

Scheduling Complete Trees on two uniform processors with any integer speed ratios and communication delays

Étude d'article
Université du Havre
DEA-ITA 2003/2004
Robert Fisch
7 avril 2004

Fiche de lecture

- Scheduling Complete Trees on two uniform processors with any integer speed ratios and communication delays
- Parallel Processing Letter, 25 février 1998
- <http://citeseer.nj.nec.com/muth93semantic.html>
- Auteurs
 - J. Blazewicz
 - F. Guinand
 - B. Penz
 - D. Trystram



Plan

- Présentation de l'algorithme de l'article
- Présentation du problème posé
 - Propriétés avec preuves
 - Remarques
- À la recherche d'un algorithme...
 - Différentes méthodes de choix
 - Analyse de sensibilité

Contexte

- Tâches unitaires
- Délais de communications unitaires
- Arbres d'arité k du type «InTree»
- Deux processeurs tel que $V(\mu P - rapide) = 1$ et $V(\mu P - lent) = a \mid a \in N, a \geq 2$
- Pas de tâche préemptives
- Chevauchement des communications



Notation

- «h» c'est la hauteur de l'arbre considéré
- «k» c'est l'arité de chaque nœud
- «n» est le nombre total de noeuds.

Calculs

- Temps minimal d'exécution: $LB = \left\lfloor \frac{(n-1) \cdot a}{1+a} \right\rfloor + 2$
- Tâches allouées sur le processeur rapide:

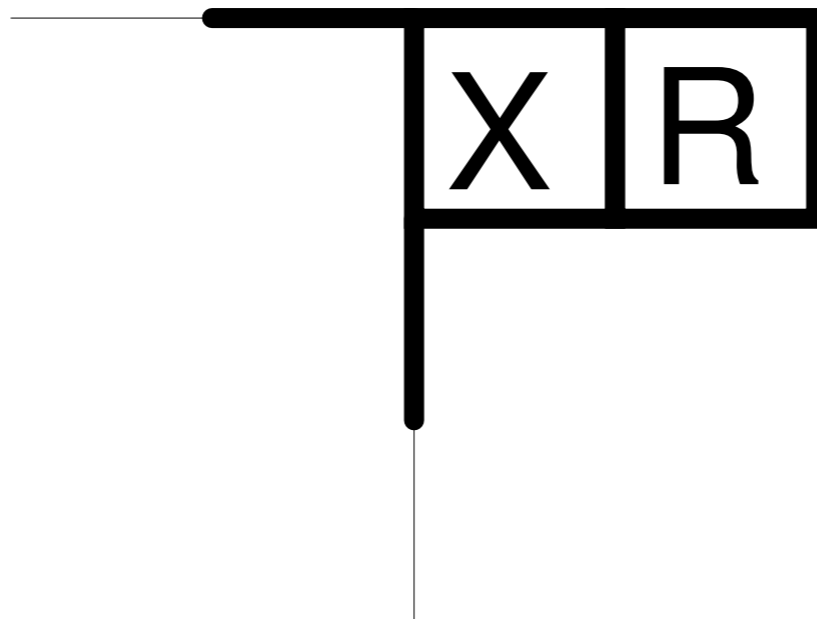
$$N_f = \left\lfloor \frac{(n-1) \cdot a}{1+a} \right\rfloor + 1$$

- Tâches allouées sur le processeur lent:

$$N_s = \left\lceil \frac{n-1}{1+a} \right\rceil - 1 = n - 1 - N_f$$

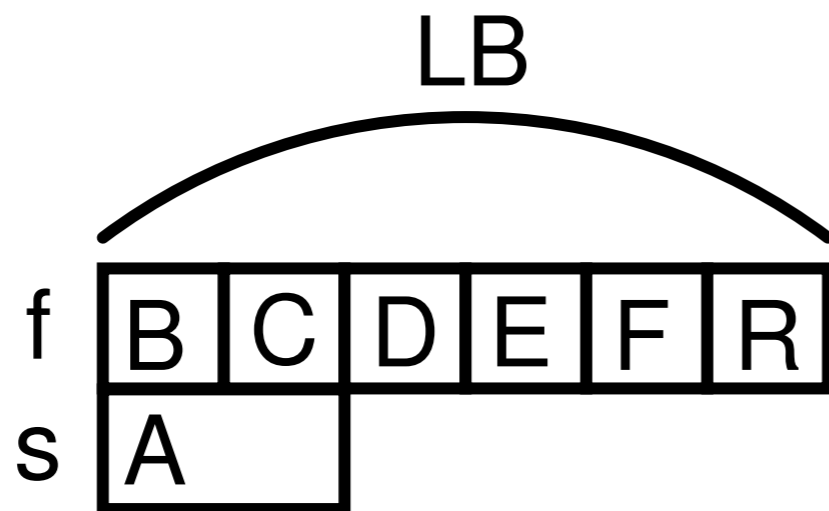
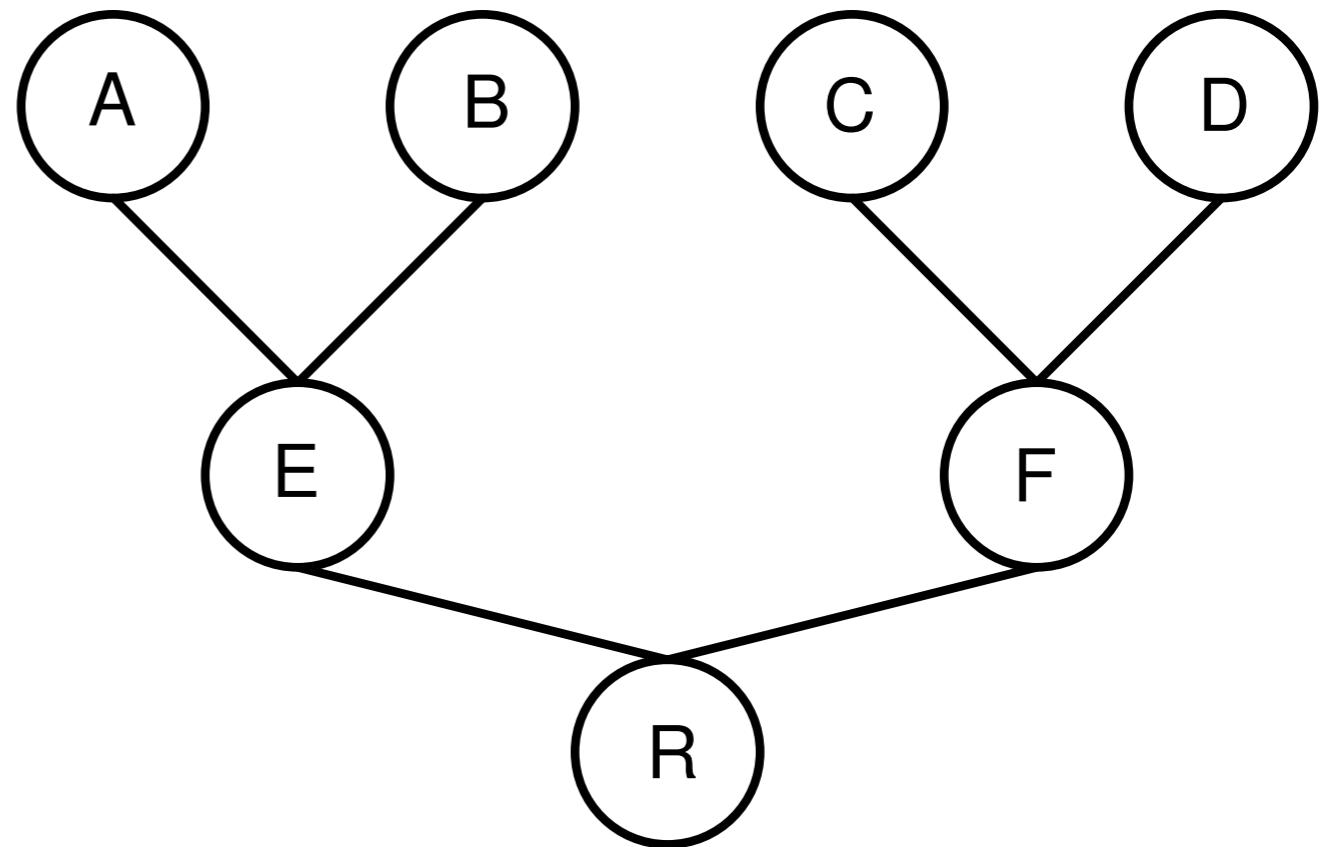
Remarque #1

- L'ordonnancement optimal est toujours tel que la racine s'exécute sur le processeur le plus rapide.



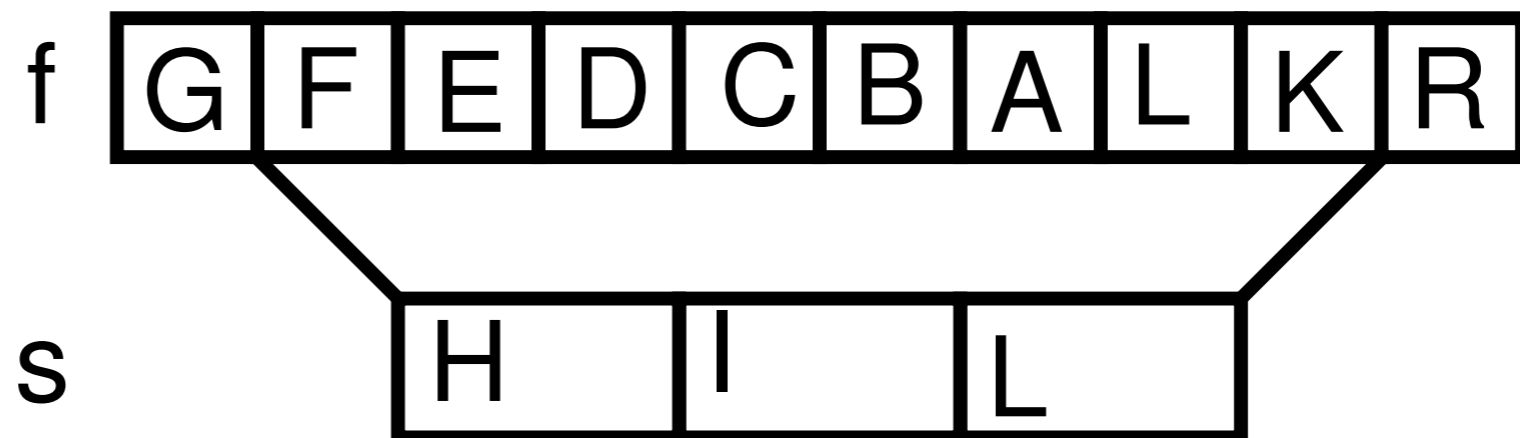
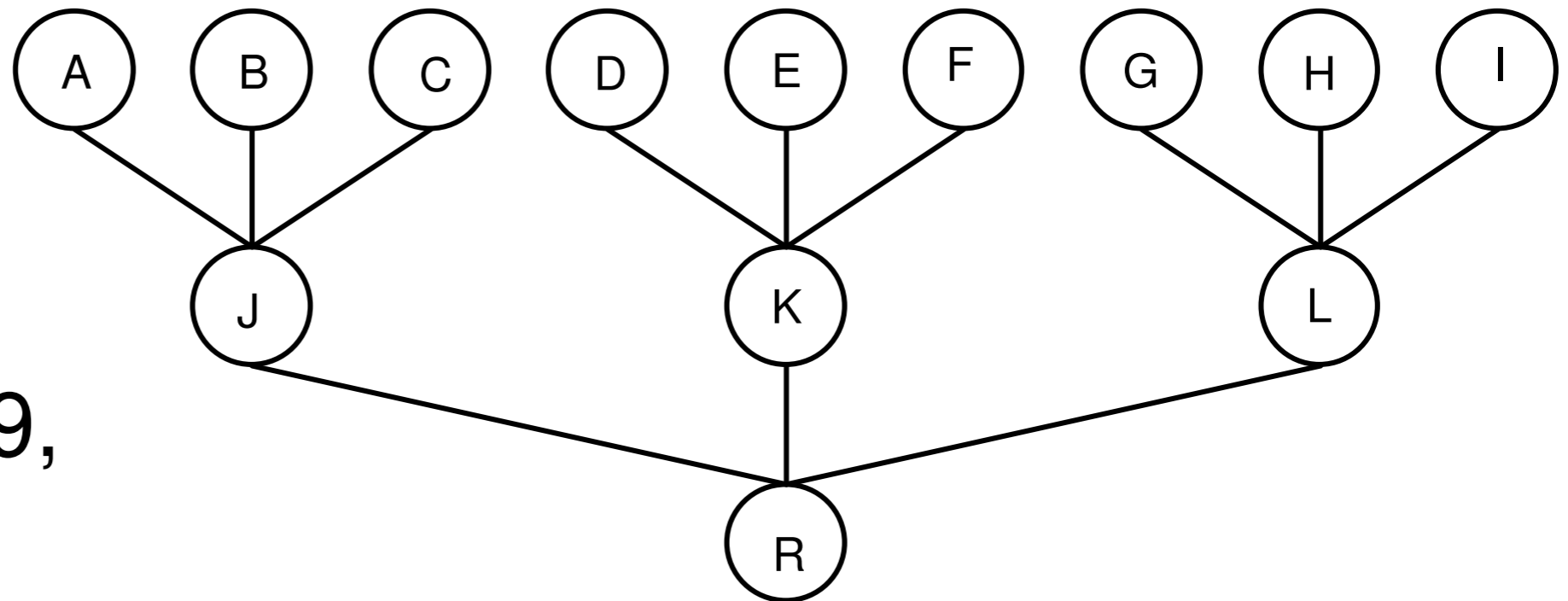
Exemple #1

- $a=2, n=7$
 $h=3, k=2$
- $LB=6, N_f=5, N_s=1$
- $A+h=5<6$



Exemple #2

- $a=2$, $n=13$,
 $h=3$, $k=3$
- $LB=10$, $N_f=9$,
 $N_s=3$
- $h-1=2 < N_s$

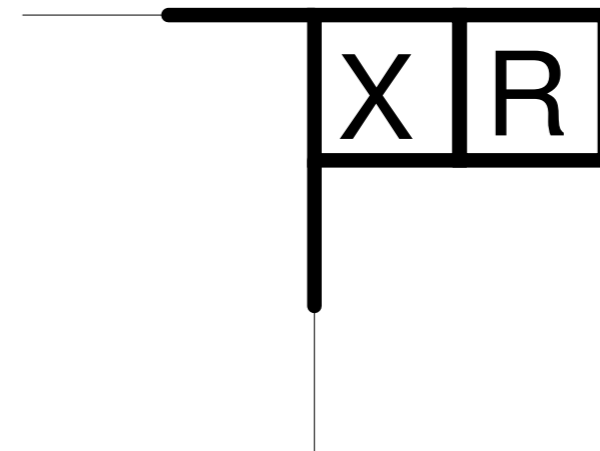


Problème posé

- m machines, partagées en deux groupes de machines identiques
 - x machines à vitesse 1
 - y machines à vitesse a
- Comment peut-on ordonnancer un arbre complet sur une telle architecture ?
- Peut-on qualifier la qualité de l'ordonnancement obtenu par rapport à un ordonnancement optimal ?

Propriété #1

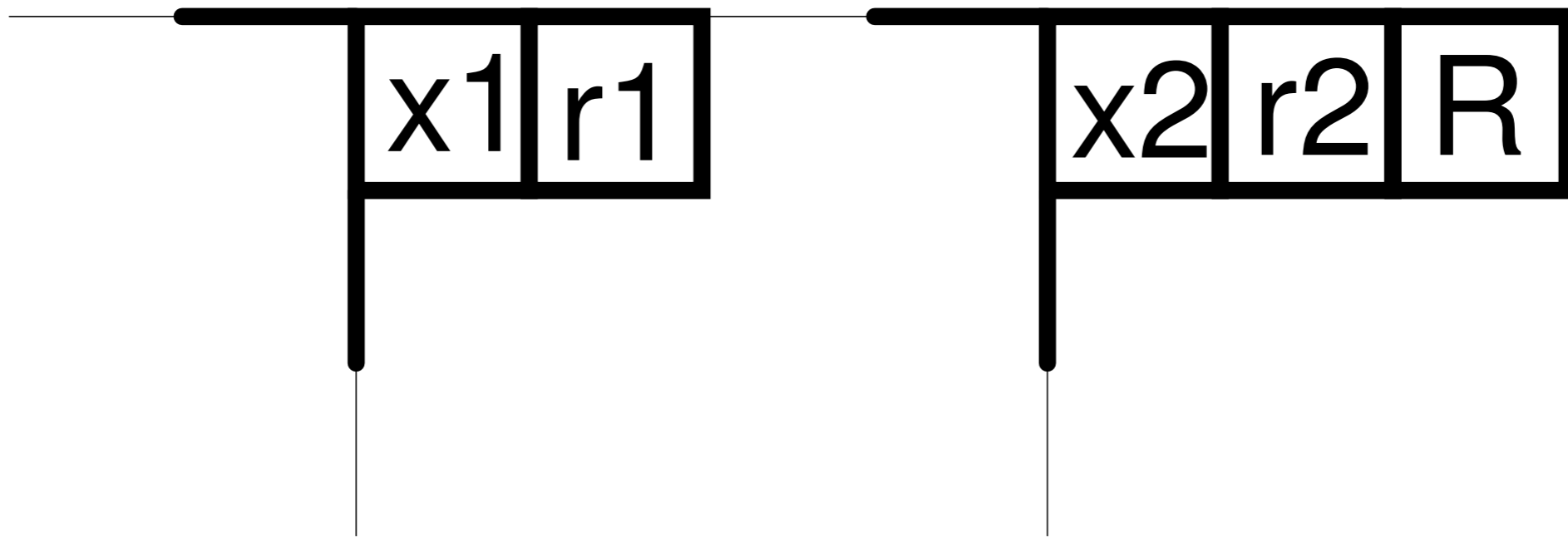
- Peu importe la valeur de k , on a toujours un ordonnancement du type:



- Si $O(k,h) = t$, alors $O(k,h+1) = k \cdot t + 1$

Preuve #1

- Trivial: concaténation des ordonnancements des sous-arbres et ajout de la racine.



q.e.d.

Propriété #2

- Supposons que tous les processeurs soient listés dans un tableau tel que $V[1]=1$ soit la vitesse du plus rapide.
 - Si $O(k,h) = t$
 - et $2 \cdot k - 3 \geq V[2]$
 - alors $O(k,h+1) = k \cdot t$

Preuve #2

- La différence avec la propriété 1 est d'une unité de temps.
- Essayer de mettre une tâche sur le deuxième processeur le plus rapide $P[2]$.
- $P[2]$ est de vitesse $V[2]$, donc on a besoin de $V[2]+1$ unité de temps à cause de la communication.

Preuve #2 (suite)

- Sur $P[1]$, on dispose de $2 \cdot k$ unité de temps.
- Si X est la tâche qui doit être mise sur $P[2]$, alors Y est son père et Y est le fils de R . $Y+R$ vont prendre 2 unités de temps.
- Comme il faut exécuter X ($V[2]+1$) unités de temps avant Y , il faut nécessairement que: $V[2]+1 \leq 2 \cdot k - 2$

q.e.d.

Remarque #2

- Les deux propriétés ci-dessus restent valables pour:
 - N'importe quel nombre de machines, pourvu qu'il y en ait au moins 2.
 - N'importe quelles vitesses de machines, pourvu que celle de la machine la plus rapide soit de 1.
 - Les rapports de vitesse ne sont pas nécessairement des entiers.

Essai #1

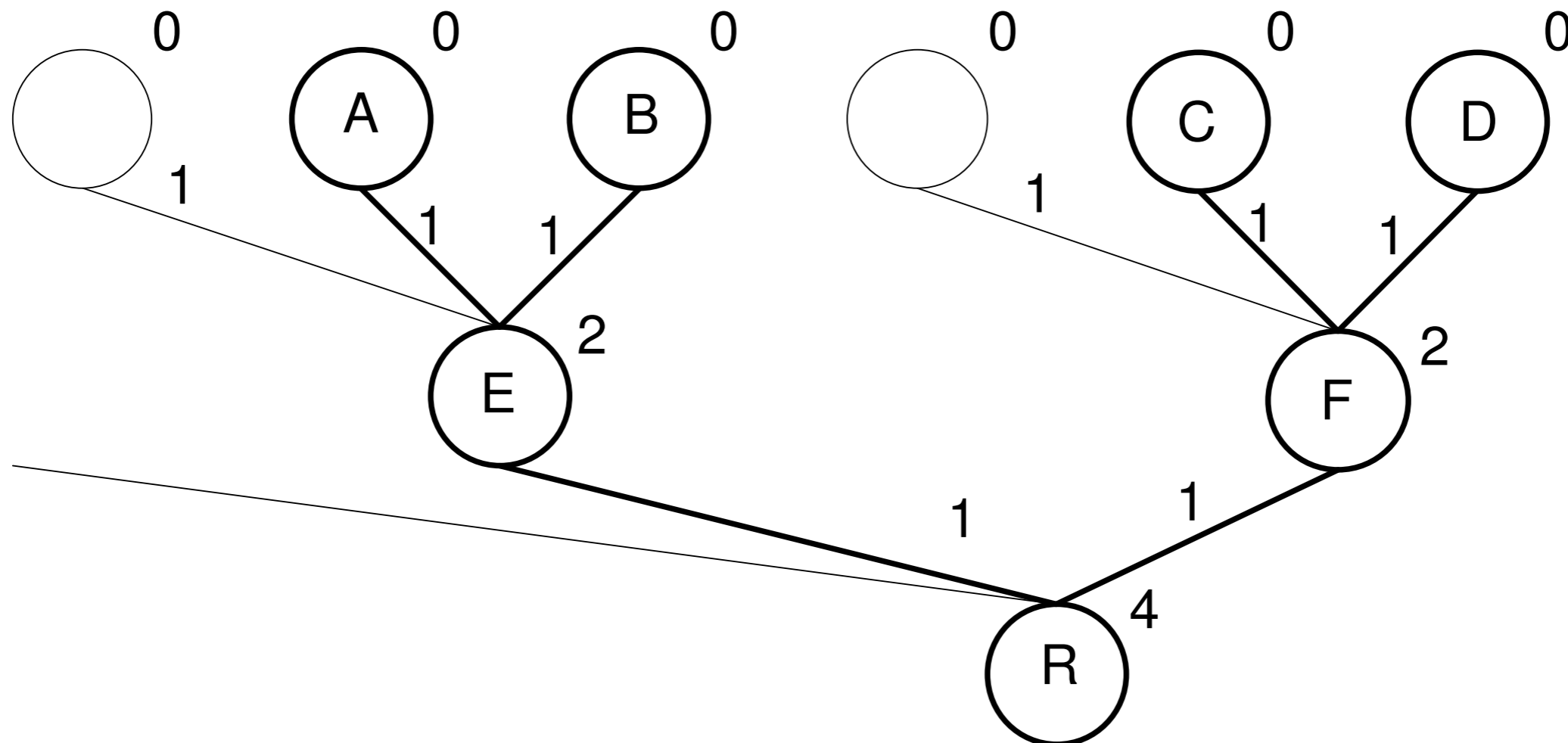
- Généralisation de la formule de LB afin de déterminer le temps minimal d'exécution.
 - Échec,
 - mais j'ai remarqué que, peu importe k , le temps d'exécution au plutôt est de: $2 \cdot h - 1$

Propriété #3

- En supposant qu'on possède un nombre assez important de machines, alors, pour toute valeur de $k \geq 2$, le temps total d'exécution au plutôt est de: $C_{th} = 2 \cdot h - 1$

Preuve #3

- Chaque nœud peut s'exécuter au plutôt 2 unités de temps plus tard que ses pères, car il faut une unité pour exécuter ses pères et une pour effectuer au plus $k-1$ communications.



Essai #2

- Ramener le cas de m machines à plusieurs cas de 2 machines.
 - Comment diviser le problème?
 - Comment décider quel tâche mettre sur quelle groupe?
 - Échec ...

Essai #3

- Extrapolation aux autres tâches de la remarque #1
- Attaque du problème par la queue:
 - Commencer à placer la racine
 - Placer logiquement les autres tâches

Algorithme

- Placer la tâche t de telle manière à ce que sa date de début soit le plus tard que possible (donc maximale) dans l'ordre de préférence que voici:
 1. Sur une machine vide
 2. Sur la machine hébergeant le successeur de t
 3. En cas d'égalité des dates, prendre la machine la plus rapide



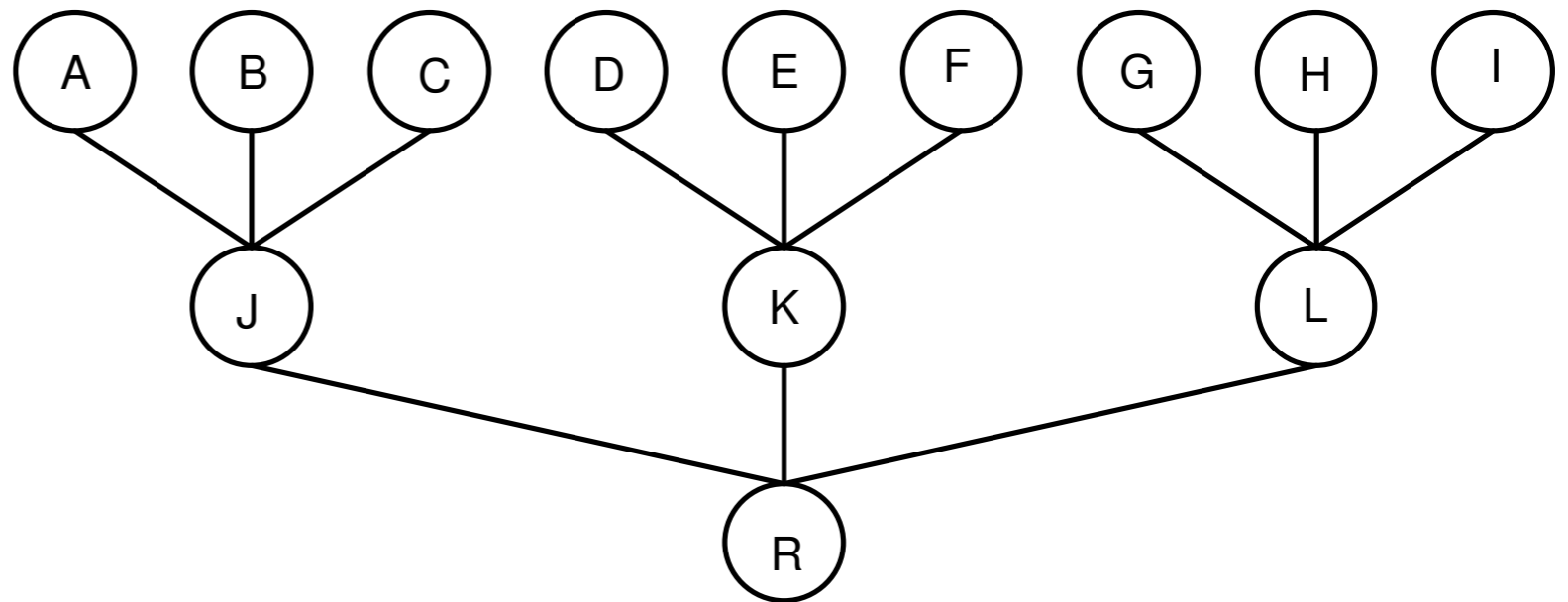
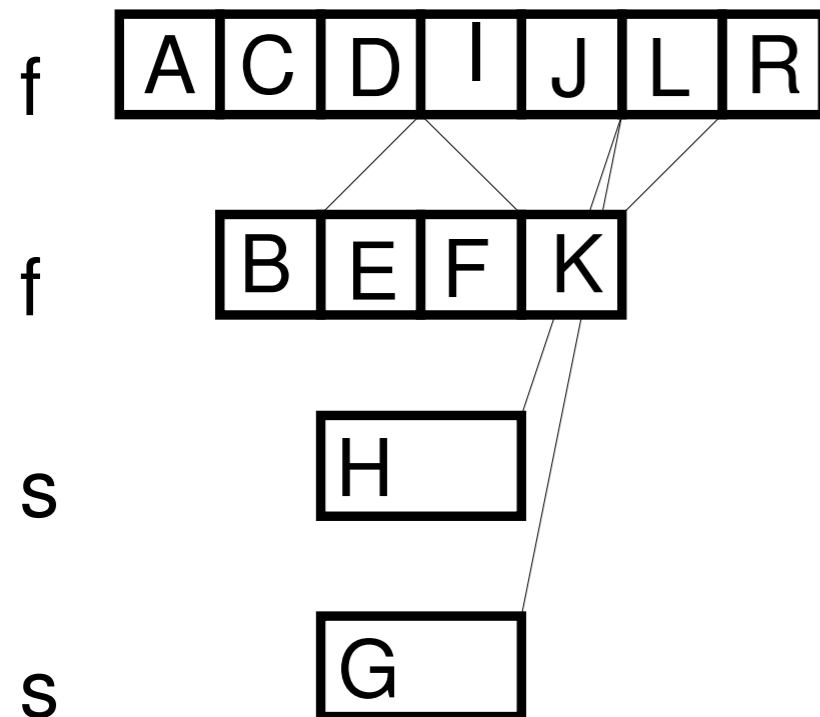
Algorithme (suite)

- Algorithme glouton
- Algorithme de liste (?)
 - Comment sélectionner les tâche?
 - De quelle manière, dans quelle ordre?
- Ne minimise pas les communications

Exemple #3

■ $V[f]=1, V[s]=2,$
 $k=3, h=3$

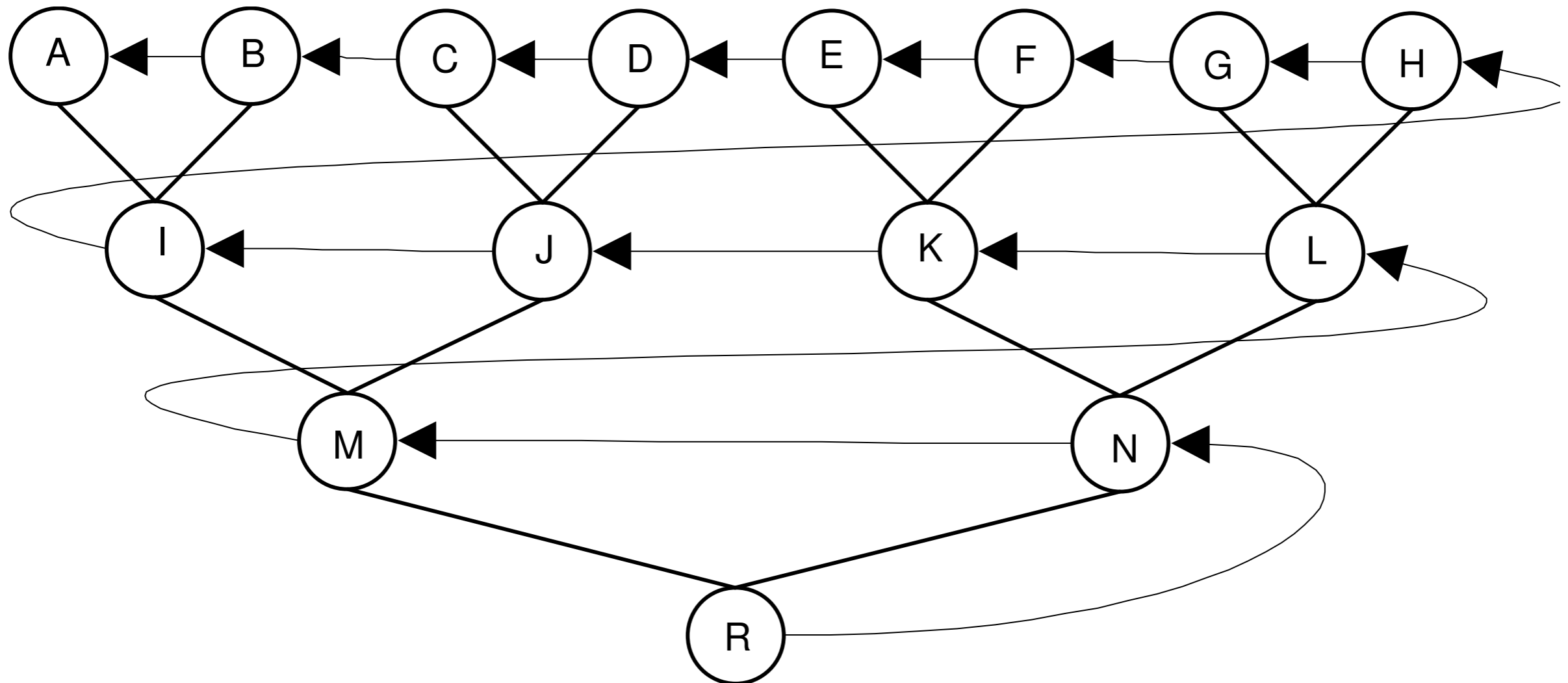
■ $C_{th} = 2 \cdot h - 1 = 5$



■ $C_{max} = 7$

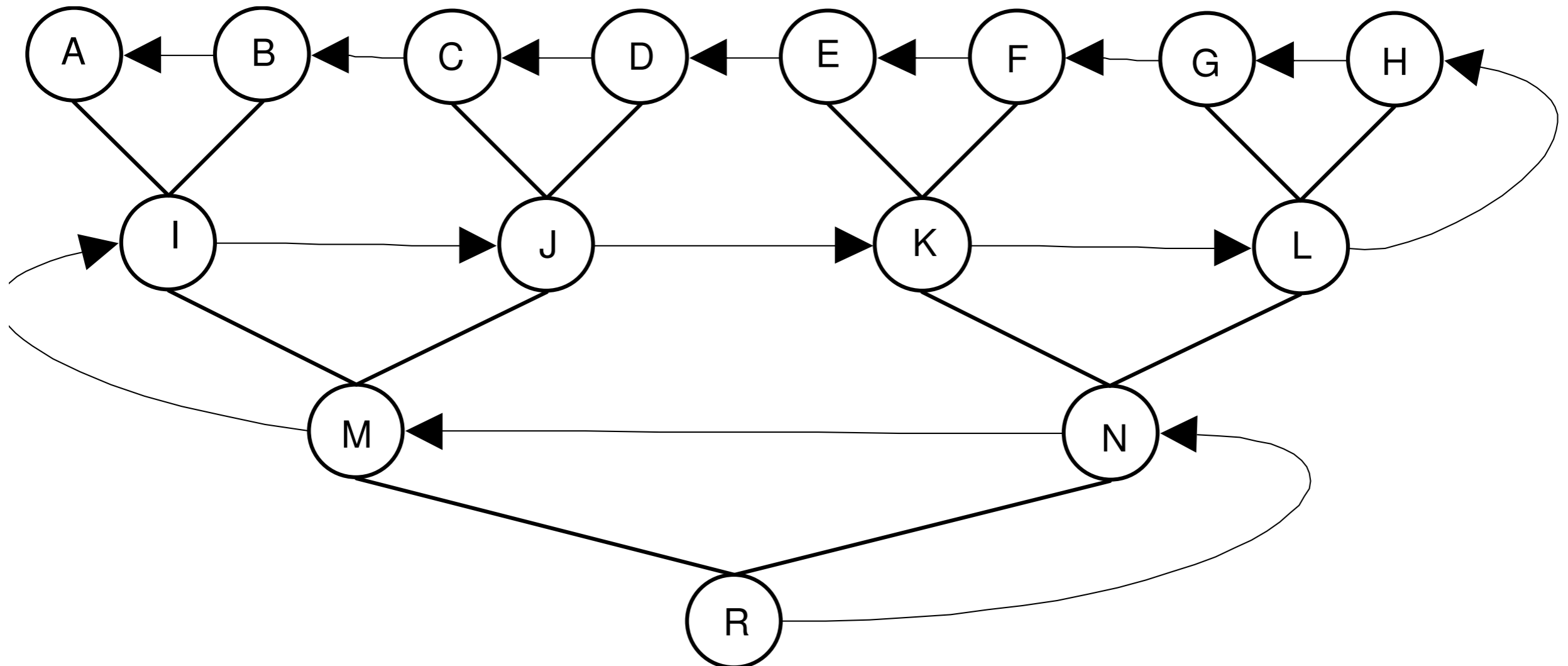
Choix #1

- En zigzag avec retour à la ligne



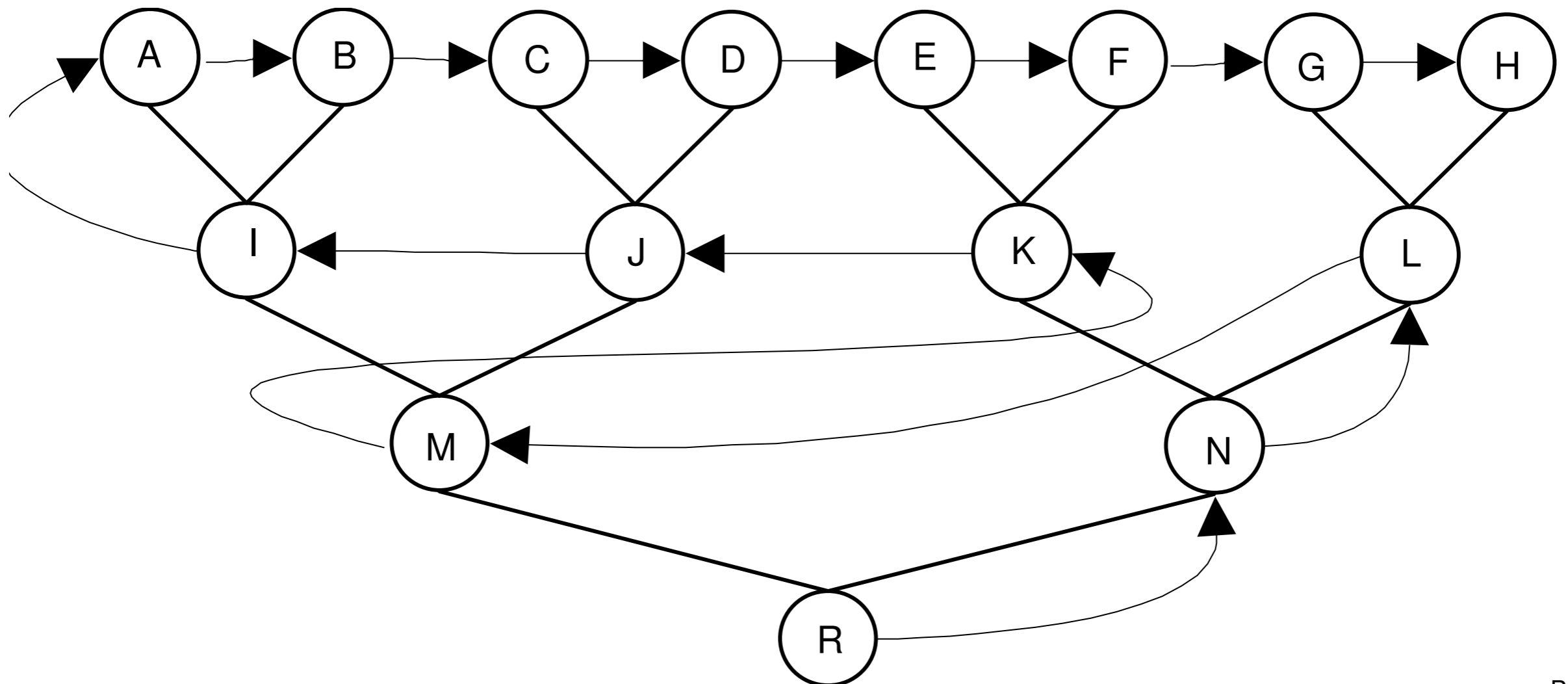
Choix #2

- En zigzag sans retour à la ligne



Choix #3

- En zigzag sans retour à la ligne et avec pénétration en profondeur



Distance maximale?

- Notée: D_{\max}
- C'est le plus grand intervalle de temps qui existe pour un choix d'ordonnancement donné.
- Pour qu'un processeur puisse être utile à un ordonnancement, il faut que sa vitesse soit $< D_{\max}$

Calcul de D_{\max}

■ Choix #1

- $D_{\max} = k^h - 1$

■ Choix #2

- $D_{\max} = k^{(h-1)} + k^{(h-2)} - 1$

■ Choix #3

- $D_{\max} = n - h - 1 = (k^h - 1)/(k - 1) - h - 1$

D_{max} pour le choix #1

h\k	1	2	3	4	5	6	7	8
1	0	1	2	3	4	5	6	7
2	0	3	8	15	24	35	48	63
3	0	7	26	63	124	215	342	511
4	0	15	80	255	624	1295	2400	4095
5	0	31	242	1023	3124	7775	16806	32767
6	0	63	728	4095	15624	46655	117648	262143
7	0	127	2186	16383	78124	279935	823542	2097151
8	0	255	6560	65535	390624	1679615	5764800	16777215
9	0	511	19682	262143	1953124	10077695	40353606	134217727
10	0	1023	59048	1048575	9765624	60466175	282475248	1073741823

D_{max} pour le choix #2

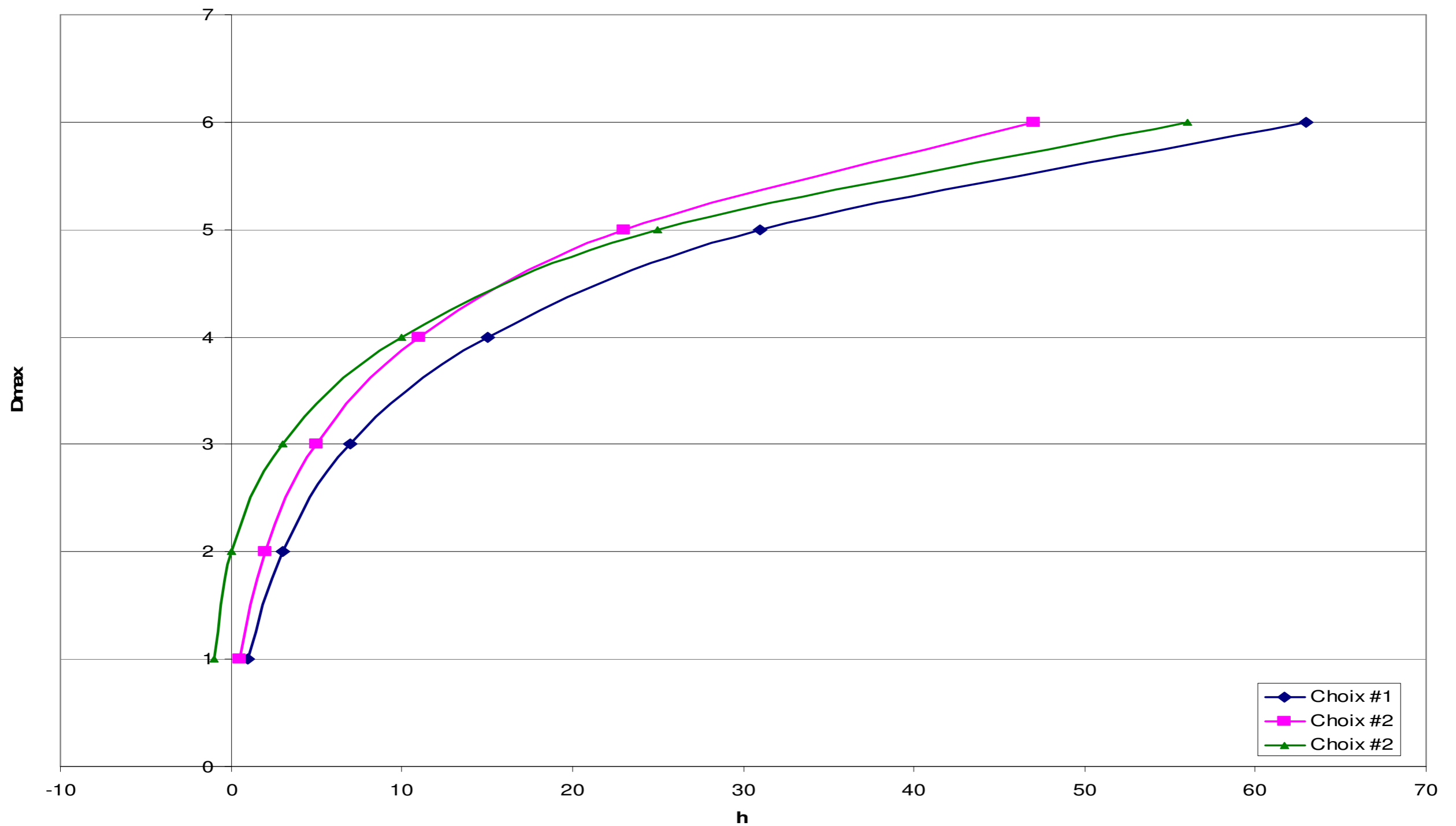
h\k	1	2	3	4	5	6	7	8
1	1	0,5	0,3333333	0,25	0,2	0,1666667	0,1428571	0,125
2	1	2	3	4	5	6	7	8
3	1	5	11	19	29	41	55	71
4	1	11	35	79	149	251	391	575
5	1	23	107	319	749	1511	2743	4607
6	1	47	323	1279	3749	9071	19207	36863
7	1	95	971	5119	18749	54431	134455	294911
8	1	191	2915	20479	93749	326591	941191	2359295
9	1	383	8747	81919	468749	1959551	6588343	18874367
10	1	767	26243	327679	2343749	11757311	46118407	150994943

D_{max} pour le choix #3

h\k	1	2	3	4	5	6	7	8
1	#DIV/0!	-1	-1	-1	-1	-1	-1	-1
2	#DIV/0!	0	1	2	3	4	5	6
3	#DIV/0!	3	9	17	27	39	53	69
4	#DIV/0!	10	35	80	151	254	395	580
5	#DIV/0!	25	115	335	775	1549	2795	4675
6	#DIV/0!	56	357	1358	3899	9324	19601	37442
7	#DIV/0!	119	1085	5453	19523	55979	137249	299585
8	#DIV/0!	246	3271	21836	97647	335914	960791	2396736
9	#DIV/0!	501	9831	87371	488271	2015529	6725591	19173951
10	#DIV/0!	1012	29513	349514	2441395	12093224	47079197	153391678

Comparaison

$k=2$



Comparaison statistiques

Pour les exemples montrées lors de la présentation des différentes méthodes de choix des tâches.

	choix #1	choix #2	choix #3
	0	0	0
	1	1	0
	1	0	2
	2	1	2
	2	3	1
	3	4	2
	3	0	0
	4	1	1
	4	3	3
	5	4	4
	5	6	6
	6	7	7
	6	9	10
	7	10	11
Total	49	49	49
Moyenne	3,5	3,5	3,5
Médiane	3,5	3	2
Écart type	2,139374	3,368405	3,653239

Sens de la réalité?

- Vitesses

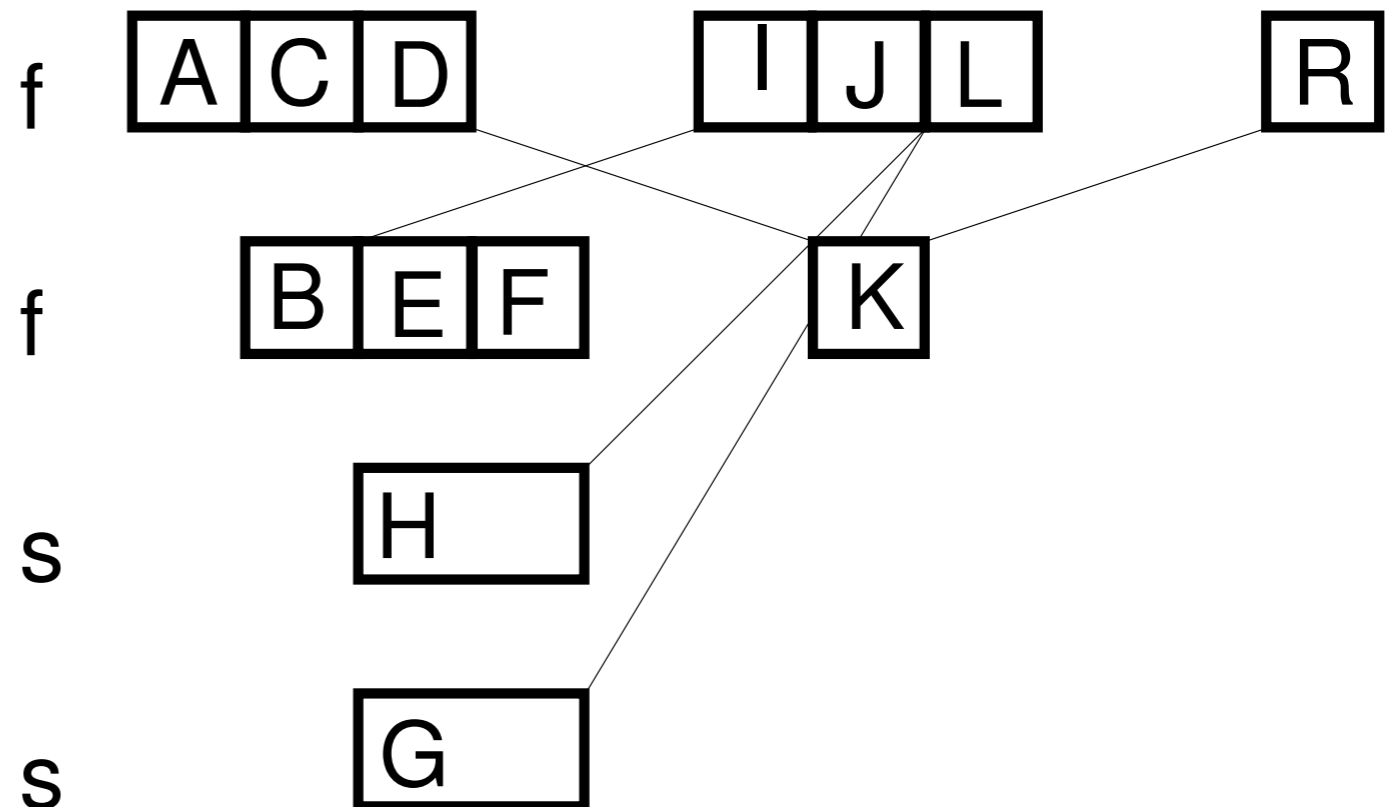
- ☐ maximale aujourd'hui: 3000 Mhz
- ☐ minimale encore exploitable: 300Mhz

- Facteur multiplicatif de seulement 10

- À quoi bon d'étudier des cas où ce facteur multiplicatif, donc la différence entre la machine la plus rapide et la plus lente, est trop grand?

Analyse de sensibilité du délais

- Délais de communication: $1 \rightarrow 3$ [$c^{te}=3$]
- C_{max} : $7 \rightarrow 11 = 7 + 2 \cdot (3 - 1)$
- la plus longue chaine de communications
 - $(c^{te} - 1)$
- Pas très bien ...



Questions?

