REGLE DE TRANSFORMATION STD-CANONIQUE

canonique	$\Leftrightarrow$	standard
max		max
s.c $\leq$		s.c $=$
variables ≥ 0		variables ≥ 0

$x \in \Re : x = x^+ - x^- (x^+, x^- \geq 0)$

$\min(f(x)) = -\max(-f(x))$

$\geq \to \leq$: multiplier par - 1

$$ax = b \Leftrightarrow \begin{cases} ax \leq b \\ -ax \leq -b \end{cases}$$

inéquation $\to$ équation : rajoute des var. d'écart s $(s \geq 0)$

$ax \leq b \Leftrightarrow ax + s = b \quad ax \geq b \Leftrightarrow ax - s = b$

$$|x| \leq b \Leftrightarrow \begin{cases} x \leq b \\ x \geq -b \end{cases}$$

$$\min z = \max\{c_1,\dots,c_n x\} \Leftrightarrow \begin{cases} \min \quad z = t \\ \text{s.c} \quad t \geq c_i x \\ \quad\quad t \geq 0 \end{cases}$$

$$\min|x| \Leftrightarrow \begin{cases} \min \quad t \\ \text{s.c} \quad t = |x| \end{cases} \Leftrightarrow \begin{cases} \min \quad t \\ \text{s.c} \quad t \leq x \end{cases} (\text{cas } t \geq |x| \text{ non convexe}$$

$\text{EX} : x_i \geq -3 \Leftrightarrow \bullet\, x_i \in \Re, -x_i^+ + x_i^- \leq 3$

$\bullet$ changement de variable :

$x_i' = x_i + 3$

Tableau initial admissible admissible si les b_i sont ≥ 0 ⊕

Dégénéré : un zéro dans les b_i

Tableau optimal : tous les b_i et tous les coûts positifs

Primal non borné : une colonne entièrement négative

PHASE I – PRIMAL
(on peut aussi rajouter un second membre(b) nul)

Données : Un tableau non admissible. Résultat : Un tableau admissible ou un certificat d'absence de solutions admissibles.

(1) Construire le PL auxiliaire et un tableau initial admissible :

- Introduire la variable auxiliaire x_0 dans toutes les contraintes vérifiant $b_i < 0$, avec comme valeur = -1.
- Ajouter la fonction objectif auxiliaire (à maximiser) $z' = -x_0$.
- Faire entrer x_0 dans la base en pivotant sur α_{j0} où

j = min {i | b_i = min { b_k | $b_k < 0$ } :

(2) Résoudre le PL auxiliaire à l'aide de la phase II de l'algorithme du simplexe et en utilisant la règle de Bland.

- Si $z' = 0$ à l'optimum, supprimer les colonnes de x_0 et de z' et la ligne de z'. Le tableau restant est admissible pour le problème de départ.
- Si $z' < 0$ à l'optimum, le problème de départ n'admet pas de solutions admissibles.

PHASE I – DUAL

Rajouter une dernière ligne au tableau = 0 partout sauf pour z=1.Ce tableau est dual-admissible et on peut appliquer la phase II-dual(avec la règle de Bland)

PHASE II – PRIMAL

Données : Un tableau admissible. Résultat : Un tableau optimal ou non borné.

(1) Choix d'une colonne (variable) entrante : Choisir une colonne hors base r ayant un coût marginal négatif $r \in \{k \in N \mid -\gamma_k < 0\}$

S'il n'existe pas de colonne entrante : STOP le tableau courant est optimal.

(2) Choix d'une colonne (variable) sortante : Choisir une ligne j minimisant les quotients caractéristique

$j \in \{k \in \{1,\dots,m\} \mid \dfrac{\beta_k}{\alpha_{kr}} = \min\{\dfrac{\beta_i}{\alpha_{ir}} \mid \alpha_{ir} > 0\}\}$ S'il n'existe pas de colonne sortante : STOP le tableau courant est non borné.

(3) Mise à jour de la base et du tableau : pivoter autour de α_{jr} et retourner en (1).

PHASE II – DUAL

Données : Un tableau dual-admissible. Résultat : Un tableau optimal ou un certificat d'absence de solutions admissibles.

(1) Choix d'une variable sortante : Choisir une ligne i avec $\beta_i < 0$, la variable basique x_j avec $j = \sigma(i)$ quitte la base. S'il n'existe pas de variable sortante : STOP le tableau courant est optimal.

(2) Choix d'une variable entrante : Choisir une colonne hors base r maximisant les quotients caractéristiques duaux $r \in \{k \in N \mid \dfrac{-\gamma_k}{\alpha_{ik}} = \max\{\dfrac{-\gamma_j}{\alpha_{ij}} \mid \alpha_{ij} < 0\}\}$

S'il n'existe pas de variable entrante : STOP le dual est non borné et le primal sans solutions admissibles.

(3) Mise à jour de la base et du tableau : Pivoter autour de α_{ir} et retourner en (1).

PROGRAMME DUAL D'UN PL CANONIQUE

A tout PL canonique	On associe un programme dual
Max $z = cx$	Min $w = yb$
s.c $Ax \leq b$ (PLP)	s.c $yA \geq c$ (PLD)
$x \geq 0$	$y \geq 0$

THEOREME DE DUALITE FAIBLE

Soit x une solution admissible d'un PL canonique et **y** une solution admissible de son dual, alors $cx \leq yb$

THEOREME DE DUALITE FORTE

Si un programme linéaire standard possède une solution optimale x de valeur z, alors son dual possède aussi une solution optimale y. De plus, z=w

ECART COMPLEMENTAIRES
$Y_E X_D = 0 \;\&\; Y_D X_E = 0$

ALGO. DUAL – TABLEAUX PARTICULIERS

Dual Admissible ⊕ sur la dernière ligne

Dual non borné

x_1	…	x_n	z	
⊕	…	⊕	0	*
			⋮	–
			0	*
⊕	…	⊕	1	*

Sans solution admissible

x_1	…	x_n	z	
⊕	…	⊕	0	*
			⋮	–
			0	*
*	…	*	1	*

LA DUALITE

Problème de maximisation:	$\leftrightarrow$	Problème de minimisation
Variable $x_j > 0$	$\leftrightarrow$	j^e contrainte de type $\geq$
Variable $x_j \in \mathbb{R}$	$\leftrightarrow$	j^e contrainte de type $=$
Variable $x_j < 0$	$\leftrightarrow$	j^e contrainte de type $\leq$
i^e contrainte de type $\leq$	$\leftrightarrow$	Variable $y_i \geq 0$
i^e contrainte de type $=$	$\leftrightarrow$	Variable $y_i \in \mathbb{R}$
i^e contrainte de type $\geq$	$\leftrightarrow$	Variable $y_i \leq 0$

REGLE DE BLAND : Lorsque plusieurs candidats sont susceptibles d'entrer ou de sortir de la base, les départager en choisissant toujours la variable x_r ayant le plus petit index r.

AJOUT DE CONTRAINTES : écrire la contrainte sous forme standard, écrire l'expression à partir du tableau optimal et l'introduire dans le tableau optimal et le rendre optimal.

SENSIBLITE DU MEMBRE DE DROITE :

la base reste optimale pour $\delta B_i^{-1} \geq -\beta$

Variation de la fonction objectif : $z' = z^* + \delta \cdot y_i$

x_D	x_E	z	
$B^{-1}A$	B^{-1}	0	$B^{-1}b$
$c_B B^{-1}A - c_D$	$c_B B^{-1}$	1	$c_B B^{-1}b$

SENSIBILITE DE LA FONCTION OBJECTIF C :

La base reste optimale tant que $-\gamma_N' = c_B' B^{-1} N - c_N' \geq 0$

Pour variable hors base :

$\delta \leq -\gamma_j \Leftrightarrow c_j' \in (-\infty, c_j - \gamma_j]$

x_N	x_B	z	
$B^{-1}N$	I	0	$B^{-1}b$
$c_B B^{-1}N - c_N$	0	1	$c_B B^{-1}b$

LES GRAPHES NON ORIENTES
V(|V|=n) (|E|=m) $\Psi : E \to P_2(V)$ **fct d'incidence.**

Matrice d'adjacence : sommets-sommets $n \times n$

$$b_{ij} = \begin{cases} 1 \quad si\ v_i\ et\ v_j\ sont\ adjacents \\ 0 \quad \sin on \end{cases}$$

Matrice d'incidence : sommets-arêtes $n \times m$

$$b_{ij} = \begin{cases} 1 \quad si\ v_i\ et\ v_j\ sont\ adjacents \\ 0 \quad \sin on \end{cases}$$

LES GRAPHES ORIENTES
V(|V|=n) (|E|=m) $\Psi : E \to V \times V$ **fct d'incidence.**

Matrice d'adjacence : sommets-sommets $n \times n$

Que pour les flèches qui sortent

$$b_{ij} = \begin{cases} 1 \quad si\ (v_i, v_j) \in E \\ 0 \quad \sin on \end{cases}$$

Matrice d'incidence : sommets-arcs $n \times m$

$$b_{ij} = \begin{cases} -1 \quad \text{si } v_i \text{ est l'ext initiale de } e_k \\ -1 \quad \text{si } v_i \text{ est l'ext finale de } e_k \\ 0 \quad \text{sinon} \end{cases}$$

DEFINITIONS ET PROPRIETES

Une arête (un arc) dont les extrémités sont confondues est une boucle. Un graphe simple est un graphe sans boucle ni arêtes (arcs) multiples. Graphe désignera un graphe simple fini. Si des boucles ou des arêtes (arcs) multiples sont autorisées, on parlera de graphe général ou de multigraphe.
Remarque : Deux arcs de sens opposés ne sont pas des arcs multiples !
Le rang de la matrice d'incidence sommets-arcs d'un digraphe sur n sommets est inférieur ou égal à n-1.

G' est un graphe partiel de G si G' = (V;E') avec $E' \subseteq E$. G' est un sous-graphe de G induit par W si G' = (W;E(W)) où $W \subseteq V$ et E(W) est l'ensemble des arêtes (arcs) ayant leurs deux extrémités dans W.

DEGRES

Soit G un multigraphe, le degré du sommet v, noté deg(v), est égal au nombre d'arêtes (d'arcs) incidentes à v.
Remarque. Si un sommet possède une ou plusieurs boucles, chacune apporte une contribution de 2 dans le calcul du degré de ce sommet. Soit G un multigraphe orienté, le degré extérieur du sommet v, noté $\deg_+(v)$, est égal au nombre d'arcs issus de v. Similairement, le degré intérieur du sommet v, noté $\deg_-(v)$, est égal au nombre d'arcs se terminant en v.
Propriété : Dans tout graphe, la somme des degrés est un nombre pair.

CHAINES ET CYCLES

Une chaîne est une suite alternée de sommets et d'arêtes
C = (u0; f1; u1; f2; u2; : : : ; uk_1; fk; uk);
où $u_i \in V \forall i, f_i \in E \forall i$ et $f_i = \{u_{i-1}, u_i\} \forall i$ Un cycle est une chaîne des les deux extrémités sont confondues. Une chaîne (un cycle) est élémentaire si chaque sommet y apparaît au plus une fois. Une chaîne (un cycle) est simple si chaque arête y apparaît au plus une fois. La longueur d'une chaîne (d'un cycle) est le nombre d'arêtes de la chaîne (du cycle).

CHEMINS ET CIRCUITS

Un chemin est une suite alternée de sommets et d'arcs
C = (u0; f1; u1; f2; u2; : : : ; uk_1; fk; uk);
où $u_i \in V \forall i, f_i \in E \forall i$ et $f_i = \{u_{i-1}, u_i\} \forall i$ Un circuit est un chemin dont les deux extrémités sont confondues.

CONNEXITE

Soit G = (V;E; Ψ) un multigraphe (non orienté). On définit sur V une relation de connexité C par
$v_i C v_j \Leftrightarrow$ soit $v_i = v_j$ soit il existe une chaîne entre vi et vj.
C est une relation d'équivalence : réflexive, symétrique, transitive.

CONNEXITE FORTE

$iCFv_j \Leftrightarrow$ soit $v_i = v_j$ soit il existe un chemin de v_i à v_j ET un chemin de v_j à v_i

NOMBRE CYCLOMATIQUE

Soit G un multigraphe avec n sommets, m arêtes et p composantes connexes. Le nombre cyclomatique v(G) de G est v(G) = m _ n + p. Pour tout multigraphe G, on a $v(G) \ge 0$. De plus, v(G) = 0 si et seulement si G est sans cycle.

THEOREMES

Soit G = (V;E; Ψ) un multigraphe à n sommets. Les affirmations suivantes sont équivalentes. (a) G est un arbre.
(b) G est sans cycle et connexe. (c) G est sans cycle et comporte n _ 1 arêtes. (d) G est connexe et comporte n _ 1 arêtes. (e) G est sans boucle et chaque paire de sommets est reliée par une et une seule chaîne simple.
Tout arbre fini sur $n \ge 2$ sommets possède au moins deux sommets pendants ou feuilles, c'est-à-dire deux sommets incidents à une seule arête.
Dans tout arbre fini sur $n \ge 2$ sommets, on peut choisir arbitrairement un sommet v1 et trouver une numérotation v2,…. ,vn des sommets restants ainsi qu'une numérotation e2,…,en des arêtes telles que, pour tout $i \ge 2$, l'arête e_i ait pour extrémités le sommet v_i et un sommet v_j avec $j < i$
La matrice d'incidence sommets-arêtes d'un arbre sur n sommets a rang n-1

KRUSKAL

Trier, classer, etc…

PRIM

De voisins en voisins, prendre l'arete la plus petite.

CHAINE DE SECTION OPTIMALE

L'inf-section de C est définie comme la valeur minimale des poids des arêtes de C. La sup-section de C est définie comme la valeur maximale des poids des arêtes de C. Pour deux sommets i et j de G donnés, le problème de l'inf-section maximale consiste à déterminer la chaîne d'inf-section maximale reliant ces deux sommets. Réciproquement, le problème de la sup-section minimale consiste à déterminer une chaîne de sup-section minimale reliant ces deux sommets.

L'arbre recouvrant de poids maximum fournit des chaînes d'inf-section maximale entre toutes paires de sommets d'un graphe. L'arbre recouvrant de poids minimum fournit des chaînes de sup-section minimale entre toutes paires de sommets d'un graphe. chaîne de section maximale=chaîne d'inf-section maximale ;chaîne de section minimale=chaîne de sup-section minimale.

PLUS COURTS CHEMINS – BELLMAN

Si C est un plus court chemin de s à t et si u appartient à ce plus court chemin, alors les sous-chemins de s à u et de u à t sont également des plus courts chemins.
Si $\lambda_j \dots \lambda_n$ vérifient $\lambda_j \le \lambda_i + c_{ij}$ $\forall (i,j) \in E$ et si C est un chemin de s à t pour lequel $\lambda_j = \lambda_i + c_{ij}$ $\forall (i,j) \in C$
alors C est un plus court chemin de s à t.
Algorithme : (1) On part d'un vecteur λ initial défini par $\lambda_s = 0$ et $\lambda_i = \infty$ pour tout $i \ne s$.
(2) On tient à jour une liste de candidats L contenant initialement le sommet s.
(3) Tant que L est non vide, on retire un sommet de L, disons i, et, pour chacun des successeurs j de i, on teste si $\lambda_j > \lambda_i + c_{ij}$. Si tel est le cas, on pose $\lambda_j = \lambda_i + c_{ij}$ et on introduit j dans L (à moins qu'il n'y soit déjà).
Théorème :on retire de L le sommet d'étiquette minimale.

DIJKSTRA

Résultat :plus court chemin de s à i et le prédécesseur immédiat p(i) du somment i dans un tel chemin
(1) $\lambda_s = 0, \lambda_i = \infty \forall i \ne s, p(i) = \mathrm{NULL} \forall i, T = V$
(2) Tant que $T \ne \phi$ faire
(2.1) Soit i le sommet de T de plus petite étiquette λ_i (départager arbitrairement en cas d'égalité).
(2.2) Si un tel sommet n'existe pas ($\lambda_j = \infty \forall j \in T$) : STOP, les sommets encore dans T ne sont pas atteignables depuis s.
(2.3) Sinon, retirer i de T et pour tout successeur j de i encore dans T tester si $\lambda_j > \lambda_i + c_{ij}$ auquel cas poser $\lambda_j = \lambda_i + c_{ij}$ et $p(j) = i$.
Théorème : le graphe G = (V;E) est sans circuit si et seulement si on peut attribuer à chaque sommet i 2 V un nombre r(i), appelé le rang de i, tel que pour tout arc (i; j) $\in$ E on ait r(i) < r(j).

ALGORITHME DU RANG

Données : Un graphe orienté G = (V;E) sans circuit.
Résultat : Pour tout sommet i $\in$ V , un rang r(i) minimal.
(1) k = 1, W = V
(2) Tant que $W \ne \phi$ faire
(2.1) Soit X l'ensemble des sommets sans prédécesseur du sous-graphe $G_W = (W, E(W))$.
(2.2) Poser r(i) = k pour tout i $\in$ X.
(2.3) Poser W = W \ X et k = k + 1.

TRI TOPOLOGIQUE

Données : Un graphe orienté G = (V;E) sans circuit,|V| = n.
Résultat : Une numérotation $v : V \to \{1,\dots,n\}$ des sommets de G compatible avec le rang.
(1) k = 1, W = V
(2) Tant que $W \ne \phi$ faire
(2.1) Soit i un sommet sans prédécesseur du sous-graphe $G_W = (W, E(W))$.
(2.2) Poser $v(i) = k, W = W$ \{i} et k=k+1.

PLUS COURTS CHEMINS DANS LES RESEAUX SANS CIRCUITS

Pour calculer un plus court chemin dans un réseau sans circuit, il suffit de numéroter les sommets de manière compatible avec le rang (en effectuant un tri topologique du graphe) et de tester la condition d'optimalité en parcourant les sommets dans l'ordre croissant de leur numéro. Le plus souvent cet algorithme s'applique dans des réseaux possédant un seul sommet sans prédécesseur, le problème étant de trouver un plus court chemin de ce sommet à tous les autres. Dans un tel cas, après avoir trié topologiquement le graphe, on pose $\lambda_1 = 0$
et pour k de 2 à n on calcul $\lambda_k = \min\{\lambda_j + c_{jk} \mid j \in \mathrm{Pred}(k)\}$

APPLICATION A LA GESTION DE PROJET :LA METHODE DU CHEMIN CRITIQUE

Déterminer la durée minimale de réalisation d'un projet revient à calculer un plus long chemin entre les sommets α et ω du graphe associé.

ALGORITHME DU CHEMIN CRITIQUE

Données : Un réseau R = (V;E; c) associé à un projet et dont les sommets ont été numérotés de manière compatible avec le rang (le sommet α a numéro 1 et le sommet ω le numéro n).
Résultat : La durée minimale D du projet ainsi que, pour chaque tâche i, la date δ_i de début au plus tôt de la tâche et la date φ_i de début au plus tard de cette même tâche.

(1) Récurrence en avançant dans le projet (calcul des dates de début au plus tôt) $\delta_1 = 0$ Pour k = 2 à n poser

$$\delta_k = \max_{j \in Pred(k)} \{\delta_j + c_{jk}\} = \max_{j \in Pred(k)} \{\delta_j + d_j\}$$

(2) Récurrence en reculant dans le projet (calcul des dates de début au plus tard) $D = \delta_n, \quad \varphi_n = \delta_n$ Pour k = n-1 à 1 poser

$$\varphi_k = \min_{j \in Succ(k)} \{\varphi_j - c_{kj}\} = \min_{j \in Succ(k)} \{\varphi_j - d_k\}.$$

TERMINOLOGIE

Une tâche i est critique si $\delta_i = \varphi_i$. Tout retard dans l'exécution d'une tâche critique se répercute en un accroissement de la durée totale de réalisation du projet.

Un chemin critique est un chemin du sommet α (début des travaux) au sommet ω (fin des travaux) composé uniquement de tâches critiques. La longueur d'un chemin critique correspond à la durée minimale nécessaire à la réalisation du projet.

TRANSBORDEMENT

s : source $b_i < 0$

p : puits $b_i > 0$

t : transbordement $b = 0$

A. Résolution du problème auxiliaire (PHASE I)

1. Construction de réseau auxiliaire

   Source « centrale » : k

   Rajouter $(s,k), c_{sk} = 1$ si n'existe pas déjà

   Rajouter $(k,p), c_{kp} = 1$ si n'existe pas déjà

   Les autres $c_{ij} = 0$

2. Construction de l'arbre solution initiale admissible

   $\{(i,k) \mid i \text{ sommet source}\} \cup \{(k,j) \mid j \text{ sommet puits}\}$

   $\cup \{\text{arc supplémentaire pour rendre l'arbre couvrant}\}$

3. Calcul des solutions primales et duales → nouveau graphe

- Solutions primales :

  prendre un noeud pendant

  $x_{ij} = b_j$ (incident sur j)

  $x_{ji} = -b_j$ (sortant de j)

  $b_i = b_i + b_j$

  enlever noeud

- Solutions duales :

  pour un $i \in V$ arbitraire $y_i = 0$

  $y_j = y_i \pm c_{ij}$ (+ : incident sur j ; - sortant de j)

4. Tant que $z_{aux} \neq 0$:

- Choisir arc entrant :

  Dans l'ordre lexicographique, trouver le premier arc qui viole la contrainte de dualité :

  $y_j - y_i \le c_{ij}$

  S'il n'en existe pas, STOP, les solutions actuelles sont optimales.

- Choisir arc sortant :

  Créer cycle avec arc entrant :

blablabla

$$C = \begin{cases} C^+, \text{arcs dirigés dans le même sens qu'arc entrant} \\ C^-, \text{arcs dirigés dans le sens opposé qu'arc entrant} \end{cases}$$

Arc sortant serra déterminé par :

$$\Delta = \min\{x_{kl} \mid (k,l) \in C^-\}$$

S'il n'en existe pas, STOP, le réseau possède un circuit à coût négatif et le problème n'a pas d'optimum fini.

- Mettre à jour les variables :
  - Primale :

$$x_{ij} = \begin{cases} x_{ij} + \Delta, \text{si } (i,j) \in C^+ \\ x_{ij} - \Delta, \text{si } (i,j) \in C^- \\ x_{ij} \quad \text{si } (i,j) \notin C \end{cases}$$

  - Duales :

    Soit (u, v) l'arc entrant. Retirer temporairement l'arc sortant.

    T_u = arbre contenant le noeud u.

    T_v = arbre contenant le noeud v

    Seuls les $y \in T_v$ changent : $y' = y - \varepsilon$

$$\varepsilon = y_v - y_u - c'_{uv}$$

A. Résolution du problème départ (PHASE II)

1. Retirer les arcs artificiels et rétablir le coût unitaire initial des arcs.
2. Calculer les solutions duales

   Attention : considérer les arcs de l'arbre solution
3. Itérer

- Recherche arc entrant :

  S'il n'en existe pas, STOP, les solutions actuelles sont optimales.
- Recherche arc sortant :

  S'il n'en existe pas, STOP, le réseau possède un circuit à coût négatif et le problème n'a pas d'optimum fini.
- Mis à jour des variables.

Buttu Andreea