Aspects parallèles des problèmes de satisfaisabilité

Pascal Vander-Swalmen

Thèse réalisée au sein des laboratoires CReSTIC et MIS
sous la direction de :

- Michaël Krajecki (CReSTIC)
  Université de Reims Champagne-Ardenne
- Gilles Dequen (MIS)
  Université de Picardie Jules Verne

Lundi 7 décembre 2009

Plan

Objectifs

- Concevoir un solveur SAT intrinsèquement parallèle
- Préserver les progrès réalisés pour la résolution séquentielle du problème SAT
- Applications nombreuses
  - « SAT has become a mature multi-faceted scientific discipline » [Biere, Heule, Maaren, Walsh - 2009]
- Spécialisation dans les architectures à mémoire partagée

Plan

Variables

Soit $\mathcal{X} = \{x_1, \ldots, x_n\}$, un ensemble de n variables Booléennes

Variables Booléennes

- Une variable Booléenne $x \in \mathcal{X}$ est définie sur $\{\text{VRAI}, \text{FAUX}\}$
- Le littéral x exprime la même valeur de vérité que la variable x
- Le littéral $\bar{x}$ exprime la valeur de vérité opposée à celle de x
- Littéral x satisfait ssi variable x vaut VRAI (littéral $\bar{x}$ contredit)
- Littéral $\bar{x}$ satisfait ssi variable x vaut FAUX (littéral x contredit)

Variables

Soit $\mathcal{X} = \{x_1, \ldots, x_n\}$, un ensemble de n variables Booléennes

Variables Booléennes

- Une variable Booléenne $x \in \mathcal{X}$ est définie sur $\{\text{VRAI}, \text{FAUX}\}$
- Le littéral x exprime la même valeur de vérité que la variable x
- Le littéral $\bar{x}$ exprime la valeur de vérité opposée à celle de x
- Littéral x satisfait ssi variable x vaut VRAI (littéral $\bar{x}$ contredit)
- Littéral $\bar{x}$ satisfait ssi variable x vaut FAUX (littéral x contredit)

Variables

Soit $\mathcal{X} = \{x_1, \ldots, x_n\}$, un ensemble de n variables Booléennes

Variables Booléennes

- Une variable Booléenne $x \in \mathcal{X}$ est définie sur $\{\text{VRAI}, \text{FAUX}\}$
- Le littéral x exprime la même valeur de vérité que la variable x
- Le littéral $\bar{x}$ exprime la valeur de vérité opposée à celle de x
- Littéral x satisfait ssi variable x vaut VRAI (littéral $\bar{x}$ contredit)
- Littéral $\bar{x}$ satisfait ssi variable x vaut FAUX (littéral x contredit)

Variables

Soit $\mathcal{X} = \{x_1, \ldots, x_n\}$, un ensemble de n variables Booléennes

Variables Booléennes

- Une variable Booléenne $x \in \mathcal{X}$ est définie sur $\{\text{VRAI}, \text{FAUX}\}$
- Le littéral x exprime la même valeur de vérité que la variable x
- Le littéral $\bar{x}$ exprime la valeur de vérité opposée à celle de x
- Littéral x satisfait ssi variable x vaut VRAI (littéral $\bar{x}$ contredit)
- Littéral $\bar{x}$ satisfait ssi variable x vaut FAUX (littéral x contredit)

Contraintes

Soit $\mathcal{C} = \{c_1, \ldots, c_m\}$, un ensemble de m clauses

Clauses

- Une clause $c \in \mathcal{C}$ est une disjonction de littéraux (OU logique)
- $c = l_1 \vee \ldots \vee l_j$ est satisfaite ssi $\exists i \in \{1, j\} / l_i$ est satisfait
- $c = l_1 \vee \ldots \vee l_j$ est contredite ssi $\forall i \in \{1, j\}, l_i$ est contredit
  $\Rightarrow$ **conflit**

Exemple de clause

$x_2 \vee \bar{x_5} \vee x_4$

Contraintes

Soit $\mathcal{C} = \{c_1, \ldots, c_m\}$, un ensemble de m clauses

Clauses

- Une clause $c \in \mathcal{C}$ est une disjonction de littéraux (OU logique)
- $c = l_1 \vee \ldots \vee l_j$ est satisfaite ssi $\exists i \in \{1, j\} / l_i$ est satisfait
- $c = l_1 \vee \ldots \vee l_j$ est contredite ssi $\forall i \in \{1, j\}, l_i$ est contredit
  $\Rightarrow$ **conflit**

Exemple de clause

$x_2 \vee \bar{x_5} \vee x_4$

Contraintes

Soit $\mathcal{C} = \{c_1, \ldots, c_m\}$, un ensemble de m clauses

Clauses

- Une clause $c \in \mathcal{C}$ est une disjonction de littéraux (OU logique)
- $c = l_1 \vee \ldots \vee l_j$ est satisfaite ssi $\exists i \in \{1, j\} / l_i$ est satisfait
- $c = l_1 \vee \ldots \vee l_j$ est contredite ssi $\forall i \in \{1, j\}, l_i$ est contredit
  $\Rightarrow$ **conflit**

Exemple de clause

$x_2 \vee \bar{x_5} \vee x_4$

Contraintes

Soit $\mathcal{C} = \{c_1, \ldots, c_m\}$, un ensemble de m clauses

Clauses

- Une clause $c \in \mathcal{C}$ est une disjonction de littéraux (OU logique)
- $c = l_1 \vee \ldots \vee l_j$ est satisfaite ssi $\exists i \in \{1, j\} / l_i$ est satisfait
- $c = l_1 \vee \ldots \vee l_j$ est contredite ssi $\forall i \in \{1, j\}, l_i$ est contredit
  $\Rightarrow$ **conflit**

Exemple de clause

$x_2 \vee \bar{x_5} \vee x_4$

Formule logique

Formule sous CNF

- Une formule sous CNF $\mathcal{F}$ est une conjonction de clauses (ET logique)
- $\mathcal{F} = c_1 \wedge \ldots \wedge c_m$ est satisfaite ssi $\forall i \in \{1, m\}, c_i$ est satisfaite
- $\mathcal{F} = c_1 \wedge \ldots \wedge c_m$ est contredite ssi $\exists i \in \{1, m\} / c_i$ est contredite

Exemple de formule sous CNF

$(x_2 \vee \bar{x_5} \vee x_4) \wedge (x_1 \vee x_3) \wedge (\bar{x_4} \vee \bar{x_2} \vee \bar{x_1} \vee x_5)$
Cette formule contient 3 clauses et 5 variables différentes

Formule logique

Formule sous CNF

- Une formule sous CNF $\mathcal{F}$ est une conjonction de clauses (ET logique)
- $\mathcal{F} = c_1 \wedge \ldots \wedge c_m$ est satisfaite ssi $\forall i \in \{1, m\}, c_i$ est satisfaite
- $\mathcal{F} = c_1 \wedge \ldots \wedge c_m$ est contredite ssi $\exists i \in \{1, m\} / c_i$ est contredite

Exemple de formule sous CNF

$(x_2 \vee \bar{x_5} \vee x_4) \wedge (x_1 \vee x_3) \wedge (\bar{x_4} \vee \bar{x_2} \vee \bar{x_1} \vee x_5)$

Cette formule contient 3 clauses et 5 variables différentes

Formule logique

Formule sous CNF

- Une formule sous CNF $\mathcal{F}$ est une conjonction de clauses (ET logique)
- $\mathcal{F} = c_1 \wedge \ldots \wedge c_m$ est satisfaite ssi $\forall i \in \{1, m\}, c_i$ est satisfaite
- $\mathcal{F} = c_1 \wedge \ldots \wedge c_m$ est contredite ssi $\exists i \in \{1, m\} / c_i$ est contredite

Exemple de formule sous CNF

$(x_2 \vee \bar{x_5} \vee x_4) \wedge (x_1 \vee x_3) \wedge (\bar{x_4} \vee \bar{x_2} \vee \bar{x_1} \vee x_5)$
Cette formule contient 3 clauses et 5 variables différentes

SAT

Problème de décision

- Soit $\mathcal{F}$ une formule sous CNF, $\mathcal{X}$ l'ensemble de ses variables et $\mathcal{C}$ l'ensemble de ses clauses
- Problème SAT : Existe-t-il une interprétation des éléments de $\mathcal{X}$ sur $\{\text{VRAI}, \text{FAUX}\}$ telle que toutes les clauses de $\mathcal{C}$ soient satisfaites ?
- Réponses possibles : oui ou non

Complexité

- Polynômiale : 1-SAT, 2-SAT, Horn-SAT
- k-SAT ($k \geq 3$) : $\mathcal{NP}$. *$\mathcal{NP}$-complet* pour 3-SAT [Cook - 1971]
- Accroissement exponentiel de l'espace de recherche associé à une augmentation linéaire de la taille du problème

SAT

Problème de décision

- Soit $\mathcal{F}$ une formule sous CNF, $\mathcal{X}$ l'ensemble de ses variables et $\mathcal{C}$ l'ensemble de ses clauses
- Problème SAT : Existe-t-il une interprétation des éléments de $\mathcal{X}$ sur $\{\text{VRAI}, \text{FAUX}\}$ telle que toutes les clauses de $\mathcal{C}$ soient satisfaites ?
- Réponses possibles : oui ou non

Complexité

- Polynômiale : 1-SAT, 2-SAT, Horn-SAT
- k-SAT ($k \geq 3$) : $\mathcal{NP}$. $\mathcal{NP}$-*complet* pour 3-SAT [Cook - 1971]
- Accroissement exponentiel de l'espace de recherche associé à une augmentation linéaire de la taille du problème

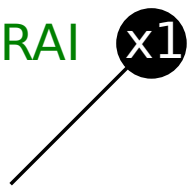
Plan

Algorithme DLL [Davis, Logemann, Loveland - 1962]

$$\mathcal{F} = \left\{ \begin{array}{lll} c_1 : \bar{x_1} \vee x_4 & c_2 : \bar{x_4} \vee \bar{x_2} & c_3 : x_1 \vee x_5 \vee \bar{x_6} \\ c_4 : \bar{x_4} \vee \bar{x_3} \vee x_7 & c_5 : \bar{x_3} \vee \bar{x_7} & c_6 : x_1 \vee x_4 \\ c_7 : \bar{x_5} \vee x_6 & c_8 : x_3 \vee x_2 \vee x_6 & \end{array} \right\}$$

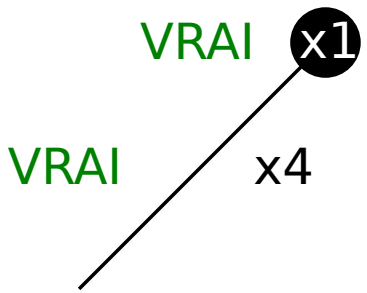
Algorithme DLL [Davis, Logemann, Loveland - 1962]

$$\mathcal{F} = \left\{ \begin{array}{lll} c_1 : \bar{x_1} \vee x_4 & c_2 : \bar{x_4} \vee \bar{x_2} & c_3 : x_1 \vee x_5 \vee \bar{x_6} \\ c_4 : \bar{x_4} \vee \bar{x_3} \vee x_7 & c_5 : \bar{x_3} \vee \bar{x_7} & c_6 : x_1 \vee x_4 \\ c_7 : \bar{x_5} \vee x_6 & c_8 : x_3 \vee x_2 \vee x_6 & \end{array} \right\}$$



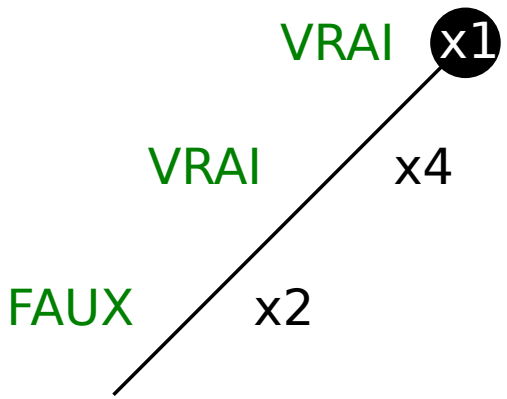
Algorithme DLL [Davis, Logemann, Loveland - 1962]

$$\mathcal{F} = \left\{ \begin{array}{lll} c_1 : \bar{x_1} \vee x_4 & c_2 : \bar{x_4} \vee \bar{x_2} & c_3 : x_1 \vee x_5 \vee \bar{x_6} \\ c_4 : \bar{x_4} \vee \bar{x_3} \vee x_7 & c_5 : \bar{x_3} \vee \bar{x_7} & c_6 : x_1 \vee x_4 \\ c_7 : \bar{x_5} \vee x_6 & c_8 : x_3 \vee x_2 \vee x_6 & \end{array} \right\}$$



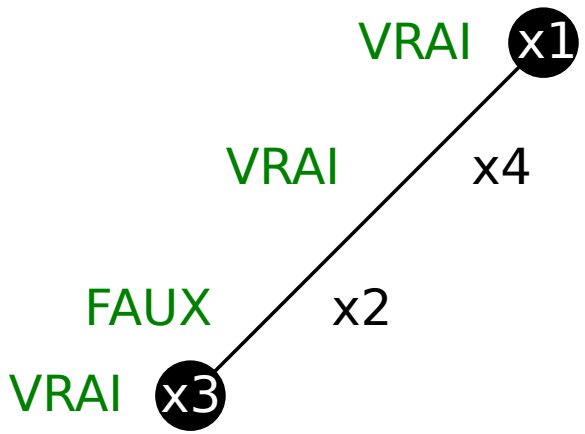
Algorithme DLL [Davis, Logemann, Loveland - 1962]

$$\mathcal{F} = \left\{ \begin{array}{lll} c_1 : \bar{x_1} \vee x_4 & c_2 : \bar{x_4} \vee \bar{x_2} & c_3 : x_1 \vee x_5 \vee \bar{x_6} \\ c_4 : \bar{x_4} \vee \bar{x_3} \vee x_7 & c_5 : \bar{x_3} \vee \bar{x_7} & c_6 : x_1 \vee x_4 \\ c_7 : \bar{x_5} \vee x_6 & c_8 : x_3 \vee x_2 \vee x_6 & \end{array} \right\}$$



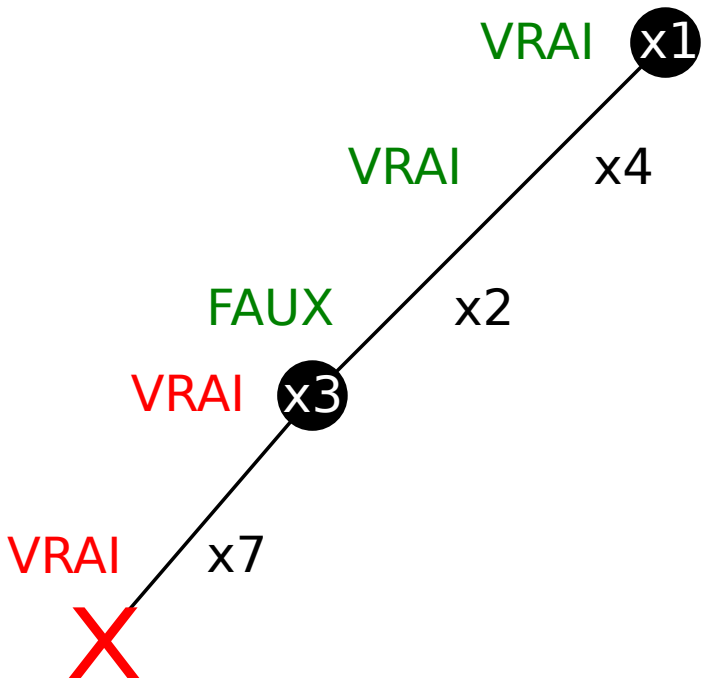
Algorithme DLL [Davis, Logemann, Loveland - 1962]

$$\mathcal{F} = \left\{ \begin{array}{lll} c_1 : \bar{x_1} \vee x_4 & c_2 : \bar{x_4} \vee \bar{x_2} & c_3 : x_1 \vee x_5 \vee \bar{x_6} \\ c_4 : \bar{x_4} \vee \bar{x_3} \vee x_7 & c_5 : \bar{x_3} \vee \bar{x_7} & c_6 : x_1 \vee x_4 \\ c_7 : \bar{x_5} \vee x_6 & c_8 : x_3 \vee x_2 \vee x_6 & \end{array} \right\}$$



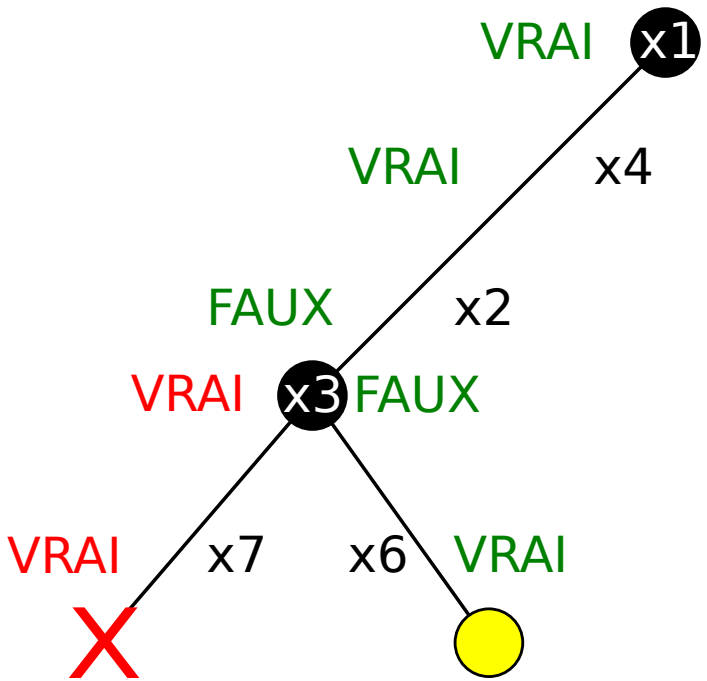
Algorithme DLL [Davis, Logemann, Loveland - 1962]

$$\mathcal{F} = \left\{ \begin{array}{lll} c_1 : \bar{x_1} \vee x_4 & c_2 : \bar{x_4} \vee \bar{x_2} & c_3 : x_1 \vee x_5 \vee \bar{x_6} \\ c_4 : \bar{x_4} \vee \bar{x_3} \vee x_7 & c_5 : \bar{x_3} \vee \bar{x_7} & c_6 : x_1 \vee x_4 \\ c_7 : \bar{x_5} \vee x_6 & c_8 : x_3 \vee x_2 \vee x_6 & \end{array} \right\}$$



Algorithme DLL [Davis, Logemann, Loveland - 1962]

$$\mathcal{F} = \left\{ \begin{array}{lll} c_1 : \bar{x_1} \vee x_4 & c_2 : \bar{x_4} \vee \bar{x_2} & c_3 : x_1 \vee x_5 \vee \bar{x_6} \\ c_4 : \bar{x_4} \vee \bar{x_3} \vee x_7 & c_5 : \bar{x_3} \vee \bar{x_7} & c_6 : x_1 \vee x_4 \\ c_7 : \bar{x_5} \vee x_6 & c_8 : x_3 \vee x_2 \vee x_6 & \end{array} \right\}$$



Méthodes de résolution

Algorithmes complets

- Basés sur `DLL`
- Exploration systématique et implicite de l'espace de recherche
- Garantie de la réponse quelque soit la nature de la formule
- `kcnfs` [Dequen, Dubois - 2003], `March` [Heule, Van Maaren - 2006]

Algorithmes incomplets

- Heuristiques
- Bonnes performances dans certains cas
- Non garantie de la réponse mais relaxation de la complexité calculatoire
- `WalkSAT` [Selman, Kautz, Cohen - 1996], `Survey Propagation` [Braunstein, Mézard, Zecchina - 2005]

Méthodes de résolution

Algorithmes complets

- Basés sur `DLL`
- Exploration systématique et implicite de l'espace de recherche
- Garantie de la réponse quelque soit la nature de la formule
- `kcnfs` [Dequen, Dubois - 2003], `March` [Heule, Van Maaren - 2006]

Algorithmes incomplets

- Heuristiques
- Bonnes performances dans certains cas
- Non garantie de la réponse mais relaxation de la complexité calculatoire
- `WalkSAT` [Selman, Kautz, Cohen - 1996], `Survey Propagation` [Braunstein, Mézard, Zecchina - 2005]

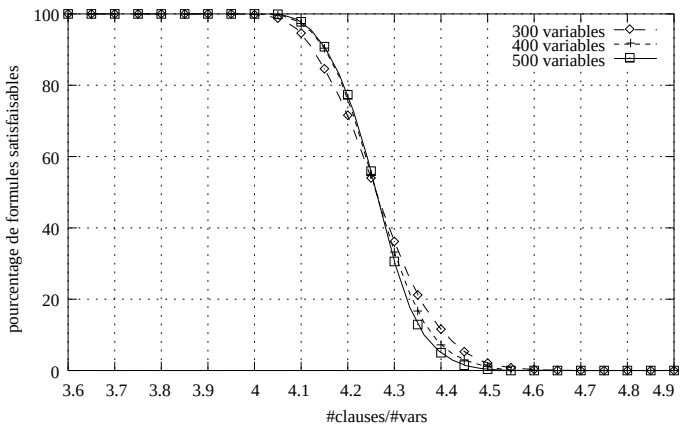
Plan

Formules générées aléatoirement

- Étude théorique du problème
  - Étude des espaces de solutions
  - Compréhension de la difficulté du problème
  - Confrontation directe avec l'explosion combinatoire
- Modèle k-SAT aléatoire (clauses de longueur k)
- Rapport $\frac{m}{n}$ (m le nombre de clauses et n le nombre de variables)

Formules générées aléatoirement

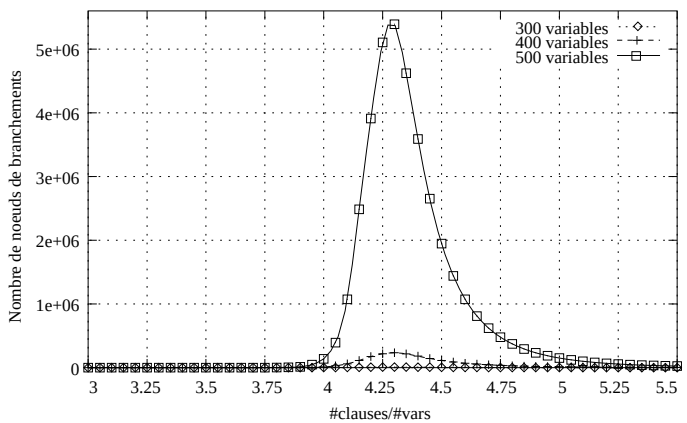
- Étude théorique du problème
  - Étude des espaces de solutions
  - Compréhension de la difficulté du problème
  - Confrontation directe avec l'explosion combinatoire
- Modèle k-SAT aléatoire (clauses de longueur k)
- Rapport $\frac{m}{n}$ (m le nombre de clauses et n le nombre de variables)



Phénomène de transition de phase

Formules générées aléatoirement

- Étude théorique du problème
  - Étude des espaces de solutions
  - Compréhension de la difficulté du problème
  - Confrontation directe avec l'explosion combinatoire
- Modèle k-SAT aléatoire (clauses de longueur k)
- Rapport $\frac{m}{n}$ (m le nombre de clauses et n le nombre de variables)



Pic de difficulté relatif à la transition de phase

Formules industrielles

Domaines d'application

- Vérification de modèles (circuits électroniques)
- Intelligence artificielle
- Ordonnancement
- Cryptographie, ...

Particularités de CDCL

- Apprentissage de clauses à partir des conflits
- Mesure et exploitation de l'activité des variables et des clauses
- Politiques de redémarrages
- Propagation unitaire rapide

Formules industrielles

Domaines d'application

- Vérification de modèles (circuits électroniques)
- Intelligence artificielle
- Ordonnancement
- Cryptographie, ...

Solveurs SAT CDCL $\longrightarrow$ résolution efficace en pratique
`zChaff` [Zhang, Madigan, Moskewicz, Malik - 2001]
`Minisat 2` [Sörensson, Eén - 2006]

Particularités de CDCL

- Apprentissage de clauses à partir des conflits
- Mesure et exploitation de l'activité des variables et des clauses
- Politiques de redémarrages
- Propagation unitaire rapide

Formules industrielles

Domaines d'application

- Vérification de modèles (circuits électroniques)
- Intelligence artificielle
- Ordonnancement
- Cryptographie, ...

Solveurs SAT CDCL $\longrightarrow$ résolution efficace en pratique
`zChaff` [Zhang, Madigan, Moskewicz, Malik - 2001]
`Minisat 2` [Sörensson, Eén - 2006]

Particularités de CDCL

- Apprentissage de clauses à partir des conflits
- Mesure et exploitation de l'activité des variables et des clauses
- Politiques de redémarrages
- Propagation unitaire rapide

Plan

Définition

Parallélisme

Faire collaborer plusieurs unités de calcul sur le même problème

Différents modèles

- Exécution (classification de Flynn)
- Architecture (accès mémoire)
- Programmation (gestion des échanges)
- Modèles liés

Notions

- Inter blocage
- Équilibrage
- Grain de parallélisme

Définition

Parallélisme

Faire collaborer plusieurs unités de calcul sur le même problème

Différents modèles

- Exécution (classification de Flynn)
- Architecture (accès mémoire)
- Programmation (gestion des échanges)
- Modèles liés

Notions

- Inter blocage
- Équilibrage
- Grain de parallélisme

Définition

Parallélisme

Faire collaborer plusieurs unités de calcul sur le même problème

Différents modèles

- Exécution (classification de Flynn)
- Architecture (accès mémoire)
- Programmation (gestion des échanges)
- Modèles liés

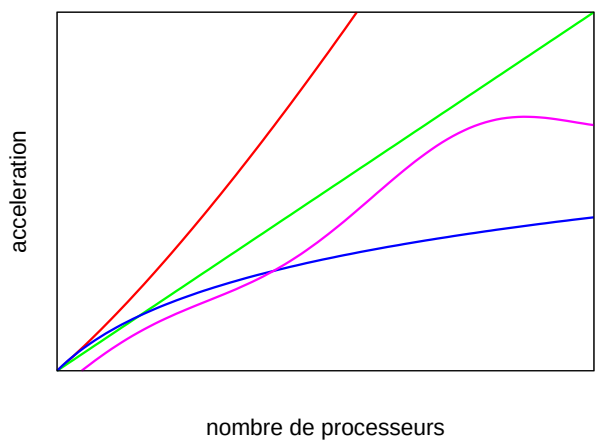
Notions

- Inter blocage
- Équilibrage
- Grain de parallélisme

Performances

Critères

- Temps séquentiel : T_{seq}
- Temps parallèle sur n processeurs : $T_{//}(n)$
- Accélération sur n processeurs, $Acc(n) = \frac{T_{seq}}{T_{//}(n)}$
- Efficacité sur n processeurs : $Eff(n) = \frac{T_{seq}}{T_{//}(n) \times n}$



Plan

Types de coopération

Modèle concurrentiel

- Mise en concurrence de plusieurs exécutions
- Exploite l'indéterminisme ou le paramétrage
- `ManySat` [Hamadi, Jabbour, Sais - 2008], `gNovelty+-T` [Gretton, Pham - 2009]

Modèle collaboratif

- Travail divisé sur plusieurs unités de calcul
- Problématique de l'équilibrage de charge
- Approche la plus répandue depuis 1994 : `//satz` [Li, Jurkowiak - 2001], `ySat` [Feldman, Dershowitz, Hanna - 2004], `MiraXT` [Lewis, Schubert, Becker - 2007], `PMinisat` [Chu, Stuckey - 2008], `JackSat` [Singer, Monnet - 2007]

Types de coopération

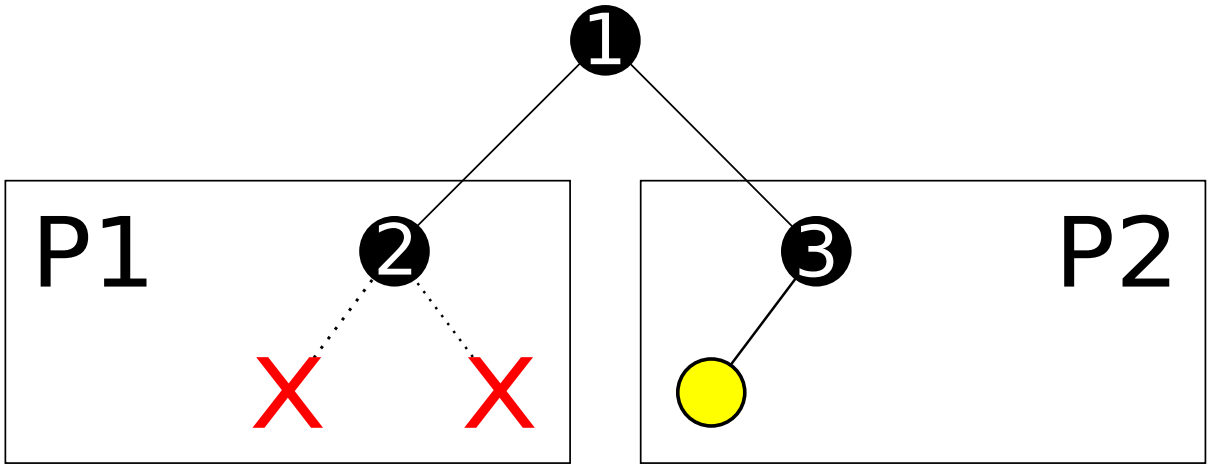
Modèle concurrentiel

- Mise en concurrence de plusieurs exécutions
- Exploite l'indéterminisme ou le paramétrage
- `ManySat` [Hamadi, Jabbour, Sais - 2008], `gNovelty+-T` [Gretton, Pham - 2009]

Modèle collaboratif

- Travail divisé sur plusieurs unités de calcul
- Problématique de l'équilibrage de charge
- Approche la plus répandue depuis 1994 : `//satz` [Li, Jurkowiak - 2001], `ySat` [Feldman, Dershowitz, Hanna - 2004], `MiraXT` [Lewis, Schubert, Becker - 2007], `PMinisat` [Chu, Stuckey - 2008], `JackSat` [Singer, Monnet - 2007]

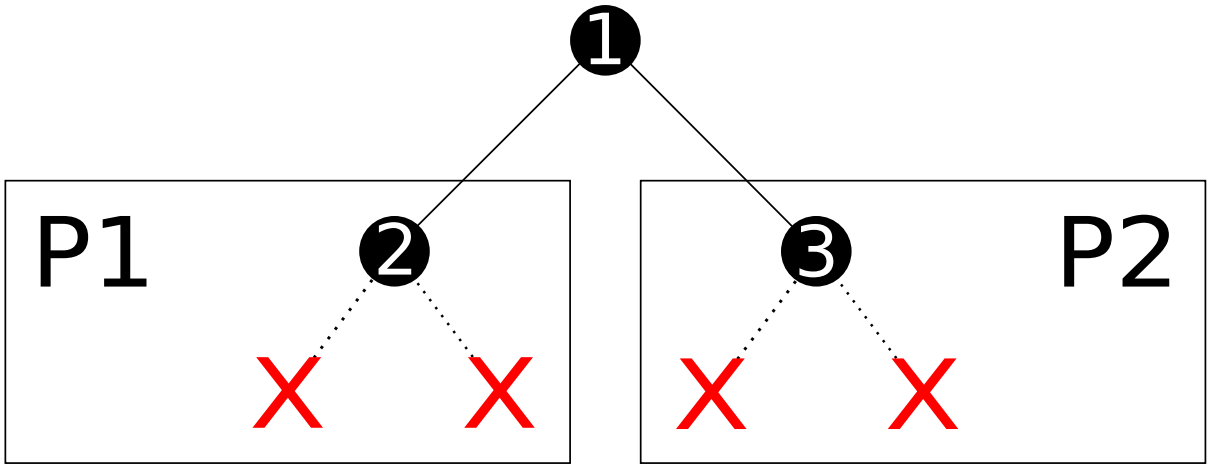
Accélération



Si la formule est satisfaisable

Accélération super linéaire possible

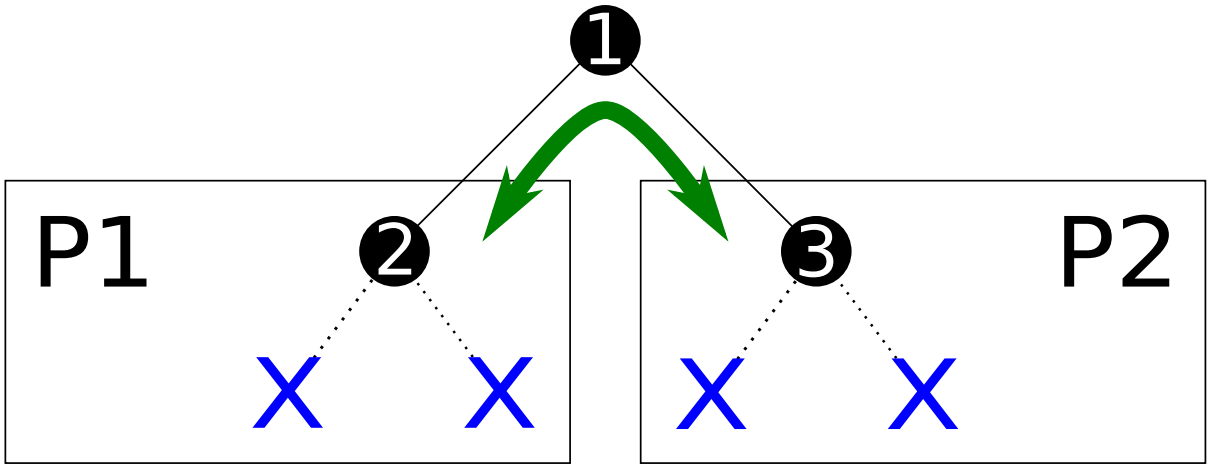
Accélération



Si la formule est insatisfaisable

Accélération linéaire au mieux (nombre de nœuds identique)

Accélération

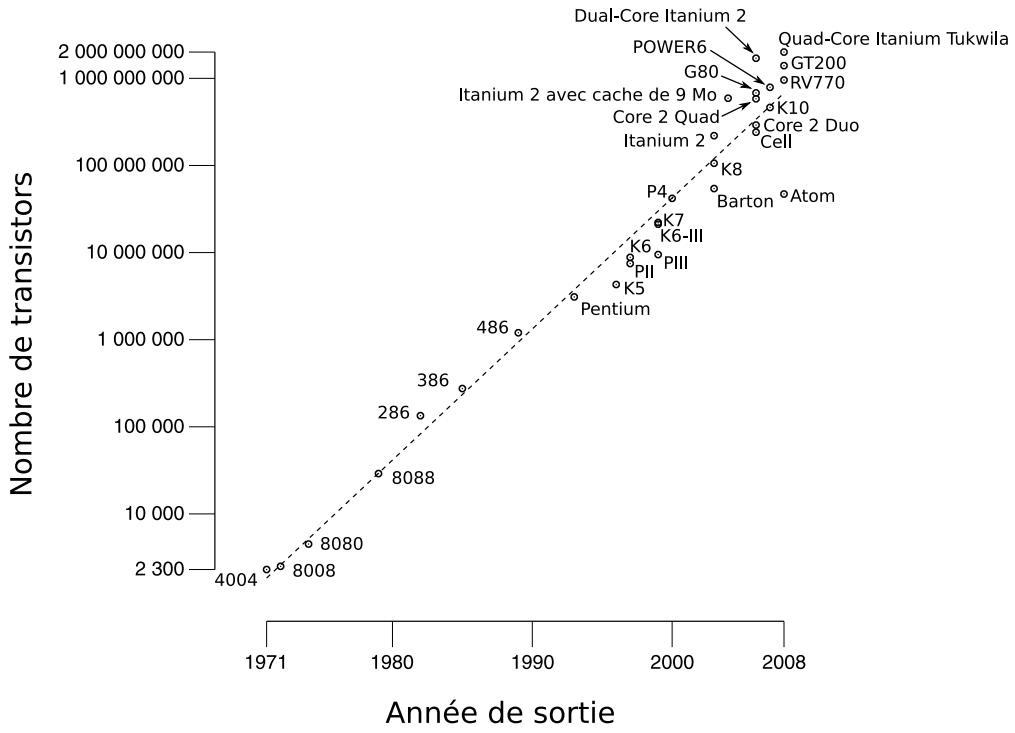


Si la formule est insatisfaisable, avec échange d'information

Accélération super linéaire possible grâce à la diminution du nombre de nœuds

Plan

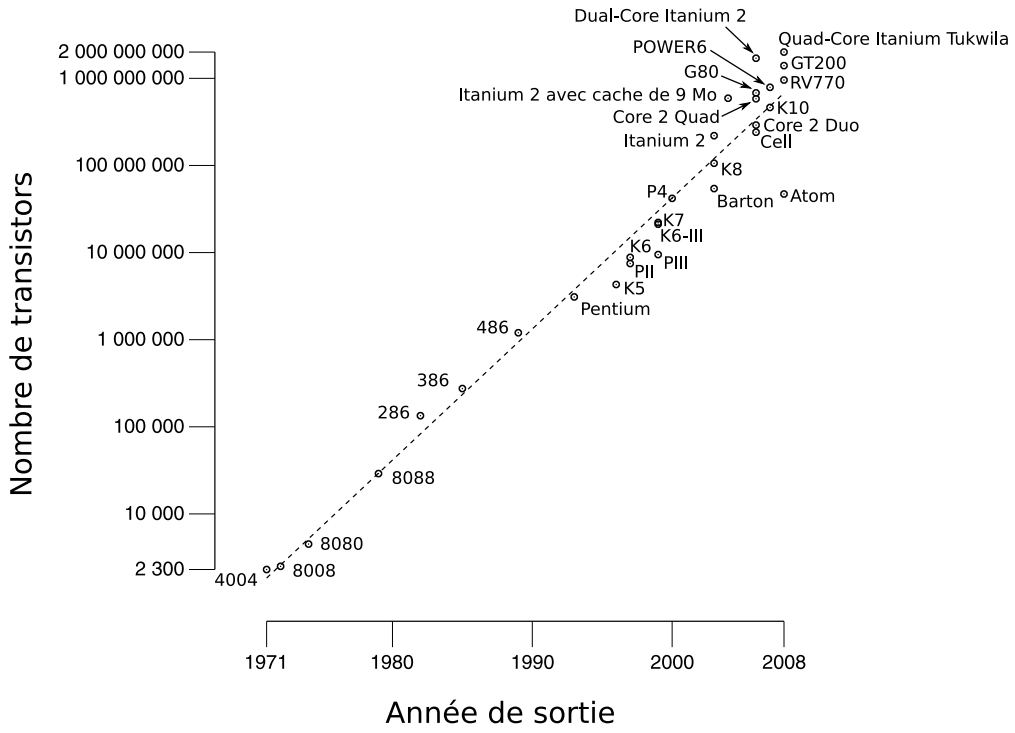
Les processeurs



Limite

- Dégagement thermique : limite de 4 GHz
- Solution : augmenter le nombre de cœurs de calcul

Les processeurs

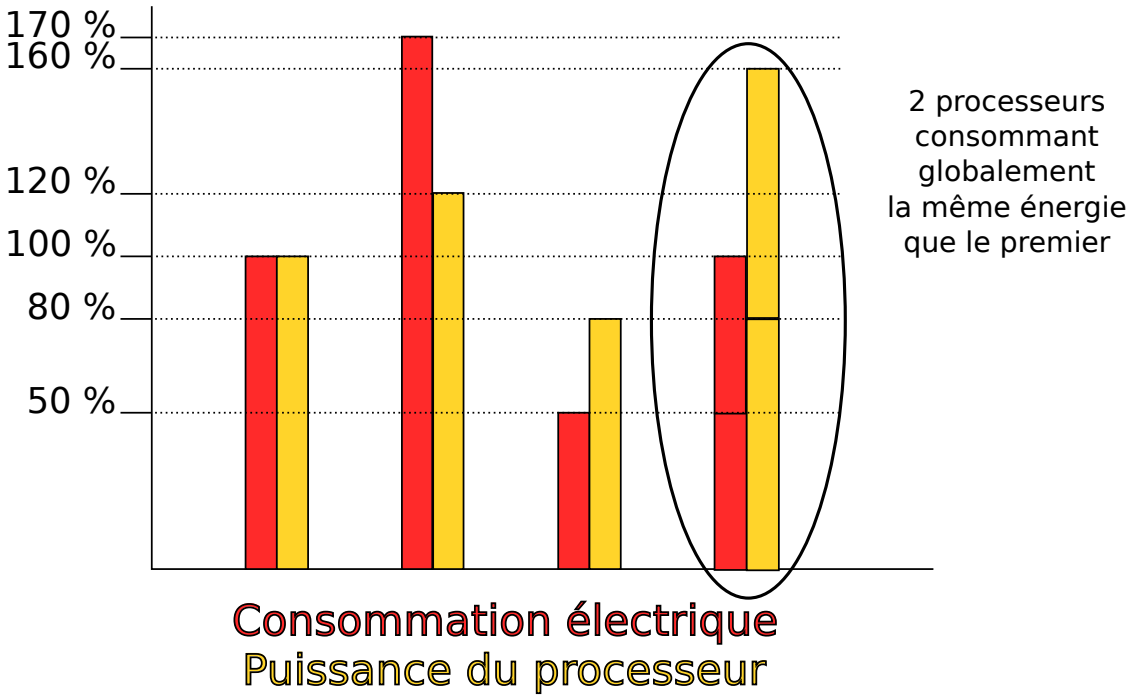


Limite

- Dégagement thermique : limite de 4 GHz
- Solution : augmenter le nombre de cœurs de calcul

Plan

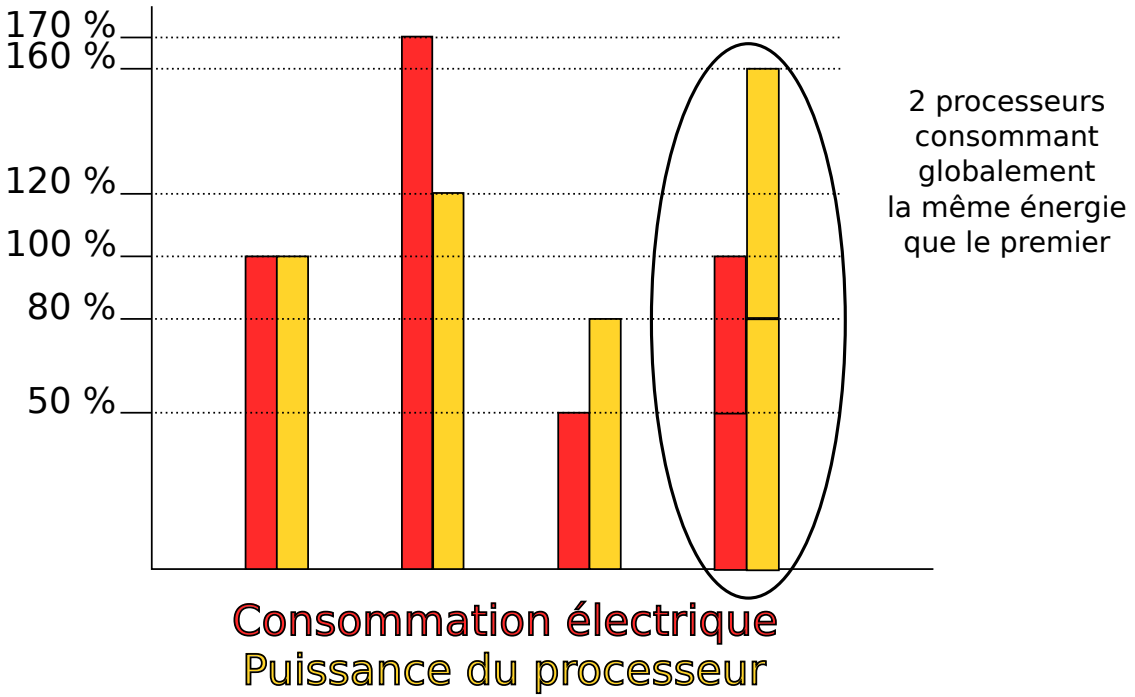
Puissance *vs* Consommation



Multi-cœurs

- Même consommation ⇒ Plus de performance
- Problème : Performance à tirer au niveau de l'algorithme
- Solution : Penser en algorithmique parallèle

Puissance *vs* Consommation



Multi-cœurs

- Même consommation ⇒ Plus de performance
- Problème : Performance à tirer au niveau de l'algorithme
- Solution : Penser en algorithmique parallèle

Plan

Motivations

Progrès dans SAT

- Étudié depuis des décennies
- Très concurrentiel
- Riche en apports théoriques et techniques
- Nombreux solveurs existants

Processeurs multi-cœurs

- Devient un standard
- Faible coût
- Accès rapide et uniforme à la mémoire (machine CC-UMA)
  $\Rightarrow$ Modèle de programmation PRAM
- Processeur efficace si bien exploité
- Intégration aux super-calculateurs

Motivations

Progrès dans SAT

- Étudié depuis des décennies
- Très concurrentiel
- Riche en apports théoriques et techniques
- Nombreux solveurs existants

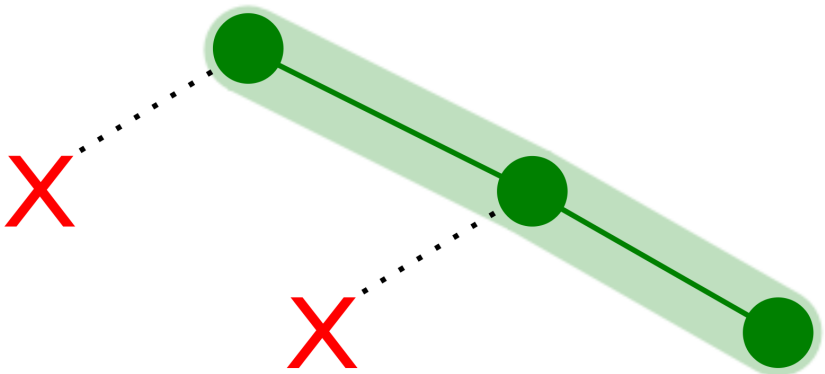
Processeurs multi-cœurs

- Devient un standard
- Faible coût
- Accès rapide et uniforme à la mémoire (machine CC-UMA)
  $\Rightarrow$ Modèle de programmation PRAM
- Processeur efficace si bien exploité
- Intégration aux super-calculateurs

Plan

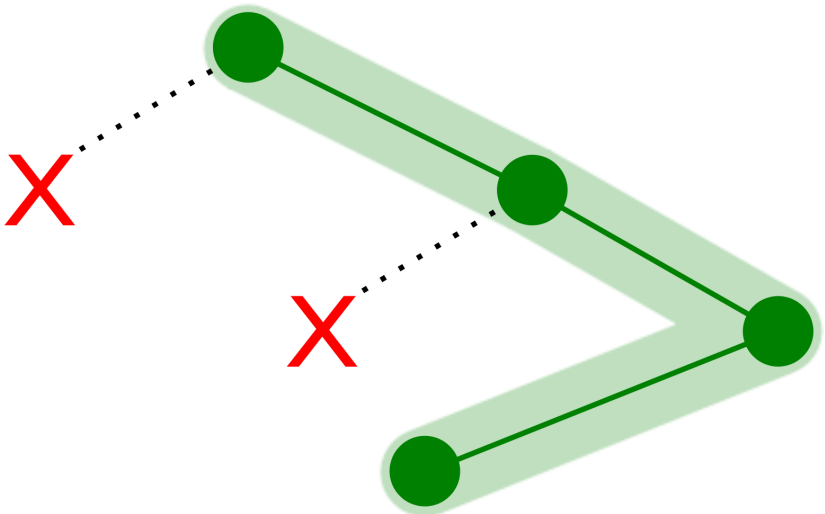
Chemin de guidage

PSATO [Zhang, Bonacina, Paola - 1994]



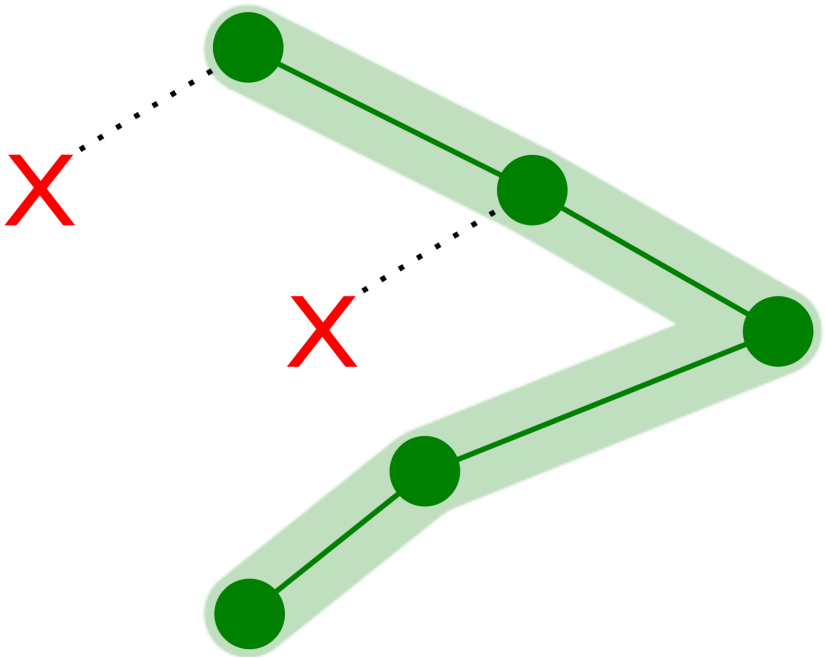
Chemin de guidage

PSATO [Zhang, Bonacina, Paola - 1994]



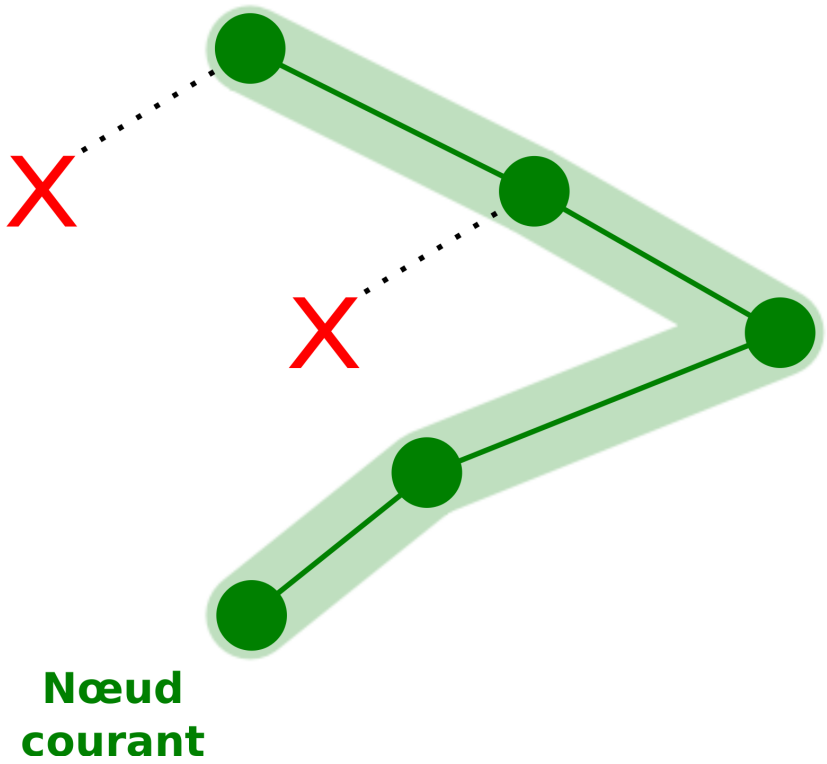
Chemin de guidage

PSATO [Zhang, Bonacina, Paola - 1994]



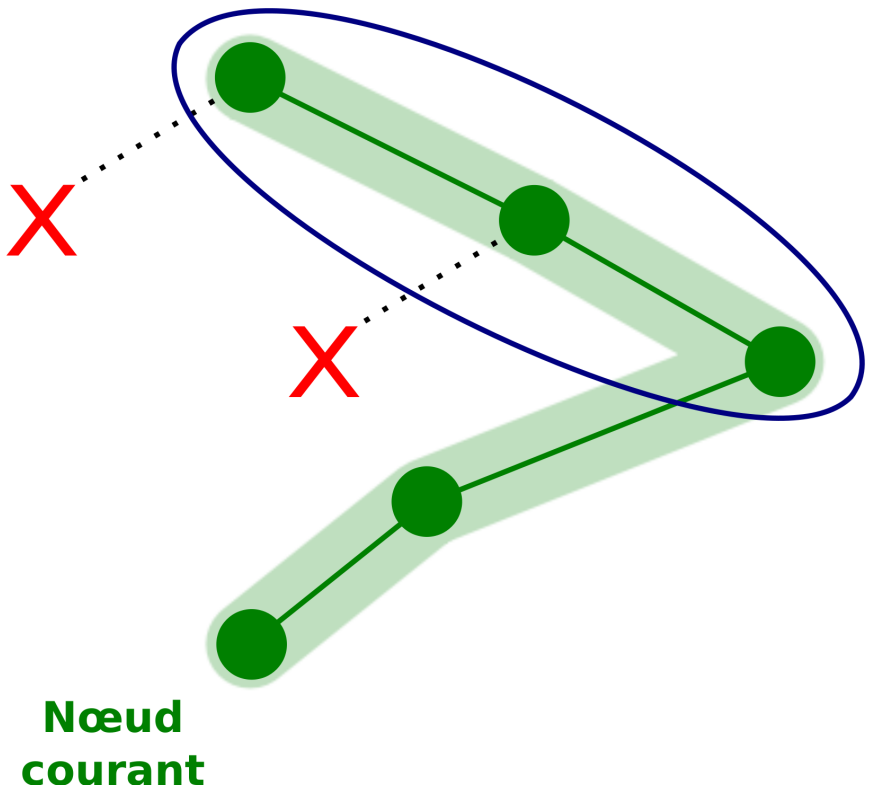
Chemin de guidage

PSATO [Zhang, Bonacina, Paola - 1994]



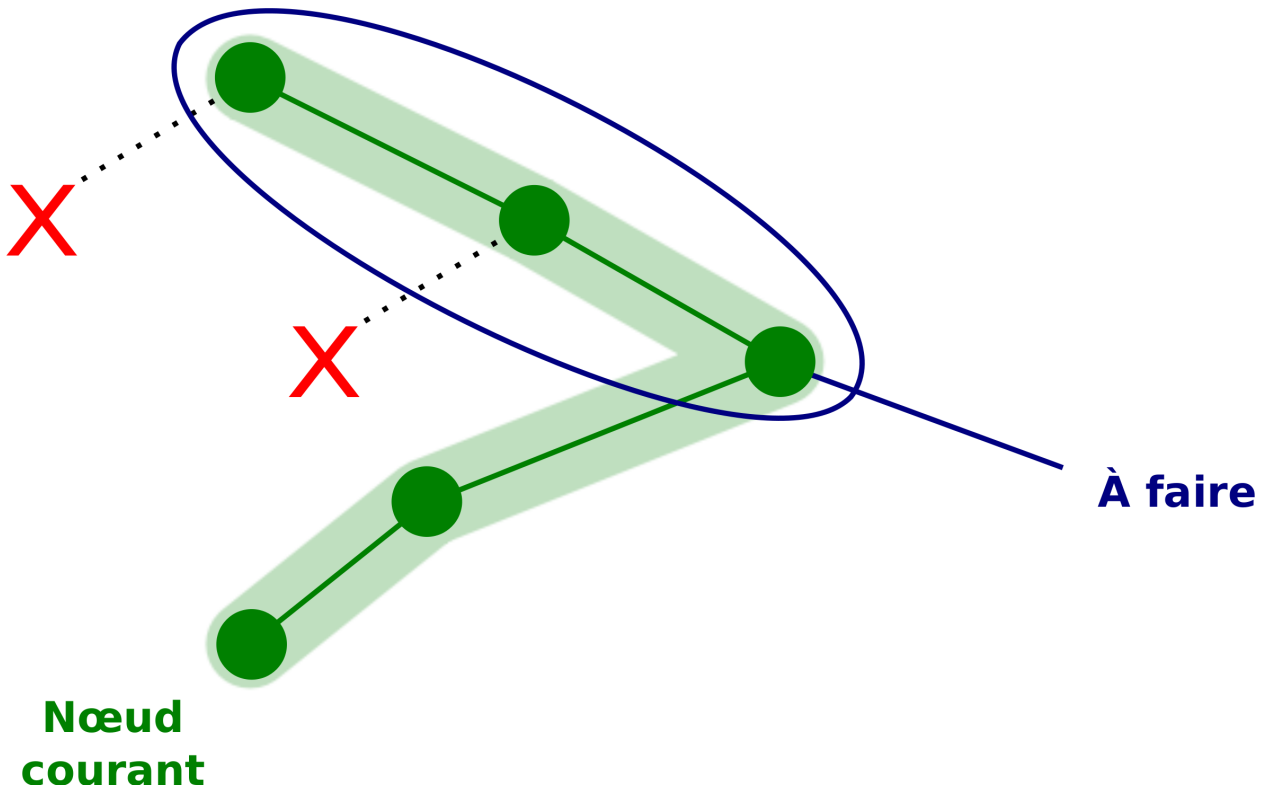
Chemin de guidage

PSATO [Zhang, Bonacina, Paola - 1994]



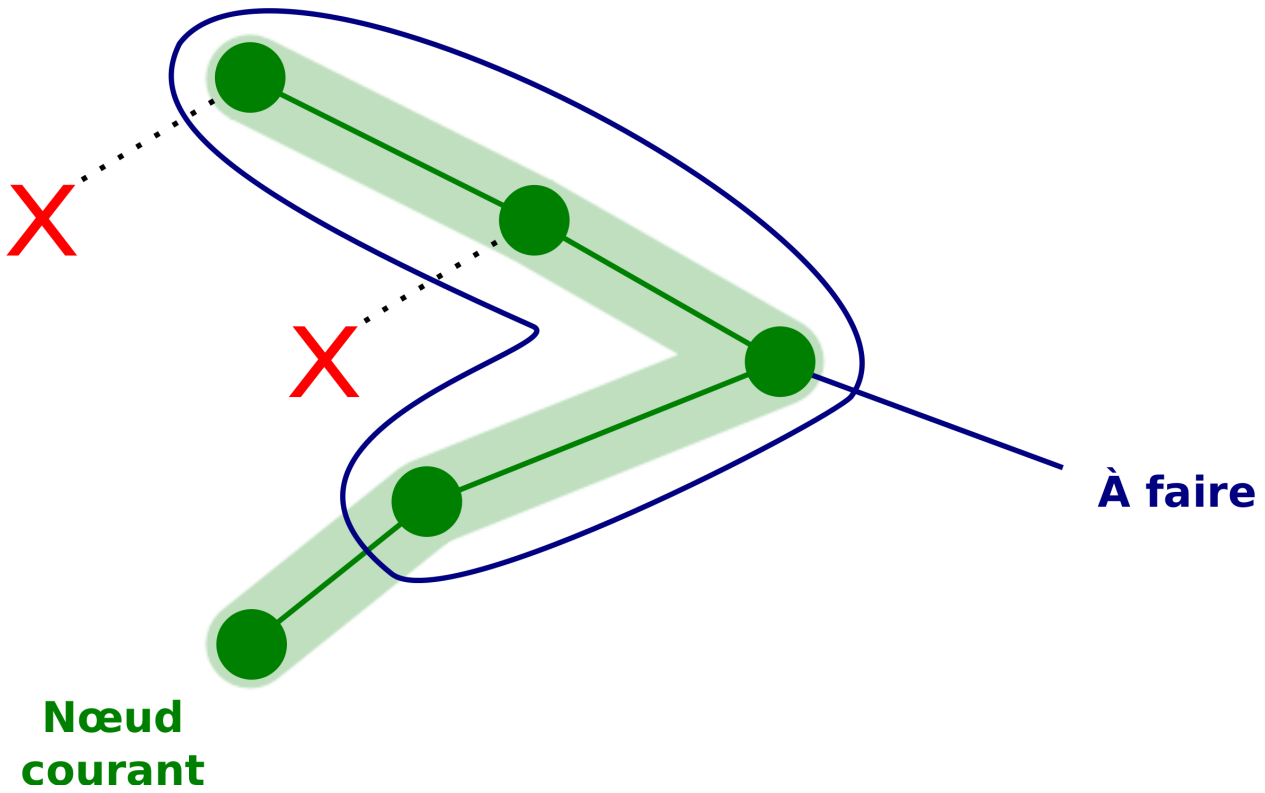
Chemin de guidage

PSATO [Zhang, Bonacina, Paola - 1994]



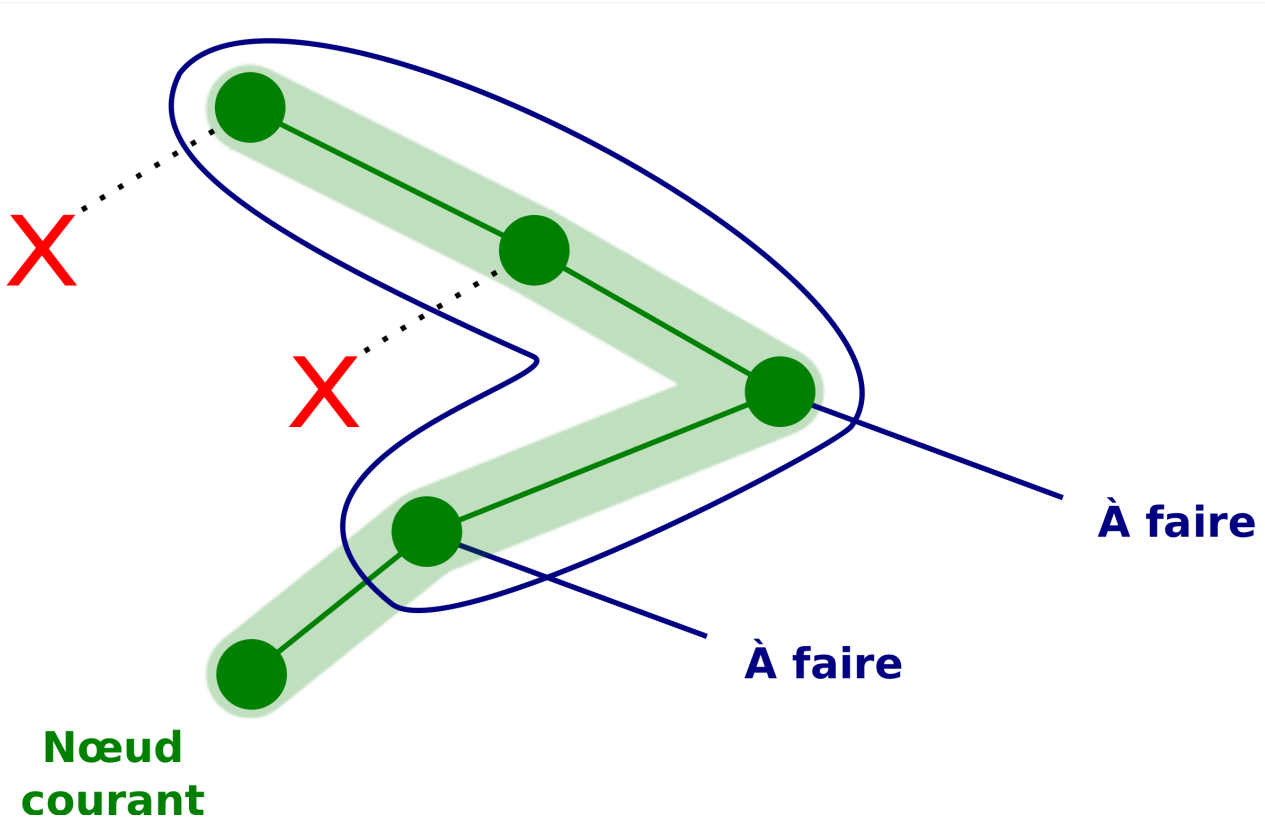
Chemin de guidage

PSATO [Zhang, Bonacina, Paola - 1994]



Chemin de guidage

PSATO [Zhang, Bonacina, Paola - 1994]



Chemin de guidage

Avantages

- Facile et naturel à construire
- Génère des espaces disjoints
- Adapté pour la mémoire distribuée

Inconvénients

- Construction séquentielle
- Déséquilibre des tâches
- Parallélisme à gros grain exclusivement
- N'exploite pas la mémoire partagée
- Passage à l'échelle limité
- Gestion spécifique pour l'équilibrage de charge

Chemin de guidage

Avantages

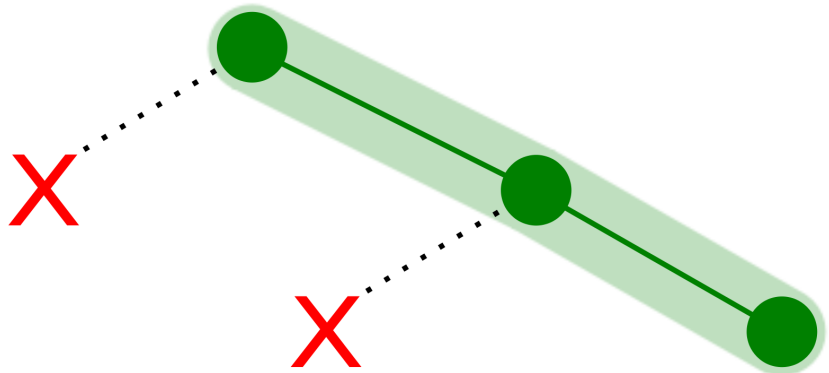
- Facile et naturel à construire
- Génère des espaces disjoints
- Adapté pour la mémoire distribuée

Inconvénients

- Construction séquentielle
- Déséquilibre des tâches
- Parallélisme à gros grain exclusivement
- N'exploite pas la mémoire partagée
- Passage à l'échelle limité
- Gestion spécifique pour l'équilibrage de charge

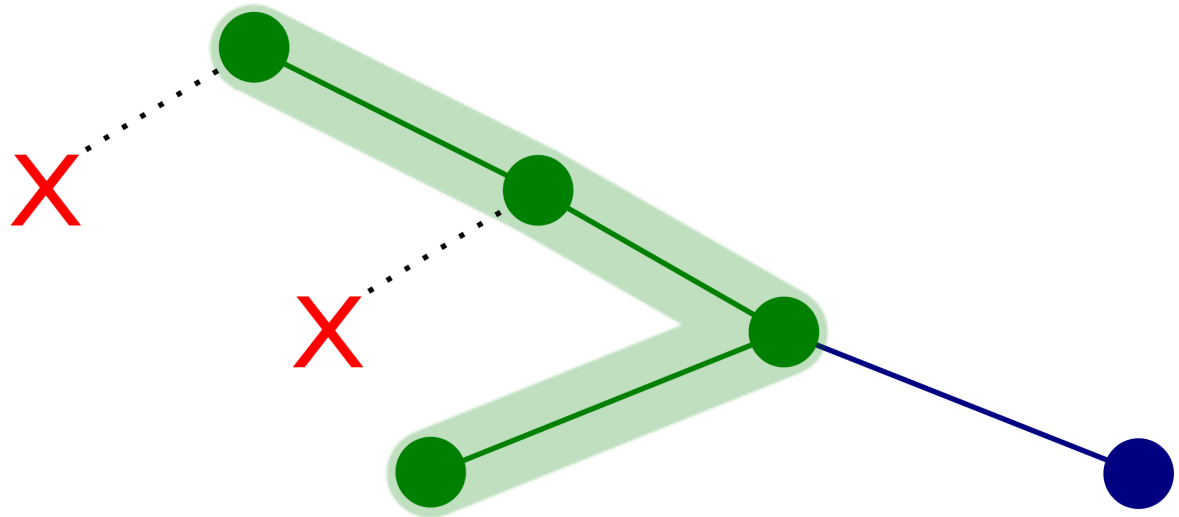
Arbre de guidage

Arbre de guidage au sein de MTSS [IWOMP'08, IJPP'09]



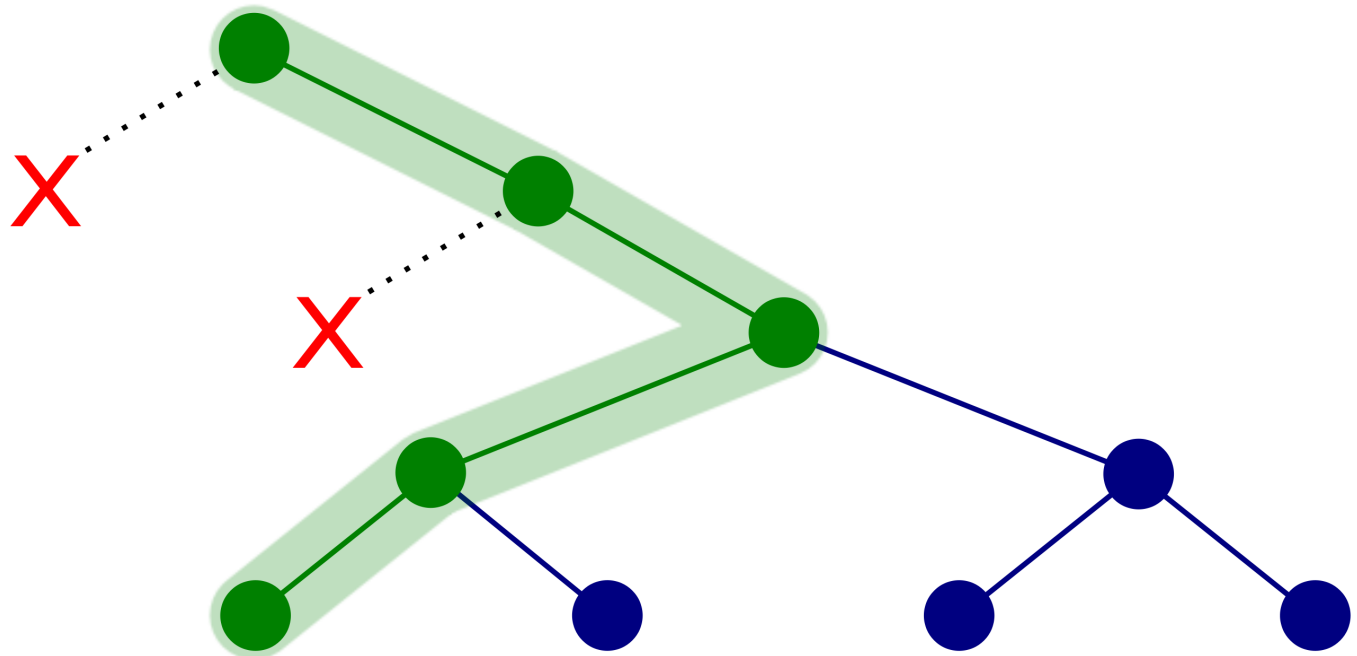
Arbre de guidage

Arbre de guidage au sein de MTSS [IWOMP'08, IJPP'09]



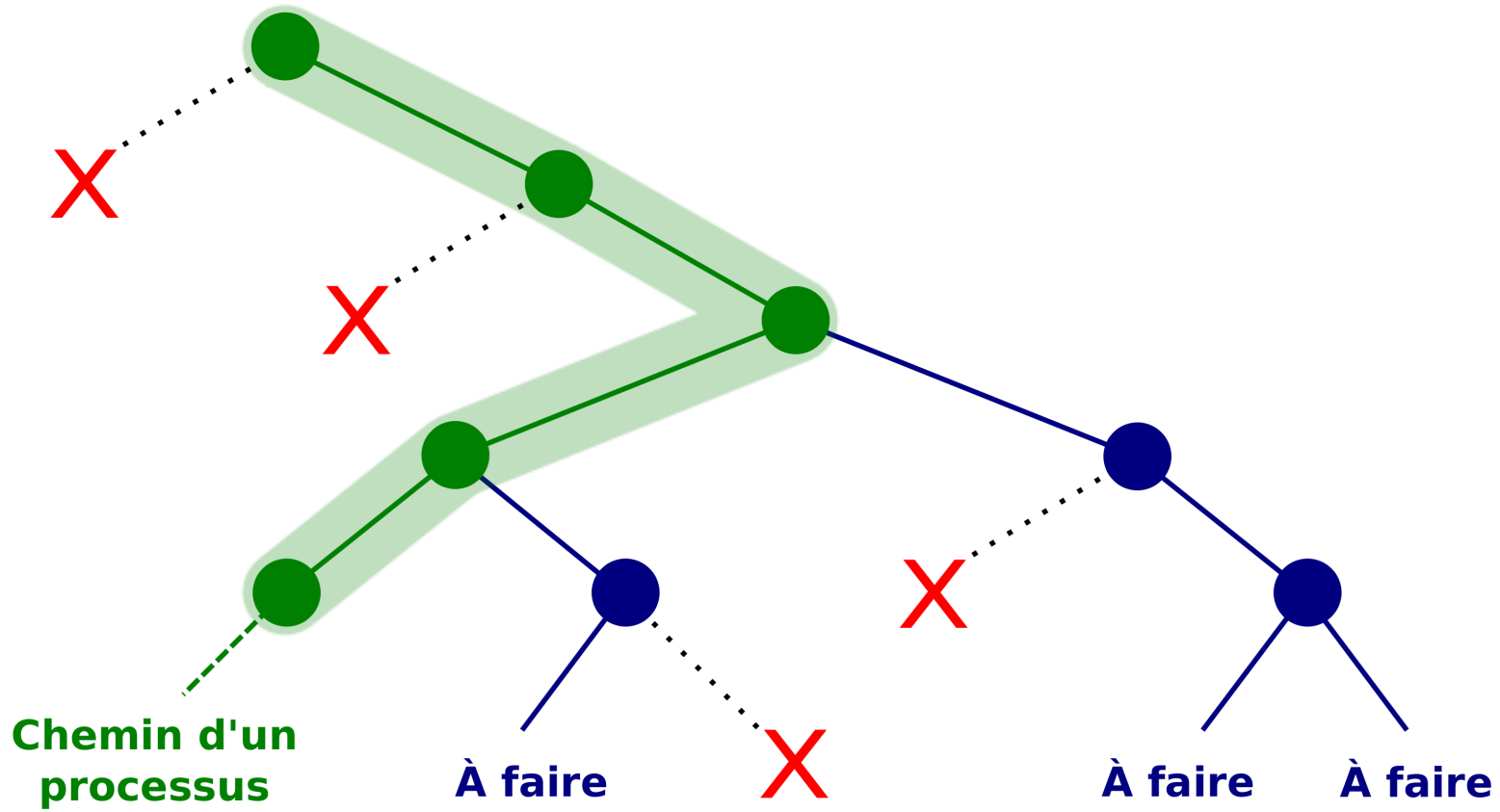
Arbre de guidage

Arbre de guidage au sein de MTSS [IWOMP'08, IJPP'09]



Arbre de guidage

Arbre de guidage au sein de MTSS [IWOMP'08, IJPP'09]



Arbre de guidage

Avantages

- Dynamiquement construit en parallèle
- Équilibrage de charge naturel
- Grain fin
- Nombreuses tâches

Plan

Deux types de processus

[IWOMP'08, IJPP'09]

Le processus riche

Décide si la formule admet une solution en un temps fini mais exponentiel (solveur complet type DLL)

Les processus pauvres

Donnent une information partielle ou globale sans garantie à propos de la formule (propagation unitaire, traitement local, recherche incomplète, heuristique de branchement, ...)

Idée principale

Le processus riche est un solveur SAT utilisant les informations calculées par les processus pauvres et déposées dans l'arbre de guidage afin de conclure partiellement ou globalement sur la formule

Deux types de processus

[IWOMP'08, IJPP'09]

Le processus riche

Décide si la formule admet une solution en un temps fini mais exponentiel (solveur complet type DLL)

Les processus pauvres

Donnent une information partielle ou globale sans garantie à propos de la formule (propagation unitaire, traitement local, recherche incomplète, heuristique de branchement, ...)

Idée principale

Le processus riche est un solveur SAT utilisant les informations calculées par les processus pauvres et déposées dans l'arbre de guidage afin de conclure partiellement ou globalement sur la formule

Deux types de processus

[IWOMP'08, IJPP'09]

Le processus riche

Décide si la formule admet une solution en un temps fini mais exponentiel (solveur complet type DLL)

Les processus pauvres

Donnent une information partielle ou globale sans garantie à propos de la formule (propagation unitaire, traitement local, recherche incomplète, heuristique de branchement, ...)

Idée principale

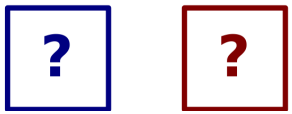
Le processus riche est un solveur SAT utilisant les informations calculées par les processus pauvres et déposées dans l'arbre de guidage afin de conclure partiellement ou globalement sur la formule

Principe général de MTSS

Avantages

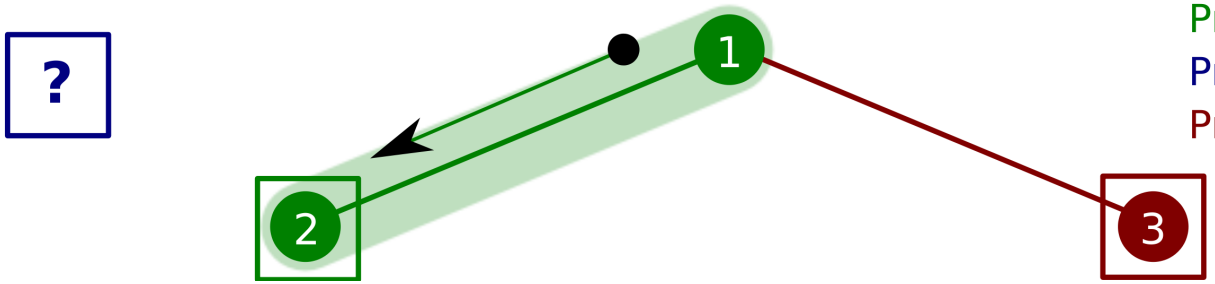
- Cadre général de solveur SAT parallèle
- Utilisation possible des techniques et algorithmes existants
- Grain parallèle dépend des tâches allouées aux processus pauvres

Exemple d'exécution



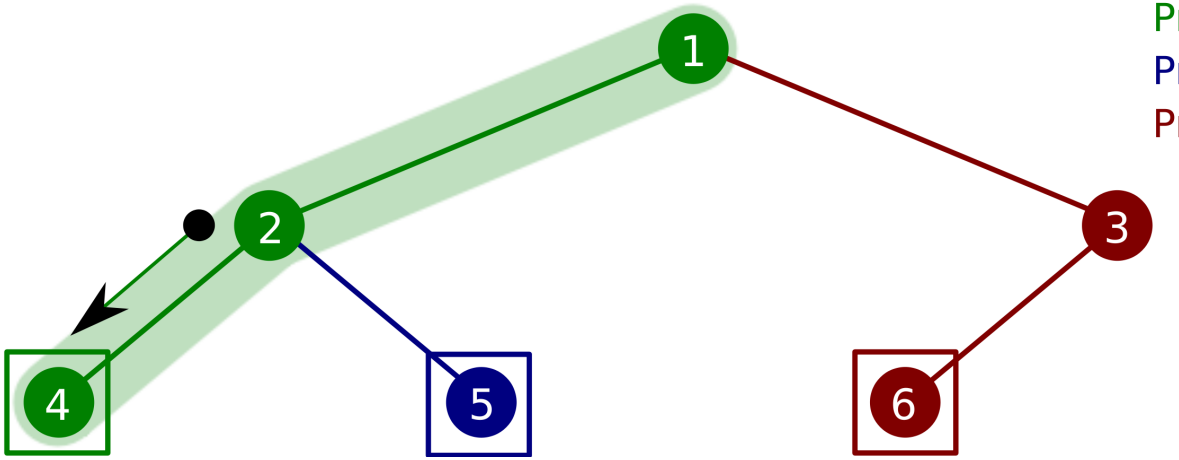
Processus Riche
Processus Pauvre 1
Processus Pauvre 2

Exemple d'exécution



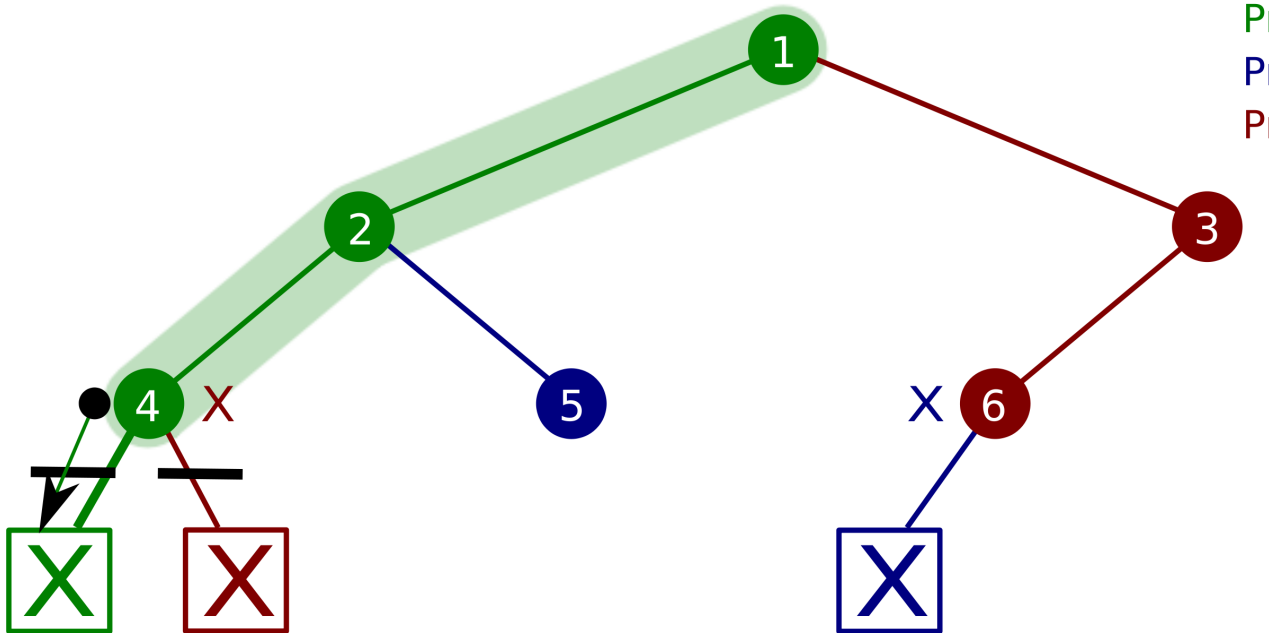
Processus Riche
Processus Pauvre 1
Processus Pauvre 2

Exemple d'exécution



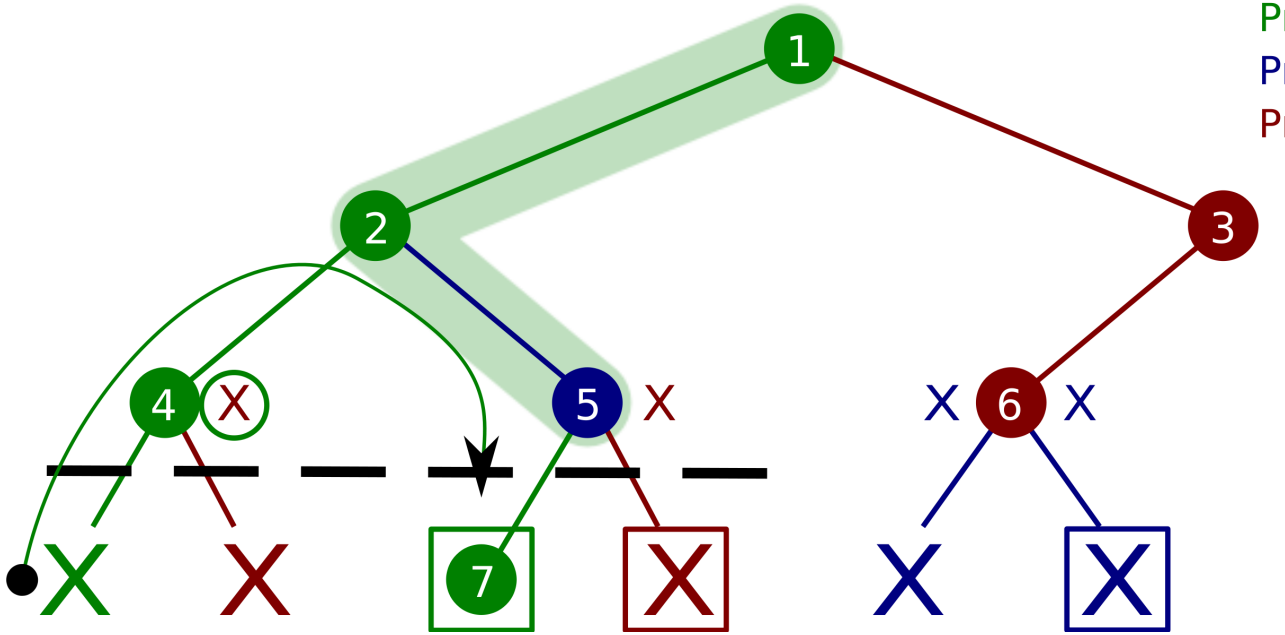
Processus Riche
Processus Pauvre 1
Processus Pauvre 2

Exemple d'exécution



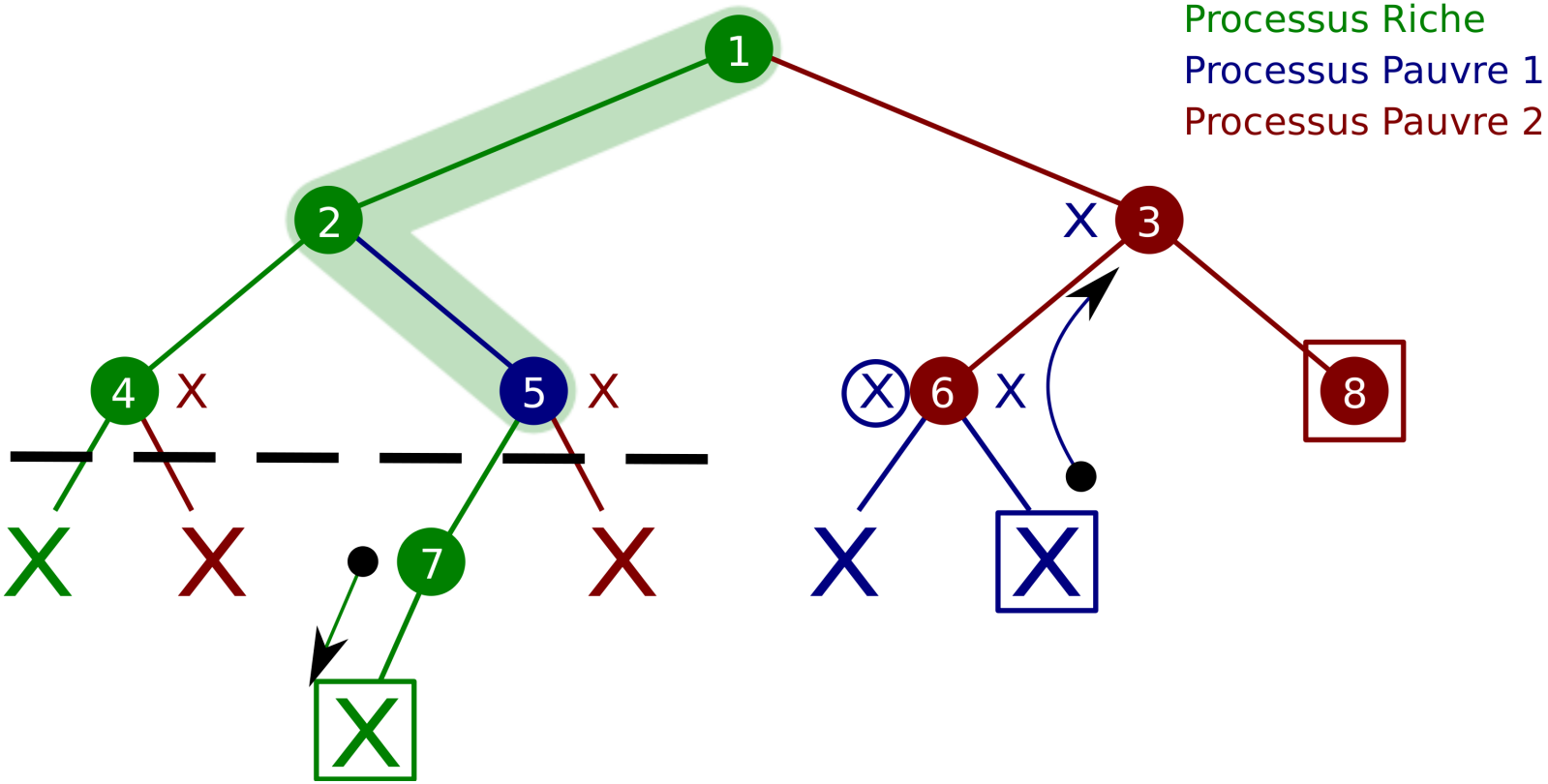
Processus Riche
Processus Pauvre 1
Processus Pauvre 2

Exemple d'exécution

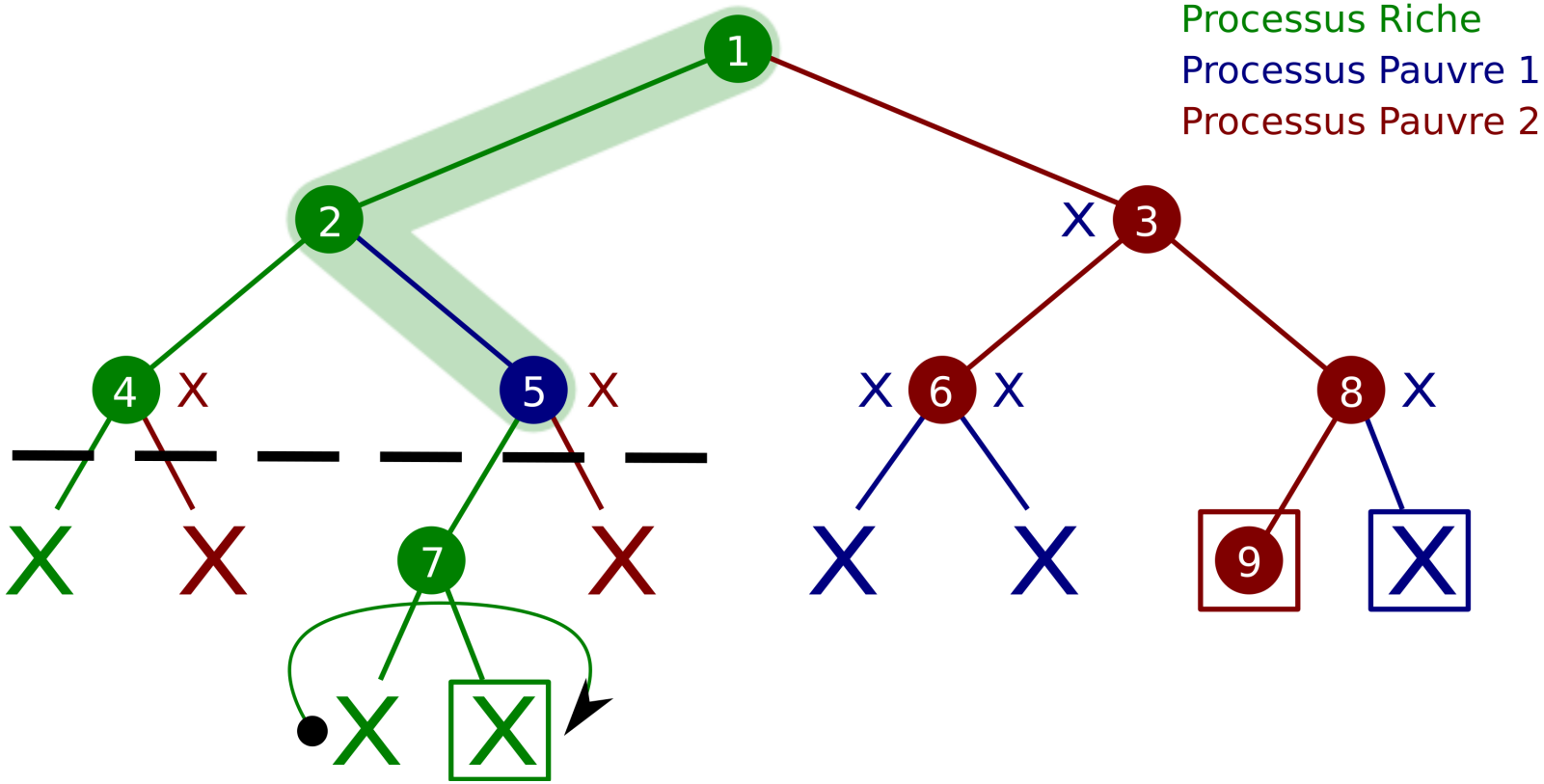


Processus Riche
Processus Pauvre 1
Processus Pauvre 2

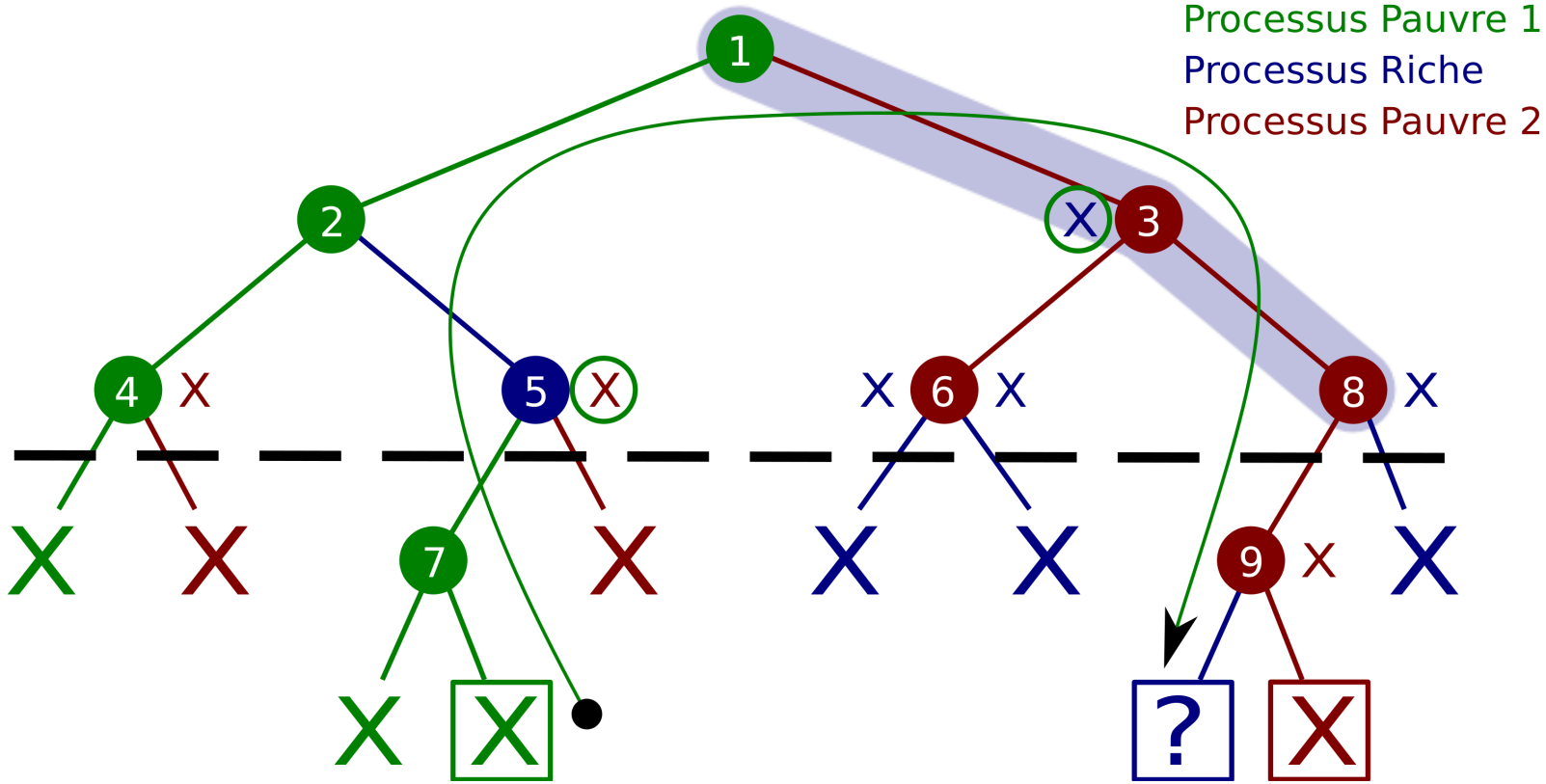
Exemple d'exécution



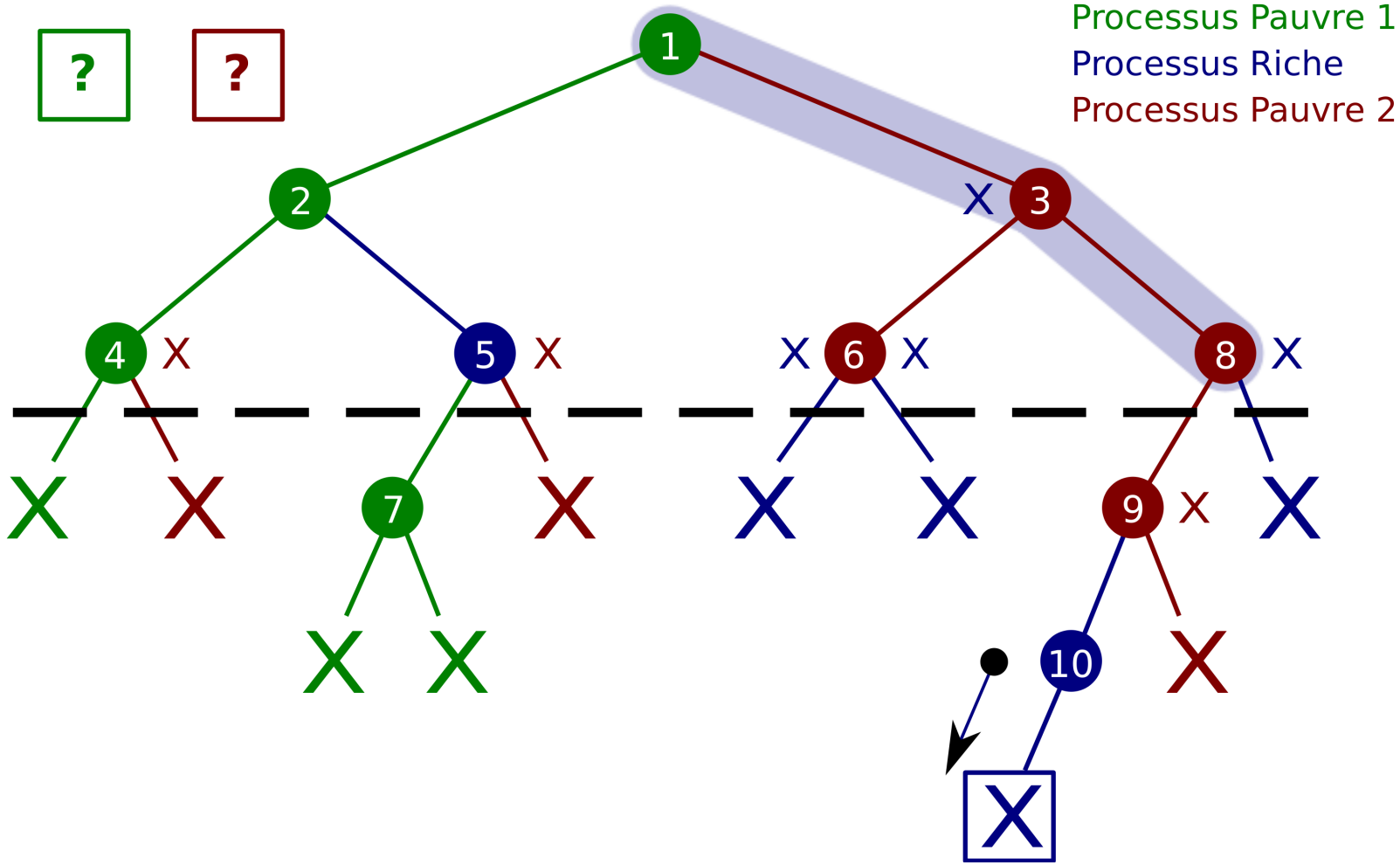
Exemple d'exécution



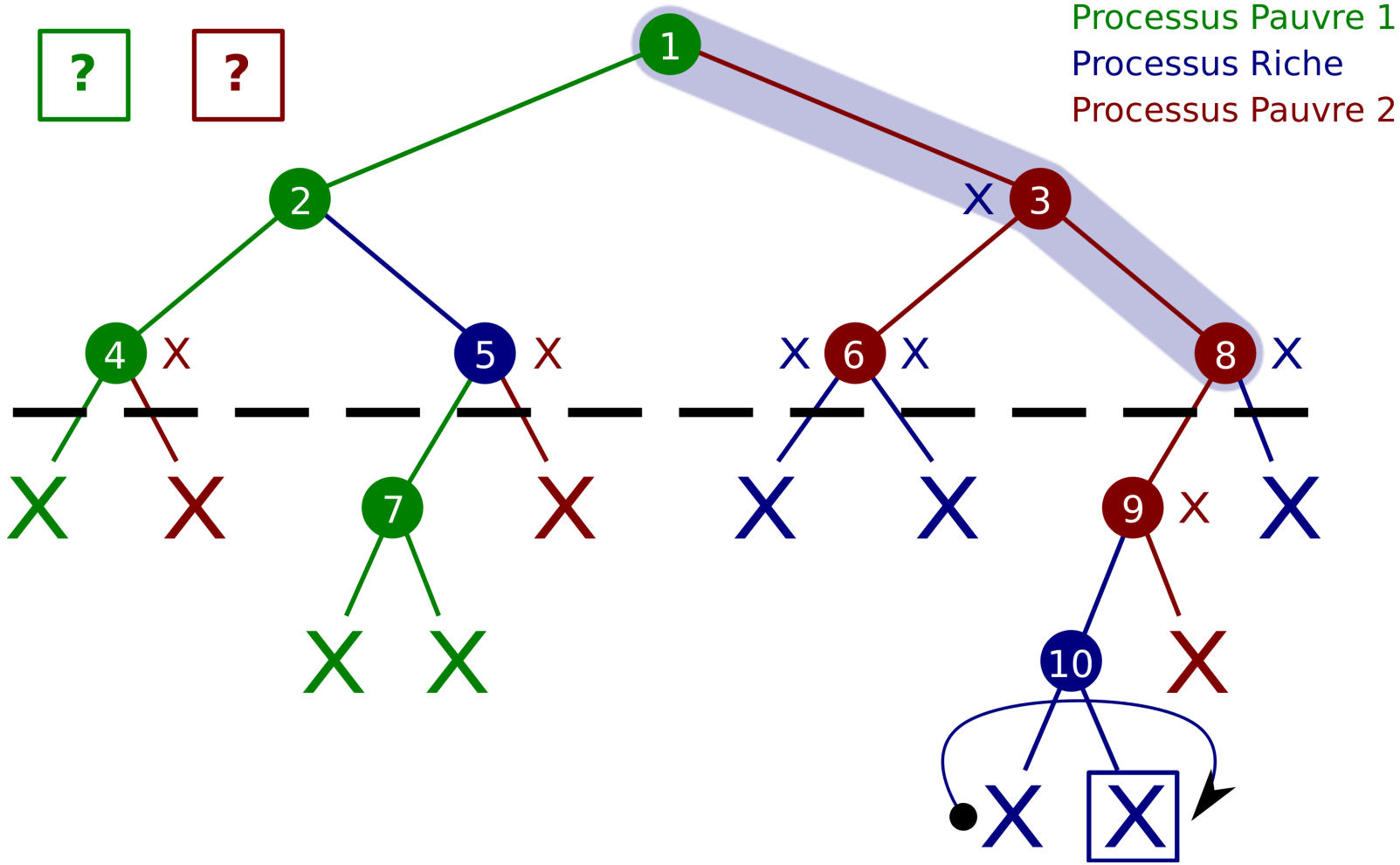
Exemple d'exécution



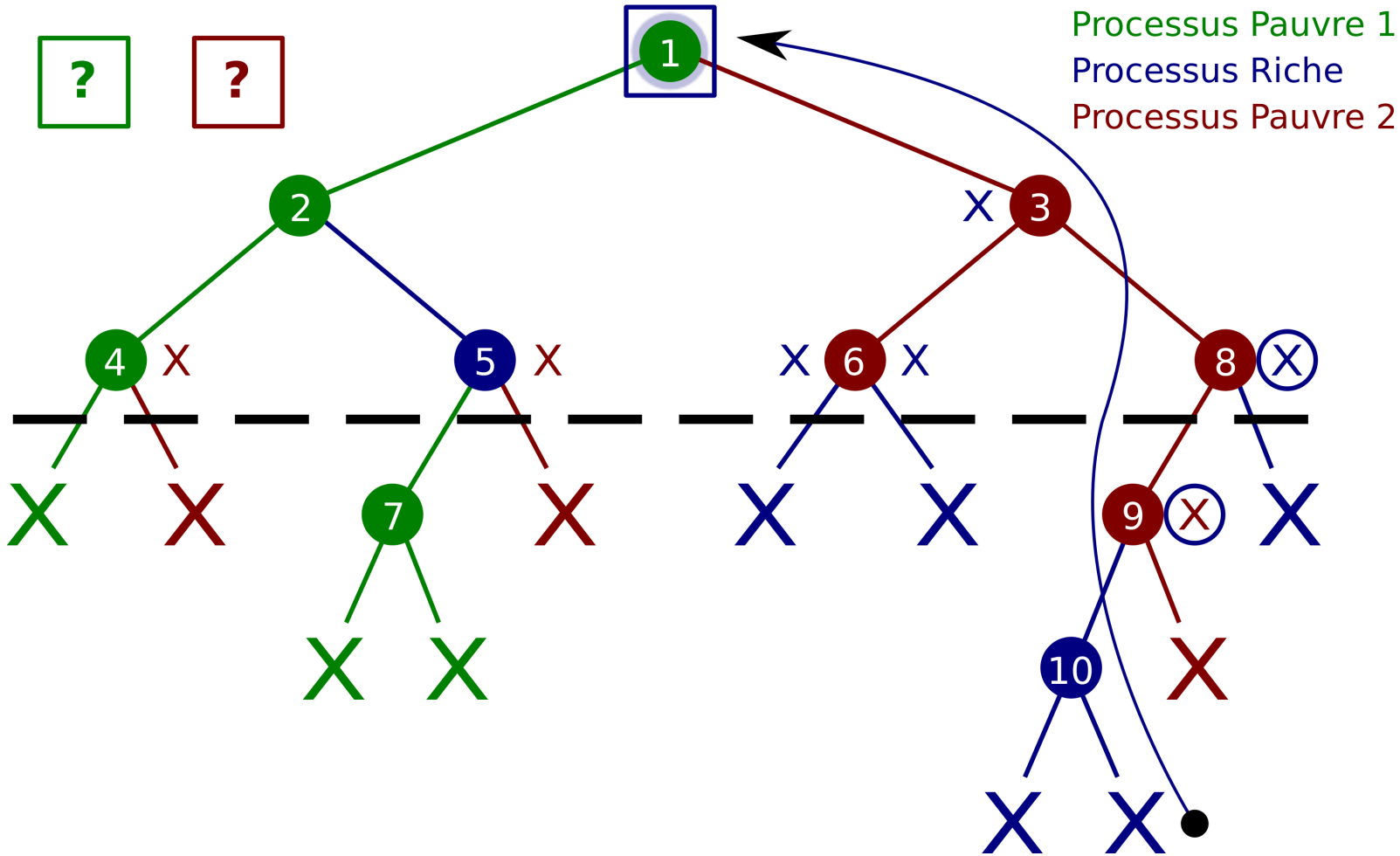
Exemple d'exécution



Exemple d'exécution



Exemple d'exécution



Plan

Processus pauvres

Extension

- Fonctions internes pour plus de vélocité
- Appels à des programmes externes

Outil de parallélisation

- Mutation des processus de MTSS en solveurs externes
- Parallélisme à gros grain
- Arbre de guidage : cadre général de parallélisation et d'hybridation de techniques pour résoudre SAT
- Partage de clauses (solveurs industriels)

Processus pauvres

Extension

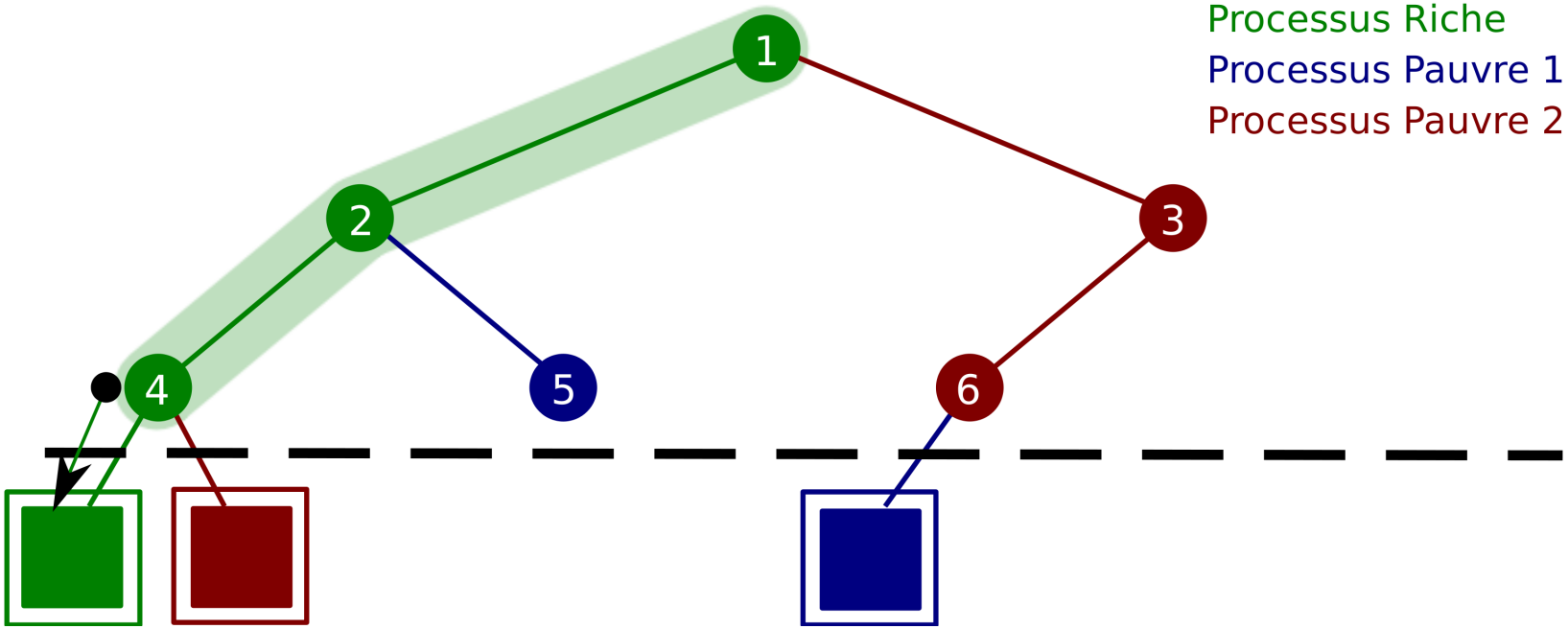
- Fonctions internes pour plus de vélocité
- Appels à des programmes externes

[HPCS'09, ICTAI'09]

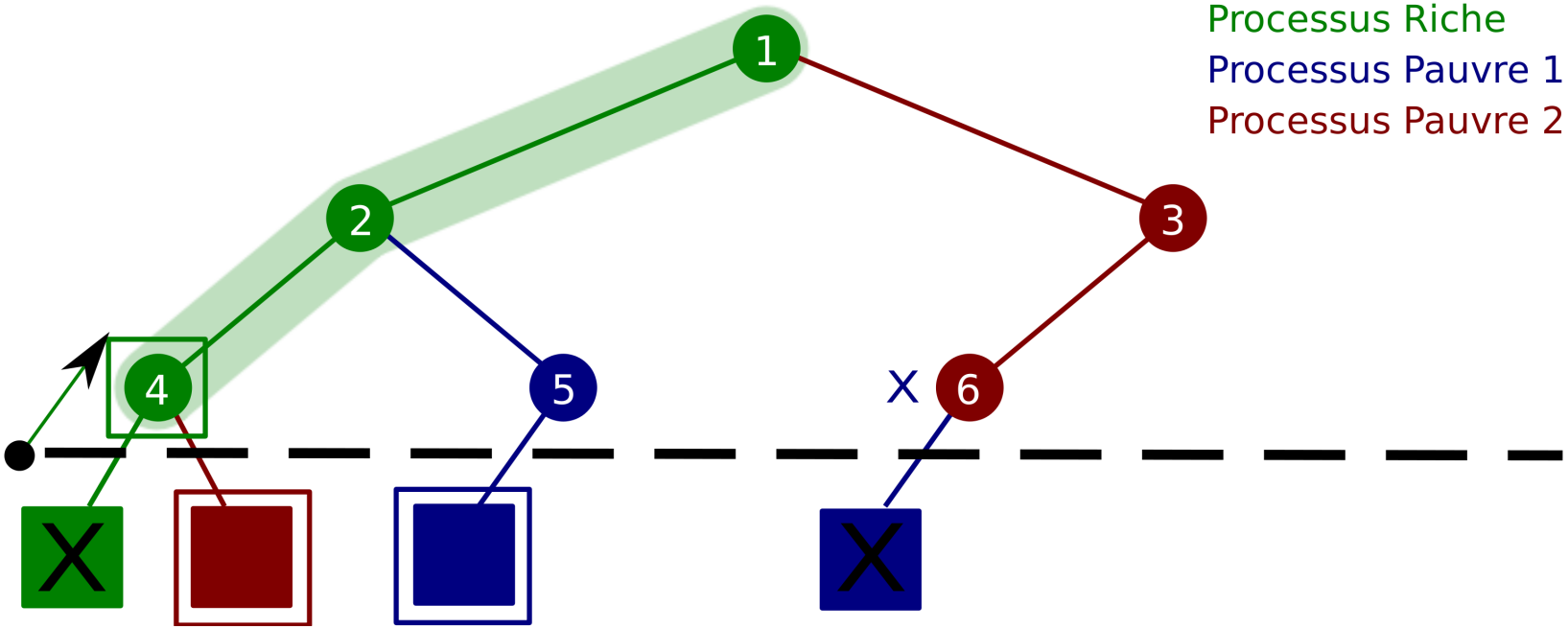
Outil de parallélisation

- Mutation des processus de MTSS en solveurs externes
- Parallélisme à gros grain
- Arbre de guidage : cadre général de parallélisation et d'hybridation de techniques pour résoudre SAT
- Partage de clauses (solveurs industriels)

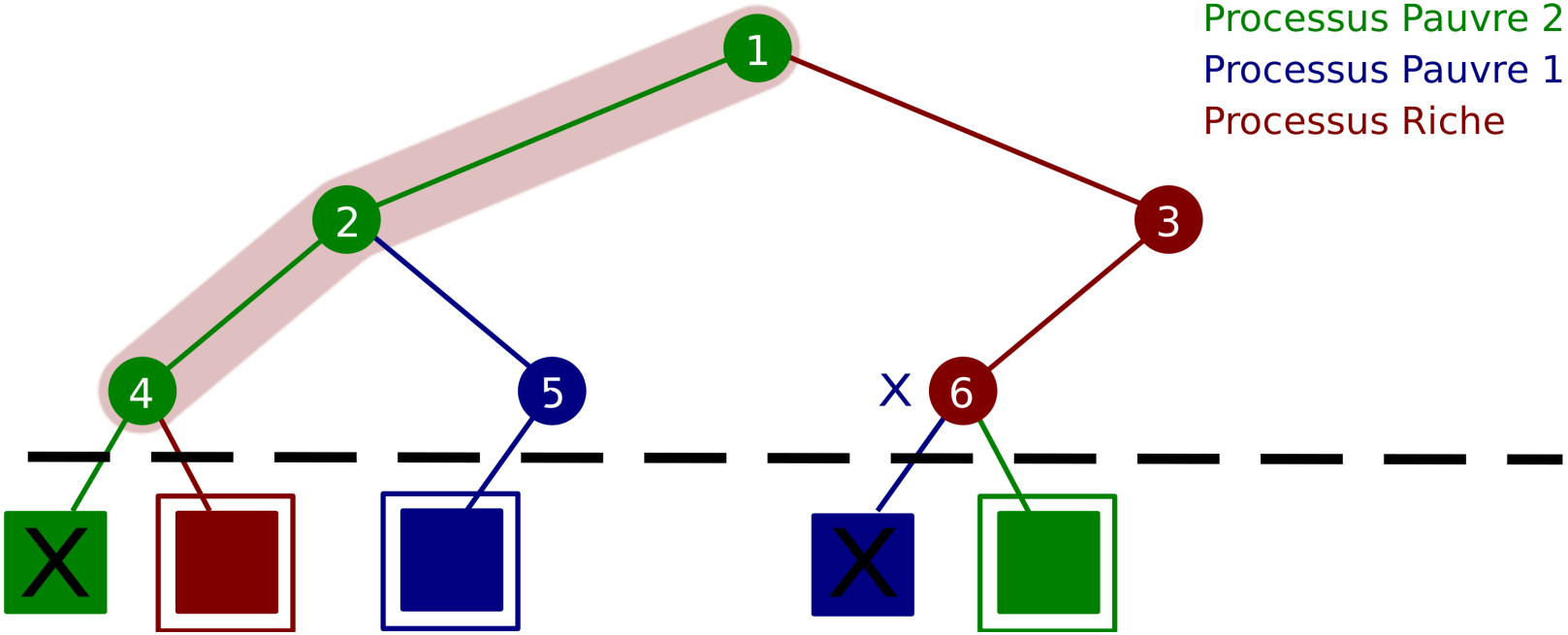
Exemple d'exécution



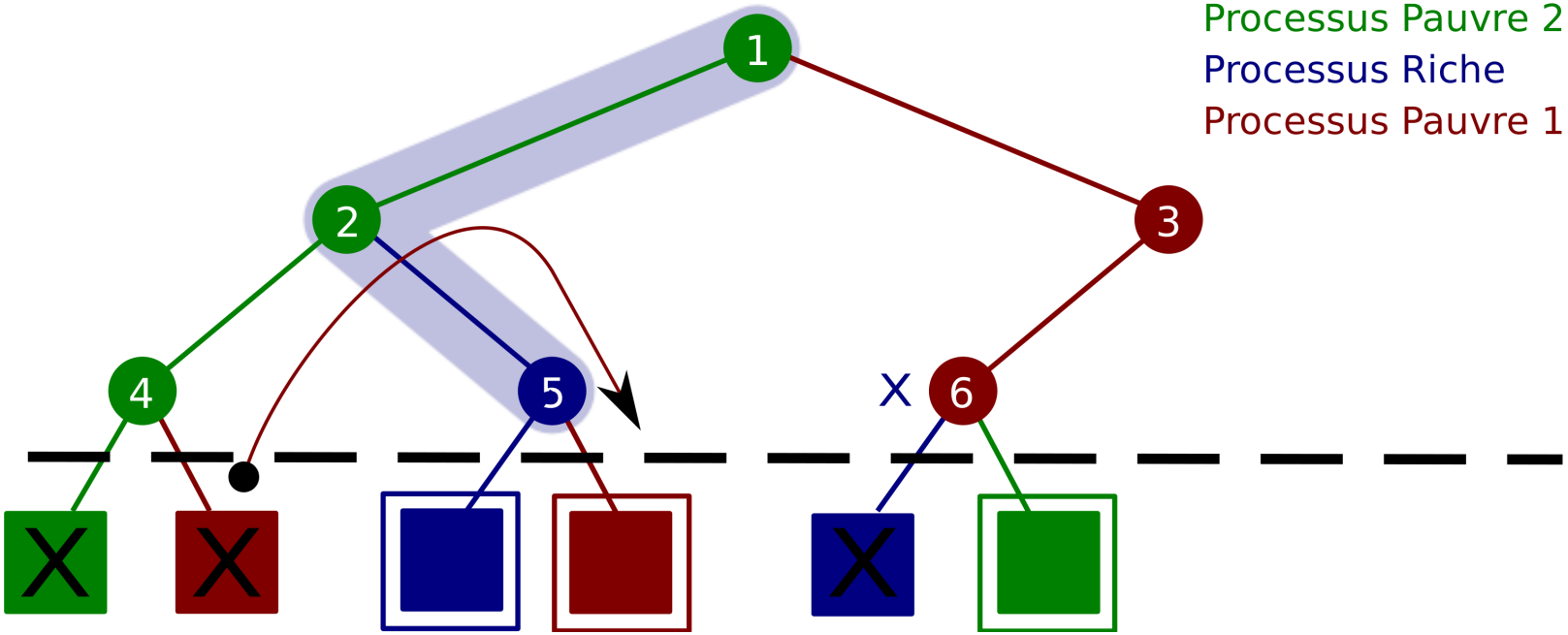
Exemple d'exécution



Exemple d'exécution



Exemple d'exécution



Plan

Machine

Benchmarks réalisés sur ROMEO II du centre de calcul régional de Champagne-Ardenne, hébergé à l'URCA

ROMEO II

- Processeurs Itanium 2 double-cœurs
- 96 cœurs de calcul
- 6 nœuds CC-UMA de 8 cœurs
- 2 nœuds CC-NUMA (16 et 32 cœurs)
- Minimum de 2 Go de RAM par cœur
- OS : GNU/Linux Red Hat retravaillée par Bull
- Compilateur ICC 10.1 (avec OpenMP)

Formules

Formules aléatoires

- 500 variables, 3-SAT
- Pic de difficulté
- Insatisfaisables

Formules industrielles

- SAT-Race 2008
- 28 formules satisfaisables
- 28 formules insatisfaisables

Formules

Formules aléatoires

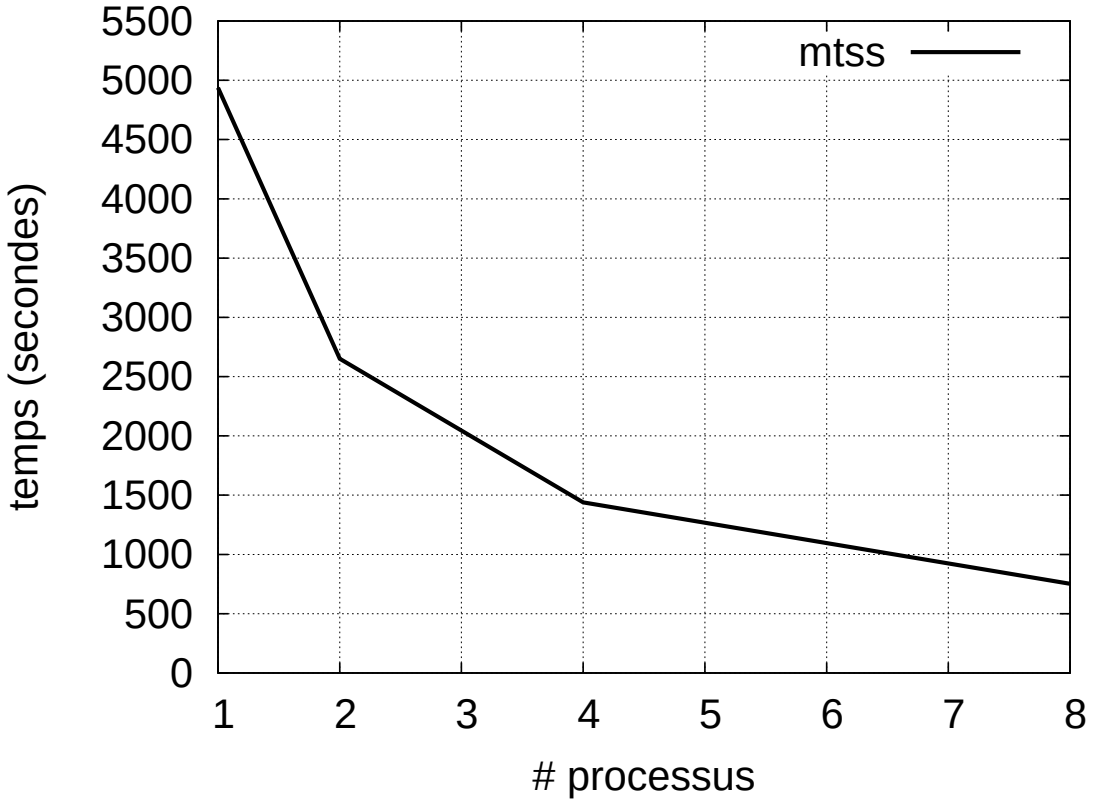
- 500 variables, 3-SAT
- Pic de difficulté
- Insatisfaisables

Formules industrielles

- SAT-Race 2008
- 28 formules satisfaisables
- 28 formules insatisfaisables

Plan

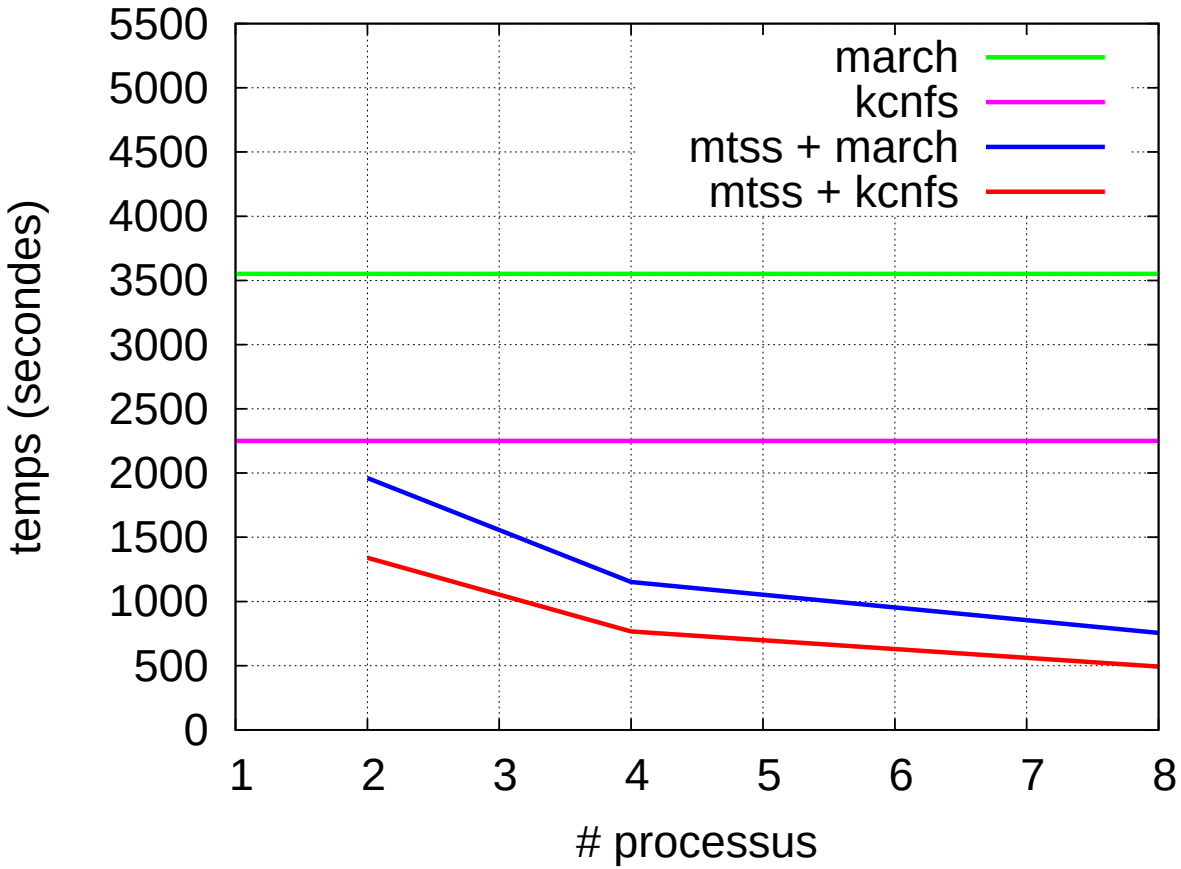
Temps de calcul



Efficacité de 80 % sur 8 cœurs de calcul (8 processus)

Plan

Solveurs pour formules aléatoires



Efficacité de 60 % avec 8 processus

Solveurs industriels

Solveur choisi

- `Minisat 2` (CDCL)
- Apprentissage de clauses
- Politique de redémarrage
- Échange de clauses à chaque redémarrage (API)

Solveurs industriels (1/2)

Résultats

Temps de calcul et accélérations de certaines formules pour 1 et 4 processus

		1 proc.	4 processus	
formule	valeur	temps (s.)	temps (s.)	accélération
ibm-2002-29r-k75.cnf	SAT	2211,06	86,17	**25,66**
ibm-2004-29-k55.cnf	SAT	1296,47	16,45	**78,81**
mizh-sha0-36-1.cnf	SAT	10576,5	1779,22	**5,94**
palac-sn7-ipc5-h16.cnf	SAT	4741,42	403,42	**11,75**
post-c32s-gcdm16-22.cnf	SAT	561,45	437,2	1,28
schup-l2s-motst-2-k315.cnf	SAT	183,87	220,64	0,83
goldb-heqc-dalumul.cnf	INSAT	11300,9	1122,68	**10,07**
manol-pipe-c9n_i.cnf	INSAT	618,2	45,18	**13,68**
manol-pipe-g10nid.cnf	INSAT	10005,5	1020,88	**9,8**
post-c32s-col400-16.cnf	INSAT	109,05	204,03	0,53
schup-l2s-abp4-1-k31.cnf	INSAT	64,61	45,42	1,42
simon-s03-fifo8-400.cnf	INSAT	185,21	167,19	1,11

Solveurs industriels (2/2)

Résultats

Temps de calcul et accélérations de certaines formules pour 8 et 16 processus

		8 processus		16 processus	
formule	valeur	temps (s.)	accélération	temps (s.)	accélération
ibm-2002-29r-k75.cnf	SAT	26,2	**84,39**	16,46	**134,33**
ibm-2004-29-k55.cnf	SAT	16,06	**80,73**	11,09	**116,9**
mizh-sha0-36-1.cnf	SAT	1777,41	5,95	398,81	**26,52**
palac-sn7-ipc5-h16.cnf	SAT	118,67	**39,95**	411,79	**11,51**
post-c32s-gcdm16-22.cnf	SAT	215,31	2,61	337,22	1,66
schup-l2s-motst-2-k315.cnf	SAT	154,32	1,19	97,61	1,88
goldb-heqc-dalumul.cnf	INSAT	853,35	**13,24**	495,48	**22,81**
manol-pipe-c9n_i.cnf	INSAT	27,36	**22,6**	20,99	**29,45**
manol-pipe-g10nid.cnf	INSAT	629,5	**15,89**	500,6	**19,99**
post-c32s-col400-16.cnf	INSAT	118,12	0,92	119,45	0,91
schup-l2s-abp4-1-k31.cnf	INSAT	36,21	1,78	27,58	2,34
simon-s03-fifo8-400.cnf	INSAT	156,23	1,19	146,46	1,26

Plan

Conclusion

Apports

- MTSS développe une approche méthodologique de résolution parallèle pour le problème SAT
- MTSS s'appuie sur la notion d'arbre de guidage : objet intrinsèquement parallèle
- MTSS repose sur le principe riche / pauvres : comportement intrinsèquement parallèle
- MTSS peut exploiter un grain fin ou plus grossier en fonction des problèmes à traiter

Performances

- Bonne efficacité pour machines multi-cœurs
- Outil de parallélisation
- Accélérations super linéaires (formules satisfaisables ou non)

Conclusion

Apports

- MTSS développe une approche méthodologique de résolution parallèle pour le problème SAT
- MTSS s'appuie sur la notion d'arbre de guidage : objet intrinsèquement parallèle
- MTSS repose sur le principe riche / pauvres : comportement intrinsèquement parallèle
- MTSS peut exploiter un grain fin ou plus grossier en fonction des problèmes à traiter

Performances

- Bonne efficacité pour machines multi-cœurs
- Outil de parallélisation
- Accélérations super linéaires (formules satisfaisables ou non)

Plan

Perspectives

Théorique

- Topologie de l'arbre de guidage
- Heuristique de MTSS avec les solveurs externes
- Fonctions des pauvres
  - Techniques de pré-traitements
  - Recherche incomplète
  - ...

Architectures matérielles

- Mémoire distribuée (MPI)
- Utilisation des nouvelles architectures parallèles (GPU ou FPGA)

Perspectives

Théorique

- Topologie de l'arbre de guidage
- Heuristique de MTSS avec les solveurs externes
- Fonctions des pauvres
  - Techniques de pré-traitements
  - Recherche incomplète
  - ...

Architectures matérielles

- Mémoire distribuée (MPI)
- Utilisation des nouvelles architectures parallèles (GPU ou FPGA)

Perspectives

Théorique

- Topologie de l'arbre de guidage
- Heuristique de MTSS avec les solveurs externes
- Fonctions des pauvres
  - Techniques de pré-traitements
  - Recherche incomplète
  - ...

Architectures matérielles

- Mémoire distribuée (MPI)
- Utilisation des nouvelles architectures parallèles (GPU ou FPGA)

Merci de votre attention