Impact de la contrainte d'incompatibilité sur la complexité et l'approximation des problèmes d'ordonnancement en présence de tâches-couplées

Gilles SIMONIN

LIRMM - UM2 - Algorithmique et Performances des Réseaux (APR)

Jury composé de :

Evripidis Bampis, Professeur, Université d'Evry Val d'Essonne . . . . . Rapporteur
Nadia Brauner Vettier, Professeur, Laboratoire G-SCOP . . . . . Rapporteur
Claire Hanen, Professeur, Laboratoire d'informatique de Paris 6 (LIP6) . . . . . Examinateur
Christophe Paul, Directeur de Recherche au CNRS, LIRMM . . . . . Examinateur
Rodolphe Giroudeau, Maître de conférences, Université Montpellier II . . . . . co-Directeur de Thèse
Jean-Claude König, Professeur, Université Montpellier II . . . . . Directeur de Thèse

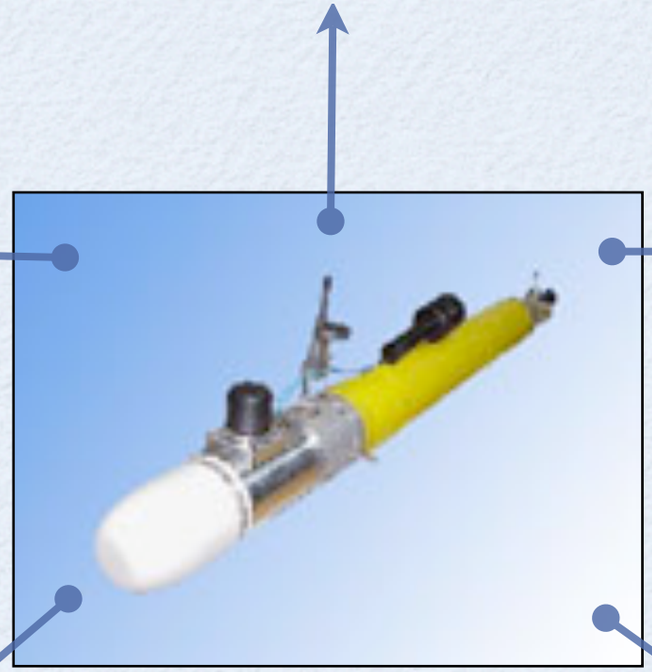
Table des matières

Introduction - présentation de notre problème

La torpille TAIPAN

Vérification des pipelines

Mesure de la salinité et température de l'eau



Détecter et analyser les sources d'eau douce

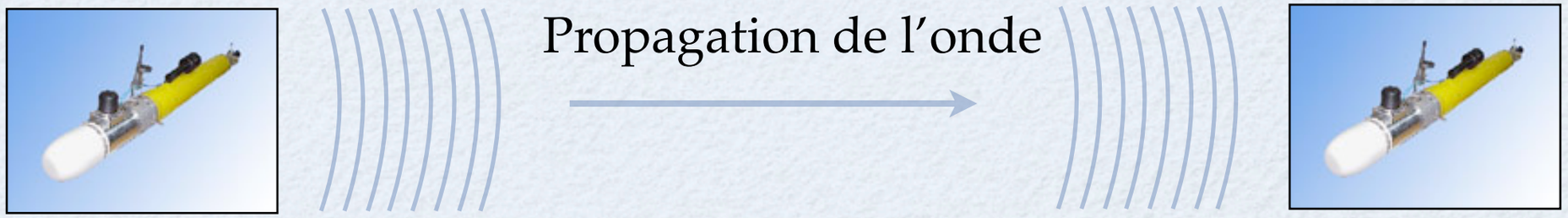
Images acoustiques de l'eau

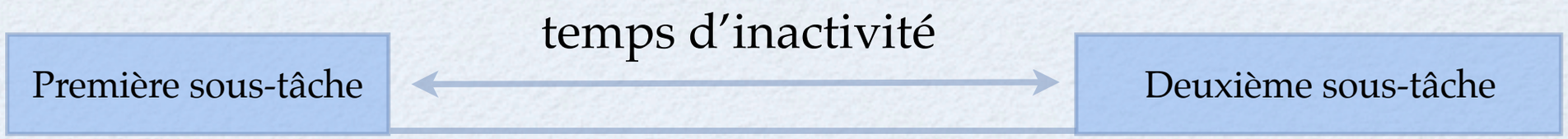
Mesure des vitesses des écoulements sous-marin

La torpille possède

- un monoprocesseur
- des capteurs
- deux ensembles de tâches : acquisition et traitement
- des contraintes de précédence entre ses tâches
- des contraintes d'incompatibilité entre des acquisitions simultanées

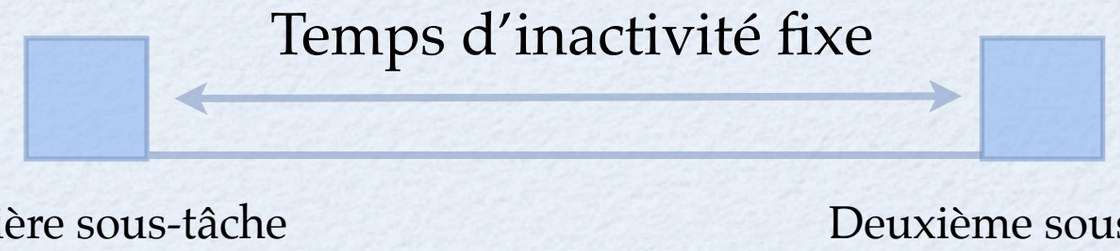
En pratique, les acquisitions s'effectuent de la manière suivante





Introduction - modélisation

Modélisation



Tâches d'acquisition ⟷ Tâches-couplées

Définition d'une tâche-couplée (Shapiro [1980])

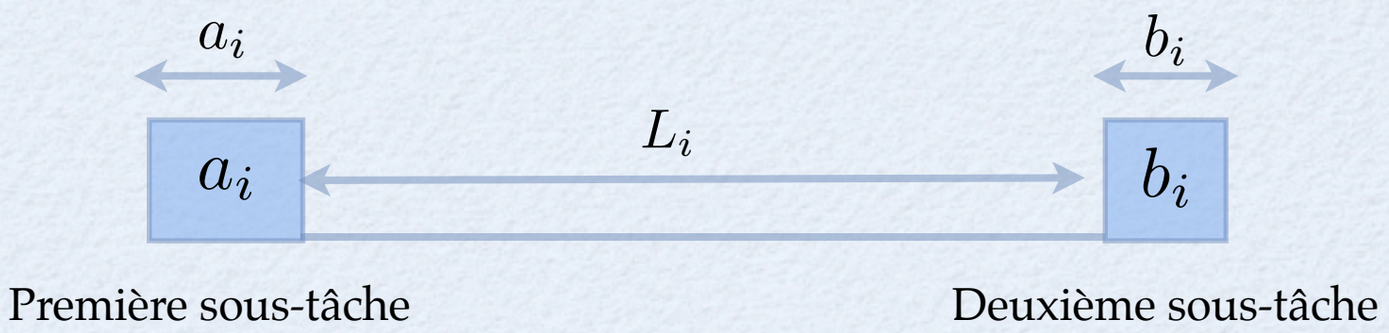
- Deux sous-tâches a_i et b_i
- Un délai d'inactivité incompressible et indilatable L_i

Introduction - modélisation

Définition d'une tâche d'acquisition

- Deux sous-tâches a_i et b_i de durée d'exécution a_i et b_i
- Un délai d'inactivité fixe incompressible et indilatable L_i

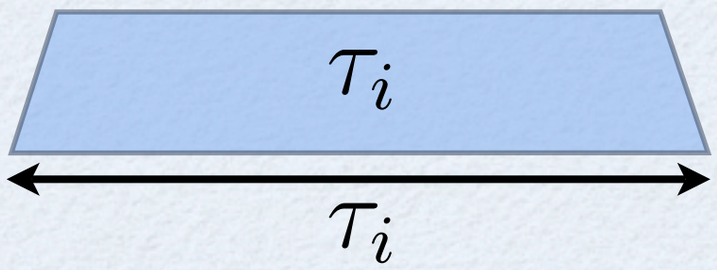
Représentation d'une tâche d'acquisition $A_i = (a_i, L_i, b_i)$



Définition des tâches de traitement

- Une durée d'exécution notée τ_i
- Elles sont préemptives

Représentation d'une tâche de traitement τ_i

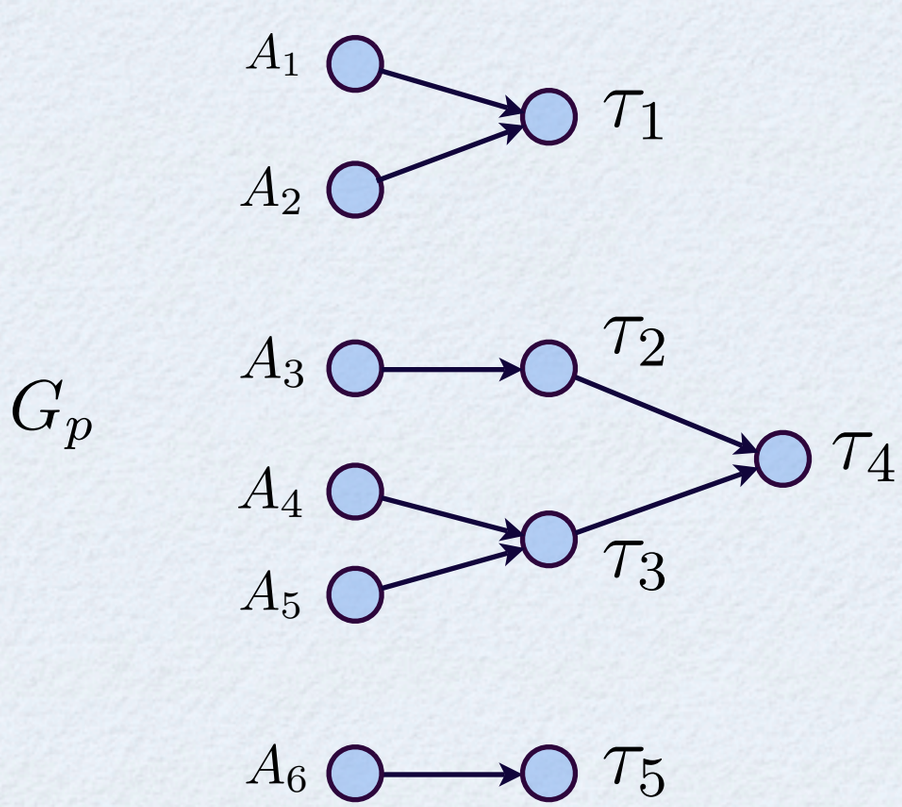


La torpille possède

- un monoprocesseur
- des capteurs
- deux ensembles de tâches : acquisition et traitement
- des contraintes de précédence entre ses tâches
- des contraintes d'incompatibilité entre des acquisitions simultanées

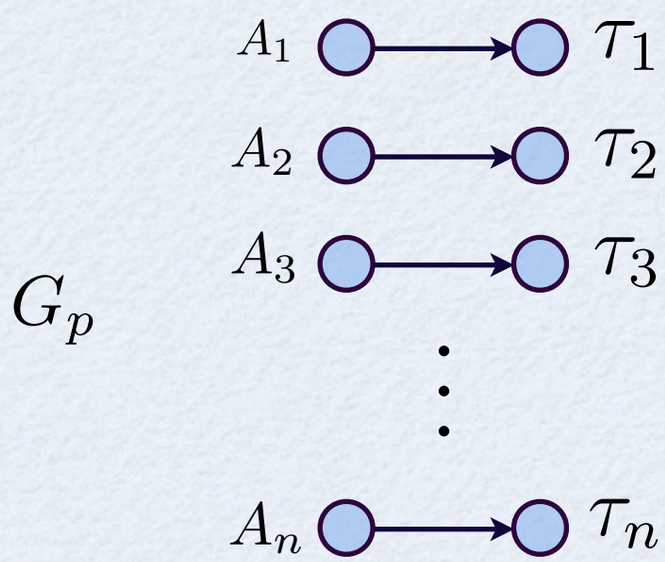
Contraintes de précédence

- $G_p = (V, E)$ un graphe orienté non connexe
- $V = \mathcal{A} \cup \mathcal{T}$ représente l'ensemble des sommets
- Les tâches d'acquisition A_i seront toujours des sources



Contraintes de précédence particulières

- $G_p = (V, E)$ un graphe orienté non connexe
- $V = \mathcal{A} \cup \mathcal{T}$ représente l'ensemble des sommets
- Les tâches d'acquisition A_i seront toujours des sources
- Pour chaque $A_i \in \mathcal{A}$, une tâche de traitement τ_i lui succède



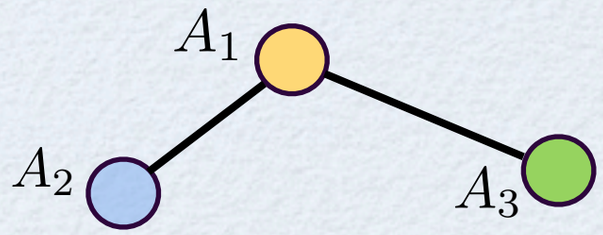
La torpille possède

- un monoprocesseur
- des capteurs
- deux ensembles de tâches : acquisition et traitement
- des contraintes de précédence entre ses tâches
- des contraintes d'incompatibilité entre des acquisitions simultanées

- Les tâches d'acquisition sont soumises à un graphe de compatibilité

Contrainte d'incompatibilité

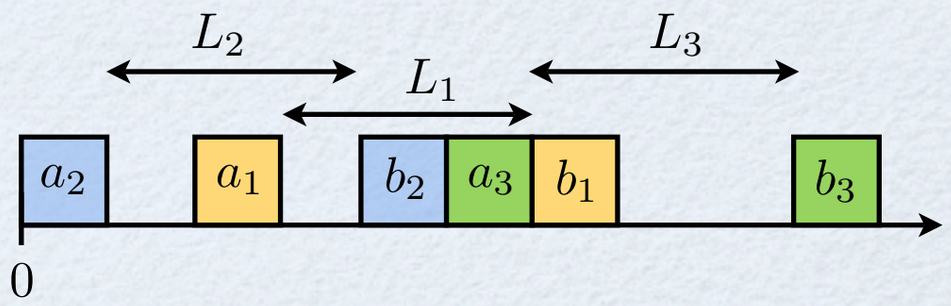
- $G_c = (\mathcal{A}, E)$ un graphe non orienté
- $\mathcal{A}$ est l'ensemble des tâches d'acquisition
- E est l'ensemble des arêtes reliant deux tâches d'acquisition compatibles



Graphe de compatibilité G_c

Tâches d'acquisition compatibles

Deux tâches d'acquisition A_i et A_j sont dites compatibles lorsque leur temps d'inactivité respectif L_i et L_j peuvent avoir lieu en même temps.



Exemple

Nos problèmes peuvent être représentés par la notation de Graham [1979] : $\alpha|\beta|\gamma$

$$1|prec, tâches - couplées, (a_i, L_i, b_i) \cup (\tau_i, pmtn), G_c|C_{max}$$

- 1 : monoprocesseur
- $prec$: contrainte de précédence
- $tâches - couplées$: modèle utilisé
- (a_i, L_i, b_i) : tâches d'acquisition
- $(\tau_i, pmtn)$: tâches de traitement
- G_c : graphe de compatibilité
- C_{max} : fonction objective, l'ordonnancement le plus court

Une notation plus simple sera utilisée ici

$$a_i, L_i, b_i \quad \text{ou} \quad a_i, L_i, b_i, G_c \quad \text{ou} \quad (a_i, L_i, b_i) \cup \tau_i, G_c$$

Introduction - objectifs

Impact de l'introduction du graphe de compatibilité et du graphe de précédence

Étude théorique : complexité / approximation

↓

Se focaliser sur les problèmes limites

Combinatoire importante des paramètres

↓

Obtenir une visualisation globale de la classification

Introduction - état de l'art

- Ordonnancement de tâches-couplées sur monoprocesseur
  - Domaines d'application :
    - Émission de signal radar : guidage d'arme, traçage d'une cible [Shapiro, Potts, Ahr]
    - Appareil utilisant des impulsions magnétiques, par exemple : TAIPAN [El Jalaoui]
    - Problème d'une cellule robotisée à une machine en mode sans attente [Brauner et al.]
  - Problèmes de complexité :
    - A.J. Orman and C.N. Potts : On the complexity of coupled-task scheduling. *Discrete Applied Mathematics*, 72 :141–154, 1997.
    - J. Blazewicz, K.H. Ecker, T. Kis, and M. Tanas. A note on the complexity of scheduling coupled-tasks on a single processor. *Journal of the Brazilian Computer Society*, 7(3) : 23–26, 2001.

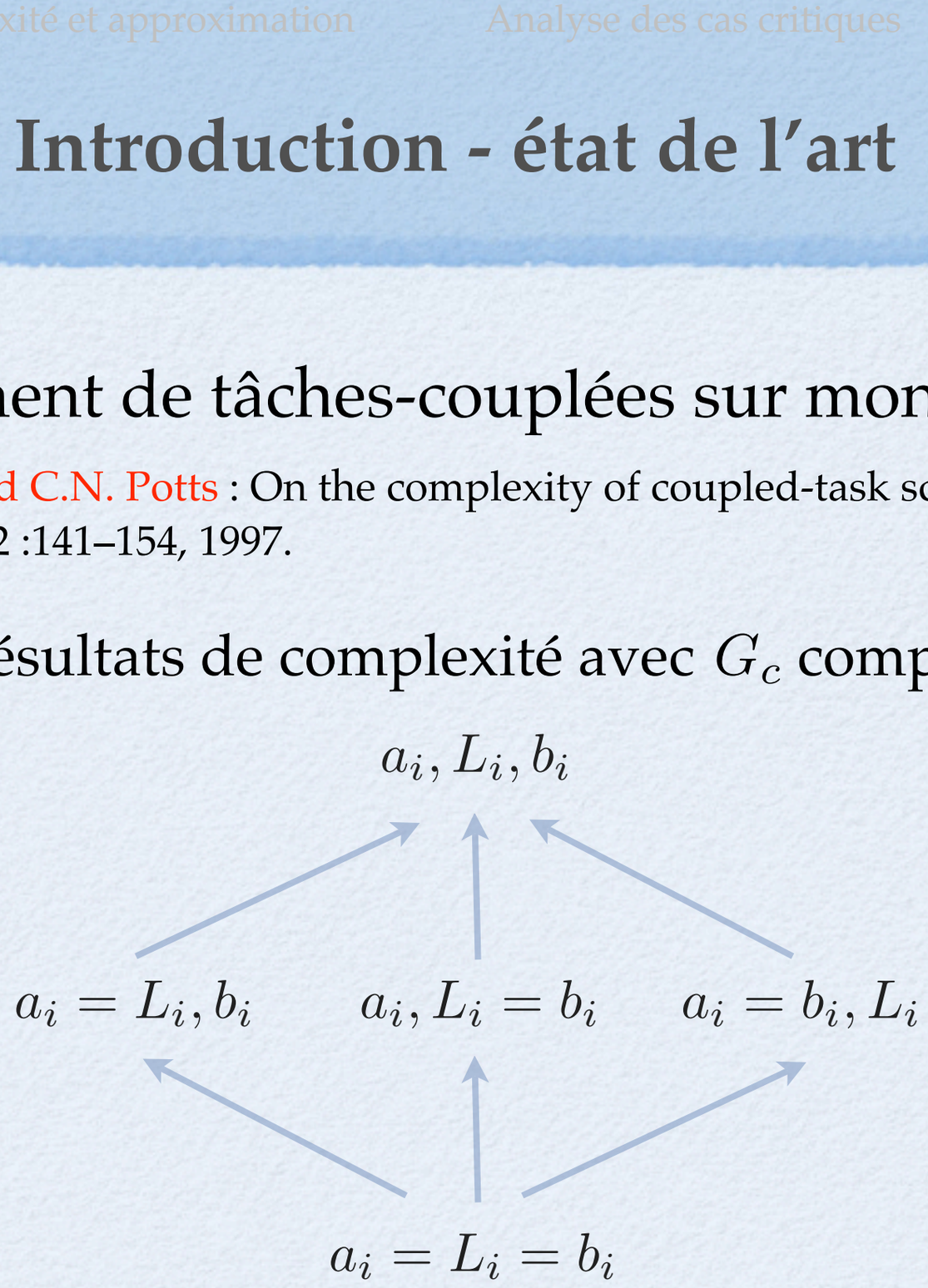
Introduction - état de l'art

- Ordonnancement de tâches-couplées sur monoprocesseur
  - Problèmes de complexité :
    - A.J. Orman and C.N. Potts : On the complexity of coupled-task scheduling. *Discrete Applied Mathematics*, 72 :141–154, 1997.
    - J. Blazewicz, K.H. Ecker, T. Kis, and M. Tanas. A note on the complexity of scheduling coupled-tasks on a single processor. *Journal of the Brazilian Computer Society*, 7(3) : 23–26, 2001.
  - Problèmes d'approximation :
    - A. A. Ageev and A. V. Kononov. Approximation algorithms for scheduling problems with exact delays. In *WAOA*, pages 1–14, 2006.
    - D. Ahr, J. Békési, G. Galambos, M. Oswald, and G. Reinelt. An exact algorithm for scheduling identical coupled-tasks. *Mathematical Methods of Operations Research*, 59 : 193–203(11), June 2004.

Introduction - état de l'art

- Ordonnancement de tâches-couplées sur monoprocesseur
  - A.J. Orman and C.N. Potts : On the complexity of coupled-task scheduling. *Discrete Applied Mathematics*, 72 :141–154, 1997.

Résultats de complexité avec G_c complet

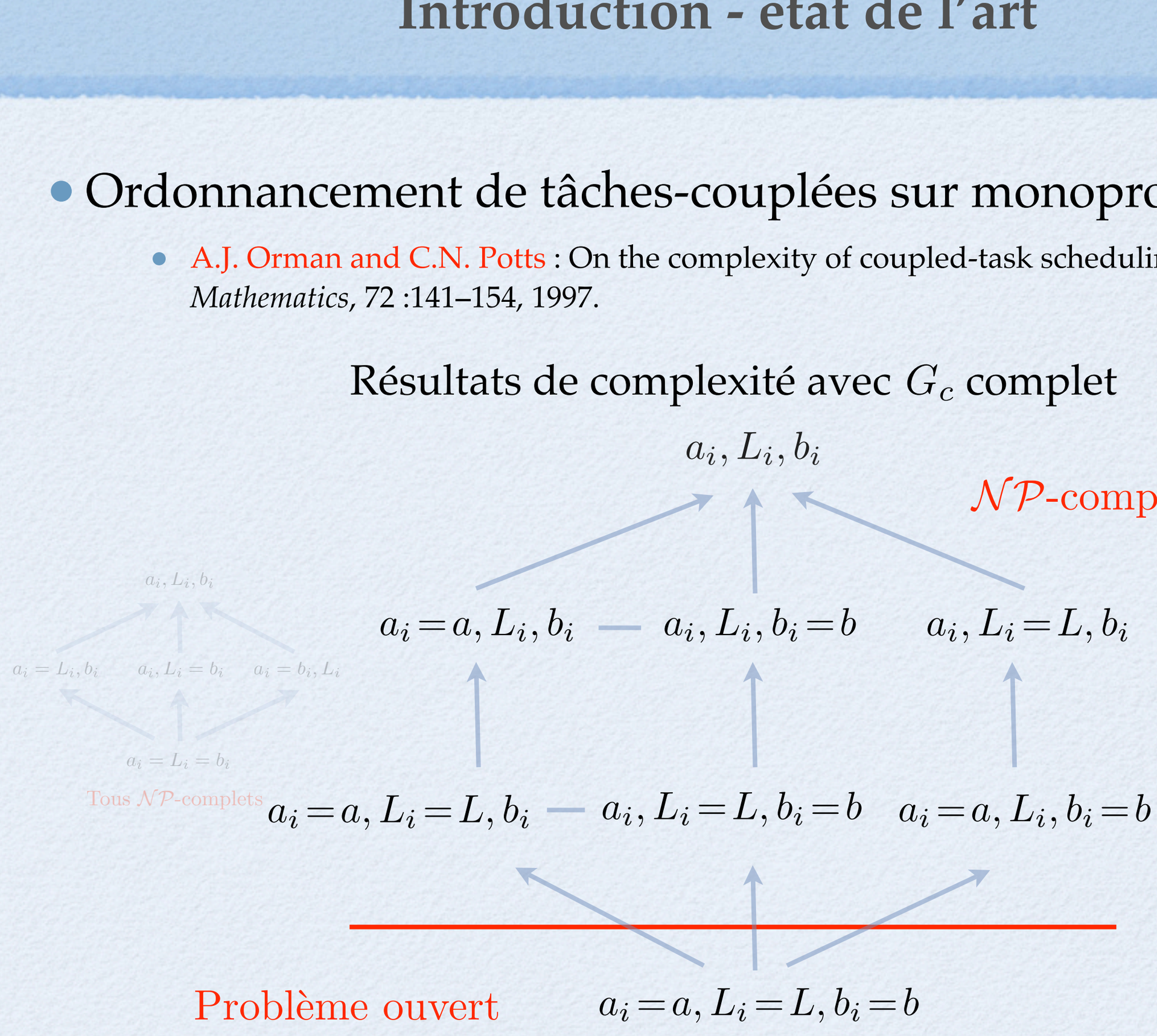


Tous $\mathcal{NP}$-complets

Introduction - état de l'art

- Ordonnancement de tâches-couplées sur monoprocesseur
  - A.J. Orman and C.N. Potts : On the complexity of coupled-task scheduling. *Discrete Applied Mathematics*, 72 :141–154, 1997.

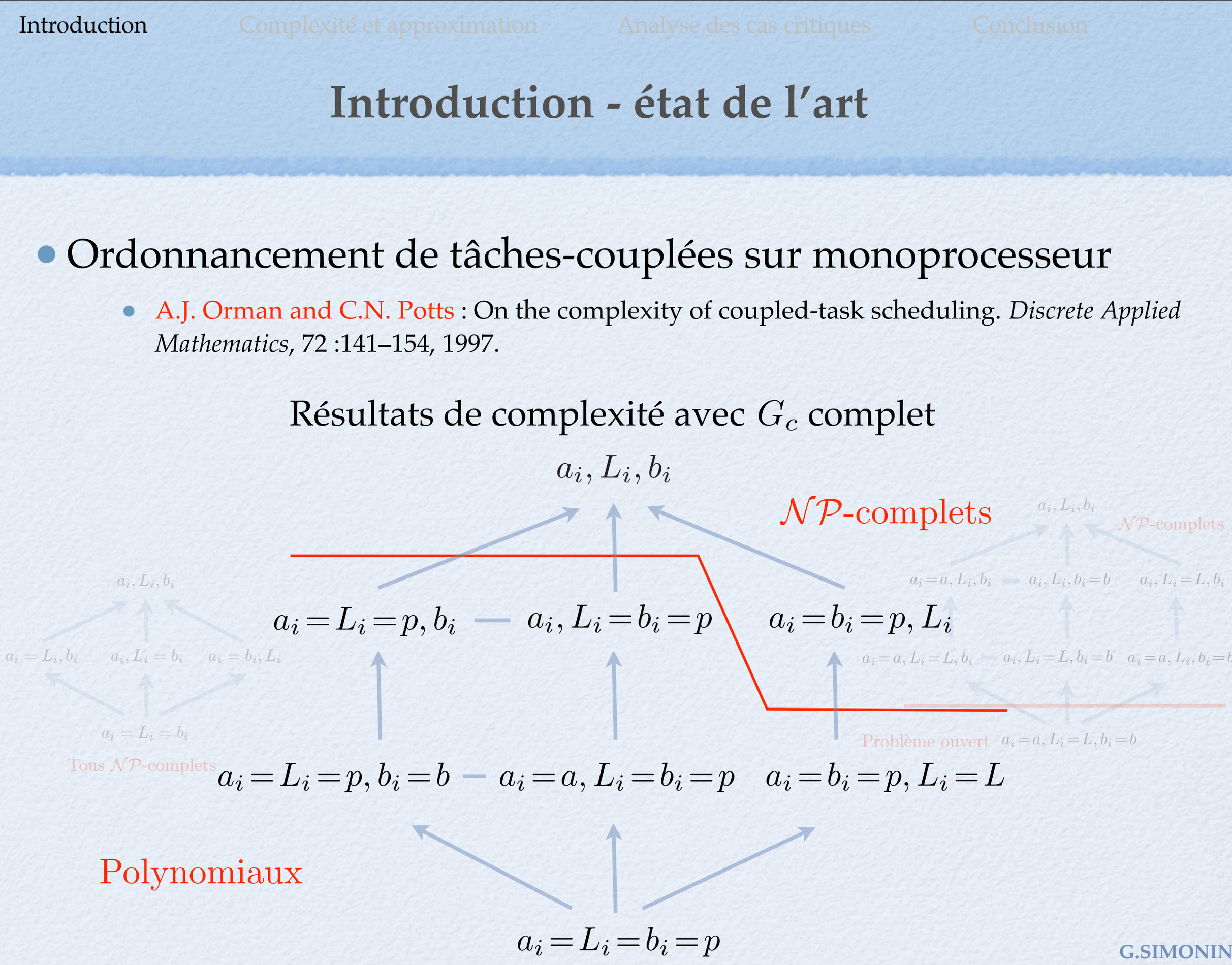
Résultats de complexité avec G_c complet



Introduction - état de l'art

- Ordonnancement de tâches-couplées sur monoprocesseur
  - A.J. Orman and C.N. Potts : On the complexity of coupled-task scheduling. *Discrete Applied Mathematics*, 72 :141–154, 1997.

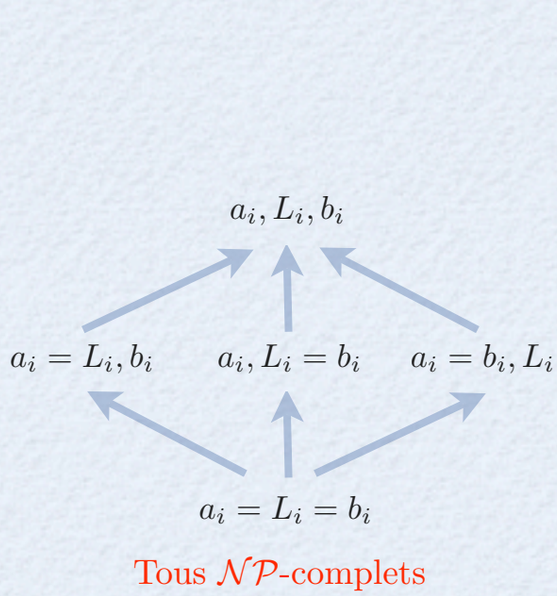
Résultats de complexité avec G_c complet

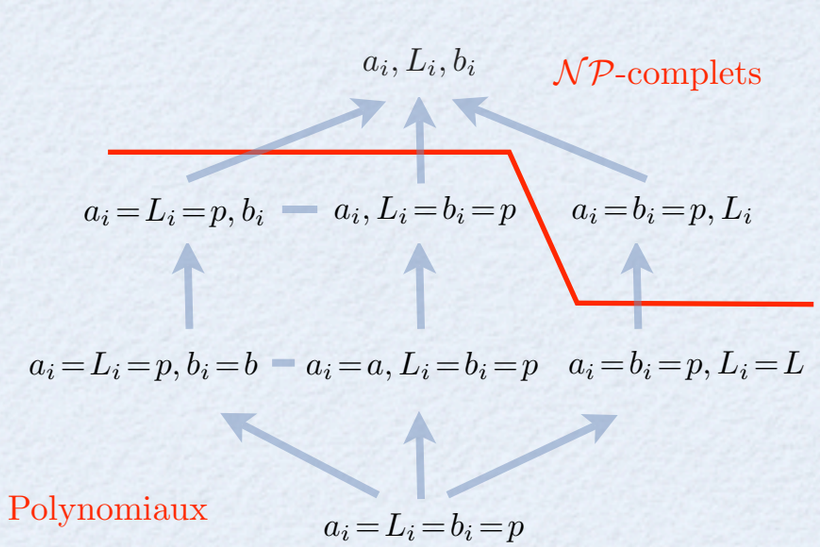


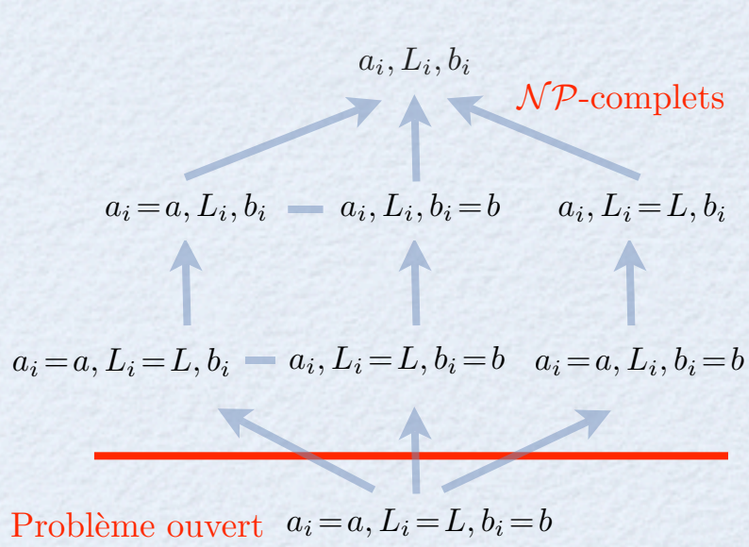
Introduction - état de l'art

- Ordonnancement de tâches-couplées sur monoprocesseur
  - A.J. Orman and C.N. Potts : On the complexity of coupled-task scheduling. *Discrete Applied Mathematics*, 72 :141–154, 1997.

Résultats de complexité avec G_c complet







Introduction - état de l'art

- Ordonnancement de tâches-couplées sur monoprocesseur
  - A.J. Orman and C.N. Potts : On the complexity of coupled-task scheduling. *Discrete Applied Mathematics*, 72 :141–154, 1997.

Résultats de complexité avec G_c complet

Spécification

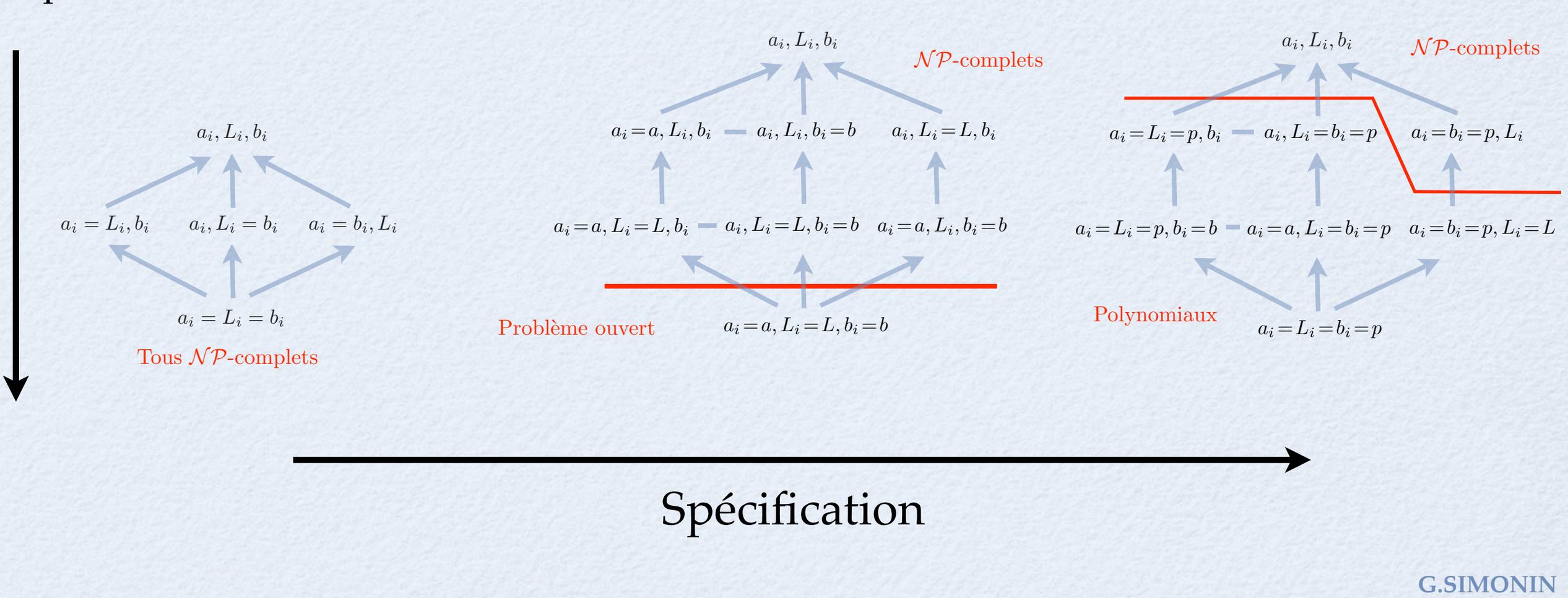


Table des matières

Avec graphe de compatibilité - objectifs

- Type de problème
  - n tâches d'acquisition
  - Un graphe de compatibilité G_c quelconque
  - Différents problèmes selon les paramètres (a_i, L_i, b_i)
- Nos objectifs
  - Classifier chaque problème rencontré
  - Développer des algorithmes d'approximation

Avec graphe de compatibilité - complexité

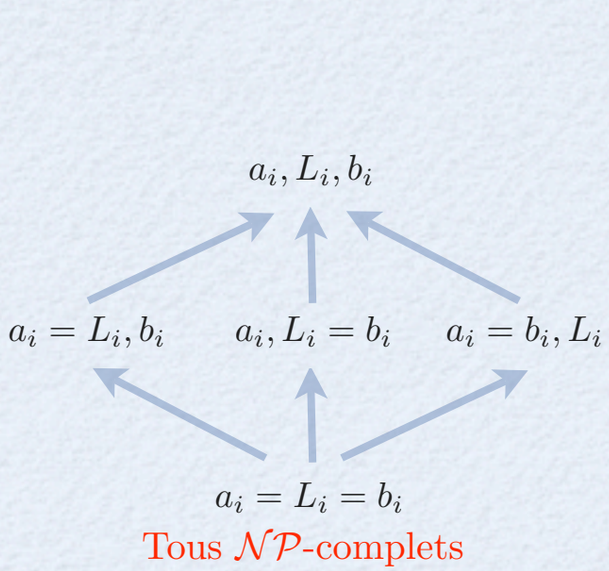
Résultats de complexité par Orman et Potts

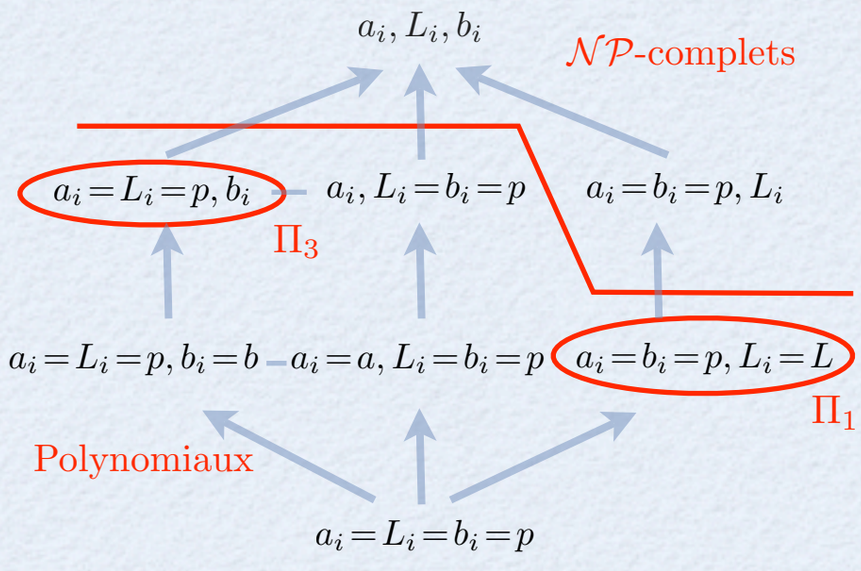
Avec graphe de compatibilité - complexité

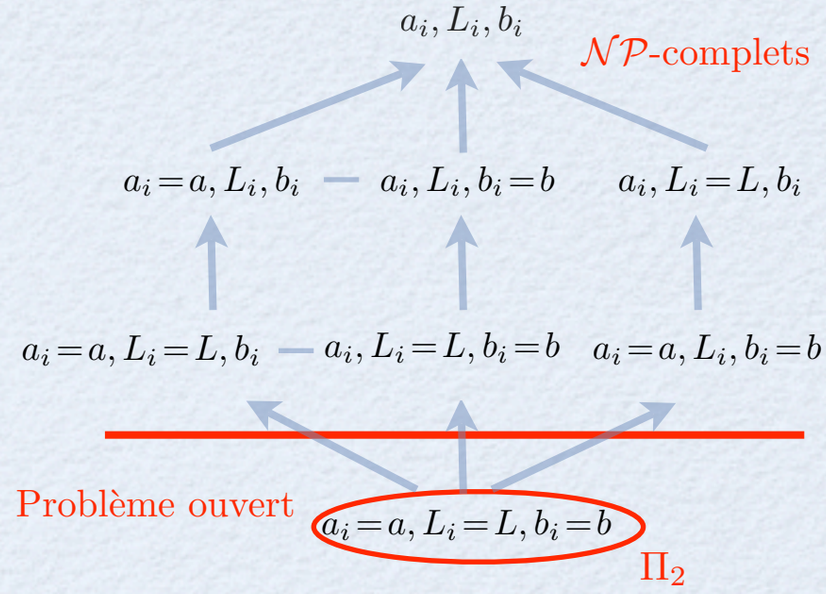
Résultats de complexité par Orman et Potts

Avec graphe de compatibilité - complexité

Résultats de complexité par Orman et Potts







Quelle est la complexité avec la relaxation de G_c ?

Trois cas intéressants

Avec graphe de compatibilité : problème Π_1

Étude du problème Π_1

$$(a_i = b_i = p, L_i = L), G_c$$

Étude d'un cas particulier

$$(a_i = b_i = p, L_i = 2p), G_c$$

Nous le noterons Π'_1

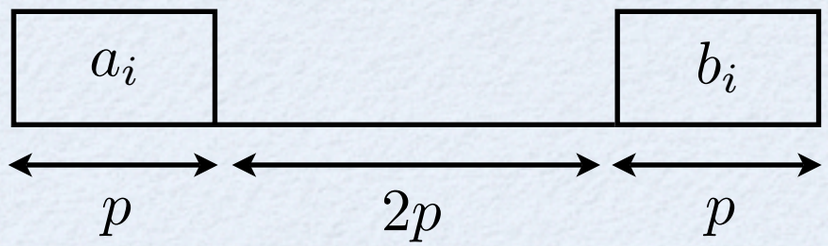
Avec graphe de compatibilité : problème Π_1

Étude d'un cas particulier

$$(a_i = b_i = p, L_i = 2p), G_c$$

Nous le noterons Π_1'

- Toutes les tâches d'acquisition sont de la forme

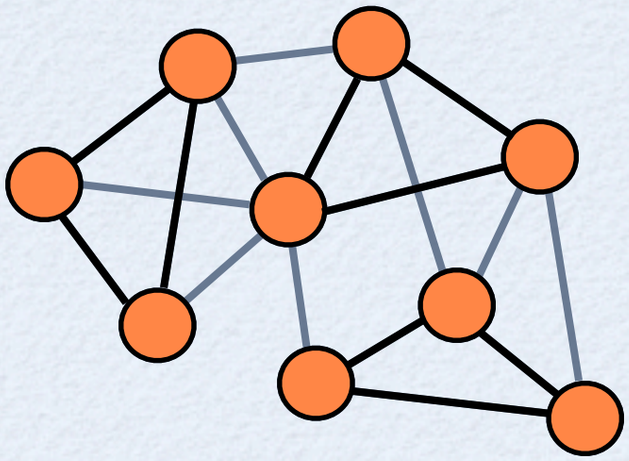


- Montrons que notre problème est $\mathcal{NP}$-complet

Avec graphe de compatibilité : problème Π_1

- Montrons que notre problème est $\mathcal{NP}$-complet ($C_{max} = 2np$)

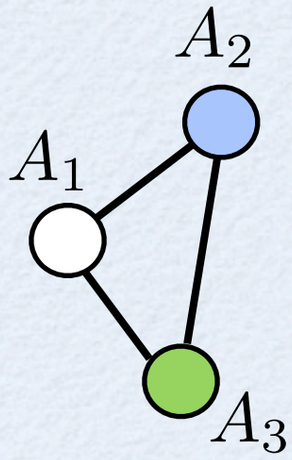
$$\text{TRIANGLE PACKING} \quad \propto \quad (a_i = b_i = p, L_i = 2p), G_c$$



Graphe G

$\downarrow$

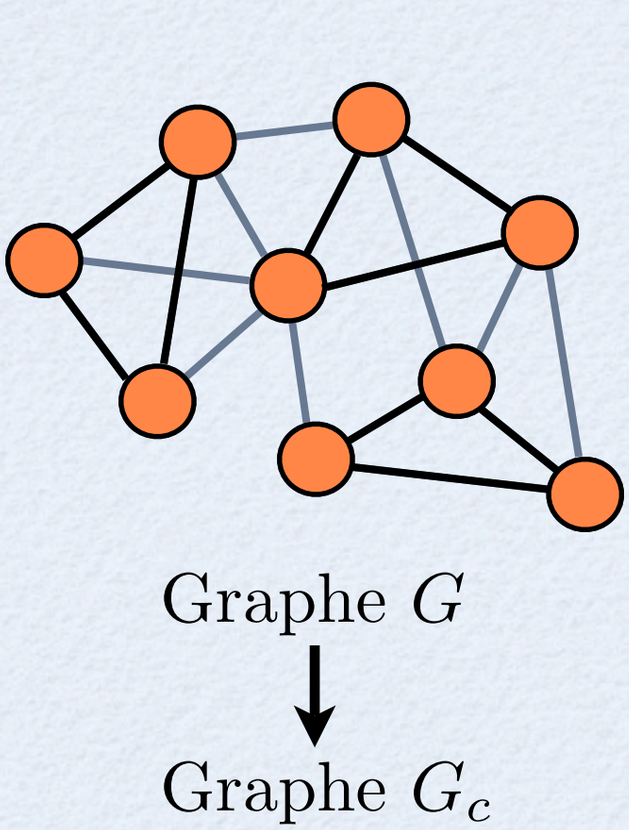
Graphe G_c

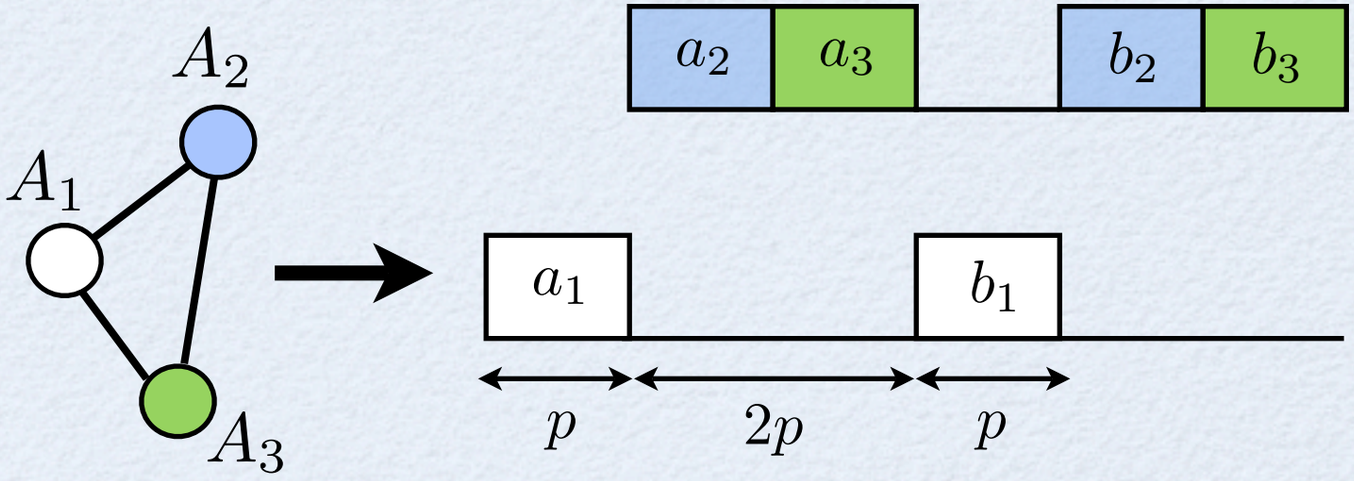


Avec graphe de compatibilité : problème Π_1

- Montrons que notre problème est $\mathcal{NP}$-complet ($C_{max} = 2np$)

$$\text{TRIANGLE PACKING} \quad \propto \quad (a_i = b_i = p, L_i = 2p), G_c$$

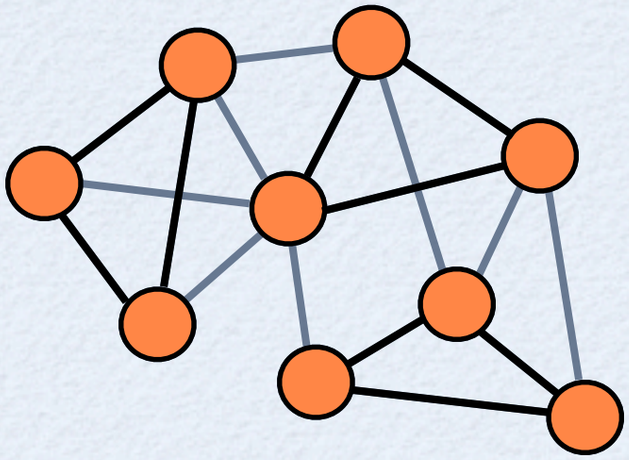




Avec graphe de compatibilité : problème Π_1

- Montrons que notre problème est $\mathcal{NP}$-complet ($C_{max} = 2np$)

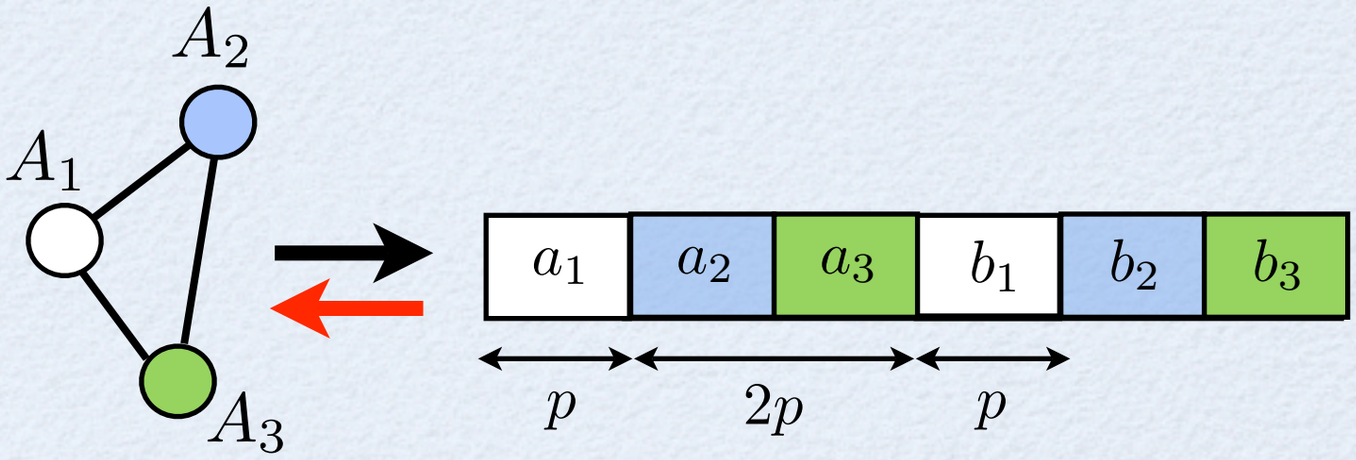
$$\text{TRIANGLE PACKING} \quad \propto \quad (a_i = b_i = p, L_i = 2p), G_c$$



Graphe G

$\downarrow\uparrow$

Graphe G_c



□

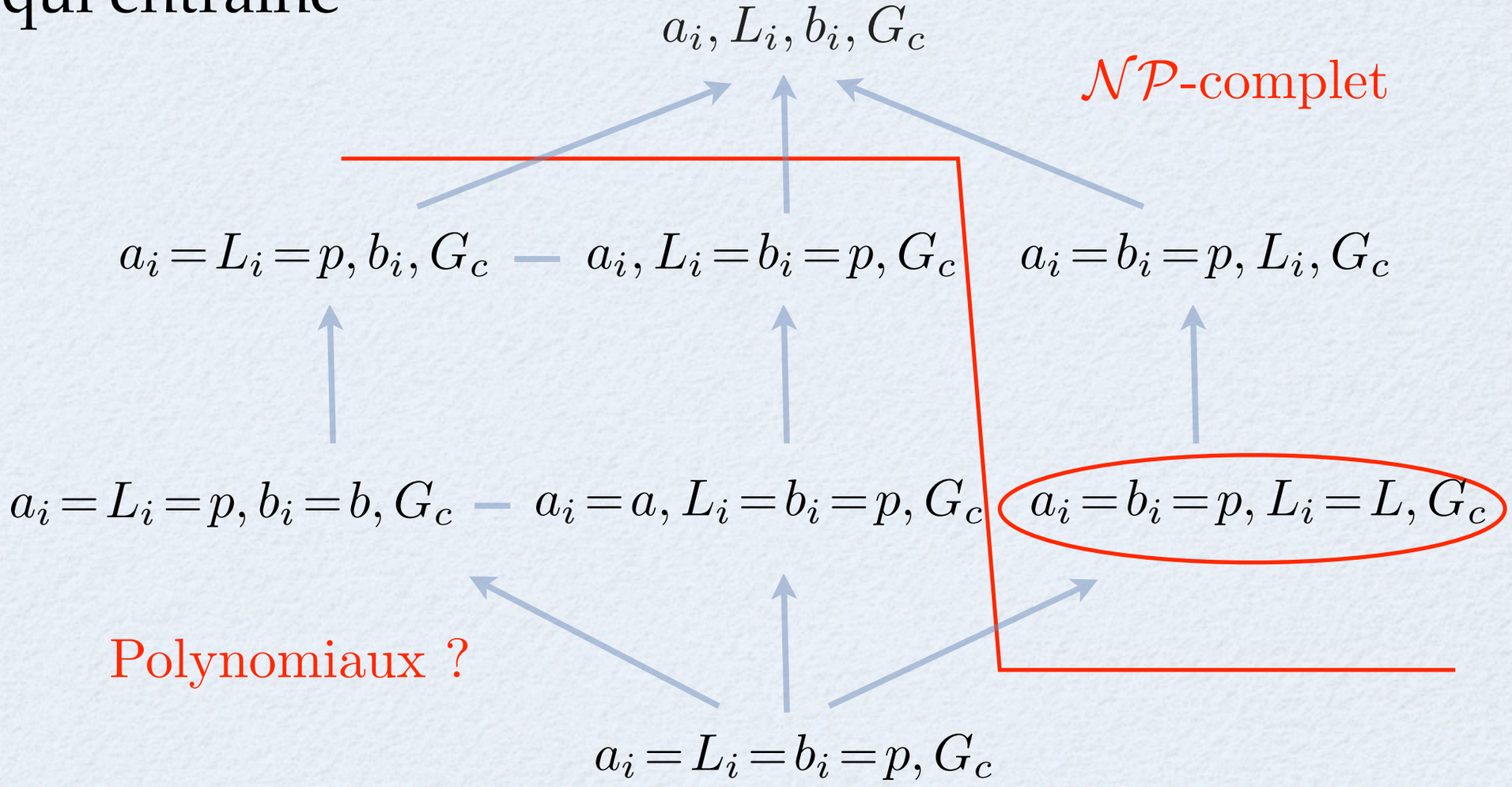
Ainsi

$$(a_i = b_i = p, L_i = 2p), G_c \quad \mathcal{NP}\text{-complet}$$

$$\downarrow$$

$$(a_i = b_i = p, L_i = L), G_c \quad \mathcal{NP}\text{-complet}$$

Ce qui entraîne



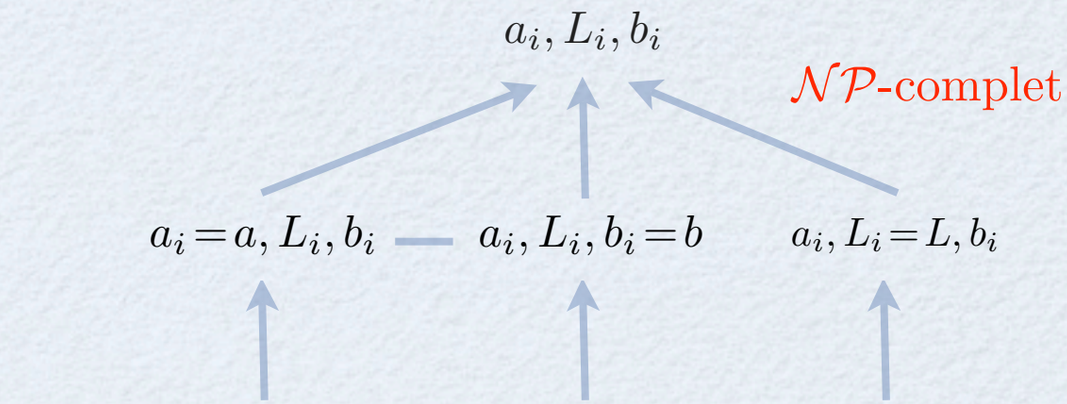
Ainsi

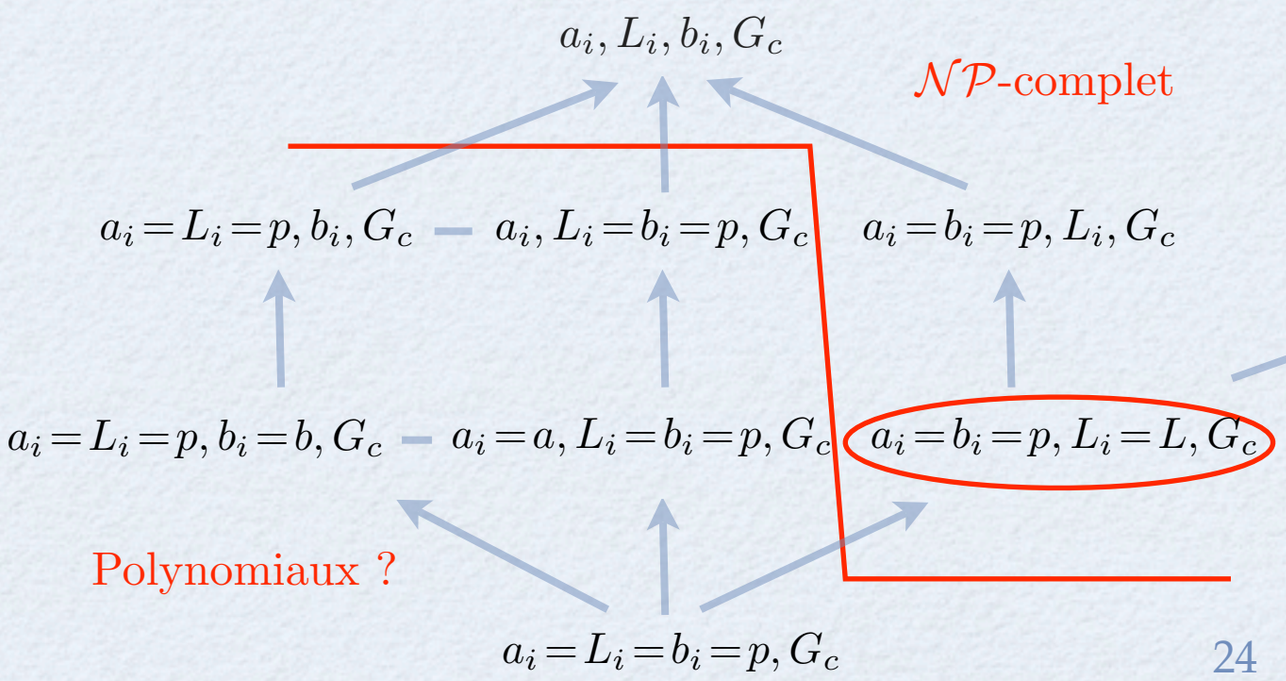
$$(a_i = b_i = p, L_i = 2p), G_c \quad \mathcal{NP}\text{-complet}$$

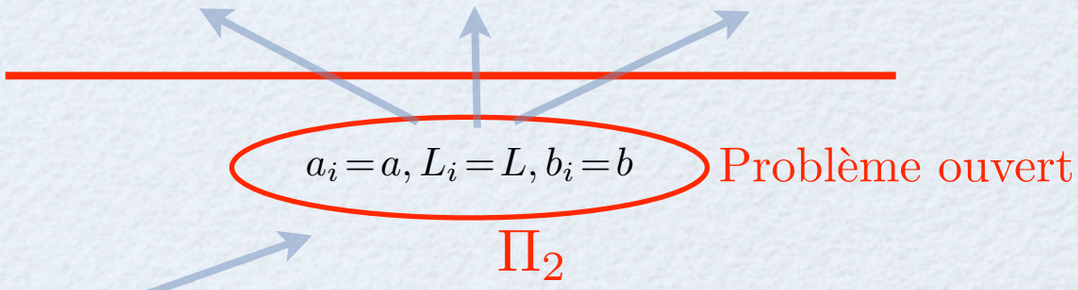
$$\downarrow$$

$$(a_i = b_i = p, L_i = L), G_c \quad \mathcal{NP}\text{-complet}$$

Ce qui entraîne







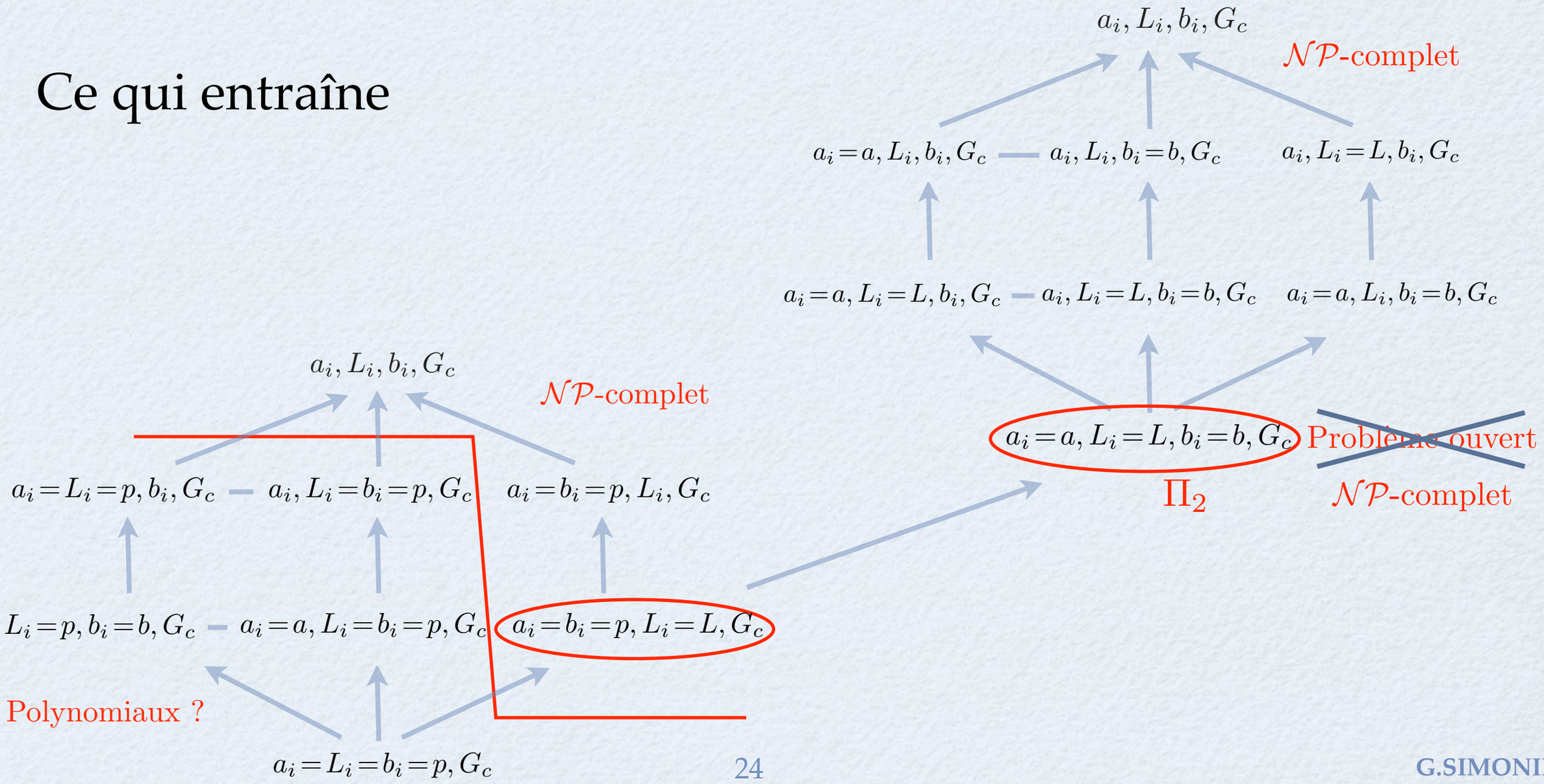
Ainsi

$$(a_i = b_i = p, L_i = 2p), G_c \quad \mathcal{NP}\text{-complet}$$

$$\downarrow$$

$$(a_i = b_i = p, L_i = L), G_c \quad \mathcal{NP}\text{-complet}$$

Ce qui entraîne



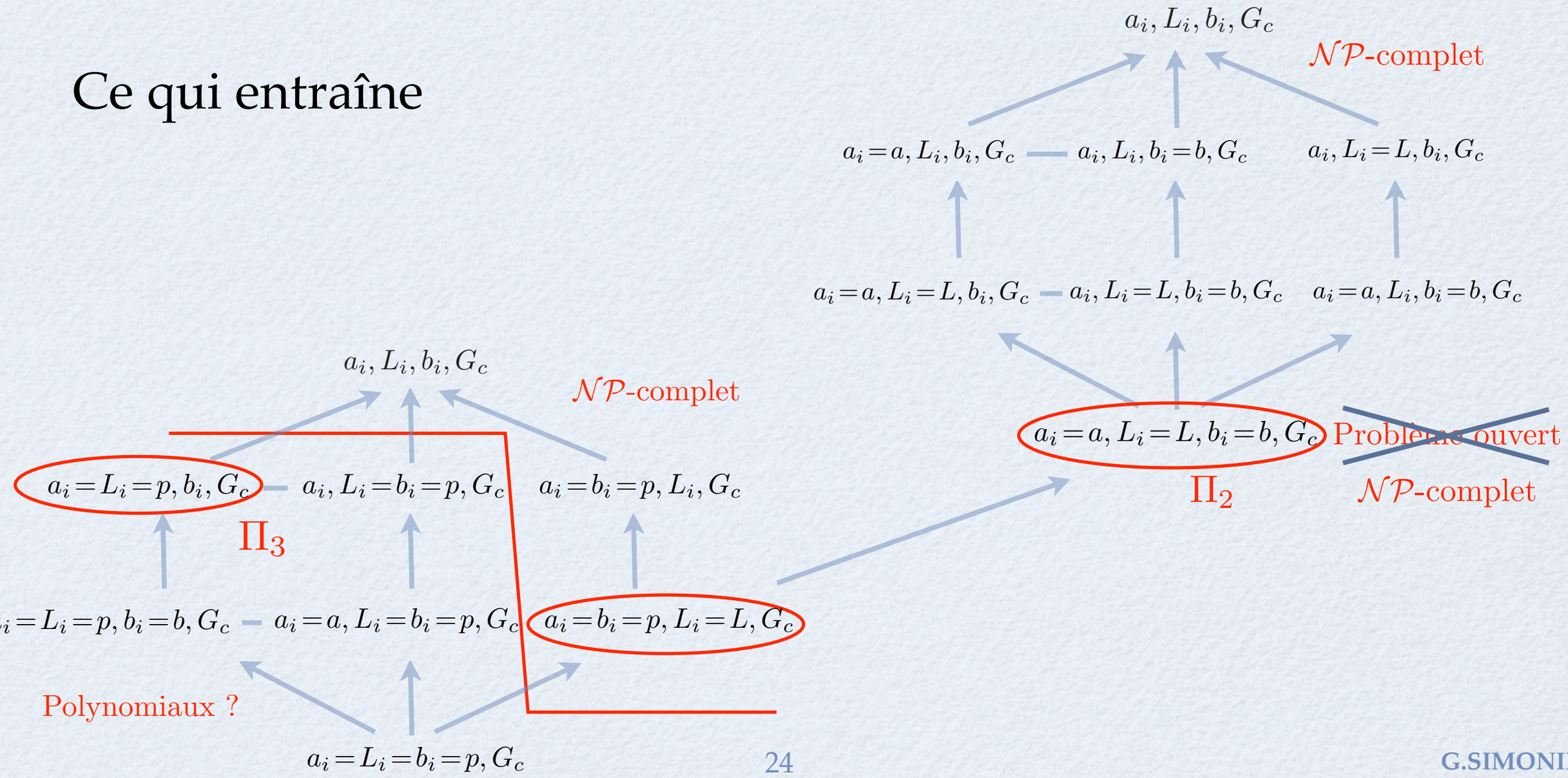
Ainsi

$$(a_i = b_i = p, L_i = 2p), G_c \quad \mathcal{NP}\text{-complet}$$

$$\downarrow$$

$$(a_i = b_i = p, L_i = L), G_c \quad \mathcal{NP}\text{-complet}$$

Ce qui entraîne

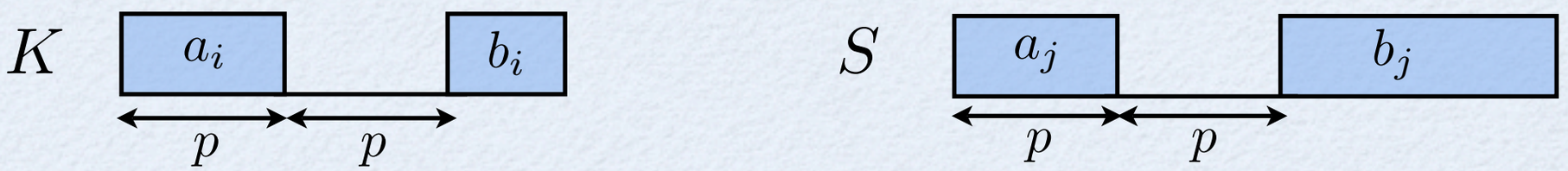


Avec graphe de compatibilité : problème Π_3

Étude du problème Π_3

$$(a_i = L_i = p, b_i), G_c$$

- Toutes les tâches d'acquisition sont de la forme



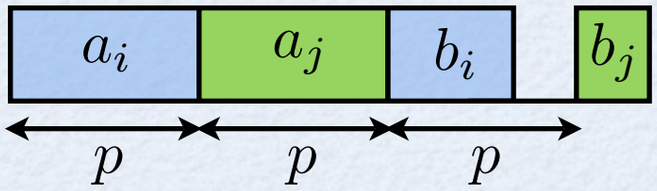
- Soit K l'ensemble des tâches d'acquisition telles que $b_i \leq p$
- Soit S l'ensemble des tâches d'acquisition telles que $b_i > p$

Avec graphe de compatibilité : problème Π_3

- Trois types d'ordonnancement possible

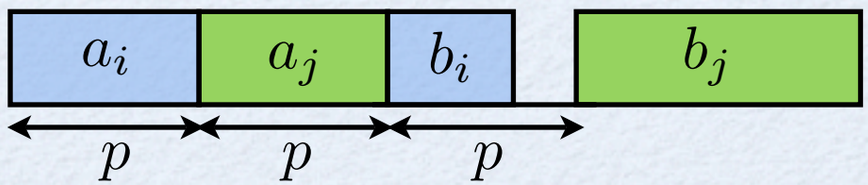
tâche première — tâche seconde

$K - K$



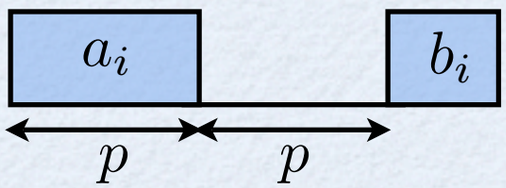
$\longrightarrow 3p + \min\{b_i, b_j\}$

$K - S$



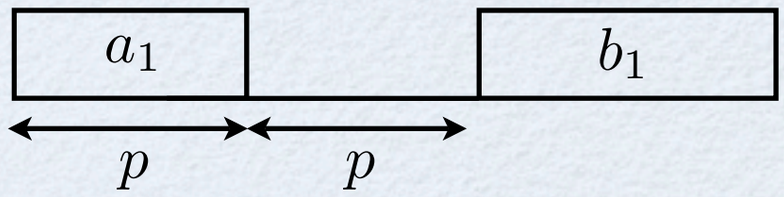
$\longrightarrow 3p + \max\{b_i, b_j\}$

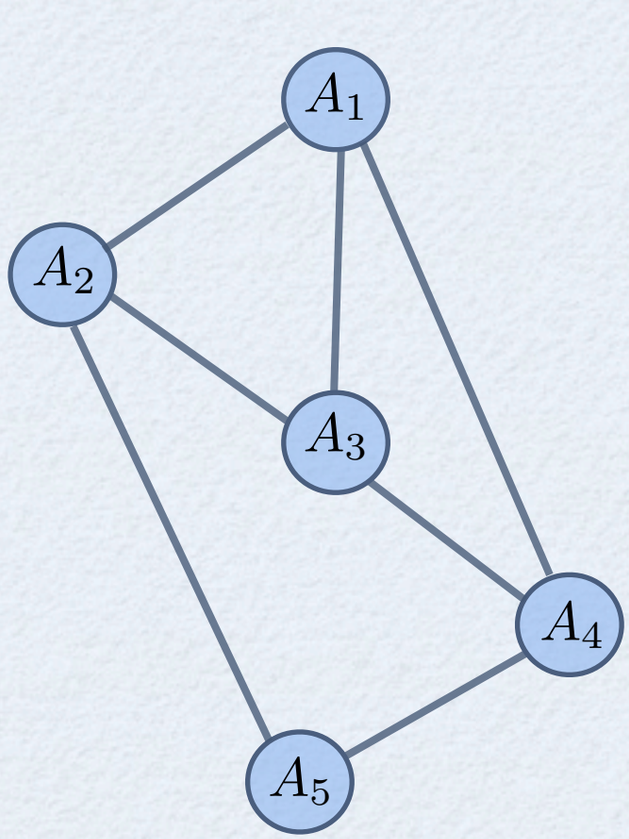
K ou S



$\longrightarrow 2p + b_i$

Utilisation de l'algorithme sur un exemple : $p = 5$

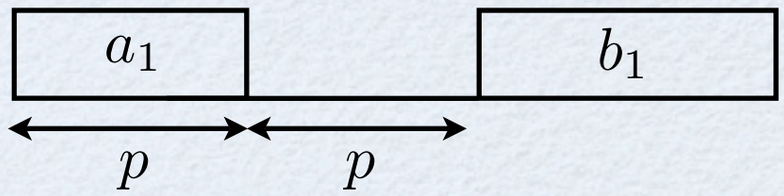


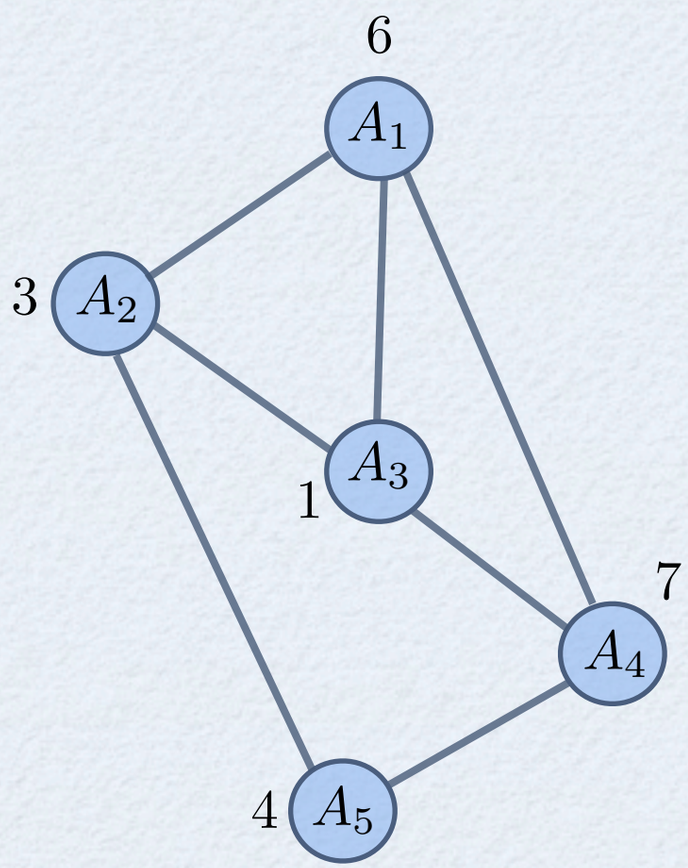


Graphe G_c

Utilisation de l'algorithme sur un exemple : $p = 5$

1) Pondérer chaque sommet par les b_i

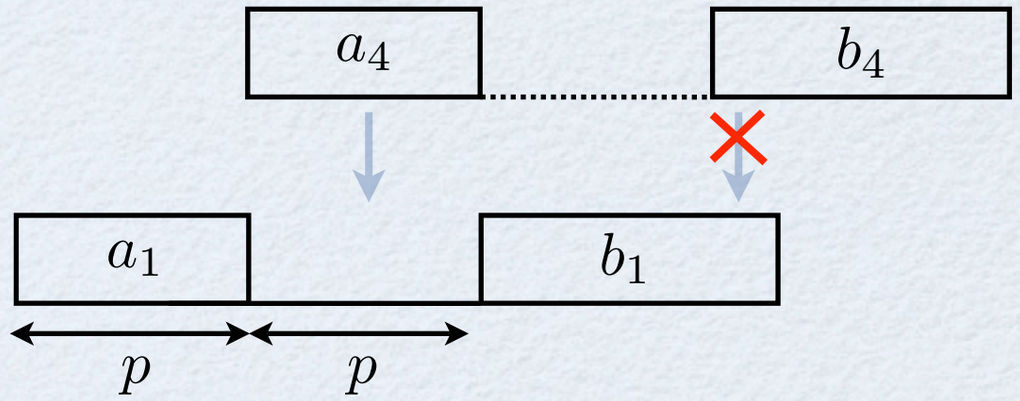


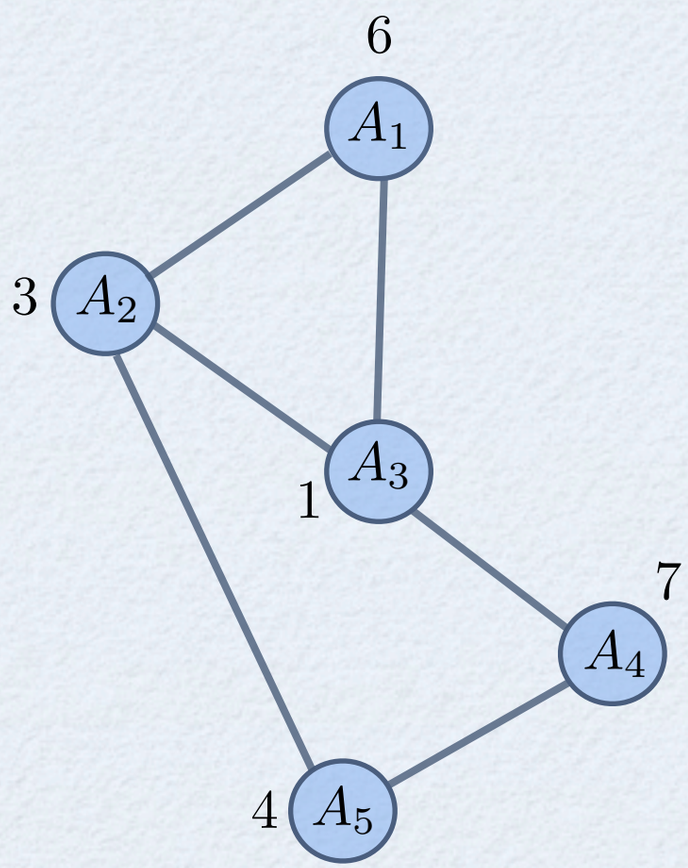


Graphe G_c

Utilisation de l'algorithme sur un exemple : $p = 5$

1) Pondérer chaque sommet par les b_i
2) Enlever les arêtes inutiles

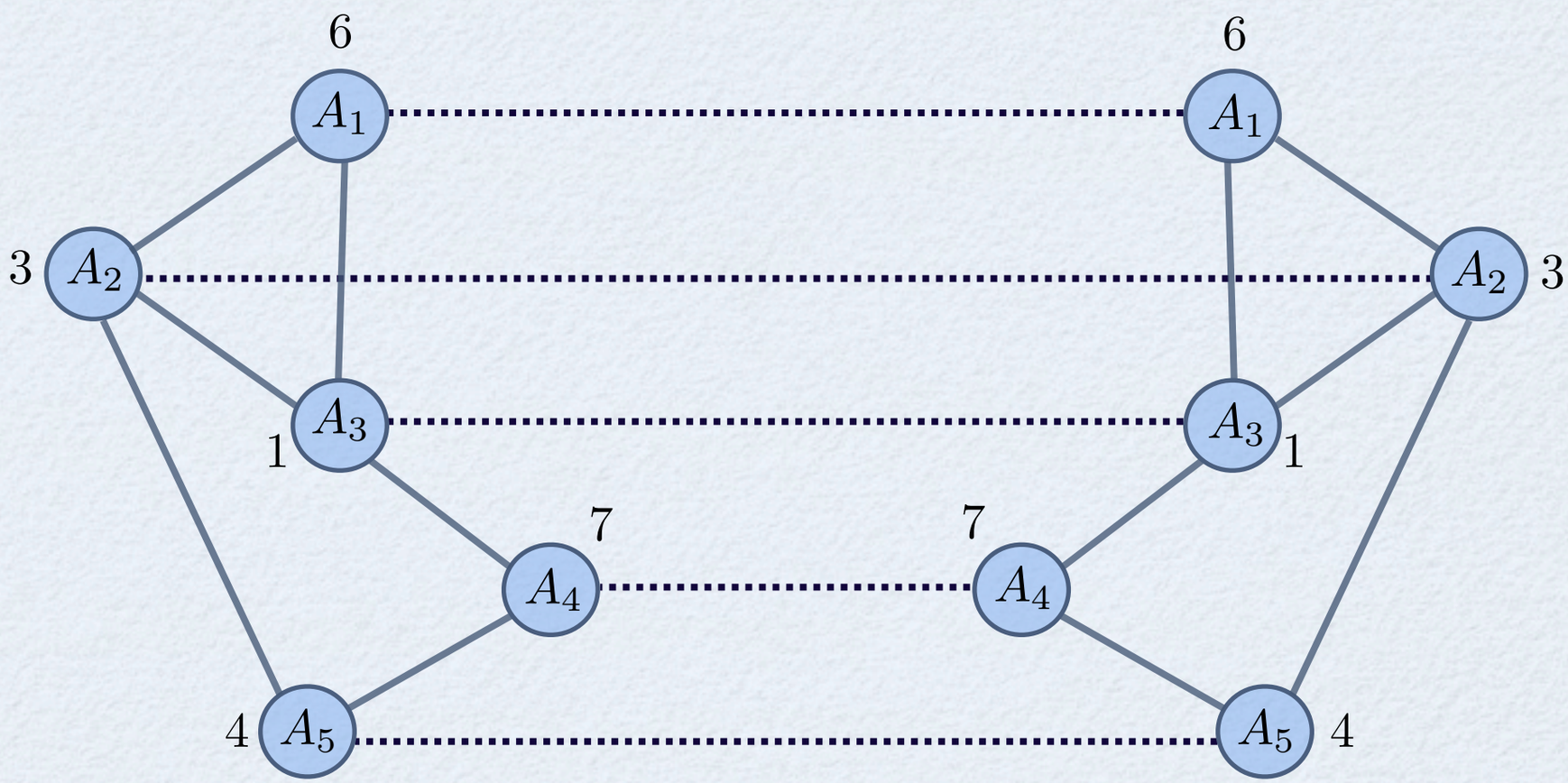




Graphe G_c

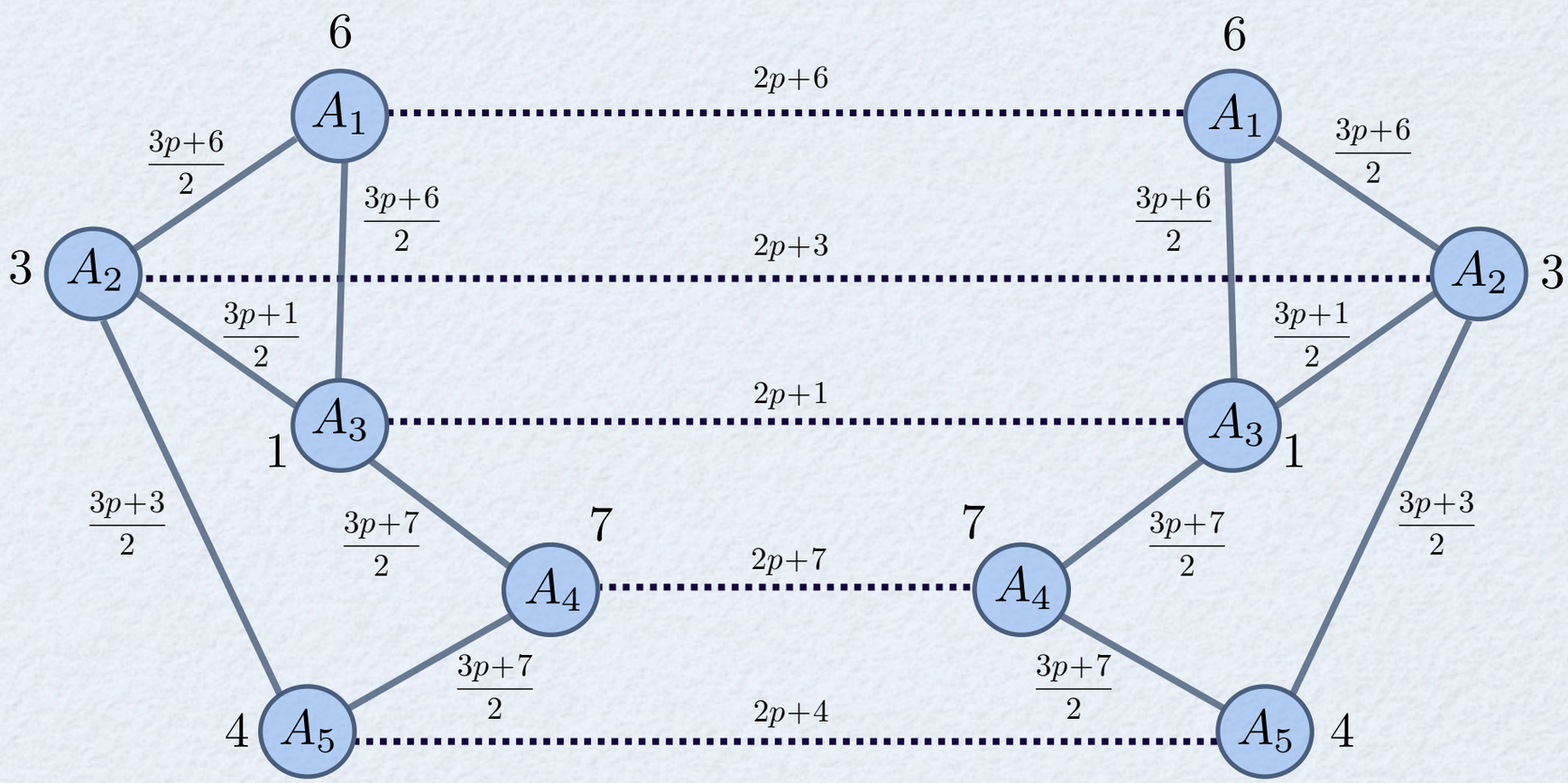
Utilisation de l'algorithme sur un exemple : $p = 5$

1) Pondérer chaque sommet par les b_i
2) Enlever les arêtes inutiles
3) Dédoubler le graphe



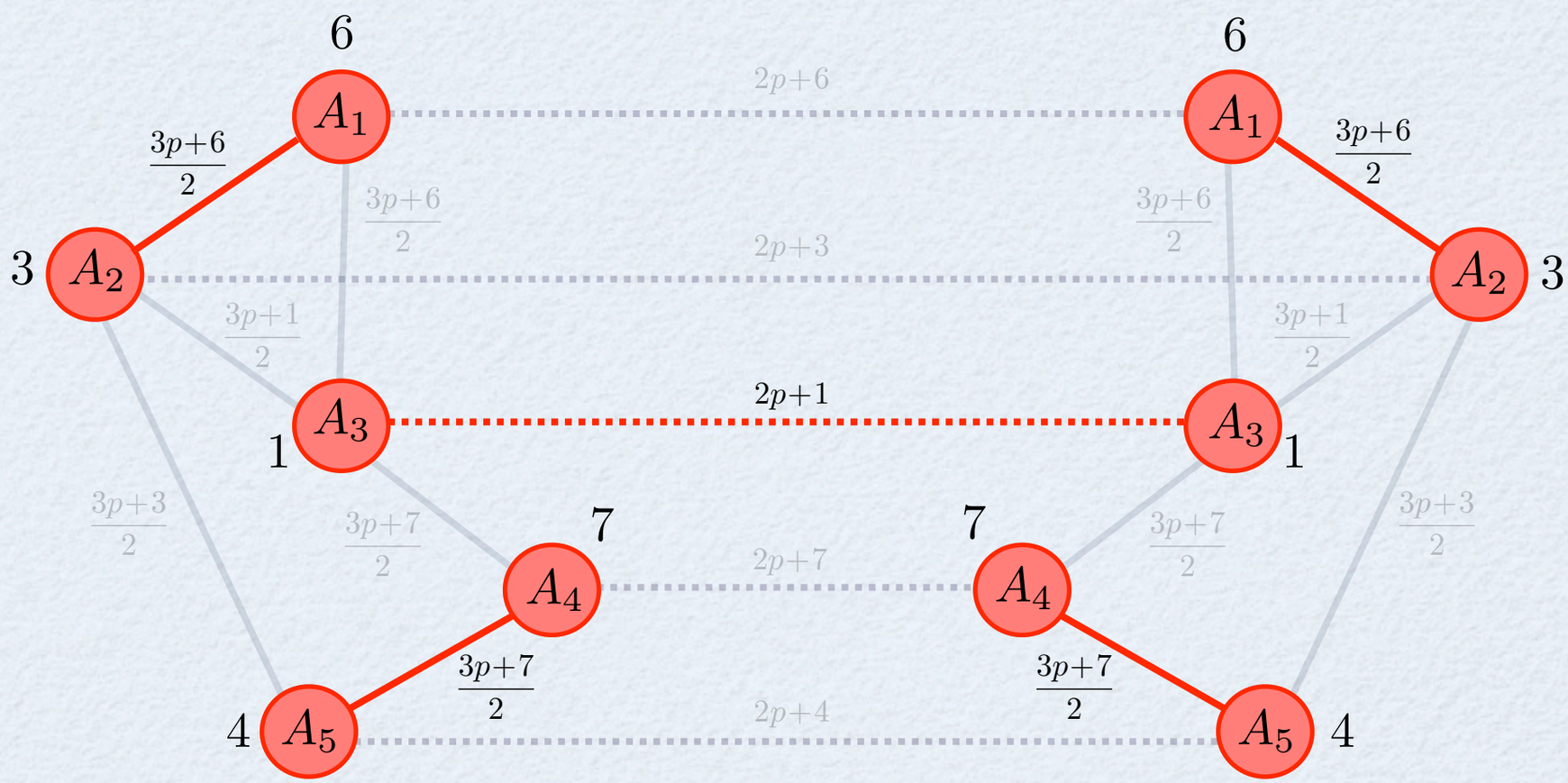
Utilisation de l'algorithme sur un exemple : $p = 5$

1) Pondérer chaque sommet par les b_i
2) Enlever les arêtes inutiles
3) Dédoubler le graphe
4) Pondérer les arêtes



Utilisation de l'algorithme sur un exemple : $p = 5$

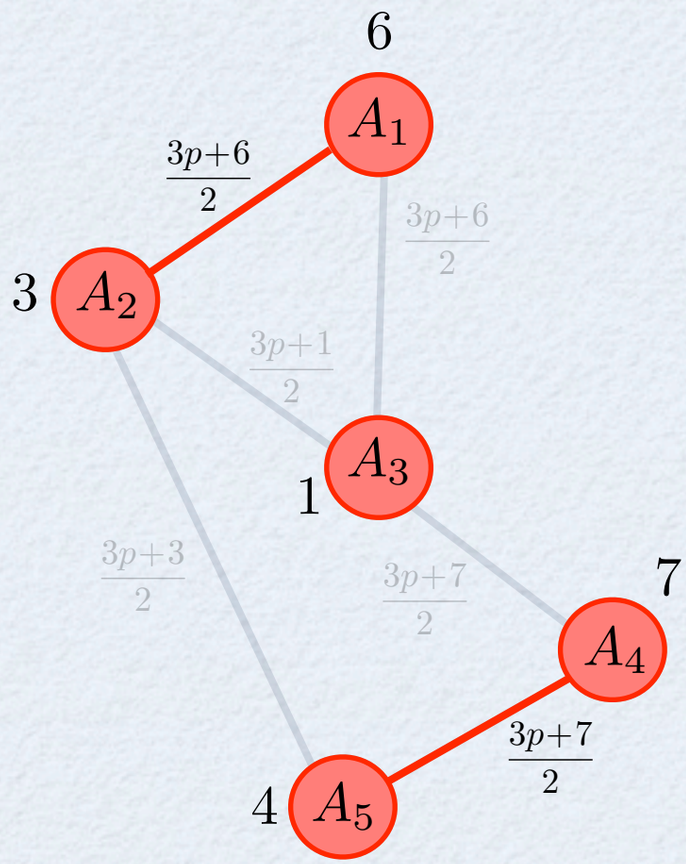
1) Pondérer chaque sommet par les b_i
2) Enlever les arêtes inutiles
3) Dédoubler le graphe
4) Pondérer les arêtes
5) Couplage parfait de poids minimum [Edmonds]



Graphe G_c Graphe G'_c

Utilisation de l'algorithme sur un exemple : $p = 5$

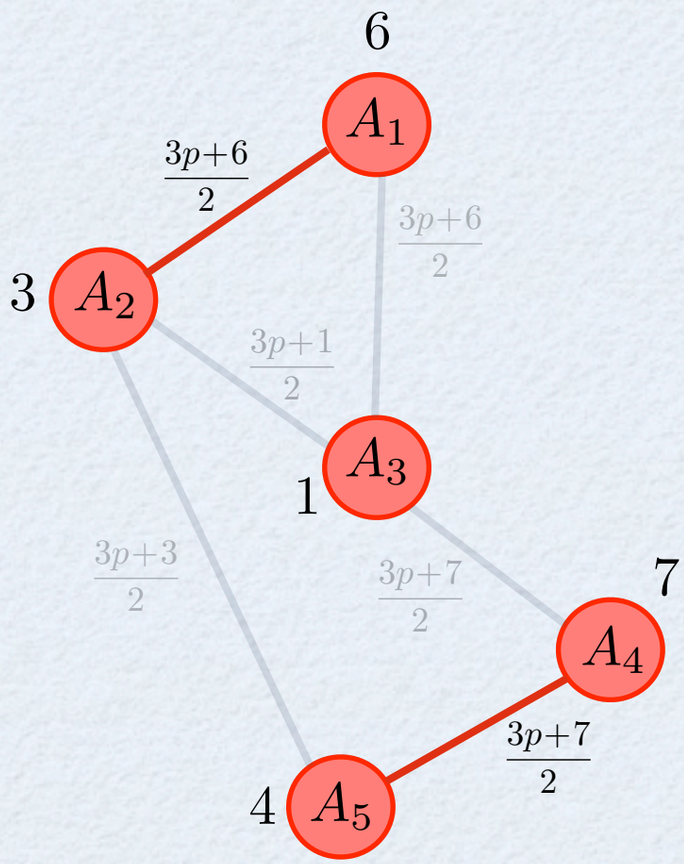
1) Pondérer chaque sommet par les b_i
2) Enlever les arêtes inutiles
3) Dédoubler le graphe
4) Pondérer les arêtes
5) Couplage parfait de poids minimum [Edmonds]
6) Diviser le graphe et ordonnancer selon le couplage



Graphe G_c

Utilisation de l'algorithme sur un exemple : $p = 5$

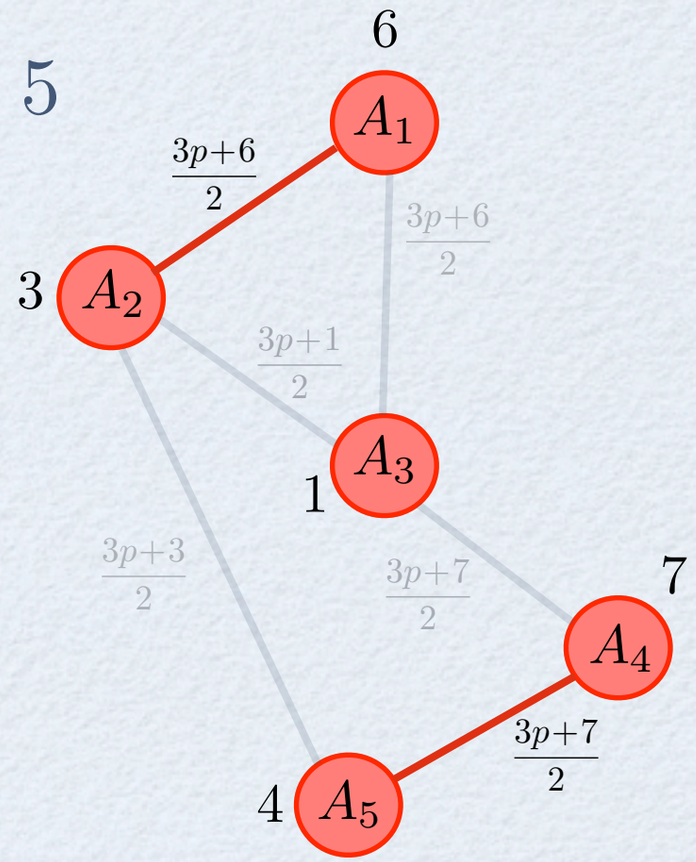
1) Pondérer chaque sommet par les b_i
2) Enlever les arêtes inutiles
3) Dédoubler le graphe
4) Pondérer les arêtes
5) Couplage parfait de poids minimum [Edmonds]
6) Diviser le graphe et ordonnancer selon le couplage



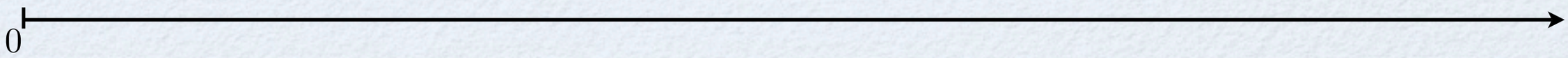
Graphe G_c

Utilisation de l'algorithme sur un exemple : $p = 5$

1) Pondérer chaque sommet par les b_i
2) Enlever les arêtes inutiles
3) Dédoubler le graphe
4) Pondérer les arêtes
5) Couplage parfait de poids minimum [Edmonds]
6) Diviser le graphe et ordonnancer selon le couplage

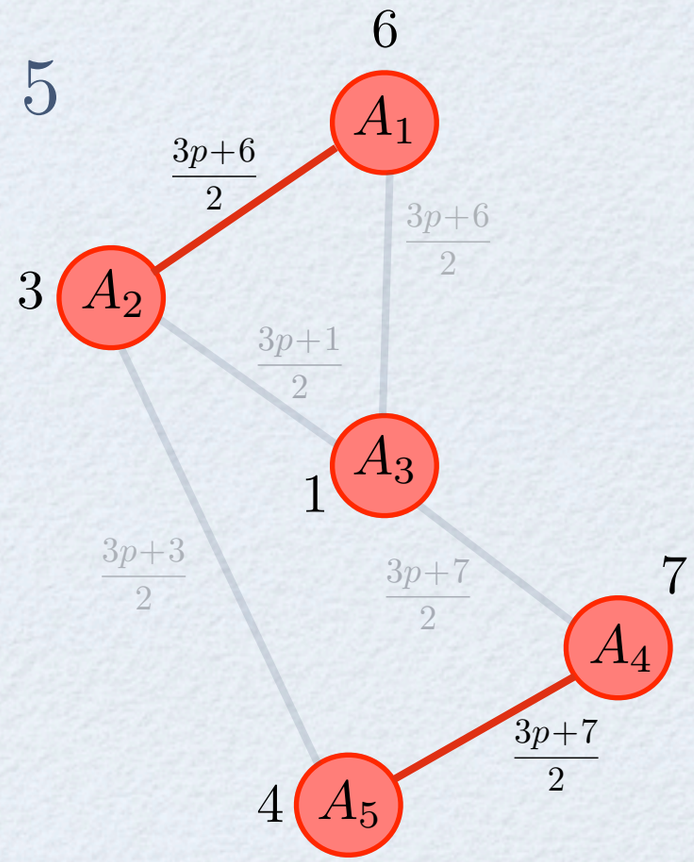


Graphe G_c

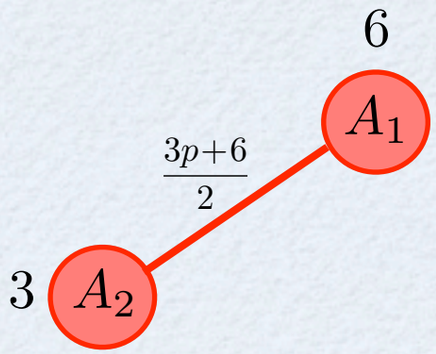


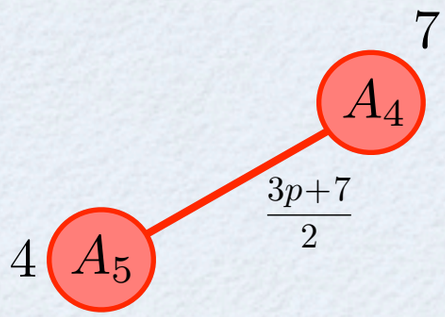
Utilisation de l'algorithme sur un exemple : $p = 5$

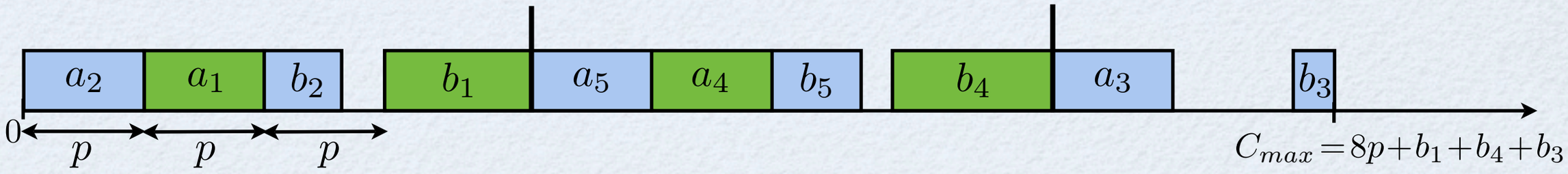
1) Pondérer chaque sommet par les b_i
2) Enlever les arêtes inutiles
3) Dédoubler le graphe
4) Pondérer les arêtes
5) Couplage parfait de poids minimum [Edmonds]
6) Diviser le graphe et ordonnancer selon le couplage



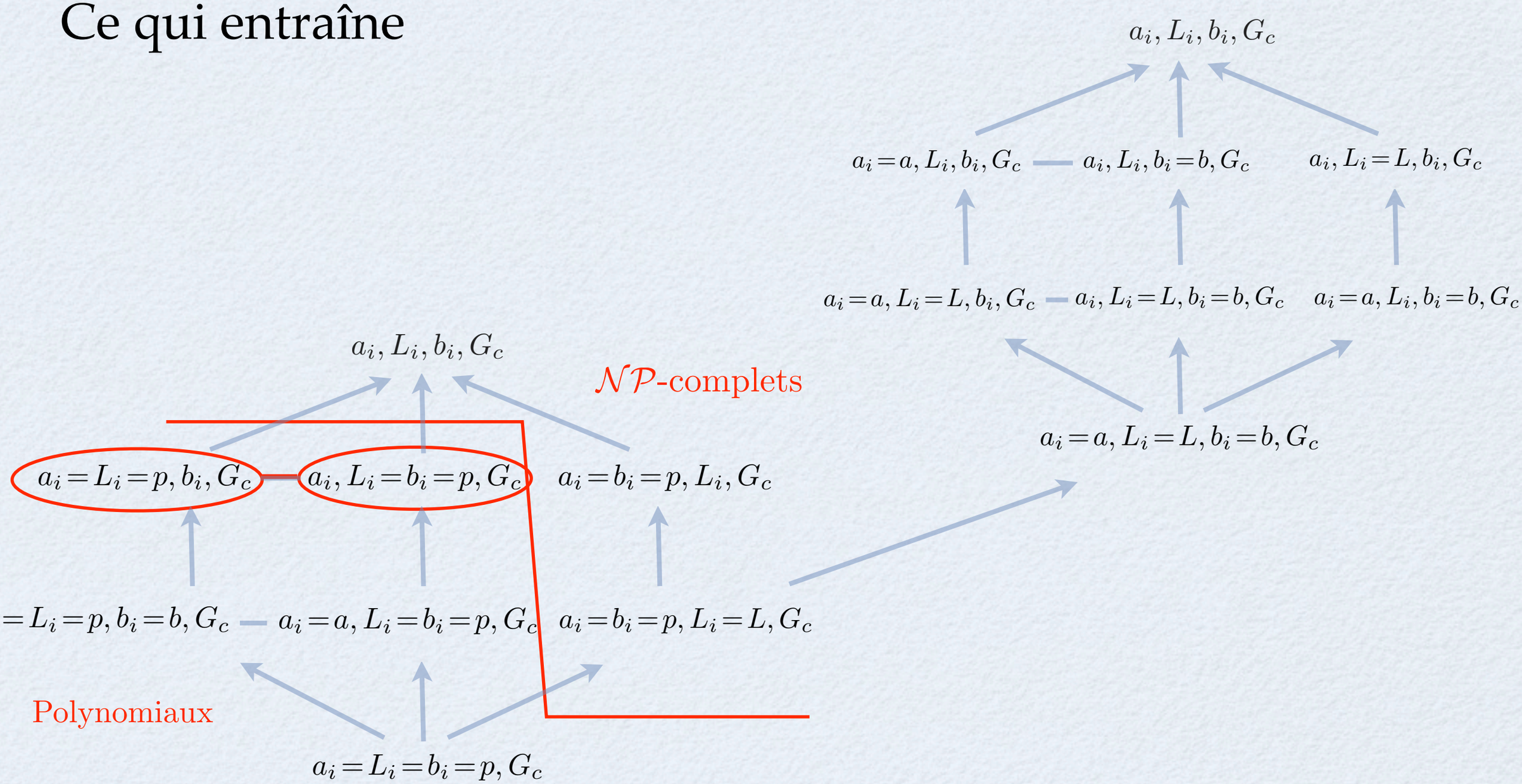
Graphe G_c







Ce qui entraîne



Avec graphe de compatibilité : approches spécifiques

- Techniques habituelles pour les tâches-couplées sur monoprocesseur non applicables.
- La solution :
  - Penser à remplir les temps de latence des tâches d'acquisition tout en gérant la contrainte d'incompatibilité
  - Des heuristiques qui exploitent le graphe de compatibilité

Avec graphe de compatibilité : clique de taille maximale

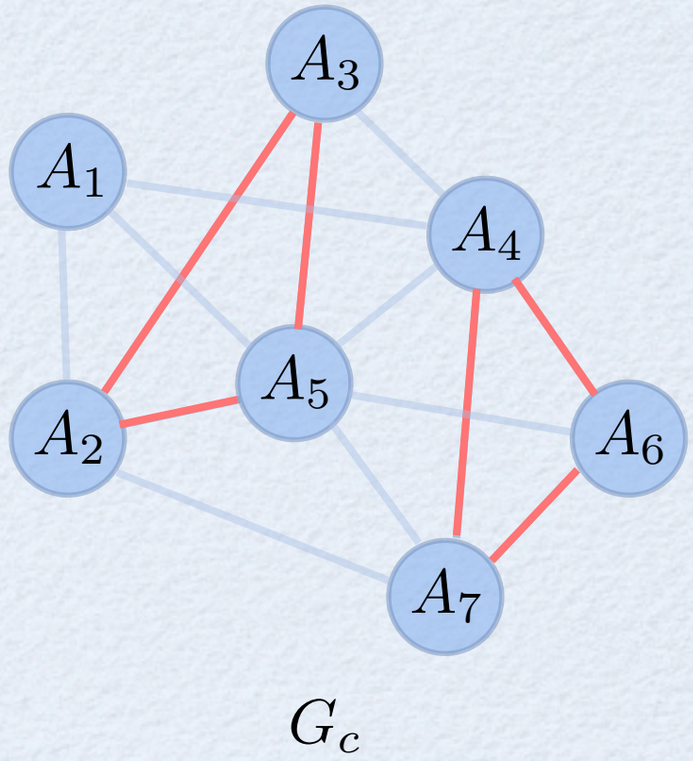
Algorithme en temps polynomial :

Données : $\mathcal{A}, \tau, G_c$

Résultat : C^h_{max}

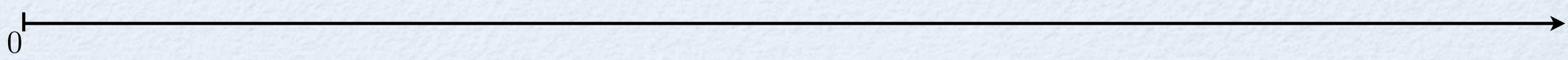
Début

- Tant qu'il reste des sommets non visités dans G_c
  - Prendre un sommet non visité A_i
  - Chercher une clique de taille maximale
- Ordonnancer les cliques par ordre décroissant



G_c

Exemple pour le triplet $(a_i = b_i = 1, L_i = L)$



Avec graphe de compatibilité : clique de taille maximale

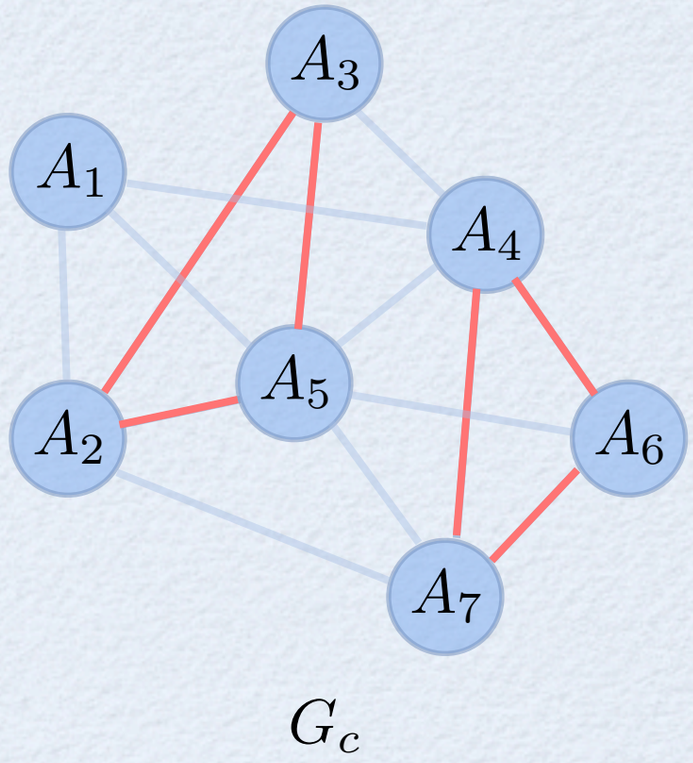
Algorithme en temps polynomial :

Données : $\mathcal{A}, \tau, G_c$

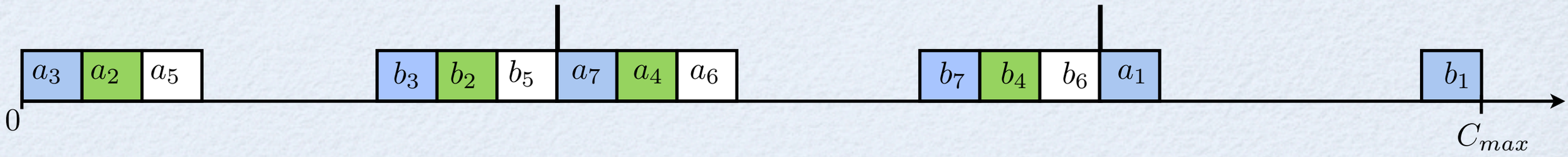
Résultat : C^h_{max}

Début

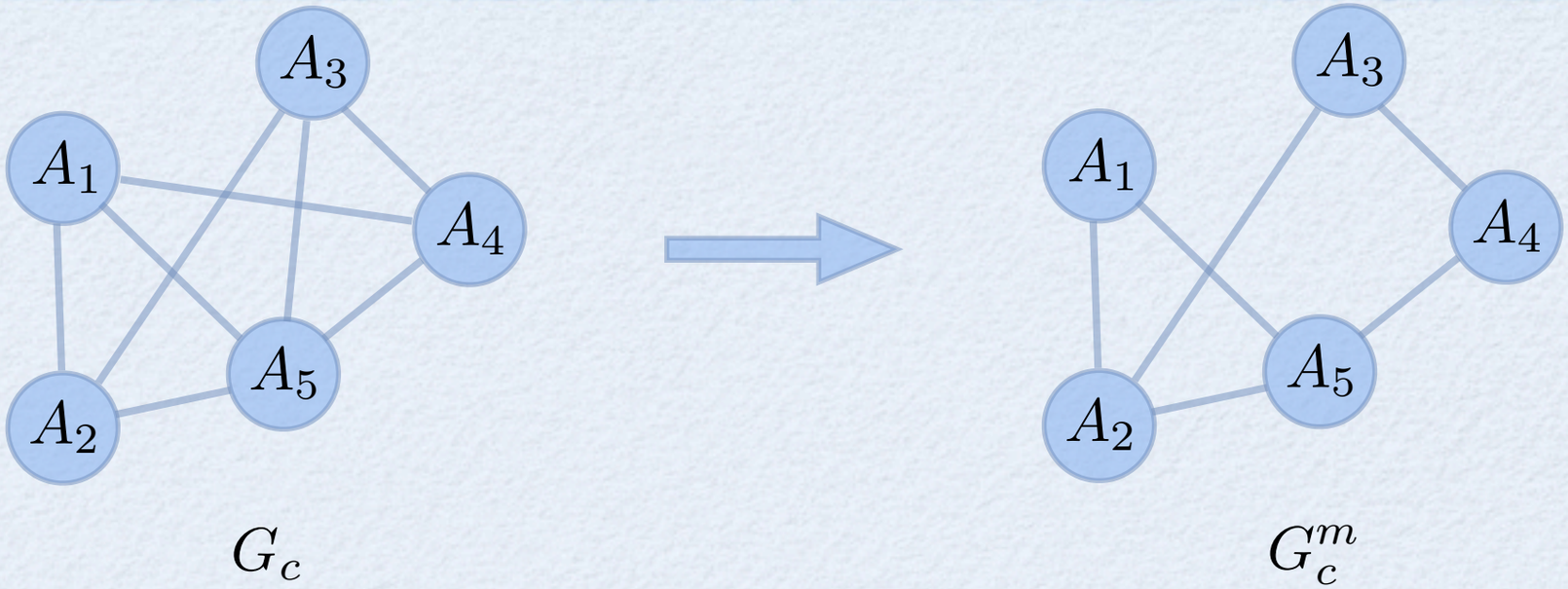
- Tant qu'il reste des sommets non visités dans G_c
  - Prendre un sommet non visité A_i
  - Chercher une clique de taille maximale
- Ordonnancer les cliques par ordre décroissant



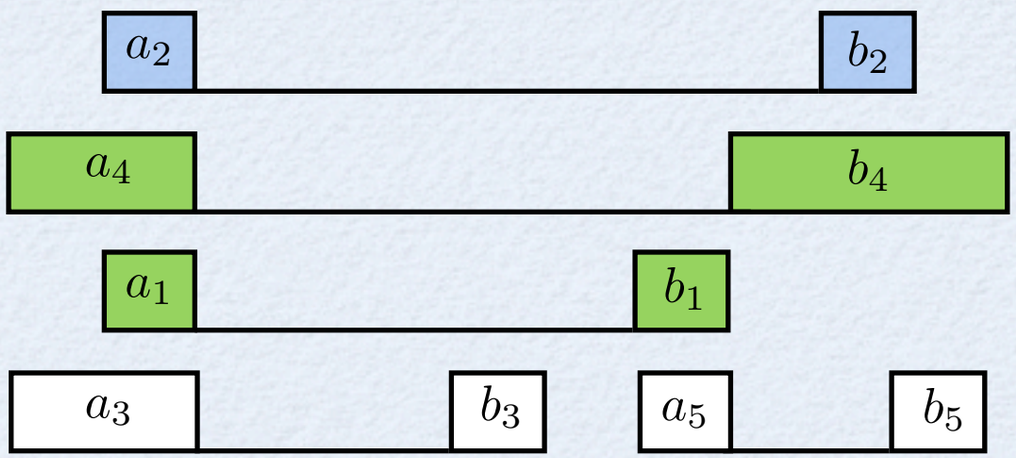
Exemple pour le triplet $(a_i = b_i = 1, L_i = L)$



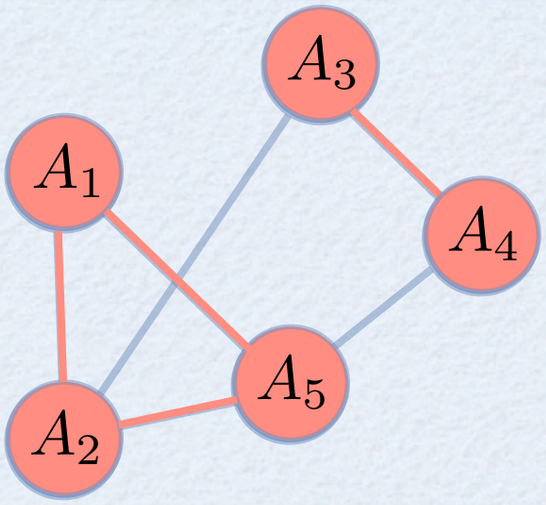
Avec graphe de compatibilité : poupées russes



Exemple pour le triplet (a_i, L_i, b_i)

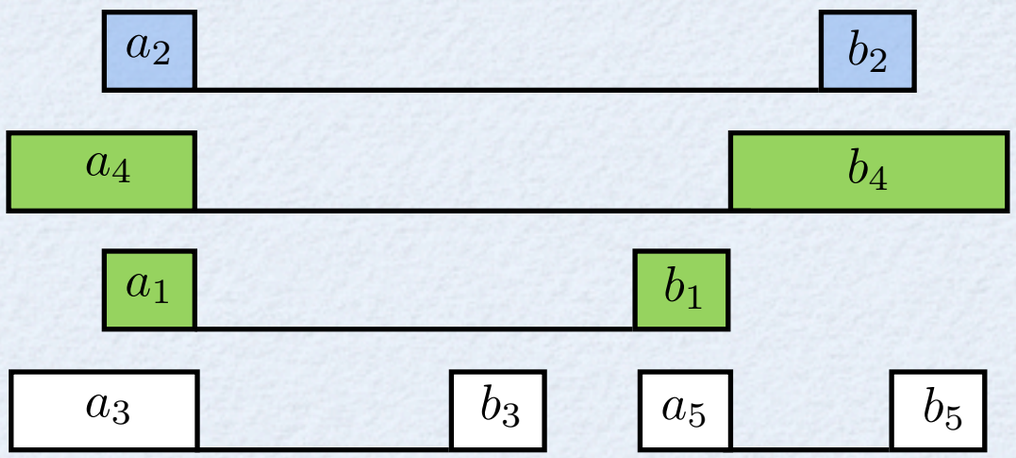


Avec graphe de compatibilité : poupées russes



G_c^m

Exemple pour le triplet (a_i, L_i, b_i)



Algorithme en temps polynomial

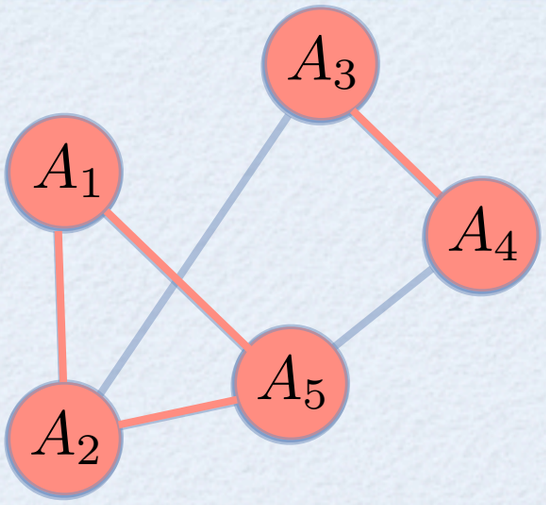
Données : $\mathcal{A}, \tau, G_c^m$

Résultat : C_{max}^h

Début

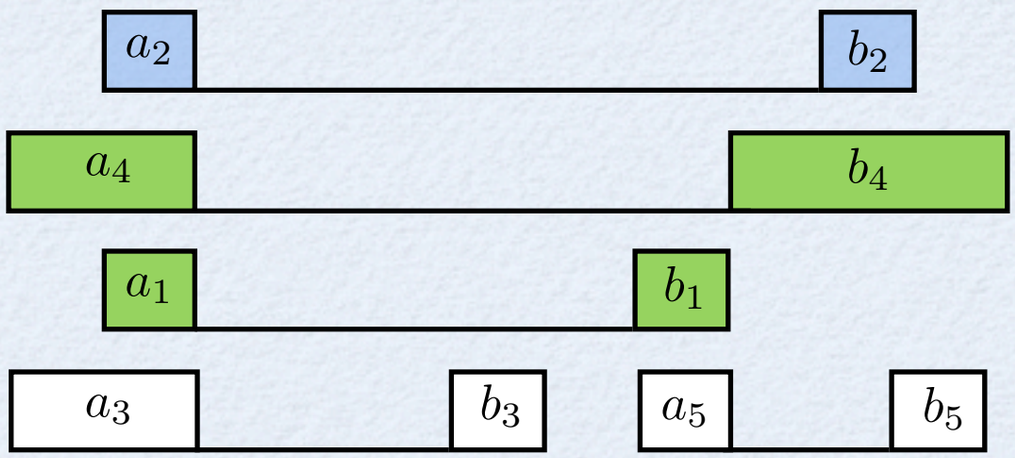
- Tant qu'il reste des sommets non visités
  - Prendre un sommet non visité A_i
  - Chercher une clique de taille maximale

Avec graphe de compatibilité : poupées russes



G_c^m

Exemple pour le triplet (a_i, L_i, b_i)



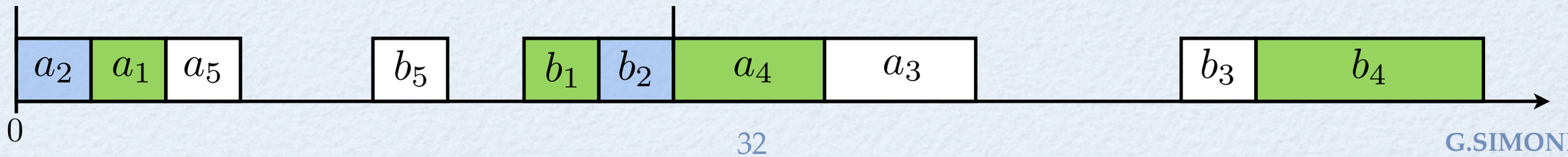
Algorithme en temps polynomial

Données : $\mathcal{A}, \tau, G_c^m$

Résultat : C_{max}^h

Début

- Tant qu'il reste des sommets non visités
  - Prendre un sommet non visité A_i
  - Chercher une clique de taille maximale
- Ordonnancer les cliques



Avec graphe de compatibilité : résultats d'approximation

- Le recouvrement des sommets par cliques de taille maximale donne :
  - une $(\frac{7}{4}+\frac{L}{4p})$-approximation pour $\Pi_1 : (a_i = b_i = p, L_i = L), G_c$

- L'heuristique des poupées russes donne :
  - une $\frac{3}{2}$-approximation pour $\Pi_3 : (a_i = b_i = L_i), G_c$
  - une $\frac{7}{6}$-approximation quand G_c est orienté biparti complet

Avec graphe de compatibilité : conclusion de la partie

- Impact important du graphe de compatibilité
- Techniques de preuves habituelles pour les tâches-couplées sur monoprocesseur non applicables
- Utilisation des problèmes de recouvrement de sommets :
  - Couplage de taille maximum
  - Recouvrement par chaînes disjointes
  - Recouvrement par cliques disjointes

Table des matières

Avec tâches de traitement - objectifs

- Type de problème
  - n tâches d'acquisition et de traitement
  - Un graphe de précédence G_p particulier
  - Différents problèmes selon les paramètres (a_i, L_i, b_i)
- Nos objectifs
  - Classifier chaque problème rencontré
  - Développer des algorithmes d'approximation

Avec tâches de traitement - complexité

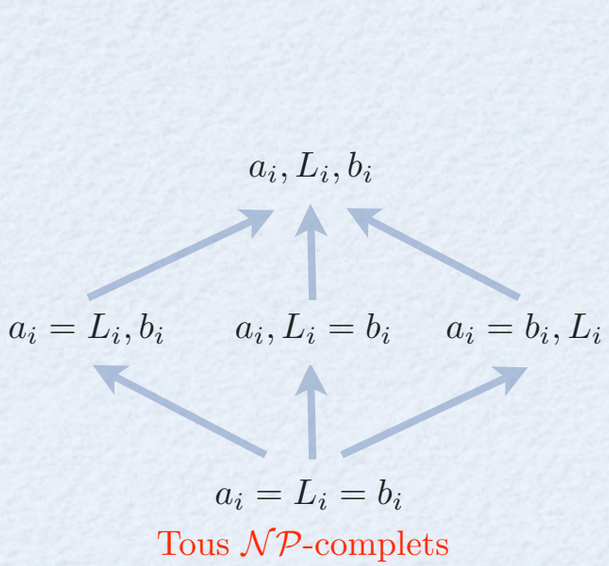
Résultats de complexité par Orman et Potts

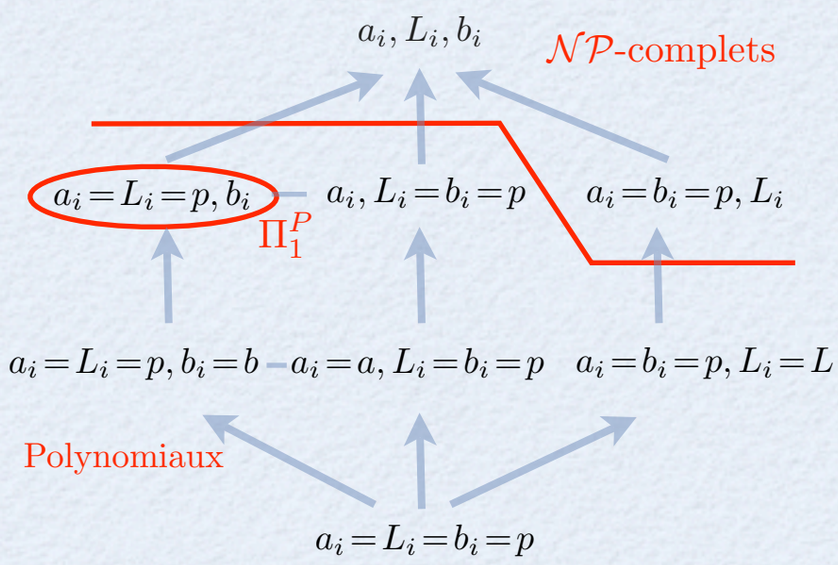
Avec tâches de traitement - complexité

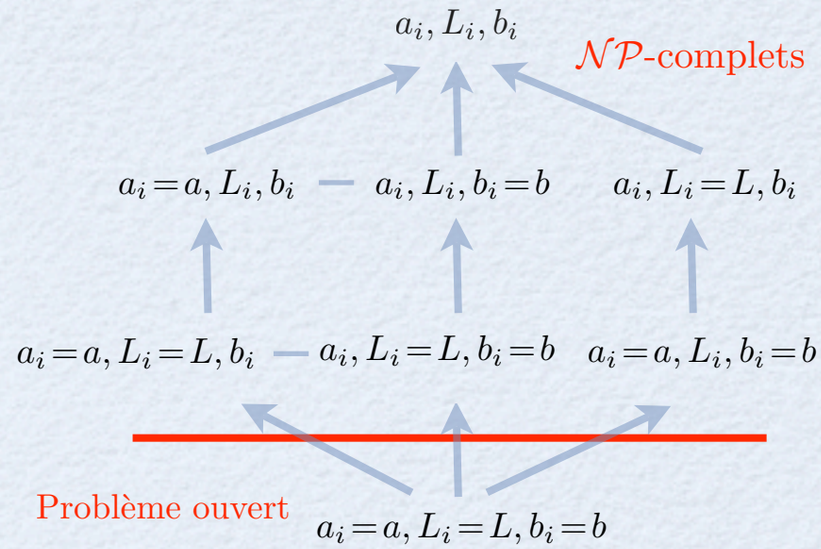
Résultats de complexité par Orman et Potts

Avec tâches de traitement - complexité

Résultats de complexité par Orman et Potts







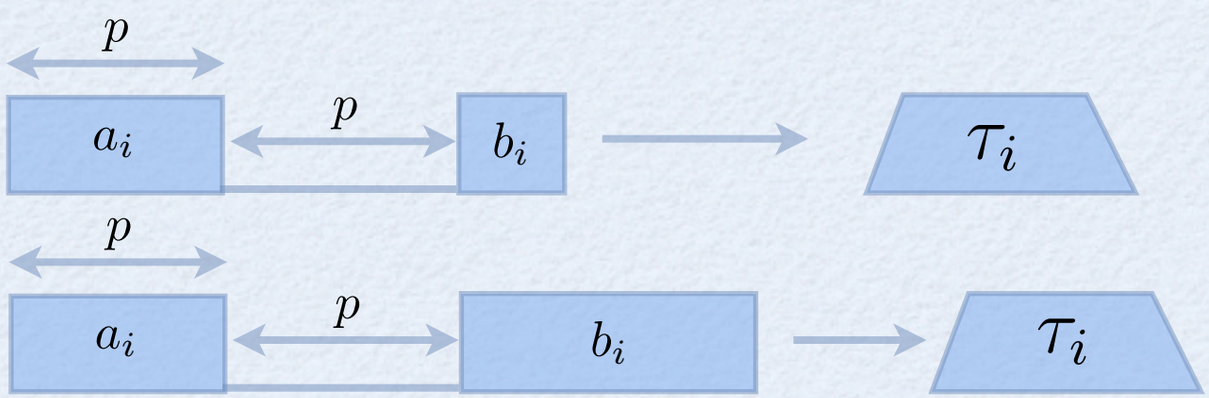
Quelle est la complexité avec G_p ?

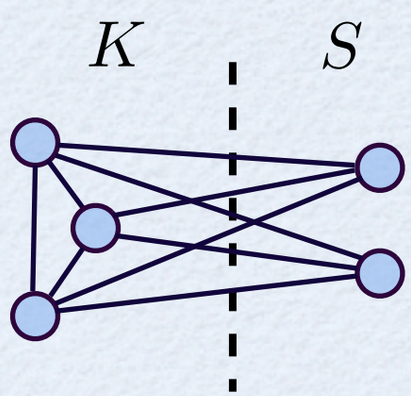
un cas intéressant

Avec tâches de traitement : problème Π_1^P

Théorème

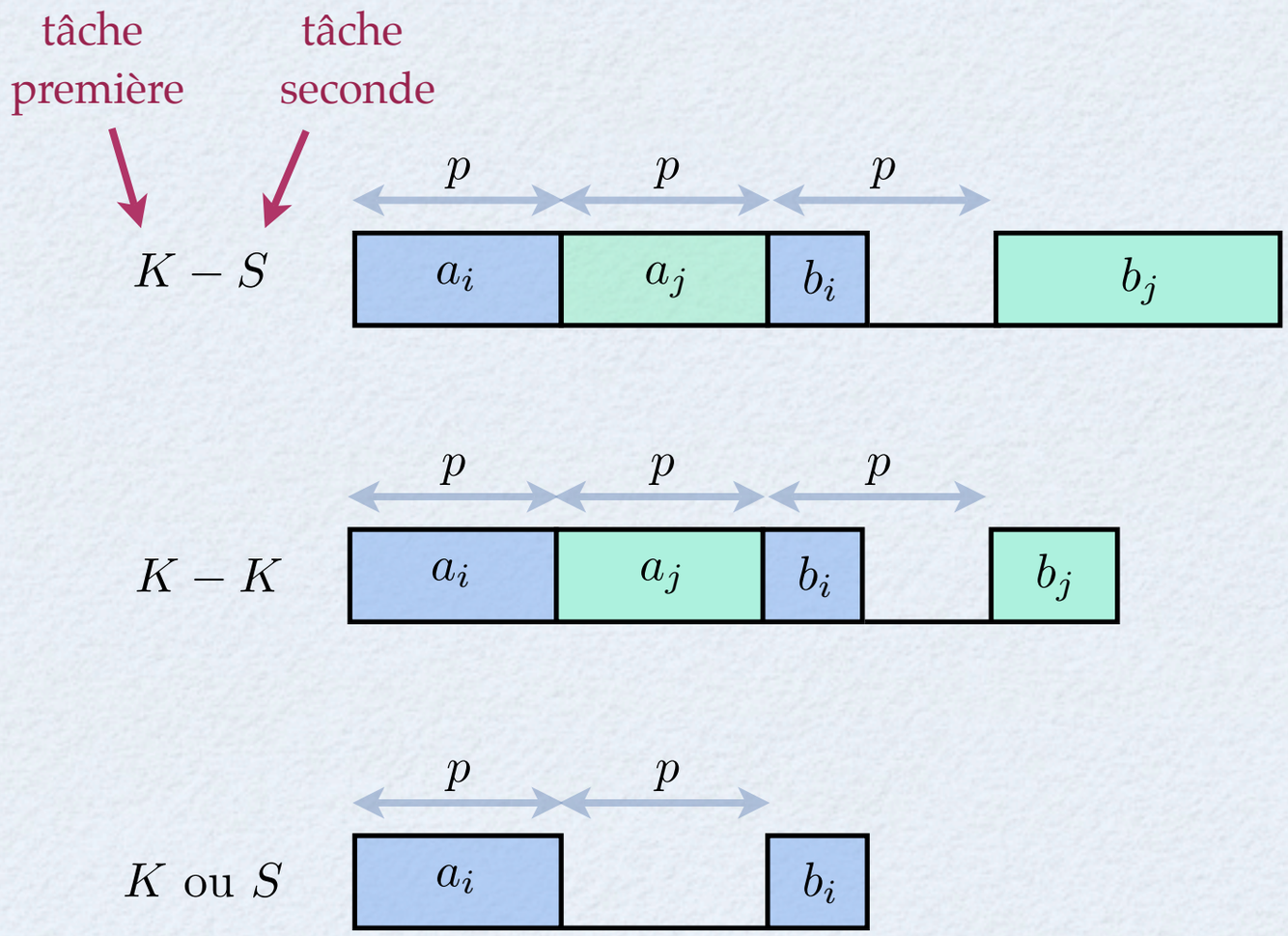
Le problème $\Pi_1^P : (a_i = L_i = p, b_i) \cup \tau_i$ est polynomial



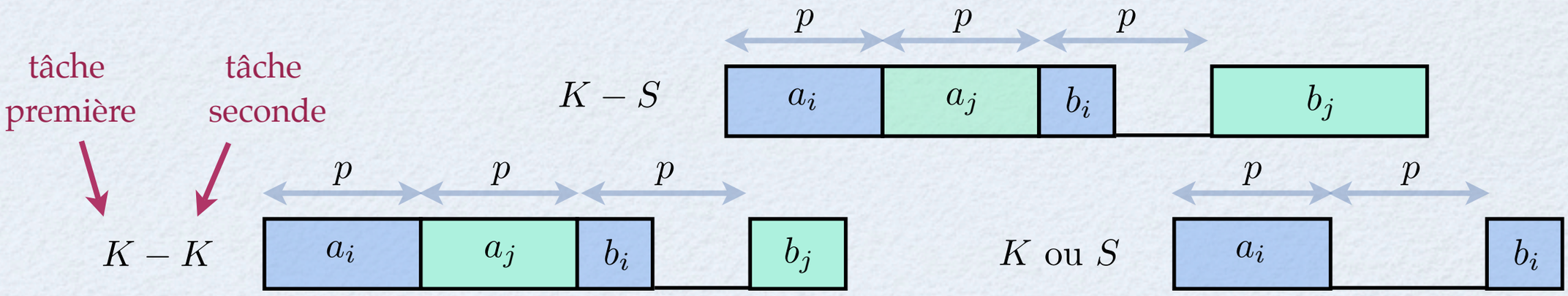


- Soit K l'ensemble des tâches d'acquisition telles que $b_i \leq p$
- Soit S l'ensemble des tâches d'acquisition telles que $b_i > p$
- K est complet, S forme un stable, un complet les relie

- Il n'y a que trois types d'ordonnancement possible



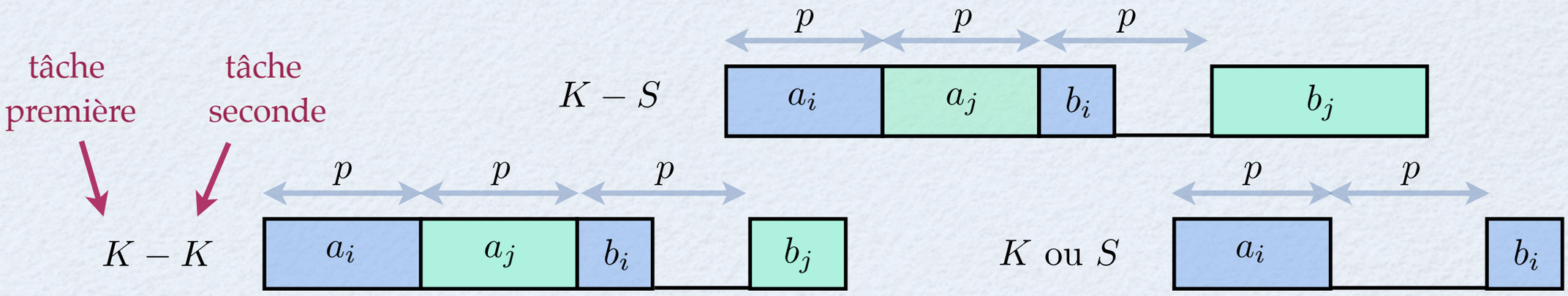
- Il n'y a que trois types d'ordonnancement possible



- Toute solution optimale est associée à un couplage M

- Il existe une solution optimale, pour laquelle nous avons :
  - Si $\tau_i^S > \tau_j^S$ alors S_i est exécutée avant S_j

- Il n'y a que trois types d'ordonnancement possible

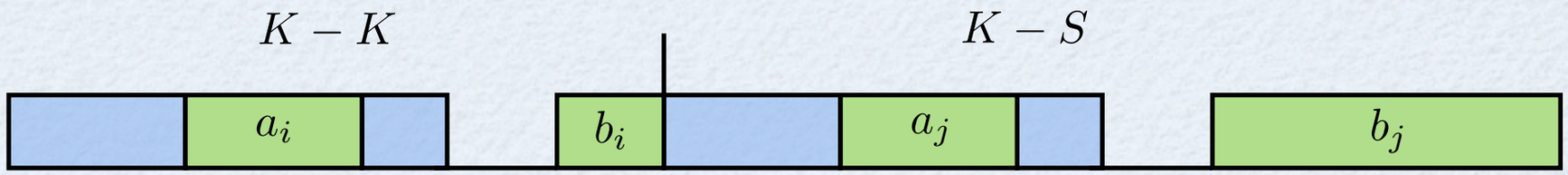


- Toute solution optimale est associée à un couplage M

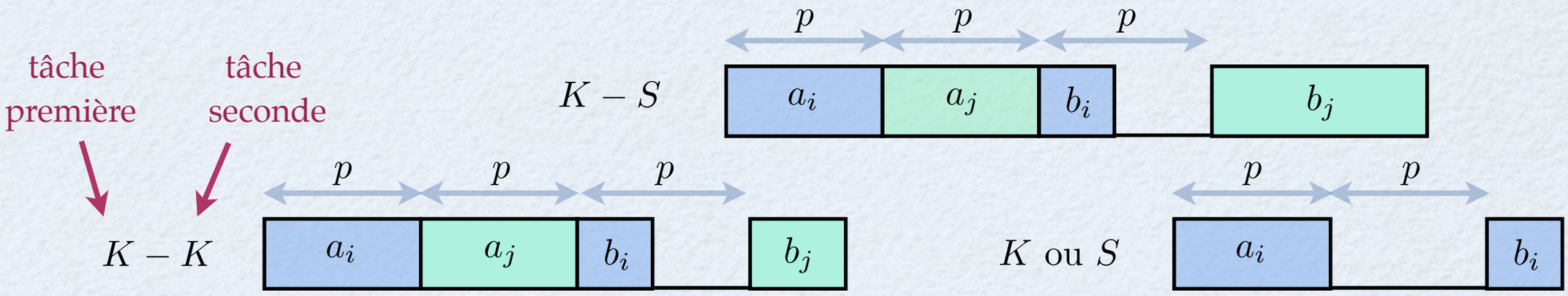
- Il existe une solution optimale, pour laquelle nous avons :

  - Si $\tau_i^S > \tau_j^S$ alors S_i est exécutée avant S_j

  - Si K_i et S_j sont deux tâches secondes, et si $\tau_i^K > \tau_j^S$, K_i est exécutée avant S_j comme tâche seconde

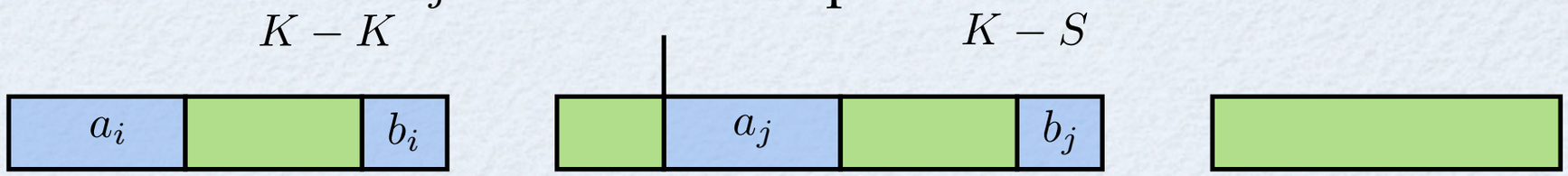


- Il n'y a que trois types d'ordonnancement possible

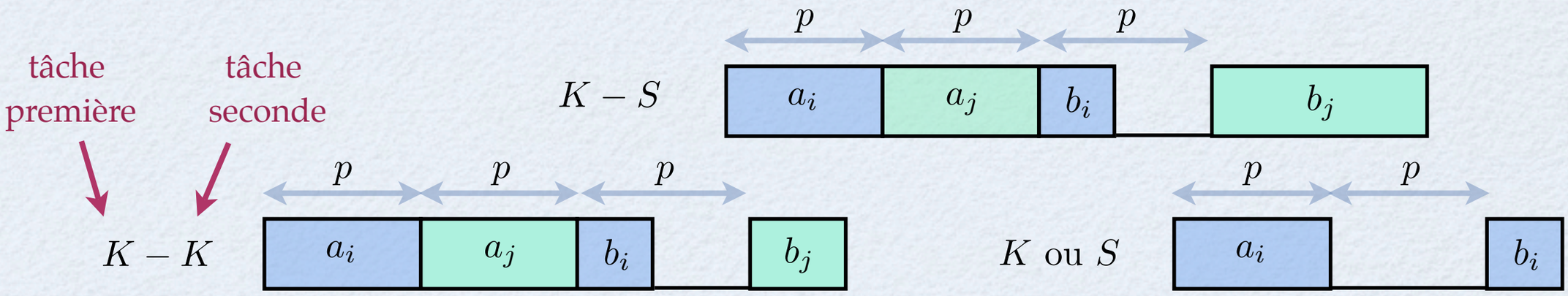


- Toute solution optimale est associée à un couplage M

- Il existe une solution optimale, pour laquelle nous avons :
  - Si $\tau_i^S > \tau_j^S$ alors S_i est exécutée avant S_j
  - Si K_i et S_j sont deux tâches secondes, et si $\tau_i^K > \tau_j^S$, K_i est exécutée avant S_j comme tâche seconde
  - Si K_i et K_j sont deux tâches premières, et si $b_i + \tau_i^K > b_j + \tau_j^K$, K_i est exécutée avant K_j comme tâche première



- Il n'y a que trois types d'ordonnancement possible



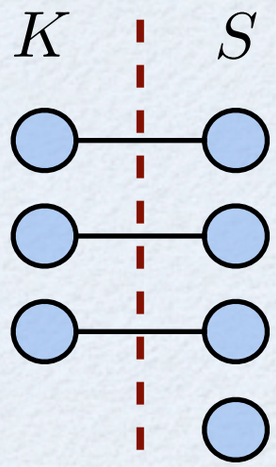
- Toute solution optimale est associée à un couplage M
- Il existe une solution optimale, pour laquelle nous avons :
  - Si $\tau_i^S > \tau_j^S$ alors S_i est exécutée avant S_j
  - Si K_i et S_j sont deux tâches secondes, et si $\tau_i^K > \tau_j^S$, K_i est exécutée avant S_j comme tâche seconde
  - Si K_i et K_j sont deux tâches premières, et si $b_i + \tau_i^K > b_j + \tau_j^K$, K_i est exécutée avant K_j comme tâche première
  - Les tâches isolées de S sont toujours exécutées à la fin de l'ordonnancement

- Il existe une solution optimale, pour laquelle nous avons :
  - Si $\tau_i^S > \tau_j^S$ alors S_i est exécutée avant S_j
  - Si K_i et S_j sont deux tâches secondes, et si $\tau_i^K > \tau_j^S$, K_i est exécutée avant S_j comme tâche seconde
  - Si K_i et K_j sont deux tâches premières, et si $b_i + \tau_i^K > b_j + \tau_j^K$, K_i est exécutée avant K_j comme tâche première
  - Les tâches isolées de S sont toujours exécutées à la fin de l'ordonnancement

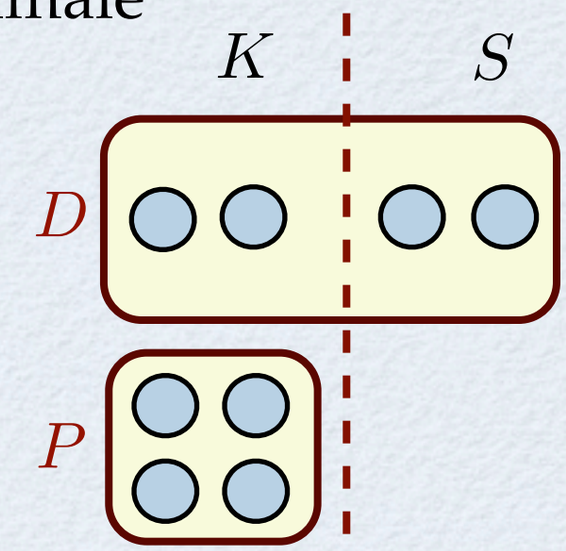
- Il existe une solution optimale, pour laquelle nous avons :
  - Si $\tau_i^S > \tau_j^S$ alors S_i est exécutée avant S_j
  - Si K_i et S_j sont deux tâches secondes, et si $\tau_i^K > \tau_j^S$, K_i est exécutée avant S_j comme tâche seconde
  - Si K_i et K_j sont deux tâches premières, et si $b_i + \tau_i^K > b_j + \tau_j^K$, K_i est exécutée avant K_j comme tâche première
  - Les tâches isolées de S sont toujours exécutées à la fin de l'ordonnancement
  - Le couplage M associé est de taille maximum
  - Nous pouvons numéroter les tâches de cette façon là :

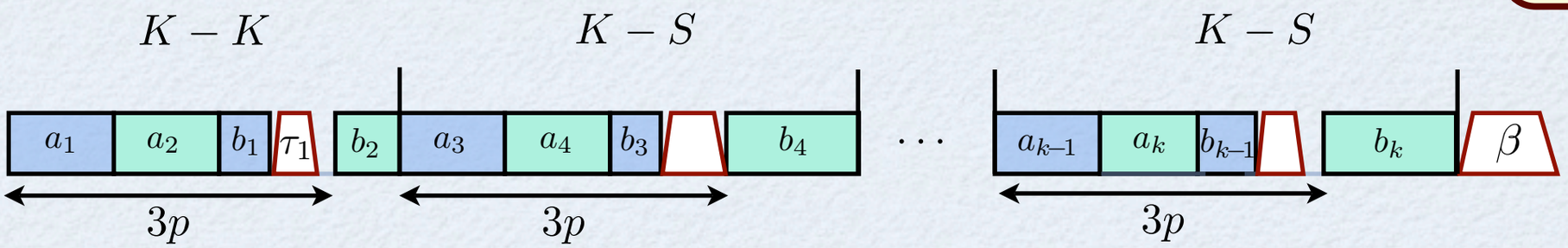
$$b_1 + \tau_1^K \geq b_2 + \tau_2^K \geq b_3 + \tau_3^K \geq \ldots \geq b_k + \tau_k^K$$
$$\tau_1^S \geq \tau_2^S \geq \tau_3^S \geq \ldots \geq \tau_l^S$$

- Si $|K| \leq |S|$, un couplage ordonné donne la solution optimale



- Si $|K| \leq |S|$, un couplage ordonné donne la solution optimale
- Si $|K| > |S|$, une étude cas par cas est nécessaire :
  - Soit P les tâches premières, et D les tâches secondes
  - Notre ordonnancement dépend de trois points :



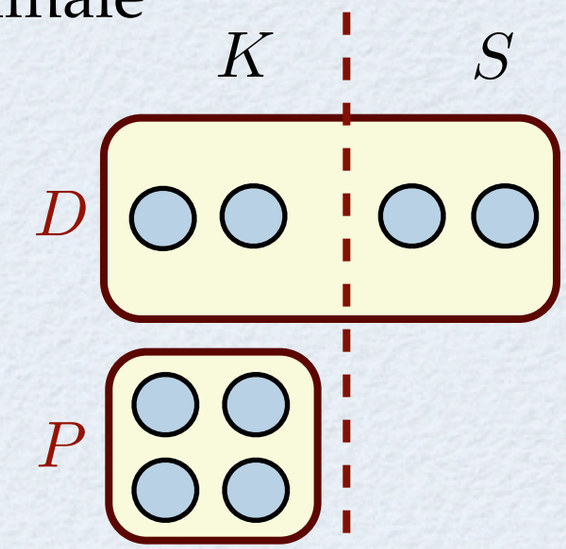


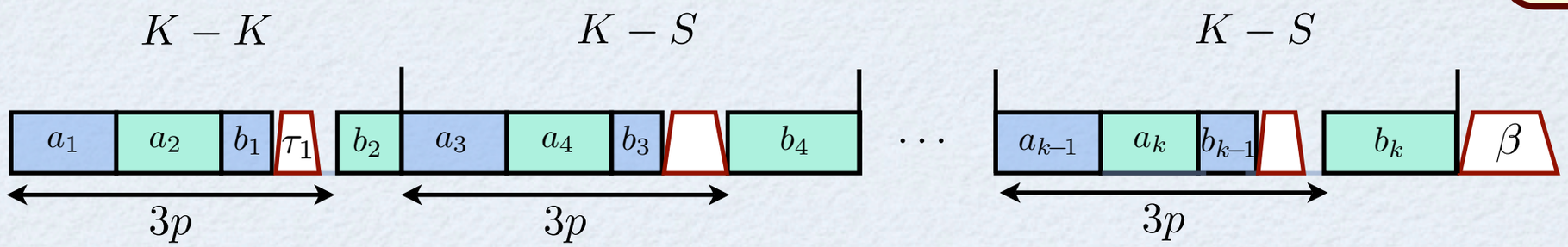
$min \sum_{A_i \in D} b_i$ est le temps minimum d'exécution des sous-tâches secondes

$b_1 + \tau_1^K$ est le plus grand $b_i + \tau_i^K$ que l'on puisse avoir

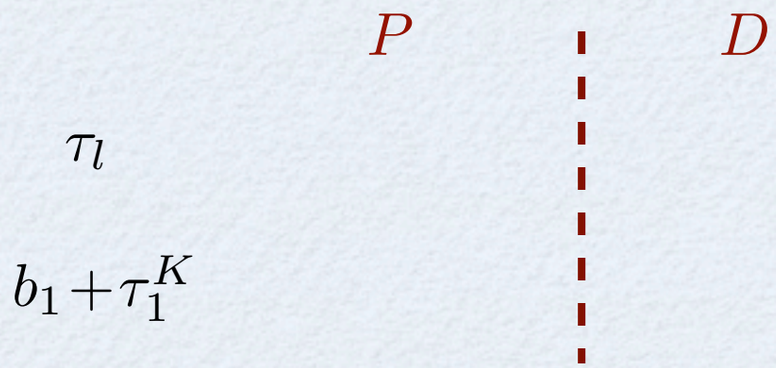
τ_l représente la tâche de traitement de plus petite durée d'exécution

- Si $|K| \leq |S|$, un couplage ordonné donne la solution optimale
- Si $|K| > |S|$, une étude cas par cas est nécessaire :
  - Soit P les tâches premières, et D les tâches secondes
  - Notre ordonnancement dépend de trois points :

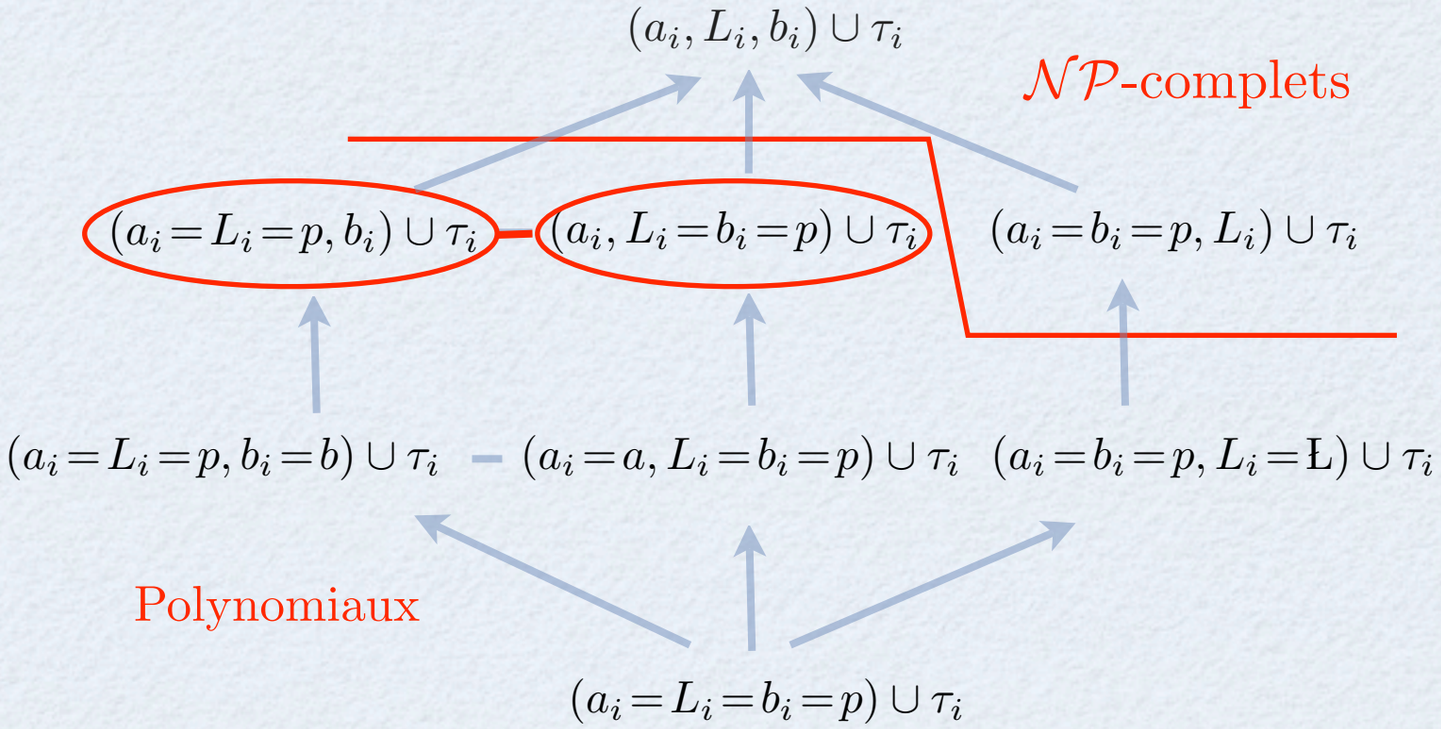




Prenons un ordonnancement avec $min \sum_{A_i \in D} b_i$



Ce qui entraîne



Conclusion

- Nous obtenons la même classification qu'Orman et Potts
- Les preuves deviennent beaucoup plus techniques

Table des matières

- Introduction
  - Présentation du problème
  - Modélisation
  - État de l'art
- Étude de la complexité et de l'approximation
  - Présence d'un graphe de compatibilité quelconque
  - Présence des tâches de traitement
  - Présence d'un graphe de compatibilité quelconque et des tâches de traitement
- Analyse des cas critiques
  - Limite entre polynomialité et NP-compétude selon les paramètres
- Conclusion

Avec graphe G_c et tâches de traitement - objectifs

- Type de problème
  - n tâches d'acquisition et de traitement
  - Un graphe de compatibilité G_c quelconque
  - Un graphe de précédence G_p particulier
  - Différents problèmes selon les paramètres (a_i, L_i, b_i)
- Nos objectifs
  - Classifier chaque problème rencontré
  - Développer des algorithmes d'approximation

Avec graphe G_c et tâches de traitement - complexité

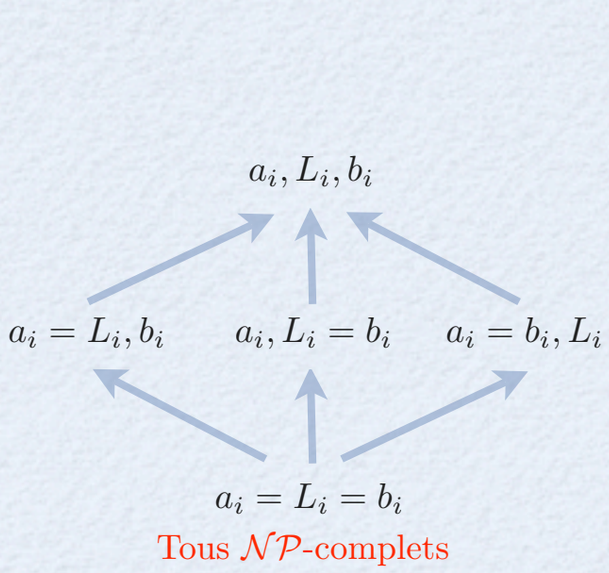
Résultats de complexité par Orman et Potts

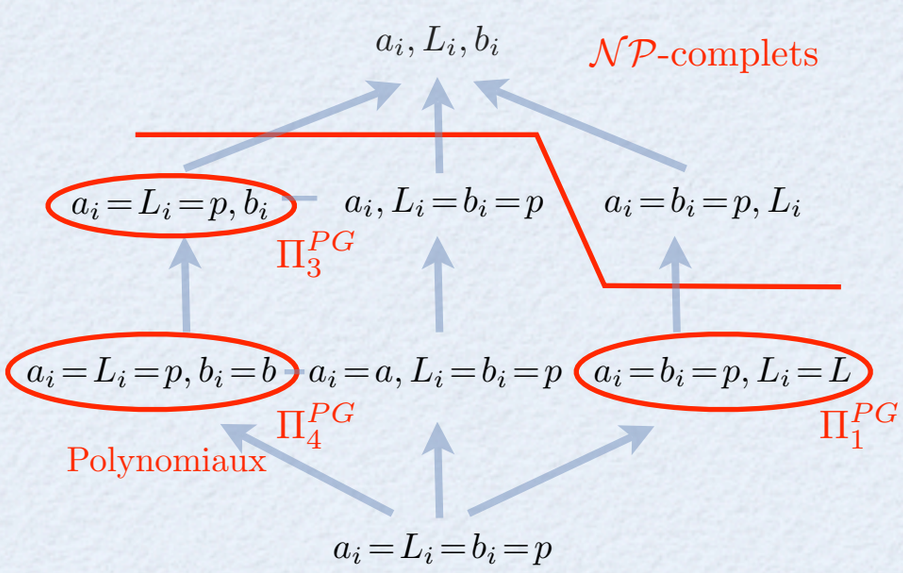
Avec graphe G_c et tâches de traitement - complexité

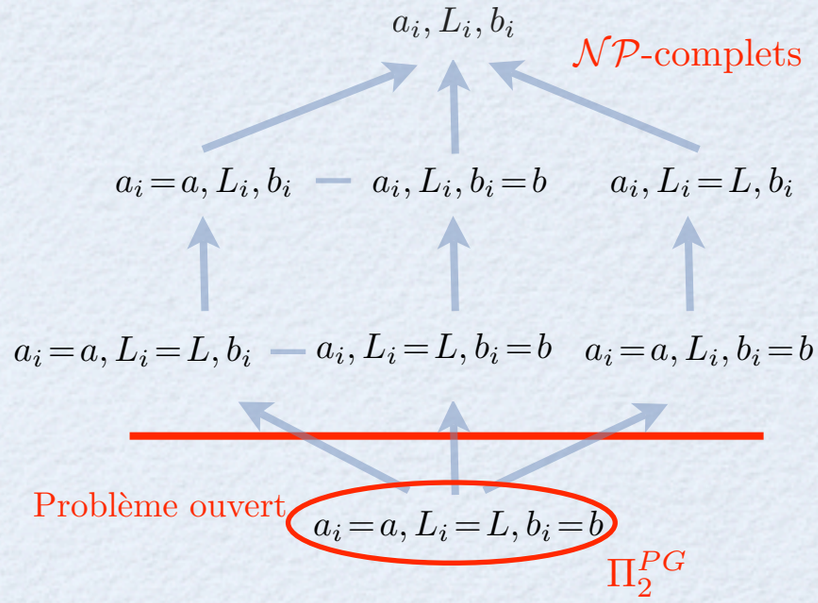
Résultats de complexité par Orman et Potts

Avec graphe G_c et tâches de traitement - complexité

Résultats de complexité par Orman et Potts







Quelle est la complexité avec G_c et G_p ?

Quatre cas intéressants

Avec graphe G_c et tâches de traitement : problème Π_1^{PG}

Étude du problème Π_1^{PG}

$$1|\ prec,\ coupled-task,\ (a_i = b_i = p, L_i = L) \cup (\tau_i\ , pmtn), G_c|C_{max}$$

Étude du cas spécifique $\Pi_1^{PG'}$

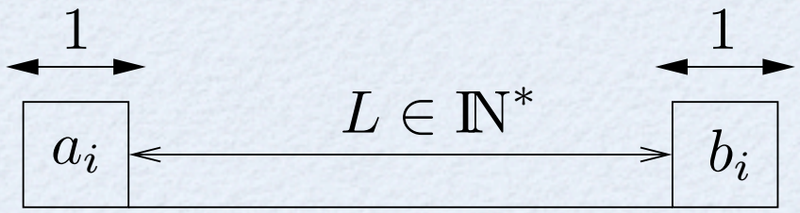
$$1|\ prec,\ coupled-task,\ (a_i = b_i = 1, L_i = L) \cup (\tau_i, pmtn), G_c|C_{max}$$

Avec graphe G_c et tâches de traitement : problème Π_1^{PG}

Étude du cas spécifique $\Pi_1^{PG'}$

$$1|\ prec,\ coupled-task,\ (a_i = b_i = 1, L_i = L) \cup (\tau_i, pmtn), G_c|C_{max}$$

- Toutes les tâches d'acquisition sont de la forme



- Le cas $L = 1$ est polynomial → Couplage de taille maximum
- Montrons que pour $L \geq 2$ notre problème $\Pi_1^{PG'}$ est $\mathcal{NP}$-complet

Théorème

Le problème de décision $\Pi_1^{PG'}$ est $\mathcal{NP}$-complet

Preuve

- Approche similaire à celle de Lenstra et al. pour $P|prec, p_j = 1|C_{max}$
- Réduction faite à partir du problème $\mathcal{NP}$-complet Clique

Preuve

- Approche similaire à celle de Lenstra et al. pour $P|prec, p_j = 1|C_{max}$
- Réduction faite à partir du problème $\mathcal{NP}$-complet Clique

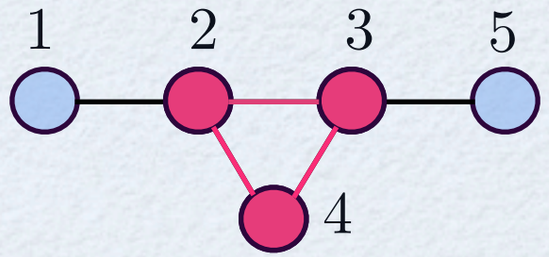
Illustration de la preuve

$n = 5 \qquad L_i = L = 2$

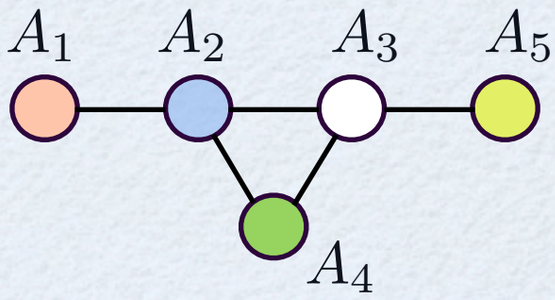
$\tau_i = L = 2$

$C_{max}^{opt} = 20 \qquad K = L + 1 = 3$

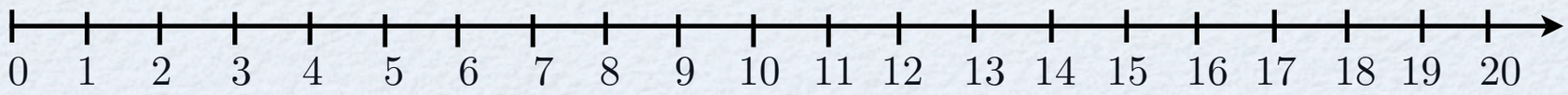
G_p : $A_1 \to \tau_1$, $A_2 \to \tau_2$, $A_3 \to \tau_3$, $A_4 \to \tau_4$, $A_5 \to \tau_5$



Graphe G



Graphe G_c



Preuve

- Approche similaire à celle de Lenstra et al. pour $P|prec, p_j = 1|C_{max}$
- Réduction faite à partir du problème $\mathcal{NP}$-complet Clique

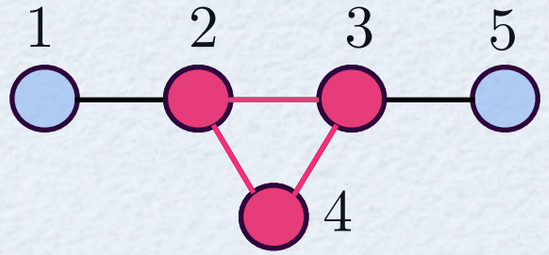
Illustration de la preuve

$n = 5 \qquad L_i = L = 2$

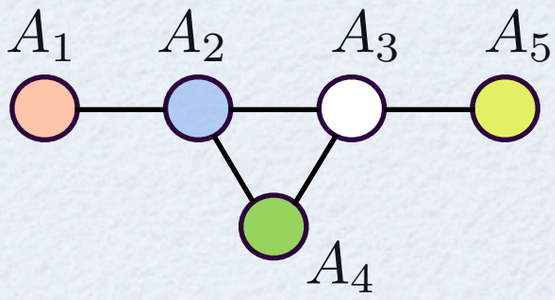
$\tau_i = L = 2$

$C^{opt}_{max} = 20 \qquad K = L + 1 = 3$

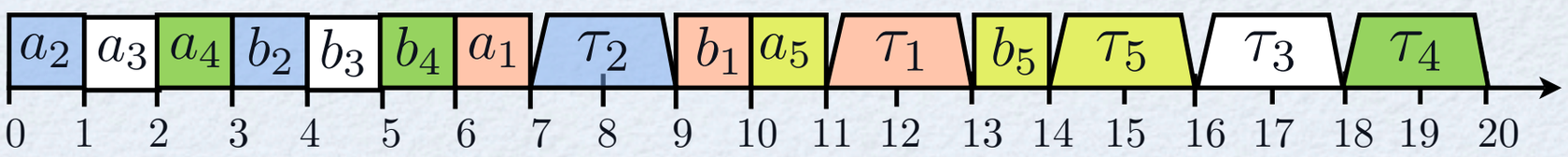
G_p : $A_1 \to \tau_1$, $A_2 \to \tau_2$, $A_3 \to \tau_3$, $A_4 \to \tau_4$, $A_5 \to \tau_5$



Graphe G



Graphe G_c

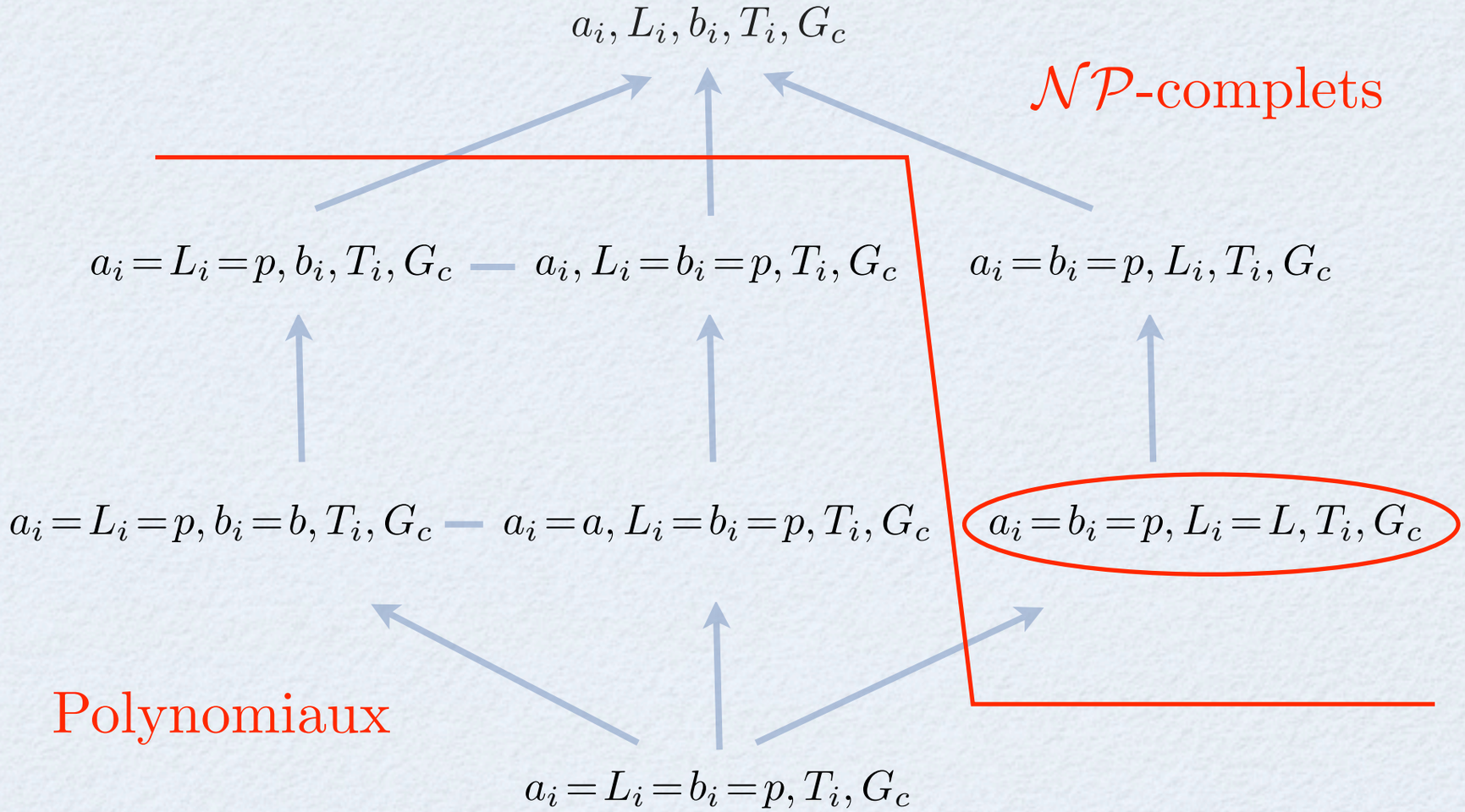


Ainsi

$1|\ prec,\ tâches-couplées,\ (a_i=b_i=1, L_i=L)\cup(\tau_i, pmtn), G_c|C_{max}$ $\mathcal{NP}$-complet

$1|\ prec,\ tâches-couplées,\ (a_i=b_i=p, L_i=L)\cup(\tau_i, pmtn), G_c|C_{max}$ $\mathcal{NP}$-complet

Ce qui entraîne

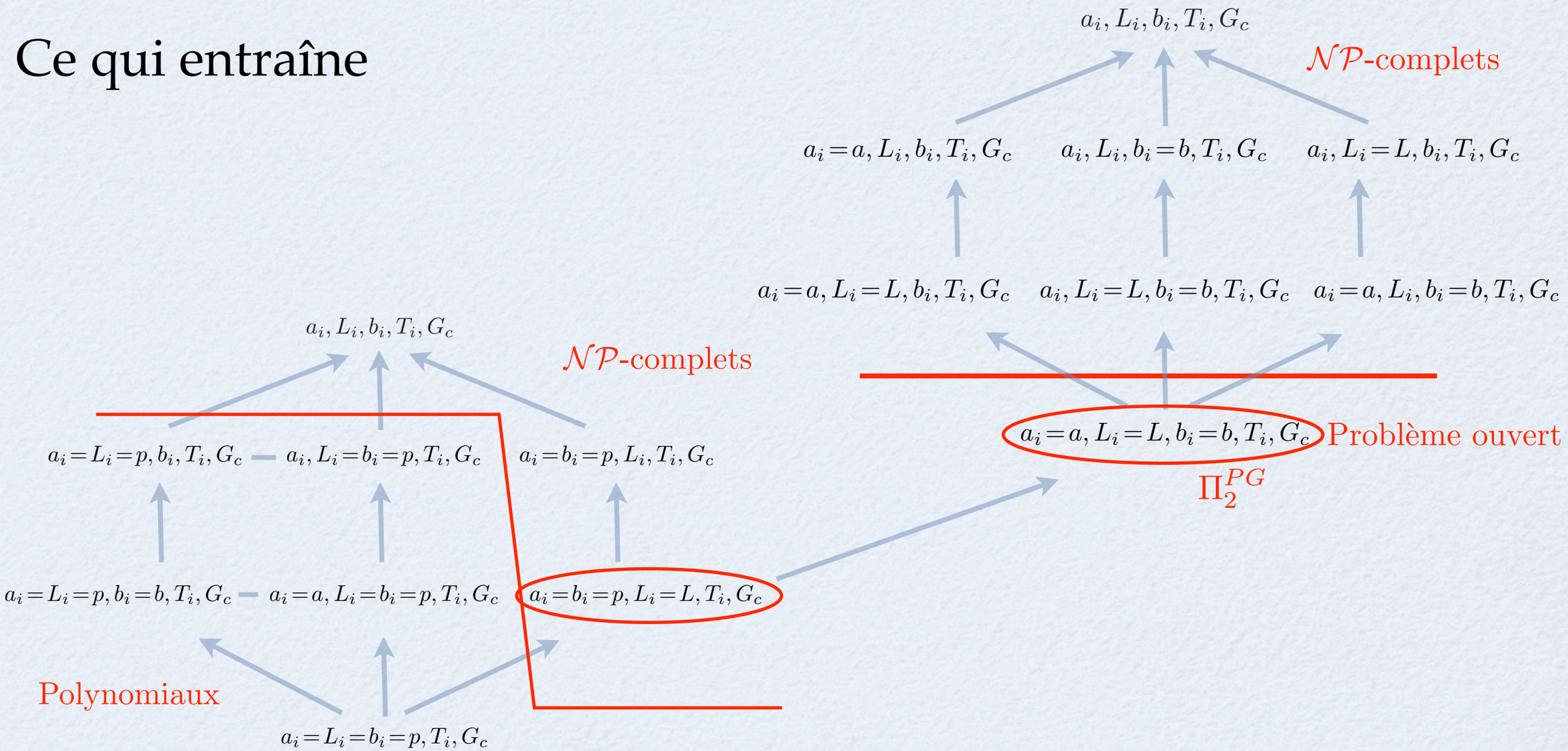


Ainsi

$1|\ prec,\ tâches-couplées,\ (a_i=b_i=1, L_i=L)\cup(\tau_i, pmtn), G_c|C_{max}$ $\mathcal{NP}$-complet

$1|\ prec,\ tâches-couplées,\ (a_i=b_i=p, L_i=L)\cup(\tau_i, pmtn), G_c|C_{max}$ $\mathcal{NP}$-complet

Ce qui entraîne

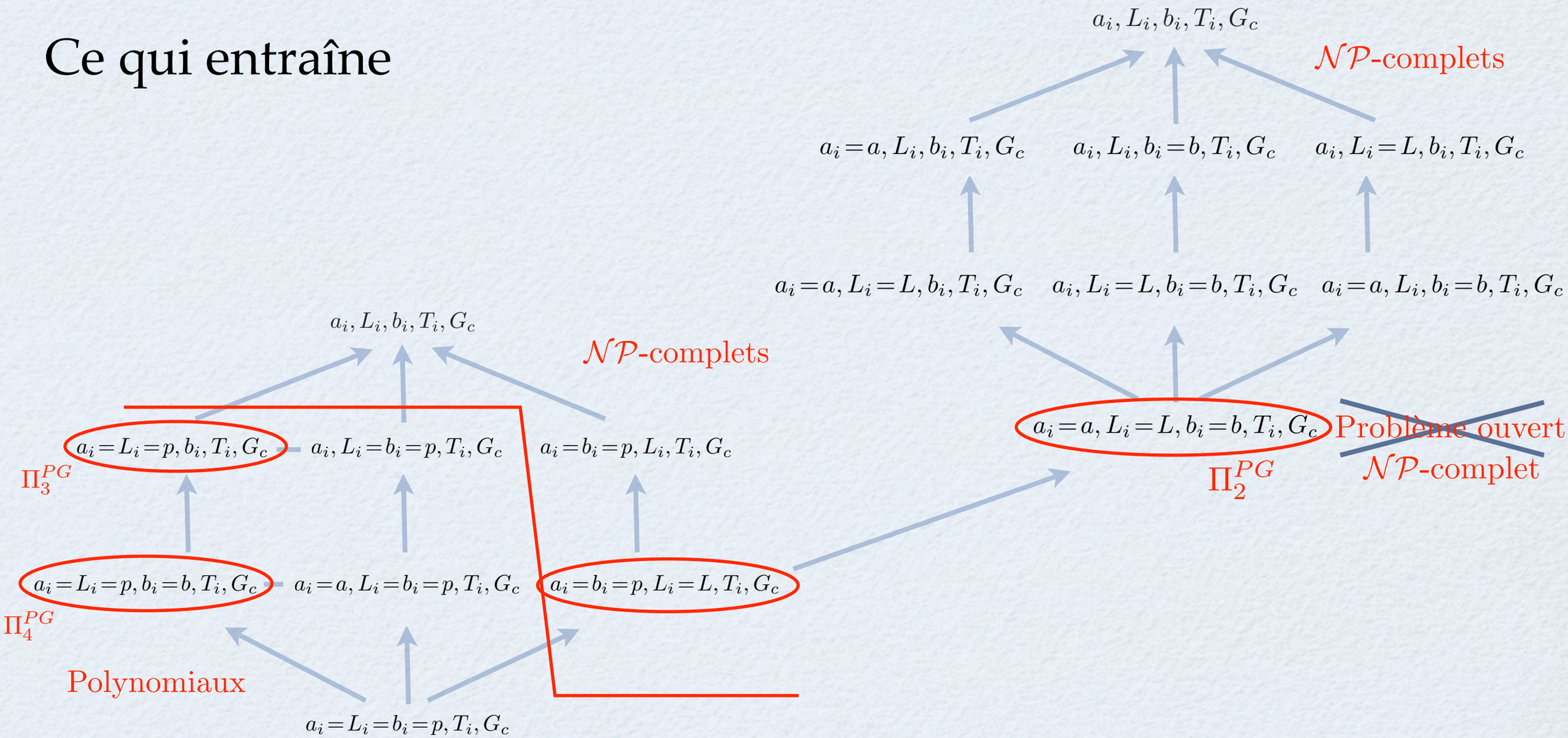


Ainsi

$1|\ prec,\ tâches-couplées,\ (a_i=b_i=1, L_i=L)\cup(\tau_i, pmtn), G_c|C_{max}$ $\mathcal{NP}$-complet

$1|\ prec,\ tâches-couplées,\ (a_i=b_i=p, L_i=L)\cup(\tau_i, pmtn), G_c|C_{max}$ $\mathcal{NP}$-complet

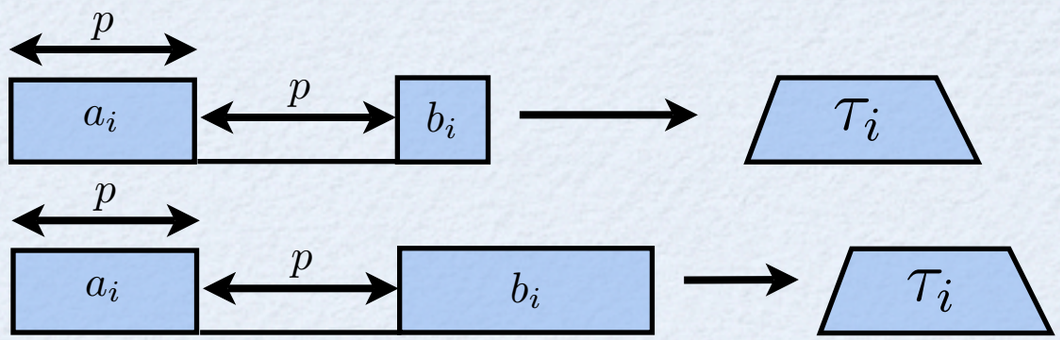
Ce qui entraîne



Avec graphe G_c et tâches de traitement : Π_3^{PG} et Π_4^{PG}

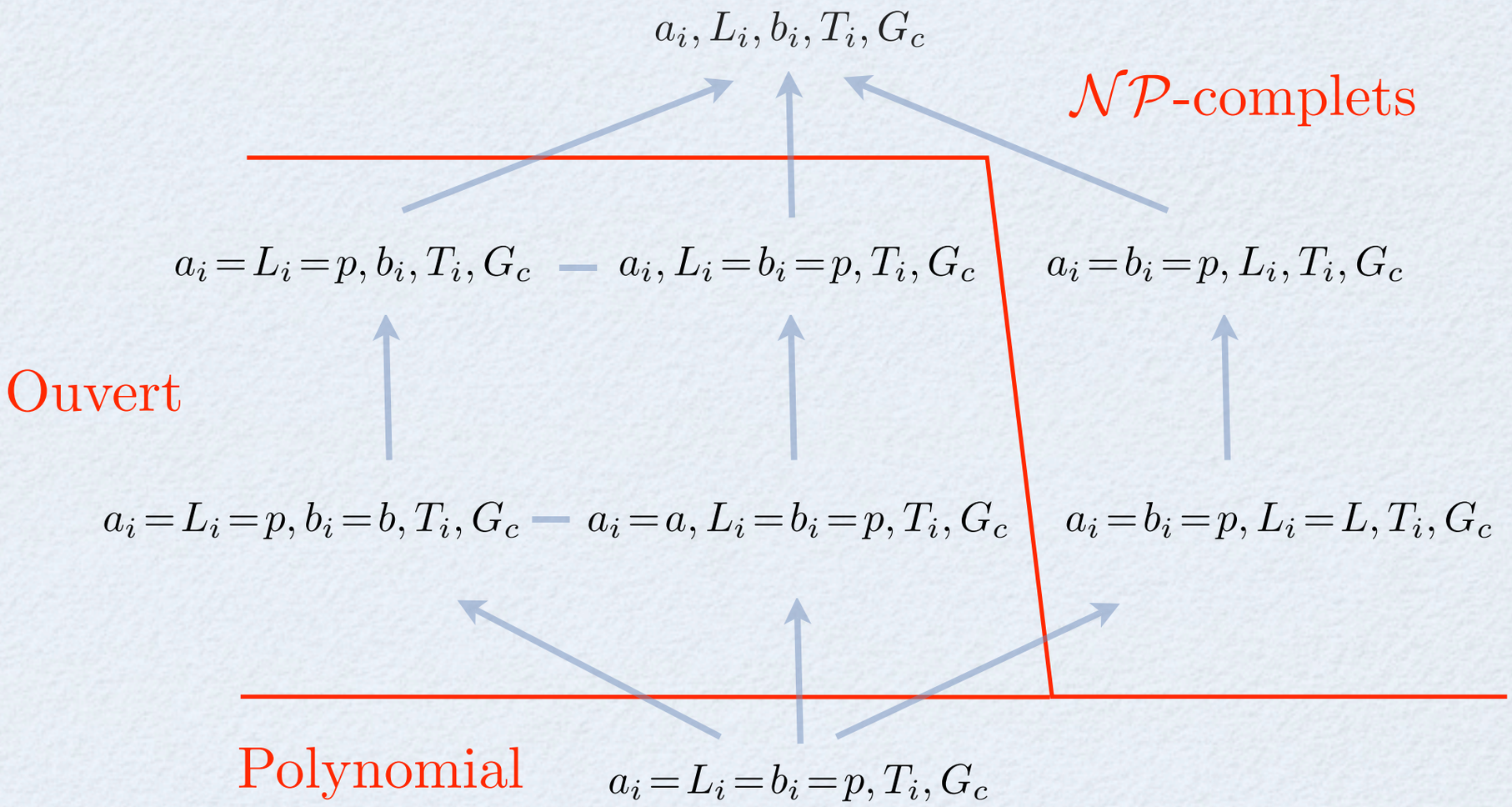
Conjecture

Le problème Π_3^{PG} : $(a_i = L_i = p, b_i) \cup \tau_i, G_c$ est $\mathcal{NP}$-complet



- Le problème avec le graphe de compatibilité est polynomial
- Le problème avec le graphe de précédence est polynomial
- Aucune solution optimale n'est associée à un couplage de taille maximum
- $\Pi_4^{PG} : (a_i = L_i = p, b_i = b) \cup \tau_i, G_c$ reste également ouvert

Au final, nous avons



Avec graphe G_c et tâches de traitement : conclusion

- Impact du graphe de compatibilité plus significatif
- Difficulté dans la classification de certains problèmes due aux tâches de traitement
- Analyse plus fine due aux tâches de traitement
- Utilisation des mêmes heuristiques d'approximation

Table des matières

Analyse des cas critiques : objectifs

- Deux problèmes changent de complexité selon les contraintes

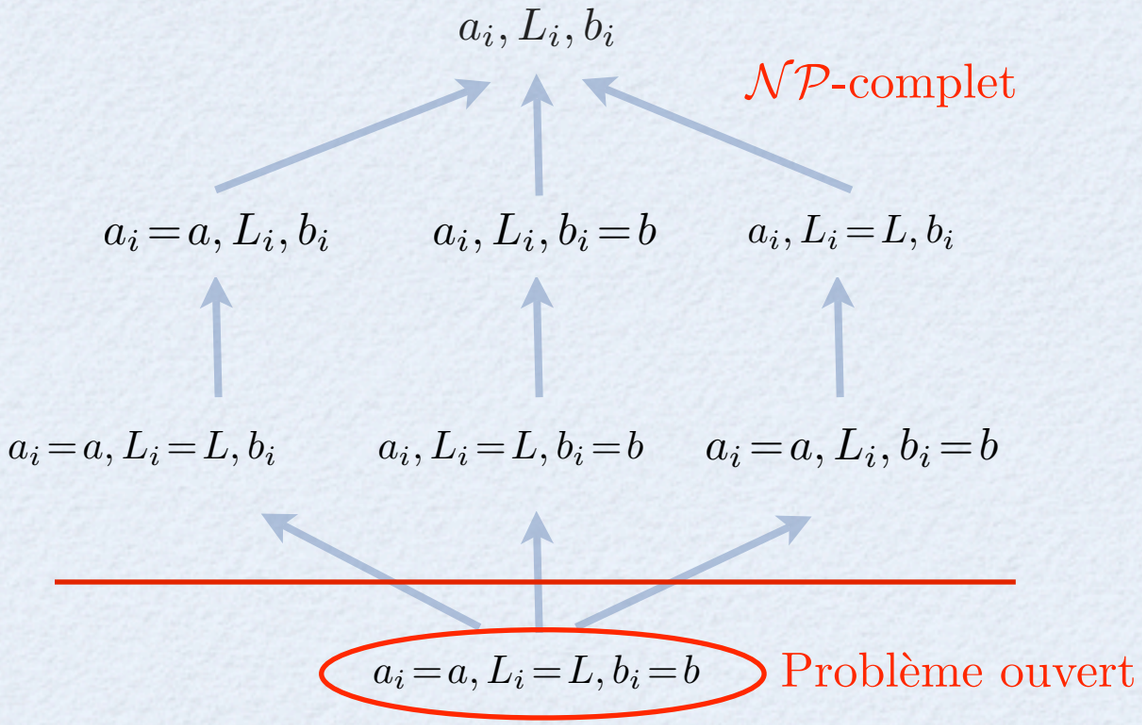
$$\Pi_2 = 1|a_i = a, L_i = L, b_i = b, G_c|C_{max}$$

$$\Pi_1^{PG} = 1|(a_i = b_i = p, L_i = L) \cup \tau_i, G_c|C_{max}$$

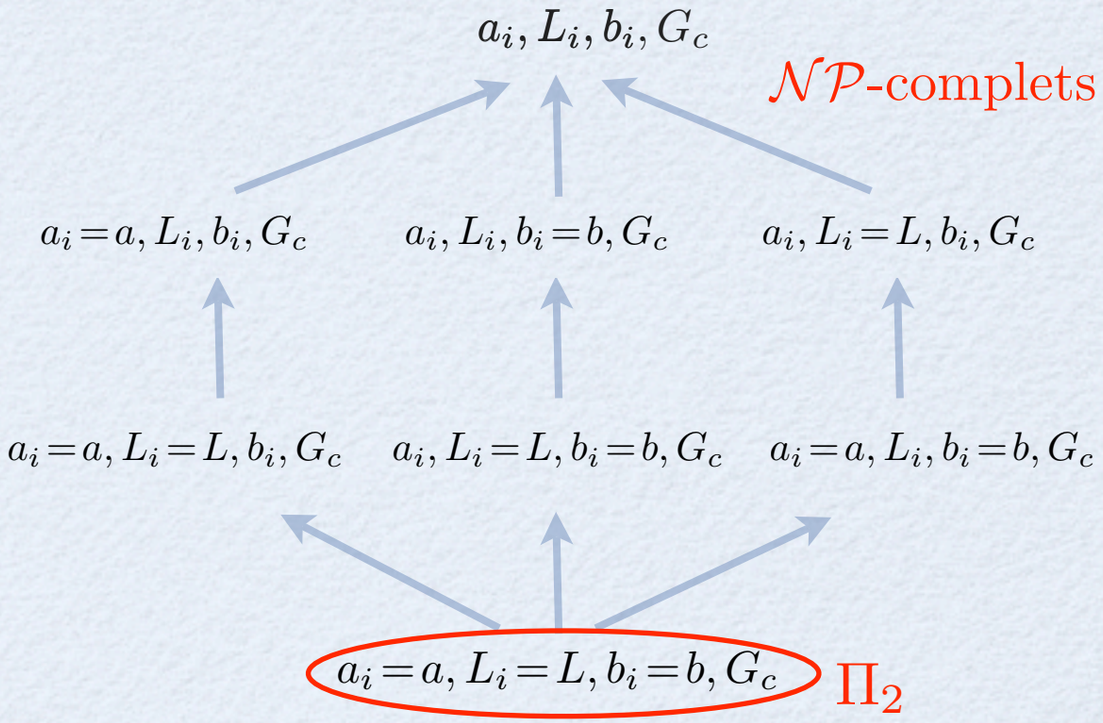
- Objectif de cette partie
  - Faire varier la valeur des paramètres pour trouver la limite entre les cas polynomiaux et ceux NP-complets
  - Développer des algorithmes d'approximation pour ces cas critiques.

Cas critique : problème Π_2

Orman et Potts [1997]



Simonin et al. [2009]



Cas critique : problème Π_2

- Nous allons étudier le cas particulier

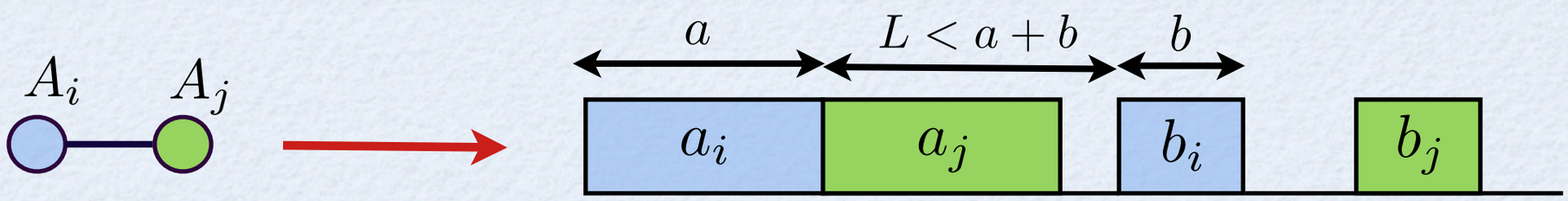
$\Pi_2 : 1|(a_i = a, L_i = L, b_i = b), G_c|C_{max}$ lorsque $L < a+b$ ou $L \geq a+b$

Théorème

Le problème de décision Π_2 est polynomial lorsque L<a+b

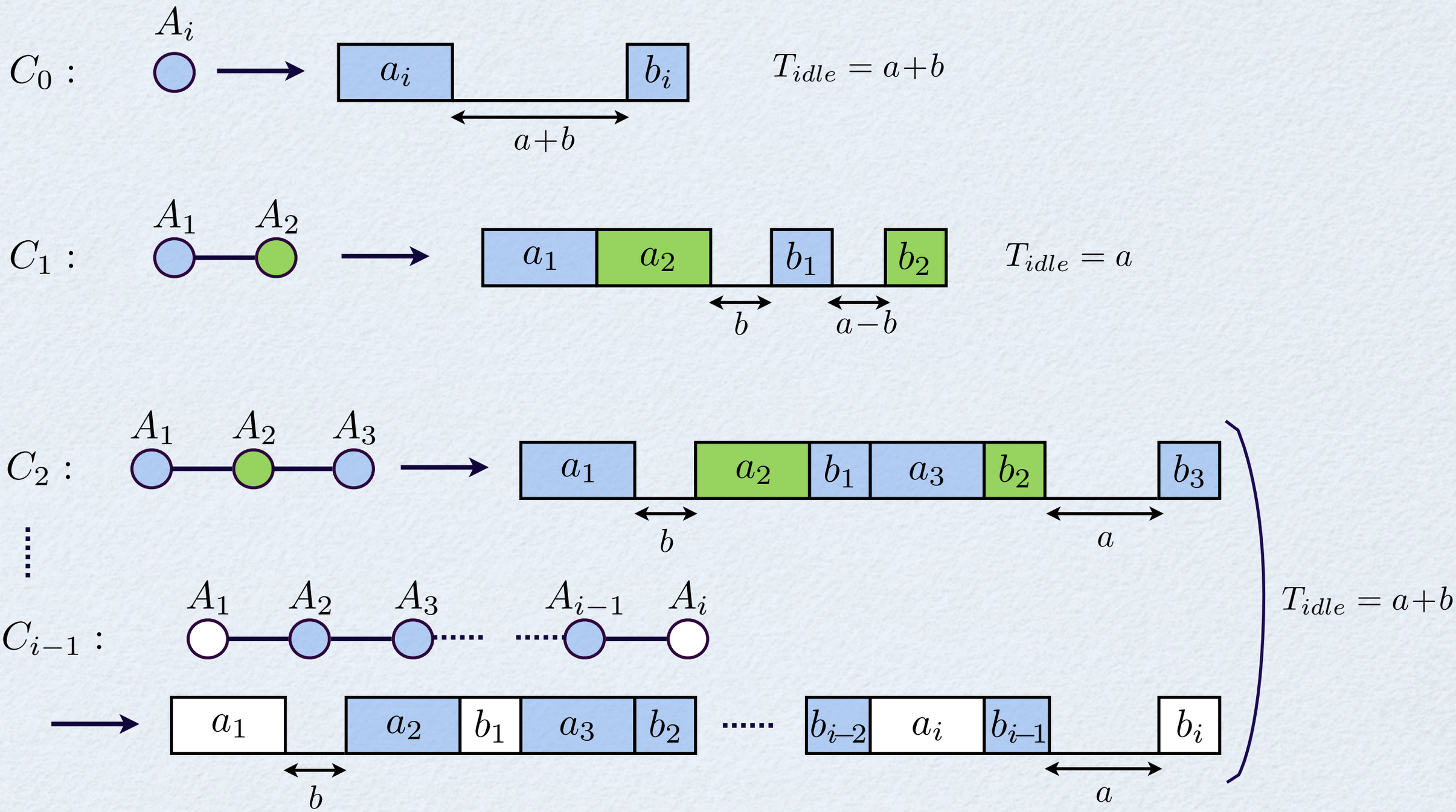
Preuve

Un couplage de taille maximum dans G_c donne une solution optimale



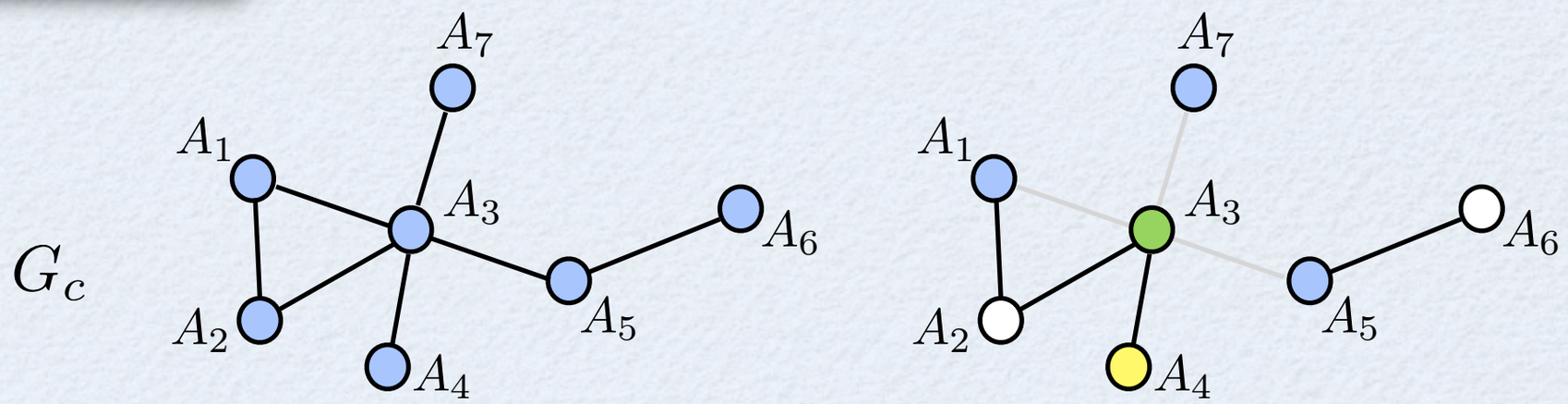
- Nous allons étudier le problème lorsque $L = a+b$, avec $a > b$

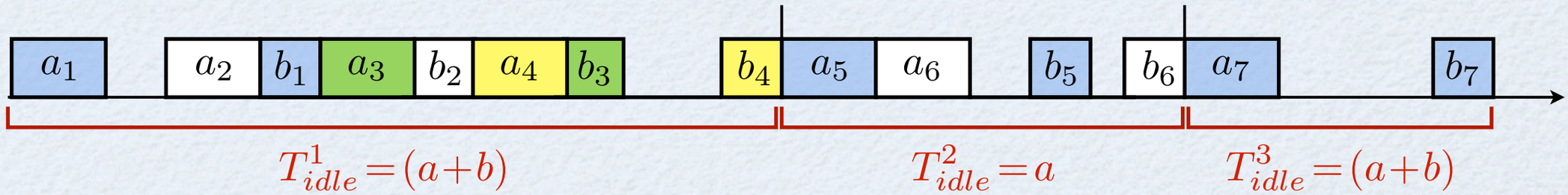
- Il existe trois ordonnancements possibles selon le recouvrement



Un ordonnancement $\longleftrightarrow$ Une partition en chaînes disjointes de G_c

Exemple





$$C_{max} = T_{seq} + T_{idle} = 7(a+b) + (a+b) + a + (a+b)$$

Un ordonnancement optimal $\longleftrightarrow$ Couverture minimum en chaînes disjoints liée à un ordonnancement (Min-CCDLO)

Définition de Min-CCDLO

- INSTANCE : Soit $G=(V,E)$ un graphe d'ordre n
  Soit a et b deux entiers avec $b < a$
- RESULTAT : Une Partition $\mathcal{P}$ des sommets de G en chaînes disjointes
- OBJECTIF : Minimiser $\sum_{p\in\mathcal{P}} w(p)$ où $w(p) = a$ si p est une arête, et $w(p)=a+b$ autrement

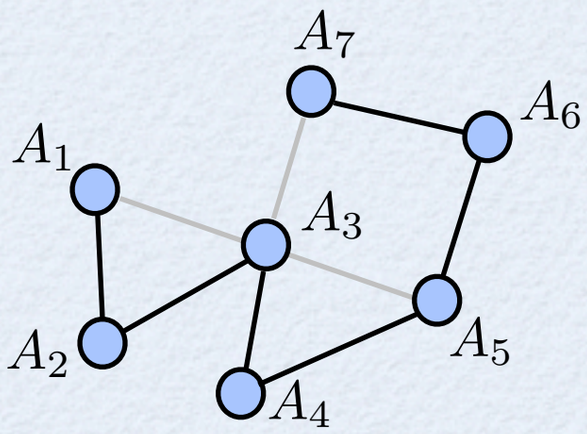
Cas critique : complexité de Min-CCDLO

Théorème

Le problème de décision Min-CCDLO est $\mathcal{NP}$-complet

Preuve

Réduction de Chaîne Hamiltonienne vers Min-CCDLO où $C_{max} = (n+1)(a+b)$



$\longleftrightarrow$ $T_{idle} = a+b$ $\longleftrightarrow$ $C_{max} = (n+1)(a+b)$

□

Cas critique : approximation de Min-CCDLO

Résultats d'approximation obtenus pour Min-CCDLO

- Un couplage maximum donne :
  - une $[\frac{5}{4}, \frac{3}{2}]$- approximation en fonction de a et b

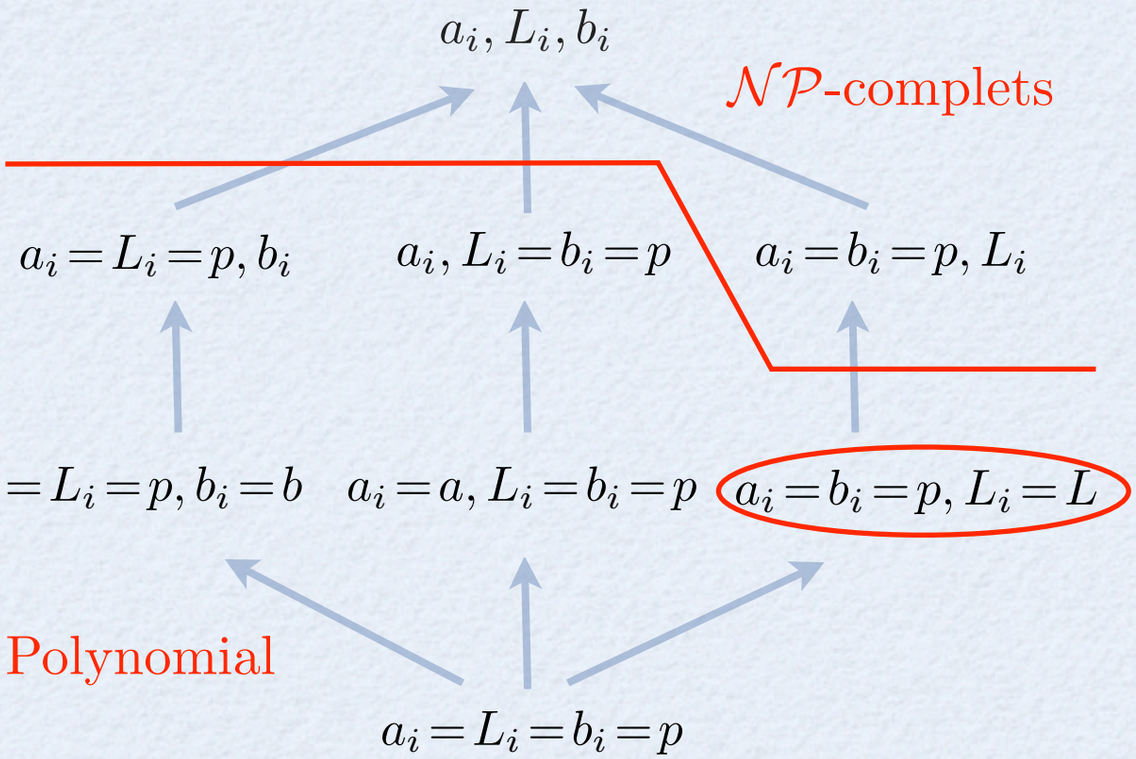
- Min-CCDLO est proche du problème Couverture Minimum par Chaînes Disjointes (Min-CCD)
- Pour une ρ_{CCD}-approximation de Min-CCD, nous obtenons :

$$\rho_{CCDLO} \leq min \left\{ \rho_{CCD} \times \left(\frac{a+b}{a} \right), \frac{3a+2b}{2a+2b} \right\}$$

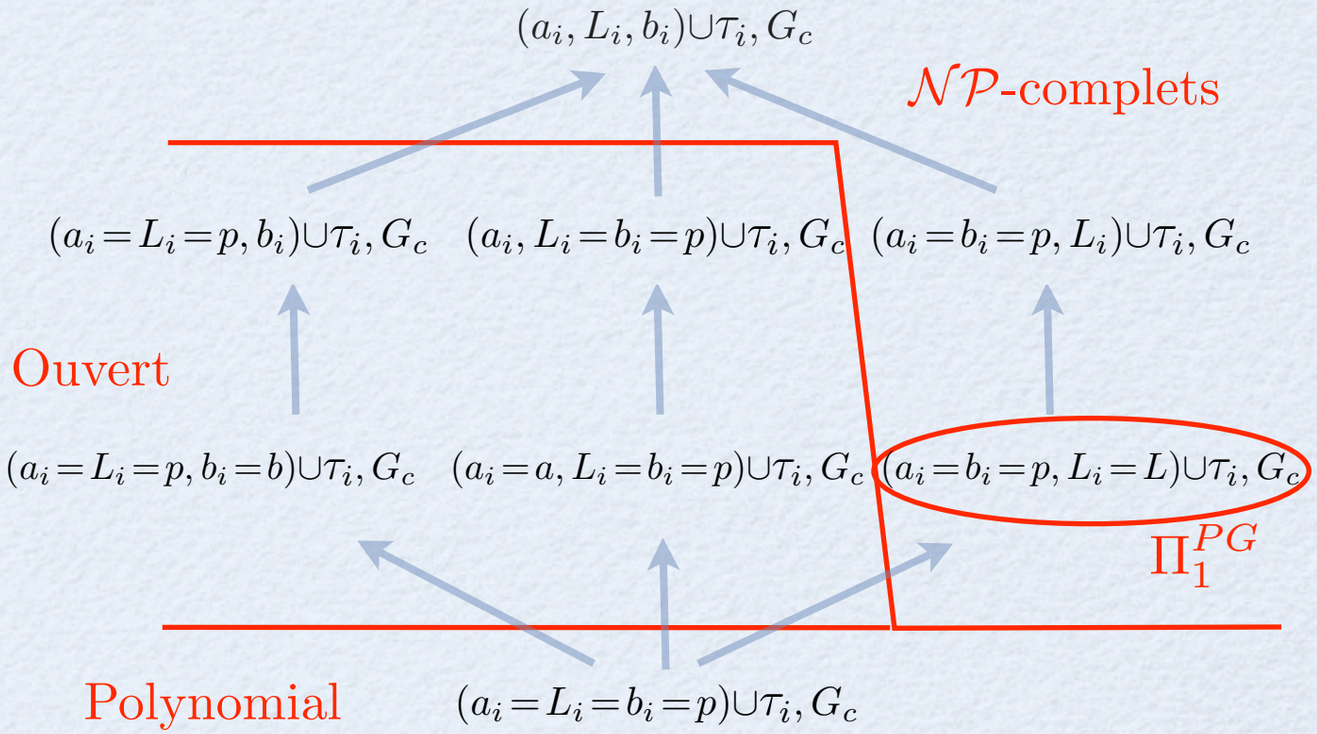
- Lorsque $\rho_{CCD} = 1$ pour certaines topologies : $\rho_{CCDLO} \leq \frac{1+\sqrt{3}}{2} \cong 1,37$

Cas critique : problème Π_1^{PG}

Orman et Potts [1997]



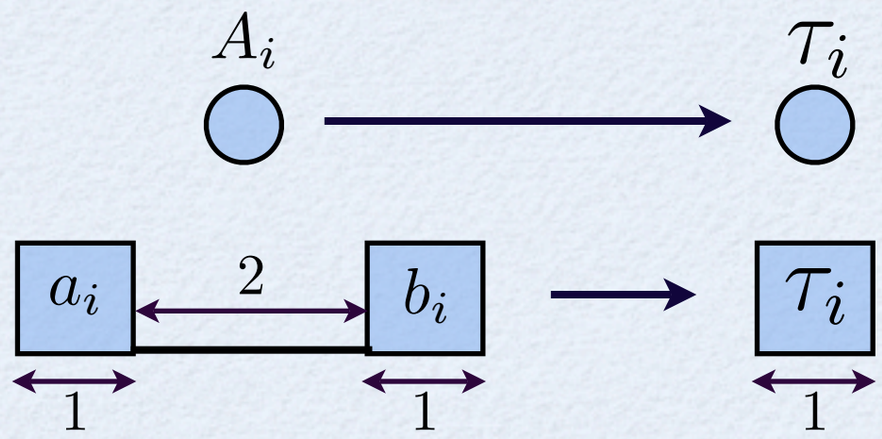
Simonin et al. [2009]



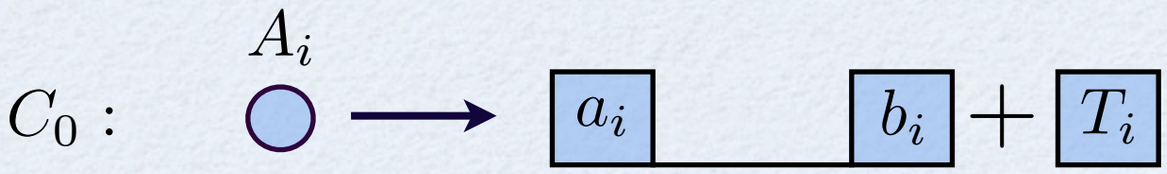
Cas critique : complexité

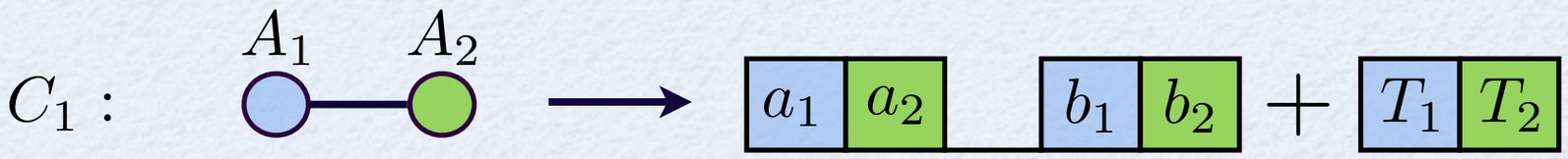
- Nous avons vu que :
  - le problème est polynomial lorsque $L = 1$
  - le problème est $\mathcal{NP}$-complet lorsque $L \geq 2$

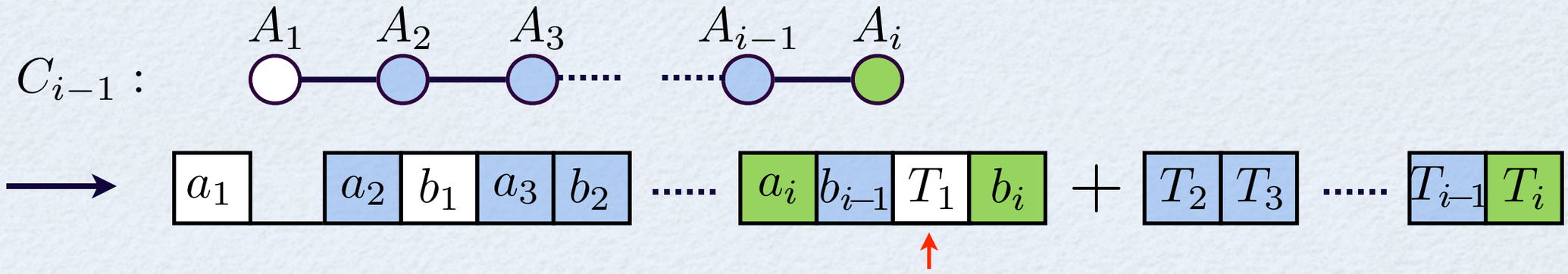
- Approximation d'un cas particulier où :
  - Toutes les tâches-couplées ont un temps d'inactivité $L = 2$
  - Toutes les tâches de traitement ont un temps d'exécution $\tau_i = 1$

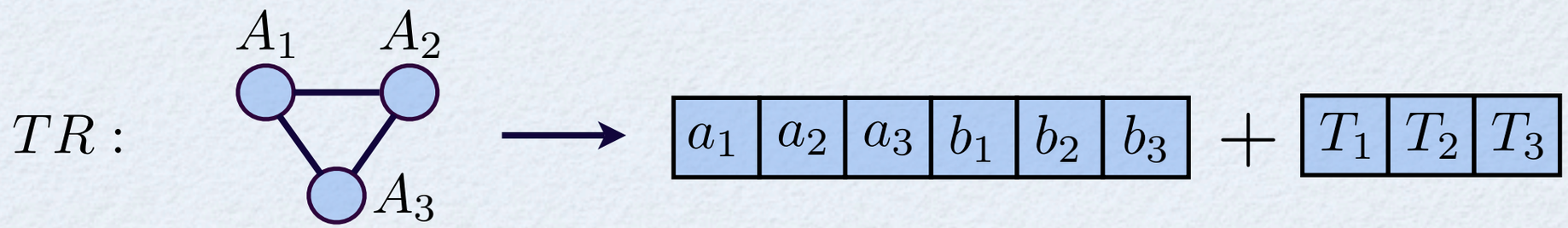


- En fonction de G_c, il y a quatre ordonnancements possibles









- Nous avons deux problèmes :
  - Sans triangle
  - Avec triangles

- Les deux cas sont $\mathcal{NP}$-complets
  - Sans triangle → Chaîne Hamiltonienne
  - Avec triangles → Triangle Packing

- Stratégies d'approximation :
  - Recouvrir par des chaînes disjointes [Steiner]
  - Recouvrir un maximum de sommets avec des arêtes et des chaînes de longueur 2 → 2-recouvrement

Cas critique : 2-recouvrement

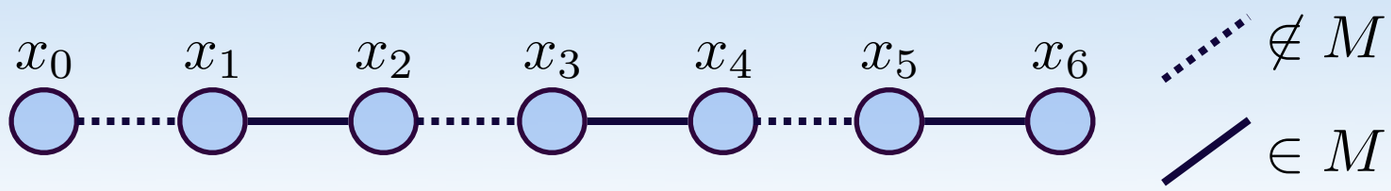
Définition [2-recouvrement]

Ensemble d'arête tel que les composantes connexes induites sont :

- des sommets isolés
- des arêtes
- des chaînes de longueur 2

Définition [Chaîne M-alternée]

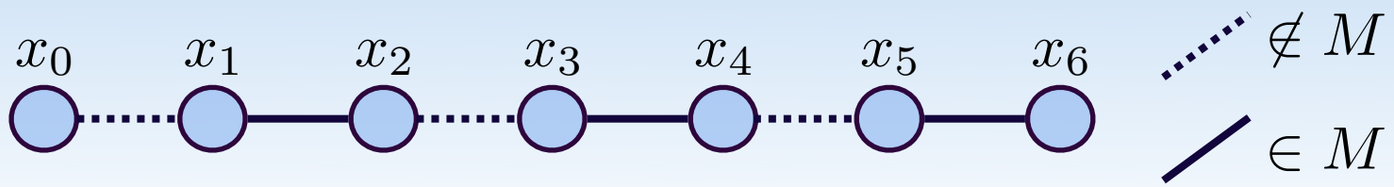
Similaire à une chaîne alternée classique [Berge]



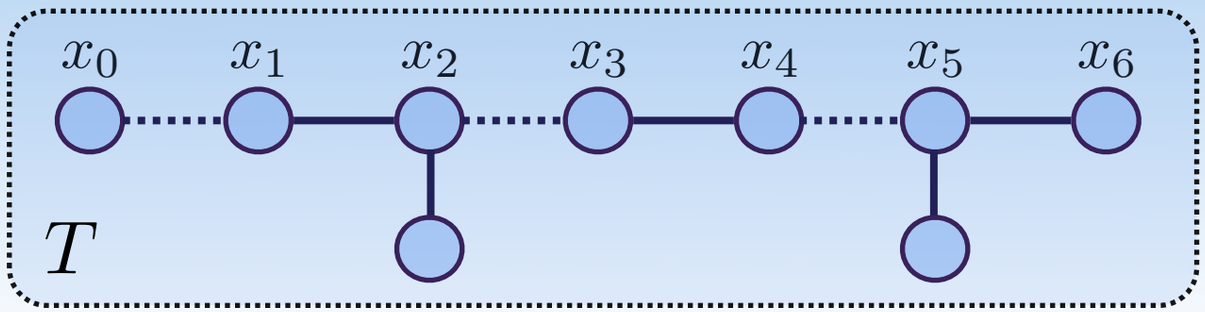
Cas critique : 2-recouvrement

Définition [Chaîne M-alternée]

Similaire à une chaîne alternée classique [Berge]



Définition [Colonne vertébrale d'une chaîne M-alternée]



Cas critique : 2-recouvrement

Définition [Chaîne M-alternée augmentante]

C est une chaîne M-alternée augmentante si le nombre de sommets saturés par M augmente en changeant l'appartenance à M des arêtes de C

Lemme

$C = x_0, x_1, \ldots, x_k$ est une chaîne M-alternée augmentante si et seulement si :

- $\exists\ x_{2i-1}, i \in I\!N^*$ tel que $d_M(x_{2i-1}) \neq 2$

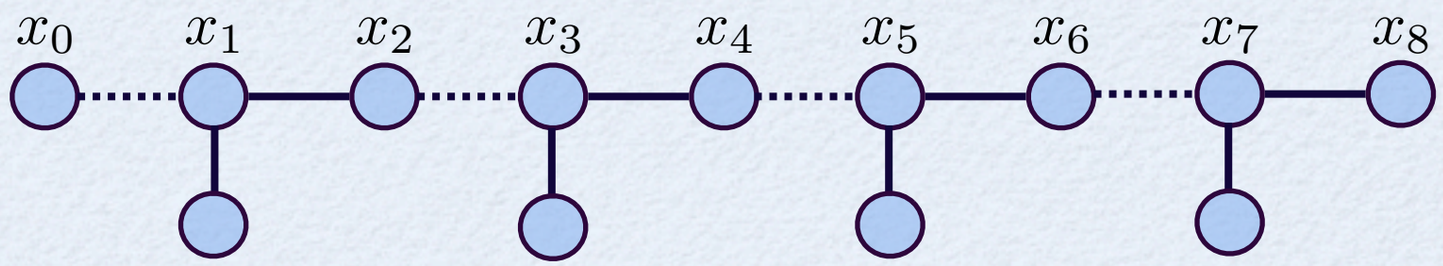
Cas critique : 2-recouvrement

Lemme

$C = x_0, x_1, \ldots, x_k$ est une chaîne M-alternée augmentante si et seulement si :

- $\exists\ x_{2i-1}, i \in \mathbb{N}^*$ tel que $d_M(x_{2i-1}) \neq 2$

- Exemple de chaîne non augmentante



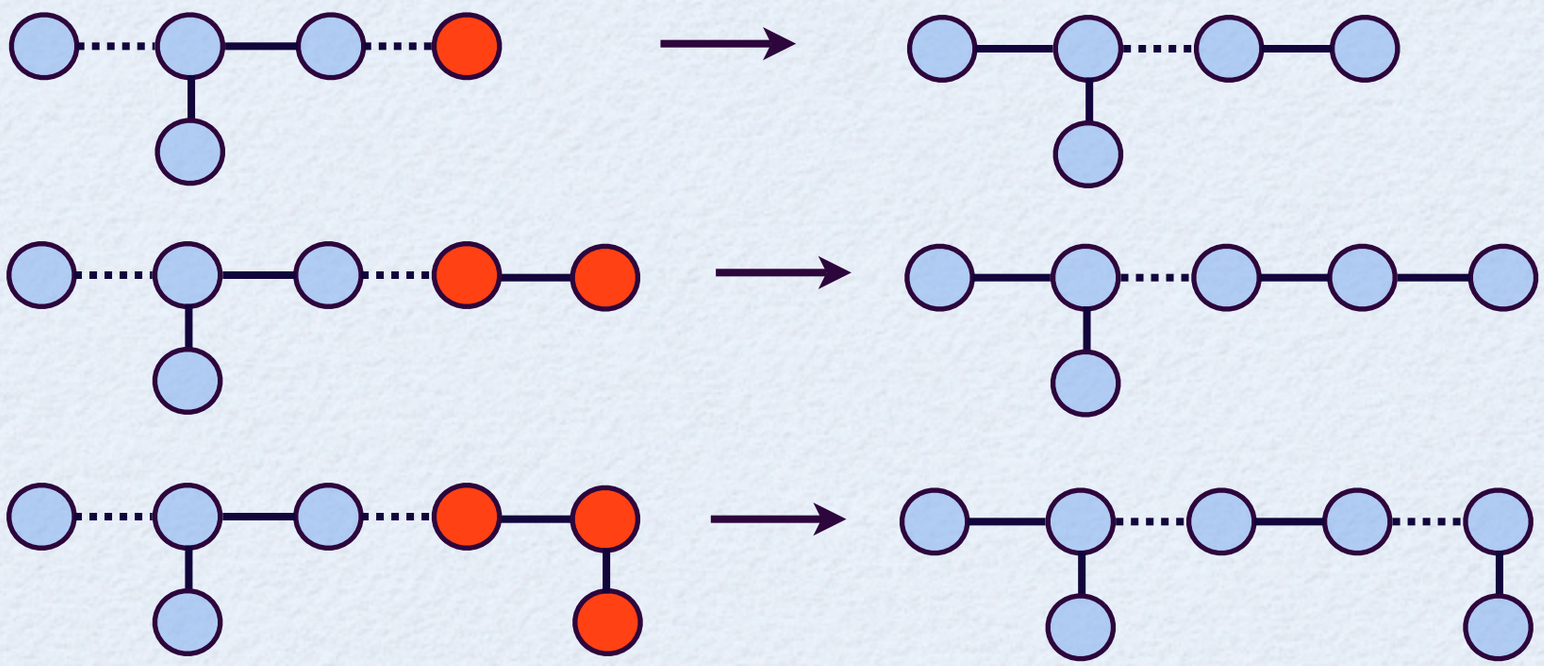
Cas critique : 2-recouvrement

Lemme

$C = x_0, x_1, \ldots, x_k$ est une chaîne M-alternée augmentante si et seulement si :

- $\exists\ x_{2i-1}, i \in \mathbb{N}^*$ tel que $d_M(x_{2i-1}) \neq 2$

- Chaînes augmentantes :



Cas critique : 2-recouvrement

Lemme

$C = x_0, x_1, \ldots, x_k$ est une chaîne M-alternée augmentante si et seulement si :

- $\exists\ x_{2i-1}, i \in \mathbb{N}^*$ tel que $d_M(x_{2i-1}) \neq 2$

Théorème

Un 2-recouvrement M est maximum dans un graphe G

$\Leftrightarrow$ G ne possède pas de chaîne M-alternée augmentante

Cas critique : approximation

- Étude de deux cas

Sans triangle

La couverture optimale contient :

C_0 : A_i

C_1 : A_1 A_2

C_{i-1} : A_1 A_2 ..... A_{i-1} A_i

$$\rho \leq \frac{C^h_{max}}{C^{opt}_{max}} < \frac{13}{12}$$

Avec triangles

La couverture optimale contient :

C_0 : A_i

TR : A_1 A_2 A_3

C_1 : A_1 A_2

C_{i-1} : A_1 A_2 .... A_{i-1} A_i

$$\rho \leq \frac{C^h_{max}}{C^{opt}_{max}} < \frac{10}{9}$$

Table des matières

Conclusion

- Présentation d'un nouveau problème d'ordonnancement avec tâches-couplées
- Apparition de la contrainte d'incompatibilité et/ou des tâches de traitement
- Conclusion sur la complexité :
  - Classification des problèmes selon la théorie de la complexité
  - Quelques problèmes restent ouverts
  - Le graphe de compatibilité a plus d'impact que les tâches de traitement
  - Les tâches de traitement induisent des preuves plus complexes

Conclusion

- Conclusion sur l'approximation :
  - Développer des algorithmes avec des garanties de performances non triviales basées sur :
    - couplage de taille maximum
    - recouvrement des sommets par des cliques de taille maximale
    - poupées russes
- Le ratio dépend souvent d'un des paramètres
- L'introduction des tâches de traitement n'a pas d'influence sur le ratio
- Extension de la notion de couplage par des chaînes de longueur au plus 2

Perspectives

- A court terme :
  - Établir la complexité des problèmes ouverts
  - Visualisation globale de l'approximation
  - Étude de la complexité et l'approximation selon la topologie de G_c

- À long terme :
  - Étude des problèmes on-line
  - Étude des problèmes cycliques
  - Introduire des contraintes de précédence pour les tâches d'acquisition
  - Étude stochastique
  - Architecture multi-processeurs

MERCI POUR VOTRE ATTENTION

G.SIMONIN