Polycopie du module logistique de distribution
Filière: SCM3

École Nationale des Sciences Appliquées de Tétouan

Abdellah El Fallahi

La première version de ce cours date de: Septembre 2015

La dernière modification date de:

2 décembre 2024

Table des matières

Chapitre 1

Introduction générale

La logistique de distribution, vise à optimiser l'acheminement des biens depuis un ou plusieurs entrepôts vers les clients finaux. Cette optimisation est basée sur l'utilisation des nouvelles techniques de la recherche proportionnelle, l'informatique et le management. En fait, plusieurs méthodes d'optimisation ont été développées par la communauté scientifique dans le domaine de la logistique de distribution. Ces méthodes peuvent aller des méthodes simples aux méthodes très avancées basées sur les nouvelles découvertes en mathématiques, informatique et l'intelligence artificielle. Généralement, ces méthodes prennent en considérations toutes les conditions de livraison qui doivent être convenu dans le contrat commercial.

L'étude de la logistique de distribution s'intéresse à la fois à la circulation des flux physiques à travers le réseau de distribution (gestion des transports, gestion des stocks ...), mais aussi à la gestion des infrastructures logistiques qui composent ce réseau (implantations, gestion d'entrepôt ...).

A1- Les caractéristiques de la logistique de distribution.

La logistique de distribution a pour finalité, la gestion des flux de marchandises. En effet, on vise à accomplir, dans les meilleures conditions économiques et les meilleurs délais, la livraison des commandes clients. Elle se traduit par l'organisation et la réalisation des acheminements des marchandises depuis les lieux de prélèvement chez les fournisseurs qui peuvent être une fabricants, un entrepôt ou toute autre infrastructure jusqu'au le client final. La complexité de la logistique de distribution a imposé la répartition des opérations de distribution ont plusieurs axes pour faciliter son management. Généralement, les activités de la logistique de distribution sont répartis en trois axes principaux à savoir :

— La détermination des réseaux de distribution qui cherche à :

1. Fixer le choix des infrastructures de stockage et de transbordement (entrepôts, plate forme logistique, lieux de transbordement , ports, aéroport, ...)
2. Choisir les moyens de transport à utiliser : bateaux, camions, avions ....
3. Déterminer les itinéraires de distribution entre les fournisseurs et les clients
4. Satisfaire les demandes des clients à moindre coût

— Gestion des flux de transport

1. Emballage et colisage de la marchandise,
2. Remplissage des véhicules : consolidation de la marchandise, fragmentation des lots volumineux, ...
3. Détermination des moyens de chargement et déchargement des camions : chariots, crues, ...
4. Organisation des tournées de véhicules,
5. Gestion du retour des moyens de transport vides
6. ...

— Management des stocks et des plates-formes logistique sur la totalité du réseau de distribution : localisations, configurations, détermination des ressources, ...

A2- Les enjeux de la logistique de distribution

Étant donnée la complexité des réseaux de distributions contemporains, les enjeux principaux de la logistique de distribution peuvent être résumés par les points suivants :

— La globalisation des chaînes logistique a fait exploser le nombre des intervenants dans ces chaînes. Et par la suite, un des points crucial et le management de la multiplicité des intervenants, qui impose une bonne organisation de la circulation des informations et des marchandises (maîtrise des flux documentaires, planification des opérations physiques à travers le réseau de distribution ...) ;

— La diversification des itinéraires empruntés par les marchandises exige un management de la Multi-modalité des opérations de transport. L'éventualité d'utiliser différents modes de transport successifs pour les acheminements exige, selon la nature et taille des colis, de faire le bon choix des emballages, des UTI (Unités de Transport Intermodal) et de prévoir que les moyens de manutention adéquats sont disponibles à chaque point de transbordement ;

— La concurrence farouche entre les différentes chaînes logistiques pousse celles-ci à respecter minutieusement les cahiers des charges clients. Les produis doivent être livrés en quantité et en qualité demandé, dans les délais impartis. Il faut par conséquent adopter les bonnes pratiques (moins de ruptures des charges, gestion anticipé de certaines formalités administratives, Inter modalité et accélération des temps de transbordement ... ;

— La souci de la marge bénéficiaire impose aux chaînes logistiques une bonne maîtrise des coûts logistiques. Pour ce faire, une réduction des parcours s'impose pour faire moins de Km, Il faut opérer une bonne détermination des routes, bien organiser les tournées, réduire le nombre de retours à vides des camions en leur proposant un fret de retour ; bon choix des prestataires (les prestations achetées doivent correspondre aux besoins) ; meilleure combinaison de moyens ; meilleur taux de remplissage des véhicules ; recours aux stratégies logistiques collaboratives (GPA, GMA, Cross-docking ...) ; optimisation des coûts des derniers Km ; recours au DRIVE ;

— Maitrise des risques liés à l'acheminement (risques de transport, manutention et entreposage). Il convient de réduire le nombre de rupture de charge lors de l'acheminement, de bien protéger les marchandises et de respecter les conditions de transport pour les denrées périssables. Moins de manipulations engendrent moins de risques et par ailleurs, des coûts d'assurances maîtrisés ;
— Logistique des retours. Organisation de la collecte et du le retour des emballages vides ...

A3- Les contraintes de la logistique de distribution

Vu la disparités des produits à transporter et les différentes connexions dans une chaîne logistique qui est devenu de plus en plus globalisée, la logistique de distribution est contrainte de faire face à plusieurs contraintes. Ces contraintes peuvent être des contraintes liées aux bien transportés, des contraintes de réglementation que se soient des produits ou des règles de chaque pays, des contraintes géographiques ou bien techniques.

— Contraintes liées aux marchandises. A chaque type de marchandise il faut prendre une série de mesures appropriées pour éviter de toute type d'avarie pouvant découler des propriétés de la marchandise.
  1. denrées alimentaires,
  2. marchandises périssables,
  3. marchandises dangereuses.
— Contraintes réglementaires.

1. Obligations documentaires liées à la nature des produits à distribuer (licences, certificats d'origine, certificats de circulation ...) ;
2. obligations documentaires liés au type d'expédition (documents à produire suivant le mode de transport) ;
3. réglementation applicable en cas de litiges ;

— Contraintes géographiques. Le climat, l'environnement socioculturel, et tout simplement la météo peuvent amener à reconsidérer certains choix du logisticien ;
— Contraintes techniques.

1. Manque d'infrastructures,
2. Absence des moyens de manutention adéquats dans les points de transbordement et au lieu de déchargement

Toutes ces contraintes peuvent impacter le choix de la politique de distribution et auront un effet sur l'optimisation des opérations de distribution. Vu l'importance de ces contraintes alors ils doivent être prises en considération par les logisticiens afin d'éviter tout type de problème qui peut entraver le bon fonctionnement de la chaîne de distribution.

B- Les missions du logisticien de distribution

En s'appuyant sur tout ce qu'a vu avant, le logisticien de distribution et un professionnel de logistique et transport, qui pour mission la maîtrise de la chaîne logistique de distribution. Il organise l'acheminement des marchandises et planifie les déroulements des opérations au niveau de chaque maillon de la chaîne de distribution. Le logisticien de distribution est chargé donc de :

— L'organisation des livraisons des commandes clients de porte à porte (door to door), du fabricant au distributeur (business to business) ou du distributeur au consommateur (business to customer) et inversement ;
— La détermination du réseau de distribution (choix des entrepôts de prélèvement, choix de moyens de transports, choix des itinéraires de transport, organisation des opérations de transport multimodale ...) ;
— La sécurité des colis acheminés (choix des emballages adéquats pour protéger les marchandises au cours du transport, surveillance des opérations de chargement, déchargement et arrimage des colis sur les véhicules de transport ...) ;
— L'organisation des opérations de transport et de livraison, avec le souci d'assurer une utilisation optimale des véhicules de transport (meilleur taux de remplissage, meilleur ordonnancement des trajets, moins de Km de parcours ...) ;

— La mise en œuvre des stratégies logistiques collaboratrices (GPA ; GMA, Cross-dockinq, CFPR ...), afin d'optimiser l'emploi des ressources de distribution et de réduire par la même occasion les coûts logistiques ;
— du choix des prestataires logistiques (transporteurs, transitaires). en général, le logisticien de distribution conclut un accord commercial avec ces prestataires qui alors, agissant en qualité de mandataire ou de commissionnaire prennent à leur charge la réalisation de certaines opérations (emballage, transport, manutention, entreposage, déclaration en douane export ...) ;
— La planification, la validation et du déclenchement des opérations auprès des prestataires logistiques, par transmission d'un ordre de travail (ordre d'expédition, ordre de transport, ordre de transit ...) ;
— Le suivi des flux de transport et de livraison (dates de départ, dates d'arrivée, itinéraires) et de la traçabilité des marchandises. il doit par conséquent pouvoir à tout moment produire un rapport sur la situation des acheminements et la position des marchandises ;
— de la validation des coûts logistiques de distribution (contrôles des postes facturés par les prestataires et validation des coûts avant le paiement des factures par la finance) ;

Pour assurer les bonnes conditions de transport de la marchandise, le logisticien de distribution doit avoir une connaissance parfaite des caractéristiques des produits à transporter ou à livrer (poids, volume, périssables, secs, à conserver sous une température dirigée, destination ...). Les caractéristiques des produits transportés détermineront les moyens de transports adéquats (emballages, véhicules, ...) ainsi que les itinéraires de transport. Aussi, le logisticien doit avoir un œil vigilant sur l'utilisation des moyens de transport surtout en ce qui concerne le retour des véhicules qui génère généralement beaucoup de perte à cause des retours vide, et cette activité doit être intégré dans la planification de la distribution.

C- contenu du cours

Dans ce cours on va traiter les aspects suivants de la distribution :

1. Notion de complexité des problèmes d'optimisation combinatoire
2. Calcul des chemins optimaux dans un graphe
3. Étude des problèmes de localisation
4. Étude des problèmes d'acheminement
5. Étude des problèmes de tournées de véhicules (Vehicle routing problems en anglais) (VRP)
6. Étude des problèmes de remplissage

7. Pour la résolution des différents problèmes d'optimisation sujets de ce cours en utilisera les méthodes exactes et approchées de type heuristique et méta-heuristique les plus connues et utilisées dans le domaine de la distribution.

Les pré-requis de ce cours sont :

— Théorie des graphes
— Recherche opérationnelle
— Programmation mathématique
— Développement informatique : au moins la maîtrise d'un langage de programmation de préférence le C-plus plus ou java.

Chapitre 2

Complexité des problèmes combinatoire

2.1 Introduction

L'optimisation combinatoire, aussi appelée optimisation discrète, est une branche de l'optimisation en mathématiques appliquées. Elle est strictement liée à la recherche opérationnelle, l'algorithmique et la théorie de la complexité. En fait, on qualifie généralement de « combinatoires » les problèmes dont l'ensemble de solutions possibles est composés de plusieurs combinaisons des valeurs qui peuvent prendre ses variables de décision. Cette multiplication du nombre des combinaisons possibles rendent l'exploration de tout l'espace de solutions une tâche très délicate et couteuse, sinon impossible dans un temps raisonnable. C'est le cas par exemple lorsque l'on cherche à établir un plan de production en affectant une certaines tâches à des machines. Si le nombre de tâches et de machines est relativement petit donc se problème d'affectation peut être résolu facilement. Cependant, une augmentation du nombre de tâches et de machines augmente considérablement le nombre de combinaisons à tester de sorte que le temps de résolution devient excessivement long ou mieux dire est corrélé exponentiellement avec la taille du problème à traiter.

On se basant sur la complexité des algorithmes de résolution des problèmes combinatoires, on peut les classer en plusieurs classes selon la complexité de l'algorithme capable de les résoudre. Les classes aux quelles appartiennent l'ensemble des problèmes d'optimisation combinatoire ils sont basées soient sur la notion du temps d'exécution de l'algorithme ou bien d'espace.

2.1.1 Les quatre familles de classes de complexité en temps et en espace

Suivant qu'il s'agit de temps et d'espace, de machine déterministes ou non déterministes, on distingue quatre familles principales de classes de complexité :

- **TIME(t(n))** : La classe des problèmes de décision qui peuvent être résolus en temps de l'ordre de grandeur de t(n) sur une machine déterministe.
- **NTIME(t(n))** : La classe des problèmes de décision qui peuvent être résolus en temps de l'ordre de grandeur de t(n) sur une machine non déterministe.
- **SPACE(s(n))** : La classe des problèmes de décision qui requièrent pour être résolus un espace de l'ordre de grandeur de s(n) sur une machine déterministe.
- **NSPACE(s(n))** : La classe des problèmes de décision qui requièrent pour être résolus un espace de l'ordre de grandeur de s(n) sur une machine non déterministe.

Si on prend en considération le temps comme critère on peut distinguer les classes de complexité suivants :

- La classe $\mathcal{P}$: cette classe contient les problèmes dites polynomiaux , i.e, les problèmes qu'on peut résoudre en utilisant un algorithme polynomial, c'est-dire d'ordre $O(n), O(n^2)$...
- la classe $\mathcal{NP}$ contient l'ensemble des problèmes polynomiaux non déterministes, i.e., pouvant être résolus par un algorithme de complexité polynomiale pour une machine non déterministe. Intuitivement, cela signifie que la résolution des problèmes de $\mathcal{NP}$ peut nécessiter l'exploration d'un grand nombre (éventuellement exponentiel) de solutions, mais que l'examen de chaque cas doit pouvoir être fait en temps polynomial.
- EXPTIME, la classe des problèmes décidés en temps exponentiel par une machine déterministe ;
- NEXPTIME, la classe des problèmes décidés en temps exponentiel par une machine non-déterministe.

Si le critère à prendre en considération c'est l'espace alors on peut distinguer les classes suivantes :

- L, la classe des problèmes qui peuvent être résolus en espace logarithmique sur une machine déterministe ;
- NL, la classe des problèmes qui peuvent être résolus en espace logarithmique sur une machine non-déterministe ;
- PSPACE, la classe des problèmes qui peuvent être résolus en espace polynomial sur une machine déterministe. En fait cette classe est égale à NPSPACE d'après le théorème de Savitch ;
- EXPSPACE, la classe des problèmes qui peuvent être résolus en es-

pace exponentiel. Cette classe est égale à NEXPSPACE d'après le théorème de Savitch.

Inclusions des classes

il est très bien connu dans la littérature qu'on a les inclusions suivantes

$$L \subseteq NL \subseteq NC \subseteq P \subseteq NP \subseteq PSPACE = NPSPACE \subseteq EXPTIME \subseteq NEXPTIME \subseteq EXPSPACE = NEXPSPACE \quad (2.1)$$

Problèmes C-complets ou C-difficiles

Soit C une classe de complexité (comme P, NP, PSPACE, etc.). On dit qu'un problème est C-difficile si ce problème est au moins aussi difficile que tous les problèmes dans C. Un problème C-difficile qui appartient à C est dit C-complet.

Formellement, on définit une notion de réduction : soient Π et Π' deux problèmes ; une réduction de Π' à Π est un algorithme (ou une machine) d'une complexité donnée (qu'on sait être inférieure à celle de la classe) C transformant toute instance de Π' en une instance de Π. Ainsi, si l'on a un algorithme pour résoudre Π, on sait aussi résoudre Π'. On peut donc dire que Π est au moins aussi difficile à résoudre queΠ', à la complexité de la réduction près.

Π est alors C-difficile si pour tout problème Π' de C, Π' se réduit à Π. La notion de C-difficile peut varier selon le type de complexité que l'on autorise pour la réduction. Dans beaucoup de cas on s'intéresse aux réductions polynomiales, c'est-à-dire celle demandant uniquement un espace et un temps polynomial pour être effectuée. Mais on peut également s'intéresser à d'autres types de réductions comme celles utilisant un espace logarithmique, afin d'étudier par exemple le lien entre les classes P et L.

La relation de réduction étant réflexive et transitive, elle définit un préordre sur les problèmes. Ainsi, on la note usuellement avec le symbole $\leq$: on a $\Pi' \leq \Pi$ si Π' se réduit à Π. On peut apposer au symbole une lettre précisant le type de réduction que l'on s'autorise : par exemple $\Pi \leq_P \Pi'$ signifie habituellement qu'il existe une réduction polynomiale de Π vers Π'. Les problèmes C-difficiles sont les majorants de C et les problèmes C-complets sont les plus grands éléments de C.

Les problèmes $\mathcal{NP}$-complets sont un cas particulier important de ce concept. De manière standard, ils sont définis en autorisant uniquement des réductions polynomiales, c'est-à-dire que l'algorithme qui calcule le passage d'une instance de Π' à une instance de π est polynomial. Le théorème de Cook-Levin, dû à Stephen Cook et à Leonid Levin, énonce qu'il existe un problème NP-complet. Plus précisément, Cook a montré que le problème SAT est NP-complet. On a par la suite établi la NP-complétude de nombreux autres problèmes.

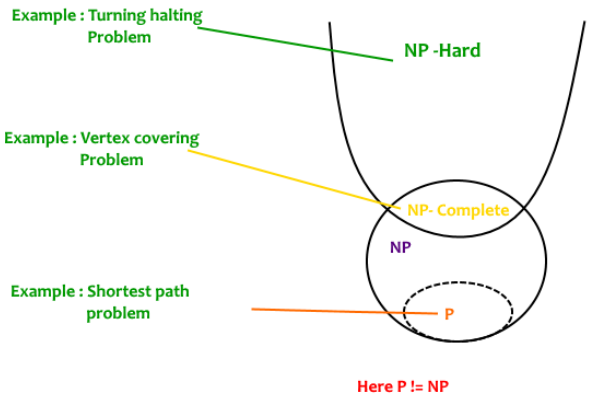


FIGURE 2.1 – Les classes de la complexité/ temps

Chapitre 3

Problème de localisation des entrepôts

Objectifs pédagogiques :
les objectifs pédagogiques de cette parti du cours sont :

- Connaître l'importance de la localisation des installations logistiques : entrepôts
- Étude et analyse des différents problèmes de localisation des entrepôts
- Modélisation des problèmes de localisation : programmation linéaire
- Méthodes de résolutions des problèmes de localisation : méthodes exactes et approchées
- Notions de couverture : couverture totale et maximale

3.1 Importance de la localisation

Les chaînes logistique sont désormais convaincues que leur performance commerciale et l'acquisition de nouveaux clients dépend principalement de leur capacité à maîtriser et optimiser l'ensemble des opérations liées au transport. Les entrepôts et certainement le maillon le plus important d'un réseau de distribution. En effet, un réseau de transport bien conçu, impliquera une économie importante des ressources. La localisation des entrepôts est sans doute la clé pour la réussite d'un réseau de distribution. Par conséquent, tout transporteur est amené à bien prendre ces décisions de localisation qui sont de plus en plus compliquées tenant compte des contraintes suivantes :

- Les marchés sont plus étendus, mais souvent avec une clientèle dispersée,
- Les livraisons à distance sont plus faciles, grâce à l'ouverture des frontières,
- Les sites possibles sont donc plus nombreux

La localisation est un problème complexe, crucial à cause des enjeux financiers, et avec des compromis à trouver entre plusieurs critères à ma-

nipuler. Ces critères sont liés généralement aux différents coûts associés au management de l'entrepôt en soi-même et aussi aux coûts de transport depuis et vers les entrepôts. Parmi les critères les plus important on citera à titre d'exemple :

Exemple de choix :

- Installation dans un pays à bas coût de main d'œuvre (Low cost contries LCC) :
- augmentation des coûts d'expédition si on est loin des zones de consommation
- Manque de main d'œuvre qualifiée
- Installation près des consommateurs ou des fournisseurs
- Augmentation des coûts d'approvisionnement entre les fournisseur et les plates formes logistiques.
- Augmentation des coûts de distributions entre les plates formes logistiques et les clients de celles-ci.
- Les risques associés au rupture de la chaine logistique d'un produit en cas des aléas
- ...

Classification et exemples :

objectifs : trouver l'emplacement optimal d'installations (usines etc.) sur un ensemble de sites possibles, à fin de :

- Minimiser le nombre d'installations pour couvrir les demandes des clients,
- Maximiser la demande couverte avec un nombre fixe d'installations,
- Minimiser les coûts (construction, fonctionnement, charges, coûts de livraison aux clients, etc).
- Réduire l'ampleur de la rupture de la chaîne logistique en cas d'aléas

Problèmes à résoudre :Parmi les questions les plus fréquentes dans le domaine de la localisation dans toute sa généralité on peut mentionner :

- Combien d'installations faut-il placer ?
- Où doit-on placer chaque installation ?
- Comment affecter les demandes aux sites créés ?
- Quelle taille pour un site ou une plate forme logistique ?
- ...

Les problèmes de localisation peuvent touchés tous les domaines socio-économique. La réussite d'un projet de mise en place d'un site ou entrepôt est une décision stratégique pour l'entreprise.

Exemples de problèmes de localisation :

— **Problèmes associés :**
— Transports (placement d'entrepôts, de hubs),
— Placement d'unités de production,
— Logistique de secours (interventions, ambulances),
— Télécommunications (relais TV, BTS),
— Localisation des centres hospitaliers,
— Localisation des écoles,
— Les centres de polices,
— ...

Critères de classification en localisation

— Localisation planaire, discrète, et sur un réseau
— Localisation continue : les sites peuvent être partout dans le plan (relais de TV).
— Discrète : nombre fini de sites, et on connaît la matrice des "distances" au sens large entre sites ou
— distancier : en km, en temps, en coût, etc.
— Sur un réseau : localisation discrète sur les nœuds d'un réseau
— Type de graphe : problèmes un peu plus faciles sur certains types de réseau. Exemple : stations de pompage sur un arbre de pipelines.
— Nombre de sites à localiser. Peut être une variable du de décision ou une donnée.
— Le cas particulier d'un seul site est facile (méthodes du barycentre).
— Problèmes statiques ou dynamiques.
— Pour les problèmes dynamiques :
— Les paramètres et les variables évoluent dans le temps.

Les types des problèmes de localisation :

— Problèmes déterministes ou probabilistes.
— Paramètres connus (demande connues, ...)
— Paramètres variables dans le cas probabiliste (modèles probabilistes, plus complexes).
— Problèmes mono ou multi-produits. Quand la nature des produits n'est pas importante (transport en vrac), on se ramène à un produit fictif pour simplifier : palettes, m^3 de liquide ...
— Secteur privé ou public.
— Le privé maximise le profit ou minimise le coût, et la même entité paie les coûts d'installation puis de fonctionnement.
— Secteur public : facilité pour l'acquisition du terrain

Public : notion de service plus importante. L'entité ne paie souvent que l'installation ou le fonctionnement.
Objectifs simples ou multiples. Si objectifs antagonistes, il faut souvent une approche d'optimisation multi-critère. Capacité finie ou infinie. La capacité d'un site est le plus souvent limitée. Parfois elle est « infinie » (= suffisante) : un relais de TV couvre tous les habitants de sa zone. Affectation des demandes. Chaque demande est traitée par un site (assignment) ou par plusieurs (allocation). Second cas fréquent si sites de capacité finie Problèmes à un ou plusieurs niveaux. Exemple : réseaux de distribution à 1 étage (usines →clients) ou 2 (usines →entrepôts (hub)→clients). Installations indésirables. Placement des installations près des clients, sauf les "indésirables« . Décharge loin d'une ville →coût de transport élevé d'où choix entre le coût de transport et nb de personnes affectées par la décharge La recherche d'un compromis entre plusieurs objectifs Optimisation multi-objective

Interactions entre sites :

Deux sites intéressants pour un hypermarché peuvent desservir des zones en commun :

- ouvrir les 2 sites en même temps n'est pas souhaitable,
- Problème du choix entre les deux sites,
- Les interactions entre sites voisins compliquent beaucoup les problèmes de décision en localisation,
- Exemple pour les BTS en télécommunication (problème d'interférence)

Chapitre 4

Localisation des entrepôts

4.1 Introduction

La détermination de l'emplacement des entrepôts est un des aspects primordiaux à prendre en compte pour assurer l'efficience de la chaîne logistique d'un produit donné. En effet, l'emplacement d'un entrepôt est un facteur qui influe non seulement sur la construction elle-même, mais surtout sur l'aspect stratégique et qui peut être décisif pour la réussite ou l'échec de la chaîne logistique. En effet, une bonne localisation d'un entrepôt permettra la bonne exécution de la fonction logistique d'un entrepôt qui peut être résumée comme suit :

- Réception de tous les produits compris dans l'activité industrielle de l'entreprise propriétaire de l'entrepôt,
- Exécution d'un contrôle de qualité immédiat,
- Contrôle et inventaire des produits stockés,
- Stockage approprié des marchandises,
- Préparation des commandes destinées aux entrepôts régionaux et/ou aux clients,
- Expédition rapide des commandes.

La localisation d'un entrepôt est une décision stratégique dans chaîne logistique, et par la suite la détermination de l'emplacement d'un entrepôt doit être faite avec beaucoup d'attention. Une bonne localisation de l'entrepôt garantira son bon fonctionnement. Le problème de localisation d'entrepôt consiste à :

- Trouver le nombre d'entrepôts nécessaire pour répondre aux besoins des clients,
- Détermination de l'emplacement adéquat de chaque entrepôt,
- Détermination de la capacité de chaque entrepôt,

Pour une bonne localisation des entrepôts il va falloir prendre en considération les paramètres suivants :

1. Le nombre de centre de distribution à installer,

2. Le mode d'alimentation des entrepôts,
3. Les coûts unitaire de transport en amont et en aval d'un entrepôt,
4. Les itinéraires d'acheminement,
5. La quantité de la marchandises à entreposer,
6. Le nombre de voyage par unité de temps.

L'objectif de la localisation est la bonne détermination des lieux d'installation des entrepôts qui permettra l'optimisation de plusieurs critères. Parmi les critères les plus important on trouvera :

— Minimisation des coûts d'installation,
— Minimisation des coûts de transport de la marchandise en amont et en aval,
— Minimisation des coûts de transport entre l'entrepôt et les clients finaux,
— Optimisation de la main d'œuvre.

4.2 Modélisation des problèmes de localisation

La modélisation du problème de localisation consiste à donner un modèle mathématique, généralement linéaire, qui prendra en considération les objectifs ainsi que les contraintes liées au problème à modéliser. En ce qui suit on donnera le modèle mathématiques classique utilisé en localisation des entrepôts.

4.2.1 Paramètres et variables de décision du modèle

les variables et paramètres du modèle peuvent être présentées comme suit :

— I - ensemble des fournisseurs, avec l'indice i
— J- ensemble des entrepôts potentiels, avec l'indice j
— K - ensemble des clients, avec l'indice k
— p_i - offre du fournisseur i
— d_k - demande du client k
— cs_{ij} - coût de transport d'une unité du produit du fournisseur i à l'entrepôt j
— cc_{jk} - coût de transport d'une unité du produit de l'entrepôt j au client k
— F_j - coût fixe pour créer l'entrepôt j
— x_{ij} - variable représentant le flux du fournisseur i à l'entrepôt j
— y_{jk} - variable représentant le flux de l'entrepôt j au client k
— z_j - variable binaire indiquant la création ou non de l'entrepôt j, zj =1 si l'entrepôt j est créé et zj = 0 sinon

— s_j - capacité de l'entrepôt j

Dans un but de minimiser les coûts d'installation ainsi que les coûts de transport en amont et en aval d'un entrepôt en considérant les contraintes classiques de demandes et de capacité la formulation mathématique est la suivante :

$$obj = \min \sum_{j \in J} F_j Z_j + \sum_{i \in I} \sum_{j \in J} C s_{ij} x_{ij} + \sum_{j \in J} \sum_{k \in K} c c_{jk} y_{jk} \tag{4.1}$$

$$\sum_{i \in I} x_{ij} = \sum_{k \in K} y_{jk} \qquad \forall j \in J \tag{4.2}$$

$$\sum_{j \in J} x_{ij} \leq p_i \qquad \forall i \in I \tag{4.3}$$

$$\sum_{j \in J} = d_k \qquad \forall k \in K \tag{4.4}$$

$$\sum_{k \in K} y_{jk} \leq S_j \qquad \forall j \in J \tag{4.5}$$

$$x_{ij} \leq p_i Z_j \qquad \forall i \in I \& \forall j \in J \tag{4.6}$$

$$y_{jk} \leq d_k Z_j \qquad \forall j \in J \& \forall k \in K \tag{4.7}$$

$$x_{ij} \geq 0, y_{jk} \geq 0, Z_j \in \{0,1\} \qquad \forall i \in I, \forall j \in J \& k \in K \tag{4.8}$$

Pour résoudre ce problème on peut utiliser toutes les méthodes d'optimisation. Avant de résoudre il faut spécifier le type de localisation désirée soit continue ou bien discrète.

4.3 Localisation continue

dans le cas de la localisation continue les emplacements des entrepôts peuvent être localiser à n'importe quel point de l'espace. Dans la cas de la localisation discrètes les emplacements potentiels sont données à l'avance sur un réseaux par exemple. pour résoudre un problème de localisation continue la méthode la plus connu est celle connue sous le nom de la méthode du barycentre. Cette méthode cherche à localiser un entrepôt en se basant sur les coordonnées de clients à servir ainsi que les demandes des clients.

4.3.1 la méthode du barycentre

La méthode du barycentre à pour objectif la détermination du centre de gravité d'un ensemble de clients : On a besoin des coordonnées des clients pour chaque client i ces coordonnées sont (x_i,y_i) et le poids de chaque client est donné par Q_i l'emplacement de l'entrepôt ou bien le barycentre des clients

est :

$$X_G = \frac{\sum_i Q_i \times X_i}{\sum_i Q_i} \qquad\qquad Y_G = \frac{\sum_i Q_i \times Y_i}{\sum_i Q_i}$$

Exercice d'application une entreprise de distribution cherche à localiser l'emplacement optimal pour servir huit clients dont les données sont données par le tableau suivant :

TABLE 4.1 – DataExercice

clients	C1	C2	c3	C4	C5	C6
cordonnées	3 ;5	2 ;5	1 ;5	2 ;2	2 ;6	2 ;7
demandes	15	13	12	7	16	33

Résolution : par application directe des formules précédentes on trouve :

$$X_G = \frac{15*3+13*2+12*1+7*2+16*2+33*2}{15+13+12+7+16+33} = \frac{195}{80} = 2.437$$

$$Y_G = \frac{15*5+13*5+12*5+7*2+16*6+33*7}{15+13+12+7+16+33} = \frac{541}{80} = 6.76$$

4.4 Localisation discrète

dans ce deuxième cas on sait à l'avance les emplacements potentiels des entrepôts. Alors, pour résoudre le problème dans ce cas on va utiliser la méthodes nommée Scores charge-distance (load-distance scores).

4.4.1 Principe de la méthode

les données nécessaires pour l'application de la méthode sont :
— m sites possibles pour une nouvelle installation,
— n entités (clients, fournisseurs, etc.) à desservir,
— mesure simple de « charge » Lj pour chaque entité j. A définir :

1. tonnage entrant ou sortant,
2. nombre de déplacements/semaine,
3. patients venant consulter, etc.

— distances dij entre tout site i et toute entité j.

Objectif : choisir un site i* minimisant un score estimant le travail de déplacement des charges.

4.4.2 Démarche de la méthode

— calculer pour tout site i candidat pour abriter un entrepôt le score charge-distance

$$ld(i) = \sum_{j=1,n} L_j \times d_{ij}$$

— choisir le i^*

$$ld(i^*) = \min\{ld(i), i \in 1, ..., m\}$$

Exemple : Localisation d'un dispensaire dans une ville.

— entités : quartiers.
— charges : nombre d'habitants (patients potentiels).
— distances : du centre des secteurs aux sites possibles.

Exercice une ville souhaite installer un nouveau dispensaire pour desservir 7 quartiers dont les informations sont présentées par la tableau 4.3 coordonnées (x,y) représentent les cordonnées cartésiennes de chaque quartier et la population donne le nombre d'habitants par quartiers. Pour installer ce dispensaire la ville dispose de deux lots de terrain au centre du quartiers F et C. déterminer le meilleur site en utilisant la méthode score-charge-distance. Comparer le résultat avec la solution donnée par la méthode du barycentre.
Résolution Comme on peut constater le site F ayant un score plus petit

TABLE 4.2 – données du problème de localisation

quartier	Cordonnées (x,y)	Population
A	(2.5 ;4.5)	2
B	(2.5 ;2.5	5
C	(5.5 ;4.5)	10
D	(5 ;2)	7
E	(8 ;5)	10
F	(7 ;2)	20
G	(9 ;2.5)	14

que celui de C et par la suite on installera le dispensaire au quartier F.

4.4.3 La méthode de centration

la méthode de centration comme celle du score-charge-distance permet de choisir parmi plusieurs emplacements potentiels celui qui minimise la distance totale parcouru entre les clients et le centre de distribution. Soit A, B, C et D les points de livraison. Soit X le nombre de livraisons par point. On calcule pour chaque emplacement le nombre de kilomètres parcourus.

Localisation en A :

$$AB \times Xb + AC \times Xc + AD \times Xd$$

TABLE 4.3 – résolutiondu problème de localisation

quartier	Cordonnées	Population	site en C=(5.5 ;4.5)		site en F=(7 ;2)			
j	(xj ;yj)	Lj	dcj	Lj×dcj	dFj	Lj× dFj	Lj×xj	Lj×Yj
A	(2.5 ;4.5)	2	3	6	7	14	5	9
B	(2.5 ;2.5	5	5	25	5	25	12.5	12.5
C	(5.5 ;4.5)	10	0	0	4	40	55	45
D	(5 ;2)	7	3	21	2	14	35	14
E	(8 ;5)	10	3	30	4	40	80	50
F	(7 ;2)	20	4	80	0	0	140	40
G	(9 ;2.5)	14	5.5	77	2.5	35	126	35
total		68		239		168	453.5	205.5

Localisation en B :

$$BA \times Xa + BC \times Xc + BD \times Xd$$

Localisation en C :

$$CA \times Xa + CB \times Xb + CD \times Xd$$

Localisation en D :

$$DA \times Xa + DB \times Xb + DC \times Xc$$

La localisation choisie est celle dont la somme des distances pondérées parcourues est la plus faible. Pour bien comprendre le fonctionnement de cette méthode on résoudra le problème suivant. Supposant qu'on a à servir 4 clients qui constituent les sommets d'un réseau de distribution, on cherche à localiser l'entrepôt dans un des quatre points de livraison. la figure 4.1 présente le réseaux considéré.

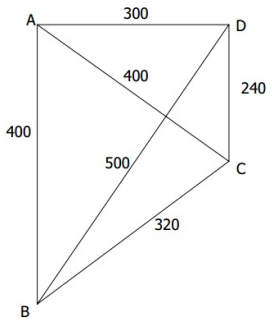


FIGURE 4.1 – Méthode de centration

les valeurs sur les arc donnent les distances entre les quartiers deux à deux. On calcule la somme de distance entre un quartiers et les autres quartiers en supposant que le nombre de voyage est un pour tous les clients, on trouve les valeurs suivantes :

- A = 400 + 400 + 300 = 1100 km
- B = 400 + 500 + 320 = 1220 km
- C = 400 + 240 +320 = 960 km
- D = 240 + 300 + 500 = 1040 km

selon la méthode de centration on va choisir le quartier ayant la distance total la plus petite, dans notre cas on va sélectionner le quartier C pour installer l'entrepôt.

4.5 Réseaux à entrepôts multiples

Le problème de localisation de plusieurs entrepôts est plus compliqué que celui d'un seul entrepôt. La complexité de ce problème réside dans le nombre de cas possible à traité et aussi les différentes contraintes liées à chaque entrepôt. 0 titre d'exemple si on considère l'exemple suivant : Exemple pour localiser 10 entrepôts sur 20 sites possibles on aura 200 000 solutions à traiter. Vu le coût des installations et le budget disponible pour l'ouverture des entrepôts on peut prendre la décision de ne pas couvrir un certains clients si cela est faisable et n'aura pas de conséquences sur la qualité de livraison.

En fait, vue la stratégie à suivre on peut distingué entre deux stratégies celle de la couverture total et maximale. Dans le cas de la couverture total on est obligé de couvrir tout le monde, par contre dans la cas de la couverture maximale on cherchera la meilleurs implantations d'un nombre fini d'entrepôts pour couvrir la demande maximale.

4.5.1 couverture totale

Pour la couverture totale il faut couvrir tous les clients. Généralement, la couverture est contrainte de la distance entre un centre et le client qu'on cherche à couvrir. Par exemple, un centre ne peut pas couvrir les clients dont la distance est supérieur à un seuil dc. les données relatives à ce problème sont les suivants :

- m points de demande ou "clients" (indexés par i)
- n sites potentiels (indexés par j)
- Dc distance de couverture (distance max de service)
- booléens $a_{ij} = 1$ ssi i est couvert par le site j (distance $\leq$ Dc) et 0 sinon.
- c_j coût d'ouverture du site j.

Objectif : déterminer les sites à ouvrir pour couvrir toutes les demandes, avec un coût total minimal.

4.5.2 Modélisation du problème de couverture toale

les variables de décision sont :

$$X_j = \begin{cases} 1 & \text{si le site j est ouvert} \\ 0 & \text{sinon.} \end{cases}$$

$$obj = \min \sum_{i=1,n} c_j x_j \tag{4.9}$$

$$\sum_{j=1,n} a_{ij} x_j \geq 1 \qquad \forall i = 1, ...m \tag{4.10}$$

$$x_j \in \{0, 1\} \qquad \forall j = 1, ..., n \tag{4.11}$$

où :
la matrice $A = (a_{ij})_{i=1,..,n;j=1,..m}$ est la matrice de couverture et x les variables de décision. Pour résoudre ce problème de couverture totale on utilisera dans un premier temps la méthode connue sous le nom techniques de réduction de la taille du problème.

principe de la méthode de réduction de taille

- soit L_i la ième ligne de A et C_j la j ème colonne de A.
- par convention, $C_k \geq C_j \Longleftrightarrow a_{ij} \geq a_{ij}$ pour tout i
- Un site k domine un site j $\Leftrightarrow F_k \leq F_j$ et $C_k \geq C_j$
- k couvre tous les clients de j sans coûter plus cher !
- Donc on peut éliminer la colonne j et on force x_j de prendre la valeur 0.

le tableau suivante donne un exemple de dominance entre le site k et j.

TABLE 4.4 – le site K domine le site j

A	1	...	k	...	j	...	n
1			0		0		
2			1		1		
3			1		0		
...			0		0		
m			1		1		

4.5.3 Techniques de réduction : client qui domine un autre client

Un client k domine un client i ssi $L_k \leq L_i$, tout site j qui couvre k ($a_{kj} = 1$) couvre aussi i. Si la contrainte (2) pour k est vérifiée, celle pour i aussi. On peut donc supprimer la ligne (contrainte) i de A.

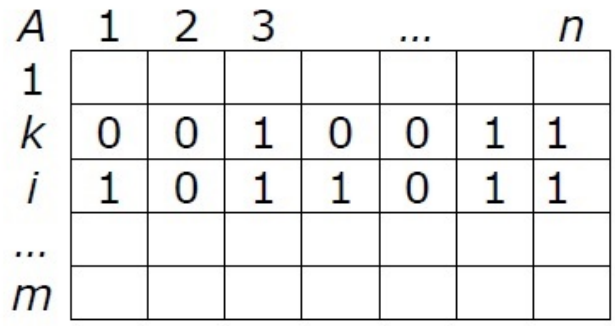


FIGURE 4.2 – notion de dominance entre deux clients

4.5.4 Techniques de réduction : Ouverture évidente

Si une ligne i de A a un seul 1, en a_{ij}, alors seul le site j couvre le client i. On peut forcer x_j à prendre la valeur 1 et supprimer toute autre contrainte k satisfaite (telle que $_{akj} = 1$). l'utilisation des techniques de réductions passe par les étapes suivantes :

- teste des lignes et colonnes en ordre quelconque
- si on trouve des réductions, on doit tout retester en effet, une réduction peut en créer d'autres
- stop quand on ne trouve plus de réductions.

Exercice On considère le graphe de la figure **??** dont on cherche à ouvrir des entrepôts pour couvrir 6 clients. Les sommets du graphe représentent les clients à couvrir et aussi les emplacement potentiels pour installer des entrepôts, les valeurs sur les arcs donnent les distances entre les sommets. On suppose aussi que les entrepôts ont le même coût d'installation.

le modèle mathématique correspondant à ce problème est donnée para 4.4

après réduction on obtient : Site A dominé par B et F par C : on force$X_A = X_F = 0$. Seul C couvre F : $X_C = 1$ et suppression des contraintes satisfaites (2),(3),(5),(6). (1) et (4), si A couvert alors D aussi : supprimer (4). Après réduction, le PL devient :

$$MinX_D + X_B + 1$$

$$X_B + X_D \geq 1$$

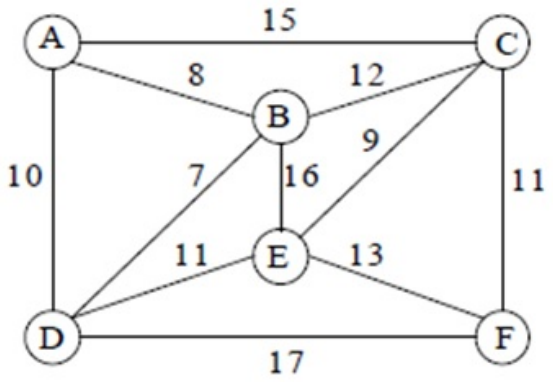


FIGURE 4.3 – problème de couverture total

$$X_B, X_D \in \{0, 1\}$$

par résolution de ce système on trouvera deux solutions à deux sites : soit $X_B = X_C = 1$ ou bien $X_C = X_D = 1$

4.6 Couverture Totale

dans plusieurs situations le problème de couverture totale peut être inadapté. Il peut avoir plusieurs inconvénients comme :

— trop de sites à ouvrir
— clients de même poids, même si demandes faibles.
— ...

Pour surmonter les inconvénients de la couverture totale on peut opter par couverture maximale qui vise à maximiser la demande totale couverte, avec un nombre limité de sites ouverts. ce problème est NP-difficile. La modélisation de la couverture totale est similaire à celle de la couverture totale on ajoutant quelques variables et restrictions. les paramètres et variables additionnels sont :

— p nombre max. de sites à ouvrir (donné),
— d_i demande du point i (donnée)
— z_i variable 0-1 indiquant si le client i est couvert ou non.

le modèle mathématique pour ce cas de figure est lst donnée para la figure **??**

Où :

— l'équation (1) : demande totale satisfaite, à maximiser.

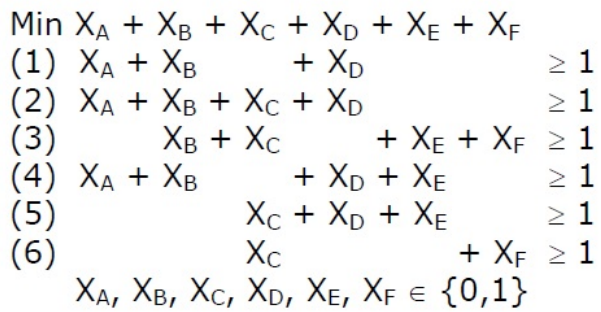


FIGURE 4.4 – model mathématique

— Contraintes (2) : Un client i est couvert si et seulement s'il exite un site ouvert qui lu couvre.
— Par définition, $Z_i = 1 \Leftrightarrow$ client i couvert.

$$Z_i = 1 \Leftrightarrow \sum_{1=1,\ldots,n} a_{ij} X_j \geq 1$$

Contrainte (3) : ouverture d'au plus p sites, on peut avoir des clients non couverts Si moins de p sites suffisent, la fonction-objectif sera égale à la demande totale

4.6.1 Résolution des problèmes de couverture

Vu que les problème de localisation son généralement de type programme linéaire en nombre entiers ou binaires donc on peut utiliser un logiciel de programmation linéaire. L'algorithme du Simplex n'est pas utilisable directement, il est nécessaire de la compléter avec un des algorithmes de coupe. On peut aussi utiliser les algorithmes de Branch & bound, branch& cut et Branch& Price. Aussi, on peut utliser les méthodes en arborescentes (xj = 0 ou xj = 1)+réductions. Ces problème deviennent très dur à résoudre lorsque la taille du problème dépasse un certain seuil. Exemple de problème traitable : 100 clients, 20 sites. **Note :** les problèmes de partitionnement (A.x = 1) sont encore plus durs car pas toujours faisables. Exemple : découpage électoral, secteurs commerciaux. Lorsque les méthodes exactes s'averent difficile à utiliser ou bien les résultats obtenus ne sont pas encouragent à ce moment on peut utiliser les méthodes heuristiques.

$$(1)\quad Max \sum_{i=1,m} d_i z_i$$

$$(2)\quad \forall i = 1 \ldots m : z_i \leq \sum_{j=1,n} a_{ij} x_j$$

$$(3)\quad \sum_{j=1,n} x_j \leq p$$

$$(4)\quad \forall j = 1 \ldots n : x_j \in \{0,1\}$$

$$(5)\quad \forall i = 1 \ldots m : z_i \in \{0,1\}$$

FIGURE 4.5 – model mathématique de la couverture maximale

4.6.2 Résolutions heuristiques des problèmes de couvertures

pour les problèmes de grande taille les méthodes exactes ne sont pas généralement applicables, pour cela les méthodes heuristiques sont alors nécessaires.Les heuristiques sont connues par leur rapide convergence vers des solutions de bonne qualité même si ne garantissent pas une solution optimale du problème traité. Parmi, les méthodes heuristiques on trouves celle appelées les heuristiques gloutonnes sont simple et efficaces :

- suite de décisions définitives (sans retours en arrière)
- choix le plus avantageux à chaque étape selon un certain critère (à définir)
- exemple, Plus Proche Voisin pour le TSP.

4.6.3 Heuristique gloutonne de Chvàtal (couverture totale)

Le principe générale de l'algorithme est l'ouverture à chaque étape du site ayant le faible coût par nouveau client non couvert.

Exemple d'application couverture totale On cherche à ouvrir un ensemble de site pour couvrir la demande de 6 quartiers A, B, C, D, E Et F. Les sites à ouvrir seront au centres de quelques quartiers clients. Le graph de la figure 4.6 donne les distance entre les sites ainsi que la demande de chaque quartiers. La demande de chaque client est donnée par d sur chaque sommet du graphe. On considère la que la distance de couverture est égale à 12 et que la capacité des entrepôts est illimitée on cherche les site à ou-

Algorithm 1 : Algo chvatal pour la couverture totale

cost =0 ; $nb_{site-ouvert}$:=0 ; $nb_{client-couvert}$:=0 ;
Ouverts = $\emptyset$
repeat
Calculer pour chaque site j non ouvert le ratio R(j)
R(j) est le ratio de F_j par le nombre de clients qui sont couverts par j et qui ne sont pas encore couverts par d'autres sites ouverts.
$R(j) = \frac{F_j}{nb-clients-couverts-par-site-j}$
sélectionner le j ayant le plus petit $R(j)$
mettre j dans Ouverts
$Cost := Cost + F_j$
$nb_{site-ouvert} := nb_{site-ouvert} + 1$;
$nb_{client-couvert} := nb_{client-couvert} + 1$;
éliminer du pb le site j et les nouveaux clients couverts
until $\{nb_{clients-couvert} = nbclient\}$

vrir pour couvrir la totalité des clients en moindre coût. On suppose que les coûts d'ouverture des sites sont A(8), B(7.5), C(10), D(11), E(13) et F(12). En appliquant l'algorithme de chvatal on trouvera :

	cost	nb_couvert	R
A	8	3	2,666666667
B	7,5	4	1,875
C	10	4	2,5
D	11	4	2,75
E	13	2	6,5
F	12	2	6

on constate que le ration le plus bas est celui de B donc on ouvre le site B, ensuite on éimine le site B et les quartiers couverts qui sont : A, B, C et D. et donc il ne reste que deux sites qui sont E et F. Puisque E ne couvre pas F et F ne couvre pas E donc on va ouvrir E et F. Et par la suite les sites à ouvrir sont B, E et F.

Exemple de couverture maximale dans cette exemple on va prendre le graphe précédent mais on considère que les coût d'ouverture des sites sont les mêmes avec une distance de couverture est égale à 11.5. le tableau de la figure 4.7 résume les calcules faits pour chaque site. On constate qu'on ouvre dans le premier lieu le site C, ensuite on élimine les sites C, E,F ; On suite on ouvre le site A ou B ou D pour couvrir le reste de la demande.

Exercice d'application On considère le problème de couverture totale défini par une matrice binaire quelconque A, mxn. On a m points de demandes (clients), n sites potentiels.

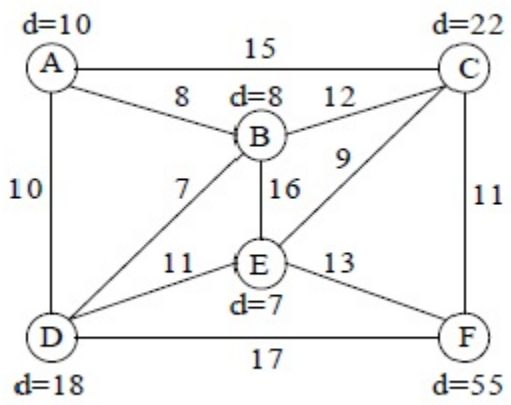


FIGURE 4.6 – notion de dominance entre deux clients

Site choisi	Clients couverts	Qté couverte si ouvert en 1er	Couverte si ouvert après C
A	A, B, D	36	36
B	A, B, D	36	36
C	C, E, F	84	déjà ouvert
D	A, B, D, E	43	36
E	C,D,E	47	18
F	C,F	77	0

FIGURE 4.7 – solution couverture maximale

Questions

- Rappeler très brièvement en termes d'ensembles la signification des lignes et colonnes de A. A quelle condition existe-t-il des solutions ?
- En supposant le problème réalisable, que signifie le fait que la somme d'une ligne i soit égale à un entier k $\leq$n ? Que la somme d'une colonne j soit égale à un entier k $\leq$m ? Que peut-on conclure si une colonne j ne contient pas de zéros ? Si toute paire de colonnes n'a aucune ligne avec deux 1 ?
- Simplifier au maximum la matrice suivante en précisant les éliminations (quelle ligne ou colonne est dominée par quelle autre). Puis résoudre optimalement le problème en précisant les sites ouverts et les sites qui couvrent chaque client. la matrice de couverture est donnée par 4.8

Algorithm 2 : Algo chvatal pour la couverture maximale

cost =0 ; $nb_{site-ouvert}$:=0 ; $nb_{client-couvert}$:=0 ;
Ouverts = $\emptyset$
$Qte_couvert := 0$
repeat
Calculer pour chaque site j non ouvert la quantité de demande qu'il peut ouvrir $Qtecouvert_j$
$Qtecouvert_j = \sum_i demande_i$ tel que le client i est couvert par j et n'est pas encore couvert
calculer le ratio $R(j) = \frac{F_j}{QteCouvert_j}$
sélectionner le j ayant le plus petit $R(j)$
mettre j dans Ouverts
$Cost := Cost + F_j$
$Qte_couvert := Qte_couvert + 1$
$nb_{site-ouvert} := nb_{site-ouvert} + 1$;
$nb_{client-couvert} := nb_{client-couvert} + 1$;
éliminer du pb le site j et les nouveaux clients couverts
until $\{nb_{clients-couvert} = nbclient$ or $nb_{site-ouvert} \geq p\}$

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

FIGURE 4.8 – Matrice de couverture A

Exercice 3 Pendant la conception de la ville de Tamasna, les responsables ont prévu qu'elle aura M quartiers. Le nombre d'enfants estimé pour chaque quartier i est donné par hi, et on a prévu l'ouverture de n écoles au maximum dans cette ville pour scolariser ses enfants (n<=m). La distance entre deux quartiers i et j est donnée par d_{ij}. Une école ouverte ne peut couvrir qu'au maximum k quartiers dont le quartier qui l'abrite. Les élèves d'un quartier doivent être affectés à la même école. Pour cette première version on considère qu'on n'a pas de contrainte concernant la distance de couverture.

Questions :

1. Modéliser le problème de localisation des écoles sous la forme d'un programme linéaire
2. On suppose maintenant que M=6, n=6 et k=3. Les autres données

du problème sont présentées dans le tableau

3. Modéliser ce problème sous la forme d'un problème de couverture totale. Résoudre ce problème en utilisant la méthode de chvatal, la distance de couverture est de 12. les données de cette exercice sont données par le tableau de la figure 4.6.3.

	Q1	Q2	Q 3	Q4	Q5	Q6
Q1	0	8	15	10	24	27
Q2		0	12	7	16	33
Q3			0	19	9	11
Q4				0	11	17
Q5					0	13
Q6						0
Nb élèves	120	135	95	80	105	115
Coût d'installation	8	7	8.5	6	9	5.5

FIGURE 4.9 – données exercice 3

Les P centres

L'objectif de ce problème linéaire est de minimiser la distance maximale entre un client et le centre auquel il est rattaché, et non plus la somme des distances. Les contraintes restent cependant les mêmes. Dans la formulation de ce problème, il n'y a alors que la fonction à minimiser qui change. La distance maximale à minimiser est

$$Z \geq \sum_j X_{ij} \times d_{ij} \forall i$$

dans ce type de problème on considère que tous les clients sont accessibles (pas de distance de couverture) mais on tient compte des distances.

- Problème des p-centres (NP-difficile)
- Fréquents en logistique de secours. Données :
  1. m clients à servir (indice i), sans distance de couverture,
  2. n sites (indice j) mais on peut ouvrir p sites au maximum,
  3. distances d_{ij} au sens large entre nœuds et sites.
- Objectif
- Placer les sites en minimisant la distance maximale aux clients.
- Exemple : placer 4 casernes de pompiers dans une ville pour minimiser le temps d'intervention sur un incendie.

Modélisation du problème de P-centres

ce problème est PL dit "mixte" : variables entières et réelles.

Variables de décision :

$$X_j = \begin{cases} 1 & \text{si le site j est ouvert} \\ 0 & \text{sinon.} \end{cases}$$

$$Y_{ij} = \begin{cases} 1 & \text{si le client i est affecté au site j} \\ 0 & \text{sinon.} \end{cases}$$

Variable réelle z : distance maximale d'intervention

modèle mathématique

$$(1) minZ \tag{4.12}$$

$$(2) \forall i \in \{1, ..., m\} : \sum_{j=1,...,n} Y_{ij} = 1 \tag{4.13}$$

$$(3) \sum_{j} X_j = P \tag{4.14}$$

$$(4) \forall i \in \{1, ..., m\} : \forall j \in \{1, ..., n\} : Y_{ij} \leq Xj \tag{4.15}$$

$$(5) \forall i \in \{1, ..., m\} : Z \geq \sum_{j=1,...,n} d_{ij} Y_{ij} \tag{4.16}$$

$$(6) \forall i \in \{1, ..., m\} : \forall j \in \{1, ..., n\} : X_j \& Y_{ij} \in \{0, 1\} \tag{4.17}$$

$$(7) Z \geq 0 \tag{4.18}$$

4.6.4 P médians

Le problème p-médian a pour objectif de minimiser la somme totale des coûts de distribution entre les clients et leur site de rattachement (le coût totale du transport). C'est alors un problème linéaire et pour formuler ce problème, il faut tout d'abord expliciter les contraintes sous forme d'équations ou inéquations. Ici sont explicitées des cas généraux. Le problème du p-median est une variante de la classe des problèmes de localisation dans le domaine de l'optimisation combinatoire. Malgré une riche littérature dans ce domaine et les nombreuses méthodes développées pour résoudre ce genre de problème, il n'existe pas de méthode pour résoudre des instances de très grandes tailles. Il est même difficile de résoudre la relaxation linéaire classique associée à ce problème. Bien entendu, plusieurs heuristiques ont été développées et donnent des résultats satisfaisants, mais sans une évaluation de la borne inférieure (nous traitons un problème de minimisation) les résultats des heuristiques qui ne se basent pas sur la résolution de la relaxation linéaire ne peuvent pas être interprétés. Une des bornes inférieures connues dans la littérature est la valeur de la solution optimale de la relaxation linéaire, bien entendu, on pourra l'améliorer avec l'ajout de contraintes valides. Une raison de plus qui rend indispensable la résolution de cette relaxation linéaire, et il est crucial de pouvoir la résoudre dans des temps raisonnables. Étant donné

un graphe dirigé G = (V,A) et des coûts c(u, v) associés à chaque arc (u, v), le problème du p-médian (pMP) consiste en la sélection de p sommets appelés des centres et l'affectation des sommets non sélectionnés aux centres tout en minimisant la somme totale de l'affectation. Ce problème est $\mathcal{NP}$-Complet même lorsque la fonction coût c est une métrique . Il est polynomial dans certaines classes de graphes comme les arbres où la fonction coût a une certaine forme. Il est aussi polynomial dans des classes de graphes définies par des structures interdites

- On veut que chaque client i soit affecté à un seul site j
- Tout client i affecté ne peut l'être que si le site j existe et est ouvert $x_{ij} \leq y_j$
- Ensuite, on vérifie l'ouverture exacte de p sites
- Les dernières contraintes sont des contraintes d'intégrité de x et y

Modèle mathématiques P-médians

$$(1)\min \sum_{i=1,..m} \sum_{j=1,..,n} q_i \times d_{ij} \times Y_{ij} \tag{4.19}$$

$$(2)\forall i \in \{1,...,m\} : \sum_{j=1,...,n} Y_{ij} = 1 \tag{4.20}$$

$$(3)\sum_{j} X_j = P \tag{4.21}$$

$$(4)\forall i \in \{1,...,m\} : \forall j \in \{1,...,n\} : Y_{ij} \leq Xj \tag{4.22}$$

$$(5)\forall i \in \{1,...,m\} : Z \geq \sum_{j=1,...,n} d_{ij} Y_{ij} \tag{4.23}$$

$$(6)\forall i \in \{1,...,m\} : \forall j \in \{1,...,n\} : X_j \& Y_{ij} \in \{0,1\} \tag{4.24}$$

$$(7)Z \geq 0 \tag{4.25}$$

Chapitre 5

Problèmes d'acheminement

L'acheminement de la marchandise désigne un problème quantitatif logistique dans lequel des produits doivent être répartis d'un ou plusieurs lieux de stockage entre plusieurs points de distribution.Parmi les problèmes d'acheminement on trouvera le transport truckload, entre des sources, avec des disponibilités données, et des destinations, avec des demandes données. le problèmes d'acheminement se modélisent par des graphe, réseaux, dont les arcs du réseau ont des coûts et éventuellement des capacités (débit maximum). La détermination des coûts des arcs est connu sous le non du problème de détermination des chemins optimaux entre deux sommets d'un graphe. On peut distingué deux catégories des problèmes d'acheminement : (i) les problèmes sans capacité et (ii)les problème avec capacité. Parmi les problème de distribution sans capacités on peut citer :

— problème de transport avec un réseau à 2 couches
— problème d'affectation dont le cas où les quantités demandées sont égales à = 1
— les problèmes de transbordement (réseau quelconque)

Pour les problème de distribution avec capacités on peut citer à titre d'exemple :

— pb du flot maximal
— pb du flot de coût minimum
— pb de multiflots (flots non miscibles)

Contrairement aux problèmes de localisation et de tournées, ces problèmes (sauf le dernier) disposent d'algorithmes rapides (polynomiaux). ils peuvent être résolus par programmation linéaire.

5.1 Chemins optimaux

Dans le cas d'un graphe orienté valué numériquement, la valeur d'un chemin est la somme des valeurs des arcs constituant le chemin. Le problème du " chemin optimum " sera donc de trouver un des chemins dont la valeur est

la plus faible.Gardons bien en tête que : l'objectif est avant tout d'obtenir ce chemin, à savoir une suite de sommets ou une suite d'arcs, et non pas simplement de calculer la valeur la plus faible qui peut être obtenue (celle-ci peut d'ailleurs aisément être retrouvée dans le graphe à partir du chemin lui-même) ;
On considère un graphe orienté valué G = (X,U,C) où :

- X ensemble de n nœuds, U ensemble de m arcs
- C_{ij} est le "poids" ou "coût" de l'arc (i,j). Le "coût" peut être un vrai coût, une distance, une durée, etc. Si G non orienté, on remplace chaque arête [i,j] par deux arcs (i,j) et (j,i) de même coût. Le coût d'un chemin est défini en général comme le coût total de ses arcs

On veut calculer un chemin de coût minimal (ou plus court chemin ou PCC) entre 2 nœuds donnés de G. Le problème a un sens si G ne contient pas de circuit de coût négatif (appelé circuit absorbant). On peut alors se limiter aux chemins élémentaires (sans nœuds répétés), avec au plus n - 1 arcs. Cette propriété est exploitée par tous les algorithmes.

xemple :
on peut être intéresser au calcul du PCC dans les réseaux de transport où :

- Nœuds = villes ou carrefours,
- arcs = routes ou rues,
- coûts = distances ou temps.

Chemin de fiabilité maximale :
par exemple en temps de guerre, un espion doit aller d'une ville s à une ville t. Les routes forment un graphe G = (X,U,P) où p(i,j) est la probabilité de passer l'arc (i,j) sans être pris. L'objectif est de trouver un chemin qui maximise la probabilité d'arriver en t (chemin de fiabilité maximale). Ici, le "coût" d'un chemin est inhabituel (produit des valeurs des arcs), et on est en maximisation.

Chemin de coût moyen optimal :

Un caboteur doit choisir un cycle sur n ports. On connaît pour chaque couple de ports (x,y) le profit prévu b(x,y) et la durée d(x,y). L'objectif est de trouver un cycle μ avec un profit maximum $C(\mu)$ par unité de temps. ce profit est donnée par l'équation suivante :

$$C(\mu) = \frac{\sum_{(x,y)\in\mu} b_{(x,y)}}{\sum_{(x,y)\in\mu} d_{(x,y)}}$$

un problème de cycle de coût moyen maximal dans un graphe bivalué G = (X,U,B,D). Pour résoudre les problèmes de PCC on peut faire la différences entre plusieurs problèmes.

Les algorithmes de chemins optimaux :
On considère le cas où le coût d'un chemin est la somme des coûts des arcs. Il existe des algorithmes pour 3 types de problèmes :

- Pb A : trouver un PCC entre deux nœuds s et t.
- Pb B : trouver un PCC d'un nœud s vers tout autre nœud, donc une arborescence de n - 1 chemins.
- Pb C : trouver un PCC entre tout couple de nœuds, sous forme d'une matrice $D(n \times n$ appelée distancier.

Un algorithm pour un problème peut être utilisé pour résoudre les deux autres, parfois en l'exécutant plusieurs fois.
Algorithmes vus : Dijkstra et Bellman, pour le pb B. Ils calculent pour tout nœud x un label V(x), coût des PCC entre le nœud de départ s et le nœud x.
L'algorithme de Dijkstra est du type à fixation de labels (label setting algorithm) : à chaque itération, il calcule la valeur définitive d'un label V(x).
Celui de Bellman est du type à ajustement de labels (label correcting algorithm) : il peut changer jusqu'à la dernière itération le label de chaque noeud.

5.1.1 Algorithme à fixation de labels de Dijkstra

le Principe de l'algorithme de Dijkstra qui suppose que les coûts non-négatifs. et le suivant :

Au début, le nœud de départ s a un label nul, les autres un label $+\infty$. L'ensemble F des nœuds fixés (de label définitif) est vide. A chaque itération, on cherche le nœud non fixé x de label minimal et on le fixe. Ensuite, pour tout successeur y de x, on met à jour V(y) si le chemin passant par x améliore V(y), c'est-à dire si V(x) + C(x,y) $<$ V(y). On note alors qu'on arrive à y en venant de x, grâce à un tableau P de prédécesseurs : P(y) := x. Si on veut un PCC de s vers un seul nœud t, on stoppe quand t est fixé. Sinon, on doit faire n itérations pour fixer tous les nœuds. A la fin, V et P définissent alors un PCC de s vers tout autre nœud. L'algorithme de Dijkstra et donné par le pseudo-code 28.

Exercice d'application dans le graphe de la figure 5.1.1 on cherche à trouver le plus court chemin entre le noeud 1 et le noued 8. la solution de cette exercice est donnée par le tableau suivant.

5.1.2 Algorithme de bellman-ford

L'algorithme de Bellman-Ford, aussi appelé algorithme de Bellman–Ford–Moore1, est un algorithme qui calcule des plus courts chemins depuis un sommet

Algorithm 3 Algorithme de Dijkstra

Require: Les coût des arc doivent être non négatifs
Ensure: un chemin optimla entre deux sommets

```
Données : Un graphe orienté pondéré G = (X, A, C) et un sommet s ∈ X
Résultat : Le plus court chemin de s vers tous les autres sommets de G
// V : Tableau stockant les étiquettes des sommets de G
Initialiser V à +∞
V[s] = 0
// P : Tableau permettant de retrouver la composition des chemins
Initialiser P à 0
P[s] = s
repeat
  // Recherche du sommet x non fixé de plus petite étiquette
  V_min = +∞
  for y allant de 1 à N faire do
    if y non marqué et V[y] ≤ V_min then
      x ⟵ y
      V_min ⟵ V[y]
    end if
    // Mise à jour des successeurs non fixés de x
    if V_min ≤ +∞ then
      Marquer x
      for tout successeur y de x do
        if y non marqué et V[x] + C[x, y] ≤ V[y] then
          V[y] = V[x] + C[x, y]
          P[y] = x
        end if
      end for
    end if
  end for
until { V_min = +∞}
```

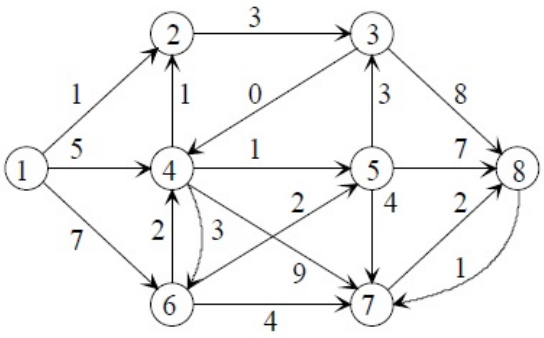


FIGURE 5.1 – Graphe Dijkstra

V[1]	V[2]	V[3]	V[4]	V[5]	V[6]	V[7]	V[8]	Nœud fixé
0	∞	∞	∞	∞	∞	∞	∞	1
-0-	1	∞	5	∞	7	∞	∞	2
-0-	-1-	4	5	∞	7	∞	∞	3
-0-	-1-	-4-	4	∞	7	∞	12	4
-0-	-1-	-4-	-4-	5	7	13	12	5
-0-	-1-	-4-	-4-	-5-	7	9	12	6
-0-	-1-	-4-	-4-	-5-	-7-	9	12	7
-0-	-1-	-4-	-4-	-5-	-7-	-9-	11	8
-0-	-1-	-4-	-4-	-5-	-7-	-9-	-11-	FIN

FIGURE 5.2 – solution Dijkstra

source donné dans un graphe orienté pondéré. Il porte le nom de ses inventeurs Richard Bellman et Lester Randolph Ford junior (publications en 1956 et 1958), et de Edward Forrest Moore qui le redécouvrit en 1959.

Contrairement à l'algorithme de Dijkstra, l'algorithme de Bellman-Ford autorise la présence de certains arcs de poids négatif et permet de détecter l'existence d'un circuit absorbant, c'est-à-dire de poids total strictement négatif, accessible depuis le sommet source.

La complexité de l'algorithme est en $o(|S||A|)$ où $|S|$ est le nombre de sommets $|A|$ est le nombre d'arcs.

Description :
L'algorithme de Bellman-Ford utilise le principe de la programmation dynamique, définie par les relations de récurrence suivantes :

$$V_0(S) = 0, \forall y \pm s : V_0(y) = +\infty$$

$$\forall k > 0, \forall y, V_k(y) = \min_{xprdy} \{V_{k-1}(y), V_{k-1}(x) + C(x,y)\}$$

il calcul des PCC d'au plus k arcs, k = 1,2,3, V0 tableau initial de labels, Vk labels à l'étape k. la troisième relation montre qu'un PCC de k arcs de s à y prolonge un PCC de k-1 arcs de s vers un prédécesseur x de y. En pratique, on calcule les tableaux Vk successifs en utilisant la troisième relation relation.L'algorithme stoppe quand les labels ne varient plus. Note : pour un graphe sans circuit (gestion de projet), il suffit d'appliquer la troisième relation niveau par niveau.

Algorithm 4 Algorithme de Bellman-Ford

```
Initialiser V à +∞, V[s] et P à 0
k := 0
repeat
  k :=k+1
  Initialiser W à + ∞ et q à 0
  for y :=1 to n do
    for each x prédécesseur de y do
      if V[x] + C(x,y) <W[y] then
        w[y] :=V[x] + C(x,y)
        P[y] := x ; q :=q+1
      end if
    end for
  end for
  // W devient V pour l'itération suivante
  V :=W
until q=0
```

Analyse de l'algorithme :

- Convergence : L'algorithme boucle s'il y a un circuit absorbant. Sinon, les labels sont stables au plus tard pour k = n-1 (nb max d'arcs d'un PCC élémentaire). L'algo fait une itération de plus constatant que q = 0.
- Optimalité : Par récurrence, on montre que V définit les PCC d'au plus k arcs en fin d'itération k.
- Complexité : Les deux for traduisent la troisième relation consultent tous les arcs (x,y) de G : une itération coûte O(m). Comme il y a au plus n itérations, l'algo est en O(mn), soit O(n^3) si G est complet.

5.2 Acheminement

le problème d'acheminement a pour but le déplacement de la marchandise entre les sources et des destination. Problème classique (transportation problem en anglais) a été étudié dès 1930 : Hitchcock (USA), Kantorovitch (URSS). Les données basique de ce problème sont

- X : m origines, disponibilités a_i avec $A = \sum_i a_i$
- Y : n destinations, demandes bj $B = \sum_j b_j$
- coûts unitaires de transport entre X et Y est donné par c_{ij}.
- Objectif : calculer un plan de transport (flux x_{ij}) pour satisfaire les demandes en minimisant le coût total.

Exemple de transport entre des origines et des destinations, dans un réseau de capacité non limitée. Exemple, transports maritimes : X ports européens, Y ports africains, c_{ij} coût de transport/t du port i au port j → graphe biparti. la figure 5.2 donne un exemple simple de graphe d'acheminement entre les sources et les destinations.

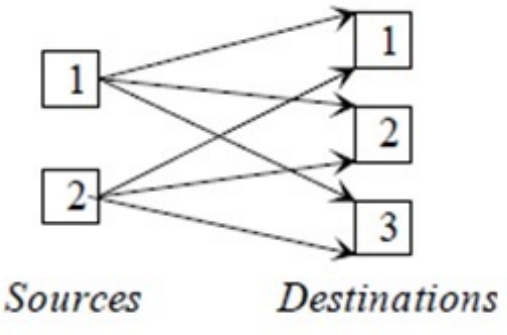


FIGURE 5.3 – Graphe bipartie donnant un plan d'acheminement

Autre exemple : distribution dans un réseau routier à deux couches. Un arc (i,j) modélise un PCC de i à j dans le réseau routier réel. Son coût c_{ij} a été calculé avec l'algorithme de Dijkstra, par exemple. Ce problème peut être modéliser par un programme mathématique linéaire (PL). Avant tout traitement de ce type de problème On suppose le problème est équilibré → (A = B), sinon :

- si B > A, source fictive de disponibilité B-A.
- si B < A, destination fictive n+1 de demande A-B.
- les arcs incidents aux nœuds fictifs ont un coût nul.

dans la solution, il faut ignorer les flux sur ces arcs.

Modèle mathématiques d'acheminement

Fonction objectif

$$(1)\min \sum_{i=1,..m} \sum_{j=1,..,n} c_{ij} \times x_{ij} \tag{5.1}$$

respect des disponibilités

$$(2)\forall i \in \{1,...,m\} : \sum_{j=1,...,n} x_{ij} = a_i \tag{5.2}$$

satisfaction des demandes

$$(3)\forall i \in \{1,...,m\} : \sum_{i} x_{ij} = b_j \tag{5.3}$$

Flux non-négatifs

$$.(4)\forall i \in \{1,...,m\}, \forall j \in \{1,...,n\} : x_{ij} \geq 0 \tag{5.4}$$

$$\tag{5.5}$$

Exemple :
on considère un exemple de transport entre trois sources et quatre destination. les données de cet exemple sont montrées dans la tableau suivant :

TABLE 5.1 – Données exemple d'acheminement

C	1	2	3	4	ai
1	7	2	9	10	8
2	6	11	7	12	6
3	15	8	3	4	9
bj	3	7	5	8	

Résolution :
Pour résoudre l'exemple précedent on peut utiliser la méthode heuristique du coin nord-ouest(CNO). On commence le remplissage des demandes en partant du haut à gauche (le CNO). Si on peut épuiser l source, on change de source. Si on peut satisfaire la destination, on passe à la suivante. Coût trouvé : 116.

TABLE 5.2 – Résolution de l'exemple d'acheminement

X	1	2	3	4
1	3	5	0	0
2	0	2	4	0
3	0	0	1	8

5.2.1 la méthode de stepping-stone

On peut utiliser la PL pour résoudre le problème, mais il existe un algorithme plus rapide, le stepping-stone (marelle).Grâce aux propriétés précédentes, il peut considérer uniquement les solutions entières en forme d'arbre. donc, Partant d'un arbre initial (comme celui de CNO), il construit des arbres successifs de coût décroissant. la solution donnée par la méthode CNO peut être présenter par le'arbre suivant. Dans l'arbre de la figure 5.2.1, si le flu x_{21}

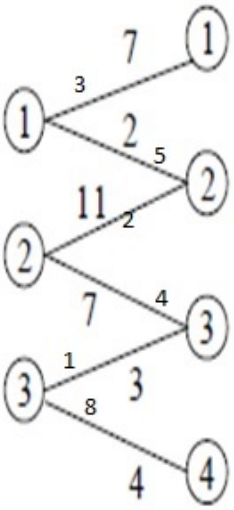


FIGURE 5.4 – arbre correspond à la solution du CNO

augmente de 0 à 1 alors pour respecter disponibilités et demandes, le flux doit diminuer sur (1,1) et (2,2) et augmenter sur (1,2). la variation de coût est appelée coût marginal de (2,1), elle vaut $D_{21} = 6 - 7 + 2 - 11 = -10$. l'augmentation de x_{21} est rentable mais limitée à 2, car x_{12} s'annule. on obtient un nouvel arbre moins coûteux, avec l'arête (2,1) en plus et (1,2) en moins.

principe de la méthode de sptepping-stone

- Le stepping-stone calcule à chaque itération un nouvel arbre moins coûteux, en cherchant un nouvel arc de coût marginal négatif D_{ij} le plus petit possible. Il s'arrête quand tous les coûts marginaux sont tous $>= 0$. On peut prouver que la solution est alors optimale.
- Pour calculer vite les D_{ij}, on munit tout dépôt i d'un potentiel u_i, et toute destination j d'un potentiel v_j. La source 1 a un potentiel 0. Puis, pour un client j voisin d'une source à u_i calculé, on pose $v_j = u_i - c_{ij}$ (le flux descend des dépôts) et, pour toute source i voisine d'un client à v_j calculé, on pose $u_i = v_j + c_{ij}$.

Exemple de potentiels sur l'arbre CNO :

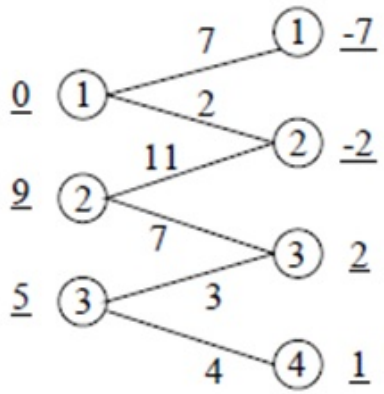


FIGURE 5.5 – Exemple d'affectation de potentiels pour les sommets d'un arbre

Calcul des potentiels sur un tableau :
Pour calculer les potentiels des sommets d'un arbre de la méthode du stepping-stone dans un tableau m xn. Détail d'une itération :

- rappeler des coûts en haut et à droite de chaque case
- écrire les flux $\neq 0$ de l'arbre actuel (encadrés)
- calculer les potentiels ui et vj
- calculer les coûts marginaux dans les cases à flux nuls
- si aucun n'est négatif, fin
- sélectionner l'arête [i,j] de coût réduit négatif minimum
- identifier le cycle créé par [i,j] avec des flèches
- mise à jour les flux sur le cycle (changement d'arbre).

la figure 5.2.1 montre un exemple d'utilisation des tableaux dans la méthode de stepping-stone. Le seul arc avec gain est (2,1), coût marginal -10. Le flux

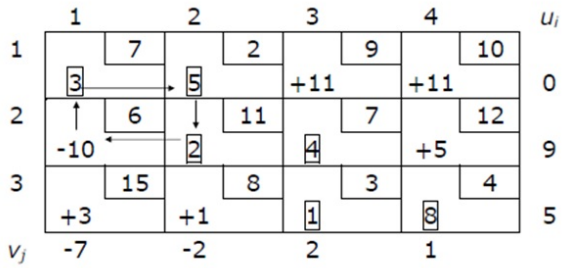


FIGURE 5.6 – Exemple de tableau de la méthode de stepping-stone

augmente sur (2,1) et (1,2), et diminue sur (1,1) et (2,2). x_{21} peut passer de 0 à 2 et x_{22} s'annule, figure 5.2.1 Nouvelle solution avec potentiels de l'itération

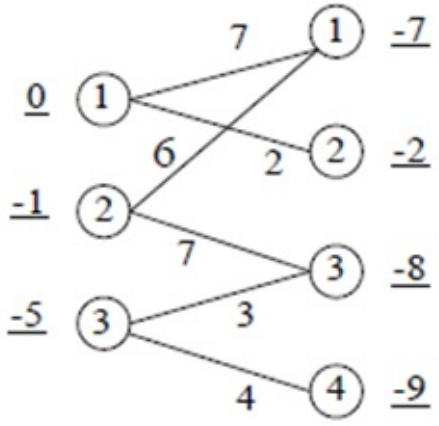


FIGURE 5.7 – deuxième arbre de la méthode stepping-stone

suivante. On a bien un arbre, de coût 116 + (-10) *2= 96. C'est l'ancien coût + $D_{21} \times x_{21}$. On obtient un nouveau tableau figure 5.2.1 Nouveau tableau. Coûts marginaux >=0 : optimum ! A la fin, il vaut mieux vérifier le respect des disponibilités et des demandes, ainsi que le coût de la solution

Dégénérescence :
Une solution dégénérée a moins de m+n-1 flux $\neq 0$: l'arbre est fragmenté en 2 sous-arbres (ou plus). Pour ne pas rater l'optimum, il faut absolument

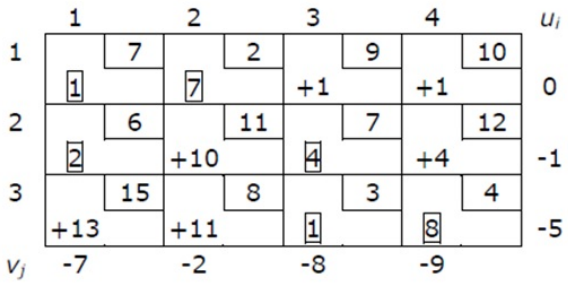


FIGURE 5.8 – deuxième tableau de la méthode stepping-stone

reconnecter les sous-arbres en incluant des arcs de flux infiniment petit e dans la solution : on peut ajouter (2,3) ou (3,2) dans l'exemple. Ces arcs sont traités ensuite exactement comme les autres. L'algorithme peut faire des changements d'arbre à gain nul (en fait e) si un arc fictif se trouve sur le cycle créé par le nouvel arc. Il faut continuer quand même les itérations et stopper seulement quand les D_{ij} sont $>= 0$. La seule précaution à prendre est de ne pas revenir sur une solution antérieure.

Exercice :
4 origines O1, O2, 03, O4 et 5 destinations D1, D2, D3, D4, D5. Chaque origine a une offre à respecter et chaque destination a une demande à satisfaire. Le tableau suivant présente les informations nécessaires Quelles quantités de marchandises à envoyer des origines vers les destinations en respectant l'offre et en satisfaisant la demande au moindre coût ?.

	D1	D2	D3	D4	D5	Offre
O1	7	12	1	5	9	12
O2	15	3	12	6	14	11
03	8	16	10	12	7	14
O4	18	8	17	11	16	8
Demande	10	11	15	5	4	

FIGURE 5.9 – Exercice d'application de la méthode stepping-stone

le tableau suivant5.2.1 donne la solution de l'exercice d'application

5.2.2 Méthode de Balas et hammer

Cette méthode est basée sur le calcul des regrets. Le regret associé à une ligne ou à une colonne est la différence entre le coût minimum et le coût

	D1	D2	D3	D4	D5	Offre
O1			12			12
O2		11				11
03	10				4	14
O4			3	5		8
Demande	10	11	15	5	4	

FIGURE 5.10 – Solution d'exercice d'application de la méthode stepping-stone

immédiatement supérieur dans cette ligne ou dans cette colonne. C'est une mesure de la priorité à accorder aux transports de cette ligne ou de cette colonne, car un regret important correspond a une pénalisation importante si on n'utilise pas la route de coût minimum. On remet constamment à jour ces regrets sur le tableau restant lorsqu'on a saturé une ligne ou une colonne en envoyant la quantité maximum sur la route de coût le plus faible dans la ligne ou la colonne de regret maximum

5.3 Problème d'affectation

lorsque un client ne peut être livré qu'à partir d'une seule source alors le problème d'acheminement se transforme ne problème d'affectation. le problème d'affectation peut être décrit par : dans un atelier à n postes et n personnes, on a mesuré pour chaque personne i et chaque poste j le nombre d'erreurs par heure c_{ij}quand on la met sur ce poste. Le but est d'affecter chaque personne à un poste et chaque poste à une personne pour minimiser le nombre total d'erreurs.

Modèle mathématiques du problème d'affectation

$$(1)\min \sum_{i=1,..m} \sum_{j=1,..,n} c_{ij} \times x_{ij} \tag{5.6}$$

Un client ne peut être affécter qu'à une seule source

$$(2)\forall i \in \{1,...,m\} : \sum_{j=1,...,n} x_{ij} = 1 \tag{5.7}$$

une source ne peut servir qu'à un seul client

$$(3)\forall i \in \{1,...,m\} : \sum_{i} x_{ij} = 1 \tag{5.8}$$

Flux non-négatifs

$$(4)\forall i \in \{1,...,m\}, \forall j \in \{1,...,n\} : x_{ij} \in \{0,1\} \tag{5.9}$$

5.4 problème du flot maximal

Le problème de flot maximum consiste à trouver, dans un réseau de flot, un flot réalisable depuis une source unique et vers un puits unique qui soit maximum. Quelquefois, on ne s'intéresse qu'à la valeur de ce flot. Le s-t flot maximum (depuis la source s vers le puits t) est égal à la s-t coupe minimum du graphe, comme l'indique le théorème flot-max/coupe-min.

Définition :
Soit un réseau avec des capacités (débit maximal permis) sur les arcs. Le problème du flot maximal consiste à faire circuler un flot de débit maximal entre deux nœuds donnés s et t, en respectant les capacités. Modèle de graphe orienté value $G = (X, U, C, s, t)$ où :

- X ensemble de n nœuds, U ensemble de m arcs
- C_{ij} capacité maximale (entière) de l'arc (i,j)
- nœud source s et nœud puits t.
- Un flot est une application ϕ de U dans $\mathcal{N}$:
- respectant les capacités des arcs :
-
$$\forall (i,j) \in U, 0 \leq \phi_{ij} \leq c_{ij}$$
- conservant le flot en chaque nœud (Kirchhoff)
-
$$\forall i \neq s,t, \sum_{\text{j succ de i}} \phi_{ij} = \sum_{\text{j préd de i}} \phi_{ji}$$
- de débit F égal au flot sortant de s ou entrant en t
-
$$F = \sum_{\text{j succ de s}} \phi_{sj} = \sum_{\text{j préd de t}} \phi_{jt}$$

Le problème du flot maximal consiste à trouver un flot maximisant F

Exemple :

Modèle mathématique du flot maximum :

C et ϕ considérés comme des matrices :

$$\begin{cases} MaxF \\ \sum_{\text{j succ i}} \phi ij - \sum_{\text{j pred i}} \phi_{ji} \\ 0 \leq \phi_{ij} \leq C_{ij}, \forall (i,j) \in U \end{cases} = \begin{cases} Fsii = s \\ 0 \forall i \neq s,t \\ -Fsii = t \end{cases}$$

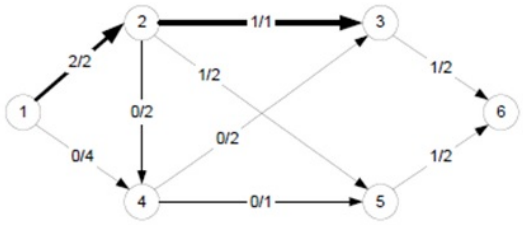


FIGURE 5.11 – Exercice d'application flot maximum

Cas particuliers :

- Dans le cas d'un graphe G non orienté. On remplace toute arête [i,j] par 2 arcs (i,j) et (j,i) de même capacité.
- si on a sources multiples ayant des disponibilités limitées. On les connecte à une super-source.
- Capacité sur un nœud x. On remplace x par x' et x", avec un arc (x', x") de même capa que x.
- Nœud j avec un seul prédécesseur i et un seul successeur k, on peut simplifier en remplaçant (i,j) et (j,k) par (i,k), avec $C_{ik} = \min(C_{ij}, C_{jk})$.

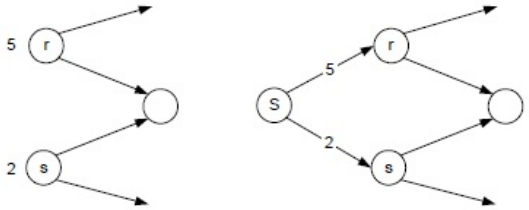


FIGURE 5.12 – remplacement de plusieurs sources par une seule

Coupes et autres concepts utiles

Une coupe (cut) de G est une partition des nœuds Z = (S,T) avec s dans S et t dans T. Un arc sortant de Z va de S à T, un arc entrant va de T à S. La capacité de la coupe est la somme des capacités de ses arcs sortants :

$$C(S,T) = \sum_{(i,j)inU, i \in S, j \in T} C_{ij}$$

Propriété 1 : Le débit de tout flot est inférieur ou égal à la capacité de toute coupe. Un flot est donc maximal si on trouve une coupe de capacité égale au débit.

Exemples d'applications :

1. Transports de marchandises : Les arcs sont des trajets avec des moyens de transport de capacités connues (tonnes / jour par exemple).
2. Transports de fluides : Les flots sont très utilisés pour les problèmes d'adduction d'eau et pour modéliser des réseaux de canalisations (oléoducs, gazoducs).

Fiabilité dans les réseaux de télécoms

3. — Si un réseau a k chemins disjoints (sans nœuds communs) entre 2 nœuds s et t, il peut résister à k-1 coupures de chemins.
   — On peut calculer k grâce à une méthode de flot : tout arc reçoit une capacité ∞ et chaque nœud sauf s et t une capacité 1. Un flot de débit k va tracer k chemins disjoints grâce aux capacités unitaires.
   — En maximisant le flot, on obtient le nb maximal de chemins disjoints de s à t.

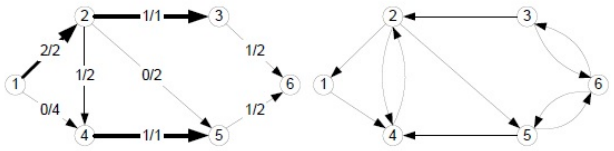


FIGURE 5.13 – chemin améliorant

Pour décrire les augmentations possibles, on peut construire un graphe d'écart $Ge(\Phi)$. Il contient les nœuds de G, les arcs non saturés, et un arc (j, i) pour tout arc (i, j) de flot non nul dans G. Une chaîne améliorante de G correspond à un chemin de s à t dans $Ge(\Phi)$. La plupart des algorithmes de flot ne génèrent pas de graphe d'écart. Avec les listes de successeurs et de prédécesseurs pour tout nœud i, ils recherchent des chaînes améliorantes en empruntant les arcs (i, j) non saturés et les arcs (j, i) de flot non nul.

5.4.1 méthode de résolution

Algorithme de Ford et Fulkerson : Principe :
On part du flot nul. Chaque itération cherche une chaîne améliorante μ de s

à t dans G. On stoppe si μ n'existe pas. Sinon, on calcule l'augmentation de flot δ. Soit $\mu+$ l'ensemble des arcs avant de μ (leur flux va augmenter) et μ - les arcs arrière (leur flux va baisser). Alors :

$$\delta = \min(\min_{(i,j)\in\mu^+}(C_{ij} - \Phi_{ij}), \min_{(i,j)\in\mu^-}(\Phi_{ij})$$

Le flux va augmenter de δ sur μ^+ et diminuer deδsur μ^-.

algorithm Ford-Fulkerson :

Algorithm 5 Algorithme de Ford

```
F :=0
Initialiser le flot Φ à 0
repeat
  Chercher une chaîne améliorante μ
  if μ existe then
    Calculer l'augmentation du débit δ
    Ajouter δ aux flux des arcs de μ+
    Diminuer de δ les flux des arcs μ−
    F := F + δ
  end if
until μ n'existe pas
```

Complexité de l'algorithme :

- On peut l'implémenter en $O(n.m^2)$. Il existe des algorithmes plus
- rapides mais plus compliqués :
- celui de Karzanov en $O(n^3)$
- celui de Ahuja en $O(m.n.logCmax)$

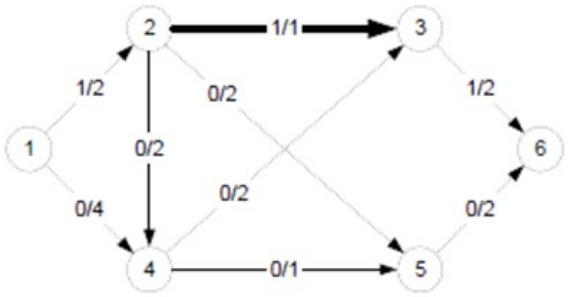
Optimalité de l'algorithme de Ford-Fulkerson :
A la fin, les nœuds marqués et non marqués forment une coupe (S,T). Soit la relation :

$$F = \sum_{(i,j)\in U, i\in S, j\in T} \Phi_{ij} - \sum_{(j,i)\in U, j\in T, i\in S-\{s\}} \Phi_{ji}$$

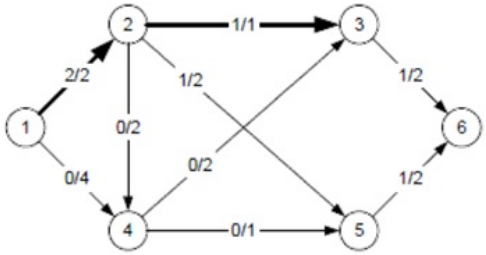
$\Phi_{ij} = C_{ij}$ pour (i,j) sortant de S, et $\Phi_{ji} = 0$ pour tout arc (j,i) entrant dans S, sinon on aurait pu marquer j. La $1^{\text{ère}}$ somme est égale à la capacité de la coupe, la seconde est nulle. On a un flot de débit égal à la capacité de la coupe, ce flot est donc maximal d'après la propriété 1. D'où les propriétés 2 et 3, la 2 est un théorème célèbre appelé max-flow, min-cut :

— Propriété 2 : Le débit maximal d'un flot de G est égal à la capacité minimale des coupes de G.
— Propriété 3 : L'algorithme de Ford et Fulkerson trouve un flot maximal de s à t.

Exemple :
Un exemple Reprenons le graphe précédent. Par convention, testons les successeurs des nœuds en ordre croissant. Partant du flot nul, on trouve la chaîne (1,2,3,6), avec $\delta = 1$ (arc (2,3)). On obtient le flot suivant, de débit F = 1 . On trouve ensuite (1,2,5,6) avec $\delta = 1$, dû à l'arc (1,2). L'augmentation

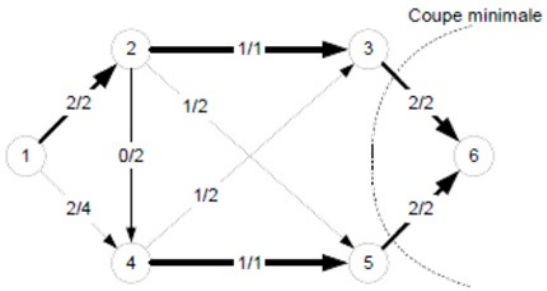


produit un débit F = 2 et sature l'arc (1,2), ce qui donne le flot suivant : Puis on trouve (1,4,3,6) et (1,4,5,6) avec $\delta = 1$. On obtient le flot suivant.



On marque 1, 4, 3, puis 2 en prenant (2,3) à l'envers, puis 5 en prenant l'arc avant (2,5). 6 n'est pas atteint : l'algorithme est terminé. Le flot max a donc un débit de 4, il Vérifie la propriété 2. en traçant la coupe minimale :

— les arcs sortant de la coupe à savoir (3,6) et (5,6) sont saturés.



— ici, la coupe n'a pas d'arcs arrière, sinon ils auraient un flot nul.
— la capacité de la coupe vaut bien 4

Exercice d'application :
Le graphe ci-dessous représente un réseau régional d'adduction d'eau. Les nœuds 1 à 4 sont des sources ou réservoirs pouvant fournir respectivement 15, 10, 15 et 15 milliers de m3 par jour. Les nœuds 10, 11 et 12 sont 3 villes dont les demandes journalières prévues dans dix ans sont respectivement de 15, 20 et 15 milliers de m3. Les disponibilités et demandes sont données près des nœuds. Les arcs sont des canalisations, aqueducs etc. avec des capacités en milliers de m3/j.

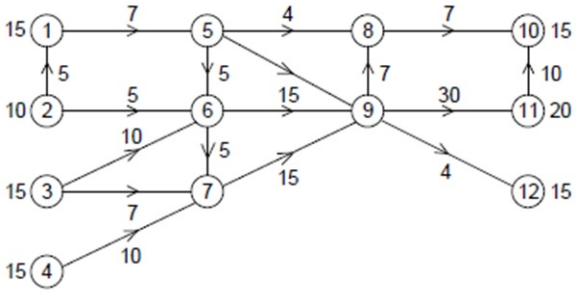


FIGURE 5.14 – graphe d'adduction d'eau

1. * Questions :
2. déterminer le flot maximal permis par ce réseaux et donner la coupe minimale.
3. Le réseau suffira-t-il à satisfaire les trois villes en 10 ans ?
4. Le conseil régional décide de refaire en priorité (en les élargissant) les deux canalisations les plus vétustes, entre les noeuds 1 et 5, et 9 et

12. Si aucune autre canalisation n'est refaite, jusqu'à quelle capacité peut-on augmenter utilement ces deux canalisations ?

5. déterminer le nouveau flot maximal après ces réparations

6. Devant le coût des travaux, le conseil souhaite finalement refaire une seule des deux canalisations laquelle recommandez-vous et pourquoi ?

5.5 Problèmes du flot maximal à coût minimal

Le problème du flot de coût minimum est un problème algorithmique de théorie des graphes, qui consiste à trouver la manière la plus économe d'utiliser un réseau de transport tout en satisfaisant les contraintes de production et de demande des nœuds du réseau. Il permet de modéliser tout un ensemble de problèmes pratiques dans lesquels il s'agit de trouver une manière optimale d'acheminer une ressource (par ex. un fluide, de l'électricité) d'un ensemble de sources à un ensemble de puits.

Le problème du flot de coût minimum est fondamental dans la mesure où la plupart des autres problèmes de flots, comme le problème de flot maximum, peuvent en être vus comme des cas particuliers. De plus, il est possible de résoudre le problème dans certains cas de manière efficace en utilisant l'algorithme du simplexe pour les réseaux.

5.5.1 Définition du problème

Définition et modélisation

Définition :
Soit un réseau G = (X,U,C,W,s,t) avec :

— X ensemble de n nœuds, U ensemble de m arcs,
— C_{ij} capacité (entière) de l'arc (i,j),
— W_{ij} coût de passage d'une unité de flux sur (i,j), une source s et un puits t.

l'objectif du flot maximum à coût minimal est de trouver les chemins à emprunter pour pouvoir far circuler entre une sources s et une destination t une quantité de flux égale à F à coût minimal.

5.5.2 Modèle mathématique

le modèle mathématique pour ce problème peut être obtenu à partir du problème du flot maximal en considérant que le flot final F est connu.

$$\begin{cases} \min \sum_{(i,j)\in U} \Phi_{ij} W_{ij} \\ \\ \sum_{\text{j succ i}} \phi ij - \sum_{\text{j pred i}} \phi_{ji} = \begin{cases} F & \text{si i =s} \\ 0 & \forall i \neq s,t \\ -F & \text{si i =t} \end{cases} \\ \\ 0 \leq \phi_{ij} \leq C_{ij}, \forall (i,j) \in U \end{cases}$$

Exemple de flots de coût minimum

1. On a F tonnes de matériel à transporter entre deux usines. Les arcs du réseau représentent des liaisons avec différents moyens de transport (route, rail, avion), chacune ayant une capacité et un coût. Le problème consiste à acheminer le matériel à coût minimal.

5.5.3 Résolution

Comme pour le flot max, il existe des algorithmes de graphes plus rapides que la PL, mais cette fois ils utilisent le graphe d'écart. Pour un flot Φ, le graphe d'écart $Ge(\Phi)$ doit tenir compte des coûts.

Graphe d'écart

— un arc non saturé (i, j) de G est conservé dans le graphe d'écart avec son coût ;
— tout arc (i, j) de flot non nul de G donne lieu à un arc inversé (j, i) de coût -W_{ij} dans Ge. Comme pour le flot maximal, un chemin π de s à t dans le graphe d'écart correspond à une chaîne améliorante μ de G avec des arcs avant et arrière.
— Une augmentation de flot le long de μ donnera un gain ou une perte selon le coût total du chemin π.

Algorithme de Buscker et Gowen :

Principe :

— On part du flot nul : débit $Q = 0$ et coût total $CT = 0$. On calcule un flot de débit F imposé, et de coût minimal.
— Différence avec Ford-Fulkerson : les augmentations se font le long de chaînes améliorantes de coût minimal.
— A chaque itération, on cherche donc un chemin π de coût minimal z, de s à t dans le graphe d'écart.

Algorithm 6 Algorithme de Buscker et Gowen

```
Initialiser le flot Q, CT et les flux des arcs à 0 à 0
repeat
  Construire le graphe d'écart H
  calculer dans H un chemin de coût min π de s à t, soit z son coût
  if π existe then
    soit μ la chaîne de G correspond à π
    Calculer l'augmentation du débit δ
    δ := min (δ,F-Q) // Note : F limite à Q
    Ajouter δ les flux des arcs μ+
    Soustraire δ les flux des arcs μ−
    Q := Q + δ ; CT := CT+ δ.z
  end if
until (μ non trouvé or (Q=F))
```

— Il faut utiliser l'algorithme de Bellman pour calculer π, à cause des coûts négatifs possibles dans H.

— Si F=constante, l'algorithme se termine quand $Q = F$ grâce au plafonnement de δ. Si $F = +\infty$, il stoppe quand il ne trouve plus de chaîne améliorante : on a alors un flot maximal et de coût minimal.

— Dans les 2 cas, le coût CT est minimal. L'optimalité de l'algorithme repose sur la propriété suivante, qu'on admettra :
Propriété 4 : à chaque itération, le flot obtenu est de coût minimal parmi tous les flots de débit Q.

Complexité :
Soit K le maximum des capacités, alors L'algorithme de Busacker et Gowen est en $O(m.n^2.K)$, mais il est rapide en moyenne.

Exercice d'application :

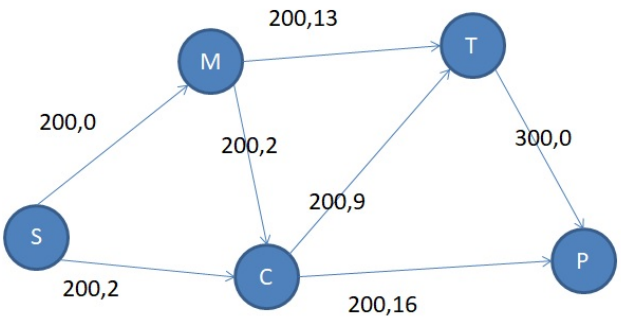


FIGURE 5.15 – Flot maximal à coût minimal

Chapitre 6

Problème de tournées de véhicules

Le problème de tournées de véhicules (aussi appelé VRP pour Vehicle Routing Problem) est une classe de problèmes de recherche opérationnelle et d'optimisation combinatoire. Il s'agit de déterminer les tournées d'une flotte de véhicules afin de livrer une liste de clients, ou de réaliser des tournées d'interventions (maintenance, réparation, contrôles) ou de visites (visites médicales, commerciales, etc.). Le but est de minimiser le coût de livraison des biens. Ce problème est une extension classique du problème du voyageur de commerce, et fait partie de la classe des problèmes $\mathcal{NP}$-complet.

Variantes du VRP Le problème VRP a été largement utilisé dans le domaine de distribution pour résoudre pluiseurs problèmes réels. Ces multiple application ont données lieu à plusieurs variantes du VRP :

- CVRP : contraintes de capacité : Les véhicules ont une capacité d'emport limitée (quantité, taille, poids, etc.) ;
- contraintes liées aux ressources et aux clients : disponibilité, localisation, compétences requises, etc. ;
- TWVRP : tournées de véhicules avec fenêtre de temps : Pour chaque client on impose une fenêtre de temps dans laquelle la livraison doit être effectuée ;
- Pickup and delivery : tournées de véhicules avec collecte et livraison : Un certain nombre de marchandises doivent être déplacées de sites de collecte vers des sites de livraisons.
- MCVRP : transport en utilisant des véhicules à plusieurs compartiments
- Split-VRP : problème dont les clients peuvent être visités plusieurs fois

6.1 Définition et modélisation

6.1.1 Données d'un problème de base en tournées

pour un problème de tournées classique les données nécessaires pour sa modélisation et résolution sont :

- un réseau G = (X,U,C), C_{ij} est souvent un temps
- un nœud dépôt s dans X
- une flotte de v véhicules de capacité W
- un sous-ensemble A de X de n nœuds-clients
- demande chaque client i et connue à l'avance q_i

Objectif :
construire des tournées de coût total minimal. Une tournée est un trajet d'un véhicule qui part du dépôt, traite certains clients et revient au dépôt. Dans sa version classique on suppose que tout client est traité par une seule tournée et en une seule visite dans cette tournée.

Contraction des problèmes :
En pratique, on contracte ces problèmes :

- ensemble S de n+1 nœuds : dépôt (index 0) et clients indexés de 1 à n
- distancier est une matrice $D_{(n+1)x(n+1)}$ qui définie les distances entre ces nœuds
- Revient à définir un graphe complet H = (S, SxS, D) sur ces nœuds : on peut aller de tout nœud i (client ou dépôt) à tout nœud j avec un coût Dij correspondant au plus court chemin de i à j.

Avantages de la contraction :
réduction de taille : exemple, réseau à 5000 nœuds mais 500 clients connus et 100 à livrer un jour donné. gain de temps : chemins optimaux pré-calculés. Le VRP est un cas particulier du problème de voyageur de commerce (Travel salesman problem en anglais). Alors, toutes les méthodes de résolution du TSP sont applicable pour résoudre le VRP.

- Le PVC (ou TSP, Traveling Salesman Problem) est le pb de tournées le plus simple : la somme des demandes n'excède pas W, on a une seule tournée.
- Toute tournée est donc une suite T = (T1, T2, ..., Tn) de clients, de coût total :

$$C(T) = C_{(0,T1)} + \sum_{i=1}^{n-1} D(T_i, T_{i+1}) + C(T_n, 0)$$

- Il y a O(n !) ordres possibles.

— Le PVC est classé NP-difficile, même si S est un ensemble de points du plan et D la distance euclidienne (PVC euclidien).

on peut distinguer entre deux types de PVC :

— PVC "orienté" (asymmétrique)

1. La solution du PL d'affectation donne une BI.
2. Si elle n'a aucun sous-cycle, c'est une solution optimale du PVC.

— PVC "non orienté" (symétrique)

1. Un arbre est un graphe connexe sans cycle.
2. Un arbre recouvrant d'un graphe H est formé d'arêtes de H et relie tous les nœuds de H.

6.1.2 Méthodes de résultions du TSP

n peut distinguer entre deux classes de méthodes pour la résolution du TSP :

— Méthodes Optimales : Elles peuvent résoudre de petits problèmes. parmi ces méthodes on peut citer les méthodes arborescentes (Little, Carpanetto).

— Bornes inférieures (BI) du coût optimal pour évaluer les heuristiques. Parfois optimales

Arbre couvrant à coût minimal :

En théorie des graphes, étant donné un graphe non orienté connexe dont les arêtes sont pondérées, un arbre couvrant de poids minimal (ACM) de ce graphe est un arbre couvrant (sous-ensemble qui est un arbre et qui connecte tous les sommets ensemble) dont la somme des poids des arêtes est minimale (c'est-à-dire de poids inférieur ou égal à celui de tous les autres arbres couvrants du graphe). L'arbre couvrant de poids minimal est aussi connu sous certains autres noms, tel qu'arbre couvrant minimum ou encore arbre sous-tendant minimum.

L'arbre couvrant minimum peut s'interpréter de différentes manières selon ce que représente le graphe. De manière générale si on considère un réseau où un ensemble d'objets doivent être reliés entre eux (par exemple un réseau électrique et des habitations), l'arbre couvrant minimum est la façon de construire un tel réseau en minimisant un coût représenté par le poids des arêtes (par exemple la longueur totale de câble utilisée pour construire un réseau électrique). Une tournée optimale est un cycle, e effet, ôtons une arête à une tournée optimale T*, on obtient un arbre recouvrant A particulier (non ramifié), et C(A*) <= C(A) <= C(T*). Amélioration de la borne de l'arbre en utilisant les 1-arbres :

— Un 1-arbre est un arbre recouvrant auquel on ajoute une arête incidente au dépôt, ce qui crée un seul cycle.

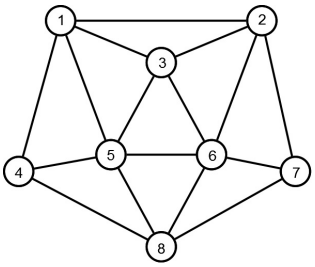


FIGURE 6.1 – Graphe à couvrir pour un arbre

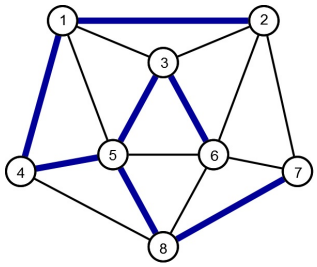


FIGURE 6.2 – Abre recouvrant un graphe

— Un 1-arbre B* de coût minimal donne une borne inférieure, car toute tournée est un 1-arbre particulier, avec tous les nœuds sur l'unique cycle.

Pour chercher les 1-arbres on peut utiliser plusieurs méthodes, parmi celles les plus connues on présentera les deux méthodes de Prim et Kruskal.

6.1.3 La méthode de Prim

L'algorithme de Prim calcule un arbre recouvrant A* de coût minimal. On obtient un 1-arbre optimal B* en ajoutant à A* la plus petite arête incidente au dépôt.

Algorithm 7 Algorithme de Prim

Initialiser T avec Sommets := un sommet de G qu'on choisit, arêtes : aucune
repeat
  Trouver toutes les arêtes de G qui relient un sommet de T et un sommet extérieur à T
  Parmi celle-ci, choisir l'arête ayant le plus petit possible
  Ajouter à T cette arête et le sommet correspondant
until (s'arrête dès que tous les sommets de G sont dans T)

l'arbre recouvrant construit par Kruskal est de poids minimal. Soit e' une arête du cycle de $T' \cup \{e\}$ qui n'est pas une arête de T.

— On note $T'' = T' \cup \{e\} - \{e'\}$
— T" est arbre recouvrant de G
— On a $Poids(T'') \geq Poids(T')$ (car T' est de poids minimal)
— $Poids(e)$‘$Poids()$

Avant l'étape d'ajout de e, T , T' et T" sont identiques

— Or e est choisi de poids minimal construisant T
— comme on n'a pas choisi e' à la place de e, c'est que en fait $Poids(e) = Poids(e')$

Algorithm 8 Algorithme de Kruskal

Initialiser T avec Sommets := tous les sommets de G qu'on choisit, arêtes : aucune
repeat
  Traiter les arêtes de G une après l'autre par poids croissant :
  if si une arête permet de connecter deux composantes connexes de T **then**
    Ajouter cette arête à T
  else
    ne Rien Faire
  end if
  Passer à l'arête suivante
  Ajouter à T cette arête et le sommet correspondant
until (s'arrête quand il n'y pas plus d'arêtes)

— donc on a aussi $Poids(T'') = Poids(T')$, c'est-à-dire que T" est de poids minimum
— Mais T" a plus d'arêtes communes avec T que T', ce qui est contraire à l'hypothèse de départ :"T' est un arbre recouvrant de poids minimal de G, différant le moins possible de T".
— conclusion : T' n'existe pas et par conséquent T est un arbre recouvrant de poids minimal de G.

Exercice à résoudre avec la méthode de Kruskal on considère le graphe de la figure 6.1.3

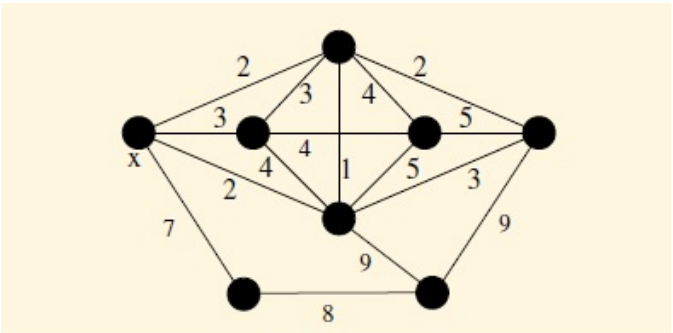


FIGURE 6.3 – Exercice à faire avec la méthode de kruskal

Exercice 2 :
On considère le graphe de la figure **??** trouver l'arbre couvrant de coût minimal ainsi que le cyclle associé.

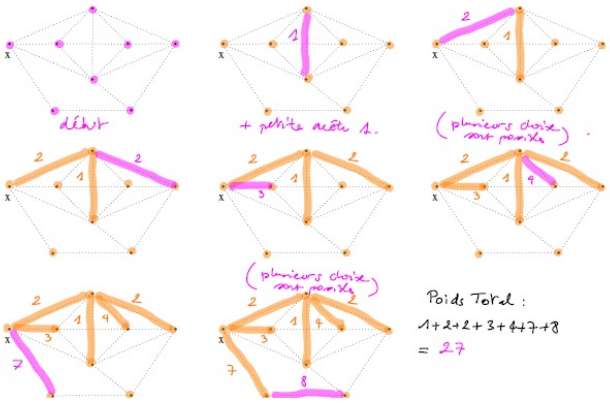


FIGURE 6.4 – Résolution de l'exercice avec la méthode de kruskal

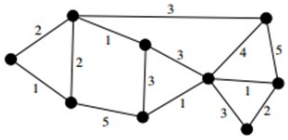


FIGURE 6.5 – Graphe à couvrir pour un arbre

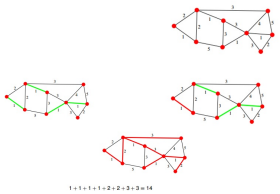


FIGURE 6.6 – solution

6.1.4 Heuristiques constructives pour le TSP

Les méthodes heuristiques construisent une solution par des décisions successives, et sont généralement des méthodes intuitives. Pour le TSP parmi les heuristiques les plus connues on peut citer :

- heuristique Plus Proche Voisin (PPV)
- heuristique de Fletcher
- méthode de l'élastique (Shamos, PVC euclidien)
- Méthodes d'insertion

Méthodes d'insertion le principe de la méthode consiste en : a partir d'une boucle sur le dépôt, elles insèrent un client à chaque étape. trois type d'insertion peuvent être considérés :

- Plus Proche Insertion (PPI),
- Plus Lointaine Insertion (PLI),
- Meilleure Insertion (MI).

Itération de base pour PPI et PLI : Chercher le client libre j le plus près (PPI) ou le plus loin (PLI) du cycle en cours de construction. insérer j entre deux nœuds i et k du cycle pour minimiser la variation de coût $D_{ij} + D_{jk} - D_{ik}$.

pour la méthode du PPV on part du dépôt puis on relie à chaque étape le client libre le plus proche du dernier nœud visité. Fletcher part de nœuds isolés et prend à chaque étape la plus petite arête ne faisant ni fourche, ni cycle avec les arêtes déjà prises.

Itération de base pour MI :
tester chaque client libre j et chaque couple de nœuds consécutifs (i, k) sur le cycle. insérer le nœud donnant la plus faible variation de coût, à la position correspondante. Les méthodes d'insertion donnent de bons résultats en moyenne. De manière anti-intuitive, PLI donne les meilleurs résultats moyens dans le cas euclidien !

Exercice TSP :
On considère le graphe de la figure 6.1.5 le dépôt est situé au point O(0,0). On cherche à trouver la tournée de coût minimal en partant de 0.

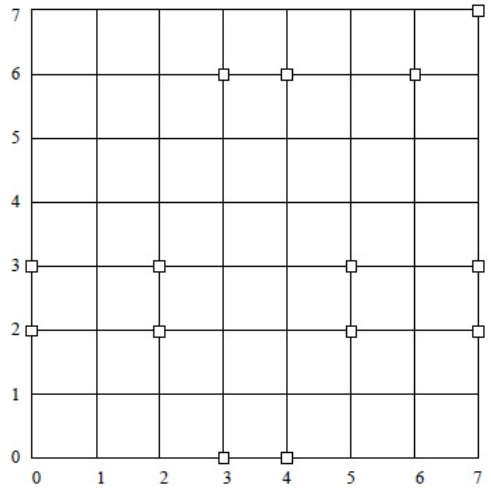


FIGURE 6.7 – Exercice TSP

Questions :

1. Résoudre avec l'algorithme de PPV
2. Algorithme de fletcher
3. Algorithme de shamos

6.1.5 Méthode de l'arbre de Christofides

Calcul d'un ARCM (algo de Prim). Puis "tour de l'arbre" et élimination des nœuds déjà visités. Résultat médiocre en moyenne, mais on peut prouver qu'elle n'est jamais à plus de deux fois l'optimum : elle calcule une tournée T telle que $C(T)/C(T*) \leq 2$.

FIGURE 6.8 – Exercice TSP

Amélioration des solutions des heuristiques

pour améliorer la qualité des solutions obtenues par les heuristique on applique les méthodes de la recherche locale. Poir le TSP et le VRP ils existent plusieurs méthode de recherche locale :

- la méthode 2-opt
- La méthode R-opt
- Echange
- ...

algorithme 2-opt :

Algorithm 9 Algorithme de 2-opt

repeat
  $\Delta_{min} := \inf$
  for each couple d'arêtes ((u,x),(v,y)) non contigües **do**
    $\Delta = D(x,v) + D(u,y) - D(x,u) - D(v,y)$
    if $\Delta < \Delta_{min}$ **then**
      $\Delta_{min} := \Delta$; p := ((u,x),(v,y))
    end if
  end for
  if $\Delta_{min} \leq 0$ **then**
    croiser réellement les arêtes de la paire p dans S
  else
    ne rien faire
  end if
until $(\Delta_{min} = \infty)$

transormation R-opt :
Déplacement d'une chaîne de nœuds consécutifs.

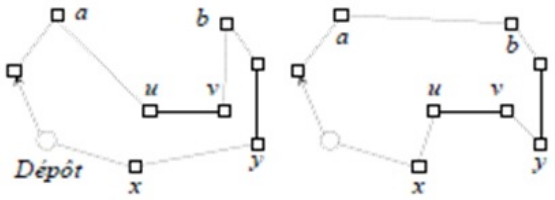


FIGURE 6.9 – Transformation r-opt

Principe de la méthode :

- on calcule le gain $\Delta = D(a,b)+D(y,v)+D(u,x)-D(a,u)-D(v,b)-D(y,x)$
- si $\Delta \leq 0$ alors :
- on déplace toute la chaîne
- sens de circulation conservé, sauf sur (u,v)
- en pratique, on se limite à des chaînes de 1 à 3 nœuds

6.2 Le Problème de Tournées de Véhicules

Le PTV (VRP, Vehicle Routing Problem) étend le PVC au cas de plusieurs tournées. Il est encore plus difficile : on ne connaît pas l'optimum pour certains pbs à 75 clients. Dans un VRP on suppose que la visite de tous les clients ne peut pas être faite en une seule tournée vue les contraintes du problème. Donc on pour servir les clients on doit organiser plusieurs tournées.

6.2.1 description et Modélisation du VRP

dans cette section nous définissons le problème du VRP. le problème est donné par un ensemble de clients $N = \{1, 2, ..., n\}$, situés en n différentes location. chaque paire de locations (i,j), où $i, j \in N$ et $i \neq j$, est associé à un temps de voyage t_{ij} ou une distance d_{ij} supposées symétriques. nous dénotons par $q_i, i = 1, 2, ..., n$ la demande au point i. Le dépôt centrale on le dénote par 0.
Fleets of vehicles
Les clients sont sevrés à partir du dépôt par un ensemble homogène et fini de véhicules. pour chaque tournée les véhicules font leur départ à partir du dépôt où ils doivent terminer aussi les tournées de livraison. Il y a un ensemble de véhicules, $V = \{1, 2, ..., m\}$ de même capacités. La capacité de chaque

véhicule k est donnée par a_k. nous dénotons par $R_i = \{r_i(1), r_i(2), ..., r_i(n)\}$ la route de chaque véhicule i, où $r_i(j)$ est l'indice du j$^{\text{ème}}$ client visité et n_i est le nombre de clients dans la tournée. nosu supposons que chaque tournée termine au dépôt, i.e. $r_i(n+1) = 0$.
les variables de décision du modèle sont :

$$x_{ij}^k = \begin{cases} 1 \text{ si le client j est visité après i par le véhicule k} \\ 0 \text{ sinon} \end{cases}$$

$$y_i^k = \begin{cases} 1 \text{ si le client i est visité le véhicule k} \\ 0 \text{ sinon} \end{cases}$$

L'objectif de ce module est de minimiser la distance totale de toutes les tournées. donc la fonction objectif peut s'écrire comme suit :

$$\min \sum_{i=0}^{n} \sum_{j=0}^{n} \sum_{k=1}^{m} d_{ij} x_{ij}^k$$

les contraintes du modèle sont :

$$\sum_{j=0}^{n} x_{0j}^n = 1 \qquad k = 1, 2, , ..., m$$

cette contrainte guarantie que chaque chaque véhicule quitera le dépôt et arrivera chez un client.

$$\sum_{i=0}^{n} x_{ip}^k - \sum_{j=0}^{n} x_{pj}^k = 0 \qquad p = 0, 1, ..., n; \qquad k = 1, ..., m$$

cette contrainte concerne le flux d'entrée et de sortie, garantie que chaque véhicule va quitter un client donné et arrivera au dépôt.

$$\sum_{k=1}^{m} y_i^k = 1, \qquad i = 1, ..., n$$

cette contrainte garantie que la demande de chaque client va être servie.

$$\sum_{i=1}^{n} q_i y_i^k \leq a_k, \qquad k = 1, ..., m$$

cette contrainte assure la non violation de la capacité du véhicule.

$$y_i^k \leq \sum_{j=0}^{n} x_{ji}^k \qquad i = 1, ..., n; \quad k = 1, ..., m.$$

cette contrainte garantie que la demande d'un client sera servie par un véhicule seulement s'il a visité.

$$x_{ij}^k, y_i^k \in \{0, 1\}, \quad i = 0, ..., n; \quad j = 0, ..., n; \quad k = 0, ..., m$$

cette contrainte garantie la bilinéarité des variables de décisions.

6.3 Méthodes de résolution du VRP

6.3.1 Heuristiques constructives inspirées du TSP

- Toute heuristique du PVC est utilisable pour construire les tournées du PTV une par une. On crée une tournée quand la capacité du véhicule est épuisée. C'est le mode séquentiel.
- Par exemple, on peut appliquer PPV en mode séquentiel. Le nombre de véhicules utilisés (= de tournées) nv fait partie des résultats.
- Si nv est imposé, on utilise le mode parallèle :
  - on construit nv tournées T1, T2, ..., Tnv en parallèle.
  - chaque itération de l'heuristique détermine les meilleures décisions pour T1, T2, ..., Tnv.
  - exemple avec PPV : on détermine le prochain client pour T1, puis T2 etc jusqu'à Tnv et on recommence.
  - Ce mode parallèle donne des tournées mieux équilibrées.

6.3.2 Heuristique de Clarke & Wright

cette méthode appelée aussi la méthode de la marguerite consiste à :

- initialiser la marguerite (n tours avec un client chacun)
- on calcule le gain de toutes le fusions deux à deux possible
- fusions de 2 tours tant que cela procure un gain à chaque étape,
- On commence par la fusion de gain maximal
- Si le véhicule de la tournée T1 peut faire T2 sans revenir au dépôt (demande totale <= W), le gain de la fusion est : e(y,u) = D(y,s) + D(s,u) – D(y,u).
- **NB** : 4 fusions possibles si réseau non orienté.
- Une itération de l'algorithme teste toutes les paires de tournées et les 4 cas par paire. Elle réalise la fusion de gain > 0 maximal, si elle existe. Sinon, c'est la fin de l'algorithme.
- Chaque itération diminuant le coût et économisant aussi une tournée, la méthode est réputée pour minimiser à la fois le coût et le nombre de véhicules utilisés.

6.3.3 Heuristique de Gillett et Miller

la méthode heuristique dite **SWEEP** de Méthode de type cluster first - route second : on construit des groupes de clients puis on calcule une tournée dans chaque groupe. Principe :

- trier les clients par angle polaire croissant / dépôt
- les balayer à partir d'un client de départ quelconque
- ceci donne des secteurs compatibles avec la capacité des véhicules
- calculer une tournée par secteur avec PPV
- améliorer chaque tournée avec une méthode de recherche locale

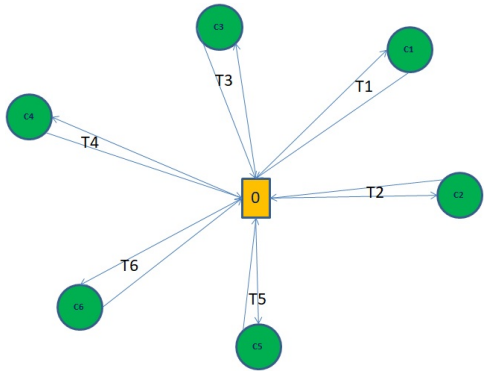


FIGURE 6.10 – Méthode de la margueritte

— cette méthode est très utile pour trouver une solution de départ bons résultats en campagne ou à grande échelle et avec un dépôt central. La figure 6.3.3 montre un exemple de décomposition de l'espace des clients sous forme de clusters, chaque cluster est constitué par un ensemble de clients dont la somme des demandes ne dépasse pas la capacité du véhicule.

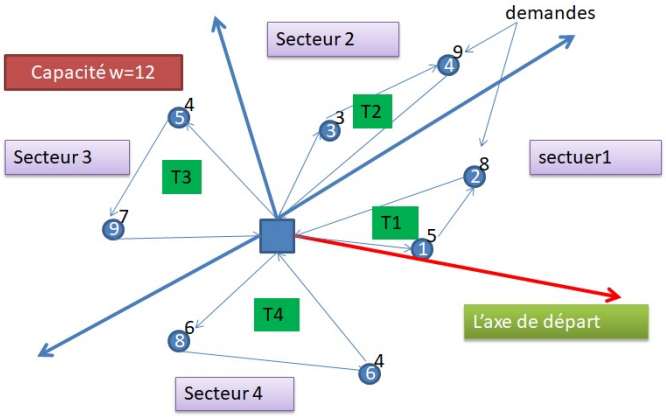


FIGURE 6.11 – Méthode de la gillet et Miller

6.3.4 Heuristique de Beasley

cette approche dite aussi route-first cluster-second a été proposé par Newton et thomas en 1974, Bodin et Berman 1979 et finalement par Beasly en 1983 construit dans un premier temps un tour géant visitant tous les clients, qui est ensuite découpé en plusieurs routes issues des dépôt. La résolution du problème de découpage peut se faire d'une manière exacte, se réduisant

à un problème de plus court chemin dans un graphe acyclique.

Principe de la méthode :

— on calcule un tour géant visitant tous les clients, puis o
— on découpe ce tour en tournées réalisables.
— Le découpage peut se faire optimalement

On construit un graphe auxiliaire H avec nc+1 nœuds indexés de 0 à nc. Si la sous-suite de clients Si ... Sj peut constituer une tournée réalisable, elle est modélisée dans H par un arc (i-1,j), valué par le coût de la tournée.
Le découpage optimal en tournées correspond à un chemin optimal du 1er au dernier nœud dans H.

Exemple d'application :
On considère le problème de distribution donner par la graphe de la figure 6.3.4, suivant avec un tour géant S=(a,b,c,d,e) à nc=5 clients. Les demandes sont entre parenthèses. Les allers-retours possibles au dépôt sont en pointillés. La capacité des véhicules est W = 10. la solution finale est donnée par

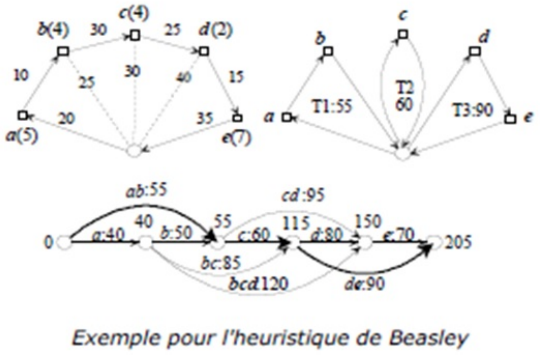


FIGURE 6.12 – graphe d'application de la méthode de Beasley

les arcs en gras, le coût total pour cet exemple est de 205.

Exercice VRP on considère un problème de distibution de la demade de cinq clients dont les demande et les distance sont données par les deux tableau de la figure 6.3.4, on suppose que les véhicules sont homogènes ayant une capacité de 90.

Questions

Distancier

From \ To	0	1	2	3	4	5
0	-	28	31	20	25	34
1		-	21	29	26	20
2			-	38	20	32
3				-	30	27
4					-	25
5						-

Customer	Quantity
1	37
2	35
3	30
4	25
5	32

FIGURE 6.13 – graphe d'application de la méthode de Beasley

— résoudre le problème en utilisant la méthode dite sweep de gillet et miller
— Résoudre le problème en utilisant la méthode de l amargueritte
— améliorer les solutions trouvées en utilisant les méthodes de recherche locale, 2-op, exchange.

Chapitre 7

Problème de remplissage

le problème de remplissage vise à rendre chaque voyage d'un véhicule profitable le maximum possible. ce problème est similaire un problème du sac-à-dos. Le problème du sac-à-dos fait partie des 21 problèmes NP-complets identifiés par Richard Karp en 1972. Ces 21 problèmes sont réputés comme les problèmes les plus difficiles en optimisation combinatoire. Un grand nombre d'autres problèmes NP-complets peuvent se ramener à ces 21 problèmes de base.
La question majeure du problème du sac-à-dos est comment faire pour remplir mon sac le mieux possible. L'énoncé de ce problème est simple : « Étant donné plusieurs objets possédant chacun un poids et une valeur et étant donné un poids maximum pour le sac, quels objets faut-il mettre dans le sac de manière à maximiser la valeur totale sans dépasser le poids maximal autorisé pour le sac ? ».

7.1 Modélisation mathématiques

Toute formulation commence par un énoncé des données. Dans notre cas, nous avons un sac à dos de poids maximal W et n objets. Pour chaque objet i, nous avons un poids wi et une valeur pi. On cherche à maximiser la valeur des objets mis dans le sac-à-dos. Cette valeur peut être donnée par l'équation suivante :

$$obj := \max \sum_{i=1}^{n} X_i \times V_i$$

$$\text{contraintes} \sum_{i=1}^{n} w_i \times x_i \leq W$$

$$x_i \in \{0,1\} \forall i \in \{1,...,n\}$$

où :

— la variable de décision

$$x_i = \begin{cases} 1 \text{ si l'objet i est mis dans le sac-à-dos} \\ 0 \text{ sinon} \end{cases}$$

— V_i : la valeur de l'objet i
— w_i : le poids de l'objet i
— W : la capacité du sac-à-dos

7.2 Méthodes de résolution

Il existe deux grandes catégories de méthodes de résolution de problèmes d'optimisation combinatoire : les méthodes exactes et les méthodes approchées. Les méthodes exactes permettent d'obtenir la solution optimale à chaque fois, mais le temps de calcul peut être long si le problème est compliqué à résoudre. Les méthodes approchées, encore appelées heuristiques, permettent d'obtenir rapidement une solution approchée, donc pas nécessairement optimale. Nous allons détailler un exemple d'algorithme de résolution de chaque catégorie.

7.2.1 Méthodes approchées

Une méthode approchée a pour but de trouver une solution avec un bon compromis entre la qualité de la solution et le temps de calcul. Pour le problème du sac à dos, voici un exemple d'algorithme de ce type :

Algorithme heuristique pour le Knapsack :

Algorithm 10 Heuristique pour le Knapsack

DEBUT
calculer la valeur (pi / wi) pour chaque objet i,
trier tous les objets par ordre décroissant de cette valeur,
sélectionner les objets un à un dans l'ordre du tri et
ajouter l'objet sélectionné dans le sac si le poids maximal reste respecté.
FIN

Exemple d'application :
on considère le problème du sac-à-dos suivant : on appliquant l'heuristique à ce problème on obtient :
étape 1 : etape 2 :
l'ordre des objets est donc le suivant 1,3,2 et 4. etape 3 (sélection des objets= :

Objets	1	2	3	4
p_i	7	4	3	3
w_i	13	12	8	10

FIGURE 7.1 – problème du sac-à-dos

Objets	1	2	3	4
p_i / w_i	0,54	0,33	0,37	0,30

FIGURE 7.2 – problème du sac-à-dos

on met dans le sac les objets 1, 3 et 2.
La solution trouvée est donc de mettre les objets 1 et 3 dans le sac, ce qui donne une valeur de 10. Cette solution n'est pas optimale, puisqu'une solution avec une valeur totale de 11 existe. Néanmoins, cet algorithme reste rapide même si le nombre d'objets augmente considérablement.
Ce type d'algorithme est aussi appelé algorithme glouton, car il ne remet jamais en cause une décision prise auparavant. Ici, lorsque l'objet 2 ne peut pas entrer dans le sac, l'algorithme n'essaie pas d'enlever l'objet 3 du sac pour y mettre l'objet 2 à sa place.

7.2.2 Méthode exacte

:
Pour trouver la solution optimale, et être certain qu'il n'y a pas mieux, il faut utiliser une méthode exacte. Généralement, les méthodes exacte consomment beaucoup de temps pour résoudre les instances de grande et moyenne taille. Il n'existe pas une méthode exacte universellement plus rapide que toutes les autres. Chaque problème possède des méthodes mieux adaptées que d'autres. parmi les méthodes exactes on peut citer :

- Branch & Bound $B\&B$
- Branch & cut $B\&C$
- Branch & price $B\&P$
- Programmation dynamique

Une $B\&B$ est un algorithme qui permet d'énumérer intelligemment toutes les solutions possibles. En pratique, seules les solutions potentiellement de bonne qualité seront énumérées, les solutions ne pouvant pas conduire à améliorer la solution courante ne sont pas explorées. Pour représenter une

PSE, nous utilisons un « arbre de recherche » constitué :

— de nœuds ou sommets, où un nœud représente une étape de construction de la solution

— d'arcs pour indiquer certains choix faits pour construire la solution.

Dans cet exemple, les nœuds représentent une étape où un certain nombre d'objets auront été mis dans le sac, d'autres laissés en dehors du sac, et d'autres objets pour lesquels aucune décision n'a encore été prise. Chaque arc indique l'action de mettre un objet dans le sac ou au contraire de ne pas le mettre dans le sac. La figure suivante représente une partie de l'arbre de recherche.

L'arbre de recherche commence par un seul nœud, où aucune décision n'a été prise pour les objets. Une fois tous les objets sélectionnés, les nœuds finaux, aussi appelés nœuds feuilles, représentent chacun une solution finale. La figure suivante reprend l'exemple précédent et représente l'arbre de recherche induit. Les nœuds en rouge représentent une solution impossible. Par

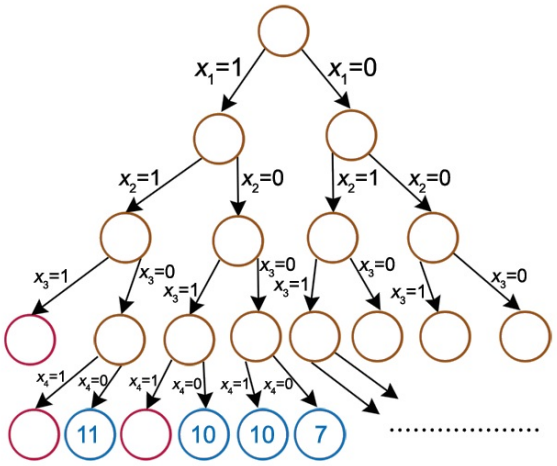


FIGURE 7.3 – $B\&B$ du sac-à-dos

exemple, pour le nœud tout à gauche en rouge, il est impossible de prendre les trois premiers objets à cause du poids maximal à ne pas dépasser. Il n'est donc pas nécessaire de développer l'étape suivante avec le dernier objet. Les nœuds en bleu sont des nœuds feuilles correspondant à une solution réalisable. À la fin de l'algorithme, il suffit de calculer la valeur du sac pour chaque nœud feuille et de prendre la solution avec la plus grande valeur.

Programmation dynamique On commence par diviser le sac de capacité k en k sous-sacs dont la capacité allant de zero à k.

— La méthode commence par la construction d'un tableau V donnant les bénéfices possible pour chaque sous-sac

— Ordonner les produits par volume croissant
— Chaque case du tableau représente le bénéfice maximum possible V(i,w) pour les i premiers objets avec une capacité w.

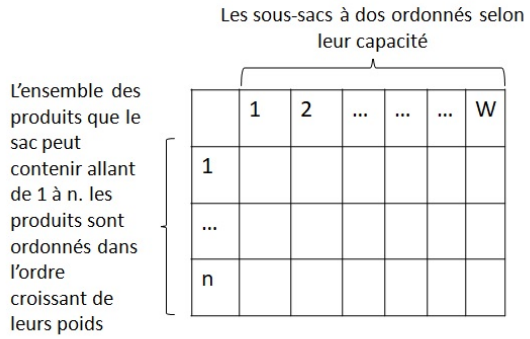


FIGURE 7.4 – Tableau de la méthode de la programmation dynamique Knapsack

7.2.3 Algorithme de la programmation dynamique

Algorithme programmation dynamique

Exercice d'application :
dans un sac-à-dos de capacité W=5, on cherche à sélectionner parmi quatre produits, dont les données sont données par le tableau suivant, choisir les produits qui maximisent la valeur du sac-à-dos. L'algorithme de la program-

TABLE 7.1 – Exercice knapsack

$Produit_i$	1	2	3	4
wi	2	3	4	5
bi	3	4	5	6

mation dynamique, sous forme de tableau, commence par ordonner les produits dans la première colonne dans un ordre croissant par poids. Ensuite, on initialise la deuxième ligne et la deuxième colonne à 0. la msie à jour des valeurs du tableau V se fait en suivant les étapes décrite dans l'algorithme 7.2.2.

Algorithm 11 Algorithme programmation dynamique pour le Knapsack

```
// Remplir la première ligne du tableau de bénéfice
for  w=0 to W do
   V[0,w] :=0
end for
// Initialiser la première colonne
for i=0 to n do
   V [i,0] :=0
end for
for  i= 1 to n  do
   for  w=1 to W do
      if wi <= w then
         if bi + V [i-1,w-wi] > V[i-1,w] then
            V[i,w] :=V[i-1,w-wi] +bi
         else
            V[i,w] :=V[i-1,w]
         end if
      else
         V[i,w] :=V[i-1,w]
      end if
   end for
end for
// Pour déterminer les produits qui sont dans le sac à dos on procède de
la manière suivante
i :=n et k :=W
for i=n to 1 do
   if  (V [i,k] ≠ V[i-1,k])  then
      // Entre l'itération i-1 et i , le produit a été ajouté
      i := i-1 , k := k-wi
      // la capacité restante après l'ajout du produit
   else
      i : =i-1
   end if
end for
```

i\ wi	0	1	2	3	4	5
0	0	0	0	0	0	0
1	0	0	3	3	3	3
2	0	0	3	4	4	7
3	0	0	3	4	5	7
4	0	0	3	4	5	7

FIGURE 7.5 – Solution de l'exercice avec la programmation dynamique Knapsack

7.3 Exercices

:

Exercice 1 :

Une entreprise de distribution dispose de trois entrepôts C1, C2 et C2 dont les capacités sont respectivement 4,5 et3. L'entreprise livre trois clients CL1, CL2 et CL3 dont les demandes sont respectivement 3, 3 et 4. Chaque client peut recevoir sa demande de plus d'un centre et chaque centre peut servir plusieurs entrepôts. Les coûts de transport unitaire entre les centres et les clients sont donnés par le tableau 1. Tableau 1 : informations du problème :
Questions :

TABLE 7.2 – Acheminement

	CL1	CL2	Cl3	offre
C1	4	3	2	4
C2	3	5	4	5
C3	2	3	1	3
Demande	3	3	4	

1. Modéliser ce problème de distribution sous la forme d'un programme linéaire
2. Résoudre le problème en utilisant la méthode CNO et la méthode de la Balas et Hammer
3. La solution donnée par la méthode CNO est elle optimale, justifier ta réponse ?
4. Améliorer la solution obtenue para CNO en utilisant la méthode de stepping stone

Exercice 2 :
Pour améliorer la qualité du service rendu à ses clients, une entreprise de télécommunication cherche à installer m BTS pour couvrir convenablement n quartiers de la ville de Tétouan. Le nombre estimé d'abonnées par quartier est donné par nbHi pour chaque quartier i. La distance entre deux quartiers i et j est donnée par dij. Une BTS ouverte ne peut couvrir convenablement qu'au maximum k abonnées. La distance de couverture entre un abonné et la BTS d'affectation doit être inférieure à 2 km.
Questions :

1. Modéliser le problème de localisation des BTS sous la forme d'un programme linéaire
2. On suppose maintenant que M=6, n=6 et k=250. Les autres données du problème sont présentées dans le tableau 1. Modéliser ce problème sous la forme d'un problème de couverture totale. Résoudre ce problème en utilisant la méthode de chvatal, on supposant que si un client est couvert par deux BTS alors il sera affecté à la plus proche BTS. On suppose que la distance entre un client un quartier est égale à la distance entre le dite quartier et son quartier d'appartenance.
3. Si on suppose maintenant que le coût d'investissement total ne doit pas dépasser 20, la solution précédente restera valable ?

TABLE 7.3 – Localisation BTS

	Q1	Q2	Q 3	Q4	Q5	Q6
Q1	0	2.5	1.5	2.2	2	2.7
Q2		0	1.2	0.7	1.6	3.3
Q3			0	2.3	1.9	1.1
Q4				0	1.1	1.7
Q5					0	1.3
Q6						0
Nb abonnés	120	135	150	251	105	125
Coût d'installation	8	7	8.5	10	9	5.5

Problème du sac-à-dos Dans une camionnette de capacité W= 65, on cherche à transporter les produits de la table suivante. L'objectif de notre

TABLE 7.4 – Knapsack Problem

Produit	A	B	C	D	E
Wight	20	30	20	80	100
Value	100	30	15	20	10

transporteur est de maximiser la valeur de chaque voyage en respectant la capacité de la camionnette. On suppose dans un premier temps que les produits sont indivisibles.
Questions.

1. Modéliser le problème sous la forme d'un programme du sac- à dos 0/1
2. Résoudre le problème en utilisant la méthode de la programmation dynamique vue en cours.

On suppose maintenant que les produits peuvent être fractionnés.

1. Modéliser le problème sous la forme d'un programme mathématique linéaire
2. Résoudre le programme en utilisant la méthode des rendements vue en cours.

Problem VRP Une société de distribution localisée à la ville de Kenitra distribue des produits cosmétiques à huit de ses clients. L'entreprise cherche à minimiser la distance parcourue par sa flotte de véhicule afin de réduire les coûts de distribution. L'objectif principal consiste à trouver un plan de distribution qui minimise la distance totale parcourue. Le tableau 2 donne les distances entre les différents nœuds du réseau de distribution de cette entreprise. L'entreprise doit faire face aux restrictions suivantes :

— La longueur d'une tournée ne doit pas dépasser 500 km
— La flotte est homogène de capacité 10 tonnes avec une vitesse moyenne de déplacement de 70 km/h.
— Un chauffeur ne peut pas travailler plus de 8h par jour.

Les demandes des clients sont données par le tableau suivant : Questions :

TABLE 7.5 – Data VRP

Client	1	2	3	4	5	6	7	8
Demande (tonnes)	2	2	2	3	4	3	4	5

1. Modéliser ce problème sous la forme d'un programme linéaire
2. Résoudre le problème de l'entreprise en utilisant la méthode Clarke & Wright (version séquentielle)
3. Décomposer la séquence suivante en utilisant la méthode de Beasly (0-1-2-6-3-4-7-5-8)

Exercice 4 La société de distribution DistMa livre 5 de ses clients à partir d'un de ses entrepôts. Le tableau suivant résume les distances entre les différents points de distribution (dépôt, clients) ainsi que les demandes des clients. L'entreprise utilise des véhicules homogènes de capacité est égale à 10. Questions :

TABLE 7.6 – Data VRP

	0	1	2	3	4	5	demande
0	0	20	8	14	16	12	0
1		0	22	18	30	26	4
2			0	32	20	22	6
3				0	20	22	3
4					0	30	2
5						0	3

1. Modéliser le problème VRP par un modèle mathématique (expliquer chaque équation du modèle) sachant que notre objectif est de minimiser la distance maximale parcourue.
2. Résoudre le problème de l'entreprise DistMa en utilisant la méthode de la Margueritte version séquentielle.
3. On donne la séquence suivante 0 1 3 2 5 4 0 décomposez la en tournées en utilisant la méthode de Baisely.

Exercice 5 Une entreprise de distribution dispose de trois entrepôts, E1, E2, E3, leurs capacités respectives sont 100, 100 et 90 unités d'emballage d'un produit donné. L'entreprise assure l'approvisionnement de cinq clients, CL1, CL2, CL3, CL4 et CL5, leurs demandes respectives sont 40, 50, 70, 30 et 90 unités d'emballage. L'objectif du gérant de cette entreprise est d'assurer un approvisionnement de ses clients à coût minimal. Le coût de transport unitaire entre chaque entrepôt et chaque client est donné par le tableau 1. Questions :

TABLE 7.7 – Data VRP

	Cl1	CL2	CL3	CL4	CL5	disponibilités
E1	4	1	2	6	9	100
E2	6	4	3	5	7	100
E3	5	2	6	4	8	90
Demande	40	50	70	30	80	

1. Modéliser ce problème de distribution sous la forme d'un modèle mathématique linéaire

2. Résoudre ce problème de distribution en utilisant la méthode de stepping stone
3. Résoudre le problème en utilisant la méthode de Balas et Hammer

Bibliographie

[1] Michel Roux, "Entrepôts et magasins : Tout ce qu'il faut savoir pour concevoir une unité de stockage". 6ème édition, 2015, Eyrolls.

[2] Lionel Amodeo et Farouk Yalaoui, " Logistique interne : Entreposage et manutention", Technosup, ISBN 2-7298-2489-8. 2005, Ellipse.

[3] El fallahi, Abdellah & Prins, Christian& Calvo, Roberto. (2008). A memetic algorithm and a tabu search for the multi-compartment vehicle routing problem. Computers & Operations Research. 35. 1725-1741. 10.1016/j.cor.2006.10.006.