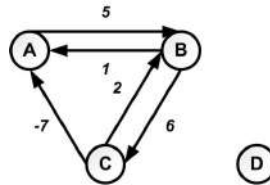


CORRIGE CE 1 TG L3 2013 (18.01.2011)

1) Recherche de plus court chemin

Recherche des plus courts chemins à l'aide de l'algorithme matriciel de Floyd.

Rappel du graphe :



Matrice des valeurs des plus courts chemins et des prédécesseurs.

En état initial, ce sont respectivement la matrice d'adjacence, et les sommets de départ (chemin en une seule étape). A chaque itération, on cherche un meilleur chemin en intégrant une indirection supplémentaire :

$$\begin{aligned}
 W_0 &= \begin{pmatrix} 0 & 5 & \infty & \infty \\ 1 & 0 & 6 & \infty \\ -7 & 2 & 0 & \infty \\ \infty & \infty & \infty & 0 \end{pmatrix} & P_0 &= \begin{pmatrix} A & A & A & A \\ B & B & B & B \\ C & C & C & C \\ D & D & D & D \end{pmatrix} \\
 W_A &= \begin{pmatrix} 0 & 5 & \infty & \infty \\ 1 & 0 & 6 & \infty \\ -7 & \boxed{-2} & 0 & \infty \\ \infty & \infty & \infty & 0 \end{pmatrix} & P_A &= \begin{pmatrix} A & A & A & A \\ B & B & B & B \\ C & \boxed{A} & C & C \\ D & D & D & D \end{pmatrix} \\
 W_B &= \begin{pmatrix} 0 & 5 & \boxed{11} & \infty \\ 1 & 0 & 6 & \infty \\ -7 & -2 & 0 & \infty \\ \infty & \infty & \infty & 0 \end{pmatrix} & P_B &= \begin{pmatrix} A & A & \boxed{B} & A \\ B & B & B & B \\ C & A & C & C \\ D & D & D & D \end{pmatrix} \\
 W_C &= \begin{pmatrix} 0 & 5 & 11 & \infty \\ \boxed{-1} & 0 & 6 & \infty \\ -7 & -2 & 0 & \infty \\ \infty & \infty & \infty & 0 \end{pmatrix} & P_C &= \begin{pmatrix} A & A & B & A \\ \boxed{C} & B & B & B \\ C & A & C & C \\ D & D & D & D \end{pmatrix} \\
 W_D &= \begin{pmatrix} 0 & 5 & 11 & \infty \\ -1 & 0 & 6 & \infty \\ -7 & -2 & 0 & \infty \\ \infty & \infty & \infty & 0 \end{pmatrix} & P_D &= \begin{pmatrix} A & A & B & A \\ C & B & B & B \\ C & A & C & C \\ D & D & D & D \end{pmatrix}
 \end{aligned}$$

Interprétation : par exemple, pour aller de C vers B, il existe un plus court chemin de poids $W_D(C, B) = -2$.

Pour arriver en B, il faut venir directement de $P_D(C, B) = A$, et pour arriver en A, il faut venir directement de $C = P_D(C, A)$.

2) Modélisation du graphe d'ordonnancement

On construit un graphe d'ordonnancement à partir d'un tableau de contraintes.

Chaque ligne de ce tableau spécifie une tâche en renseignant son nom, sa durée, les tâches dont elle dépend (celles qui doivent être terminées pour que celle-ci puisse commencer).

A partir de ce tableau, on construit un graphe orienté et valué dont chaque sommet représente une tâche et dont les arcs représentent une relation de dépendance entre deux tâches, orientés de celle qui précède vers celle qui suit. Chacun des arcs est pondéré de la durée d'exécution de la tâche qui en constitue l'extrémité initiale.

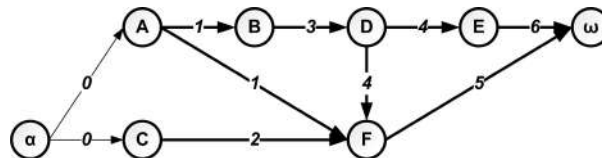
Une tâche fictive sans durée, nommée conventionnellement α et qui représente le démarrage du projet (ou du processus) est positionnée comme contrainte implicite des tâches qui n'ont pas de contraintes.

On matérialise de même une tâche fictive ω qui représente la terminaison du projet et qui a pour contrainte toutes les tâches qui ne sont explicitement contrainte d'aucune autre.

Notons que, alors que la tâche de démarrage peut éventuellement être omise (mas ceci n'est pas une recommandation...), cette tâche fictive de terminaison de projet est une obligation car elle-seule permet de représenter la durée d'exécution des tâches représentées par ses prédécesseurs dans le graphe !

3) Calcul de calendrier

Rappel du graphe d'ordonnancement des tâches :



Les **dates au plus tôt** pour démarrer chaque tâche sont calculées de proche en proche, itérativement, partant de celle $\tau_\alpha = 0$ de la tâche fictive α de début de projet, suivant la formule $\tau_i = \max(\tau_j + d_j)_{j \in \Gamma^{-1}(i)}$. On détermine finalement la date au plus tôt τ_ω de fin de projet.

Les **dates au plus tard** pour démarrer chaque tâche sont calculées selon le même principe, dans la direction opposée, depuis $T_\omega = \tau_\omega$ suivant la formule $T_i = \min(T_j)_{j \in \Gamma(i)} - d_i$.

Pour chaque tâche, la **marge totale** est la différence entre la date au plus tard et la date au plus tôt. Les tâches pour lesquelles cette marge est nulle sont les **tâches critiques**. Celles-ci forment le **chemin critique**, c'est-à-dire le plus long chemin de α vers ω lequel détermine la durée incompressible du projet, c'est-à-dire τ_ω .

La **marge libre** d'une tâche est la différence entre le minimum des dates de début au plus tôt des tâches qui dépendent de celle-ci, et sa date de terminaison au plus tôt : $\min(\tau_j)_{j \in \Gamma(i)} - (\tau_i + d_i)$.

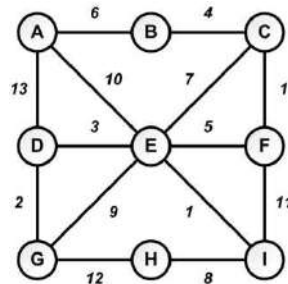
Utilisation par le manager de projet : tant que la marge totale d'une tâche n'est pas consommée, le projet ne prend pas de retard. Tant que la marge libre d'une tâche n'est pas consommée, il n'est pas nécessaire de retarder d'autres tâches qui en dépendent.

Application au cas :

tâche	au + tôt	au + tard	marge	critique
α	0	0	0	O
A	0	0	0	O
B	1	1	0	O
C	0	7	7	X
D	4	4	0	O
E	8	8	0	O
F	8	9	1	X
ω	14	14	0	O

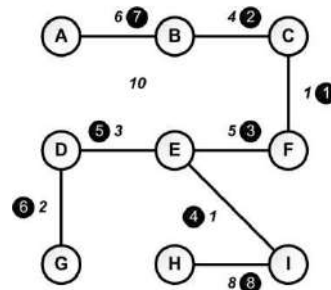
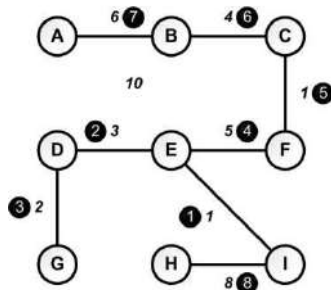
4) Arbre couvrant de poids minimal

Rappel du graphe :



L'algorithme de Prim peut suivre différents enchaînements lorsqu'il y a plusieurs arêtes de même poids pouvant être choisies, et surtout parce qu'il est possible de démarrer de n'importe quel sommet du graphe pour ensuite étendre l'arbre calculé.

Dans l'exercice proposé, les deux enchaînements (départ de E et de F) conduisent à une solution unique de poids 30 (les indices cerclés de noir représentent l'ordre d'ajout des arêtes par l'algorithme de Prim pour produire le résultat final) :



5) Modélisation sous forme de graphe

Note préliminaire :

Voici des idées pour une analyse complète. Bien entendu, nous n'attendons pas nécessairement toute cette analyse de votre part, mais au moins :

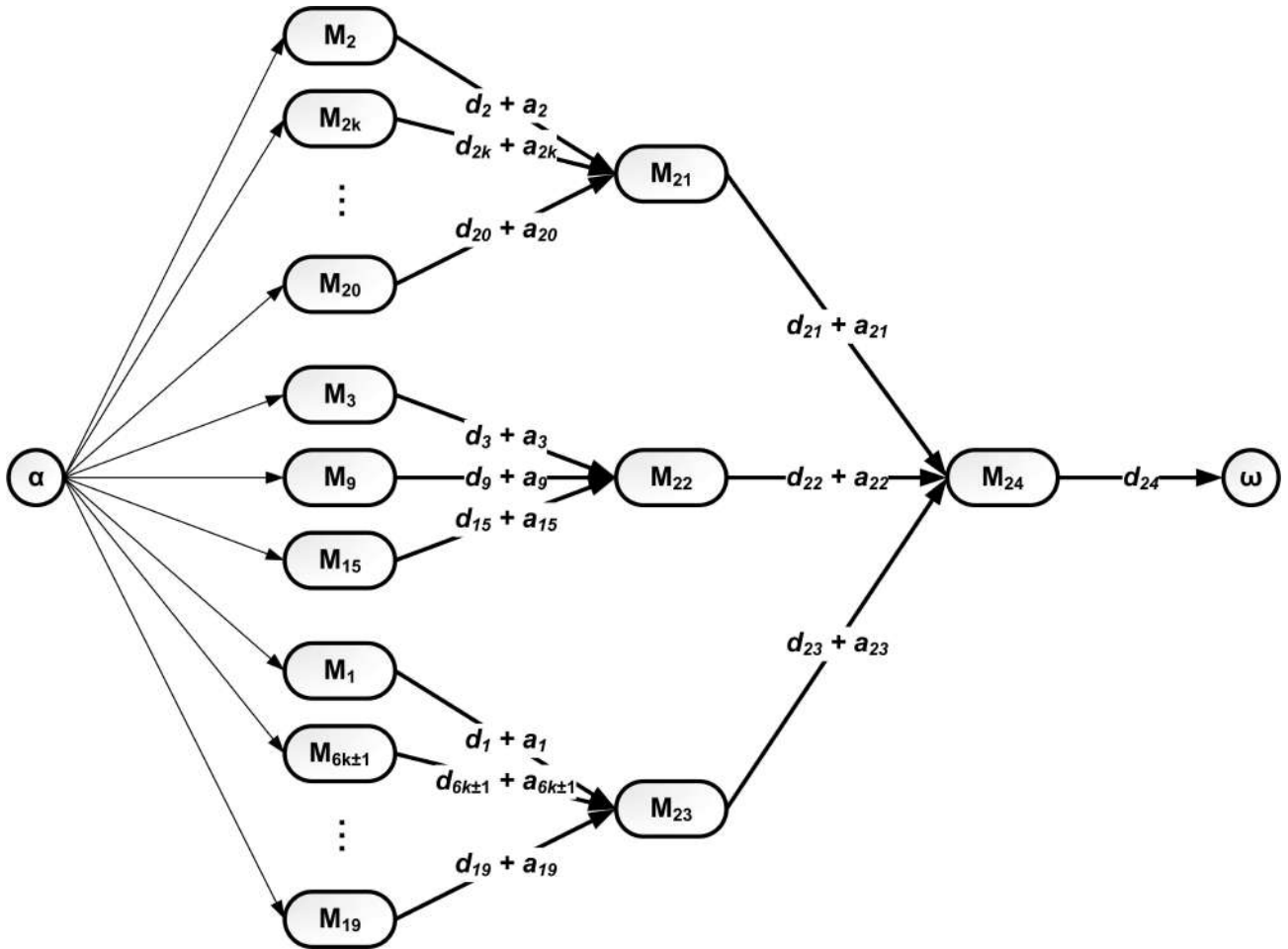
- une modélisation acceptable de la chaîne de production sous forme de graphe ;
- l'identification de pistes de résolution basées sur votre modélisation, notamment le lien avec des problèmes d'ordonnancement, la mention de dates au plus tôt ou de chemins les plus longs, ...

Nommons les machines de fabrication et d'assemblage, ainsi que, par abus de langage, un cycle élémentaire de production qu'elles effectuent :

- Les 20 machines de fabrication sont notées $(M_i)_{i \in \llbracket 1, 20 \rrbracket}$, les machines d'assemblage intermédiaires M_{21}, M_{22}, M_{23} et la machine d'assemblage de la « chose », M_{24} .
- Les temps respectifs de fabrication et d'assemblage de ces machines sont notés $(d_i)_{i \in \llbracket 1, 24 \rrbracket}$.
- Les temps d'acheminement des pièces entre les machines sont (l'énoncé n'indique pas que ce temps est constant et dans un cas réel, cela est peu probable) respectivement $(a_i)_{i \in \llbracket 1, 23 \rrbracket}$.

Commençons par représenter le graphe d'ordonnancement qui représente le fonctionnement de l'usine.

La distribution des rôles entre les machines suit quelques règles d'arithmétique modulaire dont le développement n'a pas sa place ici :



Représentation alternative possible : les machines sont représentées par 2 sommets MA et MB . Un arc de MA_i vers MB_i représente fabrication ou assemblage et sa durée est d_i ; les arcs MB vers MA , par exemple MB_{19} vers MA_{23} , ont pour valeur les durées d'acheminement. Cela rend un peu plus complexe ce qui suit...

Les dates aux plus tôt sont :

$$\begin{cases} \tau_{i \in [1, 20]} = \tau_\alpha = 0 \\ \tau_{21} = \max(\tau_i + d_i + a_i)_{i \in [1, 20] \wedge i \equiv 0(2)} \\ \tau_{22} = \max(\tau_i + d_i + a_i)_{i \in [1, 20] \wedge i \not\equiv 0(2) \wedge i \equiv 0(3)} \\ \tau_{23} = \max(\tau_i + d_i + a_i)_{i \in [1, 20] \wedge i \not\equiv 0(2) \wedge i \not\equiv 0(3)} \\ \tau_{24} = \max(\tau_i + d_i + a_i)_{i \in [21, 23]} \\ \tau_\omega = \tau_{24} + d_{24} \end{cases}$$

La durée d'un cycle de production dépend d'un chemin critique qui dépend nécessairement de trois machines particulières, $M_{p \in [1, 20]}$ l'une des machines de fabrication, $M_{q \in [21, 23]}$ l'une des machines d'assemblage intermédiaire et de la machine qui assemble finalement la « chose » M_{24} . Ce sont les machines critiques.

Nous avons donc : $\tau_\omega = d_p + a_p + d_q + a_q + d_{24}$

Les dates au plus tard sont :

$$\left\{ \begin{array}{l} T_{\omega} = \tau_{\omega} \\ T_{24} = T_{\omega} - d_{24} \\ T_{21} = T_{24} - (d_{21} + a_{21}) \\ T_{22} = T_{24} - (d_{22} + a_{22}) \\ T_{23} = T_{24} - (d_{23} + a_{23}) \\ T_{i \in \llbracket 1, 20 \rrbracket \wedge i \equiv 0(2)} = T_{21} - (d_i + a_i) \\ T_{i \in \llbracket 1, 20 \rrbracket \wedge i \not\equiv 0(2) \wedge i \equiv 0(3)} = T_{22} - (d_i + a_i) \\ T_{i \in \llbracket 1, 20 \rrbracket \wedge i \not\equiv 0(2) \wedge i \not\equiv 0(3)} = T_{23} - (d_i + a_i) \end{array} \right.$$

Intégrons à présent une hypothèse de fonctionnement en flux tendu : sauf à effectuer un stockage de pièces intermédiaires en surproduction dans la perspective de les vendre comme pièces détachées, on peut considérer que les cadences de production (temps d'acheminement compris) des machines de fabrication sont synchronisées sur celle de $M_{p \in \llbracket 1, 20 \rrbracket}$ et que celles des machines d'assemblage intermédiaire le sont sur celle de

$$M_{q \in \llbracket 21, 23 \rrbracket}.$$

Quand ces machines ont produit un lot de 20 pièces, elles peuvent attaquer la production des 20 suivantes.

Par ailleurs, si la machine M_{24} est plus lente à effectuer un assemblage que le temps nécessaire aux machines précédentes pour effectuer une fabrication ou un assemblage, il est clair que faire tourner ces dernières à plein régime entrainera une surproduction dans les étapes intermédiaires de la chaîne. En revanche, si elle est plus rapide, on pourra faire tourner les machines d'assemblage intermédiaire à plein régime (calé sur celui de $M_{q \in \llbracket 21, 23 \rrbracket}$). Le même raisonnement s'applique entre les machines d'assemblage intermédiaire et celles de fabrication. En revanche (c'est une hypothèse qui ne figure pas dans l'énoncé), on peut supposer que le temps d'acheminement entre machines est inférieur au temps de fabrication, et ne retarde donc pas la chaîne.

En conséquence, la meilleure cadence possible est fixée par le plus long des trois temps d_p , d_q et d_{24} .

Ce temps d détermine donc la cadence de production de la chaîne. Pour que la machine M_{24} tourne à plein régime il est donc nécessaire qu'elle soit la machine la plus lente de la chaîne de production. L'ensemble des calcul effectués plus haut s'uniformisent et donnent $\tau_{\omega} = 3d$.

Si l'on considère le stock S de matière première à un moment donné, compté en unité entières de quantité de matière première nécessaire pour fabriquer et assembler une unité de la « chose », alors la vitesse d'approvisionnement de ce stock doit être $\frac{1}{d}$. Si l'approvisionnement s'arrête, le dernier prélèvement sur le

stock interviendra de $S(d + \varepsilon)$, avec $d - \varepsilon$ le temps depuis le dernier prélèvement, et la production s'arrêtera τ_{ω} après, c'est-à-dire $S(d + \varepsilon) + 3d$ après l'arrêt de l'approvisionnement en matière première.

Dernières questions :

- Rebut : cela revient à ralentir le flux de production des machines concernées. Il y a une marge de manœuvre jusqu'à un certain seuil où la machine M_{24} devra ralentir son rythme. Dans ce cas seulement, d est augmentée et la vitesse d'approvisionnement de l'usine ralentie.
- Recyclage : on ajoute une machine de recyclage (des rebuts). On suppose qu'elle ne ralentit pas le reste de la chaîne. Cela revient à un problème de baignoire : le flux composé des rebuts alimente le stock, et réduit d'autant le flux d'approvisionnement de l'usine depuis l'extérieur.

La machine de recyclage fait l'objet d'un nouveau sommet dans le graphe. Les arcs sont créés des autres machines vers celle-ci (avec durée de fabrication + d'acheminement), et de celle-ci vers les autres (avec durée de recyclage + durée d'acheminement).

OPTIMISATION : Transport par palettes pour mutualiser ce coût de transport, en particulier si les machines de production sont plus lentes que les machines d'assemblage.

6) Arbre (question de cours)

Un arbre est un **graphe connexe** (implicitement non orienté) **acyclique**.

En langage SDD, on précisera qu'il s'agit d'un **arbre non enraciné**.

Une conséquence est qu'un arbre est un graphe qui comporte une arête de moins de sommets.

En revanche, la réciproque est fausse. Pour qu'un graphe à N sommets et $N - 1$ arêtes soit un arbre il faut et il suffit qu'il soit acyclique, et il faut et il suffit qu'il soit connexe (*démonstration ?*).

7) Composante connexe / fortement connexe (question de cours)

La connexité traduit la notion intuitive « d'un seul tenant ».

Un **graphe orienté** (resp. **non orienté**) est **fortement connexe** (resp. **connexe**) ssi pour tout couple de **sommets** u, v DISTINCTS, il existe un **chemin** (resp. une **chaîne**) de u vers v (resp. qui relie u et v).

Une **composante fortement connexe** (resp. **composante connexe**) d'un graphe est un **sous-graphe** fortement connexe (resp. connexe) **maximal**, c'est-à-dire qui n'est pas strictement contenu dans un autre sous-graphe de même type.