Systèmes d'exploitation

Processus. Modèles de représentation

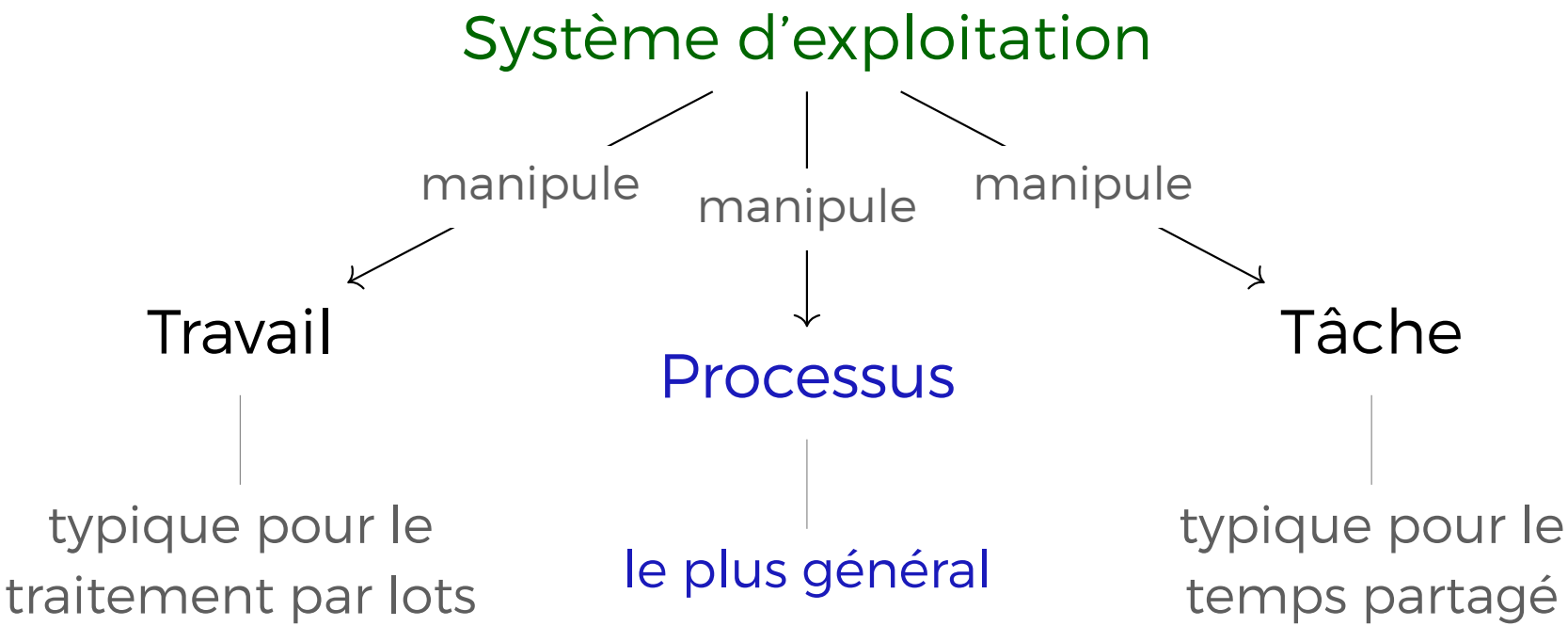
Sergiu Ivanov

sergiu.ivanov@univ-evry.fr

https://www.ibisc.univ-evry.fr/~sivanov/fr/os-ueve.html

Que manipule un système d'exploitation ?

Multiprogrammation



Quelle est la différence ?

Processus vs

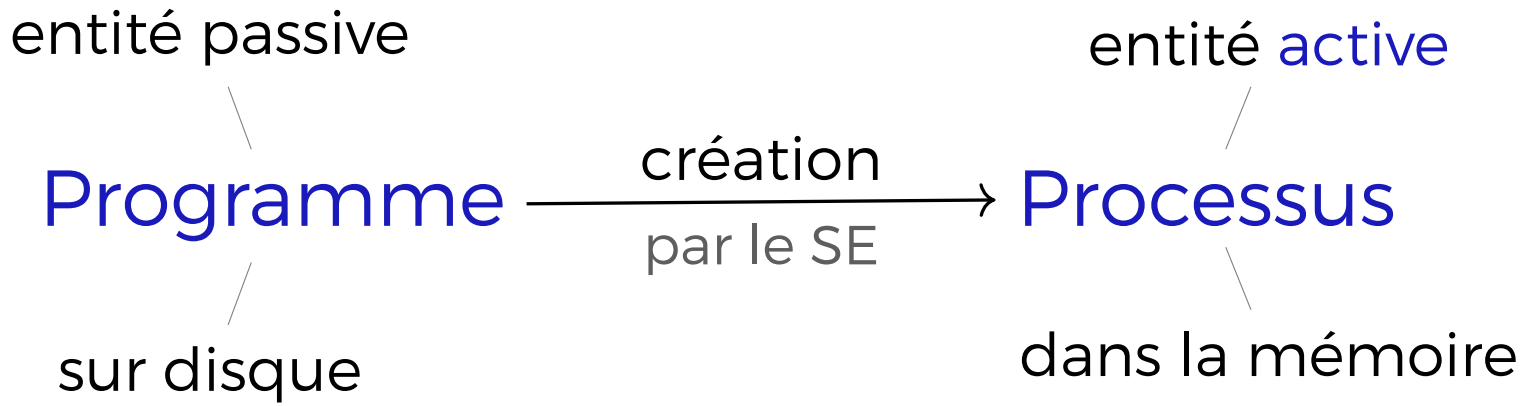
https://openclipart.org/

Quelle est la différence ?

Processus vs Programme

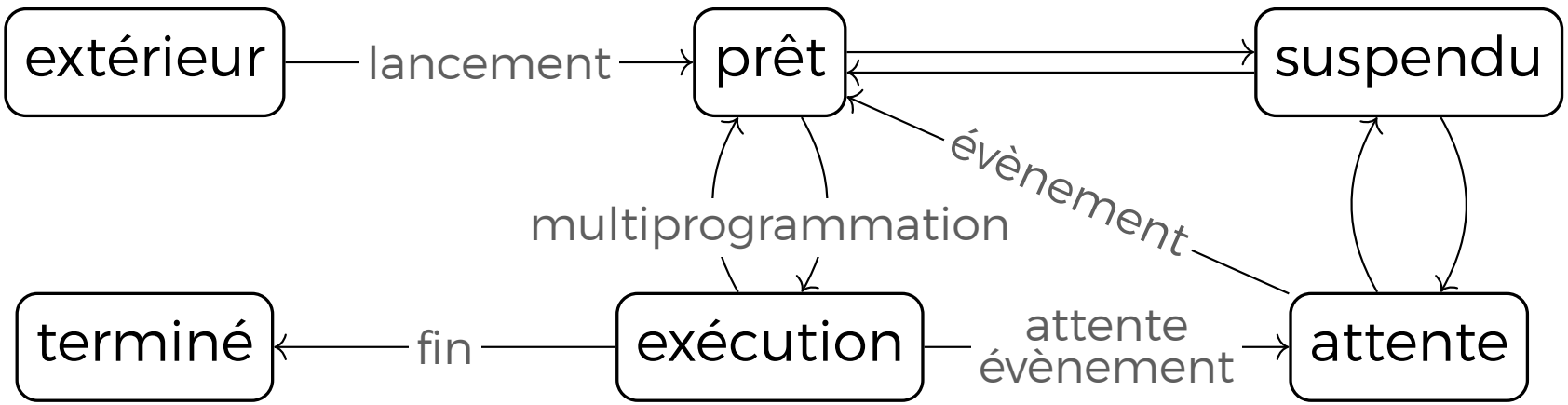
https://openclipart.org/

Processus = programme en exécution



- programme $\xrightarrow{1 \qquad n}$ processus
- programme $\sim$ texte
- processus $\sim$ mémoire $+$ processeur
  - segment code $+$ segment données $+$ pile
  - contexte

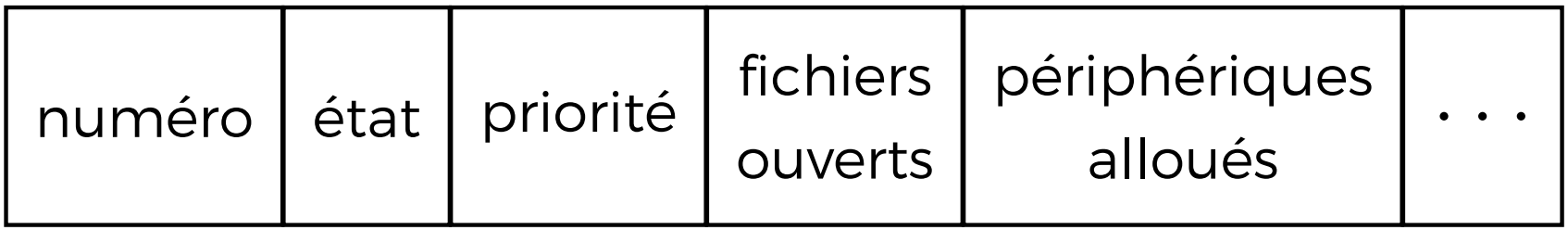
Les états des processus



- attente = attente disponibilité d'une ressource
- suspendu = attente plus longue pour des raisons de performance

Représentation des processus dans le SE

Bloc de contrôle processus = structure de données utilisée par le SE pour représenter et gérer un processus



Outline

Outline

Files d'attente de scheduling

Le SE maintient des files d'attente de processus.

- file générale : tous les processus
- prêts : processus dans l'état `prêt`
  - processus qui attendent l'exécution
- attente périphérique : processus qui attendent le retour d'un périphérique
  - processus dans l'état `bloqué`

Ordonnanceurs schedulers

Ordonnanceur = composant du SE qui choisit l'ordre d'exécution des processus

à court terme

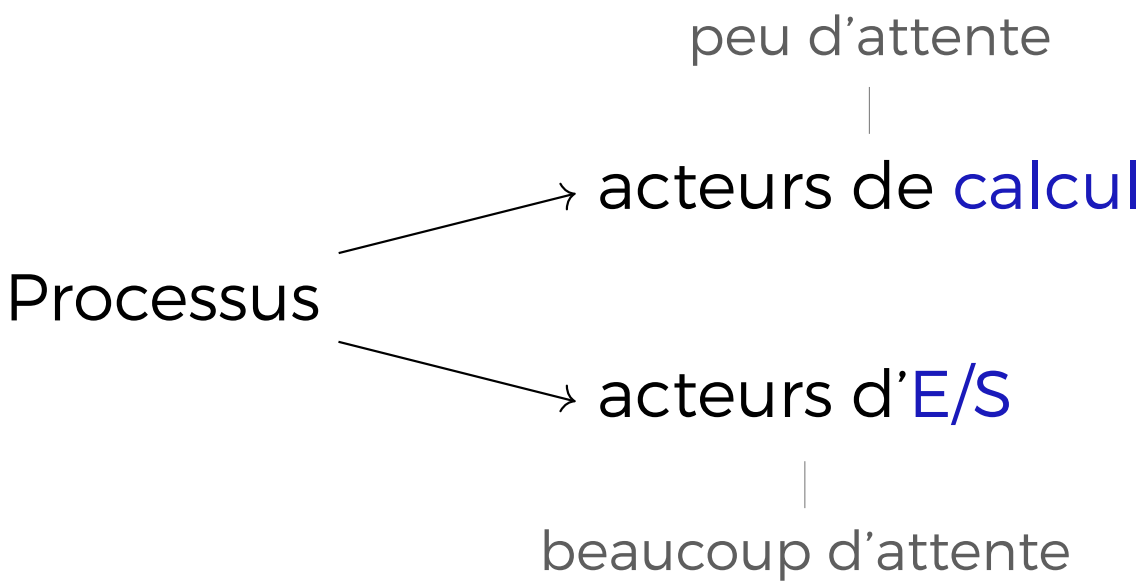
choisit parmi les processus dans l'état prêt

à long terme

choisit parmi les travaux programmés sur un dispositif de stockage

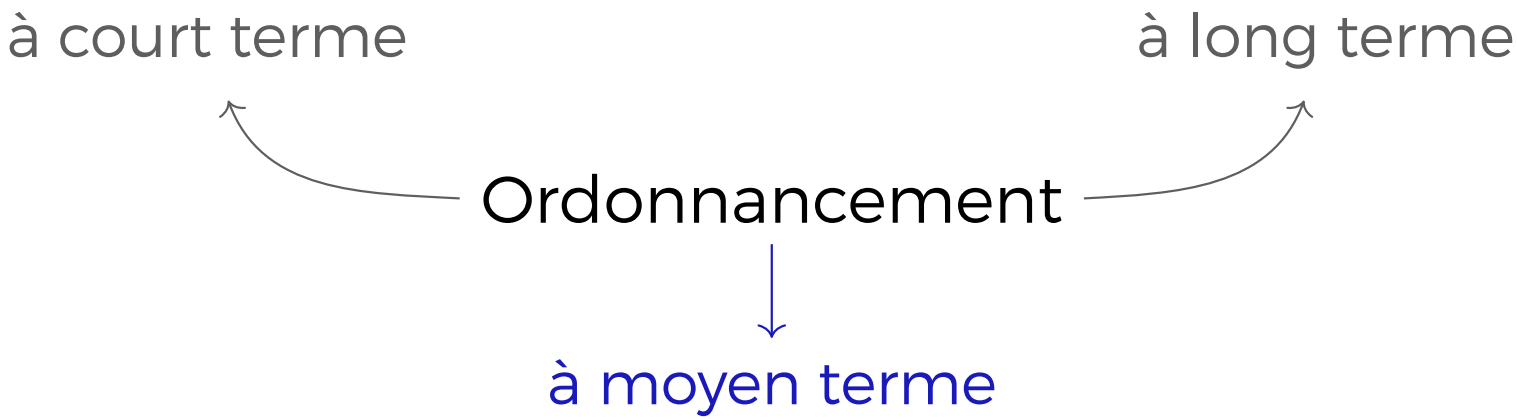
- traitement par lots

Objectifs de l'ordonnanceur



L'ordonnanceur doit réaliser un bon mélange

Swapping

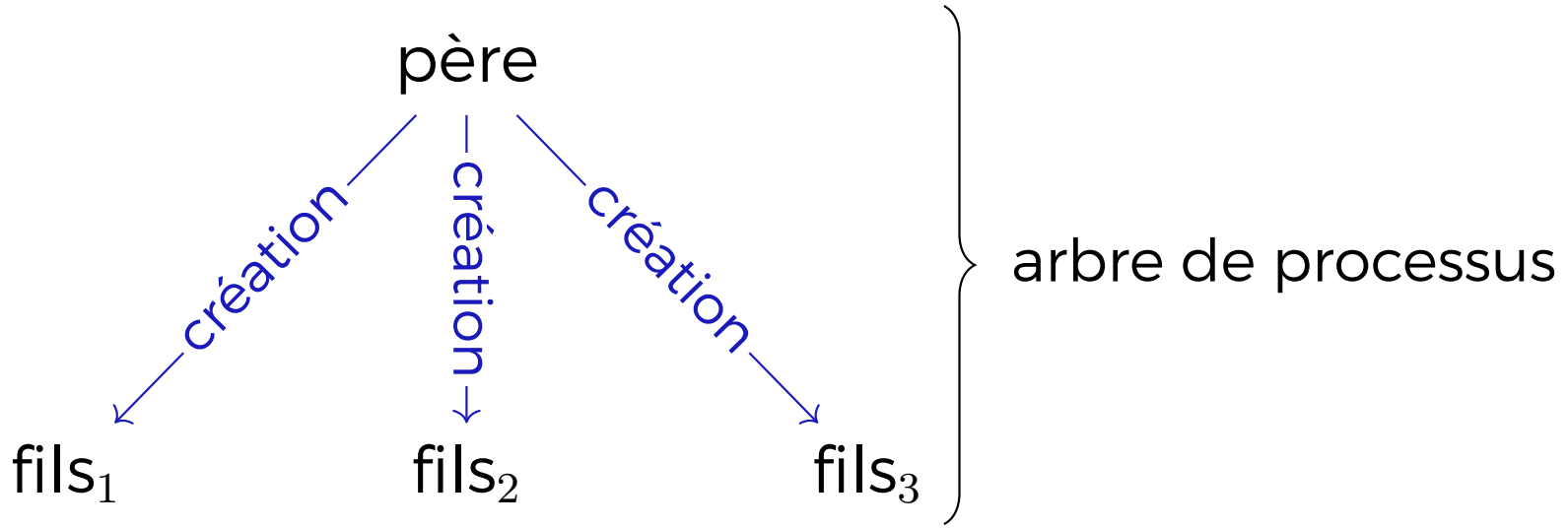


Swapping = sauvegarde de certains processus $\longrightarrow$ disque

- attente longue
- priorité réduite

Outline

Création de processus



Le fils peut être limité aux ressources du père.

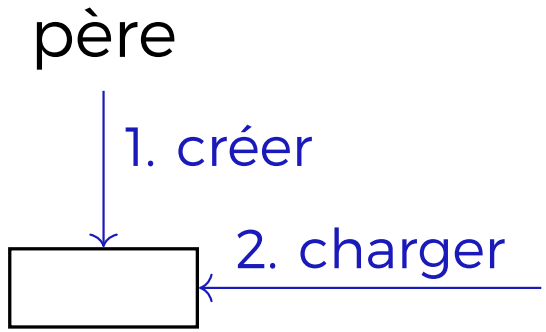
Le père peut transmettre des données d'initialisation.

Modalités de création

« CreateProcess »

typique sous Windows

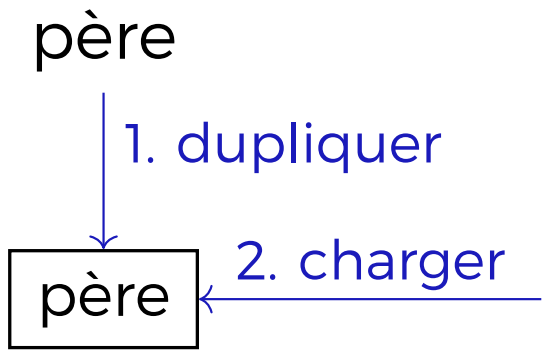
1. créer un nouvel espace mémoire
2. charger un nouveau processus



« fork »

typique sous Unix

1. dupliquer l'espace mémoire du père
2. charger un nouveau processus dans l'un des clones



Outline

Tâches et processus

Tâche T_i $=$ unité élémentaire de traitement

Processus P $=$ suite de tâches

$$P = T_1 T_2 \ldots T_n$$

La décomposition n'est pas unique

Vision par tâches : plus fine

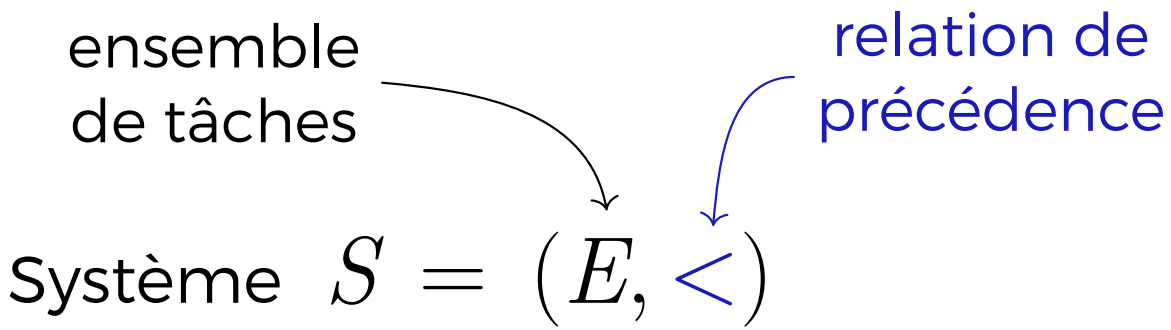
La structure d'une tâche

- lecture des données
- acquisition de ressources
- écriture des résultats
- libération des ressources

Processus $P = T_1\, T_2 \ldots T_n = d_1 f_1\; d_2 f_2 \;\ldots\; d_n f_n$

Outline

Systèmes de tâches



$T_1 < T_2$: la terminaison de T_1 doit survenir avant l'initialisation de T_2

Propriétés exigées de $<$

- $\forall T \in E$, il n'est pas vrai que $T < T$
- $\forall T_1, T_2 \in E$, soit $T_1 < T_2$ ou $T_2 < T_1$, mais pas les deux
- $\forall T_1, T_2, T_3 \in E$, si $T_1 < T_2$ et $T_2 < T_3$, alors $T_1 < T_3$

Graphes de précédence

$$S = (\{T_1, T_2, T_3\}, <) \qquad T_1 < T_2 \qquad T_1 < T_3$$

Comment dessiner cette situation ?

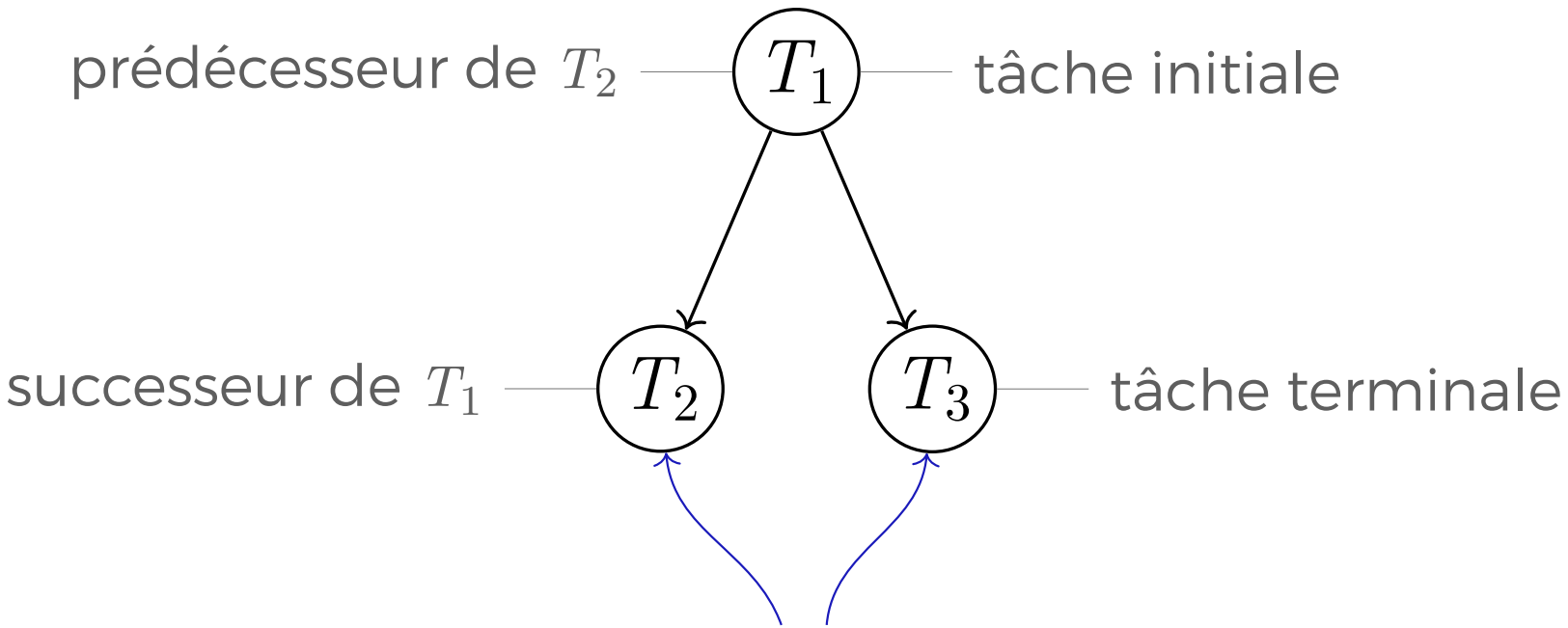
Graphes de précédence

$$S = (\{T_1, T_2, T_3\}, <) \qquad T_1 < T_2 \qquad T_1 < T_3$$

Comment dessiner cette situation ?

Graphes de précédence

$$S = (\{T_1, T_2, T_3\}, <) \qquad T_1 < T_2 \qquad T_1 < T_3$$

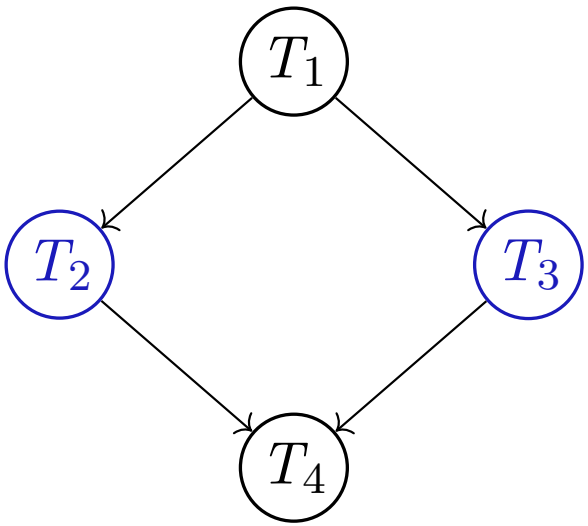


tâches indépendantes

peuvent s'exécuter en parallèle

Outline

Mots/comportements de systèmes de tâches



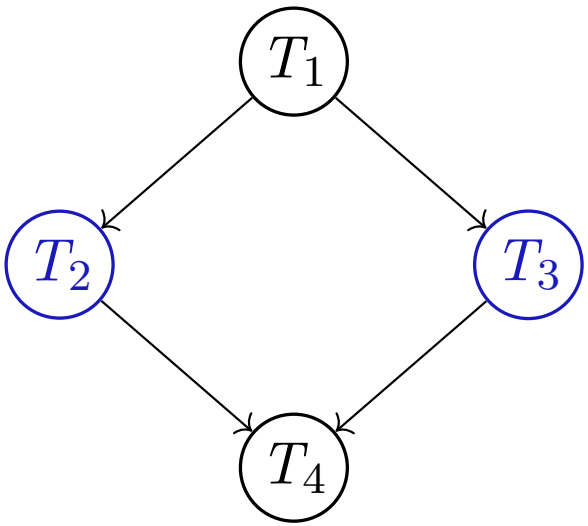
Comportements permis

$T_1\ T_2\ T_3\ T_4$ $\quad$ $T_1\ T_3\ T_2\ T_4$

$d_1 f_1\ d_2 f_2\ d_3 f_3\ d_4 f_4$ $\quad$ $d_1 f_1\ d_3 f_3\ d_2 f_2\ d_4 f_4$

D'autres comportements ?

Mots/comportements de systèmes de tâches



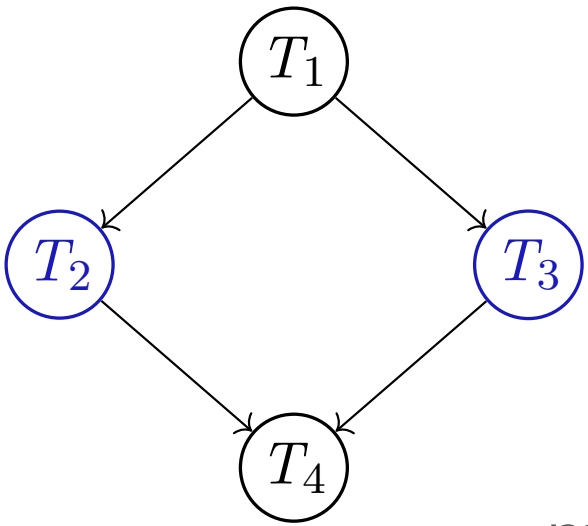
Comportements permis

$T_1\ T_2\ T_3\ T_4$ $\quad$ $T_1\ T_3\ T_2\ T_4$

$d_1 f_1\ d_2 f_2\ d_3 f_3\ d_4 f_4$ $\quad$ $d_1 f_1\ d_3 f_3\ d_2 f_2\ d_4 f_4$

D'autres comportements ?

Mots/comportements de systèmes de tâches



Comportements permis

$T_1\ T_2\ T_3\ T_4$ $\quad$ $T_1\ T_3\ T_2\ T_4$

$d_1f_1\ d_2f_2\ d_3f_3\ d_4f_4$ $\quad$ $d_1f_1\ d_3f_3\ d_2f_2\ d_4f_4$

D'autres comportements ?

mot ↓

$$L = \{\ \overline{d_1f_1\ \underline{d_2f_2\ d_3f_3}\ d_4f_4},\quad d_1f_1\ \underline{d_2\,d_3\ f_2f_3}\ d_4f_4,$$
$$d_1f_1\ \underline{d_2\,d_3\ f_3f_2}\ d_4f_4,\quad d_1f_1\ \underline{d_3\,d_2\ f_2f_3}\ d_4f_4,$$
$$d_1f_1\ \underline{d_3\,d_2\ f_3f_2}\ d_4f_4,\quad d_1f_1\ \underline{d_3f_3\ d_2f_2}\ d_4f_4\ \}$$

↑ le langage du système

Comportements initiaux

Comportement initial $=$ tout préfixe dans L

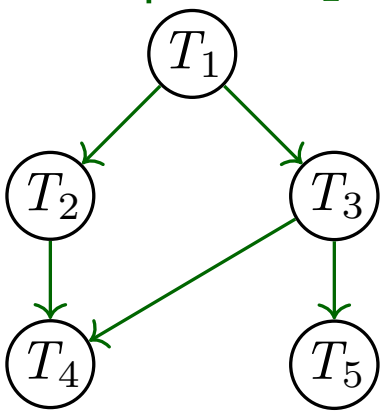
$$L = \{ \quad d_1 f_1 \; d_2 f_2 \; d_3 f_3 \; d_4 f_4, \quad d_1 f_1 \; d_2 d_3 \; f_2 f_3 \; d_4 f_4,$$
$$d_1 f_1 \; d_2 d_3 \; f_3 f_2 \; d_4 f_4, \quad d_1 f_1 \; d_3 d_2 \; f_2 f_3 \; d_4 f_4,$$
$$d_1 f_1 \; d_3 d_2 \; f_3 f_2 \; d_4 f_4, \quad d_1 f_1 \; d_3 f_3 \; d_2 f_2 \; d_4 f_4 \quad \}$$

Comportements initiaux $= \{ d_1 f_1 \; d_2 f_2 \; d_3, \;\; d_1 f_1 \, d_2 \, d_3, \;\; \ldots \}$

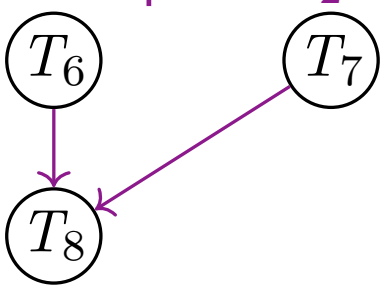
Produit $=$ composition séquentielle

Pour G_1, G_2 deux graphes, connecter toutes les tâches terminales de G_1 à toutes les tâches terminales de G_2.

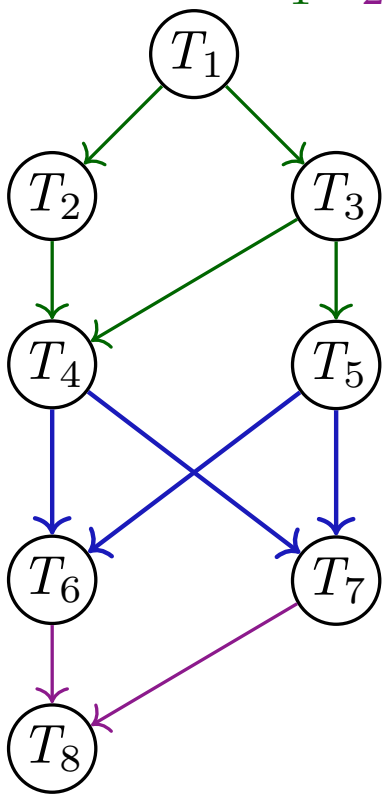
Graphe G_1



Graphe G_2



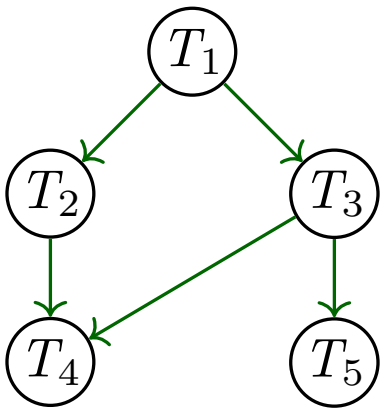
Produit $G_1\,G_2$



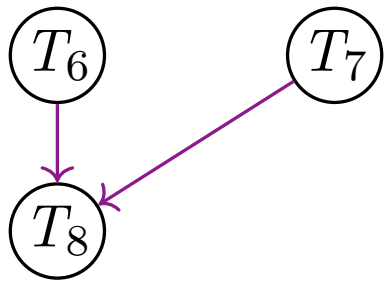
$$L(G_1\,G_2) = \{w_1\,w_2 \mid w_1 \in L(G_1), w_2 \in L(G_2)\}$$

Composition parallèle

Graphe G_1

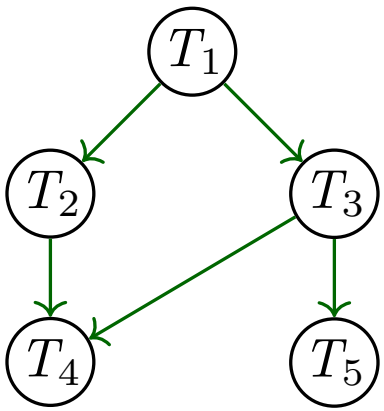


Graphe G_2

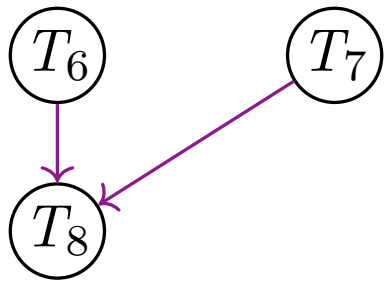


Composition parallèle

Graphe G_1

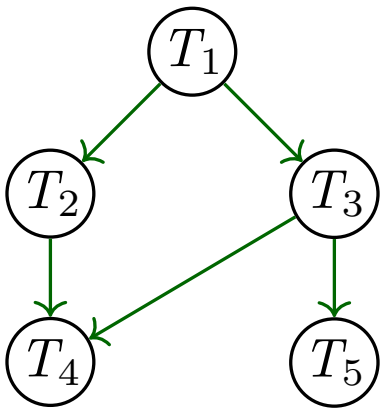


Graphe G_2

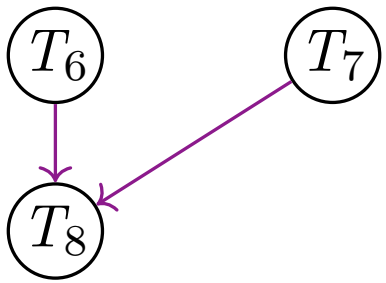


Composition parallèle

Graphe G_1



Graphe G_2

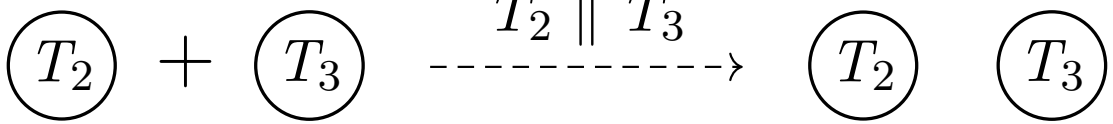


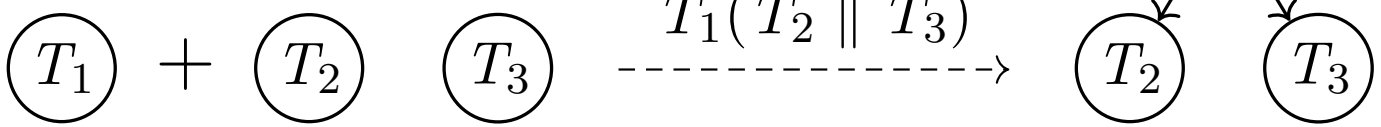
Composition parallèle $G_1 \parallel G_2 = G_1 \cup G_2$

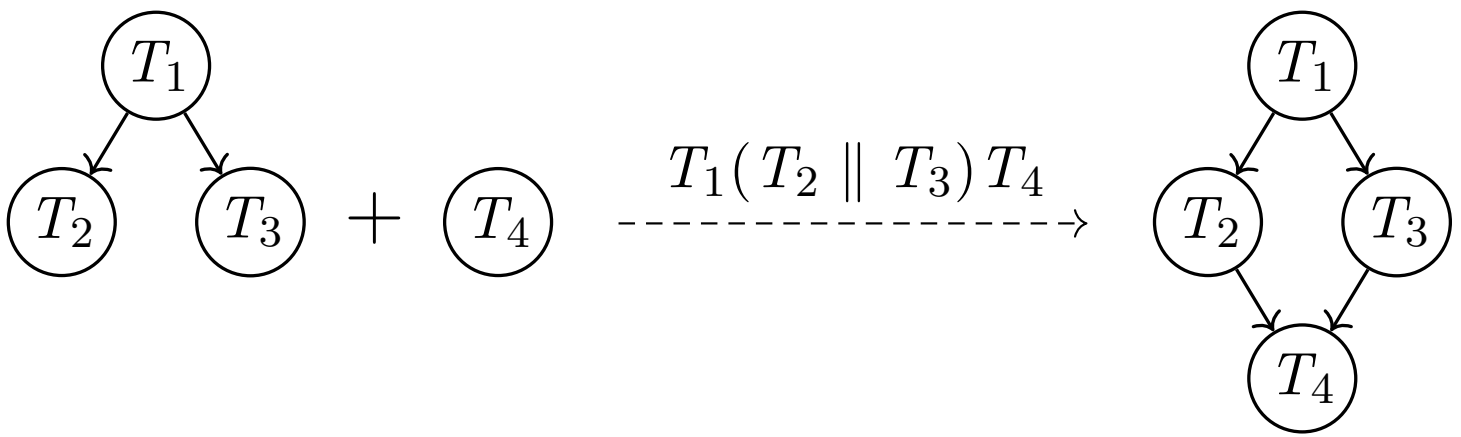
$$L(G_1 \parallel G_2) = \{\text{tous les entrelacements de } w_1 \text{ et } w_2 \mid w_1 \in L(G_1), w_2 \in L(G_2)\}$$

Synthèse de graphes de précédence

Les opérations de composition peuvent être utilisées pour construire certains graphes de précédence.

1. $T_2 + T_3 \xrightarrow{T_2 \parallel T_3}$ 

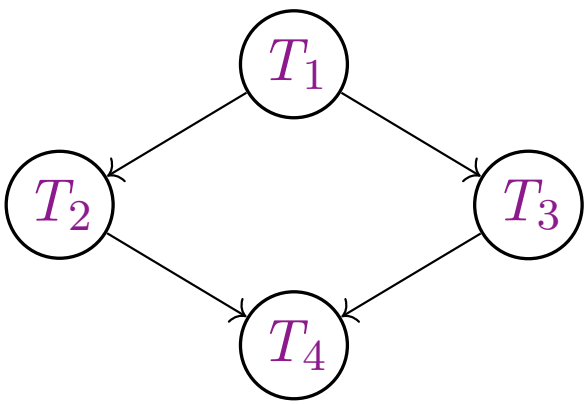
2. 

3. 

Outline

Moyens de spécifier la précédence

Graphes de précédence			Langage de programmation
Séquentiel :	$T_1 \longrightarrow T_2$	$\Longrightarrow$	$\mathtt{T}_1\,;\ \mathtt{T}_2$
Parallèle :	T_2 T_3	$\Longrightarrow$	parbegin $\mathtt{T}_2\,;\ \mathtt{T}_3$ parend



$\Longrightarrow$

$\mathtt{T}_1\,;$

parbegin $\mathtt{T}_2\,;\mathtt{T}_3$ parend;

$\mathtt{T}_4$