

Problèmes d'ordonnancement résolus en temps polynomial

Claude-Guy Quimper
en collaboration avec
Alejandro López-Ortiz

Problèmes d'ordonnancement

- Dans quel ordre doit-on exécuter les tâches?
- Qui doit exécuter les tâches?

Contraintes sur les tâches

- Date de début
- Échéance
- Temps de traitement
- Contraintes d'antériorité
- Prémption
- ...

Contraintes de ressources

- Nombre de ressources.
- Ressources exécutant plus vite certaines tâches.
- Ressources exécutant seulement un sous-ensemble des tâches.
- ...

Objectifs

- Minimiser le makespan
- Minimiser la somme des temps d'achèvement
- Minimiser les retards
- ...

Classification

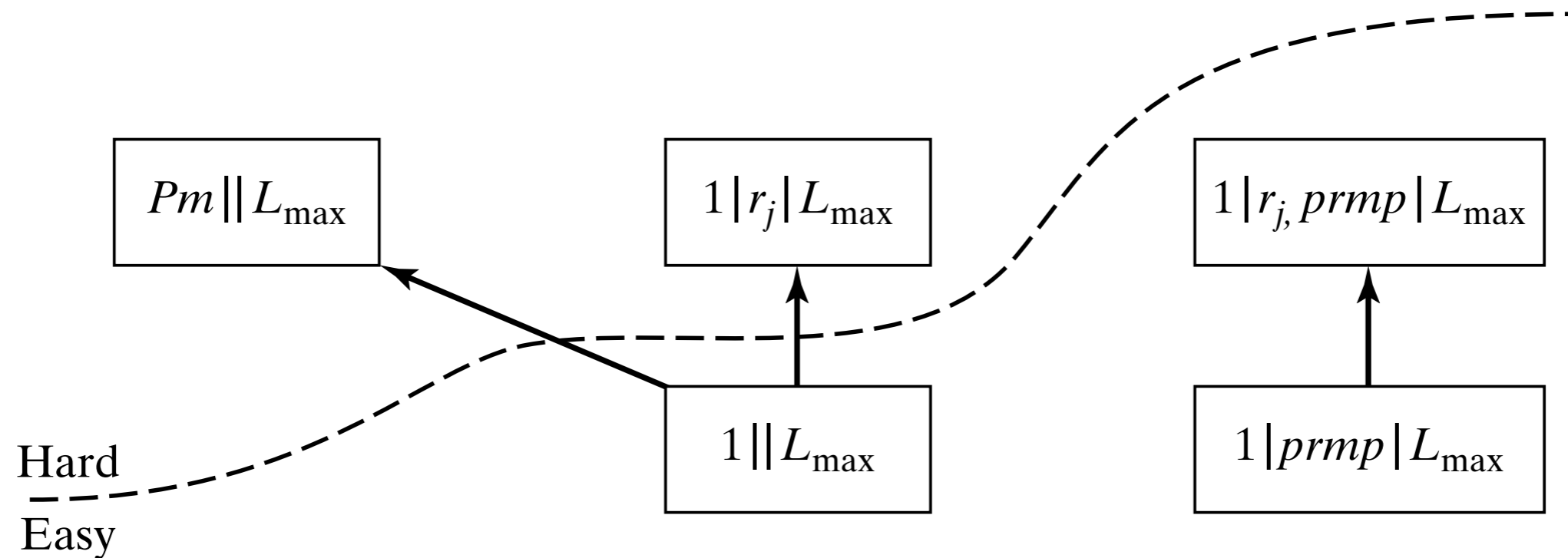


Fig. 2.9 Complexity hierarchy of problems in Example 2.4.2

Source: Michael L. Pinedo, *Scheduling, Theory, Algorithms, and Systems*, Third Edition

Un premier problème polynomial

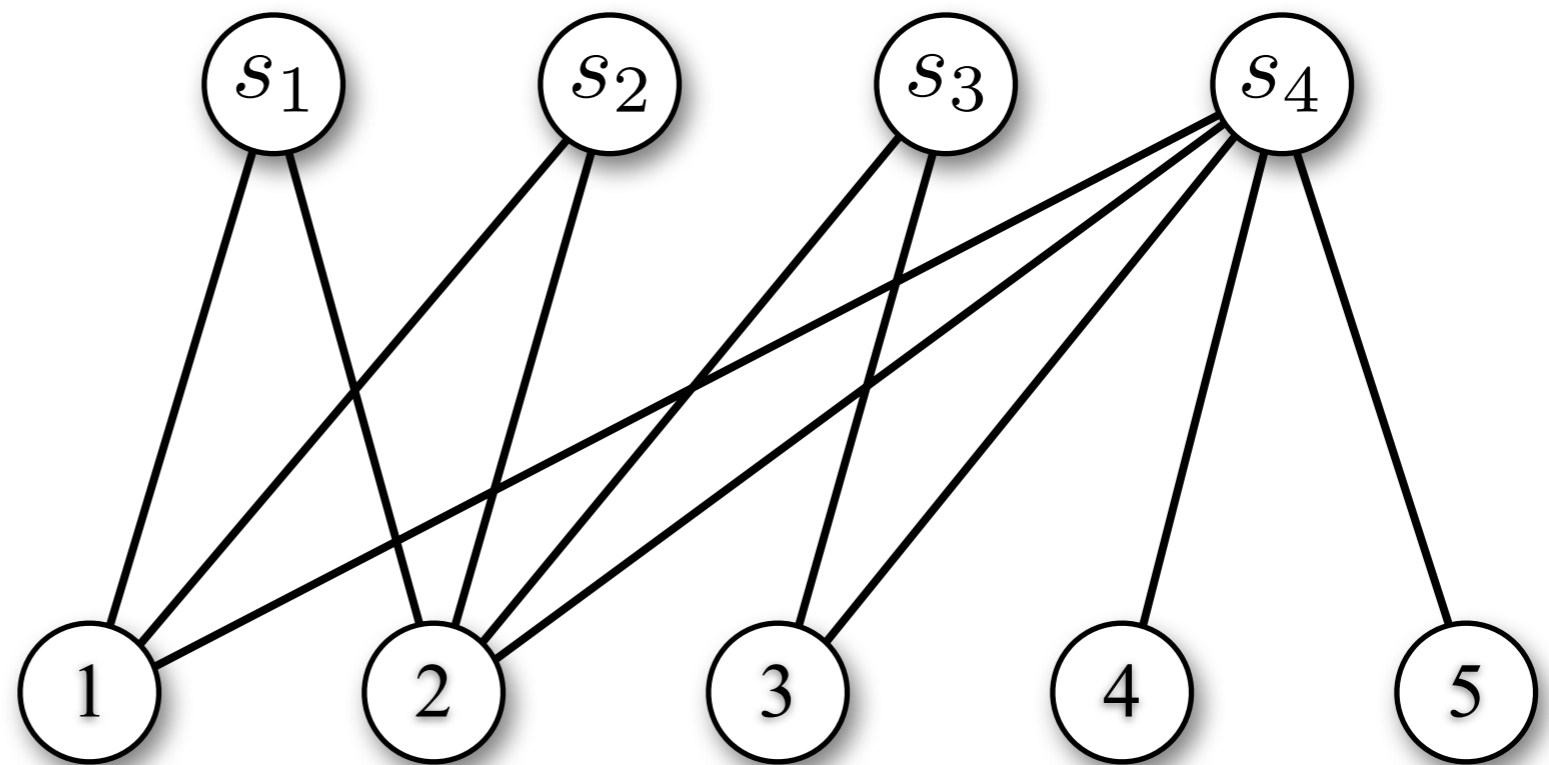
- Une seule ressource $s_1 \in [1, 2], s_2 \in [1, 2]$
- Une tâche doit commencer au plus tôt au temps a et au plus tard au temps b .
 $s_3 \in [2, 3], s_4 \in [1, 5]$
- Des temps de traitement unitaires

Un premier problème polynomial

- Couplage dans un graphe biparti convexe.

$$s_1 \in [1, 2], s_2 \in [1, 2]$$

$$s_3 \in [2, 3], s_4 \in [1, 5]$$

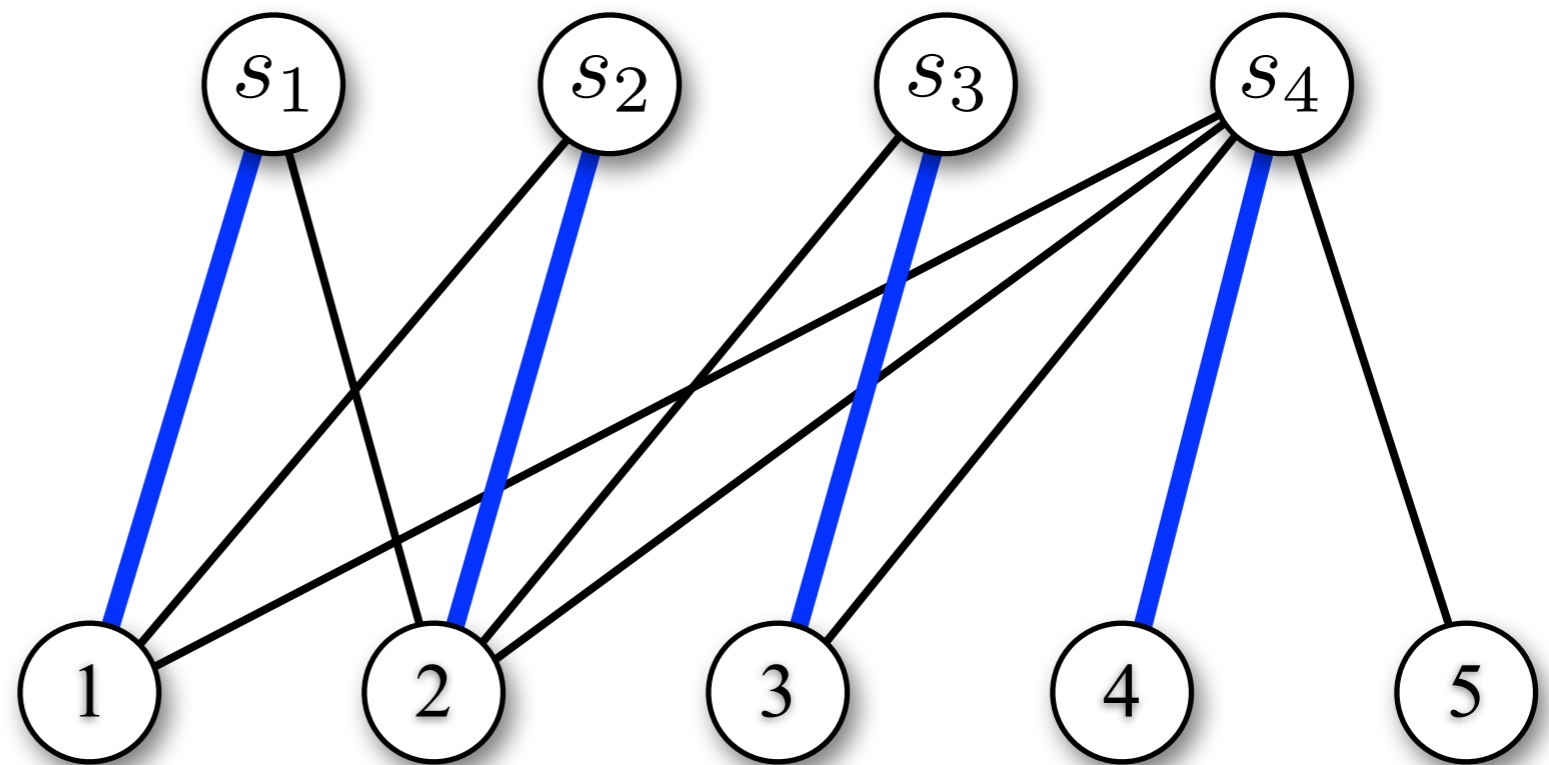


Un premier problème polynomial

- Couplage dans un graphe biparti convexe.

$$s_1 \in [1, 2], s_2 \in [1, 2]$$

$$s_3 \in [2, 3], s_4 \in [1, 5]$$

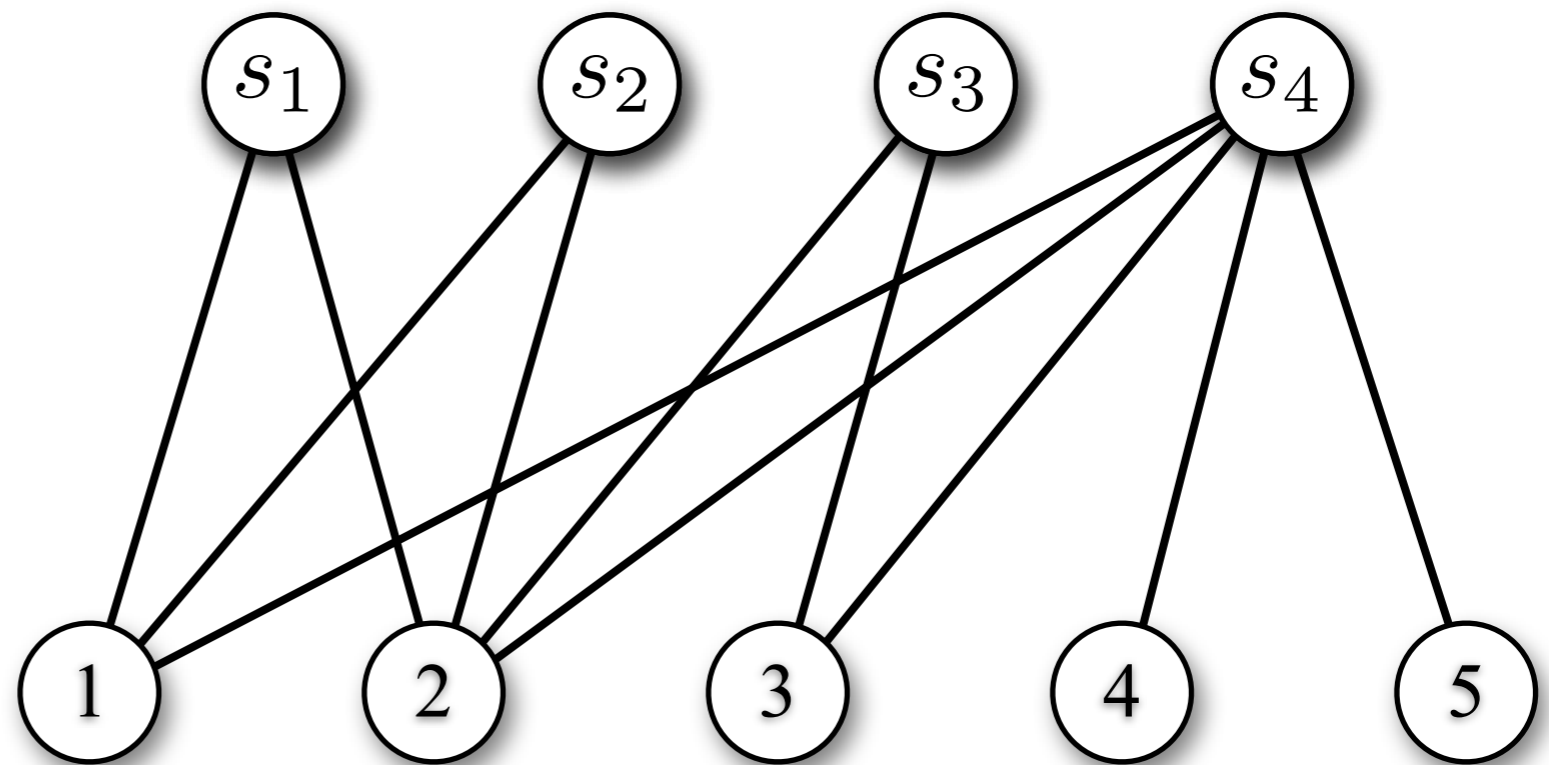


Un premier problème polynomial

- Couplage dans un graphe biparti convexe.

$$s_1 \in [1, 2], s_2 \in [1, 2]$$

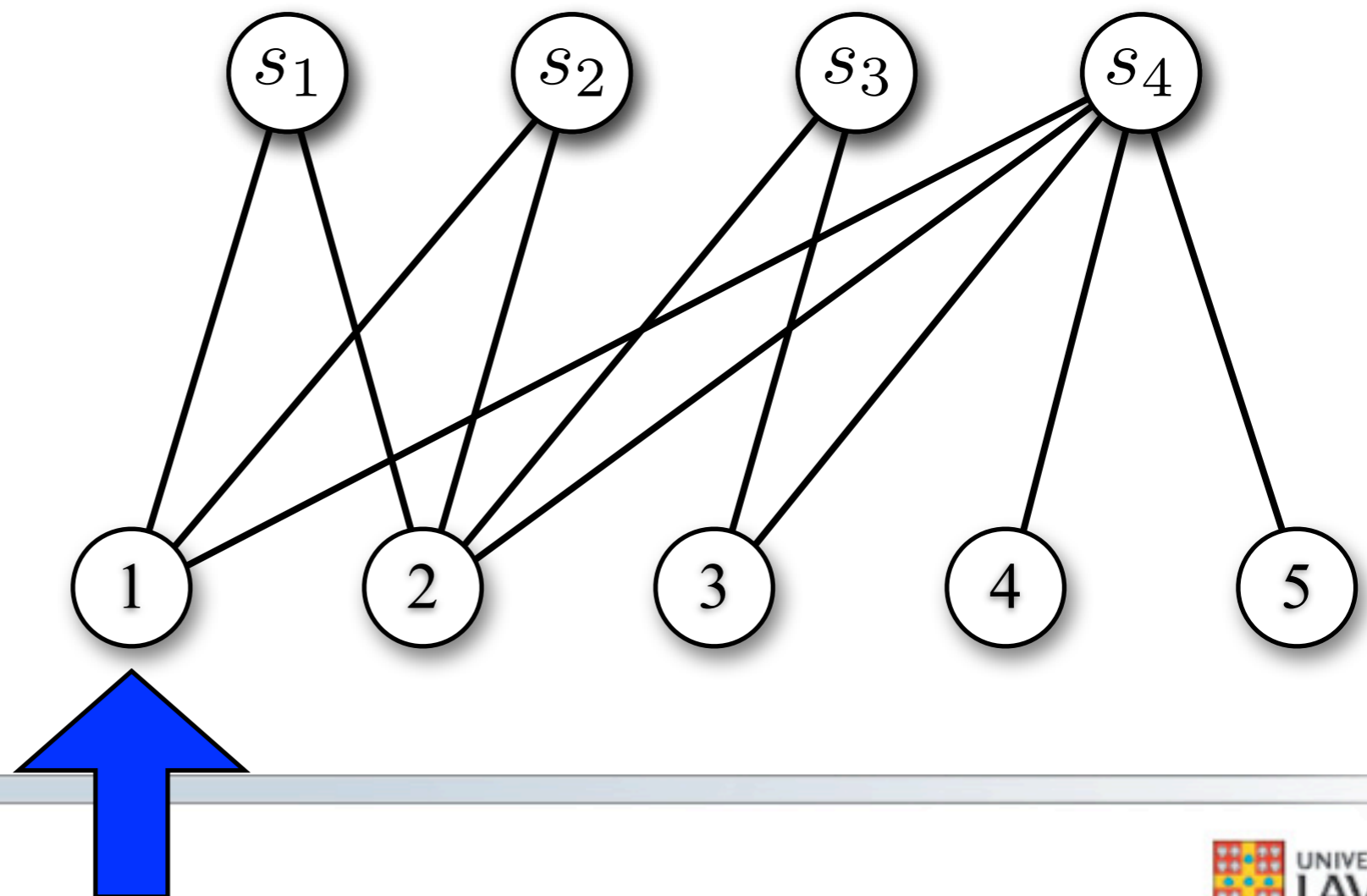
$$s_3 \in [2, 3], s_4 \in [1, 5]$$



Un premier problème polynomial

- Couplage dans un graphe biparti convexe.

$$s_1 \in [1, 2], s_2 \in [1, 2]$$
$$s_3 \in [2, 3], s_4 \in [1, 5]$$

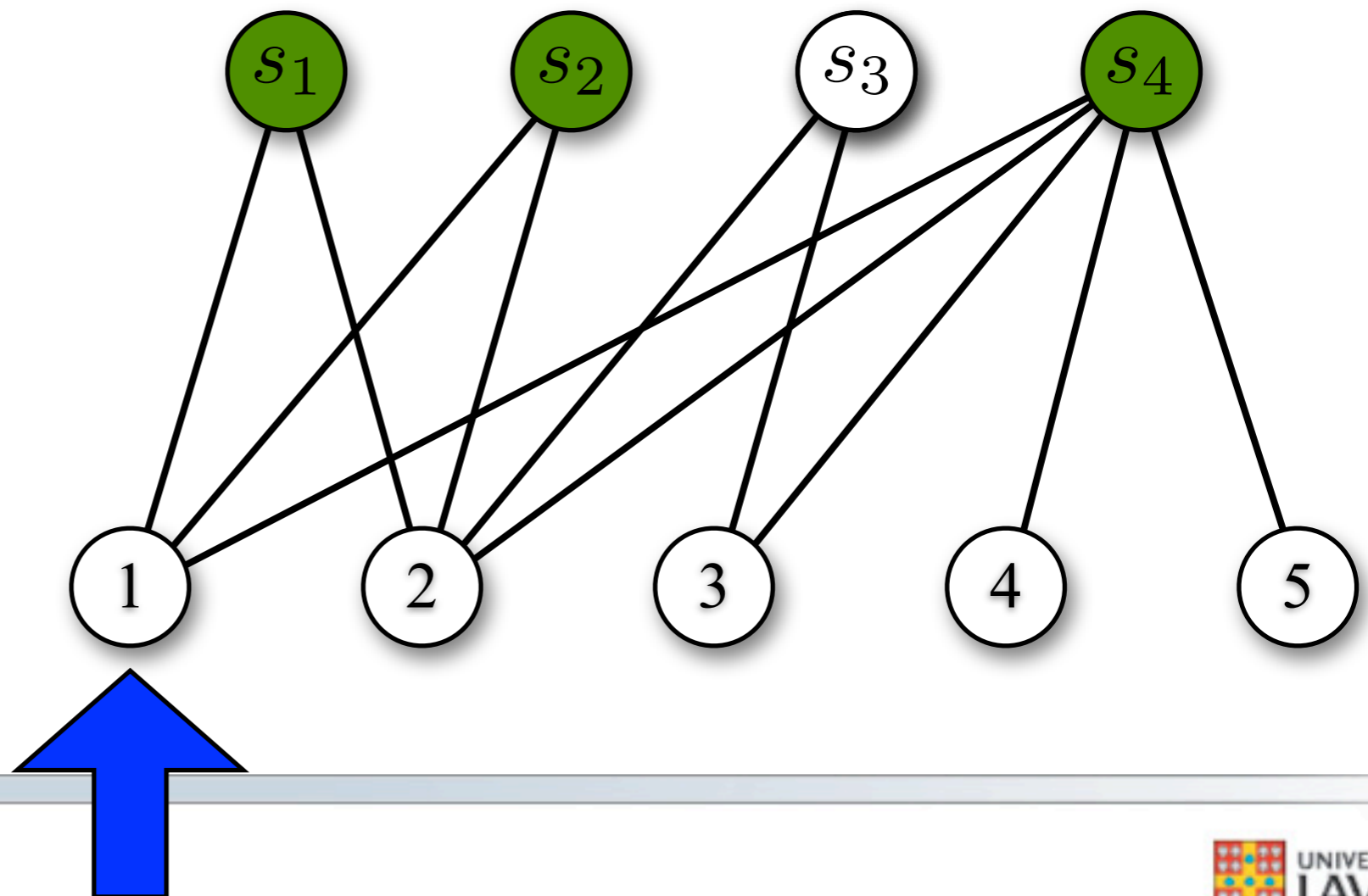


Un premier problème polynomial

- Couplage dans un graphe biparti convexe.

$$s_1 \in [1, 2], s_2 \in [1, 2]$$

$$s_3 \in [2, 3], s_4 \in [1, 5]$$

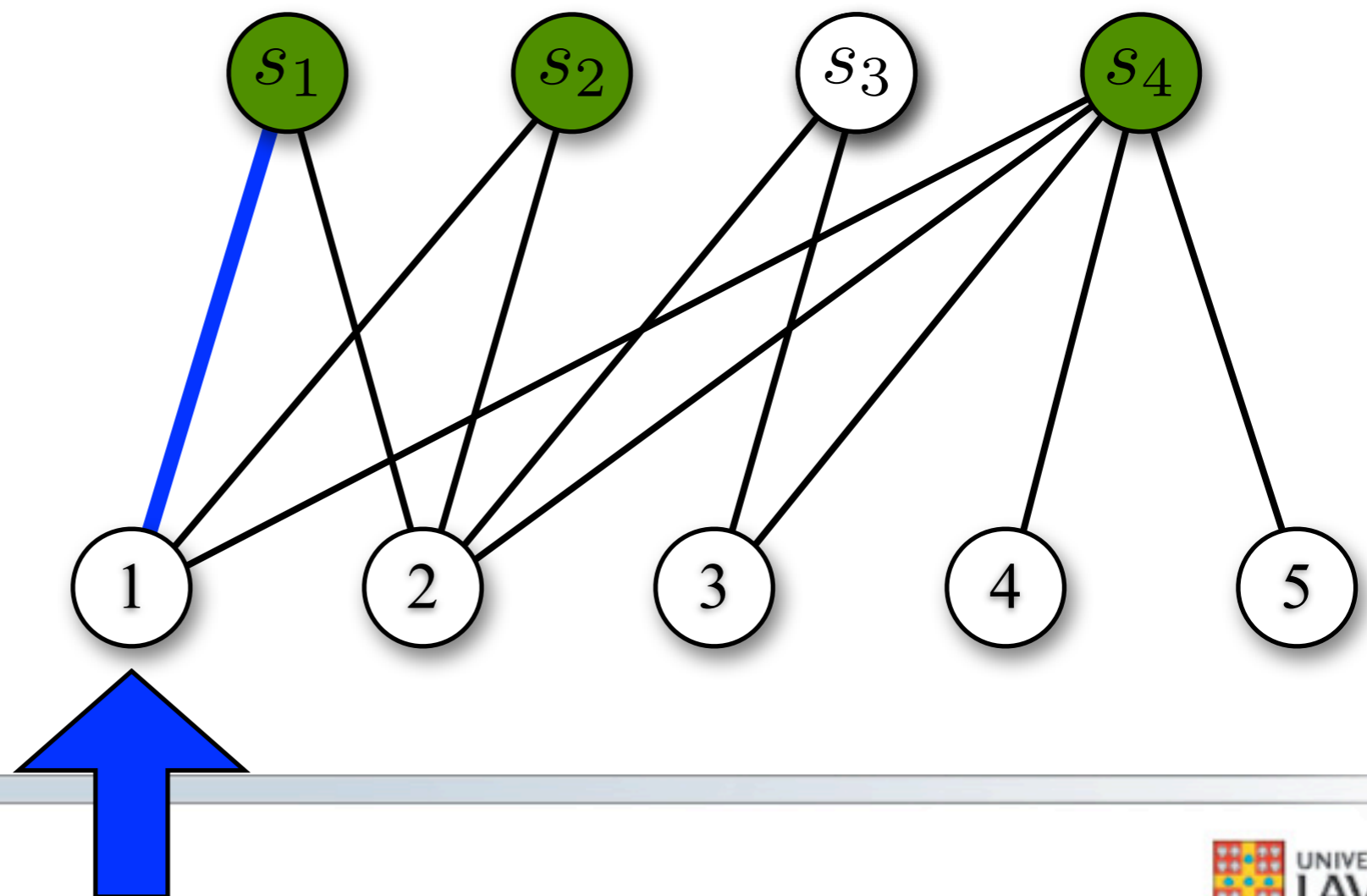


Un premier problème polynomial

- Couplage dans un graphe biparti convexe.

$$s_1 \in [1, 2], s_2 \in [1, 2]$$

$$s_3 \in [2, 3], s_4 \in [1, 5]$$

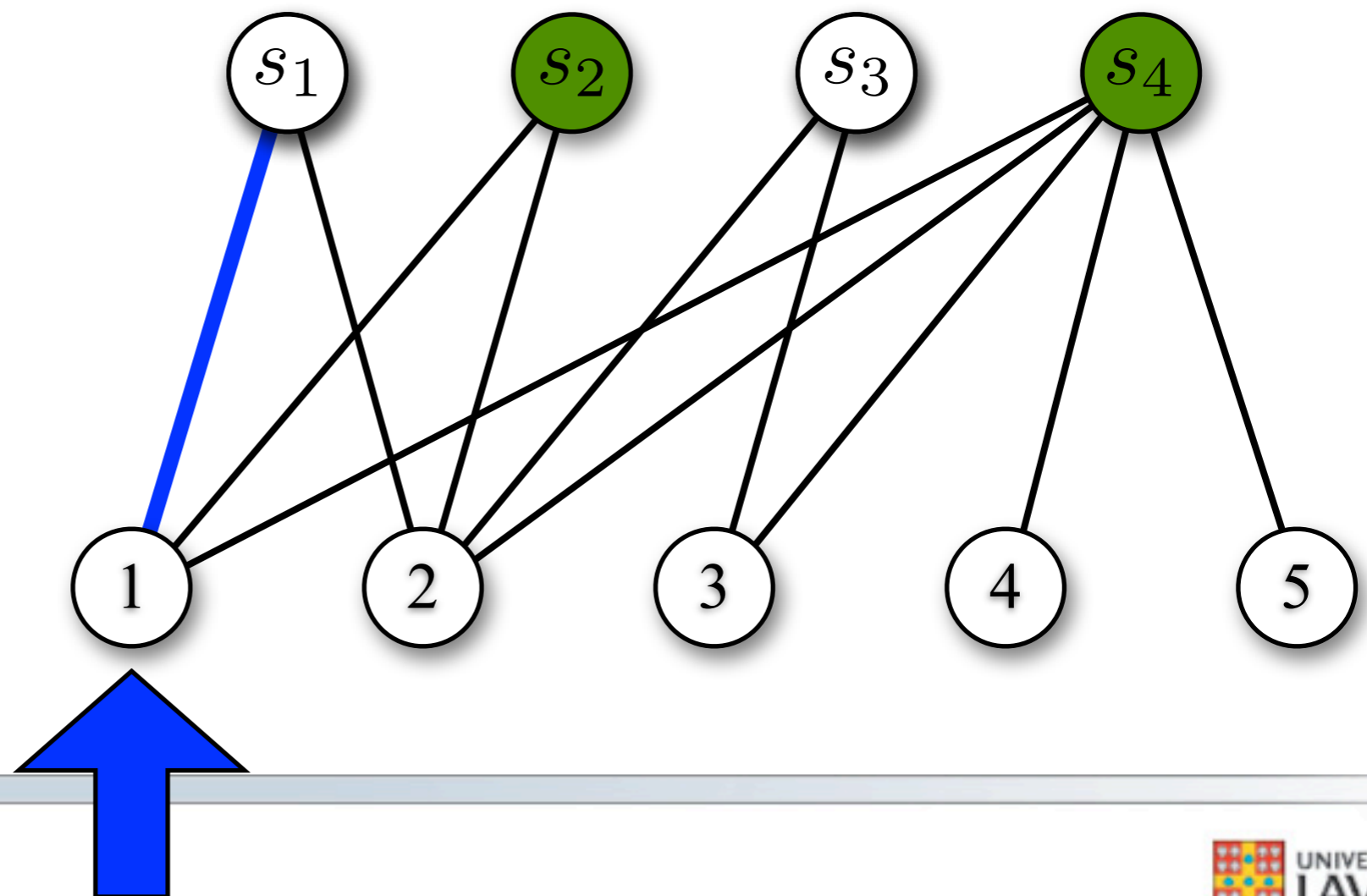


Un premier problème polynomial

- Couplage dans un graphe biparti convexe.

$$s_1 \in [1, 2], s_2 \in [1, 2]$$

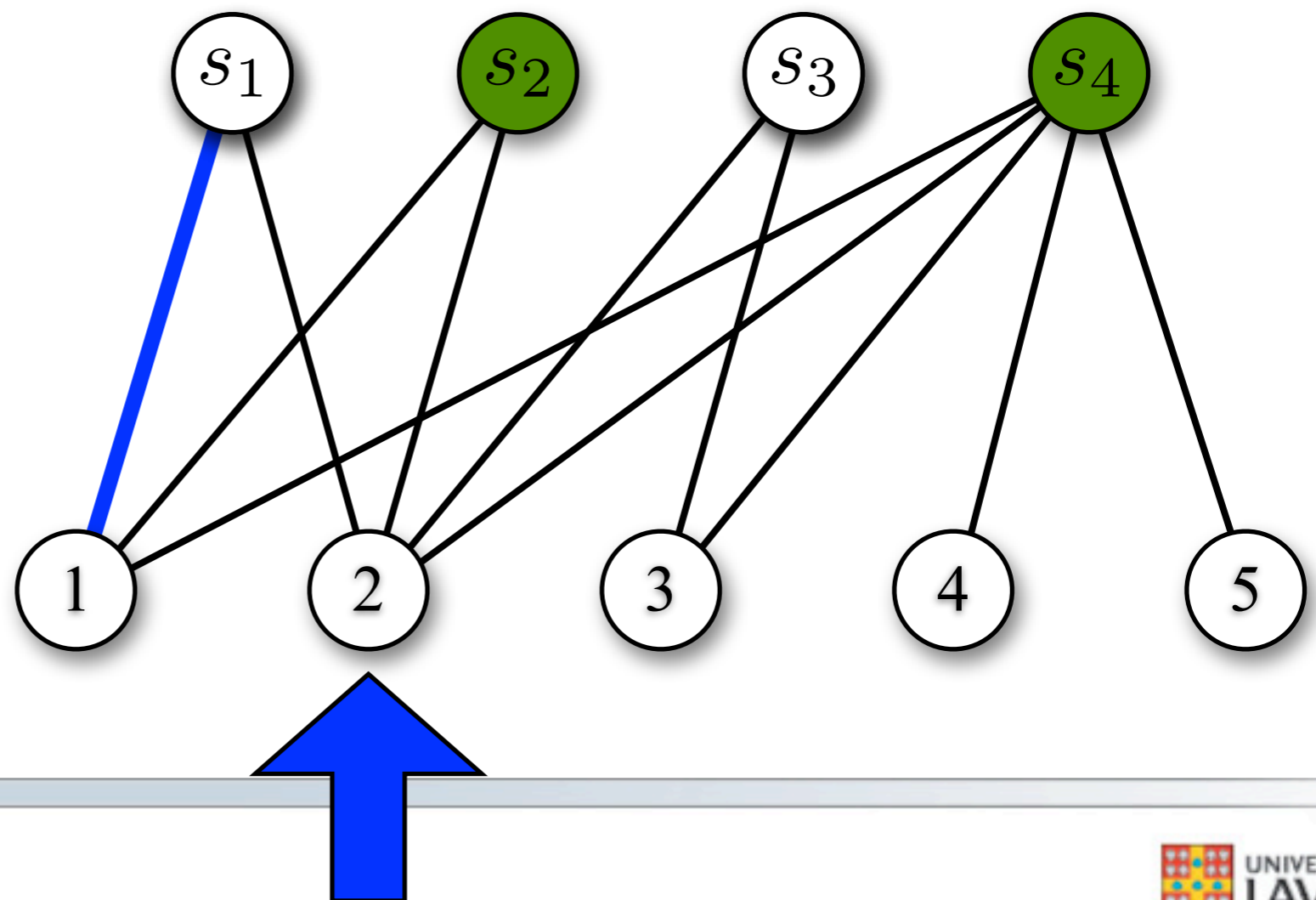
$$s_3 \in [2, 3], s_4 \in [1, 5]$$



Un premier problème polynomial

- Couplage dans un graphe biparti convexe.

$$s_1 \in [1, 2], s_2 \in [1, 2]$$
$$s_3 \in [2, 3], s_4 \in [1, 5]$$

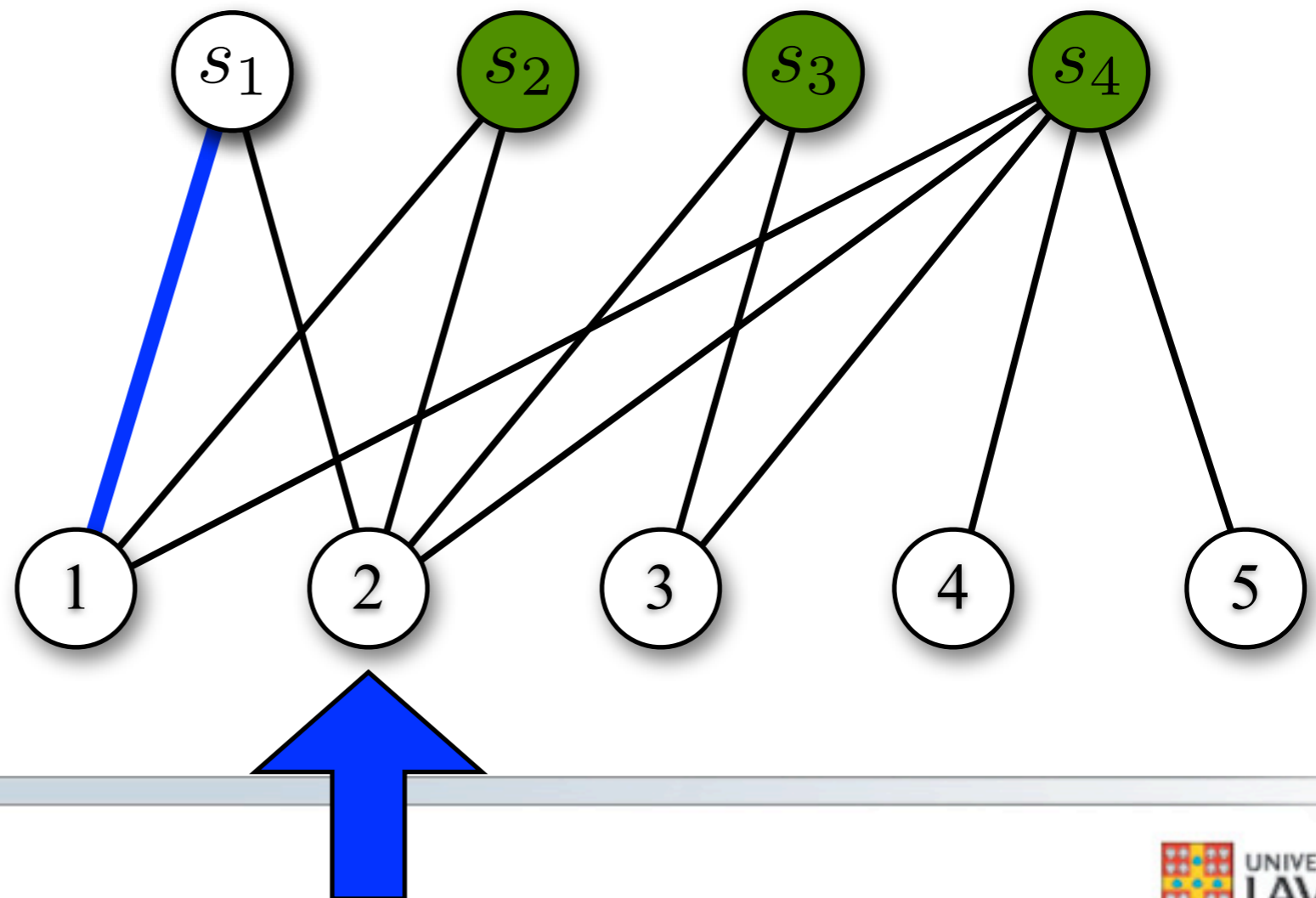


Un premier problème polynomial

- Couplage dans un graphe biparti convexe.

$$s_1 \in [1, 2], s_2 \in [1, 2]$$

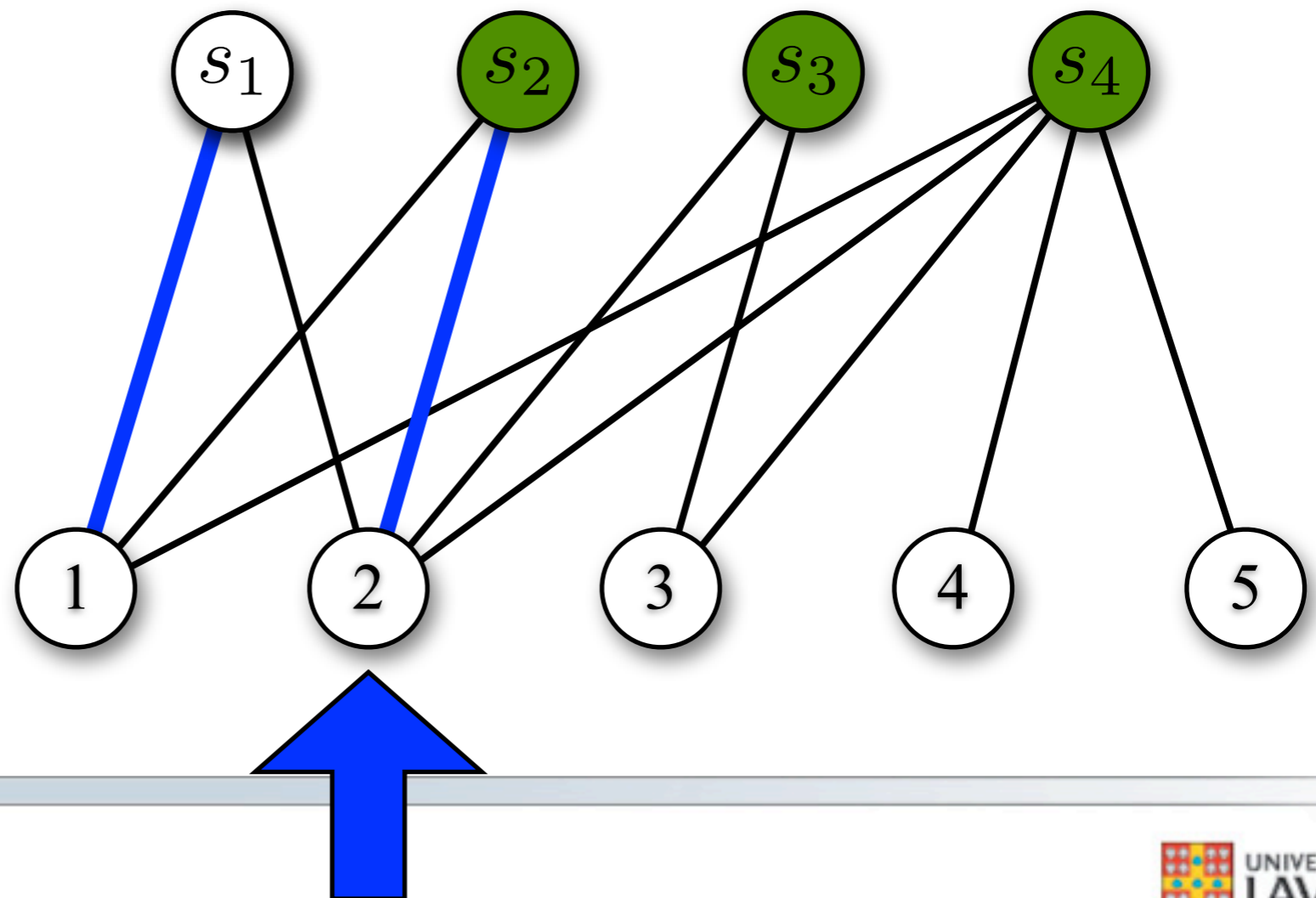
$$s_3 \in [2, 3], s_4 \in [1, 5]$$



Un premier problème polynomial

- Couplage dans un graphe biparti convexe.

$$s_1 \in [1, 2], s_2 \in [1, 2]$$
$$s_3 \in [2, 3], s_4 \in [1, 5]$$

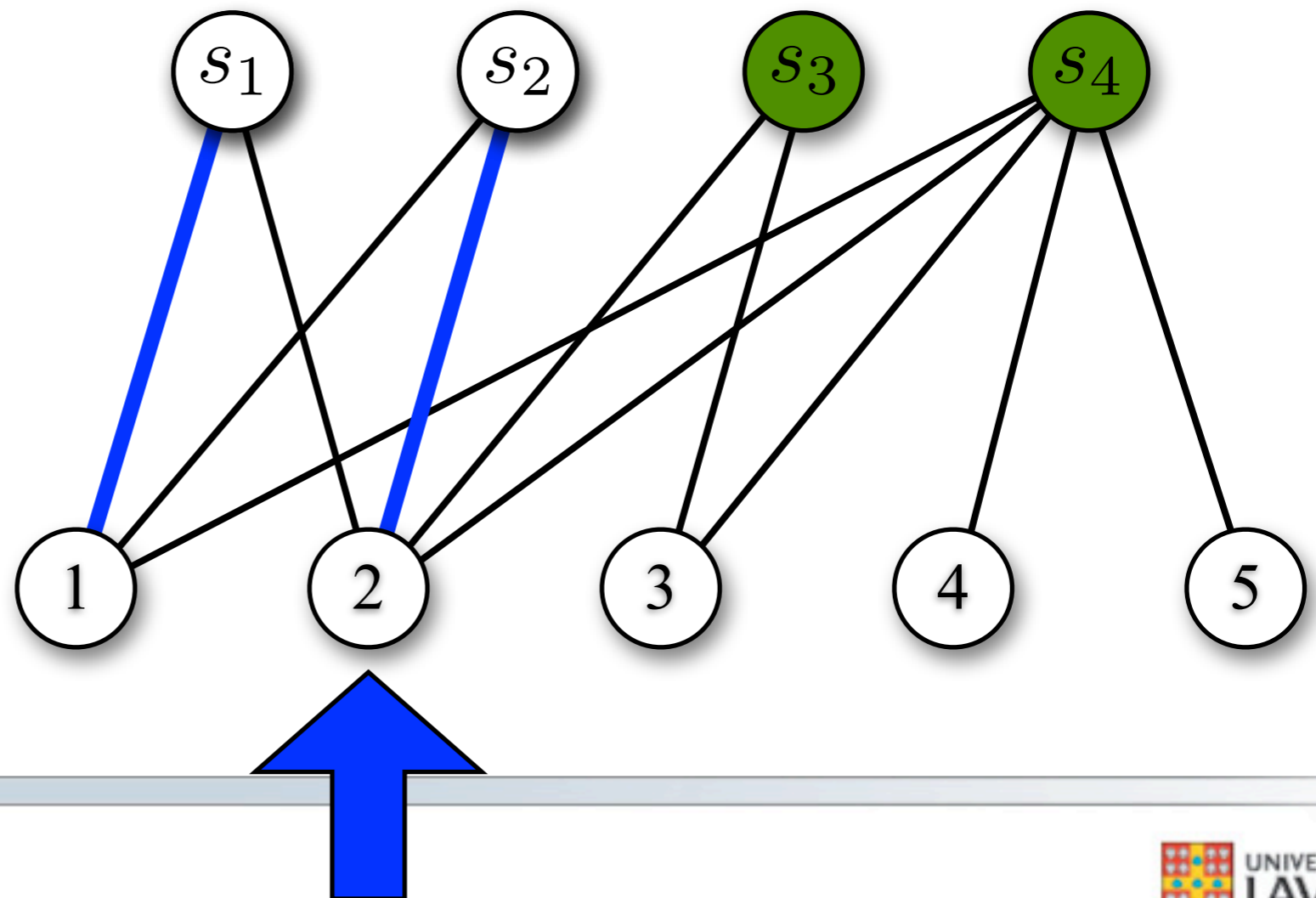


Un premier problème polynomial

- Couplage dans un graphe biparti convexe.

$$s_1 \in [1, 2], s_2 \in [1, 2]$$

$$s_3 \in [2, 3], s_4 \in [1, 5]$$

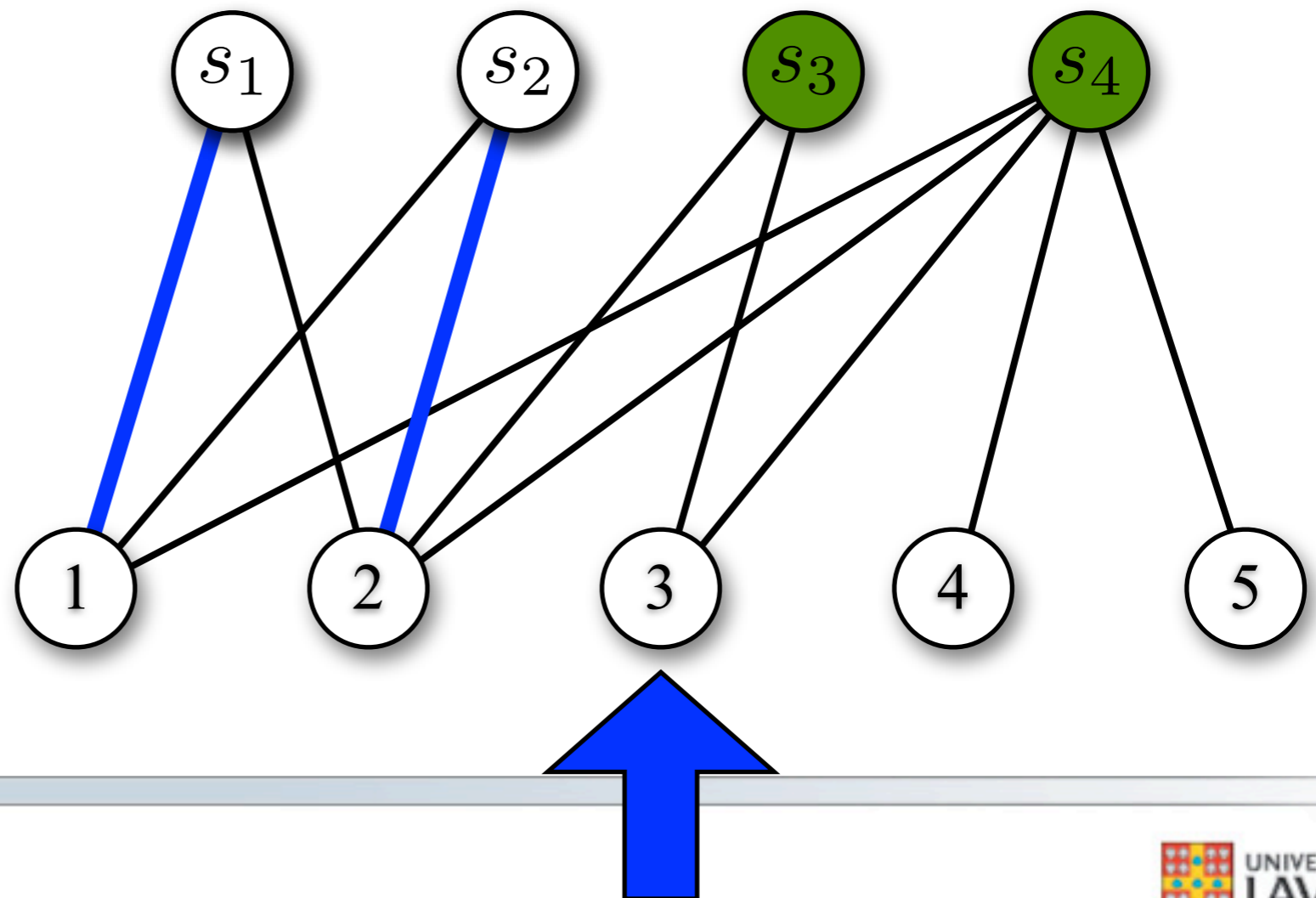


Un premier problème polynomial

- Couplage dans un graphe biparti convexe.

$$s_1 \in [1, 2], s_2 \in [1, 2]$$

$$s_3 \in [2, 3], s_4 \in [1, 5]$$

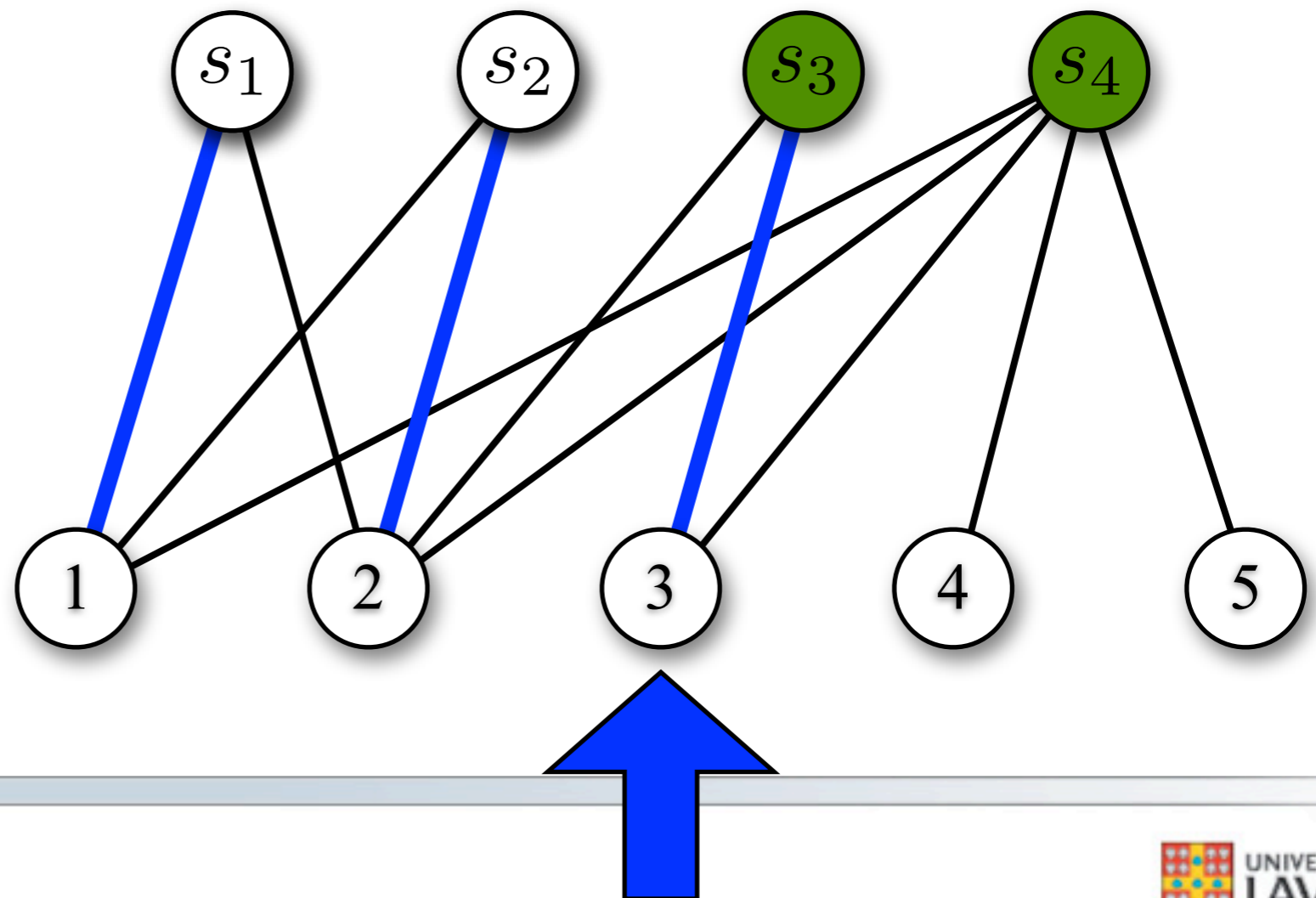


Un premier problème polynomial

- Couplage dans un graphe biparti convexe.

$$s_1 \in [1, 2], s_2 \in [1, 2]$$

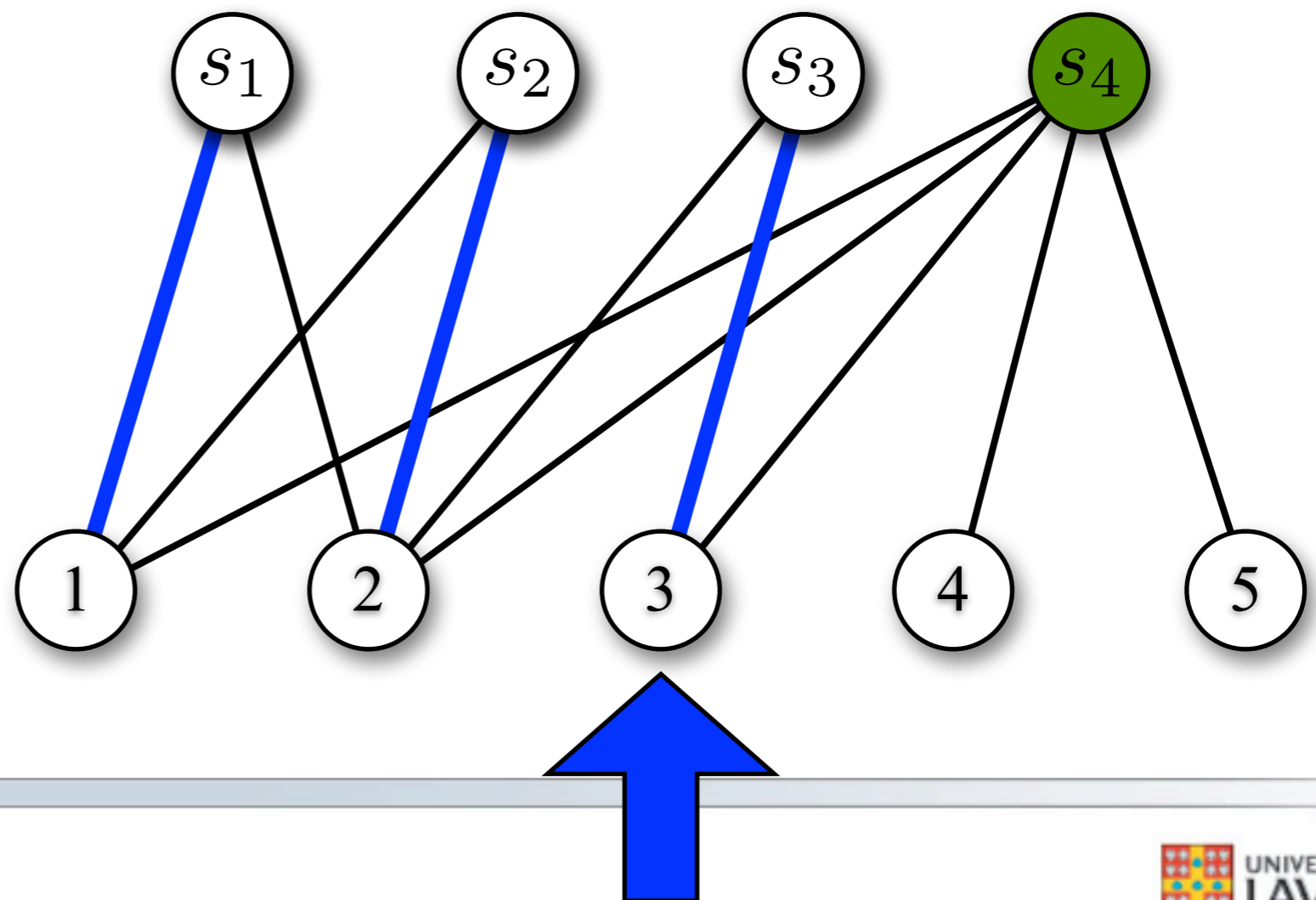
$$s_3 \in [2, 3], s_4 \in [1, 5]$$



Un premier problème polynomial

- Couplage dans un graphe biparti convexe.

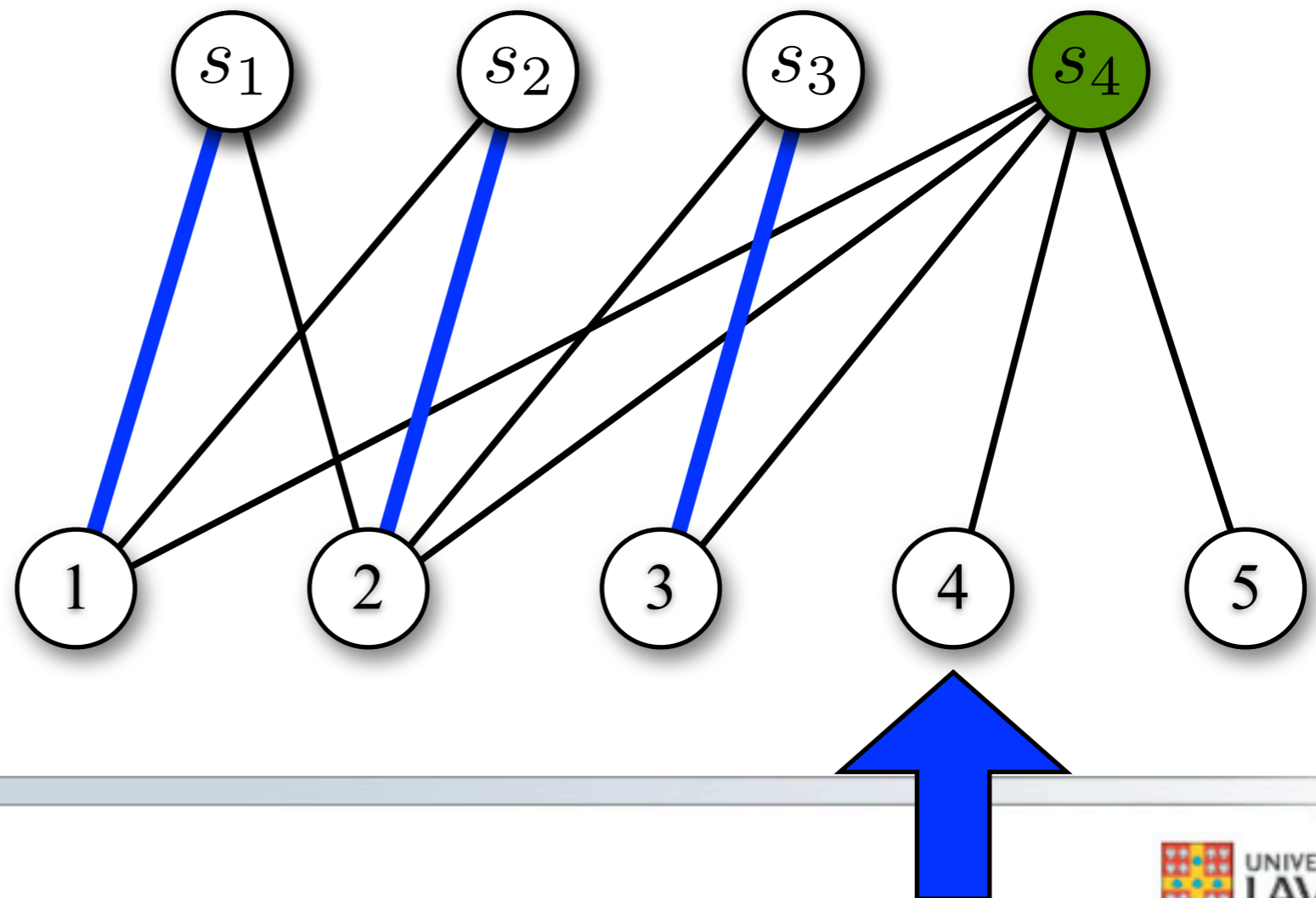
$$s_1 \in [1, 2], s_2 \in [1, 2]$$
$$s_3 \in [2, 3], s_4 \in [1, 5]$$



Un premier problème polynomial

- Couplage dans un graphe biparti convexe.

$$s_1 \in [1, 2], s_2 \in [1, 2]$$
$$s_3 \in [2, 3], s_4 \in [1, 5]$$

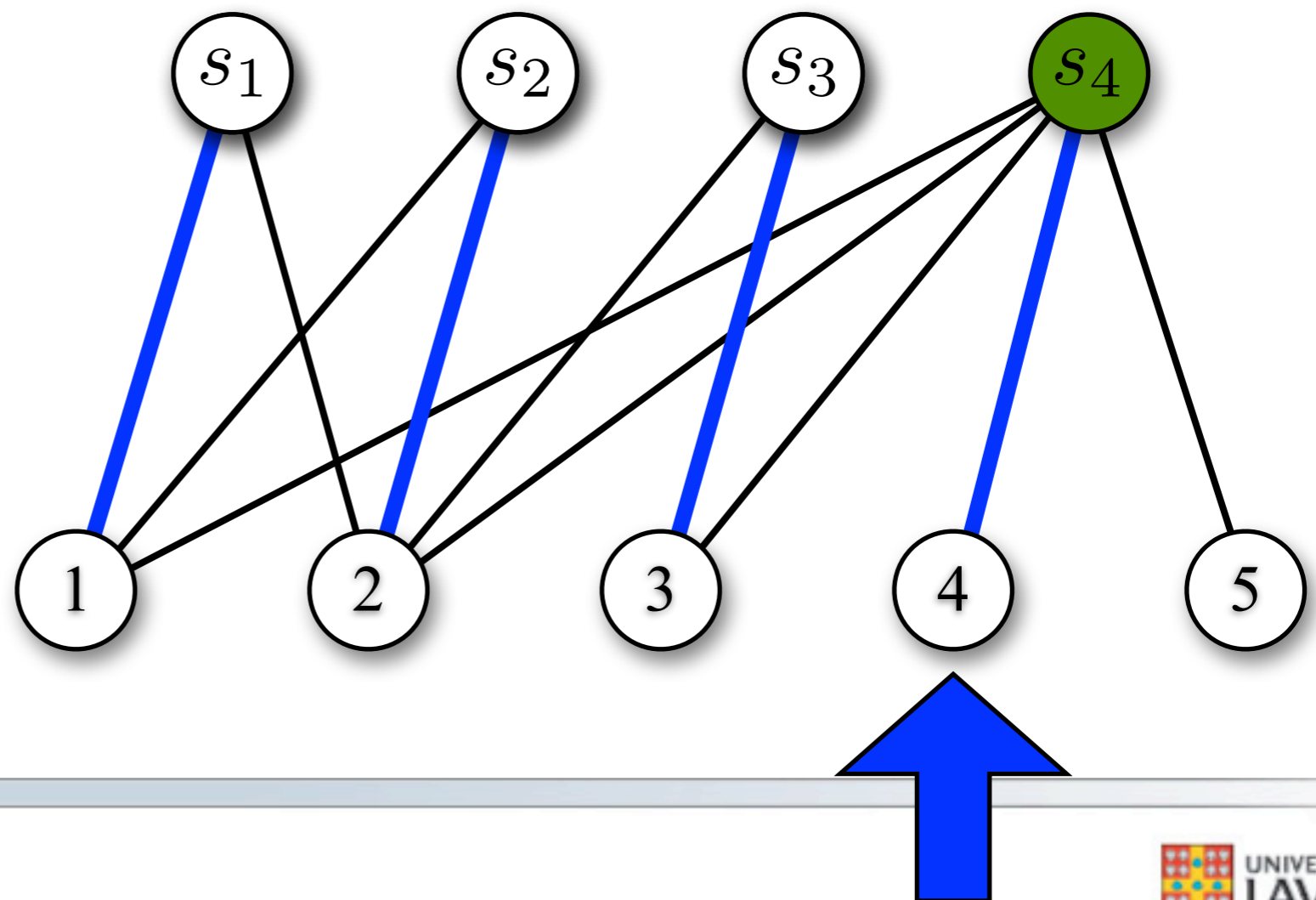


Un premier problème polynomial

- Couplage dans un graphe biparti convexe.

$$s_1 \in [1, 2], s_2 \in [1, 2]$$

$$s_3 \in [2, 3], s_4 \in [1, 5]$$

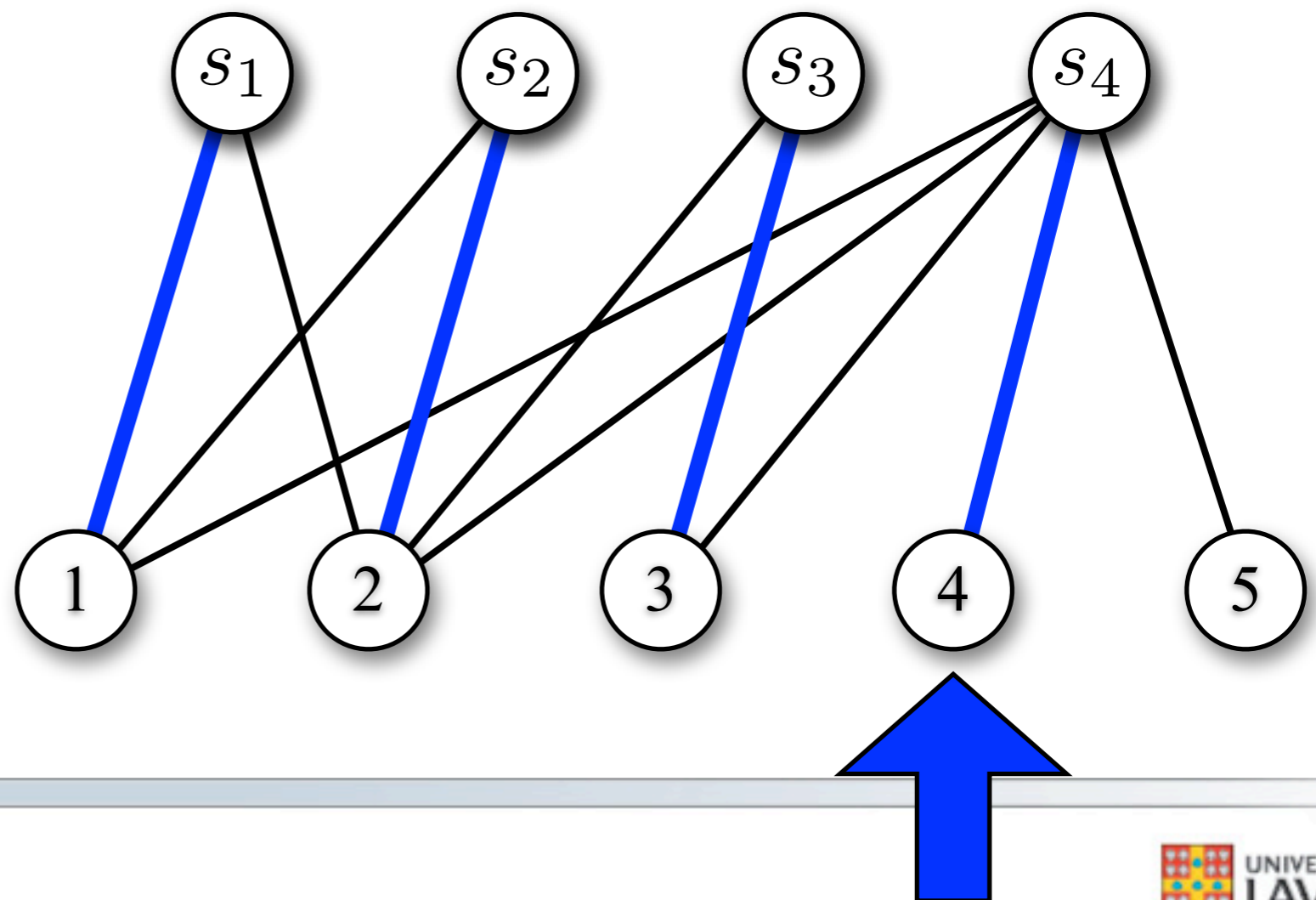


Un premier problème polynomial

- Couplage dans un graphe biparti convexe.

$$s_1 \in [1, 2], s_2 \in [1, 2]$$

$$s_3 \in [2, 3], s_4 \in [1, 5]$$

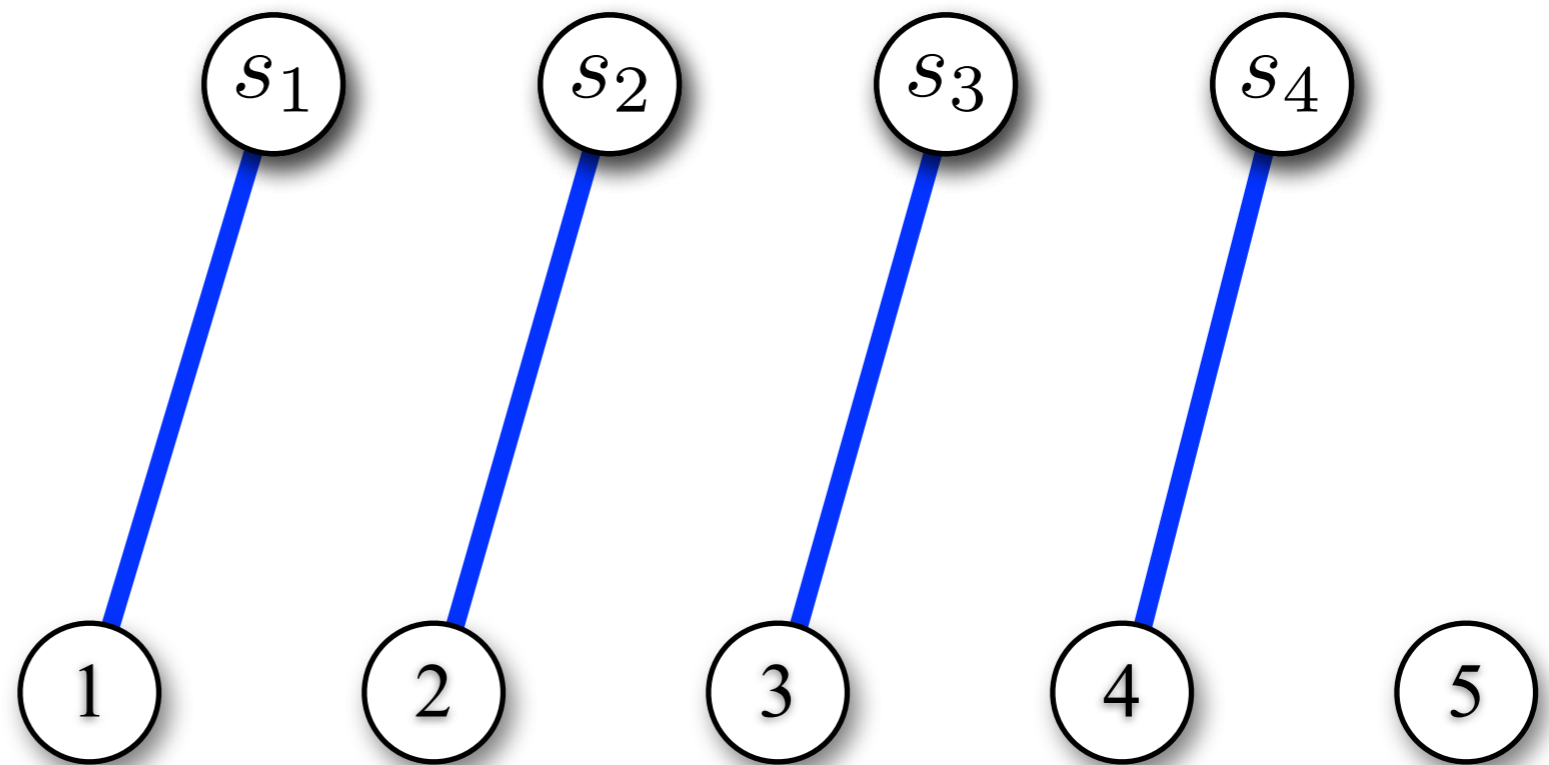


Un premier problème polynomial

- Couplage dans un graphe biparti convexe.

$$s_1 \in [1, 2], s_2 \in [1, 2]$$

$$s_3 \in [2, 3], s_4 \in [1, 5]$$



Couplage dans un graphe biparti convexe

Auteur	Année	Complexité
Glover	1967	$O(n \log n)$

Couplage dans un graphe biparti convexe

Auteur	Année	Complexité
Glover	1967	$O(n \log n)$
Lipski & Preparata	1981	$O(n\alpha(n))$

Couplage dans un graphe biparti convexe

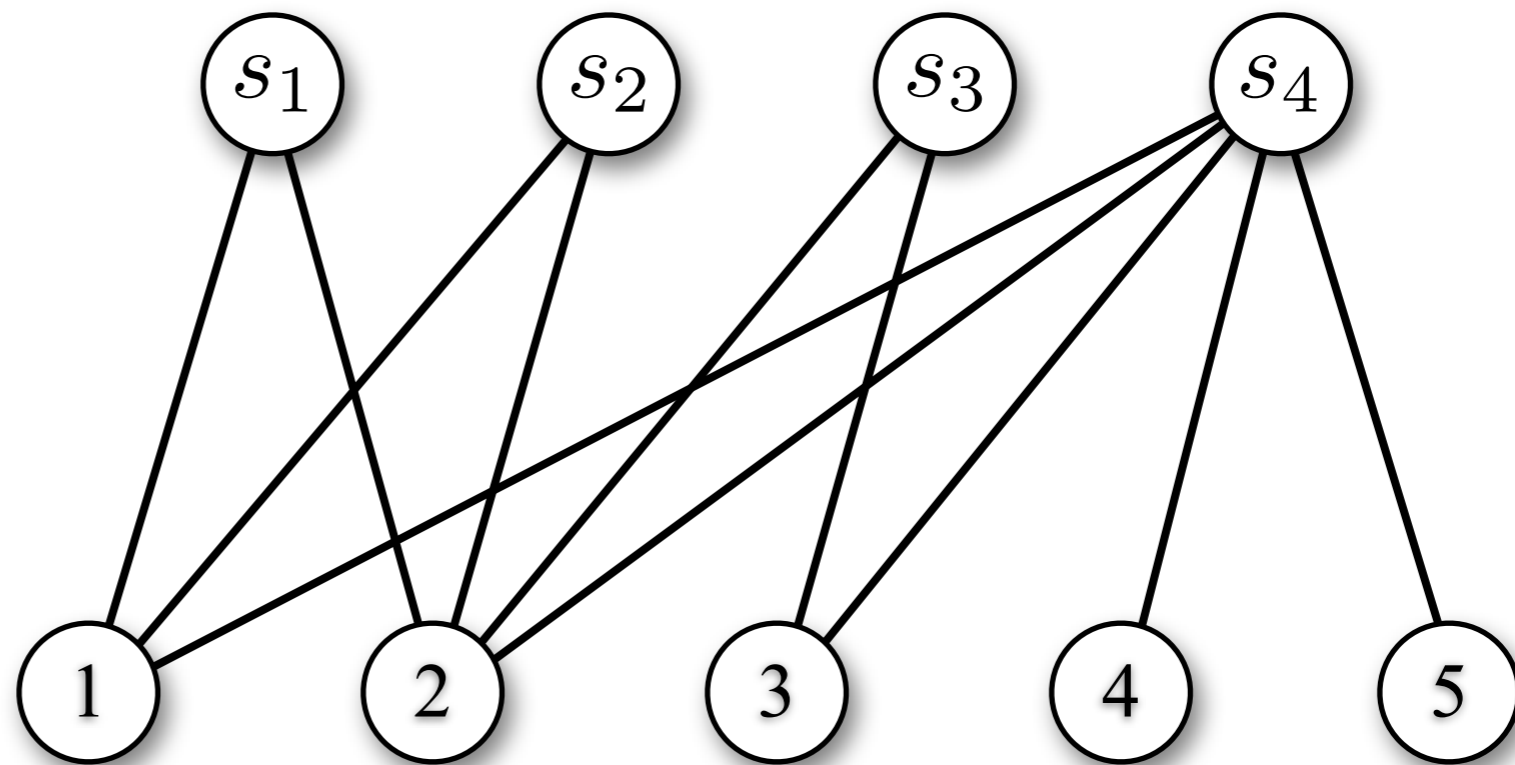
Auteur	Année	Complexité
Glover	1967	$O(n \log n)$
Lipski & Preparata	1981	$O(n\alpha(n))$
Gabow & Tarjan	1983	$O(n)$

Avec plus d'une ressource

- m ressources
- Une tâche doit commencer au plus tôt au temps a et au plus tard au temps b .
- Des temps de traitement unitaires

$$s_1 \in [1, 2], s_2 \in [1, 2]$$

$$s_3 \in [2, 3], s_4 \in [1, 5]$$

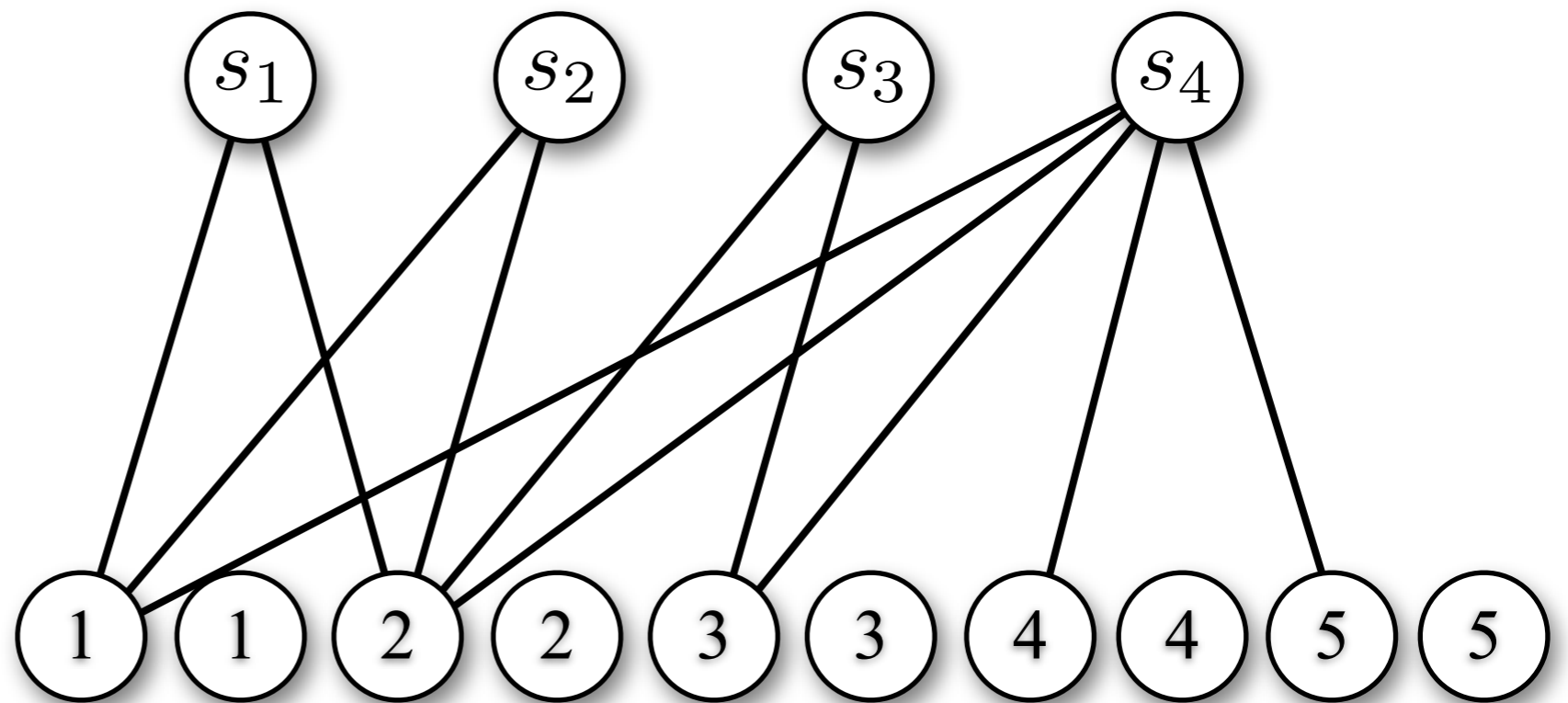


Avec plus d'une ressource

- m ressources
- Une tâche doit commencer au plus tôt au temps a et au plus tard au temps b .
- Des temps de traitement unitaires

$$s_1 \in [1, 2], s_2 \in [1, 2]$$

$$s_3 \in [2, 3], s_4 \in [1, 5]$$

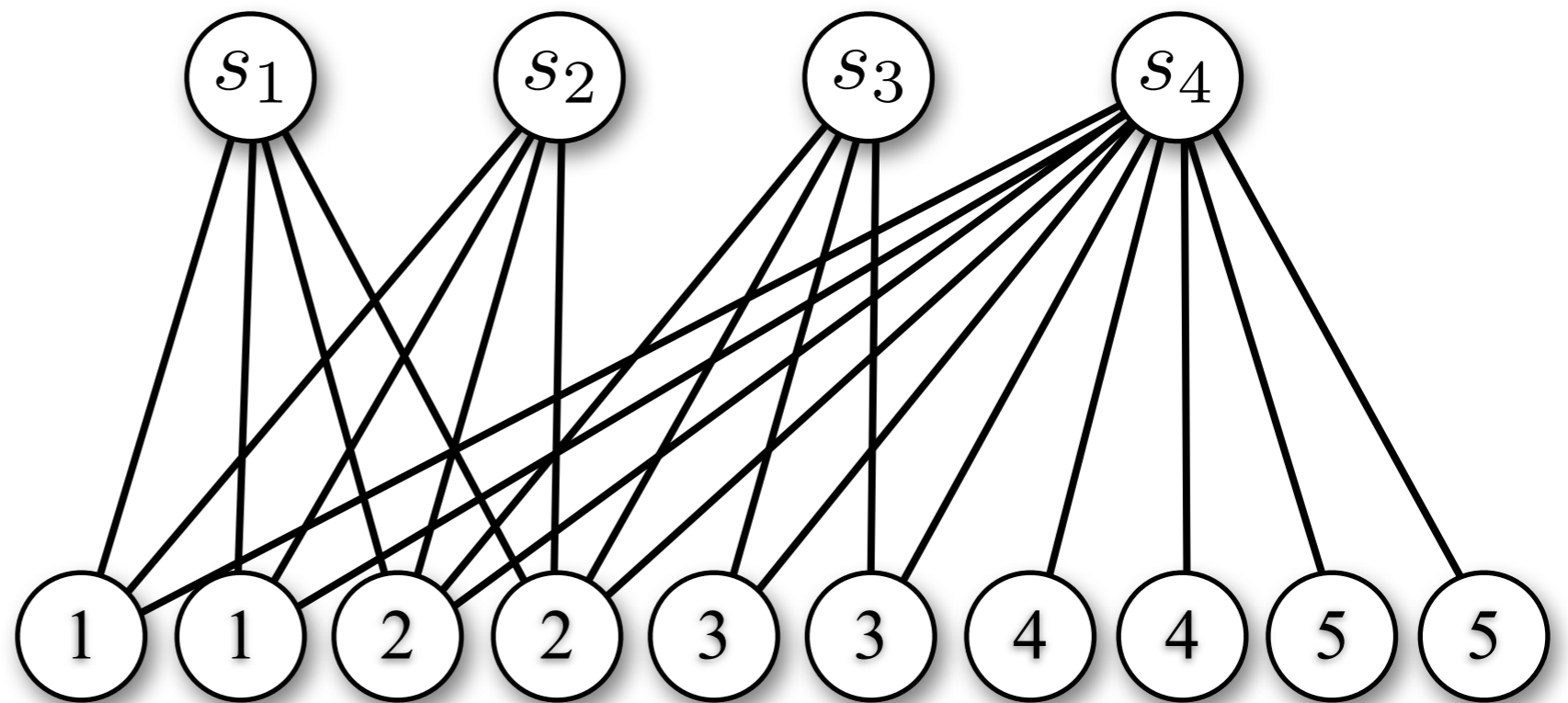


Avec plus d'une ressource

- m ressources
- Une tâche doit commencer au plus tôt au temps a et au plus tard au temps b .
- Des temps de traitement unitaires

$$s_1 \in [1, 2], s_2 \in [1, 2]$$

$$s_3 \in [2, 3], s_4 \in [1, 5]$$



Couplage dans un graphe biparti convexe

Auteur	Année	Complexité
Glover	1967	$O(n \log n)$
Lipski & Preparata	1981	$O(n\alpha(n))$
Gabow & Tarjan	1983	$O(n)$

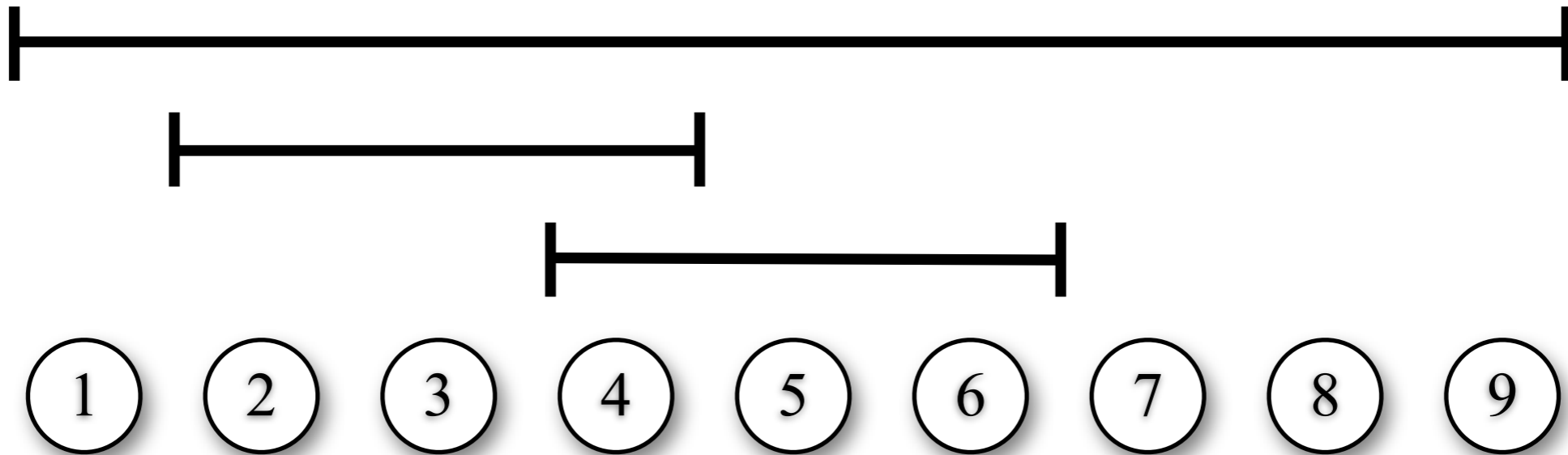
Temps de traitement identiques

- Une seule ressource.
- Une tâche doit commencer au plus tôt au temps a et au plus tard au temps b .
- Des temps de traitement identiques.
- Le problème ne peut plus être exprimé par un couplage dans un graphe biparti.

Temps de traitement identiques

Garey, Johnson, Simons, and Tarjan 1981

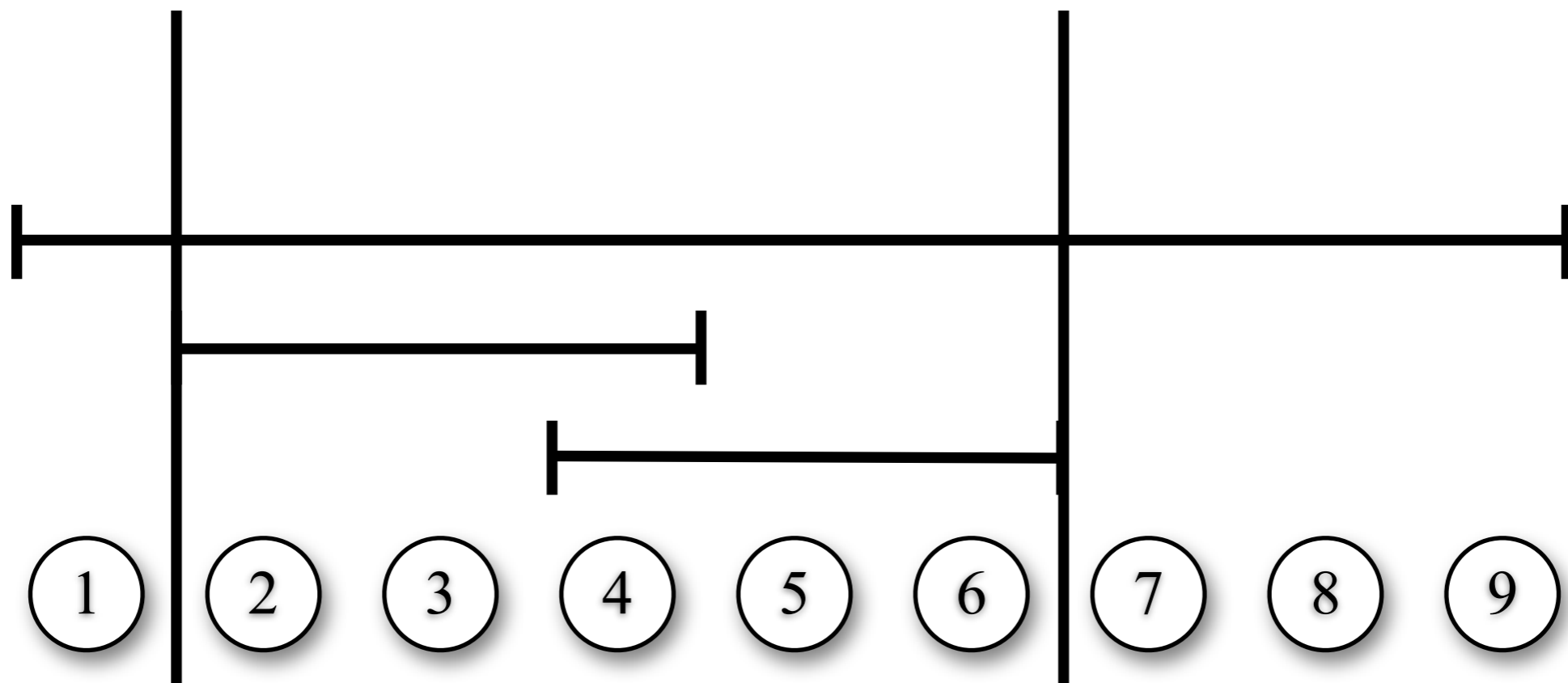
$$s_1 \in [1, 9], s_2 \in [2, 4], s_3 \in [4, 6], p = 3$$



Temps de traitement identiques

Garey, Johnson, Simons, and Tarjan 1981

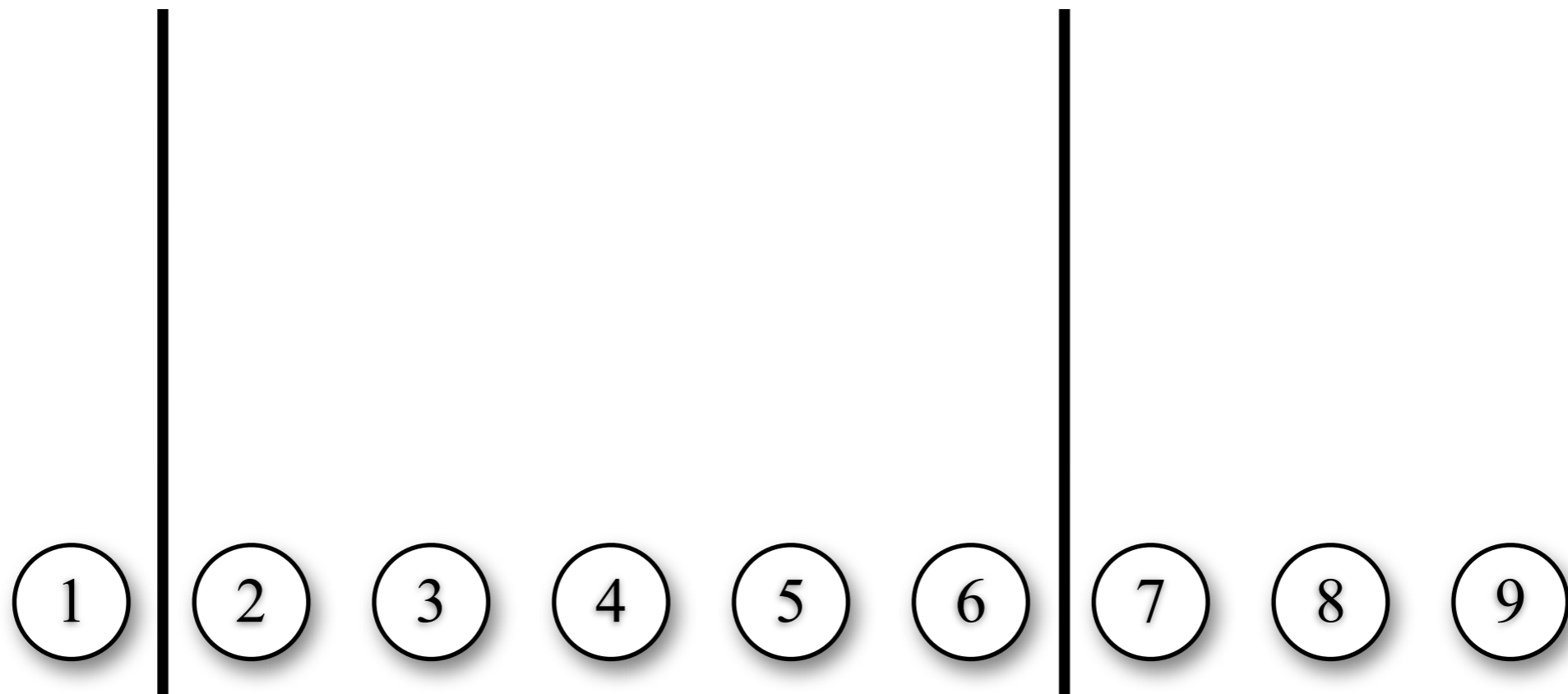
$$s_1 \in [1, 9], s_2 \in [2, 4], s_3 \in [4, 6], p = 3$$



Temps de traitement identiques

Garey, Johnson, Simons, and Tarjan 1981

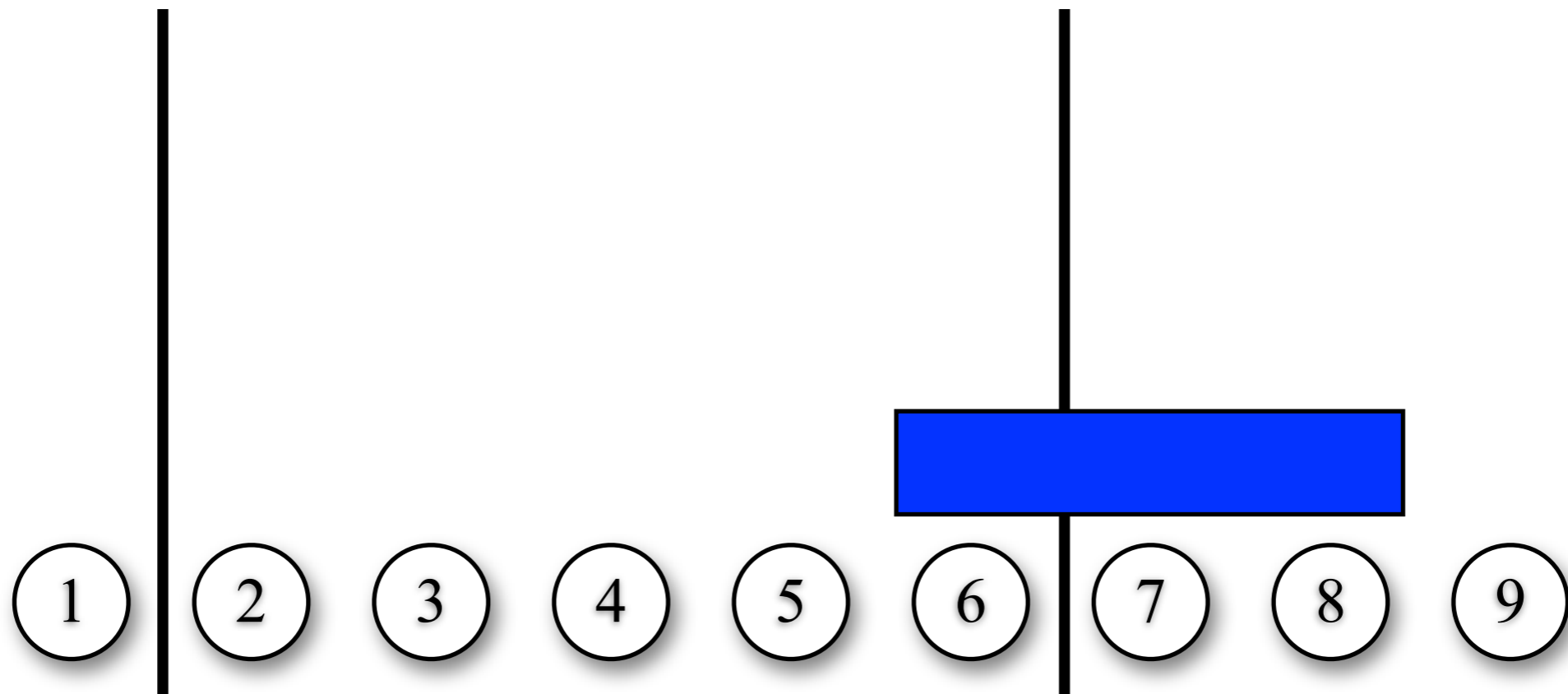
$$s_1 \in [1, 9], s_2 \in [2, 4], s_3 \in [4, 6], p = 3$$



Temps de traitement identiques

Garey, Johnson, Simons, and Tarjan 1981

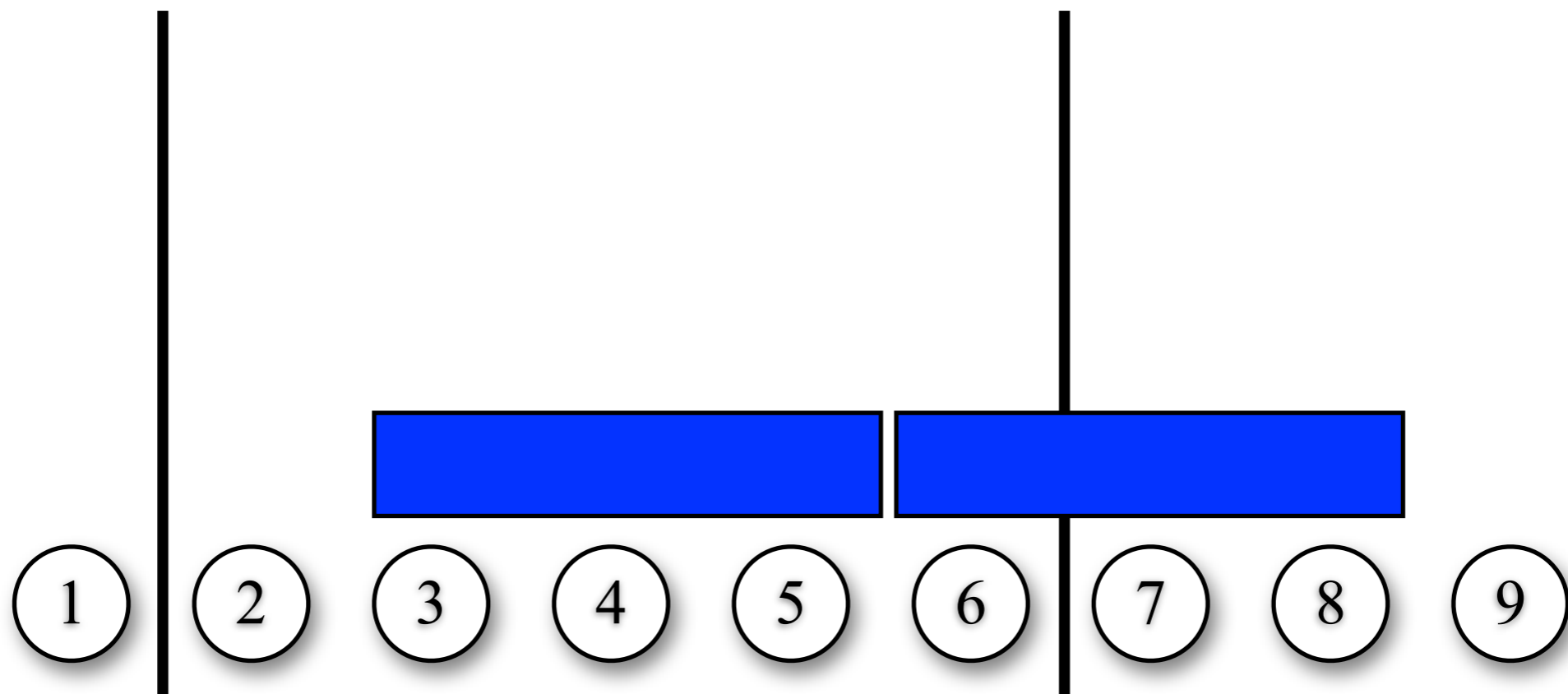
$$s_1 \in [1, 9], s_2 \in [2, 4], s_3 \in [4, 6], p = 3$$



Temps de traitement identiques

Garey, Johnson, Simons, and Tarjan 1981

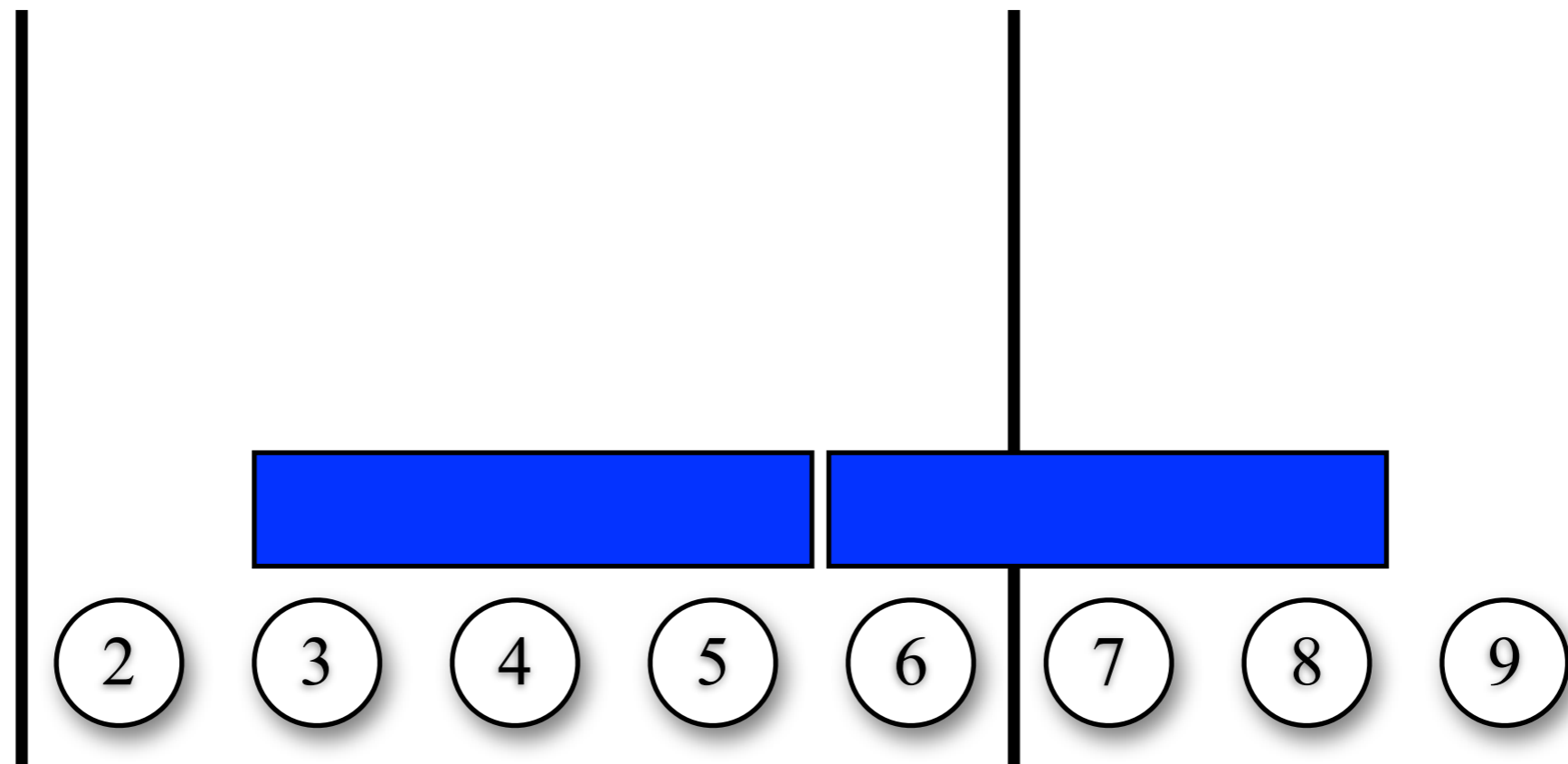
$$s_1 \in [1, 9], s_2 \in [2, 4], s_3 \in [4, 6], p = 3$$



Temps de traitement identiques

Garey, Johnson, Simons, and Tarjan 1981

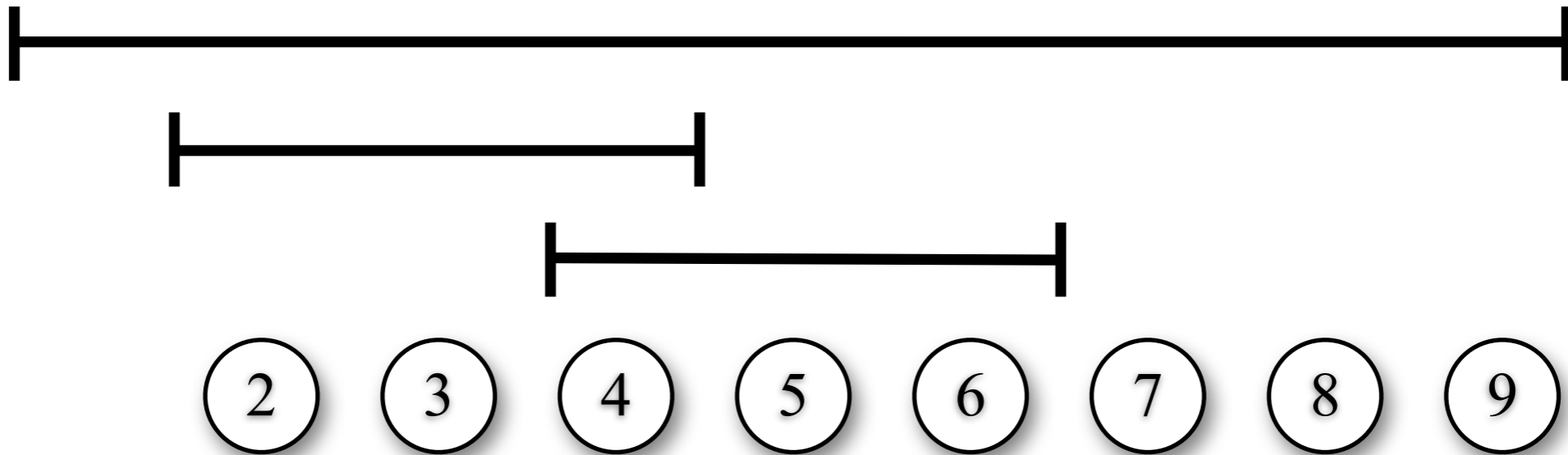
$$s_1 \in [1, 9], s_2 \in [2, 4], s_3 \in [4, 6], p = 3$$



Temps de traitement identiques

Garey, Johnson, Simons, and Tarjan 1981

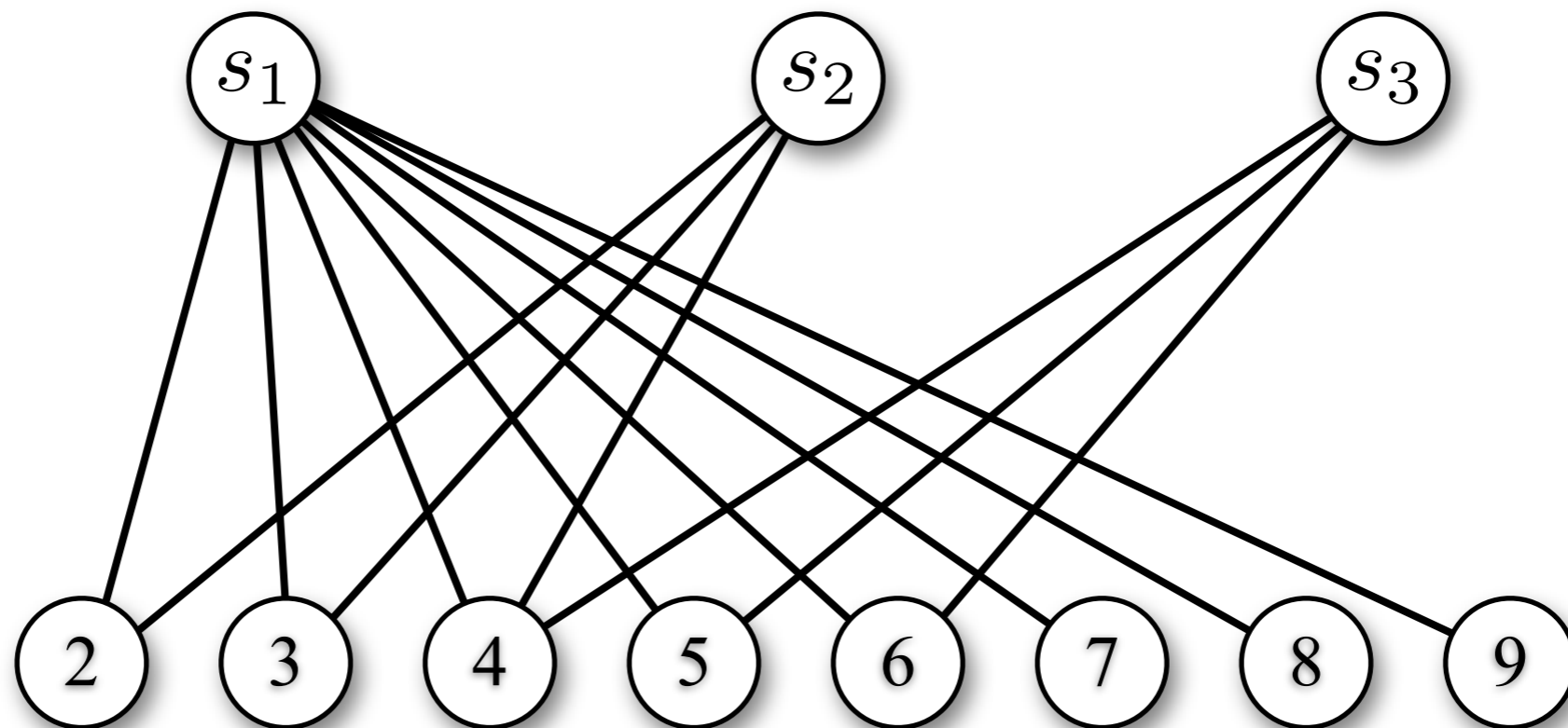
$$s_1 \in [1, 9], s_2 \in [2, 4], s_3 \in [4, 6], p = 3$$



Temps de traitement identiques

Garey, Johnson, Simons, and Tarjan 1981

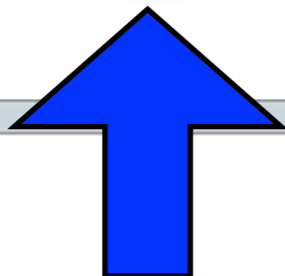
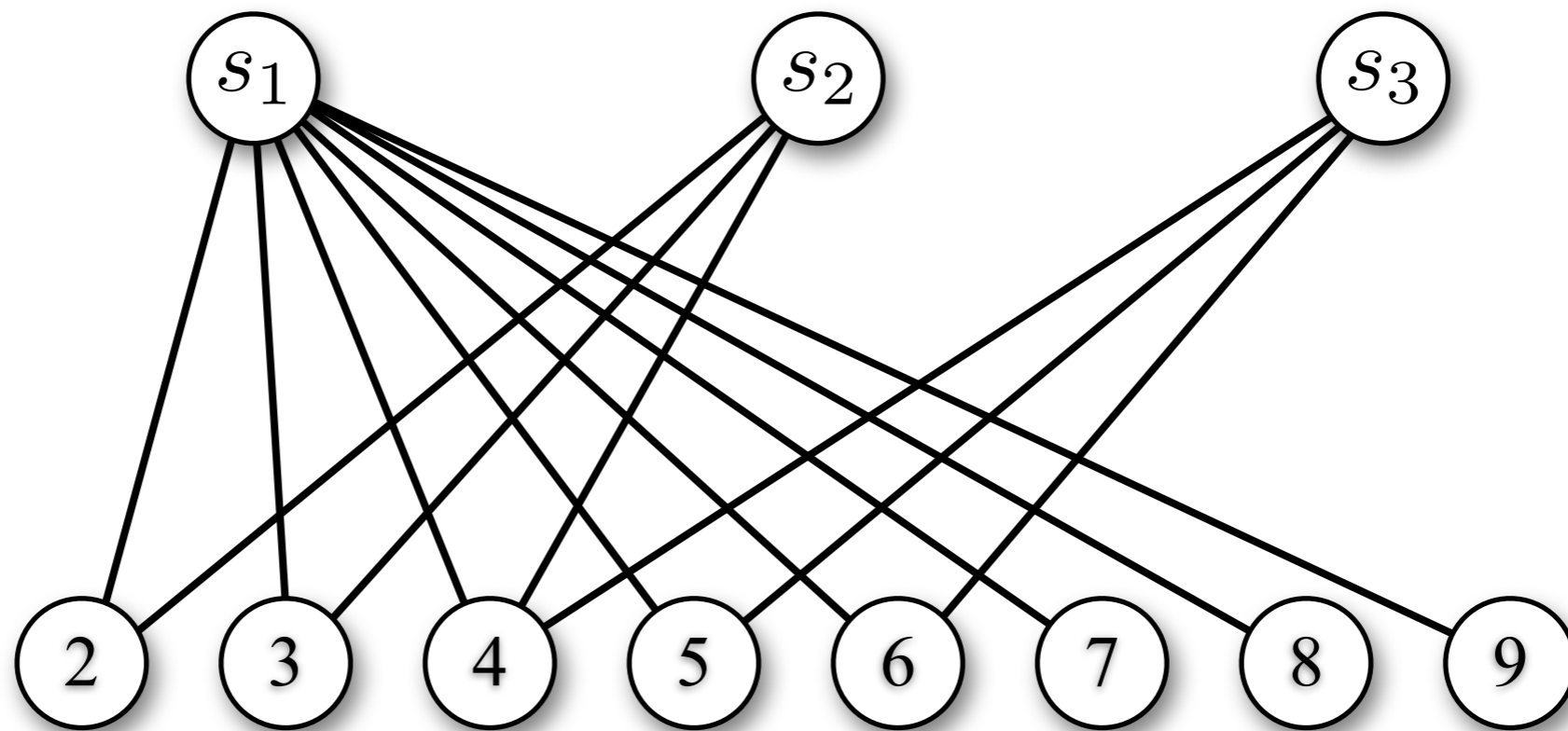
$$s_1 \in [1, 9], s_2 \in [2, 4], s_3 \in [4, 6], p = 3$$



Temps de traitement identiques

Garey, Johnson, Simons, and Tarjan 1981

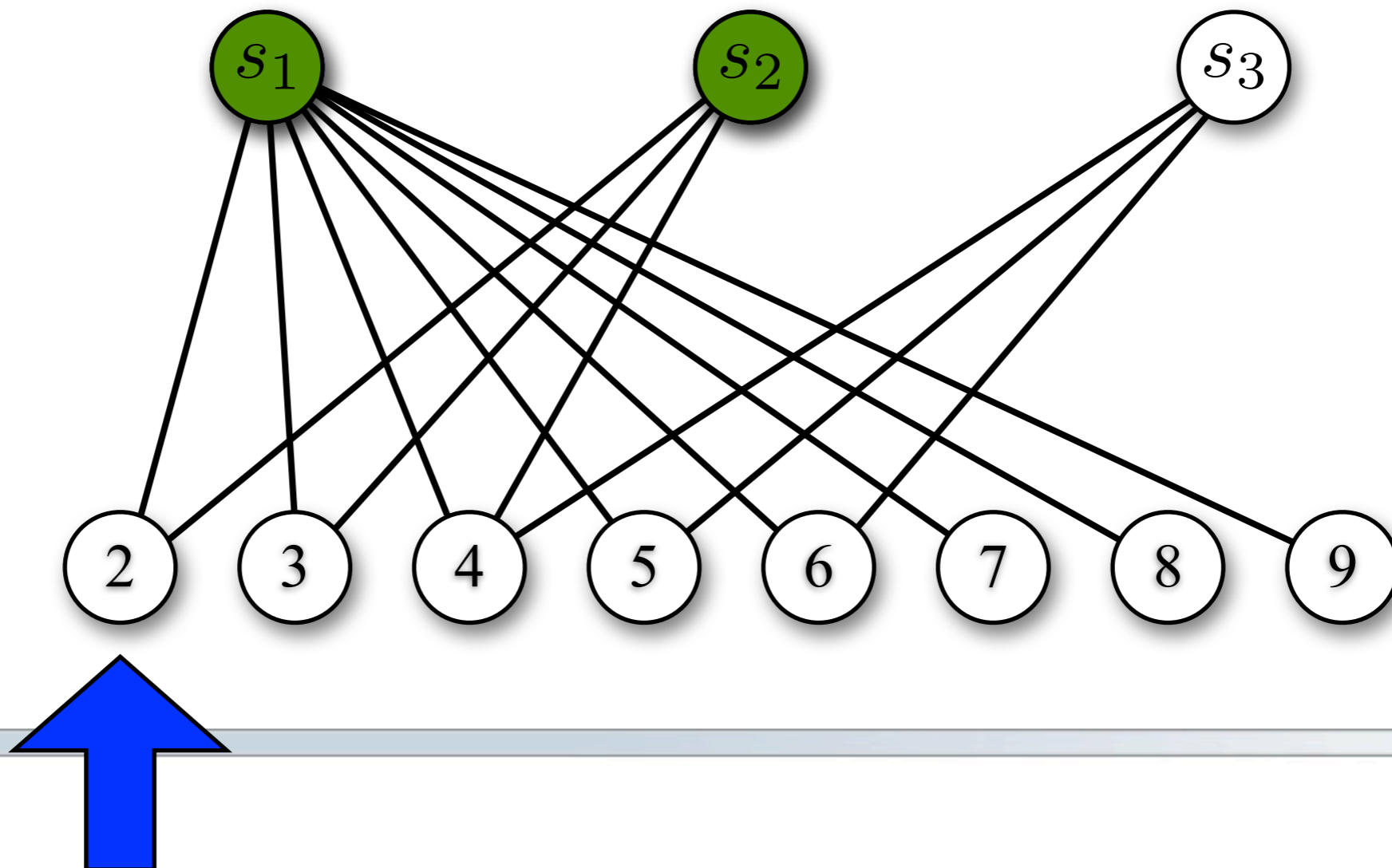
$$s_1 \in [1, 9], s_2 \in [2, 4], s_3 \in [4, 6], p = 3$$



Temps de traitement identiques

Garey, Johnson, Simons, and Tarjan 1981

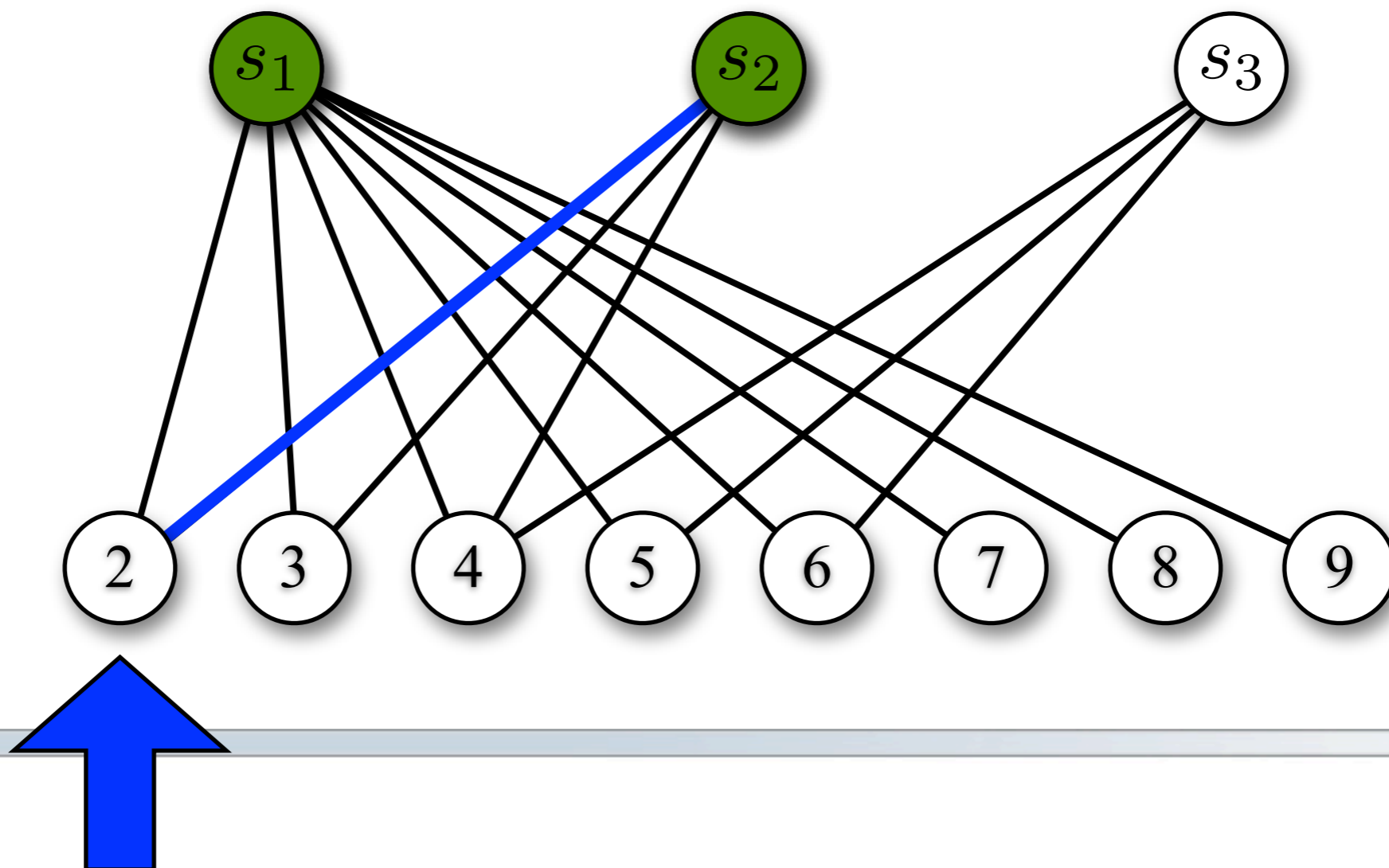
$$s_1 \in [1, 9], s_2 \in [2, 4], s_3 \in [4, 6], p = 3$$



Temps de traitement identiques

Garey, Johnson, Simons, and Tarjan 1981

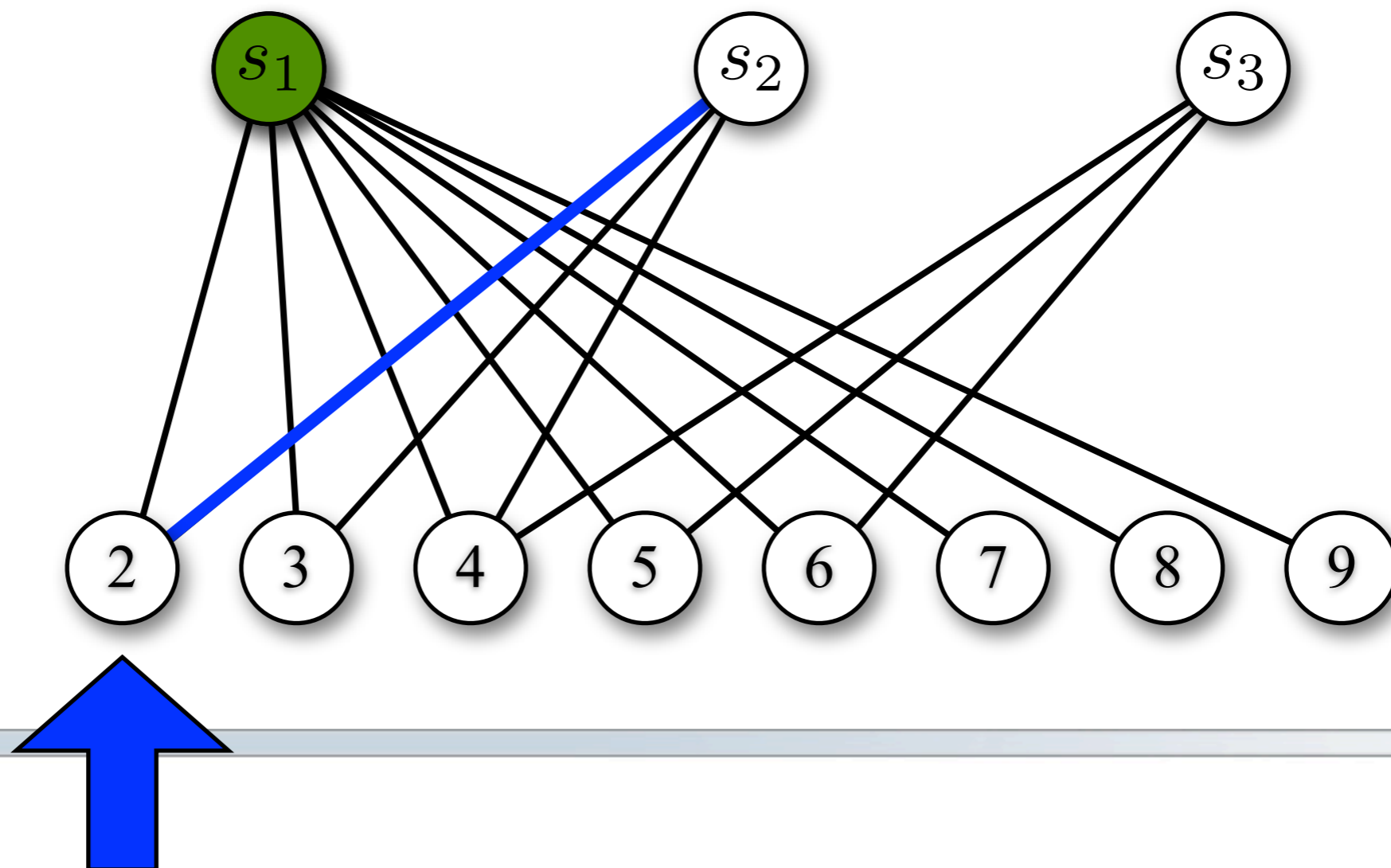
$$s_1 \in [1, 9], s_2 \in [2, 4], s_3 \in [4, 6], p = 3$$



Temps de traitement identiques

Garey, Johnson, Simons, and Tarjan 1981

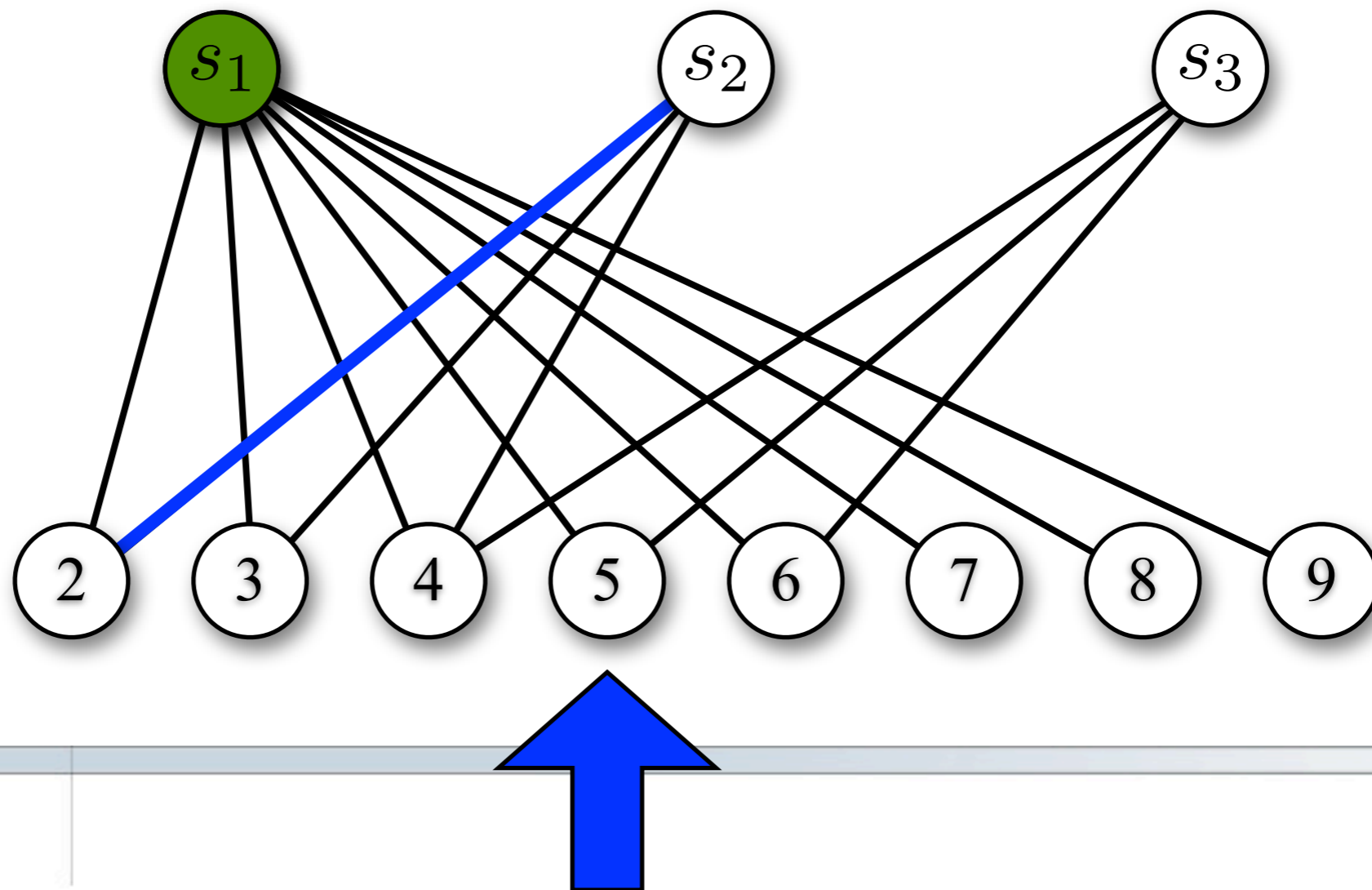
$$s_1 \in [1, 9], s_2 \in [2, 4], s_3 \in [4, 6], p = 3$$



Temps de traitement identiques

Garey, Johnson, Simons, and Tarjan 1981

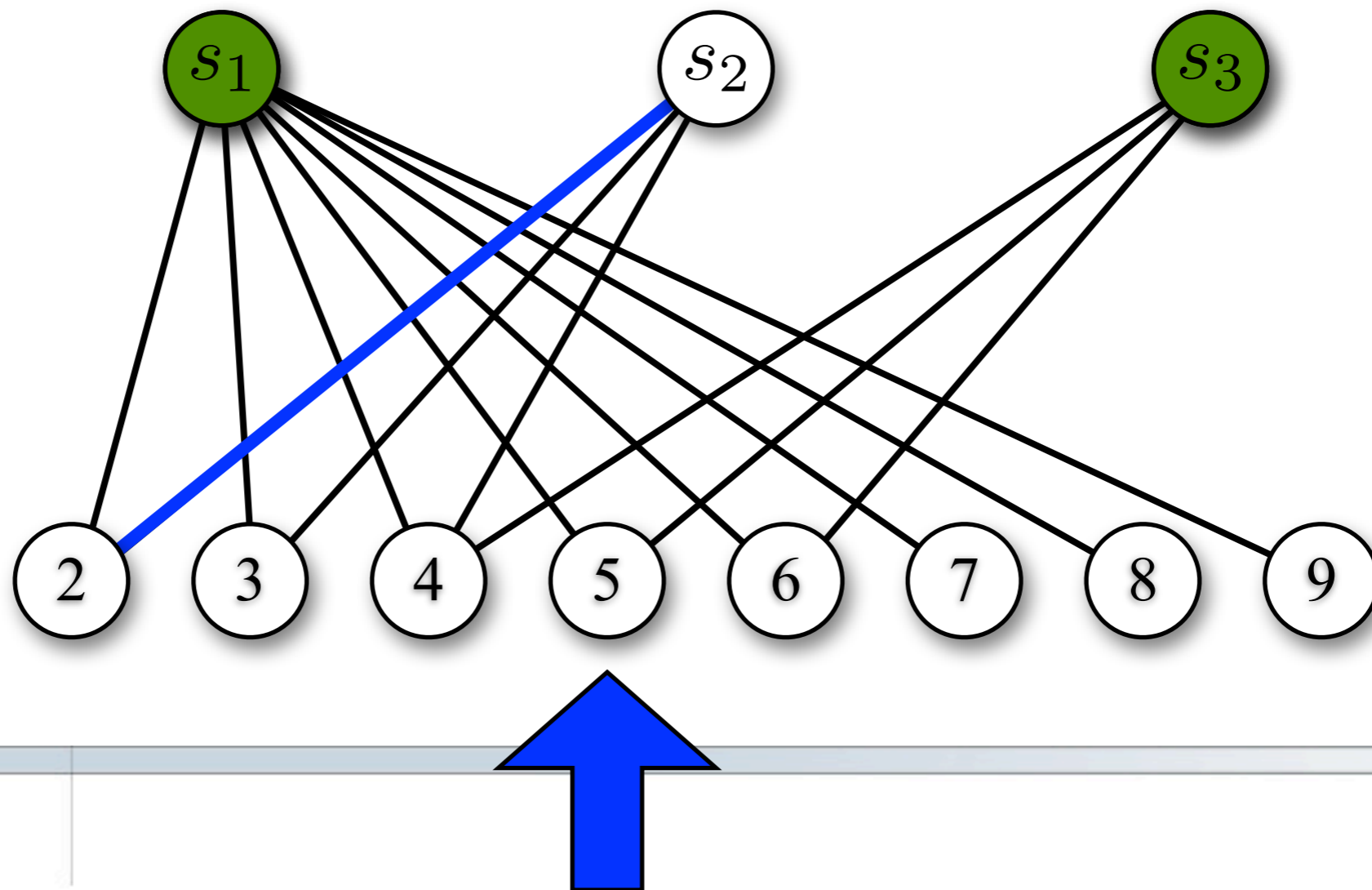
$$s_1 \in [1, 9], s_2 \in [2, 4], s_3 \in [4, 6], p = 3$$



Temps de traitement identiques

Garey, Johnson, Simons, and Tarjan 1981

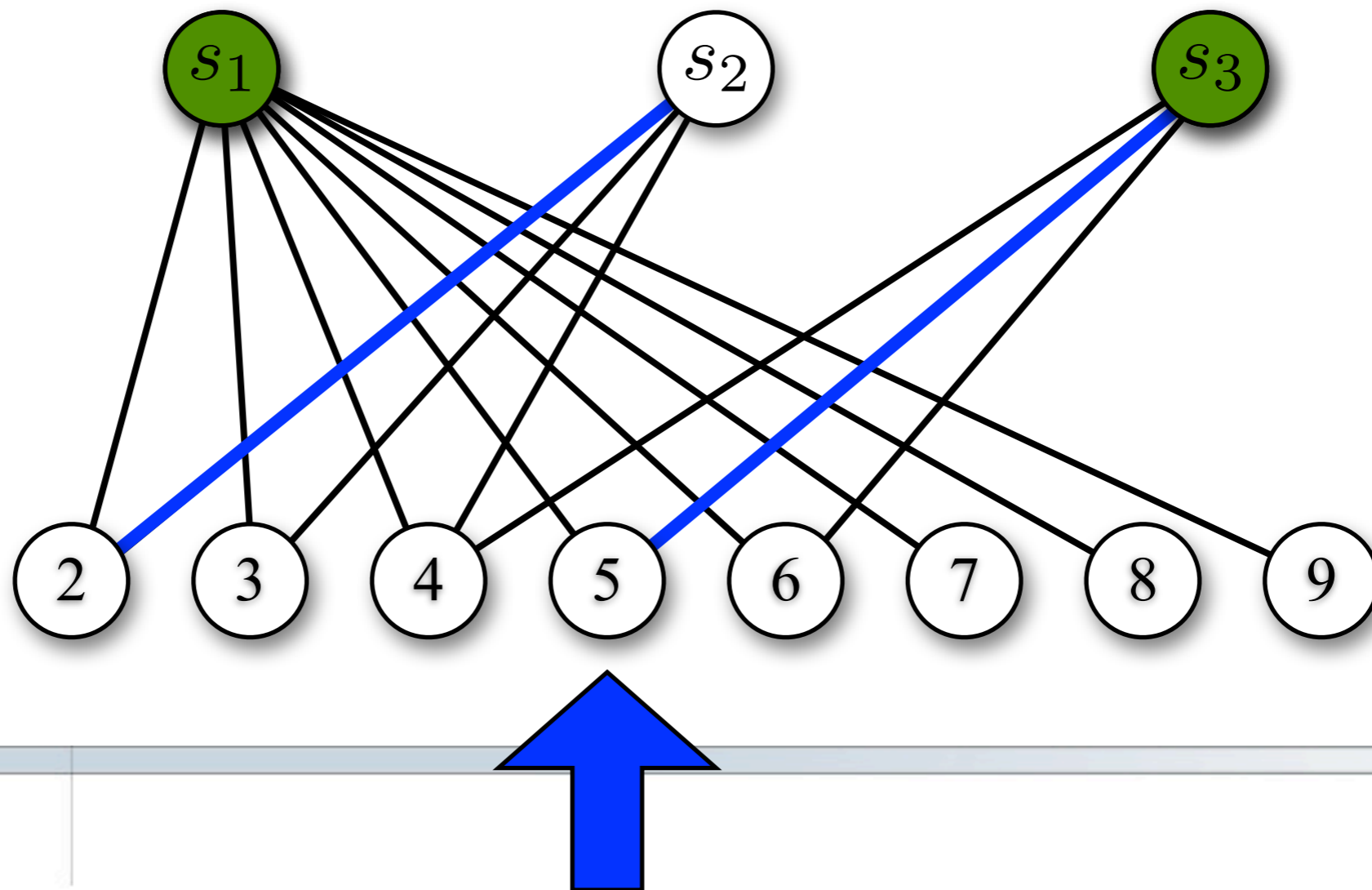
$$s_1 \in [1, 9], s_2 \in [2, 4], s_3 \in [4, 6], p = 3$$



Temps de traitement identiques

Garey, Johnson, Simons, and Tarjan 1981

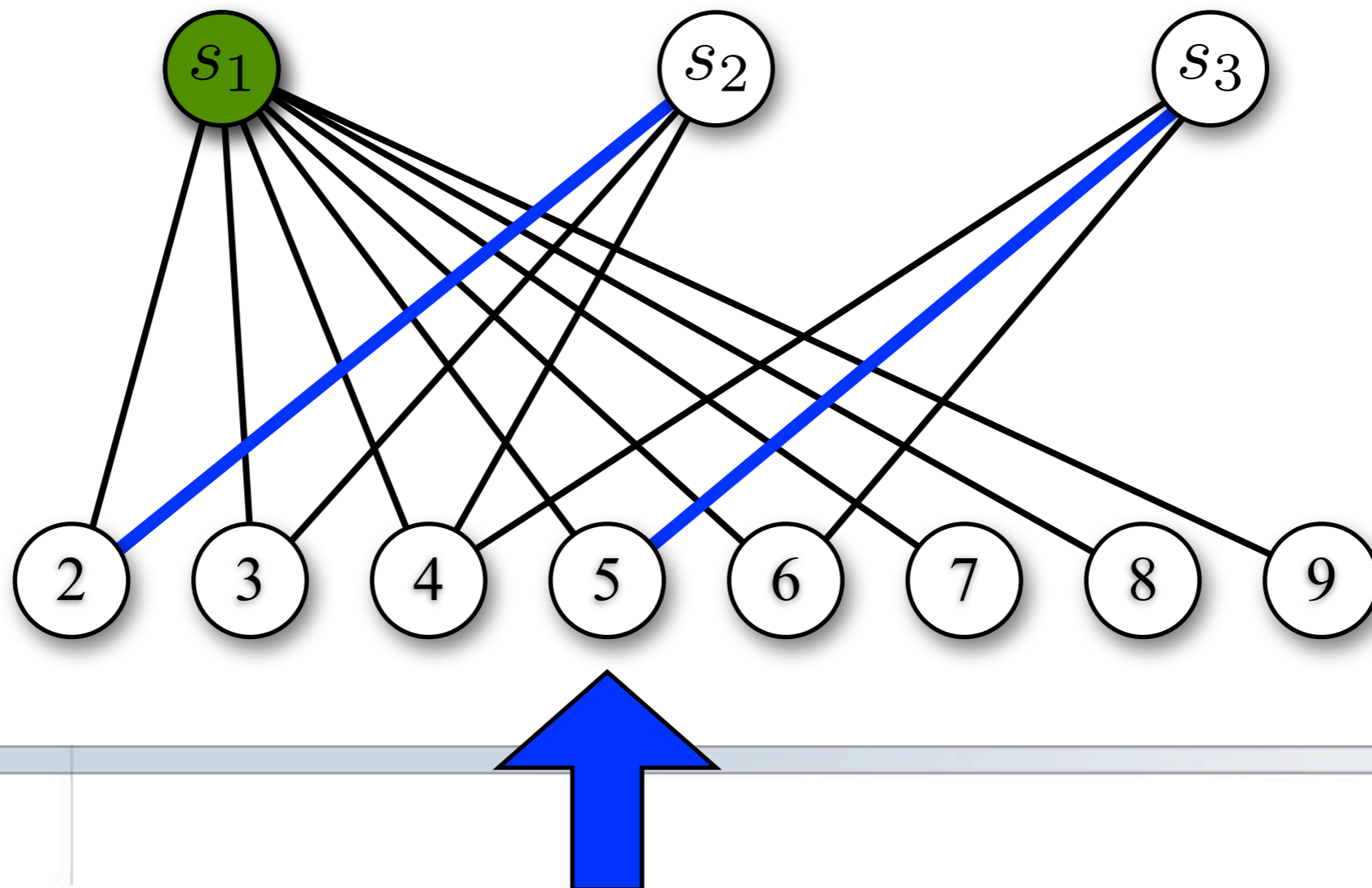
$$s_1 \in [1, 9], s_2 \in [2, 4], s_3 \in [4, 6], p = 3$$



Temps de traitement identiques

Garey, Johnson, Simons, and Tarjan 1981

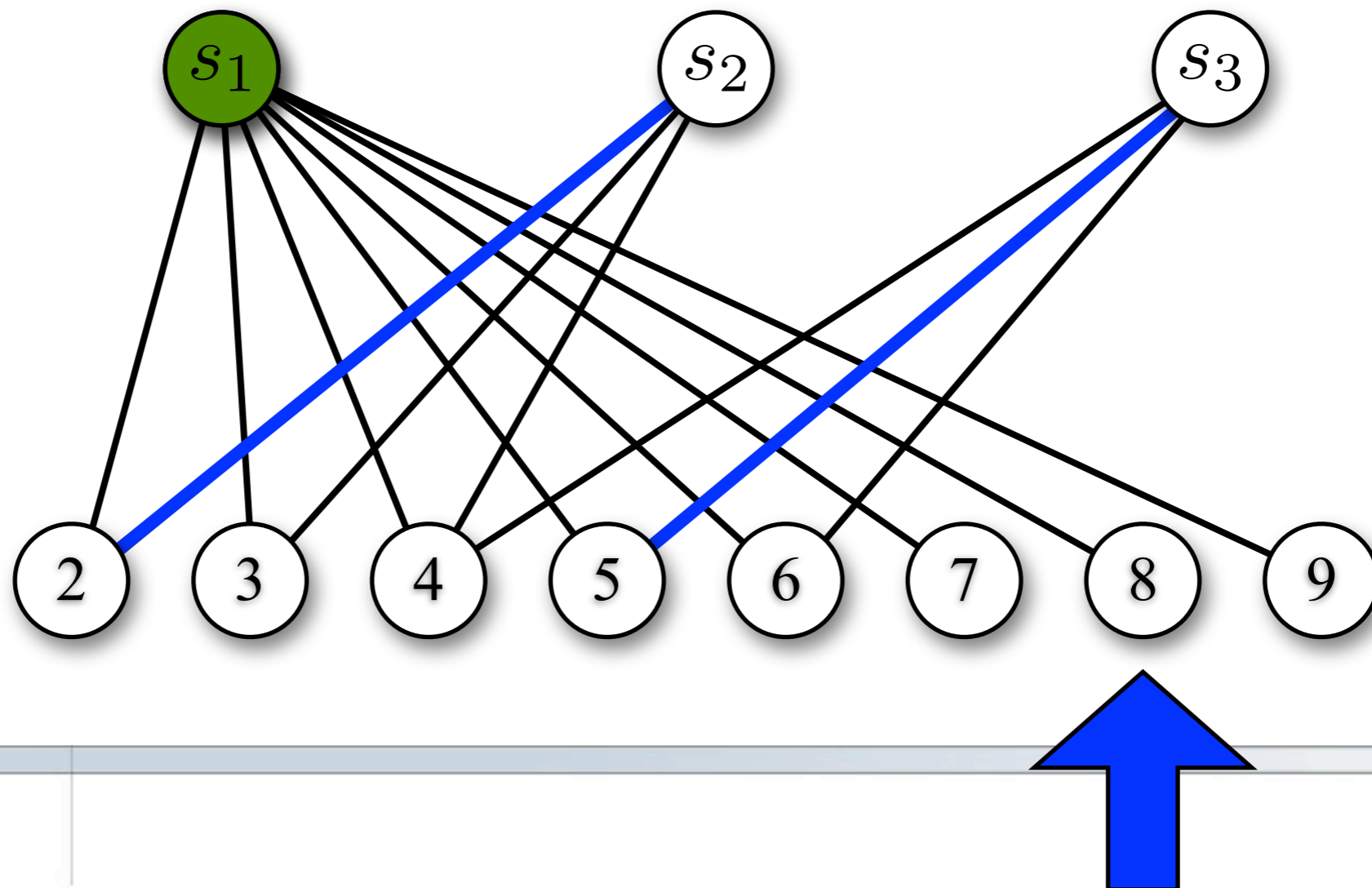
$$s_1 \in [1, 9], s_2 \in [2, 4], s_3 \in [4, 6], p = 3$$



Temps de traitement identiques

Garey, Johnson, Simons, and Tarjan 1981

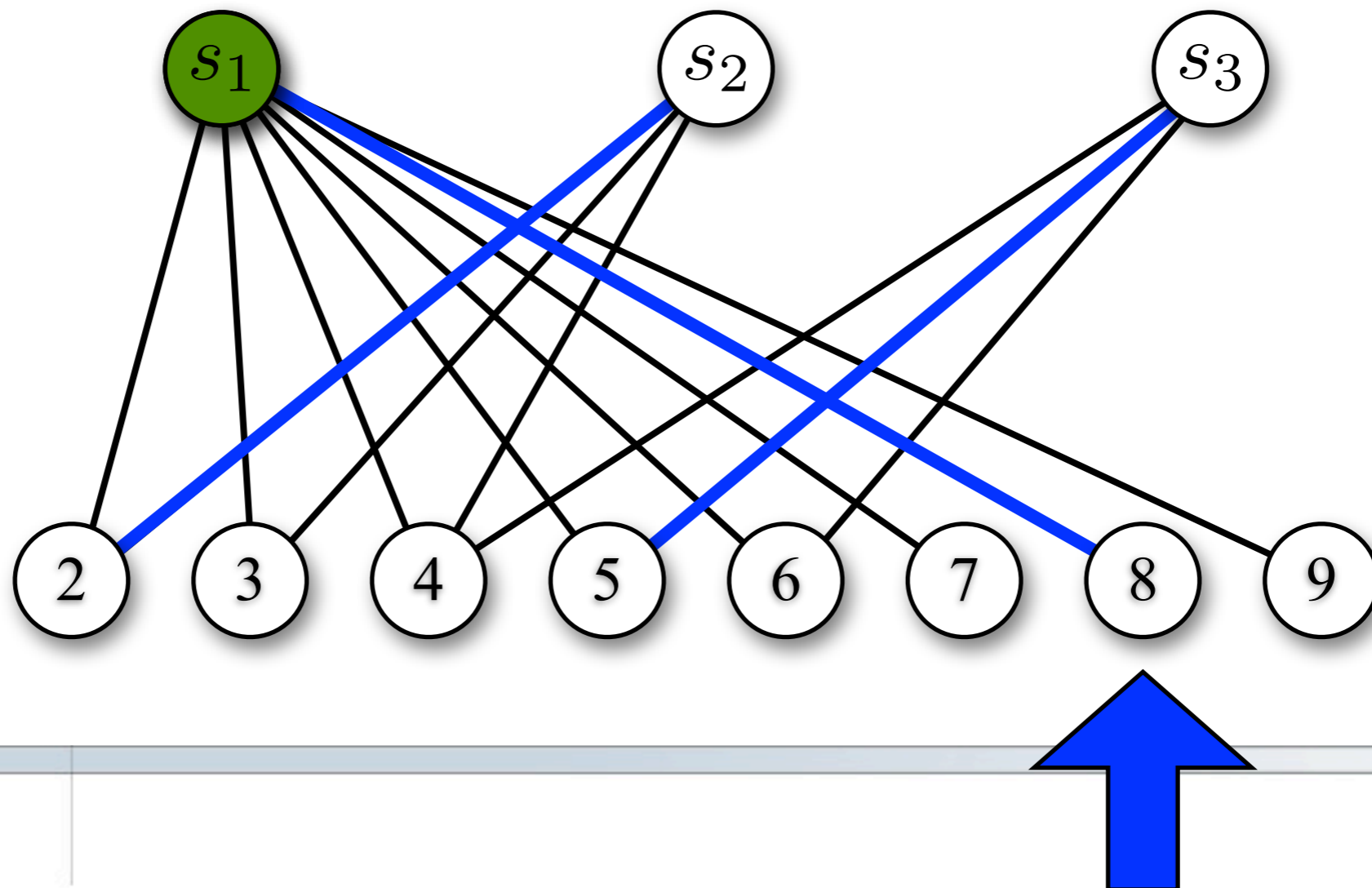
$$s_1 \in [1, 9], s_2 \in [2, 4], s_3 \in [4, 6], p = 3$$



Temps de traitement identiques

Garey, Johnson, Simons, and Tarjan 1981

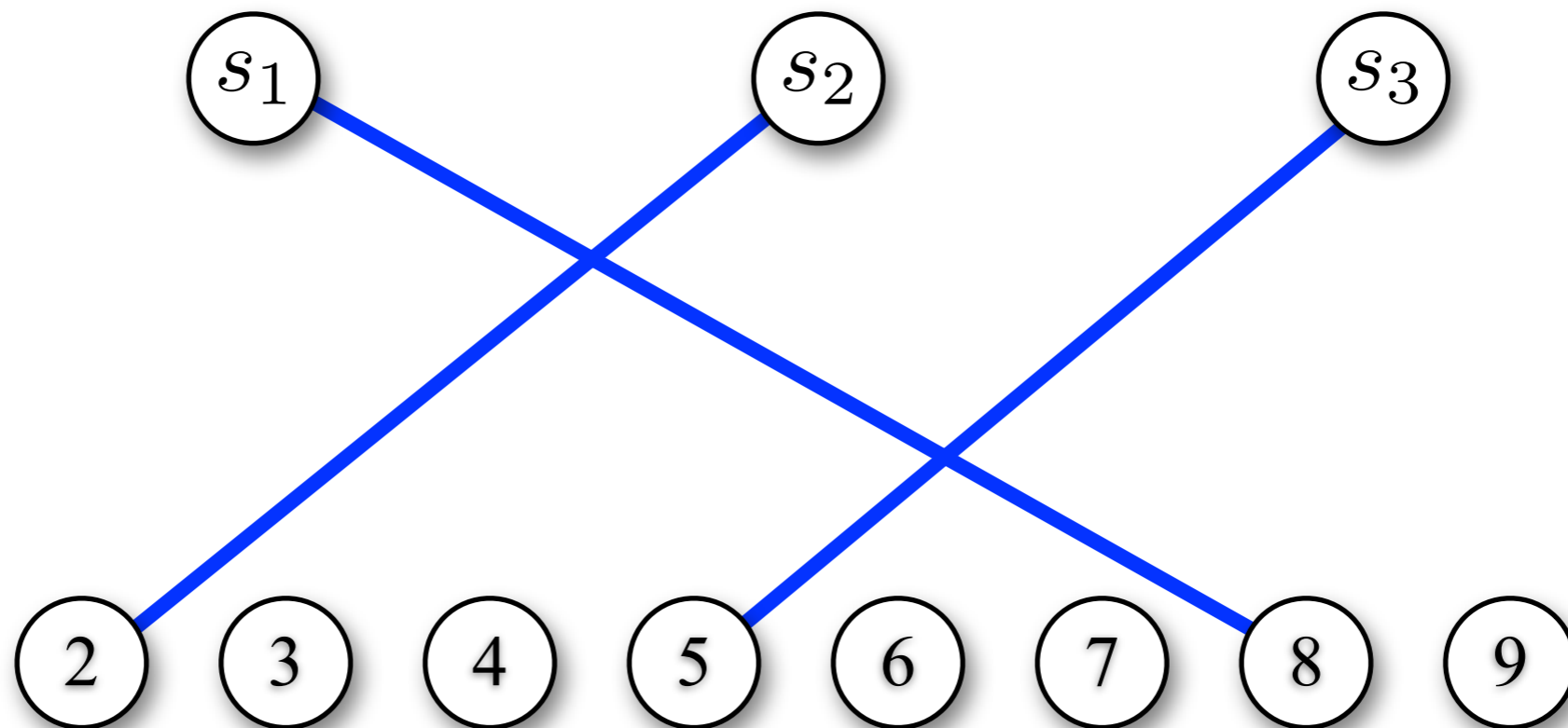
$$s_1 \in [1, 9], s_2 \in [2, 4], s_3 \in [4, 6], p = 3$$



Temps de traitement identiques

Garey, Johnson, Simons, and Tarjan 1981

$$s_1 \in [1, 9], s_2 \in [2, 4], s_3 \in [4, 6], p = 3$$



Temps de traitement identiques

Garey, Johnson, Simons, and Tarjan 1981

- Trouver les régions proscrites: $O(n \log n)$
- Ordonnancer les tâches: $O(n \log n)$
- Total: $O(n \log n)$

Généralisation à m ressources

Auteurs	Année	Complexité
Simons	1983	$O(n^3 \log \log n)$

Généralisation à m ressources

Auteurs	Année	Complexité
Simons	1983	$O(n^3 \log \log n)$
Simons & Warmuth	1989	$O(n^2 m)$

Généralisation à m ressources

Auteurs	Année	Complexité
Simons	1983	$O(n^3 \log \log n)$
Simons & Warmuth	1989	$O(n^2 m)$
Vakhania	2002	$O(d_{\max}^2 (m + d_{\max}) n \log n)$

Généralisation à m ressources

Auteurs	Année	Complexité
Simons	1983	$O(n^3 \log \log n)$
Simons & Warmuth	1989	$O(n^2 m)$
Vakhania	2002	$O(d_{\max}^2 (m + d_{\max}) n \log n)$
Dürr & Hurand	2009	$O(n^3)$

Généralisation à m ressources

Auteurs	Année	Complexité
Simons	1983	$O(n^3 \log \log n)$
Simons & Warmuth	1989	$O(n^2 m)$
Vakhania	2002	$O(d_{\max}^2 (m + d_{\max}) n \log n)$
Dürr & Hurand	2009	$O(n^3)$
López-Ortiz & Quimper	2010	$O(\min(1, \frac{p}{m}) n^2)$

Division du problème



- Si un oracle fournissait les temps de début des tâches...

Division du problème

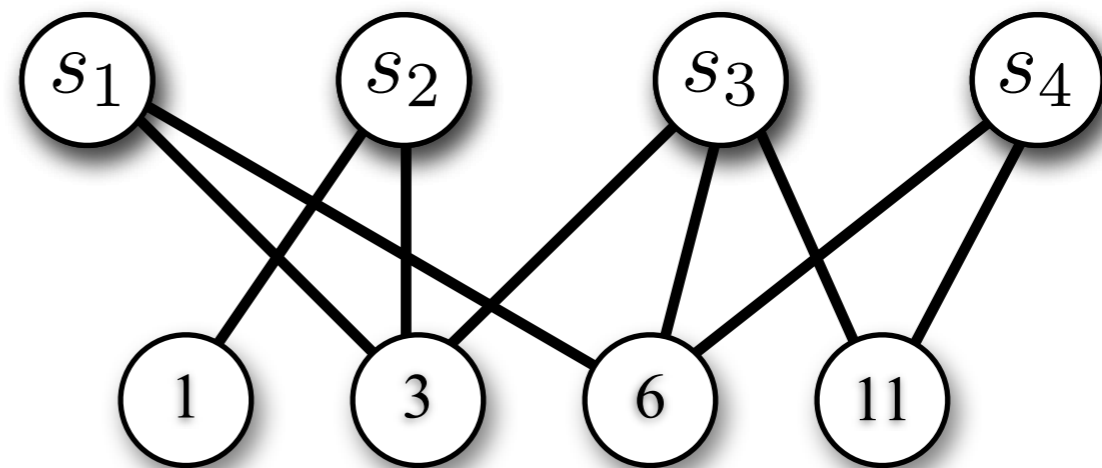


- Si un oracle fournissait les temps de début des tâches...
- alors un couplage dans un graphe serait suffisant pour ordonnancer les tâches.

Division du problème



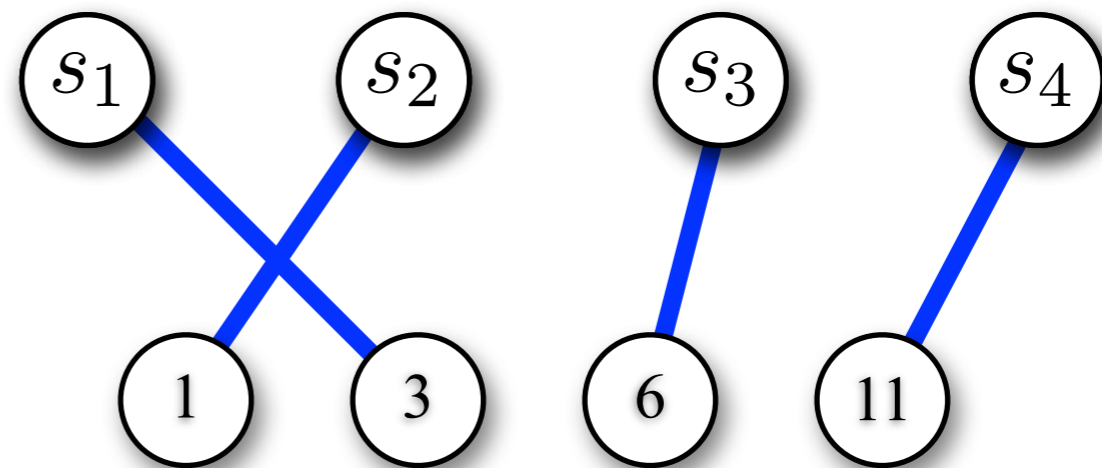
- Si un oracle fournissait les temps de début des tâches...
- alors un couplage dans un graphe serait suffisant pour ordonnancer les tâches.



Division du problème



- Si un oracle fournissait les temps de début des tâches...
- alors un couplage dans un graphe serait suffisant pour ordonnancer les tâches.



Formulation du problème

- Soit $s_i \in [r_i, u_i)$ le temps d'exécution de la tâche i
- Soit x_t le nombre de tâches débutant au temps t

Contraintes de ressources

- Pas plus de m tâches commencent dans une fenêtre de temps p .



Contraintes de ressources

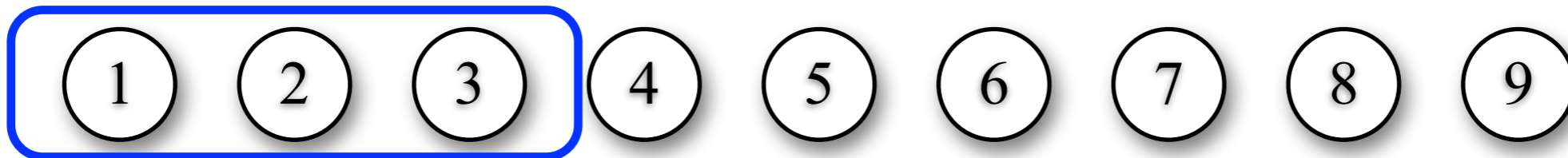
- Pas plus de m tâches commencent dans une fenêtre de temps p .



Contraintes de ressources

- Pas plus de m tâches commencent dans une fenêtre de temps p .

$$x_1 + x_2 + x_3 \leq m$$

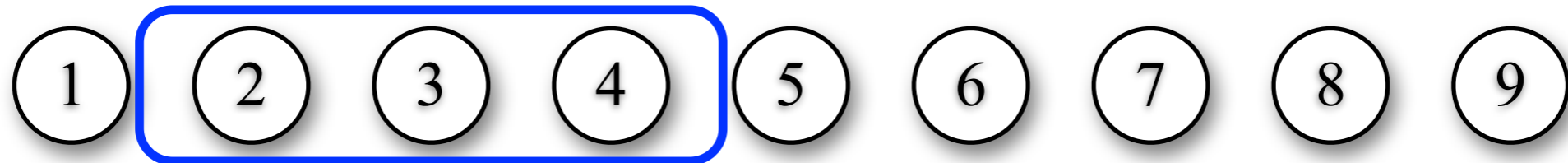


Contraintes de ressources

- Pas plus de m tâches commencent dans une fenêtre de temps p .

$$x_1 + x_2 + x_3 \leq m$$

$$x_2 + x_3 + x_4 \leq m$$



Contraintes de ressources

- Pas plus de m tâches commencent dans une fenêtre de temps p .

$$x_1 + x_2 + x_3 \leq m$$

$$x_2 + x_3 + x_4 \leq m$$

$$x_3 + x_4 + x_5 \leq m$$

...



Contraintes de ressources

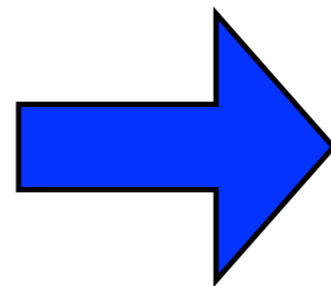
- Pas plus de m tâches commencent dans une fenêtre de temps p .

$$x_1 + x_2 + x_3 \leq m$$

$$x_2 + x_3 + x_4 \leq m$$

$$x_3 + x_4 + x_5 \leq m$$

...



$$\sum_{t=i}^{i+p-1} x_t \leq m$$

1

2

3

4

5

6

7

8

9

Formulation du problème

- Soit $s_i \in [r_i, u_i)$ le temps d'exécution de la tâche i
- Soit x_t le nombre de tâches débutant au temps t

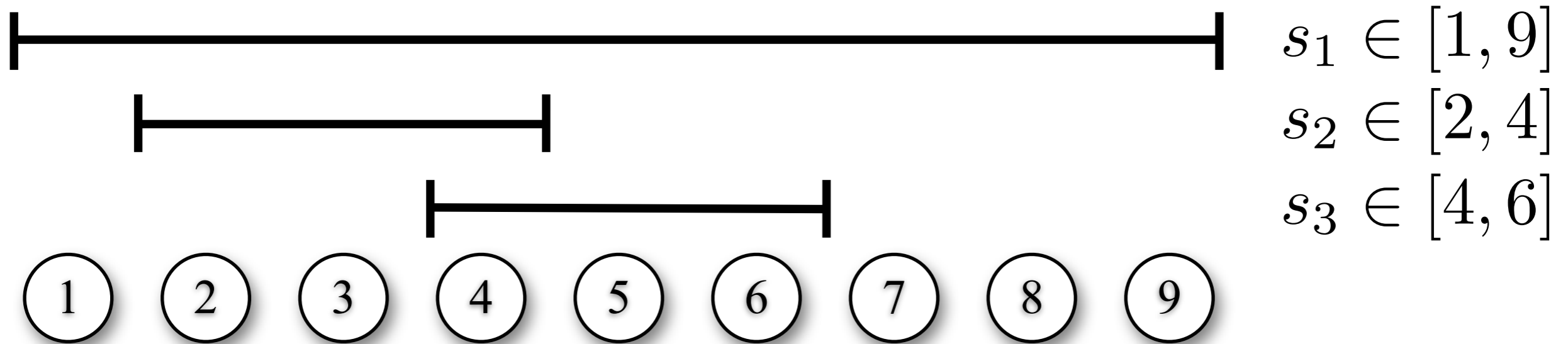
Formulation du problème

- Soit $s_i \in [r_i, u_i)$ le temps d'exécution de la tâche i
- Soit x_t le nombre de tâches débutant au temps t

$$\sum_{t=i}^{i+p-1} x_t \leq m$$

Contraintes de tâches

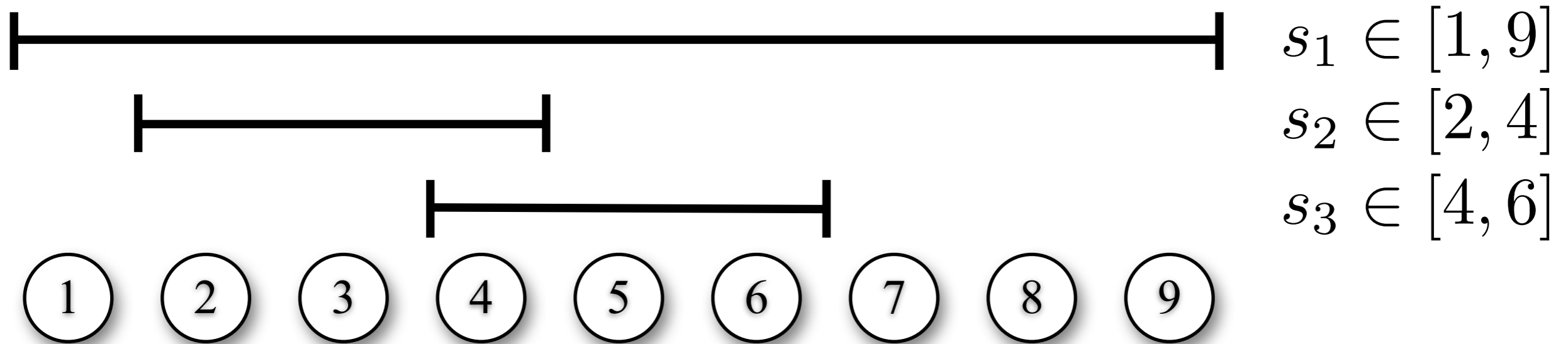
- Le début d'une tâche doit appartenir à un intervalle donné: $s_i \in [r_i, u_i)$



Contraintes de tâches

- Le début d'une tâche doit appartenir à un intervalle donné: $s_i \in [r_i, u_i)$

$$x_2 + x_3 + x_4 \geq 1$$

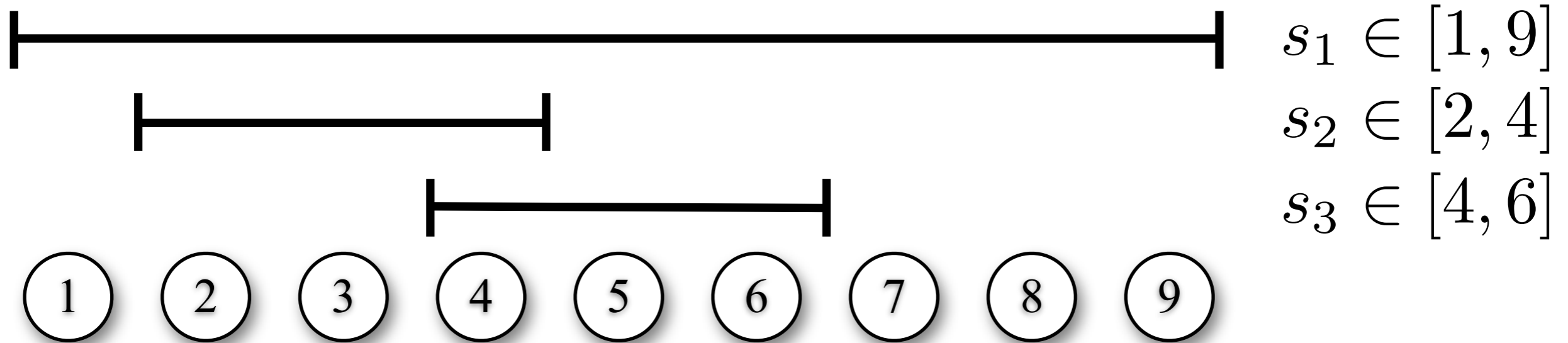


Contraintes de tâches

- Le début d'une tâche doit appartenir à un intervalle donné: $s_i \in [r_i, u_i)$

$$x_2 + x_3 + x_4 \geq 1$$

$$x_4 + x_5 + x_6 \geq 1$$



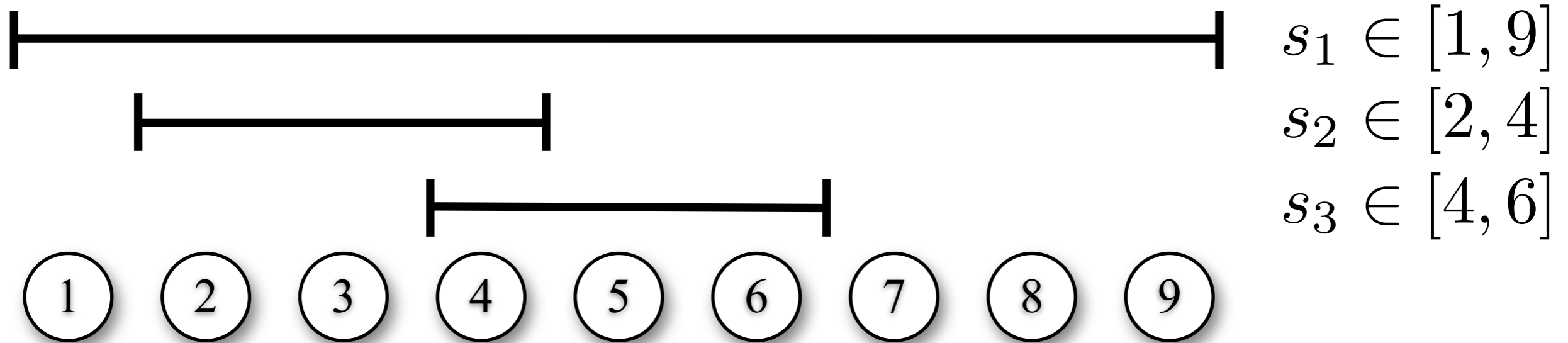
Contraintes de tâches

- Le début d'une tâche doit appartenir à un intervalle donné: $s_i \in [r_i, u_i)$

$$x_2 + x_3 + x_4 \geq 1$$

$$x_4 + x_5 + x_6 \geq 1$$

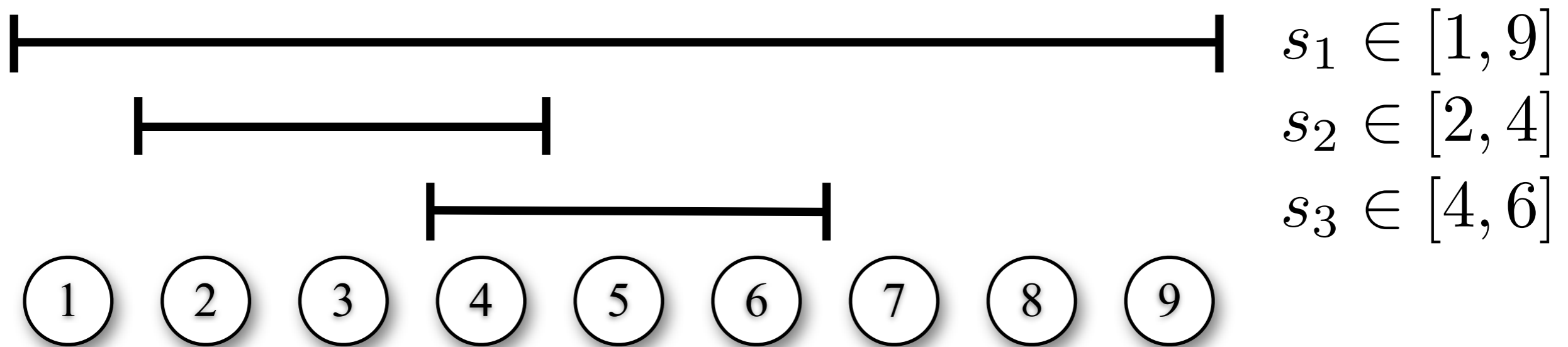
$$x_2 + x_3 + x_4 + x_5 + x_6 \geq 2$$



Contraintes de tâches

- Le début d'une tâche doit appartenir à un intervalle donné: $s_i \in [r_i, u_i)$

$$\begin{aligned} x_2 + x_3 + x_4 &\geq 1 \\ x_4 + x_5 + x_6 &\geq 1 \\ x_2 + x_3 + x_4 + x_5 + x_6 &\geq 2 \end{aligned} \quad \rightarrow \quad \sum_{t=r_i}^{u_j-1} x_t \geq |\{k \mid r_i \leq r_k, u_k \leq u_j\}|$$



Formulation du problème

- Soit $s_i \in [r_i, u_i)$ le temps d'exécution de la tâche i
- Soit x_t le nombre de tâches débutant au temps t

$$\sum_{t=i}^{i+p-1} x_t \leq m$$

Formulation du problème

- Soit $s_i \in [r_i, u_i)$ le temps d'exécution de la tâche i
- Soit x_t le nombre de tâches débutant au temps t

$$\sum_{t=i}^{i+p-1} x_t \leq m$$

$$\sum_{t=r_i}^{u_j-1} x_t \geq |\{k \mid r_i \leq r_k, u_k \leq u_j\}|$$

Formulation du problème

- Soit $s_i \in [r_i, u_i)$ le temps d'exécution de la tâche i
- Soit x_t le nombre de tâches débutant au temps t

$$\sum_{t=i}^{i+p-1} x_t \leq m$$

$$\sum_{t=r_i}^{u_j-1} x_t \geq |\{k \mid r_i \leq r_k, u_k \leq u_j\}|$$

$$x_t \geq 0$$

Formulation du problème

- Changement de variable $y_t = \sum_{i=1}^{t-1} x_i$

$$\sum_{t=i}^{i+p-1} x_t \leq m$$

$$\sum_{t=r_i}^{u_j-1} x_t \geq |\{k \mid r_i \leq r_k, u_k \leq u_j\}|$$

$$x_t \geq 0$$

Formulation du problème

- Changement de variable $y_t = \sum_{i=1}^{t-1} x_i$

$$\sum_{t=i}^{i+p-1} x_t \leq m \quad \longrightarrow \quad y_{i+p} - y_i \leq m$$

$$\sum_{t=r_i}^{u_j-1} x_t \geq |\{k \mid r_i \leq r_k, u_k \leq u_j\}|$$

$$x_t \geq 0$$

Formulation du problème

- Changement de variable $y_t = \sum_{i=1}^{t-1} x_i$

$$y_{i+p} - y_i \leq m$$

$$\sum_{t=r_i}^{u_j-1} x_t \geq |\{k \mid r_i \leq r_k, u_k \leq u_j\}|$$

$$x_t \geq 0$$

Formulation du problème

- Changement de variable $y_t = \sum_{i=1}^{t-1} x_i$

$$y_{i+p} - y_i \leq m$$

$$y_{u_j} - y_{r_i} \geq |\{k \mid r_i \leq r_k, u_k \leq u_j\}|$$

$$x_t \geq 0$$

Formulation du problème

- Changement de variable $y_t = \sum_{i=1}^{t-1} x_i$

$$y_{i+p} - y_i \leq m$$

$$y_{u_j} - y_{r_i} \geq |\{k \mid r_i \leq r_k, u_k \leq u_j\}|$$

$$y_{t+1} - y_t \geq 0$$

Formulation du problème

- Changement de variable $y_t = \sum_{i=1}^{t-1} x_i$

$$y_{i+p} - y_i \leq m$$

$$y_{r_i} - y_{u_j} \leq -|\{k \mid r_i \leq r_k, u_k \leq u_j\}|$$

$$y_{t+1} - y_t \geq 0$$

Formulation du problème

- Changement de variable $y_t = \sum_{i=1}^{t-1} x_i$

$$y_{i+p} - y_i \leq m$$

$$y_{r_i} - y_{u_j} \leq -|\{k \mid r_i \leq r_k, u_k \leq u_j\}|$$

$$y_t - y_{t+1} \leq 0$$

Plus court chemin

- Changement de variable $y_t = \sum_{i=1}^{t-1} x_i$

$$y_{i+p} - y_i \leq m$$

$$y_{r_i} - y_{u_j} \leq -|\{k \mid r_i \leq r_k, u_k \leq u_j\}|$$

$$y_t - y_{t+1} \leq 0$$

Plus court chemin

$$y_{i+p} - y_i \leq m$$

$$y_{r_i} - y_{u_j} \leq -|\{k \mid r_i \leq r_k, u_k \leq u_j\}|$$

$$y_t - y_{t+1} \leq 0$$

Plus court chemin

$$y_{i+p} - y_i \leq m$$

$$y_{r_i} - y_{u_j} \leq -|\{k \mid r_i \leq r_k, u_k \leq u_j\}|$$

$$y_t - y_{t+1} \leq 0$$

$$s_1 \in [1, 9)$$

$$s_2 \in [2, 4)$$

$$s_3 \in [4, 6)$$

$$p = 3$$

Plus court chemin

$$y_{i+p} - y_i \leq m$$

$$y_{r_i} - y_{u_j} \leq -|\{k \mid r_i \leq r_k, u_k \leq u_j\}|$$

$$y_t - y_{t+1} \leq 0$$

$$s_1 \in [1, 9)$$

$$s_2 \in [2, 4)$$

$$s_3 \in [4, 6)$$



$$p = 3$$

Plus court chemin

$$y_{i+p} - y_i \leq m$$

$$y_{r_i} - y_{u_j} \leq -|\{k \mid r_i \leq r_k, u_k \leq u_j\}|$$

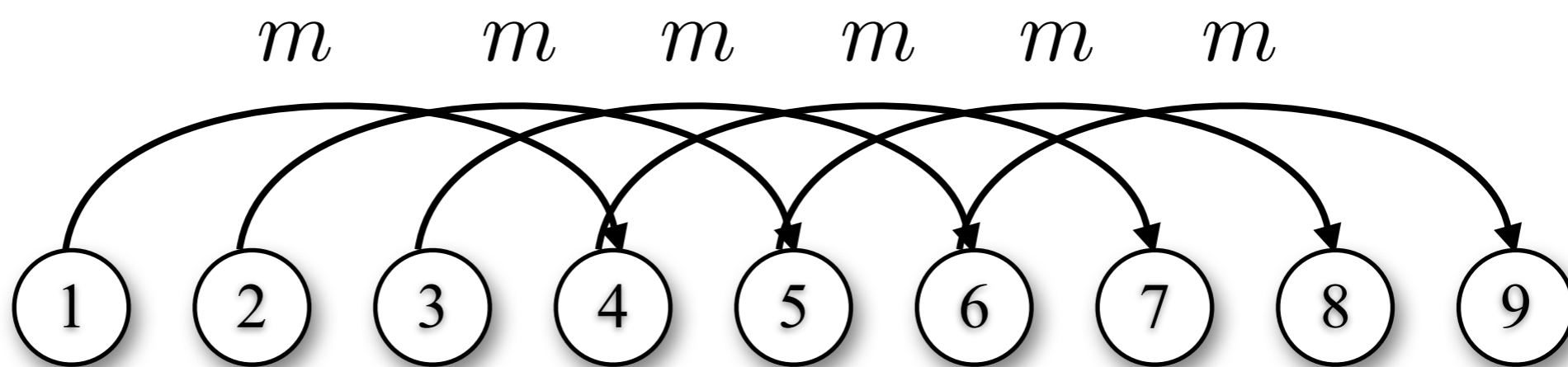
$$y_t - y_{t+1} \leq 0$$

$$s_1 \in [1, 9)$$

$$s_2 \in [2, 4)$$

$$s_3 \in [4, 6)$$

$$p = 3$$



Plus court chemin

$$y_{i+p} - y_i \leq m$$

$$y_{r_i} - y_{u_j} \leq -|\{k \mid r_i \leq r_k, u_k \leq u_j\}|$$

$$y_t - y_{t+1} \leq 0$$

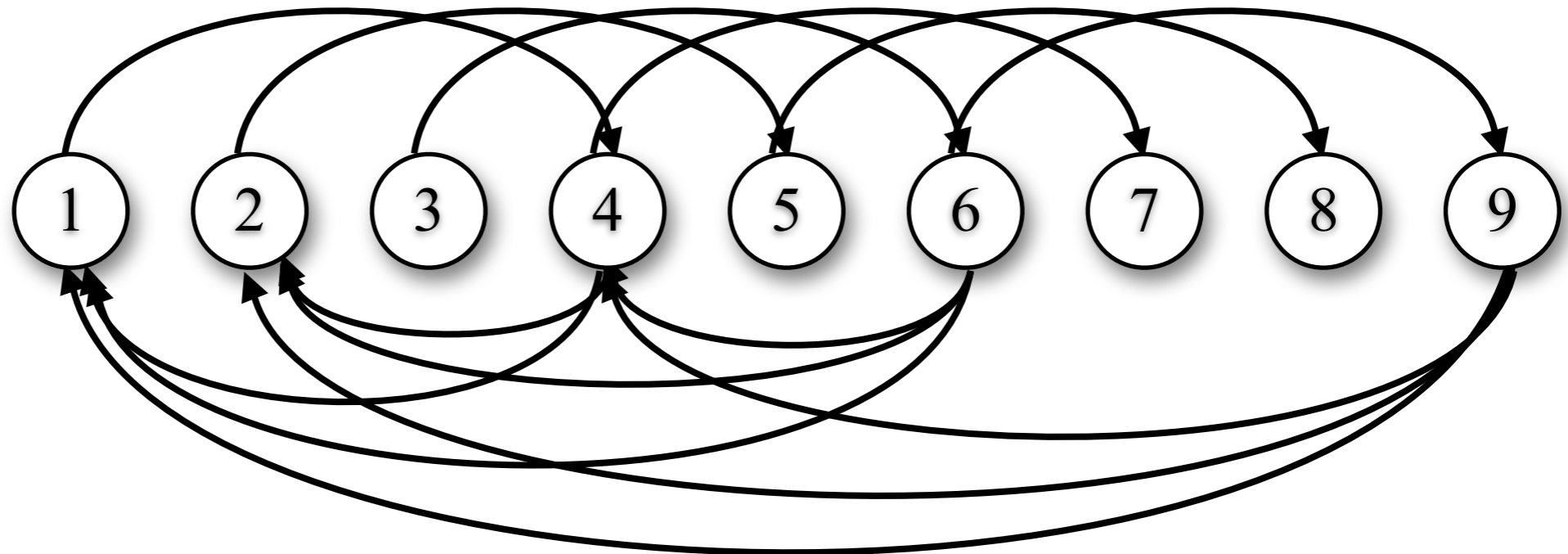
$$s_1 \in [1, 9)$$

$$s_2 \in [2, 4)$$

$$s_3 \in [4, 6)$$

$$p = 3$$

m m m m m m



$$-|\{k \mid r_i \leq r_k, u_k \leq u_j\}|$$

Plus court chemin

$$y_{i+p} - y_i \leq m$$

$$y_{r_i} - y_{u_j} \leq -|\{k \mid r_i \leq r_k, u_k \leq u_j\}|$$

$$y_t - y_{t+1} \leq 0$$

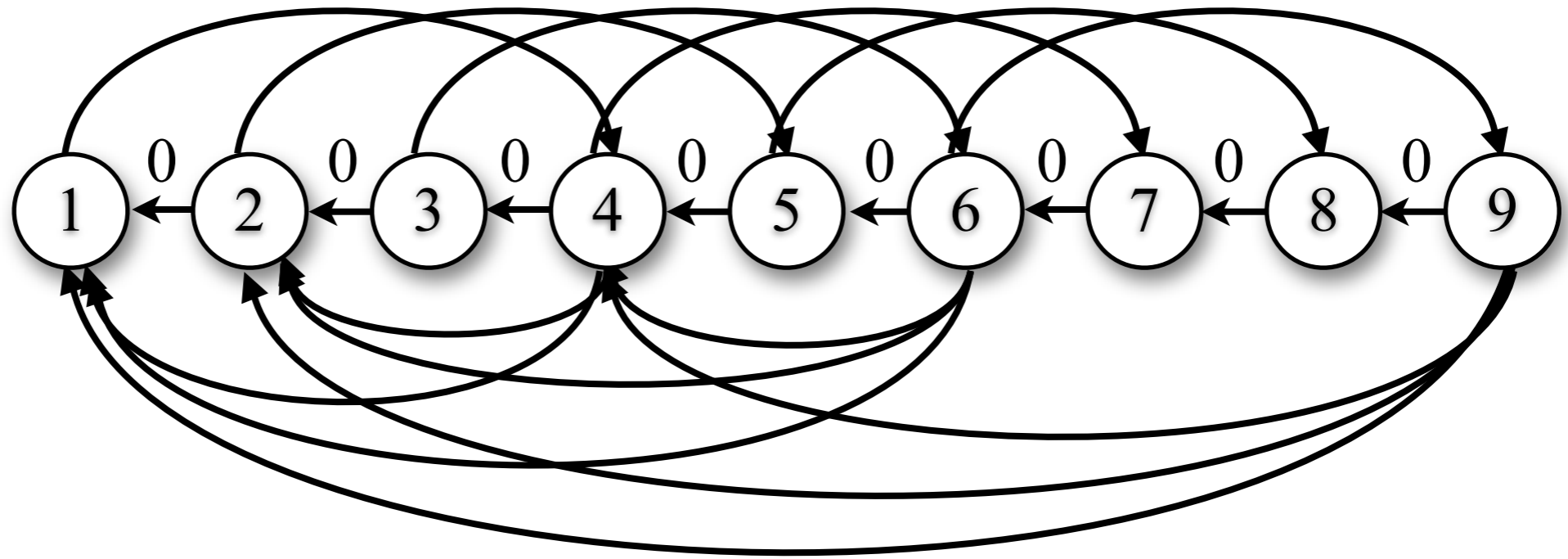
$$s_1 \in [1, 9)$$

$$s_2 \in [2, 4)$$

$$s_3 \in [4, 6)$$

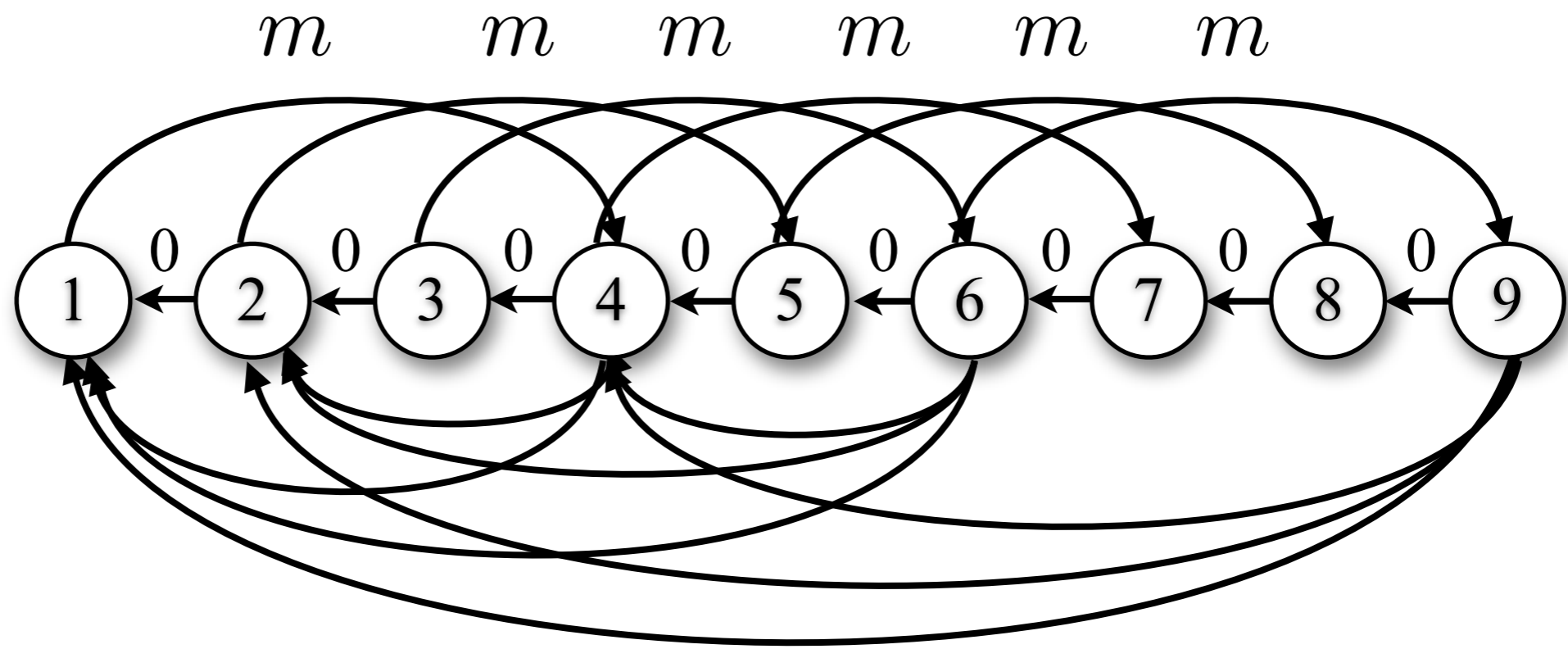
$$p = 3$$

$m \quad m \quad m \quad m \quad m \quad m$



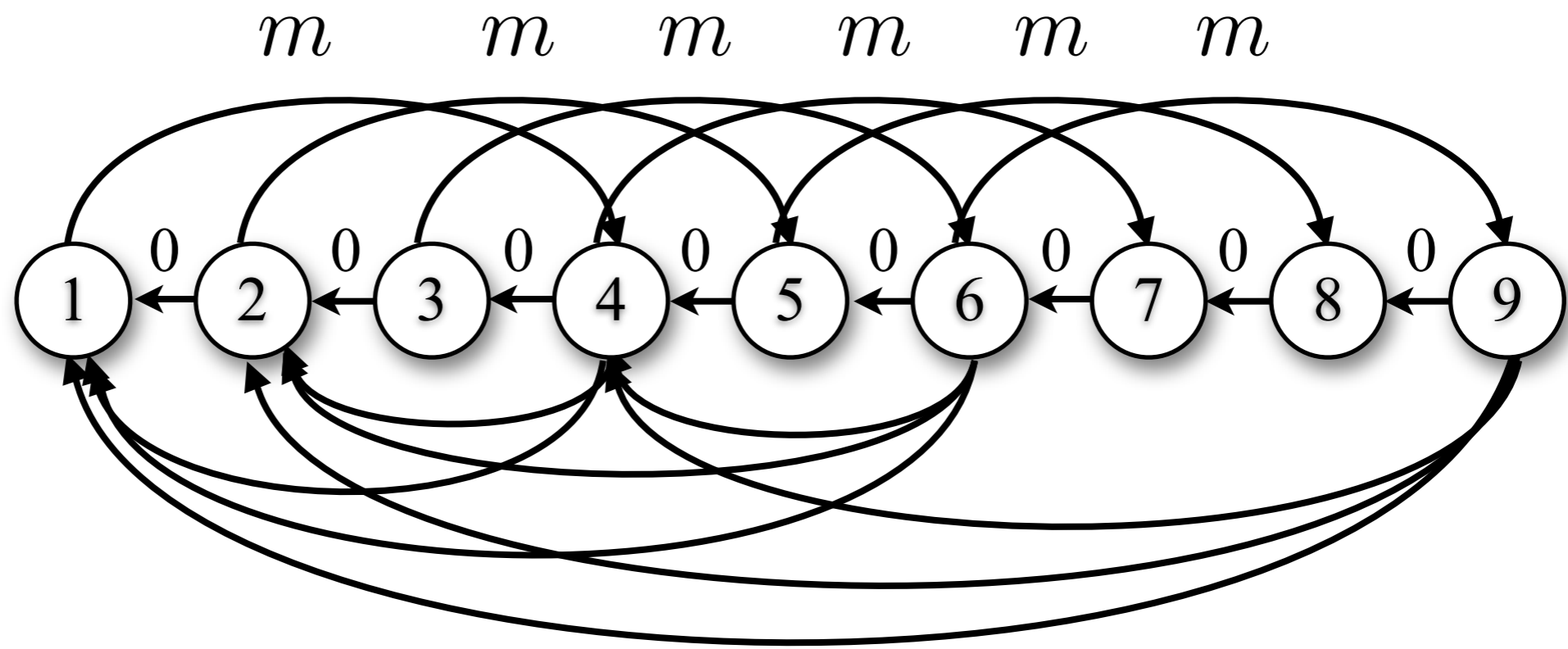
$$-|\{k \mid r_i \leq r_k, u_k \leq u_j\}|$$

Plus court chemin



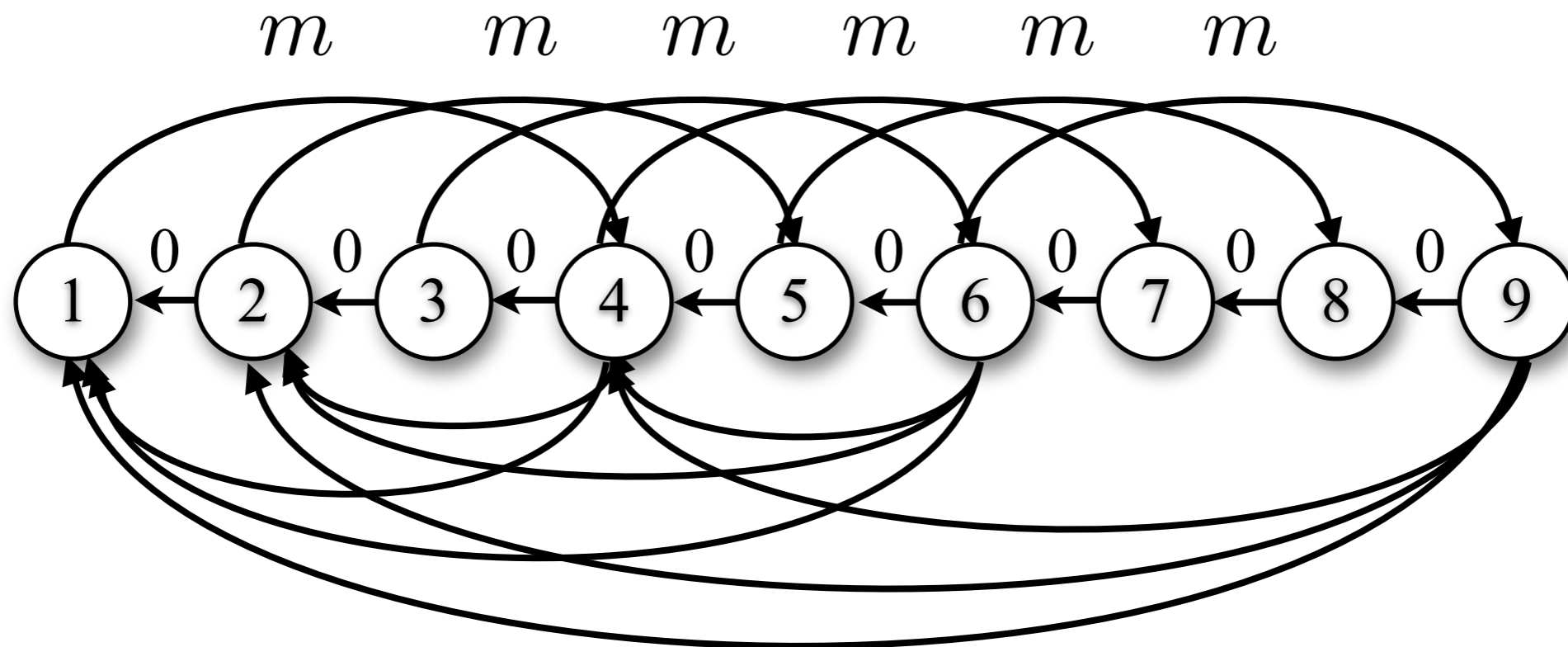
$$- |\{k \mid r_i \leq r_k, u_k \leq u_j\}|$$

Existence d'une solution



$$- |\{k \mid r_i \leq r_k, u_k \leq u_j\}|$$

Existence d'une solution

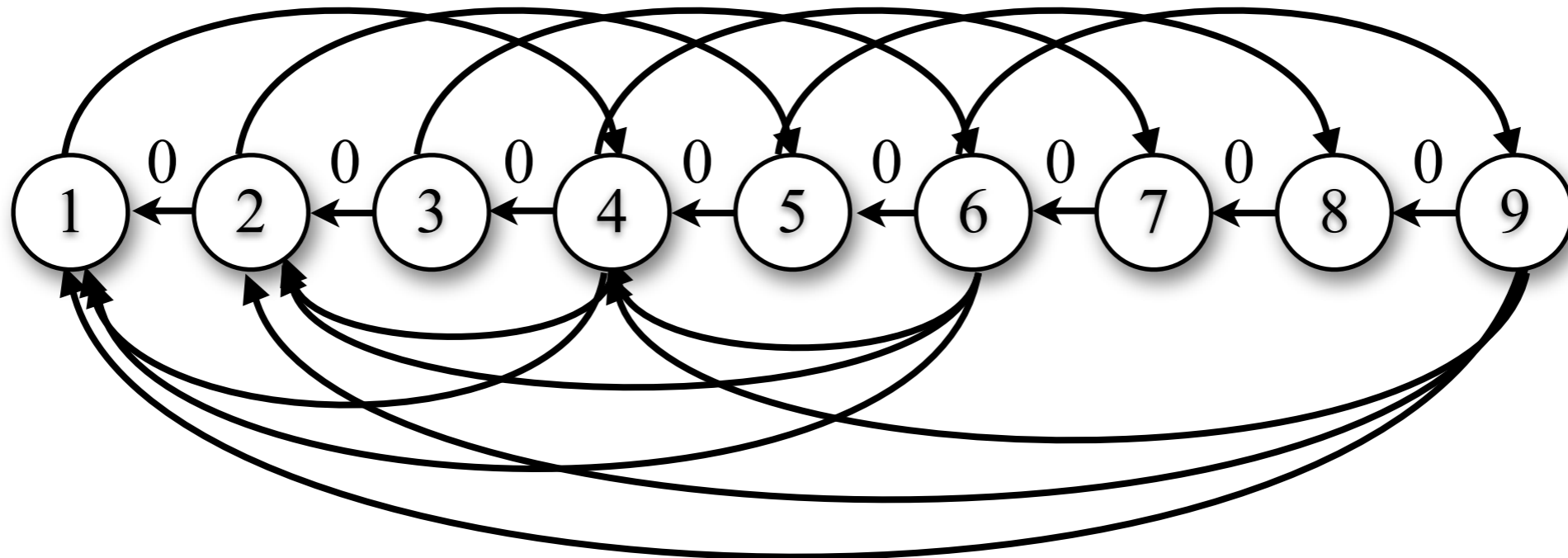


$$-|\{k \mid r_i \leq r_k, u_k \leq u_j\}|$$

- Le problème a une solution ssi le graphe n'a pas de cycle négatif.

Trouver une solution

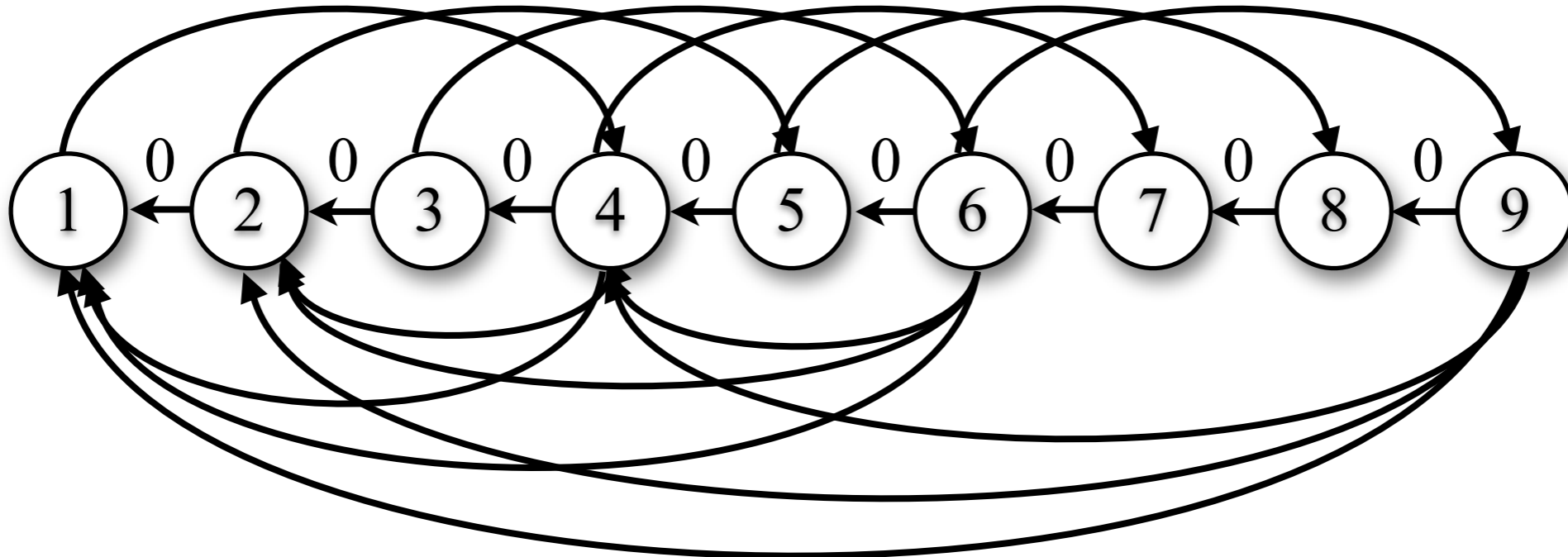
m m m m m m



$$-|\{k \mid r_i \leq r_k, u_k \leq u_j\}|$$

Trouver une solution

m m m m m m

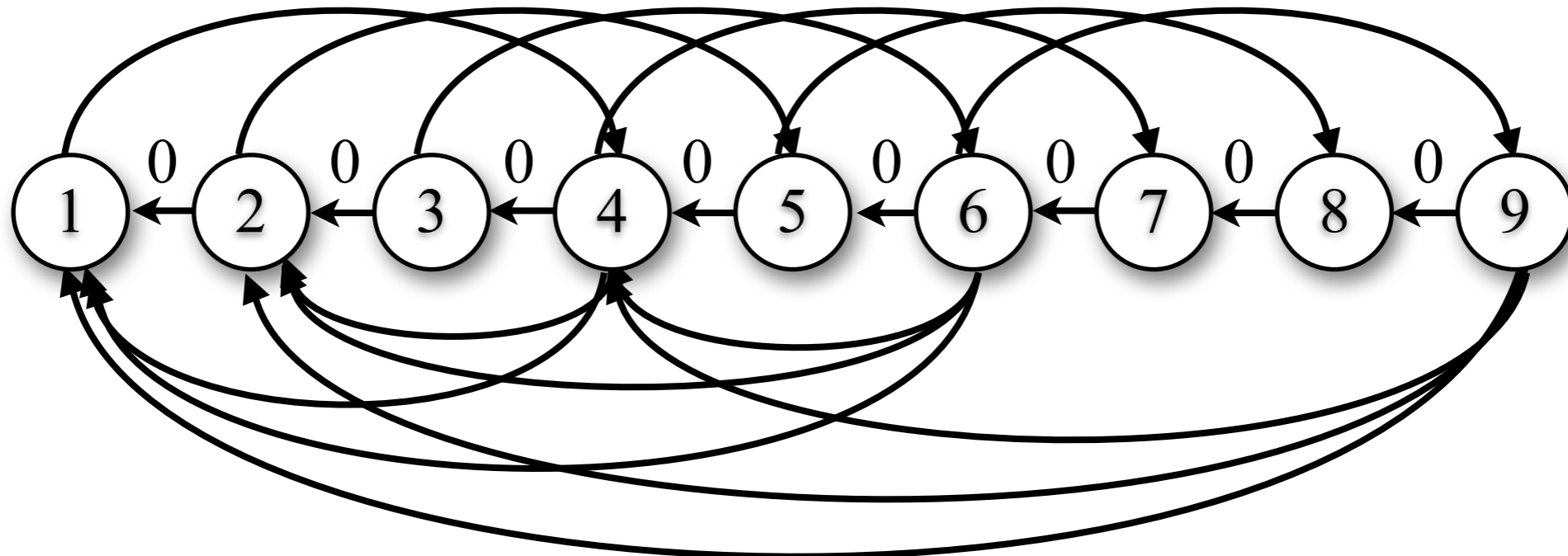


$$-|\{k \mid r_i \leq r_k, u_k \leq u_j\}|$$

- Soit $\delta(t)$ le poids du chemin le plus court entre le plus grand noeud et le noeud t .

Trouver une solution

m m m m m m



$$-|\{k \mid r_i \leq r_k, u_k \leq u_j\}|$$

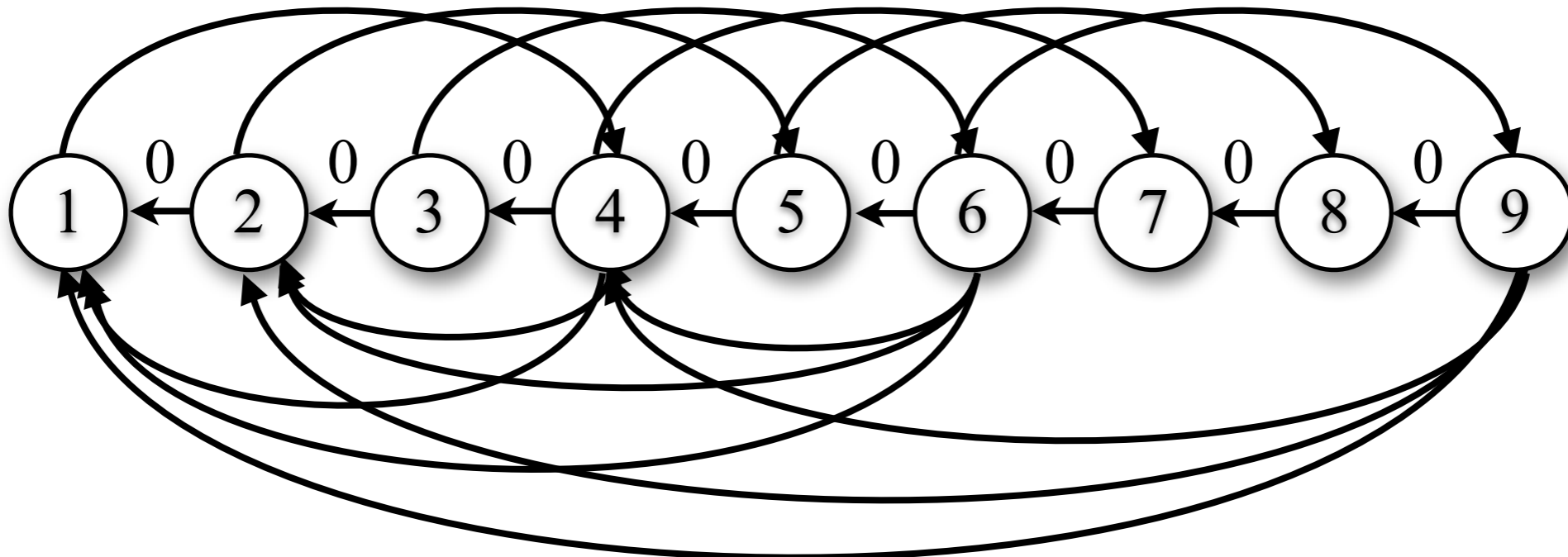
- Soit $\delta(t)$ le poids du chemin le plus court entre le plus grand noeud et le noeud t .
- $y_t = n + \delta(t)$ est une solution.

Algorithme de Bellman-Ford

- $d \leftarrow [\infty, \infty, \dots, \infty]$
- $d[\text{noeud d'origine}] \leftarrow 0$
- Répète $|V|$ fois
 - Pour toutes les arêtes (a, b)
 - Si $d[a] + w(a, b) < d[b]$ alors
 - $d[b] \leftarrow d[a] + w(a, b)$

Vecteur de distance

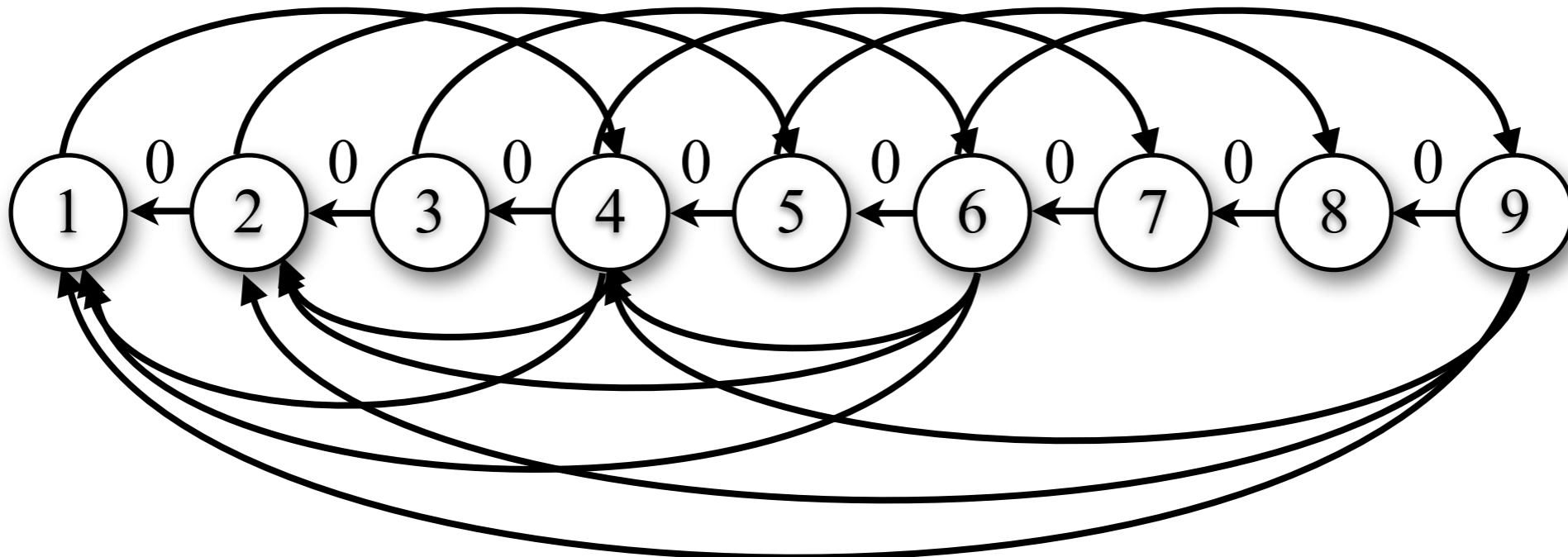
m m m m m m



$$-|\{k \mid r_i \leq r_k, u_k \leq u_j\}|$$

Vecteur de distance

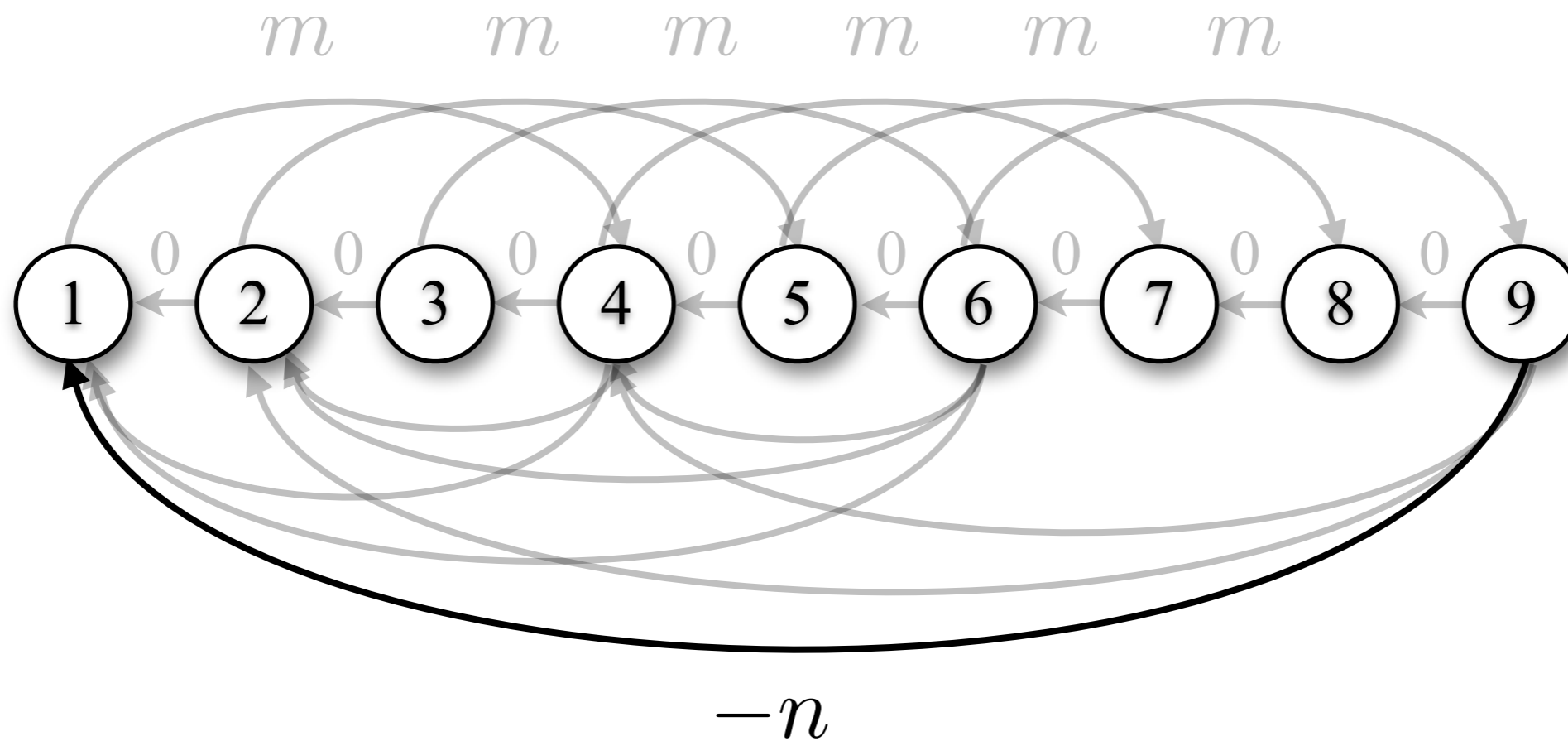
m m m m m m



$$- |\{k \mid r_i \leq r_k, u_k \leq u_j\}|$$

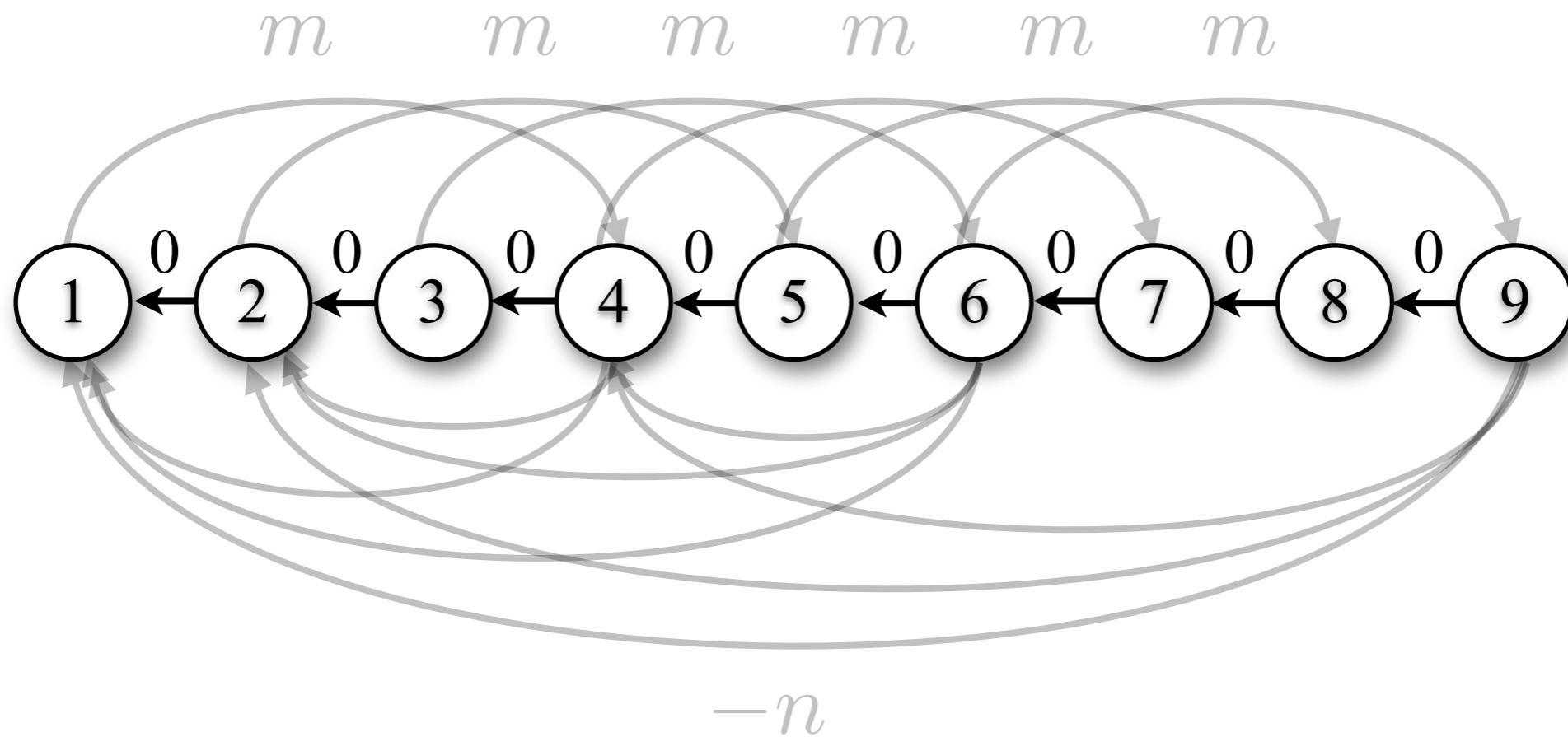
- $d[9] = 0$

Vecteur de distance



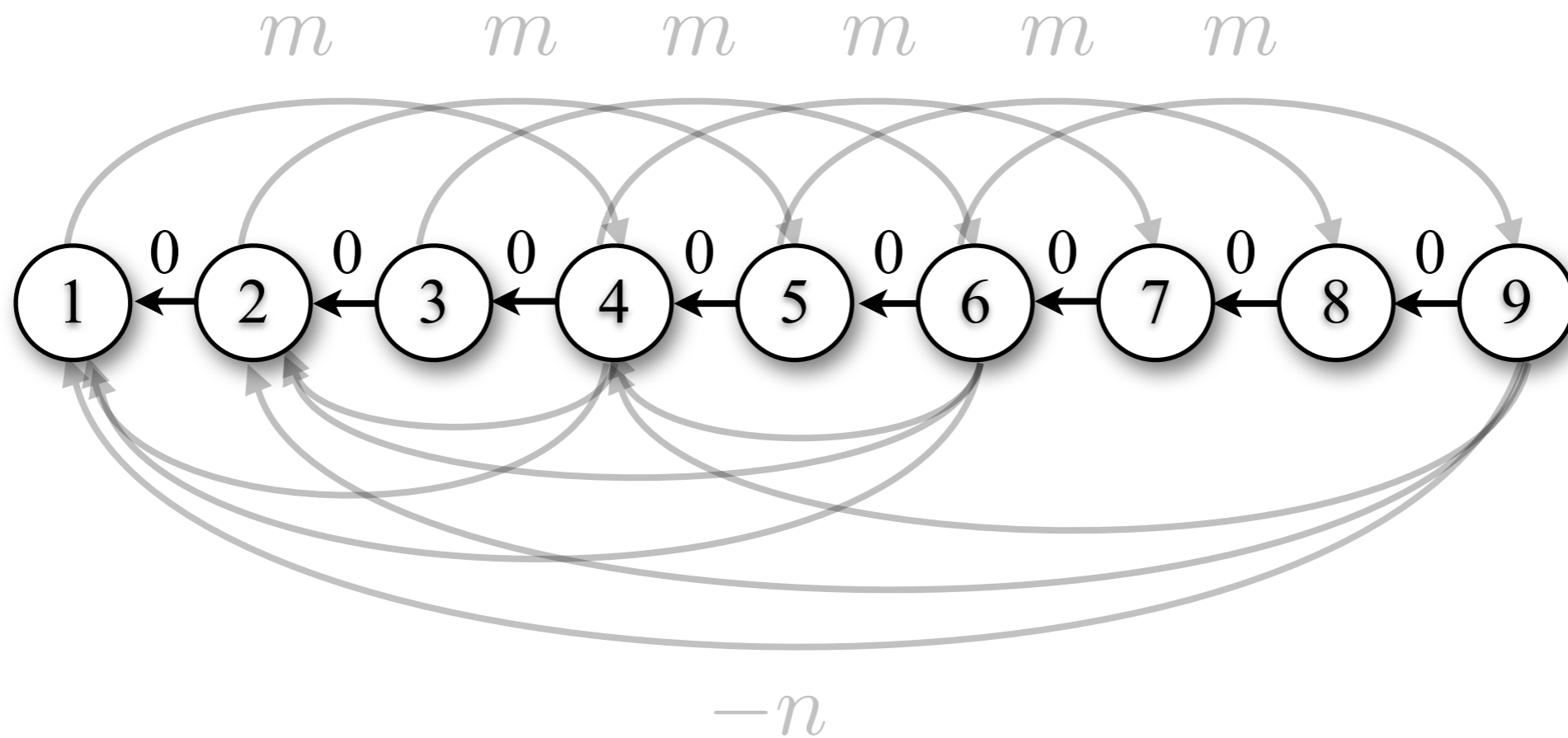
- $d[9] = 0$
- $d[1] = -n$

Vecteur de distance



- $d[9] = 0$
- $d[1] = -n$
- $d[i] \leq d[i+1]$

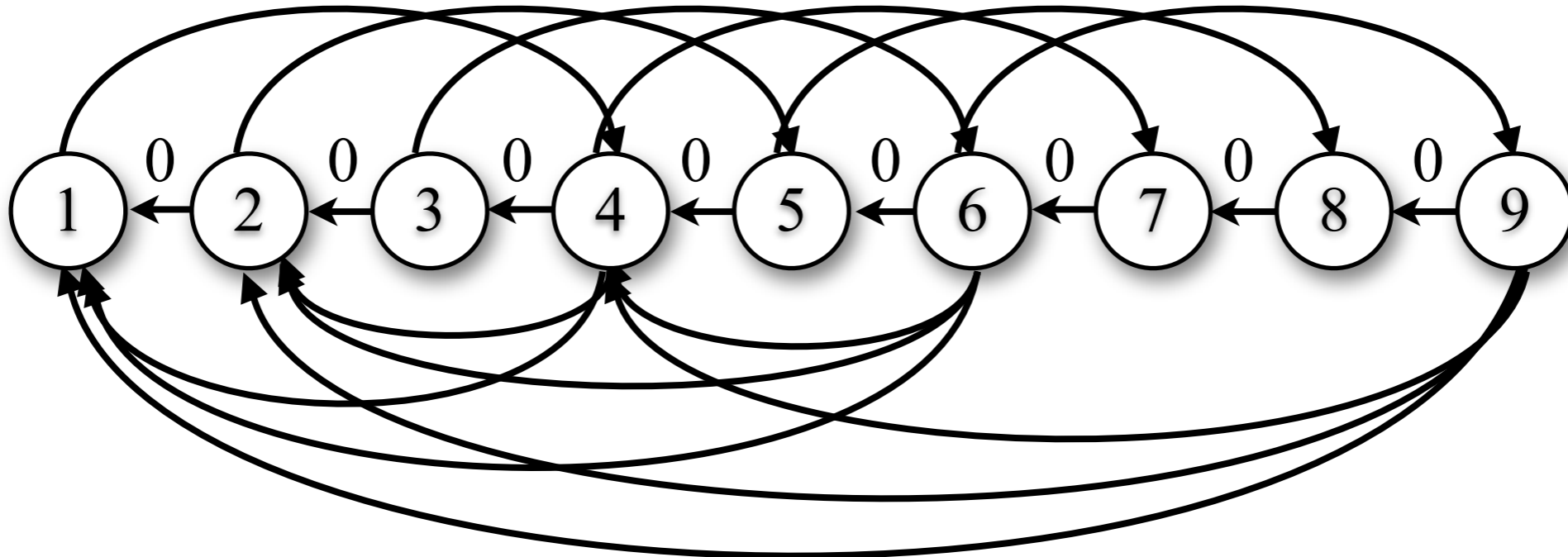
Vecteur de distance



- $d[9] = 0$
- $d[1] = -n$
- $d[i] \leq d[i+1]$
- On peut noter les n noeuds où l'inégalité est stricte.

Arêtes ascendantes

m m m m m m

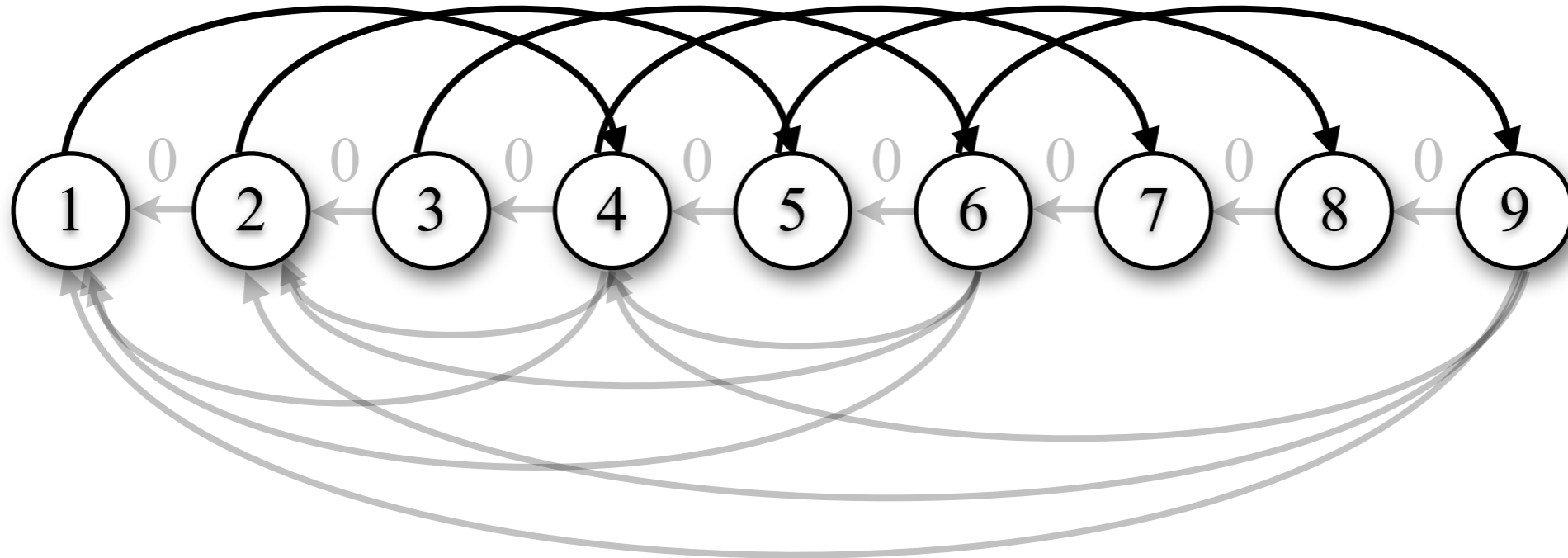


$$-|\{k \mid r_i \leq r_k, u_k \leq u_j\}|$$

- Soit $d'[x]$ le plus grand noeud atteignable à une distance $-x$.

Arêtes ascendantes

m m m m m m

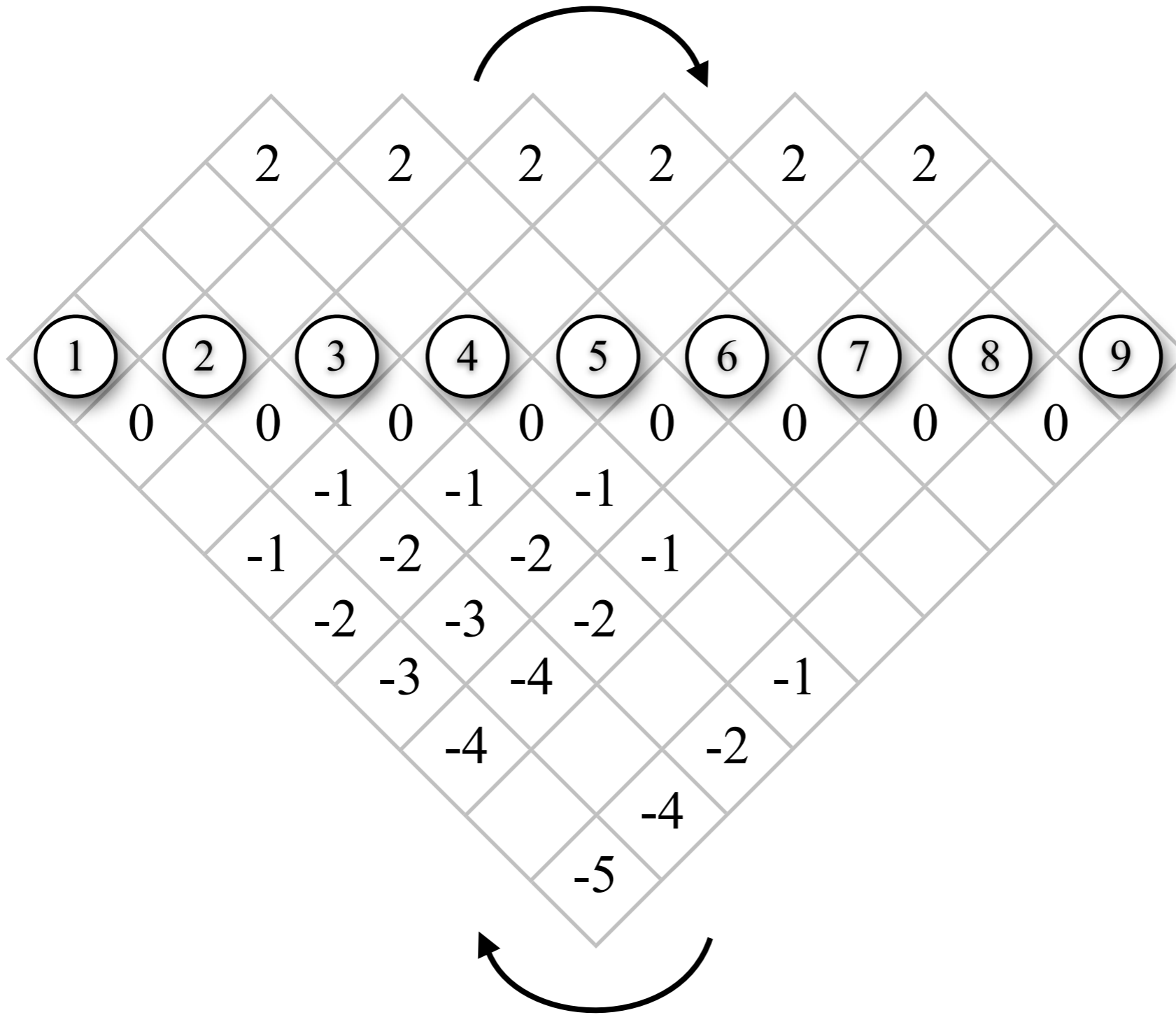


$$-|\{k \mid r_i \leq r_k, u_k \leq u_j\}|$$

- Soit $d'[x]$ le plus grand noeud atteignable à une distance $-x$.
- $d'[i] \leftarrow \min(d'[i], d'[i + m] + p)$

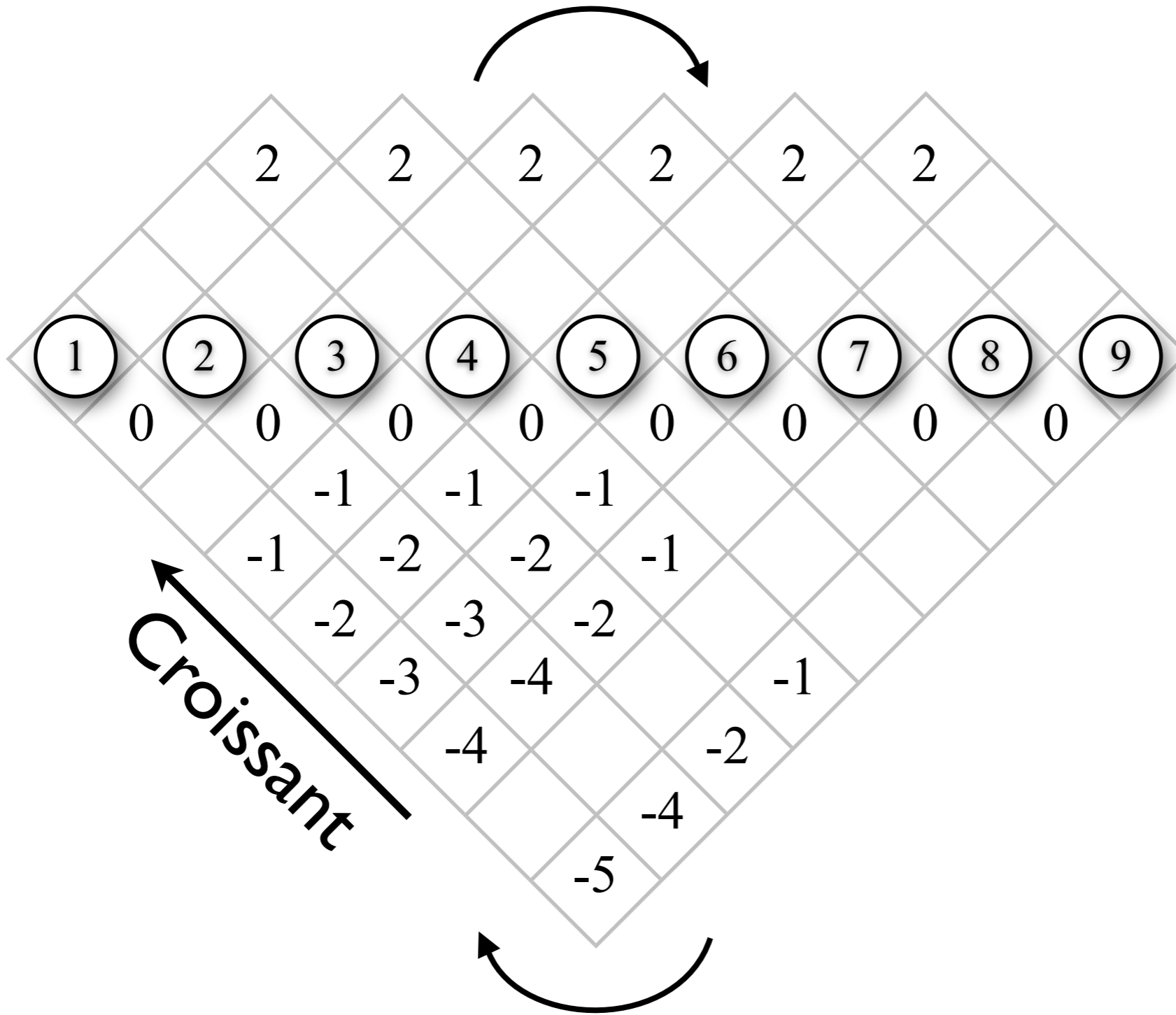
Arêtes descendantes

$s_1 \in [1, 9)$
 $s_2 \in [2, 4)$
 $s_3 \in [2, 7)$
 $s_4 \in [3, 5)$
 $s_5 \in [4, 6)$
 $p = 3$
 $m = 2$



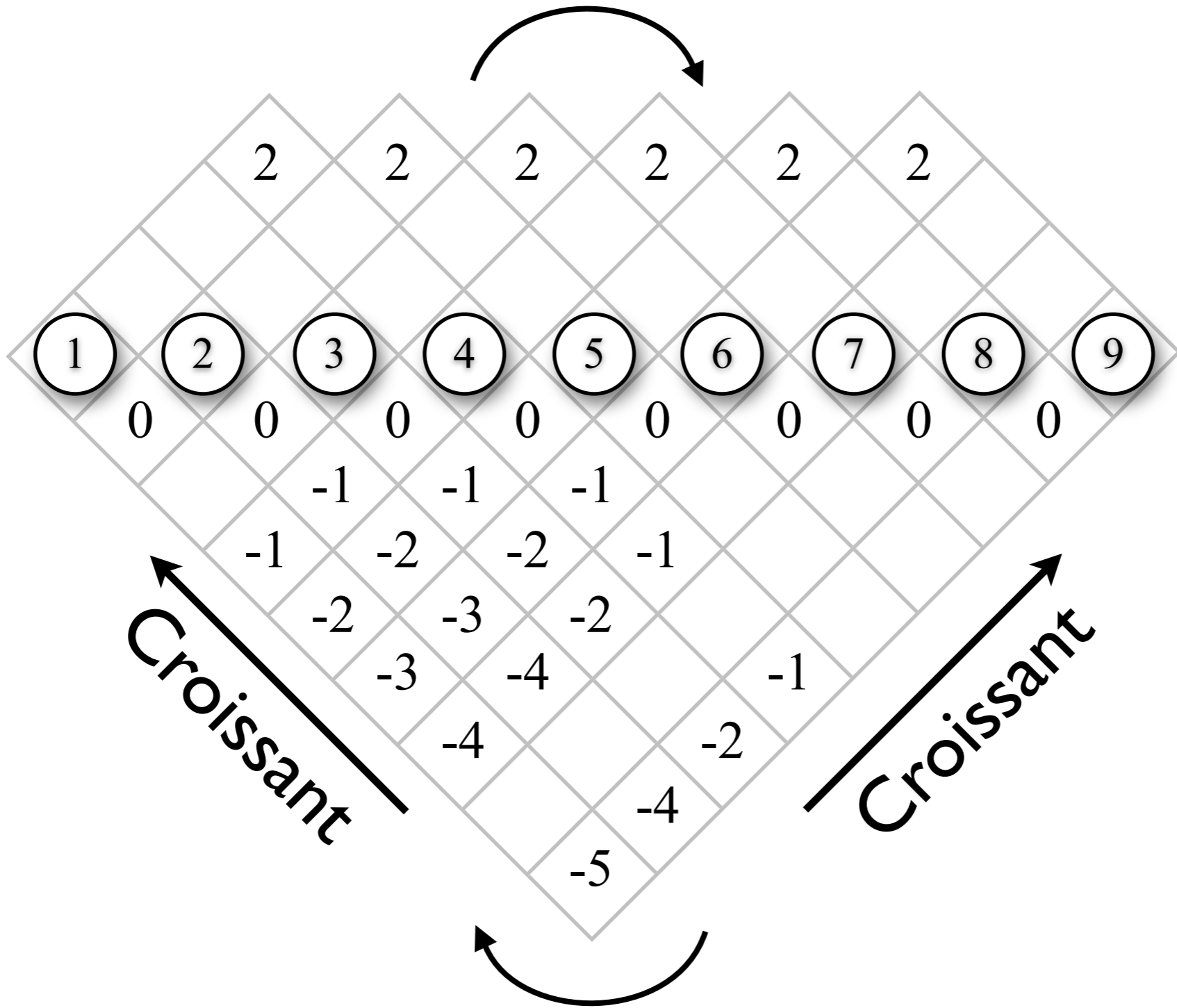
Arêtes descendantes

$s_1 \in [1, 9)$
 $s_2 \in [2, 4)$
 $s_3 \in [2, 7)$
 $s_4 \in [3, 5)$
 $s_5 \in [4, 6)$
 $p = 3$
 $m = 2$



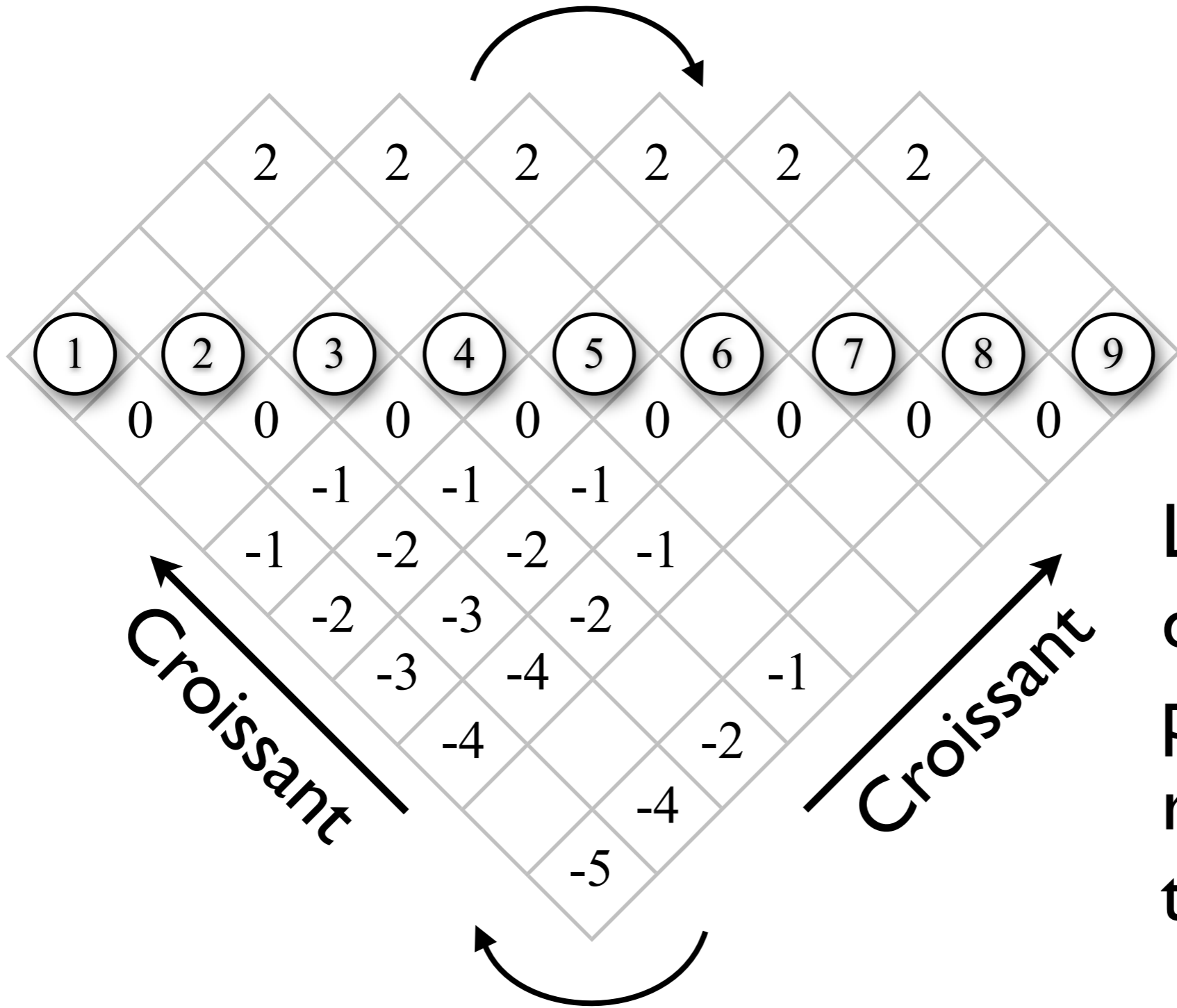
Arêtes descendantes

$s_1 \in [1, 9)$
 $s_2 \in [2, 4)$
 $s_3 \in [2, 7)$
 $s_4 \in [3, 5)$
 $s_5 \in [4, 6)$
 $p = 3$
 $m = 2$



Arêtes descendantes

$s_1 \in [1, 9)$
 $s_2 \in [2, 4)$
 $s_3 \in [2, 7)$
 $s_4 \in [3, 5)$
 $s_5 \in [4, 6)$
 $p = 3$
 $m = 2$



Les arrêtes descendantes peuvent être relaxées en temps $O(n)$

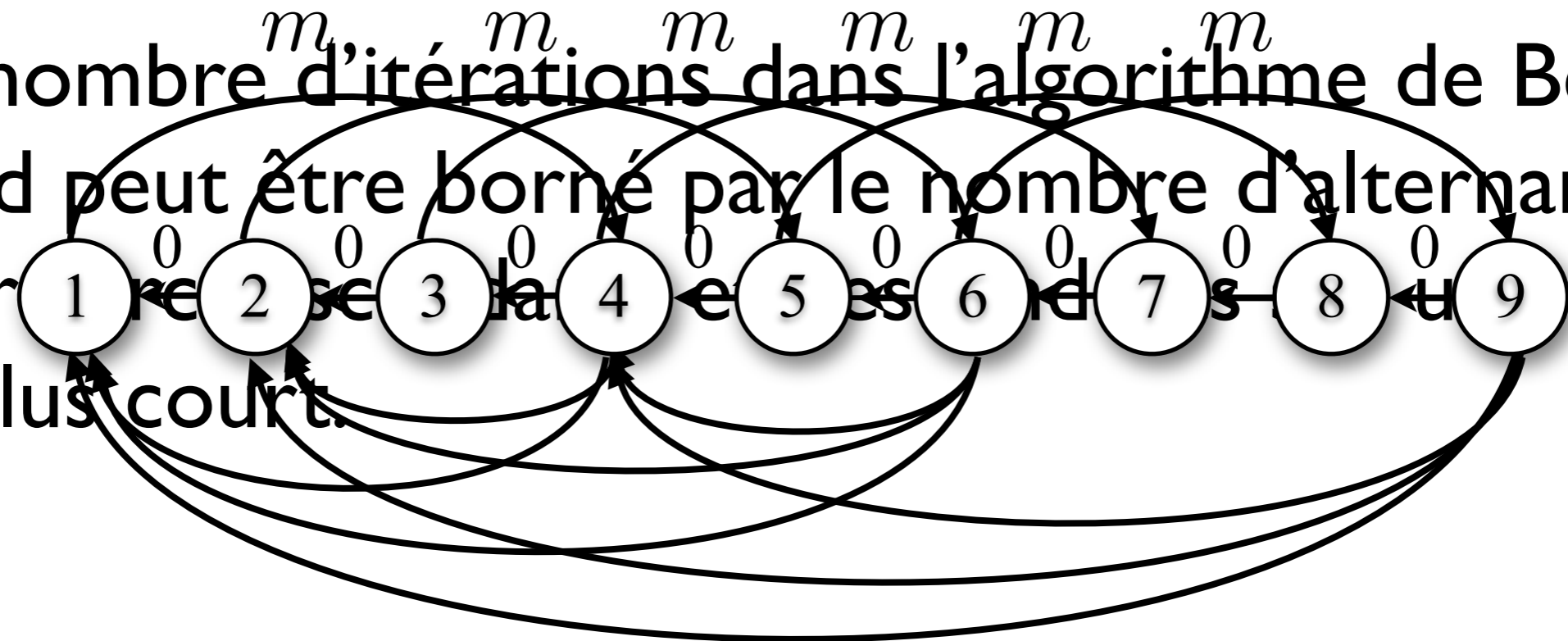
Algorithme de Bellman-Ford

- $d \leftarrow [\infty, \infty, \dots, \infty]$
- $d[\text{noeud d'origine}] \leftarrow 0$
- Répète $|V|$ fois
 - Pour toutes les arêtes (a, b) $O(n) + O(n) = O(n)$
 - Si $d[a] + w(a, b) < d[b]$ alors
 - $d[b] \leftarrow d[a] + w(a, b)$

Ascendantes
Descendantes

L'amélioration de Yen

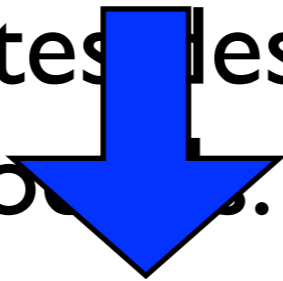
- On traite les arêtes ascendantes en ordre croissant de noeuds.
- On traite les arêtes descendantes en ordre décroissant de noeuds.
- Le nombre d'itérations dans l'algorithme de Bellman-Ford peut être borné par le nombre d'alternances entrées/sorties le plus court.



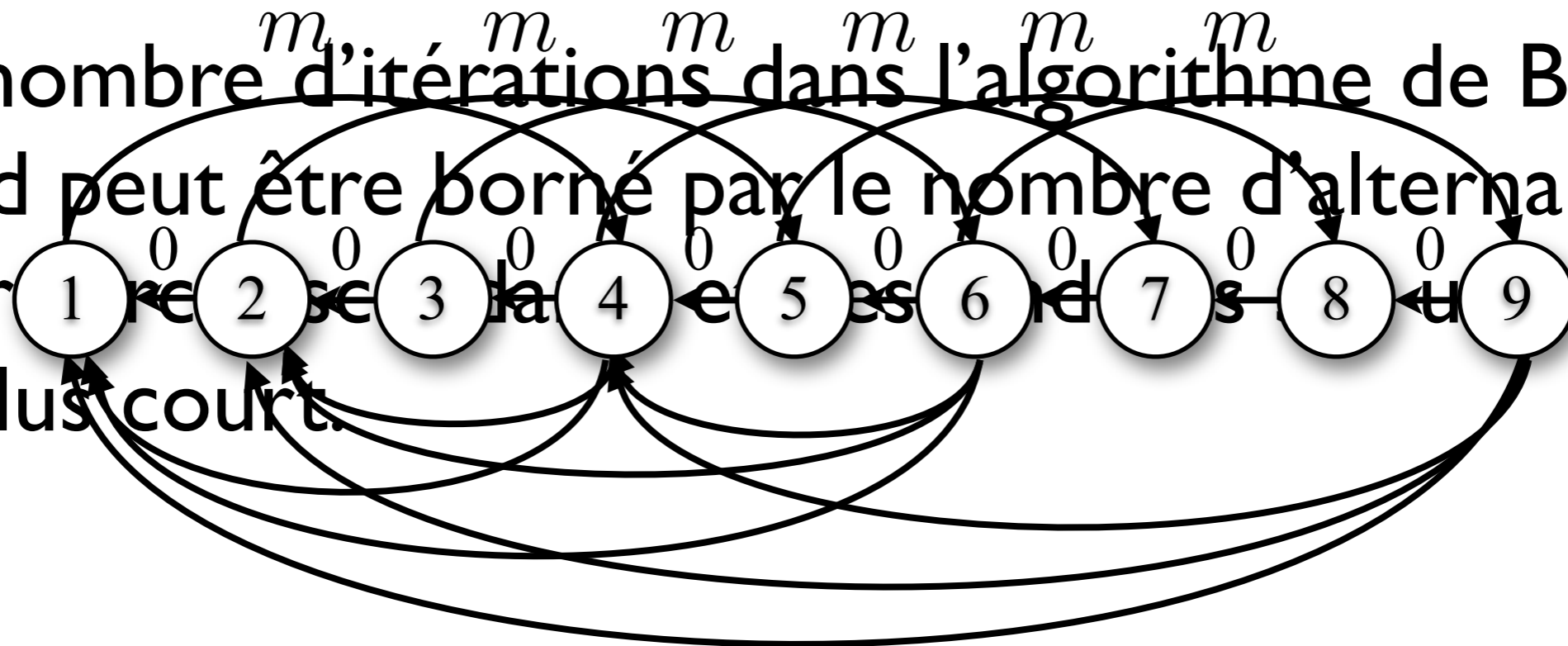
$$- |\{k \mid r_i \leq r_k, u_k \leq u_j\}|$$

L'amélioration de Yen

- On traite les arêtes ascendantes en ordre croissant de noeuds.
- On traite les arêtes descendantes en ordre décroissant de noeuds.



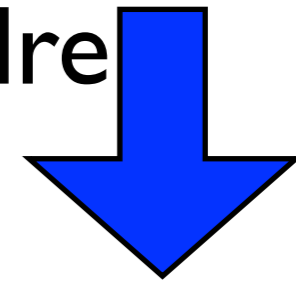
- Le nombre d'itérations dans l'algorithme de Bellman-Ford peut être borné par le nombre d'alternances entrées/sorties le plus court.



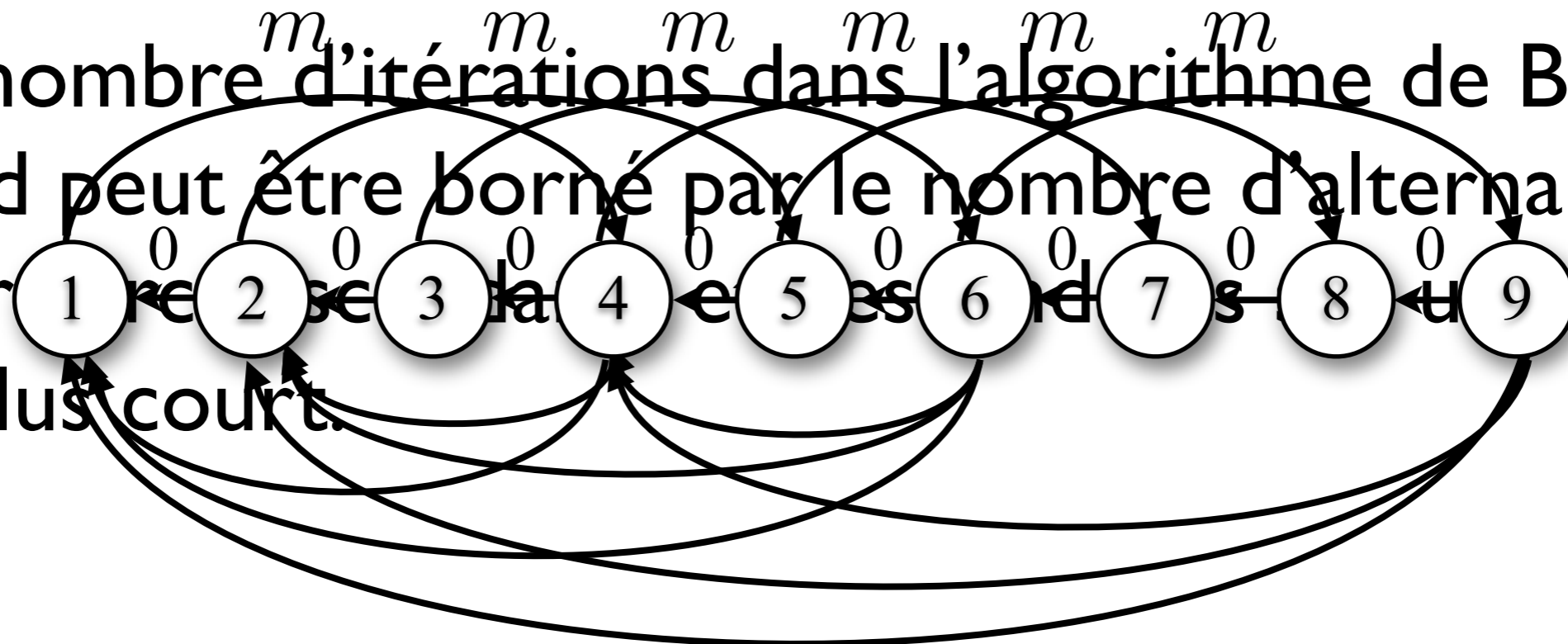
$$- |\{k \mid r_i \leq r_k, u_k \leq u_j\}|$$

L'amélioration de Yen

- On traite les arêtes ascendantes en ordre croissant de noeuds.
- On traite les arêtes descendantes en ordre décroissant de noeuds.



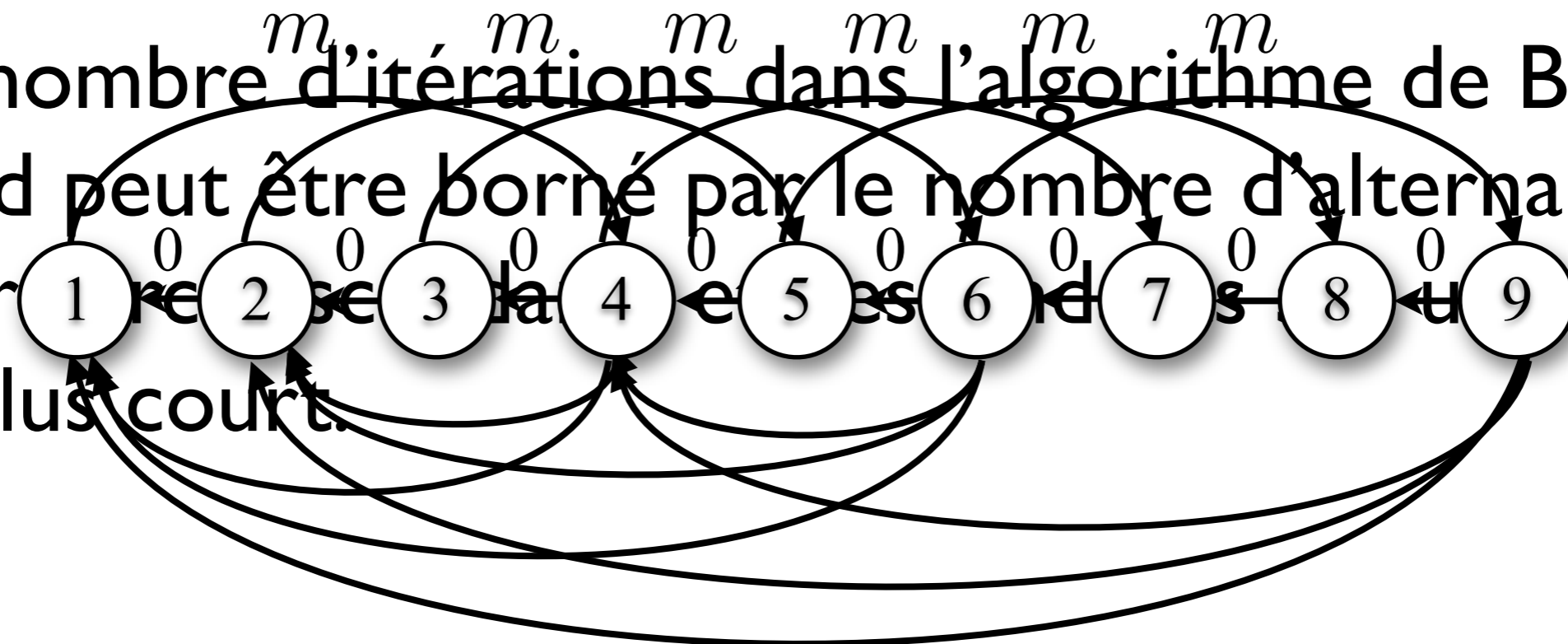
- Le nombre d'itérations dans l'algorithme de Bellman-Ford peut être borné par le nombre d'alternances entrées/sorties le plus court.



$$- |\{k \mid r_i \leq r_k, u_k \leq u_j\}|$$

L'amélioration de Yen

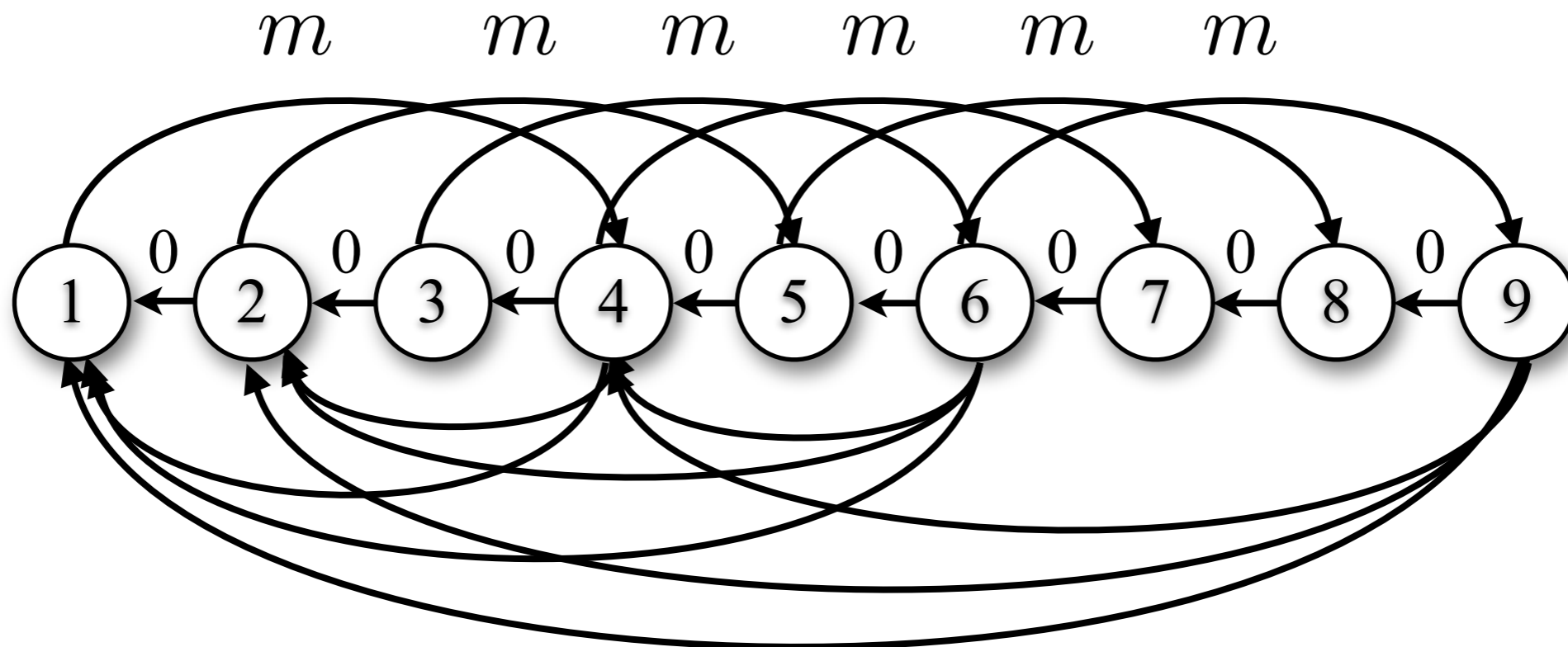
- On traite les arêtes ascendantes en ordre croissant de noeuds.
- On traite les arêtes descendantes en ordre décroissant de noeuds.
- Le nombre d'itérations dans l'algorithme de Bellman-Ford peut être borné par le nombre d'alternances entrées/sorties le plus court.



$$- |\{k \mid r_i \leq r_k, u_k \leq u_j\}|$$

L'amélioration de Yen

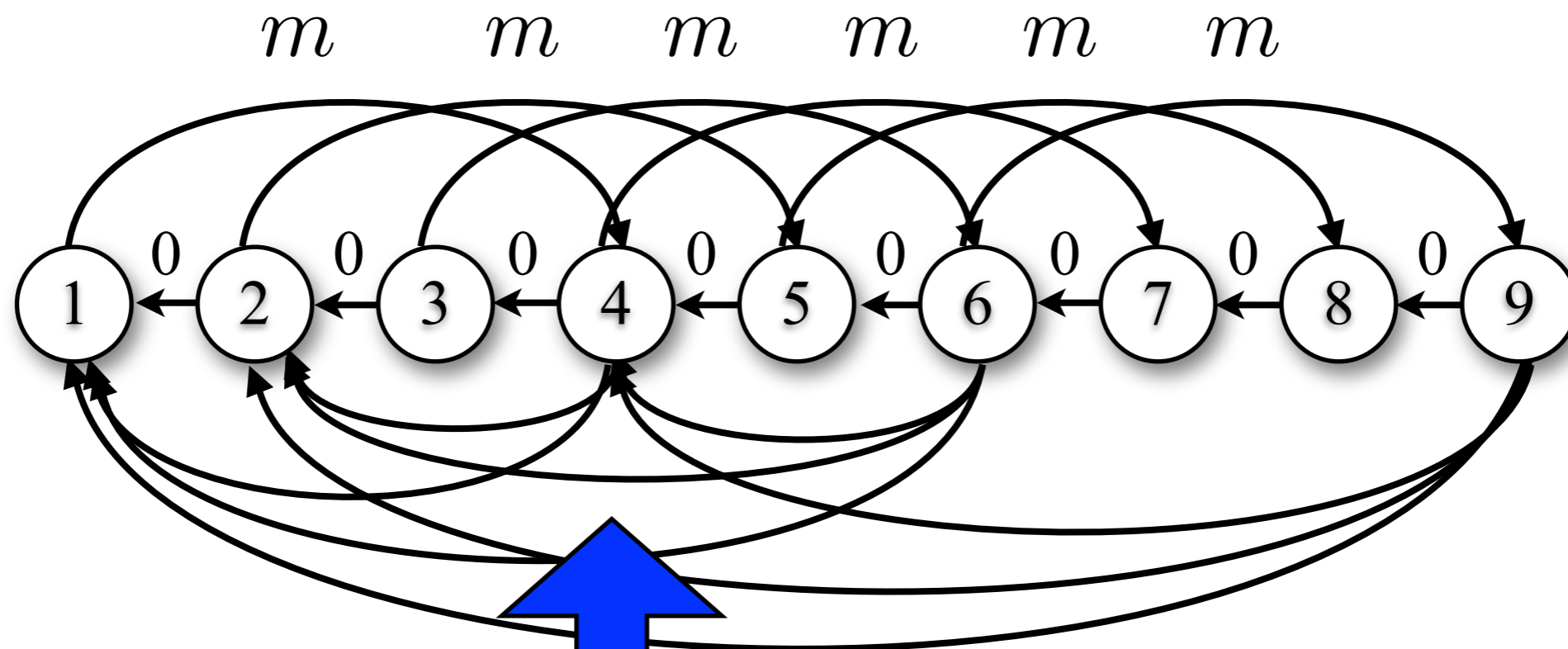
- On traite les arêtes ascendantes en ordre croissant de noeuds.
- On traite les arêtes descendantes en ordre décroissant de noeuds.



$$- |\{k \mid r_i \leq r_k, u_k \leq u_j\}|$$

L'amélioration de Yen

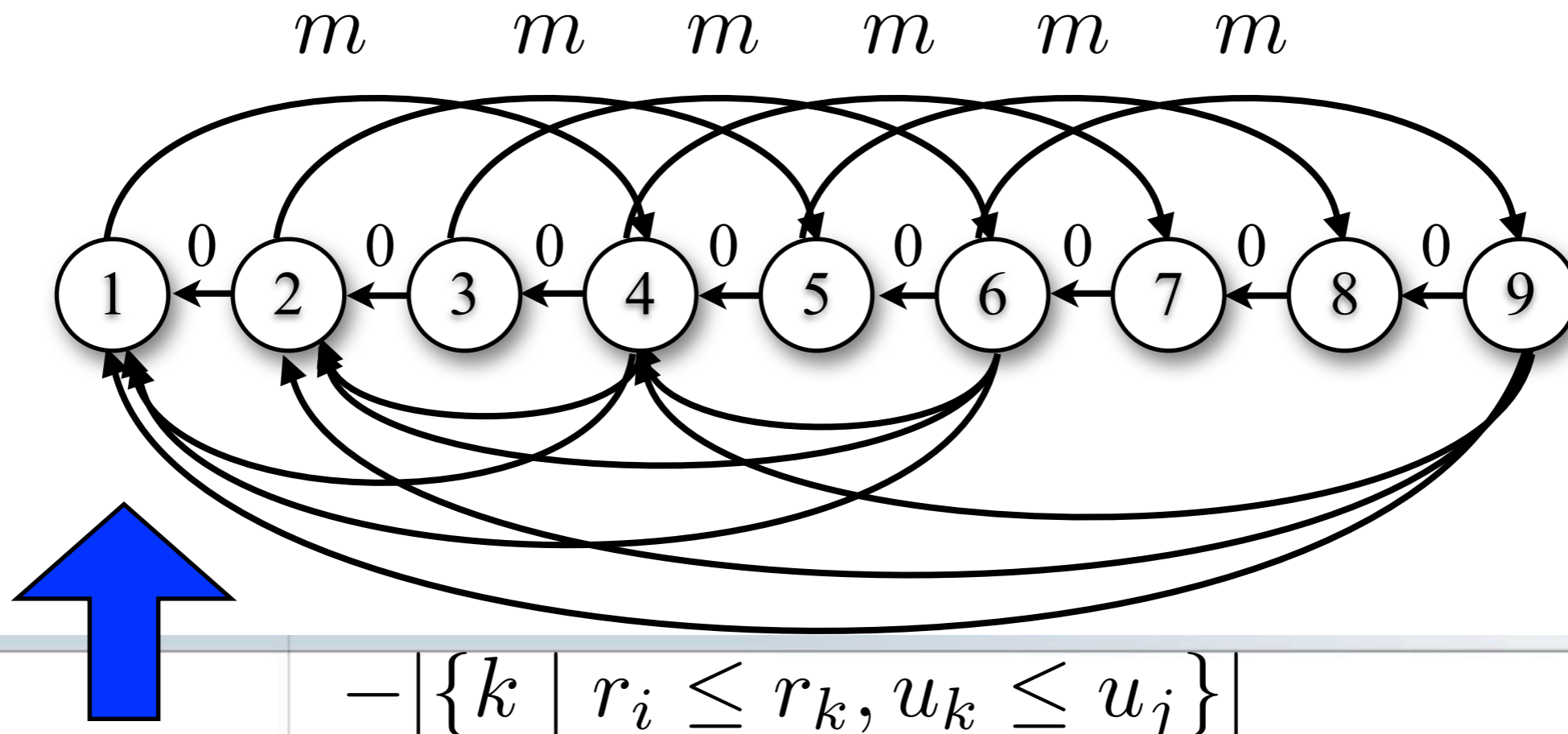
- On traite les arêtes ascendantes en ordre croissant de noeuds.
- On traite les arêtes descendantes en ordre décroissant de noeuds.



$$-|\{k \mid r_k \leq r_j, u_k \leq u_j\}|$$

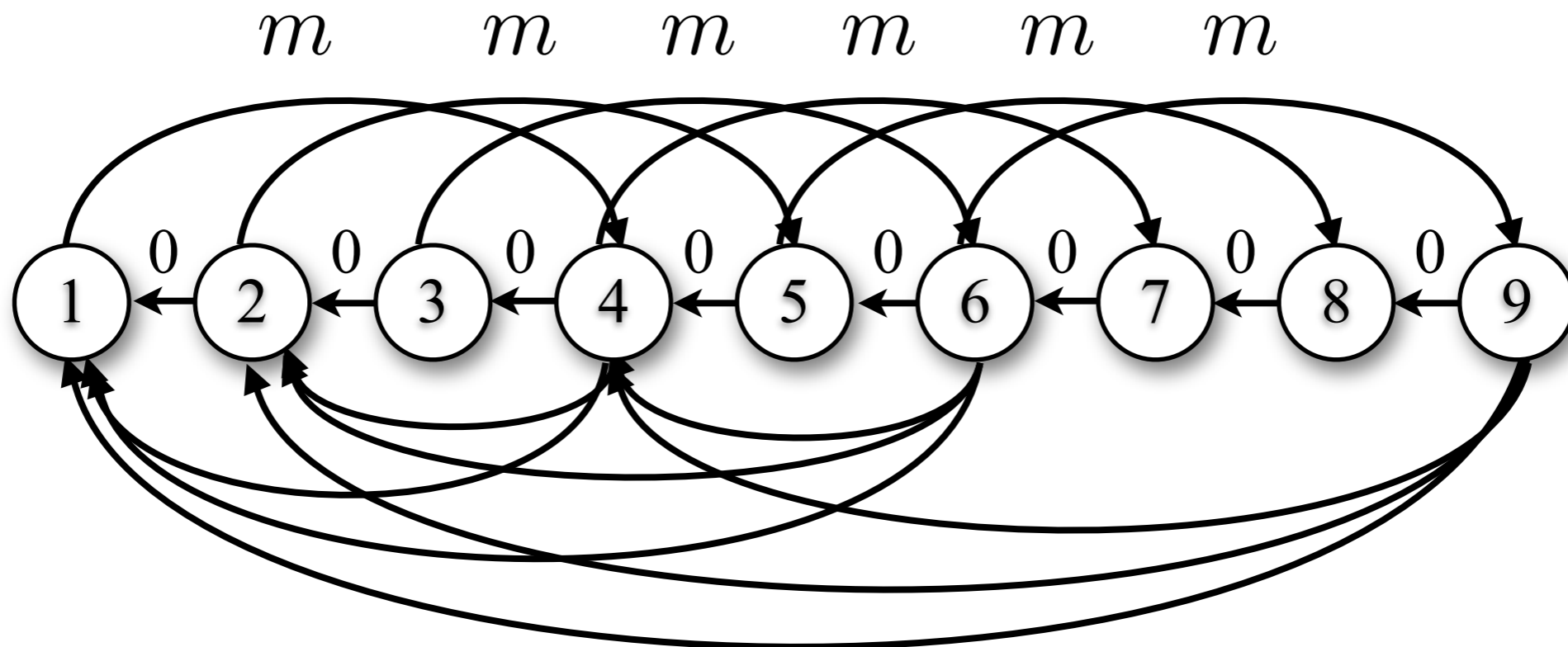
L'amélioration de Yen

- On traite les arêtes ascendantes en ordre croissant de noeuds.
- On traite les arêtes descendantes en ordre décroissant de noeuds.



L'amélioration de Yen

- On traite les arêtes ascendantes en ordre croissant de noeuds.
- On traite les arêtes descendantes en ordre décroissant de noeuds.



$$-|\{k \mid r_i \leq r_k, u_k \leq u_j\}|$$

L'amélioration de Yen

- On traite les arêtes ascendantes en ordre croissant de noeuds.
- On traite les arêtes descendantes en ordre décroissant de noeuds.
- Le nombre d'itérations dans l'algorithme de Bellman-Ford peut être borné par le nombre d'alternances entre arcs ascendants et descendants sur un chemin le plus court.

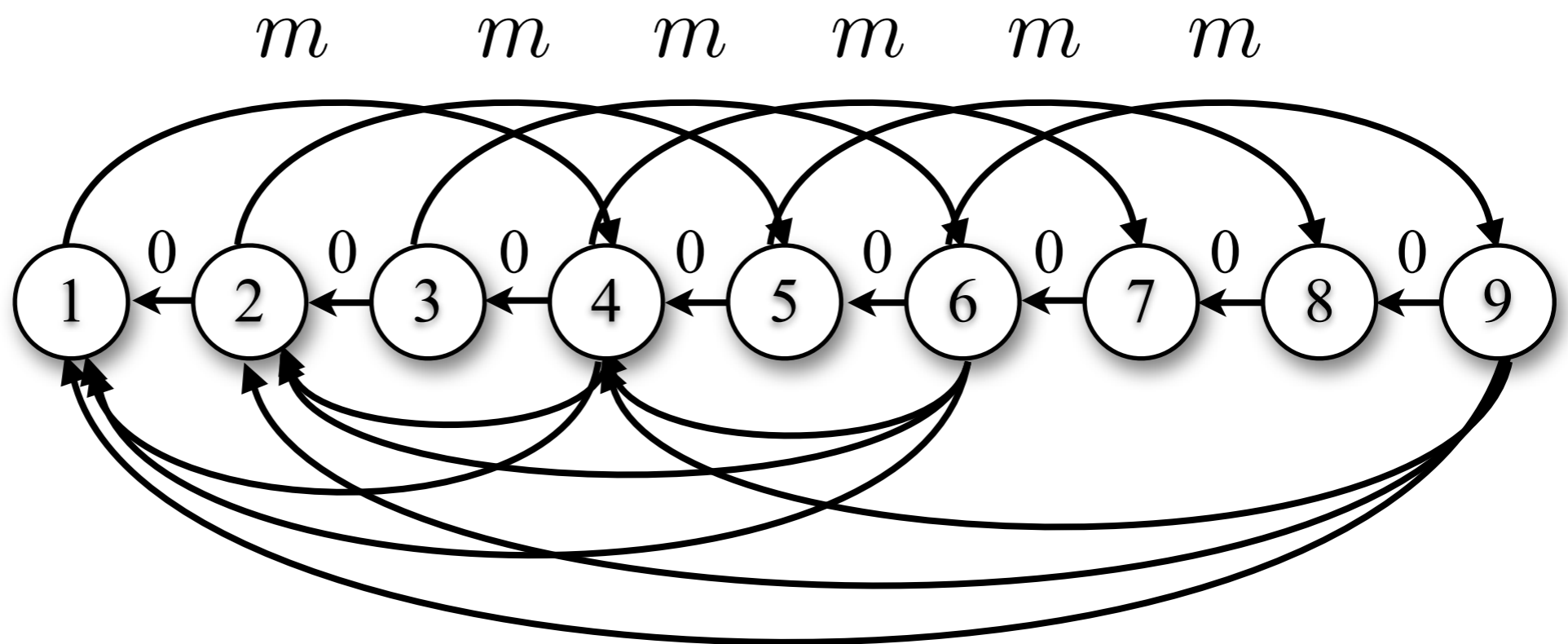
Borne sur les alternances

$$s_1 \in [1, 9)$$

$$s_2 \in [2, 4)$$

$$s_3 \in [4, 6)$$

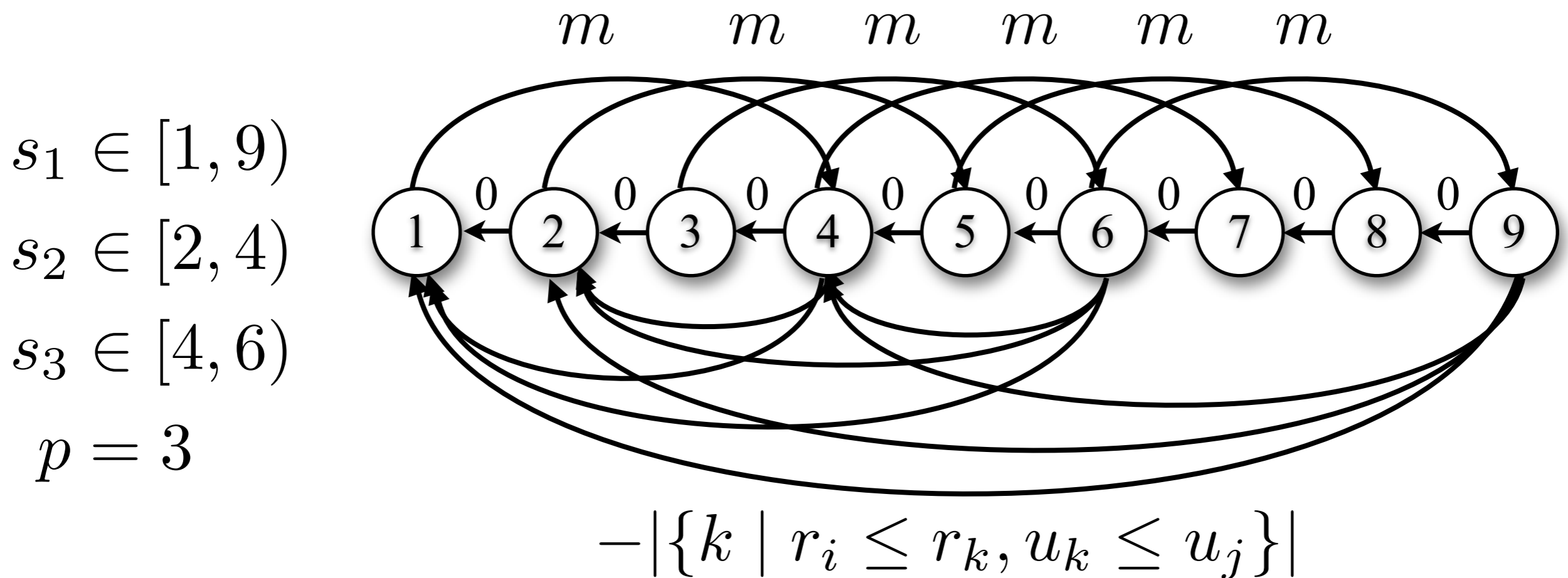
$$p = 3$$



$$- |\{k \mid r_i \leq r_k, u_k \leq u_j\}|$$

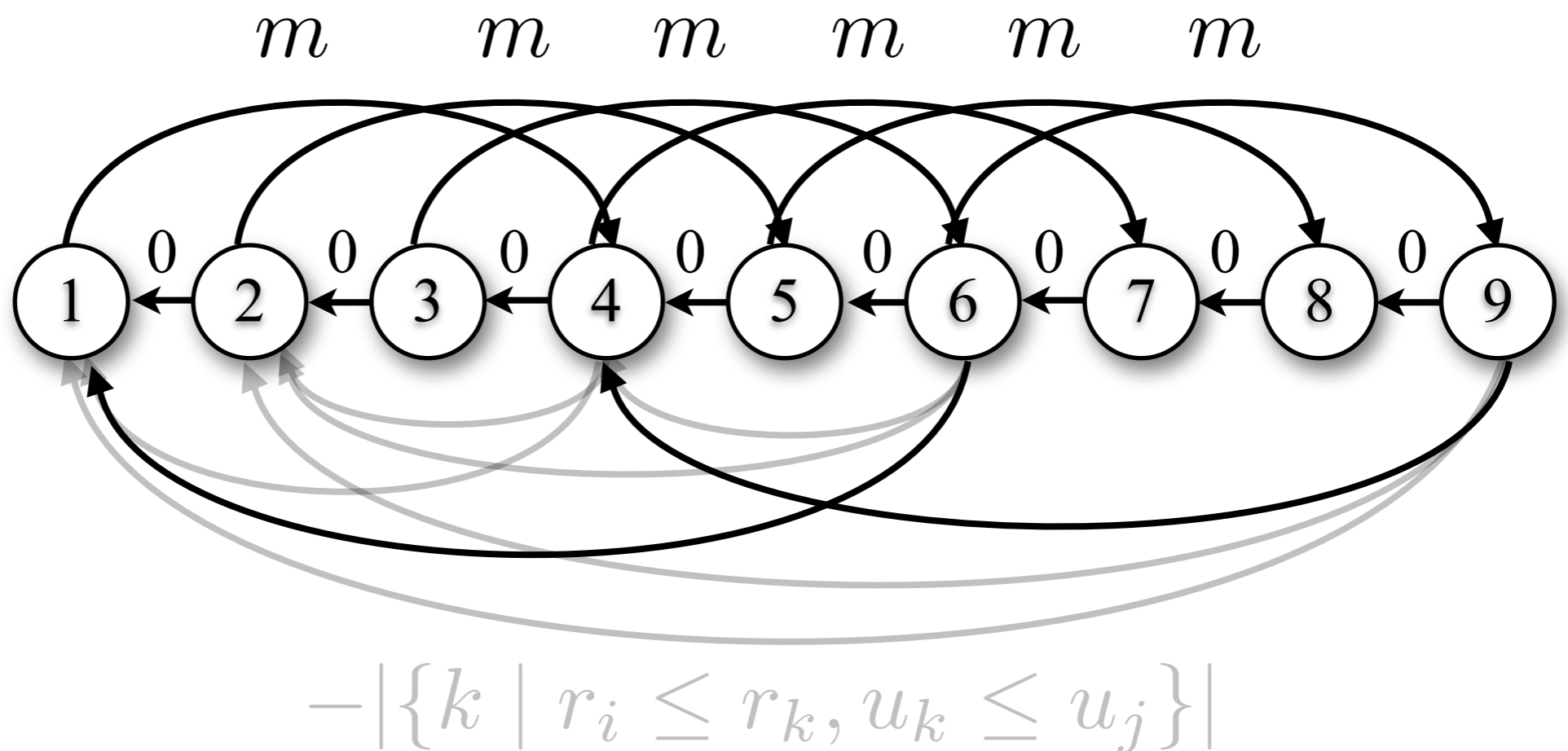
Borne sur les alternances

- Seuls n noeuds ont des arêtes descendantes entrantes



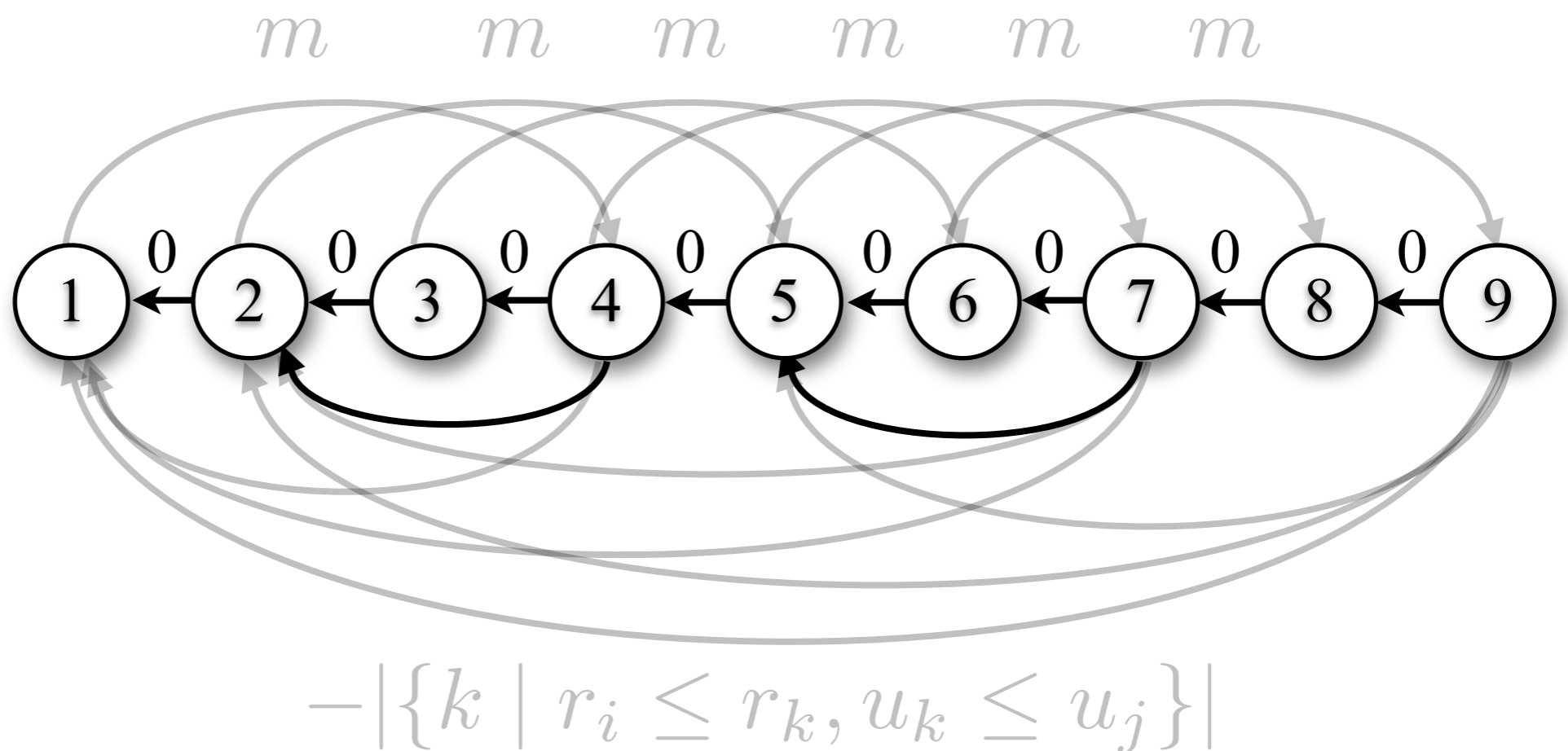
Borne sur les alternances

- Deux arêtes descendantes ne peuvent pas se croiser.



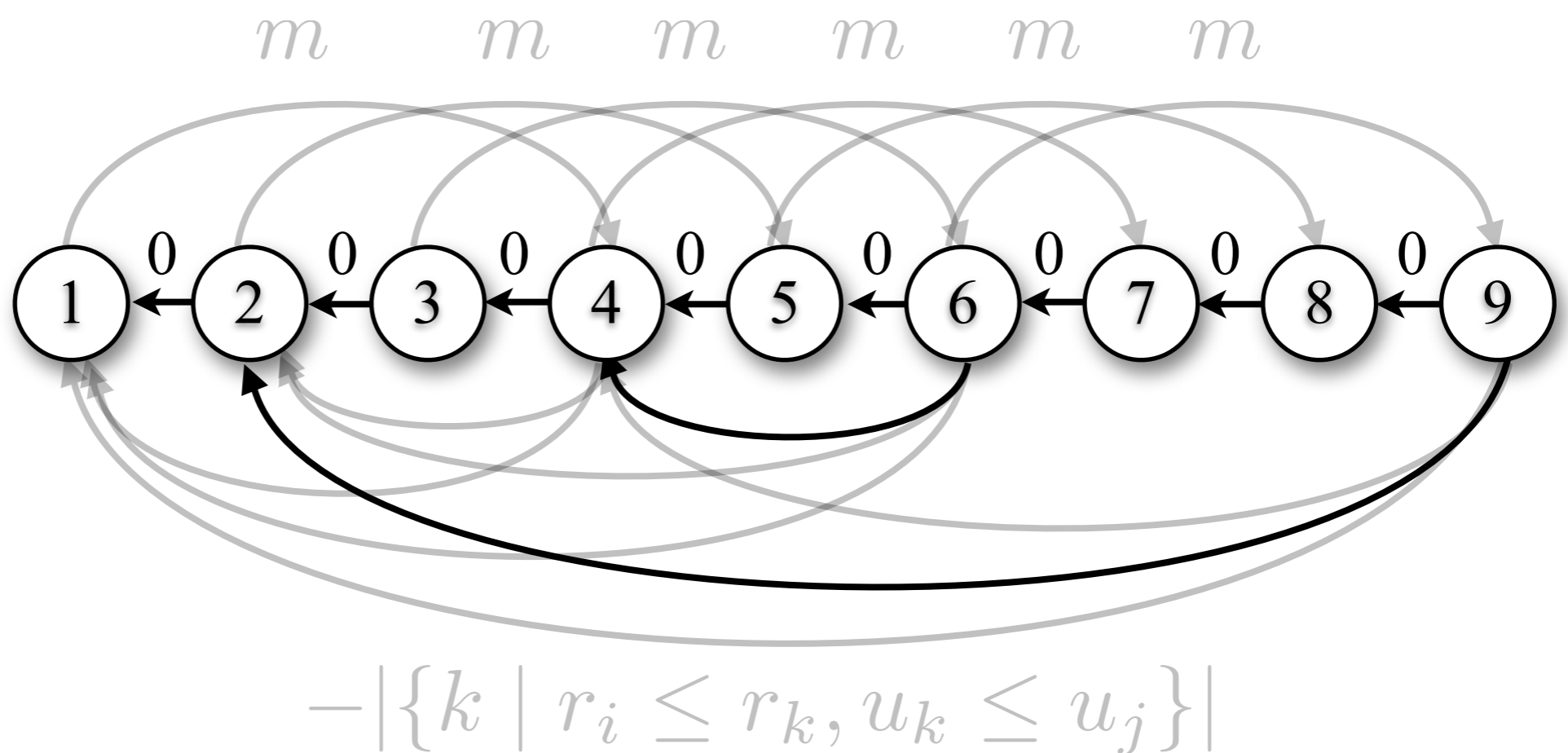
Borne sur les alternances

- Il y a au plus $\frac{n}{m}$ arêtes descendantes disjointes.



Borne sur les alternances

- Il y a au plus p arêtes descendantes imbriquées.



Borne sur les alternances

- Il y a au plus $\frac{n}{m}$ arêtes descendantes disjointes

Borne sur les alternances

- Il y a au plus $\frac{n}{m}$ arêtes descendantes disjointes
- Il y a au plus p arêtes descendantes imbriquées

Borne sur les alternances

- Il y a au plus $\frac{n}{m}$ arêtes descendantes disjointes
- Il y a au plus p arêtes descendantes imbriquées
- Il y a au plus $\frac{np}{m}$ arêtes descendantes

Borne sur les alternances

- Il y a au plus $\frac{n}{m}$ arêtes descendantes disjointes
- Il y a au plus p arêtes descendantes imbriquées
- Il y a au plus $\frac{np}{m}$ arêtes descendantes
- Puisqu'il y a aussi une borne de n arêtes descendantes,

Borne sur les alternances

- Il y a au plus $\frac{n}{m}$ arêtes descendantes disjointes
- Il y a au plus p arêtes descendantes imbriquées
- Il y a au plus $\frac{np}{m}$ arêtes descendantes
- Puisqu'il y a aussi une borne de n arêtes descendantes,
- il y a au plus $\min\left(\frac{np}{m}, n\right)$ alternances

Algorithme de Bellman-Ford

- $d \leftarrow [\infty, \infty, \dots, \infty]$
- $d[\text{noeud d'origine}] \leftarrow 0$
- Répète $\min\left(\frac{np}{m}, n\right)$ fois
 - Pour tous les arêtes (a, b)
 - Si $d[a] + w(a, b) < d[b]$ alors
 - $d[b] \leftarrow d[a] + w(a, b)$

Ascendantes
Descendantes

$$O(n) + O(n) = O(n)$$

Algorithme de Bellman-Ford

- $d \leftarrow [\infty, \infty, \dots, \infty]$
- $d[\text{noeud d'origine}] \leftarrow 0$
- Répète $\min\left(\frac{np}{m}, n\right)$ fois
 - Pour tous les arêtes (a, b)
 - Si $d[a] + w(a, b) < d[b]$ alors
 - $d[b] \leftarrow d[a] + w(a, b)$

Ascendantes
Descendantes

$$O(n) + O(n) = O(n)$$

Complexité totale

$$O\left(\min\left(1, \frac{p}{m}\right)n^2\right)$$

Objectif

- L'algorithme minimise la somme des temps de début d'exécution.

$$\min \sum_{i=1}^n s_i$$

Objectif

- L'algorithme minimise la somme des temps de début d'exécution.

$$\min \sum_{i=1}^n s_i$$

- L'algorithme minimise le makespan

Travaux à venir

- Obtenir ou prouver une borne stricte sur le nombre d'alternances sur un chemin le plus court.

Travaux à venir

- Obtenir ou prouver une borne stricte sur le nombre d'alternances sur un chemin le plus court.
- Utiliser l'algorithme pour résoudre la relaxation d'un problème d'ordonnancement NP-Difficile.

Travaux à venir

- Obtenir ou prouver une borne stricte sur le nombre d'alternances sur un chemin le plus court.
- Utiliser l'algorithme pour résoudre la relaxation d'un problème d'ordonnancement NP-Difficile.
- Obtenir un propagateur pour une contrainte globale.