Introduction aux problèmes combinatoires

H. Fargier, C. Pralet, S. Roussel, G. Verfaillie

ISAE

Plan

Chemin d'un aéroport à un autre

Données

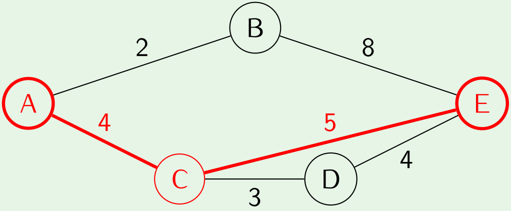
- **aéroports** de décollage et d'atterrissage,
- **points de passage**,
- **segments** possibles entre les aéroports et les points de passage,
- **longueurs** des segments.

Objectif

Trouver un **chemin** de **longueur totale minimum**

Chemin d'un aéroport à un autre

Exemple



Problème générique

Chemin le plus court dans un graphe pondéré : problème facile

Planification des décollages dans un aéroport

Données

- une seule **piste de décollage**,
- un ensemble d'**avions** qui doivent décoller sur un horizon temporel,
- des **durées minimum** entre les décollages (dépendant du type d'avion),
- les **heures de décollage** au plus tôt.

Objectif

Planifier les décollages de manière à **minimiser le délai maximum** atteint sur l'ensemble d'avions (la différence entre l'heure de décollage prévue et l'heure de décollage réelle).

Planification des décollages dans un aéroport

Exemple

Problème générique

Ordonnancement de tâches avec des durées et des heures de démarrage au plus tôt – minimisation du *makespan* ; voif le **problème du voyageur de commerce** : **problème difficile**

Maintenance d'une flotte d'avions

Données

- une **flotte d'avions**,
- des **besoins** de maintenance sur un horizon temporel,
- des **ressources** humaines et physiques,
- ses **contraintes** de maintenance.

Objectif

Construire un **plan de maintenance** satisfaisant

- les **besoins de maintenance**
- et les **contraintes**

et minimisant

- la **durée totale de non-disponibilité** de la flotte
- ou les **le nombre maximum d'avions indisponibles** au cours de l'horizon temporel

$\rightarrow$ problème **difficile**

Connectivité entre des équipements

Données

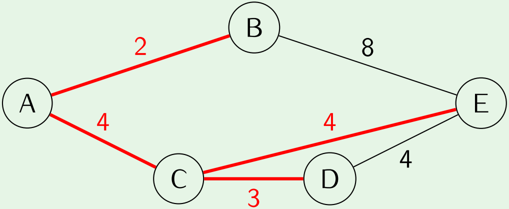
- des **équipements**,
- des **connexions** possibles entre les équipements,
- des **coûts** associés aux connexions.

Objectif

Connecter tous les équipements en **minimisant le coût total** des connexions.

Connectivité entre des équipements

Exemple



Problème générique

Trouver **l'arbre couvrant** de **poids minimum** dans un **graphe pondéré** : problème **facile**

Management d'un projet de conception

Données

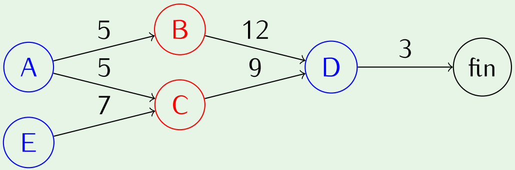
- des **tâches** de conception,
- des **durées** de réalisation associées aux tâches,
- des contraintes de **précédence** entre les tâches,
- des **ressources** : **besoins** et **contraintes**.

Objectif

Organiser les tâches dans le temps de manière à minimiser la durée totale du projet.

Management d'un projet de conception

Exemple



Problème générique

Management de projet : problème **facile** sans les contraintes de ressources (chemin le plus long dans un graphe pondéré dirigé acyclique), problème **difficile** sinon.

Planification de production

Données

- des **produits finaux** possibles,
- des **contraintes** de production (par exemple, quantité de matière premières disponibles),
- des **coûts** (par exemple, coût des matières premières),
- les **prix** de vente des produits finaux.

Objectif

Décider de la quantité de chaque produit final à produire de manière à maximiser le **profit** sur un horizon temporel donné.

Gestion de portfolio financier

Données

- une **somme d'argent** disponible,
- des **produits financiers** possibles,
- les **profits espérés** et **profits minimum**,
- les **limites** d'investissement.

Objectif

Décider de l'investissement sur chaque produit financier de manière à maximiser le **profit espéré** et éventuellement minimiser **le risque**.

Plan

Caractéristiques communes

- un ensemble S **d'alternatives** : **l'espace de recherche** ;
- un ensemble Co de **contraintes** à satisfaire (représentant soit des **limitations physiques** soit des **prérequis** de l'utilisateur) ;
  - une contrainte $c \in Co$ est une fonction de S dans $\mathbb{B}$;
  - une **solution** est une alternative satisfaisant toutes les contraintes de Co ;
- un ensemble Cr de **critères** à satisfaire du mieux possible (représentant des **préférences** de l'utilisateur) ;
  - un critère $c \in Cr$ est une fonction de S dans n'importe quel ensemble totalement ordonné qui doit être minimisé ou maximisé ;
- une **requête** classique : produire une solution satisfaisant les critères au mieux ;
  - dans le cas d'un **critère unique** c, produire une **solution optimale** (i.e. solution qui minimise ou maximise c) ;
  - dans le cas de **plusieurs critères**, **agréger** les critères pour se ramener au premier cas (problème difficile).

Cadre de représentation classique

Ensemble V de variables

- On associe à chaque variable v de V un **domaine** de valeurs d_v ;
- Une alternative est une **instanciation** de chaque variable v par une des valeurs de son domaine d_v.
- L'espace de recherche S est donc $\Pi_{v \in V} d_v$

Exemple

- $V = \{x, y, z\}$;
- $x, y, z \in \mathbb{N}$;
- $0 \leq x, y, z \leq 10$.

Cadre de représentation classique

Contrainte

Equation logique ou arithmétique sur un sous-ensemble de V

Exemple

$(x \geq y + 1) \vee (y \geq x + 2)$

Critère

Equation logique ou arithmétique sur un sous-ensemble de V

Exemple

Minimiser $max(x + 2, y + 1, z + 3)$

$\rightarrow$ Représentation **compacte**, non énumérative.

Exemple illustratif : le problème du sac-à-dos (Knapsack Problem)

Données

- un ensemble O d'**objets** à placer dans le sac-à-dos ;
- un ensemble D de **dimensions** à considérer (poids, volume, etc.) ;
- pour chaque dimension $d \in D$, une **capacité** maximum Ca_d ;
- pour chaque objet $o \in O$ et chaque dimension $d \in D$ une **consommation** $Co_{o,d}$;
- pour chaque objet $o \in O$, une **valeur** V_o représentant son importance.

Objectif

Décider **quels objets placer** dans le sac-à-dos de manière à maximiser la **somme des valeurs** des objets placés **en respectant les capacités**.

Exemple illustratif : le problème du sac-à-dos (Knapsack Problem)

Variables

$\forall o \in O : p_o \in \{0, 1\}$

Contraintes

$\forall d \in D : \sum_{o \in O} p_o \cdot Co_{o,d} \leq Ca_d$

Critère

$maximize \sum_{o \in O} p_o \cdot V_o$

Plusieurs cas

- **optimisation pure** : pas de contrainte ;
- **satisfaction pure** : pas de critère ;
- **optimisation sous contrainte** : présence de contraintes de critère (cas classique).

- **optimisation continue** : espace de recherche continu ;
- **optimisation discrète** : espace de recherche discret ;
- **optimisation dans des domaines finis** : espace de recherche fini.

Solution analytique

Solution analytique

- Dans le cas général, il n'existe pas de **solution analytique**, i.e. une expression donnant la(les) solution(s) optimale(s) en fonction des paramètres du problème.

Exemples

- Pour les polynômes de degré 3, utilisation des méthodes de Cardan
- Pour les polynômes de degré 2, calcul des racines à l'aide du déterminant
- Équation différentielle linéaire d'ordre 1

Difficultés

Difficultés

- S'il n'existe pas de solution analytique, il est possible d'utiliser la **puissance de calcul machine** pour calculer des solutions optimales ou approchées.
- Même si l'espace de recherche est **fini**, il peut être **immense**.
- L'**énumération** des alternatives (pour les vérifier, les comparer et trouver la meilleure) peut être impossible en pratique.
- Parfois, trouver **une solution** peut être un défi.

Exemple : le problème du voyageur de commerce (Traveling Salesman Problem - TSP)

Données

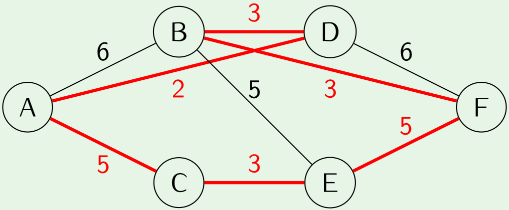
- un ensemble de **villes** qui doivent être visitées ;
- une matrice de distance entre les villes ;

Objectif

Trouver un **circuit** de **longueur minimum** qui part d'une ville v, finit dans cette ville et passe par toutes les autres villes une et une seule fois (**circuit hamiltonien**).

Exemple : le problème du voyageur de commerce (Traveling Salesman Problem - TSP)

Exemple



Espace de recherche

- Un **circuit** peut être vu comme une **permutation** sur l'ensemble des villes ;
- Taille de l'**espace de recherche** s'il y a n villes $= n!$

Temps de calcul si énumération

Hypothèse optimiste

10^{-12} secondes pour produire et évaluer un circuit.

Temps de calcul

Nombre de villes	Temps de calcul (en secondes)
10	$3.63 \cdot 10^{-6}$
15	1.31
20	2432902
30	265252859812191058636

- 2432902 sec. $=$ 0.77 année.
- 265252859812191058636 sec. $=$ 84111130077 millénaires $=$ plus de 500 fois **l'âge de l'univers** !

$\Rightarrow$ **Explosion combinatoire**

Défi

Être capable de produire des solutions optimales ou approchées **sans explorer explicitement l'espace de recherche entier**.

Plusieurs approches

Approche possible

- développer un **algorithme d'optimisation spécifique** pour chaque problème d'optimisation concret

$\rightarrow$ algorithmes efficaces pour la résolution

$\rightarrow$ coûteux en terme de développement

Une autre approche

- existence de **cadres de modélisation génériques**
- existence d'**algorithmes de résolution génériques** dédiés
- modélisation de chaque problème concret dans un **cadre adapté**
- utilisation d'un **algorithme dédié**, éventuellement paramétré pour traiter le problème concret au mieux.

Plusieurs approches

Dernière approche

- utilisation d'outils dédiés :
  - décomposition du problème en sous-problèmes plus faciles à résoudre
  - appels à plusieurs outils en parallèle ou en séquence
  - utilisation/adaptation de bibliothèques d'algorithmes existantes
  - définition d'un algorithme spécifique exploitant les caractéristiques du problème

Cadres de modélisation considérés

	Continu	Discret
Linéaire	Programmation linéaire (PL)	Programmation linéaire en nombres entiers (PLNE)
Non linéaire	Programmation non linéaire (PNL)	Programmation par contrainte (PPC)

Graphes

Méthodes de résolution dédiées

	Continu	Discret
Linéaire	simplexe Programmation linéaire (PL)	recherche arborescente + simplexe Programmation linéaire en nombres entiers (PLNE)
Non linéaire	Programmation non linéaire (PNL) recherche à base de gradient	Programmation par contrainte (PPC) recherche arborescente + propagation de contraintes recherche locale

Graphes

Plan

Programme du module

1 Introduction (Cours)

2-3 Complexité (Cours)

4-9 Programmation Linéaire en Nombre Entiers (Cours + PC)

10-12 1er **BE noté**

13-17 Programmation par contraintes (Cours + PC)

18-20 2eme **BE noté**

21-22 Examen