Analyse d'intervalles pour l'ordonnancement d'activités

Cyril Briand

Habilitation à Diriger des Recherches

Préparée au
Laboratoire d'Analyse et d'Architecture des Systèmes du CNRS - UPR 8001

Présentée à
l'Université Paul Sabatier de Toulouse

par

Cyril Briand
Maître de Conférences à l'Université Paul Sabatier de Toulouse

---

Analyse d'intervalles pour l'ordonnancement d'activités

---

soutenue le 7 décembre 2009 devant le jury :

Rapporteurs :		
	Jean-Charles BILLAUT	Professeur - Université de Tours
	Jacques CARLIER	Professeur - Université Technologique de Compiègne
	Bernard PENZ	Professeur - Université de Grenoble
Examinateurs :		
	Guy JUANOLE	Professeur - Université Toulouse III
	Marie-Claude PORTMANN	Professeur - École des Mines de Nancy
Directrice de recherche :	Colette MERCÉ	Professeur - INSA de Toulouse

Avant-propos

C'est en 1990, à l'occasion de la préparation de mon DEA, que je suis entré dans le Laboratoire d'Analyse et d'Architecture des Systèmes du CNRS (LAAS-CNRS) de Toulouse pour la première fois. J'y ai ensuite poursuivi mes activités en thèse, ayant été l'heureux bénéficiaire d'une allocation de recherche du ministère. J'appartenais alors au groupe "Systèmes de Production" (SP), dirigé à ce moment par Robert Valette, Directeur de Recherche au CNRS. Mon travail de doctorat, assez éloigné de l'ordonnancement, concernait la conception d'une structure de commande hiérarchique multiniveau pour les systèmes de production automatisée. Aujourd'hui, mes activités de recherche se déroulent toujours au LAAS, où je suis membre du groupe "Modélisation Optimisation et Gestion Intégrée de Systèmes d'Activités" (MOGISA), animé par Pierre Lopez, Chargé de Recherche au CNRS.

Dans ma fonction d'enseignant, j'ai toujours été fidèle à l'Université Paul Sabatier (UPS). J'ai d'abord été Moniteur, puis demi Attaché Temporaire d'Enseignement et de Recherche et, depuis 1995, Maître de Conférences. Je suis principalement intervenu dans les filières EEA durant mon monitorat et mon année de 1/2 ATER, puis à l'IUP "Systèmes Intelligents", depuis mon recrutement.

Si je n'ai pas été très mobile géographiquement, mes thèmes de recherche ont, eux, été assez mouvants. Le profil de recherche associé au poste de Maître de Conférences sur lequel je fus classé premier en septembre 1995 était intitulé "Vision pour la robotique", thème tout de même très lointain de celui développé durant ma thèse .... Je devais donc abandonner mes recherches passées pour intégrer l' équipe "Robotique et Intelligence Artificielle" (RIA) du LAAS et y mener de nouvelles recherches. Heureusement, cette réorientation étant tout de même assez risquée, il me fut possible de demeurer dans mon équipe originelle, pour poursuivre les travaux entrepris en thèse. Si rester dans le groupe SP ne posa aucun problème, garder le même thème de recherche fut impossible du fait de la restructuration, au printemps 1996, des activités du groupe (rebaptisé à cette époque "Organisation et Conduite de Systèmes Discrets", restructuration qui sonna le glas de mon thème. Je dus donc me reconstruire une nouvelle identité scientifique ...

Désireux de me frotter à des problèmes possédant un formalisme mathématique plus fort que ceux abordés durant ma thèse, je décidai de m'orienter vers un thème qui

m'attirait depuis longtemps : l'ordonnancement d'activités. Aujourd'hui encore, j'ai du mal à m'expliquer l'origine de mon attirance pour les problèmes d'ordonnancement. La présence dans mon groupe de recherche du quatuor Jacques Erschler, Patrick Esquirol, Pierre Lopez et Marie-José Huguet, que j'appréciais pour des raisons autant scientifiques qu'humaines, n'y est sans doute pas étrangère .... Mes premiers pas dans le domaine de l'ordonnancement, relatés brièvement dans ce mémoire, ont donc été marqués à la fois par leur vision de l'ordonnancement plutôt orientée "contraintes", ainsi que par ma culture d'automaticien des systèmes à événements discrets.

C'est en 1999 que François Roubellat, alors responsable de l'équipe, me fit la proposition de co-encadrer avec lui une thèse concernant un problème ordonnancement avec gestion de fournées. C'est avec joie que j'acceptai. Je le remercie d'ailleurs ici chaleureusement. En effet, c'est grâce à sa main tendue, ses encouragements et aussi ses critiques, que j'ai pu petit à petit progresser, m'épanouir dans mes recherches et prendre de l'autonomie (bien que la thèse ne démarra jamais du fait de l'abandon du doctorant !).

2001 est une année charnière puisque la possibilité s'est offerte à moi de co-encadrer deux thèses de doctorat. La première était une thèse en convention CIFRE et concernait l'organisation de réseaux logistiques. Je co-encadrai avec Patrick Esquirol, François Roubellat suivant lui aussi le travail en nous prodiguant ses impressions et conseils. La deuxième thèse, de nature plus théorique, concernait la flexibilité et la robustesse en ordonnancement et fut co-encadrée avec Jacques Erschler. J'en profite au passage pour le remercier de la confiance qu'il m'a témoignée en me laissant la maîtrise d'œuvre de ces recherches, ainsi que pour toute l'attention qu'il a portée à ces travaux. Merci aussi à vous, Emmanuelle et Trung, je souhaite à tout nouvel encadrant de connaître des doctorants aussi sérieux, tenaces et sympathiques que vous !

Les travaux menés dans le cadre de la flexibilité et de la robustesse en ordonnancement ont été très porteurs. Ils furent amorcés à l'époque où le sous-groupe *Flexibilité et Robustesse en Ordonnancement* du *GOThA*[1] se développait. Ce fut pour moi une expérience très riche que de participer à ce groupe, aussi bien sur le plan humain que scientifique. D'autre part, c'est sans doute aussi grâce à cette expérience que je pus amorcer en 2004 une collaboration scientifique avec Brahim Bouzouia (Directeur de Recherche au Centre de Développement des Technologies Avancées d'Alger), qui m'a conduit à accepter courant 2005 l'encadrement des travaux de doctorat de Samia Ourari, Chargée de Recherche au CDTA.

Ce mémoire d'HDR décrit mes travaux de recherche dans la période 1999 à 2009 et ignore volontairement les travaux non-connectés à l'ordonnancement ou plus marginaux. Comme nous le verrons, on considère en particulier des problèmes où la notion d'intervalles est très structurante et où des propriétés intéressantes peuvent être déduites de l'analyse de

1. Groupe d'Ordonnancement THéorique et Appliqué

ces intervalles. En cohérence avec le plan proposé par l'Université Paul Sabatier, ce mémoire est constitué de cinq chapitres. Le premier donne quelques informations administratives et décrit brièvement la face "enseignant" de mon métier. Le second, de loin le plus conséquent du mémoire, propose une synthèse de mes recherches et décrit une prospective scientifique. Le troisième dresse un bilan de mes encadrements d'étudiants. Le quatrième détaille mes communications, mes participations à la vie scientifique et mes collaborations industrielles. Enfin, le cinquième chapitre inclut 7 communications essentielles, connectées avec la synthèse proposée au chapitre 2.

Table des matières

Chapitre 1

Curriculum Vitæ

Cyril BRIAND

Maître de Conférences à l'Université Paul Sabatier de Toulouse (UPS)

61ème section - titularisé le 1.09.1996
Classe Normale - 5ème échelon (le 1.04.2004)

Tel : 05.61.33.78.18
Mèl : briand@laas.fr
URL : http ://www.laas.fr/~briand

Né le 21.08.68 à Toulouse
Marié, 3 enfants

Laboratoire d'accueil : Laboratoire d'Analyse et d'Architecture des Systèmes du CNRS de Toulouse (LAAS-CNRS) - Directeur : Raja Chatila

Équipe d'accueil : Modélisation Optimisation et Gestion Intégrée de Systèmes d'Activités (MOGISA) - Responsable : Pierre Lopez

1.1 Cursus universitaire

1995 : Thèse de Doctorat de l'Université Paul Sabatier - " Conception Orientée-Objet et basée Réseaux de Petri de la conduite temps réel des Systèmes de Production Manufacturière ", soutenue le 12 janvier 1995, Option Informatique Industrielle

1991 : DEA Automatique et Informatique Industrielle - UPS - Mention Bien

1990 : Maîtrise Électronique - Électrotechnique - Automatique de l'UPS

1989 : Licence Électronique - Électrotechnique - Automatique de l'UPS

1988 : DEUG A - Option Sciences pour l'Ingénieur à l'UPS

1986 : Bac C au lycée Joseph Savernes de l'Isle Jourdain

1.2 Parcours professionnel

Mars 2004 - Août 2005 : Délégation CNRS - Laboratoire d'accueil : LAAS-CNRS

Depuis 1996 : Maître de Conférences à l'UPS

1995-1996 : Maître de Conférences Stagiaire à l'UPS

1994-1995 : Attaché Temporaire d'Enseignement et de Recherche à l'UPS

1991-1994 : Moniteur à l'UPS

1.3 Activités d'enseignement

Depuis mon recrutement, je me suis principalement investi dans les matières énumérées ci-dessous. Les pourcentages donnent une approximation de l'importance relative de chaque enseignement au cours de ces dernières années.

- **Systèmes de Commande à Événements Discrets**
  - **Mots clés :** Machine à états, Grafcet, Statecharts, réseaux de Petri, Mise en œuvre synchrone/asynchrone
  - **Niveau d'intervention :** Licence 3ème année Professionnelle "Systèmes Intelligents" et Master 1ère année Professionnel "Systèmes Intelligents"
  - **Nature des interventions :** Cours - TD - TP
  - **Pourcentage :** 35%

- **Architecture des Microcalculateurs - Informatique Industrielle**
  - **Mots clés :** Mémoires, UC, UAL, Bus, Microcontrôleur C167, Programmation assembleur, VHDL
  - **Niveau d'intervention :** Licence Professionnelle "Systèmes Intelligents" 2ème et 3ème année
  - **Nature des interventions :** Cours - TD - TP
  - **Pourcentage :** 10%

- **Systèmes de Commande Temps réel**

- **Mots clés :** Système d'exploitation, Ordonnancement temps réel, Mécanismes de synchronisation /communication
- **Niveau d'intervention :** Master Professionnel "Systèmes Intelligents" 2ème année
- **Nature des interventions :** Cours - TD - TP
- **Pourcentage :** 10%

- **Ordonnancement**
  - **Mots clés :** Graphes Potentiels-Tâches, PERT, Méthodes Heuristiques, Suivi de projet, Incertitudes
  - **Niveau d'intervention :** Master Professionnel "Systèmes Intelligents" 2ème année
  - **Nature des interventions :** Cours - TD - TP
  - **Pourcentage :** 15%

- **Recherche opérationnelle**
  - **Mots clés :** Programmation Linéaire, PL en nombres entiers, Graphes, Heuristiques et Métaheuristiques
  - **Niveau d'intervention :** Master recherche "Systèmes Automatique et Informatique Décisionnelle" 2ème année et Master Professionel "Productique" 2ème année.
  - **Nature des interventions :** Cours - TD
  - **Pourcentage :** 30%

1.4 Formations

La majorité de mes interventions se sont déroulées au sein de l'IUP " Systèmes Intelligents " de l'UPS (`http://www.iup-ups.ups-tlse.fr/si/`).Cet IUP présente l'originalité d'offrir aux étudiants une double culture concernant les systèmes automatiques d'une part, et les systèmes informatiques d'autre part. Les compétences développées concernent la Robotique, les Systèmes Embarqués, le Temps Réel, le Traitement d'Image, l'Intelligence Artificielle, les Interaction Homme-Machine, la Réalité Virtuelle et la Synthèse d'Images. L'IUP s'appuie sur 2 grands laboratoires toulousains : le LAAS-CNRS et l'IRIT, ainsi que sur un réseau de partenaires industriels impliqués fortement dans les enseignements et dans l'actualisation des programmes.

Un part non négligeable de mes interventions s'est aussi déroulée dans le cadre du Master recherche "Systèmes Automatiques et Informatique Décisionelle" de l'UPS, plus

précisément dans le parcours "Systèmes Industriels", où j'ai dispensé des enseignements de RO, suite au départ à la retraite de Gérard Authié.

Depuis 2007, j'interviens également au sein du Master Professionel "Productique" 2ème année où j'assure des enseignements de Gestion de production et de Graphes.

Signalons enfin qu'en 2007, j'ai eu l'occasion d'assurer quelques vacations à l'INSA de Toulouse pour des enseignement en Programmation Mathématique.

1.5 Responsabilités administratives

Animations pédagogiques

1995-1999 : Co-responsable d'année de la Licence 3ème année IUP "Systèmes Intelligents " de l'UPS

1999-2004 : Co-Responsable d'année du Master Professionnel 2ème année IUP "Systèmes Intelligents" de l'UPS

La responsabilité d'année consiste à prendre en charge, au jour le jour, l'organisation des emplois du temps de la formation, la diffusion de l'information auprès des étudiants, l'organisation de réunions bilans, la vérification des notes utilisées en jury ; par ailleurs, j'étais tenu de participer aux réunions pédagogiques, aux conseils de perfectionnement de l'IUP et au recrutement des étudiants (examen de dossier et audition). Les enseignements étant assurés par des enseignants du LAAS et de l'IRIT, il y a un responsable d'année pour chaque laboratoire et pour chaque année.

2000 - Aujourd'hui : Animateur du "Projet de Grande Envergure" (PGE) du Master Professionnel 2ème année de l'IUP "Systèmes Intelligents" de l'UPS

Le PGE a pour but d'illustrer les enseignements sur les systèmes interactifs et les systèmes de commande temps réel du Master 2 "Systèmes Intelligents" à travers la réalisation d'une application, proposée par un industriel, s'inscrivant généralement au sein d'un projet de R&D. Les étudiants sont autonomes et doivent s'organiser pour prendre en charge la réalisation de l'application depuis sa spécification jusqu'à sa livraison effective. Le PGE leur donne donc l'occasion d'expérimenter en grandeur réelle les techniques de gestion de projet présentées dans la formation et de comprendre leur importance. Il constitue également une expérience humaine forte et enrichissante (pas seulement pour les étudiants !). Le travail de l'animateur consiste à démarcher les industriels pour trouver un sujet en adéquation avec les disciplines enseignées à l'IUP, affiner les besoins de l'industriel sélectionné, participer et organiser l'ensemble des revues de projet (7 revues au total), assurer une bonne communication entre les acteurs du projet et conseiller les étudiants quant à leur organisation interne.

Commission de Spécialistes

1998-2001 : Membre élu titulaire de la commission de spécialistes 61ème section de l'UPS

2001-2004 : Membre suppléant de la commission de spécialistes 61/63 de Bordeaux - Talence

2004-2008 : Membre élu suppléant de la commission de spécialistes 61ème section de l'UPS

1.6 Détail des activités d'enseignement

1.6.1 Systèmes de Commande à Événements Discrets

Cet enseignement constitue la part essentielle de mes activités et est actuellement réparti sur deux années de formation (Bac+3 et Bac+4). Les objectifs sont d'initier l'étudiant à concevoir et mettre en œuvre des machines à états finies en vue de décrire la commande de systèmes complexes. Des modèles de plus haut niveau, tels que les Statecharts, le Grafcet et les réseaux de Petri sont également présentés au niveau Master. Au cours des dernières années, j'ai assuré les cours, les TDs et les TPS de cet enseignement. Les cours ont donné lieu à la préparation de supports (planches Power-Point), distribués aux étudiants et complétés en séance par l'analyse de quelques exemples traités au tableau.

J'ai également largement contribué à la mise en place des travaux pratiques :

- En licence 3, j'ai mis en place une manipulation de commande d'un système à événements discrets simulé. Il s'agit d'un ascenseur, se déplaçant sur quatre niveaux, que j'ai implémenté en Ada avec une interface GTK. Les étudiants doivent proposer un modèle de commande de la cabine (machine à états) qu'ils doivent ensuite coder en C et valider en simulation.
- En Master 1, les TPs illustrant le cours sont composés de trois manipulations dédoublées mettant en œuvre des procédés discrets réels : une maquette d'ascenseur (machine à états - commande par FPGA), une maquette de trains (modèle Grafcet - commande par automate programmable) et une maquette d'atelier flexible (machine à états - commande logicielle sur PC linux). En 2000, j'ai assuré la mise en place de la manipulation consacrée à la commande de la maquette d'atelier flexible (composée de 2 robots, de 4 tapis de convoyage, de 2 tables d'indexage et de nombreux capteurs / actionneurs) et à la définition d'un sujet de manipulation. La mise en place de la manipulation nécessita plusieurs mois de travail puisqu'il fallut travailler aussi bien sur la configuration matérielle que sur la configuration logicielle : mise en place des cartes d'E/S, analyse des E/S, écriture de modules Linux temps réel (utilisation de RTAI) permettant de discrétiser complètement la commande des robots, validations et tests.
- Je signale également qu'avant l'actuelle habilitation, la formation IUP démarrait au niveau Licence 2. J'intervenais alors à ce niveau dans les enseignements d'initiation à la logique séquentielle où les objectifs étaient de rappeler les éléments de logique combinatoire et de présenter les circuits séquentiels de base tels que les

bascules, les compteurs et les registres. Dans ce cadre, j'ai préparé les textes des 4 sujets de manipulations dont deux étaient mis en œuvre par logique câblée sur platines logiques, un par circuits logiques programmables (CPLD) et le dernier par programmation en C de machine à états.

1.6.2 Architecture des Microcalculateurs - Informatique Industrielle

Cet enseignement s'adresse aux étudiants de Licence 3 de l'IUP SI. Il est réalisé en lien avec les TP de logique séquentielle précédemment évoqués (le microcalculateur étant utilisé dans un objectif de commande d'un SED). Jusqu'en 2002, il visait à présenter l'architecture d'un micro calculateur de type PC via la programmation en langage assembleur. Depuis 2002, on s'intéresse à présenter l'architecture d'un microcontrôleur Siemens C167 et à la commande en C de ses unités périphériques. Les éléments périphériques basiques sont présentés (port série, timer, convertisseur analogique / numérique, ports parallèles programmables, contrôleur d'interruption, etc.). Pour chacun des deux supports, ma contribution a consisté à concevoir et mettre en place les cours (création de supports) et TD, ainsi que les sujets de manipulations associés, dont certains étaient conçus comme des mini-projets de 6h, différent chaque année.

Depuis 2005, je n'assure plus cet enseignement.

1.6.3 Systèmes de Commande Temps Réel

Il s'agit d'une part de décrire les principales caractéristique d'un OS Temps Réel et d'autre part d'initier les étudiants à la programmation de tâches temps réel (utilisation de RTAI - Linux). Le cours présente les spécificités d'un SE temps réel (prédictabilité), analyse plusieurs méthodes d'ordonnancement temps réel de processus (Rate Monotonic, EDF, ...) et décrit les spécificités de quelques SE temps réel (VxWorks, RTAI, ...). Ce cours a donné lieu à la préparation de planches Power-Point servant de support et à la rédaction de sujets de TDs.

Les TPs du Master 2 de l'IUP ont pour but d'illustrer aux étudiants comment implémenter soit matériellement, soit logiciellement, un système de commande temps réel. Quatre manipulations de 8h ont été définies. La première manipulation consiste en une description VHDL d'une architecture de commande d'un SED avec gestion du temps explicite. La deuxième manipulation est consacrée à la réalisation d'un générateur de signaux à fréquence, déphasage et amplitude réglable. Les étudiants implémentent un module de noyau Linux contenant plusieurs tâches temps réel et analysent les performances de leur système du point de vue temporel (fréquence maximale accessible). La troisième manipulation s'intéresse à la commande d'un robot mobile (Pekee), commande mise en œuvre sous Windows NT en C++. Il s'agit de montrer aux étudiants quels sont les mécanismes disponibles sous Windows permettant de gérer des contraintes temporelles et de montrer leurs limites. La dernière manipulation traite de la commande d'une application

complexes sous Linux en C++. Les étudiants proposent une modélisation par réseau de Petri de la commande, qu'ils décomposent ensuite en tâches et qu'ils synchronisent à l'aide des primitives offertes sous Linux. Les quatre manipulations se déroulent en parallèle dans la salle, une rotation étant définie pour chaque binôme étudiant. Pour ces TPs, j'ai participé à la rédaction des sujets et à leur mise en place.

1.6.4 Ordonnancement

Jusqu'en 2000, des cours, TD et TP d'ordonnancement de production étaient dispensés aux étudiants de M1 de l'IUP SI. J'ai pris en charge la totalité de ce service où il s'agissait de sensibiliser l'étudiant aux divers niveaux de décision caractérisant un système de production, en ciblant notamment la fonction ordonnancement. Mon intervention s'est traduite par un réaménagement du programme des cours et TD et une adaptation des TPs. Ces derniers étaient donnés sous forme d'un bureau d'études de 6h basé sur l'utilisation du logiciel Preactor 400 et visant à résoudre certains problèmes d'ordonnancement de plus en plus complexes.

Depuis 2000, ce cours est remplacé par un cours "Ordonnancement de projet", intégré dans le module "gestion de projet" du M2 de l'IUP SI, cours que j'ai également pris en charge. Celui-ci est consacré à l'étude de méthodes permettant de construire un planning en intégrant les contraintes de ressources et de temps, relatives aux activités du projet. La modélisation d'un problème d'ordonnancement de projet par graphe potentiels-tâches est décrite. Les flots de ressource sont également présentés. On montre également comment suivre l'avancée du déroulement du projet du point de vue des coûts, des délais et du risque. Quelques méthodes sont également présentées permettant de tenir compte, lors de la définition d'un planning, des incertitudes intrinsèques au projet. Un BE de 6h, basé sur l'utilisation du logiciel Microsoft-Project, a également été conçu où les étudiants, structurés par équipe, définissent un planning pour organiser et rationaliser les activités liées au développement et à l'intégration des divers modules de leur "Projet de Grande Envergure".

1.6.5 Recherche Opérationnelle

Voilà 4 ans que j'interviens au sein du Master Recherche SAID pour les enseignements de PL, PLNE et Graphes. Cette intervention m'a demandé un gros investissement car elle sortait du champ habituel de mes enseignements, et même de ma formation initiale (davantage orientée 61 que 27). J'ai ainsi passé beaucoup de temps à recueillir, consulter, comprendre et trier les divers matériels disponibles dans les livres et sur le web, pour élaborer mon propre support de cours. Ce travail, s'il m'a demandé beaucoup de temps, a été réellement passionnant et m'a beaucoup profité, aussi bien sur le plan pédagogique, que dans l'exercice de mon travail de recherche.

J'interviens dans le Master 2 Professionel "Productique" depuis 2 ans. J'ai réutilisé les compétences acquises lors de mes enseignements au master recherche pour élaborer un cours portant sur les graphes et leurs applications en logistique. Là aussi, j'ai pris beaucoup de plaisir à essayer de démontrer aux étudiants l'intérêt des graphes, et

des méthodes d'optimisation combinatoire en général, pour la résolution de problèmes logistiques concrets

Chapitre 2

Synthèse des activités et projet de recherche

2.1 Introduction

Faire la synthèse d'un ensemble de travaux de recherche est un exercice délicat. L'idéal est de trouver une articulation favorisant une présentation logique des idées et des concepts, et mettant en relief une cohérence d'ensemble. Dans mon cas, la totalité des travaux présentés s'intéresse à la résolution de problèmes d'ordonnancement ou de séquencement : problème à 1 machine, flowshop de permutation, ordonnancement de projet et réseaux logistiques. Je me rends compte également aujourd'hui que dans chacun de ces problèmes, la notion d'*intervalle* a souvent été centrale et structurante : c'est bien dans la façon d'analyser les intervalles "caractéristiques" d'un problème que réside l'originalité des résultats présentés dans ce mémoire. Les intervalles étudiés sont de natures diverses : intervalles d'exécution dans le cas de problèmes à machine unique, intervalles de durées opératoires dans le cas de flowshop, intervalles d'écart temporel dans le cas de l'ordonnancement de projet et intervalles de délais et/ou de quantités (liés à une demande) dans le cas de réseaux logistiques. Pour toutes ces raisons, il m'a semblé que "*Analyse d'intervalles pour l'ordonnancement d'activités*" était le titre qui reflétait le mieux le contenu de cette synthèse.

Ce manuscrit fait le point sur les résultats essentiels obtenus au cours des 10 dernières années. Elle est organisée en 5 sections. L'ordre de présentation des travaux a été choisi en fonction des thèmes et n'obéit donc pas à une logique chronologique. S'agissant d'une synthèse, certains aspects sont parfois rapidement balayés, l'accent étant mis sur les points jugés “marquants”. Pour plus de détail, le lecteur est invité à se référer aux articles insérés

dans le chapitre 5.

Ce chapitre est structuré comme suit. Dans un premier temps (partie 2.2), quelques notations, concepts et notions nécessaires à la compréhension du mémoire sont précisés. La partie 2.2.3 s'intéresse aux problèmes à 1 machine et au problème flowshop à 2 machines. Nous montrons comment, à partir d'une analyse d'intervalles, il est possible de mettre en évidence des conditions logiques efficaces pour réduire l'espace de recherche associé à la résolution de ces problèmes de séquencement. La partie suivante (partie 2.5) présente comment, en utilisant certaines des conditions logiques précédentes, il est possible de caractériser un vaste ensemble de solutions dont la performance au pire est bornée. Il est également expliqué comment cet ensemble peut être manipulé pour améliorer la performance au pire et dans quelle mesure cet ensemble peut être qualifié de *robuste*. Les parties 2.6 et 2.7 s'intéressent à des problèmes plus appliqués. La première traite du problème d'ordonnancement de projet (noté RCPSP du fait de la dénomination anglaise *Resource-Constrained Project Scheduling Problem*). On s'intéresse en particulier à la prise en compte de contraintes d'intervalles minimum et maximum, entre les dates de début et/ou de fin des activités, et à la proposition de nouvelles heuristiques de résolution. La seconde décrit les travaux que nous avons réalisés dans le cadre de l'organisation coopérative de réseaux logistiques. Une aide à la décision est en particulier proposée dans le but de supporter les coopérations entre les couples fournisseur - donneur d'ordre du réseau logistique. Cette aide concerne le dimensionnement des délais et des quantités de produits relatifs aux commandes transitant entre fournisseurs - donneurs d'ordres.

La dernière section de ce chapitre donne une prospective pour ces travaux de recherche, celle-ci correspondant naturellement à ma vision présente de l'avenir. Elle s'inscrit dans la continuité de mes axes de recherche actuels.

2.2 Quelques définitions

2.2.1 Notations et problèmes d'ordonnancement considérés

Plusieurs définitions d'un problème d'ordonnancement sont données dans la littérature. Nous donnons ici celle proposée dans [Esquirol & Lopez 99] :

« Le problème d'ordonnancement consiste à organiser dans le temps la réalisation de tâches, compte tenu de contraintes temporelles (délais, contraintes d'enchaînement, ...) et de contraintes portant sur l'utilisation et la disponibilité de ressources requises. »

Le mot *ordonnancement* désigne soit une solution au problème d'ordonnancement, soit par abus de langage, le processus ayant conduit à la détermination de cette solution. Une solution d'ordonnancement décrit les dates prévues pour l'exécution des tâches et l'allocation des ressources au cours du temps. La méthode utilisée pour l'élaboration d'un ordonnancement vise souvent à satisfaire un ou plusieurs objectifs (coûts, délais, qualité), exprimés au sein d'un ou plusieurs critères de performance. Pour une description plus détaillée de la problématique de l'ordonnancement, le lecteur peut se référer aux ouvrages [Coffman 76, Blazewicz *et al.* 96, Esquirol & Lopez 99, Leung 04, Pinedo 08].

Brièvement, les paragraphes suivants précisent les notions de tâche, travail, activité, ressource, objectif, ... et introduisent quelques notations utiles pour la compréhension de la synthèse.

Une tâche i est une entité élémentaire de travail caractérisée par une date de disponibilité r_i, une date de d'échéance d_i et une durée opératoire p_i. Sa localisation dans le temps est définie par une date de début $s_i \geq r_i$ et/ou une date de fin $f_i \leq d_i$. Dans le cas où la durée opératoire $p_i = f_i - s_i$ est connue, la valeur de la variable f_i peut être déduite de la valeur de s_i (et vice-versa). La réalisation de la tâche i nécessite l'utilisation d'une ou plusieurs ressources $k \in R$, où R est l'ensemble de ressources. Les ressources concernées par la réalisation de la tâche sont généralement supposées connues a priori. L'intensité de ressource k consommée pour la réalisation de i est notée q_{ik} (supposée constante lors de l'exécution de la tâche). Dans ce rapport nous supposons que les tâches ne peuvent pas être exécutées par morceaux, le problème est dit *non préemptif*.

Les tâches sont interdépendantes à double titre. D'une part, parce qu'elles partagent un ensemble de ressources de capacité limitée et, d'autre part, parce que des contraintes de temps entre les dates de début et de fin des tâches doivent être respectées (l'écart entre les occurrences de 2 tâches peut être borné inférieurement ou supérieurement). Ces contraintes traduisent en pratique des impératifs technologiques et organisationnels, ou correspondent à des préférences.

Dans le domaine de la production, on désigne par *travail* un sous-ensemble de tâches (ou *opérations*) à réaliser. Ce sous-ensemble correspond en général à la fabrication d'une entité physique ou à la réalisation d'un service. Des contraintes de séquencement entre les tâches membres d'un même travail doivent généralement être respectées pour assurer la cohérence de l'ordonnancement. Dans le domaine de la gestion de projet, le mot *activité* est préféré à celui de tâche.

Une ressource k est une entité matérielle (une machine, un outil, une matière première, un lieu, ...) ou humaine, disponible en quantité limitée, destinée à être utilisée pour la réalisation d'une ou plusieurs tâches. Sa disponibilité est généralement exprimée par une *capacité* propre à chaque ressource k, notée Q_k ($Q_k \geq 1$). Bien qu'ici nous la supposerons constante, cette capacité peut dépendre du temps ou de l'*état* de la ressource. Une ressource est dite renouvelable si, après avoir été utilisée par une ou plusieurs tâches, elle est à nouveau disponible en même quantité (les hommes, les machines ...). Dans le cas contraire, elle est dite consommable (matières premières, budget ...). Dans ce rapport, nous considérerons uniquement des ressources renouvelables.

L'ordonnancement produit doit assurer, d'une part, que les contraintes temporelles entre les tâches soient satisfaites et, d'autre part, que la consommation des tâches sur les ressources soit en cohérence, à tout instant, avec leurs disponibilités. Dans ce cas, l'ordonnancement est dit *admissible* (ou *faisable*).

Dans ce manuscrit, trois problèmes d'ordonnancement sont particulièrement étudiés. Tout d'abord, le problème à 1 machine. Il s'agit de déterminer une séquence de n travaux (chaque travail étant constitué d'une seule tâche), ceux-ci devant être tour à tour réalisés sur une ressource renouvelable de capacité unitaire. Chaque travail i est caractérisé par une date de disponibilité r_i, une date d'échéance d_i et une durée opératoire p_i. On cherche généralement à déterminer une séquence qui soit admissible (i.e. $f_i \leq d_i \forall i$), si possible, ou qui minimise le plus grand retard algébrique ou retard vrai (problèmes respectivement notés $1|r_i|L_{\max}$ ou $1|r_i|T_{\max}$), ou qui minimise le nombre de travaux en retard (problème noté $1|r_i|\sum U_i$). En l'absence d'hypothèse supplémentaire, chacun de ces problèmes d'optimisation est NP-difficile au sens fort.

Le deuxième problème que l'on considère est le flowshop de permutation à deux machines. Ici, chaque travail j est composé de 2 tâches, j_1 et j_2, chacune devant s'exécuter en séquence sur la machine M_1, puis sur la machine M_2. On note p_{j1} et p_{j2} respectivement les durées opératoires du travail j sur les machines M_1 et M_2. L'ordre de réalisation des travaux est supposé être le même sur chaque machine (ordonnancement de permutation). Les machines sont de capacité unitaire. On s'intéresse ici à caractériser un vaste ensemble de séquence de travaux qui minimise la durée totale d'exécution (problème noté $F2|prmu|C_{\max}$).

Si les 2 problèmes précédents sont relativement académiques, le 3ème problème que

nous considérons, le RCPSP, se rapproche davantage de problèmes pratiques. Un ensemble de n activités doit être ordonnancé dans le temps. Il existe des contraintes de temps entre les dates de début et de fin de chaque activité. L'exécution d'une activité requiert l'utilisation d'une ou plusieurs ressources, chacune pouvant être consommée dans une quantité supérieure à 1. Les ressources sont renouvelables et cumulatives. Le problème consiste à déterminer un ordonnancement des activités qui soit d'une part réalisable (du point de vue des contraintes de temps et de ressource) et qui minimise, d'autre part, la durée totale d'exécution (problème noté $PS||C_{\max}$ dans la littérature). Ce dernier problème est également NP-difficile.

2.2.2 Structures d'intervalles, sommets, bases et pyramides

Une partie des travaux présentés dans ce mémoire s'intéressant à l'analyse d'intervalles, il est nécessaire de préciser au préalable quelques définitions des notions mises en jeu [Esquirol & Lopez 02].

Une structure d'intervalles est définie par un couple $< I, C >$ où $I = i_1, \ldots, i_n$ est un ensemble d'intervalles et C est un ensemble de contraintes sur $I \times I$. Chaque intervalle i_j est défini par une date de début x_j et une date de fin y_j. Une contrainte entre deux intervalles i_j et i_k peut être exprimée soit en spécifiant un ordre total entre les dates de début et de fin des deux intervalles soit, de façon équivalente, en utilisant une des relations de l'algèbre de Allen [Allen 91], récapitulées sur la figure 2.1.

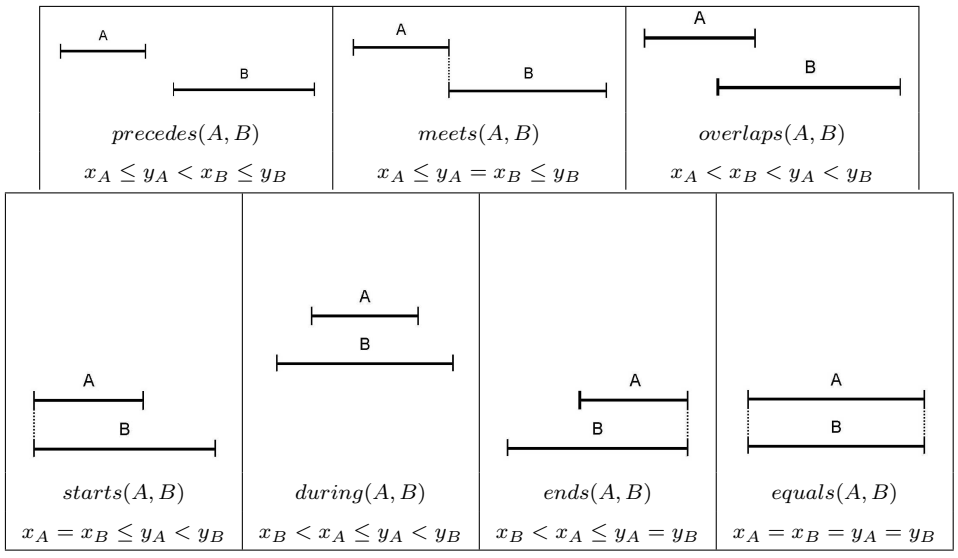


FIGURE 2.1 – Algèbre de Allen

Sur la base des relations de Allen, deux types particuliers d'intervalles peuvent être mis en évidence pour une structure d'intervalles : l'intervalle de type *sommet* et celui de type *base*.

Définition 1. *Un intervalle s est dit sommet d'une structure d'intervalles $< I, C >$ si $\nexists i \in I$ tel que $during(i, s)$.*

Définition 2. *Un intervalle b est dit base d'une structure d'intervalles $< I, C >$ si $\nexists i \in I$ tel que $during(b, i)$.*

En utilisant ces notions de sommet et base, les notions de *sommet-pyramide* (notée *s-pyramide*) et de *base-pyramide*(notée *b-pyramide*) peuvent être définies [Esquirol & Lopez 02].

Définition 3. *Une s-pyramide P_s, associée au sommet s d'une structure d'intervalles $< I, C >$, est le sous-ensemble d'intervalles $i \in I$ tel que $during(s, i)$.*

Définition 4. *Une b-pyramide P_b, associée à la base b d'une structure d'intervalles $< I, C >$, est le sous-ensemble d'intervalles $i \in I$ tel que $during(i, b)$.*

Les définitions 1 à 4 sont illustrées sur l'exemple de la figure 2.2. On identifie les sommets $\{C, D, E\}$ qui caractérisent respectivement les s-pyramides : $P_C = \{A, B, G\}$, $P_D = \{G\}$ et $P_E = \{F, G\}$. De même, on identifie les bases $\{A, B, F, G\}$ et les b-pyramides $P_A = \{C\}$, $P_B = \{C\}$, $P_F = \{E\}$, $P_G = \{C, D, E\}$.

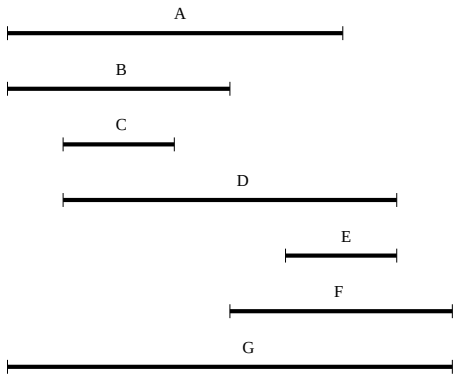


FIGURE 2.2 – Exemple d'intervalles

2.2.3 Conditions nécessaires / suffisantes et conditions de dominance

Dans le but d'améliorer l'efficacité des méthodes de résolution en ordonnancement, de nombreux chercheurs se sont intéressés à la mise en évidence de conditions logiques

permettant de restreindre l'espace de recherche à explorer. On distingue en général trois types d'approche selon que les conditions logiques utilisées sont nécessaires, suffisantes ou dominantes.

Les approches par conditions *suffisantes* caractérisent un sous-ensemble de l'ensemble des ordonnancements admissibles (resp. optimaux relativement à un critère donné). Les ensembles de solutions satisfaisant des conditions *nécessaires* comprennent tous les ordonnancements admissibles (resp. optimaux), plus certains ordonnancements non admissibles (resp. non optimaux).

Les approches par conditions *dominantes* éliminent de l'espace de recherche les solutions qui sont nécessairement dominées par d'autres. Dans le cadre de l'étude de l'admissibilité, une solution Sol_1 domine une autre solution Sol_2 si et seulement si Sol_2 admissible implique Sol_1 admissible [Erschler *et al.* 83]. Les solutions dominantes peuvent être considérées comme potentiellement meilleures que les autres dans le sens où, si aucune n'est admissible, alors cela implique qu'il n'existe pas de solution admissible pour le problème considéré.

2.3 Problèmes à machine unique

Appartenant à la classe des problèmes NP difficiles, le problème à 1 machine a été de tout temps une cible très appréciée par les chercheurs. Il reste cependant relativement académique et possède peu de débouchés directs en pratique. Une motivation principale pour continuer à étudier ce problème réside surtout dans l'espoir de concevoir de nouveaux algorithmes efficaces, utilisables pour des classes particulières de problèmes à 1 machine, ces algorithmes pouvant être ensuite réutilisés, en tant que briques élémentaires, pour résoudre des problèmes plus complexes. Dans le cadre du problème job shop par exemple, décomposer le problème général en un ensemble de sous-problèmes à 1 machine que l'on résout séparément est une stratégie payante, ainsi que l'ont démontré plusieurs chercheurs (voir [Adams *et al.* 88, Carlier & Pinson 89, Chu *et al.* 92]).

En ce qui nous concerne, le choix du problème à 1 machine en tant qu'objet d'étude a été motivé par la volonté de concevoir de nouvelles techniques pour caractériser un ensemble flexible de solutions dans un problème d'ordonnancement. Dans cette optique, nous avons étudié comment l'utilisation de conditions de dominance pouvait permettre de mettre en évidence un ensemble de solutions remarquables. Pour cela, nous sommes partis d'un théorème de dominance, démontré il y a plus d'une vingtaine d'années, au LAAS-CNRS. Ce théorème, ainsi que d'autres résultats de dominance que nous avons depuis mis en évidence dans le cadre de la thèse de Samia Ourari, sont présentés dans les sous-parties suivantes. La notion d'*ensemble flexible de solutions* est quant à elle développée plus loin, dans la partie 2.5.

2.3.1 Retour aux années 80 : le théorème des pyramides

Dans les années 80, un nouveau théorème de dominance est proposé dans [Couzinet-Mercé 79, Erschler *et al.* 83] pour la caractérisation d'un ensemble de séquences. Ce théorème, baptisé *théorème des pyramides*, a été initialement conçu par les auteurs dans le but de restreindre l'ensemble de séquences à explorer pour la recherche d'une solution admissible (la recherche étant réalisée par une technique branch-and-bound).

Ce théorème est basé sur une analyse de structure d'intervalles. On considère la structure d'intervalles $< I, C >$ où un intervalle $i_j \in I$, caractérisé par une date de début r_j et une date de fin d_j, est associé à chaque travail j du problème. On s'intéresse dans ce qui suit aux sommets de la structure d'intervalles ainsi qu'aux s-pyramides qu'ils caractérisent. Pour cela, seul l'ordre relatif défini par les dates de disponibilité r_j et d'échéance d_j de l'ensemble T des n travaux à ordonnancer est considéré. Les durées opératoires p_j ainsi que les valeurs explicites des r_j et d_j de chaque travail $j \in T$ ne sont pas utilisées.

On suppose que les sommets sont indicés par r_j croissant (par d_j croissant en cas d'égalité). Si deux sommets possèdent des dates de disponibilité et d'échéance identiques, alors ils peuvent être indicés de façon quelconque. La s-pyramide notée P_α désigne la pyramide caractérisée par le sommet s_α. On note $u(j)$ (resp. $v(j)$) l'indice de la première s-pyramide à laquelle le travail j appartient (resp. l'indice de la dernière pyramide à laquelle le travail j appartient). Un ordre partiel dominant peut alors être caractérisé grâce au théorème suivant.

Théorème 1. *L'ensemble des séquences de travaux de la forme* $\sigma = \alpha_1 \prec s_1 \prec \beta_1 \prec \cdots \prec \alpha_k \prec s_k \prec \beta_k \prec \cdots \prec \alpha_m \prec s_m \prec \beta_m$, *où :*

- s_k *est le sommet de la pyramide* P_k, $\forall k = 1 \ldots m$;
- α_k *et* β_k *sont deux sous-séquences de travaux situées respectivement à la gauche et à la droite du sommet* s_k, *telles que les travaux de la sous-séquence* α_k *sont ordonnés selon l'ordre croissant de leur* r_j, *et ceux appartenant à* β_k, *selon l'ordre croissant de leur* d_j ;
- *tout travail non sommet* j *est affecté soit à* α_k, *soit à* β_k, *pour un certain* k *tel que* $u(j) \leq k \leq v(j)$;

est dominant pour le problème de recherche d'une solution admissible.

Seul l'ordre relatif entre les dates de disponibilité et d'échéance des travaux étant considéré pour la définition des sommets et des pyramides, l'ensemble des séquences dominantes reste inchangé tant que les valeurs numériques des r_j et d_j restent cohérentes avec cet ordre, cela indépendamment des valeurs des durées opératoires, d'où la généricité du résultat. Une autre propriété intéressante du théorème est que la cardinalité de l'ensemble dominant $\mathcal{S}_{\text{dom}}$ peut être calculée grâce à la formule $card(\mathcal{S}_{\text{dom}}) = \prod_{q=1}^{N}(q+1)^{n_q}$, où n_q désigne le nombre d'intervalles non sommets appartenant exactement à q pyramides et N le nombre total de pyramides.

Considérons par exemple, le problème d'ordonnancement à quatre travaux du tableau 2.1. L'ordre relatif des dates de disponibilité et au plus tard est : $r_2 < r_5 < r_1 < d_1 < d_5 < r_3 < r_4 < d_4 < d_3 < d_2$. La structure d'intervalles associée à ce problème est représentée sur la figure 2.3. Elle contient deux sommets : $s_1 = 1$ et $s_2 = 4$. Ces deux sommets caractérisent deux pyramides : $P_1 = \{2, 5\}$ et $P_2 = \{2, 3\}$. Remarquons qu'en cohérence avec la définition de la notion de pyramide, un sommet n'appartient pas à la pyramide qu'il caractérise.

Travaux	1	2	3	4	5
r_i	6	1	21	24	4
d_i	13	37	33	31	17
p_i	4	5	8	6	7

TABLE 2.1 – Un problème à une machine $1|r_j|L_{\max}$

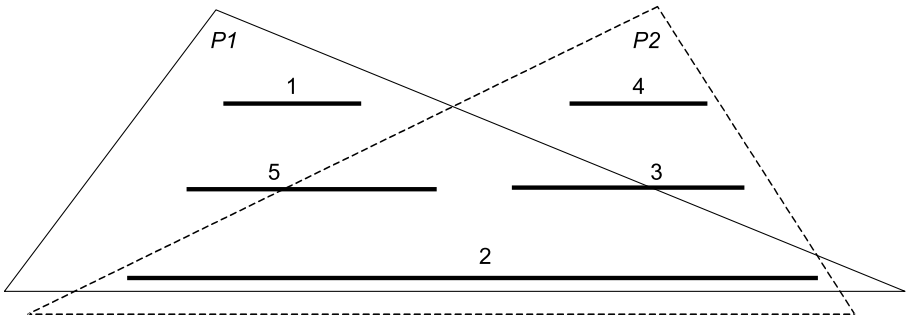


FIGURE 2.3 – Structure d'intervalles du problème du tableau 2.1

L'application du théorème à l'exemple précédent permet de caractériser un ensemble dominant $\mathcal{S}_{\text{dom}}$ de cardinalité $card(\mathcal{S}_{\text{dom}}) = (1+1)^2.(2+1)^1 = 12$, constitué des séquences (cf. figure 2.4) :

$2 \prec 5 \prec 1 \prec 3 \prec 4$, $2 \prec 5 \prec 1 \prec 4 \prec 3$, $5 \prec 1 \prec 2 \prec 3 \prec 4$, $5 \prec 1 \prec 2 \prec 4 \prec 3$,
$5 \prec 1 \prec 3 \prec 4 \prec 2$, $5 \prec 1 \prec 4 \prec 3 \prec 2$, $2 \prec 1 \prec 5 \prec 3 \prec 4$, $2 \prec 1 \prec 5 \prec 4 \prec 3$,
$1 \prec 5 \prec 2 \prec 3 \prec 4$, $1 \prec 5 \prec 2 \prec 4 \prec 3$, $1 \prec 5 \prec 3 \prec 4 \prec 2$, $1 \prec 5 \prec 4 \prec 3 \prec 2$.

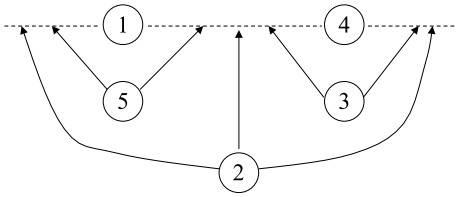


FIGURE 2.4 – Ensemble dominant de séquences associé à l'exemple

Conformément à la propriété de dominance, toute séquence $s \notin \mathcal{S}_{\text{dom}}$ possède une qualité inférieure ou égale à au moins une séquence de $\mathcal{S}_{\text{dom}}$. Remarquons aussi que, dans le cas où la date de disponibilité du travail 2 deviendrait supérieure à celle du travail 1, de sorte que 2 n'appartiendrait plus uniquement qu'à la pyramide P_2, cet ensemble reste dominant. En effet, on peut assurer que l'ensemble dominant $\mathcal{S}'_{\text{dom}}$ correspondant à cette nouvelle structure d'intervalles est strictement inclus dans $\mathcal{S}_{\text{dom}}$ [Briand *et al.* 05].

2.3.2 Une condition numérique de dominance

Dans [Briand *et al.* 06b], considérant le problème à une machine où les intervalles de réalisation des travaux sont inclus les uns dans les autres (*nested* en anglais), il est proposé une condition numérique de dominance permettant d'affiner davantage l'ensemble des séquences dominantes caractérisé par le théorème des pyramides. Cette condition numérique de dominance a été généralisée au cas mono-pyramidal quelconque dans [Briand & Bezanger 06]. On suppose ici que le problème est caractérisé par un sommet

unique $s \in V$, donc une unique pyramide noté P. D'après le théorème des pyramides, l'ensemble des séquences dominantes est de la forme $\alpha \prec s \prec \beta$. Selon que les $(n-1)$ travaux non-sommets sont séquencés dans α ou dans β, on dénombre $2^(n-1)$ séquences dominantes différentes, ce qui constitue une amélioration comparé au $n!$ séquences possibles mais reste encore énorme. Toutefois, la condition nécessaire et suffisante d'admissibilité suivante permet d'affiner un peu plus l'ensemble des séquences dominantes :

Théorème 2. *Soit un problème caractérisé par une unique pyramide P de sommet s, une séquence de la forme $\alpha \prec s \prec \beta$ est admissible si et seulement si $D^{\min} - R^{\max} - p_s \geq 0$, où :*

- $R^{\max} = \max_{a \in \alpha} (r_s, r_a + p_a + \sum_{k \in \alpha, a \prec k}(p_k))$ *représente la date de début au plus tôt de s ;*
- $D^{\min} = \min_{b \in \beta} (d_s, d_b - p_b - \sum_{k \in \beta, k \prec b}(p_k))$ *représente la date de fin au plus tard de s.*

Le fait que le théorème 2 permette une comparaison numérique des séquences est intéressant puisque la condition de dominance suivante peut être immédiatement déduite.

Corollaire 1. *Soit un problème caractérisé par une unique pyramide P de sommet s et soient deux séquences σ_1 et σ_2, de forme respective $\alpha_1 \prec s \prec \beta_1$ et $\alpha_2 \prec s \prec \beta_2$. Si $D_1^{\min} - R_1^{\max} \geq D_2^{\min} - R_2^{\max}$, alors σ_1 domine σ_2 au sens de l'admissibilité.*

On remarque qu'il est facile de montrer le lien entre le critère de minimisation du retard algébrique $L_{\max}$ et les valeurs de $R^{\max}$ et $D^{\min}$ définies plus haut. En effet, ces dernières valeurs permettent de déduire la marge libre ml_s du travail sommet s puisque $ml_s = D^{\min} - R^{\max} - p_s$. Or, cette marge libre est précisément égale à l'inverse du retard algébrique, i.e. $L_{\max} = R^{\max} - D^{\min} + p_s$.

2.3.3 Un modèle de PLNE pour le problème $1|r_j|L_{\max}$

La condition de dominance précédente est intéressante pour concevoir des algorithmes de résolution de problèmes mono-pyramidaux pour le cas "nested" (cf. [Briand *et al.* 06b]). Son principal intérêt est qu'elle peut être étendue au cas multi-pyramidal, ce qui conduit à la définition d'un modèle original de programmation linéaire en nombres entiers (PLNE) pour la résolution du problème général $1|r_j|L_{\max}$.

Cette formulation est décrite dans ce qui suit et a fait l'objet d'un article à paraitre dans la revue RAIRO (cf. [Briand *et al.* 10]), dans le numéro spécial consacré à Roadef'08.

Afin de comprendre la modélisation, nous nous intéressons en premier lieu au problème mono-pyramidal. Nous montrons ensuite comment ce modèle peut être généralisé.

Considérons donc de nouveau un problème caractérisé par une unique pyramide P de sommet s. Il est aisé de déterminer les valeurs $D^{\min}$ et $R^{\max}$ de n'importe quelle séquence de la forme $\alpha \prec s \prec \beta$, et donc de trouver la séquence minimisant $D^{\min} - R^{\max}$, par le PLNE suivant.

$$\max\ z = D - R$$

$$\text{s.t.} \quad \begin{cases} R \geq r_s \\ D \leq d_s \\ R \geq r_i + \sum_{\{j \in V \setminus \{s\} | r_j \geq r_i\}} p_j x_j^+ \ , \ \forall i \in V \setminus \{s\} \\ D \leq d_i - \sum_{\{j \in V \setminus \{s\} | d_j \leq d_i\}} p_j x_j^- \ , \ \forall i \in V \setminus \{s\} \\ x_i^- + x_i^+ = 1 \ , \ \forall i \in V \setminus \{s\} \\ x_i^- \ , \ x_i^+ \in \{0,1\} \ , \ \forall i \in V \setminus \{s\} \\ D \ , \ R \in \mathbb{Z} \end{cases}$$

Dans la formulation précédente, chaque travail non-sommet $j \in V \setminus \{s\}$ peut être séquencé à la gauche (i.e., la variable binaire x_j^+ est mise à 1 et $j \in \alpha$) ou à la droite (i.e., la variable binaire x_j^- est égale à 1 et $j \in \beta$) du sommet s, mais pas les deux (i.e. $x_i^- + x_i^+ = 1$). Les contraintes sur les variables entières D et R permettent de déterminer les valeurs $D^{\min}$ et $R^{\max}$, en cohérence avec les formules données plus haut (i.e. les travaux de α sont classés par r_j croissant et ceux de β, par d_j croissant). La fonction objectif $z = D - R$ doit être maximisée. En effet, d'après le corollaire 1, une séquence de travaux donnant à z sa valeur optimale domine forcément celle donnant à z une valeur moindre. De plus, si $z^* \geq p_s$ alors, d'après le théorème 2, la séquence est admissible. Dans le cas contraire, i.e. $z^* < p_s$, alors ont peut affirmer qu'il n'existe pas de séquence admissible pour le problème. En cohérence avec la remarque de la section précédente, signalons que le retard algébrique de n'importe quelle séquence de travaux vérifiant les contraintes du PLNE précédent est $L_{\max} = p_s - z$.

Nous considérons à présent le problème général $1|r_j|L_{\max}$. Ici, un travail non-sommet j peut appartenir à plusieurs pyramides différentes. Nous notons $u(j)$ ($v(j)$ respectivement) l'indice de la première pyramide (la dernière pyramide) à laquelle j appartient. L'indice m désigne le nombre de sommets. L'ensemble des sommets est noté $S = \{s_1, \ldots, s_m\}$ Nous savons que l'ensemble des séquences dominantes est de la forme $\sigma = \alpha_1 \prec s_1 \prec \beta_1 \prec \cdots \prec \alpha_m \prec s_m \prec \beta_m$, où s_k est le sommet de la pyramide P_k et où tout travail non-sommet j peut être séquencé soit dans α_k, soit dans β_k, pour tout $k \in [u(j), v(j)]$.

La recherche de la séquence la plus dominante se ramène alors à résoudre le PLNE suivant :

$$\max\ z = \min_{k=1,\dots,m}(D_k - R_k - p_{s_k})$$

$$\text{s.t.}\ \begin{cases}
R_k \geq r_{s_k} & , \forall k \in [1,\ m] & (1.1)\\
D_k \leq d_{s_k} & , \forall k \in [1,\ m] & (1.2)\\
R_k \geq r_i + \sum_{\{j \in P_k | r_j \geq r_i\}} p_j x^+_{kj} & , \forall i \in V \setminus S | k = u(i) & (1.3)\\
D_k \leq d_i - \sum_{\{j \in P_k | d_j \leq d_i\}} p_j x^-_{kj} & , \forall i \in V \setminus S | k = v(i) & (1.4)\\
R_k \geq R_{k-1} + \sum_{\{j \in P_{k-1}\}} p_j x^-_{(k-1)j} + p_{s_{k-1}} & & \\
\qquad + \sum_{\{j \in P_k | r_j \geq r_i\}} p_j x^+_{kj} & , \forall k \in [2,\ m] & (1.5)\\
D_k \leq D_{k+1} - \sum_{\{j \in P_{k+1}\}} p_j x^+_{(k+1)j} - p_{s_{k+1}} & & \\
\qquad - \sum_{\{j \in P_k | d_j \leq d_i\}} p_j x^-_{kj} & , \forall k \in [1,\ (m-1)] & (1.6)\\
\sum_{k=u(i)}^{v(i)} (x^-_{ki} + x^+_{ki}) = 1 & , \forall i \in V \setminus S & (1.7)\\
x^-_{ki}\ ,\ x^+_{ki} \in \{0,1\} & , \forall k \in [1\ m], \forall i \in P_k & \\
D_k\ ,\ R_k \in \mathbb{Z} & , \forall k \in [1,\ m] &
\end{cases}$$

Dans cette formulation, la variable binaire x^+_{ki} (x^-_{ki} resp.) vaut 1 si le travail non-sommet i, appartenant à la pyramide P_k (avec $u(i) \leq k \leq v(i)$), est séquencé dans α_k (dans β_k resp.), sachant qu'il ne peut pas être séquencé dans les deux à la fois (i.e., la contrainte (1.7) est satisfaite). Les contraintes (1.1)-(1.4) sont similaires à celles figurant dans le PLNE précédent à la différence que plusieurs pyramides sont prises en compte. Les contraintes (1.5) et (1.6) permettent d'éviter les recouvrements entre les séquences $\alpha_k \prec s_k \prec \beta_k$ de chaque pyramide P_k.

Le critère z revient à maximiser la plus petite marge libre associée aux sommets. Si $z^* \geq 0$ alors la séquence optimale est admissible (les dates d'échéance de tous les travaux sont respectées), sinon il n'existe pas de solution admissible. On a toujours une stricte équivalence entre ce critère et la minimisation du plus grand retard algébrique puisque : $L_{\max} = -z$ pour toute séquence réalisable.

Cette formulation est intéressante du fait du nombre relativement faible de variables et de contraintes à prendre en compte et de l'absence de coefficient de type "big M". Faisons tout d'abord le bilan sur les contraintes. Il y a une contrainte de type (1.1) et de type (1.2) pour chaque sommet. Il y a m sommets. Il faut prendre en compte une contrainte de type (1.3), (1.4) et (1.7) pour chaque travail non-sommet. Il y a $n - m$ travaux non-sommets. Pour les contraintes de non-recouvrement de type (1.5) et (1.6), il y en a une entre chaque pyramide. Il y a m pyramides et donc $m - 1$ contraintes de chaque type. Au total, on a donc $m + 3n - 2$ contraintes à prendre en compte. Concernant le nombre de variables, on a $2m$ variables entières et $(2\sum_{k=1}^{m}(n_k))$ variables binaires, où n_k est le nombre de travaux non-sommets appartenant à la pyramide P_k. On remarque que les valeurs des variables

entières peuvent être directement déduites de celles des variables binaires.

Dans [Briand *et al.* 10], une analyse expérimentale est produite qui compare, sur des instances de problèmes $1|r_j|L_{\max}$ de grandes tailles (plusieurs milliers de travaux), le temps de calcul de la méthode arborescente de Carlier [Carlier 82] et le temps mis par le solveur commercial Cplex pour résoudre le PLNE précédent. Les résultats montrent qu'en général, même si la technique basée sur la PLNE est davantage performante sur les instances comptant moins de mille travaux, la méthode arborescente reste meilleure pour les instances de taille supérieure. De plus, au delà de 4000 travaux, lorsque le nombre de sommets devient important ($m > 400$), la taille du modèle devient si importante que Cplex se trouve souvent en défaut de mémoire pour procéder à sa résolution.

Une analyse plus approfondie montre que plus le nombre de travaux est élevé et plus le nombre d'instances générées réellement "difficiles" devient faible, ce qui désavantage la méthode basée sur la PLNE. Précisons ce que nous entendons par instance difficile. On sait que la méthode arborescente de Carlier utilise intensivement la procédure de Schrage qui permet à chaque nœud de l'arborescence, étant donné les valeurs r_i et d_i des travaux pour le nœud considéré, de construire en temps polynomial une séquence de bonne qualité vis-à-vis de la minimisation du retard algébrique. À chaque nœud, une condition suffisante permet de déduire si la solution de Schrage locale est optimale ou non. Or, dans nos tests, il arrive très fréquemment, pour les instances générées comptant plus de 1000 travaux, que la solution de Schrage initiale (i.e., celle générée à partir des valeurs initiales des r_i et des d_i) soit optimale. Dans ce cas, la méthode arborescente ne génère qu'un seul nœud optimal, cela en un temps très faible. Cplex lui, n'intégrant pas la condition suffisante d'optimalité utilisée par la méthode arborescente, met davantage de temps à conclure sur l'optimalité, cela même si on lui fournit la solution de Schrage initiale.

Cependant, nous avons montré que, dans le cas où l'analyse est restreinte aux instances difficiles, la méthode basée PLNE devenait très compétitive et le plus souvent meilleure en temps de calcul (mais pas en mémoire) que la méthode arborescente de Carlier.

2.3.4 Minimisation du nombre de travaux en retard

Un avantage connu de la PLNE est de permettre la prise en compte de nouvelles contraintes relativement aisément. Cela a motivé le travail consistant à adapter la formulation PLNE précédente, valide pour le problème $1|r_j|L_{\max}$, au problème $1|r_j|\sum U_j$, pour lequel il s'agit de déterminer une séquence admissible minimisant le nombre de travaux en retard (U_j est mis à un si le travail j est en retard).

Dans ce cadre nous avons décrit deux PLNEs permettant respectivement de déterminer une borne supérieure et une borne inférieure pour le problème $1|r_j|\sum U_j$, ces deux bornes

étant par ailleurs de bonne qualité.

Intéressons nous tout d'abord au calcul de la borne supérieur. Il est très facile d'adapter le PLNE de la section précédente pour déterminer une séquence admissible qui minimise le nombre de travaux en retard, *sous l'hypothèse que les sommets sont séquencés à l'heure.* Le PLNE correspondant est le suivant :

$$\min\ z = \sum_{\{j\in V\setminus\{t_1\dots t_m\}\}}(1-\sum_{k=u(j)}^{v(j)}(x_{jk}^- + x_{jk}^+)) + \sum_{k=1}^{m} \boldsymbol{y_{s_k}}$$

$$\text{s.t.}\quad \begin{cases}
R_k \geq r_{s_k} & , \ \forall k \in [1,\ m] & (2.1)\\
D_k \leq d_{s_k} & , \ \forall k \in [,1\ m] & (2.2)\\
R_k \geq r_i + \sum_{\{j\in P_k | r_j \geq r_i\}} p_j x_{kj}^+ & , \ \forall i \in V\setminus S | k = u(i) & (2.3)\\
D_k \leq d_i - \sum_{\{j\in P_k | d_j \leq d_i\}} p_j x_{kj}^- & , \ \forall i \in V\setminus S | k = v(i) & (2.4)\\
R_k \geq R_{k-1} + \sum_{\{j\in P_{k-1}\}} p_j x_{(k-1)j}^- + \boldsymbol{y_{s_{k-1}}} p_{s_{k-1}} & & \\
\qquad + \sum_{\{j\in P_k | r_j \geq r_i\}} p_j x_{kj}^+ & , \ \forall k \in [2,\ m] & (2.5)\\
D_k \leq D_{k+1} - \sum_{\{j\in P_{k+1}\}} p_j x_{(k+1)j}^+ - \boldsymbol{y_{s_{k+1}}} p_{s_{k+1}} & & \\
\qquad - \sum_{\{j\in P_k | d_j \leq d_i\}} p_j x_{kj}^- & , \ \forall k \in [1,\ (m-1)] & (2.6)\\
\sum_{k=u(i)}^{v(i)}(x_{ki}^- + x_{ki}^+) \leq 1 & , \ \forall i \in V\setminus S & (2.7)\\
D_k - R_k \geq p_{s_k}(1-\boldsymbol{y_{s_k}}) & , \ \forall k \in [1,\ m] & (2.8)\\
x_{ki}^-\ ,\ x_{ki}^+ \in \{0,1\} & , \ \forall k \in [1\ m], \forall i \in P_k & \\
D_k\ ,\ R_k \in \mathbb{Z} & , \ \forall k \in [,1\ m] &
\end{cases}$$

Commentons un peu ce PLNE. Tout d'abord, les contraintes (2.1)-(2.6) sont identiques aux contraintes (1.1)-(1.6) du PLNE précédent, les variables D_k et R_k étant déterminées de façon identique. Autoriser un travail non-sommet à être en retard est facile en relaxant la contrainte d'égalité (1.7) et en la remplaçant par une contrainte d'inégalité (contrainte (2.7)). Comme la séquence obtenue doit être admissible, la contrainte (2.8) est introduite, i.e., $D_k - R_k \geq p_{s_k}, \forall k = 1,\dots,m$. Cependant, cette contrainte peut être trop forte puisque lorsque deux sommets consécutifs s_k et s_{k+1} sont tels que $d_{s_{k+1}} - r_{s_k} < p_{s_{k+1}} + p_{s_k}$, le PLNE est infaisable, i.e., il n'existe pas de séquence réalisable permettant d'avoir les deux sommets séquencés à l'heure. Pour éviter cette infaisabilité, on introduit pour chaque sommet s_k une variable binaire y_{s_k} (voir la contrainte (2.8)) : y_{s_k} vaut 1 si la durée opératoire du sommet est prise en compte, 0 sinon. Ainsi, le critère $\sum U_j$ peut facilement être déterminé à partir de la connaissance des variables binaires y_{s_k}, x_{ki}^+ et x_{ki}^- puisque, si $y_{s_k} = 1$ alors le sommet s_k est en retard et si $\sum_{k=u(j)}^{v(j)}(x_{jk}^- + x_{jk}^+) = 0$, alors le travail non-sommet j est en retard.

Soulignons que la résolution du PLNE précédent donne seulement une borne supérieur

du nombre de travaux en retard, cela à cause de l'hypothèse que les sommets doivent être séquencés à l'heure (leur durée opératoire pouvant éventuellement être prise égale à 0). En effet, les séquences de la forme $\sigma = \alpha_1 \prec s_1 \prec \beta_1 \prec \cdots \prec \alpha_m \prec s_m \prec \beta_m$ ne sont pas les seules à être dominantes pour le critère $\sum U_j$. Cela est facilement illustrable sur un petit exemple comportant 4 travaux dont la structure d'intervalles est représentée sur la figure 2.5. Le travail a est le sommet et, s'il est séquencé à l'heure, alors les séquences de la forme $(d \prec b \prec c \prec \mathbf{a} \prec b \prec c \prec d)$ (les travaux b, c et d étant séquencés à gauche ou à droite de a en respectant l'ordre indiqué) sont dominantes pour la minimisation de $\sum U_j$. Si l'on suppose à présent que a est en retard (i.e., a peut être séquencé après tous les autres travaux et son intervalle peut être omis), alors b et c deviennent sommets et, toujours sous l'hypothèse que ces sommets sont séquencés à l'heure, les séquences dominantes pour $\sum U_j$ sont de la forme $(d \prec \mathbf{b} \prec d \prec \mathbf{c} \prec d)$. Comme on peut l'observer, les séquences dominantes du 2ème ensemble ne sont pas toutes incluses dans celles du premier 1er ensemble (le travail d ne pouvait pas initialement être séquencé entre b et c). Se restreindre seulement au premier ensemble, comme nous le faisons dans le PLNE précédent, conduit donc à éliminer certaines séquences pouvant potentiellement être optimales. Le résultat obtenu lors de la résolution du PLNE correspond donc bien à une borne supérieure.

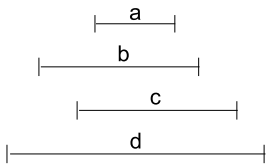


FIGURE 2.5 – Une structure d'intervalles

Néanmoins, nous avons mis en évidence un cas pour lequel l'ensemble initial des séquences dominantes contient nécessairement *toutes* les séquences dominantes pour le critère $\sum U_j$. Cette propriété est obtenue lorsque toutes les pyramides d'un problème sont *parfaites*. Nous décrivons dans ce qui suit cette notion et nous montrons comment elle permet d'aboutir au calcul d'une borne inférieure très serrée.

La définition d'une pyramide parfaite est donnée ci-dessous.

Définition 5. *Une s-pyramide P_s, associée au sommet s d'une structure d'intervalles $< I, C >$ est dite parfaite si, quels que soient deux intervalles i et $j \in P_s$, les relations $overlaps(i, j)$ et $overlaps(j, i)$ sont fausses.*

Autrement dit, une pyramide est parfaite si ses intervalles s'imbriquent les uns dans les autres sans que jamais deux d'entre eux ne se chevauchent. Dans la suite du texte, par abus de langage, nous qualifions prétentieusement de "parfait" un problème caractérisé par une structure d'intervalles formée exclusivement de pyramides parfaites.

Le problème $1|r_i, nested|-$ est un exemple de problème parfait. La contrainte *nested* indique que les dates de disponibilité et d'échéance des travaux sont telles que $r_i < r_j \Rightarrow$

$d_i \geq d_j$ ou $d_i > d_j \Rightarrow r_i \leq r_j$. Les fenêtres d'exécution des travaux sont donc imbriquées les unes dans les autres. La structure d'intervalles correspondant à ce problème est donc une pyramide parfaite unique.

Une propriété sympathique d'un problème parfait est que l'on peut facilement démontrer que, si V' est un sous-ensemble de travaux de V (i.e., $V' \subseteq V$), alors l'ensemble des séquences dominantes associé à V' est toujours strictement inclus dans l'ensemble des séquences dominantes associé à V. En d'autres termes, en recherchant parmi les séquences de la forme $\sigma = \alpha_1 \prec s_1 \prec \beta_1 \prec \cdots \prec \alpha_m \prec s_m \prec \beta_m$, celle minimisant le nombre de travaux en retard, on obtient une solution optimale pour le problème général (et non plus seulement une borne supérieure).

Si on considère un problème parfait, le PLNE à résoudre pour trouver la séquence optimisant $\sum U_j$ est le suivant :

$$\min\ z = \sum_{\{j \in V \setminus \{t_1 \ldots t_m\}\}} (1 - \sum_{k=u(j)}^{v(j)} (x_{jk}^- + x_{jk}^+)) + \sum_{k=1}^{m} \boldsymbol{y_{s_k}}$$

$$\text{s.t.} \begin{cases}
R_k \geq r_{s_k} + \boldsymbol{y_{s_k}}(\boldsymbol{r_{n_k}} - \boldsymbol{r_{s_k}}) & , \ \forall k \in [1,\ m] & (3.1) \\
D_k \leq d_{s_k} + \boldsymbol{y_{s_k}}(\boldsymbol{d_{n_k}} - \boldsymbol{d_{s_k}}) & , \ \forall k \in [,1\ m] & (3.2) \\
R_k \geq r_i + \boldsymbol{(1 - x_{ik}^+)}(\boldsymbol{r_{n_k}} - \boldsymbol{r_i}) & & \\
\qquad + \sum_{\{j \in P_k | r_j \geq r_i\}} p_j x_{kj}^+ & , \ \forall i \in V \setminus S | k = u(i) & (3.3) \\
D_k \leq d_i + \boldsymbol{(1 - x_{ik}^-)}(\boldsymbol{d_{n_k}} - \boldsymbol{d_i}) & & \\
\qquad - \sum_{\{j \in P_k | d_j \leq d_i\}} p_j x_{kj}^- & , \ \forall i \in V \setminus S | k = v(i) & (3.4) \\
R_k \geq R_{k-1} + \sum_{\{j \in P_{k-1}\}} p_j x_{(k-1)j}^- + \boldsymbol{y_{s_{k-1}}} p_{s_{k-1}} & & \\
\qquad + \sum_{\{j \in P_k | r_j \geq r_i\}} p_j x_{kj}^+ & , \ \forall k \in [2,\ m] & (3.5) \\
D_k \leq D_{k+1} - \sum_{\{j \in P_{k+1}\}} p_j x_{(k+1)j}^+ - \boldsymbol{y_{s_{k+1}}} p_{s_{k+1}} & & \\
\qquad - \sum_{\{j \in P_k | d_j \leq d_i\}} p_j x_{kj}^- & , \ \forall k \in [1,\ (m-1)] & (3.6) \\
\sum_{k=u(i)}^{v(i)} (x_{ki}^- + x_{ki}^+) \leq 1 & , \ \forall i \in V \setminus S & (3.7) \\
D_k - R_k \geq p_{s_k}(1 - \boldsymbol{y_{s_k}}) & , \ \forall k \in [1,\ m] & (3.8) \\
x_{ki}^- \ , \ x_{ki}^+ \in \{0, 1\} & , \ \forall k \in [1\ m], \forall i \in P_k & \\
D_k \ , \ R_k \in \mathbb{Z} & , \ \forall k \in [,1\ m] &
\end{cases}$$

avec :

- $r_{n_k} = \min_{\{j \in P_k\}} r_j$;
- $d_{n_k} = \max_{\{j \in P_k\}} d_j$;

Ce dernier PLNE est totalement équivalent au PLNE qui précède sauf que les termes $y_{s_k}(r_{n_k} - r_{s_k})$, $y_{s_k}(d_{n_k} - d_{s_k})$, $(1 - x_{ik}^{+})(r_{n_k} - r_i)$ et $(1 - x_{ik}^{-})(d_{n_k} - d_i)$ ont été respectivement ajoutés dans les contraintes (3.1), (3.2), (3.3) et (3.4) pour pouvoir les désactiver lorsque nécessaire. Dans les contraintes (3.1) et (3.2), ces termes permettent de désactiver les contraintes associées à un sommet s_k lorsque ce sommet est décidé être en retard (i.e., $y_{s_k} = 1$). En effet, dans ce cas, on obtient les contraintes $R_k \geq r_{n_k}$ d'une part, et $D_k \leq d_{n_k}$ qui sont toujours valides (i.e., r_{n_k} est une borne inférieure de R_k et d_{n_k} est une borne supérieure de D_k. Dans le PLNE précédent, il n'était pas nécessaire de rajouter ces termes puisque les sommets sont supposés être toujours à l'heure (même si leur durée opératoire peut être mise à 0), ce qui n'est pas le cas ici. Le raisonnement est identique pour les termes ajoutés dans les contraintes (3.3) et (3.4) : si on considère un travail i en retard, et si tous les travaux dont les intervalles d'exécution sont inclus dans celui de i sont également en retard (y compris les sommets), les contraintes associées à i nécessitent d'être désactivées pour éviter d'imposer que $R_k \geq r_i$ et $D_k \leq d_i$ alors que ce n'est pas nécessaire.

Étant donné une instance quelconque de problème $1|r_j|\sum U_j$, il est toujours possible de créer un problème parfait en diminuant certaine dates de disponibilité des travaux ou, inversement, en augmentant certaines dates d'échéance des travaux. Dans la mesure où, en procédant ainsi, les intervalles d'exécution des travaux s'élargissent, on obtient bien une relaxation du problème initial. Donc, comme on est capable de résoudre le problème relâché de façon optimale, celui-ci étant parfait, la valeur optimale obtenue est bien une borne inférieure du critère $\sum U_j$.

Nous avons distingué trois façons de relâcher un problème pour le rendre parfait selon si :

- seules les dates de disponibilité des travaux sont diminuées ;
- seules les dates d'échéance des travaux sont augmentées ;
- on adopte une stratégie mixte où à la fois les dates d'échéance sont augmentées et les dates de disponibilité sont diminuées dans le but de minimiser l'ampleur des élargissements des intervalles d'exécution (le but étant de s'écarter le moins possible du problème initial dans l'espoir d'obtenir une borne inférieure davantage proche de la valeur optimale).

L'algorithme correspondant à la stratégie mixte est donné plus loin (algorithme 1). Sa complexité temporelle est en $O((n \log n))$, c'est-à-dire la complexité liée au tri des dates de disponibilité et d'échéance des travaux membres d'une pyramide (les modifications des dates de disponibilité et d'échéance se faisant en $O(n)$). Comme il y a m pyramides, une instance de problème peut être rendu parfaite en $O(mn \log n)$. Le principe est le suivant. On considère les travaux par dates de disponibilité croissante, d'une part, et par date d'échéance décroissante, d'autre part. On détermine à chaque étape le travail i (le travail j respectivement), non encore traité, ayant la date de disponibilité la plus petite (la date d'échéance la plus grande respectivement). Trois cas sont alors considérés. Si les travaux i

Algorithme 1 "Perfectisation" d'une pyramide (stratégie mixte)

1: **Function MakePerfect** (P_k)
2: $Rtab \leftarrow$ **IncreasingSort**($\{\mathbf{r_j}|\mathbf{j} \in \mathbf{P_k}\}$) ;
3: $Dtab \leftarrow$ **DecreasingSort**($\{\mathbf{d_j}|\mathbf{j} \in \mathbf{P_k}\}$) ;
4: $left \leftarrow 1$;
5: $right \leftarrow 1$;
6: **while** $left \leq n_k$ AND $right \leq n_k$ **do**
7: $r_i \leftarrow Rtab[left]$;
8: $d_j \leftarrow Dtab[right]$;
9: $\Delta R \leftarrow r_j - r_i$;
10: $\Delta D \leftarrow d_i - d_j$;
11: **if** $i = j$ OR ($\Delta R \geq 0$ AND $\Delta D \leq 0$) OR ($\Delta R \leq 0$ AND $\Delta D \geq 0$) **then**
12: $left \leftarrow left + 1$;
13: $right \leftarrow right + 1$;
14: **else if** $\Delta R < \Delta D$ **then**
15: **if** $\Delta R \geq 0$ **then**
16: $r_j \leftarrow r_i$;
17: $right \leftarrow right + 1$;
18: **else**
19: $r_i \leftarrow r_j$;
20: $left \leftarrow left + 1$;
21: **end if**
22: **else**
23: **if** $\Delta R \geq 0$ **then**
24: $d_i \leftarrow d_j$;
25: $left \leftarrow left + 1$;
26: **else**
27: $d_j \leftarrow d_i$;
28: $right \leftarrow right + 1$;
29: **end if**
30: **end if**
31: **end while**

et j sont identiques ou si les intervalles d'exécution de i et j sont inclus l'un dans l'autre (cf. ligne 11), alors on respecte bien les propriétés d'une pyramide parfaite et on peut passer aux travaux i et j suivants. Sinon, les intervalles d'exécution de i et j se chevauchent et on ne respecte pas les propriétés d'une pyramide parfaite. Il faut donc soit réduire une date de disponibilité, soit augmenter une date d'échéance, de sorte que les intervalles d'exécution soient inclus l'un dans l'autre. Dans le cas où l'écart entre r_i et r_j est plus petit que l'écart entre d_i et d_j (cf. ligne 14), le changement le moins coûteux consiste à diminuer la date de disponibilité la plus grande parmi celles de i et j de sorte que $r_i = r_j$. À l'inverse (cf. ligne 22), dans le cas où l'écart entre r_i et r_j est plus grand que l'écart entre d_i et d_j, le changement le moins coûteux consiste à augmenter la date d'échéance la plus petite parmi celles de i et j de sorte que $d_i = d_j$. L'algorithme se termine quand les n_k travaux membres de la pyramide P_k ont tous été envisagés deux à deux.

Nous avons évalué les performances de nos modèles en considérant les instances proposées par Baptiste et al. (voir [Baptiste *et al.* 03]) ayant entre 80 et 160 travaux. Les résultats sont bons puisque l'on montre que dans la plupart des cas les bornes inférieure et supérieure sont égales, ce qui permet de conclure sur l'optimalité. Ce travail a fait l'objet d'une soumission à revue, aujourd'hui en cours de révision [Ourari *et al.* 09], la version courante de cet article étant par ailleurs insérée dans la section 5.

2.4 Flowshop de permutation à 2 machines

On s'intéresse dans cette section au problème flowshop de permutation à 2 machines dans lequel on souhaite minimiser le makespan. Il s'agit d'un problème de séquencement tout aussi académique que le problème à 1 machine mais, contrairement à ce dernier, il n'est pas NP difficile. Il est en effet possible de trouver en $O(n \log n)$ une séquence optimale en appliquant la célèbre règle de Johnson $\min(p_{i1}, p_{j2}) \leq \min(p_{j1}, p_{i2}) \iff i \prec j$. L'application de cette règle conduisant à la détermination d'une unique séquence, nous nous sommes attachés à étudier comment caractériser un ensemble flexible de solutions optimales. Le problème consiste alors à obtenir, dans un temps le plus court possible, un ensemble de solutions le plus vaste possible, ceci en évitant l'énumération fortement combinatoire des solutions de l'ensemble.

De nombreux travaux se sont intéressés à la détermination d'un ensemble de séquences optimales pour le problème $F2|prmu|C_{\max}$. Citons tout d'abord les travaux de Bellman, Esogbue et Nabeshima [Belman *et al.* 82] qui présentent un algorithme, baptisé GWR (*General Working Rule*), énumérant toutes les séquences satisfaisant la règle de Johnson. Dans [Billaut & Lopez 98], une approche est également proposée pour lister, sur la base des séquences produites par la règle GWR, *toutes* les séquences optimales d'un problème $F2|prmu|C_{\max}$. Un autre algorithme d'énumération de toutes les séquences optimales est aussi présenté dans [Benoît & Billaut 00], basé sur la notion de *séquence maximale*. Remarquons que dans les deux derniers cas, seuls des problèmes de faible taille sont traités du fait de la complexité sous-jacente à l'énumération de toutes les solutions. Enfin, Billaut et Esswein [Esswein & Billaut 02] se sont intéressés à la caractérisation de solutions pour $F2|prmu|C_{\max}$, par l'introduction de flexibilité séquentielle dans une solution optimale de Johnson.

Dans le cadre de ce mémoire, deux approches sont décrites. La première, étudiée en 1999 à l'occasion du stage de DEA de Anis Ziadi, consiste à mettre en évidence des conditions nécessaires d'admissibilité qui permettent de filtrer l'espace de solutions[1]. La seconde, mise en évidence en 2002 dans le cadre de la thèse de Hoang Trung La, utilise l'analyse d'intervalles pour mettre en évidence une condition suffisante d'admissibilité. Nous verrons que cette dernière est particulièrement remarquable puisqu'elle permet de caractériser, en temps polynomial, un vaste ensemble de séquences incluant nécessairement *toutes* les séquences optimales de Johnson.

1. Cette première approche, si elle a l'avantage d'être exhaustive (toutes les solutions admissibles sont caractérisées), présente l'inconvénient d'être beaucoup moins efficace du point de vue des temps de calcul que la seconde. J'ai tout de même choisi de la présenter dans ce mémoire car il s'agit d'un travail original, mené durant une période clé de mon parcours de chercheur marquant le passage de mon thème de recherche initial, le *pilotage de systèmes de production* (développé durant mon doctorat), à celui de l'*ordonnancement*.

2.4.1 Mise en évidence de conditions nécessaires

Un travail entrepris trois ans après mon doctorat fut d'étudier comment le formalisme des réseaux de Petri pouvait être utilisé pour résoudre des problèmes d'ordonnancement. Ayant en effet intensivement utilisé les réseaux de Petri au cours de mon doctorat, aborder la problématique d'ordonnancement avec cet angle d'attaque me parut à l'époque assez logique.

Nous nous sommes intéressés principalement à 2 problèmes de séquencement : celui de séquencement de voitures sur une ligne d'installation d'équipements optionnels (connu sous le nom de *car-sequencing*) et celui du flowshop de permutation à 2 machines. Les techniques de résolution utilisées pour ces 2 problèmes étant identiques et relativement génériques, seul le travail relatif à la 2ème étude est décrit ici.

L'idée de base consiste à décrire le problème à résoudre sous forme de problème de satisfaction de contraintes (noté CSP pour *Constraint Satisfaction Problem*), puis de le modéliser dans le formalisme des réseaux de Petri, pour enfin utiliser une procédure de renforcement de consistance basée sur un calcul d'invariants de transition (voir l'article [Briand 00]).

On considère un CSP $Y = (X, D, C, \overline{R})$ où :

- $X = \{X_1, \ldots, X_n\}$ est l'ensemble des variables de décision ;
- $D = \{D_1, \ldots, D_n\}$ est l'ensemble fini des domaines des variables, où le domaine D_i est associé à la variable X_i ;
- $C = \{C_1, \ldots, C_m\}$ est l'ensemble des contraintes, où une contrainte $C_j = \{X_{j1}, \ldots X_{jk}\} \subseteq X$ est identifiée par les tuples de variables qu'elle implique ;
- $\overline{R} = \{R_1, \ldots, R_m\}$ est l'ensemble de relations où $R_j \subseteq D_{j1} \times \cdots \times D_{jk}$ spécifie les tuples de valeurs qui ne satisfont pas la contrainte C_j.

Une façon de modéliser un CSP par un réseau de Petri $(P, T^D, Pre, Post, m_0)$ consiste à :

- créer une transition $t \in T^{D_i}$ pour chaque affectation distincte de variable ;
- associer une place initialement marquée d'un jeton à chaque variable X_i de X et créer un arc entre la place et toute transition correspondant à une affectation possible de la variable (i.e. $Pre[X_i, t] = 1 \forall t \in T^{D_i}$) ;
- associer une place à chaque tuple de valeurs interdites de $\overline{R}$, cette place ayant un marquage égal à $(|C| - 1)$, $|C|$ étant l'arité de la contrainte C (c'est à dire le nombre de variables impliquées dans la contrainte), et créer un arc entre la place et chaque transition associée à une valeur du tuple interdit.

Ce principe est illustré sur l'exemple de CSP de la figure 2.6, correspondant à un problème de coloration de graphe. La modélisation par réseau de Petri de ce CSP est décrite sur la figure 2.7.

X	D	C	$\overline{R}$
X_1	{a,b}	C_{12}	{aa,bb}
X_2	{a,b}	C_{23}	{aa,bb}
X_3	{a,b}		

FIGURE 2.6 – Un exemple simple de CSP

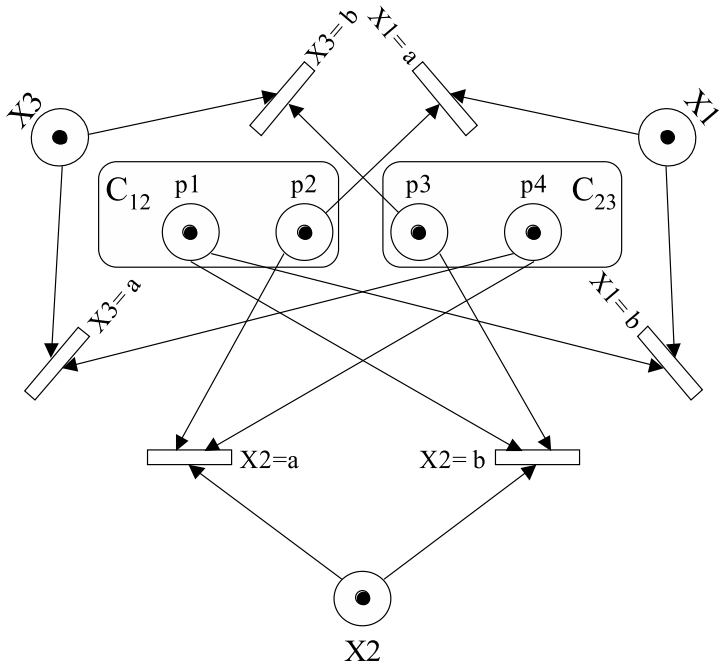


FIGURE 2.7 – Modélisation par réseau de Petri

Une propriété évidente de cette modélisation, liée au fait que tout tir de transition consomme des jetons sans jamais en produire, est que le graphe des marquages accessibles du réseau est un treillis dans lequel chaque feuille correspond à un marquage puits. D'autre part, une feuille correspond à une solution admissible si et seulement si les marques de toutes les places associées aux variables sont nulles (i.e. toutes les variables ont été instanciées). Dans le cas contraire, le marquage puits correspond à une instanciation partielle des variables, bloquante du point de vue du respect des contraintes puisqu'il n'est plus possible d'instancier de variables.

Le langage du réseau de Petri contient donc toutes les solutions admissibles, et également d'autres solutions partielles, ne pouvant pas mener à une solution admissible.

Le renforcement de contraintes consiste à enrichir l'espace des contraintes de sorte à empêcher le réseau d'évoluer vers une solution partielle indésirable. Si ce renforcement est

complet alors, plus aucune solution partielle initiale n'est accessible et le réseau ne peut donc plus évoluer que vers un marquage puits admissible.

Dans [Briand 00], une technique de renforcement est proposée, fondée sur la notion originale de *Marquage Critique Maximum*. Un marquage m_c est dit critique si, lorsqu'il est atteint, au moins une variable ne peut pas être assignée. Un tel marquage est dit maximal si, quel que soit le marquage m tel que $m > m_c$, la propriété précédente n'est plus vraie. Étant donne un marquage critique, il est possible, par un calcul d'invariant de transition, de déterminer s'il existe une séquence de tirs permettant oui ou non de l'atteindre. Dans l'affirmative, la séquence de tirs correspond à un nouveau tuple de valeurs interdit. Il est alors possible, par l'ajout d'une place dans le réseau, de l'interdire. Lorsque plus aucun marque critique maximum ne peut être atteint, le langage associé au réseau de Petri contient alors toutes les solutions réalisables.

Pour illustration, la figure 2.8 montre pour l'exemple de CSP précédent, un marquage puits non admissible (devant être interdit). Le réseau de la figure 2.9 montre comment, après ajout de 2 places (représentée en gras sur la figure), on peut garantir que toute évolution du réseau conduira nécessairement vers une solution admissible.

Évidemment, la détermination de tous les marquages critiques étant complexe et le calcul des séquences de tir étant également très combinatoire, il n'est pas possible d'utiliser une telle méthode sur des larges instances.

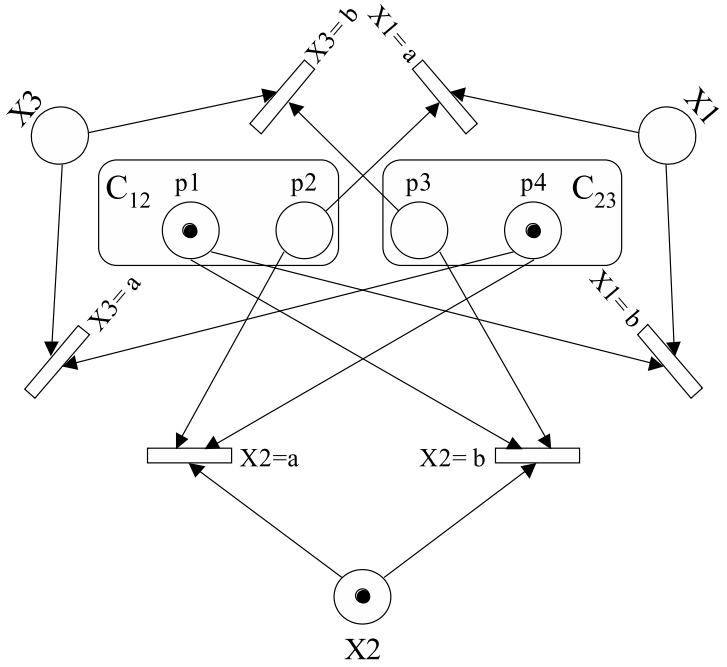


FIGURE 2.8 – Un marquage puits non admissible

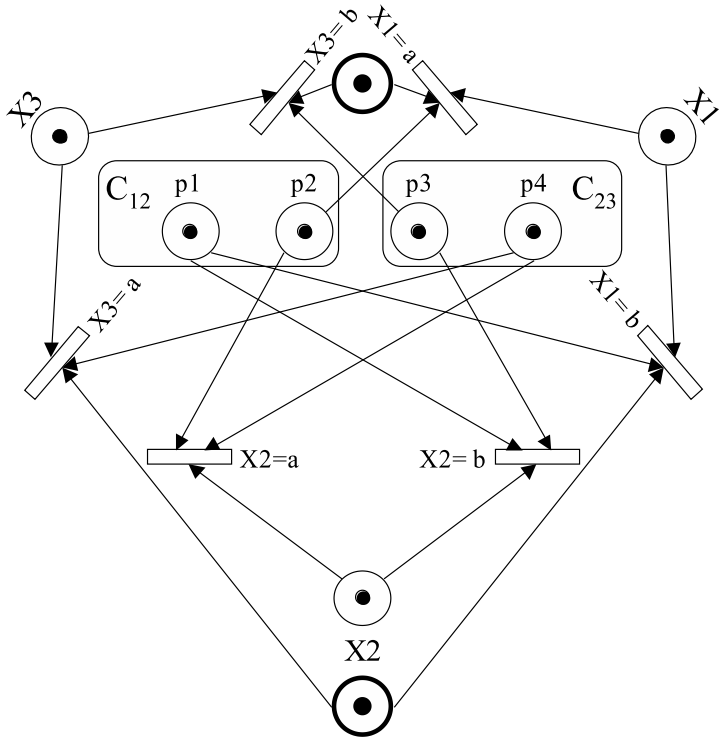


FIGURE 2.9 – Ensemble de solutions admissibles

Cette technique de résolution a été appliquée au flowshop de permutation à 2 machines. Les variables du CSP correspondent aux travaux à séquencer. Leur domaine est un intervalle de rang (i.e. $\mathrm{Dom}(X_i) = \{1 \dots n\} \forall i$). Ainsi que l'illustre la figure 2.10 sur un exemple académique à 3 travaux, il est aisé de représenter des contraintes de type :

- 2 travaux ne peuvent avoir les mêmes rangs ;
- précédence interdite ;
- séquence interdite.

On suppose que la durée totale d'exécution (makespan) optimale est connue. Une condition nécessaire et suffisante évidente pour affirmer qu'une séquence est admissible est de garantir qu'il n'existe pas de chevauchement entre les tâches i_1 et i_2 d'un même travail i, $\forall i$, lorsque les travaux sur M_1 sont calés au plus tôt et ceux sur M_2, au plus tard (cf figure 2.11).

Sur la base de cette condition, il est possible, sans énumérer toutes les séquences, de déterminer quelles sont les séquences partielles menant nécessairement à une incohérence. On sait que la tâche i_1 peut commencer à la date x s'il existe au moins un vecteur λ tel que :

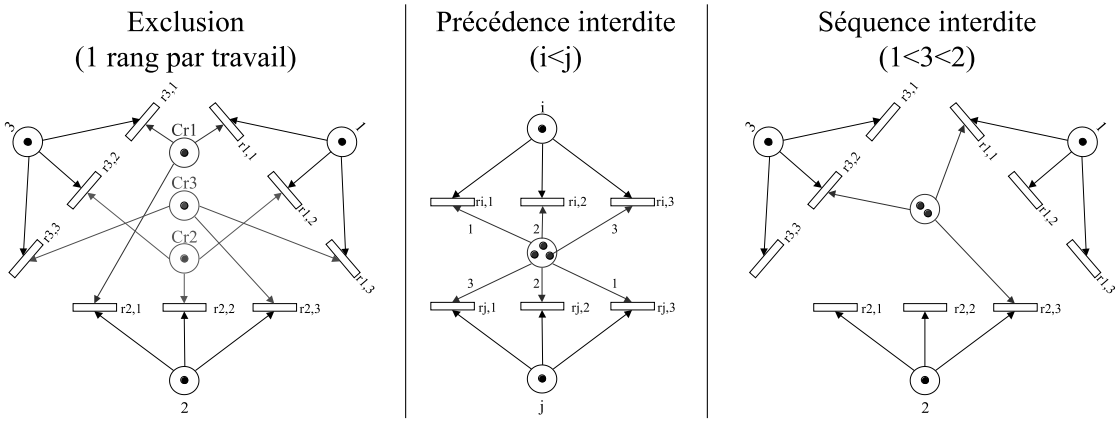


FIGURE 2.10 – Exemple de modélisation de contraintes

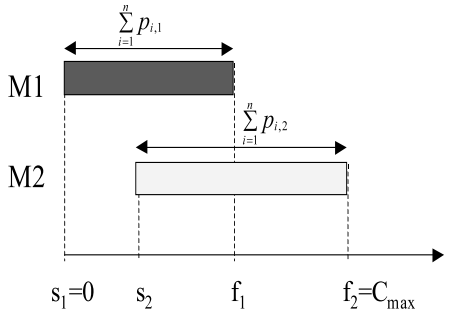


FIGURE 2.11 – Structure d'une solution ($C_{\max}$ connu)

$$\sum_{j=1,j\neq i}^{n} \lambda_j p_{j1} = x$$

Pour tout vecteur λ satisfaisant l'équation précédente, il est possible de déduire le rang du travail i puisque : $\text{rang}(i) = \sum \lambda_j + 1$. De plus on connaît les travaux précédant i et on peut donc aisément vérifier s'il s'agit d'une séquence partielle admissible ou non. Dans l'affirmative, elle doit être éliminée. En appliquant cette procédure pour tout i et tout $x \in 0 \ldots \sum p_{i1}$, on peut déduire toutes les séquences partielles interdites.

Appliquons ces principes à l'exemple élémentaire défini sur la table 2.2. On résout le système $\lambda_1 + \lambda_2 + 2\lambda_3 = x$.

On déduit :

x=0 3 de rang 1 interdit ;

x=1 3 de rang 2 précédé par 1, ou suivi par 2, interdit ;

x=2 2 de rang 2 précédé par 3, ou suivi par 1, interdit ;

Travaux	1	2	3
p_{i1}	1	1	2
p_{i2}	1	2	1

C_{max}=5

TABLE 2.2 – Un exemple de flowshop à 3 travaux

1 de rang 2 précédé par 3, ou suivi par 2, interdit ;

x=3 2 de rang 3 interdit.

Ces contraintes peuvent être ajoutées sur le réseau de Petri associé à ce problème. On obtient alors le réseau de la figure 2.12 dont le langage caractérise toutes les solutions.

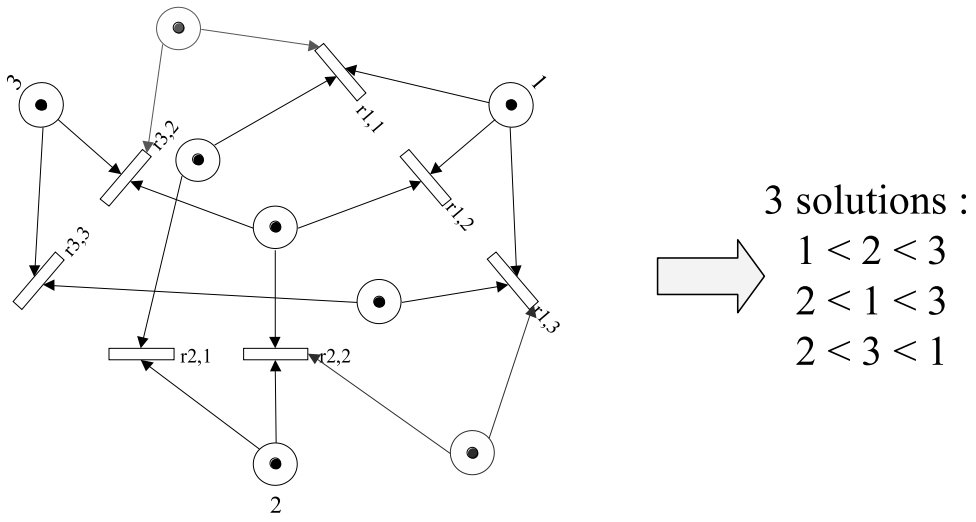


FIGURE 2.12 – Réseau de Petri obtenu après renforcement

L'approche précédente a été appliquée à des problèmes comptant jusqu'à plusieurs dizaines de travaux. Toutefois, au delà, la combinatoire devenant trop forte, les temps de calcul sont prohibitifs.

2.4.2 Mise en évidence de conditions suffisantes

Nous montrons dans cette partie comment mettre en évidence un vaste ensemble de solutions optimales pour le problème $F2|prmu|C_{\max}$, en temps polynomial, à l'aide d'une analyse d'intervalles.

Nous considérons pour cela deux structures d'intervalles, $< I_1, C_1 >$ et $< I_2, C_2 >$, dépendantes des durées opératoires p_{j1} et p_{j2}. On s'intéresse dans ce qui suit aux bases de ces structures et aux b-pyramides qu'elles caractérisent.

- I_1 est l'ensemble des intervalles associés aux travaux $j \in T$ tels que $p_{j1} \leq p_{j2}$ où un intervalle $i_j \in I_1$ associé au travail j est caractérisé par $i_j = [p_{j1}, p_{j2}]$;
- I_2 est l'ensemble des intervalles associés aux travaux $j \in T$ tels que $p_{j2} \leq p_{j1}$ où un intervalle $i_j \in I_2$ associé au travail j est caractérisé par $i_j = [p_{j2}, p_{j1}]$.

Un travail j tel que $p_{j1} = p_{j2}$ appartient donc aux deux structures d'intervalles. Considérons l'exemple du tableau 2.3 dont les structures d'intervalles sont représentées sur la figure 2.13. Les bases de $< I_1, C_1 >$ sont $b_1 = 1$, $b_2 = 2$ et $b_3 = 3$. La structure d'intervalles $< I_2, C_2 >$ ne contient qu'une seule base $b_4 = 4$. Les b-pyramides de ces deux structures sont donc $P_1 = \{5, 6\}$, $P_2 = \{6, 7\}$, $P_3 = \{8\}$ et $P_4 = \{7, 9\}$. Le travail 7 étant tel que $p_{71} = p_{72} = 7$, il appartient à la fois à $< I_1, C_1 >$ et à $< I_2, C_2 >$.

Travaux i	1	2	3	4	5	6	7	8	9
p_{i1}	1	3	8	8	2	4	7	9	5
p_{i2}	6	8	12	2	4	5	7	11	3

TABLE 2.3 – Un problème $F2|prmu|C_{\max}$

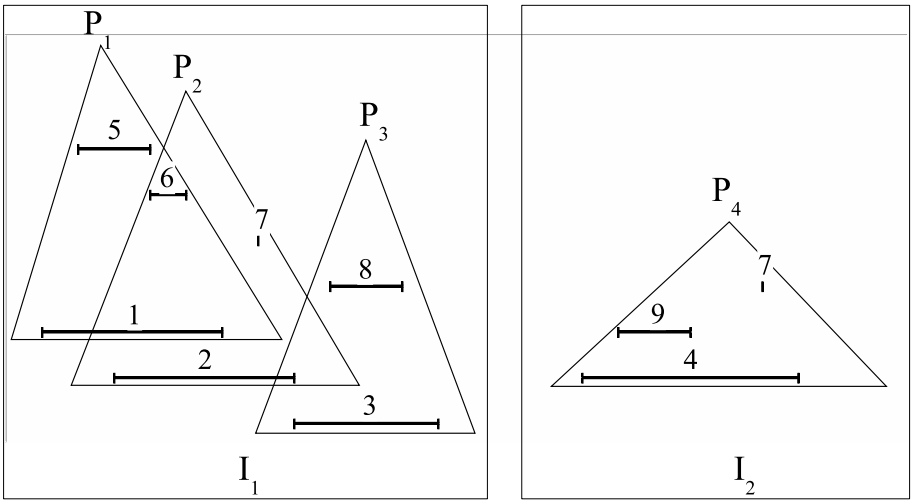


FIGURE 2.13 – Structures d'intervalles du problème du tableau 2.3

Nous supposons que les n_1 bases de $< I_1, C_1 >$ sont indexées selon l'ordre croissant des p_{j1} (dans un ordre arbitraire en cas d'égalité). La sous-séquence σ_j, avec $j \in [1, n_1]$, désigne l'ensemble des travaux placés entre b_j et b_{j+1}. Similairement, nous supposons que les n_2 bases de $< I_2, C_2 >$ sont indexées, à partir de l'index n_1+1, selon l'ordre décroissant des p_{j2} (dans un ordre arbitraire en cas d'égalité). La sous-séquence σ_k, avec $k \in [n_1 + 1, n_1 + n_2]$, correspond à l'ensemble des travaux placés entre les bases b_{k-1} et b_k.

Nous nous intéressons aux séquences de la forme $b_1 \prec \sigma_1 \prec b_2 \prec \sigma_2 \prec \cdots \prec b_{n_1} \prec \sigma_{n_1} \prec \sigma_{n_1+1} \prec b_{n_1+1} \prec \cdots \prec \sigma_{n_1+n_2} \prec b_{n_1+n_2}$ comme illustré sur la figure 2.14. La fonction

$u(j)$ (resp. $v(j)$) indique l'index de la base qui caractérise la première (resp. la dernière) b-pyramide à laquelle le travail j appartient.

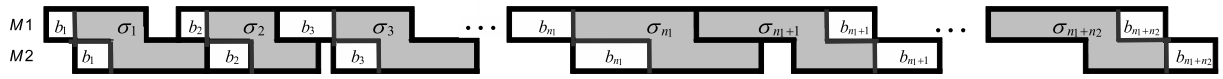


FIGURE 2.14 – Structure générale d'une séquence

Le théorème suivant peut alors être énoncé [Briand *et al.* 06a].

Théorème 3. *Toute séquence de travaux telle que :*

1. *les bases de* $< I_1, C_1 >$ *et* $< I_2, C_2 >$ *sont séquencées dans l'ordre croissant de leur indice ;*
2. *un travail* j*, non base, est placé dans n'importe quelle sous-séquence comprise entre* $\sigma_{u(j)}$ *et* $\sigma_{v(j)}$*, dans n'importe quel ordre ;*

est optimale.

Ce théorème définit une condition suffisante d'optimalité permettant de caractériser un ensemble de séquences $\mathcal{S}_{\text{opt}}$, toutes optimales. L'algorithme permettant de déterminer les bases et leur b-pyramide est en $O(n^2)$.

Si on ne tient pas compte de toutes les permutations possibles au sein d'une sous-séquence σ_i, le nombre de séquences optimales peut être approximé par une formule identique à celle décrite dans la partie précédente :

$$card(\mathcal{S}_{\text{opt}}) = \prod_{q=1}^{N} (q+1)^{n_q}$$

où n_q désigne le nombre d'intervalles appartenant exactement à q pyramides et N le nombre total de pyramides.

Une propriété intéressante de l'ensemble $\mathcal{S}_{\text{opt}}$ est son insensibilité aux variations des durées opératoires. Autrement dit, même si les durées opératoires varient, on peut affirmer que, tant que cette variation reste "compatible" avec la structure d'intervalles initiale, alors il existe toujours des solutions optimales dans $\mathcal{S}_{\text{opt}}$ (toutefois la valeur de l'optimum, elle, peut changer).

Une autre propriété intéressante est que l'ordre partiel défini par le théorème est une extension de l'ordre partiel découlant de la règle de Johnson. Ainsi, *toute séquence vérifiant*

la règle de Johnson est nécessairement incluse dans l'ensemble de séquences caractérisées par le théorème [Briand *et al.* 06a].

Revenons à l'exemple de la figure 2.13. L'application du théorème permet de mettre en évidence les affectations décrites sur la figure 2.15. Comme l'ordre dans les sous-séquences σ_i peut être quelconque, treize séquences optimales sont caractérisées (cf. figure 2.16). En comparaison, l'ordre partiel défini par la règle de Johnson n'en caractériserait que deux. Pour les valeurs de durées opératoires indiquées dans le tableau 2.3, la durée totale optimale est $C^*_{\max} = 59$ pour chacune des treize séquences.

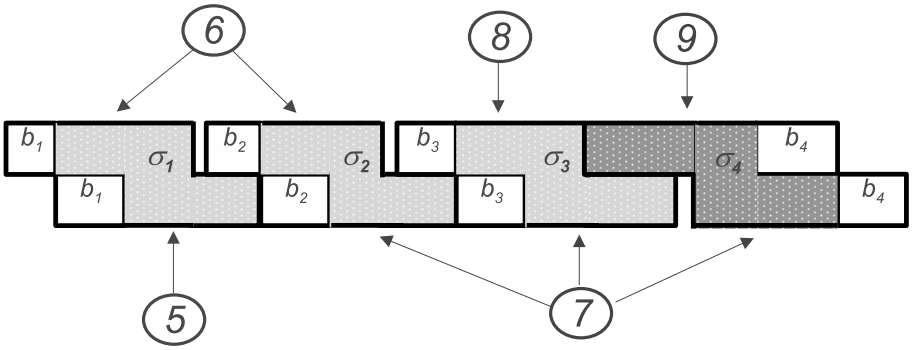
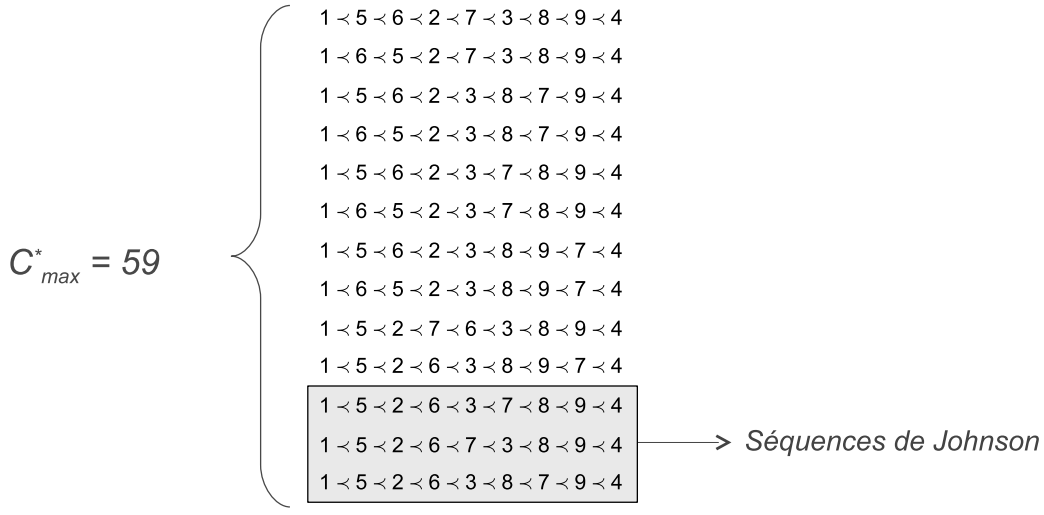


FIGURE 2.15 – Possibilité d'affectation pour l'exemple de la figure 2.13



h

FIGURE 2.16 – Ensemble des séquences caractérisées par le théorème

2.5 Flexibilité et robustesse

La plupart des méthodes d'ordonnancement opèrent sous l'hypothèse simplificatrice que les données du problème à résoudre sont déterministes : l'ensemble des tâches à ordonnancer est fini et fixé initialement, les durées opératoires, les dates de disponibilité et d'échéance des tâches sont connues et stables, ainsi que les capacités des ressources. Précisons que, si ces hypothèses simplifient considérablement le problème à résoudre, ce dernier n'en demeure pas moins fortement combinatoire et souvent non solvable en temps polynomial.

Face à une demande industrielle de plus en plus forte, et malgré la complexité sous-jacente, de nombreux chercheurs ont entrepris de remettre en cause ces hypothèses déterministes. En effet, il est tout de même difficile d'omettre que les ordonnancements produits sont avant tout destinés à être mis en œuvre dans un environnement dont la caractéristique principale est d'être perturbé. Les paramètres d'un problème d'ordonnancement fluctuent donc dans le temps. Dans ce contexte, comment assurer qu'une solution, bonne ou optimale à l'instant t, sera toujours bonne à l'instant $t+\delta t$, voire même simplement cohérente ? Pour répondre à cette question, il est nécessaire de reconsidérer la fonction ordonnancement pour la replacer dans un contexte dynamique.

Ce constat a justifié l'émergence d'une nouvelle thématique de recherche appelée *Ordonnancement Robuste* [Herroelen *et al.* 99]. En France, un groupe de travail nommé *Flexibilité et Robustesse en Ordonnancement* (FRO), animé par E. Sanlaville (LIMOS), s'est constitué pour accompagner et prendre pleinement part au mouvement international. J'ai participé à la vie de ce groupe depuis sa constitution. Ce groupe est un sous-groupe du GOThA (*Groupement de recherche en Ordonnancement Théorique et Appliqué* - voir `http://www-desir.lip6.fr/~pascual/gotha`), lui même membre du GDR RO (`http://www.lip6.fr/RO`).

Un article court paru dans le bulletin de la ROADEF [GOThA 02], un article long mis en ligne à l'adresse `http://liste.cines.fr/d_read/gotha/groupeflexibilité` et un livre rédigé de façon collégiale [Billaut *et al.* 05] ont émané de cette communauté FRO (ce livre ayant récemment été traduit par les auteurs en anglais [Billaut *et al.* 08]). Ces documents mettent en évidence la pluralité des approches robustes en ordonnancement. Celles-ci se distinguent par la nature des incertitudes considérées (et la façon de les modéliser), par le type de critère envisagé, par le fait qu'elles manipulent ou non un ordonnancement de référence (servant de guide tout au long de la mise en œuvre), et selon qu'elles tentent d'anticiper ou non les éventuelles perturbations.

Dans ce qui suit, on s'intéresse aux approches qui, dans le but de résister aux perturbations, prônent un accroissement de la flexibilité intrinsèque à une solution d'ordonnancement. La flexibilité temporelle, déjà naturellement présente au sein de la solution, peut par exemple être volontairement augmentée pour accroître la robustesse, soit

par introduction de temps morts entre certaines tâches [Goldratt 97, Mehta & Uzsoy 98], soit par surdimensionnement des durées opératoires [Gao 95], soit encore par augmentation des marges temporelles des activités [Davenport *et al.* 01, Herroelen & Leus 04].

Dans la suite du texte, on s'intéresse à un autre type de flexibilité, de nature séquentielle, et sauf ambiguïté, le mot flexibilité sera exclusivement réservé pour désigner une flexibilité de ce type.

Un ordonnancement incorpore de la flexibilité séquentielle si l'ordre des tâches qu'il définit sur chaque ressource est partiel (certaines décisions de séquencement sont laissées libres). Ce type de flexibilité est intéressant car il permet, lorsqu'une ressource se libère, de disposer de plusieurs choix quant à la prochaine tâche à traiter. Par exemple si, suite à un aléa, une tâche ne peut commencer du fait d'un composant requis est indisponible, ou parce qu'une autre tâche devant la précéder est en retard, il devient possible de réaliser une autre tâche en restant cohérent avec l'ordonnancement de référence. De plus, la flexibilité séquentielle induisant une flexibilité temporelle, il est également possible d'absorber les retards liés à la réalisation de certaines tâches, sans détérioration de performance, au prix cependant d'une perte de flexibilité séquentielle.

Les deux écueils à contourner pour construire une solution flexible sont, d'une part, d'être capable de déterminer la performance au pire de l'ensemble de solutions, d'autre part, d'être capable de *caractériser* un vaste ensemble de solutions en évitant la complexité liée à une énumération totale de ces solutions.

Dans ce cadre, une approche d'ordonnancement robuste, développée au cours de la thèse de doctorat de Hoang Trung La [La 05], a été proposée. Cette approche est de type proactif-réactif [GOThA 02] et s'inscrit dans la droite lignée de l'approche flexible d'ordonnancement ORABAID, déjà développée au LAAS-CNRS dans les années 80, et fondée sur le concept de *groupe de tâches permutables* [Thomas 80, Le Gall 89, Billaut 93, Artigues 97]. En effet, dans un cas comme dans l'autre, l'ajout volontaire de flexibilité séquentielle au sein d'un ordonnancement est considéré comme le ressort permettant de réagir aux perturbations. Néanmoins, les deux approches demeurent radicalement différentes du point de vue des techniques utilisées. En effet, nous utilisons les conditions de dominance décrites dans la section 2.3.1 afin de caractériser, sur chaque machine, un ensemble de solutions dont la performance au pire est connue, cet ensemble étant robuste *vis-à-vis* d'un ensemble de perturbations préalablement défini.

Ces travaux sont décrits dans les 4 sections suivantes. Les détails des résultats peuvent être trouvés dans l'article "A robust approach for the single machine scheduling problem" [Briand *et al.* 07], inséré dans le chapitre 5, ainsi que dans [La 05] et [La *et al.* 05].

2.5.1 Mesure de la qualité d'un ensemble dominant

Dans un ordonnancement flexible, il est fondamental d'être capable de déterminer, étant donné un critère de qualité, sa pire valeur possible. En effet, ajouter de la flexibilité séquentielle au sein d'une solution induit généralement une détérioration de la qualité. Le décideur doit donc connaître le prix à payer afin de rendre une solution robuste. C'est pourquoi nous nous sommes intéressés, dans le cadre du problème à une machine avec intervalles d'exécution, à caractériser le pire retard algébrique d'une ensemble de séquences.

Le théorème des pyramides (cf. page 18) associe à chaque structure d'intervalles, définie par un problème V à résoudre, un ensemble des séquences dominantes S_{dom}. Nous avons montré dans [Briand *et al.* 07] qu'il était possible de calculer en temps polynomial, pour chaque travail j, les valeurs L_j^{max} et L_j^{min} correspondant respectivement au pire et au meilleur retard de j parmi toutes les séquences de S_{dom}. Ce sont ces valeurs qui seront prises comme indicateurs de la qualité de l'ensemble dominant de séquences.

Dans la suite du texte, on note respectivement $u(j)$ et $v(j)$ les indices de la première et de la dernière pyramide auxquelles peut être affecté le travail j.

Calculer le retard au mieux d'un travail j demande de déterminer la séquence $Seq_j^{\min}$ minimisant la longueur du chemin critique associé à ce travail. Pour cela, seuls les travaux nécessairement séquencés avant j sont à considérer (en cohérence avec la structure de pyramides). Le travail j est affecté à la pyramide d'indice $u(j)$ (au plus tôt possible) et seul l'ensemble $Pred_j^{min}$ des travaux k tels que $v(k) < u(j)$ est pris en compte. En effet, si la condition précédente est vérifiée alors k précède nécessairement j dans toutes les séquences de l'ensemble dominant. Dans la pyramide d'indice $u(j)$, nous pouvons mettre j en première position. Déterminer L_j^{min} revient alors à minimiser la durée totale C_{max} du problème d'ordonnancement constitué des travaux de l'ensemble $Pred_j^{min}$. Le retard des travaux de $Pred_j^{min}$ n'étant pas significatif pour le calcul de L_j^{min}, nous fixons arbitrairement leur date de fin à la date d_j. La structure d'intervalles de $Pred_j^{min}$ ainsi obtenue définissant une structure en escalier, une séquence Seq_j^{min} (cf. figure 2.17) optimale est obtenue par l'application de la règle de Jackson séquençant les travaux dans l'ordre croissant de leurs dates de début (remarquons que Seq_j^{min} respecte le théorème des pyramides).

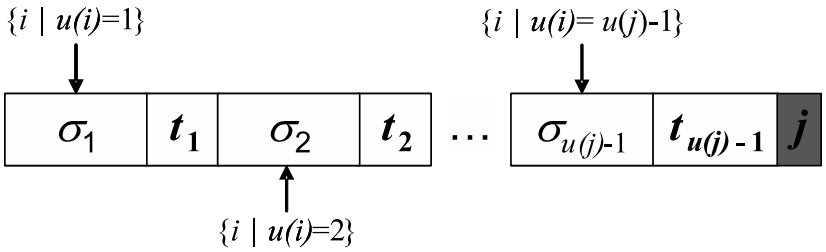


FIGURE 2.17 – Construction de la séquence au mieux $Seq_j^{\min}$

Soient $s_k^{Seq_j^{min}}$ les dates de début des travaux $k \in Pred_j^{min}$ dans Seq_j^{min}. Le makespan $C_{max}^{Seq_j^{min}}$ de cette séquence est :

$$C_{max}^{Seq_j^{min}} = Max(s_k^{Seq_j^{min}} + p_k), \forall k \in Pred_j^{min}$$

On en déduit :

$$L_j^{min} = Max(C_{max}^{Seq_j^{min}}, r_j) + p_j - d_j$$

Considérons à présent le calcul du retard au pire d'un travail j, noté L_j^{max}. De façon symétrique au cas précédent, j est affecté à la pyramide d'indice $v(j)$, c'est-à-dire le plus tard possible. D'autre part, les travaux k devant nécessairement être séquencés après k (*i.e.* les travaux k tels que $u(k) > v(j)$) ne sont pas considérés. Nous notons $Pred_j^{max}$ l'ensemble des travaux restants. Trouver L_j^{max} consiste alors à maximiser le makespan C_{max} des travaux de $Pred_j^{max}$ tout en respectant le théorème des pyramides.

Pour cela, nous affectons les travaux k tels que $v(k) < v(j)$ de façon à ce qu'ils soient séquencés le plus tard possible, c'est à dire à la pyramide d'indice $v(k)$. Les affectations des travaux aux pyramides étant établies, le problème de maximiser le C_{max} des travaux situés dans les $v(j) - 1$ pyramides d'indice inférieur à $v(j)$ se décompose alors en $v(j) - 1$ problèmes de maximisation indépendants. Ils peuvent chacun être résolus optimalement, avec respect du théorème des pyramides, en construisant une sous-séquence dans laquelle les travaux sont séquencés par ordre de d_i croissant, puis en juxtaposant les séquences obtenues. Considérons à présent la pyramide d'indice $v(j)$. Pour maximiser le retard de j, les travaux k tels que $u(k) \leq v(j) \leq v(k)$ sont affectés à cette pyramide. Le problème est alors de déterminer quelle séquence, parmi celles de l'ensemble dominant plaçant j en dernier dans la pyramide $v(j)$, possède le C_{max} le plus grand. Il est facile de montrer que cette séquence est celle plaçant le plus de travaux entre le sommet de la pyramide $v(j)$ et le travail j. Pour l'obtenir, il suffit de séquencer tous les travaux k tels que $d_k > d_j$ avant le sommet et tous les travaux restants après le sommet. En juxtaposant toutes ces sous-séquences on obtient la séquence $Seq_j^{\max}$, la pire pour le travail j (cf. figure 2.18).

On en déduit :

$$L_j^{max} = Max(C_{max}^{Seq_j^{max}}, r_j) + p_j - d_j$$

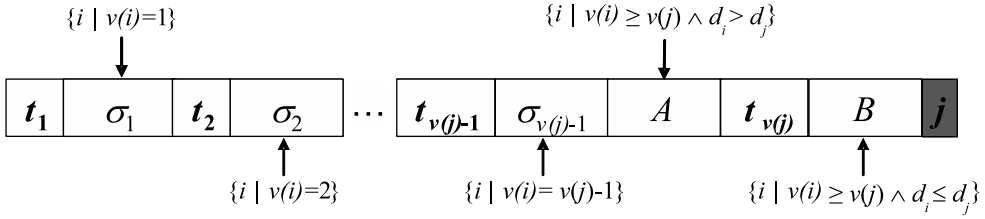


FIGURE 2.18 – Structure de la séquence au pire associée au travail j

Travail	r_j	d_j	p_j
1	10	44	5
2	13	25	6
3	11	27	7
4	20	30	4
5	30	43	3
6	0	34	6
7	30	51	2

TABLE 2.4 – Une instance de problème à une machine

Pour illustrer ces résultats, considérons l'exemple proposé par J. Carlier dans [Carlier 82] (voir table 2.4). La structure d'intervalles associée à cet exemple est donnée sur la figure 2.19, ainsi que la décomposition correspondante en pyramides. Les durées opératoires sont représentées par des rectangles. Les sommets sont grisés.

Pour cet exemple, on distingue 4 sommets : $s_1 = 2$, $s_2 = 4$, $s_3 = 5$ et $s_4 = 7$, caractérisant les 4 pyramides : $P_1 = \{1, 3, 6\}$, $P_2 = \{1, 6\}$, $P_3 = \{1\}$ et $P_4 = \emptyset$.

Pour le calcul du retard au mieux, intéressons nous au travail 5. Ce travail est un sommet et $u(5) = 3$. L'ensemble $Pred_5^{min}$ est donc constitué des travaux k tels que $v(k) < 3$, c'est-à-dire : $Pred_5^{min} = \{2, 3, 6, 4\}$. En appliquant la règle de Jackson, la séquence Seq_5^{min} obtenue est : $Seq_5^{min} = 6 \prec 3 \prec 2 \prec 4$ de makespan $C_{max}^{Seq_5^{min}} = 28$. On en déduit :

$$L_5^{min} = Max(C_{max}^{Seq_5^{min}}, r_5) + p_5 - d_5 = 30 + 3 - 43 = -10.$$

Intéressons nous à présent au calcul du pire retard pour le travail 6. Ce travail appartient aux deux pyramides de sommets 2 et 4, donc $v(6) = 2$. L'ensemble $Pred_6^{max}$ contient les travaux k tels que $u(k) \leq v(6) = 2$, c'est à dire $\{1, 2, 3, 4\}$. En appliquant la méthode décrite plus haut,on obtient la séquence $Seq_j^{\max} = 2 \prec 3 \prec 1 \prec 4 \prec 6$ et on déduit $L_6^{max} = 7$.

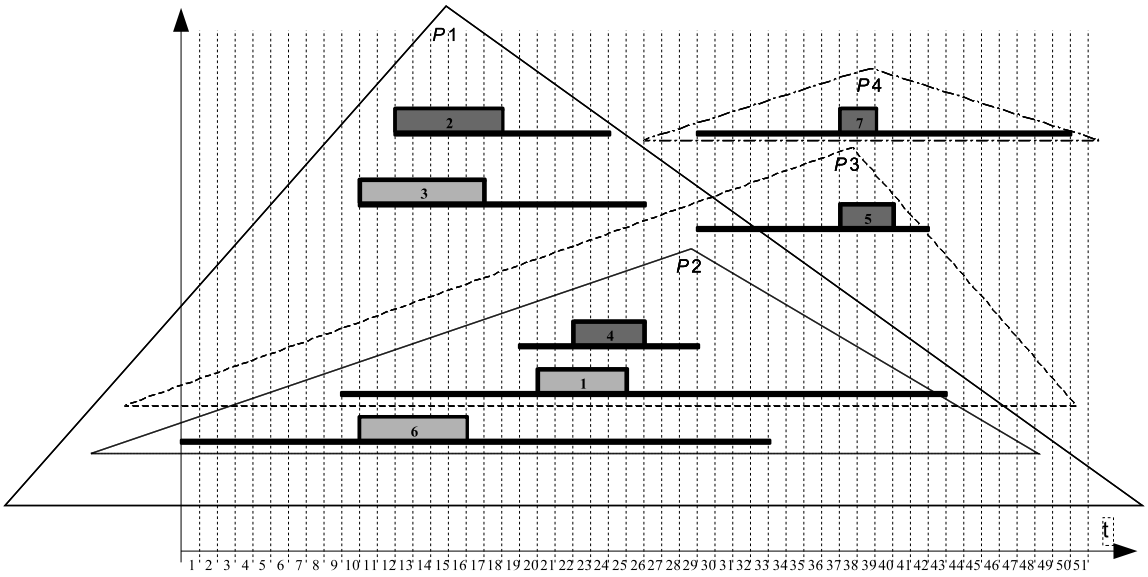


FIGURE 2.19 – Structure d'intervalles associé à l'exemple de la table 2.4

2.5.2 Le diagramme de retards

Étant donné un problème V et son ensemble des séquences dominantes S_{dom}, il est intéressant de disposer d'une représentation visuelle permettant de mettre en évidence la flexibilité et les performances de S_{dom}. Dans ce but, nous introduisons ici le diagramme de retards associant à chaque travail j de V un intervalle $[L_j^{min}, L_j^{max}]$ calculé comme décrit dans la section précédente. Étant donnée la structure d'intervalles du problème du tableau 2.4, la figure 2.20 présente le diagramme de retards obtenu.

Les séquences éventuellement partielles ayant permis le calcul de chaque valeur L_j^{min} et L_j^{max} sont indiquées au dessus de chaque borne. Le diagramme indique que, parmi toutes les séquences dominantes caractérisées, le retard algébrique L_{max} optimal est tel que :

$$Max(L_j^{min}) \leq L_{max} \leq Max(L_j^{max}), \forall j \in V$$

Par exemple, sur ce diagramme nous avons $-2 \leq L_{max} \leq 11$. Ces valeurs sont à comparer au retard optimal de -1, établi dans [Carlier 82]. Remarquons de plus que, quelle que soit la séquence de S_{dom} considérée, les travaux 1 et 7 ne seront jamais en retard.

Ce diagramme de retards permet à un décideur de visualiser les performances associées à l'ensemble des séquences dominantes S_{dom} d'un problème, du point de vue du retard

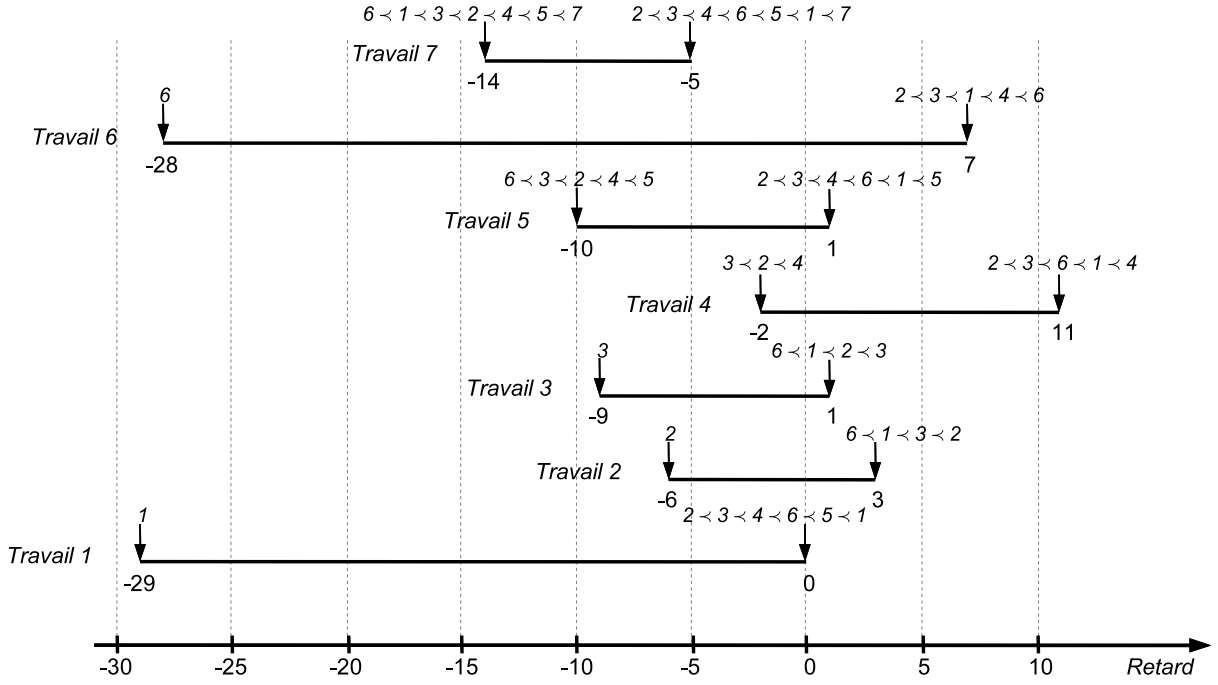


FIGURE 2.20 – Diagramme de retards obtenu pour l'exemple du tableau 2.4

algébrique, ainsi que celui de la flexibilité, d'autant plus grande que l'amplitude des intervalles de retard est importante. Dans l'hypothèse où les dates de fin des travaux correspondent à des délais de livraison (cas de la production à la commande), le décideur pourrait juger si un retard au pire est acceptable ou non. Dans la négative, une interaction serait possible, via cette représentation graphique, dans laquelle le décideur demanderait une réduction des retards qui ne lui conviendraient pas.

Une procédure d'ordonnancement aurait alors pour objectif d'éliminer, au détriment de la flexibilité, les séquences de l'ensemble dominant qui produisent les retards non acceptables. Cette idée est développée dans la partie qui suit.

2.5.3 Une méthode d'ordonnancement robuste [Briand *et al.* 07]

Nous nous intéressons ici à modifier la structure d'intervalles initiale associée à un problème, en réduisant certaines fenêtres d'exécution de travail, de sorte que l'ensemble dominant caractérisé par le théorème des pyramides ne comprenne que des solutions optimales. On suppose que la valeur optimale du $C_{\max}$ est connue (utilisation de l'algorithme de Carlier [Carlier 82] ou de l'approche par PLNE décrite dans la section 2.3.3).

On utilise pour cela un schéma classique de parcours arborescent, proche de celui utilisé

par l'algorithme de Carlier. La racine de l'arborescence est la structure d'intervalles associée au problème initial V. Chaque nœud de l'arborescence est séparé en 2 nœuds fils, chaque fils correspondant à un nouveau problème $V' = \{j' | r_{j'} \geq r_j, p_{j'} = p_j, d_{j'} \leq d_j\}$. Un nœud est étiqueté par une borne inférieure et une borne supérieure du critère $L_{\max}$, correspondant respectivement aux valeurs de $\max_{j \in T} L_j^{\min}$ et $\max_{j \in T} L_j^{\max}$, calculées comme décrit plus haut dans le texte.

Le mécanisme de séparation que nous proposons permet de garantir que l'union des ensembles dominants associés aux 2 fils est exactement égale à l'ensemble dominant du père. Il est basé sur une analyse du chemin critique C associé à la séquence produisant la borne supérieure du retard algébrique $L^{\max}$. À partir de ce chemin, sont déterminés un travail *pivot*, correspondant à un sommet de ce chemin, et un travail *libre*, non-sommet, que nous séquencerons soit à droite, soit à gauche du pivot.

Notons respectivement i et α le travail libre à pivoter et le travail pivot, déterminés comme décrit précédemment. Deux branchements sont alors envisagés : soit α précède i, soit i précède α. Compte tenu de l'ordre partiel fixé par le théorème des pyramides, ces deux précédences peuvent être imposées de la façon suivante :

- $\alpha \prec i$ peut être imposé en actualisant r_i de sorte que $r_i \leftarrow r_\alpha$; en effet, α étant un sommet, le théorème des pyramides garantit que si $r_j \geq r_\alpha$ alors α précède j dans toute séquence dominante qu'il caractérise ;
- similairement, $i \prec \alpha$ peut être imposé en actualisant d_i de sorte que $d_i \leftarrow d_\alpha$; en effet, α étant un sommet, le théorème des pyramides garantit que si $d_j \leq d_\alpha$ alors j précède α dans toute séquence dominante qu'il caractérise.

Cette procédure arborescente a été implémentée et testée sur des problèmes comptant 10, 50, 100 et 500 travaux. Pour chacune de ces familles, 320 instances ont été générées. Une vue condensée des résultats obtenus est donnée sur la table 2.5. On voit que les temps moyens de résolution des problèmes à 10-50-100 travaux sont faibles et stables. Le temps de calcul augmente pour les problèmes à 500 travaux mais reste cependant tout à fait acceptable. Le nombre moyen de solutions optimales caractérisées est très impressionnant, en particulier pour les grosses instances. On observe cependant une disparité entre les instances d'une même famille. Celle-ci s'explique en partie par le fait que, d'un problème à l'autre, le nombre de sommets que comptent les structures d'intervalles peut varier de façon importante. Or, plus une structure d'intervalles compte de sommets, moins le nombre de séquences dominantes est important.

Remarquons que les structures d'intervalles optimales caractérisent un ensemble flexible de solutions, ces dernières étant optimales vis-à-vis du problème déterministe initial. La méthode est qualifiée de robuste dans le sens où elle met en évidence une flexibilité séquentielle, potentiellement exploitable au sein d'une procédure d'ordonnancement en ligne.

		Tcpu (sec.)	Card(S_{dom})
10 jobs	Avg.	0.00018	7.27
	Max.	0.01	108
	Min.	0.001	1
50 jobs	Avg.	0.068	3.67E+22
	Max.	0.11	1.01E+25
	Min.	0.02	1
100 jobs	Avg.	0.43	1.87E+63
	Max.	0.631	5.51E+65
	Min.	0.22	36
500 jobs	Avg.	57.17	2.06E+303
	Max.	93.731	> 1.00 E.308
	Min.	25.138	3.1568E+12

TABLE 2.5 – Synthèse des résultats

De plus, il est possible de caractériser un ensemble de scénarios d'exécution vis-à-vis duquel l'ensemble dominant est *a priori* robuste. En effet, on observe qu'un ensemble dominant reste inchangé quelles que soient les valeurs des dates de début au plus tôt et au plus tard, dans la mesure où l'ordre total entre ces dates reste conservé. D'autre part, cet ensemble est complètement indépendant des durées opératoires des travaux. Ces dernières remarques restent valides quant à la construction des séquences au mieux et au pire $Seq_j^{\min}$ et $Seq_j^{\max}$. On en déduit que, si un problème possède plusieurs scénarios d'exécution possibles, tous caractérisés par un modèle à intervalles de la forme $r_i \in [\underline{r}_i, \overline{r}_i]$, $p_i \in [\underline{p}_i, \overline{p}_i]$ et $d_i \in [\underline{d}_i, \overline{d}_i]$ alors, si tous les intervalles $[\underline{r}_i, \overline{r}_i]$ et $[\underline{d}_i, \overline{d}_i]$ **sont disjoints**, il est possible de déterminer un ensemble de séquences tel que :

- les retards au mieux et au pire de l'ensemble sont déterminés (quelles que soient les valeurs réalisées des r_i, p_i et d_i dans le respect du modèle à intervalles) ;
- quelles que soit les valeurs réalisées, il existe un optimum dans l'ensemble dominant de séquences !

Supposer que les intervalles $[\underline{r}_i, \overline{r}_i]$ et $[\underline{d}_i, \overline{d}_i]$ sont disjoints ne semble pas irréaliste. En effet, dans le cas d'une production à la commande, cela impose seulement que ni les composants nécessaires à la production, ni les produits finis, ne soient livrés dans les mêmes périodes.

2.5.4 Une aide à la décision pour un compromis flexibilité / performance

On quitte à présent le monde de l'optimisation pour considérer celui de l'aide à la décision. Le but est maintenant de faciliter les choix d'un décideur qui rechercherait, dans l'ensemble dominant initial S_{dom} associé au problème, une solution correspondant à un compromis flexibilité/performance acceptable. Pour cela, le diagramme de retards est choisi comme objet d'interaction. En effet, il permet de juger de l'acceptabilité du compromis flexibilité / performance puisqu'il indique, pour chaque travail, les retards $L_j^{\min}$ et $L_j^{\max}$, et qu'un indicateur de flexibilité, précisant le nombre de séquences contenues dans l'ensemble dominant, peut être calculé.

Ainsi que décrit dans [La *et al.* 05], l'aide à la décision consiste, dans le cas où le compromis ne satisfait pas le décideur, à lui donner la possibilité de sélectionner une borne d'un travail, $L_j^{\min}$ ou $L_j^{\max}$, pour indiquer qu'il désire la réduire ou l'augmenter, en précisant la nouvelle valeur souhaitée (appelée *consigne* dans la suite du texte). Le principe de l'aide à la décision consiste à coupler une telle action à une modification de la structure d'intervalles, comme indiqué dans les paragraphes suivants. Rappelons que $u(j)$ (resp. $v(j)$) indique l'indice de la première pyramide (resp. de la dernière pyramide) à laquelle le travail j appartient.

La réduction ou l'augmentation du retard $L_j^{\max}$ est respectivement réalisée en décrémentant ou en incrémentant progressivement la valeur de $v(j)$, en vérifiant toujours l'inégalité $v_0(j) \geq v(j) \geq u(j)$, où $v_0(j)$ est l'indice de la dernière pyramide à laquelle le job j appartient initialement. La décrémentation de $v(j)$ conduit à la mise à jour de la valeur de d_j de sorte que $d_j \leftarrow \max(d_{s_{v(j)-1}}, d_{s_{u(j)}})$. L'incrémentation de $v(j)$ est elle aussi réalisée en mettant à jour la valeur de d_j de sorte que $d_j \leftarrow \min(d_{s_{v(j)+1}}, d_j^0)$, où d_j^0 est la valeur initiale de d_j.

Similairement, la réduction ou l'augmentation du retard $L_j^{\min}$ est respectivement réalisée en décrémentant ou en incrémentant progressivement la valeur de $u(j)$, en vérifiant l'inégalité $u_0(j) \leq u(j) \leq v(j)$. L'incrémentation de $u(j)$ conduit à la mise à jour de la valeur de r_j de sorte que $r_j \leftarrow \min(r_{s_{u(j)+1}}, r_{s_{v(j)}})$. La décrémentation de $u(j)$ est elle aussi réalisée en mettant à jour la valeur de r_j de sorte que $r_j \leftarrow \max(r_{s_{u(j)-1}}, r_j^0)$, où r_j^0 est la valeur initiale de r_j.

L'incrémentation ou la décrémentation des valeurs de $u(j)$ ou $v(j)$, est automatiquement réitérée jusqu'à ce que la valeur de $L_j^{\min}$ ou $L_j^{\max}$ soit la plus proche possible de la consigne donnée par le décideur. Remarquons que, quelle que soit l'action envisagée, elle se traduit toujours par une augmentation ou une diminution des bornes r_j ou d_j associées au travail j considéré. Une nouvelle structure d'intervalles est donc obtenue telle que l'ensemble dominant de séquences correspondant est inclus dans l'ensemble dominant

initial (du fait de la façon d'actualiser les valeurs de r_j et d_j). Notons également que l'augmentation ou la réduction d'un retard au pire ou au mieux d'un travail j peut avoir des conséquences sur les retards des autres travaux du fait de la méthode de calcul des $L_j^{\min}$ et $L_j^{\max}$. De façon générale, le resserrement de l'intervalle de retard d'un travail j réduit la flexibilité et peut entraîner le resserrement d'intervalles de retard d'autres travaux. Inversement, l'élargissement d'un intervalle de retard augmente la flexibilité et peut entraîner l'élargissement d'intervalles de retard d'autres travaux.

Afin de valider les principes de l'aide à la décision décrits plus haut, une interface logicielle, baptisée ADOR (Aide à la Décision pour l'Ordonnancement Robuste), a été élaborée au cours du stage de DEA de Jean-Luc Santamaria. Le langage utilisé fut Ada et, quant à l'aspect graphique, la bibliothèque gratuite GtkAda a naturellement été adoptée. Nous illustrons rapidement ci-dessous les fonctionnalités de cette interface.

La figure 2.21 représente l'interface graphique associée à la saisie d'un problème. Après validation de la saisie, le diagramme de retards initial est automatiquement calculé et affiché ainsi qu'illustré sur la figure 2.22. Le nombre de séquences contenues dans l'ensemble dominant initial est également indiqué au dessus du diagramme des retards. Pour représenter les retards, une règle graduée, munie de deux curseurs mobiles à ses extrémités, est associée à chaque travail. Les curseurs gauche et droit d'un travail j matérialisent respectivement les valeurs de $L_j^{\min}$ et $L_j^{\max}$. Une région ombrée, commune à l'ensemble des règles, indique l'intervalle dans lequel est situé le retard optimal ($L^* \in [\max(L_j^{\min}), \max(L_j^{\max})]$), ici l'intervalle $[-1, 6]$. Le sous-menu *search optimal L** du menu *Decisions* détermine, si l'utilisateur le souhaite, la valeur exacte du retard optimal (utilisation de la procédure de Carlier [Carlier 82]). Dans ce cas, la région est réduite à une seule valeur, 4 dans notre cas.

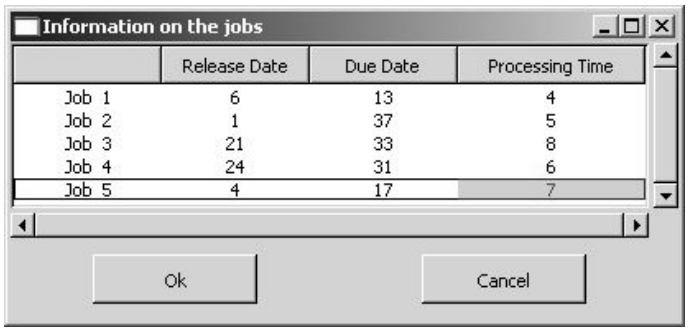


FIGURE 2.21 – Saisie des données d'un problème

Afin de permettre l'interaction, les curseurs de chaque règle peuvent être déplacés à l'aide de la souris. Cette interaction, permet au décideur d'indiquer qu'il souhaite modifier un retard au mieux ou au pire associé à un travail. Supposons par exemple que le décideur désire modifier le retard au mieux du travail 2 de sorte qu'il soit le plus proche possible de la valeur 12. Dans ce cas, il doit sélectionner le curseur gauche de la règle associée à

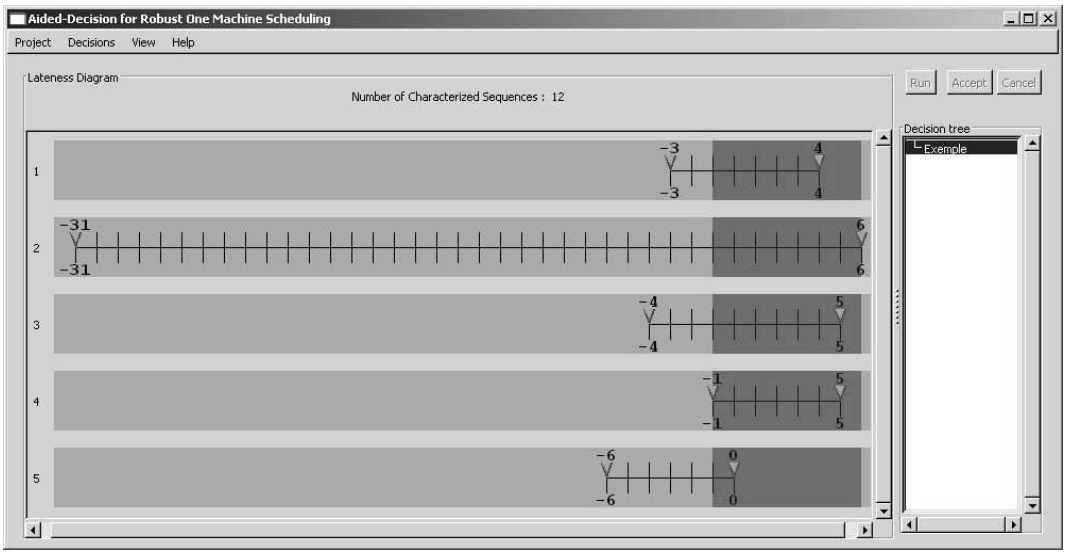


FIGURE 2.22 – Affichage du diagramme des retards initial

ce travail, puis le faire glisser à la valeur 12, et enfin cliquer sur le bouton *run*. La fenêtre est alors actualisée (cf. figure 2.23) et affiche pour chaque travail les anciennes et nouvelles valeurs des retards (un curseur grisé indique la valeur initiale d'un retard lorsqu'elle a été modifiée). On constate que, conformément à la procédure de mise à jour de $u(2)$ (voir section précédente), la valeur la plus proche de la consigne 12 (représentée par un marqueur à cette abscisse) est 3 (nouvelle position du curseur) et il reste 4 séquences dans l'ensemble dominant sur les 12 initiales. De plus, les retards au pire des travaux 1 et 4 ont été réduits. L'utilisateur peut alors, s'il le souhaite, mémoriser la structure d'intervalles correspondant à ce nouveau diagramme de retards, en cliquant sur le bouton *Accept*. Ainsi qu'illustré sur la figure 2.24, l'appui sur le bouton *Accept* provoque d'une part la création d'un nouveau nœud dans l'arbre de décision (fenêtre *Decision tree*), et d'autre part la validation des nouvelles positions des curseurs. En se plaçant sur un nœud quelconque de l'arbre de décision et en modifiant un retard, l'utilisateur peut répéter l'opération, et affiner ainsi sa stratégie de recherche, en développant soit en séquence, soit en concurrence un ensemble de décisions.

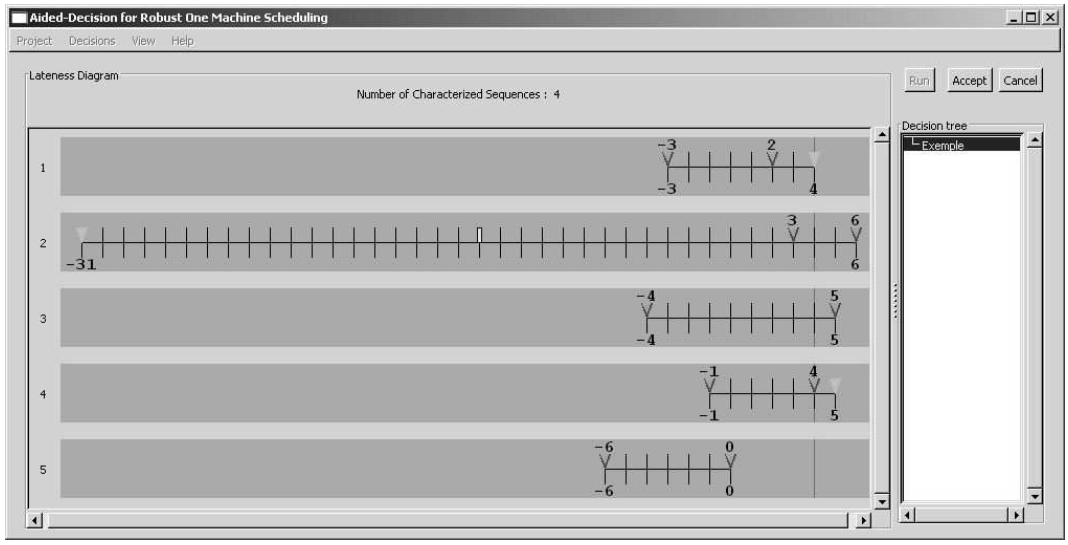


FIGURE 2.23 – Exemple d'interaction

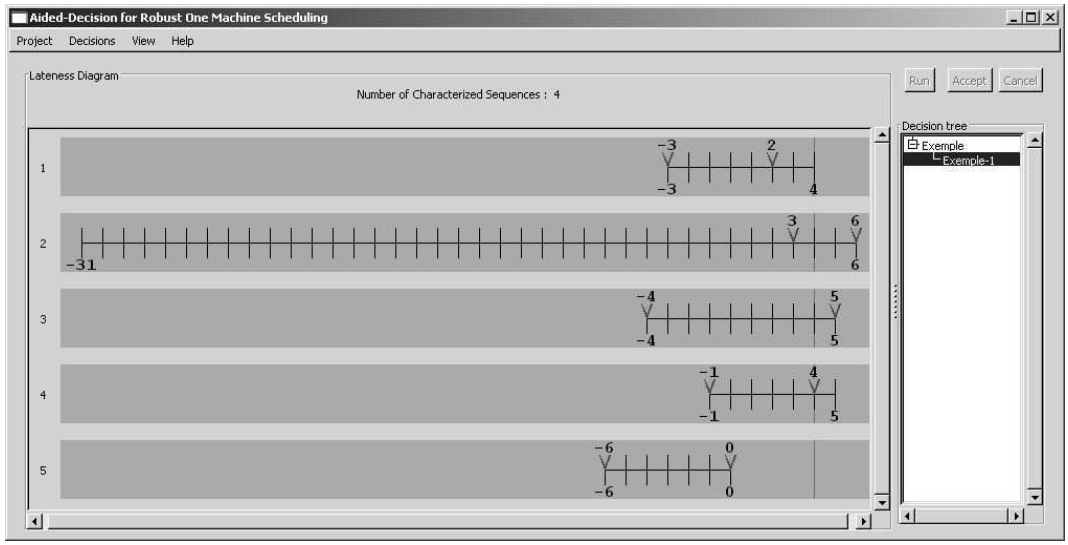


FIGURE 2.24 – Validation d'une décision

2.6 Ordonnancement de projet

Cette partie s'intéresse à un problème plus appliqué que ceux abordés jusqu'à présent : l'ordonnancement de projet (RCPSP). On s'intéresse en particulier à la détermination de nouveaux schémas de génération pour la détermination de solution approchée. On considérera, tour à tour, le cas où il n'existe pas d'intervalles temporels maximaux entre les activités du projet, et celui où il y en a. Les travaux décrits ici sont le fruit, d'une part, des stages de DEA et de Master de Damien Poinsot, Jérome Bezanger et Xiaoshu Li que j'ai encadrés et, d'autre part, de la collaboration avec Christian Artigues avec lequel j'ai co-encadré le stage de Master Recherche de Abdelhamid Younes.

2.6.1 Introduction

En ordonnancement de projet, il est courant d'utiliser des règles de priorité pour construire rapidement, par une heuristique glouton, un ordonnancement satisfaisant les contraintes de temps et de ressources. Pour cela, indépendamment de la règle de priorité choisie, deux schémas gloutons de génération sont généralement distingués : le schéma de génération *série* et le schéma de génération *parallèle*.

Le schéma de génération série nécessite n itérations (n étant le nombre d'activités). À chaque itération, l'activité de plus grande priorité est sélectionnée parmi un ensemble d'activités dites *éligibles* (une activité est dite éligible si toutes les activités la précédant ont déjà été ordonnancées). L'activité sélectionnée est ensuite ordonnancée au plus tôt, en interdisant de remettre en cause les dates de début des activités ordonnancées durant les itérations précédentes.

Le schéma de génération parallèle est relativement similaire au précédent excepté qu'à chaque itération, plusieurs activités peuvent être ordonnancées. À chaque itération k est associée une date $t_k > t_{k-1}$, égale à la plus petite date de fin des activités déjà ordonnancées. Les activités éligibles possédant une date de début au plus tôt égale à t_k sont ordonnancées à cette date, par ordre de priorité croissante. L'ordonnancement d'une activité à la date t_k empêche parfois l'exécution à la même date d'activités moins prioritaires. Celles-ci sont alors considérées lors des itérations suivantes.

Le schéma de génération série produit des ordonnancements actifs (aucune activité ne peut être ordonnancée plus tôt sans retarder l'exécution d'une autre). Les ordonnancements produits par le schéma parallèle appartiennent à la classe des ordonnancements *sans délais*. Cette classe est une sous-classe des ordonnancements actifs, mais ne possède pas, contrairement à la précédente, la propriété de dominance vis-à-vis de critères réguliers.

On remarque que l'espace de solutions, potentiellement accessible grâce aux schémas série et parallèle, est relativement limité. En effet, lors de l'ajout d'une activité, les deux schémas interdisent de déplacer des activités déjà ordonnancées (c'est à dire de remettre en cause des décisions déjà prises). La nouvelle activité est alors bien souvent placée à la suite des activités déjà ordonnancées. En conséquence, dans de nombreux problèmes, il suffit que l'ordre de sélection des activités (imposé par la règle de priorité et par la notion d'éligibilité) soit malheureux pour que, lors d'une itération, une décision de séquencement mauvaise soit prise et qu'il n'y ait plus alors aucune possibilité de trouver une solution optimale.

2.6.2 Le schéma de génération *Any-Order*

Pour pallier la limite précédente, un schéma de génération alternatif a été proposé [Briand & Bezanger 06, Briand 09]. Ce schéma est baptisé *Any-Order* dans la mesure où il permet de considérer les activités dans un ordre indifférent, non nécessairement compatible avec le graphe de précédence. On s'affranchit ainsi de la notion d'éligibilité (toutes les activités étant éligibles à tout instant). De plus, lors de l'insertion d'une activité, le schéma autorise le décalage des activités déjà ordonnancées, un espace de solutions plus vaste pouvant ainsi être exploré.

Le composant de base du schéma Any-Order est la méthode d'insertion d'activité proposée par C. Artigues et F. Roubellat dans [Artigues & Roubellat 00, Artigues *et al.* 03]. Étant donné un graphe de flots de ressources correspondant à un ordonnancement partiel, cette méthode énumère un ensemble de coupes maximales. Chaque coupe maximale définit un ensemble de coupes d'insertion valide pour une activité que l'on souhaite insérer. Une coupe d'insertion est dite valide dans la mesure où le flot de ressources qu'elle véhicule est supérieur ou égal à celui nécessaire pour réaliser la nouvelle activité. La méthode permet de trouver, en temps polynomial, la meilleure coupe valide d'insertion, c'est à dire dans notre cas, celle conduisant à une augmentation minimale de la durée du projet. Cette méthode nécessite la connaissance des dates de début au plus tôt et de fin au plus tard des activités.

Un algorithme succinct du schéma de génération Any-Order est décrit ci-dessous. Les activités sont numérotées de 1 à n. Les activités fictives, 0 et n+1, correspondent respectivement au début et à la fin du projet. On suppose de plus que $G =< V, U_{\prec}, U_r >$ est le graphe associé à l'ordonnancement partiel courant. Chaque sommet $i \in V = 0 \ldots n+1$ de G correspond à une activité. Chaque arc a orienté correspond soit à une contrainte de précédence entre deux activités (i.e. $a \in U_{\prec}$), soit à un flot de ressources entre deux activités (i.e. $a \in U_r$). Au lancement de l'algorithme, G ne contient aucun flot de ressources, excepté un flot de capacité maximale entre les activités fictives O et n+1. La partie $U_{\prec}$ de G n'est pas modifiée par l'algorithme.

Le fonctionnement de l'algorithme 2 est alors très simple à comprendre. À chaque itération, une activité quelconque j de ε est sélectionnée grâce à la règle de priorité R. On recherche alors une coupe d'insertion optimale C^* pour l'insertion de j dans la partie U_r de G. L'insertion est ensuite réalisée. La partie U_r de G ayant été modifiée et la date de début de $n+1$ pouvant par ailleurs avoir augmenté, il est nécessaire de recalculer les dates de début au plus tôt et au plus tard des activités. L'algorithme s'arrête lorsque toutes les activités ont été ordonnancées. La complexité temporelle est en $O(n^3)$.

Algorithme 2 AO_SGS(G)

$\varepsilon \leftarrow \{1...n\}$;
for i $\leftarrow$ 1 to n **do**
  $j \leftarrow$ Select_Activity(ε, R) ;
  $C^* \leftarrow$ Search_Optimal_Insertion_Cut(j, G) ;
  Insert(j, C^*,G) ;
  Forward_Backward_Propagation(G) ;
  $\varepsilon \leftarrow \varepsilon - \{j\}$;
end for

Le schéma de génération Any-Order a été testé sur les instances de disponibles dans la librairie PSPLIB (`http://129.187.106.231/psplib/datasm.html`) (480 instance de problèmes à 30 activités - 480 instances de 60 activités et 600 instances de 120 activités). Pour chaque classe de problème, onze règles de priorité ont été utilisées. Pour chaque instance et chaque règle, les résultats donnés par le schéma Any-Order ont été comparés avec ceux obtenus en appliquant un schéma de génération série classique utilisant la même règle. Le tableau 2.6 synthétise les résultats obtenus.

Classes de problème	30 activités		60 activités		120 activités	
	meilleure	équivalente	meilleure	équivalente	meilleure	équivalente
#	169	293	159	299	339	133
Pourcentage	35.21%	61.04%	33.13%	62.29%	56.5%	22.16%
		96.25%		95.42%		78.66%

TABLE 2.6 – Synthèse de la comparaison

Par exemple, pour les 480 instances de problèmes à 60 activités, toutes règles confondues, il y a eu 169 instances (soit 35,21%) pour lesquelles le schéma de génération Any-Order a permis de trouver un makespan meilleur que celui trouvé avec le schéma de génération série et 293 instances pour lesquelles les valeurs des makespan sont identiques pour les deux schémas (soit 61,04%). Le schéma de génération Any-Order es donc meilleur ou équivalent dans 96,25% des cas.

L'ensemble de ces résultats illustre l'intérêt du schéma de génération Any-Order qui produit, dans la majorité des cas, des résultats meilleurs ou équivalents à ceux obtenus par un schéma de génération série. Une autre caractéristique importante, qui n'apparaît pas sur le tableau 2.6, est que les règles de priorité basées sur l'analyse de la consommation énergétique de chaque tâche (qui donnent de mauvais résultats avec un schéma classique) deviennent très concurrentielles avec le schéma Any-Order. En effet, sur les instances à 120 activités, on constate que ce sont souvent ces règles énergétiques qui permettent de trouver des solutions meilleures que celles trouvées avec le schéma de génération série.

2.6.3 Prise en compte d'intervalles temporels maximum

L'existence d'un intervalle temporel maximum (*maximum time lags* en anglais) entre 2 activités indique que ces dernières ne doivent pas être éloignées l'une de l'autre de plus d'une distance δ_{ij}. On dit que les contraintes de précédence sont *généralisées* dans la mesure où elles peuvent toutes être génériquement exprimées par l'expression $s_i+\delta_{ij} \leq s_j$, où δ_{ij} est soit positif (cas d'un intervalle temporel minimal), soit négatif (cas d'un intervalle temporel maximum).

En pratique, ce type de contrainte est très fréquent bien que rarement pris en compte par les logiciels du marché. On peut ainsi exprimer de façon générique de multiples contraintes telles que :

- le début simultané où la fin simultanée de 2 activités (rendez-vous) ;
- la fin imposée d'une activité, x unités de temps au plus tard après la fin ou le début d'une autre ;
- la prise en compte d'une activité de maintenance, planifiée à un horaire précis.

Graphiquement, les contraintes de précédence généralisées sont représentées ainsi que l'indique la figure 2.25. Les paramètres $\delta_{ij}^{\min}$ et $\delta_{ij}^{\max}$ sont deux valeurs positives correspondant à des intervalles temporels minimum et maximum entre les dates de début de i et j.

Cette classe de problèmes est notée $PS|temp|C_{\max}$ dans la littérature, et généralement désignée sous le terme RCPSP/max. Trouver une solution optimale à ce problème est bien sûr NP-difficile. Mais ce qui rend ce problème particulièrement épineux est que la recherche d'une solution admissible est elle-même NP-difficile !

Les schémas de génération série et parallèle ont été adaptés pour traiter ce problème [Neumann *et al.* 02]. Une phase de retour-arrière, utilisée lorsqu'une incohérence

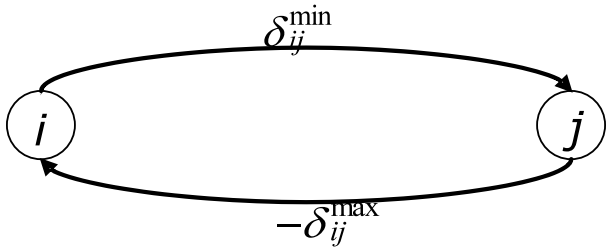


FIGURE 2.25 – Représentation graphique d'intervalles temporels minimum et maximum entre 2 activités

temporelle est détectée, est particulièrement introduite. Bien que relativement efficaces, ces schémas ne garantissent pas d'aboutir à une solution admissible. En effet, le retour-arrière peut lui même échouer et, dans ce cas, aucune solution n'est obtenue.

Dans ce qui suit, on s'intéresse à l'adaptation du schéma de génération Any-Order pour la prise en compte d'intervalles temporels maximaux. Une première question que nous nous sommes posés est de savoir si la procédure d'insertion de C. Artigues pouvait oui ou non être adaptée à ce cadre.

Durant le stage de master recherche de Xiaoshu Li, nous avons tout d'abord montré que la technique d'insertion d'activités de Christian Artigues ne permet pas la détection de création de circuit absorbant. Illustrons ceci sur un exemple très simple. Considérons 2 activités a et b, telles que $a \prec b$, de durées $p_a = 1$ et $p_b = 3$, ordonnancées sur une ressource disjonctive. Supposons en outre que l'horizon d'ordonnancement est de 13 unités de temps et que a et b sont reliées par la contrainte de précédence généralisée : $\delta_{ba}^{\max} = 7$. Le calcul des dates de début au plus tôt et au plus tard donne : $r_a = 0$, $d_a = 9$, $r_b = 1$ et $d_b = 13$. D'après la formule utilisée dans l'algorithme de C. Artigues, l'insertion d'une activité i entre a et b, ayant une durée $p_i = 9$ et une fenêtre d'exécution $[0, 13]$, ne produit aucune augmentation du makespan. En effet, $(d_b - p_b) - (r_a + p_a) \geq p_i$. Or, ainsi que l'illustre la figure 2.26, cette position d'insertion, si elle n'augmente pas la valeur du makespan, donne par contre naissance à un circuit absorbant (la contrainte d'écart maximum $\delta_{ba}^{\max} = 7$ est violée de 3 unités de temps).

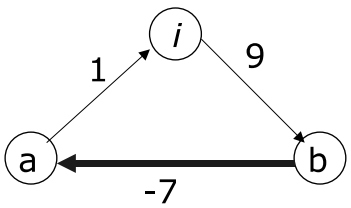


FIGURE 2.26 – Création d'un circuit absorbant

Un second travail, réalisé en collaboration avec C. Artigues, s'est intéressé à étudier le problème d'insertion, sous contraintes de délais minimaux et maximaux, en considérant explicitement l'existence de circuits. Le problème consiste à insérer une activité au sein d'un ordonnancement partiel en préservant la structure de cet ordonnancement, représentée sous forme d'un graphe de flots de ressource, dans le but de minimiser l'augmentation de makespan induite et en satisfaisant les contraintes de délais minimaux et maximaux. Nous avons proposé des conditions d'insertion basées sur une analyse de circuits élémentaires. Trois types de circuits élémentaires sont à distinguer pour pouvoir conclure quant à la validité d'une position d'insertion. Nous avons ensuite montré que la variante de décision de ce problème, où l'on recherche une position d'insertion valide n'induisant pas une augmentation de makespan supérieure à une valeur préalablement fixée, était NP-complet. Ce résultat de complexité est basé sur une réduction du problème *INDEPENDENT SET*, lui même NP-complet, vers le problème d'insertion considéré. Dans le cas où seules des contraintes de délais minimaux sont prises en compte, nous avons enfin proposé une procédure d'insertion optimale s'exécutant en temps polynomial. Cette procédure généralise les procédures d'insertion antérieurement proposées, prenant seulement en compte des contraintes de précédence entre les activités. Pour plus de détail sur ce travail, le lecteur peut se référer à l'article [Artigues & Briand 09], inclus dans le chapitre 5.

2.6.4 Relaxation des contraintes temporelles

Comme la recherche d'une position d'insertion optimale, respectant les contraintes de délais minimaux et maximaux, est NP-difficile, il est nécessaire de relâcher les contraintes du problème. Nous présentons ci-dessous une méthode où toutes les contraintes de délais sont relâchées. On remarque cependant qu'il est également possible de ne relâcher que les contraintes de délais maximaux, piste partiellement débroussaillée en 2008 lors du stage de Master recherche de Abdelhamid Younes (stage co-encadré avec C. Artigues).

Pour la relaxation de toutes les contraintes de délais minimaux et maximaux, nous considérons le graphe $G' =< V + \{n+2\}, W^-, W^+, F >$. Comme dans G, V représente les activités du projet avec $V = \{0, 1, .., n+1\}$. 0 est l'activité fictive de début, de durée nulle et de consommation de ressources maximale. Cette activité représente également l'origine des temps, autrement dit aucune activité du projet ne pourra la précéder. $n+1$ est l'activité fictive de fin de projet, de durée nulle et de consommation de ressources maximale. Le potentiel de cette activité représente également la mesure du makespan $C_{\max}$. Comme dans G, F représente les flots de ressources circulant entre les activités du graphe. F résulte des décisions d'ordonnancement. Le noeud $n+2$ correspond à une activité fictive dont le potentiel représente la valeur du retard algébrique maximum. Cette valeur indique de combien d'unités de temps au plus les contraintes temporelles sont violées / respectées (voir explication ci-dessous). W^- sont les arcs, issus de 0, entrants sur les activités $(1...n+1)$. W^+ sont les arcs, sortants des activités $(1...n+1)$, allant vers $n+2$. Ces arcs sont construits

de telle façon que, à tout arc $u =< i, j >$ de longueur δ_{ij} dans G, on associe deux arcs dans G' :

- $u^- =< 0, j > \in W^-$de longueur $\delta_{ij}^- = s_i + \delta_{ij}$
- $u^+ =< i, n+2 > \in W^+$de longueur $\delta_{ij}^+ = -(s_i - \delta_{ij})$

On remarque que ces arcs ont une longueur dépendante des dates de début des activités.

La figure 2.27 illustre le principe de la relaxation. Celle-ci intervient dans la liberté de placement d'une activité :

- si i est ordonnancée et que l'on souhaite placer j, alors la relaxation borne inférieurement la date de début au plus tôt de j.
- si j est ordonnancée et que l'on souhaite placer i, alors la relaxation borne supérieurement la date de début au plus tard de i.

Si i et j sont ordonnancées et que la contrainte $s_j - s_i \geq \delta_{ij}$ est violée, alors l'arc $< i, n+2 >$ induit un potentiel positif sur $n+2$, ce potentiel mesurant de combien d'unités de temps la contrainte est violée. Par exemple, si $s_j - s_i \geq 3$, $s_i = 2$ et $s_j = 4$, alors le potentiel induit sur $n+2$ est $s_i - (s_j - 3) = 2 - (4 - 3) = 1$. La distance entre s_i et s_j doit donc être augmentée de 1 unité pour que la contrainte soit respectée.

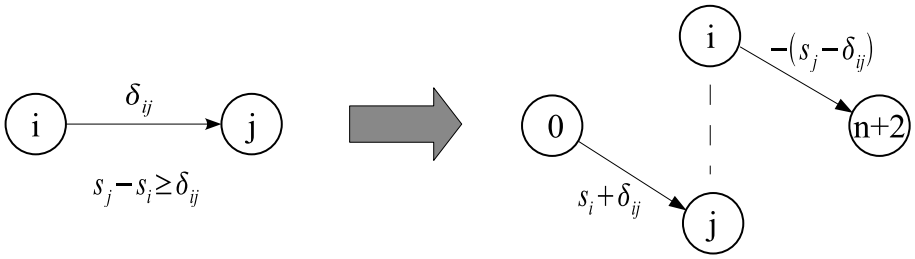


FIGURE 2.27 – Principe de relaxation

Pour illustrer ces principes, considérons l'exemple de la figure 2.28. La relaxation conduit au graphe G' représenté sur la figure 2.29. On distingue un arc en pointillé entre les activités $n+1$ et $n+2$, correspondant respectivement aux activités 7 et L sur la figure. Cet arc n'est pas obligatoire. Le rajouter permet d'interpréter le passage du $C_{\max}$ au delà d'une certaine valeur comme une violation de contraintes temporelles. Ne pas le mettre indique qu'a priori le $C_{\max}$ peut être aussi grand que l'on veut. La longueur de cet arc sera un paramètre au sein de l'heuristique de construction d'une solution.

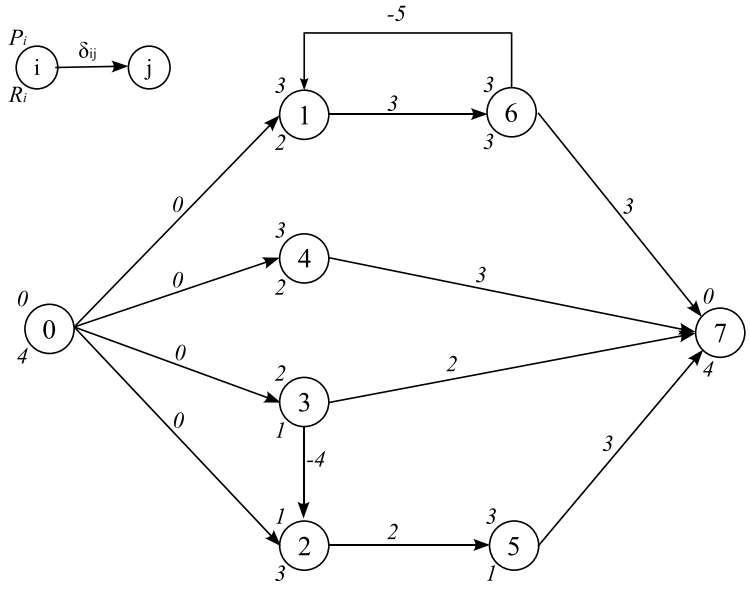


FIGURE 2.28 – Graphe RCPSP-*max*

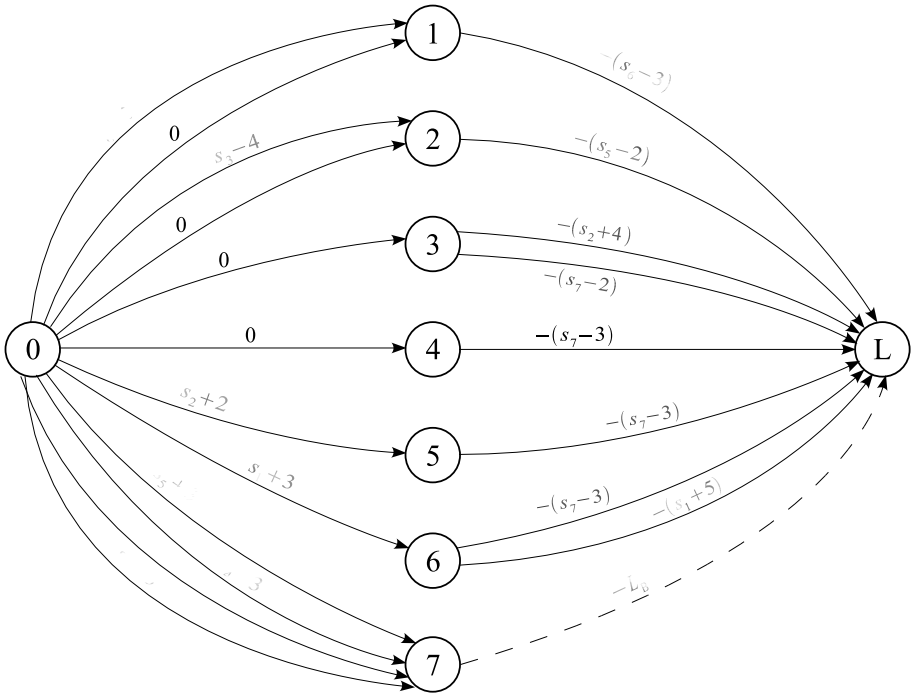


FIGURE 2.29 – Graphe relâché

2.6.5 Le schéma de génération *Any-Order/max*

Dans le cas du RCPSCP/max, l'évaluation d'une coupe d'insertion en fonction uniquement de l'augmentation du makespan produite n'est pas satisfaisante. C'est pourquoi nous proposons d'évaluer dorénavant chaque coupe par l'augmentation du retard algébrique $\delta L_{\max}$ produite (potentiel du noeud $n+2$ dans le graphe relâché). Ce retard correspond en effet à la longueur du circuit le plus long dans le graphe. Son augmentation lors d'une insertion traduit indifféremment une violation de(s) contrainte(s) temporelle(s) et/ou une augmentation du makespan.

Pour calculer l'augmentation du retard algébrique, il est nécessaire de connaître la distance δ_{ij} la plus longue entre toute paire d'activités i et j. Pour cela, il faut utiliser l'algorithme de Floyd-Warshall en $O(n^3)$.

L'insertion d'une activité i sur la coupe contenant les flots de ressources allant de la clique d'activités P à la clique S peut alors être évaluée par la formule :

$$\delta L_{\max} = \max_{x \in P}(\delta_{xi}, p_x) + \max_{y \in S}(\delta_{iy}, p_i) + \max_{y \in S, x \in P} \delta_{yx} - \delta_{0,n+2} \tag{2.1}$$

L'algorithme 3 que nous proposons est une adaptation de l'algorithme d'insertion de C. Artigues au cas du RCPSP/max . À chaque itération, la coupe courante est évaluée par la formule 2.1. La différence essentielle avec l'algorithme d'insertion classique est, qu'au lieu de passer d'une coupe à une autre en enlevant l'activité ayant la plus grande date de fin au plus tôt parmi les activités à gauche de la coupe (membre de la clique P), on enlève à présent l'activité u telle que $\delta_{vu} = \max_{y \in S, x \in P} \delta_{yx}$, c'est à dire celle imposant le circuit le plus long. La règle de passage d'une coupe maximale à une autre reste inchangée. La complexité de l'algorithme demeure également identique : $O(n^3)$.

On peut à présent décrire l'algorithme 4 correspondant à l'adaptation du schéma de génération Any-Order au cas du RCPSP/max. La structure du nouveau schéma de génération Any-Order/max reste très proche de celle de son homologue, hormis l'utilisation de la procédure Floyd-Warshall à la place de Bellman-Ford. La complexité temporelle de l'algorithme de Floyd-Warshall étant supérieure à celle de la procédure d'insertion, la complexité de Any-Order/max fait un saut à $O(n^4)$.

Une implémentation de ces algorithmes a été réalisée. Nous avons testé leur performance sur les instances disponibles sur le site internet du laboratoire de recherche opérationnelle de l'université de Karlsruhe (Allemagne). Les performances s'avèrent décevantes puisque les contraintes temporelles étant mises au même plan que l'augmentation du makespan induite par l'insertion, on obtient très souvent des solutions qui, si elles présentent un makespan de bonne qualité, sont la plupart du temps non réalisables, puisque violant à la fois les

Algorithme 3 Algorithme Search_Optimal_Insertion_Cut_Max - Adaptation de l'algorithme d'insertion au cas du RCPSP/max

repeat
  $best \leftarrow \infty$
  $j \leftarrow \text{argmax}_{\forall k \in S}\ \delta_{k0}$
  $remcap(k) \leftarrow R_k,\ \forall k \in R$
  repeat
    $(v,u) \leftarrow \text{argmax}_{y \in S, x \in P}\ \delta_{yx}$, avec $v \in S, u \in P$
    $\delta L_{\max} = \max_{x \in P}(\delta_{xi}, p_x) + \max_{y \in S}(\delta_{iy}, p_i) + \delta_{vu} - \delta_{0,n+2}$
    $remcap(k) \leftarrow remcap(k) - cap_{uk}\ \forall k \in R$
  until $remcap(k) < r_{ik},\ \forall k \in R$
  if $\delta L_{\max} < best$ **then**
    $best \leftarrow \delta L_{\max}$
    $C^* \leftarrow < P, S >$
  end if
  if $\delta_{j0} < \delta_{i0}$ **then**
    $NextMaxCut(P,S)$
  end if
until $\delta_{j0} \geq \delta_{i0}$
Return C^* **;**

Algorithme 4 Any-Order/max

$\varepsilon \leftarrow \{1...n\}$;
for i $\leftarrow$ 1 to n **do**
  $j \leftarrow$ Select_Activity(ε, R) ;
  $C^* \leftarrow$ Search_Optimal_Insertion_Cut_Max(j, G) ;
  Insert$(j, C^*$,G) ;
  Floyd-Warshall(G) ;
  $\varepsilon \leftarrow \varepsilon - \{j\}$;
end for

contraintes de délais minimaux et/ou maximaux. Dans un avenir proche, nous envisageons donc de modifier le schéma de génération Any-Order/max, en explorer davantage la piste consistant à ne relâcher que les contraintes de délais maximaux, dans l'espoir de produire des solutions davantage réalisables (voir section 2.8).

2.7 Réseaux logistiques et négociation d'intervalles

Comme dans la section précédente, cette partie s'intéresse à un problème davantage pratique : il s'agit de l'organisation de réseaux logistiques. Les travaux menés dans le cadre de la thèse de doctorat d'Emmanuelle Despontin-Monsarrat [Despontin-Monsarrat 04] sont présentés. Ils sont décrits avec plus de détails dans l'article [Monsarrat *et al.* 05] figurant au chapitre 5. Nous verrons qu'ici encore, la notion d'intervalles, de délais ou de quantités, est essentielle.

Signalons que ce travail a été réalisé dans le cadre d'une thèse en convention CIFRE avec la société OrdoSoftware de Besançon. Cette société commercialise des logiciels pour la gestion des unités de production, en particulier le logiciel d'ordonnancement en temps réel *ORDO*, issu des travaux du LAAS-CNRS. Au cours de réunions trimestrielles prévues pour faire le bilan des travaux de recherche, les ingénieurs de la société ont régulièrement contribué à l'affinement de la problématique, à la définition des orientations de recherche privilégiées, ainsi qu'à la validation des concepts et de l'approche.

2.7.1 Contexte de l'étude

La performance organisationnelle d'une entreprise repose non seulement sur sa capacité à optimiser ses propres objectifs locaux, mais aussi sur son aptitude à prendre en compte, voire anticiper, au sein de son organisation les comportements de ses partenaires. Ce deuxième axe de performance est devenu ces dernières années de plus en plus prépondérant. En effet, du fait de la nécessité de diversifier et de renouveler sans cesse les biens et/ou les services produits, une entreprise est aujourd'hui de plus en plus fréquemment amenée à collaborer avec un nombre croissant de partenaires : des fournisseurs de matières premières, de biens ou de services, des agences de travail intérimaires, des sous-traitants et bien sûr des clients. Elle s'intègre ainsi dans un réseau que l'on désigne communément sous le nom de chaînes logistiques ou de réseaux logistiques. Dans ce cadre, les décisions internes prises localement par l'entreprise sont de plus en plus dépendantes de celles prises par ses partenaires. Il est donc essentiel pour optimiser les performances de connaître au plus tôt les décisions des partenaires, de les intégrer, voire d'être en mesure de les influencer.

Les nouvelles technologies de l'information et de la communication (NTIC), en augmentant la qualité et la fréquence des échanges entre les partenaires, favorisent l'émergence de réseaux [Sardas *et al.* 02]. En revanche, elles n'apportent pas de solution générale au problème de la cohérence et de la qualité des décisions individuelles d'une entreprise, vis-à-vis de l'organisation globale en réseaux dont elle fait partie.

Le travail développé dans le cadre de la thèse d'Emmanuelle Despontin-Monsarrat

s'est intéressé au développement d'une méthode d'aide à la coopération interentreprise dans le cadre d'une production à la commande. L'approche retenue assimile le problème d'organisation globale du réseau à un processus de décision distribuée dans lequel l'organisation est progressivement construite par un ensemble de coopérations entre des couples d'acteurs du réseau d'entreprises. Nous étudions en particulier les couples d'entreprises donneur d'ordre-fournisseur pour lesquels la coopération concerne les attributs des commandes de produits.

2.7.2 Cadre de décision

La formalisation proposée pour la coopération s'appuie principalement sur la notion de cadre de décision. Un cadre de décision représente un engagement de mise à disposition par un fournisseur auprès de son donneur d'ordre, d'une quantité variable de produits, dans un ou plusieurs intervalles temporels. Soulignons que cet engagement de mise à disposition sous-tend un engagement réciproque de consommation de la part du client. Ainsi, un cadre de décision peut être assimilé à un ensemble de contraintes portant sur l'état du stock de sortie d'un fournisseur dans certains intervalles temporels, ou sur l'état du stock d'entrée d'un client, à une translation près, si les temps de transport sont pris en compte. Dans la mesure où les objectifs de délais et de quantités correspondent à des intervalles, un cadre de décision modélise une décision flexible, flexibilité qui sera utilisée par les partenaires pour gérer les nouvelles commandes et les imprévus.

La figure 2.30 illustre cette notion de cadre de décision. La commande est associée à plusieurs cadres de décision, collectivement définis. L'objectif final est de fournir 650 produits exactement, pour le 28 février précisément. Cet objectif n'est pas flexible. Par contre, on suppose que fournisseur et donneur d'ordre se sont entendus pour que cette commande soit satisfaite progressivement de sorte que le fournisseur ait délivré au donneur d'ordre :

- entre 50 et 200 unités de produits entre le 1er et le 4 février, puis ;
- entre 200 et 550 unités de produits entre le 10 et le 12 février, et enfin ;
- entre 400 et 650 unités de produits entre le 23 et 24 février.

L'aire des rectangles et leurs positions rendent compte de la flexibilité dont dispose le fournisseur pour réaliser sa production. En effet, plus la surface d'un rectangle est importante ou, à surface égale, plus la position d'un rectangle est loin de l'origine du temps, plus le fournisseur dispose de flexibilité pour respecter son engagement. Le bon dimensionnement d'un cadre de décision est le résultat d'un compromis qui satisfait tout à la fois les objectifs internes du fournisseur et ceux du donneur d'ordre. Un cadre de décision doit donc satisfaire deux points de vue.

Remarquons d'autre part qu'un cadre de décision n'est pas figé. Au fur et à mesure

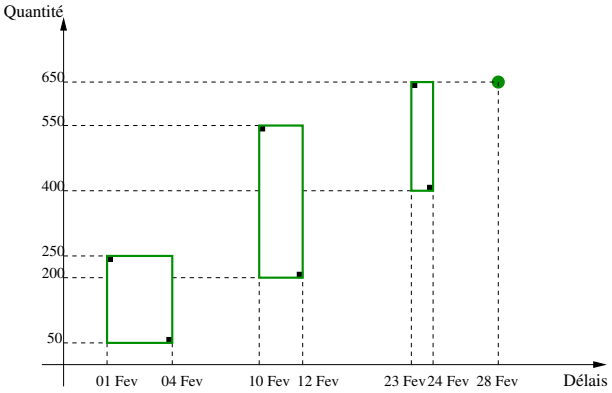


FIGURE 2.30 – Cadres de décision pour une commande

de l'avancée de la production, lorsque les besoins des clients et les plans de fabrication des fournisseurs s'affinent, les acteurs peuvent modifier les cadres de décision de sorte qu'ils soient en adéquation avec les courbes prévisionnelles de production et de demande. Il est également possible d'augmenter le nombre de cadres relatif à une commande. Ces affinements des cadres de décision se font par renégociations successives au fur et à mesure de l'avancée de la production.

Si on suppose que les cadres de décision sont les supports des coopérations donneurs d'ordres - fournisseurs, deux problématiques se posent. La première concerne l'organisation de la coopération elle-même : selon quelles règles les décideurs peuvent-ils / doivent-ils négocier pour que la coopération soit efficace ? La seconde concerne les attributs des cadres de décision : comment définir les intervalles de délais et de quantités au mieux afin de satisfaire les besoins réciproques des donneurs d'ordres et fournisseurs ? Nous montrons comment nous répondons à ces problématiques dans les parties suivantes.

2.7.3 Le protocole de coopération

Nous postulons qu'un processus de coopération comporte trois aspects distincts : la négociation, la coordination et la renégociation. Dans ce qui suit, nous illustrons ces aspects en nous appuyant sur des copies d'écran de fenêtres d'une interface graphique prototype [2].

Un processus de négociation est initié lorsqu'un des partenaires émet une proposition : par exemple si un donneur d'ordre souhaite passer une commande ou, inversement, si un fournisseur veut faire une offre commerciale à un de ses donneurs d'ordre. La figure 2.31

2. J'ai développé cette interface avec l'aide d'un stagiaire de recherche, Cédric Mocquillon, de Polytech'Tours. Elle a été conçue dans l'intention de valider avec notre partenaire industriel les concepts développés durant la thèse.

illustre un cas concret où un fournisseur reçoit une proposition de commande concernant un produit *écran plat* par le donneur d'ordre *Despontin*. De façon générale, une conversation correspond à une suite de propositions - contre-propositions et s'achève soit par une acceptation, soit par un refus. Comme on peut le voir sur l'arbre placé dans la zone gauche de la fenêtre, l'interface graphique retrace le fil de chaque négociation. Ici par exemple, dans le cadre de la négociation de la nouvelle commande du donneur d'ordre *Despontin*, il y a avait déjà eu une proposition lancée par le client, et le fournisseur avait soumis une contre-proposition. L'interface permet par simple clic, d'observer les caractéristiques de chaque proposition / contre-proposition, et donc de conserver une trace des négociations.

On peut distinguer des propositions de deux natures. Une proposition dite *pour engagement*, si elle est acceptée, induit un engagement ferme d'un fournisseur. A contrario, une proposition *pour évaluation* induit une réponse concernant la faisabilité d'une commande (sans aucune notion d'engagement). Dans le premier cas, le processus de négociation se termine soit par un engagement (acceptation), soit par un refus. Dans le deuxième cas, le processus de négociation conclut soit à la faisabilité (acceptation), soit à l'infaisabilité (refus). Conclure à la faisabilité d'une proposition n'engage aucun des partenaires, une négociation pour engagement sera tout de même nécessaire. De plus, cette dernière n'aboutira pas nécessairement au même cadre de décision que celui déterminé lors de l'analyse de faisabilité puisque le contexte aura changé.

La coordination est postérieure à la négociation, si celle-ci a effectivement débouché sur une commande (un cadre de décision a été défini). Elle consiste en l'envoi de messages permettant de synchroniser les informations disponibles entre parties. Ces messages peuvent être générés automatiquement et concernent les délais, les quantité ou les prix. Ainsi que le représente la figure 2.32, les messages à envoyer apparaissent, pour chaque commande en cours, dans l'arbre situé dans la zone droite de la fenêtre. Quatre possibilités (boutons au centre) s'offrent alors au décideur : envoyer, différer, modifier ou annuler l'envoi d'un message. Une fois envoyer, le message passe dans l'arbre de la zone gauche de la fenêtre qui retrace l'historique des messages.

À la réception d'un message, un donneur d'ordre (resp. un fournisseur) peut vérifier que la quantité délivrée par le fournisseur (resp. la quantité attendue par le client) est bien en cohérence en quantité et en heure à la fois avec les cadres de décisions et avec son besoin en composants (resp. avec son objectif de production).

Un processus de renégociation est initié unilatéralement par un des partenaires si celui-ci souhaite modifier un cadre de décision, par exemple parce que celui-ci est devenu incohérent avec son état interne. Pour entamer la renégociation, le décideur émet une proposition que le récepteur pourra, similairement au processus de négociation, soit accepter, soit refuser, soit discuter (contre-proposition).

Nous postulons également que les processus de négociation, coordination et

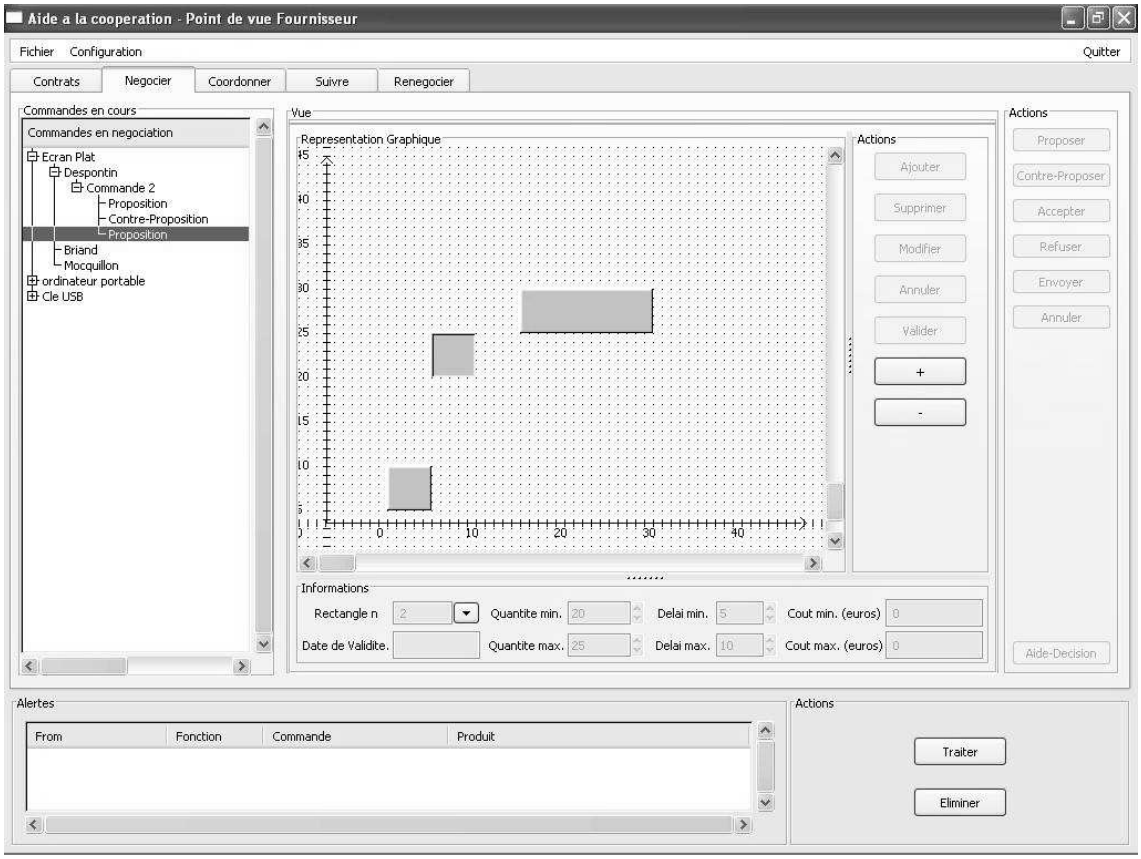


FIGURE 2.31 – Interface pour la négociation

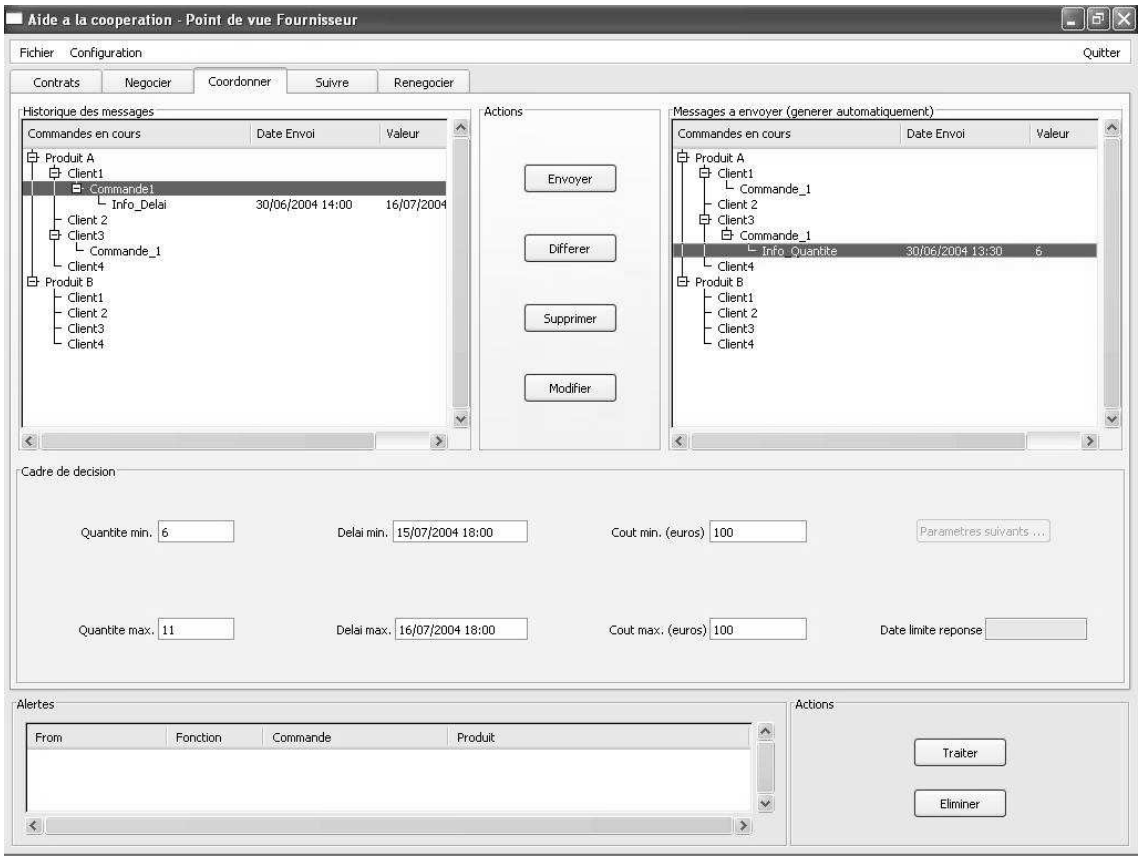


FIGURE 2.32 – Interface pour la coordination

renégociation doivent eux-mêmes satisfaire un ensemble de règles de bon comportement, ces règles ayant été collectivement définies et acceptées par les décideurs. Ces règles forment ce que nous nommons le *contrat de coopération*. Ainsi que le représente la figure 2.33, le contrat se décline en fonction de l'horizon temporel considéré : on ne négocie pas de la même manière s'il s'agit d'un cadre de décision portant sur une commande à réaliser pour le lendemain, ou s'il s'agit d'une commande à réaliser dans un mois.

2.7.4 Aide à la décision

Comment aider les décideurs, lors de leurs négociations / renégociations successives, à dimensionner des cadres de décision qui soient d'une part, adaptés à leurs besoins propres et qui, d'autre part, permettent d'atteindre un compromis satisfaisant lorsque les objectifs des parties sont contradictoires ? Ce paragraphe apporte des réponses à cette question.

Tout d'abord, nous supposons que l'entreprise est décomposée en 3 centres de décision

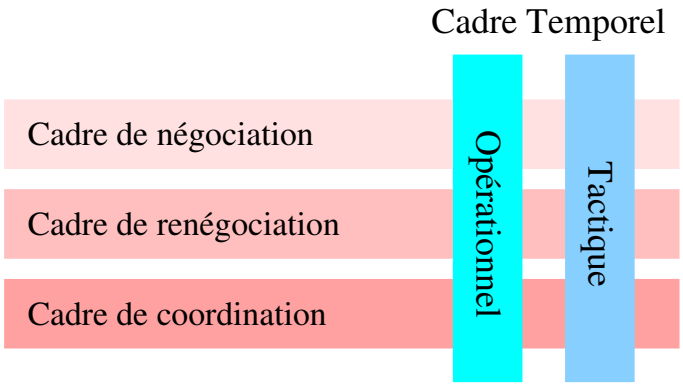


FIGURE 2.33 – Le contrat de coopération

(cf. figure 2.34) : le centre de décision *Gestion de Production*, le centre de décision *Service Vente* et le centre de décision *Service Achat*. Le centre de décision *Gestion de Production* regroupe toutes les fonctions classiques de la gestion de production d'une entreprise, et en particulier celles relatives aux systèmes de conduite et de fabrication. Le centre de décision *Service Vente* regroupe les services liés à la vente des produits d'une entreprise. Ce centre gère les relations d'une entreprise avec ses clients. Le centre de décision *Service Achat* correspond aux services liés à l'achat de composants nécessaires à la fabrication d'une entreprise. Ce centre gère les relations d'une entreprise avec ses fournisseurs.

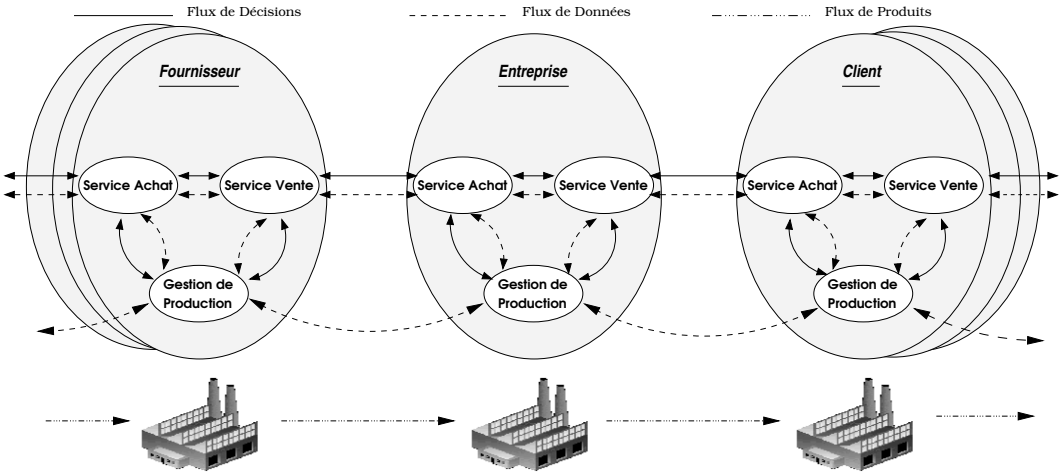


FIGURE 2.34 – Relations entre les centres de décision d'une chaîne logistique

Nous nous sommes intéressés aux centres de décision *Service Achat* et *Service Vente* de deux entreprises distinctes d'une part, et aux relations entre les centres de décision *Service Achat* et *Service Vente* d'une même entreprise d'autre part.

Nous supposons que les comportements temporels des centres *Achat* et *Vente* sont

chacun décrits par un modèle spécifique. On suppose que l'horizon est agrégé en périodes, de longueurs éventuellement différentes. Pour chaque période, le modèle du service *Vente* intègre les contraintes liées à la disponibilité des ressources, celles liées au respect des gammes opératoires (éventuellement non-linéaires) et celles liées au respect des cadres de décision (il y a donc une flexibilité quant au respect des demandes). Le modèle du service *Achat* est plus simple et intègre simplement les contraintes liées à la satisfaction des besoins en composants et évidemment celles relatives au respect des cadres de décision (il y a donc une flexibilité quant à l'approvisionnement en composants). Pour plus de détail sur ces modèles, le lecteur peut se référer à l'article [Monsarrat *et al.* 05] figurant dans le chapitre 5.

Lors de la création d'un nouveau cadre ou lors de l'adaptation d'un cadre existant, nous utilisons des mécanismes de propagation de contraintes pour mettre en évidence auprès du décideur la flexibilité décisionnelle encore disponible.

Par exemple, plaçons nous du point de vue d'un fournisseur souhaitant répondre à la sollicitation d'un client. C'est le service *Vente* de ce fournisseur qui réceptionne la proposition du client. Par propagation de contraintes appliquée au modèle de ce centre de décision, il est possible de déterminer, ainsi que l'illustre la figure 2.35, deux courbes :

- une courbe cumulée de production maximale au plus tôt qui est obtenue en supposant que tous les cadres de décision existants sont satisfaits au plus tard à leur minimum ;
- une courbe cumulée de production minimale au plus tôt qui est obtenue en supposant que l'on ne modifie pas le plan de production établi par la gestion de production.

Ces courbes mettent en évidence la capacité de production restante pour satisfaire une nouvelle commande. Pour définir un nouveau cadre de décision, un décideur aura ainsi intérêt à fixer le point supérieur gauche entre les 2 courbes, et celui inférieur droit en dessous de la courbe la plus basse. En faisant de la sorte, on assure que chaque point du cadre de décision est accessible sans violation des cadres de décision existants. Toutefois, par effet de vase communiquant, plus l'objectif de production du nouveau cadre sera situé en haut à gauche, plus les objectifs des cadres existants devront être dégradés. Sur la figure 2.35, le cadre de décision est par exemple défini par 2 rectangles ce qui correspond à une stratégie de fractionnement de la commande.

Les mécanismes mis en jeu lors l'adaptation d'un cadre existant sont relativement similaires. Ils consistent à autoriser le relâchement des objectifs de production prévus pour les autres cadres, puis à observer les conséquences sur les 2 courbes précédentes. Pour résumer, plus on accepte de dégrader les objectifs des autres cadres et plus la courbe cumulée de production maximale au plus tôt augmente pour le cadre considéré. Si la dégradation de l'objectif d'un cadre passe en dessous du minimum collectivement défini (point inférieur droit), alors la courbe cumulée de production minimale au plus tôt

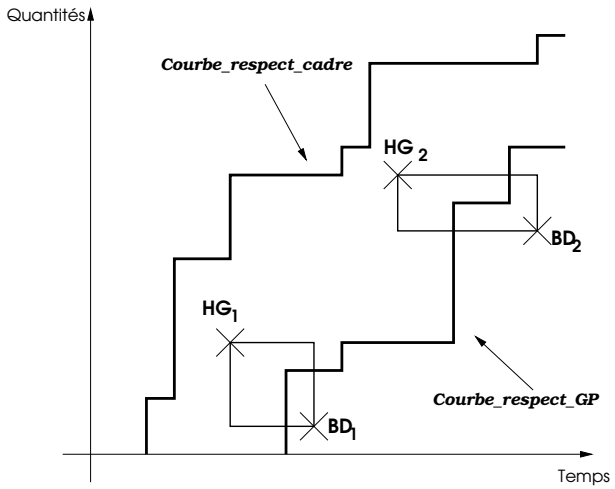


FIGURE 2.35 – Création d'un cadre de décision

augmente également pour le cadre considéré. Cependant, il sera nécessaire de renégocier le cadre ainsi relâché, avec le partenaire concerné, puisqu'il sera incohérent.

Les mécanismes d'aide à la décision décrits plus haut sont relativement génériques et peuvent être appliqués aussi bien pour un centre de décision *Vente* que pour un centre *Achat*. De plus, ils mettent en évidence que, lors de la renégociation d'un cadre existant, plusieurs stratégies s'offrent au décideur pour recouvrer la cohérence. Il peut soit adapter sa production en interne (la répartir différemment), soit adapter en amont les cadres de décision de son centre achat (modification de l'approvisionnement en composants), soit encore renégocier en aval les cadres de décision de son centre *Vente* (modification de la demande). Le processus de coopération proposé semble donc bien adapté à gérer toutes les situations.

2.8 Projet de recherche

Cette partie décrit comment je pense orienter mes recherches durant les prochaines années. Trois voies sont principalement distinguées.

2.8.1 Axe 1 : Utilisation de conditions de dominance

Le travail relatif à la mise en évidence, par analyse d'intervalles, de nouvelles conditions de dominance ou de conditions suffisantes d'optimalité a été passionnant. Dans l'immédiat, je souhaite le poursuivre en commençant par tenter d'étendre certains des résultats obtenus à d'autres problèmes comme :

- le problème à une machines où plusieurs intervalles d'exécution alternatifs sont associés à chaque travail ;
- le problème à machine parallèle avec intervalles d'exécution $(Pm|r_j|L_{\max})$;
- Le problème flowshop à deux machines où l'on souhaite minimiser la somme des dates de fin des travaux $(F2|prmu|\sum C_j)$.

Le but serait de concevoir des nouvelles techniques de résolution exacte de ces problèmes. Pour les deux premiers points, il s'agit d'analyser comment la formulation PLNE, présentée dans la section 2.3.3 pour le problème $1|r_j|L_{\max}$, pourrait être adaptée pour considérer des environnements d'ordonnancement plus complexes.

Dans ce mémoire, l'analyse d'intervalles que nous avons proposée considère exclusivement des structures d'intervalles particulières où les bornes des intervalles sont *totalement* ordonnées. À moyen terme, il serait intéressant d'envisager des structures où cet ordre ne serait plus que *partiel*. En effet, si on s'intéresse par exemple à un problème de type job shop, et que l'on considère l'ordonnancement local propre à chaque machine, il est possible d'associer des intervalles aux dates de début et de fin de chaque tâche. Pour le problème à une machine correspondant, les valeurs de r_j et d_j étant incluses dans ces intervalles, on peut définir un ordre partiel entre les dates de disponibilité et d'échéance des tâches. Dans ce cas, il devient difficile de caractériser précisément les tâches sommets, les tâches bases, et dire quelle tâche appartient à quelle pyramide puisque la même structure d'intervalles peut engendrer, selon le séquencement choisi sur les autres ressources, plusieurs structures pyramidales possibles. Cependant, une analyse de ces intervalles permet de déduire des informations quant aux structures pyramidales possibles. Ces informations peuvent à leur tour permettre de restreindre l'espace de recherche à envisager lors de la résolution.

L'exemple basique représenté sur la figure 2.36 a pour but d'illustrer cette idée. On considère 4 tâches i, j, k et l, devant être réalisées sur une ressource disjonctive unique. On suppose que le respect des contraintes temporelles entre ces tâches et celles réalisées sur les autres ressources impose (par propagation) les dates de début au plus tôt est, de début au plus tard lst, de fin au plus tôt eft et de fin au plus tard lft, indiquées sur la figure. Étant donné les relations existant entre les intervalles $[est, lst]$ et $[eft, lft]$, on peut déduire que, *quel que soit le séquencement choisi sur les autres ressources* :

- la tâche i est la base unique de 1 ou 2 pyramides dont les sommets seront j et/ou k puisque, a priori, on ne peut pas savoir si l'intervalle d'exécution de k sera inclus dans celui de j ($during(k, j)$, ou si les intervalles d'exécution de k et j se chevaucheront ($overlaps(k, j)$), auquel cas les deux tâches seront sommets ; quelle que soit la configuration, on peut déduire que la sous séquence $j \prec i \prec k$ est toujours dominée ;
- la tâche l est à la fois sommet et base d'une pyramide d'indice supérieur au(x) 1 ou 2 précédente(s), l est donc nécessairement séquencée après toutes les autres tâches, ce qui peut induire une actualisation de la date de début au plus tôt de l et des dates de fin au plus tard des autres tâches.

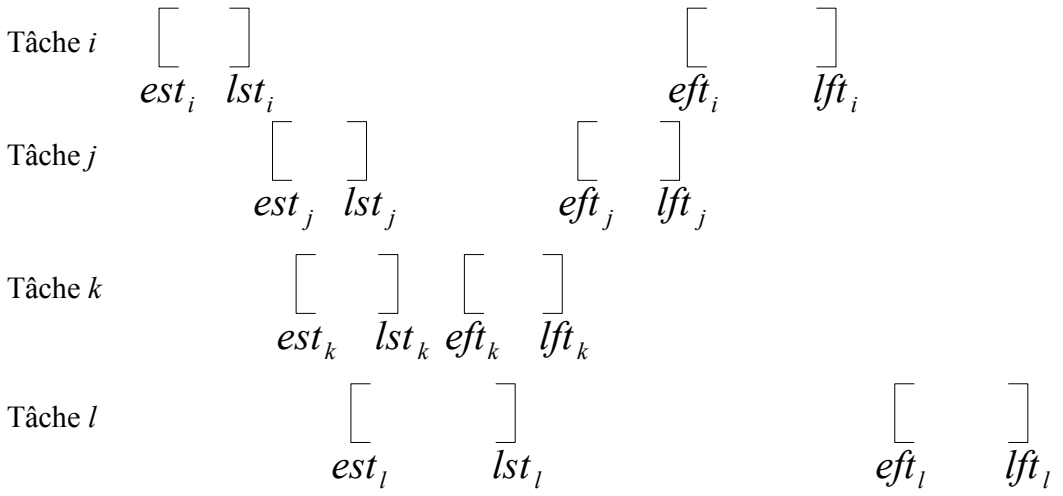


FIGURE 2.36 – Une structure d'intervalles associée à un ordre partiel

Comme on le constate sur cet exemple, l'analyse d'intervalles permet de déduire des informations quant aux diverses structures pyramidales possibles. Ces informations peuvent être exploitées pour filtrer l'espace de recherche de problèmes d'ordonnancement à ressources disjonctives (flowshop, job shop, open shop ...). L'idée serait de mettre au point des techniques de filtrage utilisant des conditions de dominance et basée sur l'analyse d'intervalles, aussi générique que celles, très fréquemment utilisées, utilisant des conditions

nécessaires d'admissibilité [Baptiste *et al.* 01]. Il serait alors possible, en s'autorisant à délaisser certaines solutions admissibles (nécessairement dominées), de procéder à un filtrage plus serré que si on utilisait exclusivement des conditions nécessaires d'admissibilité (voir figure 2.37), cette hypothèse restant bien sûr à valider. On pourrait également envisager une coopération entre ces deux techniques, l'une pouvant être complémentaire de l'autre ainsi que l'illustre la figure 2.37.

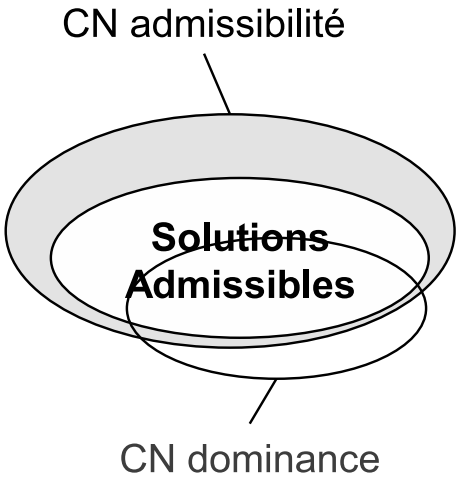


FIGURE 2.37 – Utilisation de conditions nécessaires / Utilisation de conditions de dominance

2.8.2 Axe 2 : Méthodes d'ordonnancement coopératives et robustes

Au delà de l'utilisation de méthodes robustes en ordonnancement, il semble intéressant d'étudier les interactions entre robustesse et coopération en ordonnancement. En effet, les progrès récents dans le domaine des sciences de l'information et de la communication rendent aujourd'hui la conception de systèmes d'ordonnancement coopératifs possible. Les domaines applicatifs dans lesquels une approche coopérative de l'ordonnancement serait utile sont nombreux : la production multi-entreprise (chaîne logistique), l'ordonnancement de projet multi-partenaire ou multi-projet, la construction d'emploi du temps, l'ordonnancement de grille de calcul, ...

Aujourd'hui, la plupart des systèmes existants restreignent la coopération au simple échange d'information, ce qui permet uniquement la coordination des acteurs. Au delà de cette première fonctionnalité, une évolution naturelle serait de permettre aux acteurs de collaborer dans le but de leur donner la possibilité de négocier leur décision

d'ordonnancement et, ainsi, d'améliorer leur organisation collective. Bien que naturelle, cette évolution pose de nombreux problèmes : quelles informations partagées ? comment négocier les décisions ? comment garantir l'efficacité d'un processus de négociation ? Comment assurer un bon compromis entre les objectifs de chaque partenaire et un objectif collectif ?

Le projet ANR Blanc ROBOCOOP, amorcé en janvier 2009 en partenariat avec le LI de Tours, le LITIS du Havre et IBM/ILOG, propose d'étudier ces aspects et d'apporter des éléments de réponse, sous l'hypothèse forte que les ordonnancement locaux offre une robustesse permettant de faire face aux incertitudes présentes dans le réseau d'acteurs. Ce projet donnera lieu, en septembre 2009, au lancement de la thèse de Thomas Lehaux que j'encadrerai, en lien avec les autres partenaires du projet ROBOCOOP.

On suppose que les ressources ou les travaux sont distribués parmi un ensemble d'acteurs ayant leur propre autonomie décisionnelle, leurs propres objectifs et une connaissance restreinte de leur environnement. Dans ce cadre, l'utilisation de méthodes de décision robustes semble souhaitable pour permettre à chaque acteur de faire face, non seulement aux incertitudes liées à son organisation interne, mais aussi et surtout à celles liées à son environnement (les décisions prises par les autres acteurs étant susceptibles de fluctuer dans le temps). Notre ambition est de considérer des organisations distribuées de plus en plus complexes, en considérant en premier lieu le problème à une machine où les travaux sont répartis entre plusieurs acteurs qui se partagent la machine, puis, le cas dual où les acteurs détiennent chacun une ressource et doivent se partager un ensemble de travaux, pour progressivement arriver à traiter les cas plus complexes d'ordonnancement multi-projet dans lequel les projets (et éventuellement les ressources) sont réparties entre plusieurs acteurs. Dans chacun des cas, on supposera l'existence d'un objectif global poussant les acteurs à coopérer, ce qui différenciera cette recherche d'autres approches où les acteurs sont en compétition pure. Il s'agit d'identifier, selon la nature des informations échangées, un formalisme mathématiques permettant de supporter les processus de coopération dans le but de faciliter la recherche de compromis entre la satisfaction des objectifs locaux et celle de l'objectif collectif considéré. Pour cela, des techniques de décomposition en programmation mathématique, d'optimisation multi-objectif et de programmation par contraintes seront explorées.

2.8.3 Axe 3 : Ordonnancement de projet avec contraintes de délais minimaux et maximaux

Un dernier axe que je souhaiterai continuer à développer, en collaboration avec Christian Artigues, concerne la conception de méthodes exactes et heuristiques pour l'ordonnancement de projet avec contraintes de délais maximaux.

Un premier point serait de proposer une extension du schéma de génération Any-Order, que nous proposons pour le RCPSP classique, au cas du RCPSP avec contraintes de délais maximaux. Comme discuté plus haut, il s'agit dans un premier temps de proposer une relaxation des contraintes de délais maximaux, qui permettent d'adapter et la procédure d'insertion décrite dans [Artigues & Briand 09], utilisable en présence de délais minimaux. L'objectif est de déterminer une position d'insertion minimisant, d'une part, la violation des contraintes de délais maximaux et, d'autre part, minimisant l'augmentation du makespan. Cette procédure pourrait ensuite être intégrée au sein d'un schéma de génération de type Any-Order.

Un deuxième point serait de développer des méthodes de résolution par exploration de voisinage, exploitant éventuellement les solutions construites grâce au schéma Any-Order, afin de déterminer une solution réalisable présentant un makespan le plus faible possible, ou à défaut, une solution violant le moins possible les contraintes de délais maximaux et présentant un makespan le meilleur possible (étant donné le seuil maximum de viol des contraintes de précédence autorisé). La conception de méthodes exactes, de type branch-and-bound, pourra également être envisagée.

2.9 Références bibliographiques

[Adams *et al.* 88] J. Adams, E. Balas & D. Zawack. *The shifting bottleneck procedure for job shop scheduling.* Management Science, vol. 34, no. 3, pages 391–401, 1988.

[Allen 91] J.F. Allen. *Time and time again : The many ways to represent time.* International Journal of Intelligent Systems, vol. 6, no. 4, pages 341–355, 1991.

[Artigues & Briand 09] C. Artigues & C. Briand. *The resource-constrained activity insertion problem with minimum and maximum time lags.* Journal of Scheduling, vol. X, pages xx–yy, 2009.

[Artigues & Roubellat 00] C. Artigues & F. Roubellat. *A polynomial activity insertion algorithm in a multi-resource schedule with cumulative constraints and multiple modes.* European Journal of Operational Research, vol. 127, pages 297–216, 2000.

[Artigues *et al.* 03] C. Artigues, P. Michelon & S. Reusser. *Insertion Techniques for static and dynamic resource-constrained project scheduling.* European Journal of Operational Research, vol. 149, pages 249–267, 2003.

[Artigues 97] C. Artigues. *Ordonnancement en temps réel d'ateliers avec temps de préparation des ressources.* Thèse de troisième cycle, Université Paul Sabatier, Toulouse, France, 1997.

[Baptiste *et al.* 01] P. Baptiste, C. Le Pape & W. Nuijten. Constraint-based scheduling. Kluwer Academic Publishers, Boston/Dordrecht/London, 2001.

[Baptiste *et al.* 03] P. Baptiste, L. Peridy & E. Pinson. *A Branch and Bound to Minimize the Number of Late Jobs on a Single Machine with Release Time Constraints.* European Journal of Operational Research, vol. 144, no. 1, pages 1–11, 2003.

[Belman *et al.* 82] R. Belman, A.O. Esogbue & I. Nabeshima. Mathematical aspects of scheduling and applications. Pergamon press, Oxford, 1982.

[Benoît & Billaut 00] M. Benoît & J-C. Billaut. *Characterization of the optimal solutions of the F2//Cmax scheduling problem.* Dans 2nd Conference on

Management and Control of Production and Logistic (MCPL 2000), Grenoble, France, 2000.

[Billaut & Lopez 98] J-C. Billaut & P. Lopez. *Enumeration of all optimal sequences in the two-machine flowshop.* Dans Computational Engineering in Systems Application (CESA'98), Symposium on Industrial and Manufacturing Systems, IEEE-SMC/IMACS, pages 378–382, Nabeul-Hammamet, Tunisie, 1998.

[Billaut *et al.* 05] J.-C Billaut, A. Moukrim & E. Sanlaville. Flexibilité et robustesse en ordonnancement. Hermès Science Publications, 2005.

[Billaut *et al.* 08] J.-C Billaut, A. Moukrim & E. Sanlaville. Flexibility and robustness in scheduling. ISTE - Wiley, 2008.

[Billaut 93] J-C. Billaut. *Prise en compte de ressources multiples et des temps de préparation dans les problèmes d'ordonnancement en temps réel.* Thèse de doctorat, Université Paul Sabatier, Toulouse, France, 1993.

[Blazewicz *et al.* 96] J. Blazewicz, K.H. Ecker, E. Pesch, G. Schmidt & J. Weglarz. Scheduling in computer and manufacturing processes. Springer Verlag, 1996.

[Briand & Bezanger 06] C. Briand & J. Bezanger. *An any-order SGS for project scheduling with scarce resources and precedence constraints.* Dans 10th International Workshop on Project Management and Scheduling (PMS'2006), pages 95–100, Poznan, Pologne, 2006.

[Briand *et al.* 05] C. Briand, M-J. Huguet, H.T. La & P. Lopez. *Approches par contraintes pour l'ordonnancement robuste.* Dans J-C. Billaut, A. Moukrim & E. Sanlaville éditeurs, Flexibilité et Robustesse en Ordonnancement, pages 191–215. Hermes Science Publication, 2005.

[Briand *et al.* 06a] C. Briand, H.T. La & J. Erschler. *A new sufficient condition of optimality for the two-machine flowshop problems.* European Journal of Operational Research, vol. 169, pages 712–722, 2006.

[Briand *et al.* 06b] C. Briand, S. Ourari & B. Bouzouia. *Minimizing the number of late jobs in single machine scheduling with nested execution intervals.* Dans IEEE International Conference on Services Systems and Services Management, Troyes, France, 2006.

[Briand *et al.* 07] C. Briand, H.T. La & J. Erschler. *A robust approach for the single machine scheduling problem.* Journal of Scheduling, vol. 10, pages 209–221, 3 2007.

[Briand *et al.* 10] C. Briand, S. Ourari & B. Bouzouia. *An efficient ILP formulation for the single machine scheduling problem.* RAIRO - Operations Research, vol. X, pages xx–yy, 2010.

[Briand 00] C. Briand. *Using Petri nets for constraint satisfaction.* International Journal of Decision Systems, vol. 9, no. 1, pages 77–97, 2000.

[Briand 09] C. Briand. *A new Any-Order schedule genration scheme for resource-constrained project scheduling.* RAIRO - Operations Research, vol. X, pages xx–yy, 2009.

[Carlier & Pinson 89] J. Carlier & E. Pinson. *An algorithm for solving the job-shop problem.* Management Science, vol. 35, no. 2, pages 164–176, 1989.

[Carlier 82] J. Carlier. *The one-machine sequencing problem.* European Journal of Operational Research, vol. 11, pages 42–47, 1982.

[Chu *et al.* 92] C. Chu, M.-C. Portmann & J.-M. Proth. *A Splitting up Approach to Simplify Job-shop Scheduling Problems.* International Journal of Production Research, vol. 30, no. 4, pages 859–870, 1992.

[Coffman 76] E.G. Coffman. Computer and job-shop scheduling theory. John Wiley and Sons, 1976.

[Couzinet-Mercé 79] C. Couzinet-Mercé. *Étude de l'existence de solutions pour certains problèmes d'ordonnancement.* Thèse de Doctorat, Université Paul Sabatier, Toulouse, France, 1979.

[Davenport *et al.* 01] A.J. Davenport, C. Gefflot & J.C. Beck. *Slack-based Techniques for Robust Schedules.* Dans 6th European Conference on Planning (ECP-2001), Toledo, Spain, 2001.

[Despontin-Monsarrat 04] Emmanuelle Despontin-Monsarrat. *Aide à la décision pour le pilotage et l'ordonnancement coopératifs d'ateliers.* Thèse de troisième cycle, Université Paul Sabatier, Toulouse, France, décembre 2004.

[Erschler *et al.* 83] J. Erschler, G. Fontan, C. Merce & F. Roubellat. *A New Dominance Concept in Scheduling n Jobs on a Single Machine with Ready Times and Due Dates.* Operations Research, vol. 31, no. 1, pages 114–127, 1983.

[Esquirol & Lopez 99] P. Esquirol & P. Lopez. L'ordonnancement. Economica, 1999.

[Esquirol & Lopez 02] P. Esquirol & P. Lopez. Structures temporelles pour le problème d'ordonnancement à une machine. Rapport LAAS, 2002.

[Esswein & Billaut 02] C. Esswein & J-C. Billaut. *Trade-off between flexibility and maximum completion time in the two-machine flowshop scheduling problem.* Dans International Symposium on Combinatorial Optimisation, Paris, France, 2002.

[Gao 95] H. Gao. Building robust schedules using temporal protection- an empirical study of constraint based scheduling under machine failure uncertainty. Technical Report TR-EIL-95-2, Enterprise Integration Lab, 1995.

[Goldratt 97] E. Goldratt. Critical chain. Great Barrington : The North River Press, 1997.

[GOThA 02] GOThA. *Flexibilité et Robustesse en Ordonnancement.* Le bulletin de la ROADEF, vol. 8, pages 10–12, 2002.

[Herroelen & Leus 04] W. Herroelen & R. Leus. *The construction of stable project baseline schedules.* European Journal of Operational Research, vol. 156, pages 550–565, 2004.

[Herroelen *et al.* 99] W. Herroelen, E. Demeulemeester & B. De Reyck. *A classification scheme for project scheduling.* Dans J. Weglarz éditeur, Project Scheduling - Recent Models, Algorithms and Applications, pages 1–26. Kluwer's International Series, London, 1999.

[La *et al.* 05] H.T. La, J.-L. Santamaria & C. Briand. *Une aide à la décision pour l'ordonnancement robuste en contexte mono-ressource : un compromis flexibilité/performance.* Dans 6ème Congrès de la Société Française de Recherche Opérationnelle et d'Aide à la Décision (ROADEF'05), papier long, pages 101–116, Tours, France, 2005.

[La 05] H.T. La. *Utilisation d'ordres partiels pour la caractérisation de solutions robustes en ordonnancement.* Thèse de troisième cycle, Institut National des Sciences Appliquées, Toulouse, France, janvier 2005.

[Le Gall 89] A. Le Gall. *Un système interactif d'aide à la décision pour l'ordonnancement et le pilotage en temps réel d'atelier.* Thèse de doctorat, Université Paul Sabatier, Toulouse, France, 1989.

[Leung 04] J. Y-T Leung. Handbook of scheduling : Algorithms, models, and performance analysis. Chapman & Hall/CRC, 2004.

[Mehta & Uzsoy 98] S.V. Mehta & R.H. Uzsoy. *Predictable Scheduling of a Job Shop Subject to Breakdowns.* IEEE Transactions on Robotics and Automation, vol. 14, no. 3, pages 365–378, 1998.

[Monsarrat *et al.* 05] E. Monsarrat, C. Briand & P. Esquirol. *Aide à la décision pour une coopération interentreprise.* Journal Européen des Systèmes Automatisés, vol. 38, no. 7, pages 799–818, 2005.

[Neumann *et al.* 02] K. Neumann, C. Schwindt & J. Zimmermann. Project scheduling with time windows and scarce resources. Lecture Notes in Economics and Mathematical Systems, Springer, 2002.

[Ourari *et al.* 09] S. Ourari, C. Briand & B. Bouzouia. *A mathematical programming approach to minimize the number of late jobs for the single machine scheduling problem.* in revision for the Journal of Mathematical Modelling and Algorithms, pages XX–YY, 2009.

[Pinedo 08] M. Pinedo. Scheduling : Theory, algorithms, and systems. Springer, 2008.

[Sardas *et al.* 02] J.C. Sardas, J. Erschler & G. De Terssac. *Coopération et organisation de l'action collective.* Dans Coopération et connaissance dans les

systèmes industriels sous la direction de R. Soënen et J. Perrin, pages 69–90. Lavoisier, Hermes Science, 2002.

[Thomas 80] V. Thomas. *Aide à la décision pour l'ordonnancement d'atelier en temps réel*. Thèse de troisième cycle, Université Paul Sabatier, Toulouse, France, 1980.

Chapitre 3

Formation et encadrement

Dans ce chapitre, sont listés de façon synthétique les encadraments auxquels j'ai pris part. Trois catégories d'étudiants sont distinguées : doctorants, étudiants en master recherche 2ème année et stagiaires de recherche d'été. J'aurai pu également ajouter les encadrements de stages d'élèves ingénieurs (2 par an en moyenne) pour lesquels j'ai effectué à chaque fois une visite en entreprise et une lecture du rapport de stage avant de participer au jury de soutenance.

3.1 Doctorants

à partir de septembre 2009 : Encadrement (100%) de la thèse de **Thomas Lehaux**, " Méthodes d'ordonnancement coopératives et robustes ", dans le cadre du projet ANR Blanc ROBOCOOP.

depuis février 2006 : Encadrement (100%) de la thèse de **Samia Ourari**, " Ordonnancement réactif sous incertitudes ", soutenance prévue en janvier 2010.

Novembre 2001 - janvier 2005 : Co-encadrement (70%) de la thèse de doctorat INSA de **Hoang Trung LA** " Utilisation d'ordres partiels pour la caractérisation de solutions robustes en ordonnancement ", Co-encadrant : Jacques Erschler - Soutenue le 24 janvier 2005.

Septembre 2001 - décembre 2004 : Co-encadrement (50%) de la thèse de doctorat UPS d'**Emmanuelle Despontin-Monsarrat** (CIFRE avec la société ORDOSOFTWARE) " Aide à la décision pour le pilotage et l'ordonnancement coopératifs d'ateliers ", Co-encadrant : Patrick Esquirol - Soutenue le 10 décembre 2004.

3.2 Stages de recherche de niveau Master

2008-2009 : Encadrement (100%) du stage de Master Recherche 2ème année SAID de **Franklin Cunalata** (UPS), "Flowshop biobjectif à deux machines"

2007-2008 : Encadrement (50%) du stage de Master Recherche 2ème année SAID de **Abdelhamid Younes** (UPS), "Heuristiques pour l'ordonnancement de projet avec contraintes de délais minimaux et maximaux", co-encadrant : Christian Artigues

2007-2008 : Encadrement (50%) du stage de Master Recherche 2ème année SAID de **Abdellah Sadki** (UPS), "Minimisation du nombre de travaux en retard sous contraintes de fenêtres d'exécution", co-encadrant : Samia Ourari

2005-2006 : Encadrement (100%) du stage de Master Recherche 2ème année SAID de **Xiaoshu Li** (UPS), "Insertion d'activités en ordonnancement de projet sous contraintes de précédence généralisées"

2004-2005 : Encadrement (100%) du stage de Master Recherche 2ème année SAID de **Damien Poinsot** (UPS), "Un nouveau schéma de génération de solutions pour l'ordonnancement de projet avec contraintes de précédence généralisées et ressources cumulatives"

2003-2004 : Encadrement (100%) du stage de Master Recherche 2ème année SAID de **Jean-Luc Santamaria** (École Nationale Supérieure des Ingénieurs en Arts Chimiques et Technologiques) intitulé "Aide à la décision pour l'ordonnancement robuste"

2000-2001 : Co-encadrement (60%)du stage de Master Recherche 2ème année SAID de **Emmanuelle Despontin** (UPS), "Ordonnancement multi-objectifs", co-encadrant : François Roubellat

1999-2000 : Encadrement (100%) du stage de Master Recherche 2ème année SAID de **Anas Abou El Kalam** (UPS), "Ordonnancement avec gammes alternatives"

1998-1999 : Co-encadrement (70%) du stage de Master Recherche 2ème année SAID de **Anis Ziadi** (UPS), "Représentation et calcul d'ensembles de solutions pour un problème d'ordonnancement : utilisation des réseaux de Petri" - Co-encadrant : Pierre Lopez

1995-1996 : Co-encadrement (50%) du stage de Master Recherche 2ème année SAID de **Fernando Harold Lopez** (UPS), "Analyse comparée de Méthodes de Modélisation pour la Commande Locale de systèmes de production flexibles" - Co-encadrant : Philippe Esteban

3.3 Mémoires, projets

Stage de recherche

Été 2005 : Encadrement (100%) du stage Ingénieur de Master 1ère année de **Jérome Bézanger** (ENSEEIHT) " Développement d'une méthode de recherche par voisinage pour l'ordonnancement de projet avec contraintes de ressources"

Été 2004 : Encadrement (50%) du stage Ingénieur de Master 1ère année de **Cédric Mocquillon** (Polytech'Tours) " Réalisation d'une interface graphique pour une aide à la coopération interentreprise " - Co-encadrante : Emmanuelle Despontin-Monsarrat

Printemps 2004 : Encadrement (50%) du stage Ingénieur de Master 2ème année de **Andreea Ardelean** (Polytechnica Bucarest - Roumanie) "Proposition de schémas XML pour la modélisation de problèmes d'ordonnancement " - Co-encadrant : Patrick Esquirol

Chapitre 4

Diffusion et transfert de connaissances

Ce chapitre recense l'ensemble de mes communications classées par catégories. Sont également précisées mes implications dans l'animation de la recherche et mes collaborations industrielles.

4.1 Articles publiés dans des revues à comité de lecture

1. C. BRIAND, S. OURARI, B. BOUZOUIA , " An efficient ILP formulation for the single machine scheduling problem " - *RAIRO - Operations Research, à paraître en 2010.*
2. C. ARTIGUES, C. BRIAND , " The resource-constrained activity insertion problem with minimum and maximum time lags " - *Journal of Scheduling*, Vol. 12, No 5, pp 447-460, 2009.
3. C. BRIAND , " A new Any-Order schedule generation scheme for resource-constrained project scheduling " - *RAIRO - Operations Research*, Vol. 43, No 3, pp 297-308, 2009.
4. C. BRIAND, H.T. LA, J. ERSCHLER , " A robust approach for the single machine scheduling problem " - *Journal of Scheduling*, special issue " Project Scheduling under Uncertainty " - Vol. 10, No 3, pp.209-221, 2007.
5. C. BRIAND, H.T. LA, J. ERSCHLER " A new sufficient condition of optimality for the two-machine flowshop problem " - *European Journal of Operational Research*, 169 pp 712-722, 2006.
6. E. DESPONTIN, C.BRIAND, P. ESQUIROL " Aide à la décision pour une coopération interentreprise " - *Journal Européen des Systèmes Automatisés*, 39, pp 797-816, 2005.

7. E. DESPONTIN, C.BRIAND, P. ESQUIROL " Aide à la coopération inter-entreprises pour la production à la commande " - *Document Numérique*, Vol. 8, pp. 23-36, 2004.
8. C.BRIAND, " Using Petri nets for constraint satisfaction ", *Int. Journal of Decision Systems*, Vol.9, n° 1, pp.77-97, 2000.

4.2 Article en cours de révision

1. S. OURARI, C. BRIAND, B. BOUZOUIA, "A mathematical programming approach to minimize the number of late jobs for the single machine scheduling problem", *in revision for the Journal of Mathematical Modelling and Algorithms*, mai 2009.

4.3 Contributions à des ouvrages collectifs

1. C. BRIAND, M.-J HUGUET, H.-T LA , P. LOPEZ, " Constraint-based approaches for robust scheduling ", Ed. ISTE Wiley , Control Systems, Robotics and Manufacturing Series, N° ISBN 978-1-84821-054-7, pp. 199-226.
2. C. ARTIGUES, C. BRIAND, , "Activity insertion problem in a RCPSP with minimum and maximum time lags ", in " Resource-constrained project scheduling. Models, algorithms, extensions and applications ", ISTE/Wiley, Eds. C.Artigues, S.Demassey, E.Neron, NřISBN 978-1-84821-034-9, Mai 2008, Chapitre 11, pp.171-190.
3. C. BRIAND, M.-J HUGUET, H.-T LA , P. LOPEZ, " Approches par contraintes pour l'ordonnancement robuste ", dans " Flexibilité et Robustesse en Ordonnancement ", Ed. Hermes Science , Traité IC2 Information-Commande-Communication, N° ISBN 2-7462-1028-2, 2005, pp. 191-215.
4. C. ARTIGUES, C. BRIAND, M.-C PORTMANN, F. ROUBELLAT, " Pilotage d'atelier basé sur un ordonnancement flexible ", Méthodes du pilotage des systèmes de production, Ed. Hermes Science, Traité IC2 Information-Commande-Communication, N° ISBN 2-7462-0514-9, 2002, pp.61-97.

4.4 Conférences internationales avec actes et comités de lecture

1. S. OURARI, C. BRIAND, B. BOUZOUIA, "Minimizing the number of tardy jobs in single machine scheduling using MIP", 4th Multidisciplinary International Scheduling

Conference : Theory and Applications, Dublin (Ireland), 10-12 août 2009.

2. C. BRIAND, S. OURARI, B. BOUZOUIA, "A cooperative approach for job shop scheduling under uncertainties", International Conference on Collaborative Decision Making (CDM'08), Toulouse (France), 1-4 juillet 2008, p5-15.
3. C. ARTIGUES, C. BRIAND, "Complexity of activity insertion for resource-constrained project scheduling problem with minimum and maximum time lags", 8th Workshop on Models and Algorithms for Planning and Scheduling Problems (MAPSP'2007),Istanbul (Turquie), 2-6 juillet 2007.
4. C. BRIAND, S. OURARI, B. BOUZOUIA "A new dominance condition for mono-pyramidal single machine scheduling problems", 8th Workshop on Models and Algorithms for Planning and Scheduling Problems (MAPSP'2007),Istanbul (Turquie), 2-6 juillet 2007.
5. C. BRIAND, S. OURARI, "Minimizing the number of late jobs in single machine scheduling with nested execution intervals", IEEE International Conference on Services Systems and Services Management (ICSSSM06), University of Technology of Troyes (FRANCE), 25-27 October 2006, , p.1172-1177 .
6. C. BRIAND, J. BEZANGER "An any-order SGS for project scheduling with scarce resources and precedence constraints", 10th International Workshop on Project Management and Scheduling (PMS'2006), Poznan (Pologne), 26-28 April 2006, pp.95-100.
7. E. MONSARRAT-DESPONTIN, C.BRIAND, P ESQUIROL "A decision support framework for cooperation in make-to-order production" - 6th International Conference on the Design of Cooperative Systems (COOP'04), may 11-14 2004, pp 69-82, French-Riviera, France.
8. C. BRIAND, H.T. LA, J. ERSCHLER "Robust scheduling for the two-machine flowshop problem " , 9th International Workshop on Project Management and Scheduling (PMS'04), April 26-28 2004, pp 371-374, Nancy, France
9. C. BRIAND, E. DESPONTIN, F. ROUBELLAT, " Scheduling with time lags and preferences : a heuristic ", 8th International Workshop on Project Management and Scheduling, Valence (Espagne), April 3-5 2002, pp.77-80.
10. C.BRIAND , P.ESQUIROL, " Task charts : a time model for the specification of alternative processes " International Conference on Industrial Engineering and Production Management (IEPM'2001), Quebec, August 20-23 2001, 10p.
11. C.BRIAND, " Solving the car-sequencing problem using Petri nets " International Conference on Industrial Engineering and Production Management (IEPM'99), Glasgow (GB), July 12-15 1999, Vol.1, pp.543-551.
12. C.BRIAND , P.ESTEBAN, " An object-oriented and a Petri net based approach for real time control of flexible manufacturing systems ", 4th INRIA/IEEE Symposium on Emerging Technologies and Factory Automation (ETFA'95), Paris (France), October 10-13 1995, pp.115-123.

13. C.BRIAND , D.ANDREU, " HOOD/PNO design of a batch process control system " ,Workshop on Object-Oriented Programming and Models of Concurrency, Turin (Italie), 27 Juin 1995.
14. C.BRIAND , P.ESTEBAN , M.PALUDETTO, " Flexible manufacturing system design using object and Petri net concepts " 1st IFAC Workshop New Trends in Design of Control Systems, Smolenice (Slovaquie), September 7-10 1994, pp.362-367.
15. P.ESTEBAN , C.BRIAND , M.COURVOISIER, " Integration of the real time levels of a multilevel hierarchical control : application to a flexible assembly cell ", IEEE International Symposium on Industrial Electronics (ISIE'93), Budapest (Hongrie), June 1-3 1993, pp.407-412.

4.5 Conférences Francophones avec actes et comités de lecture

1. C. BRIAND, S. OURARI, "Une formulation PLNE efficace pour $1|r_j|L_{\max}$", 10ème Congrès de la Société Française de Recherche Opérationnelle (ROADEF'2009), Nancy (France), 10-12 février 2009, pp 106-107.
2. C. BRIAND, S. OURARI, B. BOUZOUIA, "Une approche coopérative pour l'ordonnancement sous incertitudes", 7ème Conférence Internationale de Modélisation et Simulation (MOSIM'08), Paris (France), 31 Mars - 2 Avril 2008, 8p.
3. S. OURARI, C. BRIAND, "Conditions de dominance pour le problème à une machine avec minimisation des travaux en retard ", 9ème Congrès de la Société Française de Recherche Opérationnelle (ROADEF'2009), Nancy (France), 25-27 février 2008, pp 351-352.
4. S. OURARI, C. BRIAND, B. BOUZOUIA, "Vers une approche distribuée pour l'ordonnancement réactif sous incertitudes", Colloque sur l'Optimisation et les Systèmes d'Information (COSI'2007), Oran (Algérie), Juin 2007, pp.449-460.
5. S. OURARI, C. BRIAND, B. BOUZOUIA "Un concept de dominance basé sur un théorème des pyramides pour l'ordonnancement sur une machine avec minimisation des travaux en retard", 7ème congrès International de Génie Industriel, trois-Rivières (Québec - Canada), 5-8 Juin 2007.
6. S. OURARI, C. BRIAND, B. BOUZOUIA "Une condition de dominance pour l'ordonnancement à une machine avec minimisation du nombre de travaux en retard", Colloque sur l'Optimisation et les Systèmes d'Information (COSI'2006), Alger (Algérie), 11-13 Juin 2006, pp.268-278.
7. C. BRIAND, J. BEZANGER "Un nouveau schéma de génération pour l'ordonnancement de projet sous contraintes de ressources", 7ème Congrès de la

Société Française de Recherche Opérationnelle et d'Aide à la Décision (ROADEF'06), Lille (France), 6-8 Février 2006.

8. H.T. LA, J.-L. SANTAMARIA, C. BRIAND "Une aide à la décision pour l'ordonnancement robuste en contexte mono-ressource : un compromis flexibilité/performance", 6ème Congrès de la Société Française de Recherche Opérationnelle et d'Aide à la Décision (ROADEF'05), Tours (France), 14-16 Février 2005, pp 101-116
9. E. DESPONTIN, C. BRIAND, P. ESQUIROL " Aide à la coopération inter-entreprises pour la production à la commande " - Colloque Coopération, Innovation et Technologies, Troyes, 3-4 Décembre 2003, pp.21-37.
10. C. BRIAND, H.T. LA, J. ERSCHLER " Une approche pour l'ordonnancement robuste de tâches sur une machine ", 4ème Conférence Francophone de Modélisation et Simulation, Toulouse, 23-25 Avril 2003, pp.205 211.
11. E. DESPONTIN, C. BRIAND, " Ordonnancement avec prise en compte de multiples objectifs temporels : une heuristique couplée à une méthode d'exploration de voisinage ", 4ème Conférence Francophone de MOdélisation et SIMulation, Toulouse, 23-25 Avril 2003, pp.423-429.
12. C. BRIAND, H.T. LA, J. ERSCHLER " Ordonnancement de problèmes à une machine : une aide à la décision pour un compromis flexibilité vs performance " - 5ème Congrès de la Société Française de Recherche Opérationnelle et d'Aide à la Décision, Avignon, 26-28 Février 2003, pp.146-147.
13. C. BRIAND, H.T. LA, " Une procédure par séparation et évaluation progressive pour l'ordonnancement robuste de problème à une machine ", 1ère Conférence en Recherche Informatique Vietnam et Francophone, RIVF'03, Hanoi, Vietnam, 10-13 février 2003, pp 11-16.
14. C.BRIAND , F.ROUBELLAT, " Pilotage réactif et production à la commande ", 3ème Conférence Francophone de MOdélisation et SIMulation (MOSIM'01), Troyes (France), 25-27 Avril 2001, Vol.1, pp.371-377.
15. C.BRIAND, " Vers une plus grande flexibilité du pilotage des systèmes de production " 2ème Congrès sur la Modélisation des Systèmes Réactifs (MSR'99), Cachan (France), 24-26 Mars 1999, Modélisation des Systèmes Réactifs, Hermes, 1999, ISBN N° 2-7462-0017-1, pp.277-286.

4.6 Rapports internes

1. J.BEZANGER, C.BRIAND, "Développement d'une méthode de recherche par voisinage pour l'ordonnancement de projet avec contraintes de ressources", Rapport LAAS N° 05574, Septembre 2005, 16p.

2. D. POINSOT, C. BRIAND, "Un nouveau schéma de génération de solution pour l'ordonnancement de projet avec contraintes de précédence généralisées et ressources cumulatives", Rapport LAAS N° 05572, Mai 2005, 45p.
3. C. BRIAND , J.E. DOUCET , P. ESQUIROL , M.J. HUGUET , P. LOPEZ," Projet de plate-forme logicielle LORA. Spécification 1.1. Représentation de problèmes d'ordonnancement de tâches et d'affectation de ressources", Rapport LAAS N° 01551, Décembre 2001, 24p.
4. C. BRIAND, " The car scheduling problem : a Petri net approach ", Rapport interne LAAS N° 98172, May 1998, 6p.
5. D.ANDREU, C.BRIAND, M.COMBACAU, P.ESTEBAN, J.C.PASCAL, "Modélisation et mise en œuvre de la commande temps réel hiérarchisée d'une maquette d'atelier flexible", Rapport LAAS No95151, Automatique et Expérimentation en Laboratoire, Angers (France), 4-5 Avril 1995, pp.12.1-12.7
6. C.BRIAND, "Conception orientée-objet et basée réseaux de Petri de la conduite temps réel des systèmes flexibles de production manufacturière", Rapport LAAS No95082, Doctorat, Université Paul Sabatier, Toulouse, 12 Janvier 1995, N° 1925, 175p.

4.7 Participation à la vie scientifique

Actions Ponctuelles

2010 : Membre du Comité d'Organisation du 11ème congrès de la Société Française de Recherche Opérationnelle (www.roadef2010.fr) - 24-26.02.2010

Rôle : Participation à l'organisation de la conférence

Tâches : Recherche de sponsor - Organisation - Reviews

ROADEF 2010 est le onzième congrès de la Société Française de Recherche Opérationnelle et d'Aide à la Décision. Il est organisé par le LAAS-CNRS, l'Institut de Mathématiques de Toulouse et l'IRIT, avec la participation des établissements du PRES Université de Toulouse. Il aura lieu du 24 au 26 février 2010 sur le site "Manufacture des Tabacs" de l'Université de Toulouse (UT1-Capitole). Dans ce cadre je participe aux réunions d'organisation, à la recherche de sponsors et à l'évaluation des résumés.

2009-2011 : Coordinateur pour le LAAS du projet ANR BLANC ROBOCOOP "Robustesse et Coopération en Ordonnancement "

Cadre : Projet financé par l'ANR

Partenaires : LI Tours (J.-C Billaut), LAMSADE Paris (M. A. Aloulou), LITIS Le Havre (E. Sanlaville), ILOG-IBM PAris (P. Laborie).

Le projet ANR Blanc ROBOCOOP s'inscrit dans la suite logique du projet AVAPRO du GDR RO, décrit ci-dessous. Je suis coordinateur du projet pour le LAAS, les autres membres concernés au LAAS étant C. Artigues, P. Lopez et S. Ourari. J'ai participé à l'élaboration du projet et aux revues de lancement ayant eu lieu en mars 2009 à Tours et en juin 2009 à Toulouse (4 revues de projet par an). Un financement de doctorat est prévu dans ce projet qui va me conduire à encadrer, à partir de septembre 2009, un nouveau doctorant devant travailler sur les thèmes de ROBOCOOP, plus précisément, sur la coopération en ordonnancement et les apports de techniques robustes dans ce cadre. Les partenaires extérieur seront impliqués dnas ce travail et ont également en charge la partie du projet consacrée à la définition d'instances dynamiques de problèmes ordonnancement et à la définition d'indicateurs de robustesse.

2006-2007 : Membre du projet AVAPRO " Analyse et Vérification des Approches Proactives/Réactives en Ordonnancement : prise en compte des critères de qualité, robustesse et flexibilité"

Cadre : Projet financé par le GDR RO (http ://www-poleia.lip6.fr/ sourd/gdrro/)

Partenaires : LI Tours, LAMSADE, LIMOS, ILOG

Ce projet du GDR RO, amorcé en 2006 et renouvellé en 2007, avait pour but de définir la nature des outils à mettre en place pour permettre d'évaluer des approches robustes d'ordonnancement selon divers indicateurs propres à la robustesse. Au cours des réunions bi-annuelles que nous avons eu, la notion d'instance dynamique de problème a été proposé, où les paramètres du problème à résoudre évoluent dans le temps (selon un scénario propre à l'instance), et des principes quant à la façon de la construire ont été dégagés. Des outils ont également été partiellement spécifiés permettant de tester la performance de méthodes d'ordonnancement robuste pour la résolution d' instances dynamiques : il s'agit de jouer le déroulement d'une instance dynamique en temps accéléré, puis de récupérer la trace des décisions prises par la méthode, pour enfin les analyser a posteriori selon les indicateurs retenus. Cet avant-projet a conduit en 2007 à la rédaction d'une proposition d'ANR Blanc baptisé ROBUSTA, refusé, puis en 2008, à celle de l'ANR Blanc ROBOCOOP, finalement accepté.

2006-2008 : Responsable côté français de la coopération franco-algérienne CNRS-DPGRF entre le LAAS de Toulouse et le CDTA d'Alger

Cadre : Projet financé par le CNRS

Thème : Ordonnancement distribué et réactif sous incertitudes

Partenaires : Equipe "Systèmes robotisés de production" du CDTA d'Alger

Ce projet avait pour but de permettre l'avancée des travaux de recherche de Samia Ourari et, plus particulièrement, de financer la venue au LAAS Samia 1 mois par an, celle-ci étant employée en tant que Chargée de Recherche au CDTA d'Alger et développant sa recherche en autonomie dans ce laboratoire. Je mes suis également rendu à deux reprises (2006 et 2008) au CDTA pour discuter des orientations de recherche et présenter deux séminaires de recherche (le premier sur le thème de l'ordonnancement robuste, le second sur celui de l'ordonnancement distribué).

2005 : Co-organisateur des Journées "Gestion de Projet pour la conception de Systèmes (GPS)" LAAS-CNRS, Toulouse, 31 mai-1er juin 2005 (http ://www.laas.fr/ esquirol/gps/) - 50 participants

Cadre : Pré-projet interne LAAS-CNRS

Tâches : Appel à communication, Gestion des communications, Logistique

Ces journées ont été organisées dans le cadre d'un pré-projet, baptisé GPS, soumis à la direction du LAAS en mai 2004. Le projet initial prévoyait le financement d'un travail de thèse sur 3 ans ayant pour thème la gestion de projet intégrée dans un contexte de conception distribuée et d'ingénierie concourante. La limitation du projet à une pré-étude de 1 an a permis de procéder d'abord à un état de l'art dans ce domaine et en particulier d'organiser ces journées, afin de faire l'inventaire des équipes françaises et éventuellement internationales travaillant sur ce domaine et d'évaluer l'intérêt de notre sujet de recherche initial par rapport aux recherches en cours. Trois équipes ont participé à l'organisation de ces journées : MOGISA (LAAS-CNRS) (avec Patrick Esquirol et Cyril Briand), MIS (LAAS-CNRS) (avec Daniel Estève) et le LESIA (INSA) (avec Claude Baron). Les deux journées ont été organisées sous forme de présentations et de tables rondes autour des trois thèmes suivants : Conception et conduite de projet : de l'analyse du système à l'architecture du projet et à son pilotage, Ingénierie Simultanée : gestion des alternatives et prise en compte de la maturité dans l'analyse du risque, Conduite de projet distribuée : modèles et outils pour la coopération.

2003 - 2004 : Participation à la collaboration scientifique entre le groupe MOGISA du LAAS et le groupe " Ordonnancement et Conduite " du laboratoire d'Informatique de Tours

Cadre : Projet au sein de l'Action Spécifique "Recherche opérationnelle" du département STIC

Thème : Création d'un site Web consacré à l'ordonnancement (modélisation et résolution)- Projet e-OCEA (voir : www.ocea.li.univ-tours.fr)

Tâches : Recherche - Exposés - Compte rendus

Ce projet a eu pour but de mettre en commun les expertises du LI de Tours et du LAAS-CNRS quant à la représentation de problèmes d'ordonnancement et à leur résolution. J'ai eu l'occasion de présenter au LI l'expérience acquise durant le déroulement du projet LORA (décrit ci-dessous). Le but était de mener une réflexion sur les outils logiciels à mettre à disposition de la communauté afin de simplifier la description de problèmes d'ordonnancement, leur résolution et l'évaluation des méthodes de résolution.

2000 - 2004 : Participation aux projets LORA et GEMO (www.laas.fr/MOGISA/mogisa-lo.html)

Cadre : Projet interne LAAS-CNRS - Soutien d'un ingénieur de Recherche

Thème : Création d'une Plateforme logicielle pour l'ordonnancement

Rôle : Participation à la spécification et à la conception de la plateforme

Tâches : Exposés - Rédaction de bilans d'avancement -Encadrements de stagiaires

Le projet LORA s'intéresse à la représentation de processus et de ressources soujacents à la description de problème d'ordonnancement et de planification d'activités. Il a donné lieu à la rédaction d'un document de spécification d'un outil logiciel permettant de décrire des instances de problème très diverses et à la réalisation d'un démonstrateur logiciel. Cet outil devait constituer une brique de base d'une plateforme logicielle permettant de décrire une instance de problème, de la résoudre avec une ou plusieurs méthodes disponibles, puis de comparer les performances des méthodes selon divers critères. L'outil devant s'adapter à la grande variété des problèmes traités dans l'équipe (ordonnancement de production et de projet, planification, ressources cumulatives à disponibilité variant dans le temps, prise en compte de consommation énergétiques, notions d'états ajoutés aux ressources, etc.) et permettre de comparer les méthodes selon des critères également très variés, ce projet s'est rapidement avéré très ambitieux et nous ne sommes malheureusement pas arrivé à réaliser les objectifs.

2003 : Membre du Comité d'Organisation de la 4ème conférence Francophone de Modélisation et Simulation (MOSIM'03) - 23-25.04.2003 (www.laas.fr/mosim03)

Rôle : Participation à l'organisation de la conférence

Tâches : Rédaction de dossiers de subvention - Gestion des communications - Reviews

La 4ème conférence Francophone de Modélisation et Simulation (MOSIM'03) s'est tenue à l'INSA de Toulouse du 23 au 25 avril 2003. J'ai participé aux diverses réunions du comité d'organisation et j'ai plus spécifiquement contribué à la rédaction de dossiers de subvention à l'attention de l'UPS, à la gestion du processus d'examen des communications et à l'arbitrages des articles.

Actions permanentes

- Participations aux groupes de travail nationaux proches de l'ordonnancement : "Groupe de recherche en Ordonnancement Théorique et Appliqué" (http ://www-poleia.lip6.fr/ sourd/gotha/) du GDR RO, groupes "Bermudes" (http ://bermudes.univ-bpclermont.fr/) et "Organisation et Gestion de Production" du GDR MACS.
- Reviews : European Journal of Operational Research, International Journal of Advanced Manufacturing Technology, Journal Européen des Systèmes Automatisés, Congrés.
- Séminaires scientifiques internes

4.8 Relations industrielles

2007 : Projet Flight Training timetabling : Collaboration scientifique entre le groupe MOGISA du LAAS et AIRBUS ST Training

Thème : Expertise sur la construction de planning de formation pour pilotes

Rôle : Participant au projet (avec C. Artigues et P. Lopez).

Ce projet s'est déroulé en 2007 en relation avec le département AIRBUS ST Training. Dans le cadre de la refonte de leur outil GIPSI chargé de la planification opérationnelle et de l'ordonnancement des formations de pilote, nous avons été chargés de réaliser une expertise scientifique du problème de planification et d'ordonnancement de formations. Ce document avait pour but d'aider notre partenaire à mieux spécifier les besoins que leur nouvel outil de planification devait satisfaire. Nous avons produit une définition formelle du problème, un examen de sa complexité et une revue des méthodes et outils existants pour sa résolution. Dans ce projet, j'ai contribué à l'analyse d'articles en liens avec l'ordonnancement de formation de pilotes et à la rédaction d'un document de synthèse d'une cinquantaine de pages.

2006 : Projet MAINSCHED : Collaboration scientifique entre le groupe MOGISA du LAAS et la société Airbus France

Thème : Spécification et conception d'un logiciel d'aide à l'ordonnancement pour les tâches de maintenance programmée

Rôle : Participant au projet (avec P. Lopez et M. Mongeau)

Nous avons été sollicité en 2006 par Airbus France dans le cadre d'une projet de spécification et de conception d'un logiciel d'aide à l'ordonnancement pour les tâches de maintenance programmée. Plusieurs réunions ont eu lieu à Airbus et un projet de Thèse a été défini. Malheureusement, suite à la restructuration interne des services chez Airbus, ce projet n'a jamais vu le jour.

2000 - 2004 : Collaboration scientifique entre le LAAS et la société Ordosoftware de Besançon

Thème : Spécification et conception d'un logiciel pour l'Ordonnancement coopératif d'atelier

Rôle : Responsable scientifique du projet - Encadrement d'une thèse CIFRE

Dans le cadre de la thèse CIFRE d'Emmanuelle Despontin-Monsarrat, j'ai assuré le rôle de responsable scientifique du projet dans la convention de recherche passée avec la société OrdoSoftware. Le LAAS-CNRS avait plusieurs fois collaboré avec cette société dans le passé, sous la responsabilité scientifique de F. Roubellat. Cette responsabilité a été difficile car, outre les revues de projet trimestrielles, j'ai du faire face durant la période couverte par la convention à la revente de la société Ordosoftware à la société EBC Informatique puis, suite à un dépôt de bilan de cette dernière société, au rachat de l'activité liée à l'ordonnancement par la société Schneider Electric. J'ai du donc défendre à plusieurs reprises l'intérêt des recherches entreprises dans le cadre de la thèse, valoriser leur impact quant aux offres logicielles futures et m'occuper de gérer avec la délégation régionale du CNRS les évolutions de contrat.

4.9 Rapports de contrats

1. C. ARTIGUES , C. BRIAND , P. LOPEZ, "Flight training timetabling", Rapport LAAS No08055, Contrat AIRBUS ST Training, Janvier 2008, 46p.
2. C.MOCQUILLON, C.BRIAND, E.MONSARRAT-DESPONTIN, P.ESQUIROL, "Réalisation d'une interface graphique pour un module d'aide à la coopération inter-entreprise pour la production à la commande", Rapport LAAS No04482, Contrat ORDOSOFTWARE N° 1412117.00, Septembre 2004, 51p.
3. C.BRIAND, E.MONSARRAT-DESPONTIN, P.ESQUIROL, "Un outil logiciel pour la coopération : analyse des besoins fonctionnels", Rapport LAAS No04009, Contrat ORDOSOFTWARE N° 1412117.00, Janvier 2004, 34p.
4. C.BRIAND, E.MONSARRAT-DESPONTIN, P.ESQUIROL, "Principe d'une aide à la coopération inter-entreprise pour la production à la commande", Rapport LAAS No03370, Contrat ORDOSOFTWARE N° 1412117.00, Août 2003, 14p.
5. C.BRIAND, E.DESPONTIN, P.ESQUIROL, "Premiers pas pour une aide à la coopération inter-entreprise", Rapport LAAS N° 02593, Contrat ORDOSOFTWARE N° 1412117.00, Décembre 2002, 12p.
6. C.BRIAND, E.MONSARRAT-DESPONTIN, P.ESQUIROL, "Aide à la décision pour l'ordonnancement et le pilotage coopératifs d'atelier. Rapport d'étape", Rapport LAAS N° 02248, Contrat EBC Informatique N° 1412117.00, Juin 2002, 13p.

7. C.BRIAND , E.MONSARRAT-DESPONTIN, "Prise en compte de multiples objectifs temporels dans le module analyse d'ORABAID", Rapport LAAS No02244, Contrat EBC Informatique N° 1412117.00, Mars 2002, 18p.
8. C.BRIAND, E.MONSARRAT-DESPONTIN, "Ordonnancement avec prise en compte d'objectifs temporels multiples", Rapport LAAS No01552, Contrat OrdoSoftware N° 1412117.00, Juillet 2001

Chapitre 5

Exemplaire des 7 publications présentées

En cohérence avec la chronologie des thèmes développés dans la synthèse, ce chapitre inclut les 7 communications suivantes :

- C. BRIAND, S. OURARI, B. BOUZOUIA , " An efficient ILP formulation for the single machine scheduling problem " - *RAIRO - Operations Research, à paraître en 2010.*
- S. OURARI, C. BRIAND, B. BOUZOUIA, "A mathematical programming approach to minimize the number of late jobs for the single machine scheduling problem", *in revision for the Journal of Mathematical Modelling and Algorithms*, mai 2009.
- C. BRIAND, H.T. LA, J. ERSCHLER " A new sufficient condition of optimality for the two-machine flowshop problem " - *European Journal of Operational Research*, 169 pp 712-722, 2006.
- C. BRIAND, H.T. LA, J. ERSCHLER , " A robust approach for the single machine scheduling problem " - *Journal of Scheduling*, special issue " Project Scheduling under Uncertainty " - Vol.10, No3, pp.209-221, Juin 2007.
- C. BRIAND , " A new Any-Order schedule generation scheme for resource-constrained project scheduling " - *RAIRO - Operations Research*, Vol. 43, No. 3, pp 297-308, 2009.

C. ARTIGUES, C. BRIAND , " The resource-constrained activity insertion problem with minimum and maximum time lags " - *Journal of Scheduling*, Vol. 12, No 5, pp 447-460, 2009.

- E. DESPONTIN, C.BRIAND, P. ESQUIROL " Aide à la décision pour une coopération interentreprise " - *Journal Européen des Systèmes Automatisés*, 39, pp 797-816, 2005.