

INF2220

Optimisation Avancée

Master Informatique

M1 Semestre 2

Printemps 2025

MANIER Marie-Ange

Intervenants

		C	TD
• Marie-Ange MANIER	- UTBM (Univ. de Technologie de Belfort-Montbéliard)	5 h	8 h
• Marie-Ange MANIER	- UTBM + Yvette GBEDEVI - UL (univ. Lomé)	6 h	8 h
• Nadjime PINDRA	- UL	5 h	8 h

Sommaire

- Introduction générale
- Méthodes exactes : programmation linéaire, principes des méthodes de type branch and bound....
- Méthodes approchées :
 - Introduction et classification des méthodes heuristiques
 - Heuristiques
 - Recherche locale : gradient, recuit simulé, tabou ...
 - Métaheuristiques : algorithmes génétiques et évolutionnistes, algorithmes de colonies de fourmis, optimisation par essaims particulaires, algorithmes immunitaires, algorithmes quantiques ...
- Les algorithmes hybrides
- Optimisation multi-objectifs : méthodes Pareto, lexicographique, ...

Partie 1

5 h Cours

8 h TD

Marie-Ange MANIER

- Introduction générale
sur la Recherche Opérationnelle et les méthodes d'optimisation
- Méthodes exactes
programmation linéaire
principes des méthodes de type branch and bound....

Préambule

Marie-Ange MANIER

Professeure des Universités

e-mail : marie-ange.manier@utbm.fr

Enseignement:

Etablissement: UTBM (Univ. de Technologie de Belfort-Montbéliard)

Ex Responsable filière Logistique et Organisation Industrielle, Spécialité Systèmes Industriels, Pôle Industrie 4.0

Co-montage du parcours Logistique du Master Informatique, univ. Lomé

Disciplines enseignées : Ordonnancement, logistique, **recherche opérationnelle**, gestion de projets, gestion de production, implantation, gestion de stocks, simulation de flux...

Recherche:

Laboratoire : Institut FEMTO-ST (UMR CNRS)

~700 personnes, 7 départements scientifiques et techniques + 1 équipe SHS

Département Informatique des Systèmes Complexes (DISC)

Equipe OMNI

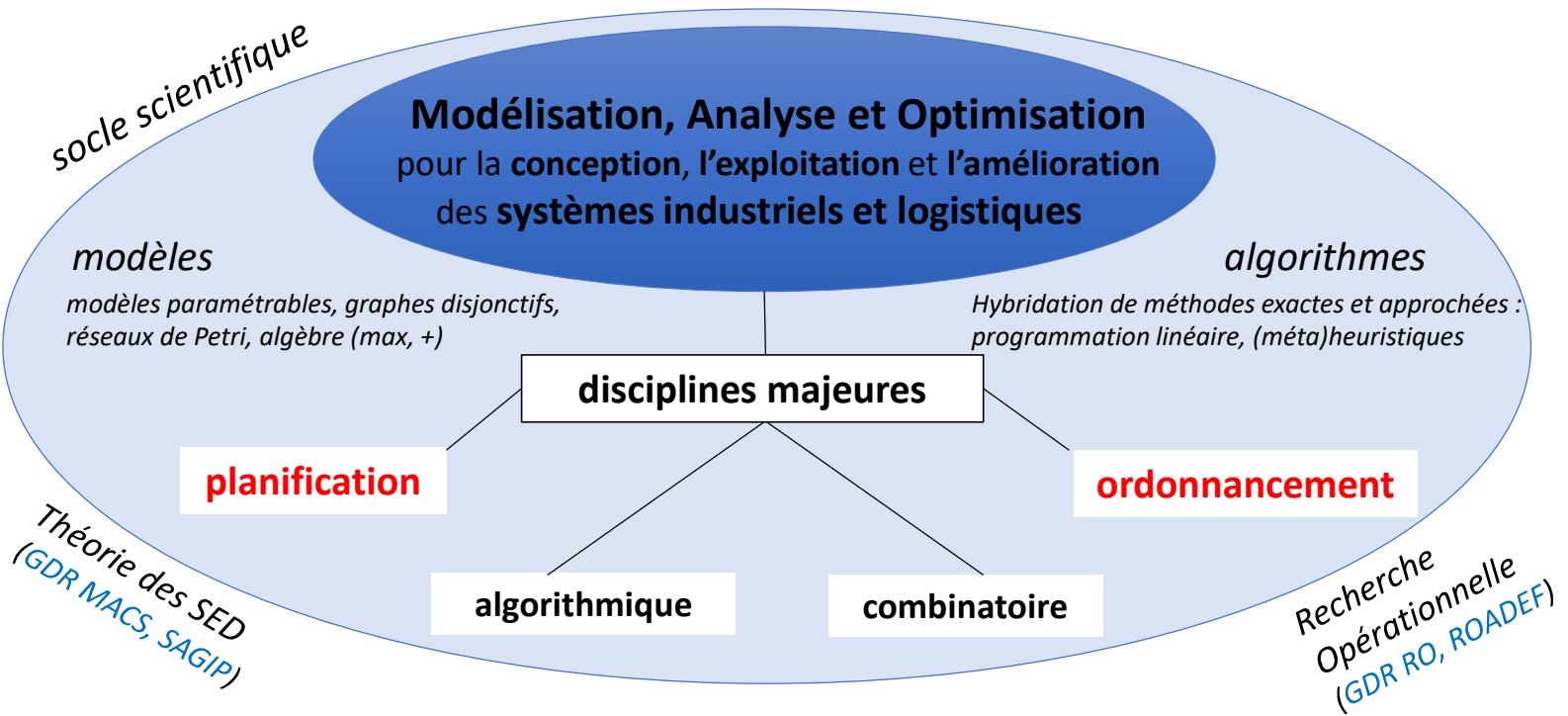
Responsable de l'axe Planification et Ordonnancement

Directrice adjointe du laboratoire

Thématisques de recherche : voir slides suivants

Laboratoire de recherche: Institut FEMTO-ST UMR CNRS
Département DISC, équipe OMNI
Axe Planification et Ordonnancement

Thématisques

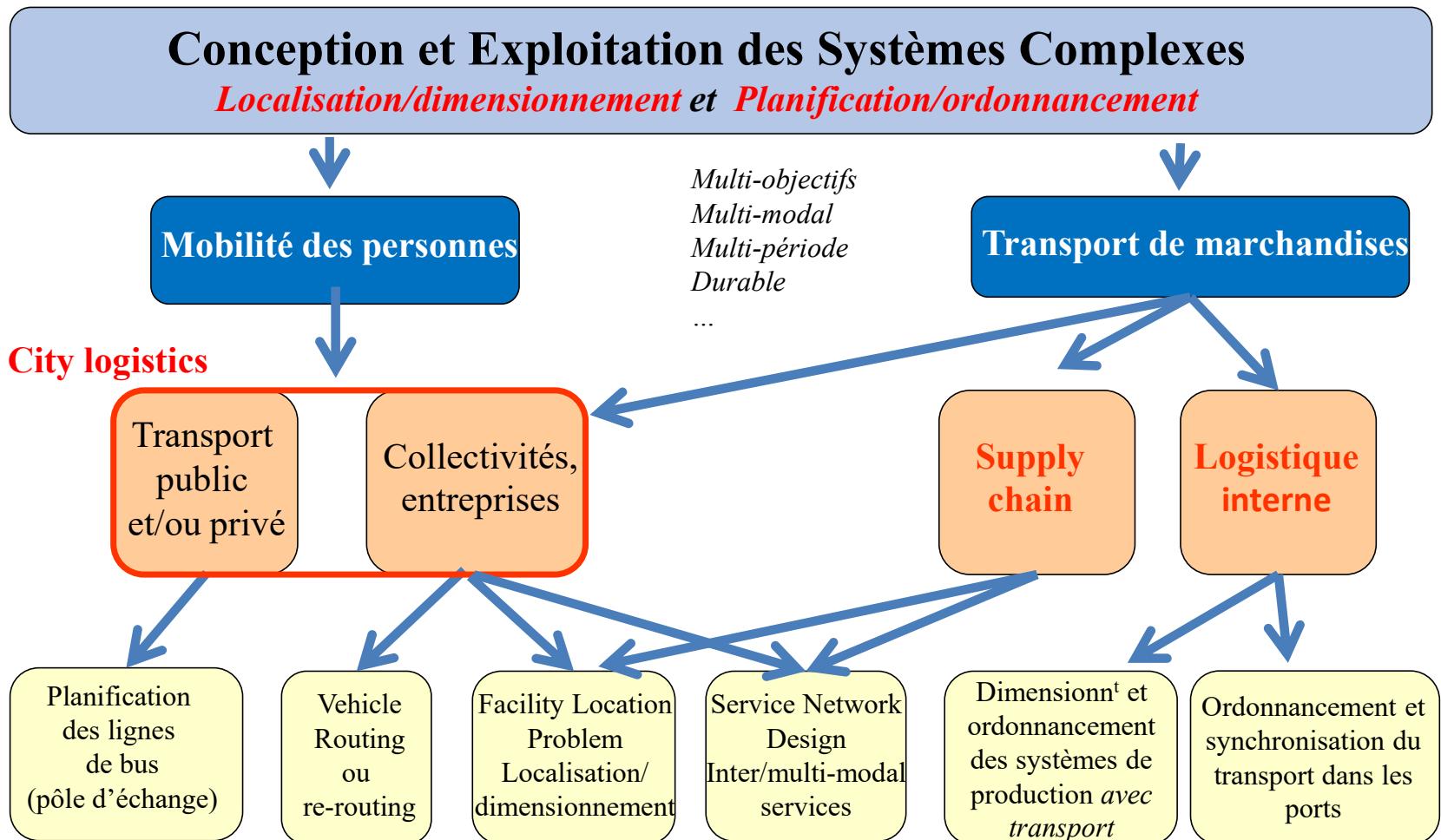


Verrous scientifiques

Dimensionnement, localisation, gestion dynamique et intégrée des flux (informations, personnes, marchandises) :
affectation, séquencement... *sous contraintes multiples et complexes*

Applications

Vue
d'ensemble



Domaines d'application

Légende année thèse:

- En noir
- En vert
- Souligné
- Entouré**

année de soutenance
thèse en cours-année de soutenance prévue
cotutelle internationale
doctorant togolais

Ordonnancement

Usine ... du Futur



Ordonnancement
d'ateliers avec transport



1998 1999 **2011** 2012



2012 **2014** 2021 **2024**



2026 **2026** 2028

Logistique portuaire



Ordonnancement des
tâches de manutention



2020



2026

Planification

Smart Cities



Logistique et
Mobilité urbaines

5 thèses (dont **1 CIFRE**)

Mobilité



2003 **2017**

Transport



2018 2019

2020

Energies renouvelables



Conception de la
chaîne logistique H2

2 thèses



2020



2024

Agriculture durable



... Planification des cultures
mixtes/intercalaires

2022-...

collaboration avec
l'université de Lomé -
Togo

Partie 1

Introduction générale

Définitions

L'optimisation est une **branche des mathématiques**, dont le but est de trouver analytiquement ou numériquement, la meilleure solution (l'optimale) à un problème donné. L'origine du mot *optimal* provient du Latin *optimum* qui signifie **le meilleur**.

En optimisation, on parle de la **fonction coût (objectif)**. C'est la fonction à minimiser/maximiser, après formulation mathématique du problème.

Un **problème d'optimisation** consiste, étant donnée une fonction $f: S \rightarrow \mathbb{R}$, à trouver :

- 1) son minimum v (resp. son maximum) dans S
- 2) un point $x_0 \in S$ qui réalise ce minimum (resp. maximum) i.e. $f(x_0) = v$.

Vocabulaire

- f est la fonction objectif
- v est la valeur optimale
- x_0 est la solution optimale
- $S = \{\text{solutions réalisables du problème}\}$
- écriture du problème : $\min_{x \in S} f(x)$ resp. $\max_{x \in S} f(x)$

Pour **résoudre** un problème d'optimisation, on peut avoir recours à des méthodes de Recherche Opérationnelle.

Définition

La **Recherche Opérationnelle (RO)** est la discipline des **méthodes scientifiques** utilisables **pour faciliter la prise de décisions** face à des problématiques qui se rencontrent dans les grandes organisations publiques ou privées. Discipline transverse associant **les mathématiques appliquées, les statistiques et l'informatique**, elle s'applique à des problèmes usuels et joue un rôle-clé dans la recherche de l'efficience.

La recherche opérationnelle permet notamment d'optimiser l'architecture et le fonctionnement des organisations. Grâce à elle, les décideurs peuvent analyser et mieux comprendre des situations complexes ou de grande dimension, aux interactions nombreuses et donc, faire des choix pertinents en toute connaissance de cause. Elle **participe à l'aide à la décision**.

Un peu d'histoire

Il est généralement admis que les premiers travaux de recherche opérationnelle ont été entrepris en Grande-Bretagne au début de la Seconde Guerre mondiale.

« **Recherche Opérationnelle** » : traduction littérale de « *Operational Research* », terme employé pour la première fois en Angleterre par Sir Robert Watson-Watt* pour désigner la recherche scientifique du rendement optimum d'une opération militaire pendant la Seconde Guerre mondiale.

*Sir Watson-Watt : premier directeur d'un centre de recherche sur les radars

Exemple d'application : Blackett (1940) pour optimiser l'emplacement des radars pour prévenir l'arrivée des bombardiers allemands.

Développement de la RO:

- Fin années 30 : émergence puis évolution de cette discipline dans les armées
- 2^{ème} moitié du XX^e siècle : engouement aussi bien dans les armées occidentales que dans le milieu civil.
- Années 70-80 : désenchantement car manque d'outils suffisamment puissants pour résoudre de manière efficace les problèmes industriels. ET pourtant, la recherche opérationnelle commence à bénéficier des **apports de l'informatique** permettant d'effectuer des calculs pour des problèmes de taille raisonnable.
- Années 90 jusqu'à maintenant :

* explosion des logiciels aussi bien scientifiques que commerciaux

* une communauté de chercheurs se structure et s'accroît progressivement.

La profusion des travaux de recherche dans le domaine de la recherche opérationnelle et l'aide à la décision conduit à la mise en application industrielle de nouvelles méthodes et l'émergence de nouveaux champs d'application.

La RO-AD (Recherche Opérationnelle-Aide à la Décision) est omniprésente dans des secteurs de plus en plus variés comme :

- l'industrie,
- le transport,
- la santé,
- les télécommunications,
- la distribution,
- la banque,
- la finance,
- l'assurance,
- l'informatique...

La RO s'appuie sur la recherche académique et sur la création de structures spécialisées à l'intérieur des grands groupes industriels mais aussi d'entreprises particulières.

Les chercheurs opérationnels ont été formés dans les mêmes écoles et entretiennent des liens privilégiés par l'intermédiaire de la sous-traitance de certaines études, du monde associatif (ROADEF* en France) et de groupements de recherche (GDR ROD** en France).

* ROADEF : société française de recherche opérationnelle et d'aide à la décision, créée en 1998.

Elle compte plusieurs centaines d'adhérents, académiques et industriels.

** GDR ROD : Groupement de Recherche du CNRS: Opérationnelle et Décision

Etapes d'une étude de Recherche Opérationnelle

- modélisation du problème :

à partir d'un problème concret, bâtir un **modèle scientifique** (en général mathématique, mais aussi graphique...) représentant schématiquement la réalité ;

- résolution du problème : résoudre le **modèle** ainsi construit.

résolution = mise en oeuvre de méthodes numériques permettant d'obtenir des réponses effectives, en général à l'aide d'ordinateurs, aux questions posées ;

- retour au problème pratique et **confrontation** des résultats du modèle et de la réalité.

Modélisation mathématique :

1. Variables de décision : $x \in \mathbb{R}^n$,

$$x = \begin{pmatrix} x_1 \\ x_2 \\ \vdots \\ x_n \end{pmatrix} = (x_1, x_2, \dots, x_n)^T$$

2. Fonction objectif : $f(x) \in \mathbb{R}$

3. Contraintes: $x \in X \subseteq \mathbb{R}^n$

- Exemple 1:**
- Un projectile est lancé verticalement à la vitesse de 50 mètres par seconde, en l'absence de vent.
 - Après combien de temps et à quelle altitude commencera-t-il à retomber ?

Variables de décision x = nombre de secondes écoulées depuis le départ du projectile.

Exemple 2 :

Fabricants de jouets en bois : soldats et trains

- Prix de vente : soldats 27 €, trains 21 €
- Matériel brut : soldats 10 €, trains 9 €
- Coûts généraux : soldats 14 €, trains 10 €
- Menuiserie : soldats 1h, trains 1h
- Finissage : soldats 2h, trains 1h
- Main d'oeuvre disponible : menuiserie 80h, finissage 100h
- Maximum de 40 soldats

Variables de décision :

x_1 nombre de soldats fabriqués

x_2 nombre de trains fabriqués

Modèle :

$$\max_x f(x) = 3x_1 + 2x_2,$$

sous contraintes

$$\begin{array}{rcl} 2x_1 + x_2 & \leq & 100 \\ x_1 + x_2 & \leq & 80 \\ x_1 & \leq & 40 \\ \hline x_1 & \geq & 0 \\ x_2 & \geq & 0 \\ x_1 & \in & \mathbb{N} \\ x_2 & \in & \mathbb{N} \end{array}$$

Objectif : **Maximiser le bénéfice total**

avec bénéfice = prix de vente – coût matériel – coûts généraux

donc pour 1 soldat, bénéfice = 27€ - 10€ - 14€ = 3€

pour 1 train, bénéfice = 21€ - 9€ - 10€ = 2€

pour x_1 soldats et x_2 trains fabriqués,

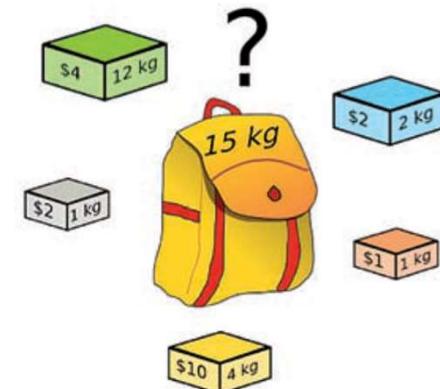
bénéfice total = 3 x_1 + 2 x_2

Les grandes classes de problèmes d'optimisation

Considérons deux cas assez différents:

1 Des marins malchanceux sont contraints d'abandonner leur navire qu'une voie d'eau destine aux profondeurs. Leur seule chance de survie est d'embarquer dans un canot pour tenter de croiser une route commerciale fréquentée. Ils doivent emmener de quoi survivre quelques jours : eau, nourriture, protection contre le froid, signaux de détresse, etc. Or le canot a une capacité limitée en volume et en poids. Ces marins vont donc devoir faire des choix pour charger leur canot, sinon ils n'iront pas loin.

2 Soit le responsable d'un budget. Considérant une espérance de revenus, comment répartir efficacement l'argent disponible entre différents projets sachant qu'ils ont une utilité différente ? Doit-il favoriser les gros projets à forte valeur ajoutée ou les petits projets un peu moins intéressants financièrement ?



Ces deux problèmes semblent différents tant dans leur formulation que dans leur portée, mais ils sont de même nature. Ils appartiennent à la catégorie de problèmes d'optimisation appelés « **sac à dos** », avec des **éléments de valeurs différentes** (matériels à charger dans le canot, projets à financer) qui se partagent une **capacité limitée** (le canot, le budget).

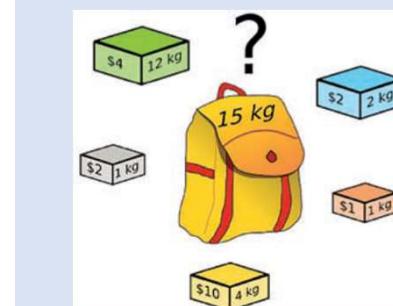
Le premier problème est plus complexe que le second car il introduit un lien entre les éléments à charger : si les marins prennent des boîtes de conserve, il faut également prendre l'ouvre boîte.

Le second problème peut être plus complexe quand certains projets sont réalisables seulement si d'autres le sont.

Les grandes classes de problèmes d'optimisation

Problèmes parmi les plus courants, qui couvrent la majeure partie des cas concrets pour lesquels la recherche opérationnelle est à même de donner une réponse pertinente :

- les problèmes de sac à dos, 1
- les problèmes de tournées, 2
- les problèmes d'affectation, 3
- les problèmes d'ordonnancement, 4
- les problèmes de file d'attente, 5
- les problèmes de flot. 6



1. Problème du sac à dos.

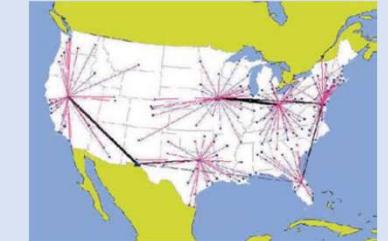
Exemples



1. Chargement optimal d'un camion.



2. Livraison de colis



3. Positionnement aux USA



4. Chaîne d'assemblage d'avions



5. Files d'attente de voitures.

Ressource	Locataire	1/1/2010	2/1/2010	3/1/2010	4/1/2010	5/1/2010	6/1/2010	7/1/2010	8/1/2010	9/1/2010	10/1/2010	11/1/2010	12/1/2010	13/1/2010	14/1/2010	15/1/2010	16/1/2010	17/1/2010	18/1/2010	19/1/2010	20/1/2010	21/1/2010	22/1/2010	23/1/2010	24/1/2010	25/1/2010	26/1/2010	27/1/2010	28/1/2010	29/1/2010	30/1/2010	31/1/2010
SCA	SCA																															
SCE	SCE																															
CUI	CUI																															
CM2	CM2																															
FRMST	FRMST																															
PRM2	PRM2																															
PERC	PERC																															
TAR	TAR																															
TOUR1	TOUR1																															
TOUR2	TOUR2																															
POLM2	POLM2																															
POLM1	POLM1																															
CPF	CPF																															
DHEI	DHEI																															
EHEI	EHEI																															
VS	VS																															



4. Ordonnancement de tâches.



6. Optimisation du trafic urbain

1) Domaine combinatoire :

le but est d'éviter l'énumération des solutions

(affecter 25 secrétaires à 25 postes $\rightarrow 25! \approx 1,5 \cdot 10^{25}$ solutions : 1 solution/ μs $\rightarrow 5 \cdot 10^9$ 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 de personnes possible.
- * *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* (grandeur aléatoire : 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* (grandeur aléatoire : 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.

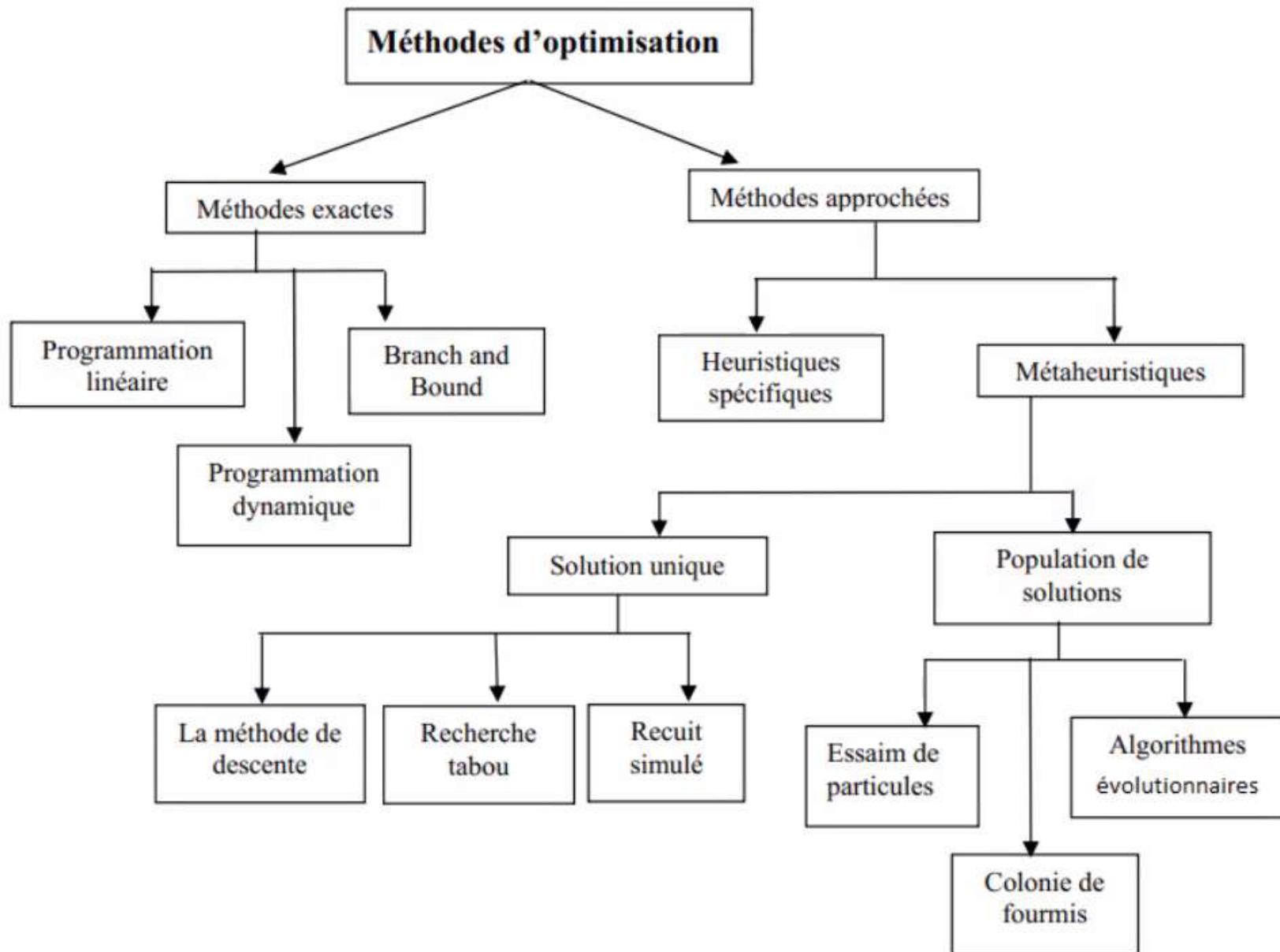
Les méthodes d'optimisation

Deux grandes familles de méthodes :

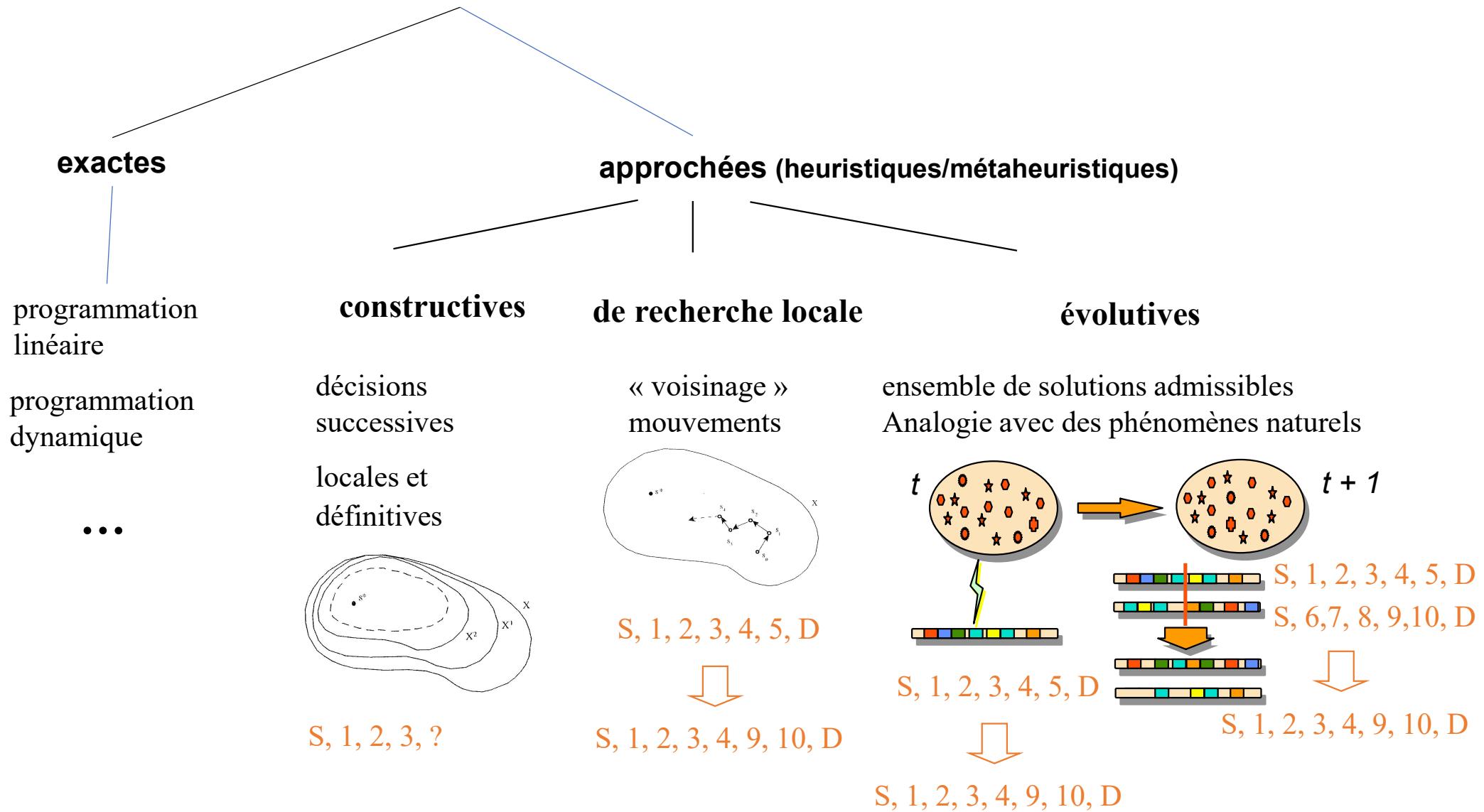
- Méthodes **exactes** *basées sur des principes mathématiques*
 \Rightarrow optimalité garantie

 programmation* mathématique (**programmation linéaire**, programmation dynamique...)
 procédures d'exploration arborescentes (*branch and bound...*),
 ...
 * *initialement, programmation signifiait planification et ordonnancement*
- Méthodes **approchées** ou heuristiques
 souvent stochastiques (utilisation du hasard) : quand les méthodes exactes ne sont pas disponibles ou sont trop coûteuses) - souvent le cas !
 \Rightarrow problèmes de grande taille, problèmes difficiles
 \Rightarrow optimalité non garantie, recherche d'une bonne solution réalisable
 en un temps raisonnable

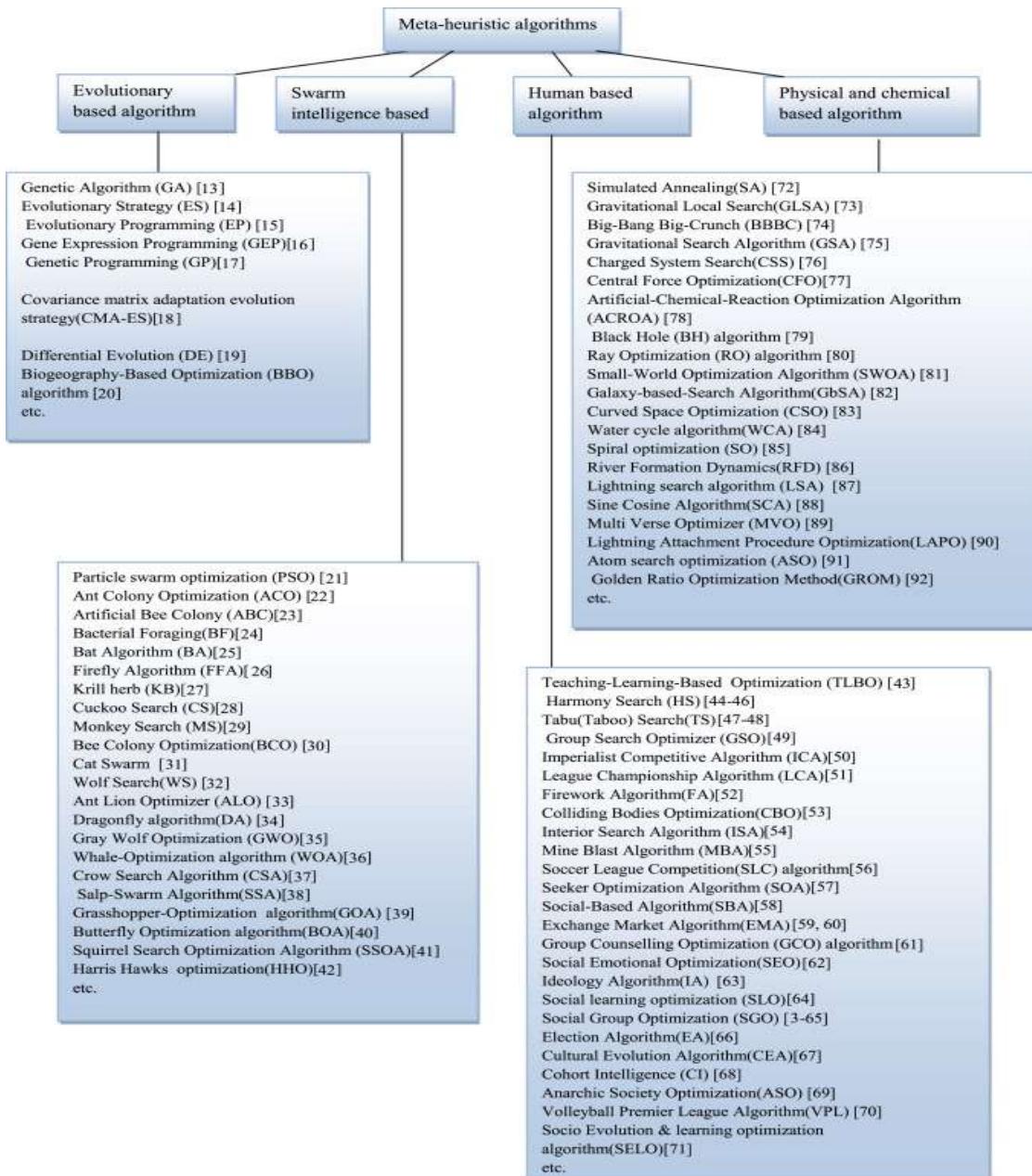
 heuristiques, méthodes déterministes et stochastiques, hybridation de méthodes, ...
 (tabou, recuit simulé, algorithmes génétiques, colonies de fourmis,...)



méthodes d'optimisation



Les méthodes d'optimisation



Comparaison des performances

	Méthodes exactes	Méthodes approchées		
	Exact Strategies	Heuristic Strategies	Metaheuristics Strategies	Probabilistic Strategies
Optimal Result	Guaranteed	Not guaranteed	Not guaranteed	Not guaranteed
Correct Result	Guaranteed	Guaranteed	Guaranteed	Not guaranteed
Execution Time	Typically higher than the other alternatives	Typically fast	Tuning-dependent but typically fast	Probably fast
Notes	Mostly used when the optimal solution is strictly necessary	Problem-oriented algorithms	Generic algorithms adapted to solve specific problems	Three main categories: -Monte Carlo -Las Vegas -Sherwood

Programmation linéaire

Présentation

La **programmation linéaire** est un outil très puissant de la **recherche opérationnelle**, capable de résoudre un grand nombre de problèmes.

Une fois un problème **modélisé** sous la forme d'équations linéaires, des méthodes assurent la résolution du problème de manière exacte.

On distingue :

- la programmation linéaire en nombres réels, pour laquelle les variables des équations sont dans **IR+**
- et la programmation en nombres entiers, pour laquelle les variables sont dans **IN**.

Bien entendu, il est possible d'avoir les deux en même temps.

Cependant, la résolution d'un problème avec des variables entières est nettement plus compliquée qu'un problème en nombres réels.

Une des méthodes les plus connues pour résoudre des programmes linéaires en nombre réels est la **méthode du Simplexe**.

En théorie, elle a une complexité non polynomiale et est donc supposée peu efficace.

En pratique, il s'avère au contraire qu'il s'agit d'une bonne méthode.

De plus, il existe de nombreux logiciels intégrant cette méthode.

Certains sont utilisés via une interface graphique alors que d'autres permettent une communication par fichiers ce qui autorise l'utilisation du programme de manière cachée dans le développement d'un autre logiciel.

Programme linéaire

La programmation linéaire permet la résolution d'un programme linéaire.

Un **programme linéaire** est un système d'équations ou d'inéquations appelées "**contraintes**" qui sont linéaires (c'est-à-dire que les **variables** ne sont pas élevées au carré, ne servent pas d'exposant, ne sont pas multipliées entre elles...). A partir de ces contraintes, on doit optimiser (minimiser ou maximiser) une fonction également linéaire appelée **objectif**.

Exemples

*Contraintes linéaires :

$$5x_1 - 2x_2 + 4x_3 = 8$$

$$x_1 + 3x_2 + 8x_3 \geq 25$$

$$9x_1 + 6x_2 - 3x_3 \leq 17$$

*Contraintes non linéaires :

$$5x_1^2 + 2x_2 + 4x_3 = 8$$

$$x_1 x_2 + 8x_3 \geq 25$$

$$x_i \in \mathbb{IN}$$

*Objectifs :

$$\max: z = 3x_1 - 2x_2 + 8x_3$$

(signifie maximiser z)

$$\min: z = -3x_1 + x_3$$

(signifie minimiser z)

Par ailleurs, les **variables** peuvent prendre n'importe quelle **valeur réelle** respectant les contraintes linéaires (**continuité**)

Exemple: problème de production de yaourt

Un fabricant produit 2 types de yaourts à la fraise A et B à partir de 3 ingrédients : Fraise, de Lait et de Sucre. Chaque yaourt doit respecter les proportions suivantes de matières premières (exprimées en Kg).

	A	B
Fraise	2	1
Lait	1	2
Sucre	0	1

On dispose de 800 Kg de Fraises, 700 Kg de Lait et 300 Kg de sucre.

La vente de 1 Kg de yaourts A et B rapporte respectivement 4 euros et 5 euros.

Le fabricant cherche à maximiser son profit.

Modélisation :

- Sur quelles quantités peut-on travailler?
- Que cherche-t-on à optimiser?
- Quelles sont les contraintes du problème?

Exemple: problème de production de yaourt

Un fabricant produit 2 types de yaourts à la fraise A et B à partir de 3 ingrédients : Fraise, de Lait et de Sucre. Chaque yaourt doit respecter les proportions suivantes de matières premières (exprimées en Kg).

	A	B
Fraise	2	1
Lait	1	2
Sucre	0	1

On dispose de 800 Kg de Fraises, 700 Kg de Lait et 300 Kg de sucre.

La vente de 1 Kg de yaourts A et B rapporte respectivement 4 euros et 5 euros.

Le fabricant cherche à maximiser son profit.

Modélisation :

- Sur quelles quantités peut-on travailler?
 - Seules valeurs non constantes : les quantités de yaourts A et B produites
 - On parle de **variables**
 - On les notera x_A et x_B
- Que cherche-t-on à optimiser?
- Quelles sont les contraintes du problème?

Exemple: problème de production de yaourt

Un fabricant produit 2 types de yaourts à la fraise A et B à partir de 3 ingrédients : Fraise, de Lait et de Sucre. Chaque yaourt doit respecter les proportions suivantes de matières premières (exprimées en Kg).

	A	B
Fraise	2	1
Lait	1	2
Sucre	0	1

On dispose de 800 Kg de Fraises, 700 Kg de Lait et 300 Kg de sucre.

La vente de 1 Kg de yaourts A et B rapporte respectivement 4 euros et 5 euros.

Le fabricant cherche à maximiser son profit.

Modélisation :

- Sur quelles quantités peut-on travailler? variables: x_A et x_B
- Que cherche-t-on à optimiser?
 - Le profit z , calculé à partir de x_A et x_B
 - On parle de **fonction objectif**: $z = 4x_A + 5x_B$
- Quelles sont les contraintes du problème?

Exemple: problème de production de yaourt

Un fabricant produit 2 types de yaourts à la fraise A et B à partir de 3 ingrédients : Fraise, de Lait et de Sucre. Chaque yaourt doit respecter les proportions suivantes de matières premières (exprimées en Kg).

	A	B
Fraise	2	1
Lait	1	2
Sucre	0	1

On dispose de **800 Kg de Fraises**, 700 Kg de Lait et 300 Kg de sucre.

La vente de 1 Kg de yaourts A et B rapporte respectivement 4 euros et 5 euros.

Le fabricant cherche à maximiser son profit.

Modélisation :

- Sur quelles quantités peut-on travailler? variables: x_A et x_B
- Que cherche-t-on à optimiser? $\text{Max } z = 4 x_A + 5 x_B$
- **Quelles sont les contraintes du problème?**
 - Première contrainte : **800 Kg de fraises disponibles**
 - La quantité utilisée dépend de la production : $2 x_A + x_B$
 $\Rightarrow 2 x_A + x_B \leq 800$

Exemple: problème de production de yaourt

Un fabricant produit 2 types de yaourts à la fraise A et B à partir de 3 ingrédients : Fraise, de Lait et de Sucre. Chaque yaourt doit respecter les proportions suivantes de matières premières (exprimées en Kg).

	A	B
Fraise	2	1
Lait	1	2
Sucre	0	1

On dispose de 800 Kg de Fraises, 700 Kg de Lait et 300 Kg de sucre.

La vente de 1 Kg de yaourts A et B rapporte respectivement 4 euros et 5 euros.

Le fabricant cherche à maximiser son profit.

Modélisation :

- Sur quelles quantités peut-on travailler? variables: x_A et x_B
- Que cherche-t-on à optimiser? Max $z = 4x_A + 5x_B$
- Quelles sont les contraintes du problème? $2x_A + x_B \leq 800$ (fraises)
 $x_A + 2x_B \leq 700$ (lait)
 $x_B \leq 300$ (sucre)
 $x_A, x_B \geq 0$ positivité

Exemple: problème de production de yaourt

Un fabricant produit 2 types de yaourts à la fraise A et B à partir de 3 ingrédients : Fraise, de Lait et de Sucre. Chaque yaourt doit respecter les proportions suivantes de matières premières (exprimées en Kg).

	A	B
Fraise	2	1
Lait	1	2
Sucre	0	1

On dispose de 800 Kg de Fraises, 700 Kg de Lait et 300 Kg de sucre.

La vente de 1 Kg de yaourts A et B rapporte respectivement 4 euros et 5 euros.

Le fabricant cherche à maximiser son profit.

Programme linéaire : $\max 4x_A + 5x_B$

$$2x_A + x_B \leq 800$$

$$x_A + 2x_B \leq 700$$

$$x_B \leq 300$$

$$x_A, x_B \geq 0$$

Forme canonique

Pour résoudre un programme linéaire de manière automatique, il faut qu'il ait une certaine forme que l'on appelle **canonique**.

Ici, on a choisi la forme canonique suivante : un programme linéaire n'a que des contraintes d'infériorité et l'on tente de maximiser la fonction objectif. De plus toutes les variables sont positives.

Exemple

Le programme linéaire suivant est sous forme canonique.

$$\text{max: } z = 3x_1 - 2x_2 + 8x_3$$

$$\begin{aligned} \text{sous: } & 5x_1 - 2x_2 + 4x_3 \leq 8 \\ & x_1 + 3x_2 + 8x_3 \leq 25 \\ & 9x_1 + 6x_2 - 3x_3 \leq 17 \\ & x_1, x_2, x_3 \geq 0 \end{aligned}$$

Généralisation

Programme linéaire avec n variables x_1, \dots, x_n et m contraintes.

$$\begin{aligned} \max & \quad \sum_{i=1}^n c_i x_i \\ \text{sous les contraintes } & \quad \sum_{i=1}^n a_{ij} x_i \leq b_j, (j = 1, \dots, m) \\ & \quad x_i \in \mathbb{R}, (i = 1, \dots, n) \end{aligned}$$

Transformations

Equivalence

Deux problèmes d'optimisation P1 et P2 sont dits équivalents si

- l'on peut construire un point admissible de P2 à partir d'un point admissible de P1 (et réciproquement),
- avec la même valeur pour la fonction objectif.

En particulier, les deux problèmes ont le même coût optimal, et on peut construire une solution optimale de P2 à partir d'une solution optimale de P1 (et réciproquement).

Transformations (*règles de réécriture*)

Tout programme linéaire quelconque peut être ramené à une forme **canonique**.

Voici quelques **exemples** de transformations possibles.

*Si une variable x_1 est **négative**, on la remplace par une variable positive $x_1' = -x_1$. Par exemple:

$$\begin{array}{l} \text{max: } z = 3x_1 - 2x_2 + 8x_3 \\ \text{sous:} \\ 5x_1 - 2x_2 + 4x_3 \leq 8 \\ x_1 + 3x_2 + 8x_3 \leq 25 \\ 9x_1 + 6x_2 - 3x_3 \leq 17 \\ x_1, x_2 \geq 0 \text{ et } x_3 \leq 0 \end{array} \quad \Leftrightarrow \quad \begin{array}{l} 5x_1 - 2x_2 + 4x_3 \leq 8 \\ x_1 + 3x_2 + 8x_3 \leq 25 \\ 9x_1 + 6x_2 - 3x_3 \leq 17 \\ x_1, x_2 \geq 0 \text{ et } x_3 \leq 0 \end{array}$$

$$\begin{array}{l} \text{max: } z = 3x_1 - 2x_2 - 8x_3' \\ \text{sous:} \\ 5x_1 - 2x_2 - 4x_3' \leq 8 \\ x_1 + 3x_2 - 8x_3' \leq 25 \\ 9x_1 + 6x_2 + 3x_3' \leq 17 \\ x_1, x_2, x_3' \geq 0 \end{array}$$

*Si une variable (réelle) n'a **pas de contrainte de signe**, on la remplace par deux variables positives x_1' et x_1'' telles que $x_1 = x_1' - x_1''$. Par exemple:

$$\begin{array}{l} \text{max: } z = 3x_1 - 2x_2 + 8x_3 \\ \text{sous:} \\ 5x_1 - 2x_2 + 4x_3 \leq 8 \\ x_1 + 3x_2 + 8x_3 \leq 25 \\ 9x_1 + 6x_2 - 3x_3 \leq 17 \\ x_1, x_2 \geq 0 \end{array} \quad \Leftrightarrow \quad \begin{array}{l} 5x_1 - 2x_2 + 4x_3' - 4x_3'' \leq 8 \\ x_1 + 3x_2 + 8x_3' - 8x_3'' \leq 25 \\ 9x_1 + 6x_2 - 3x_3' + 3x_3'' \leq 17 \\ x_1, x_2, x_3', x_3'' \geq 0 \end{array}$$

*Si le programme linéaire a une **contrainte de supériorité**, on la remplace par une contrainte d'infériorité en inversant le signe des constantes. Par exemple:

$$\text{max: } z = 3x_1 - 2x_2 + 8x_3$$

sous:

$$5x_1 - 2x_2 + 4x_3 \leq 8$$

$$x_1 + 3x_2 + 8x_3 \leq 25$$

$$9x_1 + 6x_2 - 3x_3 \geq 17$$

$$x_1, x_2, x_3 \geq 0$$



$$\text{max: } z = 3x_1 - 2x_2 + 8x_3$$

sous:

$$5x_1 - 2x_2 + 4x_3 \leq 8$$

$$x_1 + 3x_2 + 8x_3 \leq 25$$

$$-9x_1 - 6x_2 + 3x_3 \leq -17$$

$$x_1, x_2, x_3 \geq 0$$

*Si le programme linéaire a une **contrainte d'égalité**, on la remplace par deux contraintes équivalentes, l'une d'infériorité, l'autre de supériorité. Les variables du programme doivent satisfaire ces deux contraintes, ce qui revient alors à l'égalité de départ. Par exemple:

$$\text{max: } z = 3x_1 - 2x_2 + 8x_3$$

sous:

$$5x_1 - 2x_2 + 4x_3 \leq 8$$

$$x_1 + 3x_2 + 8x_3 \leq 25$$

$$9x_1 + 6x_2 - 3x_3 = 17$$

$$x_1, x_2, x_3 \geq 0$$



$$\text{max: } z = 3x_1 - 2x_2 + 8x_3$$

sous:

$$5x_1 - 2x_2 + 4x_3 \leq 8$$

$$x_1 + 3x_2 + 8x_3 \leq 25$$

$$9x_1 + 6x_2 - 3x_3 \leq 17$$

$$9x_1 + 6x_2 - 3x_3 \geq 17$$

$$x_1, x_2, x_3 \geq 0$$



$$\text{max: } z = 3x_1 - 2x_2 + 8x_3$$

sous:

$$5x_1 - 2x_2 + 4x_3 \leq 8$$

$$x_1 + 3x_2 + 8x_3 \leq 25$$

$$9x_1 + 6x_2 - 3x_3 \leq 17$$

$$-9x_1 - 6x_2 + 3x_3 \leq -17$$

$$x_1, x_2, x_3 \geq 0$$

*Tout problème de minimisation peut s'écrire comme un problème de maximisation (et vice-versa). Par exemple:

$$\max: z = 3x_1 - 2x_2 \iff \min: -z = -3x_1 + 2x_2$$

*Si le programme linéaire a une contrainte $x \geq a$, on effectue le **changement de variable** : $x = \tilde{x} + a$, ce qui transforme la contrainte $x \geq a$ en $\tilde{x} \geq 0$

Variable d'écart: variable de décision introduite dans un problème d'optimisation afin de **transformer une contrainte d'inégalité en une contrainte d'égalité**. Par exemple:

$$g(x) \leq 0 \iff g(x) + z^2 = 0 \iff \begin{cases} g(x) + y = 0 \\ y \geq 0. \end{cases}$$

Exemple / Exercice

$$\underset{x, y}{\text{Max}} \quad (-x^2 + \sin y)$$

sous contraintes :

$$6x - y^2 \geq 1$$

$$x^2 + y^2 = 3$$

$$x \geq 2$$

$$y \in \mathbb{R}$$



Exigences :

- Minimisation
- Variables non négatives
- Contraintes d'inégalité inférieure

$$\underset{\tilde{x}, y', y''}{\text{Min}} \quad (\tilde{x} + 2)^2 - \sin(y' - y'')$$

sous contraintes

$$-6(\tilde{x} + 2) + (y' - y'')^2 + 1 \leq 0$$

$$(\tilde{x} + 2)^2 + (y' - y'')^2 - 3 \leq 0$$

$$-(\tilde{x} + 2)^2 - (y' - y'')^2 + 3 \leq 0$$

$$\tilde{x} \geq 0$$

$$y' \geq 0$$

$$y'' \geq 0$$

(Attention: pas linéaire ici)

Application

Une entreprise fabrique **2** produits **X** et **Y**. Pour être fabriqué, chaque produit fini nécessite **3** produits intermédiaires **A**, **B** et **C**. Pour fabriquer un produit **X**, on a besoin de **2** produits **A**, de **2** produits **B** et de **1** produit **C**. Pour fabriquer un produit **Y**, on a besoin de **3** produits **A**, de **1** produit **B** et de **3** produits **C**. En outre, l'entreprise dispose d'une quantité limitée de **A**, **B** et **C**. Elle a **180** produits **A**, **120** produits **B** et **150** produits **C**. Sachant que le prix de revient de **X** est **3** euros et que celui de **Y** est de **4** euros, combien de produits **X** et **Y** faut-il fabriquer pour maximiser le profit ?

On modélise ce problème par un programme linéaire. Soit **x** et **y** les quantités de produits **X** et **Y** fabriqués.

La quantité totale de produits **A** utilisée est $2x + 3y$.

Cette quantité ne doit pas dépasser **180**, d'où la première contrainte: $2x + 3y \leq 180$

De même, pour les produits **B** et **C**, on obtient:

$$2x + y \leq 120$$

$$x + 3y \leq 150$$

Bien entendu, les quantités **x** et **y** sont positives.

$$x, y \geq 0$$

Enfin, on tente de maximiser le profit qui est le total des bénéfices sur la vente des produits **X** plus celui des produits **Y**.

$$\max: 3x + 4y$$

Le programme linéaire est donc le suivant :

$$\text{max: } z = 3x + 4y$$

Sous

$$2x + 3y \leq 180 \quad (\text{A})$$

$$2x + y \leq 120 \quad (\text{B})$$

$$x + 3y \leq 150 \quad (\text{C})$$

$$x, y \geq 0$$

Résolution

On dispose d'un outil pour modéliser un problème donné.

Comment résoudre ce modèle ?

- Plusieurs algorithmes existent, dont **l'algorithme du simplexe** (voir plus loin)
- Pour un problème avec **deux variables**, on peut utiliser une **Résolution graphique**, qui aide à comprendre la structure du problème

Le programme linéaire obtenu:

$$\text{max: } z = 3x + 4y$$

Sous

$$2x + 3y \leq 180 \quad (\text{A})$$

$$2x + y \leq 120 \quad (\text{B})$$

$$x + 3y \leq 150 \quad (\text{C})$$

$$x, y \geq 0$$

Résolution graphique (cas 2D)

On peut représenter le problème dans un espace à deux dimensions.

Contrainte 4:

tracer les droites $x = 0 \Rightarrow$ axe des ordonnées
et $y = 0 \Rightarrow$ axe des abscisses

Contrainte 3:

tracer droite $x + 3y = 150 \quad (\text{C})$

2 points: $x=0 \Rightarrow y=150/3=50$
 $y=0 \Rightarrow x=150$

Contrainte 1:

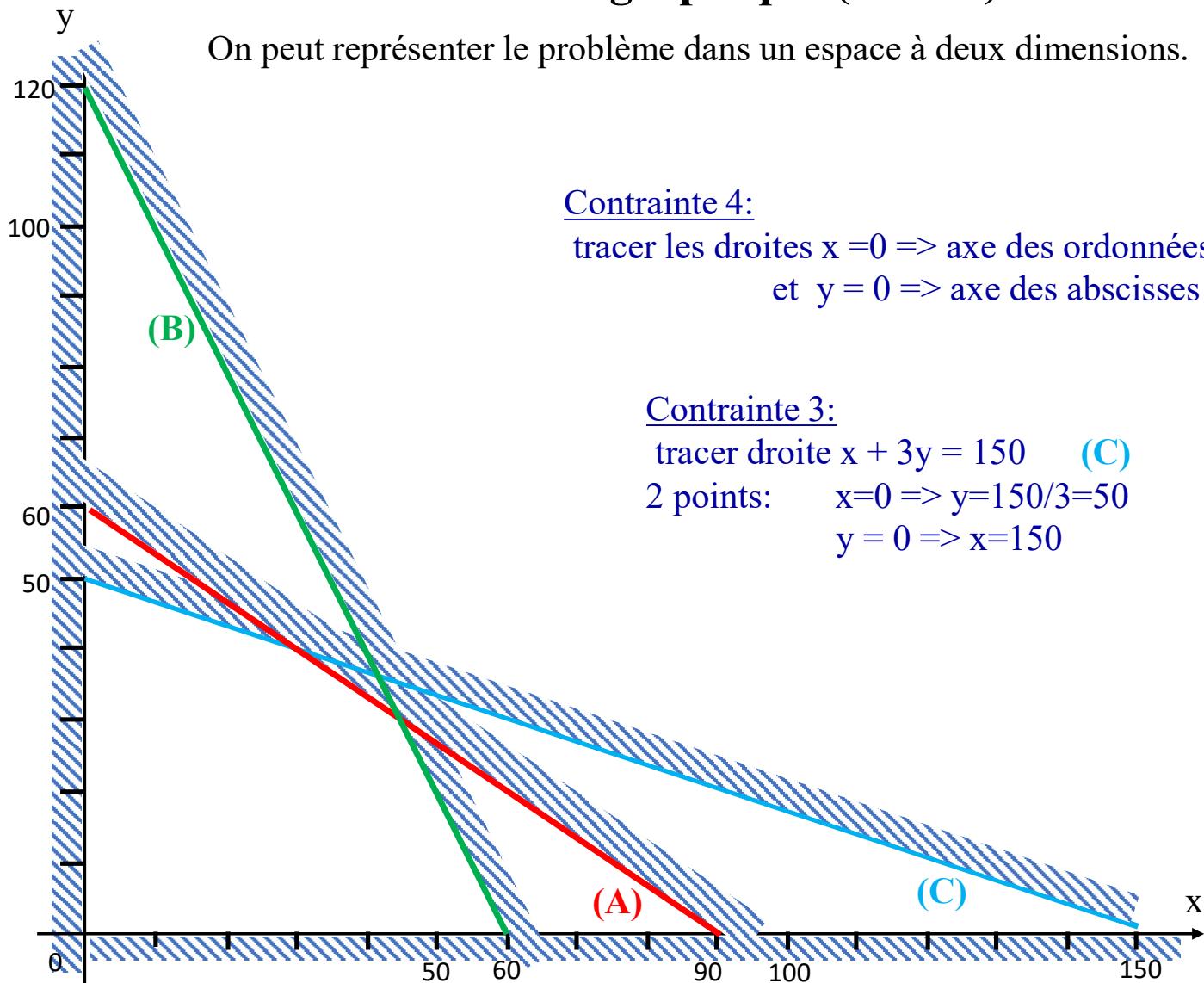
tracer droite $2x + 3y = 180 \quad (\text{A})$

2 points: $x=0 \Rightarrow y=180/3=60$
 $y=0 \Rightarrow x=180/2=90$

Contrainte 2:

tracer droite $2x + y = 120 \quad (\text{B})$

2 points: $x=0 \Rightarrow y=120$
 $y=0 \Rightarrow x=120/2=60$



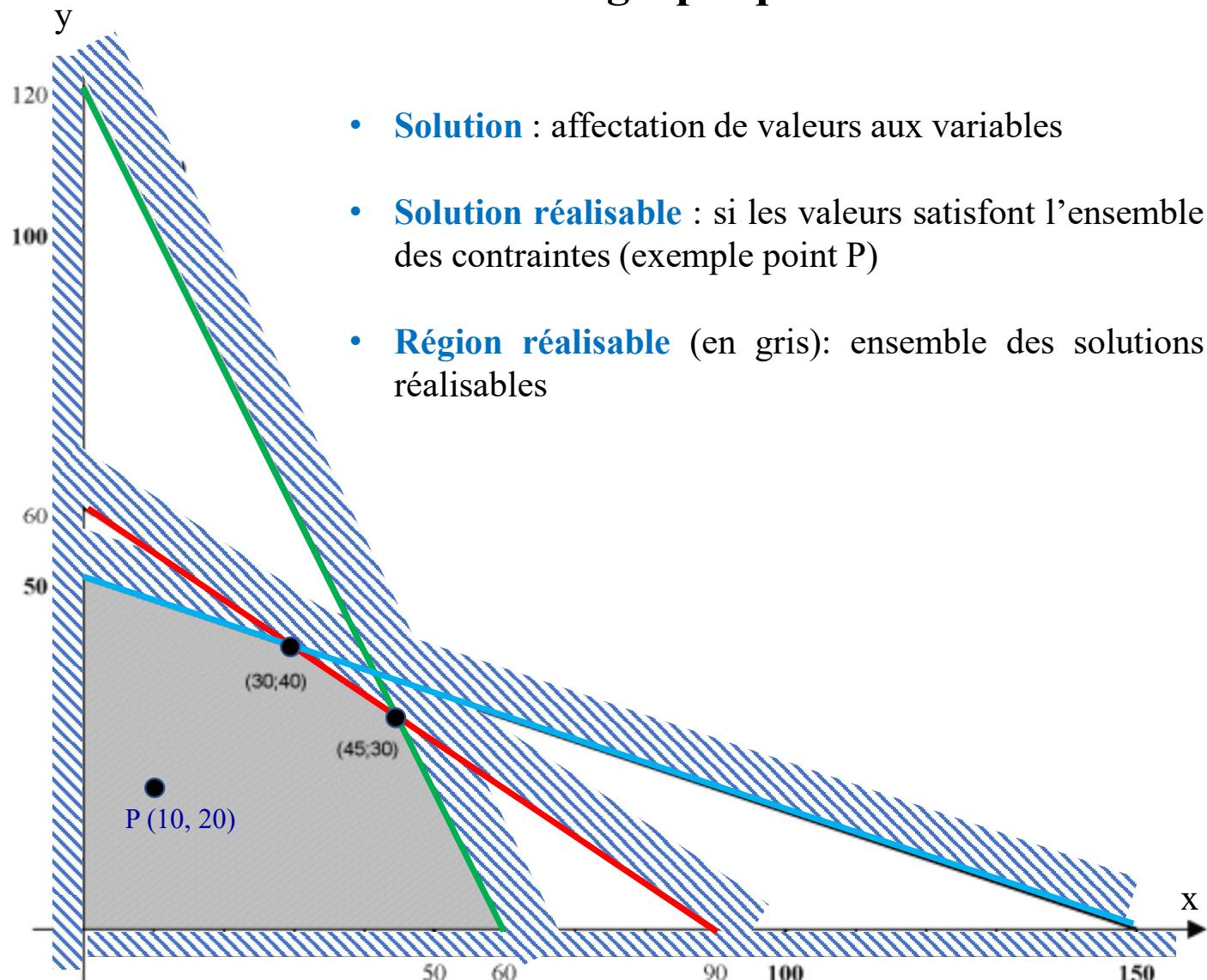
Le programme linéaire obtenu:

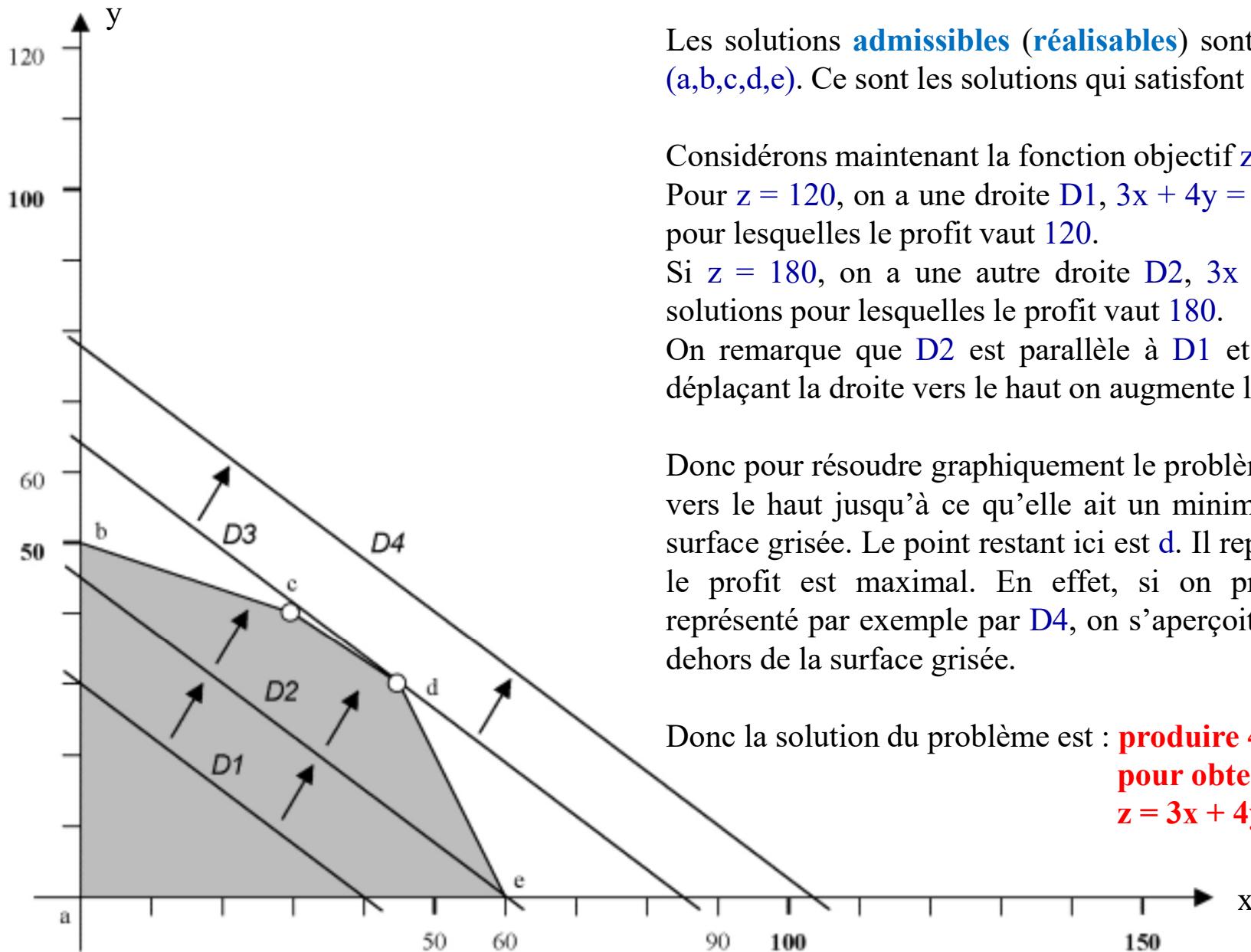
$$\text{max: } z = 3x + 4y$$

Sous
 $2x + 3y \leq 180$ (A)
 $2x + y \leq 120$ (B)
 $x + 3y \leq 150$ (C)
 $x, y \geq 0$

Résolution graphique

- **Solution** : affectation de valeurs aux variables
- **Solution réalisable** : si les valeurs satisfont l'ensemble des contraintes (exemple point P)
- **Région réalisable** (en gris): ensemble des solutions réalisables





Les solutions **admissibles (réalisables)** sont représentées par la zone grisée (a,b,c,d,e). Ce sont les solutions qui satisfont les contraintes.

Considérons maintenant la fonction objectif $z = 3x + 4y$.

Pour $z = 120$, on a une droite D_1 , $3x + 4y = 120$, qui représente les solutions pour lesquelles le profit vaut 120.

Si $z = 180$, on a une autre droite D_2 , $3x + 4y = 180$, qui représente les solutions pour lesquelles le profit vaut 180.

On remarque que D_2 est parallèle à D_1 et on s'aperçoit facilement qu'en déplaçant la droite vers le haut on augmente le profit z .

Donc pour résoudre graphiquement le problème, on va faire "glisser" la droite vers le haut jusqu'à ce qu'elle ait un minimum de points communs avec la surface grisée. Le point restant ici est **d**. Il représente la solution pour laquelle le profit est maximal. En effet, si on prend un profit plus important, représenté par exemple par D_4 , on s'aperçoit que toutes les solutions sont en dehors de la surface grisée.

Donc la solution du problème est : **produire 45 produits X et 30 produits Y pour obtenir le profit maximum de $z = 3x + 4y = 255$ euros.**

Existence d'une solution optimale

Quatre cas possibles :

Cas 1 : exemple :

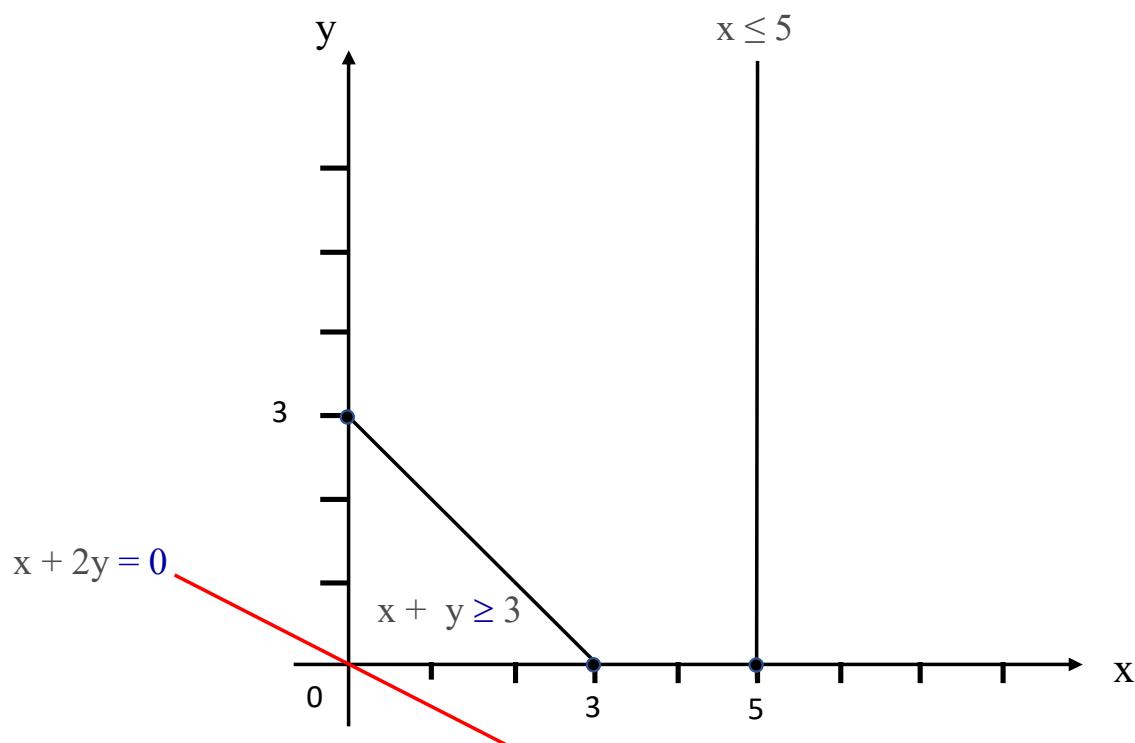
$$\min x + 2y$$

sous contraintes :

$$x \leq 5$$

$$x + y \geq 3$$

$$x, y \geq 0$$



Existence d'une solution optimale

Quatre cas possibles :

Cas 1 : exemple :

$$\min x + 2y$$

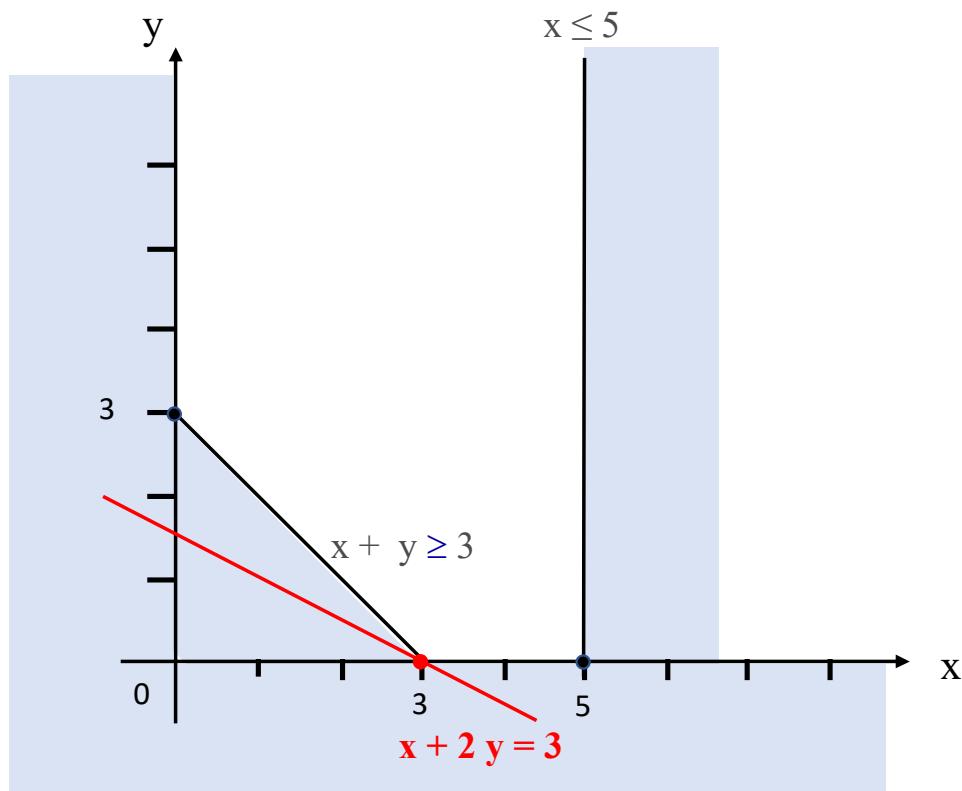
sous contraintes :

$$x \leq 5$$

$$x + y \geq 3$$

$$x, y \geq 0$$

Une solution optimale unique



Existence d'une solution optimale

Quatre cas possibles :

Cas 2 : exemple :

$$\max(x + 2y)$$

sous contraintes :

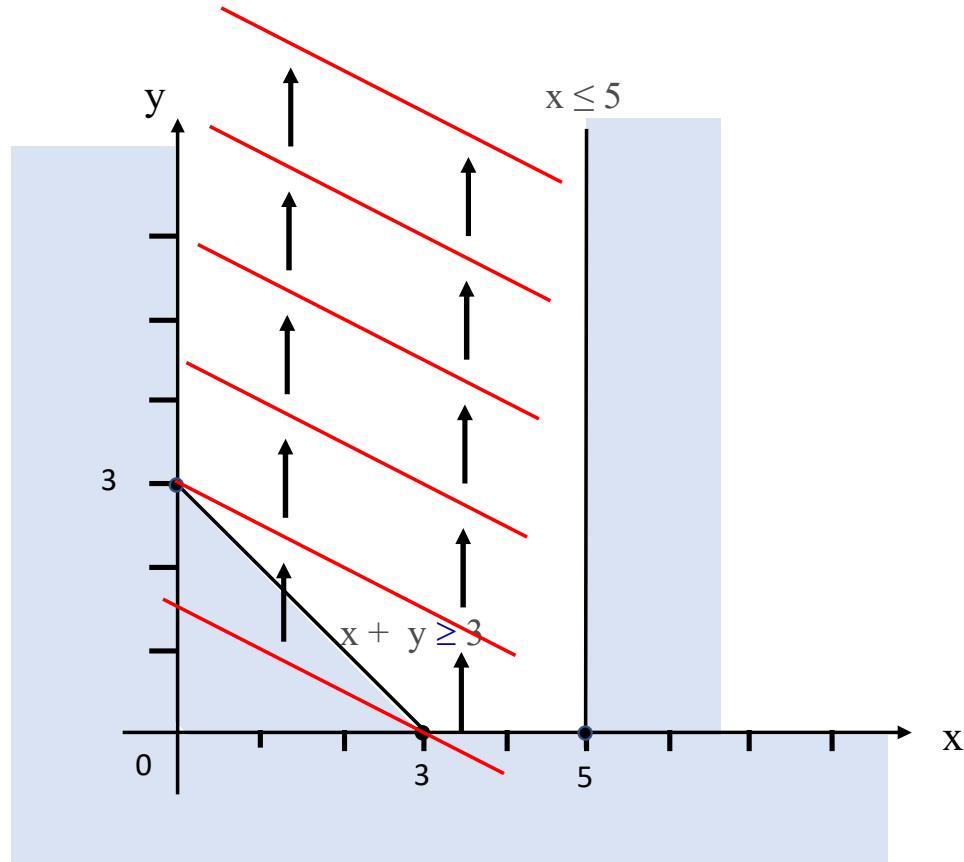
$$x \leq 5$$

$$x + y \geq 3$$

$$x, y \geq 0$$

Solution non bornée

(pas de valeur finie pour l'objectif
 $x+2y$)



Existence d'une solution optimale

Quatre cas possibles :

Cas 3 : exemple :

$$\max x + 2y$$

sous contraintes :

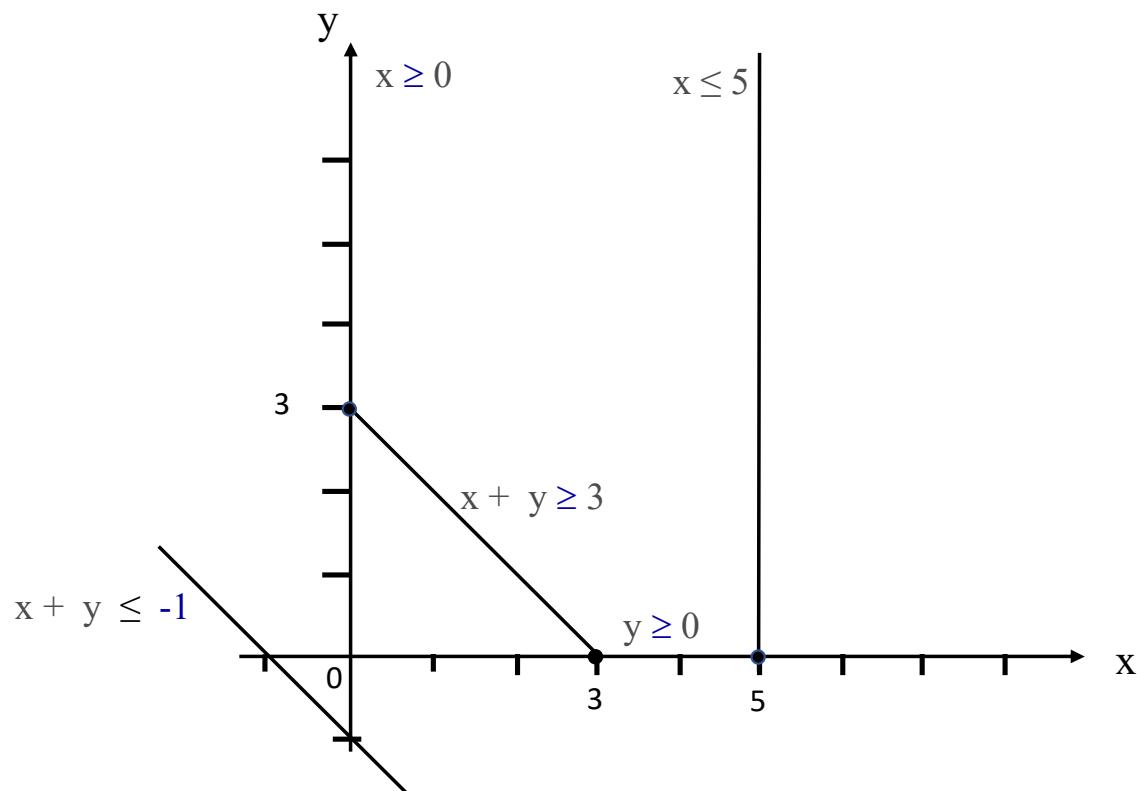
$$x \leq 5$$

$$x + y \geq 3$$

$$x + y \leq -1$$

$$x, y \geq 0$$

Pas de solution



Existence d'une solution optimale

Quatre cas possibles :

Cas 4 : exemple :

max x

sous contraintes :

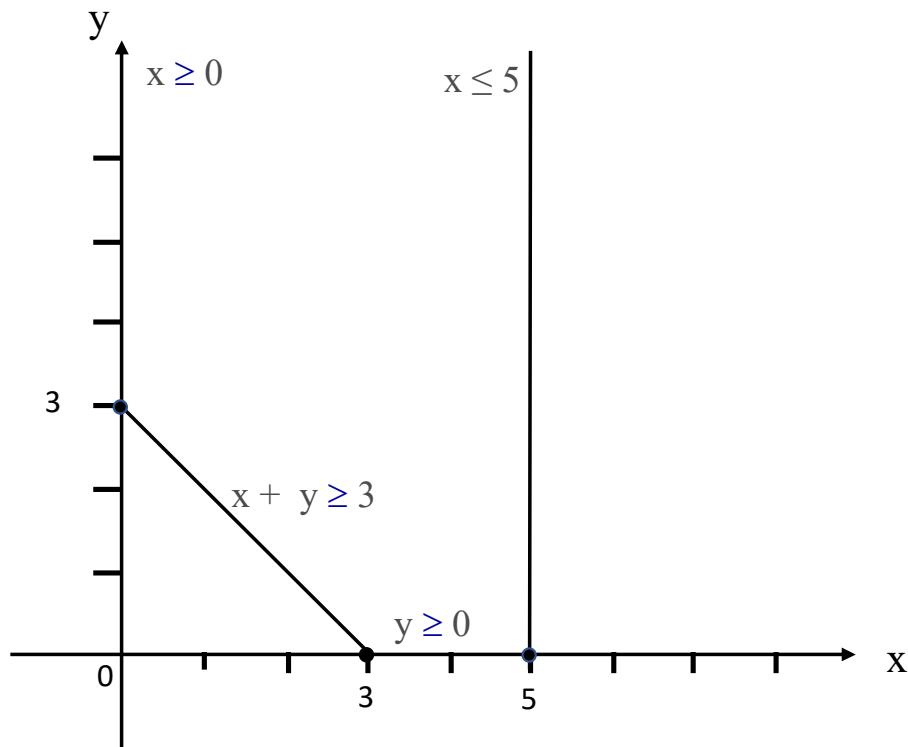
$$x \leq 5$$

$$x + y \geq 3$$

$$x, y \geq 0$$

Infinité de solutions

(valeur max de $x = 5$,
avec une infinité de valeurs pour y)



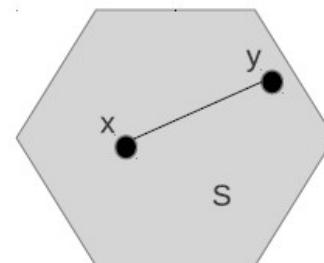
S'il en existe, il y a toujours une solution optimale sur un sommet (**point extrême**) de la région réalisable.

Donc pour trouver l'optimum, il « suffit » d'examiner les points extrêmes de la région réalisable.

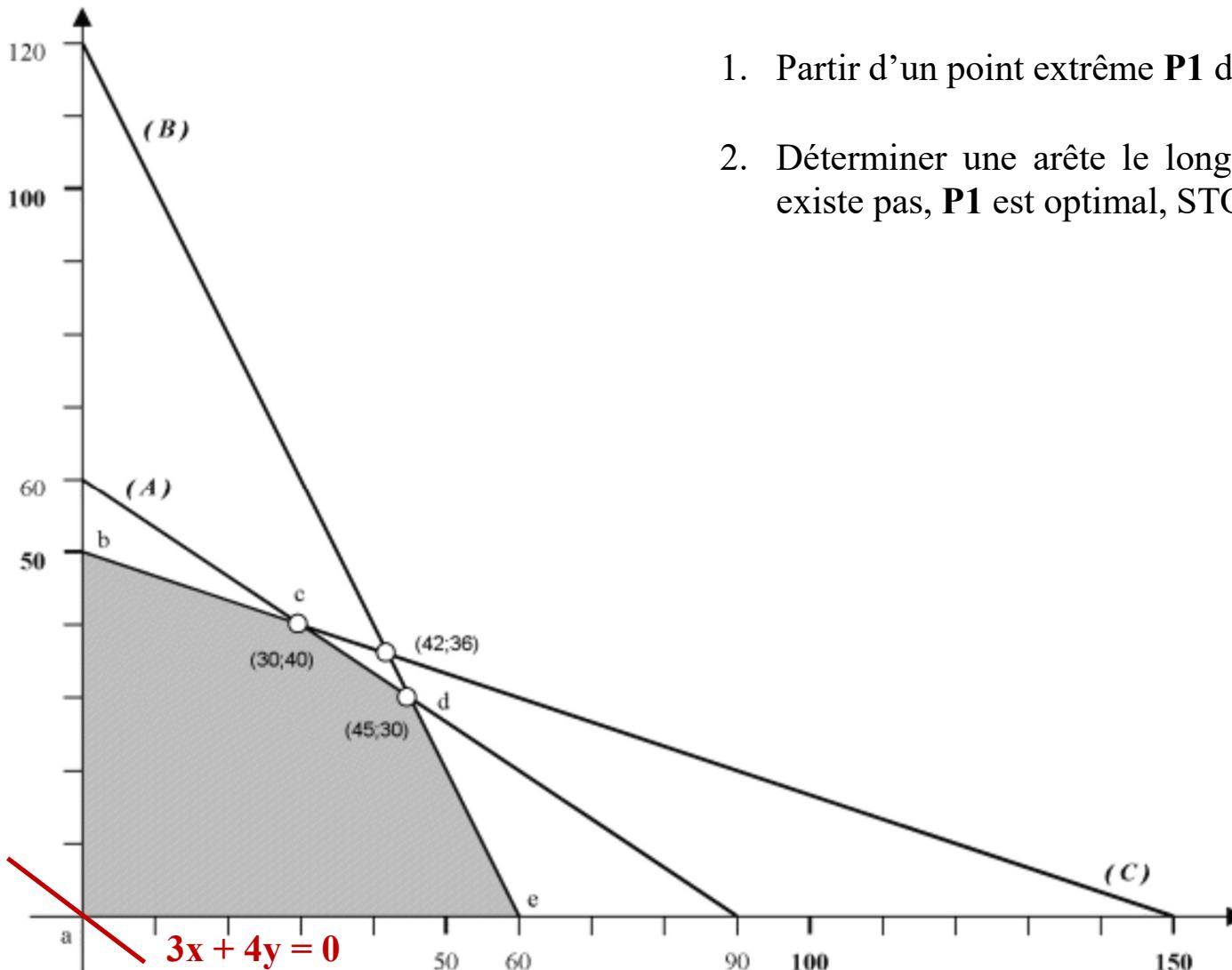
Un **polyèdre convexe*** est l'ensemble des solutions d'un système fini d'inégalités linéaires.

L'ensemble des solutions réalisables d'un programme linéaire est donc un polyèdre convexe.

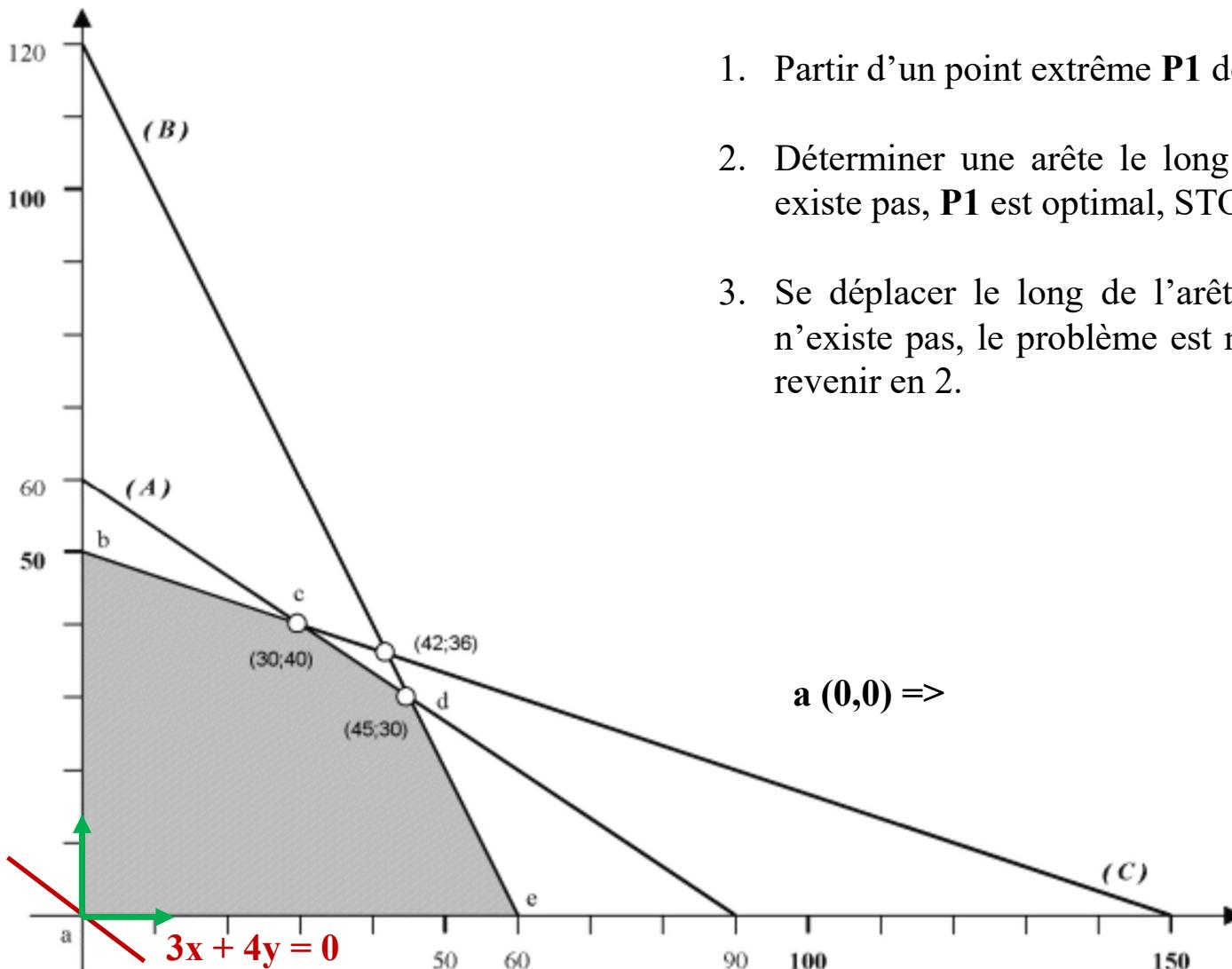
**Un polyèdre est convexe si : $\forall \mathbf{x}, \mathbf{y} \in S, \forall \lambda \in [0, 1], \lambda \mathbf{x} + (1 - \lambda) \mathbf{y} \in S$.*



Algorithme géométrique

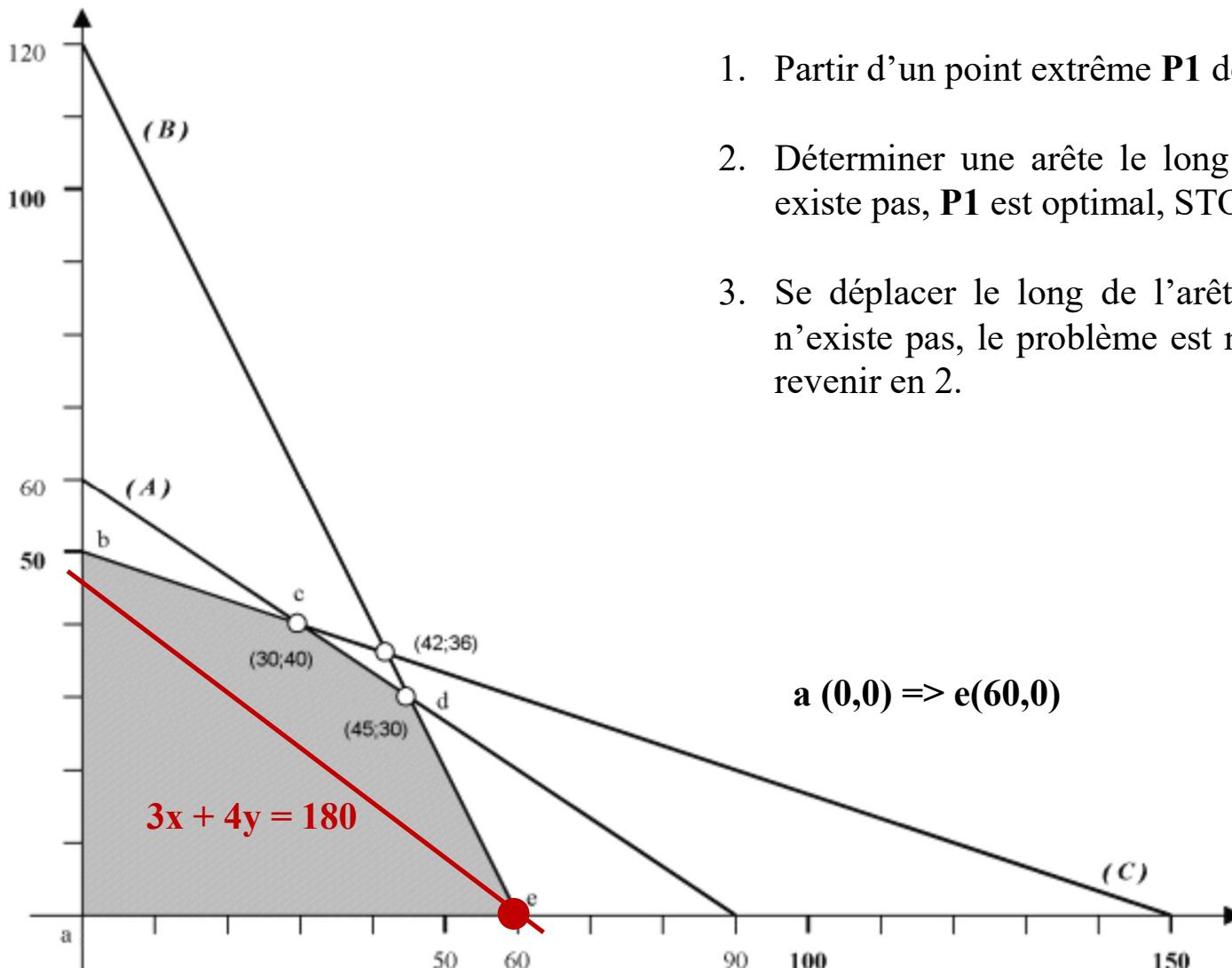


Algorithme géométrique



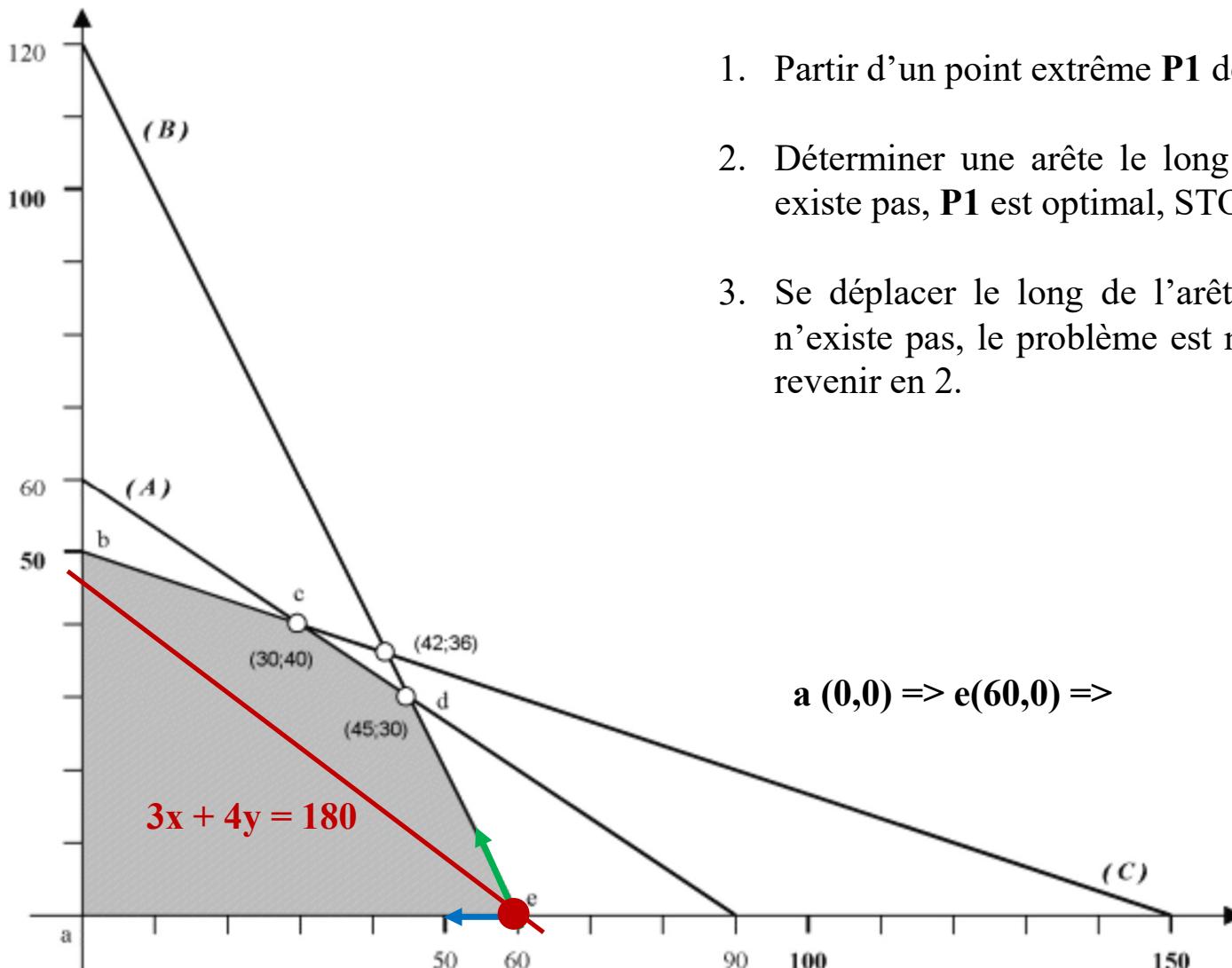
1. Partir d'un point extrême **P1** de la région réalisable (ici point a **(0,0)**).
2. Déterminer une arête le long de laquelle l'objectif augmente. S'il n'en existe pas, **P1** est optimal, STOP.
3. Se déplacer le long de l'arête jusqu'au point extrême **P2** suivant. S'il n'existe pas, le problème est non borné, STOP. Sinon, poser **P1** \leftarrow **P2** et revenir en 2.

Algorithme géométrique



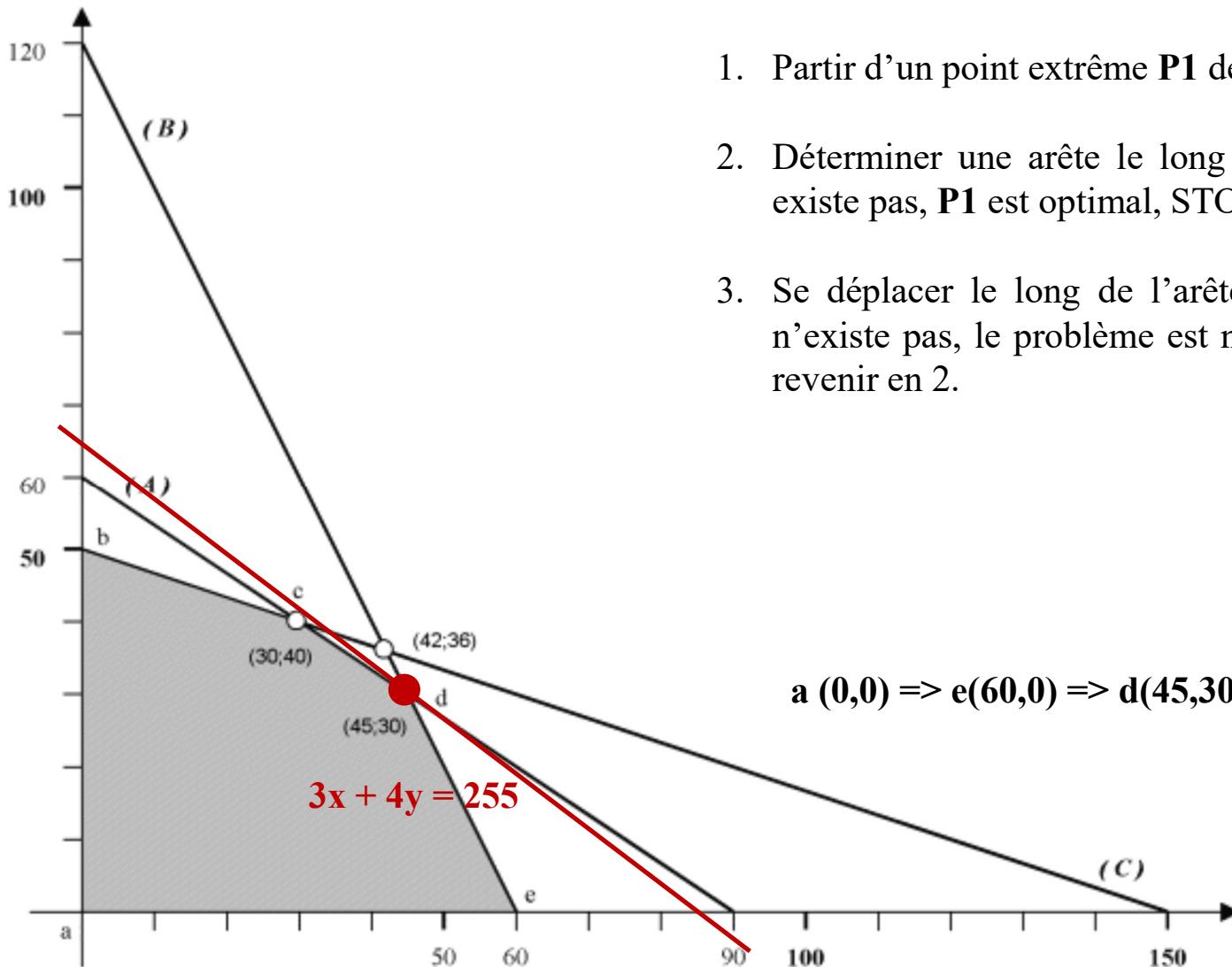
1. Partir d'un point extrême **P1** de la région réalisable (ici point a **(0,0)**).
2. Déterminer une arête le long de laquelle l'objectif augmente. S'il n'en existe pas, **P1** est optimal, STOP.
3. Se déplacer le long de l'arête jusqu'au point extrême **P2** suivant. S'il n'existe pas, le problème est non borné, STOP. Sinon, poser **P1**← **P2** et revenir en 2.

Algorithme géométrique



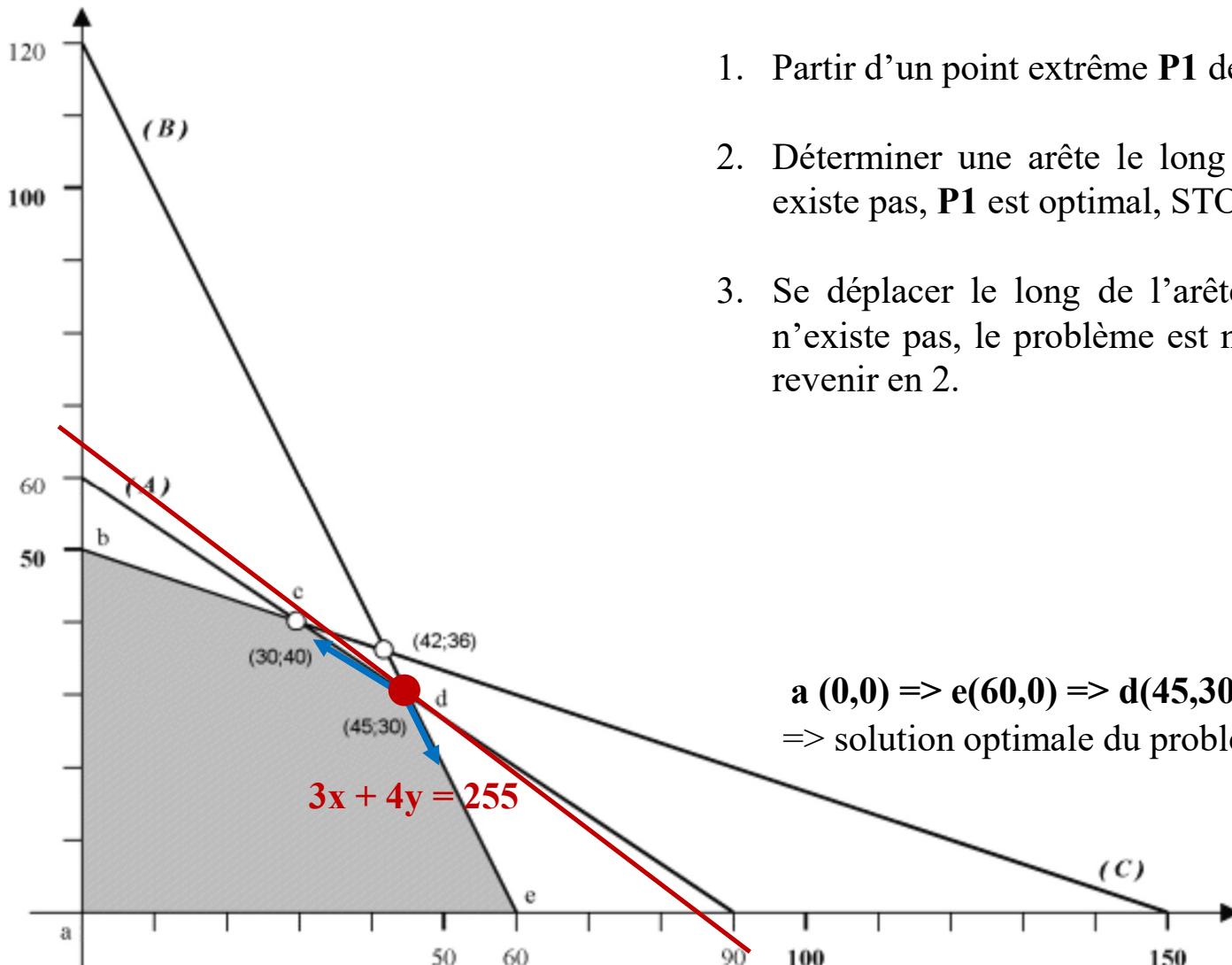
1. Partir d'un point extrême **P1** de la région réalisable (ici point a **(0,0)**).
2. Déterminer une arête le long de laquelle l'objectif augmente. S'il n'en existe pas, **P1** est optimal, STOP.
3. Se déplacer le long de l'arête jusqu'au point extrême **P2** suivant. S'il n'existe pas, le problème est non borné, STOP. Sinon, poser **P1**← **P2** et revenir en 2.

Algorithme géométrique



1. Partir d'un point extrême **P1** de la région réalisable (ici point a **(0,0)**).
2. Déterminer une arête le long de laquelle l'objectif augmente. S'il n'en existe pas, **P1** est optimal, STOP.
3. Se déplacer le long de l'arête jusqu'au point extrême **P2** suivant. S'il n'existe pas, le problème est non borné, STOP. Sinon, poser **P1**← **P2** et revenir en 2.

Algorithme géométrique



1. Partir d'un point extrême **P1** de la région réalisable (ici point **a** **(0,0)**).
2. Déterminer une arête le long de laquelle l'objectif augmente. S'il n'en existe pas, **P1** est optimal, STOP.
3. Se déplacer le long de l'arête jusqu'au point extrême **P2** suivant. S'il n'existe pas, le problème est non borné, STOP. Sinon, poser **P1** \leftarrow **P2** et revenir en 2.

a (0,0) \Rightarrow e(60,0) \Rightarrow d(45,30) \Rightarrow STOP au point (45, 30)
 \Rightarrow solution optimale du problème : produire 45 produits X et 30 produits Y avec le profit maximum de z = 3x + 4y = 255 euros

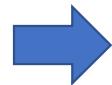
Forme standard

A partir de tout programme linéaire sous forme **canonique** (normale), on peut construire un programme linéaire sous **forme standard**.

$$\max z = \sum_{i=1}^n c_i x_i$$

$$\sum_{i=1}^n a_{ij} x_i \leq b_j (j = 1, \dots, m)$$

$$x_i \geq 0 (i = 1, \dots, n)$$



$$\max z = \sum_{i=1}^n c_i x_i$$

$$\sum_{i=1}^n a_{ij} x_i + s_j = b_j (j = 1, \dots, m)$$

$$x_i \geq 0 (i = 1, \dots, n)$$

$$s_j \geq 0 (j = 1, \dots, m)$$

Les variables supplémentaires s_j sont les **variables d'écart**.

Chaque variable d'écart est associée à une contrainte.

La méthode du simplexe

déroulement sur un exemple

Globalement, la méthode du **Simplexe** va se déplacer le long de la forme **(a, b, c, d, e)**, de sommet en sommet jusqu'à trouver le meilleur point.

Considérons le problème de production précédent. Dans un premier temps, les contraintes sont ramenées à des égalités en introduisant de nouvelles variables (les **variables d'écart**) :

forme canonique

$$\text{max: } z = 3x + 4y$$

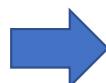
Sous

$$2x + 3y \leq 180 \quad (\text{A})$$

$$2x + y \leq 120 \quad (\text{B})$$

$$x + 3y \leq 150 \quad (\text{C})$$

$$x, y \geq 0$$



forme standard

$$\text{max: } z = 3x + 4y$$

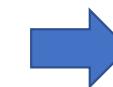
sous:

$$2x + 3y + u = 180 \quad (\text{A})$$

$$2x + y + v = 120 \quad (\text{B})$$

$$x + 3y + w = 150 \quad (\text{C})$$

$$x, y, u, v, w \geq 0$$



$$\text{max: } z = 3x + 4y$$

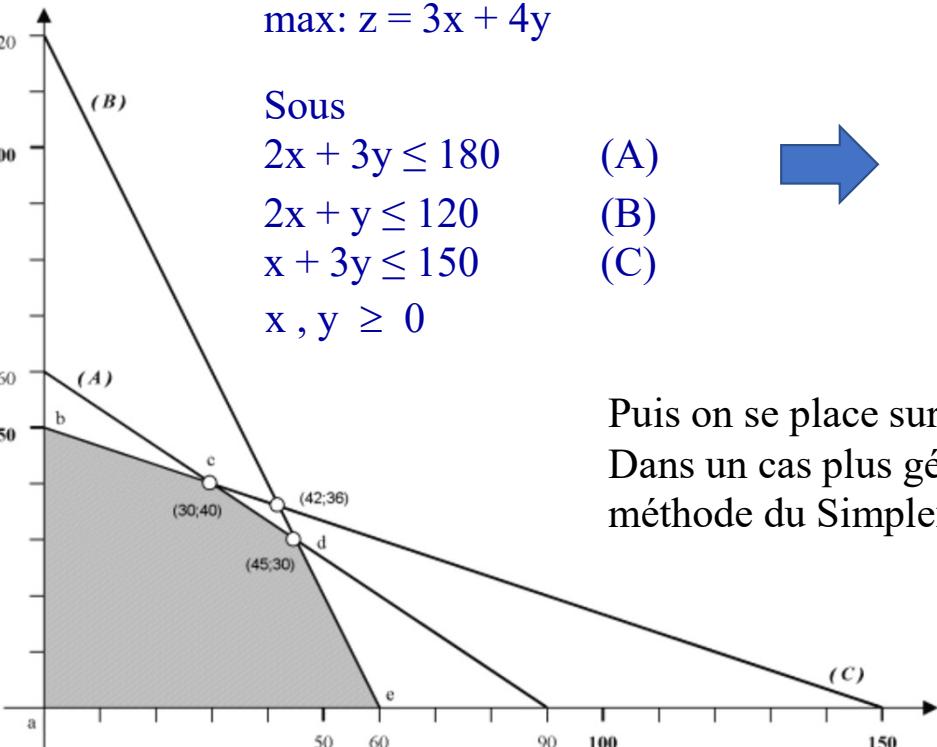
sous:

$$2x + 3y = 180 - u \quad (\text{A})$$

$$2x + y = 120 - v \quad (\text{B})$$

$$x + 3y = 150 - w \quad (\text{C})$$

$$x, y, u, v, w \geq 0$$



Puis on se place sur le point $a = (0, 0)$ qui est une solution admissible du problème.

Dans un cas plus général, il faut noter que trouver une solution admissible pour démarrer la méthode du Simplexe n'est pas forcément évident. On a alors:

$$x = 0, y = 0$$

$$u = 180$$

$$v = 120$$

$$w = 150$$

On a annulé les variables de décision pour ne garder que les variables d'écart

Remarque: Forme standard et points extrêmes

max: $z = 3x + 4y$

sous:

$$2x + 3y + u = 180 \quad (\text{A})$$

$$2x + y + v = 120 \quad (\text{B})$$

$$x + 3y + w = 150 \quad (\text{C})$$

$$x, y, u, v, w \geq 0$$

Points extrêmes : intersection d'hyperplans (contraintes)

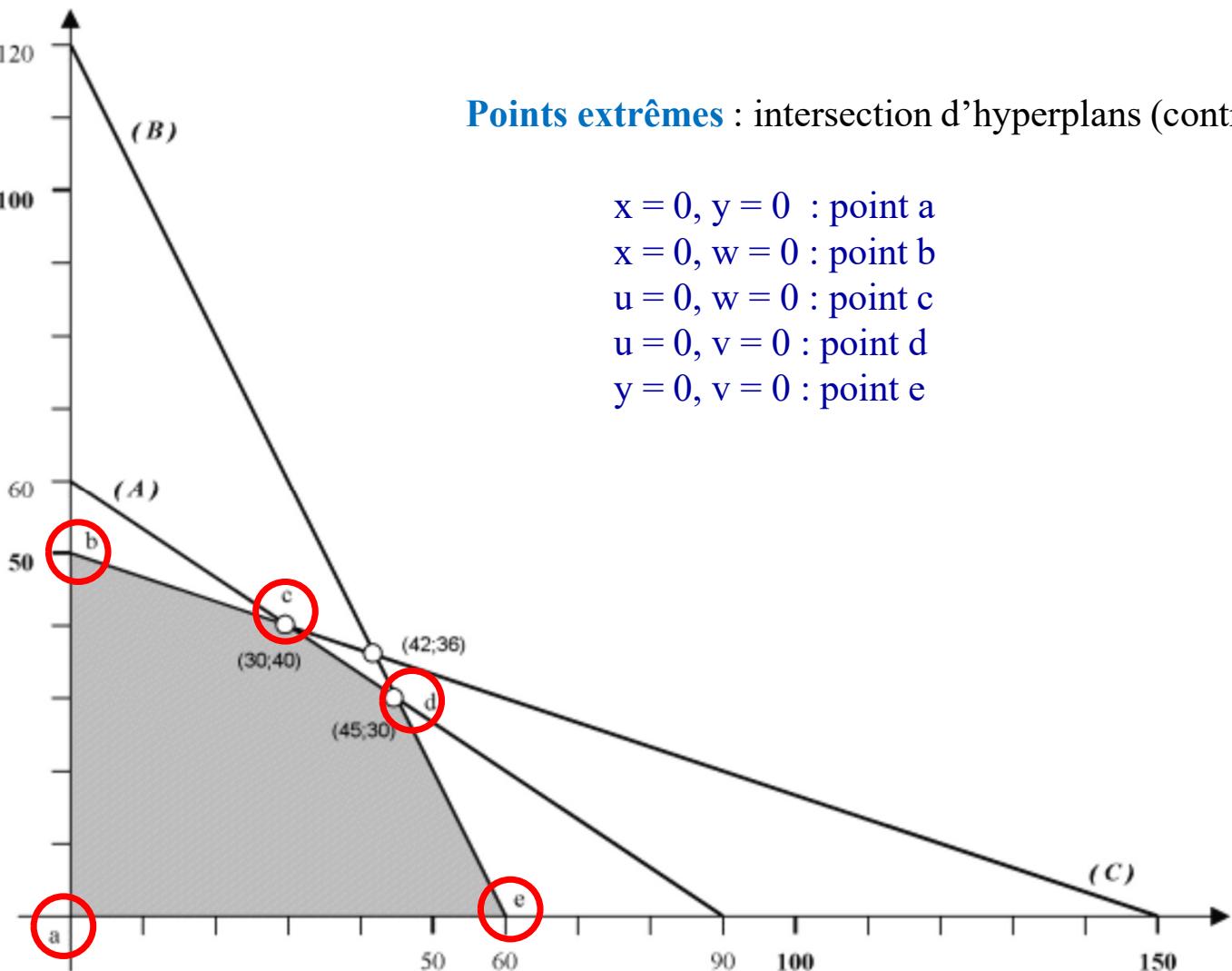
$$x = 0, y = 0 : \text{point a}$$

$$x = 0, w = 0 : \text{point b}$$

$$u = 0, w = 0 : \text{point c}$$

$$u = 0, v = 0 : \text{point d}$$

$$y = 0, v = 0 : \text{point e}$$



Dans la fonction objectif $z = 3x + 4y$, on voit que la moindre augmentation de x ou de y augmente le profit z .
Donc on va augmenter l'une des deux variables au maximum.

Par exemple, prenons y et utilisons les contraintes pour exprimer y en fonction de toutes les autres variables.

$$\text{max: } z = 3x + 4y$$

sous:

$$2x + 3y = 180 - u \quad (\text{A})$$

$$2x + y = 120 - v \quad (\text{B})$$

$$x + 3y = 150 - w \quad (\text{C})$$

$$x, y, u, v, w \geq 0$$



$$y = 60 - \frac{1}{3}u - \frac{2}{3}x$$

$$y = 120 - v - 2x$$

$$y = 50 - \frac{1}{3}w - \frac{1}{3}x$$



$$y \leq 60 \quad (\text{A})$$

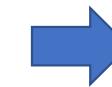
$$y \leq 120 \quad (\text{B})$$

$$y \leq 50 \quad (\text{C})$$

Comme toutes les variables sont positives, on peut en déduire, à partir de chaque contrainte, une borne maximale pour y (cf. ci-dessus). Comme y doit satisfaire les 3 contraintes, on prend ici la plus grande valeur de y possible, qui est 50.
A l'aide de la contrainte (C), on remarque que x doit rester à 0 et que w est forcée à 0.

Maintenant, les variables nulles sont x et w . On exprime les autres variables et le profit z uniquement avec x et w pour obtenir leur valeur.

$$\begin{aligned} y &= 50 - \frac{1}{3}x - \frac{1}{3}w &= 50 & (\text{C}) \\ u &= 180 - 2x - 3y = 30 - x + w &= 30 & (\text{A}) \\ v &= 120 - 2x - y = 70 - \frac{5}{3}x + \frac{1}{3}w &= 70 & (\text{B}) \\ x & &= 0 & \\ w & &= 0 & \\ z &= 3x + 4y = 200 + \frac{5}{3}x - \frac{4}{3}w &= 200 & \end{aligned}$$



On se trouve au point
 $b = (x, y) = (0, 50)$
avec un profit $z = 200$.

Ensuite, avec $b = (0, 50)$ ($x=w=0$), on reprend le même raisonnement qu'avec le point $a = (0, 0)$.

Le profit a été exprimé en fonction de x et w , par $z = 200 + 5/3x - 4/3w$.

En augmentant x , on augmente le profit. On regarde de combien on peut augmenter x .

(si on augmente w on diminue le profit z , donc w reste à 0)

$$\max: z = 3x + 4y$$

Sous :

$$2x + 3y = 180 - u \quad (A)$$

$$2x + y = 120 - v \quad (B)$$

$$x + 3y = 150 - w \quad (C)$$

$$x, y, u, v, w \geq 0$$

$$\begin{aligned} y &= 50 - 1/3x - 1/3w \\ u &= 180 - 2x - 3y = 30 - x + w \\ v &= 120 - 2x - y = 70 - 5/3x + 1/3w \end{aligned}$$



$$\begin{aligned} x &= 150 - 3y - w \\ x &= 30 - u + w \\ x &= 3/5 (70 - v + 1/3w) = 42 - 3/5v + 1/5w \end{aligned}$$



$$\begin{aligned} x &\leq 150 & (C) \\ x &\leq 30 & (A) - (C) \\ x &\leq 42 & (B) - (C) \end{aligned}$$

OU

$$\begin{cases} x - 2y = -30 - v + w, \text{ avec} \\ y = 50 - 1/3x - 1/3w \end{cases}$$

Ici, on prendra $x = 30$, la valeur maximale qu'il peut atteindre.

Les variables qui sont nulles sont u et w d'après la première contrainte ci-dessus. On calcule les nouvelles valeurs des autres variables.

$$x = 30 - u + w$$

$$= 30 \quad (A)$$

$$v = 70 - 5/3 (30 - u + w) + 1/3w = 20 + 5/3u - 4/3w$$

$$= 20 \quad (B)$$

$$y = 50 - 1/3 (30 - u + w) - 1/3w = 40 + 1/3u - 2/3w$$

$$= 40 \quad (C)$$

$$u$$

$$= 0$$

$$w$$

$$= 0$$

$$z = 200 + 5/3 (30 - u + w) - 4/3w = 250 - 5/3u + 1/3w$$

$$= 250$$



On se trouve donc au point
 $c = (x, y) = (30, 40)$
 avec un profit $z = 250$.

$$\max: z = 3x + 4y$$

Sous :

$$2x + 3y = 180 - u \quad (A)$$

$$2x + y = 120 - v \quad (B)$$

$$x + 3y = 150 - w \quad (C)$$

$$x, y, u, v, w \geq 0$$

On recommence encore une fois le même raisonnement.

Le profit est exprimé par $z = 250 - 5/3u + 1/3w$.

En augmentant w , on augmente le profit. On regarde de combien on peut augmenter w .

$$x = 30 - u + w$$

$$v = 20 + 5/3u - 4/3w$$

$$y = 40 + 1/3u - 2/3w$$

$$w = -30 + x + u$$

$$w = 3/4 (20 + 5/3u - v) = 15 + 5/4u - 3/4v$$

$$w = 3/2 (40 - y + 1/3u) = 60 - 3/2y + 1/2u$$

$$w \geq -30$$

$$w \leq 15$$

$$w \leq 60$$

(A)

(B)

(C)

Ici, on prendra $w = 15$, la valeur maximale qu'il peut atteindre.

Les variables qui sont nulles sont u et v d'après la contrainte (B). On calcule les nouvelles valeurs des autres variables.

$$w = 15 + 5/4u - 3/4v = 15 \quad (B)$$

$$x = 30 - u + (15 + 5/4u - 3/4v) = 45 + 1/4u - 3/4v = 45 \quad (A)$$

$$y = 40 + 1/3u - 2/3 (15 + 5/4u - 3/4v) = 30 - 1/2u + 1/2v = 30 \quad (C)$$

$$u = 0$$

$$v = 0$$

$$z = 250 - 5/3u + 1/3 (15 + 5/4u - 3/4v) = 255 - 5/4u - 1/4v = 255$$



On se trouve donc au point $d = (x, y) = (45, 30)$ avec un profit $z = 255$.

On reprend encore une fois le même raisonnement. Le profit est exprimé par $z = 255 - 5/4u - 1/4v$.

Ni u , ni v ne permettent d'augmenter le profit z . Cela signifie que l'on a obtenu le profit maximal.

La solution recherchée est alors représentée par le point courant d .

On aboutit bien à la même conclusion que par la représentation graphique.

Il faut produire 45 produits X et 30 produits Y pour obtenir un profit maximum de 255 euros.

La méthode du simplexe

généralisation / description formelle

Reprendons la forme standard générale :

$$\max z = \sum_{i=1}^n c_i x_i$$

$$\sum_{i=1}^n a_{ij} x_i + s_j = b_j (j = 1, \dots, m)$$

$$x_i \geq 0 (i = 1, \dots, n)$$

$$s_j \geq 0 (j = 1, \dots, m)$$

⇒ Modèle avec

- n variables de décision
- et m variables d'écart (1 par contrainte)

donc

- n+m variables et m contraintes

Sommets = **bases**

- Si on annule n variables, on obtient un système de m équations à m inconnues
- Si la matrice associée est de rang m (base), le système admet une solution unique
- **Une base = une solution**
- Si on a une base réalisable, on a un point extrême
- Pour résoudre le problème obtenu et calculer les valeurs des variables pour ces points : méthode du pivot de Gauss
- *Attention: si la solution n'a pas de valeurs positives, elle n'est pas valide*

La méthode du simplexe

généralisation / description formelle

Forme standard générale

$$\begin{aligned} \max z &= \sum_{i=1}^n c_i x_i \\ \sum_{i=1}^n a_{ij} x_i + s_j &= b_j \quad (j = 1, \dots, m) \\ x_i &\geq 0 \quad (i = 1, \dots, n) \\ s_j &\geq 0 \quad (j = 1, \dots, m) \end{aligned}$$



Forme matricielle

$$\begin{aligned} \text{Max } Z &= C \cdot x \\ A' \cdot x &= b \\ x &\geq 0 \end{aligned}$$

A = matrice m x (n+m) (matrice des contraintes)

C = (c₁, c₂, ..., c_n, 0, 0, ..., 0) vecteur ligne des coûts

b = (b₁, b₂, ..., b_m)^T vecteur des seconds membres

x = (x₁, x₂, ..., x_n, s₁, s₂, ..., s_m)^T vecteur des variables
= (x₁, x₂, ..., x_n, x_{n+1}, x_{n+2}, ..., x_{n+m})^T

bases et solutions de base

On suppose qu'il n'y a pas de contrainte redondante, c'est-à-dire pas de combinaison linéaire d'autres contraintes (le rang de A vaut m).

base de A, ou **matrice de base** : toute sous-matrice carrée inversible B, $m \times m$, de A.

N est la matrice m x n des colonnes hors base, ou ***matrice hors base***.

Le vecteur x_B a pour composantes les m *variables de base* (associées aux colonnes de B).

x_N a pour composantes les n variables hors base.

Le système de contraintes et la fonction objectif peuvent alors s'écrire de manière équivalente :

$$\begin{aligned} A.x = b &\Leftrightarrow B \cdot x_B + N \cdot x_N = b && \Leftrightarrow x_B = B^{-1} \cdot b - B^{-1} \cdot N \cdot x_N \\ Z = c \cdot x = c_B \cdot x_B + c_N \cdot x_N &= c_B \cdot B^{-1} \cdot b + (c_N - c_B \cdot B^{-1} \cdot N) \cdot x_N \end{aligned}$$

⇒ expression des variables de base x_B en fonction des variables hors base x_N , pour une base choisie B.

On a une **solution évidente en forçant x_N à 0** : on a alors $x_B = B^{-1} \cdot b$.

Cette solution du programme linéaire (PL) est la **solution de base (SB) associée à la base B**.

Elle peut violer des contraintes de positivité : on appelle solution de base réalisable (SBR) une solution réalisable dont toutes les variables sont positives ou nulles.

Algorithme du simplexe forme tableau

Forme canonique



$$\max: z = x_1 + 2x_2$$

$$\begin{array}{ll} \text{Sous} & x_1 + x_2 \leq 6 \\ & x_2 \leq 3 \\ & x_1, x_2 \geq 0 \end{array}$$

Forme standard: ajout 2 variables d'écart x_3, x_4

$$\begin{array}{rclcl} \max z = & x_1 + & 2x_2 & & \\ & x_1 + & x_2 + & x_3 & = 6 \\ & & x_2 & + & x_4 = 3 \\ x_1 & x_2 & x_3 & x_4 & \geq 0 \end{array}$$



Tableau initial pour ce PL, composé de 4 sous-tableaux

Base	T	1	2	3	4	b'	-z_B
3	1	1	1	1	0	6	
4	2	0	1	0	1	3	
	Δ	1	2	0	0	0	

Tableau initial:

*tableau central T: chargé avec matrice A.

*tableau-colonne Base: donne les **indices** des variables de base actuelles, les **variables d'écart** x_3 et x_4 .

*tableau-colonne b': contient les 2nd membres.

*ligne Δ : fonction objectif sous la forme $0 \cdot x_B + \Delta_N \cdot x_N = -z_B$ (*les variables de base ont un profit marginal nul*).

Note: la case $-z_B$ contient la valeur de la fonction objectif, mais multipliée par -1 . Actuellement cette valeur est nulle.

Tableau initial

Base	T	1	2	3	4	b'	$b'_i / T_{ie} > 0$
3	1	1	1	1	0	6	6
4	2	0	1	0	1	3	$3 \rightarrow S$
	Δ	1	2	0	0	0	

↑

e

$-z_B$

Itération 1 : elle construit le tableau de la prochaine base.

- **variable hors base x_e entrant en base** est x_2 (elle a le coût réduit > 0 le plus grand sur la ligne Δ).
Repérer sa colonne (**colonne pivot**) avec un **e** \Rightarrow *ici colonne 2 de T*
- **variable sortant de la base** : C'est celle qui minimise les b'_i / T_{ie} avec $T_{ie} > 0$ ou, comme on suppose des seconds membres positifs dans la forme standard, les $b'_i / T_{ie} > 0$. Ceci se produit à la ligne $s = 2$, dite ligne pivot. Base[s] (*ici Base[2]*) donne l'indice de la variable de base correspondante (x_4).
- entourer le pivot T_{se} (*ici T_{22}*). Pour exprimer x_2 en fonction des variables hors base, on devrait dans le cas général ranger dans le tableau suivant la ligne du pivot divisée par le pivot, pour avoir un 1 à la place du pivot. C'est inutile ici car le pivot vaut déjà 1, on se contente de recopier la ligne du pivot.

Base	T	1	2	3	4	b'	$b'_i / T_{ie} > 0$
3	1	1	1	1	0	6	6
4	2	0	1	0	1	3	$3 \rightarrow s$
	Δ	1	2	0	0	0	

↑

e

- Faire apparaître des 0 dans les T_{ie} (*ici* T_{i2}) des lignes $i \neq s$: multiplier la ligne du pivot par T_{ie} , puis la soustraire à la ligne i. les lignes obtenues sont rangées dans le tableau suivant. Ceci élimine x_e (x_2) des équations autres que la ligne s. *Ici, le traitement revient à remplacer ligne 1 par : ligne 1 - ligne 2.*
- Appliquer la même opération à la ligne Δ , y compris pour $-z_B$: la ligne du pivot est multipliée par Δ_e (=2) et soustraite à la ligne Δ pour obtenir la nouvelle ligne Δ du tableau suivant. *Ici, remplacer ligne Δ par: ligne Δ - 2*ligne 2.*
- Mettre à jour l'indice de la variable de base correspondant désormais à la ligne s dans Base : *2 au lieu de 4.*
- On obtient le tableau suivant :

Base	T	1	2	3	4	b'	
3	1	1	0	1	-1	3	<i>ligne 1 - 1*ligne 2</i>
2	2	0	1	0	1	3	
	Δ	1	0	0	-2	-6	<i>ligne Δ - 2*ligne 2</i>

$-z_B$

Base	T	1	2	3	4	b'	$b_i' / T_{ie} > 0$
$\begin{bmatrix} 3 \\ 2 \end{bmatrix}$	1	1	0	1	-1	3	$3 \rightarrow s$
	2	0	1	0	1	3	-
	Δ	1	0	0	-2	-6	$-z_B$

↑
e

Itération 2 : L'algorithme n'est pas terminé car on peut encore augmenter le profit en augmentant x_1 .

- Nouvelle colonne pivot = *colonne 1*. Et $T_{se} = T_{11}$.
- variable entrant en base : x_1 , variable sortant de la base : Base[1] donc x_3

Base	T	1	2	3	4	b'	$b_i' / T_{ie} > 0$
$\begin{bmatrix} 1 \\ 2 \end{bmatrix}$	1	1	0	1	-1	3	$3 \rightarrow s$
	2	0	1	0	1	3	-
	Δ	0	0	-1	-1	-9	$-z_B$

↑
e

*ligne 2 - 0*ligne 1*
*ligne Δ - 1*ligne 1*

L'algorithme se termine lorsque tous les profits marginaux sont négatifs ou nuls.

Base	T	1	2	3	4	b'	$b_i' / T_{ie} > 0$
1	1	1	0	1	-1	3	$3 \rightarrow \text{S}$
2	2	0	1	0	1	3	-
Δ		0	0	-1	-1	-9	

↑

e

$-Z_B$

Solution (avec variables hors base x_3 et x_4 à 0)

* ligne 1 de T : $x_1 = 3$

* ligne 2 de T : $x_2 = 3$

* $Z = 9$

Si à une itération, il n'y a pas de $b_i' / T_{ie} > 0$, alors l'optimum est non borné.

Exemple 2 Reprenons l'exemple initial suivant

forme canonique

$$\begin{aligned} \text{max: } z &= 3x + 4y \\ \text{Sous: } \\ 2x + 3y &\leq 180 \quad (\text{A}) \\ 2x + y &\leq 120 \quad (\text{B}) \\ x + 3y &\leq 150 \quad (\text{C}) \\ x, y &\geq 0 \end{aligned}$$

forme standard

$$\begin{aligned} \text{max: } z &= 3x + 4y \\ \text{sous: } \\ 2x + 3y + u &= 180 \quad (\text{A}) \\ 2x + y + v &= 120 \quad (\text{B}) \\ x + 3y + w &= 150 \quad (\text{C}) \\ x, y, u, v, w &\geq 0 \end{aligned}$$

changement
de notation

$$\begin{aligned} \text{max: } z &= 3x_1 + 4x_2 \\ \text{sous: } \\ 2x_1 + 3x_2 + x_3 &= 180 \quad (\text{A}) \\ 2x_1 + x_2 + x_4 &= 120 \quad (\text{B}) \\ x_1 + 3x_2 + x_5 &= 150 \quad (\text{C}) \\ x_1, x_2, x_3, x_4, x_5 &\geq 0 \end{aligned}$$

Simplexe: itération 1

Base	T	1	2	3	4	5	b'	$b'_i / T_{ie} > 0$
3	1	2	3	1	0	0	180	60
4	2	2	1	0	1	0	120	120
5	3	1	3	0	0	1	150	50 $\rightarrow s = 3$
	Δ	3	4	0	0	0	0	

↑

$e=2$

Solution admissible (0,0)

$-z_B$

Simplexe: itération 1

Base	T	1	2	3	4	5	b'	$b_i' / T_{ie} > 0$
3	1	2	3	1	0	0	180	60
4	2	2	1	0	1	0	120	120
5	3	1	3	0	0	1	150	50 $\rightarrow s = 3$
Δ		3	4	0	0	0	0	
		↑					$-z_B$	
		$e=2$						

=> x_2 remplace x_5 (Base[3]) dans la base

Attention : pivot $T_{32} \neq 1$

=> pour exprimer x_2 en fonction des variables hors base,
remplacer ligne du pivot par ligne pivot divisée par le pivot, pour avoir un 1 à la place du pivot

Base	T	1	2	3	4	5	b'	$b_i' / T_{ie} > 0$
3	1	2	3	1	0	0	180	60
4	2	2	1	0	1	0	120	120
2	3	1/3	1	0	0	1/3	50	50 $\rightarrow s = 3$
Δ		3	4	0	0	0	0	Ligne 3 /3
		↑					$-z_B$	
		$e=2$						

Base	T	1	2	3	4	5	b'	$b'_i / T_{ie} > 0$
3	1	2	3	1	0	0	180	60
4	2	2	1	0	1	0	120	120
2	3	1/3	1	0	0	1/3	50	50 $\rightarrow s = 3$
	Δ	3	4	0	0	0	0	
		↑					-Z _B	
		e=2						

mise à jour des lignes non pivot



Base	T	1	2	3	4	5
3	1	1	0	1	0	-1
4	2	5/3	0	0	1	-1/3
2	3	1/3	1	0	0	1/3
	Δ	5/3	0	0	0	-4/3

Ligne 1 – 3 ligne 3

Ligne 2 – ligne 3

Ligne Δ – 4 ligne 3

-Z_B

Simplexe: itération 2

Base	T	1	2	3	4	5	b'	$b'_i / T_{ie} > 0$
3	1	1	0	1	0	-1	30	30 $\rightarrow s = 1$
4	2	5/3	0	0	1	-1/3	70	42
2	3	1/3	1	0	0	1/3	50	150
	Δ	5/3	0	0	0	-4/3	-200	
		↑					$-Z_B$	
		e=1						

=> x_1 remplace x_3 (Base[1]) dans la base

mise à jour des lignes non pivot

Base	T	1	2	3	4	5	b'	$b'_i / T_{ie} > 0$
1	1	1	0	1	0	-1	30	
4	2	0	0	-5/3	1	4/3	20	Ligne 2 - 5/3 ligne 1
2	3	0	1	-1/3	0	2/3	40	Ligne 3 - 1/3 ligne 1
	Δ	0	0	-5/3	0	1/3	-250	Ligne Δ - 5/3 ligne 1
							$-Z_B$	

=> Remarque 1: les variables de décision sont entrées en base mais la méthode ne s'arrête pas car on peut encore augmenter z

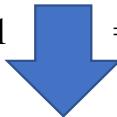
Simplexe: itération 3

Base	T	1	2	3	4	5	b'	$b_i' / T_{ie} > 0$
1	1	1	0	1	0	-1	30	- (-30)
4	2	0	0	-5/3	1	4/3	20	15 → s = 2
2	3	0	1	-1/3	0	2/3	40	60
	Δ	0	0	-5/3	0	1/3	-250	

↑ $-z_B$

e=5

x_5 remplace x_4 ET pivot $T_{25} \neq 1$



=> Ligne 2 / (4/3)

Base	T	1	2	3	4	5	b'	$b_i' / T_{ie} > 0$
1	1	1	0	1	0	-1	30	- (-30)
5	2	0	0	-5/4	3/4	1	15	15 → s = 2
2	3	0	1	-1/3	0	2/3	40	60
	Δ	0	0	-5/3	0	1/3	-250	

↑ $-z_B$

=> Remarque 2: une variable de base peut sortir puis entrer à nouveau en base: ici, x_5 (entrée 2 fois en base)

Base	T	1	2	3	4	5	b'
1	1	1	0	1	0	-1	30
5	2	0	0	-5/4	3/4	1	15
2	3	0	1	-1/3	0	2/3	40
	Δ	0	0	-5/3	0	1/3	-250

$$b'_i / T_{ie} > 0$$

$$- (-30)$$

$$15 \rightarrow s = 2$$

$$60$$



$$-z_B$$

$$e=5$$

Base	T	1	2	3	4	5	b'
1	1	1	0	-1/4	3/4	0	45
5	2	0	0	-5/4	3/4	1	15
2	3	0	1	1/2	-1/2	0	30
	Δ	0	0	-5/4	-1/4	0	-255

$$-z_B$$

mise à jour des lignes non pivot

$$\text{Ligne 1} - (-1) \text{ ligne 2}$$

$$\text{Ligne 3} - 2/3 \text{ ligne 2}$$

$$\text{Ligne } \Delta - 1/3 \text{ ligne 2}$$

On ne peut plus augmenter z, donc le simplexe s'arrête.

=> On retrouve la solution optimale z=255 (avec variables hors base x_3 et x_4 à 0, $x_1=45$, $x_2=30$ et $x_5=15$)

Fin exemple 2

Absence de base initiale évidente

Soit le programme linéaire suivant :

$$\begin{aligned} \text{Max } z &= x_1 + 2x_2 \\ x_1 + x_2 &\leq 6 \\ x_2 &\leq 3 \\ x_1 + x_2 &\geq 1 \\ x_1, x_2 &\geq 0 \end{aligned}$$



Mise sous forme standard :

$$\begin{aligned} \text{Max } z &= x_1 + 2x_2 \\ x_1 + x_2 + x_3 &= 6 \\ x_2 + x_4 &= 3 \\ x_1 + x_2 - x_5 &= 1 \\ x_1, x_2, x_3, x_4, x_5 &\geq 0 \end{aligned}$$

On n'a plus la matrice identité habituelle (à cause de $-x_5$), qui nous fournissait une base initiale évidente.

Démarrage de l'algorithme du simplexe:

* trouver par tâtonnement une sous-matrice B inversible => effort important,

OU

* 2 méthodes possibles : la **méthode des deux phases** et la **méthode du grand M**.

idée : faire apparaître une matrice identité en ajoutant aux contraintes qui en ont besoin (*ici la troisième*) une **variable artificielle** (VA) avec un coefficient 1.

Les variables x_1 à x_5 sont dites **légitimes** : elles sont nécessaires à l'obtention de la forme standard.

Les variables d'écart x_3 et x_4 nous donnent deux colonnes de matrice identité, les contraintes correspondantes (première et deuxième) n'ont donc pas besoin de VA.

On ajoute x_6 à la troisième contrainte pour compléter la base. Une telle variable est réellement artificielle et n'a aucun sens économique : à l'optimum, elle doit être nulle (hors base) sinon la contrainte n'est pas vérifiée avec égalité ! Les deux méthodes des deux phases et du grand M ont précisément pour but de s'en débarrasser.

$$\begin{aligned} \text{Max } z = & x_1 + 2x_2 \\ & x_1 + x_2 + x_3 &= 6 \\ & x_2 + x_4 &= 3 \\ & x_1 + x_2 - x_5 + x_6 &= 1 \\ & x_1, x_2, x_3, x_4, x_5, x_6 \geq 0 \end{aligned}$$

Méthode des deux phases

Phase 1: on ignore la vraie fonction objectif et on cherche à minimiser la somme des p variables artificielles qu'on a dû introduire. On résout donc le PL auxiliaire suivant, où x_A désigne le vecteur des p VA, et e un vecteur avec p composantes à 1.

$$\text{Min } z = e \cdot x_A = \sum_{j=1}^p x_A(j)$$

$$A \cdot x + x_A = b$$

$$x, x_A \geq 0$$

Remarque: ici on est en minimisation, le choix de la variable entrante dans l'algorithme du simplexe est modifié : c'est celle de plus petit coût marginal négatif.

Si tout se passe bien, on parvient à annuler la somme des VA et à trouver une base sans VA. On obtient donc une solution réalisable pour le problème de départ.

Phase 2 :

- éliminer du tableau les colonnes des VA, devenues inutiles (on a ainsi une forme simpliciale du problème initial),
- remplacer la ligne Δ par la vraie fonction objectif,
- et continuer le simplexe sur le tableau obtenu.

C'est en phase 2 qu'on peut détecter si le PL d'origine n'a pas d'optimum borné.

En réalité, d'autres cas spéciaux peuvent se produire en fin de phase 1 :

- **si $x_A \neq 0$** (fonction objectif non nulle), le PL d'origine n'a **pas de solution réalisable**.
- Si $x_A = 0$ avec des VA dans la base, on peut montrer que les contraintes associées à ces variables sont redondantes. On peut les éliminer et attaquer la phase 2.

Application à l'exemple :

$$\begin{aligned}
 \text{Max } z = & x_1 + 2x_2 \\
 & x_1 + x_2 + x_3 = 6 \\
 & x_2 + x_4 = 3 \\
 & x_1 + x_2 -x_5 + x_6 = 1 \\
 & x_1, x_2, x_3, x_4, , x_5, x_6 \geq 0
 \end{aligned}$$

La base est formée des colonnes 3, 4 et 6 (matrice identité). Mais dans l'algorithme du simplexe, les variables de base doivent avoir des coûts réduits nuls, ce qui n'est pas le cas pour x_6 .

Transformer Δ en une ligne correcte Δ' , en lui soustrayant la ligne 3 du tableau \Rightarrow cela fait apparaître le véritable coût de la solution de base actuelle : 1, c'est-à-dire qu'elle contient une VA.

Etant en **minimisation**, il faut faire entrer en base la variable de coût réduit négatif minimal. On a le choix entre $e=1$ et $e=2$. Par convention, nous prenons le plus petit indice :

Base	T	1	2	3	4	5	6	b'	$b'_i / T_{ie} > 0$
3	1	1	1	1	0	0	0	6	6
4	2	0	1	0	1	0	0	3	-
6	3	1	1	0	0	-1	1	1	$1 \rightarrow s=3$
	Δ	0	0	0	0	0	1	0	
	Δ'	-1	-1	0	0	1	0	-1	<i>ligne Δ - ligne 3</i>
		↑						$-z_B$	
			$e=1$						

Base	T	1	2	3	4	5	6	b'	$b_i' / T_{ie} > 0$
3 4 1	1	1	1	0	0	0	6	6	
	2	0	1	0	1	0	3	-	
	3	1	1	0	0	-1	1	1 → s=3	
Δ Δ'		0	0	0	0	0	1	0	
		-1	-1	0	0	1	0	-1	
		↑							$-z_B$
		e=1							

Tableau après pivotage sur T_{31} :

Base	T	1	2	3	4	5	6	b'
3	1	0	0	1	0	1	-1	5
4	2	0	1	0	1	0	0	3
1	3	1	1	0	0	-1	1	1
	Δ'	0	0	0	0	0	1	

*ligne 1 - $T_{1e} * \text{ligne } 3 = L1 - 1 * L3$*

*ligne 2 - $T_{2e} * \text{ligne } 3 = L2 - 0 * L3$*

*ligne Δ - (-1) * ligne 3*

La phase 1 se termine ici avec succès, puisque tous les coûts marginaux sont positifs ou nuls (on est en minimisation).

=> On a éliminé x_6 de la base. On peut supprimer la colonne 6 pour la suite.

Base	T	1	2	3	4	5	b'	$b_i' / T_{ie} > 0$
3	1	0	0	1	0	1	5	$5 \rightarrow s=1$
4	2	0	1	0	1	0	3	-
1	3	1	1	0	0	-1	1	- $(-1 < 0 \text{ donc non considéré})$
Δ		1	2	0	0	0	0	
Δ'		0	1	0	0	1	-1	<i>ligne Δ - ligne 3</i>
						↑	$-z_B$	
						$e=5$		

On peut faire la **phase 2** car la VA x_6 a été éjectée de la base.

Pour démarrer la phase 2,

- les colonnes des VA (ici une seule) sont supprimées du tableau en fin de phase 1, ce qui donne le tableau ci-dessus.
- La fonction objectif d'origine $x_1 + 2x_2$ est réintroduite. Elle doit être exprimée en fonction des variables hors base, comme les autres lignes. Mais ici la variable de base x_1 a un coût réduit non nul. Donc, on soustrait la ligne 3 à la ligne Δ , ce qui donne une ligne correcte Δ' avec un coefficient nul pour x_1 , x_3 , et x_4 .

Après pivotage sur T_{15} , choix pour e entre 2 et 5 : ici, on « triche » en faisant entrer en base x_5 au lieu de x_2 , pour gagner une itération.

Base	T	1	2	3	4	5	b'	$b_i' / T_{ie} > 0$
3 4 1	1	0	0	1	0	1	5	$5 \rightarrow s=1$
	2	0	1	0	1	0	3	-
	3	1	1	0	0	-1	1	-
Δ Δ'	1	2	0	0	0	0	0	
	0	1	0	0	1		-1	
								$-z_B$
						e=5		

Après pivotage sur T_{15}

Base	T	1	2	3	4	5	b'	$b_i' / T_{ie} > 0$
5	1	0	0	1	0	1	5	-
4	2	0	1	0	1	0	3	$3 \rightarrow s=2$
1	3	1	1	1	0	0	6	$ligne\ 3 - (-1)*ligne\ 1$
	Δ	0	1	-1	0	0	-6	$ligne\ \Delta' - 1*ligne\ 1$
			↑				$-z_B$	
			e=2					

Base	T	1	2	3	4	5	b'	$b'_i / T_{ie} > 0$
	1	0	0	1	0	1	5	-
	2	0	1	0	1	0	3	$3 \rightarrow s=2$
	3	1	1	1	0	0	6	6
Δ		0	1	-1	0	0	-6	
			↑					$-z_B$
			e=2					

Tableau après pivotage sur T_{22} :



Base	T	1	2	3	4	5	b'	
	1	0	0	1	0	1	5	$ligne 1 - 0 * ligne 2$
	2	0	1	0	1	0	3	
	3	1	0	1	-1	0	3	$ligne 3 - 1 * ligne 2$
Δ		0	0	-1	-1	0	-9	$ligne \Delta - 1 * ligne 2$
								$-z_B$

STOP car plus de coût marginal strictement positif **Solution optimale de coût 9 avec $(x_1, x_2) = (3, 3)$.**

Méthode du grand M

=> méthode en une phase.

Elle introduit les VA directement dans la fonction objectif du PL d'origine, pénalisées par un coût $-M$, M étant un grand nombre positif.

Avec les mêmes notations que pour les deux phases, on résout en fait le PL :

$$\text{Max } z = c \cdot x - M \cdot e \cdot x_A = \sum_{j=1}^n c_j \cdot x_j - M \sum_{j=1}^p x_A(j)$$

$$A \cdot x + x_A = b$$

$$x, x_A \geq 0$$

Si tout se passe bien, le simplexe se débarrasse des VA dès les premières itérations et ne les fait plus rentrer en base à cause de leur coût énorme. On peut ignorer les colonnes des VA dès qu'elles ne sont plus en base.

Les cas spéciaux suivants peuvent aussi se produire :

- Le PL modifié n'a pas d'optimum borné et $x_A = 0$: le PL d'origine est aussi **non borné**.
- Le PL modifié n'a pas d'optimum borné et $x_A \neq 0$: le PL d'origine n'a **pas de solution**.
- Le PL modifié a un optimum borné, mais $x_A \neq 0$: le PL d'origine n'a **pas de solution**.

Voyons ce que donne la méthode sur l'exemple.

- les tableaux T , base et b' sont identiques à ceux du début de la phase 1 dans la méthode des deux phases.
- la ligne Δ est chargée avec les coûts des variables légitimes et avec un coût $-M$ pour chaque VA.
- On rend nuls les coûts marginaux des variables de base en ajoutant la ligne 3, multipliée par M , à la ligne Δ .

Application à l'exemple : Max $z = x_1 + 2x_2 - Mx_6$

$$\begin{aligned}
 x_1 + x_2 + x_3 &= 6 \\
 x_2 + x_4 &= 3 \\
 x_1 + x_2 - x_5 + x_6 &= 1 \\
 x_1, x_2, x_3, x_4, x_5, x_6 &\geq 0
 \end{aligned}$$



Base	T	1	2	3	4	5	6	b'
3	1	1	1	0	0	0	0	6
4	2	0	1	0	1	0	0	3
6	3	1	1	0	0	-1	1	1
	Δ	1	2	0	0	0	-M	0
		M+1	M+2	0	0	-M	0	M

*ligne Δ +M*ligne 3*

$-z_B$

rendre nuls les coûts marginaux
des variables de base

Base	T	1	2	3	4	5	6	b'	$b'_i / T_{i2} > 0$
	1	1	1	1	0	0	0	6	6
	2	0	1	0	1	0	0	3	3
	3	1	1	0	0	-1	1	1	$1 \rightarrow s=3$
Δ		1	2	0	0	0	-M	0	
		M+1	M+2	0	0	-M	0	M	
			e=2					-Z_B	

Une itération mène à une solution réalisable.

Base	T	1	2	3	4	5	6	b'	
	1	0	0	1	0	1	-1	5	
	2	-1	0	0	1	1	-1	2	
	3	1	1	0	0	-1	1	1	
Δ		-1	0	0	0	2	-M-2	-2	
								-Z_B	
									$ligne 1 - ligne 3$
									$ligne 2 - ligne 3$
									$ligne \Delta - (M+2) * ligne 3$

Base T 1 2 3 4 5 6 b' $b'_i / T_{i5} > 0$

$\begin{matrix} 3 \\ 4 \\ 2 \end{matrix}$	1	0 0 1 0 1 -1
	2	-1 0 0 1 1 -1
	3	1 1 0 0 -1 1

$\begin{matrix} 5 \\ 2 \\ 1 \end{matrix}$	5
	$2 \rightarrow s=2$
	-

Δ	-1 0 0 0 2 -M-2
	\uparrow $e=5$

$\begin{matrix} -2 \end{matrix}$
$-z_B$

L'algorithme du simplexe continue en ignorant la colonne 6.

Base T 1 2 3 4 5 6 b' $b'_i / T_{i5} > 0$

$\begin{matrix} 3 \\ 5 \\ 2 \end{matrix}$	1	1 0 1 -1 0
	2	-1 0 0 1 1
	3	0 1 0 1 0

$\begin{matrix} 3 \\ 2 \\ 3 \end{matrix}$	3 $\rightarrow s=1$	$ligne 1 - ligne 2$
	-	-
	-	$ligne 3 - (-1)*ligne 2$

Δ	1 0 0 -2 0
	\uparrow $e=1$

$\begin{matrix} -6 \end{matrix}$	$ligne \Delta - 2 * ligne 2$
----------------------------------	------------------------------

Base

1
5
2

T

	1	2	3	4	5	6
1	1	0	1	-1	0	
2	-1	0	0	1	1	
3	0	1	0	1	0	

3
2
3

$$b'_i / T_{i5} > 0$$

$$3 \rightarrow s=1$$

-

 Δ

1	0	0	-2	0
---	---	---	----	---

-6

$$-z_B$$

$e=1$



Base

1
5
2

T

	1	2	3	4	5	6
1	1	0	1	-1	0	
2	0	0	1	0	1	
3	0	1	0	1	0	

3
5
3

$$b'_i / T_{i5} > 0$$

$$3 \rightarrow s=1$$

$$\begin{aligned} & \text{ligne } 2 - (-1) * \text{ligne } 1 \\ & \text{ligne } 3 - 0 * \text{ligne } 1 \end{aligned}$$

 Δ

0	0	-1	-1	0
---	---	----	----	---

-9

$$-z_B$$

$$\text{ligne } \Delta - \text{ligne } 1$$

On retrouve l'optimum calculé par la méthode des deux phases

Solution optimale de coût 9 avec $(x_1, x_2) = (3,3)$.

Programmation linéaire en nombres entiers

⇒ les **variables** sont astreintes à être **entières**.

En particulier, les variables peuvent être simplement **booléennes**, c'est-à-dire ne prendre que les valeurs 0 ou 1.

De nombreuses contraintes, en apparence non linéaires, peuvent être linéarisées grâce à des variables entières.

Par exemple, soit deux actions qui peuvent être exécutées ou non.

On peut définir deux variables binaires x et y valant 1 si l'action correspondante est effectuée.

Si les deux actions sont exclusives, on peut introduire la contrainte $x+y \leq 1$.

Définition d'un programme linéaire en nombres entiers (**PLNE**)

$$\text{Min } Z = C.x$$

$$\begin{cases} A.x = b \\ x \in \mathbb{N}^n \end{cases}$$

En général, un tel problème n'a des solutions que si A et b sont aussi **entiers**.

Si on relâche la contrainte d'intégrité sur les variables, on obtient un programme linéaire (PL) ordinaire appelé le **PL relaxé associé au PLNE**.

Si le PL relaxé avait un optimum entier, ce serait aussi l'optimum du PLNE.

Méthodes arborescente de résolution des PLNE

= méthodes de **séparation et d'évaluation** (*branch and bound*),

=> méthodes utilisées par les logiciels du commerce.

Principe : choisir une variable x et **séparer** le problème en deux sous-problèmes selon les valeurs de x .

- Pour un PLNE général, on sépare en considérant un entier p et les deux sous-problèmes $x \leq p$ et $x \geq p+1$.
- Pour un PL en 0-1, on sépare en considérant les deux cas $x=0$ et $x=1$.

Les PLNE des sous-problèmes peuvent à leur tour être séparés, ce qui forme progressivement une arborescence dont chaque nœud correspond à un sous-problème.

La majorité des sous-problèmes sont éliminés grâce à une **évaluation**.

En chaque nœud P , cette évaluation $ev(P)$ doit être une **borne inférieure** (en *minimisation*) ou une **borne supérieure** (en *maximisation*) du coût optimal du PLNE.

Une évaluation répandue est de résoudre le PL relaxé avec le simplexe.

- Pour la PLNE générale, nous voyons la **méthode de Dakin**, qui utilise le PL relaxé pour évaluer les solutions.
- Pour les PL en 0-1, nous décrivons la **méthode de Balas**, plus sophistiquée.

Méthode arborescente de Dakin

principe de la méthode

La racine de l'arbre S_0 correspond au PL relaxé. Son domaine de solutions réalisables inclut toutes celles du PLNE. On résout le PL relaxé avec l'algorithme du simplexe.

Si, par hasard, le simplexe trouve une **solution entière en S_0** , on s'arrête : on a l'**optimum** du PLNE.

Sinon, le coût maximum rendu par le simplexe donne une évaluation par excès $\text{ev}(S_0)$ du PLNE.

On initialise alors le coût de la meilleure solution entière déjà trouvée, w , à $-\infty$.

Si le PL relaxé n'a pas de solution entière, une variable non entière x_j^* est choisie dans la solution optimale du PL relaxé. Le plus souvent, on prend la variable dont la valeur est la plus proche d'un entier. Le nœud actuel est séparé en deux fils. L'un reprend le PL relaxé du nœud S_0 avec la contrainte supplémentaire $x_j \leq \text{Int}(x_j^*)$, Int désignant la partie entière. L'autre reprend également le PL relaxé de S_0 , mais avec la contrainte additionnelle $x_j \geq \text{Int}(x_j^*) + 1$. Les nœuds créés forment une arborescence.

Aux itérations suivantes, on choisit le nœud-feuille S_i de plus grande évaluation (*si maximisation*, le plus prometteur). Il est évalué en résolvant son PL relaxé avec le simplexe.

Si $\text{ev}(S_i) \leq w$, le nœud est supprimé. Sinon, si la solution est entière, on la conserve comme meilleure solution provisoire pour le PLNE, on met à jour w avec $\text{ev}(S_i)$, et on supprime le nœud. Si la solution n'est pas entière, le nœud reste dans l'arborescence pour les itérations suivantes.

Quand tous les nœuds en attente ont été supprimés, la recherche est terminée.

La dernière solution entière provisoire trouvée, s'il y a une, est la solution optimale du PLNE de départ.

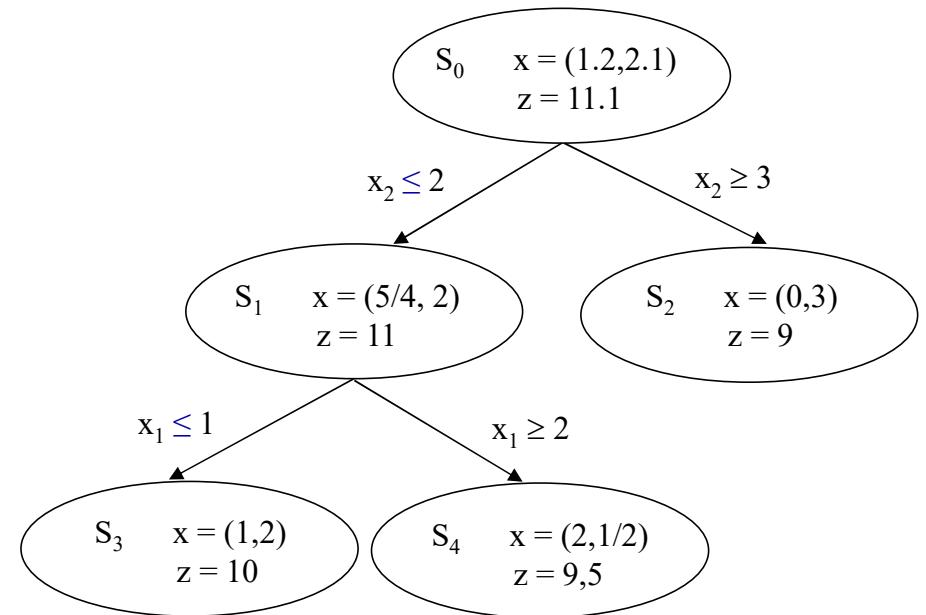
Soit **l'exemple** suivant :

$$\text{Max } z = 4x_1 + 3x_2$$

$$3x_1 + 4x_2 \leq 12$$

$$4x_1 + 2x_2 \leq 9$$

$$x_1, x_2 \in \mathbb{N}$$



En S_0 , aucune variable n'est entière, mais on sait que le coût maximal du PLNE est au plus 11.1.

On sépare x_2 , variable non entière la plus proche d'un entier : on construit deux fils, l'un S_1 avec la contrainte supplémentaire $x_2 \leq 2$, l'autre S_2 avec la contrainte $x_2 \geq 3$. On résout immédiatement les PL de ces fils.

S_1 a encore une variable fractionnaire x_1 et une évaluation de 11. S_2 fournit une première solution entière $(0,3)$ de coût 9.

S_1 doit être développé, car d'après son évaluation il est possible qu'il fournisse des solutions meilleures, de coût 10 ou 11.

On sépare donc S_1 en S_3 (PL de S_1 plus la contrainte $x_1 \leq 1$) et en S_4 (PL de S_1 plus $x_1 \geq 2$).

S_3 fournit une meilleure solution entière $(1,2)$ de coût 10.

Pour S_4 , le simplexe donne la solution $(2, 0.5)$ avec un coût de 9.5 : ce nœud peut être supprimé car cette évaluation par excès des solutions entières est moins bonne que 10.

La recherche est terminée, la dernière solution entière trouvée (dans S_3) est donc **optimale**.

Méthode arborescente de Dakin

Algorithme général

```
Initialiser une liste L avec un nœud S(0) contenant le PL relaxé du PLNE
Résoudre ce PL et garder l'optimum (variables et objectif) dans le nœud
Initialiser w, coût de la meilleure solution entière trouvée, à -∞
Répéter
    Chercher dans L le nœud S(i) d'évaluation maximale (S(0) au début)
    Si ev(S(i)) ≤ w alors
        Supprimer dans L le nœud S(i)
    Sinon
        Si la solution stockée dans S(i) est entière alors
            Conserver cette solution comme meilleure solution entière
            w := ev(S(i))
            Supprimer dans L le nœud S(i)
        Sinon
            Choisir une variable non entière  $x^*(j)$ 
            Créer un fils avec le PL de S(i) et la contrainte  $x(j) \leq \text{Int}(x^*(j))$ 
            Résoudre le PL du nœud-fils et ranger la solution dans ce nœud
            Ajouter le nœud à L
            Créer l'autre fils avec la contrainte  $x(j) \geq \text{Int}(x^*(j)) + 1$ 
            Résoudre le PL du nœud-fils, ranger la solution dans ce nœud
            Ajouter le nœud à L
        FinSi
    FinSi
Jusqu'à ce que L soit vide
Si w = -∞ alors
    Signaler que le PLNE est infaisable
Sinon
    Sortir la meilleure solution trouvée et son coût w
FinSi.
```

Méthode de Balas

pour les PL en 0-1

Le PL doit être mis sous forme canonique avec $C \geq 0$. On peut toujours revenir à ce cas en remplaçant la variable x_j par $x'_j = 1 - x_j$ quand $c_j < 0$.

$$\text{Min } z = c \cdot x \quad (c \geq 0)$$

$$Ax \leq b$$

$$X \in \{0,1\}^n$$

Principe: méthode arborescente fixant progressivement à 0 ou à 1 des variables x_j .

Il s'agit d'une méthode dite **en profondeur d'abord** : on développe toujours le dernier sommet créé dans l'arborescence.

En séparant un sommet sur une variable x_j , on traite d'abord le fils $x_j=1$.

En effet, si le sommet séparé n'est pas terminal, c'est qu'il faut fixer à 1 au moins une variable pour améliorer la meilleure solution provisoire.

$L(t)$: ensemble des variables libres (non encore fixées)

$F_1(t)$: ensemble des variables fixées à 1

$F_0(t)$: ensemble des variables fixées à 0.

$Q(t) = \{ i \mid s_i < 0 \}$: ensemble des lignes du PL courant ayant un second membre négatif.

$R(t)$: ensemble des indices de variables libres ayant un coefficient négatif sur au moins une des lignes de $Q(t)$:

$$R(t) = \{ j \in L(t), \exists i \in Q(t), a_{ij} < 0 \}.$$

Pour construire une solution réalisable du PL courant, il faut choisir dans $R(t)$ une variable et la fixer à 1.

On choisit une de ces variables pour séparer.

Constatant qu'on a une solution réalisable quand tous les seconds membres s_i sont non négatifs, Balas a proposé une mesure empirique simple de la **proximité** d'une solution, $P(t) < 0$:

$$\sum_{i=1,m} \min(0, s_i)$$

Si on fixe x_j à 1, la proximité d'une solution est alors : $P(t,j) = \sum_{i=1,m} \min(0, s_i - a_{ij})$
(calcul à faire pour chaque variable de coef négatif)

Balas propose pour la séparation la variable j^* qui augmente le plus la proximité d'une solution si on la met à 1, c'est-à-dire la plus prometteuse pour trouver une nouvelle solution rapidement : $P(t,j^*) = \max_{j \in R(t)} \{ P(t,j) \}$

Remarques :

- Si $P(t,j^*) = 0$, la solution obtenue en posant : $x_j^* = 1$ et $\forall j \in L(t) \setminus \{j^*\}$, $x_j = 0$, est une nouvelle **solution réalisable**.
- Pour départager des ex æquo ayant la même mesure de proximité, on choisit la variable de plus petit coût.
- $\exists i \in \{1, 2, \dots, m\}$ tel que $\sum_{j \in L(t)} \min(0, a_{ij}) > s_i$, le sommet peut être écarté pour **absence de solutions**.

Exemple :

$$\begin{array}{l}
 \text{Min} \quad 5x_1 \quad + 7x_2 \quad + 10x_3 \quad + 3x_4 \quad + x_5 \\
 -x_1 \quad + 3x_2 \quad - 5x_3 \quad - x_4 \quad + 4x_5 \quad \leq -2 \\
 2x_1 \quad - 6x_2 \quad + 3x_3 \quad + 2x_4 \quad - 2x_5 \quad \leq 0 \\
 x_2 \quad - 2x_3 \quad + x_4 \quad + x_5 \quad \leq -1 \\
 \end{array}$$

$x \in \{0,1\}^5$

noeud 0

$$P(0,3) = \sum_{i=1,m} \min(0, s_i - a_{i3})$$

Traitement du nœud racine S_0

Au nœud-racine S_0 , on a $L(0)=1\dots5$, $Q(0)=\{1,3\}$ et $R(0)=\{1,3,4\}$. On a l'évaluation triviale $ev(S_0)=0$. Le tableau permet de visualiser les PL résiduels et calculer les proximités.

Coefficients et seconds membres du PL courant						Second membre $s_i - a_{ij}$ si		
x_1	x_2	x_3	x_4	x_5	s_0	$x_1=1$	$x_3=1$	$x_4=1$
-1	3	-5	-1	4	-2	-1	3	-1
2	-6	3	2	-2	0	-2	-3	-2
0	1	-2	1	1	-1	-1	1	-2
						$P(0) \rightarrow$	-4	-3
								-5

Les colonnes de droite donnent les nouvelles valeurs des seconds membres si on met à 1 la variable correspondante x_j de $R(t)$. La proximité $P(t,j)$ est alors la somme des termes négatifs de la colonne j . Ici, $\max P(t,j) = -3$ pour $j^*=3$, donc on sépare sur $x_3=1$ ou 0, ce qui donne d'abord le nœud S_1 pour $x_3=1$. Le nœud S_4 avec $x_3=0$ sera construit plus tard.

Traitement du nœud S_1

Coefficients et seconds membres du PL courant						$s_i - a_{ij}$ si	
x_1	x_2	x_4	x_5	s_1	$x_2=1$	$x_5=1$	
-1	3	-1	4	3	0	-1	
2	-6	2	-2	-3	3	-1	
0	1	1	1	1	0	0	
						P(1)->	0
							-2

Traitement du nœud S_3

x_1	x_4	x_5	s_3
-1	-1	4	3
2	2	-2	-3
0	1	1	1

Traitement du nœud S_4

x_1	x_2	x_4	x_5	s_4
-1	3	-1	4	-2
2	-6	2	-2	0
0	1	1	1	-1

On a $x_3 = 1$, $L(1) = \{1,2,4,5\}$, $F1(1) = \{3\}$, $Q(1) = \{2\}$, $R(1) = \{2,5\}$, $\text{ev}(S_1) = 10$ (borne inférieure de l'objectif).

On sépare sur x_2 . le nœud S_2 obtenu pour $x_2=1$ est en fait une **première solution réalisable**.

Cette solution est $x = (0,1,1,0,0)$ de coût 17.

C'est la **meilleure solution provisoire**. (borne supérieure)

Il reste à traiter le fils droit S_3 de S_1 , correspondant à $x_3=1$ et $x_2=0$.

On a : $L(3) = \{1,4,5\}$, $F1(3) = \{3\}$, $F0(3) = \{2\}$, $Q(3) = \{2\}$, $R(3) = \{5\}$.

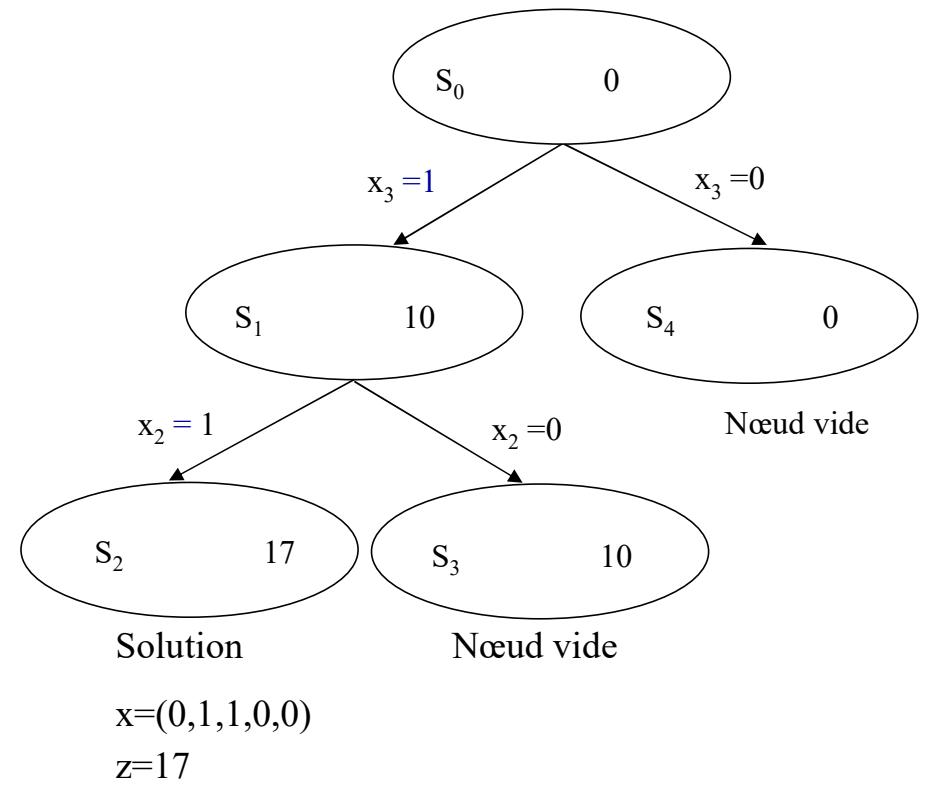
On s'aperçoit que dans la deuxième contrainte (somme des $a_{2j} < 0$) = $-2 > -3$. On laisse donc tomber le nœud S_3 car il n'a **pas de solution réalisable** (*même si toutes les variables à coef > 0 sont à 0, et les variables à coef négatif sont à 1, on a $-2 < -3$: faux*).

On remonte sur le nœud-racine S_0 , puis on construit le nœud S_4 correspondant à $x_3=0$. L'ensemble des variables libres est alors $L(4) = \{1,2,4,5\}$. Le tableau montre que la somme des coefficients négatifs dans la troisième contrainte est (somme des $a_{3j} < 0$) = $0 > -1$, donc il n'y a **pas de solution**. C'est la fin de la recherche arborescente.

La solution réalisable découverte en premier est donc optimale : $x = (0,1,1,0,0)$ de coût 17.

Synthèse :

$$\begin{array}{llllll}
 \text{Min} & 5x_1 & + 7x_2 & + 10x_3 & + 3x_4 & + x_5 \\
 & -x_1 & + 3x_2 & - 5x_3 & - x_4 & + 4x_5 & \leq -2 \\
 & 2x_1 & - 6x_2 & + 3x_3 & + 2x_4 & - 2x_5 & \leq 0 \\
 & & x_2 & - 2x_3 & + x_4 & + x_5 & \leq -1 \\
 & x \in \{0,1\}^5
 \end{array}$$



S ₀	Coefficients et seconds membres du PL courant						Second membre s _i -a _{ij} si		
	x ₁	x ₂	x ₃	x ₄	x ₅	S ₀	x ₁ =1	x ₃ =1	x ₄ =1
	-1	3	-5	-1	4	-2	-1	3	-1
	2	-6	3	2	-2	0	-2	-3	-2
	0	1	-2	1	1	-1	-1	1	-2
	P(0)->						-4	-3	-5

S ₁	Coefficients et seconds membres du PL courant					s _i -a _{ij} si	
	x ₁	x ₂	x ₄	x ₅	S ₁	x ₂ =1	x ₅ =1
	-1	3	-1	4	3	0	-1
	2	-6	2	-2	-3	3	-1
	0	1	1	1	1	0	0
	P(1)->					0	-2

S ₃	x ₁	x ₄	x ₅	s ₃	S ₄	x ₁	x ₂	x ₄	x ₅	s ₄
	-1	-1	4	3		-1	3	-6	-1	4
	2	2	-2	-3		2	-6	2	-2	0
	0	1	1	1		0	1	1	1	-1

Méthode Branch and Bound

cas général

Principe des méthodes B&B

Pour plusieurs problèmes, en particulier les problèmes d'optimisation, l'ensemble de leurs solutions est fini (en tous les cas, il est dénombrable). Il est donc possible, en principe, d'énumérer toutes ces solutions, et ensuite de prendre celle qui nous arrange. L'inconvénient majeur de cette approche est le nombre prohibitif du nombre de solutions : il n'est guère évident d'effectuer cette énumération.

La méthode de branch and bound (procédure par évaluation et séparation progressive) consiste à énumérer ces solutions d'un manière intelligente en ce sens que, en utilisant certaines propriétés du problème en question, cette technique arrive à éliminer des solutions partielles qui ne mènent pas à la solution que l'on recherche.

De ce fait, on arrive souvent à obtenir la solution recherchée en temps raisonnable. Dans le pire cas, on retombe toujours sur l'élimination explicite de toutes les solutions du problème. Pour ce faire, cette méthode se dote d'une fonction qui permet de mettre une borne sur certaines solutions pour soit les exclure soit les maintenir comme des solutions potentielles.

La performance d'une méthode de branch and bound dépend, entre autres, de la qualité de cette fonction (de sa capacité d'exclure des solutions partielles tôt).

On représente l'exécution de la méthode de branch-and-bound à travers une arborescence.

La racine de cette arborescence représente l'ensemble de toutes les solutions du problème considéré.

Méthode Branch and Bound

cas général

Principe des méthodes B&B

- Soit S l'ensemble des solutions admissibles d'un problème combinatoire P
- $f(s)$ le critère considéré pour évaluer une solution $s \in S$
- On cherche une solution $s^* \in S$ telle que $f(s^*) = \min_{s \in S} f(s)$
(on raisonne ici sur un problème de minimisation)

Hypothèses : (nécessaires appliquer la méthode de branch-and-bound)

- On suppose qu'il existe une heuristique permettant de calculer une **borne supérieure** B du minimum recherché : $f(s^*) \leq B$
- On suppose qu'on peut effectuer une **partition** $\{S_1, S_2, \dots, S_k\}$ de S telle que :
 - $S_1 \cup S_2 \cup \dots \cup S_k = S$
 - $S_i \cap S_j = \emptyset$ pour tout $i, j \in \{1, \dots, k\}$ et $i \neq j$
- On suppose qu'il est possible, pour tout $i = 1, \dots, k$, de calculer une **borne inférieure** b_i de $f(s)$ sur S_i : $\forall s \in S_i, b_i \leq f(s)$

Déroulement :

La méthode commence par considérer le problème de départ avec son ensemble de solutions, appelé la racine. Des procédures de bornes inférieures et supérieures sont appliquées à la racine. En fonction des valeurs trouvées, l'ensemble des solutions est divisé en deux ou plusieurs sous-problèmes, devenant ainsi des enfants de la racine. La méthode est ensuite appliquée récursivement à ces sous-problèmes (nœuds), engendrant ainsi une arborescence.

Si une solution optimale est trouvée pour un sous-problème, elle est réalisable, mais pas nécessairement optimale pour le problème départ. Comme elle est réalisable, elle peut être utilisée pour éliminer toute sa descendance. La recherche continue jusqu'à ce que tous les nœuds soient explorés ou éliminés.

Quelles conclusions peut-on avoir ?

Pour tout $i = 1, \dots, k$ (*à un nœud donné*)

- Si $b_i = B$ alors optimum local (*solution optimale globale si on se trouve à la racine de l'arborescence*)
- Si $b_i > B$ alors S_i ne peut contenir une solution optimale
 \Rightarrow abandonner la recherche dans S_i

(*si la borne inférieure d'un nœud dépasse la valeur d'une solution déjà connue, alors la solution optimale globale ne peut être contenue dans le sous-ensemble de solutions représenté par ce nœud*)

- Si $b_i \leq B$ alors S_i peut contenir une solution optimale
 \Rightarrow continuer la recherche dans S_i

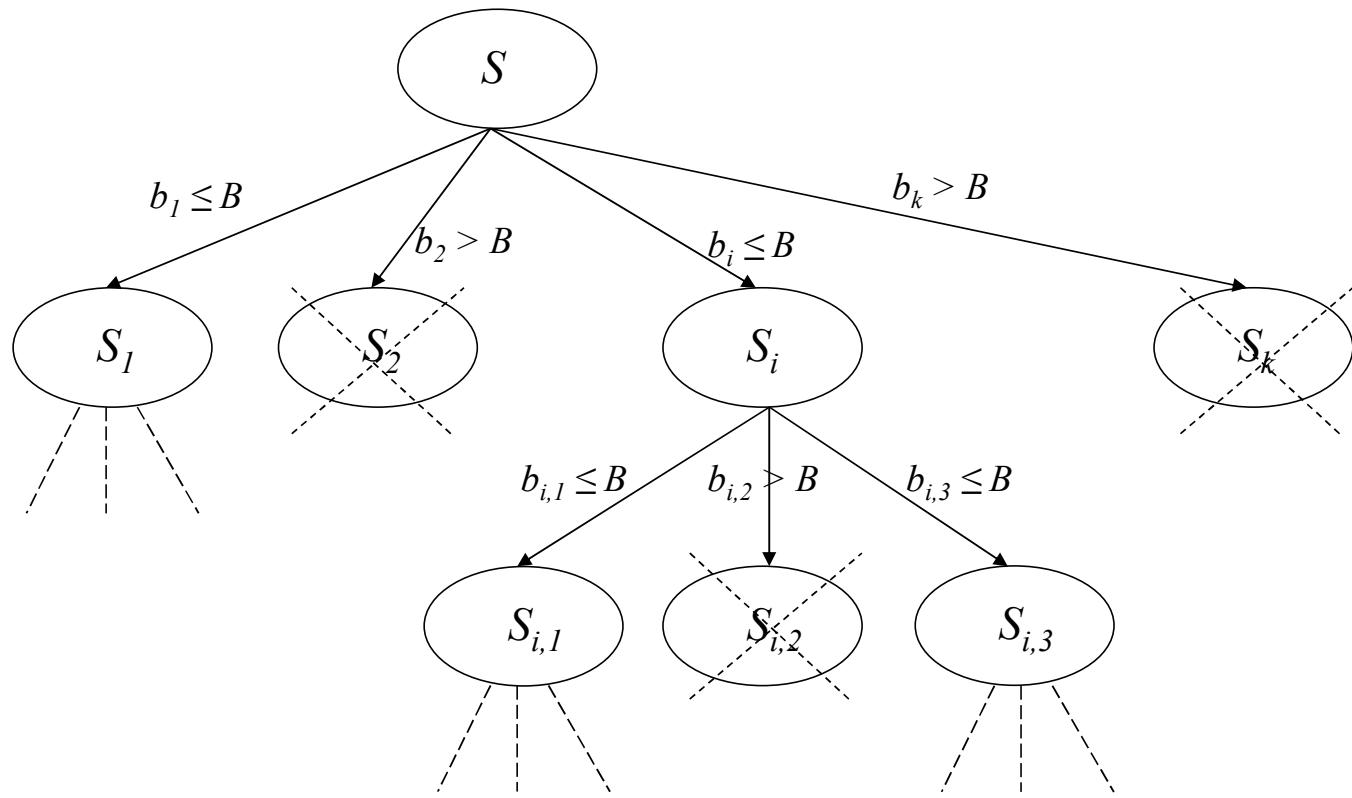
(*l'ensemble des solutions est divisé en deux ou plusieurs sous-problèmes, devenant ainsi des enfants du nœud courant*)

Continuer la recherche dans S_i veut dire qu'on effectue une **partition** de S_i en k_i sous-ensembles $S_{i,1}, S_{i,2}, \dots, S_{i,k_i}$

Pour tout $j = 1, \dots, k_i$

- Si $b_{i,j} > B$ alors $S_{i,j}$ ne peut contenir une solution optimale
 \Rightarrow abandonner la recherche dans $S_{i,j}$
- Si $b_{i,j} \leq B$ alors $S_{i,j}$ peut contenir une solution optimale
 \Rightarrow continuer la recherche dans $S_{i,j}$

Si pour tout $j = 1, \dots, k_i$, on a abandonné la recherche dans $S_{i,j}$, alors ceci veut dire qu'on abandonne la recherche dans S_i



Schémas d'évolution de l'exploration

- en largeur d'abord
- en profondeur d'abord
- meilleur noeud d'abord (ex : borne inférieure minimale)

Borne supérieure

- Si on trouve dans un noeud donné une borne supérieure $B' < B$, on remplace B par B' dans la suite des calculs
- Le calcul de nouvelles bornes est généralement fait au niveau des feuilles de l'arbre de décision
=> la borne est la valeur du critère (fonction objectif) correspondant à la feuille

... En résumé

Ce qui différencie les approches par évaluation et séparation est :

- le schéma de séparation (comment partitionner S)
- le schéma d'évolution de l'exploration de l'arbre de décision
- les méthodes de calcul des bornes supérieures et inférieures
- la fréquence de calcul des bornes supérieures
- ...

Illustration sur un exemple simple

Problème d'affectation

Soient n personnes à affecter à n tâches. Le coût de l'affectation de la personne i à la tâche j est noté par c_{ij} . Le problème consiste à affecter chaque personne à une seule tâche de telle manière à minimiser le coût total. On suppose qu'une tâche ne peut être faite que par une seule personne.

Soit la matrice suivante des coûts :

	tâche 1	tâche 2	tâche 3	tâche 4	
C =	9	2	7	8	personne a
	6	4	3	7	personne b
	5	8	1	8	personne c
	7	6	9	4	personne d

Si par exemple, on décide de faire l'affectation (1, a) ; (2, c) ; (3, b) et (4,d), on a le coût total : $1 + 8 + 3 + 4 = 16$. Par contre, si on fait l'affectation (1, b) ; (2, c) ; (3, d) et (4,a), on a le coût total : $6 + 8 + 9 + 8 = 30$.

L'application de la méthode de branch-and-bound nécessite une fonction de borne inférieure. Une des ces fonctions de borne inférieure pourrait être celle-ci : quelle que soit la solution, son coût ne peut pas être plus petit que la somme des plus petits éléments de chacune des lignes de la matrice des coûts. Ici, $2 + 3 + 1 + 4 = 10$.

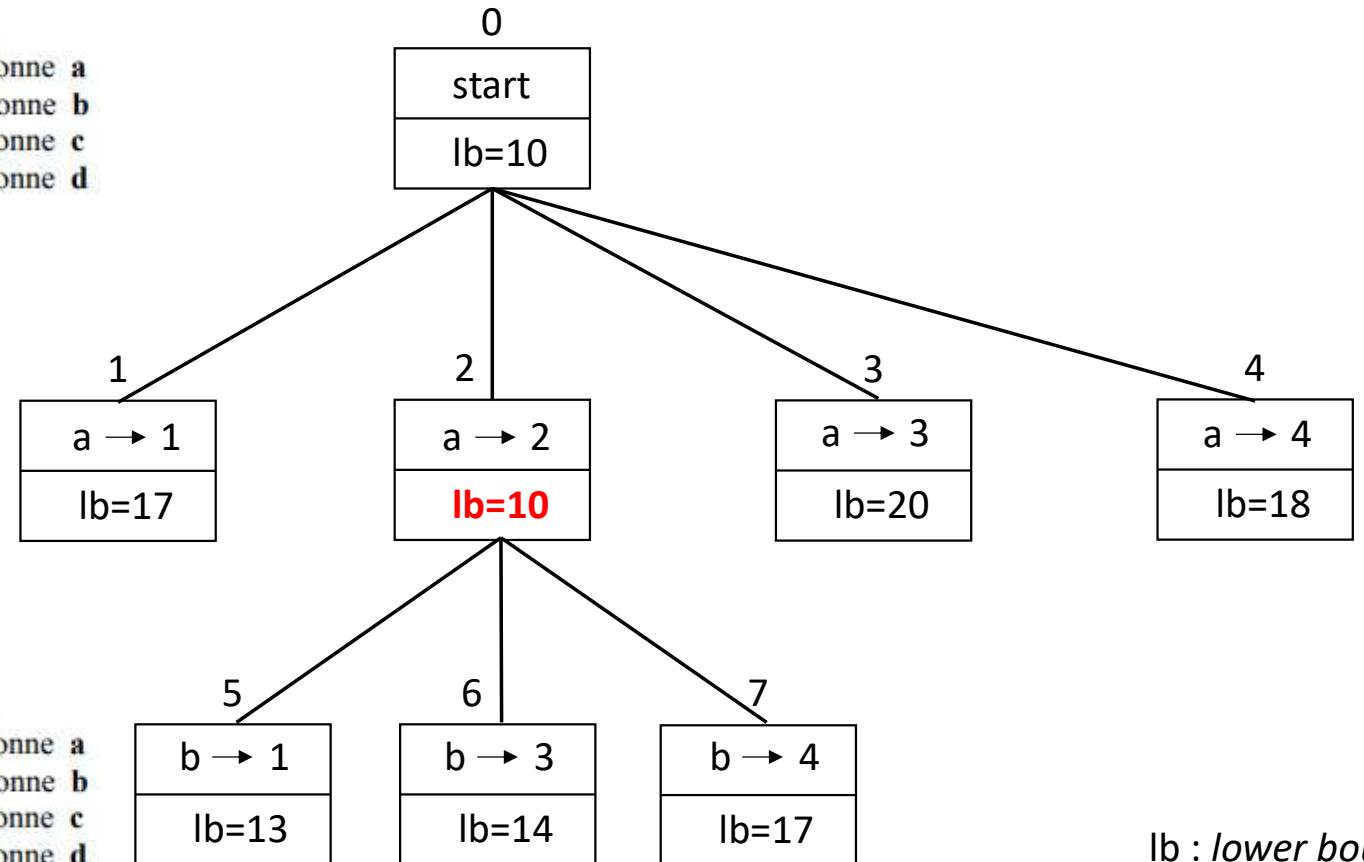
On commence par la racine qui correspond à l'état de « pas d'élément encore choisi de la matrice des coûts ». La valeur de la borne inférieure est alors égale à 10.

Les nœuds au premier niveau de l'arbre correspondent aux 4 jobs de la première ligne de la matrice à partir du moment que chacun d'eux est un choix potentiel de la première composante de la solution.

Le plus prometteur parmi eux est le nœud 2 car ayant la plus petite borne inférieure, comme illustré ci-dessous.

tâche 1	tâche 2	tâche 3	tâche 4	
9	2	7	8	personne a
6	4	3	7	personne b
5	8	1	8	personne c
7	6	9	4	personne d

C =



tâche 1	tâche 2	tâche 3	tâche 4	
9	2	7	8	personne a
6	4	3	7	personne b
5	8	1	8	personne c
7	6	9	4	personne d

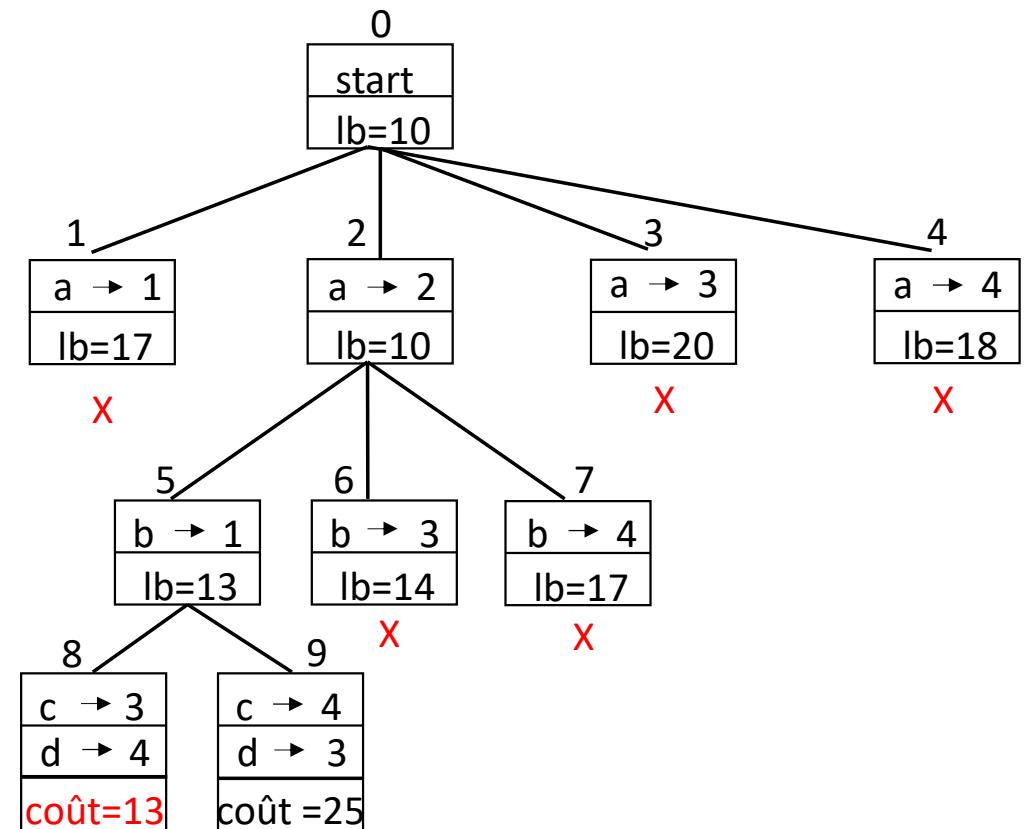
C =

lb : lower bound

En continuant, on génère les nœuds comme illustrés ici jusqu'à arriver à la solution (2,a) ; (1,b) ; (3,c) et (4,d) engendrant un coût de valeur 13.

Remarque 1: quand on arrive au noeud 8, on choisit le couple (c,3). Cela nous laisse obligatoirement avec le couple (4,d). C'est la première solution complète qu'on génère à ce stade des calculs. À partir de là, soit on génère une solution dont le coût est supérieur à la solution courante (comme c'est le cas pour le noeud 9) soit on ignore les nœuds car ne pouvant pas contenir une solution meilleure que celle dont on dispose (comme c'est le cas des nœuds 1, 3,4, 6 et 7).

Remarque 2: c'est par pur hasard que nous avons générée du premier coup la solution optimale.



	tâche 1	tâche 2	tâche 3	tâche 4	
C =	9	2	7	8	personne a
	6	4	3	7	personne b
	5	8	1	8	personne c
	7	6	9	4	personne d

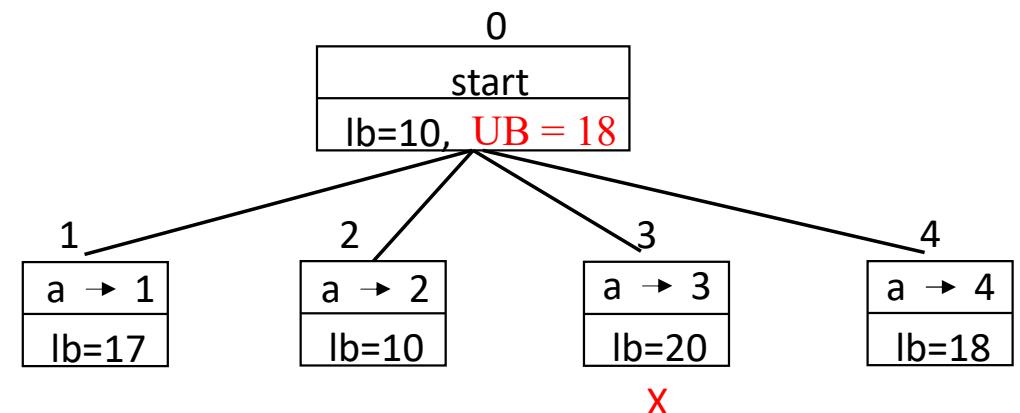
Remarque 3:

au nœud 0, prenons une solution au hasard: (1,a) ; (2,b) ; (3,c) et (4,d) engendrant un coût de valeur 18.

Soit cette solution est optimale, soit non. Dans tous les cas, on a : coût optimal ≤ 18 .

=> UB = 18 est une **borne supérieure** de la solution optimale, qui nous aurait permis de « couper » la branche à partir du nœud 3 plus tôt qu'à l'arrivée au nœud 8.

UB : upper bound



	tâche 1	tâche 2	tâche 3	tâche 4	
C =	9	2	7	8	personne a
	6	4	3	7	personne b
	5	8	1	8	personne c
	7	6	9	4	personne d

Bibliographie

- *Cahier de la recherche opérationnelle. Introduction à la recherche opérationnelle. Regard historique et applications actuelles, 2010
- *Michel Bierlaire, cours de recherche opérationnelle – optimisation, Ecole Polytechnique Fédérale de Lausanne/ENAC/TRANSP-OR
- *F. Huet, Introduction au cours « Optimisation et complexité », EFREI-L3
- *Bruno Bachelet, cours Informatique/Recherche opérationnelle, 1999-2000
- *François Clautiaux, cours Programmation Linéaire, université de Bordeaux
- *Hervé Manier, cours Logistique, université de technologie de Belfort-Montbéliard, 2010
- *Djamal Rebaïne, cours Branch and bound, université du Québec, Chicoutimi, 2005
- *Christian Prins, Marc Sevaux, Programmation linéaire avec Excel, éditions Eyrolles, 2011