Processing and analyzing large medical image sets

Johan Montagnat

Traitement et analyse de grands ensembles d'images médicales

Processing and analyzing large medical image sets

Johan Montagnat

http://www.i3s.unice.fr/~johan

Manuscrit d'Habilitation à Diriger des Recherches

December 20, 2006

	Pr	Mike	Brady
Reviewers :	Pr	Péter	Kacsuk
	Pr	Brigitte	Plateau

	Pr	Mike	Brady
	Dr	Hervé	Delingette
Jury :	Pr	Isabelle	Magnin
	Pr	Brigitte	Plateau
	Pr	Michel	Riveill

Remerciements

Je remercie chaleureusement mes rapporteurs, Brigitte Plateau, Mike Brady et Péter Kacsuk, qui ont accepté de réaliser un travail d'expertise approfondi sur ce manuscrit. J'apprécie d'autant plus leur enthousiasme à se livrer à cet exercice que j'ai pu mesurer avec mes quelques années d'expérience le poids des requêtes incessantes de relecture et d'évaluation en tous genres qui nous parviennent. Je félicite les membres de mon jury qui sont parvenus à un accord improbable de leurs agendas débordants pour venir se plier à l'exercice académique de la soutenance : Brigitte Plateau et Mike Brady encore, Isabelle Magnin, Hervé Delingette qui a guidé mes premiers pas en recherche et Michel Riveill.

Les travaux présentés dans ce manuscrit bénéficient des apports d'un grand nombre de collaborateurs sans lesquels il m'aurait été bien impossible d'assembler le présent recueil. Je veux souligner ici le rôle particulièrement important qu'ont joué Isabelle Magnin et Michel Riveill en m'accueillant dans leurs équipes et en apportant un cadre serein et dynamique indispensable à la construction d'une activité de recherche. Je remercie mes nombreux co-auteurs : N. Ayache, R. Ayani, E. Bardinet, F. Bellet, H. Benoit-Cattin, C. Blanchet, V. Breton, L. Brunie, F. Chanussot, A. Charnoz, P. Clarysse, L. Collins, J. Darcourt, E. Davila, H. Delingette, H. Duque, D. Emsellem, A.C. Evans, Z. Farkas, L. Flórez-Valencia, Á. Frohner, Y. Gaudeau, S. Georget, C. Germain, T. Glatard, J. Gotman, D. Hill, M. Janier, E. Jeannot, P. Kacsuk, B. Koblitz, M. Koulibaly, P. Kunszt, J. Lefloch, Y. Legré, F. Leprevost, M. Liljenstam, D. Lingrand, C. Loomis, I. Magnin, L. Maigne, G. Malandain, R. Medina, S. Miguet, J.-M. Moureaux, M. Orkisz, X. Pennec, J.-M. Pierson, C. Odet, A. Osorio, S. Ourselin, Q.C. Pham, B. Qiu, D. Rey, M. Riveill, A. Roche, R. Rönngren, D. Rueckert, J.-L. Roch, N. Scapel, G. Sipos, N. Santos, L. Seitz, M. Sermesant, L. Soler, R. Stefanescu, R. Texier, T. Tweed, S. Varrette, P. Vicat-Blanc Primet, D. Vray et A.P. Zijdenbos. En particulier, je veux souligner le travail considérable réalisé par Tristan Glatard dans notre équipe ; et bien entendu le soutien indéfectible de Diane Lingrand au cours de toutes ces années.

Il n'existe pas une catégorie de sciences auxquelles on puisse donner le nom de sciences appliquées. Il y a la science et les applications de la science liées entre elles comme le fruit à l'arbre qui l'a porté.

Louis Pasteur, 1871, Extrait du Salut Public

Table des matières

1 Domaine de recherche **1**

1.1 Introduction . . . 2
1.1.1 Avant propos . . . 2
1.1.2 Motivations . . . 2
1.1.3 Curriculum Vitae . . . 2
1.1.4 Thématiques de recherche . . . 5
1.1.5 Activités périphériques de gestion de la recherche . . . 13
1.2 Approche . . . 15
1.2.1 Une recherche guidée par les besoins applicatifs . . . 15
1.2.2 Le risque de la recherche applicative . . . 16
1.3 Analyse d'images médicales . . . 18
1.3.1 De l'acquisition au traitement d'images . . . 18
1.3.2 Évolution de l'analyse d'images médicales . . . 19
1.3.3 De la recherche à l'hôpital . . . 20

2 Scientific contributions **21**

2.1 Medical images segmentation . . . 22
2.1.1 PhD thesis work . . . 22
2.1.2 Axially constrained cylindrical models for vessels modeling from 3D angiograms and stents pose simulation . . . 25
2.1.3 4D deformable models for left ventricle segmentation from SPECT image sequences . . . 32
2.1.4 Current trends in model based segmentation . . . 38
2.2 Validating medical image analysis procedures . . . 39
2.2.1 Application : brain atrophy measurement from MR Images . . . 39
2.2.2 Validation . . . 39
2.2.3 Image processing . . . 40
2.2.4 Comparing brain atrophy measurement methods . . . 45
2.2.5 Validating the approach in Multiple Sclerosis studies . . . 53
2.2.6 Experimental framework . . . 57
2.3 Data grids for medical imaging . . . 58
2.3.1 Clinical use of medical data versus grid use . . . 58
2.3.2 Manipulating medical data on grids . . . 61
2.3.3 A grid medical data manager . . . 64
2.4 Compute intensive medical data analysis procedures . . . 70
2.4.1 Requirements related to intensive medical data analysis . . . 70
2.4.2 Dealing with remote interactive applications . . . 72
2.4.3 Data distribution to match computing resources availability . . . 79
2.4.4 Optimized granularity through algorithm complexity modeling . . . 79

2.4.5 Optimized granularity through non-deterministic grid modeling 89
2.4.6 Grid-enabled data-intensive workflows 97
2.4.7 Data composition strategies 101
2.4.8 Scheduling and executing workflows of services 108
2.4.9 Legacy code wrapping 113
2.4.10 Services grouping optimization strategy 116
2.4.11 Dynamic generic service factory 118
2.4.12 Experimental results 121
2.4.13 Conclusions and perspectives 134

3 Contribution au fonctionnement de la recherche **135**
3.1 Activités contractuelles et collaborations 136
3.2 Participation à l'organisation de conférences 139
3.3 Encadrement 140
3.4 Enseignement 141

4 Conclusions et perspectives **143**
4.1 Leçons tirées des principaux résultats présentés 143
4.2 Perspectives scientifiques 144

Chapitre 1

Domaine de recherche

1.1 Introduction

1.1.1 Avant propos

Dans ce document, j'utilise à dessein la forme personnelle pour introduire les réflexions qui me sont propres et qui n'engagent que ma personne. J'utilise en revanche la forme impersonnelle pour désigner les travaux de recherche réalisés. Ce n'est pas par pudeur mais bien parce que les travaux mentionnés ci-après sont le résultat de nombreuses interactions avec mes collègues et étudiants sans lesquels il m'aurait été simplement impossible d'aborder toutes les disciplines mentionnés. Qu'ils en soient ici remerciés. Je veux en particulier saluer le travail réalisé par nombre d'étudiants qui manifestent un dévouement et une énergie extraordinaire à la cause de la recherche malgré un statut très dévalorisé. Dans notre système où les chercheurs permanents sont immédiatement accaparés par des tâches périphériques de gestion, je considère qu'ils sont le moteur principal de la recherche bien qu'ils n'en aient souvent pas conscience. Ici comme ailleurs, le nerf de la recherche est l'argent. L'argent permet en grande partie de financer des étudiants et les étudiants nous font avancer.

1.1.2 Motivations

Ce document récapitule les travaux de recherche réalisés depuis mes premiers stages en laboratoire (1995) jusqu'à ma fonction actuelle de Chargé de Recherches au CNRS. Il retrace le parcours scientifique que j'ai suivi, motive le cheminement adopté qui m'a conduit de l'analyse d'images médicales à l'étude de grilles dans le domaine spécifiques des applications de la santé, et ouvre des perspectives pour les années à venir. Ce document se veut également critique vis-à-vis du système de recherche dans lequel j'ai évolué, et cherche à mettre en évidence les apports mais aussi les contraintes que celui-ci fait peser sur une carrière scientifique.

Ce document est structuré en quatre parties. Le présent chapitre 1 présente les thématiques de recherche abordées ainsi que l'approche basée sur le besoin applicatif que je défends pour conduire mes recherches. Le chapitre 2, qui détaille les différentes contributions scientifiques obtenues, est rédigé en anglais à l'instar de la quasi totalité des publications qu'il résume. Le chapitre 3 est l'occasion de présenter l'éventail des activités et des responsabilités qu'il m'a été nécessaire d'endosser, bien que ne faisant pas nécessairement parti *a priori* du bagage ou de la formation de chercheur, pour développer une activité de recherche. Enfin, le chapitre 4 apporte des conclusions et présente les perspectives ouvertes aujourd'hui.

1.1.3 Curriculum Vitae

Johan Montagnat, Chargé de Recherches 1ère classe au CNRS
http://www.i3s.unice.fr/~johan

Laboratoire I3S (UMR CNRS 6070), projet RAINBOW
2000 route des Lucioles
BP 121
06903 Sophia Antipolis cedex

Domaines de recherche
Analyse d'images médicales, modèles déformables pour la modélisation et la segmentation, systèmes distribués et grilles de calcul.

Diplômes

1999 Doctorat en Informatique de l'Université de Nice-Sophia Antipolis
"Modèles déformables pour la segmentation et la modélisation d'images médicales 3D et 4D"
préparé à l'INRIA Sophia Antipolis dans le projet EPIDAURE.
Jury : I. Magnin, P. Cinquin, D. Terzopoulos, L. Cohen, J.-P. Rigault, N. Ayache, J.-P. Thirion, H. Delingette

1996 DEA en informatique de l'ENS Cachan-Université d'Orsay
Segmentation d'image médicales volumiques à l'aide de maillages déformables contraints

1995 Magistère de l'ENS Lyon, option informatique
Maîtrise en informatique de l'Université Claude Bernard Lyon I

1994 Licence en informatique de l'Université Claude Bernard Lyon I

1991 Baccalauréat E (scientifique et technique)

Postes occupés

2005 Chargé de Recherches CNRS 1ère classe, laboratoire I3S (UMR 6070)

2004 Chargé de Recherches CNRS 2ème classe, laboratoire I3S (UMR 6070)

2001 Chargé de Recherches CNRS 2ème classe, laboratoire CREATIS (UMR 5515)

2000 Maître de conférences, Université de Nice-Sophia Antipolis

2000 Post-doctorat, Brain Image Center, Institut Neurologique de Montréal, Université McGill

Expérience professionnelle

2001-2005 Chargé de Recherches CNRS à CREATIS (imagerie médicale) puis I3S (informatique distribuée, systèmes de composants)

2003-2005 Vacataire à l'ESSI : enseignements en informatique (langages de programmation, traitement d'images, imagerie médicale)

2000 Post-doctorat au MNI : imagerie cérébrale, étude de l'évolution de la sclérose en plaques à partir d'IRM

1997-2000 Allocataire moniteur normalien : enseignements en informatique à l'ESSI et l'ESINSA (langages de programmation, algorithmique, imagerie médicale).

1996-1999 Stage de DEA puis doctorat à l'INRIA Sophia Antipolis : modèles déformables pour la modélisation et la segmentation d'images médicales

1995 Stage de 3 mois à l'Institut Royal de Technologie de Stockholm (KTH) : simulation parallèle à événements discrets

1994 Stage de 2 mois à l'Institut de Recherche en Informatique de Toulouse (IRIT) : synthèse d'images

Publications

Un *H-Number* égal à x signifie qu'au moins x publications sont citées au moins x fois (http://www.brics.dk/~mis/hnumber.html). IF est le facteur d'imact mesuré en 2005 par le *Journal Citation Report*® (http://portal.isiknowledge.com/portal.cgi). AR est le rapport d'acceptation pour une conférence (nombre de papiers acceptés / nombre de papiers reçus).

- H-number : 11 (72 citations maximum)
- 12 revues internationales : IJHPCA (IF=1.109) [77], JCMC [73], MedIA (IF=3.149) [145], JGC [132], MGV [62], PRL (IF=1.138) [154], MIM (IF=0.970) [134, 22], IVC (IF=1.383) [146], CVIU (IF=1.468) [52], CAS [192], SP (IF=0.694) [141]
- 43 conférences internationales et workshops : MICCAI'06 [86], HPDC'06 (AR=15.3%) [82], WORKS'06 [151], GELA'06 [76], EXPGRID'06 [81], HealthGrid'06 [152, 83], PDP'06 [84], GADA'05 [198], CBMS'05 [80], SCIA'05 [115],

Biogrid'05 [74], HealthGrid'05 [182], MIR'04 [78], Cluster'04 (AR=32%) [199], DEXA'04 [164], UFFC'04 [167], Vecpar'04 [20], ECCV'04 (AR=34.2%) [114], HealthGrid'04 [153, 97], ISPA'03 [137], FIMH'03 [32], HealthGrid'03 [150], Biogrid'03 [133, 57, 14], UFFC'02 [200], ICCVG'02 [61], 3DPVT'02 [136], SRMIBN'01 [21], IPMI'01 [39], SCIA'01 [116], IMVIA'01 [49], MICCAI'00 [142], ECCV'00 (AR=43.6%) [51], ICRA'00 [149], MICCAI'99 [147], MB3IA'98 [50], CVPR'97 (AR=11.4%) [138], CVRMed'97 [140], PADS'96 [173], ASS'96 [172]
- 1 chapitre de livre (Workflows for e-Science, chapitre 18) [87]).
- Rédacteur et responsable de deux chapitres du *HealthGrid whitepaper* [4].
- Éditeur de deux numéros spéciaux de MedIA (volume 9, numéro 4 [135] et volume 7, numéro 3 [122])
- Éditeur de deux actes de conférences (Springer, LNCS 2674 [123] et LNCS 2230 [105])
- 1 thèse [131]
- 1 rapport de stage de DEA [130]

Encadrements

- 2 codirections de thèse.
- 6 direction de stages de master recherche ou professionel.
- 3 codirections de stages de master recherche.

Activités contractuelles

- **European DataGrid (EDG)**. 2001-2004. Co-responsable du groupe de travail WP10 : applications biomédicales sur grille.
- **Enabling Grids for E-sciencE (EGEE)**. 2004-2006. Responsable de l'activité NA4/biomed : applications biomédicales sur grille.
- **ACI-GRID MEDIGRID**. 2002-2005. Responsable du projet.
- **ACI-MD Analyse Globalisées des données d'Imagerie Radiologique (AGIR)**. 2004-2007. Responsable des tâches "gestion de données médicales" et "architecture de gestion de tâches (*workflow*)".
- **Projet Région Rhône-Alpes : Grille pour le Traitement d'Informations Médicales (RAGTIME)**. 2003-2006. Responsable du lot applications.
- **Projet Grid5000**. 2003-2004. Contact applicatif au laboratoire CREATIS.

Organisation de conférences internationales

- **Tutorial sur les grilles de calcul et les applications au recalage d'images (MICCAI), Saint-Malo, 2004**. Co-organisateur avec Xavier Pennec et Derek Hill.
- **Functional Imaging and Modeling of the Heart (FIMH), Lyon, 2003**. Comité d'organisation, comité scientifique et édition des actes.
- **HealthGrid, Lyon, 2003**. Comité d'organisation, comité scientifique et édition des actes.
- **Functional Imaging and Modeling of the Heart (FIMH), Helsinki, 2001**. Comité d'organisation, comité scientifique et édition des actes.

Activités de relecture

J'ai participé aux comités scientifique des congrès : HealthGrid 2003 à 2006, FIMH 2001 et 2003, BioGrid 2003, EuroPar 2004, VLDB 2005, WORKS 2006 et JETIM 2006. J'ai en outre été désigné responsable de session dans les congrès : 3DPVT'02 (Padoue), session "3D imaging in Biomedicine II" le 21 juin 2002, FIMH'03 (Lyon), session 4 "motion estimation" le 6 juin 2003, HealthGrid'04 (Clermont-Ferrand), session 4 "implementation and alternative/complementary

technologies" le 17 janvier 2004, HealthGrid'06 (Valencia), session 3 "medical imaging on the grid" le 8 juin 2006.

Je suis régulièrement sollicité pour des relectures dans les journaux :

- Medical Image Analysis (MedIA)
- IEEE Transactions on Medical Imaging (TMI)
- Journal of Grid Computing (JGC)
- IEEE Transactions on Image Processing (TIP)
- Computer Methods and Programs in Biomedicine (CMPB)
- EURASIP Journal on Applied Signal Processing (JASP)
- ACM Transactions on Graphics (TOG)

et des conférences :

- HealthGrid
- Medical Image Computing and Computer Assisted Intervention (MICCAI)
- Europar
- Very Large Data Bases (VLDB)
- Workshop on Workflows in Support of Large-Scale Science (WORKS)
- BioGrid
- Computer Vision and Pattern Recognition (CVPR)
- International Conference on Computer Vision (ICCV)
- European Conference on Computer Vision (ECCV)
- British Machine Vision Conference (BMVC)
- 3D Data Visualization, Processing, and transmission (3DPVT)
- Non-Rigid and Articulated Motion Workshop (NRAMW)

Enfin, j'ai participé à l'évaluation de projets pour l'Agence National de la Recherche dans les appels Action de Recherche Amont "Masse de Données" (ARA-MD) et "jeunes chercheurs".

Jury de thèse

Membre des jury de thèse de Laurent Baduel (Université de Nice-Sophia Antipolis, 2005) et Hector Duque (INSA de Lyon, 2005).

Récompenses

Notre article intitulé "Bridging clinical information systems and grid middleware : a Medical Data Manager" a reçu le prix du meilleur article à la conférence HealthGrid'06.

Enseignement

J'ai réalisé des enseignements, essentiellement en deuxième et troisième cycles, dans les domaines de la programmation orientée objet (C++, java), la synthèse d'images, le traitement d'images, l'imagerie médicale et les grilles de calcul. Depuis 1997, ces enseignements totalisent 138 heures de cours et 540 heures de TD en plus des responsabilités administratives associées. En juillet 2006, j'ai également réalisé un cours sur les grilles de calcul à l'occasion de l'école d'été organisée par le projet SEEGRID à Budapest en juillet 2006 et je suis intervenu à l'école Grid5000 organisée à Grenoble en mars 2006.

1.1.4 Thématiques de recherche

Suite à ma formation initiale en informatique, orientée vers le parallélisme par les enseignements que j'ai suivis (du Laboratoire d'Informatique Parallèle de l'ENS Lyon de 1993 à 1995 puis du Laboratoire de Recherche en Informatique de l'Université d'Orsay en 1995-1996), mon parcours scientifique m'a amené à aborder plusieurs thématiques de recherche, *a priori* assez hétérogènes. Mes premiers travaux dans la continuité directe de mes études (domaine de la simulation parallèle par événements discrets) ont été réalisés lors d'un stage de fin de magistère à l'Institut Royal de Technologies de Stockholm dans l'équipe PARSIM (maintenant intégrée au

laboratoire IMIT[1]). Mes travaux de DEA puis de thèse, dans le projet EPIDAURE de l'INRIA Sophia Antipolis[2], m'ont ensuite conduit dans le domaine du traitement d'images qui m'était moins familier. J'y ai été mis en contact pour la première fois avec le domaine médical dans lequel je n'avais aucune formation universitaire. Mon post-doctorat au *Brain Imaging Center* de l'Institut Neurologique de Montréal (Université McGill)[3] m'ont à la fois rapproché du domaine médical et conduit à réaliser une étude sur une grande base de données d'images cérébrales pour évaluer l'efficacité statistique d'un traitement contre la sclérose en plaques. Ce travail est pour moi un révélateur du besoin grandissant de valider les algorithmes développés dans la communauté du traitement d'images médicales et de la nécessité d'une infrastructure conséquente pour y parvenir. Enfin, mon recrutement au CNRS dans le laboratoire CREATIS (traitement de l'image et du signal, interaction forte avec le domaine médical)[4] et mon implication forte dans les projets européens DataGrid (EDG) et EGEE m'ont conduit à étudier les architectures émergentes de grilles de calcul pour les applications à l'imagerie médicales. Ce travail m'a amené à établir un pont entre le domaine du traitement d'images médicale, devenu mon domaine d'expertise, et celui des systèmes distribués que j'ai essentiellement abordé lors de ma formation initiale en informatique. Dans le cadre de ma récente mutation vers l'équipe RAINBOW du laboratoire I3S[5], je poursuis cette activité de recherche transverse entre les mondes du traitement de l'image et des systèmes distribués, en m'orientant vers la thématique des systèmes de composants et des services propre à cette équipe.

Ce parcours, largement occasionné par les hasards des rencontres et des possibilités de recrutement, m'a positionné dans une activité pluridisciplinaire avec une implication en informatique (particulièrement dans le domaine des systèmes distribués) mais aussi une composante applicative importante (imagerie médicale) qui motive les recherches amont réalisées.

Simulation parallèle à événements discrets

Le domaine du parallélisme et des systèmes distribués a largement été étudié par la communauté informatique dans les années 80-90. Le premier objectif du parallélisme est l'augmentation des performances par l'utilisation de plusieurs unités de calcul en parallèle. Cette quête conduit systématiquement à trouver un compromis entre le gain induit par l'exécution parallèle et le surcoût engendré à deux niveaux :

- la distribution des tâches de calcul aux différentes unités qui induit toujours une latence (transfert de données, échanges mémoire, etc) ;
- le coût accru de développement d'un algorithme parallèle d'un point de vue de l'utilisateur.

D'autres objectifs sont ensuite apparus tels que la redondance pour augmenter la fiabilité (introduisant aussi des problèmes de cohérence), la gestion de l'hétérogénéité des systèmes distribués, la sécurité des calculs et des données, etc. Les systèmes distribués seront abordés plus en détail dans la section 1.1.4 ci-dessous.

La disponibilité de processeurs peu onéreux dont la puissance augmente à rythme élevé en suivant la demande du marché et la difficulté de mise en œuvre de nombre d'algorithmes parallèles ont cependant pesé sur le développement des systèmes distribués. Les architectures matérielles parallèles ont souvent été considérées comme trop onéreuses par comparaison à l'apport de performance qu'elles permettent d'atteindre et trop rapidement dépassées. Le coût de développement d'algorithmes parallèles ne permet d'envisager qu'un nombre restreint de programmes bénéficiant de ces architectures. Les systèmes parallèles se sont donc concentrés

[1] http://www.imit.kth.se/info/LECS/
[2] http://www-sop.inria.fr/epidaure/
[3] http://www.mni.mcgill.ca/bic.html
[4] http://www.creatis.insa-lyon.fr/
[5] http://www.i3s.unice.fr/

sur des niches dans lesquelles ils pouvaient apporter un bénéfice important, soit parce que l'algorithme à traiter se prête bien à la parallélisation, soit parce que la parallélisation s'avère indispensable en raison du coût de certains calculs.

De gros efforts ont été réalisés pour fournir aux utilisateurs des outils permettant une parallélisation aussi transparente que possible des codes applicatifs. Par exemple, la mise à disposition de bibliothèques parallèles d'analyse linéaire, permet d'instrumenter de nombreux programmes faisant appel à ce type de calcul sans nécessiter la connaissance des techniques de parallélisation. C'est l'un des objectifs du moteur de simulation parallèle à événements discrets (*PDES : Parallel Discrete Event Simulation*) sur lequel j'ai réalisé mes premiers travaux de recherche : fournir à l'utilisateur un moteur gérant la distribution des calculs, lui permettant ainsi de se concentrer sur le code applicatif concernant la simulation. Le moteur a été développé dans l'équipe PARSIM de l'Institut Royal de Technologie de Stockholm (KTH[6]).

La simulation à événements discrets consiste à utiliser un moteur d'événements qui ordonne des simulations de processus physiques. Chaque processus simulé se déroule dans un temps virtuel, ne correspondant en général pas au temps physique nécessaire au calculateur pour réaliser la simulation. Un processus simulé va envoyer des événements datés au moteur de simulation. Ceux-ci vont à leur tour générer de nouveaux calculs. Chaque événement est identifié par l'instant (dans le temps virtuel) auquel il débute. Le moteur de simulation ordonne les événements dans l'ordre chronologique dans une file d'attente pour réaliser les calculs de manière cohérente.

Le temps nécessaire aux calculs pouvant, dans de nombreux cas, être considérable en comparaison de la longueur temporelle réelle des processus simulés, des efforts de parallélisation ont été réalisés. La parallélisation est simplement réalisée en distribuant sur plusieurs processeurs les tâches de calcul déclenchées par les événements en queue dans le moteur à événements discrets. Lors d'une exécution mono-processeur, le moteur à événements discrets ne rencontre pas de problème de cohérence puisque tous les événements sont ordonnés dans la queue avant leur exécution. Mais lors de la simulation parallèle, les calculs ordonnancés se déroulant de manière asynchrone et concurrente, il se peut qu'un processus tardif envoie dans la queue un événement antérieur à l'instant du ou des derniers événements traités. On a alors un problème de cohérence temporelle (la séquentialité du temps virtuel n'est pas respectée) qui peut conduire à des calculs faux et qu'il faut corriger en repartant d'un état antérieur cohérent et en réordonnant les calculs correctement.

Pour être capable de restaurer un état antérieur à partir duquel redémarrer les calculs, il faut accepter un surcoût périodique de sauvegarde de l'état du système. Un équilibre doit être trouvé entre le surcoût induit (qui augmente avec la fréquence des sauvegardes) et le risque de devoir restaurer un ancien état (qui diminue avec la fréquence des sauvegardes). En outre, différentes stratégies de sauvegarde ont été proposées qui induisent un confort d'utilisation plus ou moins important pour le programmeur (celui-ci désirant en général se concentrer sur sa simulation sans devoir se soucier du moteur de simulation lui-même) et un coût de sauvegarde différent suivant le type de données à sauvegarder. Un autre paramètre à considérer est le coût mémoire induit par la sauvegarde des états qui peut devenir considérable.

Le travail réalisé dans ce domaine a consisté dans un premier temps à étudier et comparer différentes stratégies de sauvegarde [172]. Dans cette étude, des stratégies basées sur la sauvegarde d'états complets du système (copie de pages mémoires) et sur la sauvegarde incrémentale en prenant en considération les seuls changements réalisés (copie partielle des seules données modifiées) avec différentes stratégies relatives à la fréquence des sauvegardes ont été comparées théoriquement et expérimentalement sur une application de simulation d'un réseau de téléphonie mobile. Les expérimentations ont été réalisées sur une machine quadri-processeurs à mémoire

[6]KTH, http://www.kth.se/

partagée. Les principales conclusions de cette études sont :

- La stratégie optimale à adopter diffère en fonction de l'application et des paramètres d'exécution : il est donc important que le moteur de simulation puisse disposer de différentes stratégies et décider dynamiquement de laquelle appliquer.
- Le coût induit par la sauvegarde incrémentale diminue avec le degré de parallélisation et cette stratégie est donc préférable en cas de parallélisation massive. En outre, une implantation transparente de cette stratégie n'induit qu'un surcoût négligeable ce qui favorise largement sa mise en œuvre.

Une seconde étude, reliée à la première, a consisté à mettre en œuvre une stratégie de sauvegarde incrémentale complètement transparente pour l'utilisateur [173]. Une telle opération est rendue possible en langage C++ par la possibilité de surcharger les opérateurs du langage et notamment l'opérateur d'affectation. De cette manière, chaque modification d'une variable définie par l'utilisateur peut être interceptée par un opérateur surchargé qui aura pour rôle de sauvegarder l'ancienne valeur avant d'appliquer la modification. Une stratégie optimisée d'allocation de petites régions mémoire a été mise en place pour minimiser le temps nécessaire à l'opération de sauvegarde. L'étude a montré la viabilité de cette stratégie. La transparence d'un point de vue de l'utilisateur est presque totale, à quelques détails mineurs près.

Ce document ne détaille pas plus les travaux réalisés dans ce domaine qui n'ont pas été plus approfondis. Le lecteur intéressé pourra se référer aux publications mentionnées.

Modèles déformables pour la segmentation et la modélisation de structures anatomiques

La segmentation d'images est un problème fondamental du domaine du traitement d'images qui est sous-jacent à la majorité des algorithmes d'interprétation du contenu de l'image. Dans sa forme la plus générale, elle consiste à isoler les objets perceptibles dans une image. Un nombre considérable d'algorithmes a été proposé pour aborder ce problème à différents niveaux. Des algorithmes de détection de contours ou autres points caractéristiques permettent un première étape de transformation de l'image, identifiant la frontière des objets, mais ne livrant que peu d'information sur chaque objet individuellement puisque tous les contours sont confondus. D'autres algorithmes, parfois complémentaires, se basent sur la reconnaissance de régions homogènes (uniformes, ou présentant une texture particulière, etc) pour identifier des régions unitaires dans l'image mais celles-ci ne correspondent pas nécessairement aux objets attendus qui peuvent être composés de régions de natures très différentes. Une étape d'interprétation du contenu de l'image est donc nécessaire au delà de la simple détection de caractéristiques à partir des pixels de l'image. Il n'existe pas, loin s'en faut, d'algorithme fiable de segmentation d'image d'application générale. Seul des algorithmes très spécialisés parviennent à obtenir de bonnes performances en comparaison des capacités humaines d'interprétation d'images.

Si l'être humain est capable d'identifier et d'interpréter en une fraction de seconde les éléments d'une scène qui lui est présentée, aucun algorithme de segmentation n'est aujourd'hui capable de donner un résultat approché dans le contexte général. On sait qu'au delà de l'analyse bas niveau de l'image (détection de lignes, de points caractéristiques...), le cerveau humain fait appel à une quantité considérable de connaissances *a priori* sur les objets contenus dans une scène pour accomplir cette tâche. Le domaine de l'imagerie médicale est à ce sens révélateur : un néophyte ne verra en général que peu de choses dans une images médicale qui lui est présentée là où un radiologue aura une lecture très détaillée des structures anatomiques perceptibles. La différence entre les deux tient à leur expérience très différente de ce type d'images : le néophyte ne sais pas ce qu'il faut rechercher dans l'image.

En fait, l'étape d'interprétation du contenu de l'image joue un rôle actif dans la séparation des objets présents dans l'image. Ceci est d'autant plus important qu'une partie de l'informa-

tion peut être manquante soit par occlusion, soit, ce qui est courant dans le domaine médical, en raison de la mauvaise qualité de l'image. De nombreux algorithmes ont donc été développés qui s'appuient à la fois sur une connaissance *a priori* des objets recherchés et sur les points caractéristiques extraits de l'image. Certaines représentations par modèles déformables permettent d'atteindre ce but, bien que ce n'ait pas été la première intention de leurs concepteurs.

En effet, les premiers modèles déformables introduits à la fin des années 80 sont connus sous le nom de *snakes*. Ce sont des courbes paramétriques capables de se déformer presque librement pour s'attacher aux points de contours détectés dans une image à partir d'opérateurs simples de traitement d'images. Les déformations sont très peu contraintes : la seule information incluse dans le modèle est une indication de la régularité attendue des courbes obtenues. Un avantage des *snakes* est qu'ils produisent comme résultat un objet mathématique (courbe paramétrique) bien plus facile à manipuler qu'un ensemble de pixels identifiés par un opérateur de détection de contours par exemple. Il sont également largement moins sensibles au bruit.

Dans le cas de l'image médicale en particulier, cette approche s'avère cependant insuffisante en général. Les image médicales sont souvent de relativement mauvaise qualité (faible résolution, faible contraste, niveau de bruit élevé...), rendant l'identification des structures anatomiques particulièrement difficile. Dans de nombreux cas cependant, on peut s'appuyer sur une connaissance approximative de la forme des structures recherchées et même parfois sur une description des variations statistiques de forme rencontrées dans une population donnée. L'intégration de contraintes intrinsèques au modèle ou au processus de déformation rend la méthode beaucoup plus fiable. Cela signifie que, comme pour la totalité des méthodes de segmentation, il est nécessaire d'introduire une information spécifique sur le type d'images segmentées et sur les structures anatomiques recherchées.

Dans le domaine médical, la technique des modèles déformables présente également l'avantage, outre la segmentation, de produire un modèle géométrique synthétique et facilement manipulable des structures segmentées. Ceci est particulièrement important puisque la segmentation n'est qu'une première étape vers l'utilisation des modèles pour des besoins cliniques tels que la visualisation 3D, la quantification de paramètres anatomiques ou physiologiques, la simulation d'actes chirurgicaux, etc. Dans ce cadre, les modèles paramétriques ou les maillages présentent l'avantage de produire des représentations paramétriques concises, calculables, et facilement manipulables.

Mon implication dans le domaine des modèles déformables est issue de mon travail de doctorat. L'essentiel des résultats est consigné dans le mémoire de thèse (en français) [131] et des rapports de recherche (en anglais) [143, 148, 144]. Ce travail, initialement dédié à la segmentation du parenchyme hépatique à partir d'images scanner 3D de l'abdomen [192, 138, 140, 139, 130], a soulevé des difficultés propres à la segmentation d'images scanner 3D peu contrastées et a conduit à une étude plus générale des modèles déformables surfaciques discrets et du contrôle de leurs déformations [146, 52, 51, 149, 141, 50]. Trois autres domaines d'application ont été abordés :

- La segmentation de séquences cardiaques avec une extension du modèle proposé à la dimension temporelle et la prise en compte de différentes géométries d'images [145, 154, 142, 147].
- La modélisation de structures vasculaires pour la simulation de pose de *stents* avec l'introduction de contraintes spécifiques aux formes cylindriques recherchées [62, 61].
- La segmentation d'embryons de souris dans des images ultrasonores très haute résolution [167, 200].

L'ensemble de ces travaux a conduit à la réalisation d'un travail logiciel conséquent [49] et une partie du code est actuellement distribué sous licence GPL[7].

[7] http://www.i3s.unice.fr/~johan/crea/

Des techniques alternatives par la méthode des ensembles de niveaux ont été abordées, notamment dans le cadre d'une collaboration avec le laboratoire I3S [114, 32, 33]. Elles ont montré des difficultés similaires relatives à la contrainte de l'espace de déformation de la surface qui doivent être résolues de manière différente dans le cadre de cette représentation.

Les principales contributions de ces travaux sont :

- Un travail bibliographique de classification des modèles déformables surfaciques et des techniques de déformation [146].
- Le contrôle de la déformation des modèles déformables par une augmentation progressive du nombre de degrés de liberté de l'espace de déformation. Cette procédure permet d'améliorer la convergence du processus de minimisation sous-jacent vers la solution recherchée [141, 140].
- Le contrôle du paramétrage de la surface discrète utilisée pour optimiser la forme représentée à partir d'un nombre limité de sommets [52, 51, 50].
- L'introduction d'*a priori* de forme facilitant la reconstruction de structures connues, surtout dans les régions très bruitées ou occultées de l'image. Cet *a priori* peut être formulé intrinsèquement au modèle à travers les contraintes de régularisation [138] ou à travers les contraintes appliquées au processus de déformation [62, 61].
- L'extension du travail réalisé aux séquences temporelles (images 2D+T et 3D+T). De manière similaire à l'*a priori* de forme introduit dans l'espace, un *a priori* de trajectoire peut être utilisé pour aider à la reconstruction des objets en mouvement dans une séquence d'images [145, 142].
- La mise en œuvre de l'algorithme de segmentation dans des images provenant de différentes modalités (scanner, IRM, PET, ultrason) [145, 200] et présentant différentes géométries (parallélépipèdique, cylindrique et conique) [154, 147].

Elles sont décrites plus en détail dans la section 2.1.

Ces travaux ont montré la faisabilité de la segmentation semi-automatique de structures anatomiques 3D dans une grande variété de modalités. L'assistance humaine reste cependant souvent nécessaire pour assurer la convergence de l'algorithme de minimisation vers le résultat désiré. L'algorithme propose alors essentiellement un support pour faciliter la reconstruction, libérer l'utilisateur de l'essentiel de la tâche longue et fastidieuse, et introduire une composante 3D voire 3D+T difficilement perceptible humainement dans les images médicales. La segmentation d'images ouvre la porte à de nombreuses applications cliniques basées sur les résultats de cette première étape d'interprétation de l'image.

Validation des procédures médicales

Certains algorithmes de traitement d'images médicales ayant atteint un degré de maturité suffisant pour envisager leur utilisation dans un contexte clinique, un problème grandissant est celui de la validation des résultats obtenus à partir de ces algorithmes. Dans quelques cas, on peut s'appuyer sur une vérité terrain connue et quantifiée qui peut être comparée aux résultats calculés. C'est le cas par exemple lorsque l'on acquiert puis l'on traite l'image d'un fantôme manufacturé à des fins de test. C'est également la motivation du développement de simulateurs d'images médicales réalistes qui produisent des données dont tous les paramètres sont connus précisément. Dans la majorité des cas cependant, il n'existe pas de vérité terrain et on doit s'appuyer sur une référence plus ou moins précise et quantifiable pour la validation. Ainsi, de nombreux algorithmes de segmentation sont comparés dans la littérature à des résultats de segmentation manuelle. De tels résultats sont nécessairement subjectifs et il est indispensable d'évaluer la variabilité inter-opérateurs pour estimer si les résultats automatisés de segmentation sont fiables en comparaison des résultats obtenus par des experts.

Un autre problème émergent lié à la maturité des algorithmes est celui de leur utilisation sur

de grandes populations d'images pour quantifier l'effet d'un traitement médical. Si on considère les volumes de données impliqués dans la majorité des études médicales, allant communément de centaines à des milliers d'individus, l'évaluation manuelle des résultats et même le simple traitement de toutes les données manuellement n'est plus possible. Il devient nécessaire de mettre en place une infrastructure capable de gérer le flot de calculs engendré et d'analyser l'ensemble des résultats de manière statistique pour produire des résultats ne concernant plus chaque individu mais l'ensemble d'une population.

A l'Institut Neurologique de Montréal, nous avons conduit une étude portant sur l'évaluation d'un traitement contre la sclérose en plaques par inféron-beta [39]. Il est connu que la sclérose en plaques, comme d'autres maladies neuro-dégénératives, provoque une atrophie cérébrale pouvant atteindre environ 1% du volume du parenchyme cérébral par an. Le vieillissement normal conduit également à une atrophie mais celle-ci est moins importante, de moins de 0.5% par an chez les sujets adultes considérés. Le taux d'atrophie peut être considéré comme une mesure de la progression de la maladie. L'étude était menée grâce à une base de données d'environ 400 séquences temporelles d'IRM cérébrales de patients atteints de sclérose en plaques, chaque séquence étant constituée de 3 à 4 acquisitions réalisées tous les 6 mois. La population de patients était découpée en 3 groupes de tailles égales : un groupe placebo et deux groupes ayant reçu un traitement avec des dosages différents. Le but de cette étude était donc de :

- Mettre en place une procédure d'estimation de l'atrophie cérébrale dans des séquences d'images et en évaluer la précision.
- Mesurer l'atrophie cérébrale au sein de la population et tenter d'identifier statistiquement en aveugle les trois groupes de patients traités.

La première étape posait un problème d'évaluation des algorithmes utilisés pour lequel nous ne disposions pas de vérité terrain, et la seconde étape était liée à la mise en œuvre d'expériences médicales à grande échelle et l'analyse statistique des résultats. Ces travaux sont détaillés dans la section 2.2.

Les principales contributions de ce travail ont été :

- Le développement d'une procédure précise et fiable de la mesure de l'atrophie cérébrale à partir de techniques de recalage non-linéaires d'images et sa confrontation à une étude basée sur la segmentation du parenchyme et des ventricules cérébraux par modèles déformables.
- La mise en œuvre d'une infrastructure de calcul et de validation rapide des résultats obtenus sur plusieurs milliers d'images traitées.
- Le calcul de paramètres statistiques sur les 3 populations de patients.

Les résultats de cette étude ont montré qu'il n'existait que très peu de différence statistiquement significative entre les 3 populations étudiées. Ce résultat négatif peut avoir trois explications :

- La procédure employée sur les images disponibles ne produit pas de résultats assez précis pour mesurer l'évolution de la pathologie. Ceci semble contredit par l'étude détaillée en section 2.2 mais les mesures de variation de volume réalisées sont très faibles en comparaison de la résolution des images employées.
- L'atrophie cérébrale n'est pas une mesure adaptée. Une corrélation reconnue dans la littérature clinique a pourtant été établie entre atrophie cérébrale et évolution de la sclérose en plaques selon une échelle de progression de la maladie estimée qualitativement par les médecins.
- Le médicament a peu d'effet sur l'atrophie cérébrale. L'effet bénéfique auprès des malades semble reconnus mais ceci n'implique pas nécessairement un ralentissement de l'atrophie cérébrale.

Des études complémentaires auraient été nécessaires pour tester ces différentes hypothèses mais mon séjour trop court au MNI ne me l'a pas permis.

Cette étude a été rendue possible par le déploiement à l'intérieur du MNI d'une infrastructure permettant le stockage des très gros volumes de données impliquées d'une part (plusieurs To) et la répartition des calculs sur une trentaine de processeurs disponibles en interne pour le calcul. Cette infrastructure mise en place de manière *ad hoc* en fonction des besoins internes de l'institut ne portait pas encore de nom en ce début d'année 2000 mais serait aujourd'hui qualifiée d'*intra-grid*. Un tel système s'est révélé indispensable pour la mise en œuvre de ce type de procédure médicale : les temps de calcul auraient avoisinés 6 mois par test au lieu d'une semaine, les rendant simplement impossibles.

Grilles de calcul, application à l'imagerie médicale

Mon activité contractuelle, à commencer par mon implication dans le projet Européen DataGrid à partir de début 2001, m'a conduit à m'intéresser aux infrastructures de grilles de calcul et en particulier à leur utilisation dans le domaine de la santé. A cette époque, le domaine des grilles de calcul connaissait un essor considérable dans la communauté informatique mais l'imagerie médicale n'avait pas encore été identifiée comme un domaine applicatif potentiel.

La physique des particules a été un moteur applicatif considérable de ce type de technologie, les grilles de calcul ayant été rapidement identifiées comme un outil potentiel pour absorber le flot considérable de données qui sera produit par le LHC (*Large Hadron Collider*), la prochaine génération d'accélérateur de particules qui sera mis en servie au CERN[8] à partir de 2008. Les grilles, dans ce contexte, sont essentiellement considérées comme une extension de l'infrastructure informatique usuelle pour cette communauté : les centres de calculs qui mettent à disposition de leur utilisateur une quantité considérable de ressources (processeurs, disques...) accessibles à travers un gestionnaire par lots (*batch*). Il s'agit donc de fédérer des fermes de calculs, utilisant chacune son gestionnaire par lot, à travers une interface logicielle rendant l'accès à ces différentes ressources de manière aussi transparente que possible.

Nos premiers travaux dans ce domaine ont consisté à identifier les applications de l'imagerie médicale pour lesquels les grilles peuvent apporter un bénéfice particulier, les besoins spécifiques de ces applications en terme d'infrastructure et les perspectives ouvertes en terme de développement de nouvelles applications [74, 20, 133, 22, 21]. Ces travaux ont contribué à l'identification et dans certains cas à la formalisation de verrous relatifs à ce domaine applicatif. La gestion des données médicales (images et dossier patient associé) est rapidement apparue comme un problème central en raison de différents facteurs parmi lesquels :

- La confidentialité des données médicales, problème d'autant plus sensible que les données sont distribuées et échangées entre différents sites.
- La structuration complexe et peu normalisée des données relatives aux dossiers médicaux.
- La distribution naturelle des données dans différents centres radiologiques et les besoins de fouilles de base d'images faisant intervenir la sémantique des données et le contenu des images.
- Le volume considérable de données radiologiques produites (qui se compte en To par années à l'échelle d'un département radiologique et en Po à l'échelle nationale).

Les problèmes de sécurité et de distribution des données ont été l'objet des sujets de thèse de Ludwig Seitz [181] et d'Hector Duque [56] respectivement. D'autres spécificités du domaine médical sont la nécessité de développer des applications interactives permettant un contrôle de l'utilisateur ou encore la prise en compte de situations d'urgence nécessitant une mobilisation immédiate des ressources. Dans certains cas tels que la télémédecine ou la simulation d'interventions chirurgicales, des contraintes temps réel doivent même être prises en compte. Elles concernent les transmissions réseau aussi bien que la réalisation des calculs eux-mêmes. En col-

[8] Centre Européen de la Recherche Nucléaire, http://www.cern.ch/

laboration avec une liste d'experts représentatifs des différents domaines de la santé, nous avons travaillé à la rédaction d'un *whitepaper* qui tente d'établir l'ensemble des besoins exprimés par ces applications en termes d'infrastructures de grilles [4].

Découlant de cette analyse, nos contributions dans ce domaine ont concerné :

- La gestion des données et des méta-données médicales. En prenant en compte la nature distribuée et les contraintes d'efficacité associées [57, 56] ainsi que le caractère semi-structuré des données [164].
- L'indexation d'images médicales [78, 75] et la recherche hybride dans des bases de données basées à la fois sur les méta-données et le contenu [150, 153, 134].
- Le déploiement d'applications interactives dont l'interface doit être séparée de la partie calculatoire [137, 136].
- Les mécanismes de sécurité incluant l'authentification [198] et la mise en œuvre de politiques de contrôle d'accès [182].
- L'optimisation des transferts par contrôle de la qualité de service offerte par le réseau [199].
- Le déploiement d'applications complexes à travers un gestionnaire de flots (*workflows*) prenant en compte les données [82, 76, 86, 83, 80].
- Le développement d'applications médicales sur des infrastructures de calcul [132], incluant la simulation d'IRM [14] et le recalage d'images [97].

Ces travaux, couvrant un domaine étendu, témoignent de la jeunesse du domaine dans lequel peu d'antériorité existe. *A contrario*, le foisonnement d'activités autour des grilles de calcul ces dernières années a conduit à une explosion du nombre d'événements, de projets et de publications relatifs aux grilles qu'il devient difficile d'analyser dans son ensemble. L'Europe en particulier a acquis un avantage dans le domaine des infrastructures de grille en général et de leur application à la santé en particulier grâce à une identification précoce et un investissement conséquent dans ces domaines. Elle tente aujourd'hui de le conserver dans un domaine devenu très concurrentiel face, notamment aux États Unis d'Amérique et à l'Asie qui se sont depuis emparés de ces domaines.

1.1.5 Activités périphériques de gestion de la recherche

Les sections précédentes donnent de manière classique une énumération des publications scientifiques réalisées. Bien que pratique et quantifiable, le système d'évaluation sur le nombre et la nature des publications ne me semble refléter qu'une partie très incomplète du travail d'un jeune chercheur. J'ai été très rapidement surpris lors de ma prise de fonction en tant que chercheur permanent par la quantité de tâches périphériques afférentes à la gestion de la recherche qui viennent se substituer au travail de recherche lui-même. On peut citer des activités directement reliées telles que l'encadrement d'étudiants, l'organisation de manifestations scientifiques ou l'enseignement mais beaucoup d'autres dont l'attribution aux chercheurs est plus discutables telles que :

- L'organisation de manifestations scientifiques comprend bien d'autres éléments logistiques que les problèmes liés à la recherche elle même. Cela va jusqu'à la négociation des locations de salles et la sélection des menus chez le traiteur.
- Le montages et suivi de projets répondent à la nécessité de financer la recherche mais ils sont loin de ne faire appel qu'à des compétences scientifiques : pour la mise en œuvre de gros projets Européens, de nombreuses entreprises se sont d'ailleurs spécialisées dans la rédaction des documents administratifs et la formalisation des document techniques. Fort est de reconnaître que leur apport est considérable et que le contenu scientifique est visiblement très insuffisant à l'acceptation d'un tel projet. La gestion de projet implique une activité managériale à laquelle les chercheurs sont également bien peu préparés.

- La recherche de financements d'une manière générale, parmi la zoologie des sources de financement possibles (support régionaux, nationaux, Européens, collaboration spécifiques inter-états), toutes assorties de contraintes dont les raisons profondes n'ont certainement que bien peu à voir avec un soucis d'efficacité dans la recherche (possibilité de financer un étudiant mais surtout pas un doctorant, possibilité de financer un post-doctorant à condition qu'il ne soit pas de nationalité française ou n'ai pas séjourné dans un laboratoire français, possibilité de financer du matériel uniquement à hauteur de 30% de la facture et à condition que "quelqu'un" d'autre prenne le reste en charge...), est certainement une activité de recherche à part entière. Il faut en tout cas plusieurs années pour acquérir une vision, toujours incomplète, de cet environnement hétérogène et hétéroclite.
- Les déplacements incessants associés aux projets, et l'organisation même de ces déplacements est proportionnelle au nombre de projets suivis et terriblement consommatrice en temps.
- Les activités de communication et de valorisation sont encore des exemples auxquels les chercheurs sont bien peu préparés et qui nécessitent un travail considérable si l'on envisage de les mener à bien sérieusement.
- En informatique en particulier, le développement logiciel et l'administration système nécessitent beaucoup de temps, ou l'assistance d'ingénieurs qui font souvent défaut.
- ...
- sans mentionner courriels et réunions multiples qui représentent une véritable alternative au travail.

Dans toutes ces démarches, la lourdeur induite par des réglementations diverses et variées est souvent écrasante. Contraintes qui ont souvent pour objectif d'empêcher les abus mais qui ont également comme conséquence de pénaliser tous ceux qui, en faisant simplement leur travail, ne cherchent pas à abuser d'un système. La recherche de financement en particulier, en raison de l'éclatement des sources de financement d'une part et de l'émiettement de leur montant d'autre part est une véritable chasse au trésor incessante terriblement consommatrice en temps. Les anglo-saxons utilisent d'ailleurs comme critère secondaire d'évaluation les sommes récoltées en plus du nombre de publications. Il ne fait pas de doute que ces activités "périphériques" représentent une large majorité de mon activité aujourd'hui. Les tâches de gestion de la recherche ont très rapidement pris le devant sur l'activité de recherche elle-même et cela amène à s'interroger sur le rôle des jeunes chercheurs dans le système actuel : qu'attend-on véritablement d'un chercheur ? La réponse varie selon la perspective dans laquelle on se place (instance d'évaluation, directeur de laboratoire, organisme de financement...).

1.2 Approche

1.2.1 Une recherche guidée par les besoins applicatifs

Les travaux de recherche en informatique rapportés ici ont été largement inspirés par le domaine d'applications de l'analyse d'images médicales. Cette approche, considérant l'application comme porteuse d'un besoin qu'il convient d'identifier et de résoudre au mieux, est volontaire. L'intérêt pour le domaine d'application médical et les retombées potentielles à des applications de la santé constituent l'une de mes principales motivations pour la recherche.

Deux exemples peuvent illustrer cette approche. Les travaux rapportés sur le développements de modèles déformables surfaciques (section 2.1) sont motivés par la nécessiter de segmenter des structures anatomiques dans des images médicales tridimensionnelles. Les modèles déformables, et les techniques de minimisation d'énergie ou d'extraction d'information dans les images qui sont sous-jacentes, sont ici un outil pour atteindre ce but et non pas considérés comme un domaine de recherche en soi. Comme discuté dans l'introduction, le développement d'un outil générique de segmentation reste de toutes manières un problème largement ouvert et l'introduction d'une connaissance sur le type d'images analysées est absolument nécessaire dans ce domaine pour développer des outils fiables. De manière similaire, les travaux sur les grilles de calcul (sections 2.3 et 2.4) n'ont pas été inspirés par une formation initiale dans le domaine des systèmes distribués mais par la nécessité de traiter des bases de données d'images conduisant au déploiement d'une infrastructure adaptée. Ces travaux, qui pourraient être élargis dans un contexte plus général, se focalisent donc sur les points clés liés au domaine applicatif tels que les problèmes de gestion de données volumineuses et confidentielles, la gestion du flot de calcul dominé par les données ou encore le besoin d'interactivité. Il est à noter que dans ce cadre le travail d'identification des besoins du domaine applicatif a constitué en soi une partie importante de la problématique de recherche. Cette approche a conduit à largement enrichir la définition de l'infrastructure de grille envisagée dans le projet Européen DataGrid.

Cette approche est souvent difficile à défendre dans une communauté française Mathématique et Informatique qui plébiscite largement une approche "amont" ou "théorique" au détriment d'une approche plus pragmatique. L'apport de la recherche amont est indéniable et je partage en ce sens l'avis d'une majorité de mes collègues. La frontière entre recherche théorique et recherche appliquée est bien entendu très floue et il serait illusoire de vouloir opposer l'un à l'autre. Il me semble cependant insatisfaisant de vouloir s'abstraire systématiquement des applications et des contraintes fortes qu'elles génèrent souvent. Il est intéressant de remarquer que les travaux rapportés dans ce manuscrit seront certainement qualifiés de recherche amont dans un laboratoire de traitement du signal et de l'image appliqué à la médecine comme CREATIS (où l'application est réalisée à l'hôpital, au chevet du patient) tandis que les mêmes travaux apparaîtront comme appliqués dans un laboratoire représentatif de la communauté informatique comme l'I3S.

Je pense que partir du domaine applicatif pour exhiber des problèmes concrets et intensifier l'effort de recherche autour de ces problèmes améliore l'efficacité du système de recherche. Cette approche est certainement mieux défendue dans le monde anglo-saxon de la recherche. Mais la tendance actuelle qui consiste à financer de manière presque exclusive la recherche sur la base de contrats encadrés par des appels d'offre thématiques la supporte complètement. Ce qui m'étonne plus est le manque de reconnaissance de la recherche appliquée dans notre communauté malgré cet état de fait. L'application a pour vertu de canaliser l'effort de recherche et de lui fixer un objectif précis qui servira à orienter le travail dans un premier temps et à évaluer le résultat dans un second temps. Lors de ma soutenance de thèse, la question la plus inattendue posée par D. Terzopoulos, membre du jury, était de savoir comment l'on peut considérer qu'un travail de recherche est terminé. Je cherche encore une réponse satisfaisante à cette question, la recherche

ouvrant souvent plus de perspectives qu'elle n'apporte de réponses. Une partie de la réponse tient cependant dans la nécessité de se fixer un objectif et de confronter les résultats obtenus à une réalité mesurable.

1.2.2 Le risque de la recherche applicative

Cette discussion sur la tension existant entre recherche amont et recherche appliquée renvoie en Informatique au problème de la définition de ce qui est du domaine de la recherche, du domaine de l'ingénierie, ou du domaine de l'application. En simplifiant on pourrait dire que la recherche produit les algorithmes et les méthodes permettant aux ingénieurs de développer les outils qui seront utilisés dans un contexte applicatif. Cette chaîne complète, de la recherche à l'application est cependant difficile à construire en pratique. D'autant plus difficile que ces différentes étapes restent souvent cloisonnées dans le système français. Le succès d'une intégration des résultats de la recherche consiste à établir un pont entre ces différentes étapes qui ne peut être construit que si les différents intervenants ne s'enferment pas dans des prérogatives trop limitées et acceptent d'intervenir à la limite de leur domaine. Un algorithme, aussi élégant soit-il, sera d'un intérêt limité s'il n'est pas implantable en pratique et un outil logiciel non adapté à l'utilisation attendue par méconnaissance du domaine applicatif sera simplement renié par les utilisateurs.

Il est à noter qu'en France, différentes communautés ont des vues bien différentes sur l'organisation de la recherche et les activités qui sont considérées comme faisant partie de la recherche. Dans la communauté Informatique, l'ingénierie et le développement logiciel sont souvent considérés comme n'étant pas du ressort des chercheurs et de la responsabilité d'autres intervenants tels que "les industriels". Ce sont pourtant des activités indispensables à l'intégration des recherches et à leur mise en oeuvre. Et l'effort de développement ne peut bien souvent pas être abandonné dans l'espoir qu'il soit repris par d'autres si l'on souhaite véritablement le voir aboutir. La communauté de la Physique, que j'ai côtoyée à l'occasion des projets Européens DataGrid et EGEE, a une vue très différente de l'activité de recherche que nous méconnaissons. En Physique depuis bien longtemps, l'expérimentation tient une place très importante et le travail "d'ingénierie" qui consiste à mettre en place une expérience ainsi que le travail applicatif d'expérimentation et d'analyse des résultats sont complètement reconnus comme partie prenante de l'activité de recherche. Aujourd'hui ce travail comporte très souvent un développement logiciel conséquent. La physique théorique n'est qu'une partie du dispositif. La construction des instruments est nécessaire. Il est à noter que dans la communauté informatique, la mise en oeuvre d'une plate-forme telle que Grid5000 constitue sans aucun doute une première importante : la mise en oeuvre d'un instrument de recherche qui a aussi la vertu de fédérer les différents laboratoires participants. C'est une réussite qui démontre une prise de conscience de la communauté Informatique. Il faut dire à son crédit que la communauté de Physique est bien mieux dotée en terme d'encadrement de la recherche ce qui facilite considérablement la tâche.

La construction d'une chaîne complète entre recherche, ingénierie et application a un coût (humain donc pécunier) important. Les laboratoires d'Informatique ont bien du mal à y faire face. La tentation d'économie consistant, en l'absence d'encadrement adéquat, à laisser le chercheur en charge de toutes les étapes a cependant elle aussi un coût trop souvent ignoré : celui du manque d'efficacité des chercheurs dont l'activité se trouve trop diluée. Ajoutons à cela que les tâches n'ayant pas directement attrait à la recherche amont sont souvent non reconnues, voire déconsidérées. Il est bien difficile dans ce contexte, et souvent démotivant, de faire progresser l'application.

En outre, assurer une continuité entre les différentes étapes nécessite souvent des compétences multiples, pour établir un dialogue entre les différents intervenants et pour travailler dans une bonne direction en fonction des attentes de chacun. Dans le domaine de l'ima-

gerie médicale, une double compétence en informatique et en traitement d'images est nécessaire. Mais plus encore, une connaissance du domaine médical et des problèmes spécifiques qui lui sont liés est indispensable. A l'Institut Neurologique de Montréal comme dans une certaine mesure au laboratoire CREATIS, j'ai rencontré des environnements facilitant l'acquisition et l'intégration de ces compétences multiples par le regroupement dans un même institut de spécialistes des différents domaines concernés. Je considère que c'est un exemple à suivre pour promouvoir les recherches réalisées. Le rapprochement géographique n'est cependant qu'un facteur. Les barrières humaines et le cloisonnement entre disciplines restant souvent un frein réel.

La pluridisciplinarité, bien que largement encouragée dans de nombreux discours, n'en reste pas moins une position délicate à tenir aujourd'hui. Le risque de la pluridisciplinarité reste bien de n'être reconnu par aucune communauté plutôt que par toutes. C'est une chose que d'encourager les compétences multiples. C'en est une chose de reconnaître les travaux réalisés dans un domaine qui nous est peu familier et dont on maîtrise mal les contours. Un coût évident de la pluridisciplinarité est l'étalement des compétences : on ne devient expert en son domaine qu'après des années d'investissement. Aborder un nouveau domaine en profondeur nécessite souvent un nouvel investissement bien difficile à réaliser.

Ajoutons à cela que la recherche appliquée conduit tôt ou tard à une confrontation nécessaire avec "la réalité". Cette étape ultime, mentionnée plus haut comme un facteur d'orientation et d'évaluation des recherches obtenues, fait courir le risque d'une évaluation décevante. Les conclusions de l'étude sur la sclérose en plaque rapportée dans la section 2.2 est un exemple de résultat peu encourageant. La mise en œuvre d'algorihtmes sur des données réelles et la confrontation directe avec un problème concret remettent souvent en cause bien des résultats théoriques.

Ces éléments ont pour objectifs d'insister sur les difficultés rencontrées dans l'approche applicative poursuivie. Difficultés qui proviennent soit d'un environnement mal adapté, soit de l'approche adoptée elle même. C'est pourtant cette approche que j'essaie de défendre et de mettre en oeuvre au quotidien. Le temps consacré à cela est très important et limite indéniablement la production scientifique en terme de quantité et de qualité.

1.3 Analyse d'images médicales

1.3.1 De l'acquisition au traitement d'images

Depuis l'apparition des premiers appareils d'acquisition d'images médicales, à rayons X au début du siècle (radiographie) puis à ultrasons (échographie), une étape de traitement du signal pour sa transformation en image est nécessaire. La forme la plus primitive en a été l'impression d'un film photographique par une source de rayons X auquel il était exposé à travers le corps radiographié. Mais des étapes supplémentaires de filtrage et de reconstruction du signal se sont rapidement imposées. Par la suite et particulièrement avec l'apparition d'appareils d'acquisition de données 3D nécessitant une étape de reconstruction importante, le traitement du signal a pris une place grandissante dans la conception des appareils.

Cependant, le domaine de l'analyse informatique d'images médicales modernes, entendu comme un post-traitement de l'image produite par les appareils d'acquisition, est principalement né dans les années 80 à la suite de l'apparition d'appareils d'acquisition capables de produire des images sous forme numérique. Il a en particulier été poussé par les modalités d'acquisition 3D qui, par leur nature, traitent et produisent des données numériques dont l'impression sur film n'est qu'une étape terminale de rendu. De nombreux travaux dans ce domaine se sont attachés à produire des algorithmes de transformation et d'interprétation de l'image à des fins de visualisation, d'amélioration, de segmentation, de quantification, et d'analyse de paramètres cliniques. L'outil informatique le plus utilisé dans la communauté médicale est sans aucun doute le visualiseur, intégré à la console d'acquisition, qui est devenu le compagnon indispensable pour explorer et lire l'image. Les consoles se sont ensuite enrichies et plus un appareil d'acquisition n'est livré sans une console proposant un nombre grandissant d'algorithmes de post-traitement permettant de faciliter la lecture par transformation de la dynamique de l'image ou de réaliser des mesures anatomiques voire physiologiques.

Il demeure aujourd'hui que malgré la nature numérique de la totalité des procédés d'acquisition d'images médicales, le film reste souvent le support de prédilection des radiologues pour la lecture d'images. Le manque de sensibilité des écrans est souvent mis en cause mais même les écrans haute résolution capables de représenter des intensités codées sur 12 voire 16 bits (contre 8 bits pour les écrans classiques) ne parviennent que peu à convaincre. Le problème préoccupant pour la communauté du traitement d'images est que l'utilisation du film va souvent de paire avec la destruction des données numériques originales : peu utilisées dans la pratique clinique, elles seront souvent détruites par manque d'infrastructure d'archivage ou simplement par méconnaissance des potentialités d'un système d'informations adapté. Les consoles d'acquisition d'images médicales sont d'ailleurs complètement conçues en fonction de l'utilisation qui est demandée par le personnel médical. Bien souvent, des mécanismes de communication avec la console ou d'archivage font défaut ou sont complexes à mettre en œuvre limitant les possibilités de post-traitement. La situation évolue lentement avec la mise en œuvre des réseaux hospitalier et l'émergence de PACS (*Picture Archiving and Communication Systems*) plus ou moins heureusement connectés au RIS (*Radiological Information System*) et au HIS (*Hospital Information System*) qui contiennent l'information technique et administrative concernant les patients et les acquisitions. Ces systèmes sont en général propriétaires et l'interopérabilité n'existe que parmi les offres d'un constructeur. Le standard DICOM (*Digital Image COmmunication in Medicine*), aujourd'hui dans sa version 3, définit néanmoins un format de stockage et un protocole d'échange d'images médicales (parmi d'autres choses)[9]. Il a lentement émergé au cours des 10 dernières années mais la volonté d'un consensus avec de nombreux partenaires désireux d'imposer leur solution a conduit à un standard ouvert, extrêmement complexe et

[9] http://medical.nema.org/

généralement mal supporté. La conformité des appareils d'acquisition et de stockage des images proposés par les constructeur à ce standard est souvent déficiente, limitant ainsi largement son intérêt. De plus, DICOM est initialement conçu spécifiquement pour l'acquisition d'images et ne contient aucun élément relatif au traitement et à la traçabilité des images.

Ces différents éléments montrent qu'il existe encore un gouffre entre l'informatique déployée et utilisée dans le domaine médical et les développements réalisés dans les laboratoires de recherche. De manière compréhensible, la première préoccupation des fabricants d'appareils d'acquisition est de subvenir aux besoins exprimés par les premiers utilisateurs que sont les médecins. Mais les conséquences sont une absence de prise en compte des besoins du traitement d'images qui conduit à une méconnaissance des apports de ces outils et un obstacle de plus pour faire migrer les résultats de la recherche vers son domaine d'application.

1.3.2 Évolution de l'analyse d'images médicales

Au cours des années 90 et au contact du domaine médical, la communauté Informatique de l'analyse d'images a peu à peu réalisé la nécessité de ne pas se limiter à une étude de l'image elle même mais à la prise en compte d'un contexte médical. Un médecin n'analyse pas une image isolée mais se réfère à tout un contexte clinique et social qui aide et conditionne la lecture qu'il peut faire d'une image. Cette migration d'une recherche ciblée, en analyse d'image, vers une recherche élargie, en analyse de données médicales composées d'images et d'autres informations cliniques, conduit au développement de nouvelles applications, plus spécifiques et pertinentes du point de vue clinique mais aussi souvent plus complexes à mettre en œuvre.

Un problème grandissant avec l'émergence d'algorithmes de traitement et d'analyse d'images est également la fiabilité et la validité des résultats trouvés. De nombreuses études récentes se sont attachées à montrer la confiance que l'on pouvait attacher à tel ou tel algorithme à travers une étude précise de résultats et de leurs variations sur un ensemble de données de référence à traiter. Le problème n'est pas aisé à résoudre dans le domaine médical où l'on manque souvent de référence à laquelle les résultats de l'étude peuvent être comparés. L'interprétation humaine des images est bien souvent le seul critère établi comme référence mais il manque à la fois d'objectivité et de reproductibilité. Par exemple, un résultat de segmentation tracé manuellement par un radiologue est toujours dépendant de l'expert réalisant le travail et même pour un seul expert, le résultat variera si le travail est réalisé à plusieurs reprises. Dans certains cas, une méthode de référence existe. Il est alors possible de s'y référer mais elle ne permet pas de prouver qu'on peut obtenir de meilleurs résultats. Par exemple en ventriculographie, le volume du ventricule gauche du cœur est très communément estimé par le calcul du volume d'un ellipsoïde dont les paramètres sont déterminés à partir de deux projections du cycle cardiaque. S'il semble très probable que l'on puisse produire un résultat plus précis à l'aide d'un modèle tridimensionnel de la cavité cardiaque, il est difficile d'en convaincre les usagers. La validation d'un nouvel algorithme va alors demander une étude clinique approfondie qui sort largement du cadre du traitement d'images lui-même. Si la communauté pharmaceutique est habituée à ce genre d'étude approfondie qui peut parfois prendre plusieurs années et inclure des tests cliniques et pré-cliniques, il n'en est pas encore de même dans la communauté d'analyse d'images. La rentabilité incomparablement plus faible rend d'ailleurs des études aussi longues pratiquement inenvisageables.

Les études de validation d'algorithmes d'analyse d'images sont donc coûteuses, à la fois sur le plan humain (réunion d'experts de différentes disciplines, analyse manuelle de bases d'images) et sur le plan matériel (nécessité de disposer de jeux de données suffisants). La constitution de bases de données d'images est un avantage certain pour la conduite de ce genre d'études actuelles ou futures. Si, aujourd'hui, les jeux de données sont souvent assemblés de façon *ad hoc* en fonction des besoins d'une étude précise, on peut espérer qu'à l'avenir l'accès aux données sera facilité et

systématisé par les réseaux informatiques et que les données nécessaires pourront être mobilisées parmi les bases de données mises à disposition, tout comme la communauté bioinformatique partage des données sur différents génomes à des fins de recherches non identifiées a priori. Nous en sommes encore bien loin.

Outre les problèmes de validation entraînant en général des études sur un ensemble de données, l'étude de bases de données d'images devient une nécessité dans différents domaines : pour construire des atlas statistiques comportant des informations sur la variabilité anatomique ou physiologique entre individus, pour étudier une pathologie particulière et ses conséquences, pour comparer différents algorithmes sur des jeux de tests communs ou encore pour envisager de nouvelles applications concernant de grandes populations telles que l'épidémiologie. De telles applications font à nouveau émerger des besoins de fédération des données médicales et de contrôle d'accès en raison de la sensibilité des données qui sont des problématiques plus traditionnellement abordées dans la communauté des bases de données.

1.3.3 De la recherche à l'hôpital

Les sections précédentes évoquent les objectifs différents qui animent la vie hospitalière et la recherche en analyse d'images médicales, au moins dans le court terme. Pour le praticien hospitalier, le patient est tout proche et la tension toujours élevée. L'objectif immédiat est celui de l'efficacité des procédures, au risque de négliger des éléments tels que l'archivage des images produites. Pour l'informaticien, le patient est beaucoup plus lointain. Il ne s'agit souvent pas d'un individu mais d'une population étudiée. Les besoins et les recherches réalisées s'établissent sur le long terme. Ces approches qui s'opposent parfois conduisent à des situations parfois difficiles à résoudre sur le plan humain.

Un autre écart existe entre les développements de la recherche et les décisions publiques qui gouvernent la santé. Certaines évolutions ne deviendront possibles que lorsque certaines infrastructures et politiques de santé auront été mises en place à grande échelle (régionale, nationale, voir internationale). Mais la charnière entre ces deux mondes semble bien lointaine et inaccessible. A l'heure où le dossier médicalisé informatique est plébiscité par la classe politique, j'ignore où se discutent et se décident les contenus d'une telle réforme d'importance capitale dans notre domaine.

Outre les difficultés techniques et les barrières sociologiques mentionnées dans la section 1.2, l'interaction entre recherche et politique de santé est encore une autre étape à prendre en compte dans le déploiement et l'application des recherches effectuées.

Chapitre 2

Scientific contributions

2.1 Medical images segmentation

2.1.1 PhD thesis work

Most of my work done on deformable models for segmenting medical images and modeling anatomical structures has been reported in my thesis [131] in French or research reports [143, 148, 144] in English. This section only makes a brief summary of the main results and conclusions and does not intend to give any details. Some continuation of this work has been done since year 2000, after my PhD defense. It includes work on brain ventricles segmentation (reported in section 2.2), vessels modeling for stent pose simulation (reported in section 2.1.2), mouse embryo segmentation from high resolution images (reported below in this section), and a validation study of 4D models on SPECT images (reported in section 2.1.3).

Simplex meshes

This work was first motivated by the development of a method for segmenting the liver in from abdominal helical CT-scans. This work was part of a larger project in collaboration with IRCAD (*Institut de Recherche contre le Cancer de l'Appareil Digestif*)[1] aiming at simulating and planing liver surgery for tumors extraction. It included work on liver vessels segmentation [191] and development of a realistic simulation engine with visual and force feedback [43].

Many deformable surface representations have been proposed in the literature [146, 126]. They are characterized by the surface representation itself and the deformation scheme used. For modeling the complex shaped liver and tackling the problem of low contrast helical CT-scan images, we have been working with *simplex meshes* [48, 47], a discrete surface representation without geometrical limitations, well suited for including prior shape constraints useful for segmenting area of the images where image extracted boundaries are lacking. Precisely, we are using 2-simplex meshes of $\mathbb{R}^3$ for representing surfaces. They are regular meshes where each vertex is connected to exactly 3 neighbors, thus exhibiting a topological duality with triangulations for which each face has exactly 3 edges. The figure 2.1 displays part of a simplex mesh (plain line and black vertices) and its dual triangulation (dual line and blue vertices) on the left and a sample mesh representing a human liver on the right.

Beyond their interesting geometric properties, the simplex meshes offer a computationally efficient deformation scheme. Our work focused on controlling the meshes deformation and improving the geometrical properties of the surface by dynamically updating the surface parametrization and topology.

Deformation control

A deformable model is a template shape designed to deform according to the data to segment and some internal constraints enforcing some regularity behavior in presence of noisy or incomplete data. To the shape of a deformable model can be associated an energy, sum of an external (or data dependent) term that decreases with the closeness of the model shape with the data, and an internal term that decreases with the regularity of the surface. Both the internal and the external terms are precisely defined depending on the target application and the desired result. The searched shape of the model is defined as the one shape minimizing the energy among all possible shapes. In practice, this shape of minimal energy cannot be analytically computed and deformable models are usually iteratively deformed using an energy minimization scheme to converge toward the desired shape. Several parameters thus interfere with the result of the deformation procedure. Many deformable mesh representations have been proposed

[1] http://www.ircad.com/

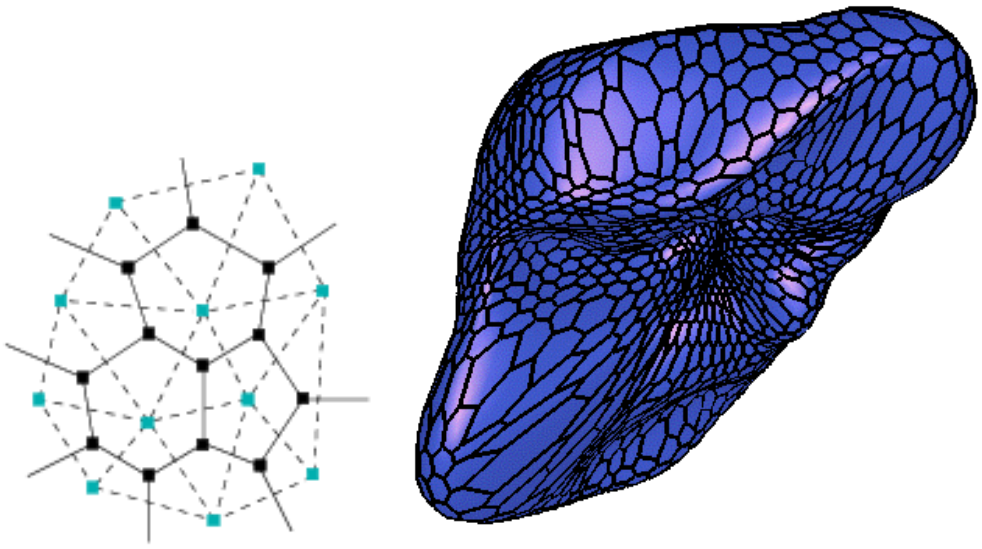


FIG. 2.1 – Left : simplex mesh and its dual triangulation. Right : sample simplex mesh representing a liver.

and their ability to segment some kind of data is always related to the kind of constraint that can be applied to the deformation process to ensure that it will converge towards the desired solution. The external energy definition depends on the data to segment. The internal energy is related to the mesh representation and is the reason for choosing simplex meshes. The energy minimization algorithm itself is very important as it is often sensitive to local minima traps.

Data term extracted from 3D medical images

Snakes seminally used gradient feature points, extracted *e.g.* using Sobel or Canny-Deriche filters, in order to attract the contour toward image object boundaries. This approach is limited to well contrasted images which is not usually the case of medical images. Many improvement have been proposed in the literature such as using 3D gradient detectors [147], region and texture information [37, 171, 209], or Gradient Vector Flow [205]. In our work, we have either used gradient-based information with additional intensity constraints for well contrasted images (such as CT-scans of the bones or MRI) or region based information for very noisy and less contrasted images (such as echographies). Pre-processing techniques such as anisotropic filtering of the images may help in reducing image noise while preserving boundary information [187, 154, 201, 5, 161]. Adaptation can be made to the case of images with cylindrical or spherical geometries as can be acquired by rotative ultrasound probes [154].

Internal regularization constraints

For each vertex in a simplex mesh, one can simply and uniquely define a local frame from its 3 neighbors enabling the location of a vertex with regard to its neighbors and 3 geometric parameters (two *metric parameters* and a *simplex angle*) [48]. As these parameters are independent from the orientation and the scale of the mesh, they intrinsically define the local shape of the surface. Moreover, the metric parameters are controlling the surface discrete parametrization while the simplex angle controls the local curvature [52]. These geometric properties enable shape constraint definitions : a deformed simplex shape evolving according to the shape

constraint alone converges towards its initial shape. Thus, in the absence of image data (due to occlusions or noise preventing correct feature extraction), the surface local shape tends to converge towards the prior shape injected in the model. In the presence of outliers, the internal shape constraint counter-balances to some extent the incorrect shape deformation. The internal shape constraint can be weighted by a scale parameter at the cost of an increasing computation time [130].

Temporal constraints for image sequences

Segmentation of temporal image sequences can be achieved by independent segmentation of each image through the approach described above. However, an important information that is temporal continuity of the motion is lost and taking into account the whole image sequence enables the introduction of motion priors just like shape priors were introduced to take into account spatial continuity of the surface. Indeed, each mesh vertex trajectory through time is a 3D line in $\mathbb{R}^3$ that can be described by shape parameters similar to those of simplex meshes. Thus, the position of a vertex at a given time instant depends on its two temporal neighbors position plus a metric parameter, a torsion angle and a curvature angle. Through this representation, temporal smoothing constraints or motion shape constraints can be applied to a temporal model [145, 142]. Vertices submitted to the motion constraint alone tend to return to their reference trajectory when they are moved away. It was shown that temporal constraints improve sequences segmentation by introducing an extra level of prior information.

Controlled energy minimization

The energy function of a deformable model directly depends on the number of parameters of the model and the degrees of freedom enabled by the deformation process. A large number of degrees of freedom ensures a highly deformable model capable of representing a large range of shapes. However, it also increases the dimensionality of the energy function and makes it more difficult to minimize and more sensitive to local minima that can trap the deformation process. Hence, the energy function is in the vast majority of cases non convex and deformable models are sensitive to their initialization (the initial energy state). Many approaches have been proposed in the literature that either limit the number of parameters of the model or the possible deformations applied to it in order to make it less sensitive to small perturbations. A trade-off has to be found between the model variability and the deformation process robustness.

An interesting approach is the Graduated Non Convexity method proposed by Blake and Zisserman [17]. The intend is to progressively transform the energy function from a convex function to the final one and to perform an iterative energy minimization with each intermediate energy function thus computed. The convergence and uniqueness of the minima can be demonstrated in some particular cases. In other cases, this process was demonstrated to improve the convergence although the global minima of the functional is not necessarily reached. We have proposed a *local deformation approach with global constraints* along the same lines by merging the *Iterative Closest Point* algorithm used for registration of data [16, 207] and the deformation process used with deformable models. The ICP is often used to apply a global transformation with a limited number of degrees of freedom (such as a rigid transformation, a similarity, or an affine transformation) to the data while a simplex mesh deformation involves a very large number of degrees of freedom (proportional to the mesh number of vertices). In our approach, a single weighting parameter enables a continuous transition from an ICP transformation to an unconstrained deformation. By progressively increasing this parameter as the model converges, one get a far better control on the deformation process [141, 140]. This algorithm especially makes sense when segmenting data for which the model gives a reasonable prior of the shape

to recover as it is usually the case in medical imaging.

Applications

Given that appropriate features extraction techniques are added for different imaging modalities, the simplex meshes technique developed proved to be a generic segmentation tool that was useful for many segmentation tasks. We have been developing different gradient and region based feature extraction techniques, enabling the segmentation of various medical image modalities such as CT-scans, MRI, echographies or nuclear medicine images. The tool was applied in quite different contexts as illustrated in figure 2.2. From top to bottom, left to right, are shown :

- Segmentation of the liver parenchyma from 3D helical angiography CT-Scan (3D rendering and model intersection with one slice).
- Segmentation of the heart left ventricle from MRI (temporal model intersection over one slice through time), ultrasound (3D rendering of one time instant), and SPECT sequences (temporal model).
- Segmentation of the brain cavity of the Tautavel skull from 3D CT-scan.
- Segmentation of the brain parenchyma and ventricles from 3D MRI.
- Segmentation of a mouse embryo from high resolution 3D ultrasound images (3D rendering and intersection over one slice).

2.1.2 Axially constrained cylindrical models for vessels modeling from 3D angiograms and stents pose simulation

This study on stent pose simulation was lead by Leonardo Flórez-Valencia as a part of his PhD thesis work at CREATIS under the supervision of Maciej Orkisz. Simplex meshes were used for vessels and stents modeling. The motivation for this work is to plan the stent pose and deployment procedure which is commonly performed to reinforce vessels pathologically deformed by stenoses (local strictures of the arterial lumen usually due to an atherosclerotic plaque, leading to hypoperfusion, ischemia and infarct of organs irrigated by the artery), or aneurysms (local distension of the artery, which rupture can lead to hemorrhage and stroke).

The stents are tubular grids that are deployed within the stenotic regions in order to push the vascular wall outwards and thus keeping open a passageway for the blood flow. The endo-prostheses are similar to the stents, but covered with a blood-proof tissue. Implanted within an aneurysm, an endo-prosthesis canalizes the blood flow and reduces the pressure on the arterial wall, thus preventing rupture. When folded, the stents and the endo-prostheses are slim and can be inserted into the artery, using a catheter. Deployed by shape-memory effect or by an inflating balloon, they become shorter and should fit to the diameter of the artery's healthy part (see figure 2.3). An appropriate pre-operative choice of the stent (or endo-prosthesis) dimensions is necessary to avoid loosening, formation of a thrombus, embolism and obstruction of branching vessels.

In this study, we attempt to simulate stent/vascular-wall interaction. The vascular lumen 3D image is first acquired using contrast-enhanced magnetic resonance angiography (MRA) technique [54]. This image is then segmented using Maracas, a software developed in CREATIS [96]. The segmentation method is based on generalized-cylinder model and provides a set of centerline points $\{\mathbf{a}_i\}$, planar contours orthogonal to the vessel centerline and estimated values of local vascular lumen radii $\{r_i\}$ [95]. The simulation of stent insertion and deployment is carried out using a simplified geometrical model "pulled on" the centerline. A more realistic representation of the stent and of the vessel wall surface, as well as the interaction between them, is then realized using a axially constrained cylindrical simplex mesh deformable model enhanced with a center-line structure that is modeled through a 1-simplex mesh of $\mathbb{R}^3$.

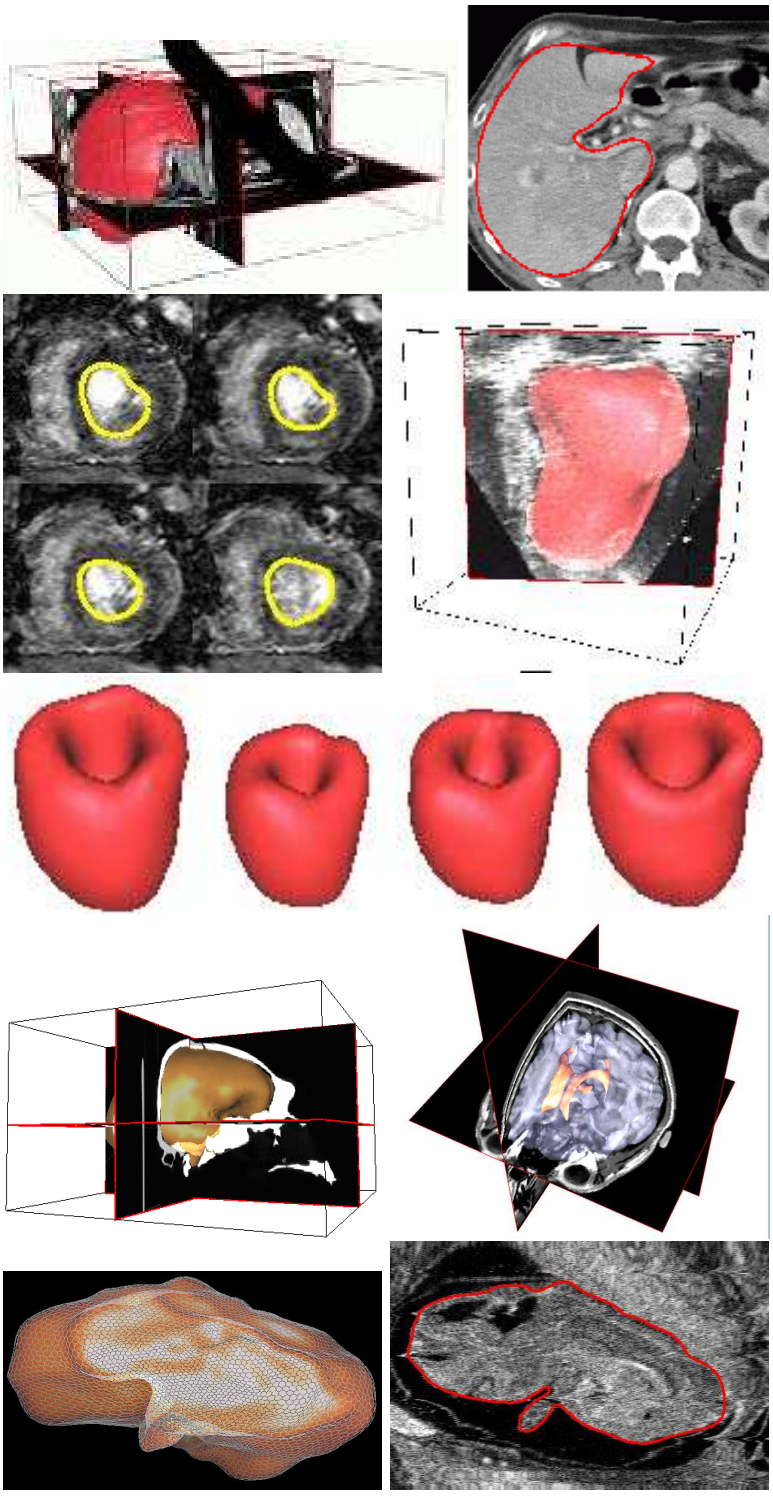


FIG. 2.2 – Segmentation results in various applications. See text for details.

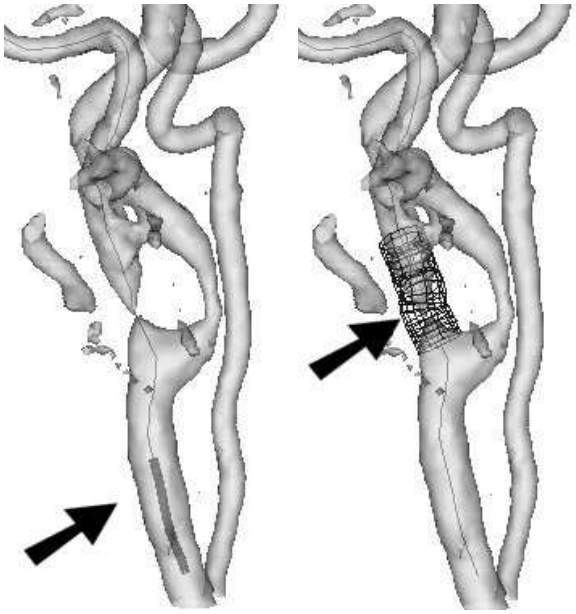


FIG. 2.3 – Simulation of stent insertion and deployment using a simplified geometrical model : a contracted stent (left) is placed along a catheter and guided until the delivery location, once expanded (right) the stent becomes shorter. The locations of the stent are indicated by arrows.

Simplex mesh initialization

The complete model is the union of the centerline $\mathcal{C}$ and of a cylindrical surface mesh $\mathcal{S} = \{\mathbf{x}_i \in I\!R^3, 0 \leq i < M\}$. It also includes the spatial relationships between the vertices. The axial shape estimated by Maracas is used for the model centerline initialization : $\mathcal{C} = \{\mathbf{a}_i\}$. The simplex mesh surface is initialized by calculating a M vertices cylindrical surface with local radii r_i.

Simplex mesh axially constrained deformation

The classical energy minimization framework for simplex meshes leads to the model iterative evolutive equation [146] :

$$\mathbf{x}_i^{t+1} = \mathbf{x}_i^t + \gamma(\mathbf{x}_i^t - \mathbf{x}_i^{t-1}) + \mathbf{d}_i^{\text{int}} + \beta\mathbf{d}_i^{\text{ext}} \tag{2.1}$$

where $\mathbf{x}_i^t$ denotes the location of vertex $\mathbf{x}_i$ at iteration t (with initial condition $\mathbf{x}_i^{-1} = \mathbf{x}_i^0$), $\mathbf{d}_i^{\text{int}}$ and $\mathbf{d}_i^{\text{ext}}$ are displacement components respectively owing to the internal and external forces, $\beta \in [0, 1]$ is an external force weight, and γ is a damping parameter. The latter plays the same role as viscosity in physically-based models and is experimentally set to $\gamma \approx 0.35$ to speed-up convergence. In our experimentation we used $\beta = 0.1$.

Model extension to generalized cylinders

Equation (2.1) defines the local displacement of each surface vertex but it does not take into account the particular shape and expected properties of the modeled object. When dealing with vessels, one expects cylindrical structures with high bending capability, for which deformations should preserve the generalized cylinder shape. In order to mimic a physical deformable cylinder, the globally constrained local deformation framework described above is extended through the

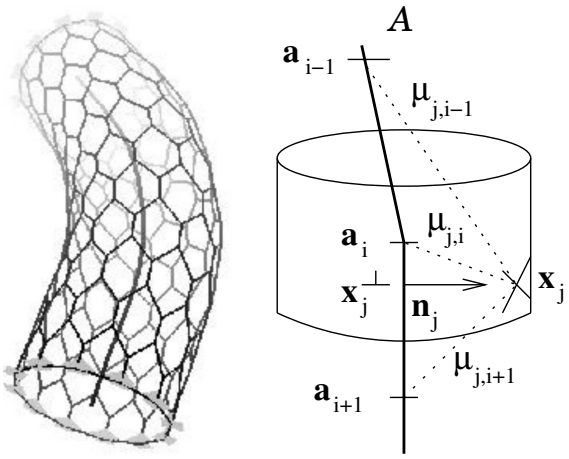


FIG. 2.4 – Left : cylinder deformed under the adapted globally constrained local deformation framework. Right : notations.

definition of an ad hoc global transformation replacing the rigid or affine transformation usually applied (see left of figure 2.4).

The surface is bound to the centerline : each surface vertex $\mathbf{x}_j$ is associated with three closest centerline vertices $\{\mathbf{a}_{i-1}, \mathbf{a}_i, \mathbf{a}_{i+1}\}$ (see notations on right of figure 2.4), except vertices at both ends of the cylinder surface, which are only bound to two centerline vertices. Conversely, we denote $\mathcal{E}_i$ the set of surface vertices bound to the centerline vertex $\mathbf{a}_i$. Each pair of vertices $(\mathbf{a}_i, \mathbf{x}_j)$ is weighted by a coefficient μ_{ij} such that $\sum_j \mu_{ij} = 1$. These coefficients are computed automatically, based on the inverse of the distance between the vertices. When the model surface undergoes some deformation, the centerline bends accordingly through an external force resulting from the surface forces. The resulting displacement is :

$$\mathbf{d}^{\text{ext}}(\mathbf{a}_i) = \sum_{\mathbf{x_j} \in \mathcal{E}_i} \mu_{ij} \mathbf{d}^{\text{ext}}(\mathbf{x}_j) \tag{2.2}$$

The centerline is considered as a 1-simplex mesh in $\mathbb{R}^3$ and the equation 2.2 is used to compute its deformation owing to the external force. Conversely, the centerline bending is reported onto the surface as the sum of an axial component (each vertex tends to follow the global motion of the axis) and a radial component (each vertex tends to align on a circle around the centerline) with :

$$\begin{array}{l} \mathbf{d}^{\text{axial}}(\mathbf{x}_j) = \sum_{k=i-1}^{k=i+1} \mu_{kj} \mathbf{d}^{\text{ext}}(\mathbf{a}_k) \\ \mathbf{d}^{\text{radial}}(\mathbf{x}_j) = \sum_{k=i-1}^{k=i+1} \mathbf{x}_j^{\perp} + \left((1-\xi) \|\mathbf{x}_j^{\perp} \mathbf{x}_j\| + \xi r_k \right) \mathbf{n}_j - \mathbf{x}_j \end{array} \tag{2.3}$$

where $\mathbf{x}_j^{\perp}$ is the orthogonal projection of $\mathbf{x}_j$ onto the centerline, $\mathbf{n}_j$ (right of figure 2.4) is the unit normal vector of the centerline in $\mathbf{x}_j^{\perp}$ (*i.e.* $\mathbf{n}_j = \mathbf{x}_j^{\perp}\mathbf{x}_j / \|\mathbf{x}_j^{\perp}\mathbf{x}_j\|$), r_k is the radius (mean distance of the surface vertices to the centerline) in $\mathbf{a}_k$ and ξ is a radial weight. In our experimentation this parameter varied between 0.3 and 0.8. With small values of ξ circularity constraint is weak and complex cross-sectional shapes can be recovered. Larger values are used when data are not reliable. When ξ tends towards one, the cross-sections tend to be circular with constant radius along the cylinder.

The surface vertices are thus submitted to the internal and external forces (local forces) plus the axial and radial forces (cylindrical forces). Let $\lambda \in [0, 1]$ weight the contributions of the local and cylindrical forces. The equation (2.1) becomes :

$$\mathbf{x}_i^{t+1} = \mathbf{x}_i^t + \gamma(\mathbf{x}_i^t - \mathbf{x}_i^{t-1}) + (1-\lambda)\left(\mathbf{d}_i^{\text{int}} + \beta \mathbf{d}_i^{\text{ext}}\right) + \lambda\left(\mathbf{d}^{\text{axial}}(\mathbf{x}_i^t) + \mathbf{d}^{\text{radial}}(\mathbf{x}_i^t)\right)$$

In our application, a strong contribution of the cylindrical forces was desired. Hence, we experimentally set $\lambda = 0.95$.

External force for MRA images

The external force $\mathbf{d}^{\text{ext}}$ is dependent on the image acquisition modality, and can use either the image gradient or an iso-value of the image intensity. An empirical study on contrast-enhanced MRA images of vascular phantoms with stenoses [94] has shown that the actual boundary does not correspond to the gradient maximum but rather to 45% of the local intra-luminal maximum of intensity. For a given vertex $\mathbf{x}_j$ this maximum is sought within a sphere $\mathcal{B}(\mathbf{a}_i, r_i)$ centered at the axis point $\mathbf{a}_i$ closest to $\mathbf{x}_j$, and having a radius equal to the initially estimated local radius r_i of the vessel. The external force $\mathbf{f}^{\text{ext}}(\mathbf{x}_j)$ is defined as a vectorial displacement : $\mathbf{f}^{\text{ext}}(\mathbf{x}_j) = \mathbf{x}_j - \mathbf{p}_j$. The point $\mathbf{p}_j$ is defined as :

$$\begin{aligned} &\mathbf{p}_j = \arg\min_{\mathbf{q}_j \in \mathcal{L}_j} \left| \mathcal{I}(\mathbf{x}_j) - I(\mathbf{q}_j) \right|, \\ &\text{with :} \quad \mathcal{I}(\mathbf{x}_j) = 0.45 \max_{\mathbf{b}_i \in \mathcal{B}(\mathbf{a}_i, r_i)} \left[I(\mathbf{b}_i) \right], \end{aligned}$$

where $I(\mathbf{x})$ is the image intensity at coordinate $\mathbf{x}$ and the search of the point $\mathbf{p}_j$ is carried out along a linear path $\mathcal{L}_j$ (a list of image voxels $\mathbf{q}_j$) described by the surface normal $\mathbf{n}_j$. The best candidate is the one having the intensity closest to the local isovalue $\mathcal{I}(\mathbf{x}_j) = \mathcal{I}_i$.

Stent modeling

Once deployed using a simplified model, the stent surface is represented using a new cylindrical simplex model combining the mapped centerline and a constant radius all along the cylinder. Once constructed, this model remains static, *i.e.* it is not submitted to deformations. In the stent area, the shape of the vessel-surface model is controlled by the stent shape rather than the external forces extracted from the image data. Therefore, the external force of the vessel-surface model is locally set to zero. Instead, the corresponding section of the vessel-surface model is attracted by the surface of the stent model.

Segmentation evaluation

The accuracy and the reproducibility of our segmentation method was evaluated on seven 3D MRA images (figure 2.5a) of physical phantoms [169]. The phantoms' internal surface represents the arterial intra-luminal shape, each one having circular reference sections (diameter of 6 mm) and two stenoses (figure 2.5b) of variable shape, position, eccentricity and known severity (between 50% and 95%). The images were acquired in realistic conditions with in-plane resolution of 0.78 mm × 0.78 mm and slice thickness between 0.75 mm and 1 mm.

The deformable model used for this simulation is applicable to vascular image segmentation and quantification (figure 2.6). For each image we obtained a stable segmentation (figure 2.5d) within a maximum of 40 iterations. However, the actual segmentation time depends on the number of simplex vertices : a model containing 100 vertices converged in 1.80 s on a PIII 800 MHz personal computer[2]. Then each resulting axis was resampled (100 evenly spaced points) and the local radii were recalculated for each mesh at these points. In the normal sections the average estimated diameter was equal to 6.25 mm±0.12 mm. Although the diameter was slightly over-estimated (4.2%), the errors remain significantly smaller than the voxel size. The standard deviation of 0.12 mm (2% of the reference diameter) gives an idea of the reproducibility, as it corresponds to measurements obtained along the reference sections within different phantoms

[2]This time value includes an iterative 3D rendering of the complete mesh after each iteration of the model evolution.

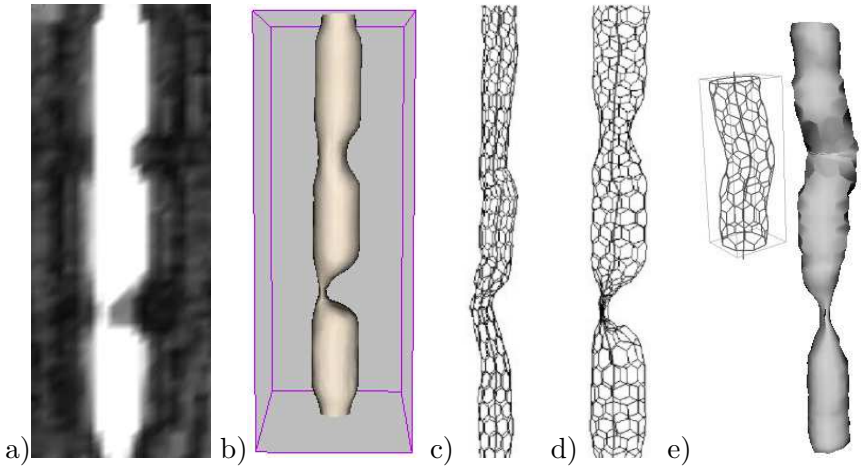


FIG. 2.5 – Vascular phantom and model evolution. From left to right : a) slice from a 3D MRA image of an arterial phantom with 2 stenoses, b) shaded-surface rendered through shape of the phantom, c) mesh initialized according to the coarse model extracted from the MRA image, d) final mesh resulting from the segmentation, and e) bended stent and deformed phantom model.

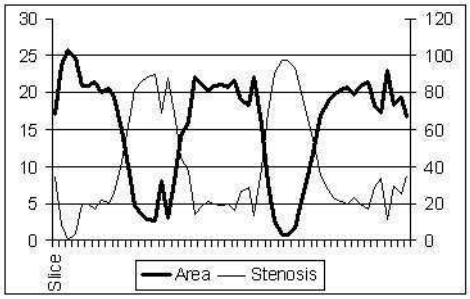


FIG. 2.6 – Quantification curves obtained from the phantom segmented by our simplex model. The scale on the left corresponds to the units of the area (mm^2), while the scale on the right corresponds to the percentage of the local narrowing of the vessel compared to a reference cross-section.

having the same diameter. The average stenosis-quantification error was equal to 7.14%. This still corresponds to a sub-voxel accuracy of the segmentation. The largest errors occurred in complex stenoses, where the shape was no more cylindrical, and in the most severe stenoses (95% of narrowing), due to the lack of signal.

Results and discussion

In images from patients both segmentation and stenting simulation have only been visually evaluated. Figure 2.7 shows an example of a stented aortic arch. In general, the simulation is very useful to assess the positioning of the stent and if the stent diameter fits to the vessel's healthy segments. However, the segmentation is sometimes inaccurate when two vessels are too close and cannot be distinguished because of an insufficient resolution of the images. In some cases we also noticed that the vessel axis is not well centered within the vessel lumen. This occurs when the vessel bends strongly. In these cases the deployed stent is not well centered, too. These results can be improved by a careful tuning of the parameters of the model, namely λ and β. Nevertheless a better solution would probably be to add an image force acting directly on the

axis and attracting it towards local gravity centers or other medialness-criterion maxima [69].

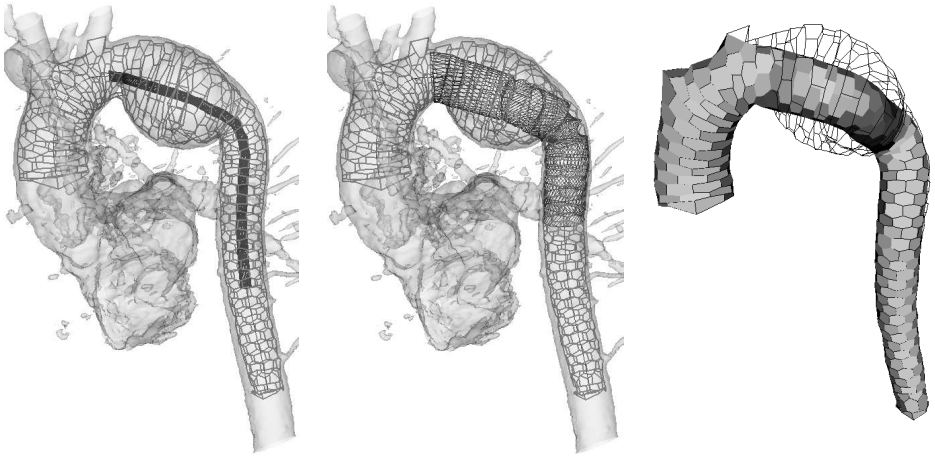


FIG. 2.7 – Simulated stent in a real aneurysm within the aorta arch. Left and middle : simplex mesh representing the result of the segmentation, with a stent placed folded and deployed. Right : Surface rendering of the simplex mesh fitting to the deployed stent .

The segmentation using the deformable model provides reasonably accurate measurements of the local diameters of the arterial lumen and of the length and curvatures of its centerline. These measurements are theoretically sufficient for the user to appropriately choose the dimensions of the stent, namely its length and its nominal diameters at the attachment sites. However, even a simple geometric simulation like ours makes this choice more intuitive, as each parameter of the stent can be interactively modified and the predictable result of this modification can be displayed in real time. Our geometric simulation tool is a step towards a complete pre-operative simulation of vascular stenting. Like other authors, it implicitly assumed that the axial rigidity of the arteries is much larger than the axial rigidity of the stents, while the radial rigidity of the stents is larger than that of the arteries. In other words, in our simulation, the axial shape of the stent is modified so that it fits to the centerline of the artery, while the arterial diameters between the attachment sites are modified so that they fit to the diameters of the deployed stent. In real situations however, the axial shape of the vessel may be modified by the stent, and the deployment of the stent may locally be limited by the rigidity of the vascular wall, *e.g.* owing to calcifications. In aneurysms, as the shape of the central part of the endo-prosthesis is not constrained by the arterial wall, the centerlines of the artery and of the endo-prosthesis may differ significantly from each other. There are frequent problems with the angulation of the endo-prosthesis near the attachment sites in strongly curved vessels. A future version of our simulation tool should therefore involve a simulation of the local mechanical interaction between the stent and the vascular wall.

The usefulness of the extraction of the patient-specific vascular geometry and of a realistic simulation of the deployment of the stents may go beyond the pre-operative choice amongst the existing stents. These tools can namely be used for the design of patient-specific stents. Furthermore, the predictive value of the simulations is to be exploited to assess the outcome of the endo-vascular remodeling in terms of hemodynamics. To this purpose, the simplex mesh simulating the remodeled surface of the vascular lumen will be used to generate a volumetric finite-elements model exploitable in the computations of the fluid dynamics.

2.1.3 4D deformable models for left ventricle segmentation from SPECT image sequences

Motion prior for segmentation of medical image sequences

As mentioned in section 2.1.1, we have introduced motion priors using geometrical parameters that describe the model vertex trajectories. The basic idea is that, given a set of surface models with the same topology representing a moving object over a sequence of image, the trajectory of each vertex is defined as a discrete line crossing all vertex positions through time that can be assimilated to a line in $\mathbb{R}^3$. Let $\mathbf{p}_{i,t}$ denote the position of vertex i at time t and $\mathbf{p}_{i,t}^{\perp}$ denote the orthogonal projection of $\mathbf{p}_{i,t}$ onto segment $[\mathbf{p}_{i,t-1}, \mathbf{p}_{i,t+1}]$. The position of point $\mathbf{p}_{i,t}$ relatively to its temporal neighbors is defined through the three parameters (see notations on figure 2.8) :

- a metric parameter $\varepsilon_{i,t} \in [0,1]$ measuring the relative position of $\mathbf{p}_{i,t}^{\perp}$ in $[\mathbf{p}_{i,t-1}, \mathbf{p}_{i,t+1}]$ $(\mathbf{p}_{i,t}^{\perp} = \varepsilon_{i,t}\mathbf{p}_{i,t-1} + (1-\varepsilon_{i,t})\mathbf{p}_{i,t+1})$;
- an angle $\varphi_{i,t}$ measuring the elevation of $\mathbf{p}_{i,t}$ above the segment $[\mathbf{p}_{i,t-1}, \mathbf{p}_{i,t+1}]$ in plane $(\mathbf{p}_{i,t-1}, \mathbf{p}_{i,t}, \mathbf{p}_{i,t+1})$;
- an angle $\psi_{i,t}$ measuring the discrete torsion of the trajectory.

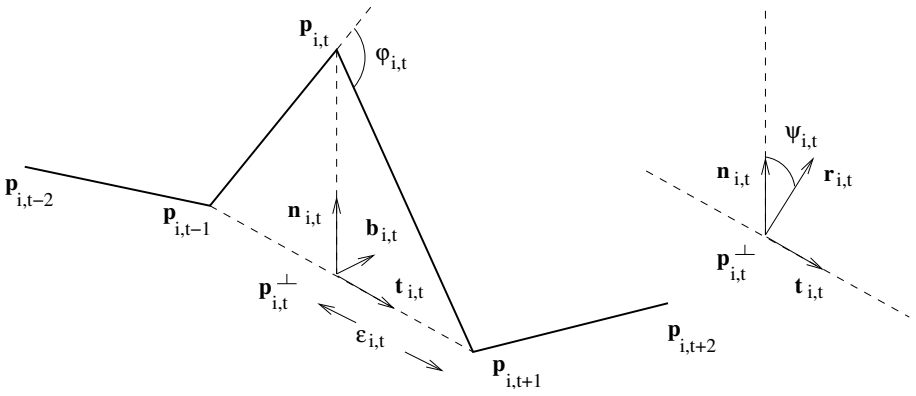


FIG. 2.8 – Trajectory geometry.

Intuitively, $\varepsilon_{i,t}$, $\varphi_{i,t}$, and $\psi_{i,t}$ correspond to discrete arc length, curvature, and torsion respectively. Let $\mathbf{t}_{i,t}$ denote the discrete tangent, $\mathbf{b}_{i,t}$ the binormal vector, and $\mathbf{n}_{i,t}$ the discrete normal to point $\mathbf{p}_{i,t}$ respectively :

$$\mathbf{t}_{i,t} = \frac{\mathbf{p}_{i,t-1}\mathbf{p}i, t+1}{\|\mathbf{p}_{i,t-1}\mathbf{p}i, t+1\|},\ \mathbf{b}_{i,t} = \frac{\mathbf{p}_{i,t}\mathbf{p}_{i,t+1} \wedge \mathbf{p}_{i,t-1}\mathbf{p}_{i,t}}{\|\mathbf{p}_{i,t}\mathbf{p}_{i,t+1} \wedge \mathbf{p}_{i,t-1}\mathbf{p}_{i,t}\|},\ \mathbf{n}_{i,t} = \mathbf{b}_{i,t} \wedge \mathbf{t}_{i,t}.$$

The metric parameter, the elevation angle, and the torsion angle are defined by :

$$\begin{aligned}
&\varepsilon_{i,t} = \frac{\|\mathbf{p}_{i,t}^{\perp}\mathbf{p}_{i,t+1}\|}{\|\mathbf{p}_{i,t-1}\mathbf{p}_{i,t+1}\|},\\
&\varphi_{i,t} = (\mathbf{p}_{i,t}\mathbf{p}_{i,t+1}, \widehat{\mathbf{p}_{i,t-1}\mathbf{p}_{i,t}}),\\
&\psi_{i,t} \text{ such that } \mathbf{n}_i = \cos(\psi_{i,t})\mathbf{r}_{i,t} + \sin(\psi_{i,t})\mathbf{t}_{i,t} \wedge \mathbf{r}_{i,t}
\end{aligned}$$

with $\mathbf{r}_{i,t} = \frac{\mathbf{t}_{i,t} \wedge (\mathbf{p}_{i,t-2}\mathbf{p}_{i,t-1} \wedge \mathbf{p}_{i,t+1}\mathbf{p}_{i,t+2})}{\|\mathbf{t}_{i,t} \wedge (\mathbf{p}_{i,t-2}\mathbf{p}_{i,t-1} \wedge \mathbf{p}_{i,t+1}\mathbf{p}_{i,t+2})\|}$.

The local deformation scheme of the simplex mesh models rewrites :

$$\begin{aligned}
\mathbf{p}_{i,t}^{\tau+\Delta\tau} &= \mathbf{p}_{i,t}^{\tau} + (1-\gamma)(\mathbf{p}_{i,t}^{\tau} - \mathbf{p}_{i,t}^{\tau-\Delta\tau}) + \\
&\quad \alpha f_{\text{int}}(\mathbf{p}_{i,t}^{\tau}) + \delta f_{\text{time}}(\mathbf{p}_{i,t}^{\tau}) + \beta f_{\text{ext}}(\mathbf{p}_{i,t}^{\tau})
\end{aligned}$$

where $f_{\text{time}}(\mathbf{p}_{i,t}) = \tilde{\mathbf{p}}_{i,t} - \mathbf{p}_{i,t}$ is an additional temporal constraint applying onto each vertex and weighted by the coefficient $\delta \in [0,1]$. To smooth trajectories over time, we set $\tilde{\mathbf{p}}_{i,t} = \frac{\mathbf{p}_{i,t-1}+\mathbf{p}_{i,t+1}}{2}$. To introduce a motion prior,

$$\begin{aligned}\tilde{\mathbf{p}}_{i,t} &= \tilde{\varepsilon}_{i,t}\mathbf{p}_{i,t-1} + (1-\tilde{\varepsilon}_{i,t})\mathbf{p}_{i,t+1} + \\ &\quad g(\mathbf{p}_{i,t-1}, \mathbf{p}_{i,t+1}, \tilde{\varepsilon}_{i,t}, \tilde{\varphi}_{i,t})(\cos(\tilde{\psi}_{i,t})\mathbf{r}_{i,t} + \sin(\tilde{\psi}_{i,t})\mathbf{t}_{i,t} \wedge \mathbf{r}_{i,t}).\end{aligned}$$

where $\{\tilde{\varepsilon}_{i,t}, \tilde{\varphi}_{i,t}, \tilde{\psi}_{i,t}\}_{(i,t)}$ are the parameters of the reference trajectory and $g = \|\mathbf{p}_{i,t} - \mathbf{p}_{i,t}^{\perp}\|$ is defined in [145].

The globally constrained deformation scheme can also be extended to the case of sequences as described in [145].

Application to cardiac images segmentation

The assessment of the cardiac function is important for the understanding and the early diagnosis of heart pathologies. In this section we illustrate the use of 4D deformable models for the segmentation of the myocardium and the Left Ventricle chamber from 3D cardiac image sequences. Based on volume estimation of the LV chamber along the cardiac cycle, the *Ejection Fraction* (EF) can be computed. The EF is an important clinical parameter measuring the ratio of blood ejected between the *End of Diastole* (ED : end of filling phase of the myocardium) and the *End of Systole* (ES : end of ejection phase). The ejection fraction value is typically $70\% \pm 10\%$ on healthy patients but is known to decrease significantly in the presence of some cardiac pathologies ([45]). Other quantitative parameters of cardiac dynamic could be extracted such as the septum wall thickness or the displacement of myocardium points.

Model measurements accuracy

The validation of segmentation outcome is a complex issue in medical imaging due to the lack of ground truth measurements for most applications ([10, 13]). To assess the accuracy of our segmentation algorithm, we propose to segment synthetic SPECT images generated by the NCAT simulator of [180]. NCAT images are produced by simulation of SPECT physics on a realistic spline-based dynamic heart phantom. The simulator produces gray level images and outputs multiple information about the observed objects, including the volume of each structure. The NCAT simulator produces realistic images in terms of geometry and physiology, although they tend to be more sharp and less noisy than real images.

For our case study, the left ventricle appears with a high intensity, while surrounding structures appear with a lower contrast. Ten image sequences have been generated simulating different heart shapes by changing the heart scale and the ratio parameters. Images of the torso generated by the simulator are cropped around the LV area, resulting in volumes of $42 \times 46 \times 30$ isotropic voxels (0.3125^3 cm^3). Each sequence is composed of 8 frames covering one heart cycle. Figure 2.9 shows 4 images (middle slice at the end of diastole) randomly selected out of the 10 simulated sequences. On the same figure are shown the associated meshes representing the LV myocardium (at the end of diastole) and the LV chamber (at the end of systole) extracted from these 4 images with our segmentation algorithm.

The segmentation algorithm is based on the deformation of 4D simplex meshes using the globally constrained local deformation scheme. We use two additional sets of 4D images to test what is the best sequence of deformation stages (from global to local) and what are the best values of the algorithm parameters. Once this manual tuning is done, we use the same sequence with the same parameter values for the segmentation of the 10 simulated 4D images. The deformation of the 4D simplex mesh is first highly constrained (rigid, similarity and finally

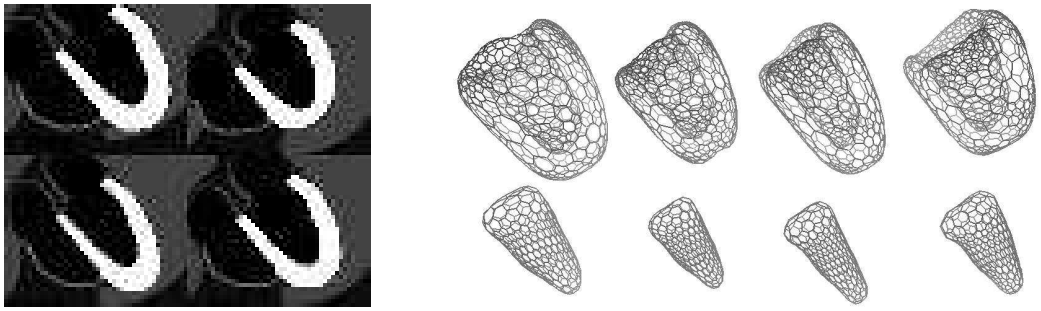


FIG. 2.9 – Left : Four simulated SPECT images. Right : simplex meshes of the LV myocardium at ED (top) and the LV chamber ES (bottom).

affine registration is performed) with strong influence of external forces. A deformation sequence terminates when the 4D mesh does not move significantly between two iterations. After the affine registration, the mesh is close enough to the myocardium boundaries to start a globally constrained deformation with a lower influence of external forces.

Figure 2.10 displays quantitative results. On the left, the diagram shows the LV myocardium volume (dashed lines) and the LV chamber volume (solid line) in the second image sequence. In each case, the thick line corresponds to the ground truth (the volume given by the simulator) and the thin line corresponds to the estimated volume of the deformed 4D model. It can be seen that, as expected, the myocardium volume has a very small variation during the heart cycle, while the LV chamber contracts significantly. From the LV chamber volume curve, the EF can be computed. On the right of figure 2.10 are displayed the average errors (expressed in % of the true volume) of the reconstructed volumes compared to the ground truth for the 10 simulated images. The volumes are averaged over the 8 instants and the error bars represent standard deviation of the volume estimates. The plain line shows the error (in %) of the computed ejection fraction compared to the ground truth.

This experience shows that the myocardium segmentation is fairly accurate on such good quality images. Over the 10 simulated images, the maximum error is lower than 4%. The LV chamber segmentation shows a more significant variation when compared to the ground truth. This is because the definition of the LV chamber is geometrically ill-posed. Indeed we lack boundary information at the base of the left chamber boundary in SPECT images. In this case, it is the model shape constraints that prevent the leakage of the mesh in the region of the chamber base. Another alternative for estimating the LV chamber could have been to compute the volume enclosed by the LV myocardium after "closing" the ventricle at the base level.

Anyway, errors are quite consistent over a given sequence (the error variance is quite low although the absolute error can reach 17% in the worst case). It results in an accurate measurement of the EF (4.7% in the worst case, which is low compared to pathological variations of the EF).

In addition to volume measurements, figure 2.11 shows the local distance between the recovered myocardium surfaces and the reference surfaces extracted by isosurface computation from the original NCAT images. The distances are computed using the M.E.S.H. software (*Measuring Error between Surfaces using the Hausdorff distance*) developed at EPFL [9]. This tool measures the asymmetric distance between two discrete surfaces (triangular meshes) using the Hausdorff distance. It provides the maximum, mean and root-mean-square (RMS) errors between two given surfaces. Left of figure 2.11 displays an example of surface comparison for the 3rd frame of image 7 in our experiments (the worst case). The colors displayed on the myocardium surface are related to the distance between the surface recovered by the segmentation algorithm and

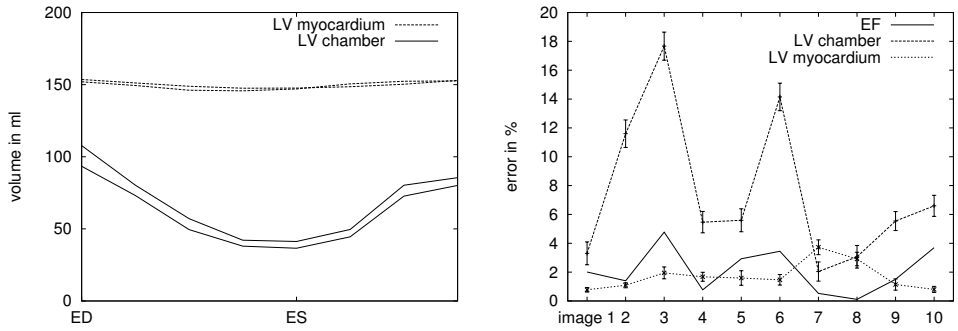


FIG. 2.10 – Left : Volume of LV myocardium and chamber on sequence 2. Right : Mean error (in % of volume) in measuring the LV myocardium volume, the LV chamber volume, and the EF.

the reference surface as computed by MESH. Darker values correspond to highest distances (up to 3.5 voxels in this case) and brighter values correspond to lowest distances. The maximum error are concentrated at the border of the base and at the apex where the mesh curvature is maximum and the smoothing constraint tends to bias the shape recovery. The diagram on the right of figure 2.11 shows the average distance (plain line with error bars corresponding to minimum and maximum distances) and the RMS distance (dashed line) between the recovered models and the reference surfaces provided to the NCAT simulator for each of the 10 sequences studied. For each sequence, the four values displayed (minimum, maximum, mean and RMS distance) have been averaged over the 8 time frames. This figures shows that the average distance is always lower than a voxel with local maxima around 3 voxels. The recovered surfaces are therefore mostly located close to the reference surfaces up to a subvoxel distance. Thus, the model accuracy is demonstrated both in local (inter-surface distances) and global (volume) measures.

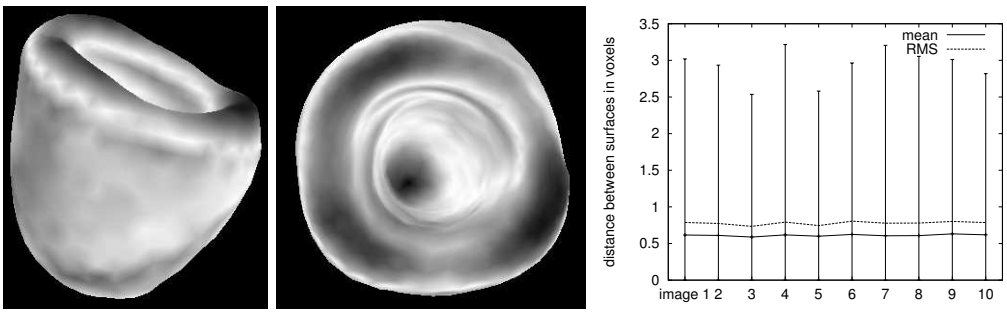


FIG. 2.11 – Left : color distance map. Right : mean and RMS distance between the recovered surface and an isosurface extracted from the NCAT source images.

Non-synthetic images segmentation

We have segmented SPECT images with a resolution of 64^3 voxels ($2 \times 2 \times 2$ mm voxels). SPECT sequences are covering one heart cycle over 8 time frames similarly to simulated images. The 4D cup-shaped LV myocardium model built for simulated images and shown on top of figure 2.13 is also used for segmenting those images. It consists of 500 vertices which is well

suited to the representation of the LV in low resolution SPECT images.

We compared images of 5 healthy patients with normal endocardium blood perfusion and one pathological patient with an abnormal perfusion due to ischemic zones.

Due to the high contrast of the LV in SPECT images, gradient forces are chosen as external forces. The 4D model is roughly initialized in a given reference position. Rigid followed by similarity registration are first performed to compensate for the differences in location and size between patients. Globally constrained deformations based on affine transformation are then applied. By progressively increasing the locality factor and lowering the external forces range, local deformations only affect a restricted neighborhood.

The segmentation of a pathological case is performed in the same way than healthy cases. Due to the poor perfusion of the myocardium in some pathologies the image contrast is much lower and the contours are weaker. The model rigidity then becomes critical for a proper reconstruction of the heart boundaries. Figure 2.12 shows the intersection of the deformed 4D model with 4 short axis slices of the pathological patient SPECT image. This figures compare the segmentation outcome with (left figure) and without (right figure) temporal constraints. Clearly segmentation errors are larger without temporal constraints (see the endocardium in slice 26 at time 2, or the epicardium at slice 32 time 4 for instance).

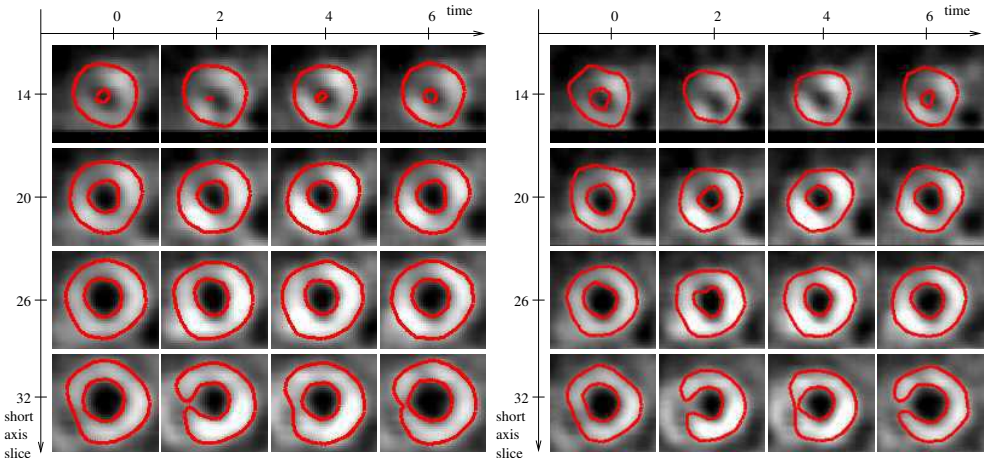


FIG. 2.12 – 4D model deformed with (left) and without (right) time constraints.

Figure 2.13 shows a frontal view of the 4D models. The figure shows the reference model (top), an healthy patient model (center) and the pathological patient model (bottom). The deformed models in 4D shows a much more regular aspect than the reference model obtained by 3D segmentation. The periodic nature of the motion clearly appears between the first and the last instant for the 4D deformed models (center and bottom line). Also the difference in the motion amplitude between healthy and pathological cases clearly appears on the reconstructed surfaces.

Conclusions

The segmentation accuracy was evaluated on simulated SPECT images for which a ground truth (organ surfaces and volumes) is known. We have shown a sub-voxel accuracy in the segmentation of the LV surface and LV chamber volume which is sufficient for clinical assessment of the cardiac function. Application of the 4D segmentation algorithm to non-synthetic SPECT, MR and US images have shown the versatility of our approach.

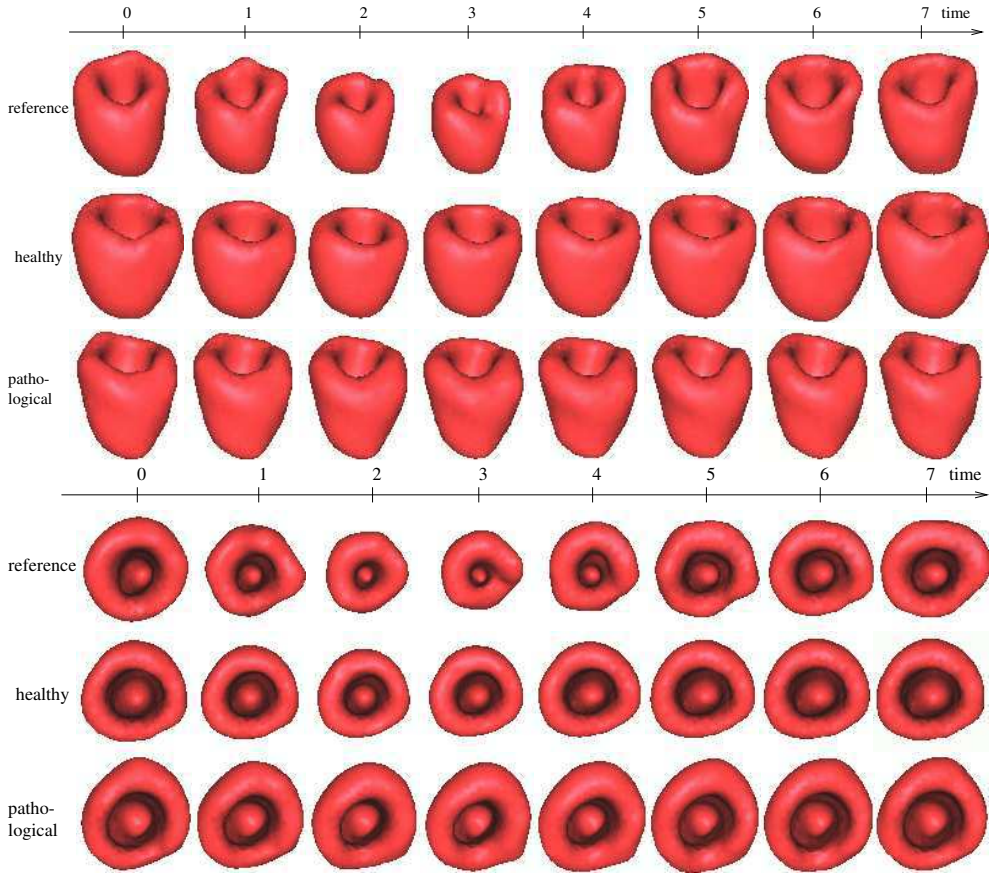


FIG. 2.13 – Frontal (three top rows) and coronal views (three bottom rows) of myocardium 4D models. In each direction, from top to bottom : reference model obtained by 3D segmentation (top), healthy case (center), and pathological case (bottom).

To increase the accuracy of the segmentation for a given image modality, it is possible to specify different set of parameters (rigidity, locality) at different parts of the mesh, for instance near the base of the LV or the apex of the endocardium where high curvature points make the surface deformation more difficult to control. For image modalities like MR or US, more sophisticated external force definition based on the matching of intensity profiles [42] or blocks [188] can also significantly improve the detection of image boundaries.

Moreover, the segmentation robustness can also be strengthen, especially when dealing with patient with strong pathologies, by relying on statistical shape appearance modeling for firing alarms when the current shape and image model is far from the expected one [165]. In such cases, one can decide to start again the segmentation based on a different initial model or with different set of parameters.

2.1.4 Current trends in model based segmentation

One current trend in model-based segmentation of medical images it to combine different data sources, when possible, in order to build more reliable and automated segmentation tools. Multi-modality images are carrying different kind of information that can be exploited in the data term to compensate for deficiencies of one modality with another one. This usually requires a prior registration processing of the different images so that the model can take into account data information translated into a single frame. Yet, last generation imaging devices are capable of performing simultaneous acquisitions with different modalities such as X-ray scanner combined with SPECT nuclear imaging.

Strengthening the model prior is also a way of improving the model behavior, especially when data are noisy or lacking. Many algorithms integrate different kinds of priors, especially statistical models built from the analysis large image databases. In our work, we have shown how we have been using shape priors to help in recovering the surface of known organs. Many other works are using higher order statistics such as variability of the mean shape through modal analysis or Principal Component Analysis of a set of shapes. More prior has also been introduced, using expected gray level distribution in the images or even known variations of these gray level samples.

Another difficulty with deformable models is often the need for fine-tuning all the algorithms parameters for a given application. This is usually a manual and tedious process. Recent work have reported the automatic tuning of the deformation algorithm by exploiting a large amount of computing resources to perform multiple segmentations while covering the parameters space and searching for the parameters set that yields to the minimal energy [158].

These different improvements in segmentation algorithms have been made possible by the joint release of image databases needed for extracting statistics and priors from references images and large computing resources needed for computing the statistical atlases and searching the algorithms parameter space.

2.2 Validating medical image analysis procedures

2.2.1 Application : brain atrophy measurement from MR Images

This research work was led at the Montreal Neurological Institute attached to the McGill University in collaboration with Dr. Louis Collins. The motivation of this study was the measurement of the effect of interferon beta drug used against Multiple Sclerosis (MS) through longitudinal MR acquisitions of the brain. Interferon beta drugs are controversial. They are recognized to reduce the output of patient's interferon gamma, that is known to stimulate the destructive activity of the immune system, but the way they achieve this result is not well understood.

A number of neurodegenerative diseases are characterized by brain tissue loss. In particular, multiple sclerosis is a neurological disorder that is associated with recurrent attacks of focal inflammatory demyelination resulting in loss of brain matter. Brain atrophy has similarly been reported as a consequence of Alzheimer's Disease [68, 67] and Schizophrenia [196] progression. Therefore, several methods for estimation of brain volume from magnetic resonance images have been proposed in the literature. Our first objective was to evaluate the robustness of state-of-the-art brain volume computation methods versus several parameters such as image acquisition settings and availability of multiple images. Our second objective was to correlate the brain volume measurements with disease severity in several patient groups with or without interferon beta treatment.

2.2.2 Validation

This section demonstrates the effort deployed to validate a medical image analysis procedure built by chaining several image analysis algorithms. It is often difficult to evaluate the effect of therapy in clinical trials of brain diseases when there is a high degree of variability in clinical signs and symptoms that vary over time and between individuals. Because of such variability, large numbers of subjects are required in order to detect subtle differences in treatment groups that can be qualitatively interpreted as different degrees in a pathology progression. A number of early quantitative studies have demonstrated that atrophy of different brain structures can in fact be quantified and that these measures correlate to varying degrees with disability [117, 89, 53], intellectual and memory dysfunction [168], dementia [100] and lower mean scores for neuropsychological tests [36, 41]. It is therefore important to develop a metric to estimate whole brain volume, validate the metric, and use it to characterize brain atrophy. Since the average rate of atrophy is approximately 1.0% per year in MS patients [119], very precise metrics are required to detect changes in cerebral volume over periods of time shorter than one year and to find correlations with disability.

Normal aging [128, 93, 179, 162, 88], anorexia [109], corticosteroid administration [15], acute dehydration [127], excessive alcohol intake [170, 163], can also affect brain volume or CerebroSpinal Fluid (CSF) volume, and thus confound the measure of disease burden. These factors were used as exclusion criteria for subject selection in the experiments described below.

Section 2.2.3 shows the workflow applied to brain MR images in order to compute voxel-based atrophy measures. It describes each stage that was carefully experimented and selected to remove from images any bias resulting from the image content (gray-level distortion) or the acquisition conditions (acquisition parameters). Sections 2.2.4 shows how the measures repeatability and validity where validated using different databases of images with controlled acquisition parameters. Four different brain volume measures have been computed over two image databases summing up to 712 brain image volumes acquired with different parameters. Finally, section 2.2.5 shows how the measure where validated in a clinical context in the absence

of ground truth. The Brain Intra-Cranial Capacity Ratio (BICCR) was evaluated on MRI data from age and sex-matched normal subjects and patients with multiple sclerosis.

2.2.3 Image processing

In this section, we present a fully automated technique to compute a head-size normalized brain volume metric based on the ratio of brain parenchyma to intra-cranial volume : BICCR (Brain Intra-Cranial Capacity Ratio). The brain volume computation methods studied are voxel-based. Each image voxel is classified as a brain tissue, CSF or image background. The number of voxels in each class multiplied by the elementary voxel volume gives an estimate of actual tissue and CSF volumes. The classification of voxels is performed using multimodal MR images carrying complementary information when available. Figure 2.14 provides an illustration showing input T_1-weighted, T_2-weighted, and PD-weighted images and a color-coded classification result.

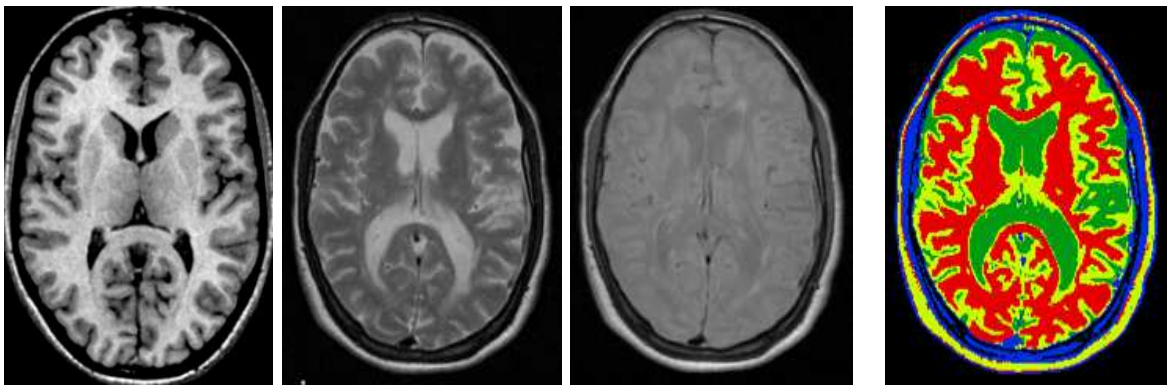


FIG. 2.14 – Brain image classification example. Left : T_1-weighted, T_2-weighted and PD-weighted images of the same brain. Right : color-coded classified image (white matter in red, gray matter in light green, and CSF in dark green).

As a voxel-based approach, this method requires preliminary processing stages that aim at correcting the image intensities by minimizing the bias and the noise due to the acquisition device. Images are also registered in a common Talairach space by a linear registration procedure. This simplifies the anatomical masking steps described below. In addition, this step ensures that the size differences between individuals are compensated for so that absolute tissue volumes are invariant to head size. Figure 2.15 diagrams the processing stages required for brain volume measurement. These processings are described in detail in the following sections. The procedure begins with a gray-level non-uniformity correction to reduce intensity biases introduced during MR acquisition. Data is then registered in a standardized brain-based stereotaxic coordinate system to facilitate anatomically driven data manipulation in subsequent processing steps. Each image volume is then intensity normalized and spatially smoothed with anisotropic diffusion filter in preparation for tissue classification. After identification of the different tissue types, the resulting volume is masked to remove skull and scalp, and finally the normalized brain volume is computed.

Artifact and noise reduction

Non-uniform intensity correction. The inhomogeneity of the MR acquisition device magnetic field introduces a bias perceptible in images as a continuous variation of gray-level intensities.

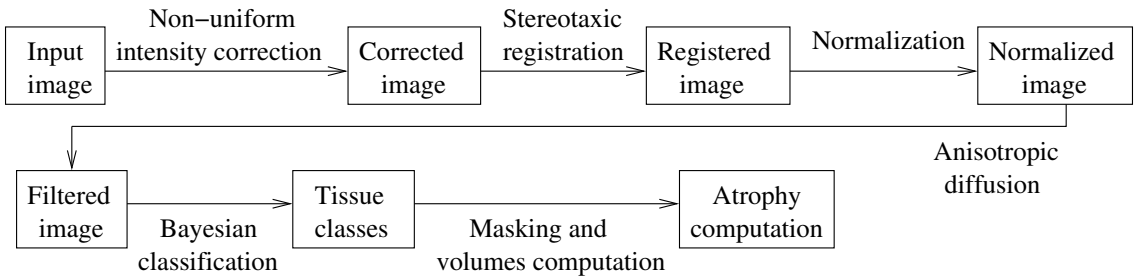


FIG. 2.15 – Diagram of the brain volume computation method stages.

The non-uniform intensity correction algorithm iteratively proceeds by computing the image histogram and estimating a smooth intensity mapping function that tend to sharpen peaks in the histogram. The intensities for each tissue type thus have a tighter distribution and are relatively flat over the image volume [189].

Intensity normalization. In preparation for intensity-based classification, each image is intensity normalized onto an average PD- or T_2-weighted target volume already in stereotaxic space. An affine intensity mapping is estimated that best maps the histogram of each image onto the template. After normalization, the histogram peaks corresponding to each tissue class have the same value in all images.

Anisotropic diffusion. It has been shown that the application of an edge preserving noise filter can improve the accuracy and reliability of quantitative measurements obtained from MRI [129, 210]. Anisotropic diffusion was pioneered by Perona and Malik [161] and was generalized for multidimensional and multispectral MRI processing by Gerig *et al* [72]. This filtering stage reduces voxel misclassification due to noise and minimizes the speckled appearance sometimes apparent in the resulting classified images.

Image registration

Image registration in a common Talairach space is needed both to compensate for gross volume variations between individuals and to align all images in the same direction so that relevant image subvolumes can be extracted for brain volume measures. A linear registration procedure based on the optimization of an objective function derived from a cross-correlation measure by a simplex algorithm is used [40, 38]. Only pose and scale parameters are optimized so that no non-linear deformation that would affect the relative volumes of tissues and CSF are allowed. After linear registration, all images are aligned in the same reference frame so that scale differences between individuals are eliminated. Thus, resulting brain volume values are not biased by inter-patient size variability.

Stereotaxic space. The target image for stereotaxic registration is a template image built from an earlier study [125] involving the averaging of more than 300 MR images. To achieve robust results, the characteristic features must be insensitive to small perturbations and noise, and independent of the orientation and position of the object. The registration algorithm proceeds to a coarse-to-fine approach by registering subsampled and blurred images. Furthermore, both blurred intensity and gradient images are used. Convolution with a 3D isotropic Gaussian kernel maintains linearity, shift invariance, and rotational invariance of features to be registered in the cross-correlation process.

Longitudinal registration. Since the goal of longitudinal studies is to quantify volume changes of the brain, the brain-based registration procedure described above is not ideal. A more accurate registration procedure is obtained by aligning a subject image onto itself rather than using

a stereotaxic target as specified above. This so called *longitudinal registration* involves three stages :

- **Stereotaxic registration**. Each image of the longitudinal study is first registered in Talairach space as described above.
- **Build target**. A subject-specific target is built by averaging all images of that subject in Talairach space. This ensures that all images contribute equally to the longitudinal target.
- **Longitudinal registration**. A scale-invariant feature of each image from each longitudinal study is registered onto the subject-specific average target. This ensures that each image of the longitudinal study is submitted to the same processing.

It is possible that the size of the voxels may change slightly over time due to shifts of the linear gradients. Since the skull is not expected to change over time, we use it as a scale-invariant feature. When T_2-weighted MR images are available, the skull is rather easy to distinguish from the background. Indeed, a single threshold and mathematical morphology operations are sufficient. In the absence of T_2-weighted images, we rely on a more complex model-based segmentation algorithm of the skull [121].

After longitudinal skull-based registration, the subtle orientation and scale differences are removed from the images. An additional stage of intensity correction is also computed to reduce the intensity variability between images as described in figure 2.16. Each source image is subtracted from the average target. Each resulting image gives a map of intensity differences between sources and target. The non-uniform intensity correction algorithm is applied to compute a smooth transformation flattening the difference intensities. The correction computed is applied on each source image to correct the residual intensity bias.

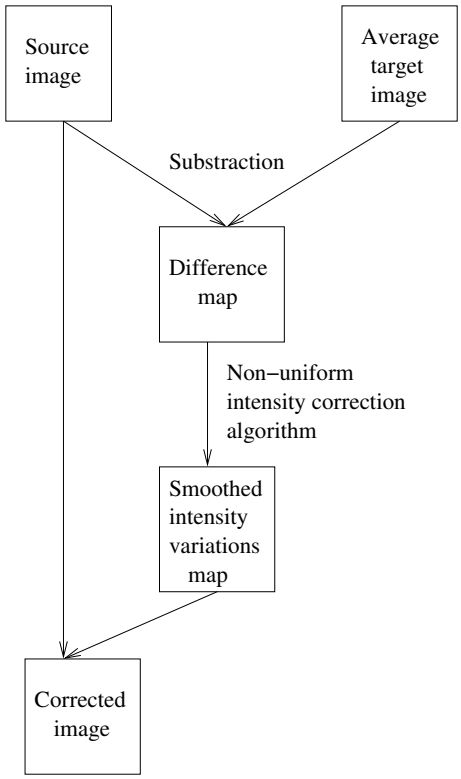


FIG. 2.16 – Residual intensity bias correction.

Classification and brain volume measures

The preliminary stages described above aim at removing any differences in input images due to patient orientation in the acquisition device, noise and sampling artifacts. Image voxels are then submitted to a Bayesian classifier [55] outputting brain tissues, CSF, and background.

Classification. The Bayes classifier is trained manually by picking a set of 20 volumes randomly among all volumes to be processed. On each sample image, 50 voxels belonging to each class (background, white matter, gray matter and CSF) are selected by hand. The resulting 4000 samples are used to compute each class mean intensity and the covariance matrices. Depending on their availability, T_1, T_2, and/or PD-weighted images can be used for the multivariate training and classification. In the following experiments, we compare T_1 versus T_2+PD and T_1+T_2+PD classification. Once trained, the Bayesian classifier proceeds by estimating for each image voxel the class corresponding to the highest likelihood, given the intensities of that voxel in all weighted images available [55]. The probability for a voxel to belong to a class depends on the distance between the voxel intensity and the class mean corrected by the class standard deviation.

Masking. Different brain volume measures have been proposed in the literature. Each of them only takes into account a particular anatomical region of the brain. This masking operation is straightforward once all images have been aligned in brain-based coordinate system of Talairach space where the desired region can be extracted with a single mask aligned in Talairach space. The external tissues of the skull and the scalp also have to be masked before voxel counting. In T_2-weighted MR images, the brain is isolated by simple thresholding and morphological operations. If T_2-weighted images are not available, we rely on a deformable surface method [121]. A rough brain skull mask is first extracted. A deformable surface is initialized inside the skull labels. It deforms in a coarse-to-fine strategy using different resolution levels to fit the internal part of the skull while getting rid of skull's holes [141]. Figure 2.17 displays an example of the result produced by the image processing pipeline : on the left column, two images from a same patient acquired with a 6 months interval are classified (center column). A subtraction overlaid on a source image (right columns) eases the identification of volume varying areas.

Brain volume measures. In order to normalize the brain volume metric for differences in head size, the fraction of brain tissues volume over CSF volume is estimated. Brain tissues include gray and white matter, and, in pathological cases, lesions. One such normalized brain volume measure is the BICCR (Brain Intra-Cranial Capacity Ratio) coefficient defined as :

$$\mathrm{BICCR} = \frac{V_{\mathrm{tissues}}}{V_{\mathrm{tissues}} + V_{\mathrm{CSF}}}$$

where V_{tissues} is the brain tissue volume contained in a horizontal slab limited by the top of the pons inferiorly ($z = -22mm$, in Talairach coordinates) and superiorly by a plane cutting through the centrum semi-ovale ($z = 58mm$). This yields an anatomically equivalent 80mm thick volume across all subjects that contains most of the cerebrum. V_{CSF} is the volume of CSF within the ventricles, cisterns and cortical sulci in the same horizontal slab. In order to study the influence of extra-cerebral CSF, we also evaluated a related measure :

$$\mathrm{BICCR}' = \frac{V_{\mathrm{tissues}}}{V_{\mathrm{tissues}} + V'_{\mathrm{CSF}}}$$

where V'_{CSF} is the volume of CSF in the ventricles, cisterns and all extra-cerebral CSF within the same anatomical region.

For this study, we have computed two other brain volume measures similar to two measures described in the literature : the BPF (Brain Parenchymal Fraction) introduced by Fisher and Rudick [60, 174], and method of Losseff *et al* [119]. The BPF is defined as the ratio of brain

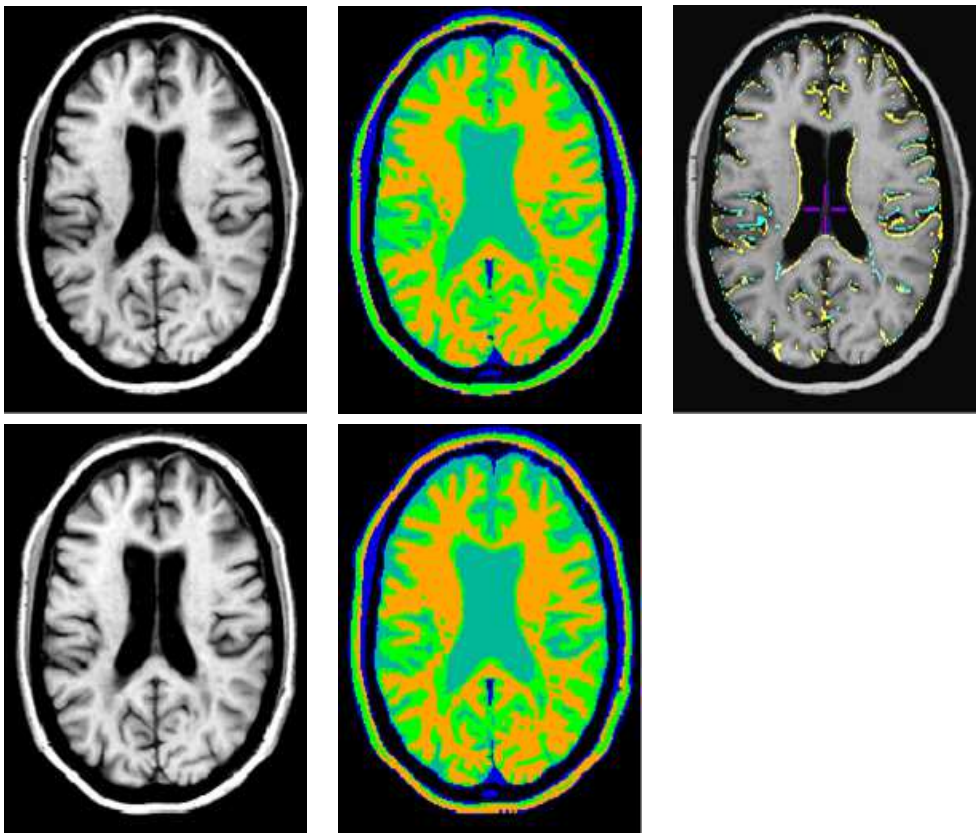


FIG. 2.17 – Sample result of the image processing pipeline : input images (left) are registered and classified (center) to determine subtle brain matter variations over the studied period (right).

tissue volume to the total volume enclosed by the brain surface :

$$\text{BPF} = \frac{V_{\text{brain tissues}}}{V_{\text{brain tissues}} + V_{\text{brain CSF}}}$$

and as such, does not contain extra-cerebral CSF that is not in sulci, as might be the case in greatly atrophied brains. The Losseff technique involved manual identification of the brain and lateral ventricles on four 5mm thick transverse slices through the center of the brain. By modifying the definition of the horizontal slab, we have adapted the BICCR procedure to estimate a metrics similar to the BPF and the method of Losseff. In the former, the slab is increased to enclose the whole brain. In the second, the slab is reduced to 20mm thick, centered on the ventricles.

2.2.4 Comparing brain atrophy measurement methods

The BICCR, BICCR', BPF and Losseff's metric are compared to determine the most relevant metrics for our study. The sensitivity of each brain volume measure is tested with regards to three factors :

- The robustness versus acquisition parameters. Various MR volumes are acquired with different resolutions, slice orientations, TE and TR parameters.
- The reproducibility is estimated by computing brain volume on multiple images of the same subject acquired in the same conditions using a scan-rescan paradigm.
- The need for multivariate data is demonstrated by using T_1-, T_2-, PD-weighted images, or only a subset of those, in the classification procedure.

Two sets of experiments based on two different image databases have been used to compute the brain volume measures. The former database comes from the University of Siena and is described in [190]. The latter originates from the International Consortium for Brain Mapping [125].

Databases description

Siena database. The Siena database is composed by a set of T_1-weighted images acquired from 16 different subjects using a Philips NT 1.5T. Eight images with different scanning parameters were acquired on each subject. Each series of scans was applied to each subject on two different occasions thus leading to 256 ($16 \times 8 \times 2$) images. For each data set, scans 1 to 6 are acquired with a 1mm to 6mm slice thickness, T_1-weighted axial 2D fast field echo, TE=11ms, TR=35ms, flip=40°, NAcq=1. Scan 7 is a 3mm slice thickness axial volume fast field echo, TE=3ms, TR=20ms, flip=30°, NAcq=1. Scan 8 is the same as scan 7 but with coronal slices. For all scans, the in-plane resolution is 1mm × 1mm and the inter-session interval is between 1 and 7 days. Half of the subjects were scanned with the slice thickness range order reversed to control for order effect.

With the wide variety of acquisition parameters, the Siena database allows a robustness study of the different brain volume measures. Computation reproducibility is estimated by computing the brain volume measure in each pair of volumes acquired with the same parameters. Robustness is probed computing brain volume measures on the 16 images of a same patient with different parameters. However, only T_1 images are available in this database. This corresponds to the least desired case for the classification algorithm and the longitudinal registration based on skull extraction.

ICBM database. The ICBM database is composed of a set of 152 subjects, each acquired with T_1-, T_2-, and PD-weighted MR images. All subjects have no known pathologies. All images are acquired using an $1 \times 1 \times 1$ mm voxel size axial slices. T_1-weighted scans are acquired

with a fast field echo sequence, TE=10ms, TR=18ms, flip=30°, NAcq=1. T_2- and PD-weighted scans are acquired with TE=120ms and 34ms respectively, TR=3.3s, flip=90°, and NAcq=1. As multivariate classification is used in tissue identification, the availability of different volumes influences the brain volume computation. This database permits the evaluation of single or multivariate classification.

Results using the Siena database

Two experiments were completed to study the behavior of the brain volume metric. The former measures the stability of the brain volume measure versus acquisition parameters. The latter studies the reproducibility of the brain volume computation over different images acquired with fixed parameters.

Robustness versus acquisition parameters

For the 32 sets of 8 acquisitions with different MR parameters, figure 2.18 shows the brain volume measure of each patient versus the acquisition number (from 1 to 8 as described above). Figure 2.19 is another display of the same data showing only the mean, minimum, and maximum brain volume values for each of the 32 sets. It appears that the acquisition parameters and slice thickness have very little effect on the brain volume measure variability. The typical variation between minimum and maximum value for a given parameter set is about 0.05 for BICCR and our implementation of BPF measures. Our implementation of Losseff and BICCR' measures are slightly less robust.

The mean and standard deviation are also estimated for each brain volume measure of each subject among the 16 images acquired. Table 2.1 shows the mean standard deviation obtained over the 16 subjects for each brain volume measure. Consistently, BICCR' shows a larger variability. The other measures are very close although Losseff's has a small advantage.

Measure	Mean standard deviation
BICCR	0.017
BICCR'	0.022
BPF-like	0.017
Losseff-like	0.014

TAB. 2.1 – Mean standard deviation of brain volume measures.

Brain volume computation reproducibility

The second experiment tests the reproducibility of the brain volume measure computation in a scan/rescan paradigm. For each given acquisition parameter and each measure, brain volume is computed on the two images available. The coefficient of variability (CoV) is the used as a normalized measure of the standard deviation. The CoV is computed as :

$$\mathrm{CoV} = 100\% \frac{\text{standard deviation}}{\text{mean}}. \tag{2.4}$$

The mean CoV over the 16 patients is computed as summarized in table 2.2. The CoV is in the order of 0.3% except for the 1mm slice thickness acquisition where it is significantly higher. This surprising result, already reported in [190], might be explained by the longer acquisition time of this volume (about 18 minutes) leading to more probable patient motion artifact in the image. Also, the acquisition number 8 (coronal slices) performs slightly better than the others.

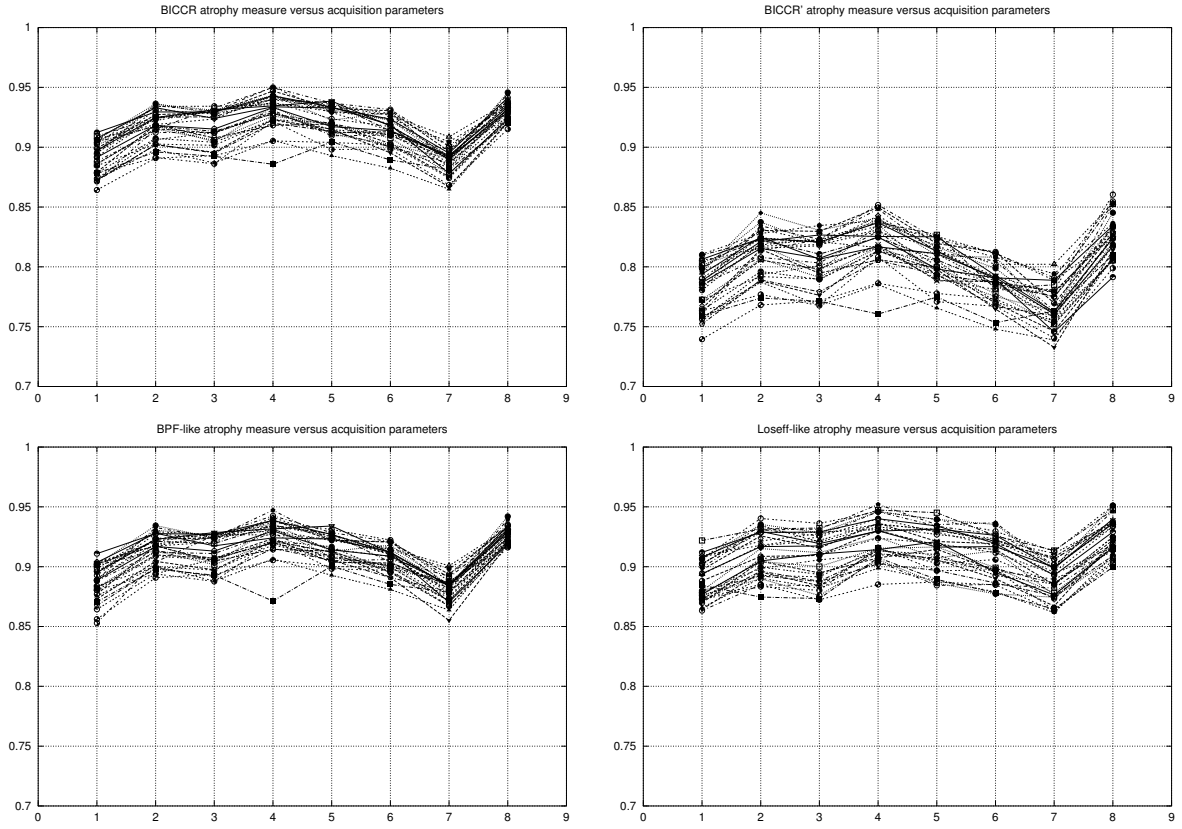


FIG. 2.18 – Brain volume value versus acquisition parameters for BICCR (top left), BICCR' (top right), BPF-like (bottom left) and Losseff-like (bottom right measures).

Acquisition	Measure	Mean CoV	Acquisition	Measure	Mean CoV
1	BICCR	0.54%	2	BICCR	0.25%
	BICCR'	0.77%		BICCR'	0.39%
	BPF-like	0.63%		BPF-like	0.30%
	Losseff-like	0.44%		Losseff-like	0.31%
3	BICCR	0.25%	4	BICCR	0.35%
	BICCR'	0.37%		BICCR'	0.62%
	BPF-like	0.25%		BPF-like	0.42%
	Losseff-like	0.29%		Losseff-like	0.39%
5	BICCR	0.23%	6	BICCR	0.34%
	BICCR'	0.38%		BICCR'	0.66%
	BPF-like	0.24%		BPF-like	0.37%
	Losseff-like	0.27%		Losseff-like	0.32%
7	BICCR	0.34%	8	BICCR	0.23%
	BICCR'	0.61%		BICCR'	0.40%
	BPF-like	0.34%		BPF-like	0.25%
	Losseff-like	0.34%		Losseff-like	0.21%

TAB. 2.2 – Mean standard deviation and CoV of brain volume measures.

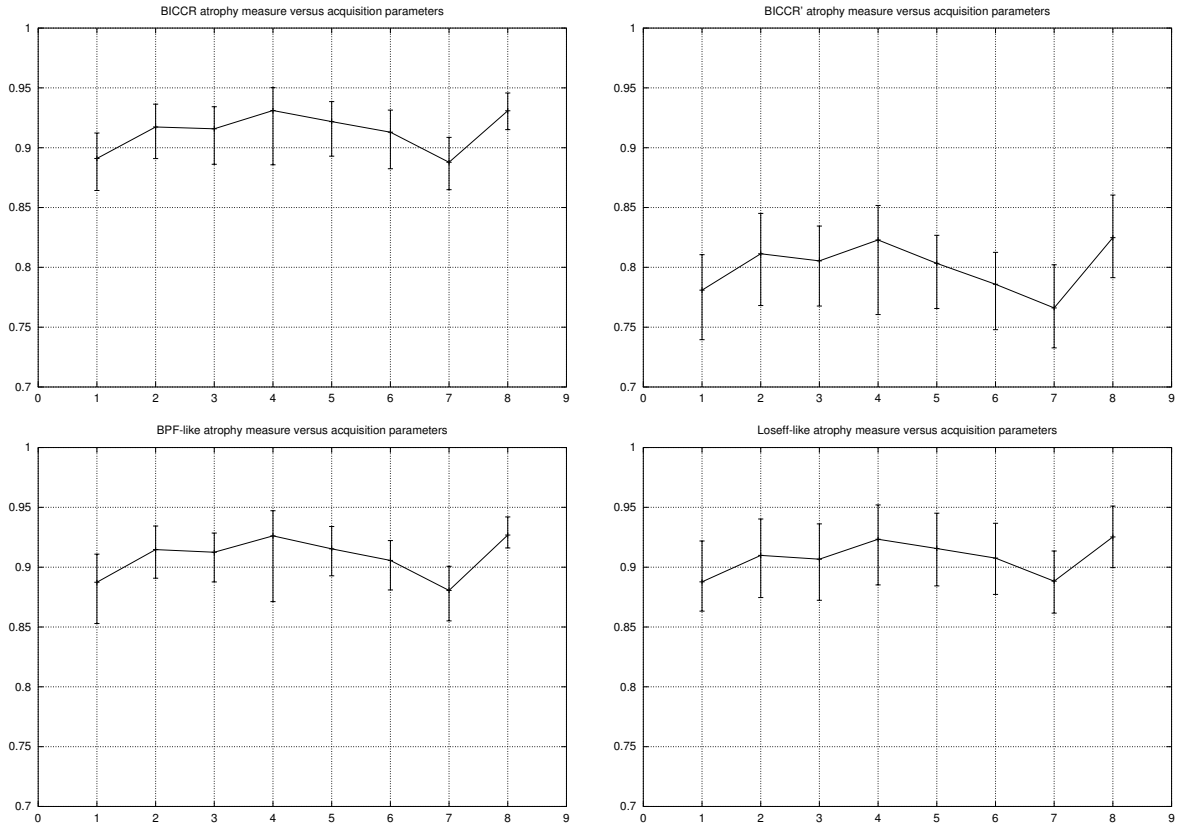


FIG. 2.19 – Brain volume value versus acquisition parameters for BICCR (top left), BICCR' (top right), BPF-like (bottom left) and Losseff-like (bottom right measures).

Table 2.3 summarizes the results by giving the mean CoV for each brain volume measure over all experiments. The mean is close to 0.3% for all measures but BICCR' (that includes extra-cerebral CSF).

Measure	mean CoV
BICCR	0.31%
BICCR'	0.53%
BPF-like	0.32%
Losseff-like	0.32%

TAB. 2.3 – Mean CoV for each brain volume measure.

Finally, figure 2.20 shows the scatter plot of points for which the `x` and `y` coordinates are the two brain volume measurements made. The $y = x$ (solid) line and the $y = x \pm 0.01$ (dashed) lines are overlaid on the scatter plot. Every brain volume measure show the same reproducibility behavior apart from the BICCR' measure demonstrating a slightly higher variability. Including extra-cerebral CSF leads to a less robust brain volume measure.

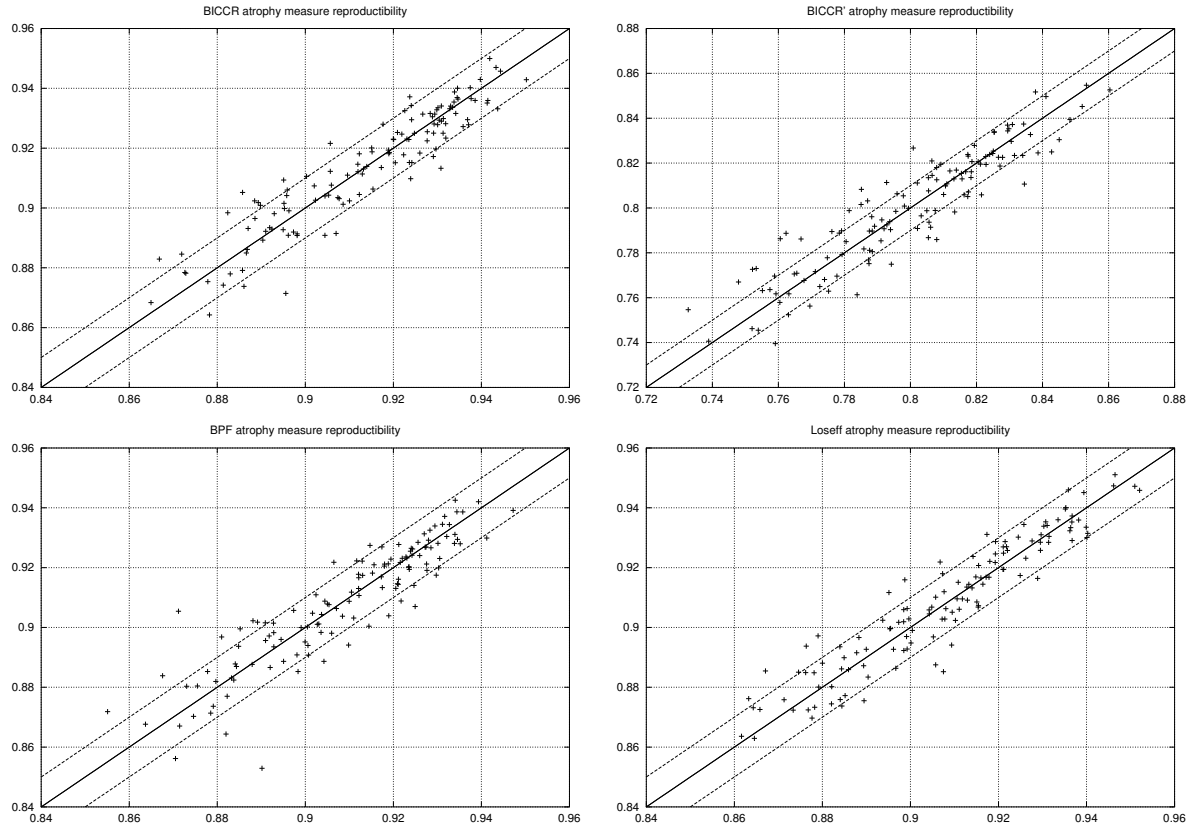


FIG. 2.20 – Scatter plot of points for which the `x` and `y` coordinates are the two brain volume measurements made. The $y = x$ (solid) line and the $y = x \pm 0.1\%$ (dashed) lines are overlaid.

Results using the ICBM database

On the ICBM database, the classifier was run three times : using only T_1-weighted images, using both T_2- and PD-weighted images, and using T_1-, T_2- and PD-weighted images. This aims at studying the effect of adding images in the multivariate classification step for the brain volume computation method. The ideal case corresponds to the situation where T_1-, T_2-, and PD-weighted images are available. However, this is not common in clinical practice. Often, only T_1-weighted images are used. In some cases such as clinical trials of multiple sclerosis, both T_2- and PD-weighted images are acquired. Table 2.4 shows the mean and standard deviation of the brain volume value for each tested measure. The mean value of the brain volume measure over the 152 images processed appears to be quite stable, although a small shift (< 0.02) of the mean value exists. However, the mean standard deviation over these images shows greater variations. Standard deviations computed from T_2+PD and T_1+T_2+PD classification are very similar but T_1 only classification leads to significantly larger variations.

	T_1+T_2+PD		T_2+PD		T_1	
Measure	Mean	s.d.	Mean	s.d.	Mean	s.d.
BICCR	0.941	0.016	0.930	0.016	0.937	0.050
BICCR'	0.891	0.029	0.876	0.027	0.877	0.071
BPF-like	0.943	0.015	0.933	0.015	0.936	0.051
Losseff-like	0.925	0.019	0.912	0.022	0.930	0.044

TAB. 2.4 – Mean and standard deviation of brain volume values for different brain volume measures using only T_1, T_2+PD, or T_1+T_2+PD images for classification.

Further analysis yields better insight on the differences between the different classification procedures. Figure 2.21 displays the correlation between brain volume measures computed using T_2+PD images versus T_1+T_2+PD images. As expected from results shown in table 2.4, the disparity is very small.

Let $v_i^{T_1}$, $v_i^{T_2+PD}$, $v_i^{T_1+T_2+PD}$, represent one of the brain volume measures computed for case $i \in [1, 152]$. The disparity between each classification type is estimated by computing the CoV for each measure pair :

$$di_{T_2+PD}^{T_1+T_2+PD} = \mathrm{CoV}(v_i^{T_2+PD}, v_i^{T_1+T_2+PD}),$$
$$di_{T_1}^{T_1+T_2+PD} = \mathrm{CoV}(v_i^{T_1}, v_i^{T_1+T_2+PD}),$$
$$\text{and } di_{T_1}^{T_2+PD} = \mathrm{CoV}(v_i^{T_1}, v_i^{T_2+PD}).$$

The mean coefficient of variation for each brain volume measure pair is computed :

$$m_{T_2+PD}^{T_1+T_2+PD} = \frac{1}{152}\sum_{i=1}^{152} di_{T_2+PD}^{T_1+T_2+PD},$$

$$m_{T_1}^{T_1+T_2+PD} = \frac{1}{152}\sum_{i=1}^{152} di_{T_1}^{T_1+T_2+PD},$$

$$\text{and } m_{T_1}^{T_2+PD} = \frac{1}{152}\sum_{i=1}^{152} di_{T_1}^{T_2+PD}.$$

If the mean disparity is high, then that means that there is a large difference, on average, between the two methods. If the value is low, then there is agreement. If the value is very low,

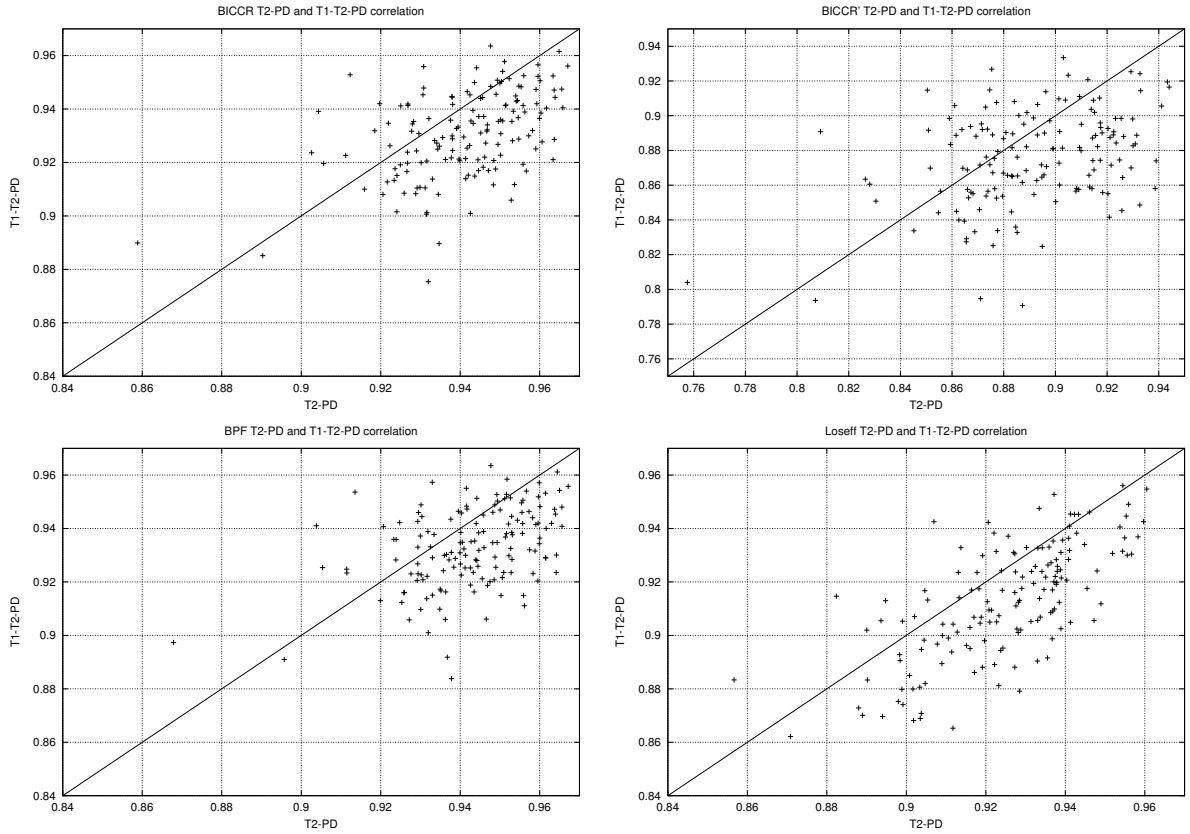


FIG. 2.21 – Correlation of brain volume measures computed using T_2+PD images versus T_1+T_2+PD images.

then it indicates that the two methods provide essentially the same information and therefore, the least expensive technique (in terms of acquisition or computation) is desirable.

The numbers in Table 2.5 demonstrate a very weak variability between T_2+PD and T_1+T_2+PD measurements. However, the variability between T_1 and T_2+PD or T_1+T_2+PD measurements is at least twice as large.

Measure	$m_{T_2+PD}^{T_1+T_2+PD}$	$m_{T_1}^{T_1+T_2+PD}$	$m_{T_1}^{T_2+PD}$
BICCR	0.10%	0.23%	0.25%
BICCR'	0.19%	0.39%	0.38%
BPF-like	0.10%	0.23%	0.24%
Losseff-like	0.11%	0.20%	0.25%

TAB. 2.5 – Mean CoV of brain volume computed using only T_1, T_2+PD, or T_1+T_2+PD images for classification.

Anisotropic diffusion effect

Tables 2.6 and 2.7 display the results of experiments similar to those of tables 2.4 and 2.5, however anisotropic diffusion was removed from the preprocessing stages. Anisotropic diffusion is an efficient, yet costly method to reduce image region inhomogeneities. This experiment confirms the former result : classification using T_2, and PD-weighted images only or by adding the T_1-weighted image are very close. There is therefore no real need for an additional T_1 acquisitions. However, at least T_2, and PD-weighted images are desirable.

	T_1+T_2+PD		T_2+PD		T_1	
Measure	Mean	s.d.	Mean	s.d.	Mean	s.d.
BICCR	0.932	0.017	0.929	0.016	0.933	0.053
BICCR'	0.877	0.029	0.873	0.028	0.872	0.074
BPF-like	0.934	0.016	0.931	0.015	0.932	0.053
Losseff-like	0.915	0.023	0.911	0.023	0.927	0.046

TAB. 2.6 – Mean and standard deviation of brain volume values for different brain volume measures using only T_1, T_2+PD, or T_1+T_2+PD images for classification, without anisotropic diffusion.

Measure	$m_{T_2+PD}^{T_1+T_2+PD}$	$m_{T_1}^{T_1+T_2+PD}$	$m_{T_1}^{T_2+PD}$
BICCR	0.03%	0.24%	0.28%
BICCR'	0.05%	0.36%	0.41%
BPF-like	0.03%	0.24%	0.27%
Losseff-like	0.04%	0.24%	0.27%

TAB. 2.7 – Mean CoV of brain volume computed using only T_1, T_2+PD, or T_1+T_2+PD images for classification, without anisotropic diffusion.

Discussion

The sensitivity of the various brain volume measures have been tested with regard to four factors :

- the robustness versus acquisition parameters,
- the computation reproducibility,
- the need for multivariate data,
- and the consequence of anisotropic diffusion in preprocessing.

In general, the BICCR and the BPF-like measures both tend to give reliable results while the Losseff's measure shows a slightly higher variation. Including extra-cerebral CSF leads to a less robust brain volume measure. That might be explained by the difficulty of precisely masking the extra-cerebral CSF from the image background. Thus, the BICCR' measure does not compete with the other ones. However, on-going work tends to prove that BICCR' is more sensitive to disability in patients suffering from multiple sclerosis.

The acquisition parameter setting does have a small effect on the mean brain volume value, and this bias is consistent between the different brain volume measures. This result implies that no changes in the acquisition parameters should occur in a longitudinal study.

In terms of brain volume computation reproducibility, the brain volume measures' CoV is in the order of 0.3% except for the 1mm slice thickness acquisition where it is significantly higher. This surprising result, already reported in [190], might be explained by the longer acquisition time of this volume (about 18 minutes) leading to more probable patient motion artifact in the image. Also, the acquisition along coronal slices performs slightly better than the other ones. Coronal slicing geometry appears to give a more reliable information on brain volume variations, possibly through a reduction of the partial volume effect.

The experiment completed on the ICBM database shows that the T_1-weighted image does not significantly reduce the measures variability when both T_2- and PD-weighted images are also available. This is due to the fact that T_2- and PD-weighted are complementary enough in the classes that appear well contrasted. PD-weighted images allow to discriminate between tissue and bony structures while T2-weighted images make the CSF interface very clear. This result is confirmed when omitting the anisotropic diffusion preprocessing stage. However, the BICCR metric is significantly less precise when based only on T_1-weighted images, even though these data greatly facilitate differentiation between gray and white matter. Although the mean brain volume value computed is fairly close to the expected value when using only T_1-weighted MRIs, the disparity of measures is much higher. This means that overestimates as well as underestimates of the actual brain volume are computed.

2.2.5 Validating the approach in Multiple Sclerosis studies

BICCR, as well as other atrophy measures, has been validated for brain tissues volume variations from MR images. BICCR addresses some limitations of other techniques :

1. in theory, BICCR is more sensitive to atrophy, since it does not exclude any extracerebral CSF from processing (as does the BPF metric) ;
2. the BICCR metric is fully automated ;
3. it is robust and can be applied to less than ideal images ;
4. with small changes in the classification step, it can be applied to data acquired with different pulse sequences ;
5. the procedure to estimate BICCR is setup in a processing workflow so that it is trivial to re-analyze data using different (possibly improved) procedures or evaluation criteria, and finally

6. it has a scan-rescan accuracy comparable to the best existing techniques (0.2%) and can thus be used for longitudinal studies and the difference in BICCR can be used to estimate the rate of brain tissue loss for the subject.

The method is validated using MRI data from normal controls and then applied in a post hoc analysis of existing data to characterize brain atrophy in a group of patients with multiple sclerosis.

Data set

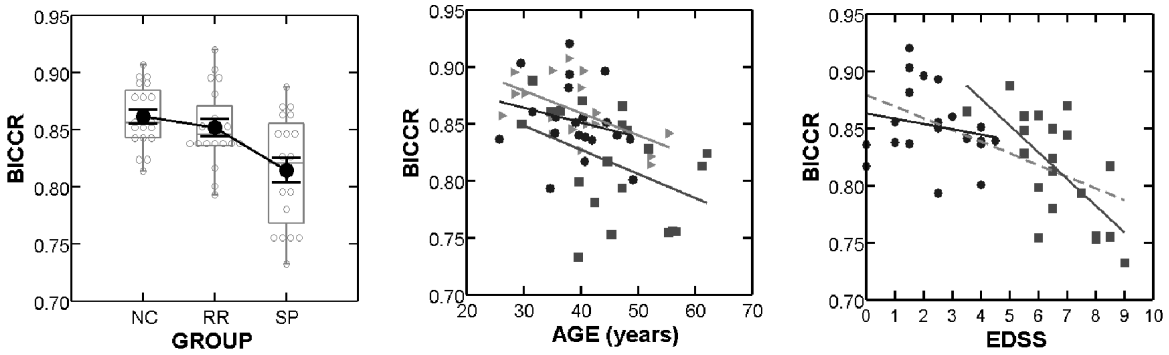
Data for this study was taken from the data base of the on-going multiple sclerosis research program at the Montreal Neurological Institute. The data base includes MR images and clinical data from normal controls and patients. The study was approved by the Montreal Neurological Institute Ethics Committee and informed consent was obtained from all participating subjects.
Subjects : In this cross sectional study, 20 normal controls (age range 22-52, 11 males, 9 females) and 40 patients with MS were selected from the database. The patients were divided into 2 groups. Twenty patients had relapsing-remitting (RR) disease, characterized by recurrent relapses with complete or partial remission (disease duration 0.5 to 23 years, Expanded Disability Status Score (EDSS [110]) range 0-4.5, age range 25-49, 7 males, 13 females). Twenty patients had secondary progressive (SP) disease characterized by progression in the absence of discrete relapses after earlier RR disease (disease duration 3 to 26 years, EDSS range 3.5-9.0, age range 29-62 years, 10 males, 10 females). No subject was included if they were treated with immunosuppresive drugs or immunomodators 2 months prior to their MRI scan, or if they suffered from anorexia, alcoholism or neurological disorders other than MS. Neurological examinations were performed on the cohort within one week of the MRI exam. Disability was evaluated using the EDSS score.
MRI acquisition : All MR data was acquired on a Philips Gyroscan operating at 1.5 T (Philips Medical Systems, Best, The Netherlands) using a standard head coil and a transverse dual-echo, turbo spin-echo sequence (TR/TE1/TE2=2075/32/90 ms, 256x256 matrix, 1 signal average, 250mm field of view) yielding PD-weighted and T2-weighted images. Fifty contiguous 3mm transverse slices were acquired approximately parallel to the line connecting the anterior and posterior commissures (AC-PC line).

Statistical Analysis

Measurement reliability is estimated with the coefficient of variability (as defined in equation 2.4). Comparisons of BICCR values between groups was achieved using Student's T for two groups, or analysis of variance (ANOVA) for three groups with Tukey's HSD (honestly significant difference) for post hoc testing. Rates of brain volume loss with respect to age (atrophy) were computed using linear regression on the cross-sectional data. The relationship between BICCR and EDSS score was evaluated using Spearman's rank correlation coefficient. The relationships between BICCR with age and disease duration were evaluated with a Pearson's correlation coefficient.

The mean BICCR value for the normal control (NC, $n = 20$) subjects is shown in left of figure 2.22 and was 86.1 ± 2.7 (mean $\pm$ s.d.). The mean CoV for scan-rescan tests of 4 NC subjects was 0.20%.

The mean BICCR value for patients with MS (83.4 ± 4.4) was significantly lower ($p = 0.005$) than normal controls. An ANOVA of the BICCR values for the NC, RR and SP groups showed a significant difference ($F = 7.9$, $p < 0.001$). The post hoc test showed that BICCR was significantly lower in the secondary progressive group (81.4 ± 4.7) than either the NC group ($p < 0.001$) or the relapsing-remitting group (85.2 ± 3.2, $p = 0.012$). The Z-score (number of



a) b) c)

FIG. 2.22 – Results : a) box and whisker plot for comparison of BICCR mean values (heavy circles), standard deviation (heavy lines) for NC, RR and SP groups : correlation of BICCR with b) age and c) EDSS. (RR=black circles, SP=gray squares, NC=gray triangles ; solid lines indicates regression line for each sub-group. Broken line indicates regression for total patient population).

standard deviations from the mean of healthy controls) was -0.371 for RR (not significantly different from NC) and -1.741 ($p < 0.001$) for SP groups. The average absolute percentage of brain tissue lost (compared to normal controls) was 1.0% for RR and 5.3% for SP groups. While the NC group was age and sex-matched to the patient groups, the SP sub-group was older than the RR sub-group (45.8 ± 9.3 versus 39.1 ± 6.0, $p = 0.02$). We therefore determined the effect of age on the BICCR metric in order to determine whether age could explain the differences between the three groups (NC, RR and SP).

As shown in middle of figure 2.22, BICCR was correlated with age in the NC group (Pearson's $r = -0.584$, $p = 0.007$). Using linear regression, the rate of atrophy in NC was estimated to be -1.96%/decade ($p = 0.007$). BICCR was similarly correlated with age in the total patient group (Pearson's $r = -0.475$, $p = 0.002$) at a rate of -2.48%/decade. Analysis of the sub-groups showed a trend to correlation with age only for the SP sub-group ($r = -0.4255$, $p = 0.06$). Using linear regression,the rate of atrophy was -1.27%/decade ($p = 0.27$) for RR and -2.13%/decade ($p = 0.06$) for SP groups. Repeating the ANOVA of BICCR values for the NC, RR and SP groups with age as a co-variate did not change the results. The age-corrected mean BICCR value for the RR group was not significantly different from NC. The age-corrected mean BICCR for the SP group remained significantly different from both the RR and NC groups.

As shown in right of figure 2.22, BICCR was negatively correlated with EDSS for the entire group of MS patients (Spearman's $r = -0.501$, $p < 0.005$). BICCR was not correlated with EDSS for RR patients but was strongly correlated with EDSS for SP patients ($r = -0.638$, $p < 0.005$).

Discussion

The results from this cross-sectional study demonstrate that there are significant differences in brain volume between MS patients and normal controls. This difference was driven by smaller brain volumes in the SP group since we did not find a significant difference in brain volume between the NC and RR groups due to the large variability in BICCR values between individuals.

The values we find compare well to the BPF values of Fisher and Rudick [174, 60]. The mean BPF and BICCR values are similar for normal controls. However, the BPF method is reported to

have a smaller intersubject variance when estimated on normal controls (approximately 0.7%). This value is much smaller than the variance for normal controls reported here (3.1%). The difference does not appear to be due to the precision of the technique. The BICCR measure has a very low CoV (0.20%) when using T2 and PD-weighted data as described here. The BICCR can thus detect changes as small as 0.47% of the total brain volume (corresponding to 5.2ml for a 1100ml brain[3]). This is similar to the BPF measure [174, 60]. The larger variance in BICCR values appears to be due to subject selection and the greater age range for our normal controls.

Although the BICCR metric presented here is similar to the BPF of Fisher, there are two differences in the way that these metrics are computed. The most important difference is that the BICCR includes all of the extra-cerebral CSF (*i.e.*, CSF between the cortex and dura, in addition to that in the sulci) as opposed to only the CSF contained within the surface enclosing the brain (which is the case for the BPF). In a simple test to compare sensitivity to disability of BICCR and a measure similar to BPF, we used morphological operators to remove extra-cerebral CSF voxels from the BICCR metric. When evaluated on 20 patients with SP MS, the magnitude of the Spearman's correlation coefficient of BICCR with EDSS dropped from -0.638 to -0.574. Using the same test in the RR group, r changes from -0.115 to -0.101.

The second difference between the BICCR and BPF methods is that the BPF classification procedure accounts for partial volumes effects[4] between tissue classes, while the BICCR method uses a discrete classification result. While the latter method should yield an unbiased result for objects that are larger than the voxel size, the BICCR method may underestimate CSF volume in regions that have dimensions on the order of the voxel size, in sulci for example.

When using volumes to compare groups, it is important to correct for differences in intracranial volume due to differences in head size. The BICCR ratio is head-size normalized by design, and accounts for the difference in head size between men and women, for example. The high precision of the BICCR method permits detection of small changes ($< 0.5\%$) in brain volume (*i.e.*, atrophy) in single subjects over a short period of time (< 1 year), and may eventually be used to monitor treatment in individual patients. Evaluation of new therapies using outcome measures based on standard clinical tests (*e.g.*, EDSS) requires large numbers of subjects to achieve the statistical power required to detect subtle between-group differences in treatment effect. The results presented here have important implications for the design of clinical trials if atrophy is deemed an acceptable surrogate for burden of disease in MS.

Conclusions

This study has confirmed that patients with MS have smaller brain volume than normal controls, and that the rate of atrophy with aging in patients with MS is greater that that of normal controls.

While pathologically non-specific, the fact that cerebral atrophy is generally correlated with irreversible neurological dysfunction makes atrophy an important surrogate to evaluate in MS using state of the art image analysis techniques. Characterization of brain atrophy will yield information complementary to other MR-based measures of focal and diffuse abnormality with varying specificity for underlying pathological changes. Brain atrophy may yield an improved endpoint for treatment trials in MS and possibly also for other neurodegenerative diseases.

[3]The standard deviation of a difference measure is $\sqrt{2}$ times larger than the standard deviation of a single measure. To be outside the 95% confidence interval for difference occuring by chance alone, the smallest detectable change is $\sqrt{2} \times 1.64 \times$ s.d. of the single measure

[4]Partial volume effects occur when more than one tissue type lies within the spatial limits of a voxel.

2.2.6 Experimental framework

The computing workflow described in section 2.2.3 is compute intensive. The total computing time needed for a single image varied between 30 minutes and 2 hours 30 minutes depending on the input images size and the algorithm parameters used. This application is also data intensive. The study on validation of atrophy measures (section 2.2.4) involved up to 712 source images (256 images from the Siena database and 456 images from the ICBM database). The later study on atrophy measurement in MS patients (section 2.2.5) involved 120 images. A third, non documented study, on interferon beta drug effect in a population of 200 normal controls and patients was the most data intensive as it required more than 2400 source images (more than 50 GB of disk space). On year's 2000 state of the art PC processors, the processing of the full MS database would have taken approximately 2 months of computation time. Of course, experiments at this scale require a lot of execution-retry tests, parameters tuning, and results comparison. Therefore, this time has to be multiplied by the number of retries needed. Moreover, the amount of data to manipulate during computations can temporarily overcome the amount of input data by a factor of 10 due to intermediate results. About half a terabyte of disk space were needed for each execution, which was a considerable amount of storage space at that time.

These image database processing procedures were made tractable by a smart use of the MNI computing resources. A common pool of approximately 30 processors were shared among the Information Technologies users through an home made workflow submission engine. The storage devices were shared over NFS in order to assemble considerable amounts of disk space by gathering RAID units. The workflow submission engine was composed of a language enabling the description of the workflow and an execution system submitting a set of input data to a given workflow and monitoring the evolution of the computations. The language was simply describing each stage of the workflow (each algorithm to apply to the data) and its data dependencies with anterior and posterior stages. The execution engine was submitting computing tasks by remote login on each of the 30 processors available. It was storing in a database the current status of computations (stage processed for each input data) in order to offer cancellation, fault tolerance and partial reruns capabilities. Indeed, a workflow being executed could be stopped, canceled or restarted from an earlier stage (given that intermediate data was stored on disk). Due to the large number of computers involved, fault tolerance was also an important issue to deal with : computer crashes could cause failure that were automatically recovered by restarting interrupted computation from the former stage.

This infrastructure made possible the longest workflow execution in less than one week. Simpler runs could take as little as one or two days to execute. These computation times where compatible with research practice and enabled the setting and tuning of experiments that would not have been possible otherwise. At a larger scale, I am today interested in the use of grid technologies to enable that kind of large scale computing needs arising from medical data analysis applications.

2.3 Data grids for medical imaging

The availability of digital imagers inside hospitals and their ever growing inspection capabilities have established digital medical images as a key component of many pathologies diagnosis, follow-up and treatment. Digital medical images represent a tremendous amount of data. In industrialized countries, an hospital produces several Terabytes of medical image data each year, bringing the total production of the European Union or the USA for instance to thousands of Terabytes (Petabytes) per year. This amount of data needs to be properly archived for both medical and legal reasons. Beyond the outstanding issue of proper storage and long term archiving of such an amount of data, automated analysis is increasingly needed as manual inspection of medical images is a complex task, and may become extremely tedious and error prone when dealing with 3D or time sequences of 3D images.

To face the growing image analysis requirements, automated medical image processing algorithms have been developed over the two past decades. Many medical applications today benefit from computerized models and image processing techniques for which computer image analysis is mandatory. Computers have thus been introduced in hospital for diagnosis and pathologies follow-up (modeling and quantitative analysis of images), simulation (*e.g.* planning difficult interventions), and real time treatments (*e.g.* computer assisted surgery). In parallel, medical image databases have been set up in health centers. Some attempts have been made to join data coming from different sources for studies involving large databases.

Grid technologies have been developed for applications with large storage and computation requirements in mind. They offer a promising tool to deal with current challenges in many medical domains involving complex anatomical and physiological modeling of organs from images or large image databases assembling and analysis. However, grid technologies are still in their youth and they often propose only very generic services for deploying large scale applications such as users authorization, authentication and accounting, data replication, fast transfer and transparent access to computing resources. These basic services are needed for any application domain but specialized higher level layers should be developed to take into account the application area specific requirements.

In the following sections, we study the requirements associated to medical image processing on grids and we describe our research activity in this area over the past 5 years. Grids address both the problem of distributed data management and intensive computing. In this chapter, we first address the data management issue which is critical in the medical area : starting from an analysis of the clinical use of medical data (section 2.3.1), we make a data-related requirements study and we show how grids can be used to help in manipulating and processing medical data in section 2.3.2. Based on this analysis, we propose a grid-enabled medical data manager that fulfills most of these application requirements in section 2.3.3. This work was initiated in the MEDIGRID project and was the research topic of Hector Duque during his PhD thesis. It was then continued in the context of the AGIR project in collaboration with the EGEE development team. In the next chapter, we will address the problem of using grid infrastructures to perform intensive data analysis.

2.3.1 Clinical use of medical data versus grid use

Clinical information systems

The medical community is routinely using clinical images and associated medical data for diagnosis, intervention planning and therapy follow-up. Medical imagers are producing an increasing number of digital images for which computerized archiving, processing and analysis are needed [74, 133]. Indeed, image networks have become a critical component of the daily clinical

practice over the years. With their emergence, the need for standardized medical data formats and exchange procedures has grown [7]. For this reason, the *Digital Image and COmmunication in Medicine* standard (DICOM)[5] was adopted by a large consortium of medical device vendors. *Picture Archiving and Communication Systems* (PACS) [98], manipulating DICOM images, and often other medical data in proprietary formats, are proposed by medical device vendors for managing clinical data. PACS are often proprietary solutions weakly standardized. PACS may be more or less connected to the *Hospital Information System* (HIS), holding administrative information about patients, and the *Radiological Information Systems* (RIS), holding additional information for the radiological departments. The DICOM standard, PACS, RIS and HIS have been developed with clinical needs in mind. They are easing the daily care of the patients and medical administrative procedures. However, their usage in other areas is very limited. The interface with computing infrastructures for instance is almost completely lacking. In addition, current PACS hardly address medical data management needs beyond clinical centers' administrative boundaries, while the patient medical folders are often wide spread over many medical sites that have been involved in the patient's healthcare. Many medical image acquisition devices are also weakly conforming to the DICOM standard, thus hardly hiding the heterogeneity of these systems.

DICOM format, protocol, storage, and security

The DICOM standard encompasses, among other things, an image format and an image communication protocol. A DICOM image usually contains one slice (a 2D image) acquired using any medical imaging modality (MRI, CT-scan, PET, SPECT, ultrasound, X-ray... [1]). A DICOM image may contain a multi-slices data set but this is rarely encountered. A DICOM image contains both the image data itself and a set of additional information (or *metadata*) related to the image, the patient, the acquisition parameters and the radiology department. DICOM metadata is stored in fields. Each field is identified by a unique tag defined in the DICOM standard. A given field may be present or absent depending on the imager that produced the image. The standard is open and image device manufacturers tend to use their own fields for various kinds of information. A couple of fields (such as image size) are mandatory but experience proved that surprises should be expected when analyzing a DICOM image. The image itself is usually stored as raw data. Most imaging devices produce one intensity value per image pixel, coded in a 12 bit format. Other format may be encountered such as 16 bit data or loss-less JPEG.

Most (reasonably modern) medical image acquisition devices are DICOM clients. DICOM servers are computers with on-disk and/or tape back-ends able to store and retrieve DICOM images. The DICOM protocol defines the communication protocol between DICOM servers and clients.

There is no standardization on DICOM storage. DICOM servers are implementing their own policy of data storage. One should not see DICOM data sets as a set of files. As stated above, a single DICOM image usually contains only one image slice. In practice, during a medical examination (a DICOM *study*), a radiologist acquires several 2D and 3D images, representing up to hundreds to thousands of slices. A study is divided in one or several *series* and series are composed by sets of slices (that can be stacked to assemble a volume when they belong to the same 3D image). Note that there is often no notion of 3D image encoded in the DICOM format : series may contain a set of slices composing several 3D images. The way a DICOM server stores these data sets on disk is irrelevant just like the way a database stores its table is usually not known from the users : the medical user is never exposed to the DICOM storage and does not

[5] DICOM, http://medical.nema.org/

need to know if different files are used for each DICOM slice, series, study, etc. Metadata are included in DICOM image headers, making them difficult to manipulate. A DICOM server will often extract this metadata and store it in a database to ease data search.

Each image acquisition device is a potential DICOM compliant medical image source. In a radiological department, one or several DICOM servers can be set up to centralize data acquired on this site. Medical data are naturally distributed over the different acquisition sites.

The DICOM security model is rather weak. DICOM files are unencrypted and transported unencrypted. Files contain patient data. The DICOM server security model is based on a per-application basis : all users having access to some DICOM client application can access to the information that the server returns to this specific application. DICOM servers are using random file names without any connection to the patient information and a proprietary data storage policy. To cope with these data protection limitations, security is often implemented in hospitals by isolating the image network from the outside world.

Clinical use of medical data

Medical data is very sensitive as it often carries information about individuals. Another particularity of medical data is its rich semantic content. Medical images represent tons of data to be manipulated. However, images by themselves are not sufficient for most medical applications. A physician is not analyzing images alone but he needs to interpret an image or a set of images in a medical context. The image content is only relevant when considering the patient age and sex, the medical record for this patient, sociological and environmental considerations, etc. In clinical practice, physicians do not access directly to image files. They identify data by associated metadata. The data is transferred mainly for visualization purposes. The physician quickly scans the slices stack in the DICOM study and focuses on the slices he or she is most interested in.

Medical image analysis

In the medical image analysis community, the needs are quite different. In the last decades, with the growing availability of digital medical data, many medical data processing and analysis algorithms were developed, enabling computerized medical applications for the benefit of the patient and healthcare practitioners. Although sharing the same data sources, the medical image analysis community has different requirements for medical system than the healthcare community. Many algorithms are developed for processing and producing image files. A common procedure for accessing all medical data sources is needed.

As in the clinical case, many image analysis application are also concerned with the data semantics and require rich metadata content. For instance, epidemiology requires the study of large data sets and the search of similarities between medical cases. Relevant images are also often identified through the use of metadata, although the search are not necessarily for nominative data but often related to the acquisition type or body region. Physicians are often interested in looking for medical cases similar to the one they are studying. A case may be identified as “similar” because the images content are similar but this is often not sufficient to discriminate between a set of acquisitions. There is a need to take into account metadata on the medical case and results of computations done on the images in the similarity criterion. Therefore, medical metadata carrying additional information on the images are mandatory.

For image analysis, 3D images are exported to disk files for ease of use. Various 3D medical image format may be used to stack different DICOM slices into a single image volume (the most common being the *analyze* file format).

Grid added value for medical applications

Grids are providing computing resources and workload systems that ease application code deployment and usage. Moreover, grids are providing distributed data management services that are well suited for handling medical data geographically spread throughout various medical centers [23, 58, 18, 92, 12]. Grid technologies offer a unique environment for sharing data and computing or storage resources. But there is more than that into grids that make them a potential tool to outperform the actual limitations of medical applications. Indeed, grids are a vector for :

- allowing distribution of large data sets over different sites and avoiding single points of failure or bottlenecks ;
- enforcing the use of common standards for data exchanges and making exchanges between sites easier ;
- enlarging the data sets available for large scale studies by breaking the barriers between remote sites ;
- allowing a distributed community to share its computational resources so that a small laboratory can proceed with large scale experiments if needed ;
- opening new application fields that were not even thinkable without a common grid infrastructure (*e.g.* large scale epidemiology or studies on rare diseases).

Grids are likely to have a deep impact on health related applications by playing a key federative role [22]. Given that application specific facilities are provided, they will help in federating research efforts over distant institutes by allowing to assemble very large data sets needed in statistical and epidemiological studies, by allowing to share computing resources between small laboratories that could not offer to maintain costly computing centers otherwise and by easing results comparisons and common procedures adoption. Grids provide a logical continuation to regional health networks [98] by allowing distant sites to collaborate and exchange their data for specific research purposes.

However, existing grid middlewares are often only dealing with data files and do not provide higher level services for manipulating medical data. Medical data often have to be manually transferred and transformed from hospital sources to grid storage before being processed and analyzed. Such manual interventions are tedious and often limit systematic use of grid infrastructures. In some cases, they may even prevent the use of grids, *e.g.* when the amount of data to transfer is too large. As a consequence, the first key to the success of the systematic deployment of medical image processing algorithms is to provide a data manager that :

- provides access to medical data sources for computing without interfering with the clinical practice ;
- ensures transparency so that accessing medical data does not require any specific user intervention ;
- ensures a high data protection level to respect patients privacy.

2.3.2 Manipulating medical data on grids

Medical data security

The primary concern when distributing medical data over a grid is privacy. Medical applications often deal with patient data that are private and should only be accessible to the patient himself, the medical team involved in his health care, and, under some restrictions, for research purposes. Therefore, a medical grid, opened to a wide community of users, should enforce strict access control. Privacy of medical data stored over a grid is particularly sensitive due to :

- the spread of data over many remote sites locally administrated ;
- the replication mechanism triggering copies of data on any grid site without notification.

There are two confidentiality levels when considering medical data : nominative data is the most critical and should only be accessible to the patient and a limited accredited medical team, while the rest of the data (such as image content) is restricted for authorized persons only although it does not allow the identification of a person alone. It is very important in a grid environment to ensure that no relation can be established between a person and his or her data except for a few accredited users. Also note that it may be difficult to determine whether some data is private (whether it carries nominative information or not). Indeed, a medical image is usually not sufficient to recognize the patient it was acquired from but in some cases (*e.g.* high resolution image of the head) it might provide information that can indirectly be used to identify the patient.

We can consider four groups of users involved in medical data manipulation :

- the patient from whom the data originates ;
- the physicians or other specialists involved in health care ;
- researchers needing an access to this data ;
- and any other grid user.

In general, no private or personal data should ever be accessible to any grid user. This group includes system administrators of grid clusters who are not accredited to manipulate medical data. This makes security enforcement complex as they have full access to resources under their responsibility. Therefore, medical data should be encrypted when transferred onto the grid, and encryption key should only be stored on secured (trusted) sites with strict controls on the persons allowed to retrieve them. Patients should always have full read access to their data. Physicians involved in the health care of a given patient need read/write access. However, any physician should not be able to access any patient data. Finally, researchers should be able to access personal data (not private ones) for research purposes if a physician with access to the data grants them this authorization.

Enabling data security

Security is always a trade off between inconvenience for the users and the desired level of protection. In order to convince users to use grids for their data storage and processing needs, the following needs have to be addressed.

Reliable authentication. Authentication is not a grid-specific problem. It is well researched and standard solutions exist. The use of a public key infrastructure (PKI [65]) with certification authorities (CA) and X509 certificates is a reasonable way to handle authentication in grid environments.

Secure transfer of data. Secure transfer is also a well researched area independently of grid technologies. It is addressed in various standardized protocols such as *SSL/TLS*, *IpSec*, *SSH* or GridFTP [3].

Secure storage of data. For storage, encryption and signing is an obvious solution. The problem in grid environments is that mechanisms are required to share decryption keys between users authorized to access data. Common encrypted storage systems lack the flexibility to deal with the dynamic nature of grid access permissions. In the context of the MEDIGRID project, we have proposed an architecture with a generic interface to grid access control mechanisms, that provides access to decryption keys based on access permissions [183, 181].

Data access control. Access control raises the most problematic issues for medical data processing in grid environments. Classic access control techniques are not designed to deal with the problems arising from the decentralized, cross-organizational nature of grid access permissions. The medical field of applications adds another inherent problem. Classical grid access control mechanisms such as CAS [159] are satisfactory for dealing with simplest cases. Nevertheless

these systems fail to provide sufficient permission granularity and flexibility for ad hoc permission granting that is required in medical applications. Furthermore such systems use centralized permission databases which represents a single point of failure.

An alternative approach is to manage grid access control using decentralized permission checking through attribute certificates. Such certificates permit resource administrators to deliver permissions in a simple way without having to resort to third party services. Local servers can easily verify the permissions granted in such certificates, using a local database that specifies the *sources of authority* (SOA) of the resources on their systems. The attribute certificates enable the local servers to trace a permission from SOA of the concerned resource to the user requesting it. The database that specifies the SOAs is managed by the access control system itself and is updated, when new resources (e.g. files) are added to the system.

The EGEE security model is based on Virtual Organizations (VO [66]). Resource providers assign permissions to those VOs, and the VOs have policies to dispatch the resources they have been assigned between their members [2]. VOMS is an extension of VO-based security providing role based access control (RBAC) [59]. Using RBAC, administrators can manage user groups (VOs) that are assigned sets of permissions and the membership of users within those groups. In the context of the MEDIGRID project, an RBAC control system that also provides a generic program execution interface was proposed. It permits users to run their own specific programs in a sandbox environment prior to giving access to a resource [184, 181].

Anonymization of medical records. Anonymization is required to provide large sets of data for medical research. Legislation imposes severe regulations as to what can be considered an anonymized information [35]. The main problem is that even if obvious sections such as name and address of the patient have been removed a medical document could be re-identified with secondary information.

Tamper-proof logging of operations performed on medical files. Traceability is clearly another important factor in medical grids. It should always be possible to know, for a given image where it originates from (which algorithm and which input image(s) were used to produce it). Indeed, physicians often need to come back to the unaltered data when studying a processed image. Conversely, for each input data it is of interest for optimizing computations to record which output has already been processed using various algorithms (computation results cache). Extending this idea further, the grid middleware could use this facility for storing either an output data or the description of how it can be obtained from some input data (and make space optimization based on the relative cost of results recomputation compared to the needed storage space). This is achievable not only with a data management system that stores metadata describing computations done, but also with an algorithm management service that make algorithms known to the grid middleware and reprocessing possible by picking algorithms out of a database where they have been registered.

Note that we have not included *secure processing of data* in this discussion. Performing computations on encrypted data without the without explicit decryption is a burning research area today. These techniques can accommodate to simple arithmetic operations but they are not mature enough to handle the complexity of image processing, not to mention the efficiency problems. To remain realistic, the features that should protect data while it is being processed on a grid are based on best effort technologies, *i.e.* on-disk encryption, access control, and anonymization. Users need to trust the servers on which their data is to be processed, to our knowledge no systems for data processing on untrusted resources exist.

Medical metadata and data semantics

Another particularity of medical data is its strong semantic content. A medical image itself is often of low interest if it is not related to a context (patient medical files, other similar cases...).

Therefore, grids should provide tools to organize, relate, and describe medical image contents. The medical information system should not only deal with data but also their semantics by providing standardized ways of describing their content. Tools to manipulate metadata attached to the data are a first step in this direction.

Therefore, the grid should provide a basic support for storing metadata (usually in relational databases) on different sites, making distributed queries over this data and triggering replicates when needed for efficiency or robustness reasons. Beyond the storage and retrieval of metadata, facilities are expected to make use of data easier from an application point of view. The middleware should allow description of data sets from queries on metadata and the definition of job inputs and/or outputs through metadata. A complete synchronization between the metadata management system and the data management system are needed to ensure coherence.

Beyond simple patient-related metadata (age, sex, etc), the metadata should also include :

- image-related metadata : image dimensions, voxels size, encoding, etc.
- acquisition-related metadata : acquisition device used, parameters set for the acquisition, acquisition date, etc.
- hospital-related metadata : radiology department responsible for this acquisition, radiologist, etc.
- medical record : history, miscellaneous information explaining how to interpret this image, etc.

In addition to these medical metadata, external information is needed for computation-related maters : to hold access control information, to ensure traceability or to improve performances by caching queries and computation results. Computation on large images are costly and data retrieval in large image databases may represent intractable computations if image analysis is needed and images have not been properly indexed. The information related metadata therefore include :

- security-related metadata : authorization, encryption keys, data access logging, etc.
- history-related metadata : image sources, algorithms, parameters, etc.
- optimization-related metadata : image index, query caching, processing caching, etc.

As can be seen, a part of the metadata is directly attached to the image, while the rest is related to the hospital or the patient. The metadata structure should therefore reflect these relations between images and metadata. Some metadata is static : it is either administrative information external to the image (e.g. patient metadata) or bound to the image (e.g. image metadata) with the same access pattern, same lifetime, etc. Other metadata is dynamically generated during computations.

Metadata is often very sensitive, even more than the image content itself : it contains all necessary information to identify patients and the security elements such as access control information and encryption keys. Most metadata can therefore only be stored on trusted and secured sites where administrators are accredited to manage such personal data. Precise metadata access policies must be enforced.

An important feature of a medical information system is its ability to retrieve relevant data for a given application. Data may be selected on the image content (by processing) or by taking into account its semantics (the metadata). Often both are needed at the same time.

2.3.3 A grid medical data manager

In the context of the MEDIGRID project and in collaboration with the EGEE European project, we have been working on a medical data manager fulfilling most of the medical applications requirements analyzed in section 2.3.2. This work was initiated by a first study and the development of a prototype [57, 132, 56]. It later led to a second design and the implementation of a service tightly coupled with the EGEE middleware and exposing a standard grid

interface [152].

EGEE grid middleware

The EGEE project is currently deploying the LCG2 middleware[6] on its production infrastructure. LCG2 is based on GLOBUS2, Condor, and the other services developed during the European DataGrid project[7]. A new generation middleware, gLite[8], is under testing and should be deployed during Summer 2006. Our Medical Data Manager service (MDM) is based on gLite.

The gLite middleware provides workload management services for submitting computing tasks to the grid infrastructure and data management services for managing distributed files. The data management is based on a set of *Storage Elements* which are storage resources distributed in the various sites participating in the infrastructure (currently, more than 180 sites distributed all over Europe and beyond). All storage elements expose a same interface for interacting with the other middleware services : the *Storage Resource Manager* interface (SRM) that is standardized in the context of the Global Grid Forum[9]. The SRM is handling local data at a file level. It offers an interface to create, fetch, pin, or destroy files among other things. It does not implement data transfer by itself. Additional services such as GridFTP or gLiteIO are coexisting on storage elements to provide transfer capabilities.

In addition to storage resources, the gLite data management system includes a *File Catalog* (FiReMAN - *File Replica MANager*) offering a unique entry point for files distributed on all grid storage elements. Each file is uniquely identified through a *Global Unique IDentifier* (GUID). The file catalog contains tables associating each GUID to file location. For efficiency and fault tolerance reasons, files may be replicated on different sites. Thus, each GUID may be associated to several locations. To ease the manipulation by users, human readable *Logical File Names* (LFN) can be associated to each file (each GUID).

Medical Data Management service design

The Medical Data Management service architecture is diagrammed in figure 2.23. On the left, is represented a clinical site : various imagers in an hospital are *pushing* the images produced on a DICOM server. Inside the hospital, clinicians can access the DICOM server content through DICOM clients. In the center of figure 2.23, the MDM internal logic is represented. On the right side, the grid services interfacing with the MDM are shown.

All middleware services requiring access to data storage do so through SRM requests sent to storage elements. To remain compatible with the rest of the grid infrastructure, our MDM service is based on a SRM-DICOM interface software. The SRM-DICOM core is receiving SRM requests and transforms them into DICOM transactions addressed to the medical servers. Thus, medical data servers can be shared between clinicians (using the classical DICOM interface inside hospitals) and image analysis scientists (using the SRM-DICOM interface to access the same data bases) without interfering with the clinical practice. An internal scratch space is used to transform DICOM data into files that are accessible through data transfer services (GridFTP or gLiteIO).

A metadata manager is also used to extract DICOM headers information and ease data search. The AMGA[10] service [176] is used for ensuring secured storage of this very sensitive data. The AMGA server holds a relation between each DICOM slice and the image's metadata.

[6] LCG2 : Large hadron collider Computing Grid middleware, http://lcg-web.cern.ch
[7] European DataGrid project, http://www.edg.org
[8] gLite middleware, http://www.glite.org
[9] Global Grid Forum, http://www.ggf.org
[10] ARDA metadata catalog project, http://project-arda-dev.web.cern.ch/project-arda-dev/metadata/

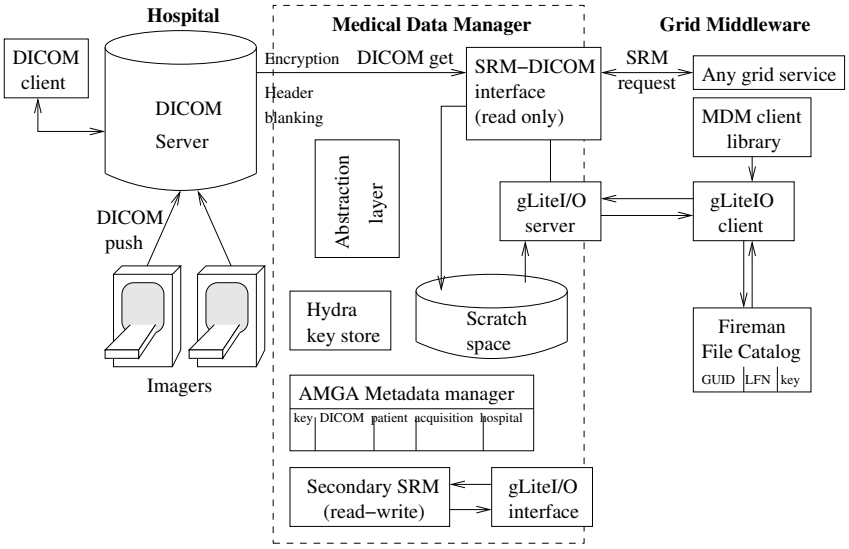


FIG. 2.23 – Overview of the medical data manager

This specialized SRM is not providing a classical Read/Write interface to a storage element. A classical R/W storage element can symmetrically receive grid files to be stored or deliver archived files to the grid on request. In the MDM, The SRM interface only accepts registration request coming internally from the hospital. To avoid interfering with the clinical data, external grid files are not permitted to be registered on the MDM storage space : only get requests are authorized from the grid side. If classical grid storage is desired (with write capability), a classical secondary SRM can be installed on the same host.

For data encryption needs, a secured encryption key catalog is also used. It is named *hydra catalog* as it uses a split key storage strategy to improve security and fault tolerance [185, 183].

An *abstraction layer* is also depicted on the diagram. Its role is to offer a higher level abstraction for accessing 3D images by associating all DICOM slices corresponding to a single volume. Indeed, most medical image processing applications are not manipulating 2D images independently but rather consider complete volumes. The abstraction layer is associating a single GUID to each volume. On a request for the volume associated to this GUID, all corresponding slices are transferred from the DICOM server and assembled in a single volume in scratch space.

Internal service interaction patterns

To fulfill its role, the MDM service needs to be notified when files are produced by the imagers and stored into the DICOM server. This notification triggers a file registration procedure that is depicted in figure 2.24. The DICOM data triggering the operation is first stored into the hospital DICOM server as usual. The DICOM header is then analyzed to extract image identifying information. This DICOM ID is used to build a *Storage URL* (SURL) as used by the grid File Catalog to locate files. The SURL is registered into the File Catalog and a GUID associated to this data on the grid side. The rest of the metadata extracted from the DICOM header is stored into the AMGA metadata server. Finally, encryption keys that are associated to the file and that will be used for data retrieval are stored into the hydra distributed database.

Once DICOM data sets have been registered into the MDM, the server is able to deliver requested data to the grid as depicted in figure 2.25. A client library is used for this purpose. To cover all application use cases, the MDM client library provides APIs for requesting files based on their grid identifier (GUID) or the metadata attached to the file. In case of request

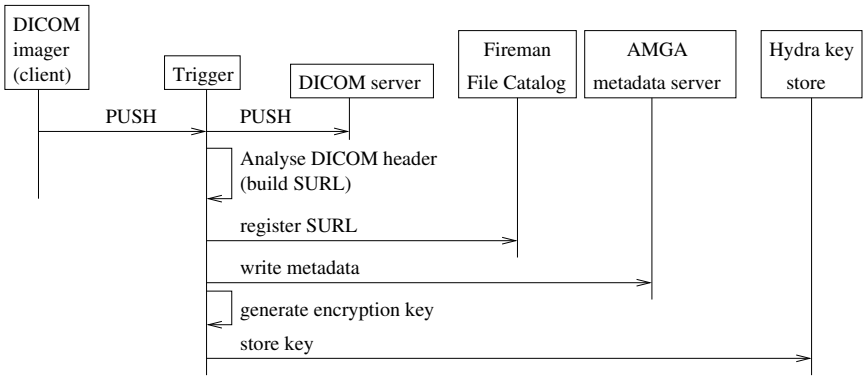


FIG. 2.24 – Triggered action at image creation

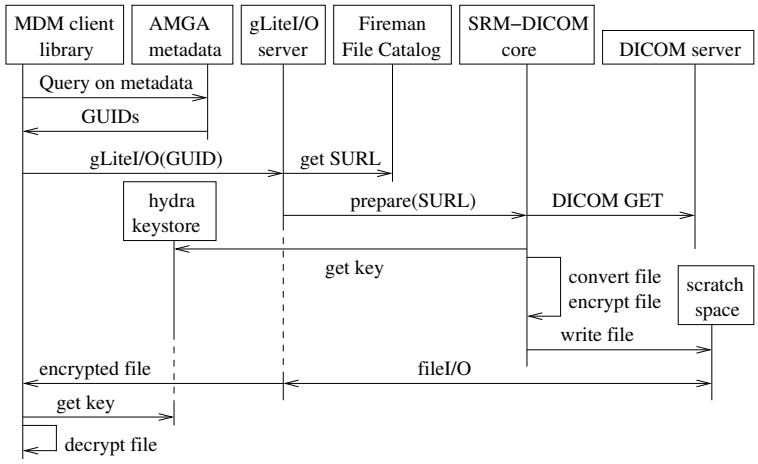


FIG. 2.25 – Accessing DICOM images

on the metadata, a database query is first made to the AMGA server and the list of GUIDs of images matching the query is returned. The SRM-DICOM server can then deliver images requested through their GUID. SRM get requests are translated into DICOM get queries. The data extracted from the DICOM server is first written to an internal scratch space. Its format is transformed into a simple 3D image file format (a human readable header including image size and encoding, followed by the raw image data). In this transformation, the DICOM header, containing patient identifying operations, is removed to preserve anonymity. The files are also encrypted before being sent out to ensure that no sensitive information is never transferred nor stored on the grid in a readable format. Files are then transferred through the gLiteIO service and returned to the client in an encrypted form. The file is only decrypted in memory of the client host, given that the client is authorized to access the file encryption keys.

MDM client

On the client side, three levels of interfaces are available to access and manipulate the data hold by the MDM. The MDM is seen from the middleware as any storage resource exposing a standard SRM interface, the standard data management client interface can be used to access images provided that their GUID is known. The files retrieved using this standard interface are encrypted. The second interface is an extra middleware layer which encompasses access to

the encryption key and the SRM. Thus images can be fetched and decrypted locally. The third and last level of interface is the fully MDM aware client library represented in figure 2.25. It provides access to encrypted files and in-memory decryption of the data on the application side, plus access to the metadata through the AMGA client interface.

Data security model

The security model of the MDM relies on several services :

- file access control ;
- files anonymization ;
- files encryption ; and
- secured access to metadata.

The user is coherently identified through a single X509 certificate and all services involved in security are using the same identification procedure. The file access control is enforced by the gLiteIO service which accepts Access Control Lists (ACLs) for fine grained control. The hydra key store and the AMGA metadata services also accept ACLs. To read an image content, a user needs to be authorized both to access the file and the encryption key. The access rights to the sensitive metadata associated to the files are administrated independently. Thus, it is possible to grant access to an encrypted file only (*e.g.* for replicating a file without accessing to the content), to the file content (*e.g.* for processing the data without revealing the patient identity), or to the full file metadata (*e.g.* for medical usage).

Through ACLs, it is possible to implement complex use cases, granting access rights (for listing, reading, or writing) to patients, physicians, healthcare practitioners, or researchers needing to process medical data, independently from each other.

Medical metadata schema

A minimal metadata schema is defined in the MDM service for all images stored. It provides basic information on the patient owning the image, the image properties, acquisition parameters, etc. There are two main indexes used : a patient ID, for all nominative information associated to patients and the image GUID for all information associated to images. The patient ID is a unique but irreversible field (such as a MD5 sum on the patient field name). Four main relational tables are used :

- The Patient table, indexed on the patient ID, contains the most sensitive identifying data (patient name, sex, date of birth, etc).
- The Image table, indexed on the image GUID, contains technical information about the image (size, encoding, etc). It establishes a relation with the patient ID.
- The Medical table, indexed on the image GUID, contains additional information on the acquisition (image modality, acquisition place and date, radiologists, etc).
- The DICOM table, indexed on the image GUID, contains the image DICOM identifiers used for querying the DICOM server.

To remain extensible, an additional Protocol table associates image GUIDs with medical protocol names. Through AMGA, the user can create as many medical protocols as needed, containing specific information related to some particular acquisition (*e.g.* a temporal protocol for cardiac acquisitions, etc). AMGA also enables per table access right control, allowing restricting access to the most sensitive data (*e.g.* the Patient table) to the minimum number of users.

Testbed and future work

The Medical Data Manager has been deployed on several sites for testing purposes. Three sites are actually holding data in three DICOM servers installed at I3S (Sophia Antipolis, France), LAL (Orsay, France) and CREATIS (Lyon, France). In addition to the DICOM servers, these sites have installed the core MDM services : a SRM-DICOM server and associated database back-end, a gLiteIO service, a GridFTP service, and all dependencies in the gLite middleware. Client have been deployed on all these three sites.

To complete the installation, an AMGA catalog has also been set up in CREATIS (Lyon) for holding all sites' metadata, and an hydra key store is deployed at CERN (Geneva, Switzerland) for keeping file encryption keys.

Given the number of services involved, the installation and configuration procedure is currently complex. It is being worked out to ease the testbed extension. The MDM service should be deployed in hospitals where little support is provided for the informatics infrastructure.

The testbed deployed has been used to demonstrate the viability of the service by registering and retrieving DICOM files across sites. For testing purposes, DICOM data registrations are triggered by hand. Registered files could be retrieved and used from EGEE grid nodes transparently, using the standard EGEE data management interface. The next important milestone will be to experiment the system in connection with hospitals by registering real clinical data freshly acquired and registered on the fly from the hospital imagers. This step involves entering a more complex clinical protocol with strong guarantee on the data privacy protection. The security cannot be neglected at any level at this point.

2.4 Compute intensive medical data analysis procedures

As a consequence of the tremendous research effort carried out by the international community these last years and the emergence of standards, grid middlewares have reached a maturity level such that large grid infrastructures where deployed (EGEE[11], OSG[12], NAREGI[13]) and sustained computing production was demonstrated. Yet, current middlewares expose rather low level interfaces to the application developers and enacting an application on a grid often requires a significant work involving computer and grid system experts.

Medical applications usually require more than a grid-wide scheduler and a batch job submission system. The main computing requirements specific to the medical area and the subsequent expectations for a medical grid are analyzed in section 2.4.1. Our works tackling these requirements are presented in the rest of this chapter. In this document, we mainly address three specific points :

- The remote execution of interactive applications with a need for a visual feedback (section 2.4.2). This work was performed with the help of Eduardo Dalila during is Master thesis at CREATIS.
- The partitioning of data to distribute computing load over the grid infrastructure illustrated through data indexing applications (section 2.4.3). This work was initiated in the context of the DataGrid project and later studied under a totally new angle thanks to the innovative approach of Tristan Glatard during is first year PhD thesis at I3S.
- The efficient processing of data-intensive workflows to manage medical applications requiring the manipulation of complete databases such as algorithms validation (sections 2.4.6 to 2.4.12). This work is the PhD research topic of Tristan Glatard. It owes a lot to his relevant contributions.

2.4.1 Requirements related to intensive medical data analysis

Considering the considerable amount of sequential, non grid-specific algorithms that have been produced for various data processing tasks, grid computing is very promising for :

- Performing complex computations involving many computation tasks (codes parallelism).
- Processing large amounts of data (data parallelism).

Indeed, beyond specific parallel codes conceived for exploiting an internal parallelism, grids are adapted to the massive execution of different tasks or the re-execution of a sequential code on different data sets which are needed for many applications. In both cases, temporal and data dependencies may limit the parallelism that can be achieved. In the case of medical data, analysis procedures often do not involve a single algorithm and a single data fragment but rather a chain of processings and full data sets that can sometimes be executed concurrently. A support for the efficient execution of such workflows, is expected from the grid middleware.

Medical applications reliability is also sometimes critical and an expert needs to assess the computation results or even to guide medical data analysis algorithms during their execution. Therefore, the grid should allow user interaction with processes submitted to distributed resources, which implies that computations should be fast enough to allow a fast interaction with a physician. This is made difficult by the remote execution of the computation whereas the application feed-back (visualization and user input) should be returned to the end user desktop machine. The most demanding cases are simulation applications for which real time constraints have to be enforced. In addition, a medical grid should be able to take into account emergency

[11] Enabling Grids for E-sciencE, http://www.eu-egee.org
[12] Open Science Grid, http://www.opensciencegrid.org
[13] NAREGI Japan National Research Grid Initiative, http://www.naregi.org

situations by allowing some users (surgeons, emergency services) to send high priority jobs, preempting running jobs when needed.

Workflows

Grids are well suited for handling transparently medical image analysis procedures described as computing workflows. Workflows are compound jobs composed of several elementary stages (each stage representing an algorithm applied onto an input data set and producing an output data set). Several stages can be processed on different machines. Stages are chained (*e.g.* the output of stage A is used as input for stage B) but are not necessarily linear (*e.g.* both stages B and C can be processed in parallel). Therefore, a grid workflow manager should offer a workflow description mechanism to design the architecture of the workflow and a smart scheduler able to exploit the workflow intrinsic parallelism by distributing processings over various grid nodes (data-flow control, load balancing, synchronization...). Workflows are of real interest when processing a large number of input data rather than a single input. Through workflows, the user can describe once for all the chain of transformations that each element of the input data set should undergo. The workflow manager can process several elements in parallel on grid nodes (thousands of concurrent input images are expected for some medical applications). Synchronization barriers may be needed to extract statistics from several processed data at some point in the process flow. Therefore, workflow managers should provide additional services such as synchronization, logs of accomplished stages for a given input, restart from a failing job, automatic resubmission of stages that failed for user-independent reasons, etc.

Our work on enabling medical imaging workflows on a grid infrastructure is described in section 2.4.6. Additional results related to optimal scheduling of data parallel problems are described in section 2.4.3.

Parallel computations

Some image processing, simulation, and modeling algorithms are very computation intensive and need a parallel implementation in order to get executed in a reasonable amount of time compatible with clinical practice constraints. Efficiency is even more critical for applications that require interactivity. In this case, the user can only remain a reasonably short amount of time in front of his computer screen, waiting for the algorithm to process data and return an output. Support for parallel computations is mandatory for these applications. Local area parallelism is widely available today through message passing interfaces. However, grid-wide parallelism involving heterogeneous machines and networks is an area that still requires investigation.

Interactive applications

Interaction with the user may be needed for guiding the algorithm, to solve legal issues when dealing with medical data, or due to the application itself (*e.g.* therapy simulator). Data compression and high-bandwidth networks should insure a limited response time which is mandatory for interactive usage. Interactive feedback often involves 3D visualization of medical scenes. This can become challenging due to the large size of 3D medical images and the complexity of meshes used for realistic 3D modeling [136, 137]. To achieve user interaction with grid jobs, a communication should be possible between the computing node(s) and the user workstation. Firewalls are often causing communication constraints and computing clusters may be isolated from the external network.

We address the problem of interactive applications in section 2.4.2.

Accessibility to grid resources

The medical community is very large and in general not very much aware of computers and grid technologies. Therefore, a tremendous effort is needed to interface existing grid middlewares and make them usable by this community. High level services such as data search engines, easy access to job submission, graphic workflow design tools, algorithms selection and execution from input data or metadata sets, are needed for grid-aware medical applications development. Comprehensive and easy-to-use interfaces built on top of these services are mandatory for grid tools to become accepted by the medical community.

Another key point in the deployment of grid technologies is the trust the end users can have in the underlying architecture. Medical data being confidential in general, the deployment of many medical applications will only be feasible if the middleware is recognized as secured and controlled. This prevents the use of middlewares with a lazy security infrastructure or not enforcing strict checking on data and computation resources usage.

2.4.2 Dealing with remote interactive applications

Context

Due to the sensitive nature of medical data, medical image analysis algorithms often need human supervision to :

- solve responsibility issues when taking decisions based on data analysis results ;
- allow a user intervention when the automated processing is inaccurate or erroneous.

Therefore, many interactive medical applications have been developed offering maximum automated assistance to free the user from tedious and error prone tasks while allowing a specialist to control the algorithm output. For instance, image segmentation and computer assisted diagnosis tools usually require user supervision. Other applications such as medical intervention simulation, augmented reality, and telemedicine also require user interaction and large computing power by nature.

Interactive applications executions on remote systems such as grid are made difficult by the need to provide a user interface decoupled from the computing application. This has motivated recent research work such as the Interactive European Grid project[14]. In this section we discuss a software architecture designed to execute remote complex medical application requiring visualization of 3D scenes and interaction with the user. The main objectives of this work are to :

- Provide a multipurpose, flexible, and transparent solution easing the development of various remote interactive applications without requiring the user to know about the internals of multiprocesses programing and network transmission.
- Ensure efficient transmission of graphic data to allow the rendering of complex 3D medical scenes efficiently.
- Allow remote execution of compute intensive applications on remote workstations or grid infrastructures dedicated to high performance computing.

In this regard, X client/server protocol is not suitable. Indeed, X transmission of graphic bitmaps is network intensive (compared to the transmission of minimal geometric information needed to describe medical models) and is not adapted for a high degree of interactivity. Moreover, 3D graphic visualization is today usually achieved locally using specialized rendering hardware available on most recent computers while remote computing facilities might not provide any specific graphics hardware.

[14]Interactive European Grid project, http://www.interactive-grid.eu

Interactive medical applications

Interactive applications follow the simple loop depicted in figure 2.26. The application is made of an iterative algorithm that progresses by small steps. At each step, the application collects some input from the user (through the mouse or another specialized device), and takes into account this input to process the following iteration. After one iteration, the algorithm state is updated and some feedback (usually through visualization) is returned to the user to inform her of the current application progress. Collecting user input and sending feedback might not necessarily happen at every algorithm iteration, depending on the application.

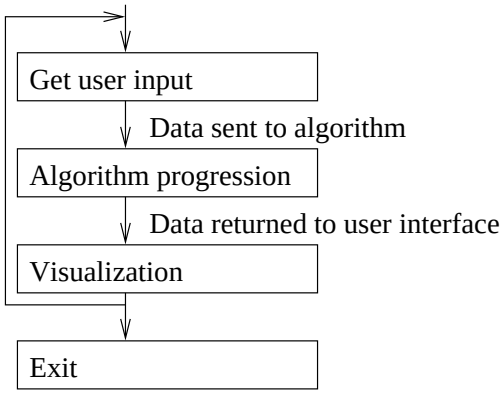


FIG. 2.26 – Generic structure of an interactive application.

In particular, medical applications often manipulate 3D data. Most modern medical image acquisition devices (Magnetic Resonance Imaging, Computed Tomography scanners, Nuclear Medicine scanners, Ultrasound devices, etc [1]) are capable of producing 3D images with very detailed information on the imaged body. To analyze such images, 3D geometrical and anatomical models have been developed [126, 146, 70].

Visualization of 3D medical images in not straightforward since a user cannot at a glance visualize the interior of a 3D volume. Different visualization techniques exist that transform the original volume in a representable data. This might be a single slice extracted out of the volume or a synthesized representation as may be obtained by surface or volume rendering techniques for example [11]. Medical applications usually display additional geometrical representations of some internal modeling of the data. This information is often much more compact than the complete 3D data set. Therefore, rather few information needs to be exchanged between the computing machine and the graphic interface.

In the literature, many works dealing with remote data visualization are reported. Many work concern post-processed data visualization [111], while other consider interaction with the data [11]. Some of them offer asynchronous objects management and real-time constraints [177, 108]. However, few of them address the specific needs of medical image processing applications [49, 136, 137], and optimized computational grids for solving medical problems is still an emerging field.

Remote 3D visualization and interaction framework

Porting interactive applications to a remote platform. Our implementation is based on C++ libraries developed in our laboratory, although the proposed framework does not depend on this specific language. Our libraries provide basic objects to describe 2D and 3D medical scenes made of a set of graphical objects and visualization facilities to display all these graphic

objects in an application window. Medical scene examples are shown in figures 2.28 and 2.29. The user can interact with the 3D scene by different means through the mouse. She can (i) change the display parameters, (ii) select an active graphic object that she can (iii) move (translate, rotate, and rescale) or on which she can (iv) apply application-dependent actions. In addition, an object-dependent menu is proposed for object specific manipulations. We will refer to a *stand-alone* application to mention a classic application designed to run on the local graphic system using these libraries by opposition to a *remote* application that is executed on a remote server. Our goal is to make as few modifications as possible in the user code to move from a stand-alone to a remote application.

In order to execute interactively, a remote application needs to create an interface window on the user local machine. Figure 2.27 illustrates the remote execution procedure. A graphic daemon is running on the user local machine and waits for incoming connection requests. From its command line interface, the user can connect to a remote machine or use some grid middleware to start the remote application on the desired target (1). At execution time, the remote application needs to know (through a command line parameter) the IP address of the local machine on which the interface should be displayed. It connects on a predefined port of the local machine (2), activates a local process that will deal with the user interface (3). It can then start the main computation loop (4), identical to figure 2.26 except that all user input is transported from the user interface to the remote process and all visualization data are sent the opposite direction through a communication channel.

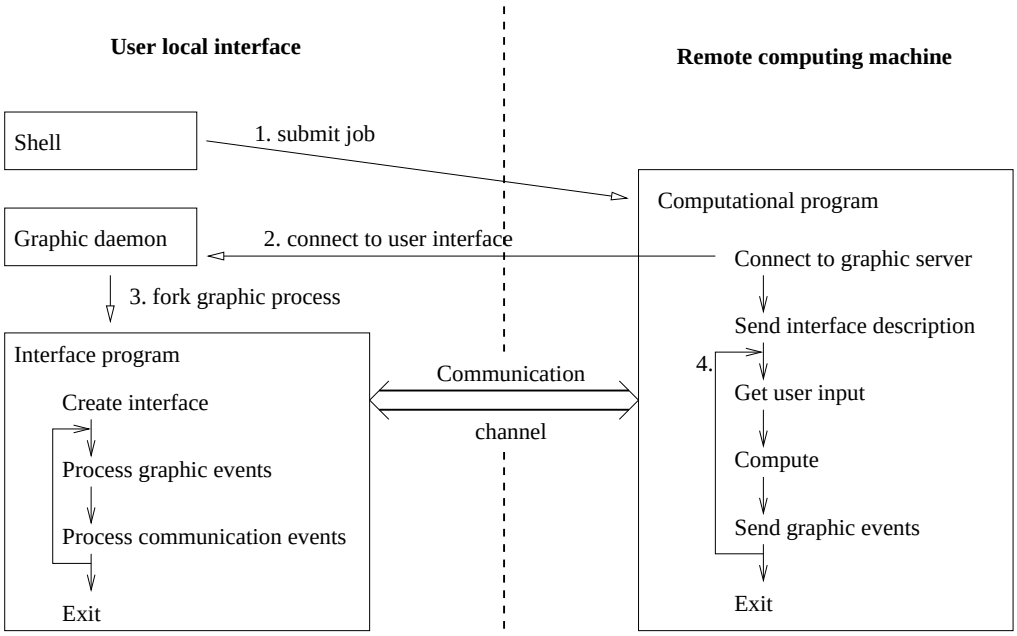


FIG. 2.27 – Remote interactive applications overview (see text).

In order to be transparent from the developer point of view, the same code should be executable either stand-alone (on the user machine, avoiding communication overhead) or remotely, depending on the execution context. To allow this, the library decouples data structures from graphic representations : to each component of the scene (medical object, model, etc) is associated two objects. The data structure object will remain on the computing host. The graphic object does not hold any data except graphic representation parameters. It is using the geometric information contained in the data object for rendering. Each graphic object redefines two redraw methods : one remote method is sending data to be displayed and one local method

is getting the data and actually performs the redraw. In case of stand-alone application, both methods are executed sequentially on the same machine and a stack is used to avoid sending data through the network loop-back interface. In case of remote execution, two instances of each graphic object composing the scene are created. The first instance lies on the computing host and is responsible for sending data to the second instance, that lies on the local host, each time a redraw event is received.

Communications. To obtain a high interactivity level, *e.g.* including real-time requirements such as needed for surgery simulation for instance, communications between the program and the user interface should be as efficient as possible. For some critical applications, a guaranteed network quality of service may even be mandatory. Both user feedback and visualization information should be exchanged between the remote and the local process. Usually, visualization (3D objects) represent much more data than user feedback (mouse clicks) and it is the most critical part in terms of performance.

A two levels communication protocol has been designed in our framework. A higher level interface protocol allows the user to create or delete interfaces widgets and graphic objects. A lower level graphic protocol is designed to send 3D data that compose the graphic scene to the local machine.

Communication channels. In our implementation we are using direct process communication through UNIX sockets for optimal performance. A C++ object encapsulates the communication channels and performs optimization such as packing small messages into a long binary chain that is transmitted using as few packets as possible. On the fly compression could be used to improve performances further. The communication layer is hidden from the user through standard method calls. The system dynamically determines whether it is running stand-alone or remotely and it either executes local code or sends a message to the graphic daemon for the corresponding code to be executed on the local machine.

Interface protocol. The interface protocol defines messages to manage interface creation, user interaction, and program termination. Messages sent from the remote to the local machine include TERMINATE (disconnect and terminate interface program), EVALUATE (evaluate transmitted script), CREATE (create a new graphic object), DELETE (delete a graphic object), and REDRAW (cause a redraw of the 3D scene). Conversely, messages sent from the local to the computing machine include TERMINATE (disconnect and terminate computation program), READ (read new data from a file/stream), WRITE (write data to a file/stream), START (start computation loop), STOP (stop computation loop). the interface communicators on the local and the remote machines periodically poll the incoming messages and respond to these commands.

Graphic protocol. Whenever a viewer window needs to be refreshed (because the viewing parameters changed, the graphic window needs to be redrawn, or the 3D scene has been updated), the local process sends redraw events that cause a redraw procedure of all graphic objects to be called. The redraw procedure needs to retrieve the data content from the remote machine for each graphic object. Therefore, each pair of graphic object instances uses predefined methods to exchange geometric data (scalar values, vectors...). In case of stand-alone execution, these methods use an internal stack to avoid network communication through the local loop-back interface. In case of remote execution, these methods send the graphics information from the remote process to the local machine. The graphic communication protocol includes an INIT message that initiates data transmission from the local side. The remote graphic object

responds by a set of SEND messages transmitting the geometric information needed to build the 3D scene. The local graphic object uses a set of GET methods to retrieve the messages sent in a known order. Data transmission ends with implicit mutual consent.

Remote interface and visualization. In our framework, the interface program is generic and does not depend on a precise application. Technically, the interface may be described by different means such as using java byte code sent through the communication channel. In our implementation we rely on a platform-independent windowing system (wxWindows[15] wrapped in the python[16] scripting language). The script is sent through the channel and executed on the local machine. This solution proves to be very flexible : a program can modify its graphical interface dynamically depending on the algorithm evolution and the user interactions. There is no limitation on the generated interface induced by the system.

When an INIT message is received by a graphic object on the remote side, it sends all graphic information needed for redisplaying that object. The following pseudo-code illustrates the redraw methods of a polygon made of a list of `n` vertices represented as vectors :

```
void Polyon::localRedraw() {
  int n = getInteger();
  Vector v1 = getVector();
  for(int i = 1; i < n; i++) {
    Vector v2 = getVector();
    draw(v1, v2); v2 = v1;
  }
}
void Polyon::remoteRedraw() {
  int n = data->getNbVertices();
  sendInteger(n);
  for(int i = 0; i < n; i++)
    sendVector(data->getVertex(i));
}
```

The INIT message precedes the entrance into the local redraw code and causes the remote redraw code to start. Only SEND messages from the remote to the local machine are then needed until all data has been exchanged and the communication ends by mutual consent. From the user point of view, message exchanges are hidden in the get/send function calls that are defined for every primitive types or array of primitive types. During a stand-alone execution, both local and remote codes are executed on the same machine. The remote code is first executed. To avoid the overhead of a communication channel, the send methods are redefined to push the addresses of sent data onto a stack. The local code then pops the data out of the stack during the get methods. Using a stack of pointers avoids useless memory copies.

When executing a remote application, two parallel processes are running on two machines : the computation process that periodically sends redraw event and the graphic process that receives and treats redraw events. Depending on the relative cost of the computation loops and the redisplay process, the two processes can easily get desynchronized. If the computation process is slower (the usual case for costly applications that require remote execution), the graphic process is just idle between two redraw events. However, if the computation process is faster, it can overflow the graphic process with redraw events and spend too much time in network transmission of the data. To avoid that, our graphic process rejects incoming redraw

[15] http://www.wxwindows.org
[16] http://www.python.org

events when it is already busy by processing such an event. The computing process can then skip data transmission and iterate several times before the 3D scene is refreshed. It is also possible to enforce a minimum redisplay rate, at the cost of slowing down the computation process, in cases where the computations would evolve too fast for the user to get a chance to react.

Transparency from the user point of view. The proposed system was designed to be as transparent as possible from the user point of view, while remaining efficient. Indeed, most messages are completely hidden by the graphic system and the user programs call methods that cause the messages to be exchanged. In case of a stand-alone execution, the behavior of the system changes transparently for efficiency reasons avoiding useless message exchanges.

Results and discussion

The interactive simplex mesh-based segmentation algorithm introduced in section 2.1.1 has been ported on our platform for testing. A generic model with the *a priori* shape of the organ is embedded into the 2D or 3D medical image. During the energy minimization procedure, the model iteratively deforms to better fit the image content [146]. At each iteration, the current shape of the model is updated and displayed on the user screen together with the input image. The user can interact by grabbing the model in areas where the automatic convergence is not satisfactory.

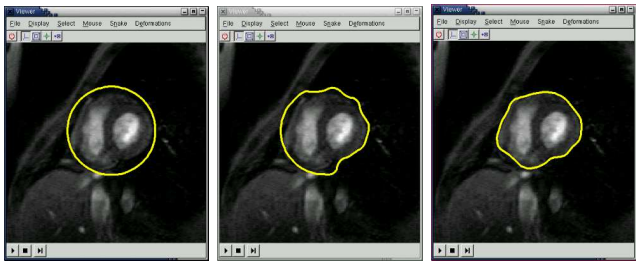


FIG. 2.28 – 2D segmentation. Left : initialization. Middle : automatic result. Right : supervised result.

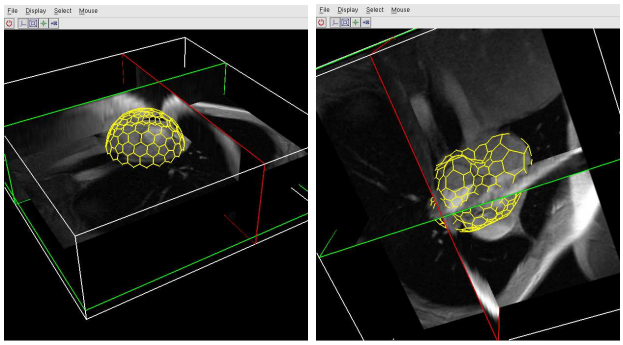


FIG. 2.29 – 3D segmentation. Left : initialization. Right : supervised result.

Figure 2.28 shows a 2D deformable contour used to segment the heart left ventricle contour from a cardiac Magnetic Resonance slice of the thorax. The initial contour (a simple circle) is overlaid on an MR slice on the left. The result of automatic segmentation is shown in the

center, while the result obtained by user interaction (a few mouse clicks while the model is being deformed) is shown on the right. Similarly, figure 2.29 shows a deformable surface used to segment the heart left ventricle from a Magnetic Resonance volume. On the left is shown the surface initialization overlaid on a 3D medical volume represented by 3 orthogonal slices and on the right is shown the result of supervised segmentation.

Figure 2.30 shows several performance measurements of the remote execution of 2D and 3D interactive segmentations. In the first row are reported the total number of bytes exchanged during a segmentation task and the corresponding network usage to measure the network load. Application-related measurements are also given : the number of iterations per seconds is dependent on the algorithm efficiency. The number of iterations per redraw depends on the relative time needed to make one computation loop and one scene redraw. The total execution time gives an application performance indicator for a fixed number of iterations (given amount of computation). Each measure is reported in 3 cases (columns) : (1) the stand-alone application that do not need network transmission of the data, (2) the remote application running the graphic daemon on the same host than the computing process, and (3) the remote application running the graphic daemon and the computing process on two different hosts connected over an IP network (with a measured 200Kbits/s bandwidth).

To interpret these results, note that the stand-alone application only involves one process. Therefore, the algorithm evolution and the graphic rendering are performed sequentially, with one rendering for each algorithm iteration. Conversely, in the remote application case, two concurrent processes are running (either on the same machine in the local case or on two different machines in the remote case). Due to the relative cost of the rendering compared to a single computation iteration with the fast models used in this testing, this is penalizing for the stand-alone application while the remote application performs several iterations for each rendering. This results in a longer execution time of the stand-alone application and a much larger number of bytes sent to the renderer. Past the initialization stage, where the background image is sent, the number of bytes transmitted for rendering is only dependent on the model size (the number of vertices : *i.e.* 50 vertices for the 2D contour and 500 vertices for the 3D surface in this example) and is fairly low (tens of bytes per second in the 2D case and thousand of bytes per second in the 3D case) even without compression (for higher resolution models, data compression could be valuable though). This results in a very fluid interaction and makes possible the remote execution of the supervised segmentation application.

As a comparison, using the X display client/server mechanism for executing the same applications requires to transmit the complete interface window bitmap picture at each redraw event. This results in so poor performances that the 2D applications can hardly be considered as interactive anymore and the 3D application just could not really be executed interactively on a remote machine using the X server capabilities alone.

	2D experiment			**3D experiment**		
Execution type	stand-alone	local	remote	stand-alone	local	remote
Number of bytes sent	1243	55	45	14667	3529	2473
Throughput (Mbits/s)	0.053	0.012	0.008	0.288	0.097	0.042
Iterations / second	42.56	209.22	186.86	19	27.56	17.03
Iterations / redraw	1	51.88	64.64	1	5.8	8.12
Execution time	4.70	0.96	1.07	10.2	7.26	11.74

FIG. 2.30 – Remote segmentation performances.

2.4.3 Data distribution to match computing resources availability

When dealing with intensively data-parallel applications such as image analysis procedures to be applied on complete image databases, a grid is naturally well suited for distributing the resulting computation load since computations are usually independent. However, deciding on the job granularity is not a straight forward issue : a maximal parallelism can be achieved by submitting small grain tasks (one data segment to be processed per task) but this also leads to a maximal load for the grid infrastructure (maximal number of tasks to handle). Conversely one can increase the tasks granularity, thus increasing each task duration but also lowering the middleware load. In addition, on large scale grid infrastructures, the overhead due to submission, scheduling, etc, can be in the order of a few minutes for each task. Reducing the number of tasks also ensures the total time spent in waiting and queuing state. A trade-off therefore has to be found and an optimal task granularity exists. This trade-off depends on the relative duration of tasks processing and the middleware overhead.

We have explored different approaches to optimize the task granularity. in the first one, introduced in section 2.4.4, knowing the exact nature of the image processing algorithms applied we predict the application execution time though an algorithm complexity model. We make measurements to model the behavior of the grid infrastructure and we are then able to analytically determine an optimized granularity. In the second approach, introduced in section 2.4.5, we make the assumption that all tasks have a constant execution time and we focus on the modeling of the grid middleware overhead at execution time. The grid is considered as a complex system which cannot be modeled in a deterministic way. We rather consider a stochastic model for the grid response time. In both cases we demonstrate that it is possible to reduce the total application execution time and to lower the middleware load which is beneficial when sharing the infrastructure with other users.

This development is illustrated through a realistic application to medical image databases exploration through content-based queries. Applications to digital medical images represent tremendous amounts of data for which automatized indexing and search tools are increasingly needed. With the arising of medical record databases in hospitals, physicians have access to precious data sets containing an history of sample data, diagnoses, medical interventions, and results, that could help in indexing and analyzing new data. However, tools are needed to manipulate and search for relevant data in these databases. Due to the amount of data and the the complexity of image analysis algorithms, these tools are both data and computationally intensive.

In this example we focus on a medical image similarity measurement application used for various medical image registration and recognition algorithms. Similarity measures are useful to analyze the similarity between the content of two images. The typical use case for this application is a physician searching for known medical cases close to a case she has to diagnosis : she want to find in the database all images with a close correlation to a sample image she is studying to be able to confirm her diagnosis by looking at other similar records. To perform efficient queries on a grid, the system needs to partition the database in subsets that will be independently processed on different processors. We study the trade-off between distributing a small number of large tasks dealing with large data sets and a large number of small and short tasks.

2.4.4 Optimized granularity through algorithm complexity modeling

We assume that the content-based search application is running on a grid of standard PC machines connected to a local area network. The medical images are available from a medical data server recording both images and metadata associated to these images. Figure 2.31 illustrates this application workflow. The user first selects a sample image. A data set of candidate

images (i.e. images of the same region body, acquired with the same imager, etc.) is determined by selecting images on their metadata in the database. The candidate images are then transported to the grid for analysis. For each candidate image, a similarity measurement is computed between the candidate and the user sample. The similarity computation measure results in a score attributed to the candidate. Once all candidates have been processed, the scores are ranked and the user can retrieve the highest score images corresponding to the most similar cases stored in the database.

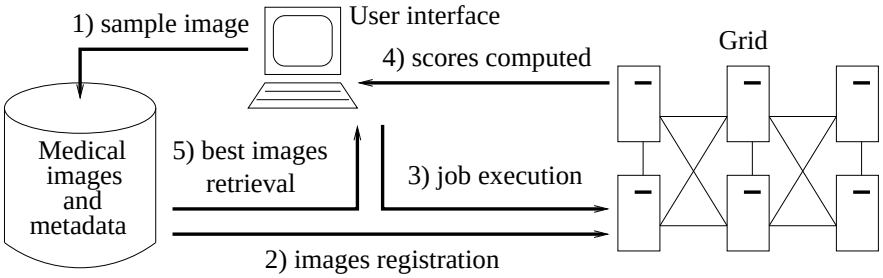


FIG. 2.31 – Content-based query on medical images

Figure 2.32 shows an example of the algorithm execution. The upper left image is the user sample image (a Magnetic Resonance Image of the thorax). The other images are candidates (all classified as thorax MRI in the system) ranked by their similarity score (2 high scores, 2 low scores shown).

FIG. 2.32 – From up to down and left to right : a sample image and ranked candidate images with high and low similarity scores

Several similarity measures may be used to measure the differences between images. Although each measurement is not very computer intensive, the comparison of a sample image against a complete database is intractable, in a reasonable time, on a single computer due to the size of medical databases. The actual cost of such a computation depends on several parameters such as the input image size and the computation precision desired as detailed below. This

application is very scalable since each computation task can process any number of similarity measurements and the optimization of the computation cost on a distributed system is not obvious.

Executing the application using a grid middleware introduces a significant overhead on the computation cost. In the case of similarity measurements, this overhead is far from negligible as compared to single tasks computations. Figure 2.33 shows the typical execution time for a same query executed at different granularities (the X axis representing the granularity level). An optimal value significantly improves the computation time.

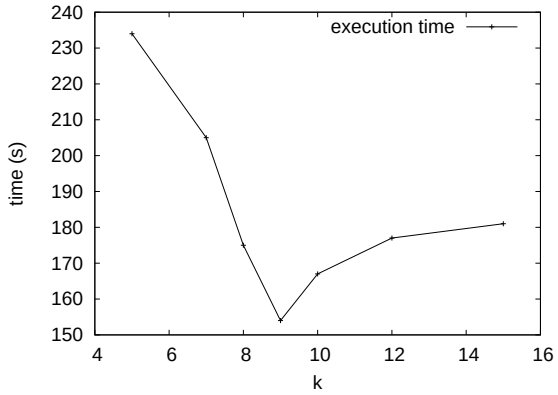


FIG. 2.33 – Time (in second) needed to answer a query depending on the application granularity

Middleware

The middleware and its capabilities have a significant impact on the application computation time. Although grid technologies are still in their youth, the growing activity around grids led to many middleware components development. Early hour middlewares focused on embarrassingly parallel applications but more sophisticated approaches are being proposed to deal with different problems. after early testing on the European DataGrid project middleware [150], which proved to be very unstable at that time, our application has been developed on our own middleware layer called μgrid[17]. μgrid is designed to be very light weighted in order to remain easy to use and maintain. It was designed to access cluster of PCs available in laboratories or hospitals. Therefore, it does not make any assumption on the network and the system installation except that independent hosts with their private CPU, memory, and disk resources are connected through an IP network and communication is possible on one port with each machine. This middleware is a research prototype although it provides the basic functionality needed for such an application. It is written in C++, C doubtlessly being the ideal language for system programming. It is built on a few standard components such as the OpenSSL library for authenticated and secured communications, and the MySQL C interface to access the MySQL database server.

μgrid includes a data manager and a job controller. A daemon is running on each host (later on referred to as *grid node*) to receive data and job related requests. A farm manager is the entry point in the system. The user can connect to the farm manager from any host through a programmable user interface or command line tools. If she is properly authenticated, her request is transmitted to a grid node for processing. The client then makes direct connection to the grid node for data or job information exchange to release the farm manager. As can be

[17] μgrid middleware, http://www.i3s.unice.fr/~johan/microgrid

seen, this solution is not a real grid implementation : it is not scalable yet as it is centralizing access on a single farm manager. This is meant to evolve in future developments. However, the farm manager is developed to remain as light weighted as possible, delegating every possible actions to the grid nodes. The middleware is implementing an active resource allocation policy. Each time a worker node is started or finishes a task, it declares itself to the farm manager. The farm manager keep track of the busy and ready nodes. Once a node becomes ready, it can receive requests from the farm manager.

Authentication. Each user is authenticated through an X509 certificate. The certificate is signed by a certification authority. A farm manager or a grid node will only accept valid and signed certificates. All communications between the user and the nodes are encrypted using the OpenSSL[18] public key interface. Each grid node also authenticates itself with a certificate. The client interface or the farm manager will not accept communications with unauthenticated hosts for security reasons.

Data management. Since we do not make any assumption on each host file system, each host is suppose to dispose of its own storage resources, not necessarily visible from the other hosts. At creation time, a grid node declares its available space to the farm manager and will send frequent updates of this value. The farm manager holds a catalog of all files known to the middleware. Each file is described by a unique grid wide identifier. Thus the user can refer to a file without needing to know its physical location. A file becomes known to the grid once it has been registered : the user transfer the file from its local machine or any external storage through the middleware interface. The file is registered (its identifier is written in the farm manager files table) and a physical copy is stored on a grid node. The farm manager holds a table giving associations between the file identifier and its physical replicates. Several replicates may exist on several nodes for a file. Indeed, when a host is responsible for executing a job, it needs to access a set of files manipulated by this job. Since all job files are not necessarily located on a single node, they are first copied onto the target node before the task is started. These multiple instances of a file are then kept on the nodes, unless disk space is lacking, for caching in case of subsequent use. This replication of files causes an obvious problem of coherence that is not handled in the current implementation : the user is responsible for creating a new file in case of modification. Our middleware controls the access to file authorization through the user certificate subject. The subject string is stored in the farm manager table on file registration allowing the system to control file access at each user level.

Job control. The farm daemon is also responsible for assigning tasks to grid nodes. When a user submits a job, she can specify some system requirements and the files needed for this job to execute. The farm manager will search for possible target nodes matching the system requirements. It sorts the possible candidates list on (1) their availability, (2) the amount of data that need to be transferred before starting the task, and (3) their processing power. After this basic scheduling, the task is assigned to the first host that becomes ready in the list. The farm manager thus orders automatic replication of files on need. Replication is actually done directly between the grid nodes owning and receiving a replica.

Tasks life cycle. Given the middleware registration procedure, the execution time for a task includes : (1) the file registration time if the task requires file from outside the grid, (2) the task scheduling and queuing time (t_{sch}), (3) the time for files replication when needed (t_{rep}), and (4)

[18] OpenSSL, http://www.openssl.org

the task execution time (t_{task}). In the later we will ignore the task registration time that depends on external components and we will consider that all needed files have been preregistered.

Let us now consider the image similarity measurement application where N candidate images should be tested against the user sample image. For optimizing the computation time, we might want to process data by subsets of k images, resulting in N/k tasks to be processed. Each task is the sequential execution of k similarity measures and its total execution time is therefore kt_{task}. Assuming that a sufficient amount of resources is available to process all tasks in parallel, the total parallel computation time will be kt_{task} while the sequential computation time would be Nt_{task}.

The grid overhead is the time needed to schedule the tasks and replicate the files for each of them : $N/k(t_{\text{sch}} + t_{\text{rep}})$. Our purpose is to find the optimal value for k such that :

$$k = \min_k \left(\frac{N}{k}(t_{\text{sch}} + t_{\text{rep}}) + kt_{\text{task}} \right) \tag{2.5}$$

In order to estimate the optimal value for k we need to estimate t_{sch}, t_{rep}, and t_{task}. t_{task} can be derived from the insight of the algorithm as shown below. t_{sch} and t_{rep} are dependent on many parameters and we have been measuring suitable values from testing runs.

Similarity measurements cost

The complexity of all similarity measures depends on the size and the dynamic range of the medical images processed. Furthermore, the complexity of computations varies from one measure to another. We give here a brief overview of the similarity measures implemented for this test and we estimate their computational complexity. All similarity measures proposed first need the computation of the joint histograms of the images to compare. Then the cost of the similarity measure itself is estimated.

Let I and J denote the two images to compare. Both images are supposed to have the same size with length l, height h, and number of slices s. Thus the total number of voxels per image is $n = l \times h \times s$. Medical images are gray level images usually coded using 8 or 16 bits. Let $r \in [2^8, 2^{16}]$ be the dynamic range of the image gray level values.

We denote n_{ij} the number of voxels with intensity $i \in [0, r[$ in the first image and $j \in [0, r[$ in the second image (*i.e.* the cell (i, j) of the joint histogram). Let us define :

$$n = \sum_{ij} n_{ij}$$

the total number of voxels, and

$$p_{ij} = \frac{n_{ij}}{n}$$

the normalized number of voxels in cell (i, j) of the joint histogram. We further define :

$p_i = \sum_i p_{ij}$	the lines normalized sum
$p_j = \sum_j p_{ij}$	the columns normalized sum
$m_I = \sum_i ip_i$ and $m_J = \sum_j jp_j$	I and J means
$\sigma_I^2 = \sum_i (i - m_I)^2 p_i$ and $\sigma_J^2 = \sum_j (j - m_J)^2 p_j$	I and J variances
$m_{J\|i} = \frac{1}{p_i} \sum_j jp_{ij}$ and $m_{I\|j} = \frac{1}{p_j} \sum_i ip_{ij}$	the conditional means
$\sigma_{J\|i}^2 = \frac{1}{p_i} \sum_j (j - m_{J\|i})^2 p_{ij}$ and $\sigma_{I\|j}^2 = \frac{1}{p_j} \sum_i (i - m_{I\|j})^2 p_{ij}$	the conditional variances

Using the above notations, we have implemented 6 classical similarity measures :

– The sum of differences, **SD** :

$$\text{SD}(I, J) = \sum_i \sum_j p_{ij}|i - j|$$

– The sum of squared differences, **SSD** :

$$\mathrm{SSD}(I,J) = \sum_i \sum_j p_{ij}(i-j)^2$$

– The coefficient of correlation, **CC** :

$$\mathrm{CC}(I,J) = \sum_i \sum_j \frac{(i-m_I)(j-m_J)}{\sqrt{\sigma_I}\sqrt{\sigma_J}}$$

– The woods criterion, **Woods** (asymmetric measure) :

$$\mathrm{Woods}(I|J) = \sum_j \frac{\sigma_{I|j}}{m_{I|j}} p_j$$

– The ratio of correlation, **RC** (asymmetric measure) :

$$\mathrm{RC}^2(I|J) = 1 - \frac{1}{\sigma_I^2} \sum_j \sigma_{I|j}^2 p_j$$

– The mutual information, **MI** (asymmetric measure) :

$$\mathrm{MI}(I|J) = -\sum_i \sum_j p_{ij} \frac{p_{ij}}{p_j}$$

For cost estimation, we will consider a time unit c roughly corresponding to the time needed for a executing floating point operation on the microprocessor (*i.e.* c is in the order of few nanoseconds on a 1GHz processor).

Joint histogram computation cost. The joint histogram is a $r \times r$ sparse matrix. Its construction means the computation and storage of all p_{ij} values. For practical reasons, the joint histogram can be stored in a 2 dimensions array if r is small enough. However, $r = 2^{16}$ would imply a 2^{32} cells histogram which is larger than most machines memory capacity. Therefore, we store the large sparse histograms as an array of r lines, each made of a linked list of non null column cells. In practice, we set the switch threshold to $r = 2^{12}$: for $r \leq 2^{12}$, the joint histogram is a 2 dimensions matrix while for $r > 2^{12}$, the joint histogram is a vector of lists. Therefore, the joint histogram computation cost and access time depends on r.

The joint histogram construction involve the initialization of the cells, the images parsing to compute the n_{ij} and the normalization to compute the p_{ij}.

Joint histogram for $r \leq 2^{12}$. The joint histogram computation using an $r \times r$ array involves :

– The initialization (allocation and affectation) of the r^2 matrix cells. The unitary cost for allocating and initializing a cell is estimated to $5c$ from empirical measurements.
– The parsing of all image voxels (n retrieval and additions). The unitary cost is estimated to $12c$.
– The normalization of all coefficients (r^2 retrieval and divisions). The unitary cost is estimated identical as above : $12c$.

This sums to :

$$H(n, r \leq 2^{12}) = 5r^2c + 12nc + 12r^2c = (17r^2 + 12n)c \tag{2.6}$$

Joint histogram for $r > 2^{12}$. The joint histogram computation using a sparse matrix involves :

- The initialization (creation of r empty lists).
- The parsing of all n image voxels and the histogram update.
- The normalization.

The unitary costs for all these operations are to some extent implementation dependent and difficult to determine theoretically due to the compiler optimizations while generating code. Therefore, we have made measurements of the average costs in our code, leading to the following estimates :

- Each list creation costs $20c$.
- The cost of each update in the histogram depends on the pixel value (all lists are not evenly balanced) and averages to $5000c$.
- The normalization of each histogram row averages to $4000c$.

This sums to :

$$H(n, r > 2^{12}) = 20rc + 5000nc + 4000rc = (4020r + 5000n)c \tag{2.7}$$

Assembling equations 2.6 and 2.7, the histogram construction cost is therefore :

$$H(n, r) = \begin{cases} (17r^2 + 12n)c & \text{if } r \leq 2^{12} \\ (4020r + 5000n)c & \text{if } r > 2^{12} \end{cases} \tag{2.8}$$

Similarity computation cost. The similarity measures computation cost is highly dependent on the cost for accessing the p_{ij} histogram value. Based on the above mentioned assumptions, this cost is estimated to be $12c$ for $r \leq 2^{12}$. In this case, the computation cost of each similarity measure may be estimated.

Let us first estimate the computation cost for the statistical values p_i, m_I, σ_I, $m_{I|j}$, and $\sigma_{I|j}$. Let $C()$ designate the cost function :

- set of $p_i, \forall i$:

$$C(p_i) = rC\left(\sum_i p_{ij}\right) = r(r \times 12c) = 12r^2c$$

- m_I, or m_J :

$$C(m_I) = C(m_J) = C\left(\sum_i ip_i\right) = rC\left(ip_i\right) = r(c + 12rc) = 12r^2c + rc \approx 12r^2c$$

- σ_I, or σ_J, given that m_I or m_J, and the set of p_i for all i have been precomputed :

$$C(\sigma_I) = C(\sigma_J) = C\left(\sum_i (i - m_I)^2 p_i\right) = rC\left((i - m_I)^2 p_i\right) = 3rc$$

- subsequently, the computation time for m_I, m_J, σ_I, and σ_J sums up to :

$$C(m_I, m_J, \sigma_I, \sigma_J) = 24r^2c + 6rc \approx 24r^2c$$

- the conditional means computation cost is :

$$C(m_{J|i}) = C(m_{I|j}) = rC\left(\frac{1}{p_i}\sum_j jp_{ij}\right) = r(2c + r(c + 12c)) \approx 13r^2c$$

– finally, the conditional standard deviation computation cost, given that the conditional mean as been precomputed, is :

$$C(\sigma^2_{J|i}) = C(\sigma^2_{I|j}) = rC\left(\frac{1}{p_i}\sum_j (j - m_{J|i})^2 p_{ij}\right) = 15r^2c$$

– therefore, the total cost for conditional means and standard deviations is :

$$C(m_{J|i}, \sigma_{J|i}) = 28r^2c$$

Given the above statistics computation cost, it is now possible to estimate the similarity measures cost :

– Sum of differences :

$$C(\mathrm{SD}, r) = C\left(\sum_i \sum_j p_{ij}|i-j|\right) = r^2(C(p_{ij}) + 3c) = 15r^2c$$

– Sum of squared differences : Following the same computation as above,

$$C(\mathrm{SSD}, r) = C(\mathrm{SD}, r) = 15r^2c \tag{2.9}$$

– Coefficient of correlation :

$$\begin{aligned} C(\mathrm{CC}, r) &= C\left(\textstyle\sum_i \sum_j \frac{(i-m_I)(j-m_J)}{\sqrt{\sigma_I}\sqrt{\sigma_J}}\right) + C(m_I, m_J, \sigma_I, \sigma_J) \\ &= r^2(7c) + 24r^2c = 31r^2c \end{aligned} \tag{2.10}$$

– Woods' criterion :

$$\begin{aligned} C(\mathrm{Woods}, r) &= C\left(\textstyle\sum_j \frac{\sigma_{I|j}}{m_{I|j}} p_j\right) + C(m_{J|i}, \sigma_{J_i}) \\ &= r(2c) + 28r^2c \approx 28r^2c \end{aligned} \tag{2.11}$$

– Ratio of correlation :

$$\begin{aligned} C(\mathrm{RC}, r) &= C\left(1 - \tfrac{1}{\sigma_I^2}\textstyle\sum_j \sigma^2_{I|j} p_j\right) + C(m_{J|i}, \sigma_{J_i}) \\ &= 2c + rc + 28r^2c \approx 28r^2c \end{aligned} \tag{2.12}$$

– Mutual information :

$$\begin{aligned} C(\mathrm{MI}, r) &= C\left(\textstyle\sum_i \sum_j p_{ij} \frac{p_{ij}}{p_j}\right) + C(p_j) \\ &= r^2(24c + 2c) + 12r^2c = 38r^2c \end{aligned} \tag{2.13}$$

The total computation cost for a similarity measure M applied over two images with n voxels and dynamic range $r \leq 2^{12}$ is therefore :

$$\mathrm{Cost}(\mathrm{M}, n, r) = H(n, r) + C(\mathrm{M}, r) \tag{2.14}$$

where $H(n, r)$ is defined in equation 2.8 and $C(\mathrm{M}, r)$ is one of equation 2.4.4, 2.9, 2.10, 2.11, 2.12, or 2.13.

For $r > 2^{12}$, the theoretical cost for similarity measures is made difficult due to the use of the sparse matrix. Indeed, the sparse matrix fullness depends on the actual images gray level dispersion. This matrix fullness has a direct effect on the p_{ij} retrieval time and therefore on the overall computation time. On one hand, the sparse matrix structure increases the time for the retrieval of a p_{ij}, but on the other hand, as we only consider non zero values of the histogram, the number of operations involved in similarity measures is often far less than n or n^2 in practice.

In our experiments on 16 bits voxel images (see table 2.9), it appears that the similarity computation time is much smaller than the joint histogram computation time and that it is in the order of magnitude of the computation time for 12 bits voxel images. We will therefore make the approximation :

$$C(\mathrm{M}, r = 2^{16}) \approx C(\mathrm{M}, r = 2^{12}) \tag{2.15}$$

Experimental validation

We have used 3 sets of images for validating the computation cost model as summarized in table 2.8. Each image's gray level range may be undersampled prior to processing. This loss of precision brings an improved computation time. Following experiments are therefore using undersampled versions of the original images to 8 and 12 bits when possible.

	set 1	set 2	set 3
Number of image	124	238	456
Dimensions	256×256	181×217×181	181×217×181
Size (n)	65536	7109137	7109137
Gray level range	[0, 411]	[-32768, 32767]	[0, 4095]
Precision (r)	412	2^{16}	2^{12}

TAB. 2.8 – Test images size and gray level ranges.

Table 2.9 shows the measured computation time (in seconds) for the similarity measures (excluding image I/O and undersampling) on pairs of the 3 above mentioned image data sets, the 6 similarity measures proposed, and every possible undersampling to 8, 12, and 16 bits. The times were measured on a 800MHz Intel Pentium III processor.

n	r	histo	SD	SSD	CC	RC	Woods	MI
set 1	2^8	0.030	0.016	0.020	0.033	0.030	0.031	0.040
n=65536	412	0.063	0.041	0.047	0.076	0.070	0.071	0.087
set 2	2^8	0.778	0.017	0.020	0.032	0.030	0.032	0.035
n=7109137	2^{12}	7.897	5.895	6.562	8.277	8.150	7.972	8.029
	2^{16}	693	4.308	4.405	7.321	9.644	9.709	11.272
set 3	2^8	1.021	0.017	0.025	0.041	0.036	0.037	0.071
n=7109137	2^{12}	8.943	5.031	5.600	9.058	8.677	8.670	9.946

TAB. 2.9 – Computations times (in seconds) for joint histogram and similarity measure computation.

Table 2.9 may be compared to the theoretical values computed using equations 2.8 to 2.15 with $c = 20ns$. Results shown in table 2.10 are consistent with table 2.9 and the model may be used for predicting computation time in an optimized scheduler.

n	r	histo	SD	SSD	CC	RC	Woods	MI
set 1	2^8	0.038	0.020	0.020	0.041	0.037	0.037	0.050
n=65536	412	0.073	0.051	0.051	0.105	0.095	9.395	0.129
set 2	2^8	1.728	0.020	0.020	0.041	0.037	0.037	0.050
n=7109137	2^{12}	7.410	5.033	5.033	10.40	9.395	9.395	12.75
	2^{16}	716	5.033	5.033	10.40	9.395	9.395	12.75
set 3	2^8	1.728	0.020	0.020	0.041	0.037	0.037	0.050
n=7109137	2^{12}	7.410	5.033	5.033	10.40	9.395	9.395	12.75

TAB. 2.10 – Theoretical computations times (in seconds) computed from equations 2.8 to 2.15 with $c = 20ns$.

Submission cost

The job submission cost is more difficult to model as it depends on a large number of parameters and inter-system interactions. Figure 2.34 shows two curves measured at run time that we are using to estimate the submission parameters. On the left is shown the time needed for files replication. On the right is shown the scheduling time curve.

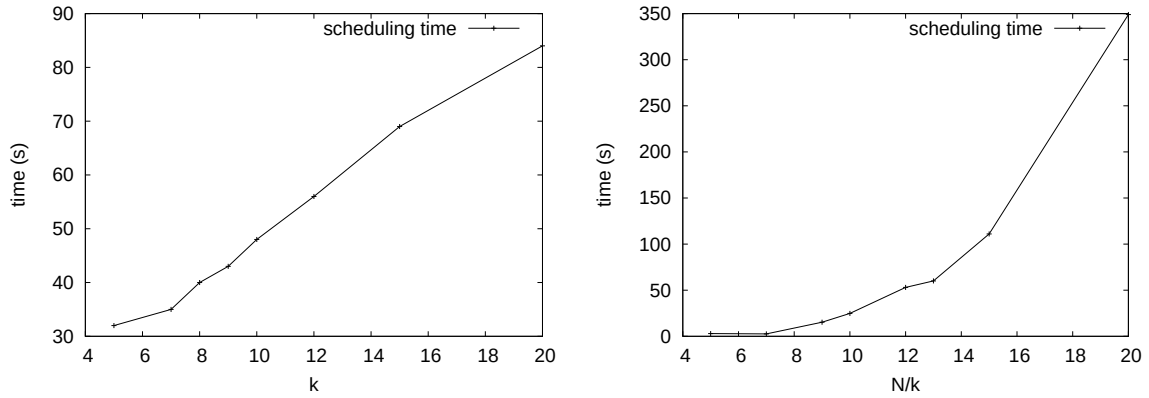


FIG. 2.34 – Left : replication time, right : scheduling time

The file replication cost linearly depends on the image size n. The larger the job granularity, the higher the number of files to transfer. Therefore, we make the assumption that the replication cost linearly depends on k and we estimate the scheduling time such that :

$$t_{\text{rep}} = n(p_1 k + p_2) \tag{2.16}$$

where p_1 and p_2 are estimated by linear regression on the data shown in figure 2.34. We found $p_1 = 5.1910^{-7}$ and $p_2 = 1.57210^{-6}$ in this case.

The scheduler time cost is clearly non-linear. It depends on the number N/k of tasks the scheduler has to deal with. Given the shape of the scheduling time curve, we match it with a quadratic function :

$$t_{\text{sch}} = \max\left(0, p_3\left(\frac{N}{k}\right)^2 + p_4\frac{N}{k} + p_5\right) \tag{2.17}$$

A non-linear regression iterative procedure leads to : $p_3 = 1.02$, $p_4 = 0.08$, and $p_5 = -54.6$.

Experiments

Figure 2.35 shows the measured computation time (plain line) and the estimated time (dashed line) in function of the granularity k for two experiments. The former involves 144 3D images of size $n = 181 * 217 * 181$ and the later 100 3D images of size $n = 150 * 160 * 65$. The 16 bits images where undersampled to 12 bits ($r = 2^{12}$) and the coefficient of correlation similarity measure was used : $t_{\text{task}} = 48r^2c + 12nc$. Inserting this value into equation 2.5 led to the dashed-line estimates shown in figure 2.35.

All experiments have been led on a farm of 8 Pentium III 1GHz PCs with 1GB of RAM and 10GB of free disk space. The estimated curves are rather approximate but sufficient to choose a reasonable value of k. However, the model failed to predict good values of k in the case of much smaller images. The estimates of t_{rep}, and t_{sch} that were done on large images are probably not valid in this case. The model requires further refinements to adapt to different situations.

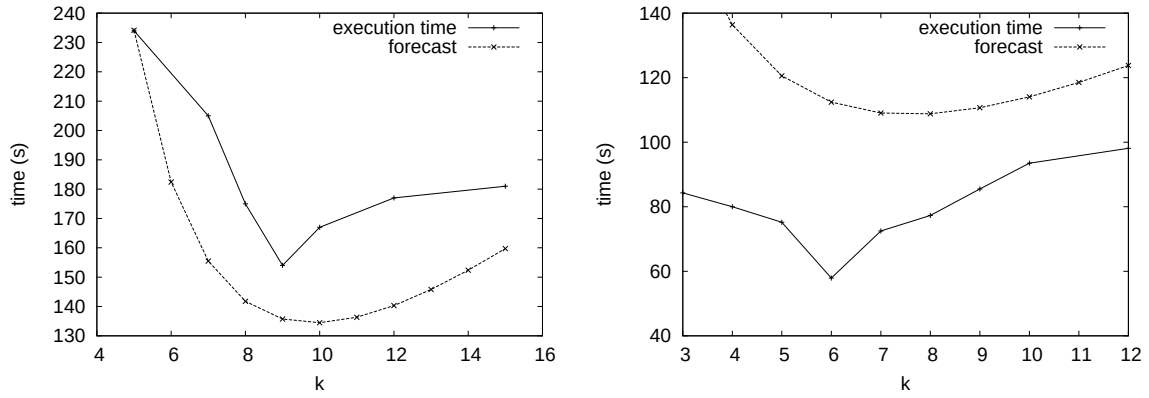


FIG. 2.35 – Estimated computation time versus measured computation time

2.4.5 Optimized granularity through non-deterministic grid modeling

An alternative approach for tackling the optimal load distribution problem is to model the grid middleware through a stochastic model that is taking into account the complexity and unpredictability of a multiuser production grid under variable load. Indeed, a grid deployed over a wide area network proves to have highly volatile resources : the probability for hardware failure is growing with the amount of resources and the network instability can temporarily disconnect full segments out of the infrastructure. Moreover, resources may appear or disappear depending on the needs for maintenance operations, addition to the infrastructure, etc. The full grid topology and status cannot be known at a given instant : grid information providers [203] rely on past and possibly outdated data on the resources.

As production grids are multiusers (and even multicommunity) systems, they are continuously loaded. The evolution of load put on grid resources is yet another hardly predictable parameter. Production systems are only accessible for end users by their user interface giving minimal access to their internal mechanisms. On production grids, we have no access to the core middleware and we can mainly submit computing tasks, monitor their evolution, and collect results, to obtain information on the grid status.

Our ultimate goal is to propose an application-independent optimization strategy for tuning the granularity of the tasks submitted to the grid, given a fixed amomunt of work to execute. This strategy aims at :

1. lowering the total execution time of a job by reducing the grid overhead (user's point of view) ;
2. reducing the total number of tasks submitted for a given job thus lowering the middleware load (infrastructure's point of view).

Problem modeling

Given a total job corresponding to a known CPU time W supposed to be divisible into any number n of independent tasks and a grid infrastructure introducing an overhead G corresponding to the submission, scheduling and queuing time of the tasks, we consider that the execution of the whole task is completed when *all the tasks* are completed. We also make the hypothesis that a task will be affected to a single processor, so that the number n of submitted tasks strictly corresponds to the number of processors involved in the execution. Thus, the goal

is to minimize the total execution time H defined as :

$$H = \max_{i \in [1,n]} \left(G + \frac{W}{n} \right)$$

If we assume G to be a fixed value, the solution is straightforward and n has to be as high as possible. However, this assumption is not realistic in most cases, due to the infrastructure's nature. A more realistic view is to assume G to be dependent both on i and time.

Related work

A complete comparative study of allocation strategies is presented in [90], where task execution times are modeled as random variables with known mean and variance. The author demonstrates the need for a dynamic allocation strategy and he points out that it is associated with an overhead which is assumed to be fixed in his study. He considers a system of initially idle processors and notices that there is a trade-off between overhead and idle time. Under those assumptions, he compares several allocations strategies by presenting simulation results. Our problem cannot be entirely addressed by such methods because (i) allocation strategies consist in optimizing the size of the batches allocated to the processors whereas we are trying to optimize the total number of processors to use, given that every processor will be given a W/n CPU time, (ii) we cannot assume the overhead G to be fixed as it varies along time in our case, and (iii) on a production infrastructure the processors are never idle. This leads to a queuing time that we take into account in the G random variable.

Scheduling techniques are largely studied in the literature. A detailed review of heuristics and a study of the impact of performance estimation on the scheduling strategy is presented in [30]. The authors assume that the scheduler has knowledge of the current topology of the grid, the number and location of copies of all input files and the list of computations and file transfers currently underway or already completed. This kind of solution is hardly usable in our case because we do not assume any a priori knowledge about the infrastructure concerning computations in progress or the current grid topology. In [178] a scheduling method taking into account the stochastic nature of the time to compute one unit of data on a distant processor, supposing that distributions are Gaussian. In particular, the authors notice that penalizing highly variable processors leads to a significant speed-up. Even if the variability of resources has to be taken into consideration, this approach cannot totally suit with our problem because (i) the distribution of G is not assumed to be Gaussian (and is actually not, as shown below), and (ii) the distribution of G has to be dynamically estimated in the multiusers infrastructure we consider. In [166], the author presents decision rules for sequential resource allocation based on dynamic programming. They consider the problem consisting in allocating machines to sequentially arriving tasks. Even if this kind of solution would constitute an interesting perspective, our problem seems not to be treatable by dynamic programming methods because we here consider a set of independent tasks being all submitted in parallel, at the same time, to the infrastructure.

Other works address the task granularity issue, noticing that there is an optimal number of processors to determine to minimize the total execution time, taking into account both computation time and communication time. In [202], they use heuristics to determine a close to optimal configuration, in which tasks are assigned to specific processors to reduce communication overhead induced by routing and contention. Even if it provides good results in their scope, their solution is strictly deterministic and models the communication function linearly in the number of processors, which cannot properly describe the overhead G we need to consider.

Works such as [124] and inside references propose performance analysis methods for task scheduling into embedded systems, considering probabilistic models of task execution times. In

this work, the authors model task execution by a generalized continuous probability distribution and propose a method not restricted to any specific scheduling policy. They consider both execution time and memory aspects. Their method is based on the construction of an underlying stochastic process and its analysis. Even if this approach is entirely probabilistic and makes no assumption on the nature of the probability function of the execution time, which well suits with our hypotheses, they assume all the tasks to be executed concurrently on a single processor.

As a conclusion, the above methods does not seem to completely match our hypotheses. We adopted a probabilistic approach to cope with the variation of the overhead G among the tasks. This approach is detailed in the next section. We address the problem of the dependency of G along time with a dedicated infrastructure monitoring system. Results and an evaluation of the proposed model on a production grid are shown.

A probabilistic model

Let us consider that G is a random variable. If we assume the probabilistic density function (p.d.f) of the random variable G to be $f_G(t)$, then the p.d.f of H will be $f_H(t)$, such as :

$$\begin{aligned} F_H(t) &= P(H < t) = \prod_{i=1}^{n} P\left(G + \frac{W}{n} < t\right) \\ &= P\left(G < t - \frac{W}{n}\right)^n = F_G\left(t - \frac{W}{n}\right)^n \\ \text{Then } f_H(t) &= \frac{dF}{dt} = n f_G\left(t - \frac{W}{n}\right) F_G\left(t - \frac{W}{n}\right)^{n-1} \end{aligned}$$

The problem can then be formulated as a minimization with respect to n of the expectation E_H of the random variable H :

$$\begin{aligned} E_H(n) &= \int_{\mathbb{R}} t f_H(t) dt = \int_{\mathbb{R}} t n f_G\left(t - \frac{W}{n}\right) F_G\left(t - \frac{W}{n}\right)^{n-1} dt \\ &= \int_{\mathbb{R}} n\left(t + \frac{W}{n}\right) f_G(t) F_G(t)^{n-1} dt = \int_{\mathbb{R}} n t f_G(t) F_G(t)^{n-1} dt + \frac{W}{n} \end{aligned}$$

Application to synthetic distributions

Let us first investigate the problem analytically considering synthetic distributions for G, in order to demonstrate the relevance of the method in a controlled environment.

If we assume G to be uniformly distributed between a minimum value a and a maximum value b for example, then an explicit solution can be provided : indeed, we then have :

$$\begin{aligned} f_G(t) &= \begin{cases} \frac{1}{b-a} \text{ if } t \in [a,b] \\ 0 \text{ else} \end{cases} \quad \text{and } F_G(t) = \begin{cases} 0 \text{ if } t < a \\ \frac{t-a}{b-a} \text{ if } t \in [a,b] \\ 0 \text{ if } t > b \end{cases} \\ E_H(n) &= \int_a^b n t \frac{1}{b-a}\left(\frac{t-a}{b-a}\right)^{n-1} dt + \frac{W}{n} = \frac{(n+1)W + bn^2 + an}{n(n+1)} \end{aligned}$$

This result is coherent as $E_H(1) = W + \frac{a+b}{2}$: the execution time on a single CPU is W and the execution suffers from a $\frac{a+b}{2}$ penalty that is the mean overhead introduced by the infrastructure. Moreover, $\lim(E_H(n))_{n\to+\infty} = b$: with an infinite amount of resources, it corresponds to the worst possible overhead introduced by the grid (b) and to the best computation time (0). Indeed, as the number of submitted tasks increases, the probability for one of the tasks to suffer from a high overhead increases. Finally, $\lim(E_H(n))_{n\to 0} = +\infty$: the limit of E_H towards zero

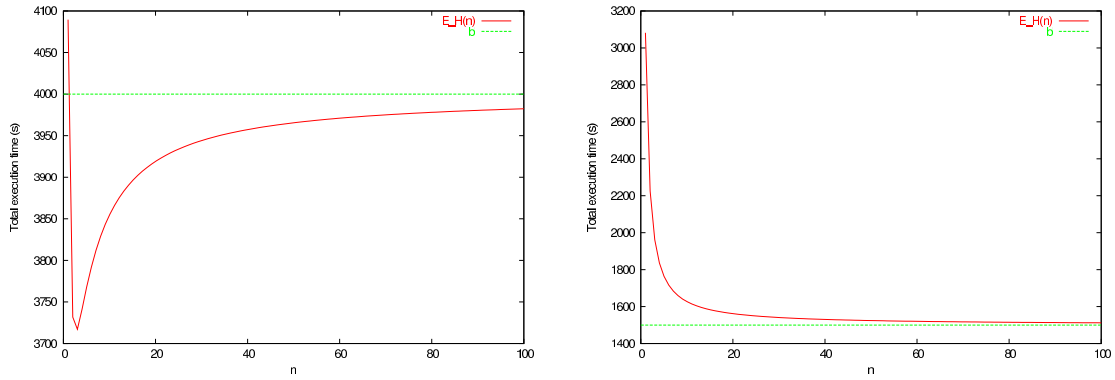


FIG. 2.36 – Representation of $E_H(n)$ for a uniform distribution with $a = 200$s, $b = 4000$s and $W = 2000$s (up) and $a = 700$s, $b = 1500$s and $W = 2000$s (down)

corresponds to the execution of the task on zero machine. In this case, the execution time of course tends to infinity.

The next step is the minimization of the expectation of H. Let us consider its derivative with respect to n :

$$\frac{dE}{dn} = -\frac{n^2W + 2nW + W - bn^2 + an^2}{n^2(n+1)^2}$$

$$\text{If } W \neq b-a, \quad \frac{dE}{dn} = 0 \text{ when } \begin{cases} n_1 = -\frac{\sqrt{(b-a)W}+W}{W-(b-a)} \\ or \\ n_2 = \frac{\sqrt{(b-a)W}-W}{W-(b-a)} \end{cases}$$

n_1 is positive if $(b-a) > W$ and negative otherwise whereas n_2 is always negative. Given that n has to be positive, there is thus a unique optimal number of tasks n_{opt} minimizing $E_H(n)$ if $(b-a) > W$ and we have :

$$n_{opt} = -\frac{\sqrt{(b-a)W} + W}{W - (b-a)}.$$

Such a configuration is represented on the left graph of figure 2.36 where we plotted $E_H(n)$ for a uniform distribution with $a = 200$s, $b = 4000$s and $W = 2000$s. On the other hand, if $(b-a) < W$ then $\frac{dE}{dn} < 0$ so that E_H is strictly decreasing and the optimal number of tasks corresponds to the maximal one. Such a configuration is represented on the right graph of figure 2.36 where we plotted $E_H(n)$ for a uniform distribution with $a = 700$s, $b = 1500$s and $W = 2000$s. If $W = b - a$, then $\frac{dE}{dn} = -\frac{2nW+W}{n^2(n+1)^2}$: it thus has no positive root and here again, the optimal number of tasks corresponds to the maximal one.

We thus can conclude from this particular example that the relative variability $V = \frac{b-a}{W}$ of the grid overhead G plays a strong role into the optimization procedure : whatever the actual mean of G is, if V is low enough, then looking for an optimal job partitioning does not make sense. Indeed, in that case, G can be seen as a fixed value with respect to W and the problem is straightforward.

If we now suppose the distribution of G to be Gaussian, with mean μ and standard deviation

σ, then :

$$\begin{aligned} f_G(t) &= \frac{1}{\sqrt{2\pi}\sigma} e^{\left(-\frac{(t-\mu)^2}{2\sigma^2}\right)} \\ \text{and } F_G(t) &= \frac{1}{\sqrt{2\pi}\sigma} \int_{-\infty}^{t} e^{\left(-\frac{(u-\mu)^2}{2\sigma^2}\right)} du \end{aligned}$$

$$E_H(n) = \int_{\mathbb{R}} nt \frac{1}{\sqrt{2\pi}\sigma} e^{\left(-\frac{(t-\mu)^2}{2\sigma^2}\right)} \left(\frac{1}{\sqrt{2\pi}\sigma} \int_{-\infty}^{t} e^{\left(-\frac{(u-\mu)^2}{2\sigma^2}\right)} du\right)^{n-1} dt + \frac{W}{n}$$

In this case, the relative variability V of the overhead G is denoted by $V = \frac{\sigma}{W}$. Minimizing $E_H(n)$ is hardly analytically feasible but we can estimate a minimum numerically. For example, if we consider $\mu = 600$s and $\sigma = 300$s, figure 2.37 displays the evolution of $E_H(n)$ with respect to n for different values of V ranging from 0.015 to 0.6. We can see on those figures that the higher the relative variability, the deeper the minimum of $E_H(n)$. One can here again conclude that the optimization procedure is particularly suited for environments with a high variability with respect to W.

Applying the model on synthetic distributions showed that it seems coherent and that it is particularly adapted to highly variable environments. But as stated in the introduction, we cannot assume the p.d.f of G to be known. Therefore it should be estimated from measures.

Experimental distributions

Our optimization method is based on the evaluation of the p.d.f of the infrastructure's latency G. Thus, our primary goal was to determine a robust procedure to measure it. Ideally a grid infrastructure should provide this measure from all the tasks submitted by the users. However from our user's point of view, we cannot access the statistics concerning all the tasks submitted to the infrastructure. Thus, the experimental method we adopted was to submit waves of dedicated "ping" tasks to the infrastructure. Those tasks do not process anything and we use them as probes to measure the grid latency, by monitoring their submission, scheduling and queuing times.

The main problem it raises is the fact that the status of the infrastructure may be disrupted by such a measure. Indeed, submitting waves of measure tasks would cause an additional load on the infrastructure, leading to inconsistent measures. To face this problem, we initially submit a limited set of "ping" tasks and then instantaneously submit a new one each time a "ping" task completed, so that the total number of measure tasks running on the infrastructure is constant, leading to a fixed perturbation.

Even if a grid potentially provides an infinite number of resources, and thus allows a theoretical infinite number of task submission, a real infrastructure is actually limited by the maximum number of simultaneous connections from the submission entity and the maximum number of tasks on the scheduler. We empirically tuned the number of "ping" tasks as a trade-off between the accuracy of the measure and the induced overhead. On the target grid infrastructure, we used 50 measurement tasks.

It is true that this kind of method is quite unfair because it introduces a significant overload on the infrastructure. But ultimately, the middleware should provide to the users such statistics computed from all the submitted tasks so that the method would not be invasive.

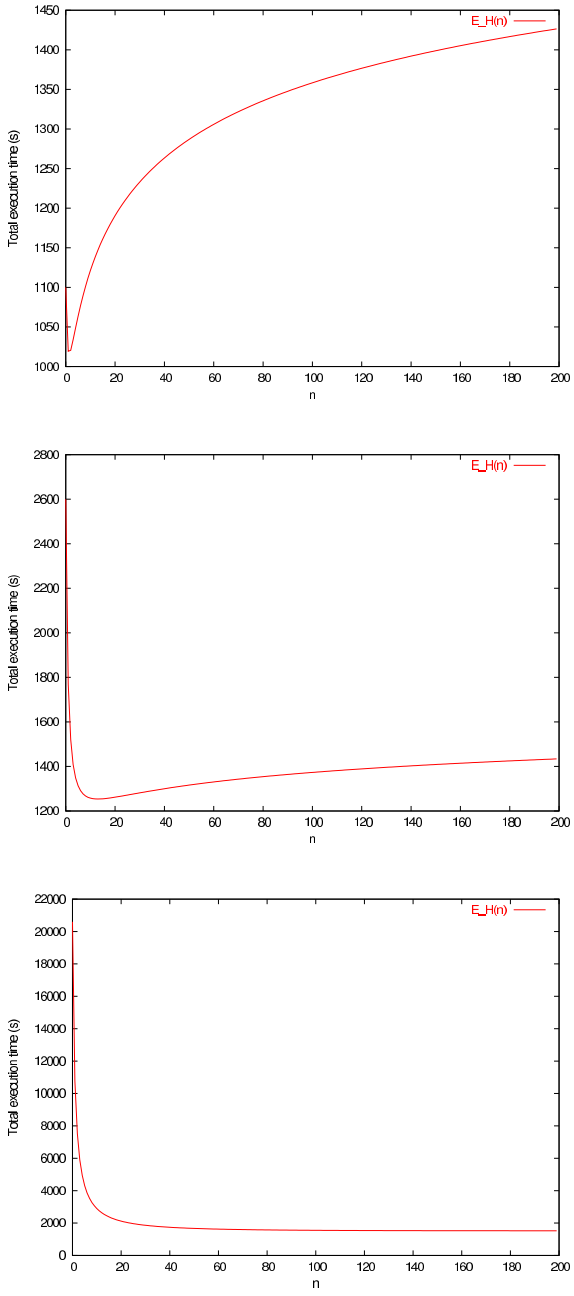


FIG. 2.37 – $E_H(n)$ for a Gaussian distribution with $\sigma = 300$s and $\mu = 600$s. From top to bottom : $V = 0.6$, $V = 0.15$ and $V = 0.015$

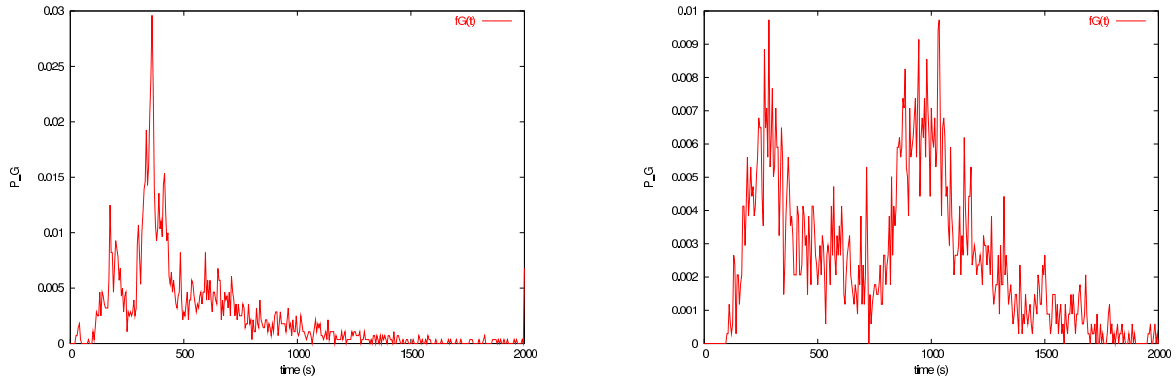


FIG. 2.38 – Examples of p.d.f of G

Timeouts

On a real large scale multiusers grid infrastructure task "losses" may occur because of *e.g.* overfull waiting queues, execution failures on distant heterogeneous machines, network problems and so on. Therefore, setting a timeout to tasks is required to avoid unreasonable waiting times. Taking into account timed-out tasks into the optimization procedure would require to propose a fault-tolerant system handling task resubmissions. Even if we know that it is a very important problem, this will be part of our future work. In this present work, we focus on the validation of the global principle and we thus decided to neglect timed-out tasks, both in the measure scope and in the validation study.

Setting the timeout of tasks to get consistent measures is not straightforward. Indeed, when a measure comes back from the infrastructure, it describes the infrastructure status at the measure's submission instant. Thus, the timeout has to be inferior to the duration while we could consider that the infrastructure's status does not vary. On the other hand, the timeout has to be large enough not to discard too many tasks. For our experiments, we fixed the timeout of tasks to the total CPU time value W of the task, so that timed-out tasks are the ones which would lead to a slowing down of the task by the grid execution.

Estimating and minimizing the probabilistic density function of G

Once we have latency measures, the next step is to determine the p.d.f of the infrastructure's latency G. We did that by considering the 50 last ping measures and gathering them into 5 seconds bins. Obtaining the corresponding p.d.f is then straightforward.

To provide an idea of the overhead times, two sample examples of the p.d.f of G at a given instant are displayed on figure 2.38. As we can see on this figure, the p.d.f is likely to strongly vary along time. Moreover, we can notice that those distributions are clearly not Gaussian. Indeed, they both are neither mono-modal, nor symmetric with respect to their mean.

Once we estimated the p.d.f of G the computation and minimization of $E_H(n)$ is straightforward. We just computed $E_H(n)$ with n ranging from 1 to a maximum value corresponding to the maximum number of tasks submittable to the infrastructure from a single user interface.

Experiments and Results

We made two experiments to evaluate our model on the EGEE infrastructure.

	Min	Max	Avg	Median
δ (seconds)	10	960	258.94	215
$\delta_{normalized}$	0.04	12.64	2.1	1.16

TAB. 2.11 – Errors between model and measures

	Min	Max	Avg
Expected	0	671	162.5
Measured	-775	1308	198.1

TAB. 2.12 – Time speed-up (s) between maximal and optimal strategies

First, we evaluated the model capability to correctly predict the execution time of a set of tasks on the grid infrastructure. We submitted and measured the total execution time of a job, having previously estimated this time with $E_H(n)$. The job is composed of 30 tasks, 67 seconds long each, thus leading to a total CPU time W of 2000 seconds.

Second, we quantified the benefit induced by the model (*optimal strategy*) compared to the naive strategy consisting in submitting a maximal number of tasks (*maximal strategy*). A total CPU time $W = 2000$s is submitted, on the one hand using the optimal number of tasks resulting from the minimization of $E_H(n)$, and on the other hand using a fixed number of 30 tasks (this corresponds to the maximum number of tasks we can submit concurrently on the infrastructure without hitting some performance loss). To avoid bias resulting from an evolution of the grid status between the two submission processes, we alternatively repeated the two strategies up to 88 times, on various day times (mornings, afternoons, nights) spread over a week and using 3 different scheduling hosts.

Experiment 1 : model versus measures. Table 2.11 shows on its upper line statistics concerning the difference δ (in seconds) between the model prediction and the effective measure. In order to quantify the accuracy of the model, we normalized this error with the predicted standard-deviation of the random variable H : $\delta_{normalized} = \frac{\delta}{\sigma_H}$. The table thus shows on its lower line the minimum, maximum, average and median ratios between the measured errors and the standard-deviation σ_H of the random variable H. One can notice that the median ratio is close to 1, that is to say that the measured error is close to the standard-deviation of H. We can thus conclude that the proposed model is effectively able to predict the execution time of a set of tasks on the grid infrastructure.

Experiment 2 : optimal strategy vs maximal strategy. Two different conclusions can be made from this experiment. Task saving : on the 88 experiments, the estimated optimal number of tasks n differed from the maximal one 37 times, that is to say in 42% of the experiments. The remaining 58% correspond to the experiments where the computed optimal n is 30. The total number of submitted tasks is 2580 for the maximal strategy and 1756 for the optimal one. One thus can see that the optimal strategy leads to a total saving of 824 tasks, representing 32% of the tasks submitted in the maximal strategy.

Time speed-up : table 2.12 shows statistics over the 88 executions on the differences (in seconds) between the maximal and the optimal strategies in cases where the computed optimal n differs from the maximal one. Negative values show that the maximal strategy was faster than the optimal one. One can notice that the average speed-up introduced by our optimization strategy is about 200s, which represents 10% of the total submitted CPU time W.

These experimental results demonstrate that (i) a significant speed-up and (ii) a substantial task saving can be obtained using statistical modeling. However, parameters such as the data transfer time and the random nature of the computing power of the resources are not considered by our model. Including them into the partitioning strategy will be part of our future work. We also plan to consider timeouts and fault tolerance elements such as resubmissions in order to propose a more complete strategy for the optimization of job partitioning on a grid infrastructure.

2.4.6 Grid-enabled data-intensive workflows

Assembling basic processing components is a powerful mean to develop new scientific applications. The reusability of data processing software components considerably reduces applications development time. In the image processing community for instance, it is common to chain basic processing operators and image interpretation algorithms to set up a complete image analysis procedure. Workflow description languages are generic tools providing a high-level representation for describing complex application control flows and the dependencies between application components. Workflow execution engines provide the ability to chain the application components execution while respecting causality and inter-components dependencies expressed within this abstract representation. Interfacing workflow managers with a grid infrastructure enables the efficient exploitation of code parallelism embedded in application workflows. It is a mean to transparently provide parallelism, without requiring specific code instrumentation nor introducing much load on the application developers side.

In addition, in many scientific areas applications exhibit a massive data-parallelism aspect that should be exploited. Taking the medical image analysis area as an example, many procedures require the processing of full image databases :

- atlases construction ;
- statistical and epidemiological studies ;
- assessing image processing algorithms ;
- validating medical procedures ;
- ...

In such data-centric applications, the workflow manager should not only efficiently handle control flows but also data flows which might well dominate the execution time.

Another important aspect for easing scientific applications migration towards grid infrastructures is to embed legacy codes into workflows. Indeed, many scientific codes represent tremendous development efforts. It is often undesirable (to take the risk of breaking the accepted validity of the code), or even impossible (when sources are not available for instance), to make any change to these codes.

We are summarizing below our research activity in the area of supporting scientific application workflows. We introduce MOTEUR, a workflow engine interfaced with grid infrastructures, specifically designed to handle data-intensive applications by transparently exploiting both application code and data parallelism. We show that MOTEUR was built in a modern Service Oriented Architecture framework to offer a maximum of flexibility. We provide experimental results for validating our approach on a medical image registration application.

State of the art and definitions

Workflow managers can be classified into two main categories : *control-centric* and *data-centric*. The control-centric managers, such as BPEL [6], are more focused on the description of complex application flows. They provide an exhaustive list of control structures such as

branching, conditions and loop operators. They can describe very complex control composition patterns and some of them are comparable to small programming languages, including a graphical interface for designing the workflow and an interpreter for its execution. Conversely, data-centric managers usually provide a more limited panel of control structures and rather focus on the execution of heavy-weight algorithms designed to process large amounts of data. The complex application logic is supposed to be embedded inside the basic application components. Although there is *a priori* no much contradiction in implementing a workflow manager that is both control and data-centric, the optimization of different managers for different needs often leads to several implementations.

Control-centric managers are commonly implemented to fulfill the e-business community needs. In this area, applications are often not so compute- nor data-intensive and can be described in a high level language suitable for non-experts. Conversely, in the scientific area complex application codes, both compute and data intensive, are frequently available. The workflow description languages are not so rich but the execution engines are better taking into account execution efficiency and data transfer issues [206]. In the remaining of this document, we will consider scientific workflow managers only.

Task-based and service-based approaches. To handle user processing requests, two main strategies have been proposed and implemented in grid middlewares :

1. In the *task-based* strategy, also referred to as *global computing*, users define computing tasks to be executed. Any executable code may be requested by specifying the executable code file, input data files, and command line parameters to invoke the execution. The task-based strategy, implemented in GLOBUS [63], LCG2[19] or gLite[20] middlewares for instance, has already been used for decades in batch computing. It makes the use of non grid-specific code very simple, provided that the user has a knowledge of the exact syntax to invoke each computing task.
2. The *service-based* strategy, also referred to as *meta computing*, consists in wrapping application codes into standard interfaces. Such services are seen as black boxes from the middleware for which only the invocation interface is known. Various interfaces such as Web Services [204] or gridRPC [155] have been standardized. The services paradigm has been widely adopted by middleware developers for the high level of flexibility that it offers (OGSA [64]). However, this approach is less common for application code as it requires all codes to be instrumented with the common service interface.

The task-based approach has been used for grid and batch computing for a very long time. To invoke a task-based job, a user needs to precisely know the command-line format of the executable and the meaning of parameters. It is not always the case when the user is not one of the developers. Input and output data are transmitted through files which have to be explicitly specified in the task description. Invoking a new execution of a same code on different data segments requires the rewriting of a new task description.

Conversely, in the service-based approach the actual code invocation is delegated to the service which is responsible for the correct handling of the invocation parameters. The service is a black box from the user side and to some extent, it can deal with the correct parametrization of the code to be executed. Services better decouple the computation and data handling parts. A service dynamically receives inputs as parameters. The inputs are not limited to files but may also be values of given types (number, text, etc). This decoupling of processing and data is particularly important when considering the processing of complete data sets rather than

[19] LCG2 middleware, http://lcg.web.cern.ch/LCG/activities/middleware.html
[20] gLite middleware, http://www.gLite.org

single data segments. Indeed, grid infrastructures are particularly well suited for data-intensive applications that require repeated processings of different data.

The service-based approach is more dynamic and flexible but it is usually used for accessing remote resources which do not necessarily benefit from grid computing capabilities. This is acceptable for most middleware services that are located and executed on a single server but application services that may require compute-intensive code execution and that are invoked concurrently in the context of the target applications, can easily overwhelm the computing capabilities of a single host. To overcome these limitations some approaches have been explored, such as submission services replacing the straight task submission [80] or generic services for wrapping any legacy code with a standard interface [104, 76].

task-based and service-based workflows. An application workflow can intuitively be represented through a directed graph of *processors* (graph nodes) representing computation jobs and data dependencies (graph arrows) constraining the order of invocation of processors (see left of figure 2.39).

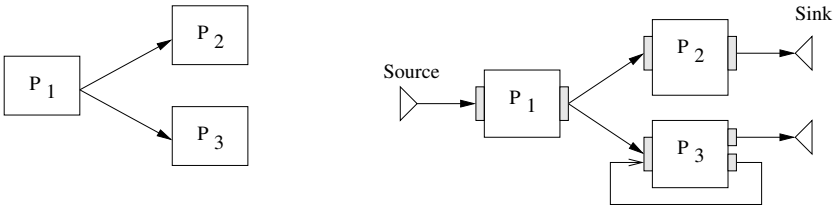


FIG. 2.39 – Simple workflow example. Task-based (left) and service-based (right).

In the task-based approach, the description of a task, or computation job, encompasses both the processing (binary code and command line parameters) and the data (static declaration). Workflow processors directly represent computing tasks. The user is responsible for providing the binary code to be executed and for writing down the precise invocation command line. All computations to be performed are statically described in the graph.

Conversely in the service-based approach, the input data is treated as input parameters (dynamic declaration), and the service hides the code invocation. This difference in the handling of data (static or dynamic declaration) makes the application composition far easier from a user point of view, as mentioned earlier. The service-based approach is also naturally very well suited for chaining the execution of different algorithms assembled to build an application. Indeed, the interface to each application component is clearly defined and the middleware can invoke each of them through a single protocol.

In a service-based workflow, each processor is representing an application component, or service. In addition to the processors and the data arrows, a service-based workflow representation requires a number of input and output ports attached to each processor. The oriented arrows are connecting output ports to input ports. Two special processor nodes are defined : *data sources* are processors without input ports (they are producing data to feed the workflow) and *data sinks* are processors without output ports (they are collecting data produced).

A significant difference between the service and task approaches of workflow composition is that there may exist loops in a service-based workflow given that an input port can collect data from different sources as illustrated in bottom of figure 2.39. This kind of workflow pattern is common for optimization algorithms : it corresponds to an optimization loop converging after a number of iterations determined at the execution time from a computed criterion. In this case, the output of processor P_1 would correspond to the initial value of this criterion. P_3

produces its result on one of its two output ports, whether the computation has to be iterated one more time or not. Conversely, there cannot be a loop in a workflow of tasks. If there were a loop, a data segment would depend on itself for its production. Hence, task-based workflows are always Directed and Acyclic Graphs (DAGs). Only in the case where the number of iterations is statically known, a loop may be expressed by unfolding it in the DAG. An emblematic task-based workflow manager is indeed called Directed Acyclic Graph Manager (DAGMan[21]). Composing such optimization loop would not be possible, as the number of iterations is determined during the execution and thus cannot be statically described. Conversely, in a workflow of services, there may exist loops in the graph of services since it does not imply a circular dependency on the data. This enables the implementation of more complex control structures.

The service-based approach has been implemented in different workflow managers. The Kepler system [120] targets many application areas from gene promoter identification to mineral classification. It can orchestrate standard Web-Services linked with both data and control dependencies and implements various execution strategies. The Taverna project [157], from the myGrid e-Science UK project[22] targets bioinformatics applications and is able to enact Web-Services and other components such as Soaplab services [186] and Biomoby ones. It implements high level tools for the workflow description such as the Feta semantic discovery engine [118]. Other workflow systems such as Triana [195], from the GridLab project[23], are decentralized and distribute several control units over different computing resources. This system implements both a parallel and a peer-to-peer distribution policies. It has been applied to various scientific fields, such as gravitational waves searching [34] and galaxy visualization [194]. The GEMLCA/P-GRADE workflow manager has recently included a so called *parametric study* extension through which it can bridge the gap between the service-based and the task-based approaches to a large extent [87].

Dynamic data sets. Task-based and service-based workflows differ in depth in their handling of data. The non-static nature of data description in the service-based approach enables dynamic extension of the data sets to be processed : a workflow can be defined and executed although the complete input data sets are not known in advance. It will be dynamically fed in as new data is being produced by sources. Indeed, it is common in scientific applications that data acquisition is an heavy-weight process and that data are being progressively produced. Some workflows may even act on the data production source itself : stopping data production once computations have shown that sufficient inputs are available to produce meaningful results.

Most importantly, the dynamic extensibility of input data sets for each service in a workflow can also be used for defining different data composition strategies as introduced in section 2.4.7. The data composition patterns and their combinations offer a very powerful tool for describing complex data processing scenarios as needed in scientific applications. For the users, this means the ability to describe and schedule very complex processings in an elegant and compact framework.

Data synchronization barriers. A particular kind of processors are algorithms that need to take into account the whole input data set in their processing rather than processing each input one by one. This is the case for many statistical operations computed on the data, such as the computation of a mean or a standard deviation over the produced results for instance. Such processors are referred to as *synchronization* processors has they represent real synchronization barriers, waiting for all input data to be processed before being executed.

[21] Condor DAGMan, http://www.cs.wisc.edu/condor/dagman/
[22] http://mygrid.org.uk
[23] http://www.gridlab.org

Services flexibility

The service-based approach enables discovery mechanisms and dynamic invocation even for *a priori* unknown services. This provides a lot of flexibility both for the user (discovery of available data processing tools and their interface) and the middleware (automatic selection of services, alternatives services discovery, fault tolerance, etc).

In the service-based framework, the code reusability is also improved by the availability of a standard invocation interface. In particular, services are naturally well adapted to describe applications with a complex workflow, chaining different processings whose outputs are piped to the inputs of each other.

Another strength of the service-based approach is to easily deal with multiple execution platforms. Each service is called as a black box without knowledge of the underlying execution infrastructure. Several services may execute on different platforms transparently, which is convenient when dealing with legacy code, whereas in the task-based approach, a specific submission interface is needed for each infrastructure.

The flexibility and dynamic nature of services depicted above is usually very appreciated from the user point of view. Given that application services can be deployed at a very low development cost, there are number of advantages in favor of this approach.

Executing services

From middleware developers point of view, the execution of workflow of services is more difficult to optimize than the execution of workflows of tasks though. As mentioned above, the service is an intermediate layer between the user and the grid middleware. Thus, the user does not know nor see anything of the underlying infrastructure. Tuning of the jobs submission for a specific application is more difficult. In addition, data transfers can drastically impact some data-intensive application performances. Services are completely independent. Consequently, for chaining two different services P_0 and P_1, P_0's output data first needs to be returned to the user before being sent back as an input to P_1. *A priori*, this mechanism does not take advantage of grid data management systems. Therefore, some precautions need to be taken when considering service-based applications to ensure good application performances.

2.4.7 Data composition strategies

Each service in a data-intensive workflow of services is receiving input data on its input ports. Depending on the desired service semantic, the user might envisage various input composition patterns between the different input ports.

Basic data composition patterns

Although not exhaustive, there are two main data composition patterns very frequently encountered in scientific applications that were first introduced in the Taverna workbench [157]. They are illustrated in figure 2.40.

Let $\mathbf{A} = \{\mathrm{A}_0, \mathrm{A}_1, \ldots, \mathrm{A}_n\}$ and $\mathbf{B} = \{\mathrm{B}_0, \mathrm{B}_1, \ldots, \mathrm{B}_m\}$ be two input data sets. The *one-to-one* composition pattern (left of figure 2.40) is the most common. It consists in processing two input data sets pairwise in their order of arrival. This is the classical case where an algorithm needs to process every pair of input data independently. An example is a matrix addition operator : the sum of each pair of input matrices is computed and returned as a result. We will denote $\oplus$ the one-to-one composition operator. $\mathbf{A} \oplus \mathbf{B} = \{A_1 \oplus B_1, A_2 \oplus B_2, \ldots\}$ denotes the set of all outputs. For simplification, we will denote $A_1 \oplus B_1$ the result of processing the pair of input data (A_1, B_1) by some service. Usually, the two input data sets have the same size ($m = n$)

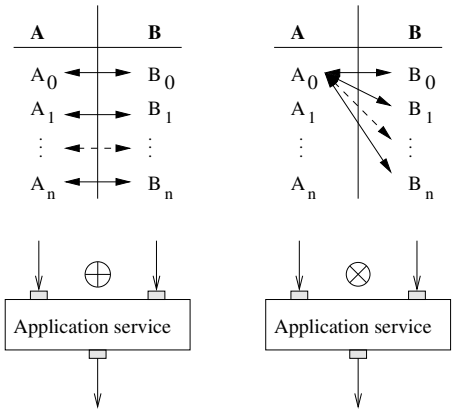


FIG. 2.40 – Action of the *one-to-one* (left) and *all-to-all* (right) operators on the input data sets

when using the one-to-one operator, and the cardinality of the results set is $m = n$. If $m \neq n$, a semantics has to be defined. We will consider that only the $\min(m, n)$ first pieces of data are processed in this case.

The *all-to-all* composition pattern (right of figure 2.40) corresponds to the case where all inputs in one data set need to be processed with all inputs in the other data set. A common example is the case where all pieces of data in the first input set are to be processed with all parameter configurations defined in the second input set. We will denote $\otimes$ the all-to-all composition operator. The cardinality of $\mathbf{A} \otimes \mathbf{B} = \{A_1 \otimes B_1, A_1 \otimes B_2 \ldots A_1 \otimes B_m, A_2 \otimes B_1 \ldots A_2 \otimes B_m \ldots\ldots A_n \otimes B_1 \ldots A_n \otimes B_m\}$ is $m \times n$.

Note that other composition patterns with different semantics could be defined (*e.g. all-to-all-but-one* composition). However, they are more specific and consequently more rarely encountered. Combining the two operators introduced above enable very complex data composition patterns, as will be illustrated below.

Combining data composition patterns

As illustrated at the left of figure 2.41, the pairwise one-to-one and all-to-all operators can be combined to compose data patterns for services with an arbitrary number of input ports. In this case, the priority of these operators needs to be explicitly provided by the user. We are using parenthesis in our figures to display priorities explicitly. If the input data sets are $\mathbf{A} = \{A_0, A_1\}$, $\mathbf{B} = \{B_0, B_1\}$, and $\mathbf{C} = \{C_0, C_1, C_2\}$, the following data would be produced in this case :

$$\mathbf{A} \oplus (\mathbf{B} \otimes \mathbf{C}) = \left\{ \begin{array}{ll} A_0 \oplus (B_0 \otimes C_0), & A_1 \oplus (B_1 \otimes C_0), \\ A_0 \oplus (B_0 \otimes C_1), & A_1 \oplus (B_1 \otimes C_1), \\ A_0 \oplus (B_0 \otimes C_2), & A_1 \oplus (B_1 \otimes C_2) \end{array} \right\}$$

Successive services may also use various combinations of data composition operators as illustrated at the right of figure 2.41. The example given corresponds to a classical situation where an input data set, say two pieces of data $\mathbf{A} = \{A_0, A_1\}$, is processed by a first algorithm (using different parameter configurations, say $\mathbf{P} = \{P_0, P_1, P_2\}$), before being delivered to a second service for processing with a matching number of data, say $\mathbf{B} = \{B_0, B_1\}$. The output data set would be :

$$\mathbf{B} \oplus (\mathbf{A} \otimes \mathbf{P}) = \left\{ \begin{array}{ll} B_0 \oplus (A_0 \otimes P_0), & B_1 \oplus (A_1 \otimes P_0), \\ B_0 \oplus (A_0 \otimes P_1), & B_1 \oplus (A_1 \otimes P_1), \\ B_0 \oplus (A_0 \otimes P_2), & B_1 \oplus (A_1 \otimes P_2) \end{array} \right\} \quad (2.18)$$

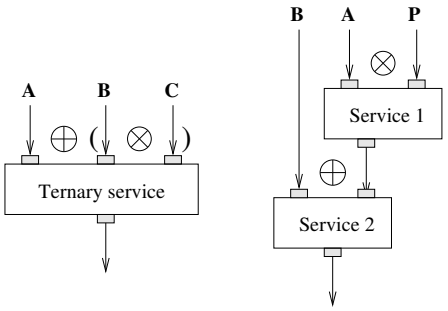


FIG. 2.41 – Combining composition operators : multiple input service (left) and cascade of services (right)

As can be seen, composition operators are a powerful tool for data-intensive application developers who can represent complex data flows in a very compact format. Although the one-to-one operator preserves the input data sets cardinality, the all-to-all operator may lead to drastic increases in the number of data to be processed.

State of the art in data composition

Taverna [157]. The one-to-one and the all-to-all data composition operators were first introduced and implemented in the Taverna workflow manager. They are part of the underlying Scufl workflow description language. In this context, they are known as the *dot product* and *cross product iteration strategies* respectively. The strategy of Taverna for dealing with input sets of different sizes in a one-to-one composition is to produce the $\min(m, n)$ first results only. However, the semantics adopted by Taverna when dealing with a composition of operators as illustrated in figure 2.41 is not straight forward.

In the ternary service (left of figure 2.41), Taverna will produce the

$$\mathbf{A} \oplus_{\text{Taverna}} (\mathbf{B} \otimes \mathbf{C}) = \{ \; A_0 \oplus (B_0 \otimes C_0), \quad A_1 \oplus (B_1 \otimes C_0) \; \}$$

output set. Given that only two input data are available on the first service port, the $\min(m, n)$ truncation rule of the one-to-one (dot product) operator applies. Note that changing the priority of operators will produce a different output. Indeed,

$$(\mathbf{A} \oplus_{\text{Taverna}} \mathbf{B}) \otimes \mathbf{C} = \left\{ \begin{array}{l} \forall i, (A_0 \oplus B_0) \otimes C_i, \\ \forall i, (A_1 \oplus B_1) \otimes C_i \end{array} \right\}$$

Taverna proposes a graphical interface for allowing the user to define the desired priority on the data composition operators.

In the case of the example given in the right of figure 2.41, the priority on the data composition is implicit in the workflow. There is no user control on it. In this case, Taverna will produce :

$$\mathbf{B} \oplus_{\text{Taverna}} (\mathbf{A} \otimes \mathbf{P}) = \{ \; B_0 \oplus (A_0 \otimes P_0), \quad B_1 \oplus (A_1 \otimes P_0) \; \} \tag{2.19}$$

More data will be produced at the output of the Service1 (namely, $A_0 \otimes P_1, A_1 \otimes P_1, A_0 \otimes P_2, A_1 \otimes P_2$) but the truncation semantics of the one-to-one operator will apply in the second service and only two output data segments will be produced. Note that this semantics differs from the one that we consider and that is illustrated in equation 2.18.

Kepler [120] and Triana [195]. The Kepler and the Triana workflow managers only implement the one-to-one composition operator. This operator is implicit for all data composition inside the workflow and it cannot be explicitly specified by the user.

We could implement an all-to-all strategy in Kepler by defining specific actors but this is far from being straight forward. Kepler actors are blocking when reading on empty input ports. The case where two different input data sets have a different size (common in the all-to-all composition operator) is not really taken into account. Similar work can be achieved in Triana using the various *data stream* tools provided. However, in both cases, the all-to-all semantics is not handled at the level of the workflow engine. It needs to be implemented inside the application workflow.

MOTEUR. We designed the MOTEUR workflow engine so that it implements the semantics of the operators defined in section 2.4.7. MOTEUR recognizes both one-to-one and all-to-all operators (it does recognize Scufl workflows) but it uses the algorithm introduced in section 2.4.7 to define the combination semantics.

Data composition algorithm

As can be seen, even considering simple examples such as the ones shown in figure 2.41, the semantics of combining data composition operators is not straight forward. Different workflow engines have different capabilities and implement different combination strategies. Our goal is to define a clear and intuitive semantics for such combinations. We propose an algorithm to implement this data combination strategy.

Taverna provides the most advanced data composition techniques. Yet, we argue that the semantics described in equation 2.19 is not intuitive for the end user. Given that two correlated input data sets **A** and **B**, with the same size, are provided, the user can expect that the data A_i will always be analyzed with the correlated data B_i, regardless of the algorithm parameters P_j considered. We therefore adopt the semantics proposed in equation 2.18 where A_i is consistently combined with B_i.

To formalize and generalize this approach, we need to consider the complete data flows to be processed in the application workflow. In the reminder of this section, we will consider the very general case, common in scientific applications, where the user needs to independently process sets of input data $\mathbf{A}, \mathbf{B}, \mathbf{C} \ldots$ that are divided into *data groups*. A group is a set of input data tuples that defines a relation between data coming from different sets. For instance :

$$\mathrm{G} = \{(A_0, B_0, C_0), (A_1, B_1, C_1), (A_2, B_2, C_2)\}$$
$$\mathrm{H} = \{(A_4, B_0), (A_1, B_2), (A_2, B_5), (A_6, B_6)\}$$

are two groups establishing a relation between 3 data triplets and 4 data pairs respectively. The relations between input data depend on the application and can only be specified by the user. However, we will see that this definition can be explicit (as illustrated above) or implicit, just considering the workflow topology and the order in which input data are delivered by the workflow data sources.

Data composition operator semantics. We consider that the one-to-one composition operator does only make sense when processing related data segments. Therefore, only data connected by a group should be considered for processing by any service. When considering a service directly connected to input data sets, determining relations between data segments is straight forward. However, when considering a complete application workflow such as the one illustrated in figure 2.42, other services (*e.g.* S_4) need to determine which of their input data segments

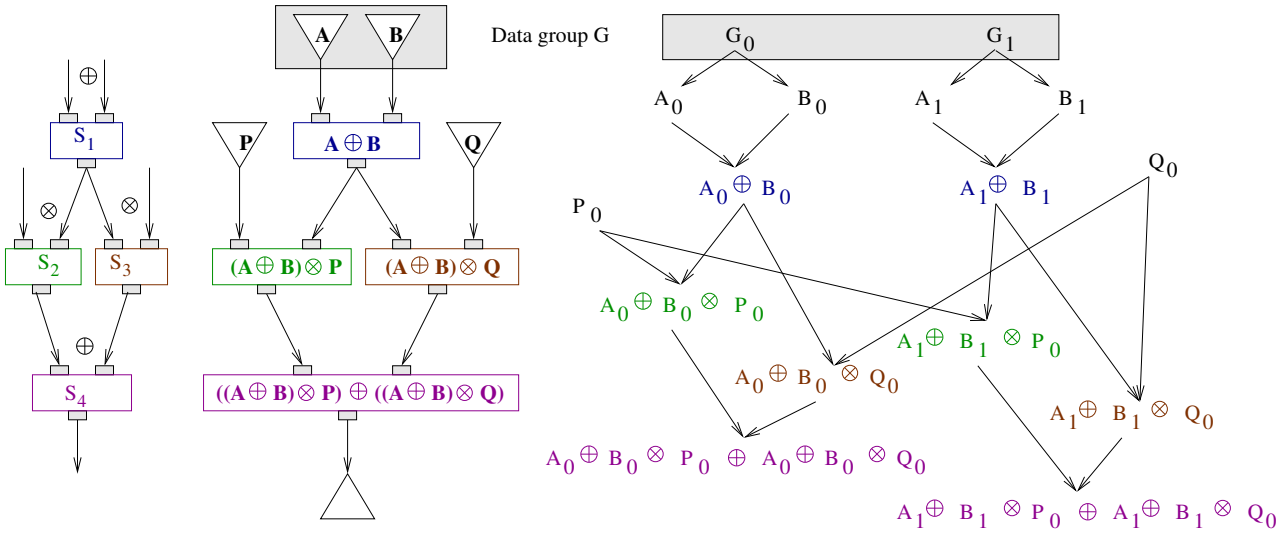


FIG. 2.42 – Workflow example (left), associated data sets directed graph (center), and part of the associated directed acyclic data graph.

are correlated. The one-to-one composition operator does introduce the need for the algorithm described below.

Conversely, the all-to-all operator does not rely on any predetermined relation between input data segments. Any number of inputs can be combined, with very different meaning (such as data to process and algorithm parameters). Each data received as input yields to one or more invocations of the service for processing.

Combination semantics. The left of figure 2.42 represents a sample workflow made of 4 application services and combining the one-to-one and the all-to-all composition operators. In the center of the figure is represented the directed graph of the data sets produced. Given 4 input data sets, $\mathbf{A}, \mathbf{B}, \mathbf{P}$ and $\mathbf{Q}$, the complete workflows produces

$$((\mathbf{A} \oplus \mathbf{B}) \otimes \mathbf{P}) \oplus ((\mathbf{A} \oplus \mathbf{B}) \otimes \mathbf{Q}).$$

as output of the S_4 service. Given the one-to-one operator semantics described above, the data set $\mathbf{A} \oplus \mathbf{B}$ produced by the first service will be non empty if and only if data in $\mathbf{A}$ and $\mathbf{B}$ are related through a group G that is represented at the top of the figure (A_i, the i$^{\text{th}}$ element of $\mathbf{A}$, is correlated with B_i, the i$^{\text{th}}$ element of $\mathbf{B}$).

Considering the service S_4, it is not trivial to determine the content of the output data set, resulting from a one-to-one composition of the two inputs $(\mathbf{A} \oplus \mathbf{B}) \otimes \mathbf{P}$ and $(\mathbf{A} \oplus \mathbf{B}) \otimes \mathbf{Q}$. Intuitively, two input data $(A_i \oplus B_i) \otimes P_k$ and $(A_j \oplus B_j) \otimes Q_l$ should be combined only if $i = j$. Indeed, combining A_i with B_i, or a subsequent processing of these data, does make sense given that the user established a relation between this input pair through the group G. Conversely, there is no relation between $A_i \oplus B_i$ on the one side and any P_k or Q_l that are combined in an all-to-all operation on the other side. Therefore, the processing of $((A_i \oplus B_i) \otimes P_k) \oplus ((A_i \oplus B_i) \otimes Q_l)$ does make sense for all k and all l.

To formalize this approach we need to consider the data production Directed Acyclic Graph that is represented in right of figure 2.42. This graph shows how all pieces of input are combined by the different processings. At the roots of the graph, the *input* data are parents of all *produced*

data. The formal relation between each data pair (A_i, B_i) is represented through a group instantiation G_i, parent of both A_i and B_i. We will name *orphan* data, input data that have no group parent such as P_0 and Q_0. The directed data graph is constructed from the roots (workflow inputs) to the leafs (workflow outputs) by applying the two following simple rules implementing the semantics of the one-to-one and the all-to-all operators respectively :

1. Two data are always combined in an all-to-all operation.
2. Two data (graph nodes) are combined in a one-to-one operation **if and only if** there exists a common ancestor to both data in the data graph.

The interpretation of the first rule is straight forward. The second rule is illustrated by the full data graph displayed at the right of figure 2.42. For instance, the data $A_0 \oplus B_0$ is produced from A_0 and B_0 because there exists a common ancestor G_0 to both A_0 and B_0. Similarly, $((A_0 \oplus B_0) \otimes P_0) \oplus ((A_0 \oplus B_0) \otimes Q_0)$ is computed because $A_0 \oplus B_0$ is a common ancestor to $(A_0 \oplus B_0) \otimes P_0$ and $(A_0 \oplus B_0) \otimes Q_0$. There exists other common ancestors such as A_0, B_0, and G_0 but it is not needed to go back further in the data graph as soon as one of them has been found. Note that in a more complex workflow topologies, the common ancestor does not need to be an immediate parent. It can be easily demonstrated by recurrence that following this rule, two input data sets may be composed one-to-one if and only if there exists a grouping relation between them at the root of the data graph.

Algorithm and implementation. To implement the data composition operators semantic introduced above, MOTEUR dynamically resolves the data combination problem by applying the following algorithm :

1. Build the directed graph of the data sets to be processed.
2. Add data groups to this graph.
3. Initialize the directed acyclic data graph :
   (a) Create root nodes for each group instance G_i and add a child node for each related data.
   (b) Create root nodes for each orphan data.
4. Start the execution of the workflow.
5. For each tuple of data to be processed :
   (a) Update the data graph by applying the two rules corresponding to the one-to-one and the all-to-all operators.
   (b) Loop until there are no more data available for processing in the workflow graph.

To implement this strategy, MOTEUR needs to keep representations of :
– the topology of the services workflow ;
– the graph of data ;
– the list of input data that have been processed by each service.

Indeed, the data graph is dynamically updated during the execution. When a new data is produced, its combination with all previously produced data is studied. In particular in an all-to-all composition pattern, a new input data needs to be combined with all previously computed data. It potentially triggers several services invocation. The history of previous computations is thus needed to determine the exhaustive list of data to produce.

The graphs of data also ensures a full traceability of the data processed by the workflow manager : for each data node, the parents and children of the data can be determined. Besides, it provides a mean to unambiguously identify each data produced. This becomes mandatory when considering parallel execution of the workflow as discussed in section 2.4.8.

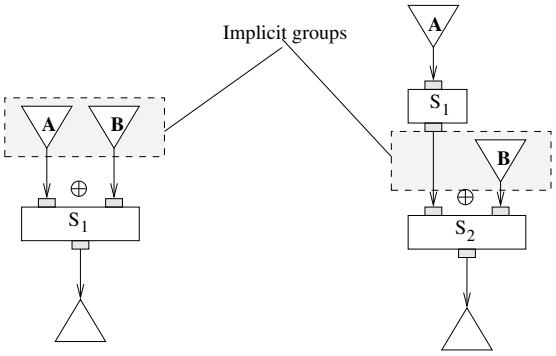


FIG. 2.43 – Implicit groups definition.

Implicit combinations. The algorithm proposed aims at providing a strict semantics to the combination of data composition operators, while providing intuitive data manipulation for the users. Data groups have been introduced to clarify the semantics of the one-to-one operator. However, it is very common that users are writing workflows without explicitly specifying pair-wise relations between the data. The order in which data are declared or send to the workflow inputs are rather used as an implicit relation.

To ease the workflow generation from the user point of view, groups can be implicitly generated when they are not explicitly specified by the user. Figure 2.43 illustrates two different cases. On the left side, the reason for generating an implicit group is straight forward : two input data sets are being processed through a one-to-one service. But there may be more indirect cases such as the one illustrated on the right side of the figure. The systematic rule that can be applied is to create an implicit group for each *one-to-one* operator whose input data are orphans. For example, in the case illustrated in left of figure 2.43, the input data sets **A** and **B** are orphans and bound *one-to-one* by the service S_1. An implicit group is therefore created between **A** and **B**. In the case illustrated in the right side of figure 2.43, the implicit group will be created between the two inputs of service S_2. There will therefore be an implicit grouping relation between each output of the first service $S_1(A_i)$ and B_i.

The implicit groups are created statically by analyzing the workflow topology and the input data sets before starting the execution of the workflow.

Coping with data fragments. So far, we have only considered the case were the number of outputs of a service matches the number of inputs. In some cases though, an application service will split input data in smaller fragments, either for dealing with smaller data sets (*e.g.* a 3D medical image is split in a stack of 2D slices) or because the service code function implies that it produces several outputs for each input. The workflow displayed in figure 2.44 illustrates such a situation. The service S_1 is splitting each input data (*e.g.* A_0) in several fragments (A_0^0, A_0^1 and A_0^2).

In the example given in figure 2.44, it is expected that service S_2 will receive the same number of data on both input ports (one-to-one composition operator). However, there is no way for the user to specify an explicit grouping between two data sets. Grouping the data sets **A** with **B** would only create a relation between A_0 and B_0. Therefore, the fragments A_0^0, A_0^1 and A_0^2, children of A_0, would all be related to B_0 and the service S_2 would produce

$$\mathbf{A} \oplus \mathbf{B} = \{A_0^0 \oplus B_0, A_0^1 \oplus B_0, A_0^2 \oplus B_0\}.$$

Instead, the implicit grouping strategy will group $S_1(A_0)$ outputs with **B**. Consequently, the grouping will result in the data graph shown in right of figure 2.44 and the output produced

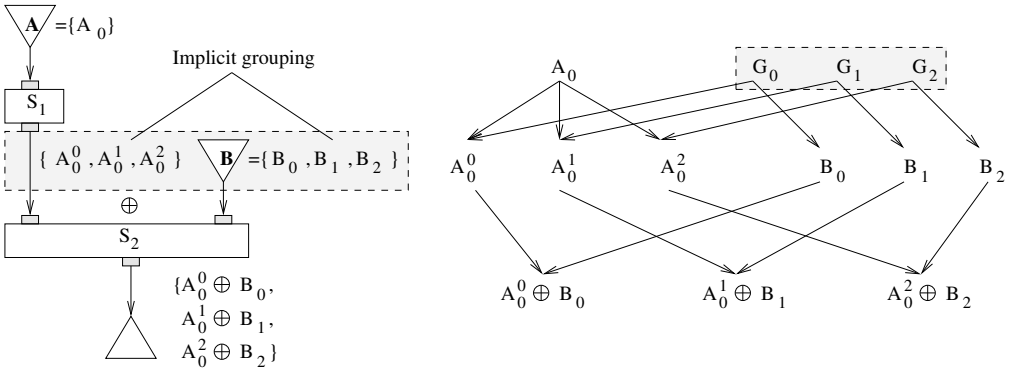


FIG. 2.44 – Implicit groups relating data fragments (A_0^0, A_0^1, A_0^2) and input **B**.

will be

$$\mathbf{A} \oplus \mathbf{B} = \{A_0^0 \oplus B_0, A_0^1 \oplus B_1, A_0^2 \oplus B_2\}$$

as expected. Note that the number of inputs to service S_2 needs to be consistent in this case.

2.4.8 Scheduling and executing workflows of services

The service-based approach is making services composition easier than the task-based approach as discussed in section 2.4.6. It is thus highly convenient from the end user point of view. However, in this approach, the control of jobs submissions is delegated to external services, making the optimization of the workflow execution much more difficult. The services are black boxes isolating the workflow manager from the execution infrastructure. In this context, most known optimization solutions do not hold.

Related work

Many solutions have indeed been proposed in the task-based paradigm to optimize the scheduling of an application in distributed environments [29]. Concerning workflow-based applications, previous works [19] propose specific heuristics to optimize the resource allocation of a complete workflow. Even if it provides remarkable results, this kind of solutions is not directly applicable to the service-based approach. Indeed, in this latest approach, the workflow manager is not responsible for the task submission and thus cannot optimize the resource allocation.

Focusing on the service-based approach, nice developments such as the DIET middleware [28] and comparable approaches [193, 8] introduce specific strategies such as hierarchical scheduling. In [26] for instance, the authors describe a way to handle file persistence in distributed environments, which leads to strong performance improvements. However, those works focus on middleware design and do not include any workflow management yet. Moreover, those solutions require specific middleware components to be deployed on the target infrastructure. As far as we know, such a deployment has only been done on experimental platforms yet [25], and it is hardly possible for an application running on a production infrastructure.

Hence, there is a strong need for precisely identifying generic optimization solutions that apply to service-based workflows. In the following sections, we are exploring different strategies for optimizing the workflow execution in a service-based approach, thus offering the flexibility of services and the efficiency of tasks. First of all, several level of parallelism can be exploited when considering the workflow execution for taking advantage of the grid computing capabilities. We describe them and then study their impact on the performances with respect to the characteristics of the considered application. Besides, we propose a solution for grouping sequential

jobs in the workflow, thus allowing more elaborated optimization strategies in the service-based workflow area.

Enabling parallelism on grids

Asynchronous services calls. To enable parallelism during the workflow execution, multiple application services have to be called concurrently. The calls made from the workflow enactor to these services need to be non-blocking for exploiting the potential parallelism. GridRPC services may be called asynchronously as defined in the standard [155]. Web Services also theoretically enable asynchronous calls. However, the vast majority of existing web service implementations do not cover the whole standard and none of the major implementations [197, 101] do provide any asynchronous service calls for now. As a consequence, asynchronous calls to web services need to be implemented at the workflow enactor level, by spawning independent system threads for each processor being executed.

Workflow parallelism. Given that asynchronous calls are possible, the first level of parallelism that can be exploited is the intrinsic workflow parallelism depending on the graph topology. For instance if we consider the simple example presented in figure 2.39, processors P_2 and P_3 may be executed in parallel. This optimization is trivial and implemented in all the workflow managers cited above.

Data parallelism. When considering data-intensive applications, several input data sets are to be processed using a given workflow. Benefiting from the large number of resources available in a grid, workflow services can be instantiated as several computing tasks running on different hardware resources and processing different input data in parallel. *Data parallelism* is achievable when a service is able to process several data sets simultaneously with a minimal performance loss. This capability involves the processing of independent data on different computing resources.

Enabling data parallelism implies, on the one hand, that the services are able to process many parallel connections and, on the other hand, that the workflow engine is able to submit several simultaneous queries to a service leading to the dynamic creation of several threads. Moreover, a data parallel workflow engine should implement a dedicated data management system. Indeed, in case of a data parallel execution, a data is able to overtake another one during the processing and this could lead to a causality problem, as we exemplified in [80]. To properly tackle this problem, data provenance has to be monitored during the data parallel execution. Detailed work on data provenance can be found in [208].

Consider the simple workflow made of 3 services and represented on top of figure 2.39. Suppose that we want to execute this workflow on 3 independent input data sets D_0, D_1 and D_2. The data parallel execution diagram of this workflow is represented on figure 2.45. On this kind of diagram, the abscissa axis represents time. When a data set D_i appears on a row corresponding to a processor P_j, it means that D_i is being processed by P_j at the current time. To facilitate legibility, we represented with the D_i notation the piece of data resulting from the processing of the initial input data set D_i all along the workflow. For example, in the diagram of figure 2.45, it is implicit that on the P_2 service row, D_0 actually denotes the data resulting from the processing of the input data set D_0 by P_1. Moreover, on those diagrams we made the assumption that the processing time of every data set by every service is constant, thus leading to cells of equal widths. Data parallelism occurs when different data sets appear on a single square of the diagram whereas intrinsic workflow parallelism occurs when the same data set appears many times on different cells of the same column. Crosses represent idle cycles.

P_3	X	D_0 D_1 D_2
P_2	X	D_0 D_1 D_2
P_1	D_0 D_1 D_2	X

FIG. 2.45 – Data parallel execution diagram of the workflow of figure 2.39

P_3	X	D_0	D_1	D_2
P_2	X	D_0	D_1	D_2
P_1	D_0	D_1	D_2	X

FIG. 2.46 – Service parallel execution diagram of the workflow of figure 2.39

As demonstrated in the next sections, fully taking into account this level of parallelism is critical in service-based workflows, whereas it does not make any sense in task-based ones. Indeed, in this case it is covered by the workflow parallelism because each task is explicitly described in the workflow description.

Services parallelism. Input data sets are likely to be independent from each other. This is for example the case when a single workflow is iterated in parallel on many input data sets. *Services parallelism* is achievable when the processing of two different data sets by two different services are totally independent. This pipelining model, very successfully exploited inside CPUs, can be adapted to sequential parts of service-based workflows. Consider again the simple workflow represented in figure 2.39, to be executed on the 3 independent input data sets D_0, D_1 and D_2. Figure 2.46 presents a service parallel execution diagram of this workflow. Service parallelism occurs when different data sets appear on different cells of the same column. We here supposed that a given service can only process a single data set at a given time (data parallelism is disabled).

When enabling exploiting both data and services parallelism, the gain of services parallelism only becomes visible if the time needed to process all data segments is varying. Figure 2.47 illustrates on a simple example what happens in the case where the processing time of some data set D_0 is twice as long as the other ones on service P_0 and the processing time of the data set D_1 is three times as long as the other ones on service P_1. On a grid, it can for example occur if D_0 has been submitted twice because an error occurred and if D_1 remained blocked in a waiting queue. The left diagram does not take into account service parallelism whereas the right one does. In this case, service parallelism improves performances beyond data parallelism as it enables some computations overlap.

Data synchronization barriers, presented in section 2.4.6, are of course a limitation to services parallelism. In this case, this level of parallelism cannot be exploited because the input data sets are dependent from each other.

Here again, we show in the next section that service parallelism is of major importance to optimize the execution of service-based workflows. In task-based workflow, this level of parallelism does not make any sense because it is included in the workflow parallelism.

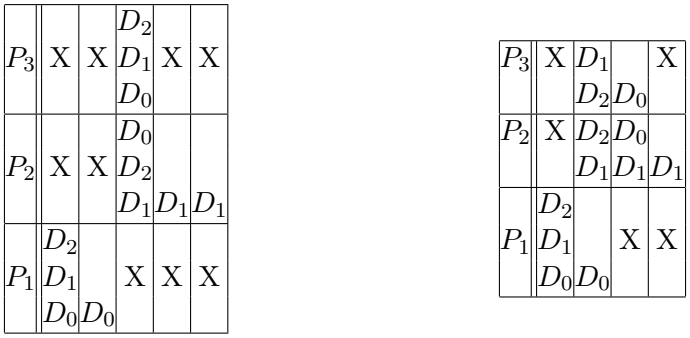


FIG. 2.47 – Workflow execution time without (left) and with (right) service parallelism when the execution time is not constant.

Theoretical performance analysis

The data and service parallelism described above are specific to the service-based workflow approach. To precisely quantify how they influence the application performances we model the workflow execution time for different configurations. We first present general results and then study particular cases, making assumptions on the type of application run.

Definitions and notations.

- In the workflow, a *path* denotes a set of processors linking an input to an output. The *critical path* of the workflow denotes the longest path in terms of execution time.
- n_W denotes the number of services on the critical path of the workflow and n_D denotes the number of data sets to be executed by the workflow.
- i denotes the index of the i^{th} service of the critical path of the workflow ($i \in [0, n_W - 1]$). Similarly j denotes the index of the j^{th} data set to be executed by the workflow ($j \in [0, n_D - 1]$).
- $T_{i,j}$ denotes the duration in seconds of the treatment of the data set j by the service i. If the service submits jobs to a grid infrastructure, this duration includes the overhead introduced by the submission, scheduling and queuing times.
- $\sigma_{i,j}$ denotes the absolute time in seconds of the end of the treatment of the data set j by the service i. The execution of the workflow is assumed to begin at $t = 0$. Thus $\sigma_{0,0} = T_{0,0} > 0$.
- Σ denotes the total execution time of the workflow

$$\Sigma = \max_{j<n_D}(\sigma_{n_W-1,j}) \tag{2.20}$$

Hypotheses. The critical path is assumed not to depend on the data set. This hypothesis seems reasonable for most applications but may not hold in some cases as for example the one of workflows including algorithms containing optimization loops whose convergence time is likely to vary in a complex way with regards to the nature of the input data set.

Data parallelism is assumed not to be limited by infrastructure constraints. We justify this hypothesis considering that our target infrastructure is a grid, whose computing power is sufficient for our application.

In this section, workflows are assumed not to contain any synchronization processors. Workflows containing such synchronization barriers may be analyzed as two sub workflows respectively corresponding to the parts of the initial workflow preceding and succeeding the synchronization barrier.

Execution times modeling. Under those hypotheses, we can determine the expression of the total execution time of the workflow for different execution policies :

– Sequential case (without service nor data parallelism) :

$$\Sigma = \sum_{i<n_W} \sum_{j<n_D} T_{i,j} \tag{2.21}$$

– Case DP : Data parallelism only

$$\Sigma_{DP} = \sum_{i<n_W} \max_{j<n_D} \{T_{i,j}\} \tag{2.22}$$

– Case SP : Service parallelism only

$$\begin{aligned} \Sigma_{SP} &= T_{n_W-1,n_D-1} + m_{n_W-1,n_D-1} \\ &\text{with :} \quad \forall i \neq 0 \quad \text{and} \quad \forall j \neq 0, \\ m_{i,j} &= \max(T_{i-1,j} + m_{i-1,j}, T_{i,j-1} + m_{i,j-1}) \\ &\text{and :} \\ m_{0,j} &= \sum_{k<j} T_{0,k} \quad \text{and} \quad m_{i,0} = \sum_{k<i} T_{k,0} \end{aligned} \tag{2.23}$$

– Case DSP : both Data and Service parallelism

$$\Sigma_{DSP} = \max_{j<n_D} \left\{ \sum_{i<n_W} T_{i,j} \right\} \tag{2.24}$$

All the above expressions of the execution times can be demonstrated recursively [79].

Asymptotic speed-ups. To better understand the properties of each kind of parallelism, it is interesting to study the asymptotic speed-ups resulting from service and data parallelism in particular application cases.

– **Massively data-parallel workflows**. Let us consider a massively (*embarrassingly*) data-parallel application (single processor P_0, very large number of input data). In this case, $n_W = 1$ and the execution time is :

$$\Sigma_{DP} = \Sigma_{DSP} = \max_{j<n_D}(T_{0,j}) \quad \ll \quad \Sigma = \Sigma_{SP} = \sum_{j<n_D} T_{0,j}$$

In this case, data parallelism leads to a significant speed-up. Service parallelism is useless but it does not lead to any overhead.

– **Non data intensive workflows**. In such workflows, $n_D = 1$ and the execution time is :

$$\Sigma_{DSP} = \Sigma_{DP} = \Sigma_{SP} = \Sigma \quad = \quad \sum_{i<n_W} T_{i,0}$$

In this case, neither data nor service parallelism lead to any speed-up. Nevertheless, none of them does introduce any overhead.

– **Data intensive complex workflows**. In this case, we will suppose that $n_W > 1$ and $n_D > 1$. In order to analyze the speed-ups introduced by service and data parallelism, we make the simplifying assumption of constant execution times : $T_{i,j} = T$. The workflow execution time then resumes to :

$$\begin{aligned} \Sigma &= n_D \times n_W \times T \\ \Sigma_{DP} = \Sigma_{DSP} &= n_W \times T \\ \Sigma_{SP} &= (n_D + n_W - 1) \times T \end{aligned}$$

If service parallelism is disabled, the speed-up introduced by data parallelism is :

$$S_{DP} = \frac{\Sigma}{\Sigma_{DP}} = n_D$$

If service parallelism is enabled, the speed-up introduced by data parallelism is :

$$S_{DSP} = \frac{\Sigma_{SP}}{\Sigma_{DSP}} = \frac{n_D + n_W - 1}{n_W}$$

If data parallelism is disabled, the speed-up induced by service parallelism is :

$$S_{SP} = \frac{\Sigma}{\Sigma_{SP}} = \frac{n_D \times n_W}{n_D + n_W - 1}$$

Theoretically, service parallelism does not lead to any speed-up if it is coupled with data parallelism : $S_{SDP} = \frac{\Sigma_{DP}}{\Sigma_{DSP}} = 1$. Thus, under those assumptions, service parallelism may not be of any use on fully distributed systems. However, section 2.4.12 will show that even in case of homogeneous input data sets, T is hardly constant in production systems due to the high variability of the overhead coming from submission, scheduling and queuing times on such large scale and multiuser platforms. The constant execution time hypothesis does not hold. This appears to be a significant difference between grid computing and traditional cluster computing. Figure 2.47 illustrates on a simple example why service parallelism do provide a speed-up even if data parallelism is enabled. It explains the experimental observations done in section 2.4.12.

State of the art in service-based workflow managers

Workflow parallelism is usually implemented in existing workflow managers.

Taverna implements data parallelism (known as *multiple threads* in this context). However, data parallelism is limited to a fixed number of threads specified in the Scufl workflow description language. It cannot dynamically adapt to the size of data sets to be processed. Service parallelism is not supported yet but this feature has been proposed for the next major release of the engine (version 2).

Kepler implements services parallelism through the *Physical Network* (PN) director. There is no data parallelism in Kepler.

Triana does not implement service nor data parallelism.

The GEMLCA/PGRADE workflow manager can exploit the three levels of parallelism reported above and submit computing tasks to different grids.

MOTEUR was designed to optimize the performance of data-intensive applications on grids by implementing the three level of parallelism in addition to the job grouping optimization presented below.

2.4.9 Legacy code wrapping

To ease the embedding of legacy-codes in the service-based framework, an application-independent job submission service is required. In this section, we briefly review systems that are used to wrap legacy code into services to be embedded in service-based workflows.

The Java Native Interface (JNI) has been widely adopted for the wrapping of legacy codes into services. Wrappers have been developed to automate this process. In [99], an automatic JNI-based wrapper of C code into Java and the corresponding type mapper with Triana [195] is presented : JACAW generates all the necessary java and C files from a C header file and compiles them. A coupled tool, MEDLI, then maps the types of the obtained Java native

method to Triana types, thus enabling the use of the legacy code into this workflow manager. Related to the ICENI workflow manager [71], the wrapper presented in [113] is based on code re-engineering. It identifies distinct components from a code analysis, wrap them using JNI and adds a specific CXML interface layer to be plugged into an ICENI workflow.

The WSPeer framework [91], interfaced with Triana, aims at easing the deployment of Web-Services by exposing many of them at a single endpoint. It differs from a container approach by giving to the application the control over service invocation. The Soaplab system [186] is especially dedicated to the wrapping of command-line tools into Web-Services. It has been widely used to integrate bioinformatics executables in workflows with Taverna [157]. It is able to deploy a Web-Service in a container, starting from the description of a command-line tool. This command-line description, referred to as the metadata of the analysis, is written for each application using the ACD text format file and then converted into a corresponding XML format. Among domain specific descriptions, the authors underline that such a command-line description format must include (i) the description of the executable, (ii) the names and types of the input data and parameters and (iii) the names and types of the resulting output data. As described latter, the format we used includes those features and adds new ones to cope with requirements of the execution of legacy code on grids.

The GEMLCA environment [46] addresses the problem of exposing legacy code command-line programs as Grid services. It is interfaced with the P-GRADE portal workflow manager [103]. The command-line tool is described with the LCID (Legacy Code Interface Description) format which contains (i) a description of the executable, (ii) the name and binary file of the legacy code to execute and (iii) the name, nature (input or output), order, mandatory, file or command line, fixed and regular expressions to be used as input validation. A GEMLCA service depends on a set of target resources where the code is going to be executed. Architectures to provide resource brokering and service migration at execution time are presented in [106].

Apart from this latest early work, all of the reviewed existing wrappers are static : the legacy code wrapping is done offline, before the execution. This is hardly compatible with our approach, which aims at optimizing the whole application execution at run time. We thus developed a specific grid submission Web-Service, which can wrap any executable at run time, thus enabling the use of optimization strategies by the workflow manager.

This section introduces a generic application code wrapper compliant with the Web Services specification. It enables the execution of any legacy executable through a standard services interface. The subsequent section 2.4.10 proposes a code execution optimization strategy that can be implemented thanks to this generic wrapper. Finally, section 2.4.11 proposes a service oriented architecture of the system, based on a service factory.

Generic web service wrapper

We developed a specific grid submission Web Service. This service is generic in the sense that it is unique and it does not depend on the executable code to submit. It exposes a standard interface that can be used by any Web Service compliant client to invoke the execution. It completely hides the grid infrastructure from the end user as it takes care of the interaction with the grid middleware. This interface plays the same role as the ACD and LCID files quoted above, except that it is interpreted at the execution time.

To accommodate to any executable, the generic service is taking two different inputs : a descriptor of the legacy executable command line format, and the input parameters and data of this executable. The production of the legacy code descriptor is the only extra work required from the application developer. It is a simple XML file which describes the legacy executable location, command line parameters, input and output data.

Legacy code descriptor

The command line description has to be complete enough to allow dynamic composition of the command line from the list of parameters at the service invocation time and to access the executable and input data files. As a consequence, the executable descriptor contains :

1. The name and access method of the executable. In our current implementation, access methods can be a URL, a Grid File Name (GFN) or a local file name. The wrapper is responsible for fetching the data according to different access modes.
2. The access method and command-line option of the input data. As our approach is service-based, the actual name of the input data files is not mandatory in the description. Those values will be defined at the execution time. This feature differs from various job description languages used in the task-based middlewares. The command-line option allows the service to dynamically build the actual command-line at the execution time.
3. The command-line option of the input parameters : parameters are values of the command-line that are not files and therefore which do not have any access method.
4. The access method and command-line option of the output data. This information enables the service to register the output data in a suitable place after the execution. Here again, in a service-based approach, names of output data files cannot be statically determined because output file names are only generated at execution time.
5. The name and access method of the sandboxed files. Sandboxed files are external files such as dynamic libraries or scripts that may be needed for the execution although they do not appear on the command-line.

Example

An example of a legacy code description file is presented in figure 2.48. It corresponds to the description of the service `crestLines` of the workflow depicted in figure 2.53. It describes the script `CrestLines.pl` which is available from the server `legacy.code.fr` and takes 3 input arguments : 2 files (options `-im1` and `-im2` of the command-line) that are already registered on the grid as GFNs at execution time and 1 parameter (option `-s` of the command-line). It produces 2 files that will be registered on the grid. It also requires 3 sandboxed files that are available from the server.

Discussion

This generic service highly simplifies application development because it is able to wrap any legacy code with a minimal effort. The application developer only needs to write the executable descriptor for her code to become service aware.

But its main advantage is in enabling the sequential services grouping optimization introduced in section 2.4.10. Indeed, as the workflow enactor has access to the executable descriptors, it is able to dynamically create a virtual service, composing the command lines of the codes to be invoked, and submitting a single job corresponding to this sequence of command lines invocation.

It is important to notice that our solution remains compatible with the services standards. The workflow can still be executed by other enactors, as we did not introduce any new invocation method. Those enactors will make standard service calls (e.g. SOAP ones) to our generic wrapping service. However, the optimization strategy described in the next section is only applicable to services including the descriptor introduced above. We call those services MOTEUR services.

```
<description>
  <executable name="CrestLines.pl">
    <access type="URL">
      <path value="http://legacy.code.fr"/>
    </access>
    <value value="CrestLines.pl"/>
    <input name="floating_image" option="-im1">
      <access type="GFN"/>
    </input>
    <input name="reference_image" option="-im2">
      <access type="GFN"/>
    </input>
    <input name="scale" option="-s"/>
    <output name="crest_reference" option="-c1">
      <access type="GFN"/>
    </output>
    <output name="crest_floating" option="-c2">
      <access type="GFN"/>
    </output>
    <sandbox name="convert8bits">
      <access type="URL">
        <path value="http://legacy.code.fr"/>
      </access>
      <value value="Convert8bits.pl"/>
    </sandbox>
    <sandbox name="copy">
      <access type="URL">
        <path value="http://legacy.code.fr"/>
      </access>
      <value value="copy"/>
    </sandbox>
    <sandbox name="cmatch">
      <access type="URL">
        <path value="http://legacy.code.fr"/>
      </access>
      <value value="cmatch"/>
    </sandbox>
  </executable>
</description>
```

FIG. 2.48 – Descriptor example

2.4.10 Services grouping optimization strategy

We propose a services grouping strategy to further optimize the execution time of a workflow. Services grouping consists in merging multiple jobs into a single one. It reduces the grid overhead induced by the submission, scheduling, queuing and data transfers times whereas it may also reduce the parallelism. In particular sequential processors grouping is interesting because those processors do not benefit from any parallelism. For example, considering the workflow of the *Bronze Standard* application presented on figure 2.53 we can, for each data set, group the execution of the `crestLines` and the `crestMatch` jobs on the one hand and the `PFMatchICP` and the `PFRegister` ones on the other hand.

Grouping jobs in the task-based approach is straightforward and it has already been proposed for optimization [19]. Conversely, jobs grouping in the service-based approach is usually not possible given that (i) the services composing the workflow are totally independent from each other (each service is providing a different data transfer and job submission procedure) and (ii) the grid infrastructure handling the jobs does not have any information concerning the workflow and the job dependencies. Consider the simple workflow represented on the left side of figure 2.49. On top, the services for P_1 and P_2 are invoked independently. Data transfers are handled by each service and the connection between the output of P_1 and the input of P_2 is handled at the workflow engine level. On the bottom, P_1 and P_2 are grouped in a virtual single

service. This service is capable of invoking the code embedded in both services sequentially, thus resolving the data transfer and independent code invocation issues.

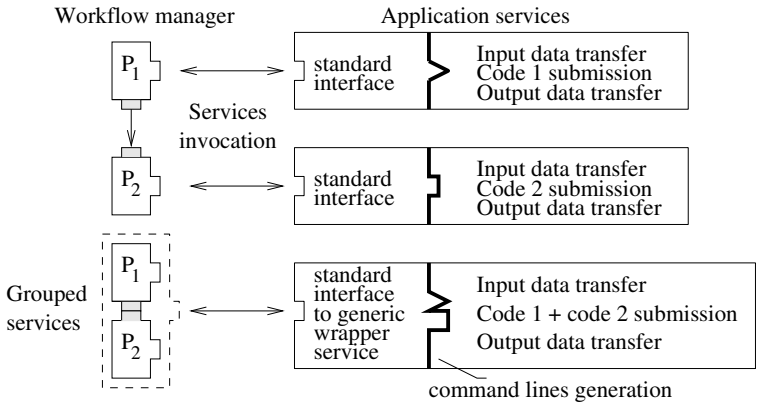


FIG. 2.49 – Classical services invocation (top) and services grouping (bottom).

Grouping strategy

Services grouping can lead to significant speed-ups, especially on production grids, as it is demonstrated below. However, it may also slow down the execution by limiting parallelism. We thus have to determine efficient strategies to group services.

In order to determine a grouping strategy that does not introduce any overhead, neither from the user point of view, nor from the infrastructure one, we impose the two following constraints : (i) the grouping strategy must not limit any kind of parallelism (user point of view) and (ii) during their execution, jobs cannot communicate with the workflow manager (infrastructure point of view). The second constraint prevents a job from holding a resource just waiting for one of its ancestor to complete. An implication of this constraint is that if services A and B are grouped together, the results produced by A will only be available once B will have completed.

A workflow may include both MOTEUR Web-Services (*i.e.* services that are able to be grouped) and classical ones, that could not be grouped. Assuming those two constraints, the following rule is sufficient to process all the possible groupings of two services of the workflow :

> Let A be a MOTEUR service of the workflow and $\{B_0,...B_n\}$ its children in the service graph. **If** there exists a MOTEUR child B_i which is an ancestor of every B_j $(i \neq j)$ and whose each ancestor C is an ancestor of A or A itself, **then** group A and B_i.

Indeed, every violation of this rule also violates one of our constraints as it can easily be shown. The grouping strategy tests this rule for each MOTEUR service A of the workflow. Groups of more than two services may be recursively composed from successive matches of the grouping rule.

For example, the workflow displayed in figure 2.50, extracted from our medical imaging application, is made of 4 MOTEUR services that can be grouped into a single one through 3 applications of the grouping rule. On this figure, notations nearby the services corresponds to the ones introduced in the grouping rule.

The first application case of the grouping rule is represented on the left of the figure. The tested MOTEUR service A is the `crestLines` algorithm. A is connected to the workflow inputs and it has two children, B_0 and B_1. B_0 is a father of B_1 and it only has as single ancestor which is A. The rule thus matches : A and B_0 can be grouped. If there were a service C ancestor of

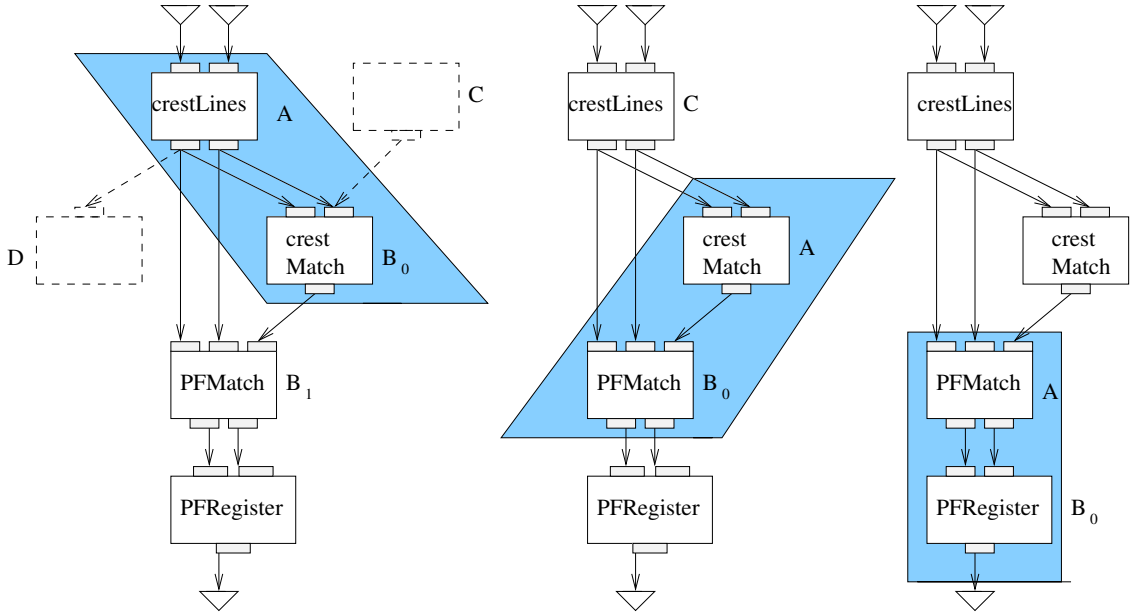


FIG. 2.50 – Services grouping examples

B_0 but not of A as represented on the figure, the rules would be violated : A and C can be executed in parallel before starting B_0. Similarly, if there were a service D the rule would be violated as the workflow manager would need to communicate results during the execution of the grouped jobs.

In the second application case, in the middle of the figure, the tested service A is now `crestMatch`. A has only a single child : B_0. B_0 has two ancestors, A and C. The rule matches because C is an ancestor of A. A and B_0 can then be grouped.

For the last rule application case, on the right of figure 2.50, A is the `PFMatch` service. It has only one child, B_0, who only has a single ancestor, A. The rule matches and those services can thus be grouped.

When A is the `PFRegister` service, the grouping rule does not match because it does not have any child. Note that in this example, the recursive grouping strategy will lead to a single job call.

2.4.11 Dynamic generic service factory

The generic web service drastically simplifies the wrapping of legacy code into application services. However, it is mixing two different roles : (i) the legacy command line generation and (ii) the grid submission. Job submission is only dependent on the target grid and not on the application service itself. In a Service Oriented Architecture (SOA) it is preferable to split these two roles into two independent services for several reasons. First, the submission code does not need to be replicated in all application services. Second, the submission role can be transparently and dynamically changed (to submit to a different infrastructure) or updated (to adapt to middleware evolutions). In addition, an application wrapper factory service further facilitates the wrapping of legacy code services and their grouping. We thus introduce a complete SOA design based on three main services as illustrated in figure 2.52.

The (blue) MOTEUR web services represent legacy code wrapping services. They are assembling command lines and invoking the (red) submission service for handling code execution on the grid infrastructure. The code wrapper factory service is responsible for dynamically ge-

```
<?xml version="1.0" encoding="utf-8" ?>
<definitions ...>
 <types>
  <schema>
   <element name="CrestLines-request">
     <complexType>
      <sequence>
       <element name="floating_image"
                type="string"... />
       <element name="reference_image"
                type="string"... />
       <element name="scale" type="string"... />
      </sequence>
     </complexType>
    </element>
    <element name="CrestLines-response">
      <complexType>
        <sequence>
          <element name="crest_reference"
                   type="string"... />
          <element name="crest_floating"
                   type="string"... />
        </sequence>
      </complexType>
    </element>
  </schema>
 </types>
 <message name="ExecuteSoapIn">
  <part name="parameters"
        element="CrestLines.pl-request" />
 </message>
 <message name="ExecuteSoapOut">
  <part name="parameters"
        element="CrestLines.pl-response" />
 </message>
 <portType name="CrestLines.plSoap">
  <operation name="Execute">
   <input message="ExecuteSoapIn" />
   <output message="ExecuteSoapOut" />
  </operation>
 </portType>
 <binding ...>
  <soap:binding transport="http://..." />
   <operation name="Execute">
    <soap:operation soapAction="http://.../Execute"
          style="document" />
     <MOTEUR-descriptor xmlns="urn:...">
      <location>http://...</location>
     </MOTEUR-descriptor>
     ....
   </operation>
 </binding>
</definitions>
```

FIG. 2.51 – WSDL generated by the factory

nerating and deploying application services. The aim of this factory is to achieve two antagonist goals :

- To expose legacy codes as autonomous web services respecting the main principles of Service Oriented Architectures.
- To enable the grouping of two of these web services in as a unique one for optimizing the execution.

On one hand, the specific web service implementation details (*i.e.* the execution of legacy code on a grid infrastructure) are hidden to the consumer. On the other hand, when the consu-

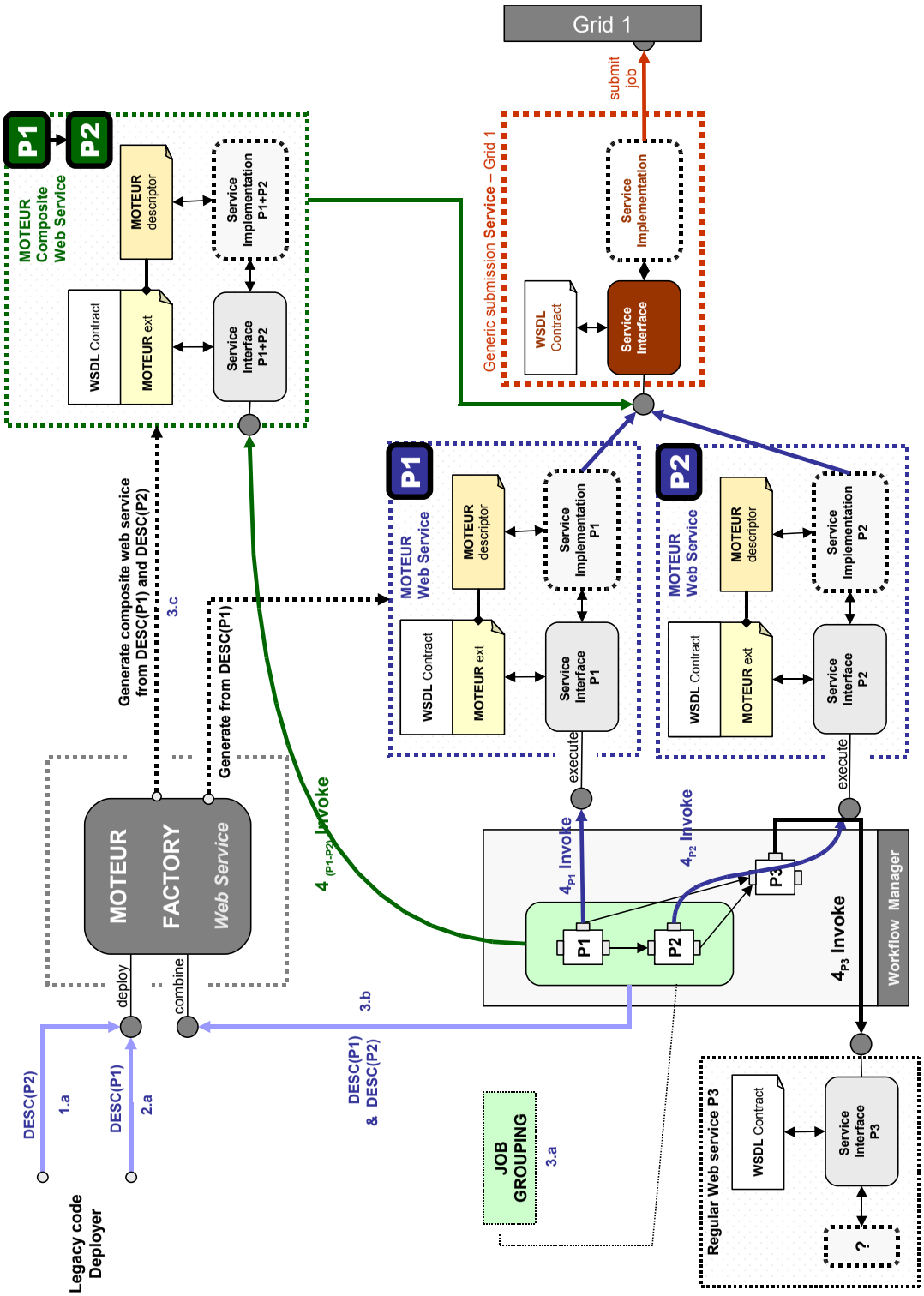


FIG. 2.52 – Services factory.

mer is a workflow manager which can group jobs, it needs to be aware of the real nature the web service (the encapsulation of a MOTEUR descriptor) so that it could merge them at run time. We choose to use the WSDL XML Format extension mechanism which allows to insert user defined XML elements in the WSDL content itself. On figure 2.52, we represent the overall architecture and some usage scenario. First, the legacy code provider submits (**1.a**) a MOTEUR XML descriptor P1 to the MOTEUR factory. The factory, then dynamically deploy (**1.b**) a web service which wraps the submission of the legacy code to the grid via the generic service wrapper. Another provider do the same with the descriptor of P2 (**2.a**). The resulting web services expose their WSDL contracts to the external world with a specific extension associated with the WSDL operation. For instance, the WSDL contract resulting of the deployment of the `crestLines` legacy code described on figure 2.48 is printed on figure 2.51.

This WSDL document defines two types (`CrestLines-request` and `CrestLines-response`) corresponding to the descriptor inputs and outputs and a single `Execute` operation. Notice that in the binding section, the WSDL document contains an extra `MOTEUR-descriptor` tag pointing to the URL of the legacy code descriptor file (`location`) and a binding to the Execute operation (`soap:operation`).

Suppose now that the workflow manager identifies a services grouping optimization (*e.g.* P_1 and P_2) (**3.a** on figure 2.52). Because of its ability to discover the extended nature of these two services, the engine can retrieve the two corresponding MOTEUR descriptors. It can ask the factory to *combine* them (**3.b**) resulting in a single composite web service (**3.c**) which exposes an operation taking its inputs from P_1 (and P_2 inputs coming from other external services) and returning the outputs defined by P_2 (and P_1 outputs going to other external services). This composite web service is of the same type than any regular legacy code wrapping service. It is accessible through the same interface and it also delegates the grid submission to the generic submission web service by sending the composite MOTEUR descriptor and the input link of P_1 and P_2 in the workflow.

2.4.12 Experimental results

The goal of this section is to present experimental results that quantify the relevance of the optimizations described above on a real service-based data-intensive application workflow. We evaluate MOTEUR's performances on two different grid infrastructures.

MOTEUR implementation

We implemented a prototype of a workflow enactor taking into account the optimizations described in section 2.4.8 : workflow, data and service parallelism and sequential processors grouping. Our hoMe-made OpTimisEd scUfl enactoR (MOTEUR) prototype was implemented in Java in order to be platform independent. It is available under CeCILL Public License (a GPL-compatible open source license) at `http://www.i3s.unice.fr/~glatard`. To our knowledge, this is the only service-based workflow enactor providing all these levels of optimization.

The workflow description language adopted is the Simple Concept Unified Flow Language (Scufl) used by the Taverna workbench [157]. This language is well disseminated in the e-Science community. Apart from describing the data links between the services, the Scufl language allows to define so-called coordination constraints. A coordination constraint is a control link which enforces an order of execution between two services even if there is no data dependency between them. We used those coordination constraints to identify services that require data synchronization.

We developed an XML-based language to describe input data sets. This language aims at providing a file format to save and store the input data sets in order to be able to reproduce

workflows execution on reference data sets. It simply describes each item of the different inputs of the workflow.

Handling the data composition patterns presented in section 2.4.7 in a service and data parallel workflow is not straightforward because produced data sets have to be uniquely identified. Indeed they are likely to be computed in a different order in every service, which could lead to wrong dot product computations. Moreover, due to service parallelism, several data sets are processed concurrently and one cannot number all the produced data once computations completed. We have implemented a data provenance strategy to sort out the causality problems that may occur. Attached to each processed data segment is a history tree referring to all the intermediate results computed to process it. This tree unambiguously identifies the data.

Finally, MOTEUR is implementing an interface to both Web Services and GridRPC instrumented application code.

Bronze Standard application

We made experiments considering the *Bronze Standard*, an application that aims at assessing medical image registration algorithms. Medical image registration consists in searching a transformation (that is to say 6 parameters in the rigid case – 3 rotation angles and 3 translation parameters) between two images, so that the first one (the floating image) can superimpose on the second one (the reference image) in a common 3D frame. Medical image registration algorithms are a key component of medical image analysis procedures.

A difficult problem, as for many other medical image analysis procedures, is the assessment of these algorithms robustness, accuracy and precision [102]. Indeed, there is no well established *gold standard* to compare to the algorithm results. Different approaches have been proposed to solve this issue. It is possible to simulate artificial images from a controlled model and to experiment the algorithm on these synthetic images [14]. However, realistic images are difficult to produce and hardly perfect enough for fine assessment of the algorithms. Phantoms (manufactured objects with properties close to human tissues for the imaging modality studied) can also be used to acquire test images. However, it is also very difficult to manufacture realistic enough phantoms.

An alternative for assessing registration algorithms is a statistical approach called the *Bronze Standard* [156]. The goal is basically to compute the registration of a maximum of image pairs with a maximum number of registration algorithms so that we obtain a largely overestimated system to relate the geometry of all the images. It makes this application very compute and data-intensive.

Suppose that we have n images of the same organ of one patient and m registration algorithms. We have in fact only $n-1$ free transformations to estimate that relate all these images, say $\bar{T}_{i,i+1}$. The transformation between images i and j is obtained using a compositions such as $\bar{T}_{i,j} = \bar{T}_{i,i+1} \circ \bar{T}_{i+1,i+2} \circ \ldots \circ \bar{T}_{j-1,j}$ if $i < j$ (or the inverse of both terms if $j > i$). The free transformation parameters are computed by minimizing the prediction error on the observed registrations :

$$\min_{\bar{T}_{1,2},\bar{T}_{2,3},\ldots,\bar{T}_{n-1,n}} \sum_{i,j\in[1,n],k\in[1,m]} d\left(T^k_{i,j}, \bar{T}_{i,j}\right)^2 \tag{2.25}$$

where $T^k_{i,j}$ is the transformation computed between image i and j by the k^{th} registration algorithm, and d is a distance function between transformations chosen as a robust variant of the left invariant distance on rigid transformation developed in [160]. The estimation $\bar{T}_{i,i+1}$ of the perfect registration $T_{i,i+1}$ is called bronze standard because the result converges toward $T_{i,i+1}$ as the number of methods m and the number of images n increase. Indeed, considering a given registration method, the variability due to the noise in the data decreases as the number of

images n increases, and the registration computed converges toward the perfect registration up to the intrinsic bias (if there is any) introduced by the method. Now, using different registration procedures based on different methods, the intrinsic bias of each method also becomes a random variable, which is hopefully centered around zero and averaged out in the minimization procedure. The different bias of the methods are now integrated into the transformation variability. To fully reach this goal, it is important to use as many independent registration methods as possible.

In this process, we do not only estimate the optimal transformations, but also the rotational and translational variance of the "transformation measurements", which are propagated through the criterion to give an estimate of the variance of the optimal transformations. These variance should be considered as a fixed effect (*i.e.* these parameters are common to all patients for a given image registration problem, contrarily to the transformations) so that they can be computed more faithfully by multiplying the number of patients.

The workflow of the bronze standard application is represented on figure 2.53. In the following experiments, we are considering $m = 4$ different registration algorithms in our implementation of the bronze standard method : (1) *Baladin* and (2) *Yasmina* are intensity-based. The former uses a block matching strategy while the later optimizes a similarity measure on the complete images using the Powel algorithm. (3) *CrestMatch* is a prediction-verification method and (4) *PFRegister* is based on the ICP algorithm. Both CrestMatch and PFRegister register features (crest lines) extracted from the input images. These algorithms are further described in [156]. The two inputs *referenceImage* and *floatingImage* correspond to the image sets on which the evaluation is to be processed. The first registration algorithm is *crestMatch*. Its result is used to initialize the other registration algorithms which are *Baladin*, *Yasmina* and *PFMatchICP/PFRegister*. *crestLines* is a preprocessing step. Finally, the *MultiTransfoTest* service is responsible for the evaluation of the accuracy of the registration algorithms, leading to the outputs values of the workflow. This service evaluates the accuracy of a specified registration algorithm by comparing its results with means computed on all the others. Thus, the *MultiTransfoTest* service has to be synchronized : it must be enacted once every of its ancestor is inactive. This is why we figured it with a double square on figure 2.53.

We chose this particular application because it is a real example of data-intensive workflow in the medical imaging field. Moreover, it embeds a synchronization barrier and thus provides an interesting case of complex service-based workflow.

Input image pairs are taken from a database of injected T1 brain MRIs from the cancer treatment center "Centre Antoine Lacassagne" in Nice, France, courtesy of Dr Pierre-Yves Bondiau. All images are 256×256×60 and coded on 16 bits, thus leading to a 7.8 MB size per image (approximately 2.3 MB when compressed without loss).

Grid5000 and EGEE infrastructures

In order to evaluate the relevance of our prototype and to compare real executions to theoretically expected results, we made experiments on two different grid infrastructures : the EGEE production grid[24] and the Grid5000 experimental platform[25]. Grids are novel and complex systems that are difficult to optimally exploit from the end users point of view as their behavior is not very well known. The Grid5000 and the EGEE infrastructures for instance have different characteristics leading to different performances and behaviors under load. We first propose a modeling of these two infrastructure to better interpret the experimental results.

[24] Enabling Grids for E-sciencE, http://www.eu-egee.org/
[25] Grid5000, http://www.grid5000.org/

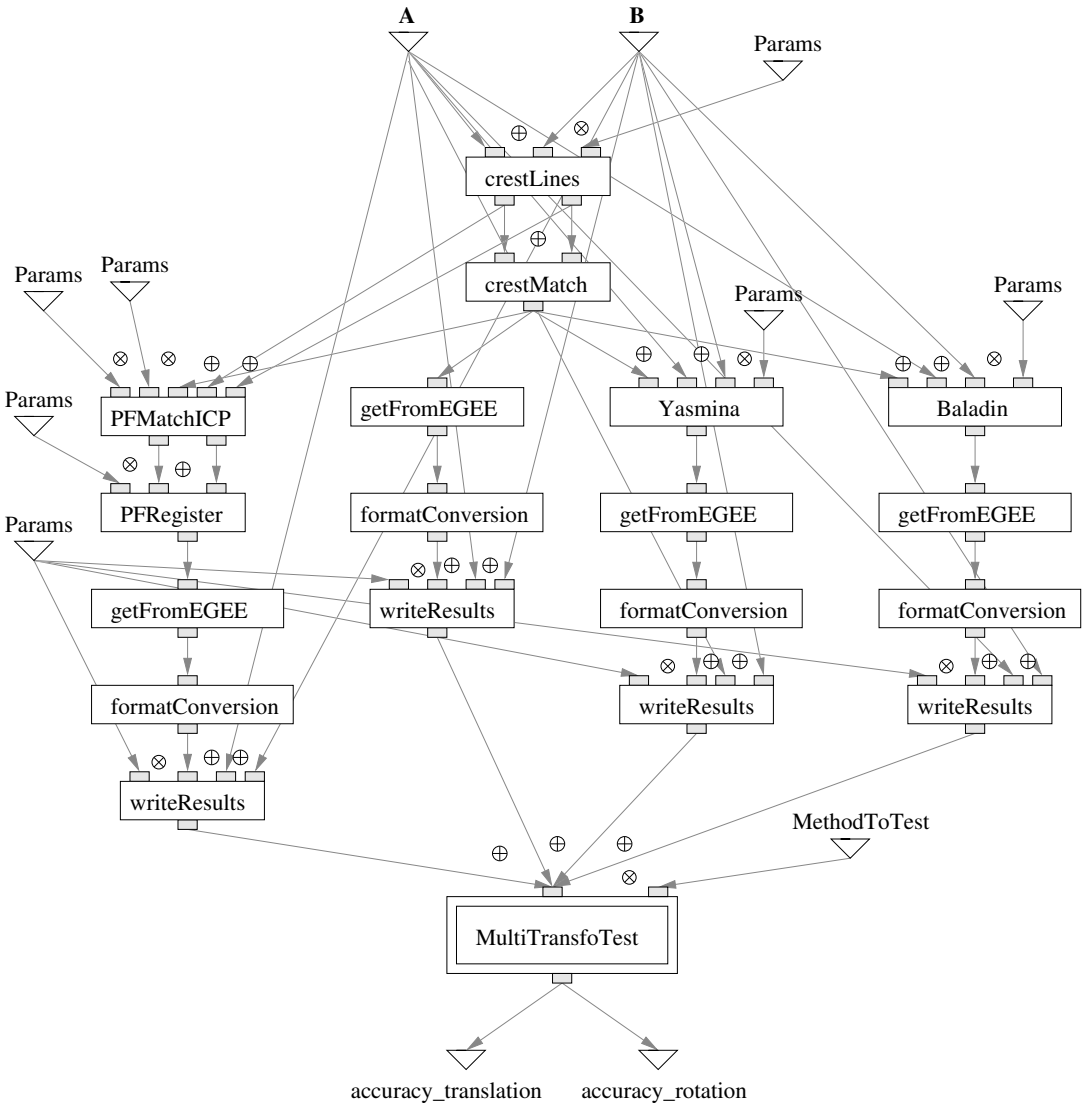


FIG. 2.53 – Workflow of the application

Grids overview. Table 2.13 summarizes the main features of both infrastructures, especially considering the *Workload Management System* (WMS) and the *Data Management System* (DMS), both strongly affecting applications.

Grid5000 is made of 9 clusters located in France, representing more than 2000 CPUs. These resources are shared by dozens of registered users. Inside each cluster, the OAR batch scheduler [24] is used as WMS. The inter-sites GridOAR scheduler is not yet available for users. Hundreds of GB of disk space are shared through NFS [175]. We mostly experimented the Grenoble cluster (12 bi-processor nodes) and the Sophia cluster (105 bi-processor nodes).

EGEE is made of more than 180 computing centers distributed all over Europe and beyond. Hundreds of users are using these resources in production mode (24/7 load of the infrastructure). A total of 18000 CPUs are available out of which 3000 CPUs are effectively accessible to our user community. The EGEE WMS is a two levels batch system : each computing center batch system is fed by higher level *Resource Brokers* (RBs) which receive and queue user computing requests before dispatching them to the available centers. A total amount of 5 PB of storage space is available through *Storage Elements* (SEs) on each site. Data transfers between SEs are handled by gridFTP.

Infrastructure	EGEE - LCG2	Grid'5000
Task description	Job Description Language	command line
Workload Management System	RB	GridOAR
	PBS, BQS, ...	OAR
CPUs	18,400	1,400
Data access	gridFTP	NFS
Storage resources	couple of PB	hundreds of GB

TAB. 2.13 – Overview of the systems

Experimental setting. Figure 2.54 displays our experimental setting. The OAR batch scheduler can receive parallel requests. Our experiments have shown that above 80 parallel connections, OAR is overloaded. To perform load experiments we thus implemented a requests sequencer. The Grid5000 clusters front-end is shared among users. To avoid overloading it, we reported our heavy-weight application on a dedicated node. On the EGEE grid, the application code is similarly executed on a dedicated *User Interface* host. Requests are sent and processed sequentially by the RB.

Given its scale and its usage in production mode, the EGEE infrastructure is more likely to be affected by variable load conditions, network interruptions, and temporary resources volatility. As a side effect, there are outliers : jobs that are lost or blocked for a considerable time before being processed. This problem is characteristic of production grid infrastructures and cannot be ignored or a single job could stop a very complex computation. Timeouts have to be set up to deal with such outliers. Due to these outliers, we did not compute any means nor standard deviations in the analysis of the experimental results shown below. We used *medians* and *interquartile ranges* (IQR) which are less sensitive to outliers instead. The IQR is defined as the interval between the 25% and the 75% lowest values. It corresponds to the range of values measured, centered on the median, after excluding one quarter of low value outliers and one quarter of high value outliers.

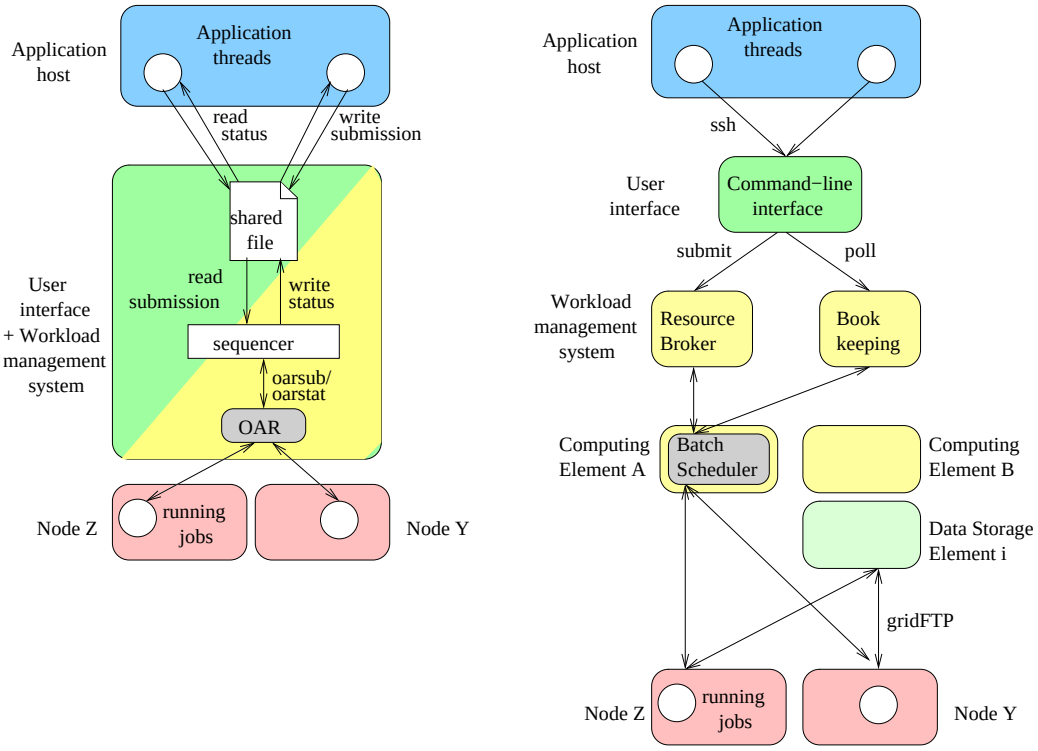


FIG. 2.54 – Grid5000 (left) and EGEE (right) system components

Workload management modeling

While grid infrastructures provide a considerable amount of computing power, the overhead introduced by the WMS when managing large amounts of jobs may cause performances loss. We are studying this overhead by comparing the difference between jobs *execution time* (t_{exec} : the waiting time for the user) and their *running time* (t_{run} : the CPU time consumed). This overhead may be significant on large scale infrastructures, thus penalizing the execution of applications with a high turn-over of jobs to process.

Experimental method. We progressively loaded the Grid5000 and the EGEE WMS by submitting an increasing number, n, of short jobs to the system. We resubmitted a new job each time a job completed, so that the total load introduced by the experiment was constant. We considered short (t_{run} =1 minute long) jobs for favoring a short turn-over of jobs and stressing conditions of the WMS. Experiments were run over 3 hours periods (a long enough period compared to the jobs duration to capture the system behavior over a statistically significant number of measurements).

Results. Figure 2.55 displays the median and the IQR of $t_{\text{over}} = t_{\text{exec}} - t_{\text{run}}$ for a growing number n of submitted jobs. This information measures the spread of the samples and gives an information about the variability of the system. For this experiment, $20,000$ jobs were submitted to the EGEE infrastructure, $32,000$ to the Sophia cluster and $28,000$ to the Grenoble one.

Modeling. As the measurements suggest an affine behavior of the median overheads, we fitted a linear model ($A.n + B$) to the experimental data by linear regression. The lines obtained are plotted on figure 2.55. The parameters of this model are shown in table 2.14, where the

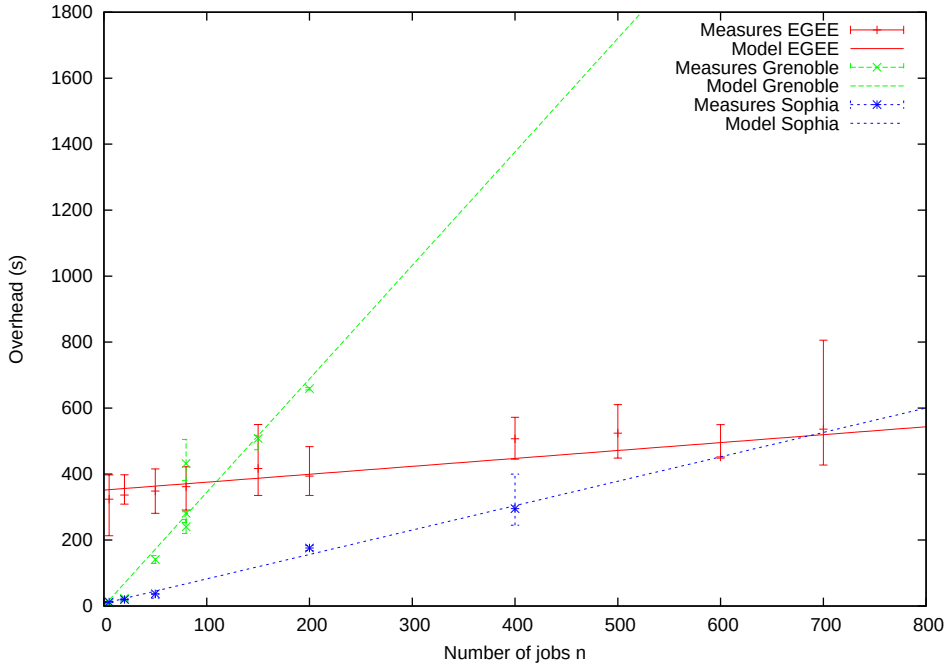


FIG. 2.55 – WMS overhead vs number of jobs

System	A (s/job)	B (s)
Grid5000 – Grenoble	3.44	0.48
Grid5000 – Sophia	0.74	8.25
EGEE	0.24	351.4

TAB. 2.14 – WMS parameters

systems are sorted from the smallest one to the largest. These parameters can be used as metrics characterizing the variation of the median of the overhead with respect to the number of jobs for each system. The B parameter measures the *nominal overhead* of the system. It corresponds to the overhead introduced by the system without any load. A measures the *scalability* of the system with respect to the number of jobs. It represents the additional time generated by the submission of 1 extra job to the system.

Discussion. The nominal overheads, B, are growing with the size of the infrastructure. The EGEE system almost has a 6 minutes overhead due to the infrastructure load and the communication costs while the nominal overhead of the Grid5000 clusters is in the order of seconds. Conversely, the scalability of the systems (A metric) is growing with their size. The EGEE overhead due to the submission of a single extra job is 0.24 second while the Grenoble cluster requires an extra 3.44 seconds per job. On all the systems evaluated, submission is done from a single entry point (the user interface) to a central workload manager (OAR or RB host). There is here a bottleneck and serious performance drops can be forecast in the scheduling when the

load reaches a critical point. Distributed WMS such as presented in [27] should thus be studied.

It also appears from the IQR bars displayed in figure 2.54 that the variability of the system response time for the Grid5000 clusters is increasing with the load. On the EGEE production infrastructure, the situation is quite different as the variability is higher, even when considering a low number of jobs due to the concurrent activity of other users. We proposed in section 2.4.5 a probabilistic framework addressing the problem of large scale systems load.

Data management performances

Experimental setting. To compare the performances of the data management systems of EGEE and Grid5000 infrastructures, we submitted to the infrastructures a number of jobs doing nothing but transferring 7.8MB files on their execution resource. This corresponds to the size of medical images manipulated in the Bronze Standard application. To limit the overhead due to concurrent job submissions, only a few of them (5) were submitted in parallel. Measures were done during 3 hours periods again. The median running time and the IQR are displayed in figure 2.56.

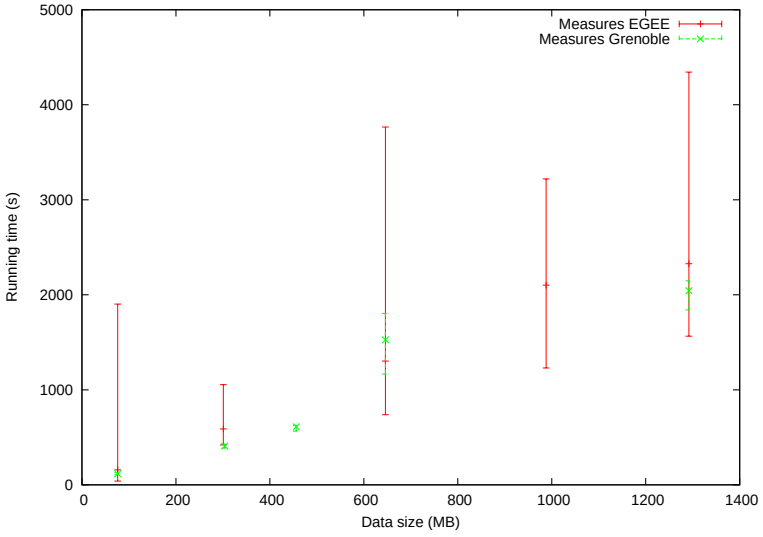


FIG. 2.56 – Median running times of data jobs

Results. Median performances of both data management systems are quite similar : the mean speed-up of the Grenoble cluster data management system with respect to the EGEE one is 1.19. This result indicates a good level of performances for the EGEE data management system as this experiment implied inter-clusters transfers, whereas only intra-clusters transfers were performed on Grid5000. However, the variability of the data transfers time on the EGEE infrastructure is far bigger than on the Grenoble one, which is not surprising given the scale of the infrastructure.

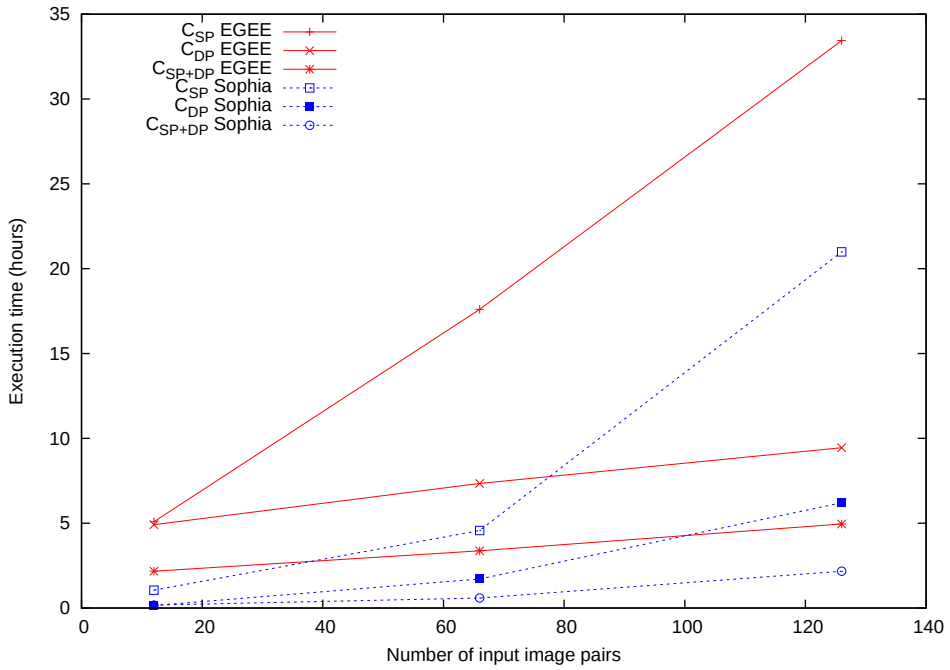


FIG. 2.57 – Comparison of EGEE and Grid5000 infrastructures on the bronze standard application

Experiments on the bronze standard application

We executed the bronze standard workflow on 3 different input data sets, with various sizes, corresponding to the registration of $n_D = 12$, 66 and 126 image pairs corresponding to images from 1, 7 and 25 patients respectively. Each of the input image pair was registered with the 4 algorithms ($n_W = 5$ in this workflow) and leads to 6 job submissions, thus producing a total number of 72, 396 and 756 job submissions respectively. We submitted each data set 6 times with 6 different optimization configurations in order to identify the specific gain provided by each optimization.

Figure 2.57 compares the results obtained both on the EGEE infrastructure and the Sophia cluster of Grid5000. MOTEUR is run on the application host located on figure 2.54. Table 2.15 displays the quantitative values measured on the EGEE infrastructure.

For a given configuration, the execution on the Sophia cluster of Grid5000 is always quicker than on the EGEE system, even for 126 input image pairs. However, we can notice on figure 2.57 that the graphical representations of the execution times with respect to the size of the input data set size for the EGEE infrastructure are almost straight lines. This could be expected as the infrastructure is large enough to scale up despite the increasing load.

The influence of data parallelism can be studied from configurations C_{SP} and $C_{\mathrm{SP+DP}}$. On the Sophia cluster, data parallelism respectively leads to a 6.04, 7.74 and 9.46 speed-ups for 12, 66 and 126 image pairs. On the EGEE infrastructure, corresponding speed-ups are 2.34, 5.22 and 6.76. On both systems, the speed-up introduced by data parallelism is growing with the number of input data sets, which is coherent with the theoretical analysis presented in

Configurations	Computation time (s)		
	12 images	26 images	126 images
NOP	32855	76354	133493
JG	22990	68427	125503
SP	18302	63360	120407
DP	17690	26437	34027
SP+DP	7825	12143	17823
SP+DP+JG	5524	9053	14547

TAB. 2.15 – Execution time for each configuration

	nominal overhead (seconds)	scalability (s/data sets)
NOP	20784	884
JG	11093	900
SP	6382	897
DP	16328	143
SP+DP	6625	88
SP+DP+JG	4310	79

TAB. 2.16 – Nominal overhead and scalability for each configuration

section 2.4.8. The influence of service parallelism can be studied from configurations C_{DP} and $C_{\text{SP+DP}}$. On the Sophia cluster, service parallelism respectively leads to a 0.86, 2.9 and 2.86 speed-up for 12, 66 and 126 input image pairs. On the EGEE infrastructure, corresponding speed-ups are 2.26, 2.17 and 1.90.

Discussion

To analyze performances, the first relevant metric from the user point of view is the speed-up, measured as the ratio of the execution time over the reference execution time. We also used the scalability and the nominal overhead metrics, as introduced in section 2.4.12, which allow a more precise interpretation of experiments on grid infrastructures.

For each configuration, we reported the nominal overhead and the scalability parameters measured on the EGEE infrastructure in table 2.16. Those values were obtained by linear regressions on measurements displayed in table 2.15. We relate our experimental results to the theoretical ones that we presented in section 2.4.8. As our data set is quite homogeneous (all the images have the same size), we make the hypothesis of constant execution times and thus refer to S_{DP}, S_{SP} and S_{SDP} speed-ups.

Impact of the data and service parallelism

DP versus NOP. Given the data-intensive nature of the application, the first level of parallelism to enable to improve performances is data parallelism. In this case, the last paragraph of section 2.4.8 predicted a speed-up $S_D = n_D$. We obtain speed-ups of 1.86, 2.89 and 3.92 for $n_D = 12$, 66 and 126 image pairs respectively. This speed-up is effectively growing with the number of input images as predicted by the theory, although it is lower than expected. Indeed, this experiment shows that the system variability (on transfer and queuing time in particular) and the increasing load of the middleware services on a multiusers production infrastructure cannot be neglected.

To go further in the analysis, we can compute from table 2.16 that in this case, data parallelism leads to a scalability ratio of 6.18 and to a nominal overhead ratio of 1.27. Data parallelism thus mainly influences the scalability ratio. It is coherent as this metric is designed to evaluate the data scalability of the system. Although a higher scalability ratio could be expected on a dedicated system to some extent (until the number of dedicated resources is reached), we can see that in our experiment the grid infrastructure smoothly accepts the increasing load (no saturation effect). This is interesting for applications such as the Bronze Standard that needs the highest number of data to be processed as possible.

(DP + SP) versus DP. One can notice is that service parallelism does introduce a significant speed-up even if data parallelism is enabled. Indeed, it leads to a speed-up of 2.26, 2.17 and 1.90 for $n_D = 12$, 66 and 126 image pairs respectively whereas the theory predicted a speed-up of $S_{SDP} = 1$. This result can be justified by noticing that the constant times hypothesis may not hold on such a production infrastructure, as already suggested in section 2.4.8. On a traditional cluster infrastructure, service parallelism would be of minor importance whereas it is a very important optimization on the production infrastructure we used.

Moreover, we can notice that in case of data parallelism, service parallelism leads to a scalability ratio of 1.62 and to a nominal overhead ratio of 2.46. This is another argument which demonstrates that service parallelism is particularly important on production infrastructures. On traditional clusters indeed, nominal overhead values may be close to 0 and such systems would therefore be less impacted by a reduction of this metric.

Impact of service grouping

To quantify the speed-up introduced by services grouping on a real application workflow, we first made experiments on the bronze standard application. The speed-up introduced by job grouping when comparing JG to NOP is 1.43, 1.12 and 1.06 for $n_D = 12$, 66 and 126 image pairs respectively. It leads to a scalability ratio of 0.98 and to a nominal overhead ratio of 1.87. Job grouping only influences the nominal overhead ratio. It is coherent because it has been designed to lower the system's overhead which is evaluated by the nominal overhead value. In addition to data and service parallelism (JG+SP+DP versus SP+DP), job grouping introduces a speed-up of 1.42, 1.34 and 1.23 for $n_D = 12$, 66 and 126 image pairs respectively. It leads to a scalability ratio of 1.11 and to a nominal overhead ratio of 1.54. Here again, job grouping mainly improves the nominal overhead ratio, which is coherent with the expected behavior. We can thus conclude that job grouping effectively addresses the problem for which it as been designed as it leads to a significant reduction of the system's overhead.

To show how services grouping is able to speed-up the execution on highly sequential applications, we also extracted a sub-workflow from our application, as shown in figure 2.53. It is made of 4 services that correspond to the `crestLines`, `crestMatch`, `PFMatchICP` and `PFRegister` ones in the application workflow. Our grouping rule groups those 4 services into a single one, as it has been detailed in the example of figure 2.50. It is important to notice that even if this sub-workflow is sequential, and thus does not benefit from workflow parallelism, its execution on a grid does make sense because of data and service parallelisms. Table 2.17 presents the speed-ups induced by our grouping strategy for a growing number of input image pairs. Experiments were lead on the grid5000 infrastructure for the two workflows described above. We can notice on those tables that services grouping does effectively provide a significant speed-up on the workflow execution. This speed-up is ranging from 1.23 to 2.91.

The speed-up values are greater on the sub-workflow than on the whole application one. Indeed, on the sub-workflow, 4 services are grouped into a single one, thus providing a 3 jobs submission saving for each input data set. On the whole application workflow, the grouping

Number of input image pairs	Speed-up on the sub-workflow	Speed-up on the whole application
12	2.91	1.42
66	1.72	1.34
126	2.30	1.23

TAB. 2.17 – Grouping strategy speed-ups

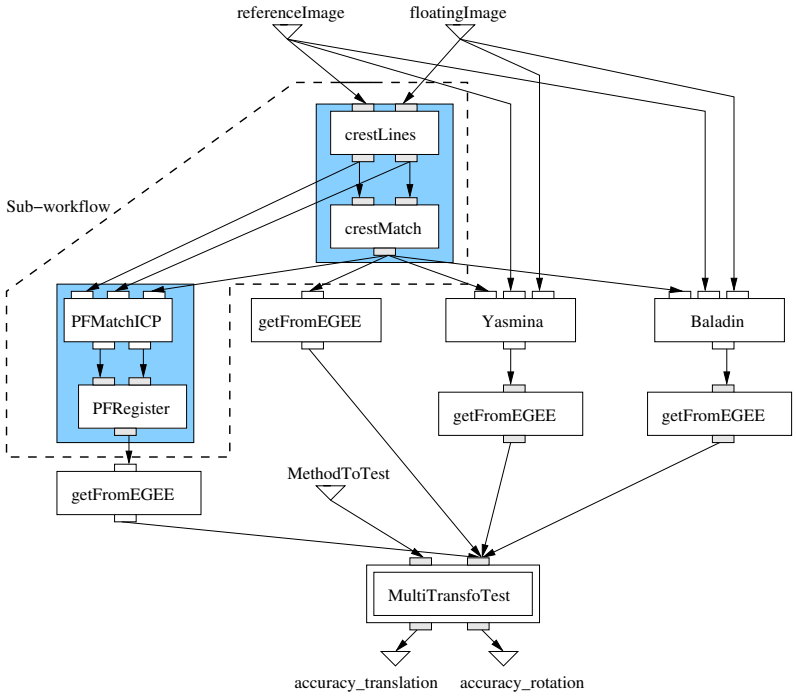


FIG. 2.58 – Workflow of the application. Services to be grouped are squared in blue.

rule is applied only twice, leading to a 2 jobs saving for each input data set, as depicted on figure 2.58.

Optimization perspectives

The nominal overhead and scalability parameters are able to quantify how an application could be improved, without any reference to the scale of the infrastructure. Indeed, an ideal system would have a null scalability ratio and a close to zero overhead.

The nominal overhead value of DP+SP+JG quantifies the potential overhead reduction that could be targeted. In the future, we plan to address this problem by grouping jobs of a single service, thus finding a trade-off between data parallelism and the system's overhead.

Besides, the scalability value of DP+SP+JG quantifies the potential data scaling improvement that could be targeted. On an ever-loaded production infrastructure, middleware services such as the user interface or the resource broker may be critical bottlenecks. The theoretical modeling does not take into account these limitations. A probabilistic modeling considering the variable nature of the grid infrastructure is probably an interesting future path to explore for further optimizing this value [84].

Largest system	Smallest system	n_0	$n_{0.5}$	$\delta(\infty)$
EGEE	Sophia	232 jobs	686 jobs	76%
EGEE	Grenoble	51 jobs	110 jobs	93%
Sophia	Grenoble	1 job	3 jobs	82%

TAB. 2.18 – Multigrids model parameters

Multigrids model

Grid5000 and EGEE exhibit different behaviors under load. When exploiting both infrastructures concurrently, it is therefore interesting to determine, given a number of jobs n to process, the optimal fractions $\delta \in [0,1]$ and $1-\delta$ of these jobs that should be submitted to each infrastructure to minimize the total execution time. Let $o_i(n)$ be the median overhead time introduced by system i when it deals with the submission of n concurrent jobs. The goal is to minimize the mean overhead time of the submitted jobs :

$$H(\delta) = \delta o_1(\delta n) + (1-\delta) o_2\left((1-\delta)n\right)$$

If we consider the linear model introduced in section 2.4.12, we get :

$$H(\delta) = \delta(A_1 \delta n + B_1) + (1-\delta)(A_2(1-\delta)n + B_2)$$

where, A_i and B_i are the model parameters of the i^{th} system. H has a unique minimum reached for the optimal proportion of jobs $\hat{\delta}$ to submit on the first system :

$$\hat{\delta}(n) = \frac{B_2 - B_1 + 2A_2 n}{2n(A_2 + A_1)} \tag{2.26}$$

We have to determine when $\hat{\delta}(n)$ is in [0,1]. Suppose that system 1 is larger than system 2. According to section 2.4.12, it implies that $B_1 > B_2$ (nominal overhead of the largest system is the highest one) and $A_1 < A_2$ (the scalability of the largest system is better). In this setting, it is straightforward to prove that $\hat{\delta}(n) < 1$, showing that the proportion of jobs to submit on the smallest infrastructure is never null (as long as it is not overwhelmed it is faster to exploit it). Moreover, $\hat{\delta}(n) > 0$ if and only if $n \geq n_0 = \frac{B_1 - B_2}{2A_2}$. Below this threshold, the number of jobs is low enough for all of them to be submitted to the smallest, but fastest, infrastructure. Beyond n_0, the number of jobs to be submitted to the largest infrastructure increases. For $n_{0.5} = \frac{B_1 - B_2}{A_2 - A_1}$, both infrastructures are loaded with the same number of jobs. Beyond, the model enters a saturation phase, where $\hat{\delta}$ tends to its asymptotic value $\hat{\delta}(\infty) = \frac{A_2}{A_1 + A_2}$. This value is inferior to 1 and denotes the remaining proportion of jobs that would always be submitted to the largest platform, even if the number of concurrently submitted jobs becomes very high.

Table 2.18 displays the different thresholds for the EGEE and grid5000 infrastructures, considering the parameters from table 2.14. Note that n_0 is high compared to the number of CPUs available on the smaller infrastructures. The $n_{0.5}$ values lead to similar interpretations. It corresponds to the abscissa where the lines cross on figure 2.55. We thus can see that the EGEE infrastructure and the Sophia cluster lead to the same overhead if 686 jobs are submitted on each infrastructure. This number of jobs is 110 when comparing EGEE to the Grenoble cluster and 3 for the Sophia versus Grenoble comparison. When considering asymptotic behavior, the Sophia cluster should handle $\delta(\infty) = 82\%$ of jobs when used concurrently with the Grenoble cluster due to their difference in size (this result is close to the proportion of nodes on the Sophia cluster in the total number of nodes on the two systems : $\frac{105}{105+12} = 89.7\%$). When comparing EGEE to the Sophia cluster, $\delta(\infty) = 76\%$: it is never efficient to submit more than three quarter of the jobs on EGEE.

2.4.13 Conclusions and perspectives

Motivated by the efficient implementation of complex medical image analysis procedures such as the Bronze Standard application, we have been studying workflow-based grid solutions. Our first experience with existing workflow managers showed that they are far from exploiting optimally grid parallelism, especially when considering massively data parallel applications. We have developed MOTEUR to improve the application performances by exploiting three different levels of parallelism (workflow, data and services parallelism) and by enabling sequential jobs grouping. MOTEUR is to our knowledge the only service-based workflow manager enabling all these levels of optimization.

The modern service-based approach is still hardly considered in production grid middlewares when it comes to user code execution. However, its advantages are many. In the case of workflow description and execution in particular, the service-based approach enables compact description of complex data composition patterns. Furthermore, it eases the dynamic deployment of the application, it makes the application independent from the grid platform through submission services and it enables job grouping.

Estimating the performance of our parallel workflow manager, is not a straight forward problem on a production system. The theoretical study does not include the infrastructure load and therefore theoretical speed-ups correspond to asymptotic values that could only be reached on an ideal system (infinite, under no load and introducing no overhead). It provides upper bounds but these can be far from the maximum performance achievable on a real multiusers production system. Comparing the situation to a parallel code running on a supercomputer, this is similar to try to estimate the speed-up of the parallelization without taking into account other users usage of the system. The parallel code is likely to be queued by a batch scheduler for a long time before being executed and may be executed in shared time with other users processes, thus introducing artificial slow downs not dependent on the code itself. we have identified metrics that are related to grid infrastructure characteristics and that give a better insight on the behavior of the grid application than raw speed-up. Our on-going research activity intends to go further by integrating a statistical model of the grid load, similar to the one introduced in section 2.4.5, into theoretical speed-up analysis. This proved to provide much more realistic speed-up bound values, varying with the infrastructure load, by compensating for other users activity.

Another perspective of this work is the extension of the multigrids submission model. The current model does not take into account data transfers that may be needed between correlated tasks submitted to different grid infrastructures. These transfers may be highly penalizing and should be accounted for in the optimization procedure.

Chapitre 3

Contribution au fonctionnement de la recherche

3.1 Activités contractuelles et collaborations

Les activités contractuelles constituent à ce jour la très grande majorité du budget de fonctionnement dont je dispose pour animer mes recherches. Elles constituent donc un volet absolument indispensable. On peut néanmoins regretter la charge administrative considérable qu'elles représentent en pratique, en particulier les projets Européens dont les procédures sont extrêmement rigides. Si le rôle des appels thématiques et des financements contractuels pour le pilotage de la recherche est indéniablement très important, un certain équilibre entre financements institutionnels récurrents et financements contractuels faciliterait néanmoins grandement l'organisation de la recherche et éviterait surtout que l'activité de recherche ne devienne pas une activité trop secondaire dans l'emploi du temps des jeunes chercheurs.

Projet Européen FP5 IST DataGrid (2001-2003)

Le projet Européen FP5 DataGrid de la division *Information Society Technologies* avait pour objectif le déploiement d'une infrastructure de grille pan-européenne pour soutenir les activités scientifiques de trois domaines applicatifs :

- la physique des hautes énergies ;
- les sciences de la terre ;
- la biologie.

DataGrid était un mastodonte du FP5 piloté par le CERN avec pour principal objectif le déploiement d'une grille capable d'absorber et de traiter le flot de données qui sera produit en 2008 lors de la mise en œuvre du *Large Hadron Collider*, le prochain accélérateur de particules en cours d'assemblage dans le sous-sol Franco-Suisse. Le projet était structuré en 12 lots de travail : 5 pour le développement d'un intergiciel, 1 pour l'intégration, 1 pour l'interaction avec le réseau, 3 groupes applicatifs, la dissémination et le management. Le CNRS était responsable du lot WP10 sur les applications en biologie dont l'objectif était l'identification des besoins en terme de grille de calcul dans ce domaine et la mise en oeuvre d'applications montrant l'impact d'une telle infrastructure dans ce domaine scientifique.

Le début de DataGrid (janvier 2001) coïncide avec ma prise de fonction au laboratoire CREATIS en tant que CR CNRS. J'ai très rapidement intégré le projet DataGrid et je suis devenu *deputy leader* du WP10. J'ai élargi la thématique du groupe de travail de la bioinformatique à l'imagerie médicale. Cette activité m'a conduit à réaliser une analyse des besoins applicatifs du domaine de l'imagerie médicale pour les systèmes distribués puis à m'impliquer dans les développements applicatifs et les services d'intergiciel. J'y ait joué en rôle d'intermédiaire entre la communauté informatique, la communauté des utilisateurs et les ingénieurs développement. Ces trois cultures représentent certainement un cocktail aux propriétés tout à fait variées et intéressantes en terme d'objectifs et de moyens.

Le projet a rencontré un grand succès et a été poursuivi dans le FP6 par le projet EGEE, puis le projet EGEE2. Deux domaines applicatifs pilotes ont été préservés dans cette transition : la physique des hautes énergies et les applications biomédicales.

Projets Européens FP6 IST I3 EGEE (2004-2006) et EGEE2 (2006-2008)

EGEE est la continuation de DataGrid dans le FP6, sous la forme d'un I3 (*Integrated Infrastructure Initiative*), toujours piloté par le CERN. Si DataGrid était un mastodonte du FP5, EGEE n'a absolument rien à lui envier dans le FP6. Avec 73 partenaires répartis dans 23 pays Européens et un budget de 32 M€ sur deux ans, EGEE représentait le plus gros projet IST du FP6. EGEE vient de perdre ce titre au profit d'EGEE2 qui, avec 91 partenaires et un budget de 36 M€, donne suite à EGEE à compter du 1er avril 2006.

De DataGrid à EGEE/2, l'objectif du projet a évolué vers la mise en œuvre d'une infrastructure de grille pérenne. Une petite moitié du budget du projet est investie dans la mise en place et la manutention de cette infrastructure (activité SA1) composée aujourd'hui d'un peu plus de 18000 processeurs répartis dans 180 centres de calcul sur 3 continents (mais principalement en Europe). Les deux autres activités principales sont le développement de l'intergiciel (activité JRA1) et les groupes applicatifs (NA4).

Dans EGEE/NA4, j'ai joué le rôle de leader du groupe "applications biomédicales". Ce sous groupe du NA4 représente à lui seul 7 partenaires dans EGEE et 15 partenaires dans EGEE2. Ce groupe a conduit à la mise en œuvre de 12 applications sur l'infrastructure EGEE dans les domaines de la bioinformatique, l'imagerie médicale et la recherche de médicaments. Il a contribué de manière importante à guider les orientations des développements intergiciels et à faire reconnaître les besoins de ce domaine.

ACI-GRID MEDIGRID (2002-2005)

J'ai été responsable de l'Action Concertée Incitative GRID (Globalisation des Ressources Informatique et des Données) MEDIGRID. Ce projet, pionnier dans son domaine, avait pour objectif l'exploration des grilles de calcul pour la manipulation et le traitement de grand volumes de données d'images médicales. Trois partenaires ont participé : le laboratoire CREATIS (UMR 5515) ainsi que le LISI et le laboratoire ERIC (désormais regroupés dans le LIRIS - UMR 5205).

MEDIGRID a permis de réaliser un gros travail exploratoire d'analyse des besoins de la communauté du traitement d'images médicales et d'expérimentation des infrastructure de grille pour répondre à ces besoins. Le projet était guidé par les besoins issus de quatre applications différentes identifiées dans la proposition. En raison de la jeunesse et de la très grande évolution des intergiciels de grille dans la période du projet, la mise en œuvre qui était planifiée sur l'infrastructure DataGrid a été très délicate. Trois applications ont néanmoins pu être déployées sur des infrastructures de grille dans la durée du projet. Les résultats ont conforté l'utilité de ce type d'outil pour répondre à des questions scientifiques issues du monde médical.

MEDIGRID a également permis de co-financer la thèse d'Hector Duque [56] qui a réalisé un travail amont sur la gestion de données médicales distribuées. Ce travail a conduit à la proposition d'une architecture efficace d'accès aux données et à la mise en œuvre d'un prototype. Le travail de thèse de Ludwig Seitz [181], financé par le LIRIS, s'est également déroulé dans la même période. Il a porté sur la sécurité et le respect de la confidentialité des données médicales.

ACI-MD AGIR (2004-2007)

En termes de thématique, l'ACI "Masse de données" AGIR (Analyse Globalisée des Images Radiologiques) s'inscrit dans la continuité du projet MEDIGRID. AGIR s'intéresse à la mise en œuvre de services intergiciels spécifiques aux applications de traitement d'images médicales et au déploiement d'applications sur la base de ces services. Le projet s'appuie sur l'infrastructure EGEE pour le déploiement des applications visées.

Je suis responsable dans AGIR de deux groupes de travail portant sur la gestion des données médicales et la gestion de flots de données sur grilles. Le premier s'inscrit dans la continuité du travail réalisé dans MEDIGRID et a permis le déploiement d'un gestionnaire de données médicales distribuées interfacé avec la grille EGEE [152]. Le second constitue une nouvelle thématique de recherche identifiée dans l'analyse issue de MEDIGRID. Elle s'est révélée être l'objet d'un grand intérêt dans la communauté des grilles. Cette activité est réalisée dans le cadre de la thèse de Tristan Glatard (bourse MESR associée à l'ACI) [85].

Le projet AGIR a été l'occasion de très nombreuses prises de contact et collaborations.

Projet région Rhône-Alpes RAGTIME (2003-2006)

Le projet Région RAGTIME (Rhône-Alpes, Grille pour le Traitement d'Images Médicales) est un sous-produit de l'ACI-GRID MEDIGRID qui permet de fédérer le travail réalisé par différents partenaires régionaux autour de cette thématique. Il réunit des acteurs des intergiciels de grille et des laboratoires travaillant sur les applications à la bioinformatique et l'imagerie médicale. Je suis responsable du lot de travail "applications" dans RAGTIME.

Projet Grid5000

J'ai également participé au projet d'infrastructure nationale de recherche Grid5000 en tant que représentant de la communauté des utilisateurs au laboratoire CREATIS (nœud Grid5000 de Lyon) puis au laboratoire I3S (nœud Grid5000 de Sophia Antipolis). Nous déployons actuellement une application à la validation de recalage d'images médicales sur Grid5000 en partenariat avec le projet Asclepios de l'INRIA Sophia Antipolis.

Collaborations

Les activités de recherche menées dans le cadre de ces activités contractuelles ont été l'occasion d'établir de nombreuses collaboration avec des partenaires nationaux et internationaux.
Collaborations nationales :

- Collaboration avec l'unité mixte CNRS-INSERM CREATIS (Pr Isabelle Magnin) portant sur le développement de services de gestion de données médicales sur grilles.
- Collaboration avec le projet ASCLEPIOS (ex EPIDAURE) de l'INRIA (Dr Xavier Pennec) sur le déploiement d'applications de traitement de données médicales.
- Collaboration avec le Centre hospitalier Antoine Lacassagne de Nice (Dr Pierre-Yves Bondiau, oncologue) sur le suivi de patients souffrant de cancer du cerveau.
- Collaboration avec le LRI (Dr Cécile Germain-Renaud) et le LORIA (Dr Emmanuel Jeannot), dans le cadre de l'ACI masse de données AGIR, sur les aspects intergiciels.
- Collaboration avec l'équipe GRAAL de l'INRIA (Dr Frédéric Desprez), portant sur l'ordonnancement de flots de calcul.

Collaborations internationales :

- Consortium Européen NA4/biomed, travaillant sur l'exploitation des données biomédicales sur grille, dans le cadre du projet EGEE.
- Collaboration avec STA SZTAKI (Pr Peter Kascuk, Hongrie), dans le domaine de la gestion des flots applicatifs.
- Collaboration avec l'Université de Manchester (Pr Carole Goble, UK), réalisant le moteur de flots Taverna (projet UK eScience MyGrid).

Récapitulatif

Projet	Période	Financement total	Part laboratoire (CREATIS, I3S)	Autre
DataGrid	2001-2003	10 M€	42 k€	
MEDIGRID	2002-2005	250 k€	120 k€	
Ragtime	2003-2006	247 k€	45 k€	
EGEE	2004-2006	32 M€	153 k€	
AGIR	2004-2007	262 k€	60 k€	1 bourse MESR
EGEE2	2006-2008	36 M€	162 k€	
Total		78.759 M€	582 k€	

3.2 Participation à l'organisation de conférences

Organisation de conférences

- **Tutorial sur les grilles de calcul et les applications au recalage d'images (MICCAI), Saint-Malo, 2004**. Co-organisateur avec Xavier Pennec et Derek Hill.
- **Functional Imaging and Modeling of the Heart (FIMH), Lyon, 2003**. Comité d'organisation, comité scientifique et édition des actes.
- **HealthGrid, Lyon, 2003**. Comité d'organisation, comité scientifique et édition des actes.
- **Functional Imaging and Modeling of the Heart (FIMH), Helsinki, 2001**. Comité d'organisation, comité scientifique et édition des actes.

Comités

J'ai participé aux comités scientifique des congrès, HealthGrid 2003 à 2006, FIMH 2001 et 2003, BioGrid 2003, EuroPar 2004, VLDB 2005, WORKS 2006 et JETIM 2006.

3.3 Encadrement

Co-encadrement de thèses

- Tristan Glatard, depuis septembre 2004, "assemblage et calcul efficace d'expériences en traitement d'images médicales par composition sur une grille de calcul". Directeurs de thèse : Michel Riveill et Nicholas Ayache
- Hector Duque, 2002-2005, "Conception et mise en œuvre d'un environnement logiciel de manipulation et d'accès à des données réparties. Application aux grilles d'images médicales : le système DSEM/DM2" [56]. Directeurs de thèse : Isabelle Magnin et Lionel Brunie.

Encadrement de stages de DEA et stages de fin d'étude ingénieurs

- Vincent Léon, stage de master recherche 2006, "Optimisation du flot d'exécution d'applications manipulant des masses de données sur grilles de calculs" [112]
- Tristan Glatard, stage de DEA 2004, "Indexation d'images médicales" [75]
- Rida Kathoun, stage de DEA 2004, "Distribution de données et de méta-données médicales" [107]
- Pascal Béringuié, stage ingénieur 2003, "segmentation cardiaque"
- Frédéric Forjan, Henri-Jean Dornbier, Thierry Coustillac, Projet ingénieur 2003, "Interface de gestion de données médicales sur une grille"
- Eduardo Davila, stage de DEA 2002, "Déportations d'algorithmes interactifs de traitement d'images médicales sur des serveurs distants" [44]

Co-encadrement de stage de DEA

- Arnaud Charnoz, stage de DEA 2003, "Segmentation du coeur dans des séquences d'images 3D TEMP par ensemble de niveaux" [31]. Direction : Diane Lingrand.
- Nicolas Scapel, stage de DEA 1999, "Changement de topologie automatique sur une surface déformable : application à la segmentation d'images médicales 3D". Direction : Hervé Delingette
- Maxime Sermesant, stage de DEA 1999, "Diffusion anisotrope et segmentation par modèles déformables sur des images échographiques 4D du coeur". Direction : Hervé Delingette.

3.4 Enseignement

J'ai participé à des activités d'enseignement depuis le début de ma thèse, en tant qu'allocataire moniteur dans un premier temps puis en tant que vacataire depuis mon recrutement au CNRS. J'ai toujours trouvé un intérêt certain pour l'activité d'enseignement, bien qu'elle n'entre pas strictement dans les prérogatives des chercheurs. La majorité de mes enseignement a été réalisée en école d'ingénieur (2ème et 3ème cycle) mais beaucoup plus rarement en master recherche, la difficulté à trouver des enseignant et donc la demande étant visiblement très importante dans le premier cas et tout simplement inexistante dans le second.

Mon activité d'enseignement se résume en volumes horaires à :

Année	cours	TD
1997-1998	21h	98h
1998-1999	7h	100h
1999-2000	7h	108h
2001-2002	10h	
2002-2003	13h	36h
2003-2004	15h	72h
2004-2005	32h	78h
2005-2006	33h	48h
Total	138h	540h

Ces enseignement se répartisse essentiellement entre :
- Programmation orientée objet (C++, java)
- Synthèse d'images
- Traitement d'images
- Imagerie médicale
- Technologies de grilles de calcul

Je suis également depuis l'année universitaire 2005 responsable de la filière de troisième année VIMM (Vision, Image et MultiMédia) de l'Ecole Polytechnique Universitaire de Nice-Sophia Antipolis. Cette nomination a été poussée par l'absence de candidat à ce poste parmi le corps professoral de cet établissement dans lequel je réalise la majorité de mes enseignements et ma volonté de contribuer au bon fonctionnement de l'organisme.

En 2006, j'ai été sollicité en tant qu'enseignant dans le domaine des grilles de calcul à l'école d'été organisée par le projet Européen SEEGRID à Budapest.

Chapitre 4

Conclusions et perspectives

4.1 Leçons tirées des principaux résultats présentés

Mes principaux travaux de recherche ont porté sur l'analyse d'images médicales et l'exploitation d'infrastructures de grilles de calcul. Dans les deux cas, l'application médicale est la principale motivation pour la réalisation de ces travaux. J'oriente mon activité en fonction de ce but concret.

Mes travaux sur la segmentation d'images médicales ont conduit à l'amélioration de la technique des modèles surfaciques déformables discrets. Le résultat est une formalisation du problème de déformation de la surface sous une forme qui permet une meilleure régularisation par l'intégration conjointe de transformations globales et de déformations locales. La force de cette approche repose sur la possibilité de passer de manière itérative de transformations globales de l'espace à des déformations locales au cours du processus de déformation.

Les travaux réalisés dans ce cadre ont conduit au développement d'un outil de segmentation semi-supervisé, intégrant un retour possible de l'utilisateur pour corriger les résultats de segmentation automatique. Cette approche pragmatique s'est révélée indispensable dans un contexte où la convergence de l'algorithme de segmentation n'est jamais garantie vers le résultat désiré et l'utilisateur médical se doit de contrôler les résultats fournis et exploités dans le cadre d'une procédure médicale. Un résultat tangible de ce travail est la mise en œuvre qui en a été réalisée par l'équipe R&D de Philips Medical Systems sur la base des publications scientifiques réalisées.

Il reste que le problème de la segmentation d'images, même restreint aux seules images médicales, est particulièrement complexe et difficile à traiter sans étudier et développer un algorithme au cas par cas, en fonction du type et de la qualité des images traitées. L'avantage de notre approche est de permettre la prise en compte de nombreux cas à travers un paramétrage flexible de l'algorithme. Ce paramétrage nécessite néanmoins une bonne compréhension du mécanisme de segmentation et ne peut donc être réalisé que par un utilisateur expert.

Les travaux sur l'analyse d'images médicales m'ont peu à peu conduit à envisager l'exploitation d'infrastructures distribuées de type grille pour mettre en œuvre des procédure d'analyse de grandes quantité de données. Ce travail a débuté par l'exploitation d'une grille interne à l'Institut Neurologique de Montréal dans le cadre de l'évaluation d'une méthode de mesure d'atrophie cérébrale utilisée pour mesurer l'évolution de scléroses en plaques. J'ai par la suite systématisé cette approche avec l'exploitation de grilles de production dans ce cadre applicatif.

Les infrastructures de grille sont prometteuses dans le cadre de nombreux problèmes d'analyse d'images médicales pour deux raisons principales :

– Ces infrastructures sont très adaptées au traitement de problèmes exprimant un parallélisme de données gros grain. C'est le cas de toutes les procédures nécessitant le trai-

tement de bases de données d'images complètes.
- Les centres médicaux ne disposent que de peu de ressources de calcul et jamais d'accès à des centres de calcul haute performance. Les infrastructures de grille permette la mutualisation de ressources disponibles et l'exploitation de ressources distantes qui peuvent être mises à disposition par différents partenaires dans le cadre de recherches médicales.

La mise en œuvre de services de gestion de données qui intègrent les contraintes de confidentialité relatives à la manipulation d'images médicales est techniquement parfaitement réalisable. Les difficultés rencontrées dans ce domaine sont souvent plus liées aux problèmes humains.

Une limitation de l'utilisation des grilles pour le traitement de grand volumes de données est néanmoins la fiabilité des traitements automatisés qui peuvent être appliqués sur des images médicales. De nombreux traitement, dont les algorithmes de segmentation mentionnés ci-dessus, nécessitent une supervision ou au moins un contrôle humain. Dans le cadre de l'analyse de bases de données composées de dizaines, centaines, voire milliers d'images, c'est autant de résultats qu'il faudra analyser et contrôler. Ceci peut devenir extrêmement fastidieux dans certains cas (la validation par un expert peut être complexe et nécessiter du temps) et les résultats sont en général assez qualitatifs (variabilité inter- et intra-experts).

Heureusement, les grilles elles-mêmes peuvent constituer un outil pour améliorer la fiabilité des procédures de traitement dans la mesure où elles permettent :
- d'explorer l'espace des paramètres d'algorithmes afin d'optimiser leurs performances dans un cadre précis ;
- de traiter des grandes bases de données quitte à rejeter une fraction non négligeable de résultats non fiables (dans les cas où il est possible d'identifier automatiquement ou manuellement à faible coût les résultats erronés) ;
- et de mettre en place des procédures statistiques d'évaluation et d'algorithmes.

La procédure du *Bronze Standard* par exemple est capable de rejeter automatiquement des résultats erronés par un test statistique sur l'ensemble des résultats produits et conduit à l'estimation de performances d'algorithmes de recalage utile pour de très nombreuses applications de traitement d'images médicales.

En revanche, nombre d'algorithmes devront être repensés pour permettre leur exploitation dans un cadre systématique et les critères statistiquement discriminant devront être identifiés pour chacun d'eux afin qu'il puissent être intégrés de manière fiable dans de telles procédures de traitement.

4.2 Perspectives scientifiques

Les grilles de calcul ont connu une évolution considérable ces dernières années qui ont permis de passer d'un stade de prototype à celui de mise en œuvre d'infrastructures de production. Pour autant, ce sont encore des technologies jeunes, qui adressent essentiellement des problèmes de bas niveau aujourd'hui, et pour lesquelles il existe un espace d'évolution considérable. La pertinence de ces infrastructures a été démontrée dans un nombre varié d'applications, y compris dans le domaine médical, mais les performances peuvent encore largement être améliorées. En terme de fiabilité pour commencer : les grandes grilles de production affichent encore un taux de fiabilité relativement bas (de l'ordre de 85%). Certaines améliorations pourront être apportées au niveau système mais le problème est plus large que cela (les problèmes d'approvisionnement électrique des grappes de calcul et de fiabilité des systèmes de climatisation sont responsable d'une fraction non négligeable des 15% d'échecs enregistrés). En termes de performances ensuite : il reste une marge significative pour améliorer les performances des intergiciels de production et distribuer les services critiques de manière à améliorer la tolérance aux pannes et à supprimer les goulots d'étranglement de tels systèmes.

Dans le cadre de l'analyse d'images médicales de manière plus spécifiques, de nombreuses fonctionnalités sont encore nécessaires pour faciliter l'exploitation d'infrastructure de grilles. Dans certains cas, l'utilisation en contexte clinique n'est même pas envisageable (en raison des contraintes légales relatives à la manipulation de données sensibles ou d'étapes manuelles de traitements qui deviennent excessivement fastidieuses dans le contexte du traitement de grands volumes de données).

Les problèmes de confidentialité et de sécurité des données exportées sur l'infrastructure distribuée de grille reste un problème pour toutes les applications manipulant de données individuelles. La nature inter-structures administratives et interdisciplinaire des grilles de production rend la mise en œuvre de politiques de gestions de données respectant les réglementations en vigueur dans différents pays participants à l'infrastructure de grille délicate. Les mécanisme de protection des données (encryption) et de contrôle d'accès basés sur les rôles (RBAC) fournissent des outils pour résoudre techniquement ces problèmes. Il reste que la prise en compte de politiques de gestion de données et de réglementations est un problème difficile. Il sera sans doute indispensable de procéder progressivement en exploitant seulement des données rendues accessibles pour des besoins de recherche dans le cadre de protocoles médicaux précis (de nombreuses bases de données d'images sont déjà disponibles aujourd'hui, telles que la base de donnée de la DDSM dans le cadre de la recherche sur le cancer du sein). L'élargissement à d'autres sources de données nécessitera probablement l'établissement d'une confiance suffisante par la communauté des utilisateurs médicaux et l'adoption des technologies grâce à des exemples phares d'exploitation des grilles de calcul pour résoudre des problèmes qui n'étaient pas traitable en raison de la fragmentation des systèmes d'information existants. Cette évolution nécessitera probablement plus de temps que la résolution des problèmes techniques sous-jacents.

Des services de haut niveau pour gérer des données médicales semi-structurées et géographiquement distribuées sont également indispensables à la mise en œuvre relativement aisée de nouvelles procédures d'analyse d'images. L'exploitation de la sémantique des données est cruciale dans le cadre de l'imagerie médicale. De nombreux acquis du domaine du web sémantique pourraient servir d'inspiration dans ce cadre.

Un autre facteur essentiel au succès des technologies de grilles dans le domaine médical est l'accessibilité de telles infrastructures pour une communauté non spécialisée. Les utilisateurs finaux ne sont intéressés que par les résultats finaux et non pas par la technologie utilisée. L'adoption des grilles ne sera possible que lorsqu'un niveau de transparence complet aura été atteint. Des interfaces spécialisées et une intégration complète avec les réseaux hospitaliers de données (PACS, RIS et HIS) sont indispensables.

L'exploitation des grilles dans le domaine médical impliquera donc bien plus que le développement de solutions techniques hautes performances et le déploiement d'infrastructures. Je pense que l'activité de recherche, pour être productive, ne peut pas se concentrer uniquement sur la technique et les performances mais doit nécessairement appréhender l'ensemble du contexte et des contraintes associées.

Bibliographie

[1] R. Acharya, R. Wasserman, J. Sevens, and C. Hinojosa. Biomedical Imaging Modalities : a Tutorial. *Computerized Medical Imaging and Graphics (CMIG)*, 19(1) :3–25, 1995.

[2] R. Alfieri, R. Cecchini, V. Ciaschini, L. dell'Agnello, Ákos Frohner, A. Gianoli, K. Lörentey, and F. Spataro. VOMS, an Authorization System for Virtual Organizations. In *European Across Grids Conference (EAGC)*, 2003.

[3] B. Allcock, J. Bester, J. Bresnahan, A.L. Chervenak, Ian Foster, Carl Kesselman, S. Meder, V. Nefedova, D. Quesnal, and S. Tuecke. Data Management and Transfer in High Performance Computational Grid Environments. *Parallel Computing Journal (PCJ)*, 28(5) :749–771, May 2002.

[4] G. Aloiso, S. Benkner, H. Bilofsky, Ignacio Blanquer Espert, Vincent Breton, M. Cannataro, I. Chouvarda, B. Claerhout, K. Dean, S. Fiore, Kinda Hassan, G. Heeren, Vicente Hernández García, J. Herveg, M. Hofmann, C. Jones, V. Koutkias, S. Lloyd, G. Lonsdale, V. López, N. Maglaveras, Lydia Maigne, A. Malousi, F. Martin-Sanchez, R. McLatchey, E. Medico, Serge Miguet, M. Mirto, Johan Montagnat, G. De Moor, K. Nozaki, W. De Neve, I. Oliviera, Xavier Pennec, J. Sanchez, T. Solomonides, M. Taillet, P. Veltri, C. De Wagster, and R. Ziegler. HealthGrid White Paper, September 2005.

[5] L. Alvarez, P.-L. Lions, and J.-M. Morel. Image selective smoothing and edge detection by nonlinear diffusion. *S.I.A.M. Numerical Analysis*, 29(3) :845–866, 1992.

[6] T. Andrews, F. Curbera, H. Dholakia, Y. Goland, J. Klein, F. Leymann, K. Liu, D. Roller, D. Smith, S. Thatte, I. Trickovic, and Sanjiva Weerawarana. Business Process Execution Language for Web Services. Technical Report version 1.1, IBM, 2003.

[7] K.P. Andriole, R.L. Morin, R.L. Arenson, J.A. Carrino, B.J. Erickson, S.C. Horii, D.W. Piraino, B.I. Reiner, J.A. Seibert, and E. Siegel. Addressing the Coming Radiology Crisis : The Society for Computer Applications in Radiology SCAR Transforming the Radiological Interpretation Process (TRIP) initiative. *Journal of Digital Imaging (JDI)*, 17(4) :235–243, December 2004.

[8] D. Arnold, S. Agrawal, S. Blackford, J. Dongarra, M. Miller, K. Seymour, K. Sagi, Z. Shi, and S. Vadhiyar. Users' Guide to NetSolve V1.4.1. Technical Report ICL-UT-02-05, University of Tennessee, Knoxville, June 2002.

[9] N. Aspert, D. Santa-Cruz, and T. Ebrahimi. MESH : Measuring Error between Surfaces using the Hausdorff distance. In *IEEE International Conference on Multimedia and Expo (ICME'02)*, volume I, pages 705–708, Lausanne, Switzerland, August 2002.

[10] K.T. Bae, M.L. Giger, C.-T. Chen, and C.E. Khan. Automatic segmentation of liver structure in CT images. *Medical Physics*, 20(1) :71–78, 1993.

[11] C. Bajaj, V. Anupam, D. Schikore, and M. Schikore. Distributed and Collaborative Volume Visualization. *IEEE Computer*, 27(7) :37–43, July 1994.

[12] Christian Barillot, R. Valabregue, J.P. Matsumoto, F. Aubry, H. Benali, Y. Cointepas, O. Dameron, M. Dojat, E. Duchesnay, Bernard Gibaud, S. Kinkingnéhun, D. Papadopoulos, M. Pélégrini-Issac, and E. Simon. NeuroBase : Management of Distributed and Heterogeneous Information Sources inNeuroimaging. In *Distributed Database and processing in Medical Image Computing workshop (DiDaMIC'04)*, Saint Malo, France, September 2004.

[13] F. Bello and A.C.F. Colchester. Measuring Global and Local Spatial Correspondence Using Information Theory. In *Medical Image Computing and Computer-Assisted Intervention (MICCAI'98)*, volume 1496 of *LNCS*, pages 964–973, Cambridge, USA, October 1998. Springer.

[14] Hugues Benoit-Cattin, Fabrice Bellet, Johan Montagnat, and Christophe Odet. Magnetic Resonance Imaging (MRI) Simulation on a Grid Computing Architecture. In *Biogrid'03, proceedings of the IEEE CCGrid03 (Biogrid'03)*, pages 582–587, Tokyo, Japan, May 2003.

[15] J. Bentson, M. Reza, J. Winter, and G. Wilson. Steroids and apparent cerebral atrophy on computed tomography scans. *Journal of Computer Assisted Tomography (JCAT)*, 2(1) :16–23, 1978.

[16] P. Besl and N. McKay. A method for registration of 3D shapes. *IEEE Transactions on Pattern Analysis and Machine Intelligence (PAMI)*, 14(2) :239–256, February 1992.

[17] A. Blake and Andrew Zisserman. *Visual Reconstruction*. MIT Press, 1987.

[18] Ignacio Blanquer Espert, Vicente Hernández García, and J.D. Segrelles Quilis. Creating Virtual Storages and Searching DICOM Medical Images through a GRID Middleware based in OGSA. *Journal of Clinical Monitoring and Computing*, 19(4-5) :295–305, October 2005.

[19] James Blythe, S. Jain, Ewa Deelman, Yolanda Gil, K. Vahi, A. Mandal, and K. Kennedy. Task Scheduling Strategies for Workflow-based Applications in Grids. In *CCGrid*, Cardiff, UK, 2005.

[20] Vincent Breton, Christophe Blanchet, Lydia Maigne, and Johan Montagnat. Grid technology for biomedical applications. In *Vecpar'04*, volume 3402 of *LNCS*, pages 204–218, Valencia, Spain, June 2004. Springer.

[21] Vincent Breton, Raoul Medina, and Johan Montagnat. Datagrid, prototype of a biomedical testbed. In *Conference on synergy between research in medical informatics, bioinformatics and neuro-informatics (SRMIBN01)*, Brussels, Belgium, December 2001.

[22] Vincent Breton, Raoul Medina, and Johan Montagnat. DataGrid, Prototype of a Biomedical Grid. *Methods of Information in Medicine (MIM)*, 42(2) :143–148, 2003.

[23] David Budgen, Mark Turner, Ioannis Kotsiopoulos, Fujun Zhu, Keith Bennett, Pearl Brereton, John Keane, Paul Layzell, Michelle Russell, and Michael Rigby. Managing healthcare information : the role of the broker. In *HealthGrid'05*, Oxford, UK, April 2005.

[24] Nicolas Capit, Georges Da Costa, Yiannis Georgiou, Guillaume Huard, and Cyrille Marti. A batch scheduler with high level components. In *Cluster computing and Grid 2005 (CCGrid'05)*, volume 2, pages 776– 783, May 2005.

[25] Franck Cappello, Frédéric Desprez, Michel Dayde, Emmanuel Jeannot, Yvon Jegou, Stéphane Lanteri, Nouredine Melab, Raymond Namyst, Pascale Vicat-Blanc Primet, Olivier Richard, Eddy Caron, Julien Leduc, and Guillaume Mornet. Grid'5000 : A Large Scale, Reconfigurable, Controlable and Monitorable Grid Platform. In *6th IEEE/ACM International Workshop on Grid Computing (Grid'2005)*, Seattle, Washington, USA, November 2005.

[26] Eddy Caron, Bruno Del-Fabbro, Frédéric Desprez, Emmanuel Jeannot, and Jean-Marc Nicod. Managing Data Persistence in Network Enabled Servers. *Scientific Programming Journal*, 2005.

[27] Eddy Caron and Frédéric Desprez. DIET : A Scalable Toolbox to Build Network Enabled Servers on the Grid. *International Journal of High Performance Computing Applications*, 2005.

[28] Eddy Caron, Frédéric Desprez, Frédéric Lombard, Jean-Marc Nicod, Martin Quinson, and Frédéric Suter. A Scalable Approach to Network Enabled Servers. In *8th International EuroPar Conference*, volume 2400 of *LNCS*, pages 907–910, Paderborn, Germany, August 2002. Springer-Verlag.

[29] Henri Casanova, Arnaud Legrand, Dmitrii Zagorodnov, and Francine Berman. Heuristics for Scheduling Parameter Sweep Applications in Grid Environments. In *9th Heterogeneous Computing Workshop (HCW)*, pages 349–363, Cancun, May 2000.

[30] Henri Casanova, Graziano Obertelli, Francine Berman, and Richard Wolski. The AppleS parameter sweep template : user-level middleware for the grid. In *ACM/IEEE conference on Supercomputing (SC'00)*, page 75, Dallas, USA, 2000.

[31] Arnaud Charnoz. Segmentation du coeur dans des séquences d'images 3D TEMP par ensemble de niveaux. Master's thesis, Université de Nice-Sophia Antipolis, Sophia Antipolis, France, June 2003.

[32] Arnaud Charnoz, Diane Lingrand, and Johan Montagnat. A levelset based method for segmenting the heart in 3D+T gated SPECT images. In Magnin et al. [123], pages 50–59.

[33] Arnaud Charnoz, Diane Lingrand, and Johan Montagnat. Segmentation du coeur dans des séquences 3D TEMP par ensemble de niveaux. In *Orasis*, pages 52–61, Gerardmer, France, May 2003.

[34] David Churches, B. S. Sathyaprakash, Matthew Shields, Ian Taylor, and Ian Wand. A Parallel Implementation of the Inspiral Search Algorithm using Triana. In *Proceedings of the UK e-Science All Hands Meeting*, Nottingham, UK, September 2003.

[35] B. Claerhout and G. De Moor. Privacy Protection for HealthGrid Applications. *Methods of Information in Medicine (MIM)*, 44(2), 2005.

[36] C.M. Clark, G. James, D. Li, J. Oger, D. Paty, and H. Klonoff. Ventricular size, cognitive function and depression in patients with multiple sclerosis. *Canadian Journal of Neurological Sciences*, 19(3) :352–356, August 1992.

[37] J.-P. Cocquerez and S. Philipp. *Analyse d'images : filtrage et segmentation*. MASSON, 1995.

[38] Louis Collins and Alan C. Evans. ANIMAL : validation and applications of nonlinear registration-based segmentation. *International Journal of Pattern Recognition and Artificial Intelligence (IJPRAI)*, 8(11) :1271–1294, 1997.

[39] Louis Collins, Johan Montagnat, A.P. Zijdenbos, and Alan C. Evans. Automated estimation of brain volume in Multiple Sclerosis with BICCR. In *Information Processing in Medical Imaging (IPMPI01)*, Davis, USA, June 2001.

[40] Louis Collins, P. Neelin, M.P. Terrence, and Alan C. Evans. Automatic 3D Intersubject Registration of MR Volumetric Data in Standardized Talairach Space. *Journal of Computer Assisted Tomography (JCAT)*, 2(18) :192–205, 1994.

[41] G. Comi, M. Filippi, V. Martinelli, G. Sirabian, A. Visciani, A. Campi, S. Mammi, M. Rovaris, and N. Canal. Brain magnetic resonance imaging correlates of cognitive impairment in multiple sclerosis. *Journal of the Neurological Sciences (JNS)*, 115 Suppl :S66–73, 1993.

[42] T.F. Cootes, C.J. Taylor, D.H. Cooper, and J. Graham. Active shape models, their training and application. *Computer Vision and Image Understanding (CVIU)*, 61(1) :38–59, January 1995.

[43] Stéphane Cotin. *Modèles anatomiques déformables en temps-réel.* PhD thesis, Université de Nice-Sophia Antipolis, Sophia Antipolis, France, 1997.

[44] Eduardo Davila. Déportations d'algorithmes interactifs de traitement d'images médicales sur des serveurs distants. Master's thesis, INSA, Lyon, France, June 2002.

[45] M. Davis, B. Rezaie, and F. Weiland. Assessment of left ventricular ejection fraction from technetium-99m-methoxy isobutyl isonitrile multiple gated radionucleide angiocardiography. *IEEE Transactions on Medical Imaging (TMI)*, 12(2) :189–199, June 1993.

[46] Thierry Delaitre, Tamás Kiss, Ariel Goyeneche, G. Terstyanszky, S. Winter, and Péter Kacsuk. GEMLCA : Running Legacy Code Applications as Grid Services. *Journal of Grid Computing (JGC)*, 3(1-2), 2005.

[47] Hervé Delingette. *Modélisation, déformation et reconnaissance d'objets tridimensionnels à l'aide de maillages simplexes.* PhD thesis, Ecole Centrale de Paris, Paris, France, July 1994.

[48] Hervé Delingette. General Object Reconstruction based on Simplex Meshes. *International Journal of Computer Vision (IJCV)*, 32(2) :111–146, 1999.

[49] Hervé Delingette, Eric Bardinet, David Rey, J.-D. Lemarchal, Johan Montagnat, Sébastien Ourselin, Alexis Roche, D. Dormont, J. Yelnik, and Nicholas Ayache. YAV++ : a software platform for medical image processing and visualization. In *Workshop on Interactive Medical Image Visualization and Analysis (IMVIA01)*, Utrecht, The Netherlands, October 2001.

[50] Hervé Delingette and Johan Montagnat. General Deformable Model Approach for Model-Based Reconstruction. In *IEEE International Workshop on Model-Based 3D Image Analysis (MB3IA98)*, Bombay, India, January 1998.

[51] Hervé Delingette and Johan Montagnat. New Algorithms for Controlling Active Contours Shape and Topology. In *European Conference on Computer Vision (ECCV00)*, pages 381–395, Dublin, Ireland, June 2000.

[52] Hervé Delingette and Johan Montagnat. Shape and Topology Constraints on Parametric Active Contours. *Computer Vision and Image Understanding (CVIU)*, 83(2), September 2001.

[53] J.L. Dietemann, C. Beigelman, L. Rumbach, M. Vouge, T. Tajahmady, C. Faubert, M.Y. Jeung, and A. Wackenheim. Multiple sclerosis and corpus callosum atrophy : relationship of MRI findings to clinical data. *Neuroradiology*, 30(6) :478–480, 1988.

[54] P. Douek, Didier Revel, S. Chazel, B. Falise, J. Villard, and M. Amiel. Fast MR angiography of the aortoiliac arteries and arteries of thelower extremity : value of bolus-enhanced, whole-volume subtraction technique. *American Journal of Radiology (AJR)*, 165 :431–437, 1995.

[55] R.O. Duda and P.E. Hart. *Pattern Classification and Scene Analysis.* Wiley-Interscience, 1973.

[56] Hector Duque. *Conception et mise en oeuvre d'un environnement logiciel de manipulation et d'accès à des données réparties. Application aux grilles d'images médicales : le système DSEM/DM2.* Ph.D.dissertation, INSA, Lyon, France, July 2005.

[57] Hector Duque, Johan Montagnat, Jean-Marc Pierson, Lionel Brunie, and Isabelle Magnin. DM2 : A Distributed Medical Data Manager for Grids. In *Biogrid'03, proceedings of the IEEE CCGrid03 (Biogrid'03)*, pages 138–147, Tokyo, Japan, May 2003.

[58] M.H. Ellisman, C. Baru, J.S. Grethe, A. Gupta, M. James, Bertram Ludäscher, M.E. Martone, P.M. Papadopoulos, S.T. Peltier, A. Rajasekar, S. Santini, and I.N. Zaslavsky. Biomedical Informatics Research Network : An Overview. In *HealthGrid'05*, Oxford, UK, April 2005.

[59] D. Ferraiolo and D. Kuhn. Role Based Access Control. In *NIST-NCSC National Computer Security Conference*, pages 554–563, 1992.

[60] E. Fisher, R. Rudick, J.A. Tkach, J.-C. Lee, T.J. Masaryk, J. Simon, J.F. Cornhill, and J.A. Cohen. Automated Calculation of Whole Brain Atrophy from Magenetic Resonance Images for Monitoring Multiple Sclerosis. *Neurology*, 52(6 :Suppl 2) :A352–3, April 1999.

[61] Leonardo Flórez-Valencia, Johan Montagnat, and Maciej Orkisz. 3D graphical models for vascular-stent pose simulation. In *International Conference on Computer Vision and Graphics (ICCVG02)*, Zakopane, Poland, September 2002.

[62] Leonardo Flórez-Valencia, Johan Montagnat, and Maciej Orkisz. 3D graphical models for vascular-stent pose simulation. *Machine Graphics and Vision (MGV)*, 13(3) :235–248, 2004.

[63] Ian Foster. Globus Toolkit Version 4 : Software for Service-Oriented Systems. In *International Conference on Network and Parallel Computing (IFIP)*, volume 3779, pages 2–13. Springer-Verlag LNCS, 2005.

[64] Ian Foster, Carl Kesselman, J. Nick, and S. Tuecke. The Physiology of the Grid : An Open Grid Services Architecture for Distributed Systems Integration. Technical report, Open Grid Service Infrastructure WG, GGF, June 2002.

[65] Ian Foster, Carl Kesselman, G. Tsudik, and S. Tuecke. A Security Architecture for Computational Grids. In *ACM Conference on Computer and Communications Security (CCCS)*, pages 83–92, San Francisco, CA, USA, 1998.

[66] Ian Foster, Carl Kesselman, and S. Tuecke. The Anatomy of the Grid : Enabling Scalable Virtual Organizations. *International Journal of Supercomputer Applications*, 5(3), 2001.

[67] N.C. Fox and P.A. Freeborough. Brain atrophy progression measured from registered serial MRI : validation and application to Alzheimer's disease. *Journal of Magnetic Resonance Imaging (JMRI)*, 6(7) :1069–1075, 1997.

[68] N.C. Fox, E.K. Warrington, P.A. Freeborough, P. Hartikainen, A.M. Kennedy, J.M. Stevens, and M.N. Rossor. Presymptomatic hippocampal atrophy in Alzheimer's disease. A longitudinal MRI study. *Brain*, 119(Pt 6) :2001–7, 1996.

[69] A.F. Frangi, W.J. Niessen, R.M. Hoogeveen, T. Walsum, and M.A. Viergever. Modeled-based quantification of 3D magnetic resonance angiographic images. *IEEE Transactions on Medical Imaging (TMI)*, 18(10) :946–956, 1999.

[70] A.F. Frangi, W.J. Niessen, and M.A. Viergever. Three-Dimensional Modeling for Functional Analysis of Cardiac Images : A Review. *IEEE Transactions on Medical Imaging (TMI)*, 20(1) :2–25, 2001.

[71] Nathalie Furmento, A. Mayer, S. McGough, S. Newhouse, T. Field, and J. Darlington. ICENI : Optimisation of component applications within a Grid environment. *Journal of Parallel Computing*, 28(12) :1753–1772, 2002.

[72] Gido Gerig, O. Kübler, Ron Kikinis, and F.A. Jolesz. Nonlinear Anisotropic Filtering of MRI Data. *IEEE Transactions on Medical Imaging (TMI)*, 11(2) :221–232, June 1992.

[73] Cécile Germain, Vincent Breton, Patrick Clarysse, Yann Gaudeau, Tristan Glatard, Emmanuel Jeannot, Yannick Legré, Charles Loomis, Isabelle Magnin, Johan Montagnat, Jean-Marie Moureaux, Angel Osorio, Xavier Pennec, and Romain Texier. Grid-enabling

medical image analysis. *Journal of Clinical Monitoring and Computing*, 19(4–5) :339–349, October 2005.

[74] Cécile Germain, Vincent Breton, Patrick Clarysse, Yann Gaudeau, Tristan Glatard, Emmanuel Jeannot, Yannick Legré, Charles Loomis, Johan Montagnat, Jean-Marie Moureaux, Angel Osorio, Xavier Pennec, and Romain Texier. Grid-enabling medical image analysis. In *proceedings of the IEEE/ACM International Symposium on Cluster Computing and the Grid (Biogrid'05)*, Cardiff, UK, May 2005.

[75] Tristan Glatard. Indexation d'images médicales basée sur le contenu : application à la recherche et à la segmentation d'images. Master's thesis, Ecole Doctorale EEA, Lyon, France, September 2004.

[76] Tristan Glatard, David Emsellem, and Johan Montagnat. Generic web service wrapper for efficient embedding of legacy codes in service-based workflows. In *Grid-Enabling Legacy Applications and Supporting End Users Workshop (GELA'06)*, Paris, France, June 2006.

[77] Tristan Glatard, Johan Montagnat, Diane Lingrand, and Xavier Pennec. Flexible and efficient workflow deployement of data-intensive applications on grids with MOTEUR. *International Journal of High Performance Computing and Applications (IJHPCA)*, 2007.

[78] Tristan Glatard, Johan Montagnat, and Isabelle Magnin. Texture based medical image indexing and retrieval : application to cardiac imaging. In *Proceedings of ACM Multimedia 2004, workshop on Multimedia Information Retrieval (MIR)*, New York, NY, USA, October 2004.

[79] Tristan Glatard, Johan Montagnat, and Xavier Pennec. An optimized workflow enactor for data-intensive grid applications. Technical Report I3S/RR-2005-32-, I3S, Sophia Antipolis, France, October 2005.

[80] Tristan Glatard, Johan Montagnat, and Xavier Pennec. Grid-enabled workflows for data intensive medical applications. In *18th IEEE International Symposium on Computer-Based Medical Systems (CBMS)*, June 2005.

[81] Tristan Glatard, Johan Montagnat, and Xavier Pennec. An experimental comparison of Grid5000 clusters and the EGEE grid. In *Workshop on Experimental Grid testbeds for the assessment of large-scale distributed applications and tools (EXPGRID'06)*, Paris, France, June 2006.

[82] Tristan Glatard, Johan Montagnat, and Xavier Pennec. Efficient services composition for grid-enabled data-intensive applications. In *IEEE International Symposium on High Performance Distributed Computing (HPDC'06)*, Paris, France, June 2006.

[83] Tristan Glatard, Johan Montagnat, and Xavier Pennec. Medical image registration algorithms assesment : Bronze Standard application enactment on grids using the MOTEUR workflow engine. In *HealthGrid conference (HealthGrid'06)*, pages 93–103, Valencia, Spain, June 2006. IOS Press.

[84] Tristan Glatard, Johan Montagnat, and Xavier Pennec. Probabilistic and dynamic optimization of job partitioning on a grid infrastructure. In *14th euromicro conference on Parallel, Distributed and network-based Processing (PDP06)*, pages 231–238, Montbéliard-Sochaux, France, February 2006.

[85] Tristan Glatard, Johan Montagnat, Xavier Pennec, David Emsellem, and Diane Lingrand. MOTEUR : a data-intensive service-based workflow manager. Technical Report I3S/RR-2006-07-FR, I3S, Sophia Antipolis, France, March 2006.

[86] Tristan Glatard, Xavier Pennec, and Johan Montagnat. Performance evaluation of grid-enabled registration algorithms using bronze-standards. In *Medical Image Computing and*

Computer-Assisted Intervention (MICCAI'06), LNCS, Copenhagen, Denmark, October 2006.

[87] Tristan Glatard, Gergely Sipos, Johan Montagnat, Zoltán Farkas, and Péter Kacsuk. *Workflow Level Parametric Study Support by MOTEUR and the P-GRADE Portal*, chapter 18. Springer, January 2007.

[88] C.R. Guttmann, F.A. Jolesz, Ron Kikinis, R.J. Killiany, M.B. Moss, T. Sandor, and M.S. Albert. White matter changes with normal aging. *Neurology*, 50(4) :972–978, April 1998.

[89] U. Hageleit, C.H. Will, and D. Seidel. Automated measurements of cerebral atrophy in multiple sclerosis. *Neurosurgery Rev*, 10(2) :137–140, 1987.

[90] T. Hagerup. Allocating Independent Tasks to Parallel Processors : An Experimental Study. *Journal of Parallel and Distributed Computing (JPDC)*, 47 :185–197, 1997.

[91] Andrew Harrison and Ian Taylor. Dynamic Web Service Deployment Using WSPeer. In *Proceedings of 13th Annual Mardi Gras Conference - Frontiers of Grid Applications and Technologies*, pages 11–16, February 2005.

[92] S. Hastings, S. Oster, S. Langella, T.M. Kurc, T. Pan, U.V. Catalyurek, and J.H. Saltz. A Grid-based image archival and analysis system. *Journal of the American Medical Informatics Association (JAMIA)*, 12 :286–295, January 2005.

[93] J. Hatazawa, M. Ito, H. Yamaura, and T. Matsuzawa. Sex difference in brain atrophy during aging ; a quantitative study with computed tomography. *Journal of American Geriatry Society*, 30(4) :235–239, 1982.

[94] Marcela Hernández-Hoyos. *Segmentation anisotrope 3D pour la quantification en imagerie vasculaire par résonance magnétique*. PhD thesis, INSA, Lyon, France, July 2002.

[95] Marcela Hernández-Hoyos, A. Anwander, Maciej Orkisz, Jean-Pierre Roux, P. Douek, and Isabelle Magnin. A Deformable Vessel Model with Single Point Initialization for Segmentation, Quantification and Visualization of Blood Vessels in 3D MRA. In *Medical Image Computing and Computer-Assisted Intervention (MICCAI'00)*, volume 1935 of *LNCS*, pages 735–745, Pittsburgh, USA, October 2000. Springer.

[96] Marcela Hernández-Hoyos, Maciej Orkisz, P. Puech, Catherine Mansard, and Isabelle Magnin. Computer-assisted analysis of three-dimensional angiograms. *RadioGraphics*, 22 :421–436, 2002.

[97] Derek Hill, Xavier Pennec, Michael Burns, Michael Parkin, Jo Hajnal, Radu Stefanescu, Daniel Rueckert, and Johan Montagnat. Intraoperable Medical Image Registration Grid Service. In *HealthGrid*, Clermont-Ferrand, France, January 2004.

[98] H.K. Huang. *PACS : Picture Archiving and Communication Systems in Biomedical Imaging*. Hardcover, 1996.

[99] Yan Huang, Ian Taylor, David M. Walker, and Robert Davies. Wrapping Legacy Codes for Grid-Based Applications. In *17th International Parallel and Distributed Processing Symposium (IPDPS)*, page 139. IEEE Computer Society, 2003.

[100] S.J. Huber, G.W. Paulson, E.C. Shuttleworth, D. Chakeres, L.E. Clapp, A. Pakalnis, K. Weiss, and K. Rammohan. Magnetic resonance imaging correlates of dementia in multiple sclerosis. *Archive of Neurology*, 44(7) :732–736, 1987.

[101] Romin Irani and S. Jeelani Bashna. *AXIS : Next Generation Java SOAP*. Wrox Press, May 2002.

[102] P. Jannin, J.M. Fitzpatrick, D.J. Hawkes, Xavier Pennec, R. Shahidi, and M.W. Vannier. Validation of Medical Image Processing in Image-guided Therapy. *IEEE Transactions on Medical Imaging (TMI)*, 21(12) :1445–1449, December 2002.

[103] Péter Kacsuk, Gábor Dózsa, József Kovács, Róbert Lovas, Norbert Podhorszki, Zoltán Balaton, and Gabor Gombás. P-GRADE : A Grid Programing Environment. *Journal of Grid Computing (JGC)*, 1(2) :171–197, 2003.

[104] Péter Kacsuk, Ariel Goyeneche, Thierry Delaitre, Tamás Kiss, Zoltán Farkas, and Tamás Boczko. High-Level Grid Application Environment to Use Legacy Codes as OGSA Grid Services. In *Proceedings of the Fifth IEEE/ACM International Workshop on Grid Computing (GRID '04)*, pages 428–435, Washington DC, USA, 2004. IEEE Computer Society.

[105] Toivo Katila, Isabelle Magnin, Patrick Clarysse, Johan Montagnat, and Jukka Nenonen, editors. *First International Workshop on Functional Imaging and Modeling of the Heart*, volume LNCS 2230, Finland, November 2001.

[106] G. Kecskemeti, Y. Zetuny, Tamás Kiss, Gergely Sipos, Péter Kacsuk, G. Terstyanszky, and S. Winter. Automatic deployment of Interoperable Legacy Code Services. In *UK e-Science All Hands Meeting*, Nottingham, UK, September 2005.

[107] Rida Khatoun. Distribution de données et de méta-données médicales. Master's thesis, Université de Nice-Sophia Antipolis, Sophia Antipolis, France, September 2004.

[108] K.H. Kim. APIs for Real-Time Distributed Object Programming. *IEEE Computer*, 33(6) :72–80, June 2000.

[109] K. Kohlmeyer, G. Lehmkuhl, and F. Poutska. Computed tomography of anorexia nervosa. *American Journal of Neuroradiology (AJNR)*, 4(3) :437–8, 1983.

[110] J.F. Kurtzke. Rating neurologic impairment in multiple sclerosis : an expanded disability status scale (EDSS). *Neurology*, 83 :1444–1452, 1983.

[111] W. Lefer and Jean-Marc Pierson. A Thin Client Archietcture for Data Visualization on the World Wide Web. In *International Conference on Visual Computing (ICVC'99)*, Goa, India, February 1999.

[112] Vincent Léon. Optimisation du flot d'exécution d'applications manipulant des masses de données sur grilles de calcul. Master's thesis, Ecole Polytechnique Universitaire, Sophia Antipolis, France, September 2006.

[113] Jianzhi Li, Zhuopeng Zhang, and Hongji Yang. A Grid Oriented Approach to Reusing Legacy Code in ICENI Framework. In *IEEE International Conference on Information Reuse and Integration (IRI'05)*, pages 464– 469, Las Vegas, Nevada, USA, August 2005.

[114] Diane Lingrand, Arnaud Charnoz, Malik Koulibaly, Jacques Darcourt, and Johan Montagnat. Toward accurate segmentation of the LV myocardium and chamber for volumes estimation in gated SPECT sequences. In Tomas Pajdla and Jiri Matas, editors, *European Conference on Computer Vision*, volume LNCS 3024, pages 267–278, Prague (Czech Republic), May 2004. Springer.

[115] Diane Lingrand and Johan Montagnat. Levelset and B-spline deformable model techniques for image segmentation : a pragmatic comparative study. In Heikki Kalviainen, Jussi Parkkinen, and Arto Kaarna, editors, *14th Scandinavian Conference on Image Analysis*, volume LNCS 3540, pages 25–34, Joennsu, Finland, June 2005. Springer.

[116] Diane Lingrand, Johan Montagnat, Louis Collins, and Jean Gotman. Compensating Small Head Displacements for an accurate fMRI Registration. In Ivar Austvoll, editor, *Scandinavian Conference on Image Analysis*, pages 10–16, Bergen, Norway, June 2001.

[117] L.A. Loizou, E.B. Rolfe, and H. Hewazy. Cranial computed tomography in the diagnosis of multiple sclerosis. *Journal of Neurological Neurosurgie and Psychiatry (JNNP)*, 45(10) :905–921, 1982.

[118] Phillip Lord, Pinar Alper, Chris Wroe, and Carole Goble. Feta : A light-weight architecture for user oriented semantic service discovery. In *European Semantic Web Conference*, 2005.

[119] N.A. Losseff, L. Wang, H.M. Lai, D.S. Yoo, M.L. Gawne-Cain, W.I. McDonald, D.H. Miller, and A.J. Thompson. Progressive cerebral atrophy in multiple sclerosis. A serial MRI study. *Brain*, 119(Pt 6) :2009–2019, December 1996.

[120] Bertram Ludäscher, Ikay Altintas, Chad Berkley, Dan Higgins, Efrat Jaeger, Matthew Jones, Edward A. Lee, Jing Tao, and Yang Zhao. Scientific Workflow Management and the Kepler System. *Concurrency and Computation : Practice &amp ; Experience*, 2005.

[121] D. MacDonald, D. Avis, and Alan C. Evans. Multiple Surface Identification and Matching in Magnetic Resonance Images. In *Visualization in Biomedical Computing (VBC'94)*, volume 2359, pages 160–168, Rochester, USA, October 1994. SPIE.

[122] Isabelle Magnin, Patrick Clarysse, Johan Montagnat, Jukka Nenonen, and Toivo Katila. MedIA special issue on Functional Imaging and Modeling of the Heart. *Medical Image Analysis (MedIA)*, 7(3), September 2003.

[123] Isabelle Magnin, Johan Montagnat, Patrick Clarysse, Jukka Nenonen, and Toivo Katila, editors. *Second International Workshop on Functional Imaging and Modeling of the Heart*, volume LNCS 2674, Lyon, France, June 2003. Springer.

[124] Sorin Manolache, Petru Eles, and Zebo Peng. Memory and Time-Efficient Schedulability Analysis of Task Sets with Stochastic Execution Time. In *Euromicro Conference on Real-Time Systems*, Delft, The Netherlands, 2001.

[125] J.C. Mazziotta, Arthur W. Toga, Alan C. Evans, P. Fox, and J. Lancaster. A probabilistic atlas of the human brain : theory and rationale for its development. The International Consortium for Brain Mapping. *NeuroImage*, 2(2) :89–101, 1995.

[126] T. McInerney and D. Terzopoulos. Deformable models in medical image analysis : a survey. *Medical Image Analysis (MedIA)*, 1(2) :91–108, 1996.

[127] A.R. Mellanby and M.A. Reveley. Effects of acute dehydration on computerized tomographic assessment of cerebral density and ventricular volume. *Lancet*, 8303(2) :874, 1982.

[128] A.K. Miller, R.L. Alston, and J.A. Corsellis. Variation with age in the volumes of grey and white matter in the cerebral hemispheres of man : measurements with an image analyser. *Neuropathologie Applied Neurobiology*, 6(2) :119–132, 1980.

[129] J.R. Mitchell, S.J. Karlik, D.H. Lee, M. Eliasziw, G.P. Rice, and A. Fenster. Quantification of multiple sclerosis lesion volumes in 1.5 and 0.5T anisotropically filtered and unfiltered MR exams. *Medical Physics*, 23 :115–126, 1996.

[130] Johan Montagnat. Segmentation d'image médicales volumiques à l'aide de maillages déformables contraints. Master's thesis, École Normale Supérieure de Cachan, Cachan, France, September 1996.

[131] Johan Montagnat. *Modèles déformables pour la segmentation et la modélisation d'images médicales 3D et 4D.* Phd thesis, Nice-Sophia Antipolis University, Sophia Antipolis, France, December 1999.

[132] Johan Montagnat, Fabrice Bellet, Hugues Benoit-Cattin, Vincent Breton, Lionel Brunie, Hector Duque, Yannick Legré, Isabelle Magnin, Lydia Maigne, Serge Miguet, Jean-Marc Pierson, Ludwig Seitz, and T. Tweed. Medical images simulation, storage, and processing on the european datagrid testbed. *Journal of Grid Computing (JGC)*, 2(4) :387–400, December 2004.

[133] Johan Montagnat, Vincent Breton, and Isabelle Magnin. Using grid technologies to face medical image analysis challenges. In *Biogrid'03, proceedings of the IEEE CCGrid03 (Biogrid'03)*, pages 588–593, Tokyo, Japan, May 2003.

[134] Johan Montagnat, Vincent Breton, and Isabelle Magnin. Partitionning medical image databases for content-based queries on a grid. *Methods of Information in Medicine (MIM)*, 44(2) :154–160, 2005.

[135] Johan Montagnat, Patrick Clarysse, Jukka Nenonen, Toivo Katila, and Isabelle Magnin. MedIA special issue on Functional Imaging and Modeling of the Heart. *Medical Image Analysis (MedIA)*, 9(4), August 2005.

[136] Johan Montagnat, Eduardo Davila, and Isabelle Magnin. 3D objects visualization for remote interactive medical applications. In *3D Data Visualization, Processing, and Transmission (3DPVT02)*, pages 75–78, Padova, Italy, June 2002.

[137] Johan Montagnat, Eduardo Davila, and Isabelle Magnin. Efficient visualization of 3D medical scenes for remote interactive applications. In *Image and Signal Processing and Analysis (ISPA)*, Roma, Italy, September 2003.

[138] Johan Montagnat and Hervé Delingette. A Hybrid Framework for Surface Registration and Deformable Models. In *proceedings of Computer Vision and Pattern Recognition (CVPR97)*, pages 1041–1046, San Juan, Puerto Rico, June 1997.

[139] Johan Montagnat and Hervé Delingette. Reconstruction surfacique et segmentation robuste à base de maillages déformables. In *sixièmes journées Orasis (ORASIS97)*, La Colle sur Loup, France, October 1997.

[140] Johan Montagnat and Hervé Delingette. Volumetric Medical Images Segmentation using Shape Constrained Deformable Models. In *Proceedings of Computer Vision Virtual reality and Robotics in Medicine (CVRMed97)*, volume 1205 of *LNCS*, pages 13–22, Grenoble, France, March 1997. Springer-Verlag.

[141] Johan Montagnat and Hervé Delingette. Globally constrained deformable models for 3D object reconstruction. *Signal Processing (SP)*, 71(2) :173–186, 1998.

[142] Johan Montagnat and Hervé Delingette. Space and Time Shape Constrained Deformable Surfaces for 4D Medical Image Segmentation. In *international conference on Medical Image Computing and Computer Assisted Intervention (MICCAI00)*, pages 196–205, Pittsburgh, PA, USA, October 2000.

[143] Johan Montagnat and Hervé Delingette. Spatial and Temporal Shape Constrained Deformable Surfaces for 3D and 4D Medical Image Segmentation. Technical Report 4078, INRIA, Sophia Antipolis, France, November 2000.

[144] Johan Montagnat and Hervé Delingette. Topology and shape constraints on parametric active contours. Technical Report 3880, INRIA, Sophia Antipolis, France, January 2000.

[145] Johan Montagnat and Hervé Delingette. 4D deformable models with temporal constraints : application to 4D cardiac image segmentation. *Medical Image Analysis (MedIA)*, 9(1) :87–100, February 2005.

[146] Johan Montagnat, Hervé Delingette, and Nicholas Ayache. A review of deformable surfaces : topology, geometry and deformation. *Image and Vision Computing (IVC)*, 19(14) :1023–1040, December 2001.

[147] Johan Montagnat, Hervé Delingette, and Grégoire Malandain. Cylindrical Echocadiographic Image Segmentation Based on 3D Deformable Models. In *International conference on Medical Image Computing and Computer Assisted Intervention (MICCAI99)*, LNCS, pages 168–175, Cambridge, UK, September 1999. Springer-Verlag.

[148] Johan Montagnat, Hervé Delingette, Nicolas Scapel, and Nicholas Ayache. Representation, Shape, Topology and Evolution of Deformable Surfaces. Application to 3D Medical Image Segmentation. Technical Report 3954, INRIA, Sophia Antipolis, France, May 2000.

[149] Johan Montagnat, Hervé Delingette, Nicolas Scapel, and Nicholas Ayache. Surface Simplex Meshes for 3D Medical Image Segmentation. In *International Conference on Robotics and Automation (ICRA00)*, San Francisco, CA, USA, April 2000.

[150] Johan Montagnat, Hector Duque, Jean-Marc Pierson, Vincent Breton, Lionel Brunie, and Isabelle Magnin. Medical Image Content-Based Queries using the Grid. In *HealthGrid'03*, pages 138–147, Lyon, France, January 2003.

[151] Johan Montagnat, Tristan Glatard, and Diane Lingrand. Data composition patterns in service-based workflows. In *Workshop on Workflows in Support of Large-Scale Science (WORKS'06)*, Paris, France, June 2006.

[152] Johan Montagnat, Daniel Jouvenot, Christophe Pera, Ákos Frohner, Peter Kunszt, Birger Koblitz, Nuno Santos, and Charles Loomis. Bridging clinical information systems and grid middleware : a Medical Data Manager. In *HealthGrid conference (HealthGrid'06)*, pages 14–24, Valencia, Spain, June 2006. IOS Press.

[153] Johan Montagnat, Isabelle Magnin, and Vincent Breton. Medical image databases content-based queries partitioning on a grid. In *HealthGrid'04*, Clermont-Ferrand, France, June 2004.

[154] Johan Montagnat, Maxime Sermesant, Hervé Delingette, Grégoire Malandain, and Nicholas Ayache. 4D Cylindrical Echocardiographic Images Anisotropic Filtering for Model Based Segmentation. *Pattern Recognition Letters (PRL)*, 24(4-5) :815–828, February 2003.

[155] Hidemoto Nakada, Satoshi Matsuoka, K. Seymour, J. Dongarra, C. Lee, and Henri Casanova. A GridRPC Model and API for End-User Applications. Technical report, Global Grid Forum (GGF), July 2005.

[156] Stéphane Nicolau, Xavier Pennec, Luc Soler, and Nicholas Ayache. Evaluation of a New 3D/2D Registration Criterion for Liver Radio-Frequencies Guided by Augmented Reality. In *International Symposium on Surgery Simulation and Soft Tissue Modeling (IS4TM'03)*, volume 2673 of *LNCS*, pages 270–283, Juan-les-Pins, France, 2003. INRIA Sophia Antipolis, Springer-Verlag.

[157] Tom Oinn, Matthew Addis, Justin Ferris, Darren Marvin, Martin Senger, Mark Greenwood, Tim Carver, Kevin Glover, Matthew R. Pocock, Anil Wipat, and Peter Li. Taverna : A tool for the composition and enactment of bioinformatics workflows. *Bioinformatics journal*, 17(20) :3045–3054, 2004.

[158] S. Ordas, H.C. van Hassen, J. Puente, B.P.F. Lelieveldt, and A.F. Frangi. Parametric optimization of a model-based segmentation algorithm forcardiac MR image analysis : A grid-computing approach. In *HealthGrid'05*, pages 146–156, Oxford, UK, April 2005.

[159] L. Pearlman, W. Welch, Ian Foster, Carl Kesselman, and S. Tuecke. A Community Authorization Service for Group Collaboration. In *IEEE Workshop on Policies for Distributed Systems and Networks (WPDSN'02)*, 2002.

[160] Xavier Pennec, R. G. Guttman, and Jean-Philippe Thirion. Feature-Based Registration of Medical Images : Estimation and Validation of the Pose Accuracy. In *Medical Image Computing and Computer-Assisted Intervention (MICCAI98)*, volume 1496 of *LNCS*, pages 1107–1114, Cambridge, USA, October 1998. Springer.

[161] P. Perona and J. Malik. Scale-Space And Edge Detection Using Anisotropic Diffusion. *IEEE Transactions on Pattern Analysis and Machine Intelligence (PAMI)*, 12 :629–639, 1990.

[162] A. Pfefferbaum, D.H. Mathalon, E.V. Sullivan, J.M. Rawles, R.B. Zipursky, and K.O. Lim. A quantitative magnetic resonance imaging study of changes in brain morphology from infancy to late adulthood. *Archives of Neurology*, 51(9) :874–887, 1994.

[163] A. Pfefferbaum, E.V. Sullivan, D.H. Mathalon, P.K. Shear, M.J. Rosenbloom, and K.O. Lim. Longitudinal changes in magnetic resonance imaging brain volumes in abstinent and relapsed alcoholics. *Alcoholic Clinical Exp Res*, 19(5) :1177–1191, 1995.

[164] Jean-Marc Pierson, Ludwig Seitz, Hector Duque, and Johan Montagnat. MetaData for Efficient, Secure and Extensible Access to Data in a Medical Grid. In *Database and Expert System Applications (DEXA'04)*, pages 562–566, Zaragoza, Spain, September 2004.

[165] A. Pitiot, Hervé Delingette, Nicholas Ayache, and P.M. Thompson. Expert-Knowledge-Guided Segmentation System for Brain MRI. In *Medical Image Computing and Computer-Assisted Intervention (MICCAI'03)*, volume 2879 of *LNCS*, pages 644–652, Montreal, Québec, Canada, November 2003. Springer.

[166] Luc Pronzato. Optimal and asymptotically optimal decision rules for sequential screening and resource allocation. *IEEE Transactions on Automatic Control (TAC)*, 46(5) :687–697, 2001.

[167] Bo Qiu, Patrick Clarysse, Marc Janier, Johan Montagnat, and Didier Vray. Comparison of 3D Deformable Models For in vivo Measurements of Mouse Embryo from 3D Ultrasound Images. In *IEEE International Ultrasonics, Ferroelectrics, and Frequency Control (UFFC'04)*, Montreal, Québec, Canada, August 2004.

[168] S.M. Rao, S. Glatt, T.A. Hammeke, M.P. McQuillen, B.O. Khatri, A.M. Rhodes, and S. Pollard. hronic progressive multiple sclerosis. Relationship between cerebral ventricular size and neuropsychological impairment. *Archive of Neurology*, 42(7) :678–682, 1985.

[169] C.P. Renaudin, B. Barbier, R. Roriz, Didier Revel, and M. Amiel. Coronary arteries : new design for three-dimensional arterial phantom. *Radiology*, 190 :579–582, 1994.

[170] M.A. Ron, W. Acker, G.K. Shaw, and W.A. Lishman. Computerized tomography of the brain in chronic alcoholism : a Survey and follow-up study. *Brain*, 105(Pt 3) :497–514, September 1982.

[171] R. Ronfard. Region-Based Strategies for Active Contour Models. *International Journal of Computer Vision (IJCV)*, 13(2) :229–251, 1994.

[172] Robert Rönngren, M. Liljenstam, Johan Montagnat, and Rassul Ayani. A Comparative Study of State Saving Mecanisms for Time Warp Synchronized Parallel Discrete Event Simulation. In *Proceedings of the 29th Annual Simulation Symposium (ASS99)*, New Orleans, USA, March 1996.

[173] Robert Rönngren, M. Liljenstam, Johan Montagnat, and Rassul Ayani. Transparent Incremental State Saving in Time Warp Parallel Discrete Event Simulation. In *Proceedings of Parallel And Distributed Simulation (PADS96)*, pages 70–77, Philadelphia, USA, May 1996.

[174] R. Rudick, E. Fisher, C. Lee, J. Simon, and L. Jacobs. Use of the brain parenchymal fraction to measure whole brain atrophy in relapsing-remitting MS. *Neurology*, 53 :1698–1704, 1999.

[175] R. Sandberg, D Goldberg, S. Kleiman, D. Walsh, and B. Lyon. Design and Implementation of the Sun Network File System. In *USENIX Conference*, Berkeley, CA, 1985.

[176] Nuno Santos and Birger Koblitz. Metadata services on the grid. In *Advanced Computing and Analysis Techniques (ACAT'05)*, Berlin, Germany, May 2005.

[177] C. Schmidt and F. Kuhns. An overview of the Real-Time CORBA Specification. *IEEE Computer*, 33(6) :56–63, June 2000.

[178] Jennifer Schopf and Francine Berman. Stochastic Scheduling. In *Supercomputing (SC'99)*, Portland, USA, 1999.

[179] M. Schwartz, H. Creasey, C.L. Grady, J.M. DeLeo, H.A. Frederickson, N.R. Cutler, and S.I. Rapoport. Computed tomographic analysis of brain morphometrics in 30 healthy men, aged 21 to 81 years. *Annals of Neurology*, 17(2) :146–157, 1985.

[180] W.P. Segars, D.S. Lalush, and B.M.W. Tsui. A Realistic Spline-Based Dynamic Heart Phantom. *IEEE Transactions on Nuclear Science (TNS)*, 46(3) :503–506, 1999.

[181] Ludwig Seitz. *Conception et mise en oeuvre de mécanismes sécurisés d'échange de données confidentielles ; application à la gestion de données biomédicales dans le cadre d'architectures de grille de calcul/données*. PhD thesis, INSA, Lyon, France, July 2005.

[182] Ludwig Seitz, Johan Montagnat, Jean-Marc Pierson, Didier Oriol, and Diane Lingrand. Authentication and autorisation prototype on the microgrid for medical data management. In *HealthGrid*, Oxford, UK, April 2005.

[183] Ludwig Seitz, Jean-Marc Pierson, and Lionel Brunie. Key management for encrypted data storage in distributed systems. In *IEEE Security in Storage Workshop (SISW'03)*, Washington DC, USA, October 2003.

[184] Ludwig Seitz, Jean-Marc Pierson, and Lionel Brunie. Semantic Access Control for Medical Applications in Grid Environments. In *EuroPar*, volume 2790 of *LNCS*, pages 374–383. Springer, 2003.

[185] Ludwig Seitz, Jean-Marc Pierson, and Lionel Brunie. Encrypted Storage of Medical Data on a Grid. *Methods of Information in Medicine (MIM)*, 44(2), 2005.

[186] Martin Senger, Peter Rice, and Tom Oinn. Soaplab - a unified Sesame door to analysis tool. In *UK e-Science All Hands Meeting*, pages 509–513, Nottingham, September 2003.

[187] Maxime Sermesant. Diffusion anisotrope et segmentation par modèles déformables sur des images échographiques 4D du coeur. Master's thesis, Ecole Normale Supérieure de Cachan, Cachan, France, July 1999.

[188] Maxime Sermesant, Olivier Clatz, Z. Li, Stéphane Lanteri, Hervé Delingette, and Nicholas Ayache. A Parallel Implementation of Non-Rigid Registration Using a Volumetric Biomechanical Model. In *Workshop on Biomedical Image Registration (WBIR'03)*, volume 2717 of *LNCS*, pages 398–407, Philadelphia, PA, USA, 2003. Springer.

[189] J.G. Sled, A.P. Zijdenbos, and Alan C. Evans. A non-parametric method for automatic correction of intensity non-uniformity in MRI data. *IEEE Transactions on Medical Imaging (TMI)*, 17(1) :87–97, 1998.

[190] S.M. Smith, N. De Stefano, M. Jenkinson, and P.M. Matthews. Measurement of Brain Change Over Time. Technical Report TR00SMS1, Center for Functional MR Imaging of the brain, Oxford, UK, 2000.

[191] Luc Soler. *Une nouvelle méthode de segmentation des structures anatomiques et pathologiques : application aux angioscanners 3D du foie pour la planification chirurgicale.* PhD thesis, Université de Nice-Sophia Antipolis, Sophia Antipolis, France, 1998.

[192] Luc Soler, Hervé Delingette, Grégoire Malandain, Johan Montagnat, Nicholas Ayache, C. Koehl, Olivier Dourthe, B. Malassagne, M. Smith, D. Mutter, and J. Marescaux. Fully automatic anatomical, pathological, and functional segmentation from CT scans for hepatic surgery. *Computer Aided Surgery (CAS)*, 6(3), August 2001.

[193] Yoshio Tanaka, Hidemoto Nakada, Satoshi Sekiguchi, Toyotaro Suzumura, and Satoshi Matsuoka. Ninf-G : A Reference Implementation of RPC-based Programming Middleware for Grid Computing. *Journal of Grid Computing (JGC)*, 1(1) :41–51, 2003.

[194] Ian Taylor, Matthew Shields, Ian Wand, and Roger Philp. Grid Enabling Applications Using Triana. In *Workshop on Grid Applications and Programming Tools*. Held in Conjunction with GGF8, 2003.

[195] Ian Taylor, Ian Wand, Matthew Shields, and Shalil Majithia. Distributed computing with Triana on the Grid. *Concurrency and Computation : Practice &amp ; Experience*, 17(1–18), 2005.

[196] P.M. Thompson, C. Vidal, J.N. Giedd, P. Gochman, J. Blumenthal, R. Nicolson, Arthur W. Toga, and J.L. Rapoport. Mapping Adolescent Brain Change Reveals Dynamic Wave of Accelerated Gray Matter Loss in Very Early-Onset Schizophrenia. *National Academy of Sciences of the USA*, 98(20) :11650–55, September 2001.

[197] Robert A. Van Engelen and Kyle A. Gallivan. The gSOAP Toolkit for Web Services and Peer-to-Peer Computing Networks. In *Proceedings of the 2nd IEEE/ACM International Symposium on Cluster Computing and the Grid (CCGRID '02)*, page 128, Washington DC, USA, 2002. IEEE Computer Society.

[198] Sébastien Varrette, Sébastien Georget, Johan Montagnat, Jean-Louis Roch, and Franck Leprevost. Distributed Authentication in GRID5000. In *Grid Computing and its Application to Data Analysis (GADA'05)*, volume LNCS 3762, pages 314–326, Agia Napa, Cyprus, November 2005. Springer.

[199] Pascale Vicat-Blanc Primet, Johan Montagnat, and Fabien Chanussot. Flexible and Dynamic Control of Network QoS in Grid environments : the QoSINUS approach. In *Cluster 2004*, San Diego, CA, USA, September 2004.

[200] Didier Vray, A. Discher, J. Lefloch, W. Mai, Patrick Clarysse, Q.C. Pham, Johan Montagnat, and Marc Janier. 3D Quantification of Ultrasound Images : Application to Mouse Embryo Imaging In Vivo. In *IEEE Ultrasonics, Ferroelectrics, and Frequency Control Society (UFFC02)*, München, Germany, October 2002.

[201] J. Weickert. A Review of Nonlinear Diffusion Filtering. *Scale-Space Theory in Computer Vision*, LNCS 1252 :3–28, 1997.

[202] Jon Weissman and Xin Zhao. Scheduling parallel applications in distributed networks. *Cluster Computing (CC)*, 1(1) :109–118, 1998.

[203] Richard Wolski. Dynamically forecasting network performance using the Network Weather Service. *Cluster Computing (CC)*, 1(1) :119–132, 1998.

[204] (W3C) World Wide Web Consortium. Web Services Description Language (WSDL) 1.1, March 2001. http ://www.w3.org/TR/wsdl.

[205] C. Xu and J.L. Prince. Snakes, Shapes, and Gradient Vector Flow. *IEEE Transactions on Image Processing (TIP)*, 7(3) :359–369, March 1998.

[206] Jia Yu and Rajkumar Buyya. A taxonomy of scientific workflow systems for grid computing. *ACM SIGMOD records (SIGMOD)*, 34(3) :44–49, September 2005.

[207] Zhengyou Zhang. Iterative point matching for registration of free-form curves and surfaces. *International Journal of Computer Vision (IJCV)*, 13(2) :119–152, December 1994.

[208] Jun Zhao, Carole Goble, Robert Stevens, Dennis Quan, and Mark Greenwood. Using Semantic Web Technologies for Representing e-Science Provenance. In *Third International Semantic Web Conference (ISWC2004)*, pages 92–106, Hiroshima, November 2004.

[209] Song C. Zhu. Unifying snakes, region growing, and Bayes/MDL for Multi-band Image Segmentation. Technical Report 94-10, Harvard Robotics Laboratory, USA, 1994.

[210] A.P. Zijdenbos, B.M. Dawant, R.A. Margolin, and A.C. Palmer. Morphometric analysis of white matter lesions in MR images : Methods and validation. *IEEE Transactions on Medical Imaging (TMI)*, 13 :716–724, 1994.