EFREI-L3 **Introduction au cours "Optimisation et complexité"**
(recherche opérationnelle = RO)

I. Définition

RO : la recherche opérationnelle est un outil d'**aide à la décision** qui permet d'optimiser une fonction économique en présence de contraintes multiples.

remarque : la RO prend des formes très diverses en univers certain ou univers aléatoire.

règle : - les choix des objectifs + critère d'optimisation sont à la charge de l'entrepreneur
- la RO consiste à trouver l'optimum.

II. Domaines d'application

1) Domaine combinatoire : le but est d'éviter l'énumération des solutions (affecter 25 secrétaires à 25 postes → 25! ≈ 1,5 10^{25} solutions : 1 solution/µs → 5 10^{9} siècles)

- *Chemin optimal* dans un graphe valué : trouver le chemin le moins (plus) cher pour aller de A à B.
- *Problème d'ordonnancement* : pour la réalisation d'un projet consistant à accomplir des tâches soumises à des contraintes (d'antériorité, de dates…), trouver l'ordre des tâches, la solution la moins longue (et/ou la moins chère), les marges de réalisation des tâches…
- *Problème de flot maximal* dans un réseau de transport (graphe valué avec E/S) : acheminer le plus possible de marchandises de E vers S compte tenu des capacités de transport de chaque arc.
- *Problème d'affectation* : pour 25 personnes à affecter dans 25 postes : chaque personne donne ses préférences (classement de 1 à 25) et on cherche une solution satisfaisant le plus possible de personnes.
- *Problème de transport* : trouver les quantités x_{ij} à transporter depuis m usines vers n distributeurs pour minimiser la somme des coûts de transport.
- *Problème du voyageur de commerce* : trouver le parcours minimisant le coût total (déplacement + hôtel) du voyageur qui doit visiter tous ses clients et revenir chez lui.

2) Domaine aléatoire : le hasard intervient (→ probabilité, espérance, processus stochastique…)

- *Files d'attente* (grandeurs aléatoires de base : instants d'arrivée des clients, durée du service) : optimiser le compromis entre le nombre de serveurs et la durée d'attente des clients.
- *"Fiabilité" = usure et renouvellement des équipements* (grandeurs aléatoires : instants des pannes, durée des réparations) : optimiser le compromis entre le coût des arrêts du matériel et le taux de renouvellement du matériel.
- *Gestion des stocks* (grandeurs aléatoires : instants des demandes de pièces, quantités demandées) : optimiser le compromis entre le coût de stockage et le coût des ruptures de stock.

3) Domaine combinatoire continu : la variable est réelle (non entière)

- *Programmation mathématique linéaire (ou non linéaire)* : par exemple, pour calculer le nombre de produits de plusieurs types fabriqués par la même machine pour maximiser un bénéfice avec des contraintes de fabrication et de vente.

Conclusion : **RO = discipline carrefour entre**

- ✓ *mathématiques* : algèbre de Boole, algèbre linéaire, théorie des graphes, probabilités, processus stochastiques.
- ✓ *informatique* : l'informatique est un outil pour la RO (algorithmes rapides) ; la RO est un outil pour l'informatique (problème de files d'attentes).
- ✓ *économie*.

III. Bref historique et bibliographie succincte

- La RO est une technique récente : Blackett (1940) pour optimiser l'emplacement des radars pour prévenir l'arrivée des bombardiers allemands.
- La plupart des outils mathématiques existaient déjà :

    - ✓ Pascal et Fermat (1654) : espérance mathématique
    - ✓ Monge (1776) : problème des déblais et remblais
    - ✓ Borel (1921-1925) : théorie mathématique des jeux
    - ✓ Erlang (1917) : théorie des files d'attente (réseau téléphonique)
    - ✓ Kantorovitch (1930-1940) : programmation linéaire pour planification
    - ✓ König (1936) : théorie des graphes.

- Gros développements depuis l'apparition des ordinateurs (1955-56). Il existe deux écoles : américaine et française (Robert Faure).

- *Bibliographie* :

- R. Faure , J-P. Boss , A. Le Garff, *La recherche opérationnelle*, Que sais-je n° 941 (PUF), 1974.
- V. Cohen, *La recherche opérationnelle*, Que sais-je n° 941 (PUF), 1995.
- R. Faure, B. Lemaire, C. Picouleau, *Précis de recherche opérationnelle : Méthodes et exercices*, Dunod, 5ème édition, 2004 (1ère édition 1979).
- A. Alj et R. Faure, *Guide de la recherche opérationnelle. Tome 1, les fondements*, Masson 1986.
- A. Alj et R. Faure, *Guide de la recherche opérationnelle. Tome 2, les applications*, Masson 1990.
- Roseaux, *Exercices et problèmes résolus en recherche opérationnelle*
  Tome 1, Graphes : leurs usages, leurs algorithmes, Dunod 2005 (1ère édition 1983).
  Tome 2, Phénomènes aléatoires en RO, Dunod 2004 (1ère édition 1983).
  Tome 3, Programmation linéaire et extensions, Dunod 2003 (1ère édition 1985).

IV. Plan du cours (12 h cours, 14 h TD, TAI)

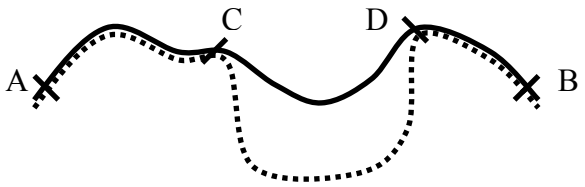
- Chapitre 1 : applications de la théorie des graphes
  - programmation dynamique pour problèmes de décision séquentielle
  - chemin optimal (voir le *cours sur les graphes*)
  - méthodes d'ordonnancement (MPM, PERT)
  - problème de flot maximal dans un réseau (Ford-Fulkerson)
  - problèmes d'affectation (méthode hongroise)
  - problèmes de transport (stepping -stone)
- Chapitre 2 : phénomènes aléatoires en recherche opérationnelle
  - files d'attente
- Chapitre 3 : programmation mathématique linéaire
  - méthode des tableaux (simplexe).

Chapitre 1. Applications de la théorie des graphes

I. Programmation dynamique

1) Généralités

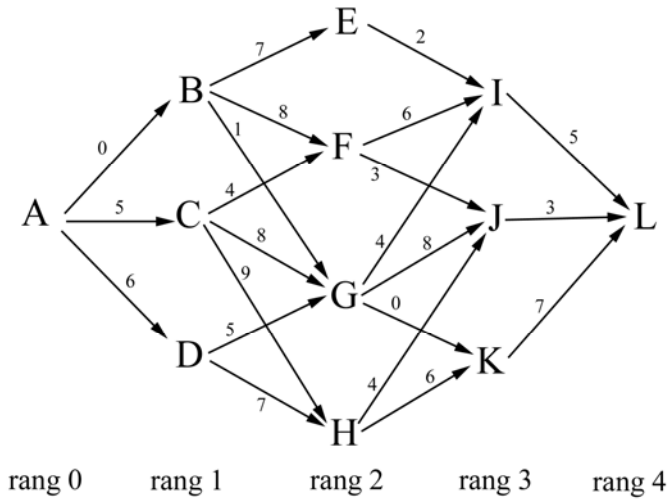
- *Principe* : "tout sous-chemin d'un chemin optimal est forcément optimal".



Dn : Soit un chemin optimal AB (trait plein). On suppose qu'il existe un sous-chemin CD en pointillés meilleur que le sous-chemin CD en trait plein → le chemin total en pointillés serait meilleur que le chemin initial optimal en trait plein, ce qui est contraire à l'hypothèse.

- *Ancien* (Fermat, Euler, Lagrange) mais mise en pratique récente : Masse (1944), Bellman (1950-1957)…
- *Intérêt* : - résoudre des problèmes de décisions séquentielles (prises l'une après l'autre).
  - atténuer le caractère combinatoire du problème (sous-chemins non optimaux non traités).
- *Application* aux problèmes **discrets** (graphes) ou continus, en univers **certain** ou aléatoire.

2) Exemple simple (Roseaux, tome 1 p. 61)



énoncé : Une voie de chemin de fer doit être construite entre les villes A et L. Trouver les villes intermédiaires pour que le coût de construction soit minimal (coût 0 = voie déjà construite).

remarques :
- pas de retour en arrière,
- les arcs vont tous d'un rang *i* à un rang *i*+1.

→ apparition de phases : à chaque phase i, on va choisir le meilleur chemin allant de A vers chacun des sommets de la phase i.

→ on construit progressivement (= dynamiquement) le chemin optimal de A à L par des décisions séquentielles.

- Algorithme. *Notations* : s_i la ville de niveau i : $s_0 = A$, $s_1 \in \{B, C, D\}$, $s_2 \in \{E, F, G, H\}$, $s_3 \in \{I, J, K\}$, $s_4 = L$

  $v_{i+1}(s_i, s_{i+1})$: valuation de s_i à s_{i+1}

  $f(s_i)$: coût minimal de construction de A à s_i

  Principe : - on calcule $f_1(s_1) = v_1(A, s_1)$ pour $s_1 \in \{B, C, D\}$

  - on calcule $f_2(s_2) = \text{Min}\,(f_1(s_1) + v_2(s_1, s_2))$ pour chaque $s_2 \in \{E, F, G, H\}$
    s_1 antécédent de s_2
  - de même : $f_3(s_3) = \text{Min}\,(f_2(s_2) + v_3(s_2, s_3))$ pour chaque $s_3 \in \{I, J, K\}$
    s_2 antécédent de s_3 ($f_3(s_3)$ ne dépend plus de s_1)

Intérêt de la méthode : - lecture de $f_i(s_i)$ dans la case mémoire allouée
- élimination des chemins non-optimaux.

 les sommets doivent être préalablement ordonnés par rang (de i à i+1).

3) Deuxième exemple : problème du sac à dos (Roseaux, tome 1 p. 71)

- *énoncé* : un randonneur veut remplir son sac à dos en maximisant la valeur nutritive totale du sac sans dépasser le poids de 15 kg. On dispose de 3 aliments avec les caractéristiques suivantes :

aliment	1	2	3
Poids p_i (kg)	6	3	9
Valeur nutritive c_i	15	10	35

Soit x_i le nombre de boites retenues pour l'aliment i. On cherche les x_i qui maximisent $c_1 x_1 + c_2 x_2 + c_3 x_3$ sous la contrainte de poids $p_1 x_1 + p_2 x_2 + p_3 x_3 \leq 15$.

- *Problème de programmation dynamique* (ou programmation mathématique linéaire, cf. chapitre 3) car on va faire un choix successif des valeurs de x_1 (phase 1), puis de x_2 (phase 2), puis de x_3 (phase 3).
- *Notation* : y_i = état du système (poids du sac) après la phase i ($y_0 = 0$)

	phase 1	*phase 2*	*phase 3*
	choix de x_1 ↓	choix de x_2 ↓	choix de x_3 ↓
$y_0 = 0$ →	y_1 →	y_2 →	y_3
retour élémentaire	$c_1 x_1$	$c_2 x_2$	$c_3 x_3$
état du sac	$y_1 = p_1 x_1$	$y_2 = y_1 + p_2 x_2$	$y_3 = y_2 + p_3 x_3$

- *But* : $\text{Max}\,(c_1x_1 + c_2x_2 + c_3x_3) = \text{Max} \sum_i \frac{c_i}{p_i}(y_i - y_{i-1})$ de la forme générale $\sum_i v_i(y_{i-1} - y_i)$

 construction du graphe : pour Oy il faut prendre l'état du système (ici le poids y_i) : le choix de y_i n'est pas toujours simple.

- Algorithme. *Notations* : $f_i(y_i)$: valeur nutritive maximale du sac après le choix de x_i

Principe : - déterminer les états (poids) de la phase 1 (y_1 = 0, 6, 12) avec les arcs de y_0 à y_1 et leurs valuations (= construire le début du graphe)
- on calcule alors $f_1(y_1) = c_1/p_1\,(y_1\text{-}y_0)$ pour y_1 = 0, 6, 12
  ⟶ à stocker en mémoire
- déterminer les valeurs de y_2 (0, 3, 6, 9, 12, 15) + les arcs de y_1 vers y_2 + leurs valuations $v_2(y_1, y_2)$ (= construction de la suite du graphe)
- on calcule alors les $f_2(y_2)$ = Max $\{f_1(y_1) + v_2(y_1, y_2)\}$ pour chaque y_2
  y_1 antécédent de y_2
- et ainsi de suite.

II. Problèmes d'ordonnancement

1) Généralités

- *But* : il s'agit d'ordonner dans le temps un ensemble d'opérations (= tâches) contribuant à la réalisation d'un projet, ces opérations étant soumises à des contraintes.

- *Contraintes* : elles sont de 3 types

  - potentielles : - contrainte de succession (la tâche *a* doit se dérouler avant la tâche *b*).
    - contrainte de date (la tâche ne peut commencer avant le 15/3, la tâche doit se terminer avant le 3/4).
  - disjonctives : imposent la non réalisation simultanée de 2 tâches (ex : si on a une seule machine).
  - cumulatives : prennent en compte les limites des facteurs de production (hommes, machines, moyens financiers).

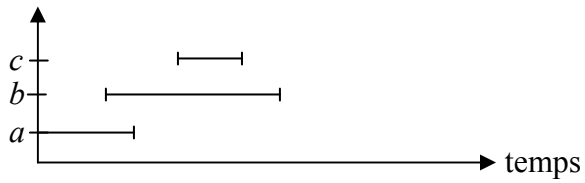
  Pour le cours on se limite aux contraintes de succession.

- *Méthodes d'ordonnancement*

au début du projet (10%) :
- analyse du projet (découpage en tâches avec les durées et les contraintes).
- ordonner les tâches en respectant les contraintes → graphe.
- trouver le chemin critique (il donne la durée minimale du projet).
- calculer les dates de démarrage au plus tôt et au plus tard des tâches → marges de réalisation.

suivi du projet (90%) :
- réactualisation permanente du graphe, des dates, des marges… pour prendre en compte les aléas extérieurs.

3 méthodes : 1) *planning à barres* (diagramme de Gantt), seule méthode avant 1958.



→ bonne visualisation du travail mais limité pour l'analyse du problème.

2) *méthode CPM* (Critical Path Method) → PERT (Program Evaluation and Review Technique; 1958). ex : projet Apollo (méthode US)
(méthode lourde pour les calculs mais graphe utile)

3) *méthode MPM* (Méthode des Potentiels METRA), Bernard Roy (1957-1959).
ex : armement du paquebot France.
(méthode simple pour les calculs mais graphe peu utile)

- *pour le cours* : les deux méthodes : - MPM (méthodes des potentiels - tâches)
  - PERT (méthode des potentiels - étapes)
  seront présentées sur le même exemple suivant (construction d'un barrage).

	opération	durée (mois)	tâches pré requises	recherche des rangs
a	construction des voies d'accès	4	-	-
b	travaux de terrassement	6	*a*	*a*
c	construction des bâtiments administratifs	4	-	-
d	commande de matériel électrique	12	-	-
e	construction de la centrale	10	*b, c, d*	*b, c, d*
f	construction du barrage	24	*b, c*	*b, c*
g	installation des galeries et conduites forcées	7	*a*	*a*
h	montage des machines	10	*e, g*	*e, g*
i	essais de fonctionnement	3	*f, h*	*f, h*

2) Méthode des potentiels - tâches (MPM)

- *Principe* : - chaque opération est un sommet (ou potentiel) du graphe (sommets connus au départ).
  - chaque arc représente une contrainte.

  ex : $a \xrightarrow{4} b$: la tache *b* ne peut pas commencer avant la date de début de $a + 4$ mois (pour une contrainte de succession : 4 = durée de *a*).

- *Construction du graphe* : c'est une étape facultative car un graphe MPM est moins parlant qu'un graphe PERT car on n'y lit pas le temps absolu.

  $\Rightarrow$ souvent : - on fait les calculs de dates et marges avec MPM (sans graphe).
  - on construit un graphe PERT.

- *Méthode de construction*

  1) il faut ordonner les sommets à partir de leur rang :

  - on part du tableau des prédécesseurs : rang 0 pour les tâches qui n'ont pas de prédécesseurs.
  - on élimine dans le tableau Préd(*x*) les tâches *a*, *c*, *d* $\rightarrow$ celles qui n'ont plus de prédécesseur sont de rang 1.
  - on itère pour avoir les sommets de rang 2, puis de rang 3, etc.

  2) on note les sommets par :

T_x	T_x^*
x	

  T_x est la date de début au plus tôt
  T_x^* est la date de début au plus tard

  et on rajoute un sommet Fin.

  3) on construit les arcs à partir du tableau Préd(*x*) et on rajoute l'arc $i \rightarrow F$.

- *Dates de début au plus tôt* T_x

  2 méthodes : 1) en utilisant le graphe : méthode de Bellman

  - pour le rang 0 : $T_a = T_c = T_d = 0$.
  - pour les autres tâches *x* : T_x = durée du chemin le plus long d'un sommet de rang 0 à *x*.

  $\rightarrow$ la durée minimale du projet est 37 mois.
  $\rightarrow$ en partant du sommet Fin et en remontant le graphe jusqu'au début on obtient

le chemin critique : $a \xrightarrow{4} b \xrightarrow{6} f \xrightarrow{24} i \xrightarrow{3}$ Fin.

→ les tâches critiques sont : *a*, *b*, *f*, *i*.

↳ ne peuvent être retardées ou rallongées sans augmenter la durée du projet.

2) sans utiliser le graphe : méthode des tableaux de Bellman

→ on remplit chaque sous-colonne de droite de *x* avec "prédécesseur de *x* + sa durée".
→ on ajoute une tâche fictive *D*(ébut) avant *a*, *c*, *d* (de durée 0).
→ algorithme : quand la colonne *j* est complète, on calcule $T_j = \max (T_i + d_i)$
où d_i est la durée de *i*. *i* préd. de *j*

0	*a*	**4**	*b*	**0**	*c*	**0**	*d*	**12**	*e*	**10**	*f*	**4**	*g*	**22**	*h*	**34**	*i*	**37**	*Fin*
0	*D*:0	0	*a*:4	0	*D*:0	0	*D*:0	4	*b*:6	4	*b*:6	0	*a*:4	12	*e*:10	10	*f*:24	34	*i*:3
								0	*c*:4	0	*c*:4			4	*g*:7	22	*h*:10		
								0	*d*:12										

- *Dates de début au plus tard* T_x^*

- but : voir si on peut retarder le démarrage d'une tâche sans rallonger la durée du projet

- 2 méthodes : 1) avec le graphe : $T_F^* = 37$ et on remonte en arrière $T_i^* = 37 - 3 = 34$
avec la relation : $T_x^* = \min (T_y^* - d_x)$
y successeur de *x*
2) sans le graphe : méthode des tableaux de Bellman

→ on remplit chaque sous-colonne de droite de *x* avec "successeur de *x* + durée de *x*"
→ on ajoute en colonne *i* la tâche Fin + durée de *i* = 3
→ algorithme : quand la colonne *j* est complète, on calcule $T_x^* = \min (T_y^* - d_x)$
y successeur de *x*

0	*D*	**0**	*a*	**4**	*b*	**6**	*c*	**2**	*d*	**14**	*e*	**10**	*f*	**17**	*g*	**24**	*h*	**34**	*i*
0	*a*:0	4	*b*:4	14	*e*:6	14	*e*:4	14	*e*:12	24	*h*:10	34	*i*:24	24	*h*:7	34	*i*:10	37	*F*:3
6	*c*:0	17	*g*:4	10	*f*:6	10	*f*:4												
2	*d*:0																		

- *Marges de réalisation* : il existe 3 types de marge pour chaque tâche.

- marge totale : retard maximum que l'on peut apporter au démarrage d'une tâche sans changer les dates de début au plus tard des tâches suivantes (= sans retarder la fin du projet) ; pour la tâche *i* : $M_i = T_i^* - T_i$.

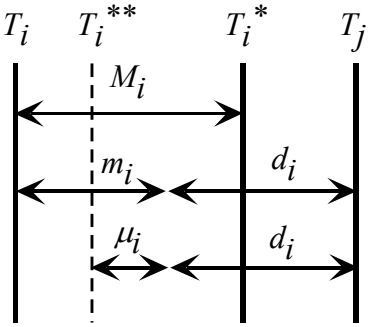
- marge libre : retard maximum que l'on peut attribuer au démarrage d'une tâche sans changer les dates de début au plus tôt des tâches suivantes (= sans changer la suite de l'ordonnancement) ; pour la tâche *i* : $m_i = \min (T_j - T_i - d_i)$.
($i \xrightarrow{d_i} j$) *j* successeur de *i*

- marge certaine (indépendante) : retard maximum que l'on peut attribuer au démarrage d'une tâche sans changer les dates de début au plus tôt des tâches suivantes, alors que les tâches précédentes ont démarré au plus tard ; pour la tâche *i* :

$T_i^{**} = \max_{k \text{ préd. de } i} (T_k^* + d_k)$ où T_i^{**} = date de début au plus tôt de i quand les tâches précédentes démarrent au plus tard ($T_i \leq T_i^{**} \leq T_i^*$).

$\mu_i = \min_{j \text{ successeur de } i} (T_j - T_i^{**} - d_i)$ ou 0 si ce minimum est négatif.

- schéma des marges :



- remarques : 1) $0 \leq \mu_i \leq m_i \leq M_i$
2) si $M_i = 0 \Rightarrow m_i = \mu_i = 0$ et si $m_i = 0 \Rightarrow \mu_i = 0$.

- valeurs des marges sur l'exemple :

tâche	a	b	c	d	e	f	g	h	i
T_i	0	4	0	0	12	10	4	22	34
T_i^*	0	4	6	2	14	10	17	24	34
M_i	0	0	6	2	2	0	13	2	0
m_i	0	0	6	0	0	0	11	2	0
T_i^{**}	-	-	0	-	-	-	4	24	-
μ_i	0	0	6	0	0	0	11	0	0

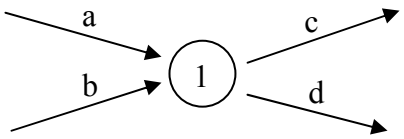
- intérêt des marges : on peut chercher à rallonger la durée d'une tâche pour diminuer son coût, et ainsi avoir 2 critères d'optimisation : durée et coût.

3) Méthode des potentiels - étapes (PERT)

- *Principe* : - chaque tâche correspond à un arc du graphe, de longueur = durée de la tâche.
  - chaque sommet correspond à une étape (un moment) du projet signifiant :
    - toutes les tâches y arrivant sont terminées,
    - toutes les tâches en partant peuvent commencer

⚠ les sommets sont inconnus au départ.

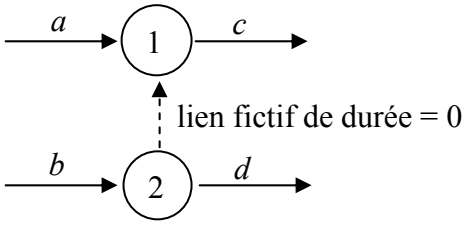
- exemple :



à l'étape 1 : a et b sont terminées, c et d peuvent commencer.

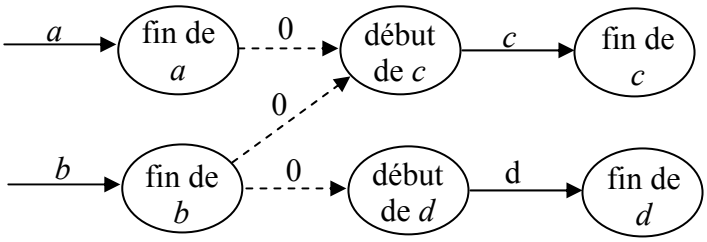
- *Tâche fictive (durée 0)* : pour le tableau des prédécesseurs suivants, *d* n'a pas *a* comme prédécesseur, le schéma précédant n'est donc pas représentatif du problème, il faut donc créer une tâche fictive.

tâche x	Préd(x)
a	
b	
c	*a, b*
d	*b*



- règle : puisque *b* apparaît 2 fois dans la colonne Préd(x), il faut démarrer *c* de *a* et pas de *b*.

- remarque : cette façon de construire les arcs *c* et *d* introduit un nombre limité de sommets ; on obtient un schéma PERT équivalent en introduisant systématiquement un sommet "début" pour la construction de chaque arc :



Problème :

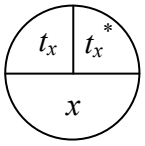
Ce schéma a beaucoup d'arcs fictifs, il faudra le simplifier pour construire un graphe PERT "minimum".

- remarque : il existe des études essayant de minimiser le nombre d'étapes/d'arcs fictifs pour obtenir un graphe PERT de taille raisonnable.

 la notion d'étape n'est pas unique : les étapes découlent de la construction du graphe.

- *Construction du graphe* :

- notation des sommets (étapes) :



où t_x = date au plus tôt de l'étape *x*
et t_x^* = date au plus tard de l'étape *x*.

 t_x et t_x^* ne sont pas des dates de début au plus tôt/tard, car une étape n'a pas de durée.

- on ajoute toujours une étape de début *D* et une étape de fin *F*.

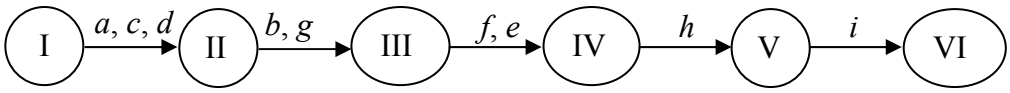
- 2 méthodes pour la construction du graphe :

1) pour chaque tâche, on introduit systématiquement une étape "début de tâche" et une étape "fin de tâche" (facile en machine) ; on a alors beaucoup d'arcs fictifs dont il faut ensuite réduire le nombre (difficile en machine).

2) pour chaque tâche, on évite l'introduction de sommets "début de tâche" (et donc l'introduction d'arcs fictifs) en appliquant la règle ci-dessus.

- <u>construction</u> : sur l'exemple du barrage

1) *recherche des rangs des tâches* (voir méthode MPM) ; on obtient alors 6 étapes :



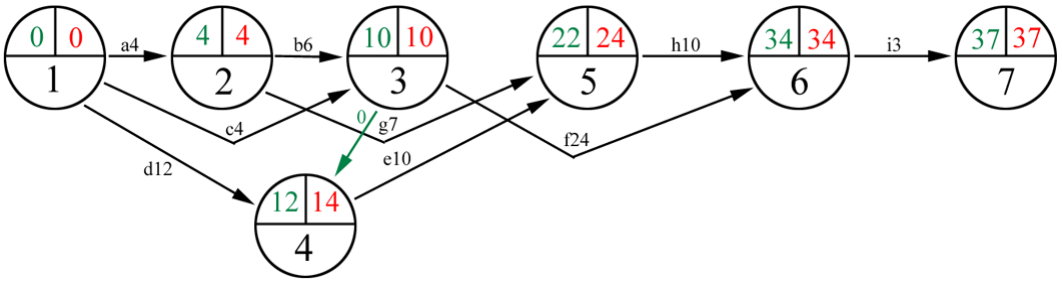
ce premier graphe PERT est incorrect :

- les contraintes ne sont pas toutes correctes (ici, *c* est avant *g*).
- il est interdit de faire terminer à une étape commune deux tâches ayant démarré à la même étape (même si ces deux tâches ont même durée car en différentiant leur début ou fin, on garde un degré de liberté pour traiter d'éventuels retards).

2) *second graphe PERT* : on construit les arcs dans l'ordre des rangs (*a*, *c*, *d*, puis *b*, *g*…) en introduisant si nécessaire des arcs fictifs.

- *e* ne peut pas partir de *b* ou *c* (car *b* et *c* sont dans *P*(*f*)) sinon, une contrainte fausse apparaît (*d* est avant *f*).
- à la fin, on ne laisse pas de sommet "en l'air", on les relie à l'étape "fin" par un arc fictif.

3) *réduction du graphe* : on élimine les arcs fictifs inutiles en fusionnant certaines étapes.



- <u>intérêt du graphe PERT</u> : ce graphe permet de visualiser facilement le temps absolu.

- *Dates au plus tôt* t_x

- la date au plus tôt de l'étape *x* est la valeur du chemin le plus long du début du projet à l'étape *x* : $t_x = \max_{y \text{ prédécesseur de } x} (t_y + d_{yx})$ où d_{yx} est la durée de la tâche entre *y* et *x*.

- → durée = 37 mois
→ chemin critique : $1 \xrightarrow{a} 2 \xrightarrow{b} 3 \xrightarrow{f} 6 \xrightarrow{i} 7$ et tâches critiques : *a*, *b*, *f*, *i*.
→ date de début au plus tôt T_i de la tâche *i* partant de l'étape *x* : $T_i = t_x$.

- *Dates au plus tard* t_x^*

  - la date au plus tard de l'étape x est la date limite de réalisation t_x^* telle que la date au plus tôt de la fin des travaux ne soit pas retardée :

  - on part de $t_7^* = 37$ et on remonte le graphe en appliquant la relation : $t_x^* = \min_{y \text{ successeur de } x} (t_y^* - d_{xy})$

    où d_{xy} est la durée de la tâche entre x et y.

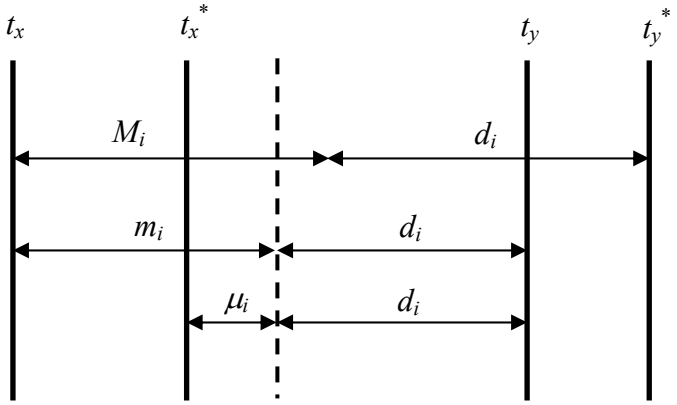
  - date de début au plus tard T_i^* de la tâche i (durée d_i) arrivant sur l'étape x : $T_i^* = t_x^* - d_i$.

- *Marges de réalisation* : mêmes définitions des marges que pour la MPM.

  - marge totale de la tâche i : $M_i = T_i^* - T_i$
  - marge libre de la tâche i : $m_i = t_y - t_x - d_i$ où x et y sont les étapes avant et après i.
  - marge certaine de la tâche i : $\mu_i = t_y - t_x^* - d_i$ (0 si négatif)

  - schéma des marges :

  $(x \xrightarrow{i} y)$

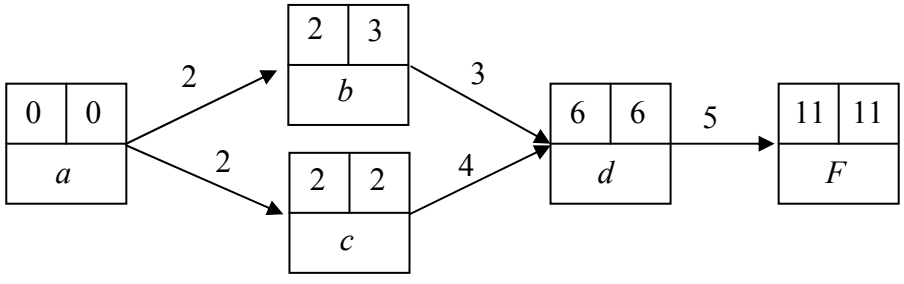


- *Non-unicité du graphe PERT - calcul des marges des sommets fictifs*

  - sur l'exemple suivant, les graphes MPM et PERT sont construits et les marges des tâches sont calculées

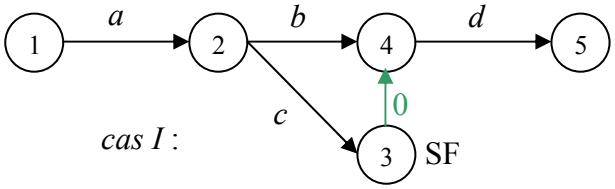
x	$P(x)$	durée
a	-	2
b	a	3
c	a	4
d	b, c	5

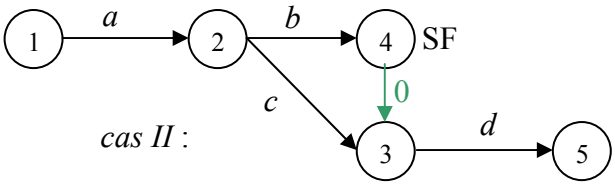
  - méthode MPM : graphe et tableau des marges

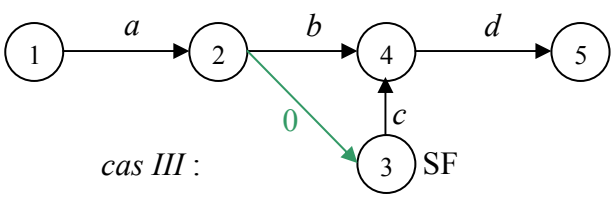


x	a	b	c	d
M_i	0	1	0	0
m_i	0	1	0	0
μ_i	0	1	0	0

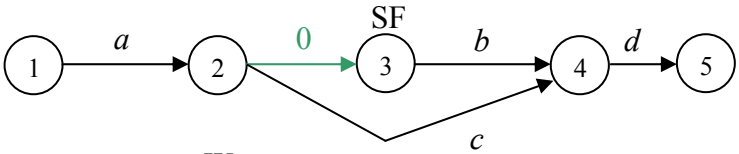
  - méthode PERT : 4 graphes équivalents sont possibles







cas III :



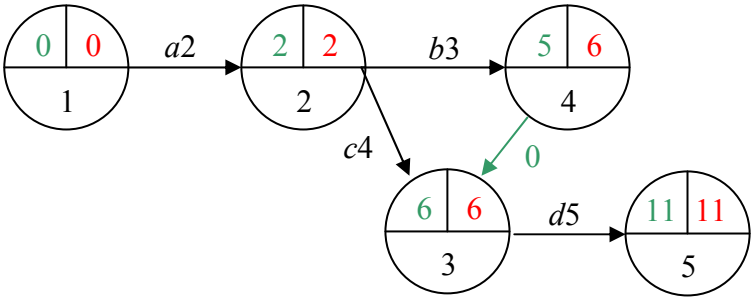
cas IV :

- <u>remarque</u> : dans chacun des cas, il y a un arc fictif et un <u>sommet fictif</u> (sommet pour lequel soit un ou des arcs réels y arrivent et aucun arc réel n'en repart, soit un ou des arcs réels en partent et aucun arc réel n'y arrive).

⚠ le calcul des marges des tâches pose parfois problème s'il y a des sommets fictifs (SF).

1) les cas I et III ne posent pas de problème (pour la raison donnée ci-dessous)

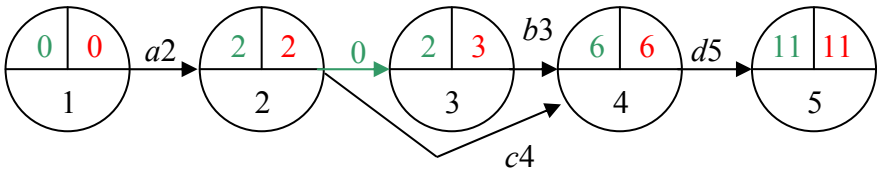
2) <u>cas II</u> :



x	a	b	c	d
T_i	0	2	2	6
T_i^*	0	3	2	6
M_i	0	1	0	0
m_i	0	**0**	0	0
μ_i	0	erreur due au SF 4	0	0

→ règle de calcul des marges pour le sommet fictif 4 : $t_{SF4} = t_{SR3} = 6$,
d'où $m_b = 6 - 3 - t_2 = 1$ et $\mu_b = 6 - 3 - t_2^* = 1$.

3) <u>cas IV</u> :



x	a	b	c	d
T_i	0	2	2	6
T_i^*	0	3	2	6
M_i	0	1	0	0
m_i	0	1	0	0
μ_i	0	**0**	0	0
		erreur due au SF 3		

→ règle de calcul des marges pour le sommet fictif 3 : $t_{SF3}^* = t_{SR2}^* = 2$,
d'où $\mu_b = 6 - 3 - t_3^* = 1$.

- remarques : 1) ces modifications des dates t_x ou t_x^* pour les calculs de marges ne modifient pas les valeurs calculées de T_i, T_i^* et M_i.
2) les cas I et III ne posent pas de problèmes car les règles de calcul des marges se trouvent appliquées d'office.

4) Comparaison rapide des deux méthodes MPM et PERT

MPM	PERT
- graphe unique	- graphes multiples à cause de la définition des étapes et des tâches fictives
- graphe peu utile	- graphe utile car on peut visualiser les dates absolues
- suivi de l'ordonnancement simple (modification d'arcs et de valuations)	- suivi plus difficile car il faut supprimer des étapes, en rajouter d'autres
- programmation simple	- programmation difficile (construction du graphe)

- remarque : les 2 méthodes sont équivalentes pour la prise en compte du caractère aléatoire de la durée des tâches : pour éviter d'avoir à définir la densité de probabilité de la variable aléatoire "durée" qui est une variable continue, on utilise la loi de probabilité discrète suivante pour la durée de chaque tâche i :

a_i = durée minimale, probabilité = 1/6
b_i = durée maximale, probabilité = 1/6
c_i = durée vraisemblable, probabilité = 4/6

$\rightarrow$ durée moyenne de $i = a_i/6 + b_i/6 + 4\ c_i/6$ et c'est cette valeur moyenne qu'on utilise dans les algorithmes pour la durée de i.

Conclusion : - la méthode PERT subsiste, car c'est la méthode la plus connue (la majorité des logiciels sont américains) et car le graphe est très utile pour visualiser l'avancement du projet.
- la méthode MPM est surtout une méthode de calcul (dates et marges) et on termine par un graphe PERT ou un diagramme de Gantt pour visualiser les résultats.

IV. Problèmes de flot maximal dans un réseau

1) Généralités

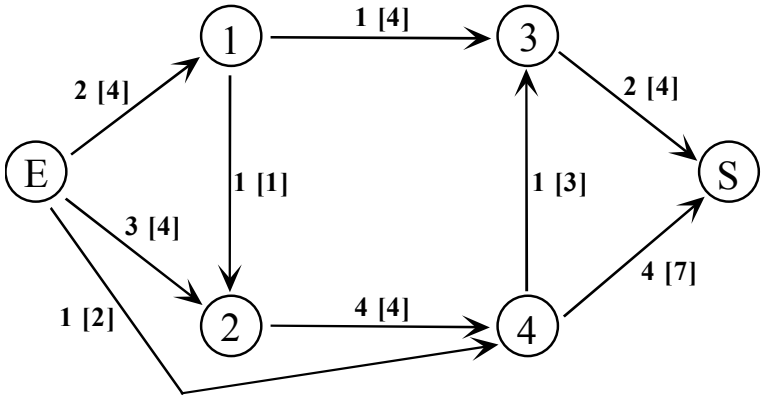
- *Réseau de transport* : graphe sans boucle, avec une entrée unique E et une sortie unique S, valué par des nombres représentant des capacités de transport :
  - débit maximum d'information dans un réseau informatique,
  - débit maximum de voitures dans un réseau routier,
  - débit maximum d'eau dans un réseau de canalisations d'eau…

- *Exemple* :

  - à chaque arc (i, j) on associe :
    - sa capacité $C(i, j)$,
    - un flot circulant $\varphi(i, j)$.

  - contrainte de capacité :

    $\varphi(i, j) \le C(i, j)$



- *Définitions – relations* :

  - flot sur un sommet i : $\varphi(i) = \sum_{x \text{ préd.de } i} \varphi(x,i)$

  - contrainte : pas de perte ou d'accumulation de "marchandises" à un sommet

    $\varphi(i) = \sum_{x \text{ préd.de } i} \varphi(x,i) = \sum_{y \text{ succ.de } i} \varphi(i,y)$ (flot arrivant en i = flot partant de i)

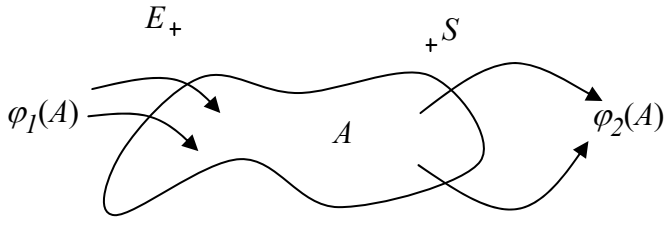
  - flot partant de E : $\varphi(E) = \sum_{y \text{ succ.de } E} \varphi(E,y)$, flot arrivant à S : $\varphi(S) = \sum_{x \text{ préd.de } S} \varphi(x,S)$

  - contrainte : pas de perte ou d'accumulation dans le réseau : $\varphi(E) = \varphi(S)$

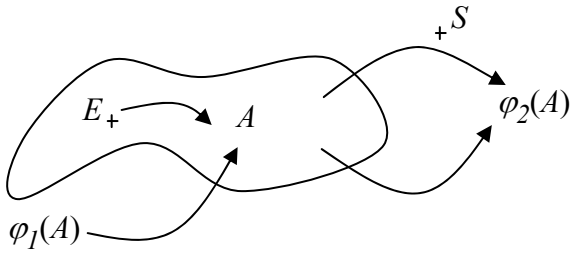
  - pour un ensemble A de sommets : - flot entrant dans A : $\varphi_1(A) = \sum_{i \notin A,\, j \in A} \varphi(i,j)$

    - flot sortant de A : $\varphi_2(A) = \sum_{i \in A,\, j \notin A} \varphi(i,j)$

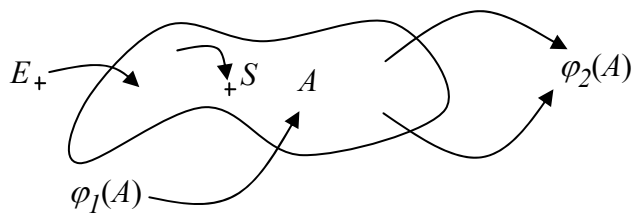
  - relations entre $\varphi_1(A)$ et $\varphi_2(A)$: 4 cas



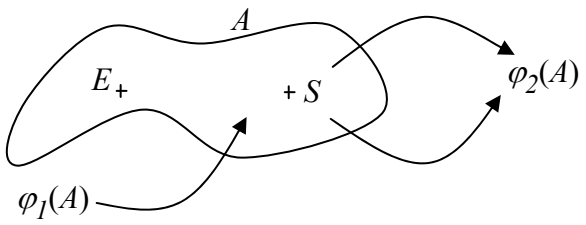
si $E \notin A$ et $S \notin A$: $\varphi_2(A) = \varphi_1(A)$



si $E \in A$ et $S \notin A$: $\varphi_2(A) = \varphi_1(A) + \varphi(E)$



si $E \notin A$ et $S \in A$: $\varphi_1(A) = \varphi_2(A) + \varphi(S)$



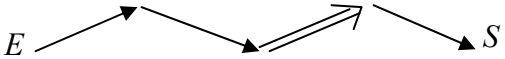
si $E \in A$ et $S \in A$: $\varphi_2(A) = \varphi_1(A)$

- *Problème* : recherche de la quantité maximale de "marchandises" transportables de E à S en calculant le flot devant circuler dans chaque arc.

  Deux étapes : 1) recherche d'un flot complet (tous les chemins de E à S sont saturés)
  2) recherche d'un flot maximal.

2) <u>Recherche d'un flot complet</u>

- *Définitions* : - un arc (i, j) est dit saturé si $\varphi(i, j) = C(i, j)$ et on notera $i \overset{5[5]}{\Longrightarrow} j$
  - capacité résiduelle d'un arc : $C(i, j) - \varphi(i, j)$
  - un chemin est saturé si au moins un de ses arcs est saturé.



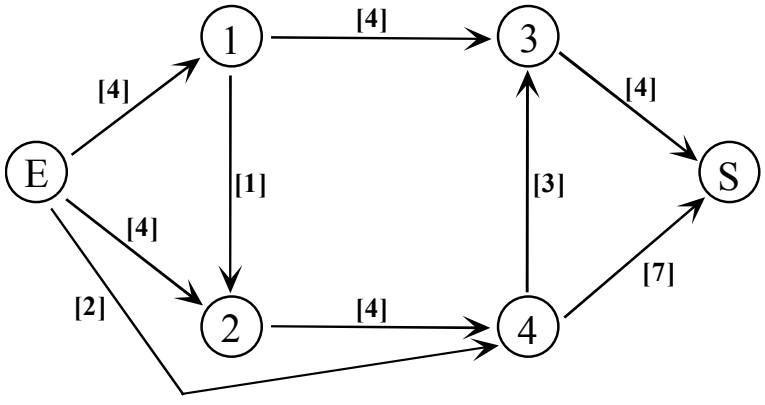
  - un flot est complet si tous les chemins de E à S sont saturés (il n'est pas forcément maximal).

- *2 méthodes de recherche* :

a) <u>résolution "manuelle"</u>

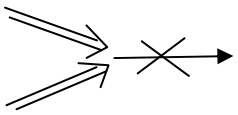
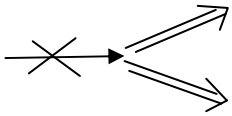
Algorithme : 1) partir d'un flot au jugé (en informatique : flot = 0 sur chaque arc) et marquer d'un double trait les arcs saturés.
2) rechercher un chemin non saturé.
3) saturer ce chemin en ajoutant au flot de chacun des arcs, le minimum des capacités résiduelles $C(i, j) - \varphi(i, j)$ de ces arcs.
4) retour à 2) jusqu'à ce que tous les chemins soient saturés.

Exemple :



b) <u>algorithme de Bloch</u> (1967)

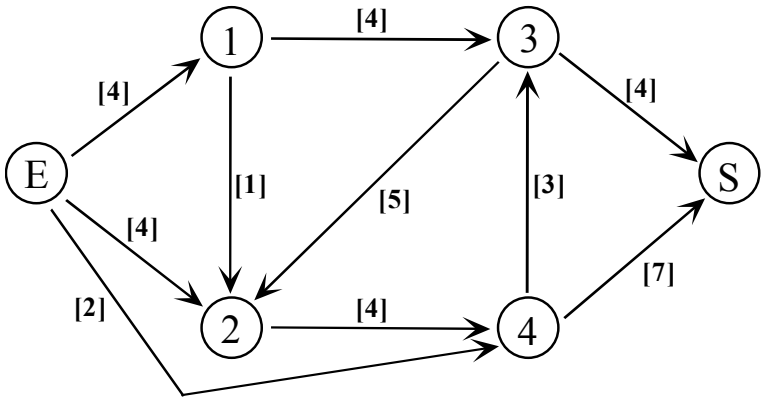
But : pour la recherche de chemins non saturés, on élimine progressivement les arcs saturés, les arcs bloqués (on ne peut plus augmenter leur flot) et les circuits (on met le flot 0 sur l'arc de plus petite capacité du circuit pour éliminer celui-ci)

exemples d'arcs bloqués :  et 

Algorithme : 1) dresser la liste des arcs dans l'ordre de la numérotation avec leur capacité.
2) choisir dans la liste des arcs praticables (non saturés, non bloqués) le premier arc de plus petite capacité résiduelle ρ_0.
3) déterminer un chemin praticable de E à S, passant par l'arc précédent, en prenant les sommets dans l'ordre de la numérotation.
- si le chemin est élémentaire (sans répétition de sommets), on augmente le flot des arcs de ce chemin de ρ_0 et on passe à 4).
- si le chemin n'est pas élémentaire (il existe un circuit), on bloque les arcs de capacité la plus faible du circuit mis en évidence et retour à 2).

4) pour chaque sommet du chemin choisi, vérifier s'il existe des arcs à bloquer, les bloquer et retour à 2).

Exemple :



3) Recherche d'un flot maximal à partir d'un flot complet (algorithme de Ford-Fulkerson, 1956)

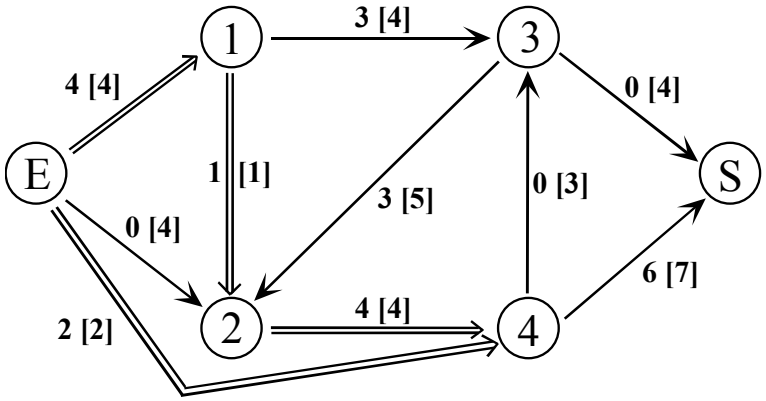
- *Algorithme* :

1) Marquer l'entrée E du réseau (+).
2) Marquer l'extrémité terminale d'un arc (i, j) non saturé dont l'extrémité initiale est marquée (mettre $+i$ à côté de j) : on pourra augmenter le flot sur cet arc.
3) Marquer l'extrémité initiale d'un arc (k, l) transportant un flot non nul et dont l'extrémité finale est marquée (mettre $-l$ à côté de k) : on pourra diminuer le flot sur cet arc.
4) Deux possibilités :

– si on n'arrive pas à marquer la sortie S, alors le flot est maximal.
– si on arrive à marquer la sortie :
* prendre une chaîne de sommets marqués de E à S ;
* calculer $d_1 = \min [C(i,j) - \varphi(i,j)]$
arcs (i,j) parcourus dans le sens des flèches
(on peut augmenter le flot sur ces arcs d'une quantité d_1) ;
* calculer $d_2 = \min [\varphi(i,j)]$
arcs (i,j) parcourus en sens opposé des flèches
(on peut diminuer le flot sur ces arcs d'une quantité d_2) ;
* augmenter de $d = \min (d_1, d_2)$ le flot des arcs parcourus dans le sens des flèches ;
* diminuer de $d = \min (d_1, d_2)$ le flot des arcs parcourus dans le sens opposé des flèches ;
* recommencer à 1) jusqu'à ce que la sortie ne puisse plus être marquée.

Exemple :

(en partant du flot complet obtenu avec l'algorithme de Bloch)



- *Optimalité du flot obtenu* :

  - <u>coupe</u> : soit L un ensemble de sommets tel que $E \in L$ et $S \notin L$

  - la coupe associée à L est l'ensemble des arcs (i, j) sortant de L : $i \in L$ et $j \notin L$
  - sa capacité est la somme des capacités de ses arcs : $C(L) = \sum_{i \in L,\, j \notin L} C(i, j)$
  - propriété : $\varphi(E) \leq C(L)$, quel que soit L (ce qui est créé en E doit sortir de L)

  - <u>flot maximal</u> : soit L_0 l'ensemble des sommets marqués ($S \notin L_0$ car le flot est maximal)

  - les arcs sortant de L_0 sont saturés (sinon le sommet suivant appartiendrait à L_0) : $\varphi_2(L_0) = C(L_0)$.
  - les arcs entrant dans L_0 sont à flot nul (sinon le sommet de départ appartiendrait à L_0) : $\varphi_1(L_0) = 0$.
  - or $\varphi_2(L_0) = \varphi_1(L_0) + \varphi(E)$ puisque $E \in L_0$ et $S \notin L_0$
  - donc $\varphi(E) = \varphi(S) = \varphi_2(L_0) - \varphi_1(L_0) = C(L_0)$ est maximal (car $\varphi(E) \leq C(L), \forall\, L$)

  - <u>complexité de l'algorithme</u> :

  - complexité de l'ordre de n^5 où n est le nombre de sommets ;
  - des algorithmes plus rapides existent (n^4 pour Dinic, n^3 pour Karsanov), ils font appel à la notion de graphe d'écart.

4) <u>Application classique</u>

- *Enoncé*

Trois dépôts A, B, C disposent de 30, 20 et 45 tonnes de marchandises ; quatre destinations D, E, F, G en demandent des quantités respectives de 10, 25, 20 et 25 tonnes. Des camions offrent les capacités de transport ci-contre. Etablir le meilleur plan de transport.

	D	*E*	*F*	*G*
A	5	5	-	20
B	-	20	10	-
C	10	5	10	10

- *Solution* : introduire une entrée et une sortie pour obtenir un réseau de transport ; la recherche d'un flot maximal donne une quantité maximale transportée de 75 tonnes de marchandises.

V. Problèmes d'affectation

1) Problème posé

- *Exemple* (R. Faure) :

Une administration veut affecter $n = 5$ personnes "A, B, C, D, E" aux postes "a, b, c, d, e" en maximisant la satisfaction générale. Les classements des postes sont donnés par la matrice c_{ij} de taille $n \times n$ (tableau I). c_{ij} (ligne i, colonne j) donne le classement du poste j par la personne i.

	a	b	c	d	e
A	1	2	3	4	5
B	1	4	2	5	3
C	3	2	1	5	4
D	1	2	3	5	4
E	2	1	4	3	5

tableau I

Exemple d'affectation :
A-a, B-c, C-b, D-e, E-d pour un coût total de 1 + 2 + 2 + 4 + 3 = 12.

- *But* : le problème consiste à choisir une case et une seule par ligne et par colonne de façon à minimiser la somme des chiffres des $n = 5$ cases choisies.

- *Formulation mathématique* : (on pose $x_{ij} = 1$ si la personne i obtient le poste j, 0 sinon)

On cherche à déterminer les x_{ij} pour minimiser le coût total $\sum_{i=1}^{n}\sum_{j=1}^{n} c_{ij}\, x_{ij}$ sous les contraintes $\sum_{i=1}^{n} x_{ij} = 1$ (le poste j est affecté à une seule personne) et $\sum_{j=1}^{n} x_{ij} = 1$ (la personne i est affectée à un seul poste).

- *Remarque* : si les personnes avaient noté les postes par une note n_{ij} de 0 (mauvais poste) à 20 (poste excellent), il faudrait maximiser $\sum_{i=1}^{n}\sum_{j=1}^{n} n_{ij}\, x_{ij}$ sous les mêmes contraintes. On revient à un problème de minimisation (et on peut alors appliquer la méthode hongroise) en remplaçant chaque note n_{ij} par la note $n'_{ij} = 20 - n_{ij}$.

2) Résolution par la méthode hongroise (Kuhn, Egervary, König) : 3 phases

- *Phase 1* : *obtention initiale de zéros*

algorithme : soustraire ligne par ligne, puis colonne par colonne le plus petit élément de la ligne ou de la colonne.
$\Rightarrow$ on obtient alors au moins un zéro par ligne et par colonne dans le tableau II.

théorème : 1) la solution optimale x_{ij} pour le tableau II est la même que celle du tableau I ;
2) si on retire u_i à la ligne i et v_j à la colonne j, alors :

$$\text{coût minimal (tableau II)} = \text{coût minimal (tableau I)} - \sum_{i=1}^{n} u_i - \sum_{j=1}^{n} v_j$$

exemple :

	a	b	c	d	e
A	0	1	2	1	2
B	0	3	1	2	0
C	2	1	0	2	1
D	0	1	2	2	1
E	1	0	3	0	2

tableau II

coût minimal (tableau II)
= coût minimal (tableau I) - 9

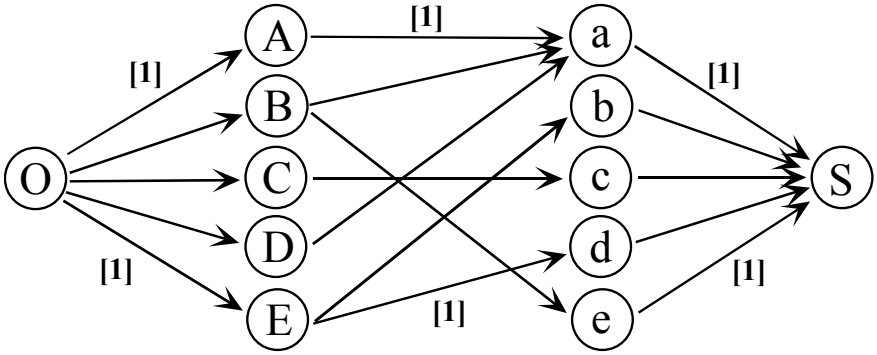
- *Phase 2* : *recherche d'une solution de coût nul* (un zéro par ligne et par colonne)

algorithme : 1) prendre la ligne contenant le moins de zéros ;
2) encadrer le premier zéro de cette ligne et barrer les autres zéros de la ligne et de la colonne du zéro encadré ;
3) retour à 1 jusqu'à impossibilité d'encadrer un zéro.

exemple : - on n'obtient pas 5 zéros encadrés, la solution n'est pas optimale ;
- coût minimal obtenu = $4 \times 0 + 2 = 2$ (tableau II), soit $2 + 9 = 11$ pour le tableau I.

vérification : on peut vérifier par l'algorithme de Ford - Fulkerson que le nombre de zéros encadrés (4) est maximal.

- on crée le réseau de transport ci-contre ;
- tous les arcs ont une capacité de 1 ;
- pour les arcs allant des personnes aux postes, on met le flot à 1 si le zéro est encadré, à 0 s'il est barré ;
- on complète les flots sur les autres arcs ;



- impossibilité de marquer la sortie par l'algorithme de Ford - Fulkerson.

- *Phase 3* : *recherche d'une solution optimale*

algorithme : 1) procédure de marquage des lignes et des colonnes :

a) marquer d'une croix les lignes ne contenant aucun 0 encadré (s'il n'y en a pas : FIN) ;
b) marquer d'une croix toute colonne qui a un 0 barré sur une ligne marquée ;
c) marquer d'une croix toute ligne qui a un 0 encadré dans une colonne marquée ;
d) retour à b et c jusqu'à ce qu'il n'y ait plus de ligne ou de colonne à marquer ;
(si toutes les colonnes sont marquées : FIN, la solution est optimale).

remarque : les lignes et colonnes marquées correspondent aux seuls sommets A, D et a marqués dans l'algorithme de Ford - Fulkerson.

2) tracer un trait horizontal sur chaque ligne non marquée et un trait vertical sur chaque colonne marquée.

3) a) choisir le plus petit élément p du tableau non rayé (il est forcément > 0) ;
b) l'ajouter aux lignes rayées et le soustraire aux colonnes non rayées.

remarques : - on obtiendra la même solution optimale x_{ij} (d'après le théorème ci-dessus) ;

- méthode plus simple : a) on augmente de p les éléments traversés par 2 traits
  b) on ne modifie pas les éléments traversés par 1 trait
  c) on diminue de p les éléments traversés par 0 trait.

exemple :

	a	b	c	d	e
A	0	0	1	0	1
B	1	3	1	2	0
C	3	1	0	2	1
D	0	0	1	1	0
E	2	0	3	0	2

tableau III

coût minimal (tableau III)
= coût minimal (tableau II) - 4 (4 colonnes non rayées) + 3 (3 lignes rayées)
= coût minimal (tableau I) - 10.

4) retour à la phase 2.

exemple : - ici la sortie de l'algorithme est en 3.1.a. car on obtient 5 zéros encadrés ;
- coût minimal (tableau III) = 0 $\Rightarrow$ coût minimal (tableau I) = 10 ;
- solution optimale : A-b (2), B-e (3), C-c (1), D-a (1), E-d (3) ;
- en utilisant une procédure récursive dans la dernière phase 2 de l'algorithme, on peut montrer qu'il existe 3 solutions de coût 10 : A-d (4), B-e (3), C-c (1), D-a (1), E-b (1)
et A-a (1), B-e (3), C-c (1), D-b (2), E-d (3).

- *Sortie en 3. 1. d.* : *obtention de la solution optimale*

(s'il existe une colonne marquée ne contenant pas de zéro encadré)

algorithme : remplacer alternativement les 0 barrés (resp. encadrés) ayant servi au marquage de cette colonne (resp. ligne) par des 0 encadrés (resp. barrés) : cf exercice du TD5.

- *Convergence de l'algorithme* : rapide car obtenue en au plus n itérations puisque chaque passage en phase 3 augmente d'une unité au moins le nombre de colonnes marquées.

VI. Problèmes de transport

1) Problème posé

- *Exemple* :

Des marchandises doivent être transportées des $m = 3$ usines I, II, III aux $n = 4$ distributeurs 1, 2, 3, 4, le transport d'une marchandise de l'origine i à la destination j ayant un coût c_{ij} donné dans le tableau ci-contre. La disponibilité a_i en marchandises de l'usine i et la demande b_j de marchandises du distributeur j sont également données dans le tableau.

	1	2	3	4	a_i
I	12	27	61	49	18
II	23	39	78	28	32
III	67	56	92	24	14
b_j	9	11	28	16	

- *But* : le problème consiste à déterminer les quantités x_{ij} à transporter de i vers j de façon que le coût total de transport $\sum_i \sum_j c_{ij} x_{ij}$ soit minimal tout en satisfaisant les deux contraintes :

- la quantité totale de marchandises partant de i est égale à la disponibilité a_i : $\sum_j x_{ij} = a_i$

- la quantité totale de marchandises reçue par j est égale à la demande b_j : $\sum_i x_{ij} = b_j$.

- pour résoudre le problème, il faut impérativement que la demande globale soit égale à la disponibilité globale : $\sum_i a_i = \sum_j b_j$. Dans l'exemple : $\sum_i a_i = \sum_j b_j = 64$.
- si ce n'est pas le cas, par exemple si $b_3 = 18$ au lieu de 28, il faut créer un distributeur fictif $j = 5$ avec une demande $b_5 = 28 - 18 = 10$ et attribuer un coût énorme (1000) pour que la somme des x_{i5} soit égale à 10 et qu'ainsi aucune marchandise réelle soit transportée vers le distributeur fictif.

- *Résolution en deux étapes* : 1) recherche d'une solution de base initiale,
  2) recherche de la solution optimale.

2) Recherche d'une solution de base initiale

- *Solution de base* : - on a $m \times n$ inconnues x_{ij} avec $m + n - 1$ équations (m équations de disponibilités + n équations de demandes - 1 équation "demande globale = disponibilité globale") ; on peut donc déterminer $m + n - 1$ inconnues.
  - une solution $\{x_{ij}\}$ est dite de base si les autres inconnues, au nombre de $m \times n - (m + n - 1) = (m - 1) \times (n - 1)$, sont nulles.

- *Algorithmes* : certains n'utilisent pas les coûts c_{ij} (coin Nord-Ouest), d'autres les utilisent (Minili, Minico, différence maximale).

- *Méthode du coin Nord-Ouest*

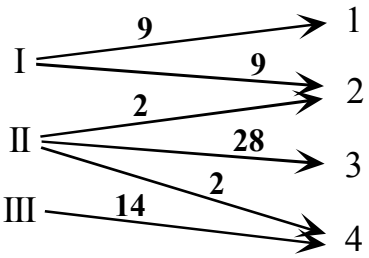
algorithme : 1) transporter sur la case Nord - Ouest la quantité maximale possible ; 2) retour à 1.

exemple :

	1	2	3	4	a_i
I	9	9	0	0	18
II	0	2	28	2	32
III	0	0	0	14	14
b_j	9	11	28	16	

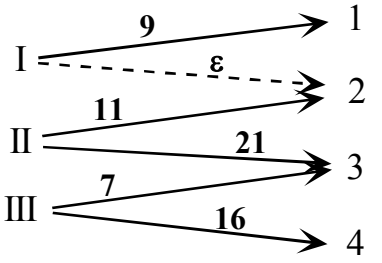
case I, 1 : Min (18, 9) = 9
case I, 2 : Min (9, 11) = 9
case II, 2 : Min (32, 2) = 2
case II, 3 : Min (30, 28) = 28
case II, 4 : Min (2, 16) = 2
case III, 4 : Min (14, 14) = 14.

remarques : - la solution est de base car on obtient $(m - 1) \times (n - 1) = 2 \times 3 = 6$ zéros,
- coût = $9 \times 12 + 9 \times 27 + 2 \times 39 + 28 \times 78 + 2 \times 28 + 14 \times 24 = 3005$,
- une solution de base donne un graphe connexe sans cycle (= arbre).



si la solution n'est pas de base (dégénérée) : rendre connexe le graphe en rajoutant par exemple l'arc I-2 avec une quantité transportée petite $\varepsilon > 0$ et faire $\varepsilon = 0$ quand l'optimum est obtenu.

	1	2	3	4	a_i
I	9	**ε**	0	0	**9+ ε**
II	0	11	21	0	32
III	0	0	7	16	**23**
b_j	9	11+ **ε**	28	16	



- *Méthodes Minili et Minico*

Minili : affecter sur la ligne I (puis sur les lignes II, III, I, II…) le maximum de marchandises à la relation de coût minimal sur la ligne.

Minico : affecter sur la colonne 1 (puis sur les colonnes 2, 3, 4, 1, 2…) le maximum de marchandises à la relation de coût minimal sur la colonne.

exemple :

	1	2	3	4	a_i
I	9	0	9	0	18
II	0	0	16	16	32
III	0	11	3	0	14
b_j	9	11	28	16	

Minili (coût 3245)

	1	2	3	4	a_i
I	9	9	0	0	18
II	0	2	28	2	32
III	0	0	0	14	14
b_j	9	11	28	16	

Minico (coût 3005)
même solution qu'avec le coin N-O

- *Méthode de la différence maximale (Balas - Hammer)*

algorithme : 1) calcul pour chaque rangée (ligne ou colonne) de la différence entre le coût le plus petit et celui immédiatement supérieur ;
2) choisir la rangée de différence maximale : affecter à la relation de coût minimal de cette rangée la quantité maximale possible ; ceci sature une ligne ou une colonne ;
3) retour à 1 jusqu'à ce que toutes les rangées soient saturées.

exemple :

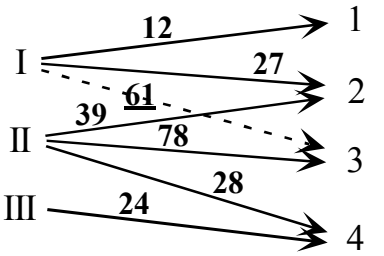
- la solution est une solution de base car elle contient 6 zéros,
- coût = $18 \times 61 + 9 \times 23 + 11 \times 39 + 10 \times 78 + 2 \times 28 + 14 \times 24 = 2906$,
- méthode efficace (résultat souvent proche de l'optimum).

	1	2	3	4	a_i
I	0	0	18	0	18
II	9	11	10	2	32
III	0	0	0	14	14
b_j	9	11	28	16	

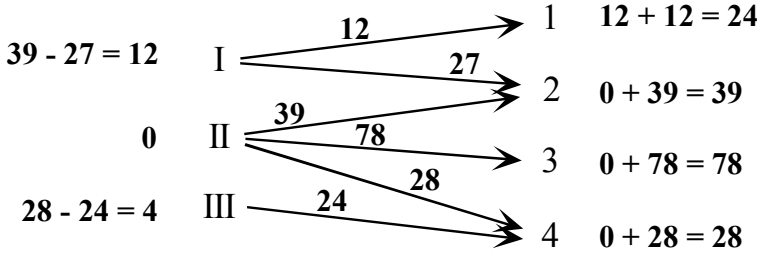
3) Recherche de la solution optimale (méthode du stepping-stone)

- *Remarque* : on part de la solution du coin Nord - Ouest car on verra que la solution donnée par la méthode de la différence maximale est optimale.

- *Méthode* :



- rajouter un nouvel arc, par exemple l'arc I-3 de coût 61 ; ceci crée le cycle I, 3, II, 2, I ;
- transporter +1 sur l'arc I-3, -1 sur l'arc II-3, +1 sur l'arc II-2, -1 sur l'arc I-2, soit un coût marginal négatif de 61 - 78 + 39 - 27 = -5 (disponibilités et demandes respectées en chaque sommet) ;
- on cherche la quantité à transporter sur l'arc I-3 qui va éliminer l'arc I-2 ou l'arc II-3 (voir plus bas) ;
- on applique la méthode des potentiels pour trouver les cycles intéressants (potentiel $v_{II} = 0$ pour la ligne II qui a le coût le plus élevé : 78).



- coût marginal d'un cycle introduit par un arc X-Y : $\delta_{X,Y} = v_{X,Y} + v_X - v_Y$
avec $v_{X,Y}$ = coût de X à Y et v_X, v_Y = potentiels des sommets X et Y.

- *Algorithme du stepping stone* :

  1) choisir la case affectée ayant le coût le plus grand et prendre 0 comme potentiel de cette ligne ;
  2) calculer les potentiels de chaque rangée ;
  3) calculer les coûts marginaux $\delta_{X,Y} = v_{X,Y} + v_X - v_Y$ des cases (X,Y) du tableau non affectées :
     - si tous les $\delta_{X,Y}$ sont ≥ 0 : l'optimum est atteint ;
     - sinon, pour le $\delta_{X,Y}$ le plus négatif : rechercher le cycle introduit par l'arc (X,Y), en utilisant des signes + et –, et déplacer la quantité maximale possible ;
  4) retour en 1.

- *Exemple* :

itération 1 :

	1	2	3	4	Potentiel
I	9 / 12	9 (-) / 27	(+) / 61	/ 49	12
II	/ 23	2 (+) / 39	28 (-) / 78	2 / 28	0
III	/ 67	/ 56	/ 92	14 / 24	4
Potentiel	24	39	78	28	

- $\delta_{I,3} = 61 + 12 - 78 = -5$
- quantité déplaçable = Min (9, 28) = 9
- gain = $-5 \times 9 = -45$
- coût total = 3005 - 45 = 2960.

itération 2 :

	1	2	3	4	Potentiel
I	9 (-) / 12	/ 27	9 (+) / 61	/ 49	17
II	(+) / 23	11 / 39	19 (-) / 78	2 / 28	0
III	/ 67	/ 56	/ 92	14 / 24	4
Potentiel	29	39	78	28	

- $\delta_{II,1} = 23 + 0 - 29 = -6$
- quantité déplaçable = Min (9, 19) = 9
- gain = $-6 \times 9 = -54$
- coût total = 2960 - 54 = 2906.

itération 3 :

	1	2	3	4	Potentiel
I	/ 12	/ 27	18 / 61	/ 49	17
II	9 / 23	11 / 39	10 / 78	2 / 28	0
III	/ 67	/ 56	/ 92	14 / 24	4
Potentiel	23	39	78	28	

- tous les $\delta_{X,Y}$ sont > 0 (la solution est optimale).

- *Remarque* : l'optimum n'est pas unique s'il existe un coût marginal $\delta_{X,Y}$ nul.

Chapitre 2. Phénomènes aléatoires en RO

Le hasard intervient : - *files d'attente* (arrivée des clients, durée du service),
- *fiabilité* des équipements (instants des pannes, durée des réparations),
- *gestion des stocks* (instants des commandes, quantités demandées)

⇒ processus stochastique : Einstein (1905), Markov (1910), Kolmogorov (1933)…

I. Processus stochastique markovien

1) Définitions

- *Processus stochastique* : ensemble de variables aléatoires (v.a.) X_t dépendant du temps t ($t \in T$).

  - soit $T = \{t_1, t_2, t_3 \ldots\}$; X_t forme une suite de v.a. (chaîne de Markov) : exemple d'une machine en marche à t ($X_t = 1$) ou en panne ($X_t = 0$) observée à des instants t_n.
  - soit T est un intervalle continu $[t_{initial}, t_{final}]$; on a réellement un processus stochastique (amplitude A_t d'un signal aléatoire, nombre N_t de personnes dans une file d'attente).
  - si les v.a. X_t prennent un nombre fini (ou dénombrable) de valeurs, on parle d'un "processus à espace d'états discret" (exemple : N_t), sinon le processus est dit "à espace d'états continu" (exemple : A_t).

- *Processus markovien* : processus pour lequel $\forall u$ les v.a. X_t pour $t > u$ ne dépendent pas des v.a. X_s pour $s < u$: X_t ne dépend que de X_u et $X_{t>u}$ (processus sans mémoire).

  - exemples : machine en panne ou marche (chaîne de Markov), N_t dans file d'attente.

2) Processus markovien particulier : processus de naissance (PN)

- *Quelques rappels sur les probabilités*

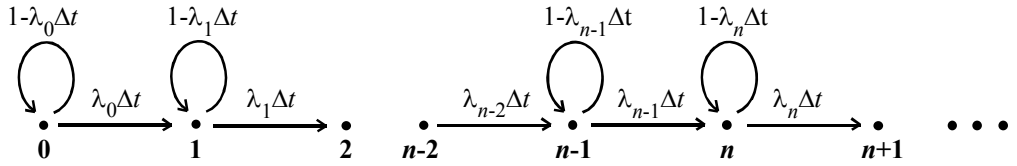
- *Définitions* : un processus de naissance correspond à l'apparition d'"individus" à des instants aléatoires ; le PN est la donnée des v.a. N_t pour $t \in T$, où N_t est le nombre d'"individus" apparus dans l'intervalle $[0, t]$.

  - PN homogène : $P(N_{t+\Delta t} = k + 1 \mid N_t = k)$ est indépendant de t ( Δt est infiniment petit).

On peut alors écrire : $P(N_{t+\Delta t} = k + 1 \mid N_t = k) = \lambda_k \Delta t$ (en négligeant les termes d'ordre supérieur)
et $P(N_{t+\Delta t} = k \mid N_t = k) = 1 - \lambda_k \Delta t$ (pas plus d'une apparition car Δt est petit)

On pose : $p_n(t) = P(N_t = n)$, c'est aussi $P(N_t = n \mid N_0 = 0)$.

- <u>diagramme des transitions (entre t et $t + \Delta t$)</u> :



- *Equations du futur* : méthode différentielle

  - <u>formule des probabilités totales</u> (FPT) : $p_n(t+\Delta t) = p_{n-1}(t)\,\lambda_{n-1}\Delta t + p_n(t)\,(1-\lambda_n \Delta t)$

  - <u>équations du futur</u> (avec $\Delta t \to 0$) : $p_n'(t) = -\lambda_n\, p_n(t) + \lambda_{n-1}\, p_{n-1}(t)$

    ($p_n(t)$ se calcule à partir de $p_{n-1}(t)$)

    si $n = 0$: $p_0'(t) = -\lambda_0\, p_0(t)$

    si $n = 1$: $p_1'(t) = -\lambda_1\, p_1(t) + \lambda_0\, p_0(t)$ (résolution de proche en proche)

- *Cas particulier* : processus de Poisson

  - <u>définition</u> : $P(N_{t+\Delta t} = k+1 \mid N_t = k) = \lambda\,\Delta t$ - indépendant de t (processus homogène)
    - indépendant du nombre k d'individus ($\lambda_k = \lambda$)

    conséquences : - probabilités d'apparition identiques dans $[t, t+\Delta t]$ et $[u, u+\Delta t]$,
    - indépendance du nombre d'individus apparus entre $[t, t+\Delta t]$ et $[u, u+\Delta t]$

  - <u>équations du futur</u> : le calcul donne $p_0(t) = e^{-\lambda t}$, $p_1(t) = \lambda t\, e^{-\lambda t}$,

    et par récurrence : $p_n(t) = \dfrac{(\lambda t)^n}{n!}\, e^{-\lambda t}$

    $\Rightarrow$ le nombre N_t d'individus apparus entre 0 et t (ou entre t_0 et t_0+t) suit une loi de Poisson de paramètre λt.

  - <u>intervalle de temps T entre 2 apparitions d'individus</u> :

    - Quelques rappels sur les probabilités
    - Fonction de répartition : $F(t) = P(T < t) = 1 - P(T > t) = 1 - p_0(t) = 1 - e^{-\lambda t}$ si $t > 0$, 0 sinon.
    - Densité de probabilité : $f(t) = F'(t) = \lambda\, e^{-\lambda t}$ si $t > 0$, 0 sinon (loi exponentielle de paramètre λ)
    - Espérance de T : $E(T) = \displaystyle\int_0^{+\infty} t\, f(t)\, dt = \frac{1}{\lambda}$ donc λ = nombre moyen d'individus apparaissant par seconde.

  - <u>conclusion</u> :

$\Leftrightarrow$	instants d'arrivée <u>indépendants</u> en nombre moyen λ par seconde (= processus de Poisson) intervalles de temps T entre 2 arrivées de densité exponentielle de paramètre λ.

3) Processus de naissance et mort (PNM)

- *Processus de mort (PM) homogène* : processus où les individus disparaissent à des instants aléatoires.

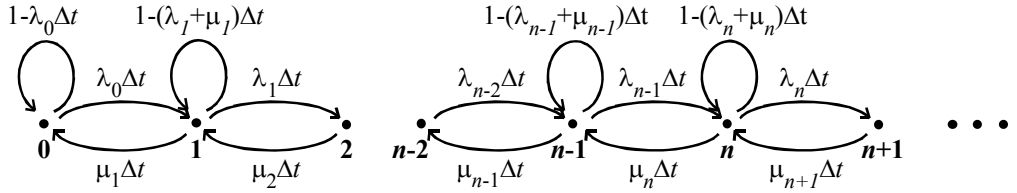
  - N individus à $t = 0$,
  - $P(N_{t+\Delta t} = k - 1 \mid N_t = k) = \mu_k \, \Delta t$ (Δt petit).

- *Processus de naissance et mort (PNM) homogène* : les individus apparaissent et disparaissent à des instants aléatoires (ex : files d'attente)

  - $N_0 = 0$ avec éventuellement une limitation à N (valeur maximale de k),
  - $P(N_{t+\Delta t} = k + 1 \mid N_t = k) = \lambda_k \, \Delta t$
  - $P(N_{t+\Delta t} = k - 1 \mid N_t = k) = \mu_k \, \Delta t$
  - $P(N_{t+\Delta t} = k \mid N_t = k) = 1 - \lambda_k \, \Delta t - \mu_k \, \Delta t$

  } (Δt petit) avec $\lambda_k > 0$ et $\mu_k > 0$ (sauf $\mu_0 = 0$)

- *Diagramme des transitions (entre t et t + Δt)* :



- *Equations du futur* :

  - FPT : $p_n(t + \Delta t) = (1 - \lambda_n \Delta t - \mu_n \Delta t) \; p_n(t) + \lambda_{n-1} \Delta t \; p_{n-1}(t) + \mu_{n+1} \Delta t \; p_{n+1}(t)$

  - équations du futur (avec $\Delta t \to 0$) : $p_n'(t) = -(\lambda_n + \mu_n) \; p_n(t) + \lambda_{n-1} \; p_{n-1}(t) + \mu_{n+1} \; p_{n+1}(t)$

    si $n = 0$: $p_0'(t) = -\lambda_0 \; p_0(t) + \mu_1 \; p_1(t)$

    si $n = N$: $p_N'(t) = -\mu_N \; p_N(t) + \lambda_{N-1} \; p_{N-1}(t)$ (c'est intégrable, cf. littérature)

- *Régime permanent* : $p_n(t)$ met un certain temps (régime transitoire) avant de se stabiliser. Ce régime transitoire est souvent court, on s'intéresse alors surtout au régime permanent pour lequel $p_n(t)$ ne dépend plus du temps : $p_n(t)$ = constante notée p_n.

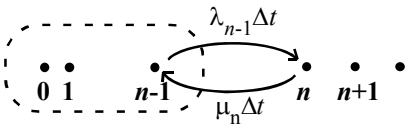
  - les équations du futur donnent alors : $p_n = \dfrac{\lambda_0 \, \lambda_1 \ldots \lambda_{n-1}}{\mu_1 \, \mu_2 \ldots \mu_n} \, p_0$. Avec la condition de normalisation $\sum_{n=0}^{N(ou+\infty)} p_n = 1$ (le système est forcément dans un des états n), on obtient p_0 et par suite p_n.

  - cas particulier fréquent : tous les λ_k sont égaux à λ et tous les μ_k sont égaux à μ (avec $\lambda < \mu$),

    alors : $p_n = \left(\dfrac{\lambda}{\mu}\right)^n p_0$ et (cas où $N \to +\infty$) : $p_0 = 1 - \dfrac{\lambda}{\mu}$ et $p_n = (1 - \dfrac{\lambda}{\mu}) \left(\dfrac{\lambda}{\mu}\right)^n$.

- *Théorème des coupes* : pour un processus markovien quelconque.

  - on réalise une coupe sur le diagramme des transitions :



  - en régime permanent (la situation de départ est oubliée), la probabilité de sortie de la coupe entre t et $t+\Delta t$ est égale à la probabilité d'entrer dans la coupe, d'où : $\lambda_{n-1}\ p_{n-1} = \mu_n\ p_n$

  - cette relation redonne directement $p_n = \dfrac{\lambda_0 \lambda_1 ... \lambda_{n-1}}{\mu_1 \mu_2 ... \mu_n} p_0$.

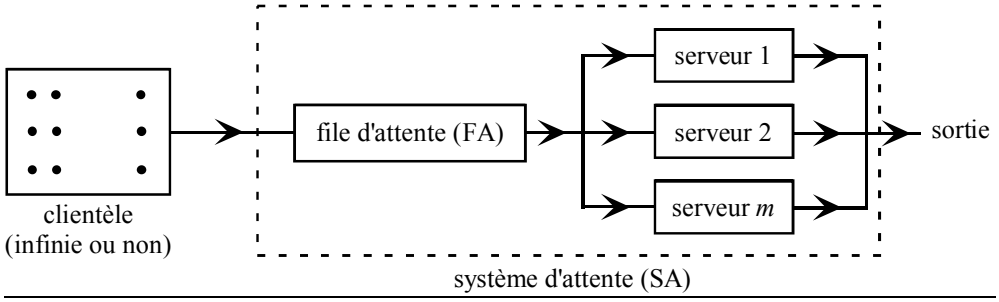
  - la relation de récurrence obtenue par la méthode des coupes est toujours plus simple (ici relation entre p_{n-1} et p_n) que celle obtenue par les équations du futur (ici relation entre p_{n-1}, p_n et p_{n+1}) ; par contre, les équations du futur permettent de calculer le régime transitoire.

II. Phénomènes d'attente

1) Description d'un système d'attente (SA)

- *Schéma* :

une seule FA !



- *Exemples* : supermarché (1 caisse), attente à la poste (1 FA), attente chez un médecin ou à une station de taxis, réseau d'ordinateur avec une seule imprimante…

- *Remarques* :
  - bien distinguer SA et FA,
  - le système est ouvert (sur l'extérieur) si la clientèle est infinie,
  - le système est fermé si la clientèle est finie (exemple du système formé par des machines en attente de réparation dans une salle),
  - historique : Erlang (1917, réseau téléphonique), E. Borel, A. Khintchine, W. Feller, informaticiens (réseaux d'ordinateurs)…

- *Variables aléatoires caractérisant un SA* :

  - loi d'arrivée X des clients :

    exemple fréquent : loi de Poisson de paramètre λ

    - instants d'arrivée indépendants,
    - λ = nombre moyen d'arrivées par seconde,
    - Proba(n clients arrivent entre 0 et t) = $\dfrac{(\lambda t)^n}{n!} e^{-\lambda t}$,
    - l'intervalle de temps T entre 2 arrivées successives suit une loi de densité exponentielle de paramètre λ : $f(t) = \lambda\, e^{-\lambda t}$ si $t > 0$ (0 sinon) et $E(T) = \dfrac{1}{\lambda}$.

- loi de la durée Y du service :

exemple fréquent : durée exponentielle de paramètre μ (bcp de services courts, peu de longs)

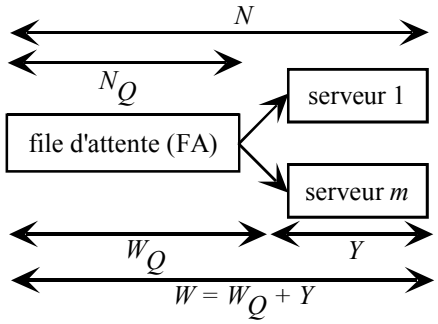
- $f_Y(y) = \mu\, e^{-\mu y}$ si $y > 0$ (0 sinon) ; $F_Y(y) = P(Y < y) = 1 - e^{-\mu y}$ si $y > 0$ (0 sinon),
- durée moyenne du service = $E(Y) = \int_0^{+\infty} y\, f_Y(y)\, dy = \frac{1}{\mu}$.

- notations : $\forall\, X$ et Y, on pose : - taux moyen d'arrivées = $\lambda = \frac{1}{E(X)}$,

- taux moyen de service (pour 1 serveur) = $\mu = \frac{1}{E(Y)}$.

- autres variables :

- N = nombre de clients dans le SA
- N_Q = nombre de clients dans la FA
- W = durée d'attente pour 1 client dans le SA
- W_Q = durée d'attente pour 1 client dans la FA.



- *Notation de Kendall d'un SA* : $X / Y / m / K / M / R$ (3 premiers paramètres obligatoires)

avec : - X : loi des arrivées (M si loi de Poisson)
- Y : loi de la durée du service (M si exponentielle)
- m = nombre de serveurs
- K = capacité d'accueil = nombre maximum de clients dans le SA (∞ par défaut)
- M = taille de la population des clients (∞ par défaut)
- R : règle de service (PAPS = FIFO par défaut).

exemples : - $M / M / 1$: arrivées indépendantes, durée du service exponentielle, 1 serveur
- $M / M / m$: idem sauf m serveurs
- $M / M / m / m$: idem + les clients repartent si les m serveurs sont occupés
- $M / M / m / K$: idem + les clients repartent s'il y a $K - m$ personnes dans la FA
- $M / M / m / K / K$: idem + cas du système formé par K machines pouvant tomber en panne et être réparées par m mécaniciens ($m < K$).

- *Critères de performance* :

- $u = \frac{E(Y)}{E(X)} = \frac{\lambda}{\mu}$: intensité du trafic (en Erlang) ;
- nombre minimal de serveurs = plus petit entier m tel que $\frac{u}{m} < 1$;
- $\rho = \frac{u}{m}$ = facteur d'utilisation = $\frac{\text{temps de service demandé } \lambda T / \mu}{\text{temps de service possible } mT}$;
- $\overline{W}$ et $\overline{W_Q}$: durée moyenne de séjour dans le SA et dans la FA ;
- $\overline{N}$ et $\overline{N_Q}$: nombre moyen de clients dans le SA et dans la FA.

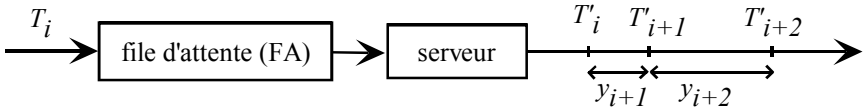
- *Equations de conservation des clients dans le SA* : formules de Little

  - pour un SA de type *G* / *G* / *m* : $\overline{N} = \lambda\, \overline{W}$ et $\overline{N_Q} = \lambda\, \overline{W_Q}$

  - pour un SA de type *G* / *G* / 1 : $\overline{W_Q} = \dfrac{\overline{N}}{\mu}$

2) Système d'attente M/M/1 (/ ∞ / ∞ / FIFO)

- *Hypothèses* :



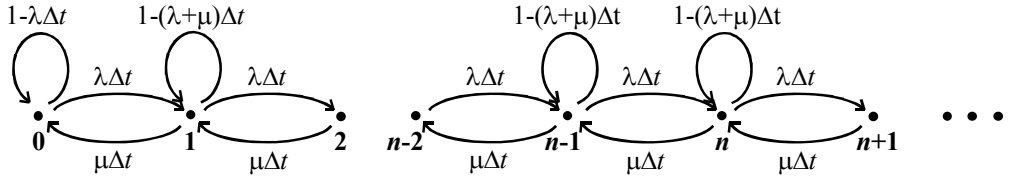
  - loi des arrivées : processus de Poisson de paramètre λ,
  - loi de la durée du service : exponentielle de paramètre μ,

  (condition de non-engorgement : $u = \dfrac{\lambda}{\mu} < 1$)

  - conséquence : les instants de sortie T'_i obéissent à une loi de Poisson de paramètre μ :

  Proba(*x* clients sortent entre 0 et *t*) = $\dfrac{(\mu t)^x}{x!} e^{-\mu t}$,

- *Diagramme des transitions du PNM* N_t :



  - $P(N_t = n) = p_n(t)$ donnée par les éq. du futur : $p_n'(t) = -(\lambda + \mu)\, p_n(t) + \lambda\, p_{n-1}(t) + \mu\, p_{n+1}(t)$

  - en régime permanent : $p_n'(t) = 0 \quad \Rightarrow \quad p_n = (1 - \dfrac{\lambda}{\mu}) \left(\dfrac{\lambda}{\mu}\right)^n$.

- *Lois et espérances de N et* N_Q :

  - pour *N* : $P(N = n) = p_n = (1 - \dfrac{\lambda}{\mu}) \left(\dfrac{\lambda}{\mu}\right)^n$ et $\overline{N} = \sum_{n=0}^{+\infty} n\, p_n = \dfrac{u}{1-u} = \dfrac{\lambda}{\mu - \lambda}$

  - pour N_Q : $N_Q = 0$ si $N = 0$ ou 1 et $N_Q = n$ si $N = n + 1$

  d'où $P(N_Q = 0) = p_0 + p_1 = 1 - \left(\dfrac{\lambda}{\mu}\right)^2$ et $P(N_Q = n) = p_{n+1} = (1 - \dfrac{\lambda}{\mu}) \left(\dfrac{\lambda}{\mu}\right)^{n+1}$ si $n \geq 1$

  $$\overline{N_Q} = \sum_{n=\emptyset 1}^{+\infty} n\, P(N_Q = n) = \frac{u^2}{1-u} = \frac{\lambda^2}{\mu(\mu - \lambda)}$$

  - remarque : $\overline{N} \neq \overline{N_Q} + 1$ car $N \neq N_Q + 1$ lorsqu'il n'y a personne dans le SA,

  $\overline{N} - \overline{N_Q} = \dfrac{\lambda}{\mu} = 1 - p_0$ = probabilité que le serveur soit occupé (= facteur d'utilisation).

- *Lois et espérances de W et W_Q* :

  - pour *W* : *densité de probabilité* exponentielle de paramètre μ - λ :

$$f_W(t) = (\mu - \lambda)\, e^{-(\mu-\lambda)t} \text{ si } t > 0 \text{ (0 sinon)},$$

fonction de répartition : $F_W(t) = P(W < t) = 1 - e^{-(\mu-\lambda)t}$ si $t > 0$ (0 sinon),

espérance : $\overline{W} = \int_0^{+\infty} t\, f_W(t)\, dt = \frac{1}{\mu - \lambda}$.

  - pour W_Q : v.a. mixte (à la fois continue et discrète car $P(W_Q = 0) \neq 0$)

$$P(W_Q = 0) = p_0 = 1 - \frac{\lambda}{\mu}$$

densité de probabilité : $f_{W_Q}(t) = \lambda(1 - \frac{\lambda}{\mu})\, e^{-(\mu-\lambda)t}$ si $t > 0$ (0 si $t < 0$),

$P(W_Q \leq t) = P(W_Q = 0) + \int_0^t f_{W_Q}(\tau) d\tau = 1 - \frac{\lambda}{\mu} e^{-(\mu-\lambda)t}$ si $t \geq 0$ (0 sinon),

espérance : $\overline{W_Q} = 0 \times (1 - \frac{\lambda}{\mu}) + \int_0^{+\infty} t\, f_{W_Q}(t)\, dt = \frac{\lambda}{\mu} \frac{1}{\mu - \lambda}$.

- *Remarque* : on retrouve bien les formules de Little : $\overline{N} = \lambda\, \overline{W}$, $\overline{N_Q} = \lambda\, \overline{W_Q}$, $\overline{W_Q} = \frac{\overline{N}}{\mu}$.

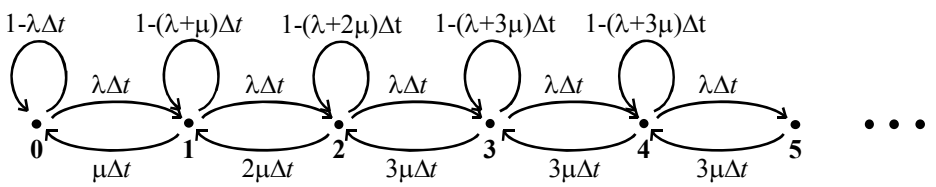
- *Exemple* : la durée moyenne de consultation chez un médecin est de 15 minutes (μ = 4 / h) ; les clients sont convoqués toutes les 20 minutes (λ = 3 / h) mais arrivent aléatoirement.
  Alors : $\overline{N} = 3$; $\overline{N_Q} = 2{,}25$; $\overline{W} = 60$ minutes ; $\overline{W_Q} = 45$ minutes.

- *Conservation des clients dans le SA* :

  - en régime permanent, pendant Δt :
    nombre moyen de clients entrant dans le SA = $\lambda\, \Delta t$
    = nombre moyen de clients sortant du SA = $\mu\, \Delta t\, (1 - p_0)$
  - on retrouve la relation : $\lambda = \mu\, (1 - p_0)$, soit $p_0 = 1 - \frac{\lambda}{\mu}$ = probabilité que le serveur se repose.

3) Système d'attente M/M/m (/ ∞ / ∞ / FIFO)

- *Diagramme des transitions entre t et t + Δt pour m = 3 serveurs* :



  - remarque : $P(N_{t+\Delta t} = k - 1 \mid N_t = k) = 3\, \mu\, \Delta t$ si $k \geq 3$ car 3 clients peuvent sortir du service en Δt.

- *Probabilités p_n en régime permanent (m serveurs)* : théorème des coupes

  - <u>pour $n \le m$</u> : $p_n = \frac{1}{n!}\left(\frac{\lambda}{\mu}\right)^n p_0$

  - <u>pour $n > m$</u> : $p_n = \frac{1}{m^{n-m}} \frac{1}{m!}\left(\frac{\lambda}{\mu}\right)^n p_0$

  la condition de normalisation $\sum_{n=0}^{+\infty} p_n = 1$ donne $p_0 = \frac{1}{\frac{1}{m!}\left(\frac{\lambda}{\mu}\right)^m \frac{1}{1-\frac{\lambda}{m\mu}} + \sum_{n=0}^{m-1} \frac{1}{n!}\left(\frac{\lambda}{\mu}\right)^n}$

- *Lois de probabilités de N, N_Q, W, W_Q* : cf. A. Alj, R. Faure, *Guide de la RO*, Tome 2, p. 222.

- *Espérances de N, N_Q, W, W_Q* :

  $\overline{W_Q} = \frac{\left(\frac{\lambda}{\mu}\right)^m}{\mu\, m\, m!(1-\frac{\lambda}{m\mu})^2} p_0$ d'où $\overline{N_Q} = \lambda\, \overline{W_Q}$ (Little)

  $W = W_Q + Y$ donne : $\overline{W} = \overline{W_Q} + \frac{1}{\mu}$ d'où $\overline{N} = \lambda\, \overline{W}$ (Little)

$\overline{W_Q}$ (2 serveurs) << $\frac{1}{2}\overline{W_Q}$ (1 serveur) !!

<u>exemple de l'attente chez un médecin</u> :

$\overline{W_Q}$ (2 serveurs) = 2 min 27 s alors que $\frac{1}{2}\overline{W_Q}$ (1 serveur) = 22 min 30 s !!

Chapitre 3. Programmation Mathématique Linéaire

Problème : on cherche à optimiser (maximiser ou minimiser) une fonction linéaire de plusieurs variables (fonction économique) sous contraintes multiples (fonctions linéaires de ces variables).

I. Exemple de programme mathématique linéaire (PML)

Enoncé : Un pays en voie de développement veut mettre en valeur une zone de 900 ha où 2 cultures sont possibles, les dattes et le blé. Les données relatives à 1 ha sont les suivantes :

	dattes	blé
- rendement en quintaux à l'hectare	75	25
- prix de vente au quintal	60	60
- main d'œuvre nécessaire (en nombre d'ouvriers)	1	2
- frais d'exploitation (hors salaires) en €	3 500	300
- eau nécessaire pour irriguer en m^3 par année	14 000	6 000

Les salaires annuels sont de 500 € par an et par personne. Les disponibilités des facteurs de production (terre, main d'œuvre et eau) sont respectivement : 900 ha, 1200 ouvriers et 14 millions de m^3 d'eau par an. Le pays cherche à maximiser le revenu national défini comme la somme des salaires versés et du bénéfice.

1) Mise en forme du problème :

- *Choix des inconnues* : x = nombre d'ha de dattes et y = nombre d'ha de blé, alors $x \geq 0$ et $y \geq 0$

  remarque : les inconnues et variables diverses d'un PML sont toujours positives ou nulles.

- *Contraintes économiques* : - pour la terre : $x + y \leq 900$ (pas = 900 !)
  - pour la main d'œuvre : $x + 2\,y \leq 1\,200$
  - pour l'eau : $14\,000\,x + 6\,000\,y \leq 14\,000\,000$, soit $14\,x + 6\,y \leq 14\,000$

- *Fonction économique* : - salaires versés : $x \times 1 \times 500 + y \times 2 \times 500$
  - bénéfice : $(x \times 75 \times 60 - x \times 3\,500 - x \times 1 \times 500)$
    $+ (y \times 25 \times 60 - y \times 300 - y \times 2 \times 500)$

  d'où le revenu national : $\Gamma = 1\,000\,x + 1\,200\,y$

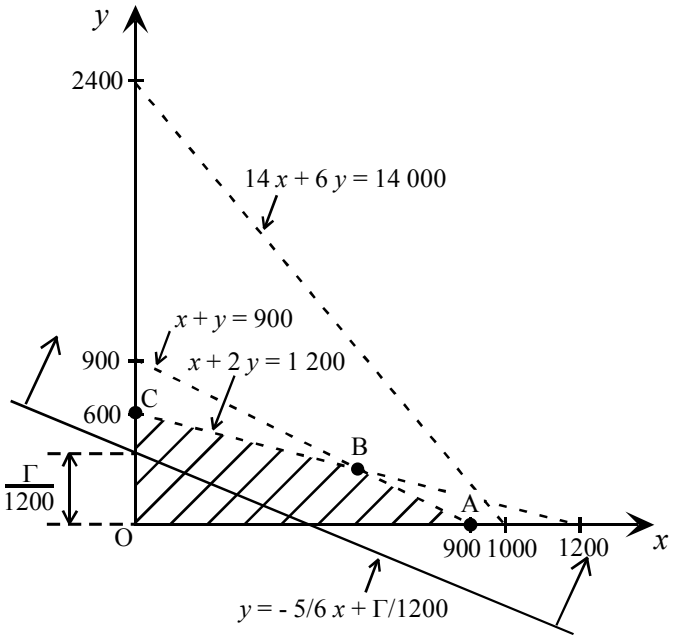
  remarque : une société privée chercherait à maximiser le bénéfice $500\,x + 200\,y$ (le choix du critère d'optimisation est à la charge du décideur)

- *PML obtenu* : on cherche

$$\text{Max } \Gamma = 1\,000\,x + 1\,200\,y$$
$$\text{sous les contraintes : } \begin{cases} x \geq 0 \text{ et } y \geq 0 \\ x + y \leq 900 \\ x + 2\,y \leq 1\,200 \\ 14\,x + 6\,y \leq 14\,000 \end{cases}$$

2) Résolution graphique :

- les contraintes définissent la zone admissible des solutions (ZAS) qui est le polygone OABC.
- la ZAS sera toujours un polygone pour un programme linéaire.
- tous les couples (x, y) de la droite d'équation $\Gamma = 1\,000\,x + 1\,200\,y$ (ou $y = -5/6\,x + \Gamma/1200$) correspondent à la même valeur de Γ.
- l'intersection de cette droite avec l'axe Oy donne la valeur de $\Gamma/1200$. Pour augmenter Γ, il faut donc déplacer la droite de pente fixe -5/6 vers le haut jusqu'à ce qu'elle ne traverse la ZAS qu'en un seul point (B), qui est le lieu du maximum de Γ.



- l'optimum d'un PML est situé forcément en l'un des sommets de la ZAS (cas non dégénéré), ici le sommet B : $x_B = 600$ ha, $y_B = 300$ ha et $\Gamma_B = 960\,000$ €.

remarques : - si le segment AB avait pour pente -5/6, tous ses points seraient des solutions optimales avec la même valeur de Γ (cas dégénéré)
- la résolution graphique est impossible s'il y a plus de 3 variables.

3) Résolution algébrique : méthode du simplexe (Dantzig)

- *Méthode* :

a) on remplace les inégalités des contraintes par des égalités, ce qui introduit des variables d'écart (VE) e_1, e_2, e_3 (forcément positives ou nulles). On part donc du système (S_0) qui contient 5 variables au lieu des 2 inconnues de départ :

$$(S_0) \quad \begin{cases} x + y + e_1 = 900 \\ x + 2\,y + e_2 = 1\,200 \\ 14\,x + 6\,y + e_3 = 14\,000 \end{cases}$$

b) chaque sommet du polygone correspond à 2 variables nulles (dites "variables hors-base") ; ainsi O correspond à $x = 0$ et $y = 0$ (les VE e_i sont non nulles et sont dites "variables dans la base"), A correspond à $e_1 = 0$ et $y = 0$, B à $e_1 = 0$ et $e_2 = 0$, et C à $x = 0$ et $e_2 = 0$. On part du sommet O ($\Gamma = 0$) et on va échanger une des variables hors-base avec une des variables dans la base (selon des critères bien définis, voir plus bas) pour se déplacer sur un autre sommet de la ZAS qui correspond à une valeur plus grande de Γ.

c) on répète alors l'opération pour atteindre le sommet optimum, suivant l'algorithme :
- α) sélection de la variable entrante (v. e.) dans la base
- β) sélection de la variable sortante (v. s.) de la base
- γ) écriture des nouvelles variables de la base en fonction des variables hors-base pour déterminer le nouveau maximum de Γ et préparer l'itération suivante.

- *Tableau initial* (T_0) : les données de départ peuvent se résumer dans le tableau (T_0) qui correspond au sommet O :

HB / B	x	y	(e_1) •	(e_2) •	(e_3) •	R	R'
e_1	1	1	1	0	0	900	
e_2	1	2	0	1	0	1 200	
e_3	14	6	0	0	1	14 000	
Γ	1 000	1 200	0	0	0	0	

- *Résolution algébrique (1ère itération)* :

  α) $\Gamma = 1\,000\,x + 1\,200\,y$: comme 1 200 > 1 000, on a intérêt à rendre y non nul pour augmenter Γ le plus possible (1er critère de Dantzig), donc y = v. e.

  β) les variables de la base s'écrivent en fonction des variables HB : $e_1 = 900 - x - y$

  $e_2 = 1\,200 - x - 2\,y$

  $e_3 = 14\,000 - 14\,x - 6\,y$

  or x reste HB ($x = 0$), donc : $e_1 = 900 - y \geq 0 \Rightarrow y \leq 900$

  $e_2 = 1\,200 - 2\,y \geq 0 \Rightarrow y \leq 1\,200 / 2 = 600$

  $e_3 = 14\,000 - 6\,y \geq 0 \Rightarrow y \leq 14\,000 / 6 \approx 2333$

  la plus grande valeur de y vérifiant les 3 contraintes est Min (900, 600, 2333) = 600, on choisit donc e_2 comme v. s. ($e_2 = 0$), c'est le 2ème critère de Dantzig.

  γ) on écrit les variables de la nouvelle base (e_1, y, e_3) en fonction des variables HB (x, e_2), (S_0) devient (S_1) :

$$(S_1)\ \begin{cases} e_1 = -1/2\,x + 1/2\,e_2 + 300 \\ y = -1/2\,x - 1/2\,e_2 + 600 \\ e_3 = -11\,x + 3\,e_2 + 10\,400 \\ \Gamma = 400\,x - 600\,e_2 + 720\,000 \end{cases} \Rightarrow \begin{cases} \text{HB} : x = 0,\ e_2 = 0 \\ \text{Base} : e_1 = 300,\ y = 600,\ e_3 = 10\,400 \\ \text{sommet : C avec } \Gamma_C = 720\,000\ € \end{cases}$$

- *Méthode des tableaux (1ère itération)* :

  α) 1er critère de Dantzig : v. e. = y car dans la ligne de Γ, le plus grand nombre positif (1 200) correspond à la colonne y.

  β) 2ème critère de Dantzig : v. s. = e_2 car la ligne e_2 correspond au plus petit nombre strictement positif dans la colonne R' remplie avec les éléments de la colonne R divisés par ceux de la colonne y (900/1, 1200/2, 14 000/6).

  γ) on peut alors remplir successivement (i) le cadre du tableau (T_1), (ii) la colonne de la v. e. y avec 0, 1, 0, 0 en gras (le 1 correspondant à la ligne y), (iii) la ligne de la v. e. y avec les éléments de la ligne de la v. s. e_2 du tableau (T_0) divisés par 2 pour obtenir le 1 de la colonne y, et (iv) les autres lignes en effectuant des combinaisons linéaires avec les lignes des tableaux (T_0) et (T_1) : $e_1(T_0) - 1 \times y(T_1)$ pour la ligne e_1, $e_3(T_0) - 6 \times y(T_1)$ pour la ligne e_3, $\Gamma(T_0) - 1\,200 \times y(T_1)$ pour la ligne Γ. On obtient le tableau (T_1) suivant :

HB / B	x	•	•	e_2	•	R	R'
e_1	1/2	**0**	1	-1/2	0	300	
y	1/2	**1**	0	1/2	0	600	
e_3	11	**0**	0	-3	1	10 400	
Γ	400	**0**	0	-600	0	-720 000	

remarques :

- chaque colonne de variable de base (•) contient un seul 1 et des 0 (test du calcul),
- le tableau (T_1) donne les mêmes résultats que la résolution algébrique,
- les variables hors base sont dans la ligne HB : $x = 0$ et $e_2 = 0$,
- les valeurs des variables de base sont dans la colonne R : $e_1 = 300$, $y = 600$, $e_3 = 10\ 400$,
- le tableau (T_1) correspond au point C car $x = 0$ et $y = 600$,
- la valeur de Γ en C est donnée dans la colonne R avec un signe moins ($\Gamma_C = 720\ 000$ €).

- *Méthode des tableaux ($2^{ème}$ itération)* :

α) 1^{er} critère de Dantzig : v. e. = x car dans la ligne de Γ, le plus grand nombre positif (400) correspond à la colonne x.

β) $2^{ème}$ critère de Dantzig : v. s. = e_1 car la ligne e_1 correspond au plus petit nombre strictement positif de la colonne R' remplie avec les éléments de la colonne R divisés par ceux de la colonne x (300/0,5 = 600, 600/0,5 = 1 200, 10 400/11 ≈ 945).

γ) on peut alors remplir le tableau (T_2) comme précédemment. Pour obtenir les lignes y, e_3 et Γ, on utilise les combinaisons linéaires suivantes : $y(T_1) - 1/2 \times x(T_2)$ pour la ligne y, $e_3(T_1) - 11 \times x(T_2)$ pour la ligne e_3, $\Gamma(T_1) - 400 \times x(T_2)$ pour la ligne Γ. On obtient le tableau (T_2) suivant :

HB / B	•	•	e_1	e_2	•	R	R'
x	**1**	0	2	-1	0	600	
y	**0**	1	-1	1	0	300	
e_3	**0**	0	-22	8	1	3 800	
Γ	**0**	0	-800	-200	0	-960 000	

remarques :

- tous les éléments de la ligne de Γ sont négatifs ou nuls, l'optimum est donc atteint,
- les variables hors base sont dans la ligne HB : $e_1 = 0$ et $e_2 = 0$,
- les valeurs des variables de base sont dans la colonne R : $x = 600$, $y = 300$, $e_3 = 3\ 800$,
- le tableau (T_2) correspond au point B car $x = 600$ et $y = 300$,
- les résultats (optimum en B et $\Gamma_B = 960\ 000$ €) sont bien ceux obtenus par la résolution graphique.

4) Programme dual :

- à tout programme linéaire (appelé primal) correspond un programme dual

exemple : à $\begin{cases} x + y \le 900 & (t) \\ x + 2\,y \le 1\,200 & (m) \\ 14\,x + 6\,y \le 14\,000 & (e) \end{cases}$ correspond $\begin{cases} t + m + 14\,e \ge 1\,000 & (x) \\ t + 2\,m + 6\,e \ge 1\,200 & (y) \end{cases}$

Max $(1\,000\,x + 1\,200\,y)$ Min $(900\,t + 1\,200\,m + 14\,000\,e)$

remarques : - on peut parfois donner une signification économique au programme dual,
- on passe de contraintes de type inégalité "≤" à des contraintes de type inégalité "≥" et d'une recherche d'un maximum à une recherche d'un minimum,
- à chaque contrainte du primal correspond une variable du dual et à chaque contrainte du dual correspond une variable du primal (voir ci-dessus).

- *Résultats (sans démonstration)* :
  - la variable correspondant à une contrainte saturée est différente de 0
    la variable correspondant à une contrainte non saturée est égale à 0
    } valable dans les 2 sens primal ↔ dual

ici : $14\,x + 6\,y \le 14\,000$ non saturée $\Rightarrow e = 0$
$x \ne 0, y \ne 0 \Rightarrow$ contraintes du dual saturées : $t + m = 1\,000$ et $t + 2\,m = 1\,200$

d'où $m = 200$ et $t = 800$, soit $\Gamma_{dual} = 900\,t + 1\,200\,m = 960\,000 = \Gamma_{primal}$.

  - l'optimum du programme primal correspond à l'optimum du programme dual

II. Méthode du simplexe pour un programme linéaire quelconque

La méthode décrite précédemment ne permet de résoudre que des PML correspondant à la recherche d'un maximum en présence de contraintes linéaires de type inégalité "inférieure ou égale". Il s'agit maintenant d'étendre la méthode au cas général de la recherche d'optimum (maximum ou minimum) en présence de contraintes linéaires de type "inégalité" (≤ ou ≥) ou "égalité".

1) Formes canonique et standard :

- *Forme canonique* : - de type I : contraintes = inégalités "≤", recherche d'un maximum
  - de type II : contraintes = inégalités "≥", recherche d'un minimum

- *Forme mixte* : les contraintes sont de type "inégalité" (≤ ou ≥) ou "égalité", recherche d'un maximum ou d'un minimum.

- *Forme standard* : toutes les équations sont des égalités ; cette forme sert à la résolution algébrique du problème (simplexe).

- *Règles pour obtenir la forme standard* :

- <u>contrainte inégalité "≤"</u> :

  - $x_1 + x_2 \leq 10$ devient $x_1 + x_2 + x_3 = 10$ avec x_3 variable d'écart (VE) ≥ 0

- <u>contrainte inégalité "≥"</u> : $x_1 + x_2 \geq 10$

  - impossible d'introduire seulement une VE x_3 vérifiant $x_1 + x_2 - x_3 = 10$ car la solution initiale $x_1 = 0, x_2 = 0$ impose $x_3 = -10 \leq 0$, ce qui est interdit ;
  - en plus de la VE x_3 on introduit une variable artificielle (VA) x_4 satisfaisant l'équation $x_1 + x_2 - x_3 + x_4 = 10$; la solution initiale du problème comprend 3 variables HB ($x_1 = 0$, $x_2 = 0, x_3 = 0$) et 1 variable dans la base ($x_4 = 10$).

- <u>contrainte égalité</u> : $x_1 + x_2 = 10$

  - on introduit seulement une VA x_3 avec le signe "+" dans l'équation $x_1 + x_2 + x_3 = 10$ car le second membre est positif : ainsi la solution initiale est $x_1 = 0, x_2 = 0$ (HB) et $x_3 = 10$ (B).

- <u>fonction économique Γ</u> :

  - une VE est affectée d'un coefficient 0,
  - une VA est affectée d'un coefficient $+M$ pour la recherche d'un minimum et $-M$ pour la recherche d'un maximum, où M est un nombre très grand positif, de façon à ce que la solution finale donne une VA égale à 0 : par exemple, la recherche de Max "$10\,x_1 + 20\,x_2 - M\,x_3$" oblige la variable x_3 à devenir nulle dans la solution finale.

- **<u>résumé</u>** :

contraintes :	$\leq$	$\Rightarrow$	+ VE
	$\geq$	$\Rightarrow$	- VE $\pm$ VA ("$\pm$" = signe du 2ème membre)
	$=$	$\Rightarrow$	$\pm$ VA ("$\pm$" = signe du 2ème membre)
fonction économique :	$0 \times$ VE		pour Min et Max
	$+M \times$ VA		pour Min
	$-M \times$ VA		pour Max

- *Exemples* : F. Ecoto, Initiation à la recherche opérationnelle, 1986

FORME CANONIQUE	FORME STANDARD CORRESPONDANTE
N° 1 : $x_1 \geq 0 \quad x_2 \geq 0$	N° 1 : $x_1 \geq 0 \quad x_2 \geq 0 \quad x_3 \geq 0 \; x_4 \geq 0 \; x_5 \geq 0$ (V.R. / V.E.)
$20x_1 + 30x_2 \leq 240$ $10x_1 + 25x_2 \leq 500$ $15x_1 + 40x_2 \leq 550$	$20x_1 + 30x_2 + x_3 = 240$ $10x_1 + 25x_2 + x_4 = 500$ $15x_1 + 40x_2 + x_5 = 550$
Max $(150x_1 + 200x_2)$	Max $(150x_1 + 200x_2 + 0x_3 + 0x_4 + 0x_5)$
2 : $x_1 \geq 0 \quad x_2 \geq 0$	2 : $x_1 \geq 0 \quad x_2 \geq 0 \quad x_3 \geq 0 \; x_4 \geq 0 \; x_5 \geq 0 \; x_6 \geq 0$ (V.R. / V.E. / V.A.)
$5x_1 + 3x_2 \leq 15$ $2x_1 + 4x_2 \leq 8$ $x_1 + x_2 \geq 4$	$5x_1 + 3x_2 + x_3 = 15$ $2x_1 + 4x_2 + x_4 = 8$ $x_1 + x_2 - x_5 + x_6 = 4$
Max $(10x_1 + 20x_2)$	Max $(10x_1 + 20x_2 + 0x_3 + 0x_4 + 0x_5 - Mx_6)$

FORME CANONIQUE	FORME STANDARD CORRESPONDANTE
N°	N°
3	3
$x_1 \geqslant 0 \quad x_2 \geqslant 0$	$\underbrace{x_1 \geqslant 0 \quad x_2 \geqslant 0}_{\text{V.R.}} \; \underbrace{x_3 \geqslant 0}_{\text{V.E.}} \; \underbrace{x_4 \geqslant 0}_{\text{V.A.}}$
$\begin{cases} 3x_1 + 5x_2 \leqslant 15 \\ 2x_1 + 3x_2 = 12 \end{cases}$	$\begin{cases} 3x_1 + 5x_2 + x_3 = 15 \\ 2x_1 + 3x_2 + x_4 = 12 \end{cases}$
$\text{Max} (5x_1 + 10x_2)$	$\text{Max} (5x_1 + 10x_2 + 0x_3 - Mx_4)$
4	4
$x_1 \geqslant 0 \quad x_2 \geqslant 0$	$\underbrace{x_1 \geqslant 0 \quad x_2 \geqslant 0}_{\text{V.R.}} \; \underbrace{x_3 \geqslant 0 \; x_4 \geqslant 0}_{\text{V.E.}} \; \underbrace{x_5 \geqslant 0}_{\text{V.A.}}$
$\begin{cases} 4x_1 + 8x_2 \geqslant 24 \\ 8x_1 + 2x_2 = 16 \end{cases}$	$\begin{cases} 4x_1 + 8x_2 - x_3 + x_4 = 24 \\ 8x_1 + 2x_2 + x_5 = 16 \end{cases}$
$\text{Max} (10x_1 + 8x_2)$	$\text{Max} (10x_1 + 8x_2 + 0x_3 - Mx_4 - Mx_5)$
5	5
$x_1 \geqslant 0 \quad x_2 \geqslant 0$	$\underbrace{x_1 \geqslant 0 \quad x_2 \geqslant 0}_{\text{V.R.}} \; \underbrace{x_3 \geqslant 0 \; x_4 \geqslant 0}_{\text{V.E.}} \; \underbrace{x_5 \geqslant 0 \; x_6 \geqslant 0}_{\text{V.A.}}$
$\begin{cases} 7x_1 + 2x_2 \geqslant 14 \\ 8x_1 + 16x_2 \leqslant 16 \\ 2x_1 + 5x_2 = 10 \end{cases}$	$\begin{cases} 7x_1 + 2x_2 - x_3 + x_5 = 14 \\ 8x_1 + 16x_2 + x_4 = 16 \\ 2x_1 + 5x_2 + x_6 = 10 \end{cases}$
$\text{Max} (20x_1 + 15x_2)$	$\text{Max} (20x_1 + 15x_2 + 0x_3 + 0x_4 - Mx_5 - Mx_6)$
6	6
$x_1 \geqslant 0 \quad x_2 \geqslant 0 \quad x_3 \geqslant 0$	$\underbrace{x_1 \geqslant 0 \quad x_2 \geqslant 0 \; x_3 \geqslant 0 \; x_4 \geqslant 0}_{\text{V R.}} \; \underbrace{x_5 \geqslant 0 \; x_6 \geqslant 0}_{\text{V.A.}}$
$\begin{cases} 8x_1 + 5x_2 + 10x_3 = 50 \\ 7x_1 + 8x_2 + 5x_3 = 40 \\ 3x_1 + 15x_2 + 5x_3 = 50 \end{cases}$	$\begin{cases} 8x_1 + 5x_2 + 10x_3 + x_4 = 50 \\ 7x_1 + 8x_2 + 5x_3 + x_5 = 40 \\ 3x_1 + 15x_2 + 5x_3 + x_6 = 50 \end{cases}$
$\text{Max} (10x_1 + 20x_2 + 30x_3)$	$\text{Max} (10x_1 + 20x_2 + 30x_3 - Mx_4 - Mx_5 - Mx_6)$
7	7
$x_1 \geqslant 0 \quad x_2 \geqslant 0$	$\underbrace{x_1 \geqslant 0 \quad x_2 \geqslant 0}_{\text{V.R.}} \; \underbrace{x_3 \geqslant 0 \; x_4 \geqslant 0}_{\text{V.E.}} \; \underbrace{x_5 \geqslant 0 \; x_6 \geqslant 0}_{\text{V.A.}}$
$\begin{cases} 5x_1 + 6x_2 \geqslant 10 \\ 2x_1 + 7x_2 \geqslant 14 \end{cases}$	$\begin{cases} 5x_1 + 6x_2 - x_3 + x_5 = 10 \\ 2x_1 + 7x_2 - x_4 + x_6 = 14 \end{cases}$
$\text{Min} (3x_1 + 10x_2)$	$\text{Min} (3x_1 + 10x_2 + 0x_3 + 0x_4 + Mx_5 + Mx_6)$
8	8
$x_1 \geqslant 0 \quad x_2 \geqslant 0$	$\underbrace{x_1 \geqslant 0 \quad x_2 \geqslant 0}_{\text{V.R.}} \quad \underbrace{x_3 \geqslant 0 \; x_4 \geqslant 0}_{\text{V.E.}} \; \underbrace{x_5 \geqslant 0}_{\text{V.A.}}$
$\begin{cases} 10x_1 + 9x_2 \geqslant 21 \\ 5x_1 + 12x_2 \leqslant 13 \end{cases}$	$\begin{cases} 10x_1 + 9x_2 - x_3 + x_5 = 21 \\ 5x_1 + 12x_2 + x_4 = 13 \end{cases}$
$\text{Min} (x_1 + 3x_2)$	$\text{Min} (x_1 + 3x_2 + 0x_3 + 0x_4 + Mx_5)$
9	9
$x_1 \geqslant 0 \quad x_2 \geqslant 0 \quad x_3 \geqslant 0$	$\underbrace{x_1 \geqslant 0 \quad x_2 \geqslant 0 \; x_3 \geqslant 0 \; x_4 \geqslant 0}_{\text{V.R.}} \; \underbrace{x_5 \geqslant 0 \; x_6 \geqslant 0}_{\text{V.E.}} \; \underbrace{x_7 \geqslant 0}_{\text{V.A.}}$
$\begin{cases} 3x_1 + 5x_2 + 10x_3 \geqslant 100 \\ 4x_1 + x_2 + x_3 \leqslant 50 \\ x_1 + x_2 + 7x_3 = -20 \end{cases}$	$\begin{cases} 3x_1 + 5x_2 + 10x_3 - x_4 + x_6 = 100 \\ 4x_1 + x_2 + x_3 + x_5 = 50 \\ x_1 + x_2 + 7x_3 - x_7 = -20 \end{cases}$
$\text{Min} (9x_1 + 2x_2 + 4x_3)$	$\text{Min} (9x_1 + 2x_2 + 4x_3 + 0x_4 + 0x_5 + Mx_6 + Mx_7)$
10	10
$x_1 \geqslant 0 \quad x_2 \geqslant 0$	$\underbrace{x_1 \geqslant 0 \quad x_2 \geqslant 0}_{\text{V.R.}} \; \underbrace{x_3 \geqslant 0 \; x_4 \geqslant 0 \; x_5 \geqslant 0}_{\text{V.E.}} \; \underbrace{x_6 \geqslant 0 \; x_7 \geqslant 0 \; x_8 \geqslant 0}_{\text{V.A.}}$
$\begin{cases} 2x_1 + x_2 \geqslant 5 \\ x_1 + 2x_2 \geqslant 10 \\ 3x_1 + 7x_2 \geqslant 3 \end{cases}$	$\begin{cases} 2x_1 + x_2 - x_3 + x_6 = 5 \\ x_1 + 2x_2 - x_4 + x_7 = 10 \\ 3x_1 + 7x_2 - x_5 + x_8 = 3 \end{cases}$
$\text{Min} (10x_1 + x_2)$	$\text{Min} (10x_1 + x_2 + 0x_3 + 0x_4 + 0x_5 + Mx_6 + Mx_7 + Mx_8)$

2) Résolution d'un problème de minimisation :

- *Problème* : on choisit le programme dual du programme précédent $\begin{cases} t + m + 14\,e \geq 1\,000 \\ t + 2\,m + 6\,e \geq 1\,200 \end{cases}$

  Min $(900\,t + 1\,200\,m + 14\,000\,e)$

- *Forme standard* : $\begin{cases} t + m + 14\,e - x_1 + x_3 = 1\,000 \\ t + 2\,m + 6\,e - x_2 + x_4 = 1\,200 \end{cases}$ avec 2 VE x_1 et x_2, 2 VA x_3 et x_4

$$\text{Min } \Gamma = 900\,t + 1\,200\,m + 14\,000\,e + 0\,x_1 + 0\,x_2 + M\,x_3 + M\,x_4$$

- *Initialisation* : - variables Hors-Base : $t = 0$, $m = 0$, $e = 0$, $x_1 = 0$, $x_2 = 0$
  - variables dans la base : $x_3 = 1\,000$, $x_4 = 1\,200$

il faut écrire Γ en fonction des variables HB !!

$\Gamma = 900\,t + 1\,200\,m + 14\,000\,e + M\,(1\,000 + x_1 - t - m - 14\,e) + M\,(1\,200 + x_2 - t - 2\,m - 6\,e)$

$\Rightarrow \Gamma = (900 - 2\,M)\,t + (1\,200 - 3\,M)\,m + (14\,000 - 20\,M)\,e + M\,x_1 + M\,x_2 + 2\,200\,M$

- *Tableau initial* (T_0) : correspond au sommet O (en dehors de la ZAS)

HB / B	t	m	e	x_1	x_2	(x_3) •	(x_4) •	R	R'
x_3	1	1	14	-1	0	1	0	1000	1000/14
x_4	1	2	6	0	-1	0	1	1200	1200/6
Γ	900-2M	1200-3M	14000-20M	M	M	0	0	-2200M	

- *Règles de Dantzig* (recherche d'un minimum) :

  - variable entrante (v.e.) : correspond au coefficient le plus < 0 dans Γ (ici e)
  - variable sortante (v.s.) : correspond au coefficient > 0 le plus petit dans R' (ici x_3)

- *Complément sur les règles de Dantzig* (recherche d'un maximum ou minimum):

- 1ère règle : s'il n'y a pas de coefficient > 0 (pour recherche de Max) ou < 0 (pour recherche de Min) dans la ligne de Γ mais qu'il existe un coefficient de variable HB nul, on choisit cette variable comme v. e.

- 2ème règle : s'il y a un coefficient nul dans la colonne R, on considère qu'il vaut ε petit > 0, ce qui donne un terme ε/a_{ij} dans la colonne R' que l'on ne prend en compte que si $a_{ij} > 0$.
  - s'il n'y a pas de terme $R/a_{ij} > 0$ dans la colonne R' mais qu'il en existe un terme tel que $R/a_{ij} = 0$, on le prend en compte (dégénérescence du problème),
  - si tous les termes R/a_{ij} sont < 0 dans la colonne R', on arrête le déroulement de l'algorithme car une variable est infinie (la zone admissible des solutions n'est pas bornée).

  référence : A. Alj et R. Faure, Guide de la RO, tome 2, les applications, Masson 1990, p. 305.

- *1ère itération :* tableau T_1 (v.e. : e, v.s. : x_3)

HB / B	t	m	•	x_1	x_2	x_3	•	R	R'
e	1/14	1/14	**1**	-1/14	0	1/14	0	1000/14	1000
x_4	4/7	11/7	**0**	3/7	-1	-3/7	1	5400/7	5400/11
Γ	-100 -4M/7	200 -11M/7	**0**	1000 -3M/7	M	-1000 +10M/7	0	-10^6 -5400M/7	

2ème itération : tableau T_2 (v.e. : *m*, v.s. : x_4)

HB / B	*t*	•	•	x_1	x_2	x_3	x_4	*R*	*R'*
e	1/22	**0**	1	-1/11	1/22	1/11	-1/22	400/11	800
m	4/11	**1**	0	3/11	-7/11	-3/11	7/11	5400/11	1350
Γ	$\frac{-1900}{11}$	**0**	0	$\frac{10400}{11}$	$\frac{1400}{11}$	-10400/11 +*M*	-1400/11 +*M*	$-12{,}08\times10^6/11$	

remarque : le sommet correspondant au tableau T_2 est dans la ZAS car le facteur *M* n'apparaît plus à l'intersection des colonnes *R* et Γ.

- *3ème itération :* tableau T_3 (v.e. : *t*, v.s. : *e*)

HB / B	•	•	*e*	x_1	x_2	x_3	x_4	*R*
t	**1**	0	22	-2	1	2	-1	800
m	**0**	1	-8	1	-1	-1	1	200
Γ	**0**	0	3800	600	300	-600+*M*	-300+*M*	-960 000

remarque : tous les coefficients de Γ sont ≥ 0, l'optimum est donc atteint ; la colonne *R* donne $t = 800$, $m = 200$, $\Gamma_{max} = 960\,000$ et la ligne HB donne les variables nulles e, x_1, x_2, x_3, x_4. La ligne Γ donne les solutions du problème primal: $x_1 = 600$ et $x_2 = 300$, $e = 3\,800$.

III. Programmation mathématique linéaire paramétrée

- *Problématique*
  - Les coefficients dans les contraintes et la fonction économique sont parfois mal connus : une petite variation de ces paramètres peut parfois modifier l'optimum, ce qui pose un problème quand par exemple cet optimum représente une décision de fabrication, une décision d'utilisation de ressources…
  - La technique de paramétrisation consiste à considérer un des coefficients comme un paramètre et à chercher l'optimum pour toutes les valeurs de ce paramètre ; on teste ainsi successivement chaque coefficient du problème.
- *Exemple : paramétrisation d'un coefficient de Γ dans le PML initial*
  - rappel du dernier tableau obtenu :

$$\begin{cases} x + y \le 900 \\ x + 2\,y \le 1\,200 \\ 14\,x + 6\,y \le 14\,000 \end{cases}$$

Max $\Gamma = 1\,000\,x + 1\,200\,y$

HB / B	•	•	e_1	e_2	•	*R*
x	1	0	2	-1	0	600
y	0	1	-1	1	0	300
e_3	0	0	-22	8	1	3 800
Γ	0	0	-800	-200	0	-960 000

  - on remplace le coefficient $c = 1\,000$ par $c = 1\,000\,(1 + \lambda)$ avec $\lambda \ge -1$

- l'écriture de Γ en fonction des variables HB donne :
  $\Gamma = 1\,000\,(1 + \lambda)\,x + 1\,200\,y = -\,400\,(2 + 5\,\lambda)\,e_1 - 200\,(1 - 5\,\lambda)\,e_2 + 960\,000 - 600\,000\,\lambda$

- 2 valeurs critiques apparaissent pour λ : $\lambda_1 = -2/5 = -0{,}4$ et $\lambda_2 = 1/5 = 0{,}2$

- <u>si $-1 < \lambda < -0{,}4$</u> : il y a un coefficient > 0 dans la ligne de Γ, il faut donc poursuivre l'algorithme avec v.e. : e_1 et v.s. : x

  ⇒ l'optimum est obtenu pour $x = 0$, $y = 600$ (point C) et $\Gamma = 720\,000$.

HB / B	x	•	•	e_2	•	R
e_1	1/2	0	**1**	-1/2	0	300
y	1/2	1	**0**	1/2	0	600
e_3	11	0	**0**	-3	1	10 400
Γ	400 +1000λ	0	**0**	-600	0	-720 000

- <u>si $-0{,}4 < \lambda < 0{,}2$</u> : les coefficients sont < 0 dans la ligne de Γ, l'optimum est obtenu pour $x = 600$, $y = 300$ (point B) et $\Gamma = 960\,000 + 600\,000\,\lambda$.

- <u>si $0{,}2 < \lambda$</u> : il y a un coefficient > 0 dans la ligne de Γ, il faut donc poursuivre l'algorithme avec v.e. : e_2 et v.s. : y

  ⇒ l'optimum est obtenu pour $x = 900$, $y = 0$ (point A) et $\Gamma = 900\,000\,(1 + \lambda)$.

HB / B	•	y	e_1	•	•	R
x	1	1	1	**0**	0	900
e_2	0	1	-1	**1**	0	300
e_3	0	-8	-14	**0**	1	1400
Γ	0		-1000 -1000λ	**0**	0	-900 000 (1+λ)

- <u>conclusions</u> :

- la décision optimale de répartition des cultures dépend du coefficient c :

- si $c < 600$:
  $x_{opt} = 0,\ y_{opt} = 600$
- si $600 < c < 1\,200$:
  $x_{opt} = 600,\ y_{opt} = 300$
- si $1\,200 < c$:
  $x_{opt} = 900,\ y_{opt} = 0$

- le revenu national optimal est donné dans le graphique ci-contre.