Titre: Title:	Extensions du problème de Weber
Auteur: Author:	Stéphane Krau
Date:	1997
Type:	Mémoire ou thèse / Dissertation or Thesis
Référence: Citation:	Krau, S. (1997). Extensions du problème de Weber [Thèse de doctorat, École Polytechnique de Montréal]. PolyPublie. https://publications.polymtl.ca/8964/

UNIVERSITÉ DE MONTRÉAL

EXTENSIONS DU PROBLÈME DE WEBER

STÉPHANE KRAU

DÉPARTEMENT DE MATHÉMATIQUES

ET DE GÉNIE INDUSTRIEL

ÉCOLE POLYTECHNIQUE DE MONTRÉAL

THÈSE PRÉSENTÉE EN VUE DE L'OBTENTION

DU GRADE DE PHILOSOPHIAE DOCTOR

(MATHÉMATIQUES

DE L'INGÉNIEUR)

AVRIL 1997

UNIVERSITÉ DE MONTRÉAL

ÉCOLE POLYTECHNIQUE DE MONTRÉAL

Cette thèse intitulée:

EXTENSIONS DU PROBLÈME DE WEBER

présentée par: KRAU Stéphane

en vue de l'obtention du diplôme de: Philosophiae Doctor

a été dûment acceptée par le jury d'examen constitué de :

M. BOURDEAU Marc, Ph.D., président

Mme JAUMARD Brigitte, T.Doctorat T.Hab., membre et directrice de recherche

M. HANSEN Pierre, D.Agr., membre et codirecteur de recherche

M. SAVARD Gilles, Ph.D., membre

Mme GUIGNARD-SPIELBERG Monique, T. d'État, membre externe

À mes parents

Jean-Claude et Blandine

REMERCIEMENTS

Un doctorat, c'est long... une vie dans la vie jalonnée d'incertains, de questionnements, de joies, d'ivresse et d'humilité. On devient docteur comme on naît acteur, un jour, sans trop y penser, par inattention, attrapé par une envie de découverte. Si l'acteur s'occupe de son âme pour bien paraître, le docteur, lui, recherche le raisonnement certain, l'argument sans failles. L'inspiration est leur maître d'oeuvre qui se nourrit dans cette vie immobile de la vie agissante. "Tout sort de ce temps silencieux, de ces heures négligées et de cette vie blanche. Le meilleur de nous arrive toujours à notre insu" (C. Bobin).

Je remercie tout particulièrement Pierre Hansen pour sa générosité, son écoute et sa curiosité dont il a fait montre à mon égard; Brigitte Jaumard pour m'avoir fait confiance sans retenues durant mon travail ainsi que pour ses qualités de directrice de thèse.

J'envoie mes salutations et remerciements à toutes les secrétaires du Gérad et du département de Mathématiques Appliquées pour leur gentillesse et leurs compétences.

Je n'oublie pas mes confrères de travail, ceux qui terminent leur mémoire ou thèse, en pensant aux discussions sans fins sur la condition d'étudiant, le comportement humain etc...

Je dédie cette thèse à mes parents Jean-Claude et Blandine Krau qui m'ont soutenu sans limites durant ces 5 ans et depuis 29 ans; à mes grands parents Edouard et Madeleine Krau; à Emmanuelle pour me supporter.

RÉSUMÉ

Le problème de Weber et ses extensions sont les plus étudiés en théorie de la localisation. On les classe en deux groupes:

(1) les problèmes à une seule source;

(2) les problèmes à plusieurs sources.

Pour les problèmes de la première classe, cela consiste à localiser une source afin de minimiser une fonction objectif reliant la source à des points préalablement fixés (appelés points de demande). Dans le premier chapitre, nous résolvons exactement le problème où les points de demande et la source se trouvent sur la sphère, lorsque la fonction objectif est la somme des distances (norme géodésique) pondérées de la source aux points de demande. Dans le second chapitre un algorithme optimal est proposé pour le problème de Weber dans le plan quand le coût reliant la source aux points de demande est une fonction concave en la distance. Nous donnons dans le troisième chapitre la première méthode exacte résolvant le problème de Weber simple dans le plan avec des contraintes de localisation (zones interdites pour la source) et des contraintes de passage (zones infranchissables).
Les problèmes de la seconde classe sont ceux où l'on doit localiser plusieurs sources auxquelles les points de demande sont rattachés. Le problème de Weber multi-sources dans le plan (minimiser la somme des distances pondérées des points de demande à la source la plus proche) avec et sans contraintes de localisation et de passage fait l'objet des deux derniers chapitres. Une méthode exacte basée sur la génération de colonnes permet dans le chapitre 4 de résoudre des problèmes de taille jamais encore approchée. Dans le dernier chapitre, une extension de cette méthode

permet de résoudre exactement et pour la première fois le problème de Weber multi-sources dans le plan avec des contraintes de localisation et de passage.

ABSTRACT

Weber's problem and its extensions has been extensively studied in continuous location theory. We can divide these problems in two groups:

(1) Problems with one source.

(2) Problems with many sources.

The first group of problems consists in the location of a point (called source) in order to minimize an objective function that depends on distances between the source and fixed points (called demand points). In the first chapter, we solve exactly Weber's problem on the sphere in which the objective function is the sum of weighted distances (geodesic norm) from the source to demand points. In the second chapter, an optimal algorithm is proposed for Weber's problem in the plane with a concave cost function. In chapter 3 we propose a first exact method for solving Weber's problem in the plane with forbidden regions either for location or for travel.
Problems of the second group are those which many sources have to be located in order to minimize the sum of distances between each demand point and the closest source. In the unconstrained case, multi-source Weber problems of large size are solved exactly in chapter four. In the fifth chapter, the constrained multi-source Weber problem (forbidden regions for location and/or to travel) is solved exactly for the first time. The solution method is a column generation approach embedded in a branch-and-bound scheme.

TABLE DES MATIÈRES

LISTE DES TABLEAUX

LISTE DES FIGURES

INTRODUCTION

Au $17^{\text{ième}}$ siècle Fermat énonce le problème suivant: étant donné trois points dans le plan, trouver un quatrième point tel que la somme des distances aux trois points donnés soit minimum. Peu après Toricelli donne la solution suivante: étant donné un triangle quelconque, le point optimal (celui qui minimise la somme des distances aux trois sommets du triangle) coïncide avec l'intersection des trois cercles circonscrits aux trois triangles équilatéraux adjacents au triangle initial. Heinen montrera cependant en 1834 que cette propriété n'est en général pas vérifiée pour des triangles ayant un angle supérieur ou égal à 120^o. Dans ce cas, le sommet correspondant à cet angle est le point optimal.

En 1750, Simpson dans "Doctrine and Application of Fluxions" associe le problème à trois points auxquels on a affecté des poids. Ce n'est qu'en 1909 que Weber [75] généralisera le problème dans son traité sur la théorie de la localisation : étant donné n points et un poids associé à chaque point, trouver un point du plan (appelé point de Weber) tel que la somme des distances pondérées aux n points (appelés points de demande) soit minimum. Une solution physique du problème est donnée par l'analogue de Varignon : une planche dans laquelle des trous (correspondant aux emplacements des points de demande) sont percés au travers desquels passent des ficelles au bout desquelles sont attachés des poids proportionnels à ces demandes. Les ficelles sont attachées ensemble et le point d'équilibre du noeud reliant toutes les ficelles est le point cherché (point de Weber).

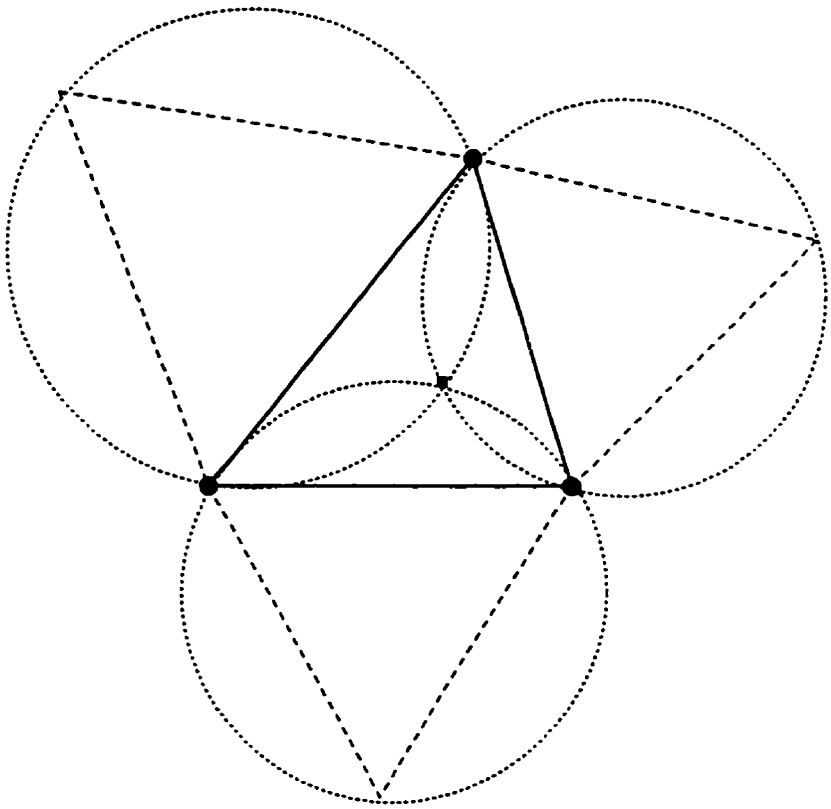


Figure 0.1 Triangles de Toricelli.

En 1937, Weiszfeld [76] propose la première méthode exacte de résolution du problème de Weber dans l'article: "Sur le point pour lequel la somme des distances de n points donnés est minimum" publié en français dans la revue japonaise Tôhoku Mathematics Journal. C'est une procédure itérative de descente basée sur les conditions nécessaires du premier ordre. L'existence de cet article ne sera découverte que 25 ans plus tard et la méthode sera finalisée par Kuhn [50] et Ostresh [59] dans les années soixante, en précisant la règle à adopter dans le cas où le point obtenu à une itération coïncide avec un point de demande. Depuis la formulation initiale de Fermat, le problème de Weber a connu plusieurs extensions:

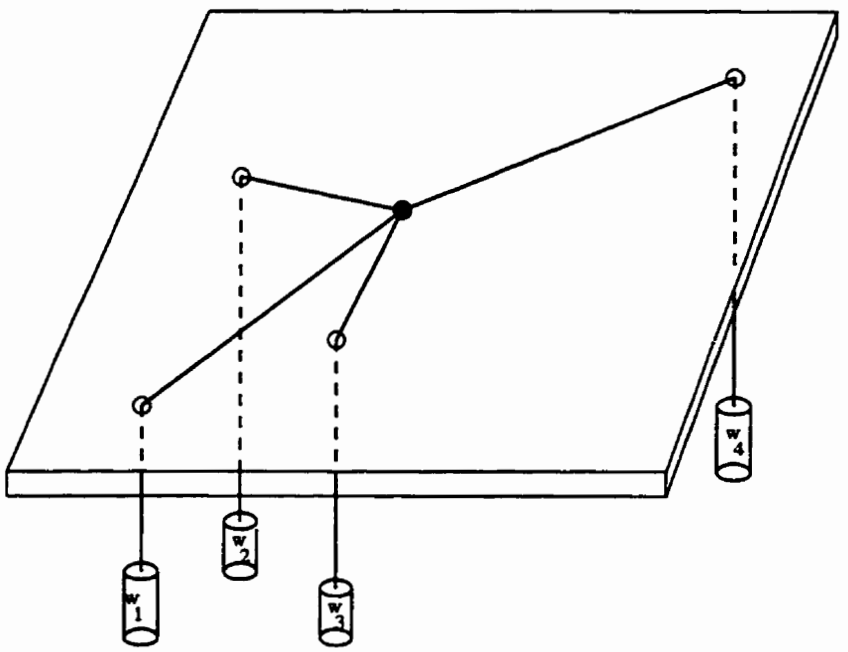


Figure 0.2 Analogue de Varignon.

- augmentation du nombre de points de demande;
- affectation d'un poids (éventuellement négatif) à chaque point de demande;
- localisation de plusieurs points de Weber minimisant la somme des distances aux points de demande (problème de Weber multi-sources et problème de Weber multiple);
- modifications de la fonction objectif.

Au cours du $20^{\text{ième}}$ siècle les économistes et les industriels se sont intéressés à plusieurs variantes du problème de Weber. Tout d'abord certains auteurs ont considéré différents coûts liant la source aux points de demande (lorsqu'on localise une usine de production desservant plusieurs villes, il est réaliste de considérer des coûts de transports qui ne sont pas forcément linéaires en la distance parcourue). D'autres auteurs ont étudié le problème de Weber sur la sphère dans lequel ils utilisent la

norme géodésique ce qui introduit une non linéarité dans la fonction objectif. Enfin, d'autres auteurs ont regardé certaines applications où des contraintes sur le lieu de la source (localisation d'une usine dans une région contenant des lacs ou autres endroits impraticables) et des contraintes de passage (obstacles infranchissables) apparaissent. Le problème de Weber, dans le milieu du $20^{\text{ième}}$ siècle, a été généralisé à plusieurs sources. On retrouve parmi les nombreuses applications de cette variante la localisation de plusieurs centres d'achats et de centres de secours desservant des villes d'une région en minimisant les coûts de transports. Il est nécessaire d'utiliser des techniques d'optimisation globale pour résoudre exactement plusieurs des variantes du problème de Weber. Nous allons proposer dans la thèse plusieurs méthodes exactes permettant de résoudre efficacement cinq variantes du problème de Weber. Une revue de la littérature concernant la variante considérée sera proposée dans chaque chapitre. Nous allons présenter ici l'objet de chaque chapitre et les résultats obtenus.

Chapitre 1: le problème de Weber sur la sphère

L'utilisation de la norme géodésique introduit une non convexité et une non concavité dans la fonction objectif. Il n'existe dans la littérature qu'un seul algorithme exact pour ce problème, peu efficace en pratique. Nous proposerons un premier algorithme exact permettant de résoudre rapidement (dans le cas contraint et non contraint) des problèmes de taille $n = 500$ lorsque les points de demande sont répartis uniformément autour de la sphère, de taille $n = 1000$ lorsque les points de demande sont répartis sur 2/3 de la sphère, ou sur un seul hémisphère.

Chapitre 2: le problème de Weber avec coût concave

Dans sa formulation il ne diffère du problème de Weber dans le plan que par la particularité de sa fonction objectif. Nous considérerons les coûts de transport des points de demande à la source comme des fonctions concaves et non décroissantes en la distance parcourue. Peu d'auteurs se sont penchés sur ce problème et il n'existe pas de méthodes spécifiques dans la littérature. C'est un problème d'optimisation globale que nous résolvons exactement en comparant une approche par programmation D.C. (différence de fonctions convexes) et une approche par énumération et séparation. Des problèmes de taille $n = 10000$ sont traités en des temps raisonnables par les deux méthodes.

Chapitre 3 : le problème de Weber avec contraintes de localisation et de passage

Deux types de contraintes sont présentes:

- des régions du plan sont interdites à la localisation de la source.
- des régions du plan sont des obstacles de passage que l'on doit contourner.

C'est un problème de minimisation d'une fonction non linéaire non convexe sur un domaine non convexe. Dans la littérature seul le problème de Weber avec des contraintes de localisation est résolu optimalement. En revanche lorsque des contraintes de passages sont ajoutées, on a seulement recours à des heuristiques. Nous développons dans ce chapitre la première méthode exacte (méthode par énumération et séparation) lorsque de telles contraintes existent. Des problèmes de taille pouvant aller jusqu'à $n = 1000$ avec 12 contraintes de localisation et de passage, des problèmes de taille $n = 100$ avec 100 contraintes de localisation et de passage sont résolus.

Chapitre 4 : le problème de Weber multi-sources

L'objectif est de trouver la localisation de p sources afin de minimiser la somme des distances pondérées des n points de demande à leur source la plus proche. Le problème est classé NP-dur. Bien que plusieurs méthodes exactes aient été développées, nous proposons une nouvelle approche par la technique de génération de colonnes dont le problème auxiliaire est un problème d'optimisation globale (résolu par une approche par programmation D.C.). Cette approche a la particularité de résoudre des problèmes jamais résolus jusqu'alors et d'être plus efficace que les méthodes existantes lorsque $p > 2$. Sa principale caractéristique étant que pour un n donné le temps de calcul diminue lorsque le nombre p de sources à localiser (nombre de variables) augmente. Des problèmes de taille $n = 150, p = 5, 50$ sont résolus.

Chapitre 5 : le problème de Weber multi-sources avec contraintes de localisation et de passage

Aucun algorithme exact n'a été proposé à ce jour. Ce chapitre regroupe les techniques développées dans les chapitres 3 et 4 pour former le premier algorithme exact de la littérature. Une approche différente de celle exposée au chapitre 4 concernant la résolution du problème auxiliaire permet de considérer des problèmes (dans le cas non contraint) de taille $n = 287, p \geq 25$. Dans le cas contraint des problèmes de taille $n = 30, p = 2, 7$ avec 12 régions interdites à la localisation et au passage sont résolus. Les quatre premiers chapitres peuvent être lus indépendamment des autres à quelques exceptions près pour lesquelles des références aux autres chapitres sont précisées. Le chapitre 5 doit être lu après les chapitres 3 et 4.

CHAPITRE 1

Le problème de Weber sur la sphère

1.1 Introduction

Le problème de Weber est un des plus étudiés en théorie de la localisation. Il consiste, dans sa formulation de base à déterminer un point (appelé source) qui minimise la somme des distances pondérées entre ce point et n points donnés (appelés points de demande).

Ce problème qui a été résolu dans le plan (distance euclidienne) par Weiszfeld en 1937 [76], a suscité de nombreuses variantes depuis. Leurs principales formulations ainsi que leurs résolutions se retrouvent dans les travaux de Hansen *et al.* [41]; Love, Morris et Wesolowsky [52]; Wesolowsky [78]. Ce problème d'optimisation est convexe et très bien résolu. La procédure de Weiszfeld [76] (méthode itérative de descente) est une des plus utilisées. Elle s'appuie sur les conditions d'optimalité du premier ordre pour une fonction convexe à deux variables et en déduit une procédure de descente dont le pas est calculé en fonction du point courant. La fonction objectif n'est pas dérivable en les points de demande et la forme initiale de la méthode de Weiszfeld ne tient pas compte de cette singularité (Kuhn [50]). Ostresh [59] modifie donc la procédure de Weiszfeld et recalcule le pas de descente afin d'assurer la convergence de l'algorithme.

Le problème de Weber sous la forme précédemment définie connaît bon nombre d'applications à caractère industriel: localiser une usine de production afin de minimiser le coût de transport vers des centres de distribution, localiser un magasin desservant plusieurs villes. Il devient moins réaliste lorsque les distances considérées sont grandes. La distance euclidienne, dans un tel cas, approxime mal la distance (appelée distance géodésique) entre 2 points de la surface du globe: localisation d'un aéroport ou d'une base militaire couvrant une large zone sur la surface de la terre.

Ce chapitre traite du problème de Weber (distance géodésique) sur la surface de la sphère. L'emploi de la distance géodésique rend le problème généralement non convexe et non concave (à la différence du plan où il y a convexité lorsque la distance euclidienne est utilisée). La résolution exacte du problème de Weber (distance géodésique) est du domaine de l'optimisation globale.

Drezner et Wesolowsky [29], Katz et Cooper [48] sont à notre connaissance les premiers à fournir des études sur ce problème. Une revue des premiers résultats et des premières propriétés est donnée par Wesolowsky [77].

Des résultats plus récents se trouvent dans le livre de Love, Morris et Wesolowsky [52] ainsi que dans l'article de Plastria [64]. Plusieurs heuristiques et un algorithme exact résolvent le problème. Dressons en premier lieu le portrait des principales heuristiques.

La voie classique à la résolution du problème de Weber dans le plan est la procédure itérative de Weiszfeld. C'est une méthode de descente de plus forte pente avec un pas dépendant de la position du point courant. Le point final vérifie les conditions d'optimalité du premier ordre. La convergence est assurée en modifiant le calcul du pas de descente lorsque le point courant se trouve sur un point de demande, lieu où la fonction objectif n'est pas dérivable (Ostresh [59], Brimberg

et Love [9]). Drezner et Wesolowsky [29], Katz et Cooper [48], Dhar et Rao [20] ont proposé plusieurs extensions de la procédure de Weiszfeld à la sphère. Elles ne garantissent cependant pas l'obtention d'un minimum local. Dans des cas particuliers ces méthodes donnent de bons résultats: Katz et Cooper [48] montrent une convergence linéaire vers un minimum local lorsque tous les points de demande sont sur un même hémisphère et que le point initial est suffisamment proche d'un minimum local.

Tsai, Chen et Lin [71] présentent une méthode qui vérifie si les points de demande se situent ou non sur un même hémisphère. Par une série de rotations d'un grand cercle (intersection d'un plan passant par le centre de la sphère et la surface de la sphère) autour du centre de la sphère, la preuve de l'existence ou non d'un tel hémisphère est établie.

Dhar et Rao [48] s'intéressent à la comparaison de trois normes de distances (norme géodésique, une approximation de la distance géodésique au carré, une approximation de la distance géodésique). Drezner et Wesolowsky [29] font un travail similaire en approximant la norme de la distance géodésique au carré. Ces études permettent dans certains cas d'obtenir un bon point de départ à une méthode de descente. Aly *et al.* [1] montrent sur un exemple avec 6 points de demande que l'utilisation de la norme euclidienne génère une erreur de 18% sur la valeur optimale de la fonction de Weber avec la norme géodésique.

Litwhiler et Aly [51] suggèrent deux méthodes itératives sans calculs de dérivées. Dans la première des deux méthodes, les points de demande sont projetés dans le plan tangent à la sphère au point courant. Un problème de Weber dans le plan est alors résolu et le point optimal est projeté sur la sphère. Ce point est pris comme point courant pour l'itération suivante. La procédure se répète jusqu'à ce que 2 points consécutifs sur la sphère soient suffisamment proches l'un de l'autre. La

seconde méthode procède par descentes successives sur une longitude puis sur une latitude. Elle se termine si aucune direction de descente n'est trouvée. Aucune de ces deux méthodes ne garantit l'obtention d'un minimum local.

Dhar et Rao [20] développent une méthode itérative, dans le cas où des contraintes sur le lieu de la source existent. Partant d'une solution du problème non contraint, la recherche d'un minimum local se poursuit sur les frontières des zones interdites.

Nous achevons le parcours des différentes méthodes par le seul algorithme exact, proposé par Drezner [27]. Cet algorithme utilise la propriété que la pente de la fonction objectif est bornée par la somme des poids (supposés tous positifs) des points de demande. La fonction objectif satisfait une condition de Lipschitz avec une constante de Lipschitz égale à la somme des poids. Des raffinements successifs d'une borne inférieure sur la fonction objectif conduisent à l'obtention (à ϵ près) du minimum global:

(1) évaluer la fonction objectif en plusieurs points choisis arbitrairement;

(2) construire en chacun de ces points un cône dont l'axe passe par le centre de la sphère;

(3) recherche du point minimisant (minimum global) la fonction minorante résultante de la somme de tous les cônes;

(4) évaluer en ce point la valeur de la fonction objectif;

(5) construire un nouveau cône minorant en ce point;

(6) itérer la procédure jusqu'à ce que la différence entre la meilleure valeur connue et la borne inférieure soit suffisamment petite.

La recherche du minimum global de la fonction minorante est un problème difficile. Il peut être formulé comme un problème de "minimax": recherche d'un point dont la plus grande distance d'un point à un ensemble de points d'un ensemble donné est minimum. Sur la sphère, le problème du "minimax" est équivalent au problème du "maximin" : recherche d'un point pour lequel la distance au point le plus proche d'un ensemble de points donnés est maximum. Drezner propose de résoudre le sous-problème par un algorithme de Drezner et Wesolowsky [30]. L'algorithme de Drezner [26] peut aussi être utilisé. Ces deux procédures sont similaires et résolvent optimalement le problème du minimax qui est en général non convexe sur la sphère. La première de ces deux méthodes comprend une méthode de descente basée sur le principe de la méthode de Demjanov [18] qui assure l'obtention d'un minimum local et la valeur correspondante de la fonction objectif. Cette valeur détermine ensuite le rayon des cercles sphériques centrés sur chaque point de demande. Une procédure spécifique détermine l'intersection de tous les cercles sphériques. Si l'intersection se réduit à un ou plusieurs points, l'algorithme s'arrête et un optimum global du sous problème est trouvé (il correspond à un point de valeur minimum). Dans le cas où l'intersection des cercles sphériques est composée d'une ou de plusieurs régions disjointes, une méthode de descente est appliquée à partir d'un point d'une de ces régions et la procédure se répète.

L'algorithme de Drezner [27] est une application au problème de Weber sur la sphère de l'algorithme de Piyavski [62] pour l'optimisation Lipschitzienne. Propriétés, variantes et comparaisons de calcul de cet algorithme ainsi que de ses variantes et d'autres algorithmes pour l'optimisation Lipschitzienne sont présentés par Hansen et Jaumard [37]; Gourdin, Hansen et Jaumard [39]. Il apparaît que ces problèmes avec seulement 2 variables semblent difficiles à résoudre lorsque la constante de Lipschitz (resp. le domaine de résolution) devient grande (resp. étendu).

L'extension du problème de Weber à plusieurs sources n'a été que très peu étudiée sur la sphère. Des heuristiques sont données par Aykin et Babu [5] et Dhar et Rao [20].

Certaines propriétés du problème de Weber ont été explorées lorsque les points de demande sont distribués uniformément sur la sphère. Drezner [28] s'intéresse à la probabilité d'avoir un minimum local sur un point de demande. Katz et Cooper [48] établissent des conditions nécessaires et suffisantes pour avoir un minimum local sur un point de demande. Aly *et al.* [1], [51] présentent des conditions suffisantes pour que l'optimum global soit à l'intérieur de l'enveloppe convexe des points de demande. Drezner [25] prouve que si les points de demande sont sur un grand cercle, un des points de demande est un optimum global.

Le but de ce chapitre est de présenter un algorithme exact et efficace pour le problème de Weber sur la sphère avec (il sera dans ce cas noté problème CWS) et sans contraintes (problème WS) sur le lieu de la source. Des problèmes avec 500 points de demande et plus sont résolus en temps raisonnable. Ce nouvel algorithme est dans ses grandes lignes une adaptation de l'algorithme (BSSS) "Big Square Small Square" développé dans le plan par Hansen *et al.* [43] pour le problème de Weber avec des fonctions coûts non décroissantes en la distance. Plusieurs améliorations majeures de cet algorithme d'énumération implicite sont étudiées et permettent une réduction considérable du temps de calcul.

Le modèle et les principales propriétés du problème de Weber sur la sphère sont présentés dans la section 2. L'algorithme est décrit dans la section 3. Les calculs de bornes sont discutés à la section 4.

1.2 Formulation du problème et propriétés

Les hypothèses du problème de Weber sur la sphère (WS) sont les suivantes:

(1) Soit S une sphère de rayon unité. Un point sur S est définie par ses coordonnées sphériques (ϕ, θ) avec $(0 \leq \phi \leq \pi)$, $(-\pi \leq \theta \leq \pi)$.

(2) La distance géodésique entre 2 points a et b sur la sphère est donnée (Donnay [23]) par:

$$d(a,b) = \arccos(\sin\phi_a \sin\phi_b \cos(\theta_a - \theta_b) + \cos\phi_a \cos\phi_b).$$

(3) Soit $J = \{a^1, a^2, ...a^n\}$ un ensemble de points de demande. $a^j \in S$.

(4) La source est associée au point $s \in S$.

(5) Un poids w_j est affecté à chaque point de demande a^j $(a^j \in S)$; w_j représente une quantité à fournir au point de demande a^j.

(6) s^* est le point qui minimise la fonction de Weber sur S:

$$F^* = F(s^*) = \min_{s \in S} F(s) = \min_{s \in S} \sum_{j=1}^{n} w_j d_j(s).$$

Où $d_j(s) = d(a^j, s)$ représente la distance géodésique entre le point de demande a^j et s.

(7) Q_h : quadrilatère sur la sphère.

(8) s_h : point réalisable de Q_h.

(9) s_h^* : localisation optimale de la source à l'intérieur et sur la frontière de Q_h.

(10) $F_h^* = \min_{s \in Q_h} \sum_{j=1}^{n} w_j d_j(s)$.

Dans le cas où des contraintes sur le lieu de la source existent, une hypothèse supplémentaire est nécessaire (problème de Weber avec contraintes (CWS)).

(12) Le point s (et éventuellement les points de demande) doit appartenir à un sous ensemble R de S (pas nécessairement convexe ni connexe). Le sous ensemble peut être approximé avec une précision suffisante par un ensemble de m triangles sphériques T_ℓ: $R = \bigcup_{\ell=1}^{m} T_\ell$.

Les distances sur une sphère possèdent des propriétés qui n'ont pas leurs équivalents dans le plan. La connaissance des principales d'entre elles et de définitions est nécessaire à la compréhension du domaine.

Propriété 1 *La somme des distances d'un point sur la sphère à un autre point et à l'antipode de ce dernier est égal à π.*

Propriété 2 *Un point sur la sphère minimise la fonction de Weber si et seulement son antipode la maximise.*

Propriété 3 *Un point de poids w que l'on remplace par son antipode de poids $-w$ ne modifie pas la localisation optimale de la source. Tout problème de Weber avec poids mixtes est équivalent à un problème de Weber avec poids positifs. Dans le cas du plan cette remarque ne s'applique pas. Le problème de Weber dans le plan avec poids mixtes a été étudié entre autres par Drezner et Wesolowsky [31], Chen et al. ([16]) et peut s'écrire comme un problème de minimisation d'une fonction D.C. (différence de fonctions convexes).*

Definition 1 *Un cercle sphérique centré en un point et de rayon r est l'ensemble des points éloignés du centre (plus petit arc joignant le centre aux points) par un arc de longueur r.*

Un cercle sphérique divise la sphère en deux parties. Un point se trouve à l'intérieur du cercle si et seulement si il est du même côté que le centre du cercle.

Definition 2 *Un domaine sur la surface de la sphère est convexe si pour toute paire de points appartenant au domaine, le plus petit arc joignant ces deux points est inclus dans le domaine.*

Definition 3 *Une fonction f définie sur un ensemble convexe C de la surface de la sphère est convexe si et seulement pour toute paire de points a et b de C et s_1, s_2 appartenant au plus petit arc entre a et b on a : $F(\lambda s_1 + (1-\lambda)s_2) \leq \lambda F(s_1) + (1-\lambda)F(s_2)$ avec $\lambda \in [0,1]$.*

Propriété 4 *La distance d'un point donné p à un point sur une sphère est une fonction convexe à l'intérieur d'un cercle de rayon $\pi/2$ et centré en p.*

La propriété 4 mentionné par Katz et Cooper [48] et Drezner et Wesolowsky [29] conduit au cas particulier suivant:

Propriété 5 *Si tous les points de demande (avec poids positifs) sont à l'intérieur d'un cercle sphérique de rayon $\pi/4$ alors la fonction objectif est convexe et possède un unique minimum local qui est global.*

Quand la propriété 5 est vérifiée, le problème de Weber se résout aisément par une extension à la sphère de la procédure de Weiszfeld [76] [Drezner et Wesolowsky [29] et Katz et Cooper [48]].

1.3 Résolution par énumération implicite

1.3.1 Idée générale de l'algorithme

L'algorithme est une généralisation à la sphère de (BSSS) (Hansen *et al.* [43]). Nous l'appellons (BRSR) "Big Region Small Region". C'est une méthode d'énumération implicite pour un problème d'optimisation continue qui fonctionne en 6 étapes:

(1) Partitionner la sphère S en régions Q_h délimitées chacune par 2 latitudes et 2 longitudes (le partitionnement initial est composé de 8 régions égales).

(2) Les régions qui ne sont pas réalisables (cas contraint) sont éliminées.

(3) Calculer sur chaque région une borne inférieure $\underline{F}_h$ de F_h^* et éliminer les régions pour lesquelles la borne inférieure est plus grande ou égal à F^*.

(4) Calculer la fonction de Weber en un point réalisable de chaque région et effectuer la mise à jour de F^* et s^* si nécessaire.

(5) Choisir la région ayant la plus petite borne inférieure et la partitionner en 4 nouvelles régions.

(6) Recommencer les tests sur les nouvelles régions Q_h obtenues jusqu'à ce que l'erreur relative $\frac{F^* - \underline{F}_h}{F^*}$ soit plus petite qu'une valeur ϵ.

Notons que $F(s) - \underline{F}_h$ tend vers 0 quel que soit $s \in Q_h$ lorsque le diamètre de Q_h (c'est à dire le plus grand arc sphérique de Q_h) tend vers 0. La convergence de l'algorithme est assurée en utilisant les résultats généraux de la méthode d'énumération implicite [Horst et Tuy [46]].

La méthode (BRSR) utilise une stratégie de type meilleur-d'abord et plusieurs moyens de calculer les bornes $\underline{F}_h$ sont discutés dans les prochaines sections. Les

étapes détaillées de l'algorithme sont:

a) Initialisation

$Q_1 \leftarrow S$;

$H \leftarrow \{1\}$ (H est l'ensemble des indices des problèmes non résolus)

$H_{new} \leftarrow \{1\}$ (H_{new} est l'ensemble des indices des problèmes pour lesquels la borne inférieure n'a pas encore été calculée);

$s^* \leftarrow$ point arbitrairement choisi dans R (si un tel point existe, sinon $s^* \leftarrow \infty$, i.e., une valeur conventionnelle);

$F^* \leftarrow F(s^*)$ si un point dans R a été trouvé, sinon $F^* \leftarrow \infty$;

b) Test de réalisabilité

Pour tout Q_h tel que $h \in H_{new}$ faire

calculer $Q_h \cap R$;

si $Q_h \cap R = \emptyset$ éliminer h de H_{new};

Fin pour;

c) Test d'optimalité

Pour tout Q_h tel que $h \in H_{new}$ faire

calculer une borne inférieure $\underline{F}_h$ sur F_h^*;

si $\underline{F}_h \geq F^*$ éliminer h de H_{new};

Fin pour;

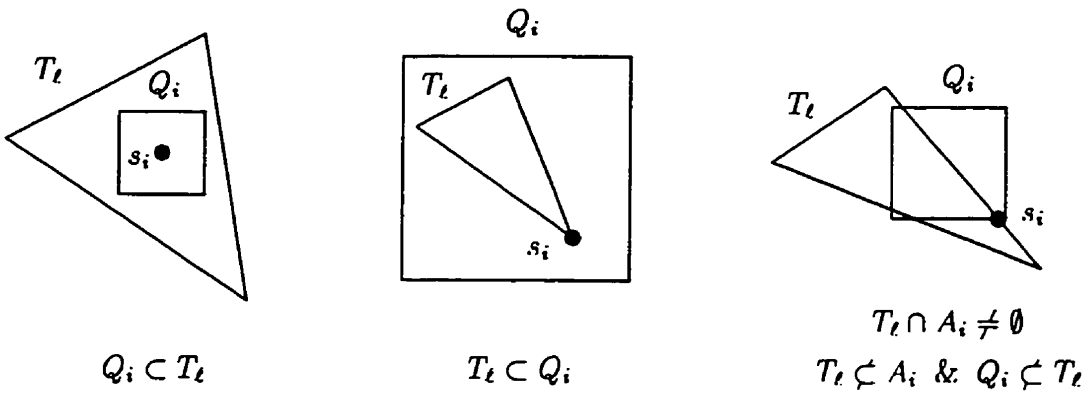


Figure 1.1 Obtention des points dans le cas contraint.

d) Test d'amélioration de la solution (voir Figure 1.1)

Pour tout Q_h tel que $h \in H_{new}$ faire

s'il existe $\ell \in \{1, 2, , \ldots, m\}$, $Q_h \subset T_\ell$,

calculer la valeur $F(s_h)$ au point central s_h de la région Q_h;

s'il existe $\ell \in \{1, 2, \ldots, m\}$, $T_\ell \subset Q_h$,

calculer la valeur $F(s)$ en un point extrême arbitrairement choisi s_h de T_ℓ;

sinon, s'il existe $\ell \in \{1, 2, \ldots, m\}$ tel que $T_\ell \cap Q_h \neq \emptyset$,

calculer la valeur $F(s_h)$ d'un point s_h sur les frontières de T_ℓ et Q_h;

si $F(s_h) < F^*$, poser $F^* \leftarrow F(s_h)$ et $s^* \leftarrow s_h$;

Fin pour;

ajouter tous les indices $h \in H_{new}$ à H;

e) Branchement et conditions d'arrêt

Si $H = \emptyset$, fin: le problème est non réalisable; sinon sélectionner Q_h tel que $\underline{F}_h = \min_{k \in H} \underline{F}_k$;

si $\frac{F^* - \underline{F}_h}{\underline{F}_h} \leq \varepsilon$: fin, une solution ε-optimale s^* de valeur F^* a été trouvée, sinon partitionner Q_h en 4 nouvelles régions $Q_{h'}$, $h' = h1, h2, h3, h4$;
Éliminer h de H et poser H_{new} égal à l'ensemble des indices des nouvelles régions;
Retour en *b)*.

Pour rendre l'étape *e)* efficace, les bornes inférieures $\underline{F}_h$ des sous problèmes ($h \in H$) sont gardées dans une liste. A l'étape *d)*, les sous problèmes tel que $\underline{F}_h \geq F^*$ peuvent être éliminés pour sauver de l'espace mémoire. Ceci est réalisé en utilisant une gestion efficace de la liste (Atkinson *et al.* [3]). Dans le cas non contraint (WS), l'étape *b)* est supprimée et l'étape *d)* est réduit au premier cas (voir figure 1.1).

Chaque région Q_h est défini par une paire de latitudes et une paire de longitudes. Les 4 nouvelles régions engendrées sont obtenues en prenant l'arc passant par le milieu des 2 latitudes et l'arc passant par le milieu des 2 longitudes.

1.4 Bornes inférieures

1.4.1 Borne 1

Une première façon de calculer une borne inférieure $\underline{F}_h$ noté $\underline{F}_h^1$ sur $F(s)$ pour $s \in Q_h$x[est de sous estimer les distances des points de demande à la source s par leurs distances à leurs points le plus proche p_j de Q_h (voir figure 1.2):

$$\underline{F}_h^1 = \sum_{j=1}^{n} d_j(p_j).$$

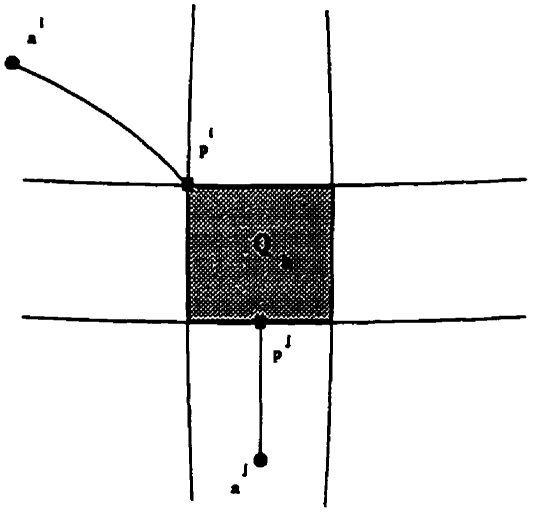


Figure 1.2 Détermination dans Q_h des points les plus proches des points de demande.

Ce calcul de borne inférieure se retrouve dans l'algorithme (BSSS). La détermination des points p_j pour chaque point de demande a^j est plus complexe sur la sphère. A chaque région Q_h, la sphère se divise en 9 zones. La région Q_h, les 4 zones latérales et les 4 zones de coins. Lorsque a^j se trouve dans une zone de coin, le point $p_j \in Q_h$ le plus près de a^j est le coin de Q_h appartenant à la zone. Pour a^j se situant dans une zone latérale, p_j est le point d'intersection entre la frontière de Q_h la plus proche de a^j et sa perpendiculaire passant par a^j. Si $a^j \in Q_h$, p_j est confondu avec a^j et ne contribue pas au calcul de $\underline{F}_h^1$. Nous allons décrire les 9 zones précisément et expliciter le calcul de p_j pour a^j dans chacune des zones.

Considérons une zone Q_h $(q^1q^2q^3q^4)$ sur un hémisphère délimité par 2 longitudes LO_1 et LO_2, 2 latitudes LA_1 et LA_2. La sphère étant initialement partitionnée en 8 triangles sphériques égaux, l'inégalité $\|LO1 - LO2\| < \pi/2$ est satisfaite. Soient C_1 et C_2 les grands cercles contenant LO_1 et LO_2 respectivement. Soit une zone liée à l'hémisphère nord $(q^1q^2q^3q^4)$ (le cas où la zone est liée à l'hémisphère sud est équivalent) et le partitionnement de la sphère (à partir de $(q^1q^2q^3q^4)$) en 9 zones NE, NO, SE, SO, N, S, O, E et Q_h (illustrées par les figures 1.3 et 1.4):

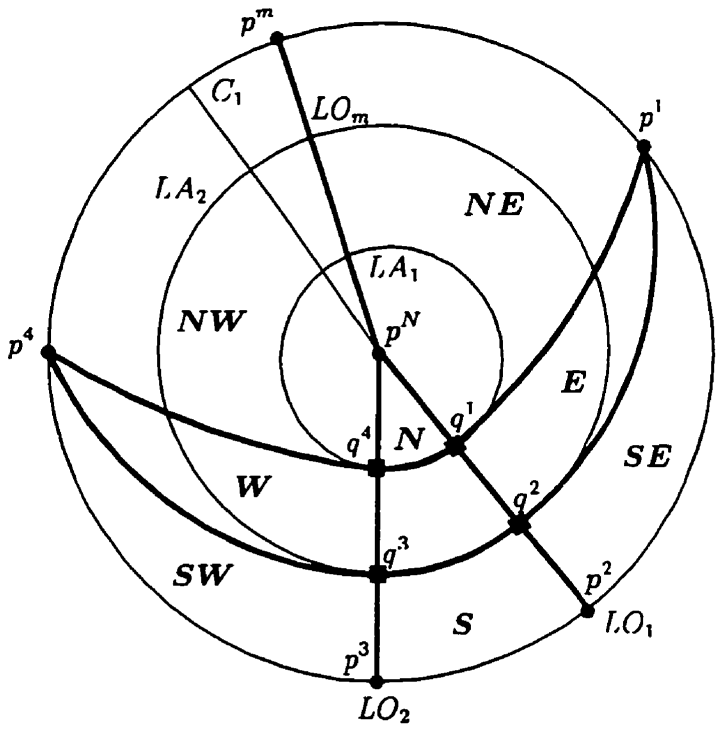


Figure 1.3 Hémisphère nord.

NE zone $(p^1q^1p^Np^mp^5)$.

- p^1 : point correspondant au pôle nord si C_1 est l'équateur.
- p^N: pôle nord de la sphère.
- p^m: point d'intersection de l'équateur avec LO_m (la longitude opposée à celle passant au milieu des arcs sphériques (q^1q^4) (q^2q^3)).
- p^5 : le point d'intersection de LO_m avec le grand cercle issue de p_1 et passant au milieu de (q^1q^2). Les angles sphériques $((p^1p^5),(p^1q^1))$ et $((p^1p^5),(p^1q^2))$ sont plus petit que π.

NO zone $(p^4q^4p^Np^mp^5)$.

- p^4: point correspondant au pôle nord si C_2 est l'équateur.

- p^5: ce point, défini précédemment pour la zone NE, est aussi le point d'intersection de LO_m avec le grand cercle issu de p^4 passant dans le milieu de (q^3q^4). Les angles sphériques $((p^4p^5),(p^4q^4))$ et $((p^4p^5),(p^4q^3))$ sont plus petits que π.

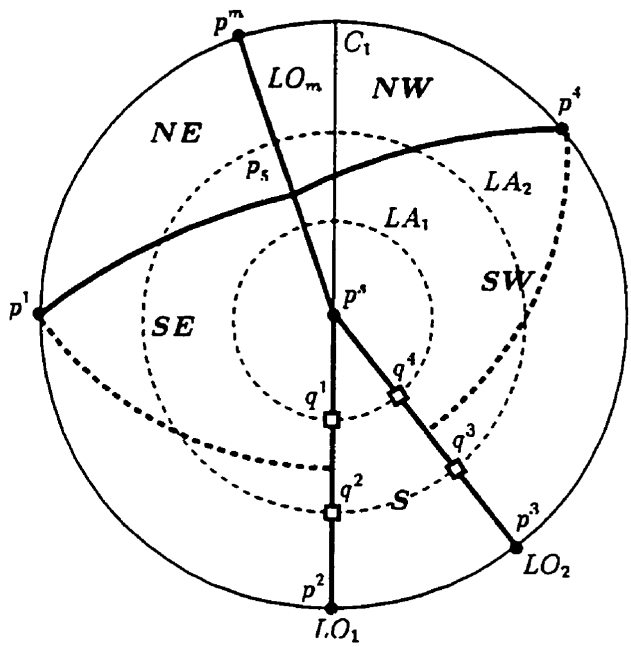


Figure 1.4 Hémisphère sud.

SE zone $(p^1q^2p^2p^Sp^5)$.

- p^2 : point d'intersection entre LO_1 et l'équateur.
- p^S : pôle sud de la sphère.

SO zone $(p^4q^3p^3p^Sp^5)$ ..

- p^3 : point d'intersection entre LO_2 et l'équateur.

N zone $(p^Nq^1q^4)$.

S zone $(q^2q^3p^3p^Sp^2)$.

$\boldsymbol{E}$ zone $(p^1q^1q^2)$.

$\boldsymbol{O}$ zone $(p^4q^3q^4)$.

Le lemme suivant explique comment le point le plus près p_j sur $(q^1q^2q^3q^4)$ du point de demande a^j est obtenu.

Lemme 1 *Soit $a^j(\phi_j, \theta_j)$ un point sur S et LA_p une latitude quelconque. Soit $p(\phi, \theta)$ un point courant sur LA_p et $p_h(\phi_p, \theta_p)$ la projection orthogonale de a^j sur LA_p. La fonction distance $d_j(p)$ croît quand $(\theta - \theta_j)$ augmente de 0 à π et décroît quand $(\theta - \theta_j)$ augmente de π à 2π. Si LA_p est un grand cercle et a^j est sur LA_p, la fonction distance $d_j(p)$ est linéaire. De plus, il correspond au cas où la pente s_j de la fonction distance est maximum (en valeur absolue) et est égal à 1.*

Démonstration:
Le signe de la première dérivée de $d_j(p) = \arccos(\sin(\phi_j)\sin(\phi_{p_h})\cos(\theta - \theta_j) + \cos(\phi_j)\cos(\phi_{p_h}))$, i.e.,

$$\frac{\sin(\phi_j)\sin(\phi_{p_h})\sin(\theta-\theta_j)}{\sqrt{1-(\sin(\phi_j)\sin(\phi_{p_h})\cos(\theta-\theta_j)+\cos(\phi_j)\cos(\phi_{p_h}))^2}}$$

est égal au signe de $\sin(\theta - \theta_j)$ car $0 \leq \phi_j \leq \frac{\pi}{2}$ et $0 \leq \phi_{p_h} \leq \frac{\pi}{2}$.

Dans le cas particulier (figure 1.5 (b)) où LA_p est vue comme l'équateur d'un pôle nord donné, la fonction distance $d_j(p)$ devient linéaire:

$$\arccos(\sin(\pi/2)\sin(\pi/2)\cos(\theta - \theta_j) + \cos(\pi/2)\cos(\pi/2)) = \theta - \theta_j.$$

Montrons que la pente de la fonction distance est bornée par 1 (voir figure 1.6). Soit le taux d'accroissement de $d_j(p)$ au point p sur LA_p pour une petite variation de θ:

$$\frac{\delta d_j(p)}{\delta \theta} = \frac{d_j(p + \delta p) - d_j(p)}{\delta \theta}.$$

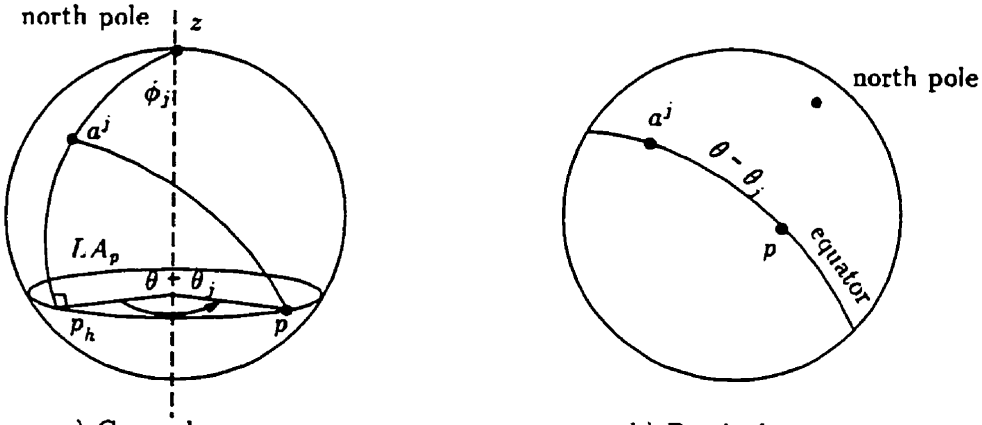


Figure 1.5 Calcul de la distance géodésique entre deux points.

Montrons que la pente de la fonction distance est bornée par 1 (voir figure 1.6): soit le taux d'accroissement de $d_j(p)$ au point p sur LA_p pour une petite variation de θ:

$$\frac{\delta d_j(p)}{\delta \theta} = \frac{d_j(p+\delta p) - d_j(p)}{\delta \theta}.$$

En utilisant l'inégalité triangulaire sphérique nous avons $d_j(p+\delta p) - d_j(p) < \delta p = r\delta\theta$ où

$$r \leq R(R = 1 : \text{ rayon de la sphère }).$$

Par conséquent:

$$s_j = \lim_{\delta\theta \to 0} \frac{\delta d_j(p)}{\delta \theta} < r \leq 1.$$

Dans le cas particulier où $d_j(p) = 0$ et LA_p est un grand cercle

$$s_j = \frac{\delta d_j(p)}{\delta \theta} = \frac{\delta p}{\delta \theta} = R = 1.$$

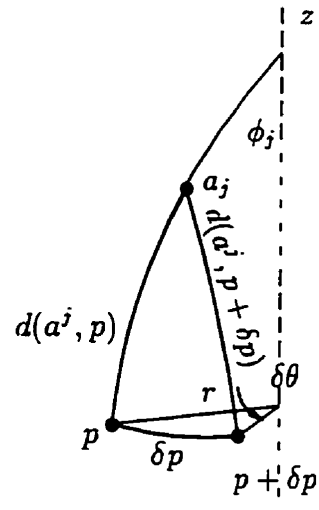


Figure 1.6 Effet sur $\delta d_j(p)$ d'une petite variation de θ.

Quand $d_j(p)$ décroît, le même raisonnement conduit à la borne inférieure -1. Le résultat s'en suit.

Proposition 1 *Le point le plus proche sur* $(q^1q^2q^3q^4)$ *d'un point de demande est respectivement* q^1, q^4, q^3, q^2 *dans NE,NO,SO,SE.*

Démonstration:

Soit a^j un point de NE. Montrons en premier lieu que q^1 est le plus près de a^j sur (q^1q^2). Puisque l'angle sphérique $[(p^1p^5)(p^1q^1)]$ est plus petit que π (voir définition de p^5), la distance $d_j(C1)$ augmente jusqu'au milieu de (q^1q^2). En effet, le point milieu est l'antipode de la projection orthogonale de p^5 sur C_1, la distance $d_j(C_1)$ ne décroîtra qu'après ce point. D'où le résultat. Le même raisonnement (en considérant LA_1) montre que q^1 est le point le plus proche de a^j sur (q^1q^4). Tout

point p intérieur à la zone $(q^1q^2q^3q^4)$ est dominé en termes de distance au point de demande a^j par l'intersection de l'arc du grand cercle passant par a^j et p et un des 2 arcs (q^1q^2), (q^1q^4). On en déduit que q^1 est le point le plus proche de $(q^1q^2q^3q^4)$ de tout point de NE. Le même raisonnement s'applique pour les autres zones de coin NO, SO, SE.

Proposition 2 *Le point le plus proche sur* $(q^1q^2q^3q^4)$ *de tout point de demande est sa projection orthogonale sur* $(q^1q^2q^3q^4)$.

Démonstration:

Appliquer le lemme 1 directement en considérant les projections orthogonales sur
$(q^1q^4), (q^2q^3), (q^1q^2), (q^3q^4)$ des points de demandes des zones N,S,E,O.

La borne $\underline{F}_h^1$ est peu précise si le rayon de Q_h est relativement étendu et lorsque les points de demande sont répartis sur toutes les zones autour de à Q_h (les distances en un point de la région sont minorées par les distances à la région). Dans la section suivante nous verrons comment améliorer la qualité de $\underline{F}_h^1$.

1.4.2 Borne 2

Examinons d'abord les conditions sous lesquelles le minimum global de la fonction de Weber soit sur un côté de Q_h.

Proposition 3 *s Soit $\bar{Q}_h$ la zone délimitée par $\bar{q}^1, \bar{q}^2, \bar{q}^3, \bar{q}^4$ les antipodes respectifs de q^1, q^2, q^3 et q^4. Soit a^j un point de demande de fort poids w_j dans une zone donnée (en dehors de Q_h et $\bar{Q}_h$) et un sous ensemble de points de demande a^i, a^k, a^l, a^m dans d'autres zones dont la somme des poids w_i, w_k, w_l, w_m ne dépasse pas w_j. La solution optimale du problème de Weber associé aux points de demande (a^i, a^k, a^l, a^m) avec $s \in Q_h$ est sur le côté de Q_h le plus proche de a^j.*

Démonstration:

Soit un point courant $p(\phi, \theta)$ de Q_h et l'arc de grand cercle (p, a^j) allant de p à a^j. Comme a^j est en dehors de $\bar{Q}_h$, la fonction distance $d_j(p)$ décroît linéairement lorsque p parcourt (p, a^j) en direction de a^j. La distance entre a^j, a^k, a^l, a^m et p peuvent croître ou décroître (ou faire les deux). D'après le lemme 1, les pentes s_j, s_i, s_k, s_l et s_m associées à ces distances satisfont:

$$s_j = 1 \text{ et } s_i, s_k, s_\ell, s_m < 1.$$

Donc

$$w_i s_i + w_k s_k + w_\ell s_\ell + w_m s_m < w_i + w_k + w_\ell + w_m < w_j = w_j s_j.$$

Lorsque l'on parcourt le grand cercle de p à a^j, la fonction objectif du problème de Weber associée aux points de demande a^j, a^j, a^k, a^l, a^m décroît. Cette fonction atteint sur Q_h son minimum en p', (i.e, le point d'intersection d'un côté de Q_h et

du grand cercle considéré). De plus, tout point p de Q_h est dominé par un point p' sur le côté de Q_h le plus proche de a^j. Remarquons que si a^j se situe sur une région latérale (N,S,O,E), il y a un seul côté à considérer, il y en a 2 si a^j se trouve sur une des zones de coin (NE,SE,NO,SO).

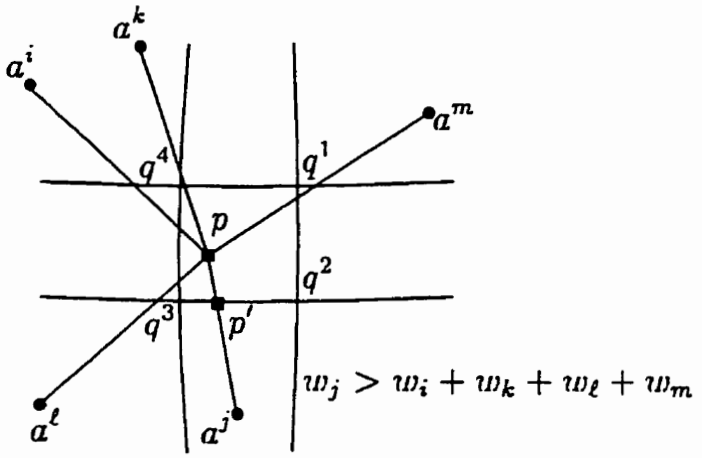


Figure 1.7 Illustration de la borne 2.

Le problème de Weber reste cependant non convexe lorsque s est restreint à se trouver sur un arc de Q_h.

Un moyen de contourner cette difficulté est de grouper un point de demande a^j associé au poids maximum en dehors de Q_h et $\bar{Q}_h$ avec le plus grand nombre de points a^j, a^k, a^l, a^m de poids total inférieur à w_j, se situant dans une zone opposée ou dans plusieurs des zones les plus éloignées lorsque la zone opposée est vide.

Éliminer successivement les points de demande et itérer la procédure avec le prochain point de demande de plus fort poids. On obtient alors un partitionnement $A_1, A_2, ...A_k$ de l'ensemble A des points de demande où le dernier ensemble A_k des points de demande est soit vide ou contient seulement des points de Q_h ou $\bar{Q}_h$. La borne $\underline{F}_h^2$ est alors obtenue en résolvant les problèmes de Weber associés aux points de demande dans A_t pour $t = 1, ..., k-1$, en suivant la procédure décrite ci-dessous.

On obtient:

$$\underline{F}_h^2 = \sum_{t=1}^{k-1} g(A_t) + \underline{F}_h^1(A_k).$$

$\underline{F}_h^1(A_k)$ est la valeur de la première borne pour les points de demande liés à A_k.

Le calcul de $g(A_t)$ s'effectue comme suit:

(1) Diviser le côté de Q_h considéré en plusieurs sous arcs (e.g. 4) $[c, d]$.

(2) Sous estimer la valeur de la fonction objectif de Weber pour l'ensemble A_t sur [c,d] en prenant pour chaque point de demande dans A_t la plus petite distance à c ou d, ou le pied de la perpendiculaire au côté de Q_h s'il appartient au sous arc.

(3) Garder la plus petite évaluation pour tous les sous arcs comme borne inférieure de la fonction de Weber sur le côté de Q_h (voir figure 1.8).

(4) Si a^j est dans une zone latérale, $g(A_t)$ est égale à la borne obtenue par les étapes précédentes. Si a^j est dans une zone de coin, itérer la procédure (1-3) pour l'autre côté de Q_h le plus proche de a^j et affecter à $g(A_t)$ la plus petite norme des deux côtés.

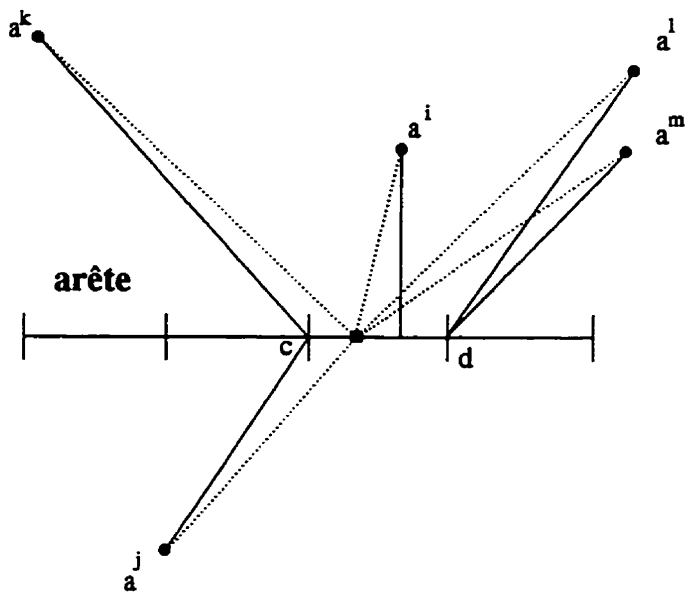


Figure 1.8 Borne sur la fonction objectif de Weber associée à un groupe de points de demande lorsque p est contraint à être sur un côté de Q_h.

1.4.3 Borne 3:

Au regard de la propriété 5, une autre façon de calculer une borne inférieure est déduite. Les points de demande sont partitionnés en 2 ensembles: un ensemble A_1 contenant tous les points de demande dont la distance maximum à un point de Q_h est inférieur à $\pi/2$ et un ensemble A_2 contenant les points de demande restant (voir figure 1.8). Le problème (WS) pour les points de demande de A_1 est convexe et peut être résolu en utilisant un logiciel de minimisation d'une fonction convexe tel que MINOS. Un borne $\underline{F}_h^3$ est obtenu en appliquant aux points de demande dans A_2 la borne $\underline{F}_h^1$:

$$\underline{F}_h^3 = \sum_{j \in A_2} d_j(p_j) + \min_{s \in Q_h} \sum_{j \in A_1} d_j(s).$$

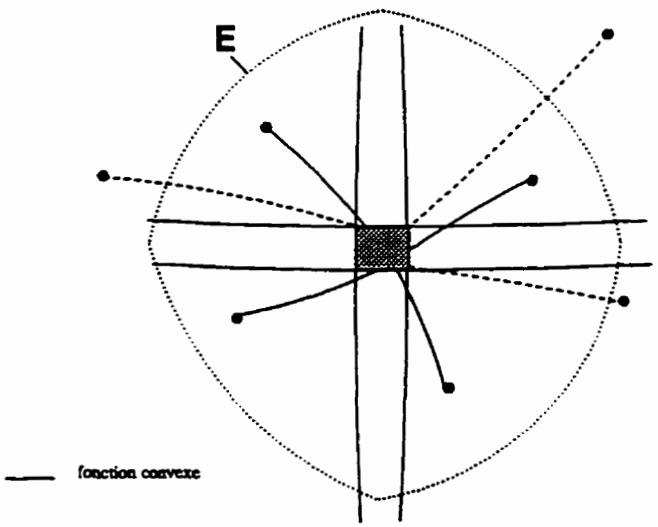


Figure 1.9 Illustration de la borne 3.

1.4.4 Borne 4

Une variante de la borne 3 est l'utilisation pour l'ensemble A_2 de la procédure pour le calcul de $\underline{F}_h^2$. Notons que lorsque la région Q_h tend vers un point (par l'effet du branchement), les bornes $\underline{F}_h^1$ et $\underline{F}_h^4$ tendent vers une valeur commune.

1.5 Résultats numériques

L'algorithme (BRSR) pour le problème de Weber décrit dans les sections précédentes a été programmé en C et testé sur SPARC 4/75-64 (28.5 mips, 64 M ram, 4.2 Mflops). Pour les problèmes tests, les points de demande sont générés aléatoirement suivant une loi de distribution uniforme, sur la sphère, sur les 2/3 de la sphère et sur l'hémisphère. Les poids sont distribués uniformément sur l'intervalle [0,2]. Une première série d'expériences vise à comparer les bornes inférieures décrites sur des problèmes de taille relativement petite (ne dépassant pas 100 points de demande).

Pour chaque valeur de n (nombre de points de demande), 10 problèmes sont résolus. L'algorithme s'arrête lorsque l'erreur relative $(F^* - \underline{F}_h)/F^*$ devient inférieure à $\epsilon = 10^{-3}$. Les résultats sont présentés dans les tables 1-3. Pour chaque valeur de n, la moyenne et l'écart type du temps de calcul et des itérations sont donnés. Une seconde série de problèmes tests donne une évaluation sur la taille des problèmes pouvant être résolus en temps raisonnable avec la meilleure des bornes (i.e borne $\underline{F}_h^2$ pour toute la sphère, $\underline{F}_h^4$ pour les 2/3 de la sphère, $\underline{F}_h^3$ pour l'hémisphère). Pour chacun des cas, des problèmes avec resp. 500, 1000, 10000 points de demande sont résolus. Pour chaque valeur de n, 10 problèmes sont testés. L'erreur relative ϵ est de 10^{-3}. Les résultats de la seconde série d'expériences sont présentés dans les tables 1-3:

(i) Le temps de calcul augmente modérément avec la taille du problème, à peu près linéairement avec le nombre de points de demande.

(ii) La borne $\underline{F}_h^2$ est meilleure que les autres pour des problèmes générés sur toute la sphère. Avec cette borne inférieure, l'algorithme résout les problèmes avec 100 points de demande en moins de 9 minutes en moyenne. L'utilisation des autres bornes augmente le temps moyen de 3 minutes en moyenne.

(iii) Les bornes $\underline{F}_h^3$ et $\underline{F}_h^4$ deviennent plus efficace quand la taille des régions contenant les points de demande décroissent. Pour des séries de problèmes sur 2/3 de la sphère, $\underline{F}_h^3$ et $\underline{F}_h^4$ ont presque la même valeur que $\underline{F}_h^2$. La qualité des bornes $\underline{F}_h^1$ et $\underline{F}_h^2$ ne dépend pas de la taille des régions où se trouvent les points de demande.

(iv) Les écarts types des nombres d'itérations et du temps de calcul sont élevés pour toutes les séries de problèmes. Cela montre que la distribution géographique des points de demande a une forte influence sur la performance de l'algorithme.

Par exemple, un problème avec 10 points de demande sur un hémisphère est résolu en 157 secondes tandis que les 9 autres de la même série sont résolus en moins de 7 secondes.

(v) A l'exception du cas sur l'hémisphère, la borne $\underline{F}_h^2$ est plus précise que $\underline{F}_h^1$, $\underline{F}_h^4$ est plus précise que $\underline{F}_h^3$. Le groupement de points est donc efficace en pratique.

(vi) Dans le cas de l'hémisphère, la borne $\underline{F}_h^3$ est meilleure que $\underline{F}_h^4$. Dans ce cas, la qualité obtenue en groupant des points de demande ne compense pas le temps nécessaire pour l'obtenir.

Une troisième série d'expérience évalue la sensibilité de l'algorithme par rapport au paramètre ϵ (erreur relative sur la fonction objectif). 5 problèmes avec 200 points de demande sur toute la sphère, 2/3 de la sphère et sur l'hémisphère sont résolus pour $\epsilon = 10^{-3}$ à $\epsilon = 10^{-5}$ (10^{-6} pour l'hémisphère). Les résultats sont donnés dans le tableau 4:

(i) L'algorithme est hautement sensible à la valeur de ϵ dans tous les cas.

(ii) Quand ϵ est divisé par 10, le nombre d'itérations augmente d'un facteur de 10 pour les points de demande sur la sphère et sur 2/3 de la sphère et par un facteur de 2 à 5 pour les points de demande sur l'hémisphère. Dans ce dernier cas le facteur augmente quand ϵ décroît.

(iii) Le temps de calcul augmente en général du même facteur que le nombre d'itérations.

Au vu de ces résultats, il apparaît que le moyen économique pour avoir une solution ϵ optimale est de résoudre rapidement le problème avec une précision modérée $\epsilon = 10^{-2}$, $\epsilon = 10^{-3}$ puis d'appliquer une procédure itérative [e.g Katz et Cooper (1980), Drezner et Wesolowsky (1978)].

Le cas contraint prend place dans la $4^{\text{ième}}$ et dernière série d'expérimentations. L'effet des contraintes de localisation est évalué. Un exemple où les 5 continents sont approximés par 24 triangles sphériques est donné. Les points de demande et la source sont contraints à être à l'intérieur de ceux-ci. La borne $\underline{F}_h^3$ est utilisée. Les résultats pour n=100-1000 sont présentés dans la table 5:

(i) Tous les problèmes sont résolus en des temps raisonnables.

(ii) Le nombre moyen d' itérations apparaît être indépendant du nombre des points de demande. Son écart type décroît quand le nombre de points de demande augmente.

(iii) la moyenne du temps de calcul augmente légèrement moins que linéairement avec le nombre de points de demande. Son écart type ne change pas avec ce nombre. Par conséquent le ratio σ/μ décroît.

(iv) Les problèmes contraints apparaissent être du même ordre de difficulté que les problèmes non contraints lorsque les points de demande sont générés sur 2/3 de la sphère mais plus facile à résoudre lorsque ces derniers sont générés sur toute la sphère.

En résumé, les résultats de calcul montrent que le problème de Weber sur la sphère est résolu efficacement par l'algorithme (BRSR) dans le cas contraint et non contraint.

Tableau 1.1 Résultats sur la sphère (1).

n	valeur opt.	borne inf. $\underline{f}^1$				borne inf. $\underline{f}^2$			
		iter		cpu		iter		cpu	
		μ	σ	μ	σ	μ	σ	μ	σ
10	12.7	1118.1	1116.9	18.1	17.3	454.7	617.1	22.3	29.5
20	25.0	4807.7	4462.0	153.7	148.9	1055.0	1202.2	102.7	117.1
30	38.7	4822.4	3897.5	223.4	179.0	747.1	475.3	110.7	72.6
40	51.8	5910.4	3808.8	356.4	229.6	793.4	336.0	158.0	66.9
50	70.8	5157.9	2757.2	392.4	214.5	659.0	360.0	163.0	90.2
60	83.9	5372.5	2718.1	484.4	249.1	672.6	431.4	197.2	122.9
70	99.0	7110.2	2673.7	766.6	289.7	1086.9	421.2	367.4	138.6
80	115.0	8035.5	3177.3	967.2	377.3	943.1	349.7	371.5	139.7
90	130.9	11178.4	3856.2	1529.1	515.8	1387.9	517.1	611.2	237.9
100	141.8	8374.7	2146.3	1247.6	319.3	1055.6	336.0	517.2	165.3
200	294.5	*	*	*	*	955.4	342.4	958.3	347.9
300	449.8	*	*	*	*	1758.0	688.5	2586.7	1045.6
400	605.6	*	*	*	*	1548.0	525.2	3018.3	985.2
500	757.7	*	*	*	*	2116.3	685.2	5233.0	1690.1

Tableau 1.2 Résultats sur la sphère (2).

n	valeur opt.	borne inf. $\underline{f}^3$				borne inf. $\underline{f}^4$			
		iter		cpu		iter		cpu	
		μ	σ	μ	σ	μ	σ	μ	σ
10	12.7	300.1	266.1	52.7	45.7	205.6	152.3	41.3	30.1
20	25.0	1123.9	1001.4	216.8	195.4	659.5	642.4	162.0	157.6
30	38.7	1090.2	815.5	245.0	181.5	716.1	611.1	215.5	182.4
40	51.8	1717.8	1352.7	439.9	331.8	917.6	730.1	322.9	250.0
50	70.8	1701.4	940.6	475.1	261.4	882.4	478.1	348.3	193.2
60	83.9	1882.9	935.7	576.2	280.2	907.5	469.6	408.6	204.7
70	99.0	2207.5	861.1	700.4	256.0	964.2	524.8	461.3	248.1
80	115.0	2566.1	944.9	919.8	358.3	1070.8	390.0	589.4	218.0
90	130.9	3969.0	1817.1	1489.4	717.3	1617.1	939.2	951.6	562.2
100	141.8	2485.7	621.4	1031.9	246.5	1118.6	95.6	720.3	58.1

Tableau 1.3 Résultats sur 2/3 de la sphère (1).

n	valeur opt.	borne inf. $\underline{f}^1$				borne inf. $\underline{f}^2$			
		iter		cpu		iter		cpu	
		μ	σ	μ	σ	μ	σ	μ	σ
10	6.8	2504.5	1329.5	38.5	21.6	1057.9	749.9	53.5	38.7
20	14.4	6832.6	3923.5	202.5	119.2	1595.9	782.2	151.4	74.9
30	22.7	5632.0	3318.4	246.0	148.5	1068.1	687.8	149.4	96.9
40	31.5	4834.9	1950.6	279.1	115.4	944.2	484.9	171.4	89.1
50	36.5	7824.3	5687.3	564.6	420.4	1323.4	851.9	300.7	195.9
60	45.8	5722.2	1995.2	487.0	176.1	890.4	332.6	243.8	92.9
70	54.9	6039.2	1801.7	599.7	180.8	978.0	240.5	320.2	78.3
80	62.6	10412.8	3472.9	1194.1	403.2	1655.0	605.5	619.3	234.0
90	71.4	9690.3	4516.0	1248.8	591.3	1573.6	704.9	641.2	291.7
100	80.0	10929.9	4688.4	1557.3	680.8	1735.7	717.8	788.0	327.6

Tableau 1.4 Résultats sur 2/3 de la sphère (2).

n	valeur optimale	borne inf. $\underline{f}^3$				borne inf. $\underline{f}^4$			
		iter		cpu		iter		cpu	
		μ	σ	μ	σ	μ	σ	μ	σ
10	8.9	227.0	189.3	40.1	32.6	160.6	154.1	31.6	29.6
20	17.6	192.8	180.3	38.7	35.5	142.9	161.3	33.4	36.7
30	31.2	760.0	1021.2	164.9	213.4	554.4	825.9	149.6	213.1
40	41.4	691.1	520.1	173.9	127.4	314.5	212.0	102.2	66.0
50	52.1	705.2	643.7	189.9	167.2	343.7	281.6	124.7	102.0
60	62.0	723.7	360.4	222.0	104.9	316.7	198.1	127.5	72.6
70	74.8	615.4	315.8	199.8	100.1	272.9	217.1	122.4	91.9
80	87.6	879.8	547.5	307.5	182.9	306.7	131.2	151.1	61.9
90	95.5	576.1	478.9	216.0	169.6	269.6	326.5	144.0	165.7
100	103.5	853.3	487.4	326.2	175.5	336.2	187.9	188.6	99.3
200	208.4	*	*	*	*	268.6	133.3	277.9	130.8
300	310.6	*	*	*	*	270.0	139.8	419.1	196.5
400	407.0	*	*	*	*	241.6	67.4	480.9	121.4
500	507.9	*	*	*	*	229.5	71.2	584.1	167.8
600	615.8	*	*	*	*	231.1	35.5	688.7	107.8
700	733.4	*	*	*	*	249.0	55.5	838.0	170.7
800	840.8	*	*	*	*	231.2	31.9	919.6	130.0
900	946.7	*	*	*	*	236.5	37.7	1066.0	141.4
1000	1038.3	*	*	*	*	230.3	34.2	1153.4	196.7

Tableau 1.5 Résultats sur l'hémisphère (1).

n	valeur optimale	borne inf. $\underline{f}^1$				borne inf. $\underline{f}^2$			
		iter		cpu		iter		cpu	
		μ	σ	μ	σ	μ	σ	μ	σ
10	8.9	1834.4	1510.1	30.3	24.7	643.5	609.9	31.5	30.1
20	17.6	3094.8	2363.5	98.1	77.0	734.9	477.9	69.6	46.1
30	31.2	5717.0	4069.4	269.4	195.2	1160.8	946.0	163.7	133.4
40	41.4	5148.4	3150.3	320.2	200.1	961.5	652.8	178.7	123.7
50	52.1	6761.3	4440.8	526.5	352.4	1265.0	930.6	296.2	222.0
60	62.0	7830.5	4989.2	727.7	473.1	1355.7	839.3	375.7	235.2
70	74.8	5937.5	1723.0	637.3	190.1	974.3	357.4	314.2	117.0
80	87.6	7404.6	2961.1	912.9	375.8	1194.3	614.5	441.4	229.4
90	95.5	5444.5	1923.3	752.6	268.4	869.1	388.8	359.5	161.9
100	103.5	8154.3	2972.4	1256.8	467.2	1444.3	618.5	666.1	287.5

Tableau 1.6 Résultats sur l'hémisphère (2).

n	valeur opt.	borne inf. $\underline{f}^3$				borne inf. $\underline{f}^4$			
		iter		cpu		iter		cpu	
		μ	σ	μ	σ	μ	σ	μ	σ
10	6.8	106.3	256.8	19.1	46	79.3	183.2	15.7	36.4
20	14.4	66.9	110.6	14.6	23.5	64.7	110.7	15.0	25.0
30	22.7	57.0	75.8	14.8	18.6	53.1	75.4	15.0	19.3
40	31.5	32.6	17.5	9.8	4.8	29.1	17.4	10.1	5.4
50	36.5	43.4	16.0	15.0	5.5	42.1	15.9	16.7	6.0
60	45.8	61.1	50.4	21.2	16.0	57.2	44.9	23.6	16.9
70	54.9	54.1	42.0	21.2	14.3	51.4	40.2	24.2	16.5
80	62.6	56.6	23.2	25.8	9.7	55.8	23.0	29.7	11.2
90	71.4	33.0	10.8	16.8	5.0	31.5	11.1	19.2	7.1
100	80.0	46.1	8.6	25.0	4.0	45.0	7.8	29.0	4.4
1000	815.8	41.0	.8	200.7	4.5	*	*	*	*
2000	1563.3	52.9	6.4	415.1	63.2	*	*	*	*
3000	2339.1	54.0	6.4	595.1	103.2	*	*	*	*
4000	3125.4	56.4	6.1	881.0	149.1	*	*	*	*
5000	3904.7	58.0	5.4	1064.7	219.2	*	*	*	*
10000	7835.3	61.0	1.0	2265.9	322.6	*	*	*	*

Tableau 1.7 Sensibilité à ϵ.

ε	sphère				2/3 sphère			
	iter		cpu		iter		cpu	
	μ	σ	μ	σ	μ	σ	μ	σ
10^{-2}	174.2	55.4	168.1	53.9	25.4	3.8	33.2	5.6
10^{-3}	1981.6	708.5	1955.8	698.9	136.8	52.6	194.5	58.8
10^{-4}	12548.6	5779.8	12577.4	5940.9	1804.0	1093.0	1840.9	1146.4
10^{-5}	16552.0	1639.0	15872.2	1724.7	13948.5	3154.4	14149.2	3610.3

ε	hémisphère			
	iter		cpu	
	μ	σ	μ	σ
10^{-2}	24.2	4.8	21.0	4.0
10^{-3}	43.2	7.4	38.8	6.7
10^{-4}	106.5	68.0	89.4	58.1
10^{-5}	450.0	567.1	341.6	427.2
10^{-6}	2287.0	3126.8	1550.9	2125.1

Tableau 1.8 Cas contraint.

n	valeur	Borne inf. $\underline{F}_h^3$			
	optimale	iter		cpu	
		μ	σ	μ	σ
100	82.0	77.0	30.5	56.4	19.4
200	169.8	103.4	20.0	104.2	17.9
300	243.5	88.2	17.3	119.0	21.0
400	332.2	102.0	24.4	168.8	33.8
500	408.5	90.6	13.9	184.5	21.8
600	485.9	85.4	7.5	199.9	14.8
700	570.0	83.5	12.2	229.1	26.0
800	651.3	76.7	10.1	243.8	26.0
900	744.0	79.5	10.0	280.6	26.7
1000	812.4	85.2	7.0	333.6	22.0

CHAPITRE 2
Le problème de Weber avec coûts concaves

2.1 Introduction

Le modèle classique de Weber présente un certain nombre d'approximations pas toujours réalistes. Par exemple, la fonction économique est généralement linéaire ou convexe en la distance alors qu'il est bien connu des économistes que le coût de transport s'exprime mieux par une fonction concave et non décroissante en la distance. De plus il n'est pas rare qu'en pratique des obstacles (villes, montagnes, lacs) interdisent la localisation et forcent à les contourner. Ces deux contraintes rendent le problème non linéaire et non convexe, avec une fonction objectif discontinue sur un domaine non convexe [11]. Il arrive aussi que pour certaines applications les demandes ou les points de demande soient connus de façon probabiliste. La fonction objectif devient alors stochastique.

Pour résoudre un bon nombre de variantes du problème de Weber, il est nécessaire de recourir à des outils d'optimisation globale. Le problème de Weber avec poids mixtes (où les usagers trouvent la source désirable, d'où des poids positifs, et les résidents proches la trouvent polluante, d'où des poids négatifs) et le problème de Weber avec distances maximum (où les fonctions de coûts cessent de croître après une distance donnée) sont des problèmes d'optimisation globale résolus exactement par Chen *et al.* [15] et Tuy *et al.* [73] [74] en utilisant la théorie des fonctions D.C. (différences de fonctions convexes). Hansen *et al.* [43] et Tuy *et*

al. [73] [74] considèrent les problèmes très généraux où la fonction économique est une somme de fonctions non décroissantes de la distance à chaque usager, celle ci pouvant être non symétrique.

2.2 Formulation du problème dans le plan

A notre connaissance le problème de Weber avec fonctions de coûts concaves et non décroissantes en la distance de chaque usager à la source n'a pas encore été étudié. C'est un problème d'optimisation globale qui peut être formulé comme suit:

$$F^* = F(x^*, y^*) = \min_{(x,y)\in S} F(x,y) = \sum_{j=1}^{n} f_j(d_j(s))$$

Où:

(i) S : enveloppe convexe des points de demande.

(ii) $J = \{a^1, ..., a^n\}$: ensemble de points de demande.

(iii) w_j : poids associé au point de demande a^j.

(iv) $s = (x, y)$: coordonnées cartésiennes du point courant.

(v) $f_j(x)$: fonction concave et non décroissante en x, associée à a^j.

(vi) $d_j(s)$: distance Euclidienne entre le point de demande a^j et la source s.

(vii) Q_h : carré du plan.

(viii) F_h^* : valeur optimale de la fonction objectif à l'intérieur de Q_h.

(ix) s_h^* : point associé à F_h^*.

(x) $\underline{F_h}$: borne inférieure sur F_h^* à l'intérieur de Q_h.

Definition 4 *On appelle enveloppe convexe des points de demande le plus petit polygone convexe qui les contient tous.*

Propriété 6 *Au moins un optimum* (x^*, y^*) *appartient à l'enveloppe convexe des points de demande.*

Preuve:

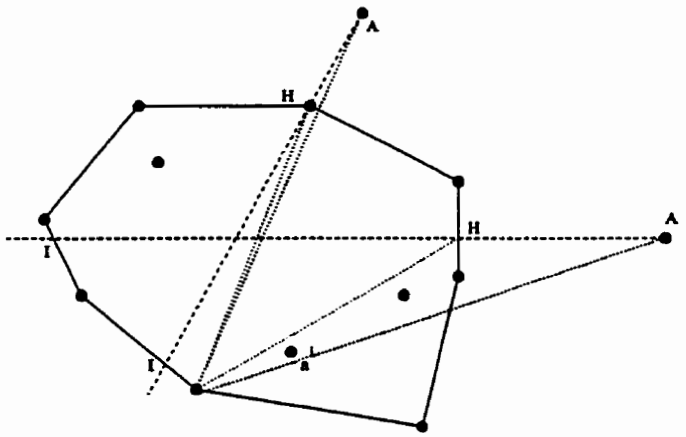


Figure 2.1 Enveloppe convexe des points de demande

Soit un point $A(x_A, y_A)$ n'appartenant pas à l'enveloppe convexe des points de demande. Supposons que ce point soit l'optimum : $F^* = \sum_{j=1}^{n} f_j(d_j(A))$ est minimum pour tout point du plan. Soit $H(x_H, y_H)$ le projeté orthogonal de A sur S (cf figure 2.1). Soit Δ la droite passant par A et H. Soit I l'autre point d'intersection entre la droite Δ et S. D'après la construction d'un polygone convexe (figure 2.1), on a pour tout j:

$$d_j(H) < d_j(A)$$

$$f_j(d_j(H)) \leq f_j(d_j(A))$$

Ce qui contredit l'hypothèse sur le point A.

Le problème de Weber avec coûts concaves et non décroissants en la distance est du domaine de l'optimisation globale. Nous proposons de le résoudre exactement par deux méthodes : un algorithme d'énumération implicite et un algorithme de minimisation de fonctions D.C. (différences de fonctions convexes).

Le chapitre comportent trois parties : la première porte sur la méthode d'énumération implicite pour laquelle deux procédures de calcul de bornes inférieures sont proposées. La seconde porte sur la résolution du problème par une approche D.C. La dernière partie présente les résultats numériques des différentes méthodes.

2.3 Approche par énumération implicite

L'algorithme proposé est schématiquement le même que "Big Square Small Square" développé par Hansen *et al.* [43] et étendu par Plastria [63] auquel on a ajouté un nouveau calcul de bornes. Sous sa forme initiale "BSSS" [43], [63] est un algorithme exact pour toute fonction de coût non décroissante en la distance. Il reste cependant limité en pratique car les bornes qu'il utilise sont souvent peu précises. L'idée de

base est de partitionner l'espace des solutions (le plan) en régions. Sur chacune d'elles, une borne inférieure et une borne supérieure de la fonction objectif sont calculées. A chaque itération la région ayant une borne inférieure minimum est choisie et est subdivisée (recherche du type meilleur d'abord [43] [63]). La meilleure localisation connue est mise à jour si nécessaire et l'algorithme réitère jusqu'à ce que la différence entre le minimum des bornes inférieure et la meilleure valeur connue soit inférieure à une petite valeur. Les principales étapes de la méthode (simplifiée ici) car elle permet aussi de prendre en compte des contraintes de zones interdites pour la localisation) sont:

Initialisation soit Q_1 le plus petit carré contenant les points de demande $\{a^1...a^n\}$ dont les côtés sont parallèles aux axes coordonnés. Soit L la longueur du côté de ce carré et ϵ la tolérance. Calculer au centre du carré la valeur de la fonction objectif. Soit F^* la plus petite valeur de la fonction objectif obtenue et s^* le point associé. Calculer une borne inférieure de la valeur de F sur le carré. Introduire le carré dans une liste L_c.

Itération courante :

(i) Sélectionner le carré Q_h de borne inférieure minimum de la liste L_c. Subdiviser Q_h en quatre carrés égaux $Q_{h_1}, Q_{h_2}, Q_{h_3}, Q_{h_4}$.

(ii) Calculer une borne inférieure $\underline{F_h}(h = h_1, h_2, h_3, h_4)$ de $F(x, y)$ sur chacun des carrés. Eliminer les carrés dont la borne inférieure est plus grande que la valeur F^*.

(iii) Calculer sur chaque carré restant la valeur de la fonction objectif en quatre points(dont le centre) (borne supérieure) et mettre à jour F^* si nécessaire.

(iv) Si l'erreur relative entre la meilleure valeur connue F^* et la borne inférieure minimum : $\frac{F^*-\underline{F_h}}{F^*}$ est plus petite qu'une tolérance ϵ, fin. Le point ϵ-optimal s^* est solution du problème de Weber associé à la valeur F^*.

Dans la version de base de "BSSS", la borne inférieure se calcule de la manière suivante:

Soit Q_h un carré défini par 4 côtés parallèles aux axes de coordonnées et par 4 sommets A, B, Cet$D$ et p_j^1 le point du carré le plus proche du point de demande a^j. Q_h découpe le plan en 9 régions (cf figure 5.1) : NO, N, NE, O, Q_h, E, SO, S, SE.

$$\underline{F_h} = \sum_{i=1}^{n} w_j f_j(d_j(p_j^1)).$$

En considérant une fonction de coût concave et non décroissante en la distance $(f_j(d_j(s)))$ associée au point de demande a^j on propose d'améliorer la borne inférieure $\underline{F_h}$ sur Q_h. Soit $\underline{f_j}(x,y)$ une fonction minorante de $f_j(d_j(s))$ sur Q_h et soit la borne inférieure suivante:

$$\underline{F_h} = \min_{(x,y)\in Q_h} \sum_{j=1}^{n} \underline{f_j}(x,y).$$

On appellera LB_1 et LB_2 les deux procédures à venir pour le calcul de la fonction $\underline{f_j}(x,y)$.

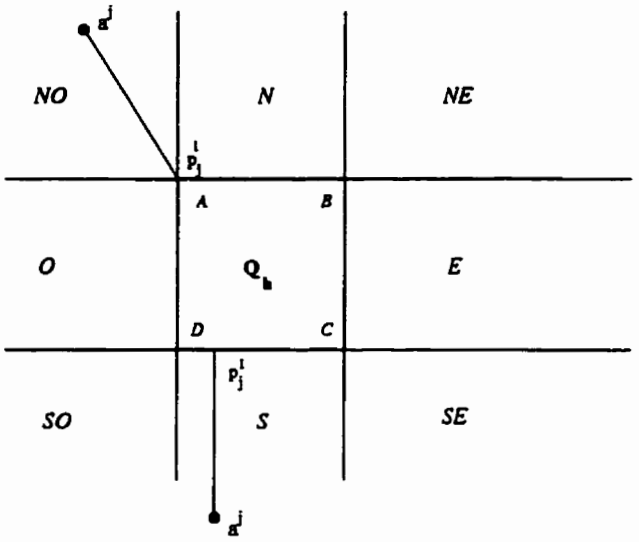


Figure 2.2 Carré Q_h

2.3.1 Procédure LB_1 : minoration par un plan

Soit $\underline{f}_j^1(x,y)$ le meilleur plan sous estimant $f_j(d_j(s))$ sur le carré Q_h. L'équation du plan $\underline{f}_j^1(x,y)$ dépend de la position du point de demande a^j par rapport au carré Q_h.

On peut distinguer 3 classes de zones à partir de Q_h pour la construction de la fonction minorante: (NO, NE, SO, SE), (N,S,O,E)), (Q_h) .

Cas 1 a^j est localisé dans les régions N, E, O, S (cf figure 2.3):

Soit p_j^1 le projeté orthogonal de a^j sur Q_h et p_j^2 la deuxième intersection entre la droite (a^j, p_j^1) et Q_h (cf figure 2.3) . Soit $\underline{f}_j^1(x,y)$ le plan passant par les points $(p_j^1, f_j(d_j(p_j^1)))$ et $(p_j^2, f_j(d_j(p_j^2)))$ généré par les droites perpendiculaires à (a^j, p_j^1) et parallèles au plan $z = 0$. Pour chaque point extrême du rectangle, la valeur du plan minorant est:

p_j	$d_j(p_j^1)$	$d_j(p_j^2)$	$\underline{f}_j^1(x_A, y_A)$	$\underline{f}_j^1(x_B, y_B)$	$\underline{f}_j^1(x_C, y_C)$	$\underline{f}_j^1(x_D, y_D)$
N	$y_j - sy_A$	$y_j - sy_D$	$f_j(d_j(p_j^2))$	$f_j(d_j(p_j^1))$	$f_j(d_j(p_j^1))$	$f_j(d_j(p_j^2))$
O	$sx_A - x_j$	$sx_C - x_j$	$f_j(d_j(p_j^1))$	$f_j(d_j(p_j^1))$	$f_j(d_j(p_j^2))$	$f_j(d_j(p_j^1))$
E	$x_j - sx_B$	$x_j - sx_A$	$f_j(d_j(p_j^2))$	$f_j(d_j(p_j^2))$	$f_j(d_j(p_j^1))$	$f_j(d_j(p_j^1))$
S	$sy_D - y_j$	$sy_A - y_j$	$f_j(d_j(p_j^1))$	$f_j(d_j(p_j^2))$	$f_j(d_j(p_j^2))$	$f_j(d_j(p_j^1))$

Cas 2 a^j est localisé dans les régions $N0, NE, SO, SE$. Soit p_j^1 le côté de $ABCD$ le plus près de a^j et D la droite passant par a^j et p_j^1, V_j le côté opposé à p_j^1 et p_j^2 la projection orthogonale de V_j sur D. Soit Δ la droite $[(p_j^1, f_j(d_j(p_j^1)))(p_j^2, f_j(d_j(p_j^2)))]$ (cf figure 2.4). Soit $\underline{f}_j^1(x, y)$ le plan contenant Δ et les droites orthogonales à Δ et parallèles à $z = 0$.

Cas 3 a^j est à l'intérieur de Q_h. On prendra 0 comme valeur minorante de $f_j(d_j(s))$.

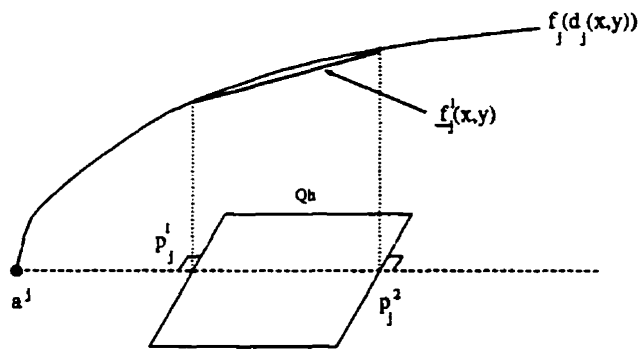


Figure 2.3 Construction de $f_j^1(x, y)$ (cas 1)

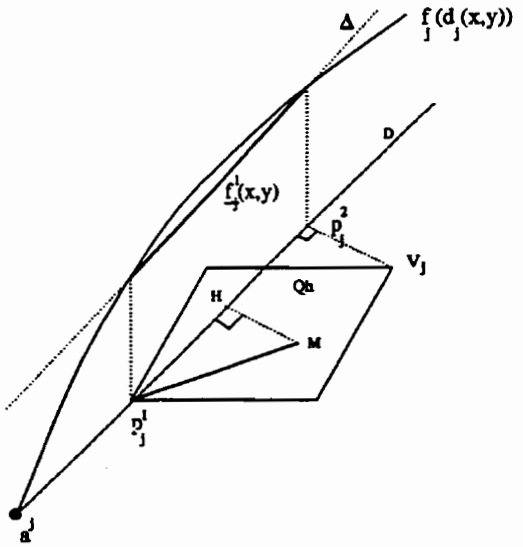


Figure 2.4 Construction de $f_j^1(x,y)$ (cas 2)

Proposition 4 *$\underline{f}_j^1(x,y)$ le plan contenant Δ et les droites orthogonales à Δ et parallèles à $z=0$ est le meilleur plan sous-estimant $f_j(d_j(s))$ sur Q_h.*

Preuve:

Soit $M(x,y)$ un point de Q_h et $H(x',y')$ son projeté orthogonal sur D:

$$d_j(s) > d_j(H)$$

$$f_j(d_j(s)) \geq f_j(d_j(H))\ d_j(s) > d_j(H)$$

et

$$\underline{f}_j^1(x',y') = \underline{f}_j^1(x,y)$$

$f(x)$ étant concave:

$$f_j(d_j(s)) \geq \underline{f}_j^1(x',y')$$

Soit la pente du plan :

$$s_j^p = \frac{f_j(d_j(p_j^2)) - f_j(d_j(p_j^1))}{d_j(p_j^2) - d_j(p_j^1)}$$

On en déduit l'expression analytique du plan:

$$\underline{f}_j^1(x,y) = \underline{f}_j^1(x',y') = f_j(d_j(p_j^1)) + s_j^p(d_j(H) - d_j(p_j^1))$$

Puisque la somme de fonctions affines est affine, $\sum_{j=1}^{n} \underline{f}_j^1(x,y)$ atteint son minimum en un des points extrêmes de Q_h. Il suffit alors d'évaluer pour chaque point de demande a^j la valeur de $\underline{f}_j^1(x,y)$ sur A, B, C et D:

$$\underline{F}_h^1 = \min\{\sum_{j=1}^{n} \underline{f}_j^1(x_A,y_A), \sum_{j=1}^{n} \underline{f}_j^1(x_B,y_B), \sum_{j=1}^{n} \underline{f}_j^1(x_C,y_C), \sum_{j=1}^{n} \underline{f}_j^1(x_D,y_D)\}.$$

2.3.2 Procédure $LB2$: minoration par un cône

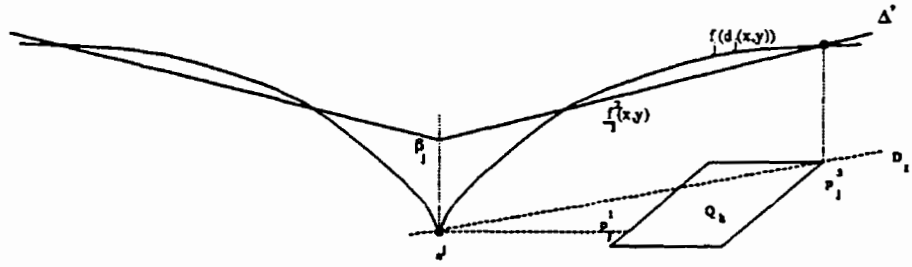


Figure 2.5 Construction du cône minorant.

Soit $\underline{f}_j^2(x,y)$ un cône sous estimant $f_j(d_j(s))$ sur Q_h (cf figure 2.5), p_j^1 et p_j^3 les deux points, respectivement le plus près et le plus loin de a^j et appartenant tous les deux à Q_h. Soit Δ' la droite passant par les points (a^j, β_j) et $(p_j^3, f_j(d_j(p_j^3))$ de pente $s_j^c = \frac{f_j(d_j(p_j^3)) - f_j(d_j(p_j^1))}{d_j(p_j^3) - d_j(p_j^1)}$ dont la projection orthogonale dans le plan(z=0) est D_1 (la droite passant pas a^j et p_j^3).

$$\underline{f}_j^2(x,y) = \beta_j + s_j d_j(s).$$

$$\underline{f}_j^2(x,y) = f_j(d_j(p_j^1)) + s_j^c(d_j(s) - d_j(p_j^1))$$

$$C(x,y) = \sum_{j=1}^{n} \underline{f}_j^2(x,y) \qquad (x,y) \in Q_h$$

On en déduit :

$$\underline{F}_h^2 = \min_{(x,y)\in Q_h} C(x,y).$$

$C(x,y)$ est convexe (la somme de fonctions convexes reste convexe). L'obtention du minimum de $C(x,y)$ sur Q_h est un problème de minimisation d'une fonction convexe sur un convexe. D'où la procédure à deux étapes suivante pour l'obtention du minimum de $C(x,y)$ sur Q_h.

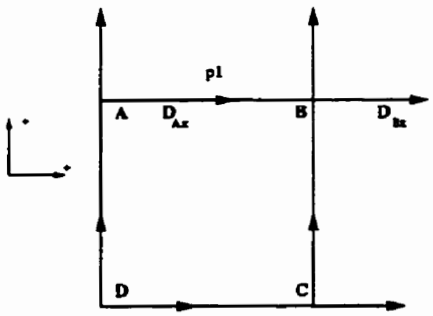
(1) Optimisation sur les côtés de Q_h

$$\min_{(x,y)\in env(Q_h)} C(x,y) = \min\{\min_{(x,y)\in[AB]} C(x,y), \min_{(x,y)\in[BC]} C(x,y), \min_{(x,y)\in[CD]} C(x,y),$$

$$\min_{(x,y)\in[DA]} C(x,y)\}.$$

Soient D_{Ax}, D_{Bx} les gradients de $C(x,y)$ aux points A et B suivant l'axe des x (cf figure ci_dessous) et $p_1(x_1,y_1)$ le point qui minimise $C(x,y)$ sur le segment $[A,B]$.

Le tableau suivant donne les différents lieux de p_1 en fonction du signe des gradients D_{Ax} et D_{Ay}.



D_{Ax}	D_{Bx}	$p_1(x, y_A)$
-	+	$x_1 \in]x_A, x_B[$
-	-	$p_1 = B$
+	+	$p_1 = A$

Dans le premier cas, p_1 s'obtient par l'algorithme de Weiszfeld (réduit à une dimension, une des coordonnées étant fixée) sur l'arête. Les points optimaux p_2, p_3, p_4 des autres côtés du carré s'obtiennent en suivant le même raisonnement. Soit p_o le point qui minimise $C(x, y)$ sur les 4 côtés de Q_h :

$$p_o(x_o, y_o) = \text{Argmin}(C(x_1, y_1), C(x_2, y_2), C(x_3, y_3), C(x_4, y_4)).$$

(2) Optimisation à l'intérieur de Q_h Soit l'algorithme de Weiszfeld (procédure itérative de descente) de point de départ p_o et p_o' le point issu de p_o après une itération. Si p_o' est à l'extérieur de Q_h alors p_o est le point qui minimise $C(x, y)$ sur Q_h sinon la procédure de Weiszfeld converge vers le minimum global de $C(x, y)$, à l'intérieur de Q_h.

2.4 Approche par programmation D. C.

Tout problème de minimisation d'une fonction continue peut s'écrire comme un problème de minimisation d'une fonction D.C. [73] [74]. Dans cette partie, nous proposons de minorer $F(x,y)$ par $DC(x,y)$ (une fonction D.C) et d'en déduire une procédure de résolution de $F(x,y)$.

Soit r_j un nombre réel positif. Nous pouvons toujours construire un cône $K_j(x,y)$ centré sur a^j (un point de demande) coupant $f_j(d_j(s))$ à une distance r_j de a^j (cf figure 2.6).

Soit $DC(x,y)$ une fonction D.C minorante de $F(x,y)$ sur S. Une fonction D.C peut être construite de la manière suivante:

$$DC(x,y) = \sum_{j=1}^{n} DC_j(x,y) = \sum_{j=1}^{n} \underbrace{K_j d_j(s)}_{\text{convexe}} - \underbrace{g_j(x,y)}_{\text{convexe}}$$

avec:

$$g_j(x,y) = \begin{cases} 0 & \text{lorsque } d_j(s) < r_j \\ K_j d_j(s) - f_j(d_j(s)) & \text{lorsque } d_j(s) \geq r_j \end{cases}$$

$DC_j(x,y)$ coïncide avec $f_j(d_j(s))$ pour tout point en dehors du cercle (a^j, r_j). $DC_j(x,y)$ coïncide avec le cône pour tout point à l'intérieur du cercle.

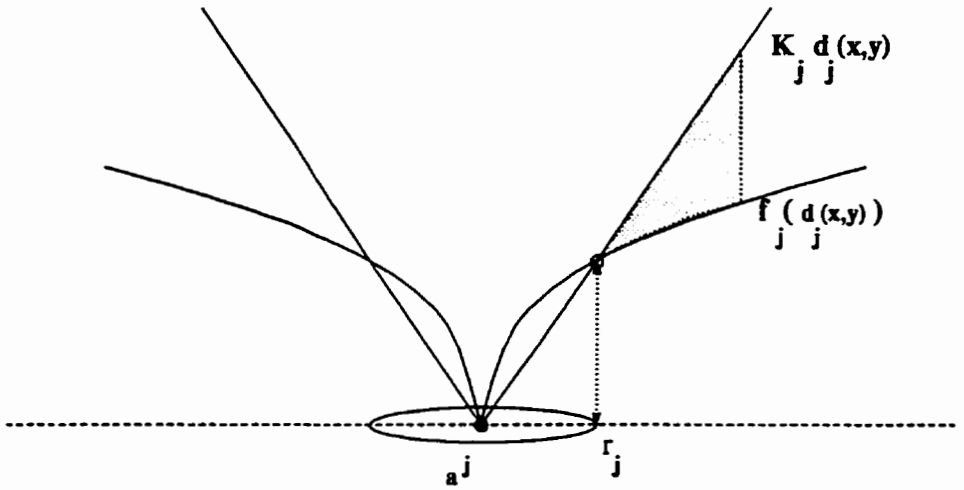


Figure 2.6 Décomposition D. C.

Puisque la somme de fonctions D.C reste D.C [74] [73] :

$$DC(x,y) = \sum_{j=1}^{n} DC_j(x,y).$$

est une fonction D.C. minorante de $F(x,y)$ pour tout $(x,y) \in S$ et on a:

$$\min_{(x,y)} DC(x,y) \leq \min_{(x,y)} F(x,y).$$

Nous pouvons reformuler le problème D.C. ($\min_{x,y} DC(x,y)$) comme un problème de minimisation d'une fonction concave sur un convexe [74] [73] [15]:

$$\min_{(x,y)} \sum_{i=1}^{n} K_j d_j(s) - \sum_{i=1}^{n} g_j(x,y)$$

devient

$$P_e \begin{cases} \min_{x,y,t} t - \sum_{j=1}^{n} g_j(x,y) \\ \text{sujet à} \\ \begin{cases} \sum_{j=1}^{n} K_j d_j(s) - t \leq 0 \\ (x,y) \in S, t \geq 0. \end{cases} \end{cases}$$

P_e est résolu optimalement par une méthode d'approximation extérieure (Chen *et al.* [16]) dans laquelle le domaine convexe est approximé extérieurement à chaque itération par un polytope de plus en plus petit. Le minimum de la fonction concave sur le polytope est obtenu en un de ses points extrêmes [46] ce qui donne une borne inférieure sur la solution. Une borne supérieure est calculée simultanément et la meilleure valeur connue est mise à jour si nécessaire. Le polytope est ensuite raffiné par un plan de coupure éliminant le point minimum précédemment obtenu et l'algorithme réitère jusqu'à ce que la différence entre la valeur minimum de la fonction obtenue sur le polytope courant et la meilleure valeur connue (correspondant à un point réalisable) est inférieure à une petite valeur (ϵ).

Nous pouvons déduire des résultats précédents une procédure permettant de résoudre le problème de Weber:

(1) initialiser r_j pour chaque point de demande. Soit $r_j = d_{ij} = \min_{i=1\ldots n(j\neq i)}(d_j(a^i))$.

(2) résoudre le programme P_e par l'algorithme de Chen *et al.*. Soit (x^*, y^*) le point obtenu.

(3) mise à jour de J'.

$$J' = \{j = 1, n : 0 < d_j(x^*, y^*) < r_j\}$$

Si $J' = \emptyset$, $DC(x, y)$ coïncide avec $F(x, y)$ et la localisation optimale de la source (x^*, y^*) est trouvée, fin. Sinon faire :

$$(j \in J', r_j = d_j(x^*, y^*)/2).$$

Mise à jour des pentes K_j pour $j \in J'$ et aller en (2).

2.5 Résultats numériques

Dans cette partie, les deux algorithmes sont testés pour différentes valeurs de n (nombre de points de demande) allant de 10 à 10000. Pour chaque valeur de n, 10 problèmes sont testés et les résultats sont présentés dans les tableaux ci après 2.1,2.2.

(i) iter1 : nombre de coupes utilisées pour la minimisation de la fonction concave sur le convexe.

(ii) iter2 : nombre d'itérations de l'algorithme D.C.

L'algorithme D.C est programmé en fortran, l'algorithme d'énumération implicite est codé en langage C. Les expériences de calcul sont obtenues sur SPARC4 : 28.5 Mips, RAM = 64 M.

Plusieurs conclusions apparaissent clairement:

- L'algorithme d'énumération implicite est plus rapide que l'algorithme D.C dans tous les cas et de plus en plus quand la taille du problème augmente.
- Le temps pour cet algorithme croît presque linéairement avec le nombre de points de demande.
- Le compromis temps de calcul/précision est identique pour les bornes LB_1 et LB_2.

Tableau 2.1 Résultat pour l'algorithme D.C.

algorithme D. C.							
n	OBJ	dij					
		iter1		iter2		cpu	
		$\bar{it}$	σ	$\bar{it}$	σ	$\bar{cpu}$	σ
10	8.38	40.2	23.02	1.2	.4	.48	.3
20	16.87	111.6	67.78	1.8	.67	2.67	1.82
50	47.86	145.7	184.35	1.7	.9	8.23	11.05
100	93.58	219	146.36	2.1	.53	24.18	16.75
200	192.04	269.7	218.6	2.2	.6	60.73	53.07
500	474.83	294.1	159.7	2.4	.66	161.62	90.09
1000	957.06	452.6	313.3	2.6	.8	572.09	409.18
2000	1910.59	435.9	140.6	2.7	.45	1057.6	351.6
10000	9549.66	1450.3	295.44	6.8	.97	15221.7	3173.11

Tableau 2.2 Résultat pour l'algorithme d'énumération implicite.

Enumération implicite									
n	OBJ	Plan				Cône			
		iter		cpu		iter		cpu	
		$\bar{it}$	σ	$\bar{cpu}$	σ	$\bar{it}$	σ	$\bar{cpu}$	σ
10	8.38	30	4.09	.21	.02	26.3	2.93	.28	.04
20	16.87	39.1	8.85	.54	.12	31.7	4.77	0.7	0.14
50	47.86	54.5	24.81	1.88	.87	38.6	11.07	2.32	.79
100	93.58	67.9	30.2	4.66	2.11	43	10.93	5.39	1.61
200	192.04	75.4	29.5	10.37	4.12	44.4	12.3	11.35	3.51
500	474.83	86.1	30.29	29.35	10.59	48.4	13.12	31.04	8.33
1000	957.06	99.6	24.6	68.30	16.8	51.1	10.05	69.15	13.42
2000	1910.59	106.2	23.85	145.61	32.33	52.1	9.10	144.69	24.18
10000	9549.66	113.7	15.11	785.36	105.38	52.7	5.72	768.8	68.17

CHAPITRE 3

Le problème de Weber avec contraintes de localisation et de passage

3.1 Introduction

En pratique, les problèmes de transport et de localisation sont sujets à plusieurs contraintes. Les montagnes, les lacs, les zones urbaines sont inadéquates pour la localisation de la source (contrainte de localisation). D'autres part, des zones sujettes à congestion ou naturellement infranchissables doivent être évitées (contraintes de passages). Aneja et Parlar [2], Butt et Cavalier [11] [12] étudièrent ces deux types de contraintes. Les contraintes de localisation sont généralement représentées dans la littérature par des formes géométriques simples: polygones (triangles et quadrilatères). Hurter *et al.* [47], Schaefer et Hurter [70] considèrent des disques comme zones permises.

Hansen *et al.* [43] étudient le cas où la zone permise est une union de polygones simples (triangles et quadrilatères). Sans perte de généralité, toute zone permise peut être approximée (à la précision voulue) par un ensemble de telles formes géométriques. Ces derniers auteurs constatent que le nombre de polygones n'influence pas beaucoup le temps de résolution. Hurter *et al.* [47] observent que si le lieu optimal de la source du problème de Weber non contraint se situe dans une zone permise du problème de Weber contraint alors il est optimal pour ce dernier problème. Sinon, le lieu optimal de la source du problème de Weber contraint est *visible* de celui du problème de Weber non contraint (i.e., il existe une droite

joignant les deux lieux qui ne contient pas de point intérieur permis). De plus la fonction objectif du problème de Weber sur la frontière d'un polygone convexe est quasi-convexe, ce qui permet d'obtenir par une recherche unimodale la solution du problème de Weber contraint. Des problèmes ayant 100 points de demande et 100 triangles et quadrilatères (zones permises) sont résolus en quelques secondes. Aneja et Parlar [2] considèrent le cas où la zone interdite est un ensemble de polygones simples (convexes ou non).

Les contraintes de passage sont peu étudiées dans la littérature. Le plus court chemin (appelé distance réalisable) d'une origine fixe (localisation de la source) à une destination fixe (localisation d'un point de demande) est obtenu par la construction d'un *graphe de visibilité* (e.g. Viegas et Hansen [79]). Les sommets sont les points extrêmes des polygones représentant les zones infranchissables, l'origine et la destination. Un arc existe entre deux sommets si les points du plan associés sont visibles l'un de l'autre (la droite les reliant ne contient aucun point intérieur de zones infranchissables). Le poids sur chacun des arcs est la longueur du chemin entre les deux points. Le plus court chemin entre deux sommets (deux points du plan) est obtenu par l'algorithme de Dijkstra [22].

Aneja et Parlar [2] considèrent le problème de Weber avec des zones (polygones convexes et non convexes) interdisant la localisation et le passage. Ils proposent une heuristique de type : "recuit simulé" en utilisant le graphe de visibilité pour évaluer la fonction économique.

Butt et Cavalier [12] proposent une autre heuristique basée sur une adaptation de l'algorithme de Weiszfeld [76]. Elle converge vers un optimum local. Cette heuristique est appelée plusieurs fois à partir de points réalisables initiaux choisis aléatoirement.

A notre connaissance, aucun algorithme exact n'a été proposé à ce jour pour résoudre le problème de Weber avec contraintes de passage et de localisation. Nous proposons dans ce chapitre la première méthode exacte.

Le coût de transport sera pris comme une fonction proportionnelle à la distance réalisable. La méthode proposée est construite sur le schéma de l'algorithme "BSSS" présenté dans les deux premiers chapitres.

3.2 Formulation du problème

Les hypothèses du problème de Weber avec contraintes de localisation et de passage sont:

(i) P : un plan.

(ii) Soit $J = \{a^1, ...a^n\}$ un ensemble de n clients (ou groupe de clients) dans le plan et des poids w_j $(j = 1.....n)$ associés à ces clients, qui représentent l'importance de la demande.

(iii) Soient m_1 régions P_i interdites pour la localisation. Ces zones sont les intérieurs de polygones (convexe ou non). Chacune est décrit par une liste de ses sommets $(b^{i1}, b^{i2}, ..., b^{ie_i}\ i = 1, m_1)$.

(iv) Soient m_2 régions R_i interdites pour la localisation et le passage. Ce sont les intérieurs de polygones (convexes ou non). Elles sont définies par une liste de leurs sommets $(c^{i1}, c^{i2}, ...c^{in_i}; i = 1, m_2)$. Soit Z_i le plus petit rectangle avec des côtés parallèles aux axes de coordonnées contenant R_i (figure 3.1):

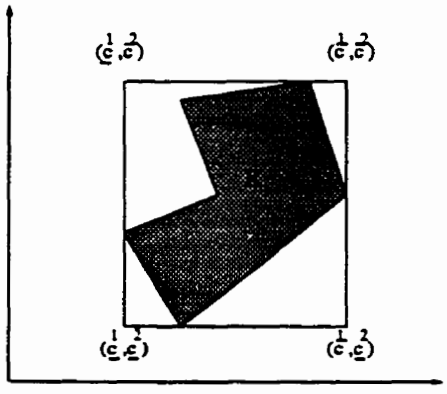


Figure 3.1 Z_i

$$\underline{c}_1^i = \min_{j=1,2..n_i} c_1^{ij} \quad \bar{c}_1^i = \max_{j=1,2..n_i} c_1^{ij}$$
$$\underline{c}_2^i = \min_{j=1,2..n_i} c_2^{ij} \quad \bar{c}_2^i = \max_{j=1,2..n_i} c_2^{ij}$$

(iv) Soit une source à localiser en un point réalisable $s(x, y)$, i.e, $s \in P \setminus (\cup_{i=1}^{m_1} P_i \cup_{i=1}^{m_2} R_i)$.

(v) Soit $\underline{d}_j(s)$ la longueur du plus court chemin réalisable (respectant les contraintes de passage) du point a^j à s. La source doit être localisée afin de minimiser le coût total de transport $F(s)$ défini par:

$$F(s) = \sum_{j=1}^{n} f_j(s) = \sum_{j=1}^{n} w_j \underline{d}_j(s)$$

(vi) $d_j(s)$: distance euclidienne entre le point de demande a^j et s.

(vii) Q_h : un carré du plan.

(viii) $d(a, b)$: distance euclidienne entre les points a et b.

(ix) $p(x, y)$: un point courant du plan.

(x) $p_j(x, y)$: le point sur Q_h le plus proche de a^j (en considérant les contraintes de passage).

(xi) p^h : point central du carré Q_h.

(xii) $\underline{d}_j(s)$: la plus courte distance entre a^j et s.

(xii) $\underline{d}_j(Q_h)$: plus petite distance réalisable entre a^j et Q_h.

(xiii) $s_h(p)$: projeté d'un point p du plan sur Q_h ($p \notin Q_h$).

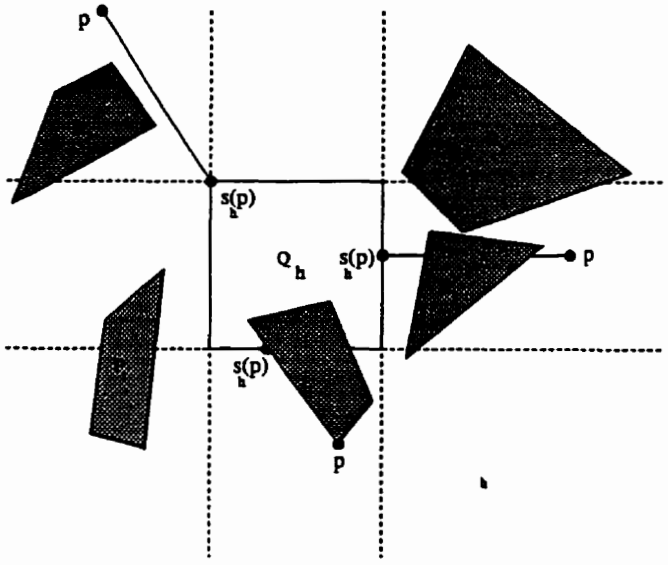


Figure 3.2 Projeté de p sur Q_h

(xiv) V : ensemble des sommets des polygones R_i, $i = 1, m_2$.

Sans perte de généralité, la région dans laquelle la source est localisée peut être restreinte à l'enveloppe convexe des points de demande et des points extrêmes des polygones R_i. En effet, un argument simple montre que tout point à l'extérieur de cet ensemble convexe est dominé par au moins un point sur sa frontière. Aneja et Parlar dans leur article "Algorithms for Weber facility Location in the Presence of Forbidden Regions"[2] mentionnent dans l'étape 1 de l'algorithme 2 que le lieu optimal se trouve à l'intérieur du plus petit rectangle contenant les points de demande seulement. Le contre-exemple suivant met en défaut leur affirmation.

Soient 4 points de demande localisés en $a^1(1,1)$, $a^2(1,2)$, $a^3(7,2)$ et $a^4(7,1)$. Une zone R_1 (interdite pour la localisation et le passage) : un carré dont les sommets sont $c_1^1(2,0)$, $c_1^2(2,4)$, $c_1^3(6,4)$, $c_1^4(4,0)$. Tous les poids w_j sont égaux à 1 et les fonctions $f_j(s)$ valent $\underline{d}_j(s)$. L'enveloppe convexe des points de demande est le rectangle dont les sommets sont a^1, a^2, a^3, a^4. Tout point de R_1 sur la frontière intérieure, (i.e. sur la droite joignant c_1^1 à c_1^4) a une valeur de $8 + 2\sqrt{2} + 2\sqrt{5}$ et est optimal. En effet si s est à gauche de R_1 et d'ordonnée inférieure à 2, le plus petit chemin de a^3 et a^4 à s passe par c^4 et c^1 et est de longueur $\sqrt{5} + 4 + d(c^1, s)$ ou $\sqrt{2} + 4 + d(c^1, s)$. La somme des distances est alors une constante plus la distance de c^1 à s. Le problème donné est équivalent à un problème de Weber avec 3 points de demande a^1 a^2 et c^1 de poids $w_1 = w_2 = 1$ et 2 respectivement. Par la propriété de la majorité, le lieu optimal se trouve en c^1. Un argument semblable s'applique quand s est à droite de R_1. Si s est d'ordonnée supérieur ou égale à 2, les distances réalisables aux points de demande sont plus grandes que si s est d'ordonnée inférieure à 2 et de même abscisse. Ce qui complète le contre-exemple.

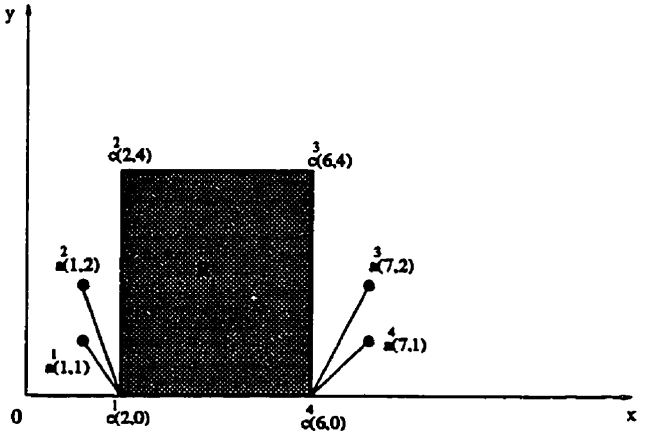


Figure 3.3 Exemple.

3.3 Problèmes auxiliaires

Le problème de Weber avec contraintes de localisation et de passage sera résolu par un algorithme d'énumération implicite (en variables continues). C'est une généralisation de "Big Square Small Square" développé par Hansen *et al.* [43]. L'idée de "BSSS" est de partir d'une partition (en carrés) du plan puis de calculer une borne inférieure et une borne supérieure (solution réalisable) sur la meilleure localisation de la source sur chaque carré. La meilleure localisation connue de la source est mise à jour à chaque itération (si nécessaire). Les carrés dont la borne inférieure est plus grande que la meilleure valeur connue (meilleure localisation) sont éliminés. A chaque itération, le carré de borne inférieure minimum est subdivisé en 4 carrés égaux. L'algorithme réitère jusqu'à ce que la précision désirée sur le carré (aire minimum) ou sur la valeur de la fonction objectif (différence entre la meilleure valeur connue et sa borne inférieure) soit atteinte. Avant de présenter l'algorithme d'énumération implicite, cinq problèmes auxiliaires sont examinés:

(i) **résolution d'un problème de Weber simple sans contraintes sur Q_h** : trouver le point de Q_h qui minimise la somme des distances pondérées à des points fixes du plan.

(ii) **visibilité :** trouver si 2 points s et s' du plan sont visibles l'un de l'autre, i.e si le segment $[s, s']$ est entièrement réalisable.

(iii) **distance réalisable d'un point p à Q_h** trouver le plus court chemin de p à Q_h qui contient seulement des points réalisables.

(iv) **point dominant sur Q_h associé à a^j** : dernier sommet v^j (entièrement visible de Q_h) appartenant à $\cup R_i$ ($i = 1, m_2$) rencontré sur le plus court chemin entre a^j et Q_h qui a la propriété suivante :

$$\forall s \in Q_h \ , \ \underline{d}_j(s) \geq \underline{d}_j(v^j) + \max_{p \in Q_h} d(v^j, p)$$

(v) visibilité sur Q_h: trouver si tous les points de Q_h sont entièrement visibles d'un point p du plan.

Propriété 7 *Si 2 points s et s' ne sont pas visibles l'un de l'autre, le plus court chemin de s à s' est une somme de segments $([s, v_k], [v_k, v_l], ..., [v_m, s'])$ où v_k et v_l sont des sommets de V.*

3.3.1 Problème de Weber simple sans contraintes sur Q_h

Soit $\{p_j\}$ $(j \in J')$ un ensemble de points fixes du plan. On cherche s_h^* $(s_h^* \in Q_h)$ le point qui minimise:

$$F_{J'}(s_h^*) = \min_{(x,y) \in Q_h} F(s) = \min_{(x,y) \in Q_h} \sum_{j \in J'} w_{p_j} d(p^j, s).$$

$F_{J'}(s_h)$ est une fonction convexe (somme pondérée de distances euclidiennes). La minimisation de $F_{J'}(x, y)$ se déroule en 2 étapes:

(1) minimisation sur la frontière de Q_h.

(2) minimisation à l'intérieur de Q_h.

Le point qui minimise $F_{J'}(x, y)$ sur la frontière de Q_h est obtenu en considérant les gradients de la fonction sur les sommets de Q_h (cf chapitre 2). Ce point est alors pris comme point de départ pour l'algorithme itératif de descente de Weiszfeld [76](cf chapitre 2 section 2.3.2).

3.3.2 Visibilité

Soient $s(s_1, s_2)$ et $s'(s'_1, s'_2)$ deux points du plan (sans perte de généralité : $s'_1 > s_1$) et les polygones R_i, $(i = 1, m_2)$.

(1) Pour tout $i = 1, m_2$:

(1.1) si $\underline{c}_1^i > s'_1$ ou $\overline{c}_1^i < s_1$ ou $\underline{c}_2^i > \max(s_2, s'_2)$ ou $\overline{c}_2^i < \min(s_2, s'_2)$ aller en (1).

(1.2) si $s_2 > s'_2$ et $(\underline{c}_1^i, \underline{c}_2^i)$ est au dessus de la droite joignant s à s' ou $(\overline{c}_1^i, \overline{c}_2^i)$ est en dessous de cette droite, aller en (1).

Si $s'_2 > s_2$ et $(\overline{c}_1^i, \underline{c}_2^i)$ est au dessus de la droite joignant s et s' ou $(\underline{c}_1^i, \overline{c}_2^i)$ est en dessous de cette droite, aller en (1).

(1.3) En considérant toutes les arêtes $[c^{ik}, c^{i(k+1)}]$ du polygone R_i $(k = 1, (e_i - 1))$ et $[c^{in_i}, c^{i1}]$. Si une arête coupe le segment $[s, s']$, s et s' ne sont pas visibles, fin. Sinon aller en (1).

(1.4) S'il n'existe plus de polygones à examiner, s et s' sont visibles. Fin.

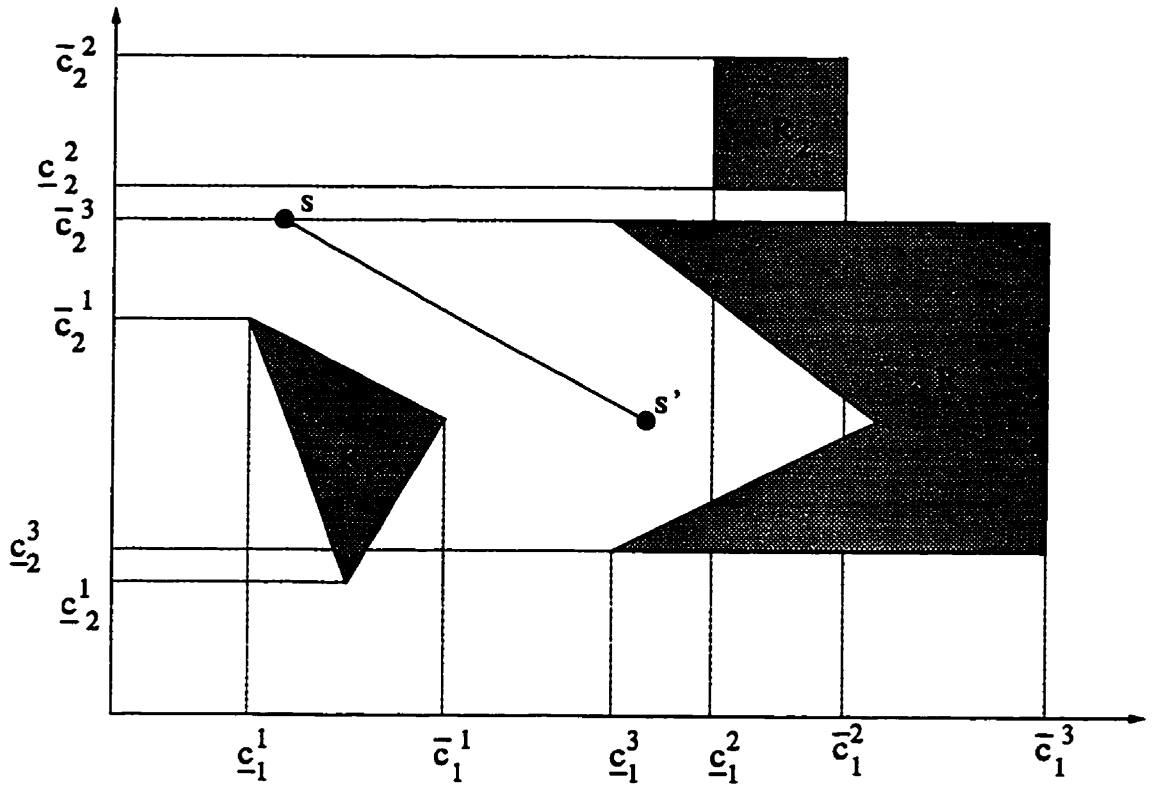


Figure 3.4 Visibilité

3.3.3 Visibilité totale sur Q_h par rapport à p

Soit p un point sur une des zones latérales de Q_h (Q_h entièrement réalisable) (cf figure 3.6). Soit D le rectangle de sommets (d^1, d^2, d^3, d^4) construit de la manière suivante:

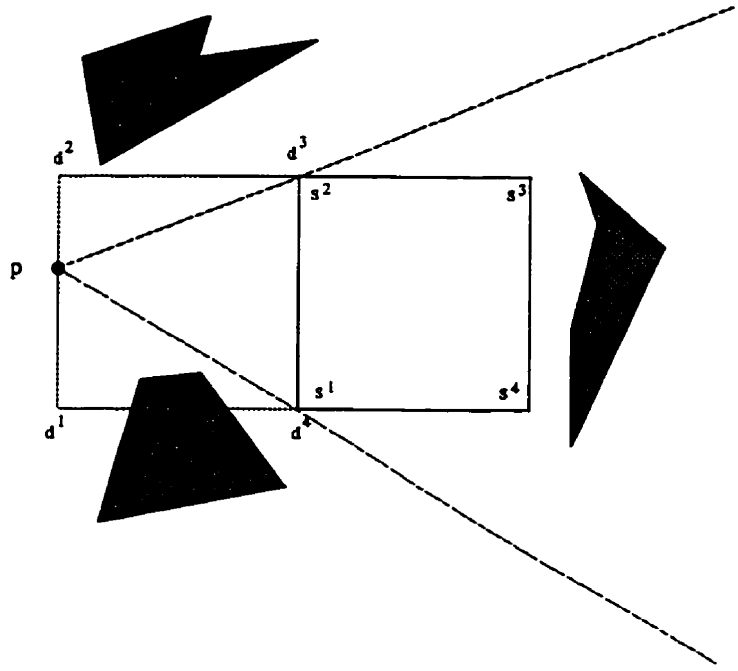


Figure 3.5 Visibilité totale sur un carré (cas 1).

$$d^1 = (v_1^k, s_2^1)\ ,\ d^2 = (v_1^k, s_2^2)$$
$$d^3 = s_2\ ,\ d^4 = s_1$$

(1) pour tout $i = 1, m_2$:

(1.1) si $\vec{c}_1^i < d_1^1$ ou $\underline{c}_2^i > d_2^2$ ou $\underline{c}_1^i > d_1^3$ ou $\vec{c}_2^i < d_2^1$ aller en (1).

(1.2) si le point $(\vec{c}_1^i, \underline{c}_2^i)$ ou le point $(\vec{c}_1^i, \vec{c}_2^i)$ est intérieur au triangle (p, d^3, d^4):

Il n'y a pas de visibilité totale sur Q_h par rapport à p.

Sinon aller en (1).

(2) p visible de $s_h(p)$: Q_h est entièrement visible de p.

Le même raisonnement peut être fait lorsque p se trouve dans une autre des régions du plan (cf figure 3.6).

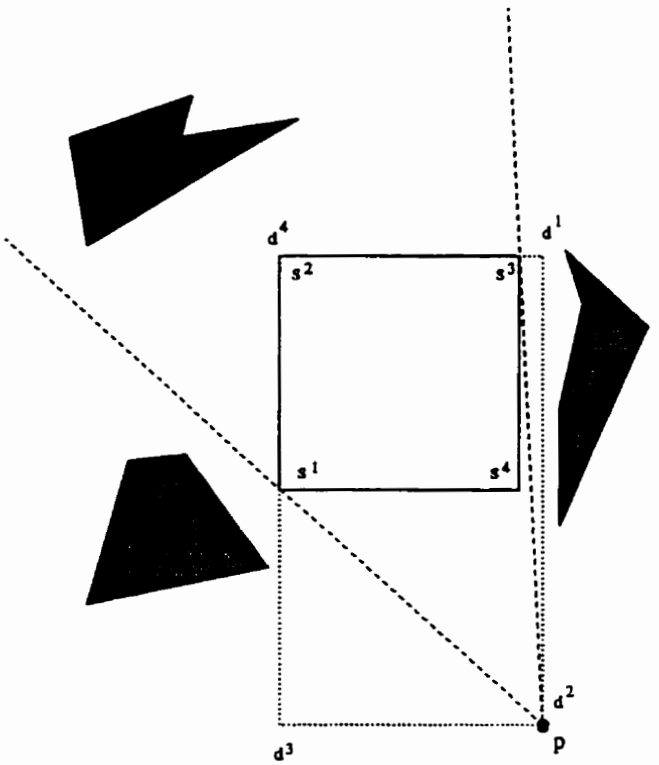


Figure 3.6 Visibilité totale sur un carré (cas 2).

3.3.4 Distance réalisable de a^j à Q_h

On considère Q_h partiellement ou entièrement réalisable (i.e $Q_h \cap (\sum_{i=1}^{m_1} P_i \cup \sum_{i=1}^{m_2} R_i) \neq \emptyset$).

En considérant les sommets de V (sommets des polygones R_i, $i = 1...m_2$). Soit $\underline{d}_j(Q_h) = M$ un nombre arbitrairement grand. Trier les sommets v_k par ordre croissant des distances par rapport à a^j. Soit $n_s = \sum_{i=1}^{m_2} e_i$ le nombre de sommets de V.

(1) Pour tout $k = 1....n_s$.

(1.1) si $\underline{d}_j(v_k) + d(v_k, p^h) > \underline{d}_j(Q_h) + \frac{\sqrt{(2)}}{2} d(s^1, s^2)$ (cf figure 5.2):
$\underline{d}_j(Q_h)$ est la distance réalisable, Fin.

(1.2) si $\underline{d}_j(v_k) + d(v_k, s_h(v_k)) \geq \underline{d}_j(Q_h)$ aller au sommet suivant dans la liste.

(1.3) si v_k est visible de $s_h(v_k)$:

$$\underline{d}_j(Q_h) = \underline{d}_j(v_k) + d(v_k, s(v_k))$$

Aller en (1).

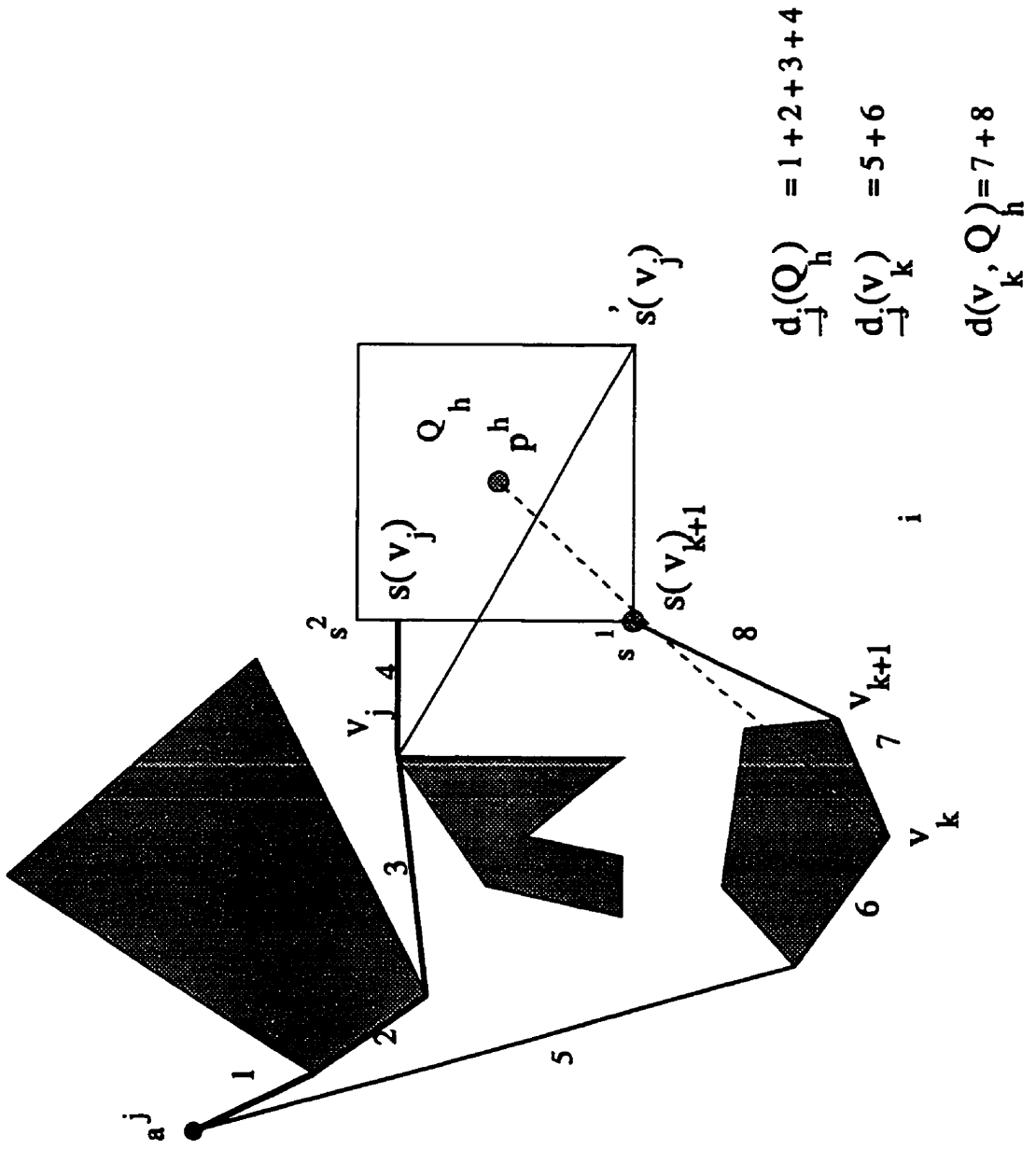


Figure 3.7 Distance réalisable et point dominant associé à a^j sur Q_h

3.3.5 Point dominant sur Q_h par rapport à a^j

Soit Q_h entièrement réalisable.

Soit $\underline{d}_j(Q_h) = M$ un nombre arbitrairement grand. Trier les sommets v_k par ordre croissant des distances par rapport à a^j.
d_carré_sup = M (un nombre arbitrairement grand), n_s le nombre de points de V.

(1) pour tout $k = 1...n_s$:

(1.1) si $\underline{d}_j(v_k) + d(v_k, p^h) > \underline{d}_j(Q_h) + \frac{\sqrt{(2)}}{2} d(s^1, s^2)$ (cf figure 5.2):

$\underline{d}_j(Q_h)$ est la distance réalisable.

(1.2) si $\underline{d}_j(v^j) + \max_{s \in Q_h} d(v^j, s) - \frac{\sqrt{(2)}}{2} d(s^1, s^2) > \underline{d}_j(v_k) + d(v_k, p^h)$ (cf figure 5.2):

- si v_k et $s_h(v_k)$ sont visibles:
  - si $\underline{d}_j(v_k) + d(v_k, s_h(v_k)) < \underline{d}_j(v^j) + \max_{s \in Q_h} d(v^j, s)$: Fin, il n'y a pas de point dominant associé à a^j sur Q_h.
  - Si d_carré_sup $=> \underline{d}_j(v_k) + d(v_k, s_h(v_k))$:
    d_carré_sup $= \underline{d}_j(v_k) + d(v_k, s_h(v_k))$.

(1.3) si $d_j(Q_h) = M$ et si v_k est visible de $s_h(v_k)$:

- $\underline{d}_j(Q_h) = \underline{d}_j(s_h(v_k)) = \underline{d}_j(v_k) + d(v_k, s_h(v_k))$
  - si v_k est entièrement visible de tous les points du carré:
    - si $\underline{d}_j(s'_h(v^j)) = \underline{d}_j(v^j) + d(v^j, s'_h(v^j)) <$ d_carré_sup :
      $v^j = v_k$
      aller en (1).

3.4 Algorithme

Le schéma de l'algorithme est le même que celui présenté au chapitre 2. C'est une méthode d' énumération et de séparation largement inspirée de l'algorithme BSSS développé par Hansen *et al.* [43]. L'idée est de partir du plus petit carré de côtés parallèles aux axes de coordonnées contenant tous les points de demande et de le subdiviser en quatre carrés égaux. Une borne inférieure et une borne supérieure sur la valeur minimum de la fonction objectif sont ensuite calculées sur chacun de ces carrés. A chaque itération un carré ayant la borne inférieure minimum est sélectionné puis subdivisé en quatre nouveaux carrés égaux, une borne inférieure et une borne supérieure sont ensuite obtenues sur ces carrés, la meilleure valeur connue est mise à jour si nécessaire, les carrés de borne inférieure plus grande que la meilleure valeur connue sont éliminés tandis que les autres carrés (réalisables) sont stockés dans une liste. L'algorithme s'arrête lorsque la liste est vide, c'est à dire quand il n'y a plus de carrés à sélectionner. Cet algorithme utilise une stratégie de type meilleur-d'abord.

Rappelons les principales notations:

- S : le plus petit carré contenant tous les points de demande et les sommets des polygones R_i, $(i = 1...m_2)$.
- J_E : ensemble des points de demande d'où le carré est entièrement visible.
- J_F : $J \setminus J_E$.
- F_h^* : valeur optimale de $F(x, y)$ sur Q_h.
- $\underline{F}_h$: borne inférieure sur F^* à l'intérieur et sur les frontières de Q_h.
- $\overline{F}_h$: borne supérieure sur F^* à l'intérieur et sur les frontières de Q_h

- F^* : meilleure valeur connue de la fonction économique.
- p_r^h : point réalisable à l'intérieur de Q_h.
- s_h^* : localisation optimale de la source à l'intérieur de Q_h.

(i) Partitionnement initial de S en 4 carrés égaux: Q_1, Q_2, Q_3, Q_4, $H = \{1, 2, 3, 4\}$.
$H' = \emptyset$.

(ii) Pour tout $h \in H$:

(I) si $Q_h \cap (\sum_{i=1}^{m_2} R_i) = Q_h$:

(I.1) Borne inférieure ($\underline{F}_h = 0; J_E = J_F = \emptyset$) :

(I.1.1) Pour $j = 1...n$:

- si Q_h est entièrement visible de a^j (procédure 1.3.4) :
  $J_E = J_E \cup a^j$
- sinon :
  $J_F = J_F \cup a^j$

(I.1.2) Pour $a^j \in J_F$:

s'il existe $v^j \in V$ dominant sur Q_h (procédure 1.3.6) :

- $\underline{F}_h = \underline{F}_h + w_j \underline{d}_j(v^j)$.
- $J_E = J_E \cup v^j . J_F = J_F \setminus a^j$.

(I.1.3) Pour $a^j \in J_F$:

calcul de $\underline{d}_j(Q_h)$ (procédure 1.3.5) :

- $\underline{F}_h = \underline{F}_h + w_j \underline{d}_j(Q_h)$.

(I.1.4) Résolution du problème de Weber simple (algorithme de Weiszfeld [76]) sans contraintes pour l'ensemble J_E (procédure 1.3.1):

- $\underline{F}_h = \underline{F}_h + \min_{s \in Q_h} \sum_{p_j \in J_E} w_j d(p_j, s)$.
  Si cardinal $J_E = n$: le problème est résolu exactement sur Q_h.

(I.2) Borne supérieure ($\overline{F}_h = 0$) :

(I.2.1) si cardinal(J_E) $= n$:

Le problème de Weber avec contraintes est résolu exactement sur Q_h ($F^* = \overline{F}_h = \underline{F}_h$).

- si $F^* > \overline{F}_h$, mise à jour de la meilleure solution connue :

  $F^* = \overline{F}_h$.

  $s^* = s_h^*$.

(I.2.2) si cardinal(J_E) $\neq n$: et si $\underline{F}_h < F^*$:

- Pour tout $j = 1...n$:

  calculer $\underline{d}_j(p_r^h)$ (procédure 1.3.5).

  $\overline{F}_h = \overline{F}_h + \sum_{j=1}^{n} w_j \underline{d}_j(p_r^h)$.

(I.3) Mise à jour des listes H et H' :

$H' = H' \cup h$.

$H = H \setminus h$.

(I.4) Mise à jour de la meilleure solution connue :

si $\overline{F}_h < F^*$:

- $F^* = \overline{F}_h$
- $s^* = p_r^h$
- pour tout $l \in J$:

  si $\underline{F}_l > F^*$:

  $H' = H' \setminus l$

(II) si $Q_h \cap (\sum_{i=1}^{m_2} R_i) \neq Q_h$ et $Q_h \cap (\sum_{i=1}^{m_2} R_i) \neq \emptyset$:

(II.1) Borne inférieure ($\underline{F}_h = 0$, $J_E = J_F = \emptyset$) :

(1) pour $j = 1...n$:

- calcul de $\underline{d}_j(Q_h)$ (procédure 1.3.5).
- $\underline{F}_h = \underline{F}_h + w_j \underline{d}_j(Q_h)$.

(II.2) Borne supérieure ($\overline{F}_h = 0$) :

(II.2.1) pour $j = 1...n$:

- calculer $\underline{d}_j(p_r^h)$ (procédure 1.3.5).
- $\overline{F}_h = \overline{F}_h + w_j \underline{d}_j(p_r^h)$.

(II.3) Mise à jour des listes H et H' :

$H' = H' \cup h$.

$H = H \setminus h$.

(II.4) Mise à jour de la meilleure solution connue :

si $\overline{F}_h < F^*$:

- $F^* = \overline{F}_h$.
- $s^* = p_r^h$.
- pour tout $l \in J$: élimination de carrés:
  - si $\underline{F}_l > F^*$:

    $H' = H' \setminus l$:

(III) si $Q_h \cap (\sum_{i=1}^{m_2} R_i) = \emptyset$:

- $H' = H'$
- $H = H \setminus h$

(iii) sélectionner $h \in H'$: $\underline{F}_h = \min_{k \in H'} \underline{F}_k$. Si $H' = \emptyset$, fin.

(iv) partitionner Q_l en 4 carrés égaux : $\{Q_{l1}, Q_{l2}, Q_{l3}, Q_{l4}\}$ et aller en (ii).

A chaque itération le carré de borne inférieure minimum est sélectionné de la liste des sous-problèmes, puis est subdivisé en quatre carrés égaux. Soit Q_h un de ces carrés. Trois possibilités se présentent alors : Q_h est entièrement réalisable, Q_h est partiellement réalisable, Q_h n'est pas réalisable.

Quand Q_h est entièrement réalisable, les points de demande sont séparés en deux groupes: un groupe contenant les points de demande entièrement visibles de

Q_h et les points de demande admettant un point dominant sur Q_h (cf section 3.3.5), un groupe contenant les points de demande restant. La contribution du premier groupe pour la borne inférieure est obtenue en résolvant un problème de Weber contraint sur Q_h (cf (I.1.4), celle du second groupe est la somme des plus petites distances réalisables pondérées des points de demande du groupe à Q_h. La valeur de la fonction objectif au centre du carré est la borne supérieure associé à Q_h.

Quand Q_h est partiellement réalisable, la somme des plus petites distances réalisables pondérées des points de demande à Q_h est prise comme borne inférieure. Une borne supérieure est obtenue en évaluant la fonction objectif en un point réalisable de Q_h.

Quand Q_h n'est pas réalisable, le carré est éliminé.

Notons que $F(x,y) - \underline{F}_h$ tend vers 0 quel que soit $(x,y) \in Q_h$ lorsque la diagonale du carré Q_h tend vers 0. La convergence de l'algorithme est alors assurée en utilisant les résultats généraux de la méthode d'énumération implicite [Horst et Tuy [46]].

3.5 Résultats numériques

Les problèmes test sont résolus sur SPARC SUN 20 et le code du programme est écrit en langage C. Nous allons tout d'abord résoudre le problème test de littérature de Aneja et Parlar [2] (cf figure 3.8). C'est un problème avec 18 points de demande et un nombre d'obstacles allant jusqu'à 12.

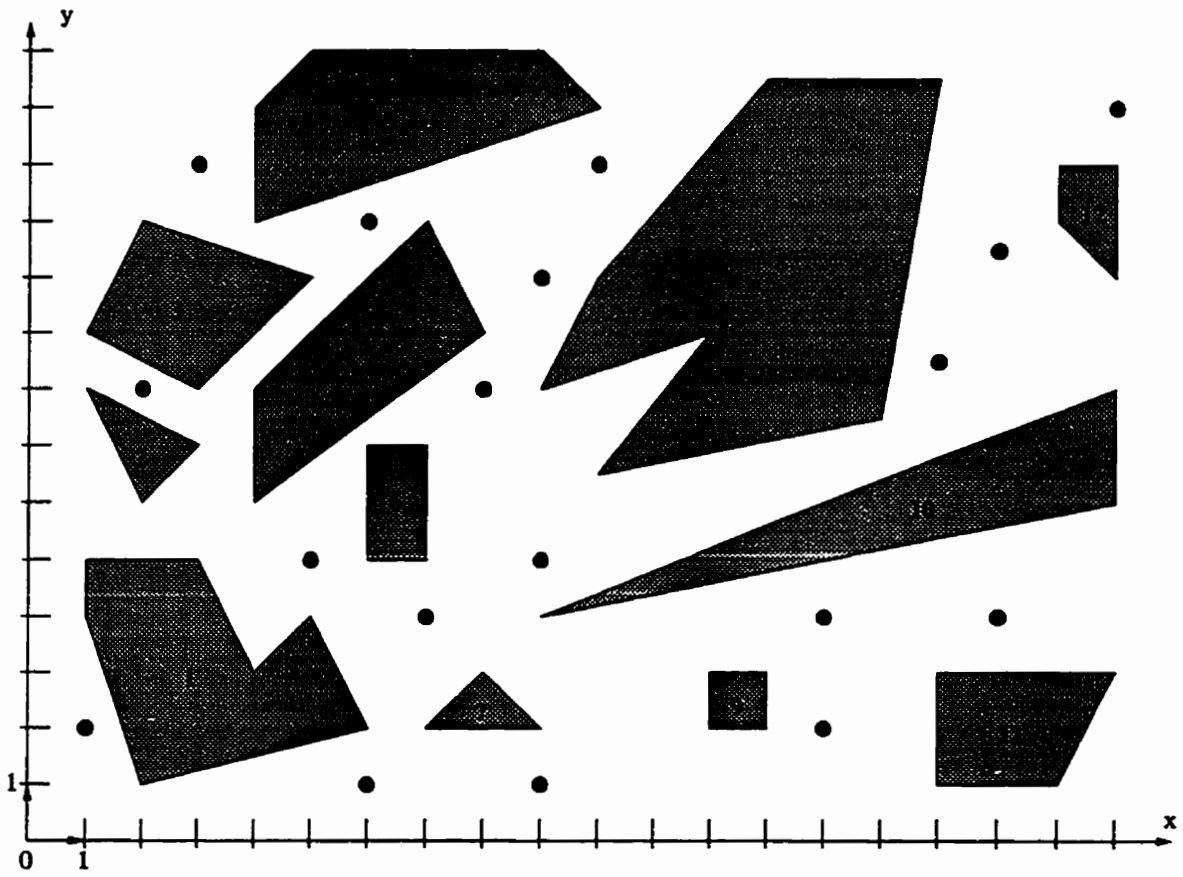


Figure 3.8 Exemple d'Aneja et Parlar

Tableau 3.1 Résultat pour le problème d'Aneja et Parlar

nb. d'obstacles	x^*	y^*	F^*	cpu (sec.)	it	opt
1-12	8.767	4.981	119.139	2.55	28	20
1-10	8.767	4.981	119.105	2.27	28	20
1-8	9.188	5.486	116.398	2.14	27	32
1-6	9.266	6.253	114.561	1.01	25	20
1-4	9.217	6.153	113.766	0.69	27	21
1-2	9.037	6.115	111.689	0.17	13	9
$\emptyset$	8.913	6.356	110.007	.01	1	4

La dernière colonne (opt) représente le nombre de carrés résolus optimalement (cardinal(J_E) = n). Ces résultats correspondent à ceux obtenus par Aneja et Parlar [2] (à 2 décimales près). Cependant on note que pour les problèmes avec 12 et 10 contraintes, le point optimal est le même alors que les auteurs obtiennent deux points différents.

Pour éprouver l'efficacité de l'algorithme en fonction du nombre de points de demande, le prochain tableau donne les résultats obtenus sur des problèmes de taille allant jusqu'à 1000 points de demande avec les 12 contraintes de passage du problème d'Aneja et Parlar [2]. Cinq problèmes sont résolus pour chaque série.

Tableau 3.2 Résultats pour $n = 100...1000, m_2 = 12$

n	m_2	itérations		temps		F^*		opt. carré	
		moy.	σ	*moy.*	σ	*moy.*	σ	*moy*	σ
100	12	47.6	3.	23.7	1.7	776.1	26.7	20.	4.9
200	12	52.4	9.9	55.5	11.6	1533.2	55.7	11.8	2.9
300	12	50.2	2.1	83.8	3.7	2319.3	54.4	10.4	2.6
400	12	52.2	2.6	126.1	7.6	3103.3	71.2	6.	1.6
500	12	52.2	2.5	166.7	9.4	3901.9	76.8	4.6	1.2
600	12	55.4	7.4	230.7	42.1	4680.6	62.5	4.2	2.4
700	12	56.2	6.	287.4	41.6	5463.2	63.4	3.2	1.7
800	12	59.8	5.1	376.9	45.8	6271.2	53.9	2.	1.6
900	12	56	4.2	415.3	48.5	7047.8	52.	2.4	1.6
1000	12	60.4	8.3	544.2	102.9	7838.4	39.4	1.2	1.1

Le dernier tableau concerne la sensibilité de l'algorithme au nombre de contraintes. Nous présentons les résultats obtenus pour des problèmes ayant 100 points de demande et un nombre de 50 ou 100 contraintes de passage (carrés de dimension alátoire). Chaque série est formée de 5 problèmes tests.

Tableau 3.3 Résultats pour $n = 100$, $m_2 = 50, 100$

n	m_2	itérations		temps		F^*		opt. carré	
		moy.	σ	*moy.*	σ	*moy.*	σ	*moy*	σ
50	100	83.8	43.4	127.1	66.4	629.4	315.8	25.2	14.1
100	100	105	64.1	414.6	246.9	643.4	323.7	67.2	61.2

L'algorithme proposé résout exactement et pour la première fois le problème de Weber avec des contraintes de passage. Le temps de calcul demeure modéré pour les problèmes de grande taille. La sensibilité au nombre de contraintes de passage est plus marquée à cause de l'explosion de l'arbre de visibilité.

CHAPITRE 4
Le problème de Weber multi-sources

4.1 Introduction

La généralisation du problème de Weber simple conduit à deux problèmes largement étudiés dans la littérature: le Weber multiple et le Weber multi-sources.

Le problème du Weber multiple est de localiser des sources afin de minimiser la somme des distances pondérées des sources aux points de demande et des sources aux sources. C'est un problème bien connu en programmation convexe pour lequel un algorithme polynômial exact a été récemment développé (Rosen *et al.* [65]).

Le problème de Weber multi-sources consiste à localiser p sources et à allouer n points de demande aux p sources sous la contrainte et avec l'objectif suivant:

- Relier chaque point de demande à une et une seule source.
- Minimiser la somme des distances pondérées entre les points de demande et les sources auxquelles elles sont reliées.

Une particularité du problème de Weber multi-sources est que si l'allocation optimale des points de demande aux sources est connue, les localisations des sources sont obtenues rapidement: il suffit pour chaque ensemble de la partition de résoudre le problème de Weber simple correspondant. Réciproquement, si les localisations des

sources sont connues, l'allocation optimale est déduite immédiatement en associant à chaque point de demande la source la plus proche.

A la différence du problème de Weber simple, le problème de Weber multi-sources est NP-dur (Meggido *et al.* [55]). Le nombre de partitions est égal au nombre de Stirling du second ordre (n : nombre de points de demande, p : nombre de sources):

$$S(m,p) = 1/m \sum_{K=0}^{m} \left(\begin{array}{c} m \\ K \end{array} \right) (-1)^K (m-K)^n.$$

qui est rapidement très élevé, par exemple $S(15,3) = 2375101$.

Le problème est bien résolu pour $p = 2$ par plusieurs méthodes: Chen *et al.* [16], Ostresh [60] , Drezner [24]. Les problèmes avec $n = 10000$ sont résolus en environ 45 minutes sur Sparc10 [16]. Pour $p = 3$ et $n = 50$ Ostresh [61] résout optimalement des problèmes de la littérature en environ 20 minutes; Chen [16] dans le même temps résout des problèmes avec $n = 30$ et Kuenne *et al.* [49] en quelques minutes de plus des problèmes avec $n = 25$. Lorsque $p > 3$ seul l'algorithme de Rosing [66] arrive en un temps raisonnable à résoudre des problèmes avec $n = 30$ et $p = 5$ ou $n = 25$ et $p = 6$. Pour des tailles plus importante on a bien souvent recours à l'utilisation d'heuristiques.

Dans ce chapitre, nous proposons de résoudre exactement le problème de Weber multi-sources par la technique de génération de colonnes [36]. Le problème auxiliaire déterminant la (les) colonne(s) de coût minimum est un problème de Weber simple avec distance maximum (cf Love *et al.* [52] Drezner *et al.* [32]). Il peut être reformulé en un problème de minimisation d'une fonction concave sur un convexe.

Nous commencerons par faire la revue de la littérature en mettant en évidence les principales méthodes existantes (section 4.2). La seconde partie de ce chapitre

sera consacrée à la présentation de l'algorithme de génération de colonnes, du problème auxiliaire et de sa résolution. La dernière section portera sur les résultats obtenus lors de la résolution de problèmes tests de la littérature.

4.2 Revue de la littérature

4.2.1 Heuristiques

Une des principales heuristiques pour le problème du Weber multi-sources a été proposée en 1964 par Cooper [17]: étant donné une localisation initiale arbitraire des sources et une allocation des n points de demande aux sources les plus proches, la procédure de Cooper [17] est la suivante:

(1) Pour chaque source et les points de demande associés, calculer la localisation optimale de la source ainsi que le coût de l'ensemble (procédure de Weiszfeld modifiée). En déduire la valeur de la fonction objectif du problème de Weber multi-sources. Si aucune des sources ne change de position, fin. Sinon aller en (2).

(2) Réallouer les points de demande aux sources en suivant la règle: chaque point de demande est affecté à la source la plus proche. Aller en (1).

La procédure de Cooper dans son principe a été reprise par plusieurs auteurs (Eilon *et al.* [34] Murtagh *et al.* [56]). La fonction objectif du problème de Weber multi-sources n'est ni convexe ni concave (Cooper [17]) et possède en général un nombre important de minima locaux. L'obtention d'un minimum global est rendue difficile par le peu de relief de la région entourant le minimum (Rosing [67]). La recherche

d'une bonne heuristique semble être un bon compromis entre le temps de calcul et la qualité de la solution.

Eilon *et al.* [34] améliorent l'algorithme de Cooper en incluant un processus de décision à l'étape de réallocation. Ils résolvent plusieurs versions d'un problème de 50 points de demande et de 1 à 5 sources. En prenant 200 allocations initiales différentes ils obtiennent des solutions pouvant varier en termes de valeur de la fonction objectif jusqu'à 40.9 %.

Soit la formulation mathématique proposée par Ostresh [60]

$$\min_{(X_i \in P, i=1,p), a_{ij}} \sum_{i=1}^{p} \sum_{j=1}^{n} a_{ij} w_j d_j(X_i)$$

$$\begin{cases} \sum_{i=1}^{p} a_{ij} = 1 \ \forall j \\ a_{ij} \in \{ 0,1 \} \ \forall i,j \end{cases}$$

où:

- P : plan.
- a_{ij} = variable d'allocation qui vaut 1 si le point de demande a^j est affecté à la source X_i, qui vaut 0 sinon.
- $I = \{X_1 ... X_p\}$: ensemble des sources à localiser.
- $J = \{a^1 ... a^n\}$: ensemble des points de demande.
- w_j : poids affecté au point de demande a^j.
- $d_j(X_i)$: distance euclidienne entre le point de demande a^j et la source X_i.

C'est un problème d'optimisation en variables mixtes, dont la fonction économique n'est pas dérivable en les points de demande et qui n'est pas continue lorsqu'une variable d'allocation change de valeur. Il y a $2n + np$ variables au problème.

Chen [14] propose de réduire le nombre de variables en approximant la fonction objectif:

$$\min_{X_i \in P, i=1..p} \sum_{j=1}^{n} w_j \min_{X_i \in I} \{d_j(X_i)\}$$

A l'aide de la propriété (Charalambous et Bandler [13]) du minimum de plusieurs fonctions positives :

$$\min\{C_1, ..., C_N\} < \{\sum_{j=1}^{p} C_j^{-N}\}^{-1/N} \quad \text{N : entier}$$

Cette approximation fait disparaître la partie allocation du problème, il reste $2n$ variables. En prenant N=100 et une méthode de descente (quasi Newton :Broyden-Fletcher-Shanno [4]), des problèmes de taille $n = 30$ et $p = 5$ sont résolus approximativement.

Love et Juel [53] reformulent la fonction objectif en son programme dual associé:

$$G^*(a) = \max_U g(U) = -\sum_{j=1}^{n} \sum_{i=1}^{p} a^j U_{ij}$$

sujet à :

$$\begin{cases} \sum_{j=1}^{n} U_{ij} = 0, i = 1, ...p \\ \|U_{ij}\| \leq a_{ij}, j = 1....n, i = 1....p \end{cases}$$

où :

$U_{ij} = (u_{ij}1, u_{ij}2)\ \forall i, j$:

$G^*(a)$ est une fonction concave [53]. Le problème de Weber multi-sources se reformule en une minimisation d'une fonction concave sur un convexe:

$$\min G^*(a)$$

sujet à

$$D \begin{cases} \sum_{i=1}^{p} a_{ij} = 1, j = 1.....n \\ a_{ij} \geq 0, j = 1....n, i = 1....p \end{cases}$$

Le domaine convexe D est formé d'inégalités liées aux variables d'allocations. Une fonction concave atteint son minimum en un de ses points extrêmes (Horst and Tuy [46]). Le parcours des points extrêmes adjacents ne nous garantit pas d'obtenir un minimum global. Ces points extrêmes sont définis par les variables d'allocations. Les auteurs développent, à partir d'une solution initiale, 5 heuristiques permettant des échanges de points de demande entre les sources. Cela revient à parcourir des points extrêmes du domaine qui ne sont pas adjacents, à augmenter l'étendue de la recherche.

Brimberg et Mladenovic [7] incluent dans l'une de ces 5 heuristiques une méthode tabou. Cette dernière permet, lorsqu'il n'existe plus de direction de descente, d'accepter des échanges faisant croître la fonction objectif (en interdisant les échanges inverses par la suite durant un nombre donné d'itérations à l'aide d'une liste tabou) jusqu'à trouver d'autres échanges faisant décroître la fonction objectif. Un parcours d'optima locaux est envisageable en dimensionant convenablement la liste tabou. On constate, sur des problèmes de la littérature (Ruspini $n = 75$ [68], Eilon *et al.* $n = 50$ [34]), une amélioration sensible des résultats en comparaison des autres méthodes. Les auteurs [7] développent une heuristique dont l'idée est de partir de la solution obtenue par la méthode de Cooper couplée avec un "Multi Start" puis de définir un voisinage (variable) autour du vecteur solution et de regarder s'il existe une direction de descente. S'il y en a une, une meilleure solution est trouvée et l'algorithme se répète. S'il y en a plus, une méthode tabou est appelée et une direction de montée est trouvée. Une nette amélioration des résultats concernant des problèmes tests de la littérature ($n = 75$: [68]), ($n = 50$: [34]) (en comparaison avec l'algorithme de Eilon et al [34]) est établie.

Pour des problèmes de plus grande taille $n > 200$, Hansen *et al.* [45] proposent de résoudre d'abord un problème de la p médiane associé (NP-dur) : la différence

avec le problème du multi-sources est que les sources doivent être localisées en des points préalablement définis. Les points de demande sont choisis comme candidats à cet effet. Le problème de la p-médiane ainsi défini est très bien résolu par l'algorithme de Hanjoul *et al.* [43]. Une partition des points de demande est obtenue et les coûts de chaque ensemble sont calculés par la procédure de Weiszfeld. Une solution du problème multi-sources est ainsi obtenue. Une récente comparaison de cet algorithme avec ceux de Bongartz *et al.* [6], Eilon *et al.* [34] montre pour des problèmes de grande taille (n = 287, p = 2.....50) et (n = 654, p = 2.....50) que cette heuristique accumule pour la première série de problèmes une erreur sur la meilleure valeur connue de 0.47%, de 0.31% pour la seconde série tandis que les deux autres heuristiques accumulent respectivement 507%, 137.36% et 487.58%, 126.96% d'erreur.

Murtagh *et al.* [56] proposent de résoudre le programme en relaxant les variables d'allocations. La formulation est la même que celle d'Ostresh [60] et de Bongartz *et al.* [6]. C'est un problème de minimisation d'une fonction non linéaire avec contraintes linéaires. Les auteurs choisissent le logiciel MINOS [56] pour résoudre le problème relaxé. Une localisation initiale des sources est faite puis le problème est résolu par MINOS avec les sources fixées. L'allocation optimale des points de demande aux sources est obtenue. Cette solution est choisie comme point de départ de la méthode de descente en considérant toutes les variables du problème relaxées. Une solution entière est obtenue en associant à chaque point de demande la source la plus proche.

Bongartz *et al.* [6] relaxent eux aussi les variables d'allocations. Ils exploitent cependant davantage la structure du problème en définissant des directions de descente:

d'après un vecteur solution (variables X_i et a_{ij} $\forall i, j$), un sous espace est défini par

des contraintes d'allocations saturées ($a_{ij} = 0$ où1) et par des localisations des sources sur des points de demande (les lieux où la fonction est non dérivable). Ils établissent des conditions nécessaires (à l'intérieur du sous espace) que doit vérifier un point stationnaire et simultanément obtiennent une direction de descente pour chacune de ces conditions violées. Arrivé à un optimum local, les contraintes d'allocations et les localisations des sources peuvent varier et un autre sous espace peut être obtenu. Une recherche similaire d'un point stationnaire et d'un minimum local est réalisée dans le nouveau sous-espace si une première direction de descente est trouvée. Cette méthode est comparée avec 3 autres méthodes existantes: méthode de Cooper [17] améliorée (Eilon *et al.* [34], Murtagh *et al.* [56], Chen. [14]). Les résultats obtenus sur 31 problèmes test ($n \leq 50$ et $p \leq 10$) montrent que la méthode de Bongartz *et al.* [6] est la plus performante puisqu'elle atteint à chaque fois la meilleure valeur connue de chaque problème contrairement aux autres accumulant les erreurs respectives 2.810%, 4.434%, 2.262 %.

4.2.2 Méthodes exactes

L'algorithme exact qui résout les problèmes de plus grandes tailles ($p > 3$) est celui de Rosing [66] dont le principe est une généralisation de l'algorithme d'Ostresh [60]. La méthode d'Ostresh [60] ($p = 2$) est basée sur la nécessité d'avoir à l'optimum une bipartition d'ensembles disjoints de points de demande (polygones de Thissen). Or il existe $4n(n-1)/2$ bipartitions d'ensembles disjoints, obtenues en prenant toutes les paires de points et leur droite associé (droite passant par les deux points). Chacune de ces droites sépare le plan en deux demi plans définissant une bipartition. Le facteur 4 est le nombre de possibilités de mettre 2 points de demande dans 2 ensembles. Le coût et la localisation de la source associée à la bipartition sont obtenus en résolvant un problème de Weber simple sur chacun des ensembles.

Pour $p > 2$ la seule méthode exacte existante résolvant des problèmes de taille moyenne a été proposée par Rosing [66]. Cet algorithme est une généralisation de ceux d'Ostresh [60] et de Drezner [24] à plus de 2 facilités. Il consiste à diviser le plan de toutes les façons possibles en deux demi-plans (deux ensembles de points de demande) à l'aide d'une droite passant par au moins deux points de demande. Chaque ensemble obtenu à l'étape précédente est ensuite divisé de la même manière. L'opération se répète jusqu'à obtenir toutes les partitions possibles de p ensembles disjoints: tous les ensembles convexes disjoints éligibles N_e dont le nombre est très inférieur à celui des partitions ($n = 15$ et $p = 3$, $S(n,p) = 2375101$, $N_e(n,p) = 3200$). Le problème est ensuite résolu par un programme linéaire en nombre entiers du type problème de partitionnement avec contrainte additionnelle de cardinalité de la solution où chaque colonne de la matrice des n premières contraintes représente un ensemble de points de demande. Le problème est résolu par une relaxation linéaire puis par une méthode d'énumération implicite si la solution de la relaxation continue est fractionnaire. Notons que les solutions des problèmes tests proposés par Rosing [66] ($n = 25, p = 6$ et $n = 30, p = 5$) sont obtenues sans branchement.

Soit E_t un ensemble composé d'au moins un point de demande et c_t le coût associé à cet ensemble. Soit y_t une variable de décision associée à cet ensemble qui vaut 1 si E_t est choisi et 0 dans le cas contraire. A l'ensemble E_t on associe une colonne t dont la $i^{\text{ième}}$ ligne vaut 1 si le point de demande a^j appartient à l'ensemble E_t, et 0 sinon.

$$\min_{y_t} \sum_{t=1}^{N_e} c_t y_t$$

sujet à:

$$\begin{cases} \sum_{t=1}^{N_e} b_{jt} y_t \geq 1 & j = 1, 2..., n \\ \sum_{t=1}^{N_e} y_t = p \\ y_t \in \{0, 1\} \end{cases}$$

$$b_{jt} = \begin{cases} 1 & \text{si le point de demande } a^j \text{ est dans } E_t. \\ 0 & \text{autrement.} \end{cases}$$

$$y_t = \begin{cases} 1 & \text{si l'ensemble } E_t \text{ est choisi.} \\ 0 & \text{autrement.} \end{cases}$$

Les n premières contraintes expriment la condition que chaque point de demande est relié à une seule source. Le signe d'inégalité associé à chacune des contraintes est choisi afin d'avoir des variables duales positives. Notons qu'à l'optimum toutes les contraintes sont saturées (les coefficients c_j étant tous positifs). La dernière contrainte (contrainte d'égalité) assure un nombre p de sources.

Kuenne et Soland [49] proposent une méthode d'énumération implicite qui porte sur les affectations des points de demande aux sources. L'ensemble des solutions est partitionné en sous-ensembles. Chaque sous ensemble est caractérisé par des affectations de points de demande à des sources, des points de demande et des sources non encore affectés. La variable de branchement est l'affectation d'un point de demande à une source. Pour chaque groupe de points de demande (associés à une même source), le minimum de la fonction de Weber simple est trouvé et une borne inférieure sur les affectations des autres points de demande est calculée. Une borne supérieure (affectation réalisable de points de demande aux sources) est simultanément calculée. La taille maximale des problèmes résolus par cet algorithme

d'énumération implicite est : $(n = 15, p = 3)$.

Chen *et al.* [16] proposent une formulation D. C. (différence de fonctions convexe):

$$\min_{X_i \in P, i=1..p} \sum_{j=1}^{n} \sum_{i=1}^{p} w_j d_j(X_i) - \sum_{j=1}^{n} w_j \max_{i=1,2...p} \sum_{k=1(k\neq j)}^{p} d_j(X_k).$$

En introduisant une variable t, le problème se reformule en un programme de minimisation d'une fonction concave sur un convexe:

$$\min_{X_i \in P, i=1...p, t} t - \sum_{j=1}^{n} w_j \max_{i=1,2..p} \sum_{k=1(k\neq j)}^{p} d_j(X_k)$$

sujet à

$$D \begin{cases} \sum_{j=1}^{n} \sum_{i=1}^{p} w_j d_j(X_i) - t \leq 0 \\ t \geq 0 \ , \ (X_i) \in P \end{cases}$$

Chen *et al.* [16] ont développé une méthode d'approximation extérieure. Le domaine convexe D est approximé extérieurement par un polytope. A chaque itération le point minimum de la fonction est obtenu sur le polytope. Si ce point appartient au domaine, l'optimum est atteint. Sinon une coupe dans le polytope éliminant l'optimum est obtenue. Une solution réalisable est calculée simultanément. Le polytope est ainsi raffiné quand on passe à l'itération suivante. Le critère d'arrêt porte sur la différence relative entre la meilleure valeur connue et la borne inférieure (obtenue sur le polytope courant). Cette méthode est à la base de la résolution du problème auxiliaire par la méthode de génération de colonnes qui sera présentée dans les sections suivantes.

4.3 Formulation mathématique

Rappelons d'abord la notation utilisée:

(i) n : nombre de points de demande.

(ii) p : nombre de sources.

(iii) $J = \{a^1, a^2, ..., a^n\}$ l'ensemble des points de demande.

(iv) $I = \{X_1, X_2, ..., X_p\}$ l'ensemble des sources.

(v) (x_i, y_i) : coordonnées de la source X_i.

(vi) $s(x, y)$: point courant du plan.

(vii) $d_j(s)$: distance euclidienne entre a^j et s.

(viii) w_j : poids associé à a^j.

(ix) E : enveloppe convexe des points de demande.

(x) $\{u_1, u_2..., u_{n+1}\}$: variables duales optimales du programme linéaire courant.

Le modèle mathématique est semblable à celui proposé par Rosing [66] (cf section précédente) mais avec plus de colonnes:

$$P^{MW} \begin{cases} \min \sum_{t=1}^{2^n-1} c_t y_t \\ \text{sujet à:} \\ \begin{cases} \sum_{t=1}^{2^n-1} b_{jt} y_t \geq 1 & j = 1, 2..., n \\ \sum_{t=1}^{2^n-1} y_t = p \\ y_t \in \{0, 1\} \end{cases} \end{cases}$$

Le nombre de colonnes étant exponentiel en le nombre de points de demande $(2^n - 1)$, une résolution directe est à exclure dès que n devient suffisamment grand. On a

donc recours pour résoudre la relaxation continue du programme P^{MW} à la technique de génération de colonnes (Gilmore et Gomory [36]). Pour satisfaire la condition d'intégralité des variables y_t, la méthode sera combinée avec une procédure d'optimisation par séparation.

4.4 Méthode de génération de colonnes

4.4.1 Principe de la méthode

La résolution d'un programme linéaire par le technique de génération de colonnes se déroule en trois étapes:

(1) Initialisation de la matrice du programme linéaire.

(2) Résolution du programme linéaire courant. Sauvegarde des variables duales optimales.

(3) A l'aide des variables duales, construction des colonnes de coûts réduits négatifs (cas d'une minimisation) par la résolution d'un problème auxiliaire. Chargement des colonnes dans le programme et retour en (1). Si aucune colonne de coût réduit négatif n'est trouvée, l'optimum de la relaxation est atteint.

La convergence de la méthode vers un optimum global repose sur la capacité d'obtenir la colonne de coût réduit optimal (le plus négatif). Le choix des colonnes initiales fera l'objet d'une discussion dans la dernière section. En pratique la résolution du programme linéaire est effectuée par le logiciel CPLEX (méthode révisée du simplexe [58]). Le problème auxiliaire est ici un problème d'optimisation globale (minimisation d'une fonction concave sur un convexe), résolu exactement par une méthode d'approximation extérieure [16].

4.4.2 Initialisation du programme linéaire

Il existe plusieurs façons de choisir la base initiale. La première est de partir d'une solution réalisable du problème de Weber multi-sources puis de remplir la matrice de colonnes unité afin d'obtenir une base de départ. Une autre façon de procéder, qui fera l'objet d'une discussion (cf section 4.7), est de choisir une à une les colonnes de départ.

Dans tous les cas le programme linéaire initial peut être vu comme un programme ayant un nombre déterminé L où $(L > n+1)$ de colonnes:

$$P^0 \left\{ \begin{array}{l} \min\limits_{y_t} \sum\limits_{t=1}^{L} c_t y_t \\ \text{sujet à:} \\ \underbrace{\begin{bmatrix} . & . & b_{1t} & . & . \\ . & . & . & . & . \\ . & . & . & . & . \\ . & . & . & . & . \\ . & . & b_{nt} & . & . \\ 1 & 1 & 1 & 1 & 1 \end{bmatrix}}_{M^0} \begin{bmatrix} y_1 \\ . \\ . \\ . \\ y_L \end{bmatrix} = \begin{bmatrix} 1 \\ 1 \\ 1 \\ 1 \\ p \end{bmatrix} \\ y_t \geq 0 \qquad t = 1, \ldots L \end{array} \right.$$

Soit $(u_1^0, ..., u_{n+1}^0)$ le vecteur de variables duales à l'optimum du programme linéaire P^0.

4.4.3 Définition du problème auxiliaire

Soit un ensemble E_t de points de demande associé à la colonne $t = (b_{1t}, ..., b_{nt})$ et l'expression de son coût réduit:

$$\overline{c_t} = c_t - \sum_{j=1}^{n} b_{jt} u_j - u_{n+1}$$

L'équation permettant d'obtenir la colonne E_t^* de coût réduit minimum s'écrit:

$$\overline{c_{t^*}} = \min_t (c_t - \sum_{j=1}^{n} b_{jt} u_j - u_{n+1})$$

avec:

$$c_t = \min_{(x,y)\in E} \sum_{j=1}^{n} b_{jt} w_j d_j(s)$$

Proposition 5 *$\overline{c_{t^*}}$ peut être obtenu par une minimisation d'une fonction D.C. (différence de fonctions convexes).*

Preuve:

$$\overline{c_{t^*}} = \min_t (c_t - \sum_{j=1}^{n} b_{jt} u_j) - u_{n+1} \tag{4.1}$$

$$\overline{c_{t^*}} = \min_{x,y,b_{jt}} \sum_{j=1}^{n} b_{jt}(w_j d_j(x,y) - u_j) - u_{n+1} \tag{4.2}$$

$$\overline{c_{t^*}} = \min_{x,y} \sum_{j=1}^{n} \underbrace{\min(w_j d_j(x,y) - u_j, 0)}_{g^j(x,y)} - u_{n+1} \tag{4.3}$$

$$\overline{c_{t^*}} = \min_{x,y} \sum_{j=1}^{n} [\underbrace{(w_j d_j(x,y) - u_j)}_{g_1^j(x,y)} - \underbrace{\max(w_j d_j(x,y) - u_j, 0)}_{g_2^j(x,y)}] - u_{n+1} \tag{4.4}$$

$$\tag{4.5}$$

L'obtention de $\overline{c_{t^*}}$ est un programme D.C. car la somme de fonction D.C. est D.C.:

$$\overline{c_{t^*}} = \min_{x,y} [\sum_{j=1}^{n} g_1^j(x,y) - \sum_{j=1}^{n} g_2^j(x,y)] - u_{n+1}$$

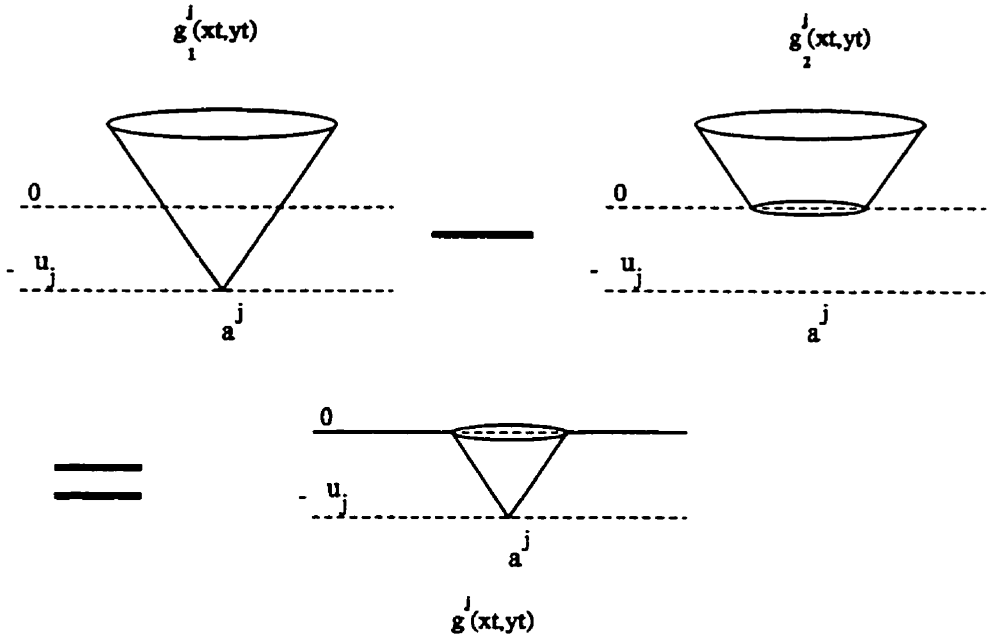


Figure 4.1 Illustration de la proposition.

Proposition 6 *Ce programme peut être reformulé en un programme de minimisation d'une fonction concave sur un convexe:*

Preuve:

En utilisant la même technique que dans Chen *et al.* [16]:

$$SP \begin{cases} \min\limits_{x,y,r} F_a(x,y,r) = r - \sum\limits_{j=1}^{n} \max(w_j d_j(x,y) - u_j, 0) - \sum\limits_{j=1}^{n} u_j - u_{n+1} \\ \text{sujet à:} \\ D \begin{cases} g(x,y,r) = -r + \sum\limits_{j=1}^{n} (w_j d_j(x,y)) \leq 0 \\ r \geq 0,\ u(x,y,r) \in E \times [0, \mathbb{R}^+], \end{cases} \end{cases}$$

On peut aussi voir le problème auxiliaire comme un problème de Weber avec distance maximum. Dans ce cas $g^j(x,y)$ est la fonction coût associée au point de

demande a^j. Ce type de problème est bien connu en théorie de la localisation. Il représente le cas où la fonction coût est linéaire en la distance puis devient constante au delà d'un certain rayon. Ce problème trouve son application lors de la localisation d'un centre de pompier ou de soins desservant des habitations (la fonction coût représente le fait qu'à partir d'une certaine distance, le centre de secours n'est plus d'aucune utilité pour l'habition concernée). C'est un problème de Weber avec une fonction de coût concave ou quasi-concave en la distance qui peut être résolu par la méthode d'énumération implicite proposée dans le deuxième chapitre. Nous allons dans la section suivante présenter seulement la méthode D.C..

4.4.4 Résolution du problème auxiliaire, obtention de la colonne de coût réduit minimum

Soit l'algorithme $A1$ résolvant le problème SP analogue à celui de Chen *et al.* [16]:

Algorithme A_1 :

Initialisation Soit un polytope $P_1 \supseteq \{(x,y) \in E\} \times [0,R]$. Soit $w(x_w, y_w, r_w)$ un point intérieur à D et $F_a(x_w, y_w, r_w)$ la valeur de la fonction objectif associée. Trouver la liste $V(P_1)$ des sommets du polytope P_1. $k = 1$.

Itération :

(1) Trouver le minimum de la fonction sur le polytope courant : $v^k \leftarrow \arg\, min F_a(V(P_k))$; $\underline{F_a} \leftarrow F_a(v^k)$; si $v^k \in D$ fin, le minimum est trouvé: $F_a^* = F_a(v^k)$, $U^* = v^k$, aller en (5).

(2) Trouver un point d'intersection (ou proche de l'intersection) p^k du segment $[w, v^k]$ avec $g(x,y,r) = 0$ tel que $p^k \in intD$.

Soit $\nu^k \in [w, p^k]$ satisfaisant $\|\nu^k - p^k\| \leq 2\ \eta\ \eta > 0$ et $\nu^k \in D$.

Si $F_a(\nu^k) < F_a^*$ alors $U^* = \nu^k$, $U^* = (x^*, y^*, r^*)$; $F_a^* = F_a(\nu^k)$;

Si $(F_a^* - \underline{F_a})/F_a^*$ aller en (5).

(3) Trouver l'équation du plan $l_k(p)$ passant par p^k à l'aide du vecteur gradient de g au point p^k (si p^k est un point extrême de D), du sous-gradient de g au point p^k si p^k est proche de D. Soit $L_k(p)$ le demi-espace borné par $l_k(p)$ ne contenant pas p^k.

(4) $P_{k+1} \leftarrow P_k \cap l_k(p)$. Mise à jour de la liste $V(P_{k+1})$.

$k \leftarrow k + 1$. Aller en (1).

(5) Colonne entrante:

$$\begin{pmatrix} b_{1t^*} \\ b_{2t^*} \\ \cdots \\ b_{nt^*} \\ 1 \end{pmatrix}$$

$$\begin{cases} b_{jt^*} = 1 & \text{si } d_j(x^*, y^*) < u_j/w_j. \\ b_{jt^*} = 0 & \text{sinon.} \end{cases}$$

fin itération

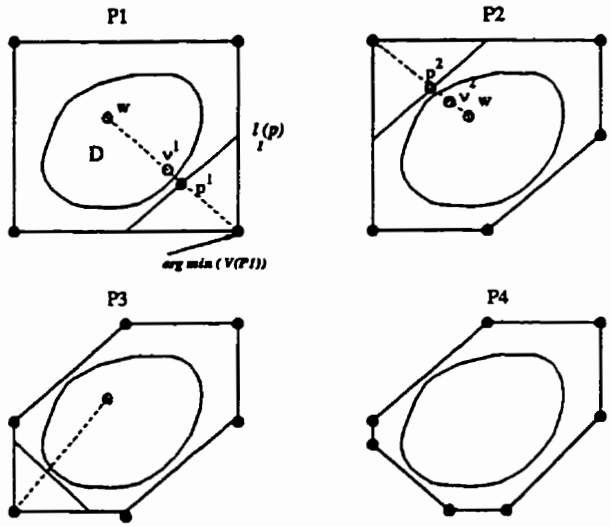


Figure 4.2 Illustration de la méthode

Il peut être utile de fournir durant un certain nombre d'itérations des colonnes de coût réduit négatif en plus de celle de coût réduit minimum. Il suffit lors de la résolution de SP par l'algorithme $A1$ de sauvegarder les points dont l'image par la fonction objectif est négative. Soit $U = (x, y, t)$, un point obtenu lors de la procédure $A1$. La procédure d'accélération $A2$ suivante construit une colonne de coût réduit négatif en partant de (x, y).

Algorithme $A2$:

(1) $E_s = \emptyset$. Pour tout $j = 1...n$ faire:
Si $d_j(x, y) < u_j/w_j$: $E_s = E_s \cup \{a^j\}$. Fin faire.

(2) Calcul du point de Weber simple associé à l'ensemble E_s. Soit (x_s, y_s) ce point.

(3) Pour tout $j \in J$, $cpt = 0$ faire:
Si $d_j(x_s, y_s) \leq u_j/w_j$ et $a^j \in E_s$: $E_s = E_s \setminus \{a^j\}$, $cpt \leftarrow cpt + 1$.
Si $d_j(x_s, y_s) < u_j/w_j$: $E_s = E_s \cup \{a^j\}$, $cpt \leftarrow cpt + 1$.

(4) Si $cpt > 0$ aller en (2) sinon la colonne entrante est :

$$\begin{pmatrix} b_{1t} \\ b_{2t} \\ \dots \\ b_{nt} \\ 1 \end{pmatrix}$$

$$\begin{cases} b_{jt} = 1 & \text{si } d_j(x_s, y_s) < u_j^k / w_j. \\ b_{jt} = 0 & \text{sinon.} \end{cases}$$

L'idée de cet algorithme est de partir d'un ensemble de points de demande dont la colonne associée a un coût réduit négatif et d'obtenir (en enlevant des points de demande à cet ensemble) une colonne ayant un coût réduit plus négatif (cf figure 4.3).

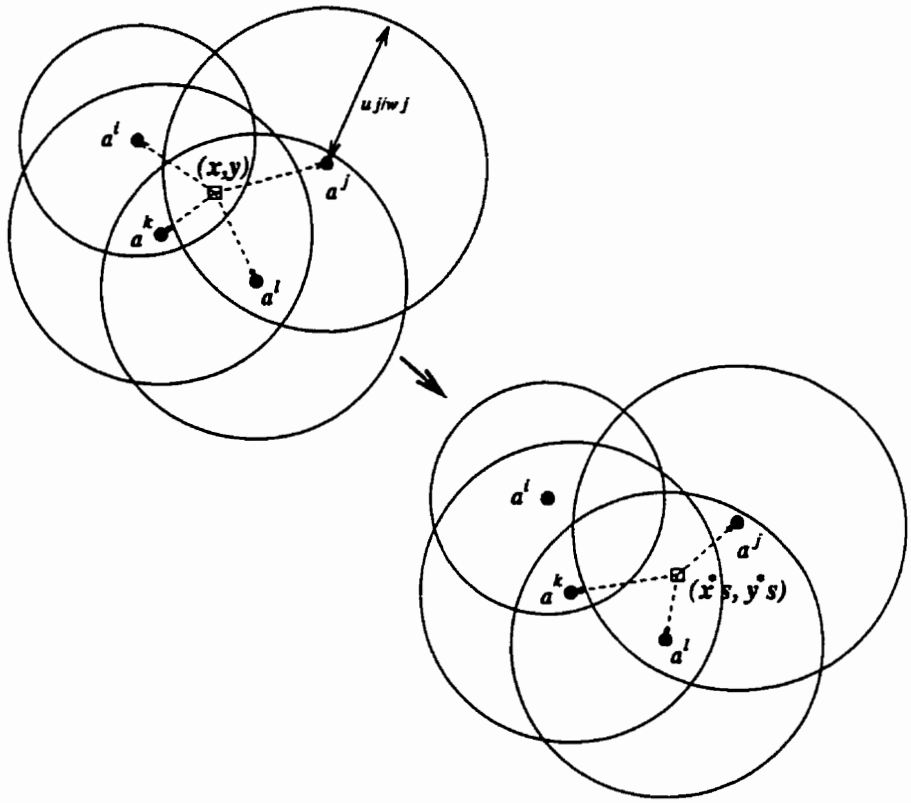


Figure 4.3 Illustration de l'algorithme $A2$

4.5 Règle de branchement

Si tous les vecteurs solutions ($y_t > 0$) sont entiers, le problème de Weber multi-sources est résolu . S'il existe plusieurs variables y_t ($0 < y_t < 1$), le problème est fractionnaire et on doit faire appel à une procédure de branchement.

La règle de branchement utilisée est celle proposée par Ryan et Foster [69] pour résoudre le problème de partitionnement (sans génération de colonnes). Elle consiste, à partir de 2 points de demande, de partitionner l'ensemble des solutions en deux parties: une partie contenant les ensembles de points de demande vérifiant la propriété: si un des deux points de demande s'y trouve, l'autre s'y trouve aussi,

et une autre partie regroupant les ensembles de points de demande vérifiant: si un des deux point de demande s'y trouve, l'autre ne s'y trouve pas. Soient a^j and a^l ces deux points de demande. La formulation mathématique de ce branchement est (pour toute colonne t appartenant au programme courant):

$$\textbf{branche 1} : b_{jt} = b_{lt} \in \{0, 1\}.$$

$$\textbf{branche 2} : b_{jt} + b_{lt} \leq 1.$$

Le choix des points de demande (a^j et a^l) dépend des colonnes fractionnaires ($0 < y_t < 1$) dans la base optimale du programme linéaire courant. Lorsqu'une base optimale est obtenue, le choix des points de demande doit répondre aux deux exigences suivantes:

- Il y a une ou plusieurs colonnes fractionnaire(s) de la base optimale qui incluent les deux points de demandes (ou aucun des deux).
- Il y a une ou plusieurs colonnes fractionnaire(s) de la base optimale qui incluent seulement un des deux points de demande.

En pratique, on pénalise par un fort coefficient (afin de ne pas perdre la réalisabilité) les colonnes de la matrice courante ne satisfaisant pas la condition choisie des modifications à la formulation du problème auxiliaire.

4.5.1 Modification du problème auxiliaire après branchement

Soit $G^1(X^1, E^1)$ le graphe dont les sommets représentent les points de demande et les arêtes les contraintes d'égalité (du type branche 1). C'est un graphe à plusieurs composantes connexes. Chaque composante connexe du graphe regroupe les sommets dont les variables b_{jt} associées prennent la même valeur. Soit $B1^k = (g_1^k, ..., g_{m_k}^k)$

l'ensemble des composantes connexes de G^1 regroupant plus de 2 points de demande (ce qui évite de considérer les points de demande qui ne sont pas liés aux contraintes d'égalité).

Soit $G^2(X^2, E^2)$ le graphe issu du graphe G^1 en fusionnant les sommets d'une même composante connexe. Soit E^2 l'ensemble des arêtes de G^2 représentant les contraintes d'inégalités (du type branche 2). Soit $B2^k = (e_1^k, ..., e_{n_k}^k)$ l'ensemble des ensembles transversaux minimals de G^2.

Soit l'expression de la fonction objectif du problème auxiliaire correspondant à la résolution du programme linéaire P^k :

$$\min_{x,y} \sum_{j \in B1^k} \min(w_j d_j(x,y) - u_j^k, 0) + \sum_{j \notin B1^k}^{n} \min(w_j d_j(x,y) - u_j^k, 0).$$

En regroupant ensemble les fonctions coûts des points de demande d'une même composante connexe de $B1^k$, la fonction objectif réarrangée s'écrit :

$$\min_{x,y} [\min(\sum_{j \in g_1} (w_j d_j(x,y) - u_j^k), 0) + ... + \min(\sum_{j \in g_{m_k}} (w_j d_j(x,y) - u_j^k), 0)$$
$$+ + \sum_{j \notin B1^k}^{n} \min(w_j d_j(x,y) - u_j^k, 0)]$$

4.6 Algorithme général

Soient les principales notations suivantes:

(i) I_ℓ: liste des indices des relaxations des problèmes à traiter.

(ii) P^0 : relaxation linéaire de P^{MW}.

(iii) Y^0 : vecteur solution optimal de la relaxation continue de P^0.

(iv) F_r^0 : valeur optimale de la fonction objectif de P^0.

(v) P^k $(k > 0)$: programme linéaire à variables entières à l'itération k.

(vi) P_i^k : programme P^k réduit par l'élimination de points de demande dans le problème auxiliaire.

(vii) $\underline{F_k}$ $(k > 0)$: borne inférieure sur la valeur optimale de la fonction objectif de P^k.

(viii) F^k : valeur optimale de la fonction objectif de P^k.

(ix) Fi_r^k: valeur optimale de la relaxation continue associé au programme P_i^k.

(x) Yi^k: vecteur solution optimal de la relaxation continue de Pi^k.

(xi) F^k : valeur optimale de la fonction objectif de P^k.

(xii) I_F: liste des valeurs des fonctions objectifs des relaxations des problèmes de la liste I_ℓ.

(xiii) $\{a_k^j, a_k^l\}$: variables de branchement issues de P^k.

(xiv) M^k: matrice de colonnes associé à la relaxation de P^k.

(xv) $\{u_1^k, ..., u_{n+1}^k\}$: variables duales optimales de la relaxation du programme linéaire P^k.

(xvi) F^*: valeur de la fonction objectif de P^{MW} associée à Y^*.

Y^*: meilleur vecteur solution entier connu du programme P^{MW}.

Les principales étapes de l'algorithme sont les suivantes:

(1) On résout optimalement par la technique de génération de colonnes la relaxation linéaire du programme initial P^{MW} (cf section 4.3, 4.4). Soit Y^0 son vecteur solution optimale associé à F_r^0. Si Y^0 est un vecteur binaire alors le programme

P^{MW} est résolu ($F^* = \underline{F}^0$ et $Y^* = Y^0$) et l'algorithme s'arrête. Sinon on sauvegarde le programme P^0 et on sélectionne à l'aide de la base optimale deux points de demande pour le branchement (cf section 4.5).

(2) On résout par la méthode de génération de colonnes les relaxations continues des deux problèmes issus de P^0 et des conditions de branchement (cf section 4.5). Si une (resp. deux) relaxation(s) admet(tent) un vecteur solution optimal et fractionnaire, deux (resp. quatre) nouveaux sous-problèmes (se caractérisant des deux précédents par une condition de branchement supplémentaire) sont créés. Si une des deux relaxations (ou les deux) admet(tent) une solution entière, une mise à jour si nécessaire de la meilleure valeur connue et du meilleur vecteur solution connu est obtenue.

(3) A chaque itération, le sous-problème ayant une borne inférieure minimum est sélectionné. Sa relaxation continue est résolu par la technique de génération de colonnes. Si la solution obtenue est entière, on mets à jour la meilleure valeur connue. Les sous-problèmes ayant une borne inférieure plus grande que la meilleur valeur connue sont éliminés. Les autres sous-problèmes sont sauvegardés dans une liste. L'algorithme s'arrête lorsqu'il n'y a plus de problèmes dans la liste.

Soit le schéma général de l'algorithme ainsi que ses procédures principales:

Initialisation $k = 0$, $B1^0 = \emptyset$, $B2^0 = \emptyset$. Initialisation des colonnes de la matrice M^0. $I_\ell = \{0\}$. Résolution de P^0 (cf section 4.4.2) par l'algorithme C.

Si Y^0 est entier, le problème de Weber multi-sources est résolu : $Y^* = Y^0$ et $F^* = F_r^0$.

Sinon: $\underline{F}^0 = F_r^0$, $I_F = \{F_r^0\}$, $I_\ell = \{0\}$. Soit $\{a_0^r, a_0^s\}$ la paire de points de demande issue de la base optimale de la relaxation linéaire du programme P^0. Sauvegarde de la matrice finale M^0 de ce programme. $N = 0$.

Itération tant que $I_\ell \neq \emptyset$:

Sélection de l'indice du programme linéaire à construire :

$\{k \in I_\ell | \underline{F}^k = \min_h(\underline{F}^h)\ h \in I_F\}$. $I_\ell = I_\ell \setminus k$. $I_F = I_F \setminus \underline{F}^k$.

branche 1 $b_{rt} = b_{st} \in \{0,1\}$, $N \leftarrow N + 1$:

$B1^N \leftarrow B1^k$, $B2^N \leftarrow B2^k$, Mise à jour de l'ensemble $B1^N$. Aller en procédure B.

branche 2 $b_{rt} + b_{st} \leq 1$, $N \leftarrow N + 1$:

$B1^N \leftarrow B1^k$, $B2^N \leftarrow B2^k$, Mise à jour de l'ensemble $B2^N$. Aller en procédure B.

fin itération

Procédure B (traitement de P^N issu du branchement) :

(1) $\underline{F}^N = \infty$, $M^N \leftarrow M^0$. Pénaliser les colonnes de M^N ne satisfaisant pas aux conditions des ensembles $B1^N$ et $B2^N$. Pour chaque e_i dans $B2^N$ ou si $B2^N = \emptyset$, :

faire:

(2) Modifier la fonction objectif du problème auxiliaire et éliminer du problème auxiliaire les points de demande appartenant à e_i : $J = J \setminus \{j \in e_i\}$.

(3) Obtention de Fi_r^N et Yi^N par la procédure C.

(4) Si Yi^N entiers et $F^* > Fi_r^N$ alors $F^* = Fi_r^N$, $Y^* = Yi^N$. $J = J \cup \{j \in e_i\}$. Si Yi^N fractionnaire et $Fi_r^N < \underline{F}^N$ alors $\underline{F}^N \leftarrow Fi_r^N$, $J = J \cup \{j \in e_i\}$. **Fin faire.**

(5) Si Y_{n_N} fractionnaire et $\underline{F}^N < F^*$, choisir les variables de branchement (a_N^r, a_N^s). $I_\ell = I_\ell \cup N$. $I_F = I_F \cup \underline{F}^N$.

Procédure C (étapes de la méthode de génération de colonnes) :

(1) Résolution par CPLEX du programme linéaire courant (méthode révisée du simplexe). Soit $(u_1^k, ..., u_{n+1}^k)$ les variables duales, F^k la valeur de la fonction objectif, X^k le vecteur solution, à l'optimum.

(2) Génération de colonnes de coût réduit négatif par l'algorithme A et la méthode d'accélération B (cf section précédente). S'il en existe aucune, fin.

(3) Chargement des colonnes dans la matrice du programme linéaire courant, retour en (1).

Une solution entière est obtenue à la fin de chaque relaxation linéaire en prenant les ensembles de points de demande associés aux variables y_t valant 1 et en résolvant par l'heuristique de Cooper [17] (location-allocation) le problème de Weber multi-sources avec les points de demande et les sources restants.

4.7 Résultats numériques

Nous avons vu que l'on pouvait résoudre le problème auxiliaire par un algorithme D.C. ou par un algorithme d'énumération implicite (voir chapitre 2). L'algorithme D.C. est programmé en langage fortran, l'algorithme d'énumération et le programme maître sont écrits en langage C. Les résultats sont obtenus sur Sparc10 lorsque $n <= 50$, sur Sparc20 pour $n > 50$. Tous les temps de calcul sont donnés en secondes. Nous présentons en premier lieu les résultats concernant les problèmes déjà testé dans la littérature : Eilon *et al.* [34] (50 points) , Rosing [66] (30 points).

Une étude a été faite sur l'influence des paramètres: nombre de colonnes lors de l'initialisation de la matrice du programme P^0, qualité d'une solution initiale, tolérance sur les coûts réduits des colonnes entrantes, nombre de colonnes entrantes. Pour la meilleure combinaison de ces paramètres, quelques problèmes générés aléatoirement avec un nombre de points de demande variant de 50 à 150 et des sources variant de 5 à 50 sont résolus.

Le temps de résolution pour résoudre un problème par la technique de génération de colonnes dépend de la matrice initiale et de la complexité de l'algorithme de résolution du problème auxiliaire. Nous avons utilisé l'algorithme D.C. pour la résolution de problèmes ($n \leq 150$) puis celui d'énumération implicite lorsque $n = 287$ (Bongartz *et al.* [6]) en ayant remarqué auparavant que ce dernier était le plus rapide des deux.

Plusieurs choix de matrice initiale (M^0) sont suggérés:

(1) base M^0 est une matrice identité.

(2) base + heur. M^0 est formé de p colonnes, solution heuristique du problème

de Weber multi-sources et (n+1- p) autres colonnes issues de la matrice identité.

(3) base + heur. + col. M^0 est composée des p premières colonnes de (2) et de colonnes issues de celles-ci de la manière suivante:

- toutes les colonnes possibles semblables aux p premières exceptées pour une ou deux lignes.

(4) base + heur. + col. + mul. Le problème auxiliaire SP génère plusieurs colonnes ("multiple pricing")

(5) base + heur. + col. + mul. + tol. Excepté pour la dernière itération, la colonne optimale (de coût réduit minimum) n'est pas nécessairement calculée.

Les tables suivantes représentent les résultats pour les données de Rosing [66] et de Eilon *et al.* [34]. pour chacun des cas précédemment définis.

Tableau 4.1 Tests sur les problèmes de Rosing: $n = 15..30, p = 2....6$.

n	p	OBJ	base				base + heur.			
			it.	cpu tot.	cpu prob.	cpu cplex	it.	cpu tot.	cpu prob.	cpu cplex
	2	214.2	89	111	105	6	60	81	74	7
	3	143.1	72	96	93	3	32	141	139	2
15	4	113.5	40	204	202	2	13	218	216	2
	5	97.2	18	177	177	0	15	238	237	1
	6	81.2	14	258	257	1	15	200	198	2
	2	287.6	139	220	207	13	108	197	192	5
	3	210.1	86	140	136	4	61	127	119	8
20	4	169.4	91	157	152	5	52	107	103	4
	5	134.2	72	138	134	4	25	85	83	2
	6	116.3	56	417	414	3	20	119	118	1
	2	355.3	226	476	441	35	168	397	370	27
	3	252.2	190	405	381	24	91	238	228	10
25	4	207.4	123	276	266	10	94	243	233	10
	5	169.8	104	260	252	8	41	234	228	6
	6	152.6	76	305	298	7	47	677	674	3
	2	447.7	407	1054	935	119	255	684	632	52
	3	307.3	406	1118	1022	96	230	667	628	39
30	4	254.1	170	471	446	25	199	573	542	31
	5	220.0	149	474	457	17	120	434	419	15
	6	191.6	98	330	324	6	54	249	243	6

Tableau 4.2 Tests sur les problèmes de Rosing: $n = 15..30, p = 2....6$.

n	p	OBJ	base + heur + col				base + heur + mul			
			it.	cpu tot.	cpu prob.	cpu cplex	it.	cpu tot.	cpu prob.	cpu cplex
	2	214.2	24	69	40	29	30	48	45	3
	3	143.1	26	72	44	28	12	22	19	3
15	4	113.5	6	79	62	17	9	48	47	1
	5	97.2	8	429	414	15	7	169	168	1
	6	81.2	5	406	393	13	11	144	143	1
	2	287.6	26	107	56	51	51	104	93	11
	3	210.1	39	117	71	46	30	66	62	4
20	4	169.4	18	76	47	29	20	50	48	2
	5	134.2	16	66	43	23	11	43	41	2
	6	116.3	11	245	223	22	12	135	134	1
	2	355.3	72	248	188	60	80	200	180	20
	3	252.2	78	272	220	52	35	104	99	5
25	4	207.4	33	149	112	37	36	119	114	5
	5	169.8	31	148	116	32	20	133	130	3
	6	152.6	22	342	316	26	20	239	236	3
	2	447.7	90	371	291	80	130	361	315	46
	3	307.3	116	461	383	78	93	309	283	26
30	4	254.1	58	258	208	50	34	149	142	7
	5	220.0	30	214	171	43	43	166	154	12
	6	191.6	36	218	172	46	33	143	140	3

Tableau 4.3 Tests sur les problèmes de Rosing: $n = 15..30, p = 2....6$

n	p	OBJ	base + heur + col + mul				base + heur + col + mul + tol			
			it.	cpu tot.	cpu prob.	cpu cplex	it.	cpu tot.	cpu prob.	cpu cplex
	2	214.2	9	52	24	28	9	47	20	27
	3	143.1	11	51	25	26	11	48	21	27
15	4	113.5	4	75	57	18	4	75	60	15
	5	97.2	5	291	276	15	5	285	270	15
	6	81.2	4	407	394	13	4	362	350	12
	2	287.6	20	109	62	47	20	100	46	54
	3	210.1	21	94	49	45	21	85	41	44
20	4	169.4	11	68	39	29	11	65	37	28
	5	134.2	8	54	32	22	8	50	26	24
	6	116.3	4	78	58	20	4	75	55	20
	2	355.3	26	139	88	51	26	133	71	52
	3	252.2	34	159	112	47	34	147	103	44
25	4	207.4	17	116	79	37	17	104	69	35
	5	169.8	13	94	63	31	13	83	52	31
	6	152.6	8	184	157	27	8	181	156	25
	2	447.7	39	222	143	79	39	210	126	84
	3	307.3	40	204	133	71	40	187	116	71
30	4	254.1	25	159	112	47	25	145	95	50
	5	220.0	20	164	117	47	20	150	106	44
	6	191.6	18	180	132	48	18	169	128	41

Tableau 4.4 Tests sur les problèmes de Eilon *et al.* : $n = 50, p = 3....8$.

n	p	OBJ	base				base + heur.			
			it.	cpu tot.	cpu prob.	cpu cplex	it.	cpu tot.	cpu prob.	cpu cplex
	3	105.2	1495	8698	7029	1669	2728	19457	12867	6590
50	4	84.1	1098	6315	5529	786	1875	11793	9140	2653
	5	72.2	533	3758	3612	146	968	6001	5393	608
	6	60.9	576	3884	3692	192	903	5418	4875	543
	7	54.5	367	2477	2391	86	270	2075	2011	64
	8	49.9	283	2221	2184	37	222	2028	1985	43

n	m	OBJ	base + heur + col				base + heur + mul			
			it.	cpu tot.	cpu prob.	cpu cplex	it.	cpu tot.	cpu prob.	cpu cplex
	3	105.2	720	4816	3837	979	1002	6598	3688	2910
50	4	84.1	775	5081	4116	965	856	5337	3414	1923
	5	72.2	215	2040	1850	190	273	1645	1476	169
	6	60.9	262	2656	2446	210	157	1301	1247	54
	7	54.5	114	1249	1139	110	117	820	786	34
	8	49.9	98	1021	102	210	121	875	832	43

n	m	OBJ	base + heur + col + mul			
			it.	cpu tot.	cpu prob.	cpu cplex
	3	105.2	129	1001	813	188
50	4	84.1	60	649	570	79
	5	72.2	67	680	610	70
	6	60.9	27	381	335	46
	7	54.5	11	212	174	38
	8	49.9	14	254	220	34

n	m	OBJ	base + heur + col + mul + tol			
			it.	cpu tot.	cpu prob.	cpu cplex
	3	105.2	132	972	785	187
50	4	84.1	60	579	497	82
	5	72.2	71	613	538	75
	6	60.9	26	327	286	41
	7	54.5	15	257	221	36
	8	49.9	14	226	189	37

4.7.1 Problèmes de la littérature

- La version de base de l'algorithme suffit à résoudre tous les problèmes tests. Les solutions pour $p \leq 5$ coïncident avec celles obtenues par Eilon *et al.* [34]

- Quand p augmente, les problèmes deviennent de plus en plus faciles à résoudre, à la fois en terme de nombre d'itérations et de temps de calcul.

- Le temps de calcul pour le problème auxiliaire est largement supérieur au temps pris par CPLEX pour résoudre le programme linéaire (environ 4 fois plus pour $p = 3$ et 50 fois plus pour $p = 8$). La différence devient moindre lorsque les règles d'accélération sont utilisées.

- L'utilisation de l'heuristique seule (lors de l'initialisation de la matrice M^0), détériore le temps de calcul. Il apparaît que les colonnes unité, lorsque p est petit, sont loin des structures des colonnes optimales. La différence est moindre lorsque p augmente. Ceci implique que les variables duales optimales de P^0 sont loin de l'optimum bien que leur somme soit optimale (lorsque l' heuristique résout optimalement le problème).

- L'ajout de colonnes par la règle issue de perturbations des colonnes de la solution trouvée par l'heuristique apparaît être une bonne accélération. Le temps de calcul et le nombre d'itérations sont tous deux divisés par un facteur proche de deux. Les exemples les plus marquant sont les problèmes avec $n = 50$ et $p = 3, 7, 8$.

- L'influence de la tolérance (..+tol) joue un rôle sensible dans certains problèmes ($n = 50$, $p = 3, 4$).

- En combinant toutes les règles d'accélération, le temps pour résoudre les problèmes de Eilon *et al.* citekn:eil71 , est divisé par presque 9 et le nombre d'itérations

est divisé par jusqu'à 20. Pour les problèmes de Rosing, cette règle est moins efficace (à cause de la taille moyenne des problèmes considérés).

- Dans tous les cas (excepté pour $p = 2$), notre algorithme est plus rapide que ceux de la littérature (Rosing [66], Ostresh [61], Chen *et al.* [16], Kuenne et Soland [49]).

4.7.2 Nouveaux problèmes

On a testé les problèmes suivants sur SS20/514MP (sparc20). Des problèmes avec $n = 287, 150, 100, 80, 50$ et $p = 5, ...50$ sont résolus. Ces derniers sont générés aléatoirement sur un carré [0,10] avec des poids égal à 1. Le dernier tableau donne les résultats obtenus sur le problème de Bongartz *et al.* [6] ($n = 287$). Pour ce problème on est partis d'une solution heuristique (H dans le tableau) obtenue par une méthode de recherche à voisinages variables développée par Hansen et Mladenović (article à paraître). La solution optimale est donnée dans la $4^{\text{ième}}$ colonne (F^* (dans le tableau). Le nombre d'itérations de la relaxation continue (avant branchement) est dans l'avant dernière colonne (it. dans le tableau). Les cas de branchements sont présentés dans la dernière colonne (gap (Br.) dans le tableau).

Tableau 4.5 Problèmes de Eilon *et al.* : $n = 50, p = 3....25$.

n	obj	p	Relax.				Branchement		
			it.	cpu tot.	cpu prob.	cpu cplex	it.	cpu	gap
	105.21	3	181	350.58	287.72	62.86	*		
	84.15	4	70	181.13	160.37	20.76	*		
	72.23	5	61	183.56	168.04	15.52	*		
	60.97	6	30	115.84	105.68	10.16	*		
	54.50	7	25	120.19	111.15	9.04	*		
	49.93	8	17	92.88	84.8	8.08	*		
	45.68	9	15	82.74	70.59	12.15	*		
	41.68	10	15	80.24	69.57	10.67	*		
	38.02	11	16	89.74	76.9	12.84	*		
	35.05	12	21	114.82	106.82	8.	*		
50	32.30	13	14	65.64	59.96	5.68	*		
	29.65	14	14	77.57	72.12	5.45	*		
	27.62	15	13	79.31	74.44	4.87	*		
	25.74	16	14	80.17	75.33	4.84	*		
	23.98	17	12	102.05	97.46	4.59	*		
	22.28	18	9	40.86	36.93	3.93	*		
	20.63	19	8	108.43	104.82	3.61	*		
	19.35	20	7	88.76	82.93	5.83	*		
	18.08	21	4	50.99	47.85	3.14	*		
	16.82	22	3	41.72	37.92	3.8	*		
	15.61	23	3	49.64	46.28	3.36	*		
	14.44	24	3	31.69	28.37	3.32	*		
	13.30	25	2	31.53	28.81	2.72	*		

Tableau 4.6 Résultats pour $n = 80, p = 5....45$.

n	p	Relax.				Branchement		
		it.	cpu tot.	cpu prob.	cpu cplex	it.	cpu	gap
	5	198	1156.92	1037.97	118.95	*		
	10	33	434.75	416.94	17.81	*		
	15	25	529.15	516.4	12.75	*		
	20	25	422.54	411.07	11.47	*		
80	25	15	392.11	383.00	9.11	*		
	30	8	220.6	214.05	6.55	*		
	34	12	241.23	234.4	6.83	2	849.9	26.161476 - 26.167317
	35	7	142.24	136.04	6.2	*		
	39	4	154.19	148.48	5.71	2	1021.83	21.590449 - 21.607725
	40	5	236.16	230.07	6.09	*		
	45	3	99.13	93.58	5.55	*		

Tableau 4.7 Résultats pour $n = 100, p = 5....45$.

n	p	Relax.				Branchement		
		it.	cpu tot.	cpu prob.	cpu cplex	it.	cpu	gap
	5	293	2811.06	2342.9	468.16	*		
	10	203	2639.12	2525.26	103.86	*		
	15	32	677.63	657.57	20.06	*		
	20	26	867.25	849.52	17.73	*		
100	25	30	722.23	705.73	16.5	*		
	30	17	607.81	594.84	12.97	*		
	35	30	688.22	666.21	22.01	*		
	40	10	322.19	313.32	8.87	*		
	45	14	466.67	458.7	7.97	*		

Tableau 4.8 Résultats pour $n = 150, p = 5....50$.

n	p	Relax.				Branchement		
		it.	cpu tot.	cpu prob.	cpu cplex	it.	cpu	gap
	5	429	14065.92	9709.43	4356.49	*		
	10	251	8316.48	7876.4	440.04	*		
	15	332	10131.21	9781.2	350.01	*		
	20	207	8418.3	8257.88	160.48	*		
150	25	96	5187.1	5122.25	64.93	*		
	30	87	4502.45	4456.31	46.14	*		
	35	90	4782.15	4734.92	47.23	*		
	40	57	3864.67	3833.78	30.89	*		
	45	60	3097.56	3067.9	29.66	*		
	50	30	2059.09	2036.03	23.06	*		

Tableau 4.9 Résultats pour $n = 287, p = 3....100$.

n	p	H	F^*	cpu	it (Relax.)	gap (Br)
	100	441.241744	441.241744	826.01	31	*
	95	480.859192	480.859192	754.23	27	*
	90	529.212633	529.212633	866.95	41	*
	85	588.386532	588.367951	900.06	52	*
	80	657.793588	655.378827	705.27	45	*
	75	730.105160	730.035409	3518.88	52	0.331321 (2)
	70	814.223752	814.223752	706.54	54	*
	65	924.554700	924.554700	1099.46	82	*
	60	1055.138941	1055.138941	1680.79	96	*
	55	1203.985415	1203.984871	1734.54	145	*
	50	1402.583644	1402.583644	1454.90	122	*
	45	1630.311497	1630.311497	1261.75	102	*
	40	1900.857097	1900.836091	1773.53	160	*
	35	2238.183877	2238.183877	1693.05	139	*
287	30	2716.903811	2716.903811	3660.94	300	*
	25	3348.710076	3348.710076	4728.18	356	*
	20	4148.844314	4148.844314	2277.04	159	*
	19	4342.064807	4342.064807	19314.65	1005	*
	18	4547.365127	4547.365127	4384.21	292	*
	17	4755.189033	4755.189033	3285.40	202	*
	16	4981.960824	4981.960824	6712.14	336	*
	15	5224.702819	5224.702819	4262.43	211	*
	14	5469.647790	5469.647790	3636.07	185	*
	13	5725.185282	5725.185282	2593.04	135	*
	12	6033.047434	6033.047434	11143.60	406	*
	11	6351.590990	6351.590990	5545.89	256	*
	10	6705.035556	6705.035556	8557.51	276	*
	9	7088.128333	7088.128333	8016.4	250	*
	8	7564.294907	7564.294907	3976.08	183	*
	7	8160.320305	8160.320305	8522.16	173	*
	6	8787.556838	8787.556838	17916.06	305	*
	5	9715.627471	9715.627471	27028.17	317	*
	4	10661.476589	10661.476589	14730.47	210	*
	3	12095.442160	12095.442160	36522.06	221	*

CHAPITRE 5

Le problème de Weber multi-sources avec contraintes de localisation et de passage

5.1 Introduction

Le problème de Weber multi-sources sans contraintes a été résolu récemment par des techniques d'optimisation globale. Chen *et al.* [16] formulent le problème du multi-sources comme un programme D. C. (la fonction objectif est une différence de fonctions convexes et les termes de gauche des contraintes sont des différences de fonctions convexes) puis comme une minimisation d'une fonction concave sur un convexe. Ce dernier problème est résolu par une méthode d'approximation extérieure et d'énumération de sommets. Le nombre de variables est deux fois le nombre de sources à localiser (i.e. le nombre de coordonnées des sources) ajouté à la variable introduite pour la réduction du problème en un programme de minimisation d'une fonction concave sur un convexe. Il est bien connu que les méthodes d'approximations extérieures pour des problèmes non convexes sont très sensibles à la dimension de l'espace (puisque le nombre de sommets du polytope approximant augmente exponentiellement avec la dimension). Pour cette raison des problèmes avec 2 sources et un nombre de points de demande allant jusqu'à 1000 sont résolus; 3 sources avec un nombre de points de demande allant jusqu'à 50. Dans ce dernier cas l'algorithme d'Ostresh ([60]) est plus rapide. En combinant la technique de génération de colonnes d'un programme linéaire (Gilmore et Gomory [36]) avec

l'optimisation globale nous avons multiplié par 10 la taille des problèmes résolus (cf chapitre 4): un problème de partitionnement similaire à celui de Rosing [66] est utilisé mais en considérant tous les ensembles possibles de points de demande (2^n-1). Le problème auxiliaire qui choisit la colonne de coût réduit le plus négatif est formulé en un problème de Weber avec distance maximum. Le problème auxiliaire peut aussi être formulé en un programme D. C. (différence de fonctions convexes), réductible en un programme de minimisation d'une fonction concave sur un convexe qui se résout par une méthode d'approximation extérieure (chen *et al.* [16]). Dans de rares cas, quand la solution optimale du programme linéaire a une solution fractionnaire, une règle de branchement (Ryan et Forster [69]) est utilisée (2 points de demande sont choisis et les cas où ils sont affectés à une même source ou à une source différente partagent l'ensemble des solutions). Des problèmes allant jusqu'à 287 points de demande et un nombre quelconque de sources sont résolus exactement. La propriété originale de cette approche est que la difficulté de résolution décroît lorsque le nombre de source augmente, à la différence des méthodes existantes.

L'objet de ce chapitre est d'étendre les résultats au cas où les contraintes de localisation et de passages sont présentes (cf chapitre 3).

Le chapitre est structuré en quatre parties. Nous donnons en première partie la formulation mathématique du problème ainsi qu'une liste des symboles. Dans la deuxième partie l'algorithme général est présenté, la résolution du problème auxiliaire est explicité dans la troisième partie et la dernière partie est réservée aux résultats numériques.

5.2 Formulation du problème

Le problème de Weber multi-sources avec zones interdites (MWFZ) et des contraintes de passage (MWBT) peut être formulé de la manière suivante

(i) n points de demande a^j pour $j = 1...n$ sont localisés dans le plan. Des constantes w_j pour $j = 1...n$ sont associées à ces points et représentent la quantité de demande pour chacun d'entre eux.

(ii) m_1 zones sont interdites pour la localisation. Ce sont les intérieurs de polygones P_i $(i = 1, m_1)$ définis par une liste de leurs sommets $(b_1^1, ..., b_{e_i}^i)$ où e_i est le nombre de côtés de P_i.

(iv) m_2 zones sont interdites pour la localisation et le passage. Ce sont les intérieurs de polygones R_i $(i = 1, m_2)$ définis par une liste de leurs sommets $(c_1^i, ..., c_{e_i}^i)$ où e_i est le nombre de côtés de R_i.

(v) En l'absence de contraintes de passage, le coût de transport d'une source localisée en s à un point de demande a^j est proportionnelle à la distance euclidienne $d_j(s)$ entre ces deux points: en présence de contraintes de passage, le coût devient proportionnel à la distance réalisable $\underline{d}_j(s)$ (le plus court chemin de s à a^j sans traverser d'obstacles de passage).

(vi) p sources doivent être simultanément localisées en des points $s_1, ...s_p$ du plan, en dehors des régions interdites à la localisation et au passage afin de minimiser la somme des distances réalisables pondérées des points de demande à la source la plus proche.

(vii) $p(x)$: point courant du plan (on suppose $x = (x_1, x_2)$).

(viii) $d_j(p)$: distance euclidienne entre a^j et $p(x, y)$.

(ix) $d(a,b)$: distance euclidienne entre les point a et b.

(x) Q_h : carré du plan (h est un nombre générique).

(xi) $s_h^*(x_h^*, y_h^*)$: localisation optimale de la source (lors de la résolution du problème auxiliaire) à l'intérieur de Q_h .

(xii) s^* : localisation optimale de la source (lors de la résolution du problème auxiliaire).

Une formulation sous forme d'un programme linéaire généralisé avec un nombre exponentiel de colonnes du problème de Weber multi-sources avec contraintes s'énonce de la façon suivante (cf chapitre 4):

$$P^{MWC}\begin{cases} \min \sum_{t=1}^{2^n-1} \bar{c}_t y_t \\ \text{sujet à:} \\ \begin{cases} \sum_{t=1}^{2^n-1} b_{jt} y_t \geq 1 & j = 1, 2..., n \\ \sum_{t=1}^{2^n-1} y_t = p \\ y_t \in \{0,1\} \end{cases} \end{cases}$$

L'espace des solutions correspond à tous les groupes possible de p ensembles de points de demande distincts couvrant tous les points de demande. Soit E_t un ensemble de points de demande:

- y_t vaut 1 si cet ensemble est pris dans la solution et vaut 0 sinon.
- $\bar{c}_t$ est le coût associé à cet ensemble (en considérant les contraintes de passage et de localisation).
- b_{jt} vaut 1 si le point de demande a^j appartient à E_t et vaut 0 sinon.

P^{MWC} est un programme linéaire avec un nombre exponentiel de colonnes qui ne peut être résolu directement. Nous proposons la technique de génération de colonnes combinée avec une méthode d'énumération implicite pour le résoudre. Le schéma de l'algorithme proposé dans ce chapitre est identique à celui énoncé pour la résolution du cas non contraint (cf chapitre 4). Seul le problème auxiliaire est résolu différemment.

5.3 Algorithme général

Ayant largement détaillé l'algorithme au cours du chapitre 4, nous allons dans cette section présenter uniquement les principales étapes de la méthode.

On relaxe les variables y_t ($y_t \geq 0$) du programme P^{MWC}. On obtient ainsi un programme linéaire en variables continues P_r^{MWC} avec un nombre exponentiel de colonnes.

1. Résolution du programme P_r^{MWC} :

(i) Initialisation du programme linéaire initial (construction de L colonnes) à partir d'une solution réalisable. Obtention de la solution optimale par une méthode du simplexe généralisée.

(ii) Construction (à partir des variables duales optimales du programme linéaire courant) et résolution du problème auxiliaire calculant la (es) colonne (s) de coût(s) réduit(s) minimum. Si aucune colonne de coût réduit négatif n'est obtenue, aller en (2).

(iii) Chargement de la (les) colonne(s) obtenues en (ii) dans le programme linéaire courant et obtention des variables duales optimales par la résolution de ce programme. Aller en (ii).

2. Branchement :

(i) si le vecteur solution de la relaxation continue est entier, fin: le problème est résolu. Sinon aller en (ii).

(ii) (règle de Ryan et Foster [69]) : sélection de deux points de demande a^j et a^k et création de deux sous-problèmes dont la borne inférieure est la valeur optimale de la relaxation continue du programme linéaire courant.

- a^j et a^k sont liés à la même source.
- a^j et a^k sont liés à deux sources différentes.

(ii) Stockage des sous-problèmes dans une liste.

(iv) Sélection du sous-problème de borne inférieure minimum (valeur de la relaxation continue et résolution de la relaxation continue par génération de colonnes).

(v) mise à jour si nécessaire de la meilleure valeur connue et du vecteur solution correspondant (dans le cas où la solution optimale d'un sous problème est entière).

(vi) Sauvegarde des sous problèmes (dans la liste des sous problèmes) ayant une borne inférieure plus petite que la meilleure valeur connue et un vecteur solution avec au moins une composante fractionnaire.

(vii) Élimination de la liste des sous problèmes de borne inférieure plus grande que la meilleure valeur connue. Si la liste est vide, fin: le problème est résolu.

Soit E_t un ensemble de points de demande associés à la colonne $t = (b_{1t}, ..., b_{nt})$ dans le programme P_r^{MWC} et l'expression de son coût réduit:

$$\overline{c}_t^* = \min_{(x,y)\in\mathbb{R}^2\setminus(P\cup R)} \sum_{j=1}^{n} \min(w_j \bar{d}_j(x_t, y_t) - u_j, 0) - u_{n+1}$$

l'obtention de $\overline{c_t}$ (appelé problème auxiliaire) est un programme D.C (cf chapitre 4) qui peut être résolu par une méthode d'approximation extérieure (cf Chen *et al.*, chapitre 4).

$\overline{c_t}$ peut aussi être trouvé par la résolution directe du problème de Weber simple avec distance maximum sous des contraintes de passage et de localisation.

Le chapitre 3 de la thèse porte sur la résolution du problème de Weber simple dans le plan avec contraintes de passage et de localisation. Une méthode exacte d'énumération implicite "BSSS" auquel on a ajouté un nouveau calcul de borne a été proposé. C'est un algorithme permettant de résoudre exactement les problèmes de Weber simple avec des fonctions de coûts non décroissantes en la distance. Le problème de Weber simple avec distance maximum sous contraintes de passage et de localisation fait partie de ce type de problèmes.

Nous allons donc adapter l'algorithme d'énumération implicite "BSSS" au problème de Weber simple avec distance maximum sous contraintes de passage et de localisation. Le schéma de la méthode est identique à celui proposé lors du chapitre 3 excepté pour le calcul de la borne inférieure.

5.4 Résolution du problème auxiliaire et construction de la colonne de coût réduit minimum

Soit les principales étapes de l'algorithme d'énumération implicite de l'espace des solutions de l'algorithme proposé au chapitre 3 :

(1) Partitionner le plus petit carré contenant l'enveloppe convexe des points de demande et les sommets de polygones en 4 carrés.

(2) Calculer sur chacun des carrés une borne inférieure et une borne plus grande que la valeur de la fonction objectif.

(3) Eliminer les carrés dont la borne inférieure est supérieure à la meilleure valeur connue et les carrés non réalisables. Sélectionner le carré Q_h (réalisable) de borne inférieure minimum et le subdiviser en quatre nouveaux carrés.

(4) Si la précision sur la taille du carré sélectionné est atteinte ou si tous les carrés ont été éliminés, fin. Sinon aller en (2).

5.4.1 Borne inférieure

5.4.1.1 Avant branchement

L'essentiel de la méthode "BSSS" repose sur le calcul d'une borne inférieure et d'une borne supérieure sur la valeur optimale de la fonction objectif à l'intérieur d'un carré Q_h du plan.

Soit $\underline{d}_j(p)$ la distance du plus court chemin réalisable (contournant les obstacles) entre le point de demande a^j et un point courant réalisable $p(x,y)$ de Q_h:

Si a^j et $p(x,y)$ sont visibles l'un de l'autre:

$$\underline{d}_j(p) = d_j(p)$$

sinon:

$$\underline{d}_j(p) = \underline{d}_j(v^j) + d(v^j, p))$$

. Où $v^j(x',y')$ est le dernier sommet de polygone visité par le plus court chemin réalisable entre a^j et $p(x,y)$.

Soit $f_j(x,y) = \min(w_j \underline{d}_j(x,y) - u_j, 0)$ la fonction économique associée au point de demande a^j (cf. figure 5.1).

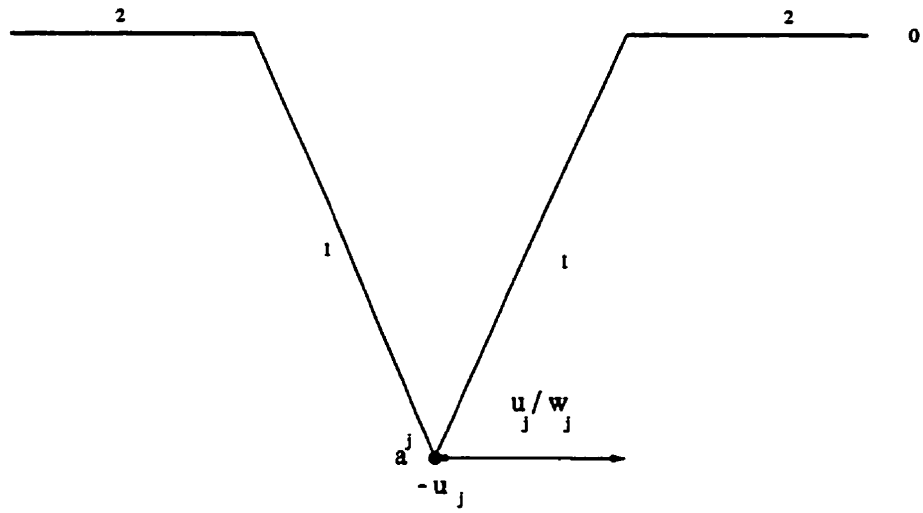


Figure 5.1 Fonction $f_j(x,y)$

La fonction $f_j(x,y)$ se décompose en deux régions : la région (1) et la région (2) (cf figure 5.1). Lorsque $\underline{d}_j(p)$ est plus petit que u_j/w_j (u_j: variable duale associée à la $j^{\text{ième}}$ contrainte), $f_j(x,y)$ se comporte comme une fonction convexe, vaut zéro pour des distances $\underline{d}_j(x,y)$ plus grande que u_j/w_j et est généralement concave en la distance. En considérant le cas particulier où Q_h est entièrement réalisable, nous allons partitionner les points de demande en 4 groupes $g1, g2, g3, g4$.

(1) Q_h entièrement réalisable :

g1: $g1 = \{a^j j = 1...n : \min_{(x,y)\in Q_h} \underline{d}_j(x,y) \geq \frac{u_j}{w_j}\}$.

g2: l'ensemble des points de demande a^j, $j = 1...n$ vérifiant :

- $\max_{(x,y)\in Q_h} \underline{d}_j(p) \leq \frac{u_j}{w_j}$.
- v^j est point dominant associé à a^j sur Q_h ou bien a^j est visible de tous les points de Q_h.

g3: l'ensemble des points de demande a^j $j = 1...n$ vérifiant :

- $\min_{(x,y)\in Q_h} \underline{d}_j(p) \leq \frac{u_j}{w_j}$.
- $\max_{(x,y)\in Q_h} \underline{d}_j(p) \geq \frac{u_j}{w_j}$.
- il existe un sommet v^j comme point dominant sur Q_h associé à a^j.

g4: tous les autres points de demande.

Soit $F_h(x,y)$ l'expression du coût réduit sur Q_h :

$$F_h(x_h^*, y_h^*) = \min_{(x,y)\in Q_h} \sum_{i=1}^{4} \sum_{a^j \in gi} F_h^i(x,y)$$

Où $F_h^i(x_h^*, y_h^*) = \min_{(x,y)\in Q_h} \sum_{a^j \in gi} \min(w_j \underline{d}_j(x,y) - u_j, 0)$ pour $i = 1..4$

On a d'après la figure 5.1 et la définition de $g1$:

$$F_h^1(x,y) = 0$$

L'ensemble $g2$ est composé de points de demande a^j dont l'expression de la distance au point courant sur Q_h s'écrit:

$$D_j = \underline{d}_j(v^j) + d(v^j, p).$$

où

$$D_j = d_j(p).$$

La fonction objectif $F_h^2(x,y)$ s'écrit comme la somme d'une constante et d'une distance euclidienne. $F_h^2(x,y)$ est une fonction convexe dont le minimum peut être obtenu par une méthode de descente sur Q_h (cf chapitre 3 section 2.3.2). Soit $p^2(x_2, y_2)$ ce point. On a :

$$F_h^2(x_2, y_2) = \min_{(x,y)\in Q_h} F_h^2(x,y) = \sum_{a^j \in g2} D_j(x_2, y_2)$$

L'ensemble $g3$ est composé de points de demande a^j pour lesquels le carré Q_h chevauche les régions (1) et (2) de leur fonction coût $f_j(x,y)$ (cf figure 5.2). Soit $C_j(x,y)$ le cône (construit de la même manière que dans le chapitre 2) sous estimant $f_j(x,y)$ sur Q_h (cf figure 5.2).

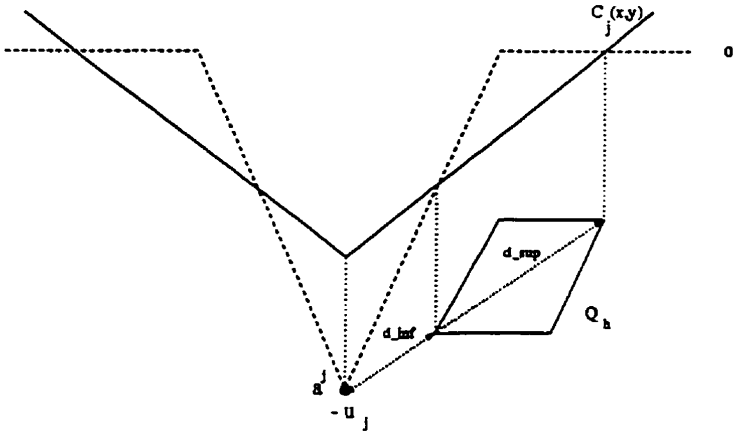


Figure 5.2 Fonction $C_j(x,y)$

$$F_h^3(x,y) > \sum_{a^j \in g3} C_j(x,y) \text{ pour } (x,y) \in Q_h$$

$C(x,y) = \sum_{a^j \in g3} C_j(x,y)$ est une fonction convexe dont le minimum est obtenu par une méthode de descente sur Q_h (cf chapitre 3 section 2.3.2). Soit $p_3(x_3,y_3)$ ce point. On a:

$$\min_{(x,y)\in Q_h} F_h^3(x,y) > C(x_3,y_3).$$

L'ensemble $g4$ est composé de points de demande a^j qui ne sont pas visibles de tous les points de Q_h ou qui ne possèdent pas de point dominant sur le carré Q_h. On en déduit :

$$\min_{(x,y)\in Q_h} F_h^4(x,y) > \sum_{a^j \in g4} \min(w_j \underline{d}_j(Q_h) - u_j, 0)$$

où $\underline{d}_j(Q_h)$ est la distance minimale réalisable du point de demande à Q_h.

Puisque :

$$\min_{(x,y)\in Q_h} F_h(x,y) > \sum_{i=1}^{4} \min_{(x,y)\in Q_h} F_h^i(x,y).$$

On a:

$$\min_{(x,y)\in Q_h} F_h(x,y) > \underline{F}_h$$

Où $\underline{F}_h = 0 + F_h^2(x_2,y_2) + C(x_3,y_3) + \sum_{a^j \in g4} \min(w_j \underline{d}_j(C) - u_j, 0)$.

(2) Q_h est partiellement réalisable :

$$\min_{(x,y)\in Q_h} F_h(x,y) > \underline{F}_h$$

Où $\underline{F}_h = \sum_{a^j \in J} \min(w_j \underline{d}_j(Q_h) - u_j, 0)$.

5.4.1.2 Après branchement

Un sous-problème est caractérisé par 2 ensembles de points de demande:

- E1 : ensemble des points de demande appartenant à la même source.
- E2 : ensemble de points de demande appartenant à des sources différentes.

Ces contraintes sur les points de demande sont traitées dans le problème auxiliaire (cf chapitre 4). L'ensemble des points de demande J subit des modifications (cf chapitre 4). Soit J' l'ensemble des points de demande pour E_1 et E_2 fixés. L'expression de la borne inférieure devient:

$$\min_{(x,y)\in Q_h} F_h(x,y) > \underline{F}_h$$

Où $\underline{F}_h = \min(\sum_{a^j \in E_1} (w_j \underline{d}_j(Q_h) - u_j), 0) + \sum_{a^j \in J'} \min(w_j \underline{d}_j(Q_h) - u_j, 0)$.

5.4.2 Colonne entrante

Soit s^* la solution optimale du problème auxiliaire et t la colonne optimale correspondante:

$$\begin{pmatrix} b_{1t^*} \\ b_{2t^*} \\ \cdots \\ b_{nt^*} \\ 1 \end{pmatrix}$$

$$\begin{cases} b_{jt^*} = 1 & \text{si } \underline{d}_j(s^*) < uj/w_j. \\ b_{jt^*} = 0 & \text{sinon.} \end{cases}$$

5.5 Résultats numériques

Les problèmes-tests sont résolus sur SPARC SUN 20 et le programme est écrit en langage C.

Soit le problème de Weber multi-sources avec contraintes de localisations et de passages d'Aneja et Parlar [2]. Nous allons donner en premier lieu les résultats obtenus pour un nombre de sources allant de 2 à 7. Les temps sont donnés en secondes. Il n'a fallu aucun branchement pour obtenir la solution optimale entière. Dans les figures qui suivent, les ronds et les carrés représentent respectivement les points de demande et les sources.

Tableau 5.1 Résultats pour le problème d'Aneja et Parlar : $n = 18, p = 2...7$

n	p	F^*	cpu (total)	itération	cpu (pb auxiliaire)	cpu (Cplex)
18	2	90.38	926.6	65	924.2	2.3
18	3	65.88	1037.8	55	1036	1.8
18	4	49.38	798.7	55	796.9	1.8
18	5	41.38	525.	29	523.7	1.3
18	6	34.63	559.	30	557.7	1.2
18	7	29.89	442.6	24	441.4	1.1

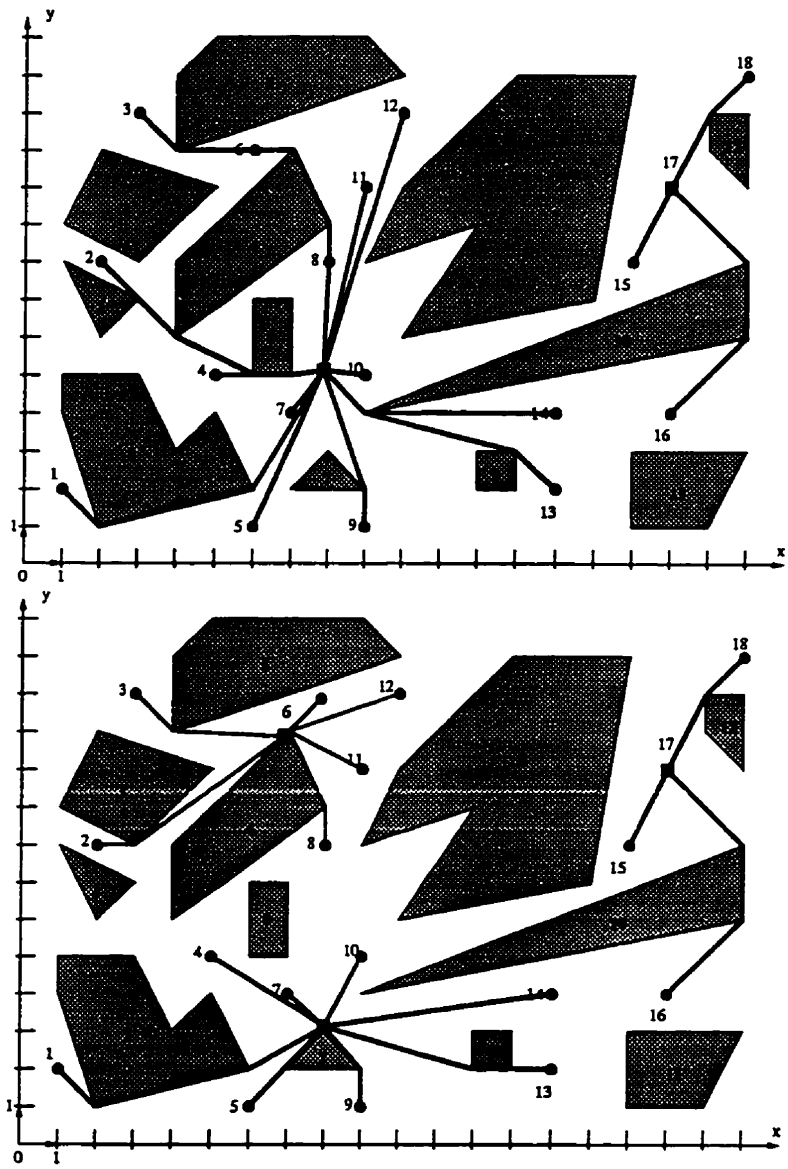


Figure 5.3 Résultats pour le problème d'Aneja et Parlar $n = 18$, $p = 2, 3$.

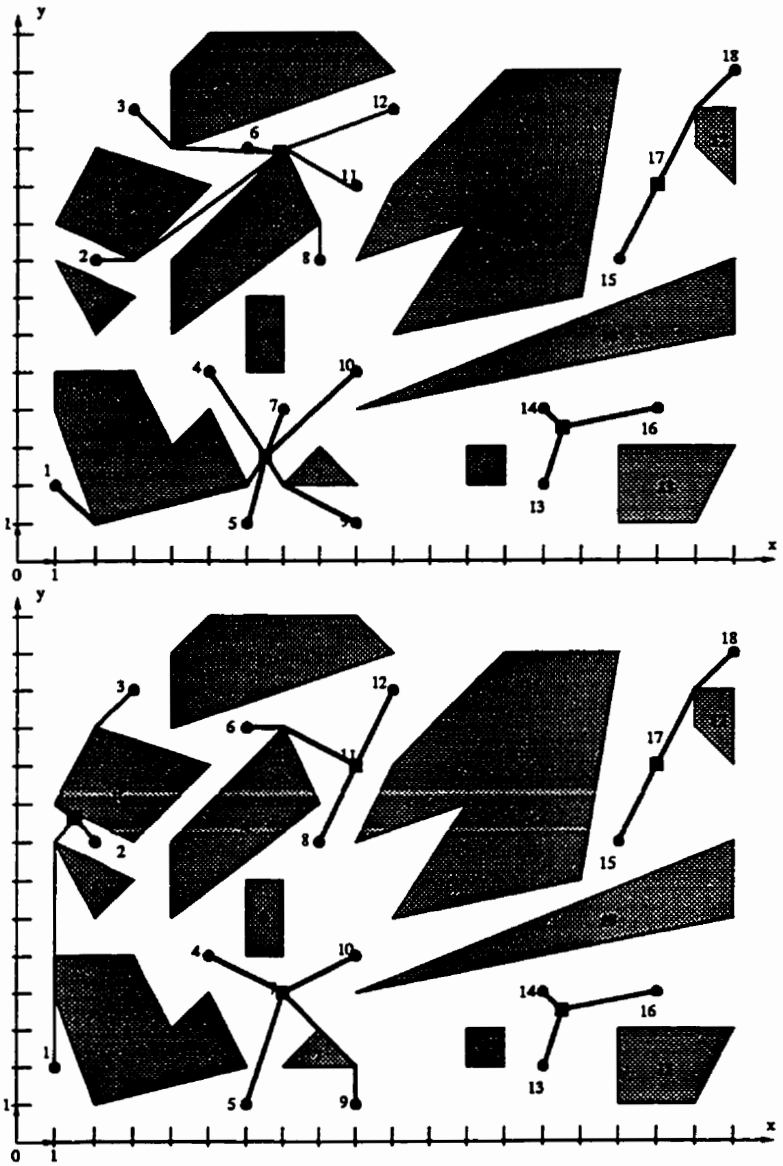


Figure 5.4 Résultats pour le problème d'Aneja et Parlar $n = 18$, $p = 4, 5$

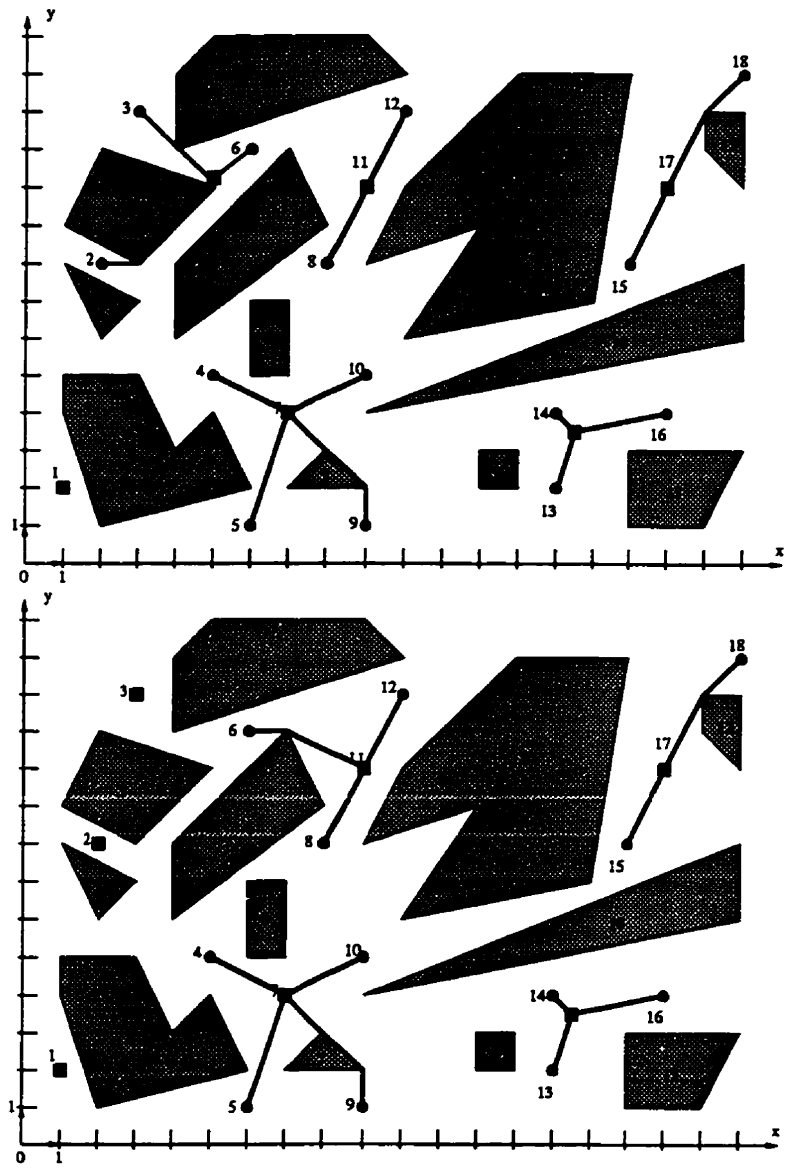


Figure 5.5 Résultats pour le problème d'Aneja et Parlar $n = 18$, $p = 6, 7$

Tableau 5.2 Coordonnées des sources du problème d'Aneja et Parlar

n	p	X_1	X_2	X_3	X_4	X_5	X_6	X_7
18	2	(17,10)	(7.82,5.03)					
18	3	(17,10)	(6.75, 10.75)	(7.94,3.12)				
18	4	(17,10)	(6.75, 10.75)	(6.57,2.61)	(14.48,3.47)			
18	5	(17,10)	(1.56, 8.72)	(9,10)	(7,4)	(14.48,3.47)		
18	6	(17,10)	(14.48,3.47)	(9,10)	(7,4)	(4.88,10.42)	(1,2)	
18	7	(17,10)	(9,10)	(7,4)	(14.48,3.47)	(2,8)	(3,12)	(1,2)

Pour éprouver la sensibilité de l'algorithme au nombre de points de demande, un problème de 30 points de demande et 2 (jusqu'à 7 sources) sont résolus. Aucun branchement n'a été nécessaire. Le problème est tout d'abord résolu dans le cas non contraint (avec les techniques d'accélération du chapitre 4) puis en considérant 12 contraintes de localisation (cf.tableau 5.4) puis 12 contraintes de passage (cf. tableau 5.5).

Tableau 5.3 Résultats (cas non contraint): $n = 30, p = 2, 7$

n	p	F^*	cpu (total)	itération
30	2	117.7	29.8	80
30	3	85.8	21.2	58
30	4	72.	19.1	36
30	5	60.6	15.8	30
30	6	52.6	17.1	36
30	7	46.3	12.6	24

Tableau 5.4 Résultats : $n = 30, p = 2, 7, m_1 = 12, m_2 = 0$

n	p	F^*	cpu (total)	itération
30	2	117.7	813.	149
30	3	85.8	945.5	249
30	4	72.	486.6	117
30	5	60.6	388.5	92
30	6	52.6	314.7	77
30	7	46.3	243.8	63

Tableau 5.5 Résultats : $n = 30, p = 2, 7, m_1 = 0, m_2 = 12$

n	p	F^*	cpu (total)	itération
30	2	134.1	8888.9	199
30	3	92.3	12437.8	240
30	4	74.2	6568.4	124
30	5	63.7	5580.2	91
30	6	55.6	5249.2	86
30	7	49.3	4296.7	58

Comme attendu, les contraintes de passage augmentent considérablement le temps de calcul en comparaison des contraintes de localisation.

C'est la première fois que le problème de Weber multi-sources avec contraintes de localisation et de passage est résolu optimalement. La taille des problèmes reste néanmoins modeste car les performances de la technique de génération de colonnes restent très liées aux performances de l'algorithme de résolution de son problème auxiliaire (problème de Weber simple avec distance maximum sous des contraintes de localisation et de passage) dont la complexité augmente rapidement avec le nombre de contraintes et le nombre de points de demande.

CONCLUSION ET DÉVELOPPEMENT

Les méthodes utilisées dans la thèse ont permis de résoudre efficacement, pour la première fois, et de façon exacte, certains problèmes majeurs de la théorie de la localisation:
le problème de Weber simple dans le plan avec contraintes de localisation et de passage;
le problème de Weber multi-sources avec contraintes de localisation et de passage.

Pour le problème de Weber multi-sources, des problèmes de taille jamais encore considérée ont été résolus et les premiers résultats significatifs de la littérature concernant le problème de Weber sur la sphère et le problème de Weber avec coût concave font l'objet des deux premiers chapitres.

Les trois premiers chapitres (le problème de Weber sur la sphère, le problème de Weber avec coût concave, le problème de Weber avec contraintes de localisation et de passage) ont été principalement résolu par des adaptations de l'algorithme "BSSS" développé par Hansen *et al.* (1984). C'est un algorithme d'énumération par séparations et évaluations pour lequel on a trouvé de nouvelles bornes. Dans le cas du problème de Weber sur la sphère, plusieurs bornes ont été étudiées pour résoudre des problèmes (avec ou sans contraintes de localisation) dont le lieu des points de demande se trouve sur un hémisphère, sur 2/3 de la sphère, et sur toute la sphère. Dans le cas du problème de Weber avec coût concave, la borne inférieure sur chaque fonction coût a été calculée à partir d'une fonction minorante (le meilleur plan minorant et un cône minorant). La programmation D.C. n'a pas connu les résultats escomptés

pour la résolution du problème de Weber avec coût concave mais reste néanmoins une des plus efficaces pour la résolution du problème de Weber multi-sources avec deux sources. Pour le problème de Weber simple avec contraintes de localisation et de passage la technique "BSSS" et le principe de la distance réalisable (plus court chemin entre deux points qui ne traversent pas de régions infranchissables) sont à la base de la méthode proposée.

Les deux derniers chapitres (le problème de Weber multi-sources, le problème de Weber multi-sources avec contraintes de localisation et de passage, i.e. des problèmes avec un nombre exponentiels de variables) ont été traités par la technique de génération de colonnes couplée avec une méthode d'énumération par séparations et évaluations. Cette méthode a résolu des problèmes de taille 10 fois supérieure ($n = 287$, $p > 2$) à ceux de la litérature. La particularité de cette méthode est qu'elle devient plus rapide à mesure que le nombre de sources augmente (pour un nombre fixe de points de demande). On remarque aussi qu'il y a eu très peu de problèmes qui ont nécessité des branchements. Les contraintes de passage augmentent considérablement le temps de résolution du problème de Weber multi-sources en comparaison aux contraintes de localisation.

Une future extension à cette thèse (spécifiquement pour le problème de Weber multi-sources) est un problème à deux niveaux: une usine (fixe) alimentant des dépôts (à localiser) fournissant en divers services des points de demande. Les coûts de transports sont considérés non linéaires et des contraintes de capacités sont ajoutées aux dépôts.

Par ailleurs on a noté lors de la résolution du problème de Weber multi-sources une forte dégénérescence dans le programme primal (cf chapitre 4). De récents travaux en collaboration avec Olivier du Merle (1995) nous ont permit de résoudre par stabilisation des variables duales dans l'algorithme de génération de colonnes

des problèmes de taille allant jusqu'à 1050 points de demande (article à paraître)...

BIBLIOGRAPHIE

[1] ALY, A.A., KAY, C.D., & LITWHILER, D.W. (1979). Location Dominance on Spherical surfaces. *Operations Research*, **27**, 972–981.

[2] ANEJA Y. P., PARLAR M. (1994). Algorithms for Weber Facility Location in the Presence of Forbidden Regions and (or) Barriers to Travel. *Transportation Science* **28**, 70-76.

[3] ATKINSON, M.D., SACK, J.-R., SANTORO, N., & STROTHOTTE, T. (1986). Min-Max Heaps and Generalized Priority Queues. *Communications of the ACM*, **29**, 996–1000.

[4] AVRIEL M. (1976). Nonlinear Programming. Analysis and Methods. *Prentice Hall, Englewood Cliffs, NJ*

[5] AYKIN, T., & BABU, A.J.G. (1987). A Constrained Large-Region Multifacility Location Problems. *Journal of the Operational Research Society*, **38**, 241–252.

[6] BONGARTZ I., CALAMAI P.H AND CONN R. (1994). A Projection Method for l_p Norm Location - Allocation Problems, *Mathematical programming* **66**, 283-312.

[7] BRIMBERG J., MLADENOVIC N. (1994). Application of Tabu Search in the Multisource Weber Problem. *Les cahiers du Gerad.* G-94-39.

[8] BRIMBERG J., MLADENOVIC N. (1995). A Variable Neighbourhood Algorithm for Solving Location-Allocation Problem. *Les cahiers du Gerad* G-95-40.

[9] BRIMBERG, J., & LOVE, R.F. (1993). Global Convergence of a Generalized Iterative Procedure for the Minisum Location Problem with ℓ_p Distances. *Operations Research*, **41**, 1153-1163.

[10] BRON C. AND KERBOSCH J. (1973). Finding all Cliques of an Undirected Graph. *Communications of the ACM* **16**, 575-579.

[11] BUTT E. S. (1994). Facility Location in the Presence of Forbidden Regions and Congested Regions. A Thesis in Industrial Engineering and Operations Research. *Department of Industrial and Management Systems Engineering and Graduate Program in Operations Research.*

[12] BUTT E., S. (1996). An Efficient Algorithm for Facility Location in the Presence of Forbidden Regions, *European Journal of Operational Research*, **90**, 56-70.

[13] CHARAMBOUS C., BANDLER J. W. (1976). Non-Linear Optimization as a Sequence of Least P-th Optimization with Finite Values of P. *International Journal of Systems Science* **7**, 377-391.

[14] CHEN R. (1983). Solution of Minisum and Minimax Location-Allocation Problems with Euclidean Distances. *Naval Research Logistics Quaterly* **30**, 449-459.

[15] CHEN, P.-C., HANSEN, P., JAUMARD, B., & TUY, H. (1992). Weber's Problem with Attraction and Repulsion. *Journal of Regional Science*, **32**, 467–486.

[16] CHEN P. C., HANSEN P., JAUMARD B., TUY H. (1992). Solution of the Multisource Weber and Conditional Weber Problems by D. C. Programming, *Les cahiers du Gerad.* G-92-35.

[17] COOPER, L. (1963). Location Allocation Problems. *Operations Research* **11**, 331-343.

[18] DEMJANOV, V.F. (1968). Algorithms for some Minimax Problems. *Journal of Computer and Systems Science*, **2**, 352–380.

[19] DHAR, U.R., & RAO, J.R. (1980). A Comparative Study of Three Norms for Facility Location Problem on a Spherical Surface. *Zeitschrift fur Operations Research*, **8**, 173–183.

[20] DHAR, U.R., & RAO, J.R. (1982). An Efficient Algorithm for Solving Area-Constrained Location Problems on a Sphere. *Operations Research*, **19**, 23–32.

[21] DHAR, U.R., & RAO, J.R. (1982). Domain Approximation Method for Solving Multifacility Location Problems on a Sphere. *Journal of the Operational Research Society*, **33**, 639–645.

[22] DIJKSTRA E. W. (1959). A Note on Two Problems in Connection with Graphs. *Numerische Mathematic* **1**, 269-271.

[23] DONNAY, J.D.H. (1945). *Spherical trigonometry.* New York: Interscience.

[24] DREZNER Z. (1984). The Planar Two-Center and Two-Median Problems. *Transportation Science.* **18.**

[25] DREZNER, Z. (1981). On Location Dominance on Spherical Surface. *Operations Research*, **29**, 1218–1219.

[26] DREZNER, Z. (1983). Constrained Location Problems in the Plane and on a Sphere. *IIE Transactions*, **15**, 300–304.

[27] DREZNER, Z. (1985). A Solution to the Weber Location Problem on the Sphere. *Journal of the Operational Research Society*, **36**, 333–334.

[28] DREZNER, Z. (1989). Stochastic Analysis of the Weber Problem on the Sphere. *Journal of the Operational Research Society*, **40**, 1137–1144.

[29] DREZNER, Z., & WESOLOWSKY, G.O. (1978). Facility Location on a Sphere. *Journal of the Operational Research Society*, **29**, 997–1004.

[30] DREZNER, Z., & WESOLOWSKY G.O. (1983). Minimax and maximin facility location problems on a sphere. *Naval Research Logistics Quaterly*, **30**, 305–312.

[31] DREZNER, Z. & WESOLOWSKY, G.O. (1990). The Weber Problem on the Plane with some Negative Weights. *Information Systems and Operational Research*, **29**, 87–99.

[32] DREZNER Z., MEHRTEZ A., WESOLOWSKY G. O. (1991). The Facility Location Problem with Limited Distances. *Transportation Science* **25**, 183-187.

[33] DU MERLE OLIVIER (1995). Points intérieurs et plans coupants: mise en oeuvre et développement d'une méthode pour l'optimisation convexe et la programmation linéaire structurée de grande taille. *Ph.D. THESIS.*

[34] EILON S., WATSON-GANDY C.D.T., CHRISTOFIDES N. (1971). Distribution Management, *Mathematical Modelling and Practical analysis* Hofner, New York.

[35] GILMORE P.C. AND GOMORY R. E. (1963). A Linear Programming Approach to the Cutting Stock Problem - Part2. *Operations Research* **11**, 863-888.

[36] GILMORE P. C. AND GOMORY R. E. (1961). A Linear Programming Approach to the Cutting Stock Problem. *Operations Research* **9**, 849-859.

[37] GOURDIN, E., HANSEN, P., & JAUMARD, B. (1995). *Global Optimization of Multivariate Lipschitz Functions: Survey and Computational Comparison of Algorithms.* In preparation.

[38] HANJOUL P., PEETERS D. (1985). A Comparaison of Two Dual-Based Procedures for Solving the P-median Problem. *European Journal of Operational Research* **20**, 387-396.

[39] HANSEN, P., & JAUMARD, B. (1994). Lipschitz Optimization. In R. Horst & P. Pardalos (Eds.), *Handbook of Global Optimization* (pp. 407-493). Dordrecht: Kluwer.

[40] HANSEN, P., PEETERS, D., RICHARD, D., & THISSE, J.F. (1985). The Minisum and Minimax Location Problems Revisited. *Operations Research*, **35**, 1251–1265.

[41] HANSEN, P., LABBE, M., PEETERS, D., & THISSE, J.-F. (1987). Facility Location Analysis, *Fundamentals of Pure and Applied Economics*, **22**, 1–70.

[42] HANSEN P., JAUMARD S., KRAU S. (1996). An Algorithm for Weber's Problem on the Sphere. *Location Science* **3**, 217-235.

[43] HANSEN P., PEETERS D., RICHARD D. and THISSE J. F. (1985). The Minisum and Minimax Location Problems Revisited, *Operations Research* **35**, 1251-1265.

[44] HANSEN P., PEETERS D., THISSE J. F. (1982). An Algorithm for a Constrained Weber Problem. *Management Science* **28** , 1285-1295.

[45] HANSEN P., MLADENOVIC N., TAILLARD. E.(1996). Heuristic Solution of the Multisource Weber Problem as a p-median Problem. *Les cahiers du Gérad* G-96-10.

[46] HORST, R., & TUY H. (1993). *Global Optimization: Deterministic Methods.* Berlin: Springer.

[47] HURTER A. P., SCHARFER M. K., WENDELL R. E. (1975). Solutions of Constrained Location Problems. *Management Science* **22**, 51-56.

[48] KATZ, I.N., & COOPER, L. (1980). Optimal Location on a Sphere. *Computers and Mathematics with Application*, **6**, 175–196.

[49] KUENNE R.E, SOLAND R.M. (1972). Exact and Approximate Solutions to the Multisource Weber Problem, *Mathematical Programming* **3** , 193-209.

[50] KUHN H. (1973). A Note on Fermat's Problem. *Mathematical Programming* **4**, 98-107.

[51] LITWHILER, D.W., & ALY, A.A. (1979). Large Region Location Problems. *Computers and Operations Research*, **6**, 1–12.

[52] LOVE, R.F., MORRIS, J.G., & WESOLOWSKY, G.O. (1988). *Facilities Location: Models and Methods.* New York: North-Holland.

[53] LOVE R.F, JUEL H. (1982). Properties and Solution Methods for Large Location-Allocation Problems. *Journal of the Operational Research Society* **33**, 443-452.

[54] MARANAS, C. D., FLOUDAS, C. A. (1994). A Global Optimization Method for Weber's Problem with Attraction and Repulsion, *Large Scale Optimization: State of the Art, eds.* W. W. Hager. D. W. Hearn and P. M. Pardalos, Kluwer Academic Publishers, Dordrecht, The Netherlands. 259-293.

[55] MEGGIDO AND SUPPOWITZ K. J. (1984). On the Complexity of some Common Geometric Location Problems. *SIAM Journal on Computing* **13**, 182-196.

[56] MURTAGH B. A., NIWATTISYAWONG S. R. (1982). An Efficient Method for the Multi-Depot Location-Allocation Problem. *Journal of the Operational Research Society* **33** ,629-634.

[57] MURTAGH B. A. AND SAUNDERS M. A. (1982). A Projected Lagrangian Algorithm and Its Implementation for Sparse Nonlinear Constraints. *Mathematical Programming Study in Constrained Minimization* 84-117.

[58] MURTY KATTA G., (1983). Linear Programming. *New York : Wiley.*

[59] OSTRESH, L.M. (1978). On the Convergence of a Class of Iterative Methods for Solving the Weber Location Problem. *Operations Research*, **26**, 597–609.

[60] OSTRESH L. M. (1973). An efficient Algorithm for Solving the two Center Location Allocation Problem. *Journal of Regional Science* **15**, 209-216.

[61] OSTRESH L. M., (1973) Multi-Exact Solutions to the M-center Location-Allocation Problem. in: G. Rushton, M.F. Goodchild and L. M. Ostresh Jr. (eds.). *Computer Programs for Location-Allocation Problems.* Monograph Number 6, Department of Geography, University of Iowa, Iowa City, IA.

[62] PIYAVSKII, S.A. (1972). An Algorithm for Finding the Absolute Extremum of a Function. *USSR Computational Mathematics and Mathematical Physics*, **12**, 57–67. (*Zh. vȳchisl Mat. mat. Fiz.*, **12**, 888–896.)

[63] PLASTRIA F., (1992) The Generalized Big Square Small Square Method. *European Journal of Operational Research* , **62** 163-174.

[64] PLASTRIA F. (1993). Continuous Location Theory, Anno 1992: A progress report. *Studies in Locational Analysis*, **5**, 85–127.

[65] ROSEN J. B., XUE G. L. AND P. M. PARDALOS. (1992) A Polynomial Time Dual Algorithm for the Euclidean Multifacility Location Problem, in E. Balas, G. Cornuejols and R. Kannan (eds.), *Integer Programming and Combinatorial Optimization* Carnegie-Mellon University Press, Pittsburgh, 227-236.

[66] ROSING K. E. (1992). An Optimal Method for Solving the (Generalized) Multi Weber Problem. *European Journal of Operational Research* **58**, 414-426. North Holland.

[67] ROSING K. E. (1990). Towards the Solution of the (Generalised) Multi-Weber Problem. *Economisch-geografisch Instituut.*

[68] RUSPINI E. H. (1970). Numerical Methods for Fuzzy Clustering. *Information Sciences 2* 319-350.

[69] RYAN D. M., FOSTER B. A. (1981). An Integer Programming Approach to Scheduling. *Computer Scheduling of Public Transport.*

[70] SCHAEFER M. K., HURTER A. P. (1974). An Algorithm for the Solution of a Location Problem with Metric Constraints. *Naval Research Logistic Quaterly.* **21**, 625-636.

[71] TSAI, W.H., CHERN, M.-S., & LIN, T.-M. (1990). An Algorithm for Determining Whether m Given Demand Points are on a Hemisphere or not. *Transportation Science*, **25**, 91–97.

[72] TUY H. Al-KHAYYAL F. AND ZHOU F. (1995). a D.C Optimization Method for Single Facility Location Problems. *Journal of Global Optimization* **7** , 209-227.

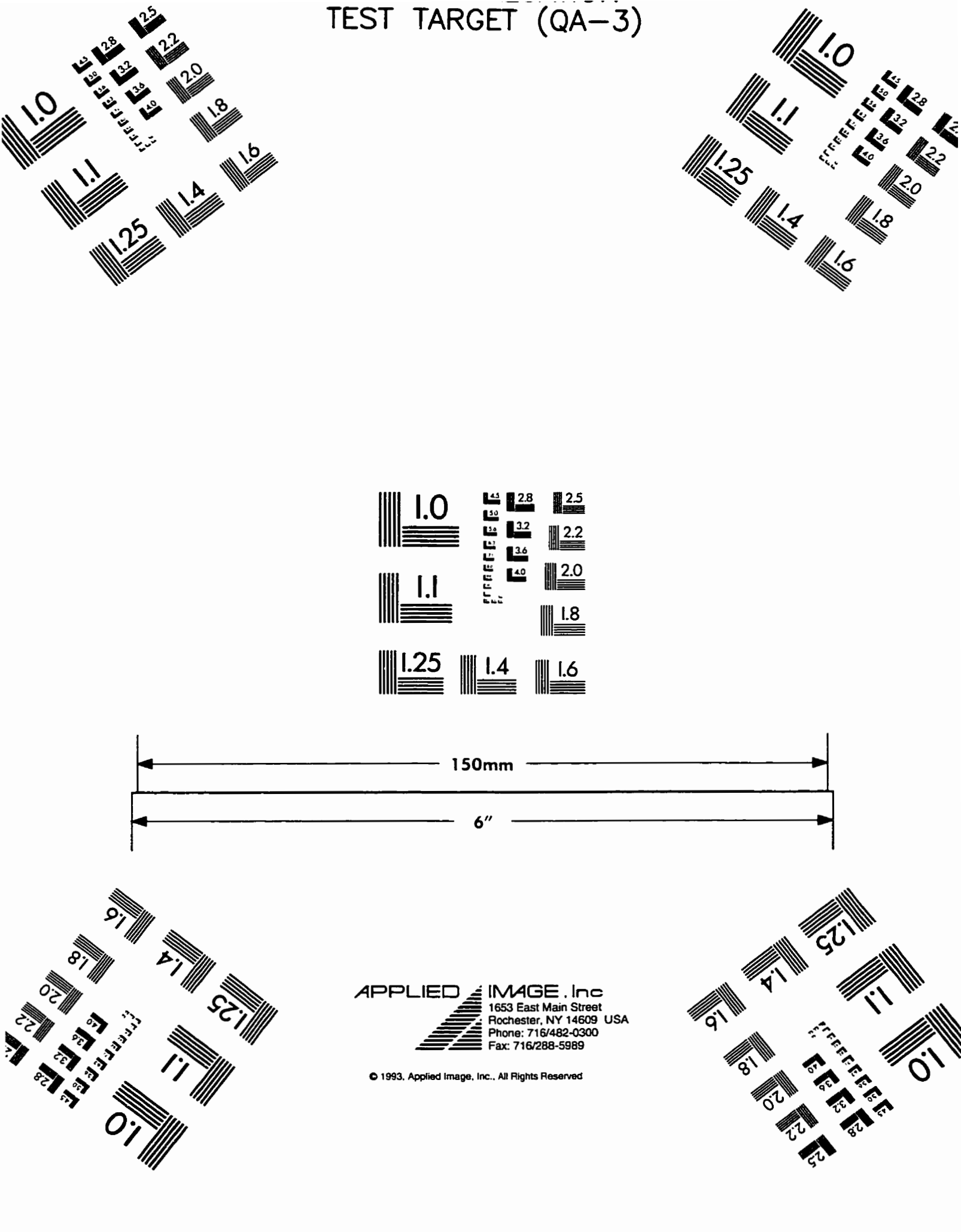
[73] TUY H. (1987). Global Minimization of a Difference of Convex Functions, *Mathematical Programming Study* **30**, 150-182.

[74] TUY H. (1983). On Outer Approximation Methods for Solving Concave Minimization Problems. *Acta Mathematica Vietnamica* **8**, 3-34.

[75] WEBER A. (1929). Ueber den Standort der Industrien. *Theorie of the Location of Industries.*

[76] WEISZFELD, E. (1937). Sur le point pour lequel la somme des distances de n points donnés est minimum. *Tôhoku Mathematical Journal*, **43**, 335–386.

[77] WESOLOWSKY, G.O. (1983). Location Problems on a Sphere. *Regional Science and Urban Economics*, **12**, 495–508.


TEST TARGET (QA-3)
1.0
1.1
1.25
1.4
1.6
1.8
2.0
2.2
2.5
2.8
3.2
3.6
4.0
150mm
6"
APPLIED IMAGE, Inc
1653 East Main Street
Rochester, NY 14609 USA
Phone: 716/482-0300
Fax: 716/288-5989
© 1993, Applied Image, Inc., All Rights Reserved