Treball final de grau

GRAU DE MATEMÀTIQUES

Facultat de Matemàtiques i Informàtica
Universitat de Barcelona

Problemes de localització d'instal·lacions: Teoria i algorismes

Autor: Júlia Folguera Profitós

Director: Dr. Miquel Bosch Gual
Realitzat a: Departament de matemàtiques i informàtica
(Matemàtica aplicada i Anàlisi)

Barcelona, June 19, 2019

2010 Mathematics Subject Classification. 90C10, 90C25, 90B80

Abstract

This project adresses various location problems, which is an area of optimization theory, mainly the Fermat-Weber problem and the uncapacited location problem. Regarding the first one, we study an iterative algorithm which produces the exact solution. For the second one, we explain some approximation algorithms to solve it. At the same time, there is an introduction to the linear optimization theory required. Finally, in both cases we introduce variations of the problems and we explain some ways to solve them given the theory previously explained.

Resum

El treball tracta diversos problemes de localització — que és una àrea de l'optimització — principalment el problema de Fermat-Weber i el problema de localització no capacitat. Pel primer, s'estudia un algorisme iteratiu que dona la solució exacta. Pel segon, s'expliquen diversos mètodes aproximatius per a resoldre'l, alhora que s'introdueix la teoria de programació lineal necessària. Finalment, en ambdós casos s'introdueixen variacions dels problemes i s'expliquen algunes formes de resoldre'ls a partir de la teoria anterior.

2010 Mathematics Subject Classification. 11G05, 11G10, 14G10

Agraïments

Voldria agrair al professor Miquel Bosch el seu recolzament i el seu ajut durant tot el treball. També voldria donar gràcies als companys que m'han acompanyat durant aquests últims quatre anys, especialment al Joel i a l'Anaïs que m'han donat el suport moral i emocional en els moments més estressants.

Índex de continguts

1 Introducció

L'optimització o programació matemàtica és una branca de les matemàtiques consistent en, donada una funció objectiu $f(x)$ i un conjunt de possibilitats que pot ser infinit $\mathcal{A}$, trobar l'element en $\mathcal{A}$ que maximitza o minimitza la funció. Aquesta és una branca ampla de les matemàtiques, per això en aquest treball només s'abordaran els problemes de localització, consistents en trobar la millor localització possible d'un punt donats uns altres punts, uns costos i unes certes condicions. L'aplicació pràctica d'aquests problemes és evident, i s'ha utilitzat en àmbits molt diferents, des de la planificació de noves centrals elèctriques a França, fins a l'anàlisi de restes arqueològiques de ciutats de l'antic Egipte. Malgrat tota la varietat d'aplicacions que existeix, en aquest treball anomenarem el punt que volem localitzar *fàbrica* i els que ens donen *ciutats*, seguint la notació de la major part de literatura del tema.

Aquests problemes, com es veurà més endavant, van començar històricament amb Fermat i el problema de Fermat-Weber, que busca la millor localització d'una sola fàbrica, però aviat en van aparèixer moltes variacions i va ser necessària una classificació dels possibles problemes.

La primera diferenciació entre problemes és produeix en canviar l'espai on es possible localitzar un nou punt i les relacions entre els punts donats. D'aquesta manera podem dividir els models en analítics, continus, de xarxa o discrets. El models analítics suposen una demanda uniforme sobre l'espai i que podem localitzar les noves fàbriques en qualsevol punt de l'espai. En els continus també pots localitzar les noves fàbriques en qualsevol punt, però la demanda es produeix només en uns punts donats, les ciutats. Un exemple d'aquest tipus de problema, que es tracta en el treball, és el problema de Fermat-Weber. Per altra banda, en els models de xarxa es parteix d'un graf que representa la demanda i podem localitzar les noves fàbriques només en algun dels vèrtexs. Finalment, en els models discrets et donen uns punts sobre l'espai corresponent a les ciutats, i altres corresponents a possibles localitzacions de les fàbriques.

Una altra forma de classificar els problemes és segons l'objectiu. Així tenim el problema de la mediana, que busca minimitzar la distància mitjana entre cada ciutat i la seva fàbrica més propera; el problema del conjunt de cobertura, que busca cobrir tota la demanda amb el mínim de fàbriques possibles; el problema del centre, que minimitza el màxim de les distàncies entre ciutats i fàbriques; el de models justos, que busca minimitzar les desigualtats en l'accessibilitat a les fàbriques; el P-problema, en què es vol cobrir el màxim de demanda possible obrint només P fàbriques, o el problema de cost fix, que busca minimitzar uns costos fixos, com poden ser els d'obertura de noves fàbriques o els costos de servei.

L'objectiu principal d'aquest treball era introduir la teoria d'optimització matemàtica mitjançant problemes específics i se n'estudien, principalment, dos. El primer és el de Fermat-Weber, que és un problema de mediana continu i es pot resoldre amb un algorisme

iteratiu que s'explica.

Després s'estudia el problema de localització mètric no capacitat, que és un dels més treballats i el més genèric d'ells. De fet, habitualment es construeixen algorismes per aquest problema i després amb petites variacions s'adapten a altres problemes semblants. Es tracta d'un problema discret de cost fix i cobertura total, és a dir es busca cobrir tota la demanda minimitzant el cost total. No es coneix cap algorisme que, amb temps polinòmic, arribi a la solució òptima, ja que és un problema $NP - hard$. Malgrat això sí que existeixen algorismes aproximatius que arriben, en temps polinòmic, a una solució que tot i no ser òptima, té un cost inferior a k cops l'òptim, on k és un valor conegut en cada algorisme. En el treball es presenten diversos algorismes aproximatius que intenten minimitzar el factor d'aproximació k i finalment es dona el valor mínim de k suposant que $P \neq NP$.

En aquest treball s'ha donat especial importància al valor del factor d'aproximació k així com en la part teòrica dels problemes i algorismes relacionada amb l'optimització lineal, però no s'ha demostrat la complexitat dels algorismes, que en qualsevol cas és polinòmica.

2 Problema de Fermat-Weber

2.1 Plantejament i història del problema

Malgrat haver-hi moltes teories al voltant de l'origen i progressió del problema de Fermat, la més habitual atribueix a Pierre de Fermat l'enunciat del problema qui, tal com s'explica en (4), va proposar el següent repte:

"Que aquell que no aprovi els meus mètodes intenti trobar la solució d'aquest problema: donats tres punt del pla, trobar-ne un quart tal que la suma de les distàncies fins als tres punts donats sigui mínima"

S'acostuma a atribuir la primera solució trobada a Evangelista Torricelli al 1645 i es tracta d'una solució amb regle i compàs. Posteriorment, Simpson va proposar una modificació a la resolució geomètrica de Torricelli afegint el que es va anomenar rectes de Simpson.

Una altra aproximació que es va fer del problema és considerar el seu *dual*. Això és, un problema formulat com un màxim (no un mínim) amb la mateixa solució. Es creu que va ser tractat explícitament per primer cop a "The Ladies Diary or Woman's Almanack", al 1755 amb el següent enunciat:

"Als tres costats d'un camp equiangular hi ha tres arbres a distàncies de 10, 12 i 16 unitats l'un de l'altre. Troba l'àrea del camp si és la màxima que admeten les dades"

La seva corresponent formulació matemàtica és:

$$\max_{U,V}\left\{-\sum_{i=1}^{2} a_i u_i + b_i v_i : \sum_{i=1}^{2} u_i = , \ \sum_{i=1}^{2} v_i = 0, \ \sqrt{u_i^2 + v_i^2} \leq w_i \ i = 1, 2\right\}$$

La utilització del problema dual està implícita en les rectes de Simpson i es fa servir també en alguns algorismes del problema de Weber, encara que no es tractaran en aquest treball.

Tornant al problema original, la primera generalització que se'n va fer va ser considerar que no només ens donen tres punts, sinó m. Es tracta del problema:

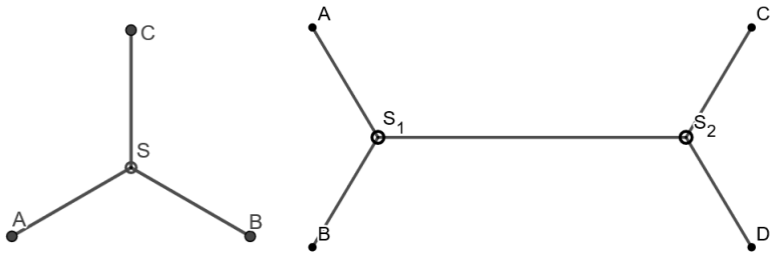
$$\min_{x\in\mathbb{R}^2}\left\{\sum_{i=1}^{m} ||x - A_i||_2 : A_i \in \mathcal{A}\right\}$$

$\mathcal{A} = \{A_1, ..., A_m\}$ el conjunt de punts donats, i $||\cdot||_2$ la distància euclidiana.

Fagnano troba la solució geomètrica en el cas $m = 4$, que, si els punts formen un polígon convex és simplement la intersecció de les diagonals. Malgrat això, aquest problema no té solució geomètrica amb regla i compàs per $m \geq 5$, tal com ho demostra Bajaj a l'article de *Discrete and Computational Geometry*, *The algebraic degree of geometric optimization problems*.

El problema d'Steiner és una reformularització popularitzada al 1941 que converteix el problema a teoria de grafs: "Donats m vèrtexs, trobar l'arbre amb pes mínim que els relacioni tots". Per fer-ho, s'utilitzen els punts d'Steiner, que en el nostre problema seria

els punts on localitzar les fàbriques. Per $m = 3$ el problema és equivalent al de Fermat, però per $m > 3$ és semblant a una variació posterior, el problema de multi centres que es tracta a la secció 2.3.2.

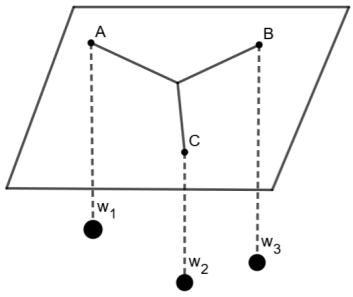


Per altra banda, Thomas Simpson va proposar al 1750 una generalització consistent en afegir pesos a cada punt donat:

$$\min_{x\in\mathbb{R}^2}\left\{\sum_{i=1}^{m} w_i||x-A_i||_2 : A_i \in \mathcal{A}\right\}$$

On $w_i \in \mathbb{R}^+$. Aquest problema s'anomena el problema de Weber ja que va ser popularitzat per l'alemany Alfred Weber al 1909 qui va donar-li un context econòmic, interpretant dos dels punts donats com a centres de matèria primera, el tercer com a lloc de venda i el punt que cal trobar com una nova fàbrica de producció. D'aquesta manera, els pesos podrien ser els costos de transportació.

Malgrat no existir solució exacta, n'hi ha una de física. Aquesta consisteix en, sobre una fusta que representa el pla, foradar els punts donats pel problema. Passem després per cada forat un fil que tingui al seu extrem el pes w_i que li pertoca. Finalment, si unim els fils amb un nus, trobem la solució. Aquest mètode s'anomena Varignon Frame i és utilitzat també per geògrafs i demògrafs.



Habitualment, però, es considera com a problema de Fermat i de Weber les corresponents generalitzacions en $\mathbb{R}^n$, és a dir, considerem com a problema de Weber:

$$\min_{x\in\mathbb{R}^n}\left\{W(x)=\sum_{i=1}^{m} w_i||x-A_i||_2 : A_i \in \mathcal{A}\right\} \tag{2.1}$$

Durant els anys posteriors es va redescobrir i se'n van trobar diferents solucions i aproximacions. La que més repercussió va tenir potser és la de l'hongarés Endre Weiszfeld (Andrew Vázsonyi) qui va donar un mètode iteratiu que, tot i passar desapercebut durant els primers anys, va ser redescobert més tard i millorat o modificat per resoldre altres

problemes en múltiples ocasions.

2.2 Algorisme de Weiszfeld

En aquesta secció es construirà l'algorisme i es demostrarà la seva convergència cap al punt mínim.

L'objectiu es trobar el punt M que compleixi l'equació 2.1. Com que W és una funció convexa, té un únic mínim, que és el que volem trobar. És condició necessària que $W'(M) = 0$, és a dir, $\sum_{i=1}^{m} w_i \frac{A_i - M}{||M - A_i||} = 0$. Aïllant M, trobem:

$$M = \frac{\sum_{i=1}^{m} \frac{w_i A_i}{||M - A_i||}}{\sum_{i=1}^{m} \frac{w_i}{||M - A_i||}} = f(M) \tag{2.2}$$

Veiem que $f(x)$ està definida a $\mathbb{R}^n \setminus \mathcal{A}$. Es pot demostrar que, en el cas en que el mínim no està en $\mathcal{A}$ i aquest no està contingut en un hiperplà, el conjunt de punts que en alguna iteració cau dins de $\mathcal{A}$, anomenat *conjunt de Kuhn* és numerable, i per tant la probabilitat de que això succeeixi és nul·la.

De tota manera, cal definir el cas $x \in \mathcal{A}$. Volem aconseguir la direcció amb la pendent més decreixent possible. Considerarem $y = A_j + t * K$, on K és un vector unitari, que ens marcarà la direcció, i volem definir K tal que $\frac{d(W(y))}{dt}$ sigui mínima, quan y es pròxim a A_j. $\frac{d(W(A_j + tK))}{dt} = w_j ||K|| + \sum_{i=1, i \neq j}^{m} \frac{w_i (A_j + tK - A_i)^T}{||A_j + tK - A_i||} K$.
Si escrivim:

$$B_j = \sum_{i=1, i \neq j}^{m} \frac{w_i (A_j - A_i)^T}{||A_j - A_i||} \tag{2.3}$$

Tenim que $\frac{dW}{dt} = w_j + B_j K$ quan t és pròxim a zero. Cal buscar ara el K que minimitza $w_j + B_j K$.

Proposició 2.1. *Sigui $a > 0$ i $B \in \mathbb{R}^n$ aleshores $\min\{a + B^T K : K \in \mathbb{R}^n, ||K|| = 1\}$ s'assoleix en $K = -\frac{B}{||B||}$ i $\max\{a + B^T K : K \in \mathbb{R}^n, ||K|| = 1\}$ en $K = -\frac{B}{||B||}$*

Així doncs el K que ho minimitza és $K = -\frac{B_j}{||B_j||}$. Per altra banda, substituint K i tornant a derivar,

$$\frac{d(W(y))}{dt} = w_j + \big(\sum \frac{w_i (A_j - A_i)^T}{||A_j + tK - A_i||} - B_j + B_j\big) K + \sum t \frac{w_i}{||A_j - A_i + tK||}$$

$$\leq w_j + \big(\sum \frac{w_i (A_j - A_i)^T (||A_j - A_i|| - ||A_j - A_i + tK||)}{||A_j + tK - A_i||} + B_j\big) K + \sum t \frac{w_i}{||A_j - A_i + tK||}$$

Imposant $t \leq \min\{\frac{1}{2}||A_j - A_i|| : A_i \in \mathcal{A}\}$, i aplicant desigualtat triangular obtenim:

$$\begin{aligned}\frac{d(W(y))}{dt} &\leq w_j + \big(\sum_{i=1}^{m} \frac{w_i(A_j - A_i)^T||K||t}{||A_j - A_i|| \cdot \big|||A_j - A_i|| - t\big|} + B_j\big)K + \sum_{i=1}^{m} \frac{w_i t}{\big|||A_j - A_i|| - t\big|} \\ &\leq w_j + \big(\sum_{i=1}^{m} \frac{w_i(A_j - A_i)^T t}{\frac{1}{2}||A_j - A_i||^2} + B_j\big)K + \sum_{i=1}^{m} \frac{w_i t}{\frac{1}{2}||A_j - A_i||} \\ &\leq w_j + \sum_{i=1}^{m} \frac{2w_i||A_j - A_i||t}{||A_j - A_i||^2} - \frac{||B_j||^2}{||B_j||} + \sum_{i=1}^{m} \frac{2w_i t}{||A_j - A_i||} \\ &= w_j - ||B_j|| + \sum_{i=1}^{m} \frac{4w_i}{||A_j - A_i||} t\end{aligned} \tag{2.4}$$

Per tant, es prou petit si $t < \frac{||B_j|| - w_j}{\sum_{i=1}^{m} \frac{4w_i}{||A_j - A_i||}}, min\{\frac{1}{2}||A_j - A_i||\}$ i definim:

$$t_j = \max\left\{0, \min\left\{\frac{||B_j|| - w_j}{\sum\limits_{i=1}^{m} \frac{4w_i}{||A_j - A_i||}}, \frac{1}{2}\min\{||A_j - A_i||\}\right\}\right\} \tag{2.5}$$

Finalment definim la funció recursiva de l'algorisme de Weiszfeld com:

$$R(z) = \begin{cases} \dfrac{\sum\limits_{i=1}^{m} \frac{w_i A_i}{||z - A_i||}}{\sum\limits_{i=1}^{m} \frac{w_i}{||z - A_i||}} & \text{si } z \in \mathcal{A} \\ & \\ & \text{si } z = A_j \in \mathcal{A} \end{cases} \tag{2.6}$$

2.2.1 Convergència de l'algorisme

Per demostrar la convergènca cap al mínim veurem primer que en cada iteració decreix W i després veurem que el límit de la successió és el mínim.

Proposició 2.2. *Si $R(z) \neq z$ i $R(z) \notin \mathcal{A}$, aleshores $W\big(R(z)\big) < W(z)$.*

Demostració. Considerem la funció $h_P(z) = \sum\limits_{i=1}^{m} \frac{w_i||z - A_i||^2}{||P - A_i||}$. La funció té un únic mínim, ja que és convexa, i és $R(P)$, obtingut derivant i igualant a zero. Com que només té un

mínim, en concret, $h_P(R(P)) < h_P(P)$ i per altra banda:

$$\begin{aligned}
h_P(R(P)) &= \sum_{i=1}^{m} \frac{w_i||R(P)-A_i||^2}{||P-A_i||} = \sum_{i=1}^{m} \frac{w_i(||R(P)-A_i|| - ||P-A_i|| + ||P-A_i||)^2}{||P-A_i||} \\
&= \sum_{i=1}^{m} \frac{w_i(||R(P)-A_i|| - ||P-A_i||)^2 + 2||P-A_i||(||R(P)-A_i|| - ||P-A_i||)}{||P-A_i||} \\
&= \sum_{i=1}^{m} \frac{w_i(||R(P)-A_i|| - ||P-A_i||)^2}{||P-A_i||} + \sum_{i=1}^{m} w_i||P-A_i|| + 2\sum_{i=1}^{m} w_i||R(P)-A_i|| \\
&- 2\sum_{i=1}^{m} w_i||P-A_i|| \geq W(P) + 2W(R(P)) - 2W(P) = 2W(R(P)) - W(P).
\end{aligned}$$

Així doncs, tenim:

$$W(R(P)) < W(P)$$

□

Per altra banda, en el cas que tinguem $z = A_i$, tenim que $t_j = 0$ si i només si $A_j = M$, i per tal com s'ha construït és clar que $W(A_j) > W\left(A_j + t_j \frac{B_j}{||B_j||}\right)$ si A_j no és el límit. Finalment veiem la convergència:

Lema 2.3. $A_j = M \Leftrightarrow w_j \geq ||B_j||$

Demostració. Si A_j és el mínim, com que la funció és convexa, és també el mínim absolut, és a dir per qualsevol direcció tk que prenguem $W(A_j+tk) > W(A_j)$ i en concret $w_j - ||B_j||$ tampoc serà negatiu, per tant, $w_j \geq ||B_j||$.
Per altra banda si tenim $\frac{dW}{dt} \leq w_j - ||B_j|| < 0$. En qualsevol direcció que prenguem, $W(A_j + tk) < W(A_j)$ i per tant, $A_j \neq M$ □

Lema 2.4. *Si $A_j \neq M$ aleshores $\exists \epsilon, \delta > 0$ tals que $||R(z) - A_j|| \geq (1+\epsilon)||z - A_j|| \quad \forall 0 < ||z - A_j|| < \delta$ i $||R(A_j) - A_j|| \geq \delta$*

Demostració.

$$\lim_{z \to A_j, z \neq A_j} \frac{||R(z) - A_j||}{||z - A_j||} = \lim_{z \to A_j, z \neq A_j} \frac{\left|\left| \frac{\sum_{i=1}^{m} \frac{w_i(A_i - A_j)}{||z-A_i||}}{\sum_{i=1}^{m} \frac{w_i}{||z-A_i||}} \right|\right|}{||z - A_j||} = \lim_{z \to A_j, z \neq A_j} \left|\left| \frac{\sum_{i=1}^{m} \frac{w_i(A_i - A_j)}{||z-A_i||}}{\sum_{i=1}^{m} \frac{w_i||z-A_j||}{||z-A_i||}} \right|\right|$$

$$= \frac{\lim_{z \to A_j, z \neq A_j} \left|\left| \sum_{i=1}^{m} \frac{w_i(A_i - A_j)}{||z-A_i||} \right|\right|}{\lim_{z \to A_j, z \neq A_j} \left|\left| \sum_{i=1, i \neq j}^{m} \frac{w_i||z-A_j||}{||z-A_i||} + w_j \right|\right|} = \frac{||B_j||}{w_j} > 1$$

On l'última desigualtat és degut al lema anterior.
Ara, aplicant la definició de límit, $\forall \delta > 0 \quad \exists \epsilon > 0$ tal que $\forall z$, si $||z - A_j|| < \delta$ aleshores $||R(z) - A_j|| \geq (1+\epsilon)||z - A_j||$. Si prenem

$$\delta \leq ||R(A_j) - A_j|| = \left|\left|A_j - t_j \frac{B_j}{||B_j||} - A_j\right|\right| = t_j$$

tenim l'enunciat. □

Proposició 2.5. *Sigui $z_0 \in \mathbb{R}^n$ i definim $z_k = R(z_{k-1}) \forall k \in \mathbb{N}$, aleshores $\lim_{k\to\infty} z_k = M$.*

Demostració. Com que $\lim_{||P||\to\infty} W(P) = \lim_{||P||\to\infty} w_i||P - A_i|| = +\infty$ i W és decreixent i no negativa, la seqüència $\{W(z_k)\}_{k\geq 0}$ és convergent.

Cal veure ara que convergeix al mínim. Considerem el conjunt $\mathcal{Z} = int(\bigcup_{k\geq 0} z_k)$. Existeix una subseqüencia de $\{z_k\}_k$ que convergeix a un punt de $\mathbb{R} \setminus (\mathcal{A} \setminus \{M\})$. Per veure-ho suposem el contrari, és a dir que $\mathcal{Z} \subseteq \mathcal{A} \setminus \{M\}$. Però $|\mathcal{Z}| \neq 0$, ja que altrament no convergiria i per el lema anterior, $|\mathcal{Z}| > 1$. Així doncs, $\exists A_1, A_2 \in \mathcal{Z}$ tal que $A_1 = \lim_{k\to\infty} R(z_{kl})$ i $A_2 = \lim_{k\to\infty} z_{kl}$.

Com que $\lim_{p\to A, p\neq A} ||R(p) - A|| = 0 \quad \forall A \in \mathcal{A}$, $\lim_{k\to\infty} R(z_{kl}) - A_2 = 0$ i concloem $A_1 = A_2$, fet que contradiu amb la nostra assumpció i hem demostrat que existeix una subsuccessió convergent a $P \in \mathbb{R}^n \setminus (\mathcal{A} \setminus \{M\})$ i, tal com està definida $R(z)$ per $z \notin \mathcal{A}$, R és contínua en un entorn de P. Finalment,

$$W(P) = \lim_{k\to\infty} W(P_k) = \lim_{l\to\infty} W(P_{kl}) = \lim_{k\to\infty} W\left(R(P_{kl})\right) = W\left(R(P)\right)$$

On la primera igualtat és certa per continuïtat de W, la segona per convergència i la última per continuïtat de $W \circ R$ a l'entorn de P. Així doncs, la successió convergeix al mínim, com volíem demostrar. □

2.2.2 Criteri de parada

Malgrat que un criteri de parada podria ser quan la distància entre iteracions és inferior a una tolerància, que seria quan consideraríem tenir un punt fix, aquest mètode no ens dona cap informació sobre la proximitat a la solució real, o la millora que s'està tenint. Per això en aquesta secció s'explica un criteri de parada proposat en (8) que dona una cota a la millora que cada iteració ens dona i demana aturar quan és inferior a la tolerància.

Lema 2.6. *Si els punts de $\mathcal{A}$ no són colineals, $W(x)$ és una funció estrictament convexa, és a dir $W(tx + (1-t)y) < tW(x) + (1-t)W(y) \quad \forall x, y \in \mathbb{R}^n \quad \forall t \in (0,1)$.*

Demostració. Utilitzant la desigualtat de Cauchy-Schwarz quan els punts no són colineals obtenim $||t(x-a) + (1-t)(y-a)|| < t||x-a|| + (1-t)||y-a||$ i per tant,

$$\begin{aligned} W(tx + (1-t)y) &= \sum_{i=1}^{m} ||tx + (1-t)y - A_i|| = \sum_{i=1}^{m} ||t(x - A_i) + (1-t)(y - A_i)|| \\ &< \sum_{i=1}^{m} t||x - A_i|| + (1-t)||y - A_i|| = tW(x) + (1-t)W(y) \end{aligned}$$

□

Lema 2.7. *Una funció $f : \mathbb{R}^n \to \mathbb{R}$ és estrictament convexa si i només si $f(x) > f(y) + \nabla f(y)^T(x-y)$.*

Demostració. Farem primer la demostració per $n = 1$, és a dir suposem $f : \mathbb{R} \to \mathbb{R}$ Demostrem primer la necessitat és a dir suposem que es compleix $f(tx + (1-t)y) < tf(x) + (1-t)f(y)$. Dividint a banda i banda entre t, obtenim

$$\frac{f(tx + (1-t)y) - f(y)}{t} < f(x) - f(y)$$

I fent el límit $t \to 0$, tenim $f(y) - \nabla f(y)^T(y-x) < f(x)$.

Demostrem ara el recíproc prenent x, y qualssevol i per hipòtesis,

$$\begin{aligned} tf(x) &> t[f(tx + (1-t)y) + f'(tx + (1-t)y)((1-t)x - (1-t)y)] \\ (1-t)f(y) &> (1-t)[f(tx + (1-t)y) + f'(tx + (1-t)y)t(y-x)] \end{aligned}$$

Sumant les dos obtenim $tf(x) + (1-t)f(y) > f(tx + (1-t)y)$.

Volem veure ara el cas general. Considerem x, y qualsevol i $g(t) = f(tx + (1-t)y)$. Es compleix $g'(t) = \nabla f(tx + (1-t)y)^T(y-x)$. Com que f és convexa, $g : \mathbb{R} \to \mathbb{R}$ també n'és i podem aplicar el cas $n = 1$ en g i tenim $g(0) > g(1) + g'(1)$ és a dir $f(y) > f(x) + \nabla f(x)^T(y-x)$. Per veure la suficiència, veiem que $\forall x, y \in \mathbb{R}^n \quad \forall t_1, t_2 \in (0,1)$:

$$\begin{aligned} g(t_1) &= f(t_1 x + (1-t_1)y) > f(t_2 x + (1-t_2)y) + \nabla f(t_2 x + (1-t_2)y)^T(y-x)(t_1 - t_2) \\ &= g(t_2) + \nabla g(t_2)(t_1 - t_2) \end{aligned}$$

I ja hem vist que això implica g és convexa, i també per tant f. □

Així doncs, es compleix $W(y) > W(x) + \nabla W(x)^T(y-x) \quad \forall x, y$.

Definició 2.8. *L'*envoltura convexa tancada*, d'un conjunt X és el conjunt de totes les combinacions convexes de punts de X, és a dir*

$$\text{conv}(X) = \left\{ \sum_{i=1}^{k} \lambda_i x_i : k < n,\ x_1, \cdots x_k \in \mathbb{R}^n,\ \lambda_1 \cdots \lambda_k \geq ,\ \sum_{i=1}^{k} \lambda_i = 1 \right\}$$

L'envoltura convexa tancada d'un conjunt X és el conjunt convex més petit que conté X.

Finalment podem donar una cota superior a la proximitat al mínim de cada iteració:

Proposició 2.9. *$\forall k \in \mathbb{N}$, $W(x_k) - W(M) \leq \max\{||x_k - A_i|| : i = 1 \cdots m\}||\nabla W(x_k)||$, on M és el mínim.*

Demostració. Considerem $\theta := conv(\mathcal{A})$ Fixem $k \in \mathbb{N}$ i considerem $y \in \theta$, $y \neq x_k$ qualsevol. Sigui K un vector unitari que marca la direcció entre x_k i y i $\alpha > 0$ tal que $y = x_k + K\alpha$. Com que W és convex podem aplicar la desigualtat del lema 2.7:

$$\begin{aligned} W(y) &\geq W(x_k) + \nabla W(x_k)^T(y - x_k) = W(x_k) + \alpha \nabla W(x_k)^T K \\ &\geq W(x_k) + \alpha \nabla W(x_k)^T \left(-\frac{\nabla W(x_k)}{||\nabla W(x_k)||} \right) = W(x_k) - \alpha ||\nabla W(x_k)|| \\ &\geq W(x_k) - \max\{||x_k - y|| : y \in \theta\}||\nabla W(x_k)|| \\ &\geq W(x_k) - \max_{i=1\cdots m} \{||x_k - A_i||\} \, ||\nabla W(x_k)|| \end{aligned}$$

On la última desigualtat es deu a la proposició 2.1.

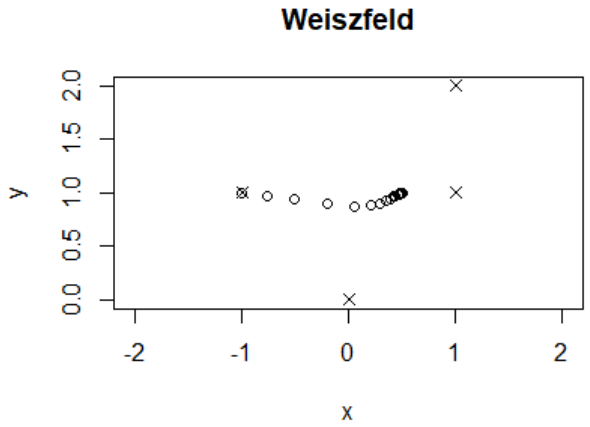
Finalment, com que la desigualtat anterior és certa $\forall y \in \theta$, podem prendre $y = M$. $\square$

2.2.3 Programació de l'algorisme

En l'annex es pot trobar l'arxiu *"Weiszfeld.c"*, que és la programació de l'algorisme de Weiszfeld de la següent manera:

1. Elegim el punt inicial i una tolerancia tol.
2. Apliquem la funció d'iteració $R(z)$ obtinguda en l'equació 2.6.
3. Si $\max\{||x_k - A_i|| : i = 1 \cdots m\}||\nabla W(x_k)|| < tol$ parem el programa, ja que hem trobat la solució. Altrament, i si no excedim el nombre màxim d'iteracions, tornem al pas 2.

S'ha programat l'algorisme en $\mathbb{R}^n$ i m centres donats. Com a exemple, resol el cas $\mathcal{A} = \{(-1,1),(0,0),(1,2),(1,1)\}$, que dona com a resultat $(0.5,1)$ des del punt inicial $(1,0)$ amb tolerància 10^{-8} en 54 iteracions, i des del punt inicial $(1,1)$ amb la mateixa tolerància en 59 iteracions.
La següent imatge mostra cada iteració de l'algorisme i com la successió va convergint:



2.3 Generalitzacions del problema

Al llarg del temps s'han proposat diverses generalitzacions del problema, bàsicament basades en la utilització d'altres distàncies, augmentar la quanitat de punts a escollir, canviar l'espai on s'optimitza del pla a l'esfera o discretitzar el problema.

2.3.1 Canvi de distància

Alguns cops usar altres distàncies enlloc de la euclidiana facilita molt la resolució del problema. Aquest és el cas en que es fa servir el quadrat de la distància euclidiana.

Si considerem $z = (z_1, ..., z_n)$, $A_i = (\alpha_i^1, ..., \alpha_i^n)$, es pot plantejar i resoldre el problema com segueix:

$$\min\Big\{\sum_{i=1}^{m} w_i||z - A_i||_2^2\Big\} = \min\Big\{\sum_{k=1}^{n}\sum_{i=1}^{m} w_i(x_k - \alpha_i^k)^2\Big\} = \sum_{k=1}^{n}\min\Big\{\sum_{i=1}^{m} w_i(x_k - \alpha_i^k)^2\Big\}$$

Per tant, podem separar cada component i trobar el mínim per cada una d'elles. Derivant i igualant a zero, la solució és:

$$\left(\frac{\sum_{i=1}^{m} w_i\alpha_i^1}{\sum_{i=1}^{m} w_i}, \ldots, \frac{\sum_{i=1}^{m} w_i\alpha_i^n}{\sum_{i=1}^{m} w_i}\right)$$

Que es tracta del centre de gravetat. Altres distàncies, com la rectilínia que té com a solució la mediana de cada coordenada, es soluciona de forma semblant.

Per altra banda, distàncies més complicades com les p-normes, $d_p(x,y) = \left(\sum_{i=1}^{n}(x_i - y_i)^p\right)^{\frac{1}{p}}$ utilitzen mètodes iteratius, sobretot basats en les normes per blocs o bé adaptant l'algorisme de Weiszfeld usant aproximacions hiperbòliques, $d_p^H(x,y) = \sqrt[p]{\sum_{i=1}^{n}(x_i - y_i)^p + \epsilon}$. Es pot demostrar que aquests mètodes convergeixen $\forall p \in [1,2]$.

2.3.2 Problema multi centres

Es tracta de la generalització del problema on no només s'ha d'optimitzar la localització d'un punt, sinó de varis.

$$\min_{z_j, j=1,...,t}\left\{\sum_{i=1}^{m}\sum_{j=1}^{t} w_{i,j}||z_j - A_i|| + \sum_{j=1}^{t-1}\sum_{k=j+1}^{t} v_{j,k}||z_j - z_k||\right\} \tag{2.7}$$

On es vol trobar la millor localització de t centres donats $A_1, ..., A_m \in \mathcal{A}$, amb un pes entre cada centre donat A_i i cada centre z_j, $w_{i,j}$; i un pes entre cada parell de centres z_i i z_j, $v_{i,j}$.

Aquest problema ha estat àmpliament estudiat i existeixen diversos algorismes o formulacions lineals segons la distància que s'escull.

3 Problema de localització no capacitat

El problema de localització no capacitat és un problema discret que modelitza la situació en què tenim un conjunt de llocs de consum, com podrien ser ciutats i volem obrir noves fàbriques de manera que es cobreixi tota la demanda amb el mínim cost possible. Per fer-ho, elegim a priori les possibles noves localitzacions i escollim d'entre elles el subconjunt que menys cost tingui; considerant el cost de transport dels productes entre les fàbriques i les ciutats que la tenen assignada i el cost d'obertura de cada fàbrica. Matemàticament es pot escriure de la següent manera:

Considerem

- El conjunt finit $\mathcal{D}$ de punts donats, corresponents a les ciutats.
- El conjunt finit $\mathcal{F}$, corresponent a les possibles localitzacions de les fàbriques d'entre les quals haurem d'elegir el subconjunt que minimitza la funció objectiu.
- $f_i \in \mathbb{R}_+^n$ que es té en compte quan triem el punt $i \in \mathcal{F}$, corresponent al cost d'obrir la possible fàbrica i.
- $c_{i,j} \in \mathbb{R}_+^n$ que és el cost que té assignar $i \in \mathcal{F}$ a $j \in \mathcal{D}$. Correspondria, per exemple, al cost de transport entre la possible fàbrica i i el centre de consum j.

El nostre objectiu és trobar $F \subseteq \mathcal{F}$ i $\sigma : \mathcal{D} \to F$ - que és l'assignació dels centres de venta i les fàbriques obertes, representant com es reparteix la demanda entre les fàbriques - de manera que es minimitzin els costos descrits anteriorment, és a dir:

$$\min_{F \subseteq \mathcal{F}} \left\{ \sum_{i \in F} f_i + \sum_{j \in \mathcal{D}} c_{\sigma(j),j} \right\} \tag{3.1}$$

A diferència del *problema de localització capacitat*, en el no capacitat no es limita la capacitat de producció de les fàbriques. Habitualment es considera $c_{i,j}$ proporcional a una distància, i en aquest cas s'anomena *problema de localització mètric no capacitat*. Això succeeix quan es compleix:

$$c_{i,j} < c_{i,j'} + c_{i',j'} + c_{i',j} \ \forall i \in \mathcal{F} \ \forall j \in \mathcal{D} \tag{3.2}$$

Malgrat que es pot demostrar que és NP-Hard, existeixen diferents algorismes que iterativament troben una solució aproximada. Aquests, s'anomenen algorismes d'aproximació:

Definició 3.1. *Considerem un problema d'optimització i $k \geq 0$. Sigui $OPT(I)$ la solució òptima del problema per a l'entrada I, i $A(I)$ la solució per a un algorisme de temps polinòmic tal que*

$$\frac{1}{k} OPT(I) \leq A(I) \leq kOPT(I) \ \forall I$$

*Aleshores diem que A és un k-*algorisme d'aproximació *i k n'és el* factor d'aproximació*.*

La primera desigualtat s'utilitza en els problemes de maximització i la segona en els de minimització. Cal tenir en compte, a més, que si el cost òptim de la funció és 0, l'algorisme aproximatiu ha de donar el valor exacte. Per altra banda, com major sigui

el cost d'optimització, major pot ser també la diferència entre aquest i el trobat per l'algorisme d'optimització.

En aquest capítol se n'expliquen varis, que estudien el problema des de perspectives diferents. Per començar, es reescriu el problema per a convertir-lo en un que ha esta més estudiat, el conjunt de cobertura ponderat. Per altra banda l'algorisme de Shmoys Tardos i Aardal passa a forma contínua el problema, el resol amb mètodes d'optimització lineal continus i discretitzar la solució. Per acabar s'estudien tres algorismes primals-duals, que resolen de forma paral·lela el problema dual i l'original.

3.1 Problema del conjunt de cobertura ponderat

En aquesta secció s'introdueix el problema del conjunt de cobertura ponderat, es reescriu el problema de locaclització no capacitat com a problema del conjunt de cobertura i es resol amb un algorisme aproximatiu, l'algorisme voraç. Considerem:

- Un parell $(U, \mathcal{S})$ on U és un conjunt finit i $\mathcal{S} = \{S_i, i = 1, \ldots, m\}$ conté subconjunts de U tals que $\bigcup_{S \in \mathcal{S}} S = U$ amb els quals volem recobrir U.
- Els pesos $c : \mathcal{S} \to \mathbb{R}^+$.

El *problema del conjunt de cobertura ponderat* busca $\{S_j : j \in J \subseteq I\}$ tal que minimitzi

$$\min \left\{ \sum_{j \in J} c(S_j) : J \subseteq I, \ \bigcup_{j \in J} S_j = U \right\} \tag{3.3}$$

Fixant $U = \mathcal{D}$, $S = \mathcal{P}(D)$ i $c(D) = \min \left\{ f_i + \sum_{j \in D} c_{i,j} : i \in \mathcal{F} \right\}$ per $D \subseteq \mathcal{D}$, podem interpretar el problema de localització no capacitat com a problema del conjunt de cobertura. Per tant, resolent el problema de cobertura es resoldrà també el problema de localització.

3.1.1 Algorisme voraç

L'algorisme voraç consisteix, simplement, en afegir en cada iteració el conjunt amb la màxima quantitat d'elements que encara no han estat coberts. Formalment:

1. Fixem $J = \emptyset$ i $W = \emptyset$.
2. Escollim $j \in I \setminus J$ tal que $S_j \setminus W \neq \emptyset$ i $\frac{c(S_j)}{|S_j \setminus W|}$ és mínim. Fixem $J = I \cup \{j\}$ i $W = W \cup S_j$.
3. Si $W = U$, aturem el programa. La solució és $\{S_i : i \in J\}$. Altrament repetim el pas 2.

Òbviament, el programa acaba en algun punt per les seves hipòtesis. Vegem que la solució que trobem té un cost no superior a $p = 1 + \frac{1}{2} + \cdots + \frac{1}{r}$ amb $r = \max\{|S| : S \in \mathcal{S}\}$ el cost òptim. D'aquesta manera haurem demostrat que és un p-algorisme d'aproximació.

Teorema 3.2. *L'algorisme voraç del problema del conjunt de cobertura ponderat és un p-algorisme d'aproximació.*

Demostració. Sigui $(U, \mathcal{S}, c)$ una instància del problema, i reordenem els conjunts de $\mathcal{S}$ de manera que $\{S_1, \ldots, S_k\}$ sigui la solució trobada amb l'algorisme i per a cada i, S_i és el conjunt que s'incorpora en la iteració número i. Considerem també $W_j = \bigcup_{i=0}^{j} S_i$ per a $j = 1, \ldots, k$.

Per a tot element d'U, u, definim $j(u) = \min\{i \in \{1, \ldots, k\} : u \in S_i\}$, és a dir, la primera iteració en la qual cobrim u i $y(u) = \frac{c(S_{j(u)})}{|S_{j(u)} \setminus W_{j(u)-1}|}$. Fixem $S \in \mathcal{S}$ i $k' = \max\{j(u) : u \in S\}$. Tenim:

$$\begin{aligned}
\sum_{u \in S} y(u) &= \sum_{i=1}^{k'} \sum_{u \in \{v \in S : j(v) = i\}} y(u) = \sum_{i=1}^{k'} \frac{c(S_i)}{|S_i \setminus W_{i-1}|} \sum_{u \in \{v \in S : j(v) = i\}} 1 \\
&= \sum_{i=1}^{k'} \frac{c(S_i)}{|S_i \setminus W_{i-1}|} |S \cap (W_i \setminus W_{i-1})| = \sum_{i=1}^{k'} \frac{c(S_i)}{|S_i \setminus W_{i-1}|} (|S \cap W_{i-1}| - |S \cap W_i|) \\
&\leq \sum_{i=1}^{k'} \frac{c(S)}{|S \setminus W_{i-1}|} (|S \cap W_{i-1}| - |S \cap W_i|)
\end{aligned}$$

L'última desigualtat es dóna per l'elecció del pas 2, on es tria S_j que minimitza y, i si la desigualtat no fos certa, hauríem elegit S_i abans que S. Sigui $s_i = |S \setminus W_{i-1}|$, aleshores,

$$\sum_{u \in S} y(u) \leq c(S) \sum_{i=1}^{k'} \frac{s_i - s_{i+1}}{s_i} \leq c(S) \sum_{i=1}^{k'} \left(\frac{1}{s_i} + \cdots + \frac{1}{s_{i+1} + 1} \right)$$

Si considerem $p(r) = 1 + \frac{1}{2} + \cdots + \frac{1}{r}$,

$$\begin{aligned}
\sum_{u \in S} y(u) &\leq c(S) \sum_{i=1}^{k'} (p(s_i) - p(s_{i+1})) = c(S)(p(s_i) - p(s_{k'+1}) \\
&\leq c(S)p(s_1) \leq c(S)p(r)
\end{aligned}$$

On l'última desigualtat es deu al fet que $s_1 = |S| \leq \max\{|S| : s \in \mathcal{S}\} = r$. Ara, si considerem el conjunt de cobertura òptim $\mathcal{O}$ podem comparar el cost d'aquest amb el de la solució trobada per l'algorisme,

$$\sum_{S \in \mathcal{O}} c(S)p(r) \geq \sum_{s \in \mathcal{O}} \sum_{u \in S} y(u) \geq \sum_{u \in U} y(u) = \sum_{i=1}^{k} \sum_{u \in \{v \in U : j(e) = i\}} y(u) = \sum_{i=1}^{k} c(S_i)$$

Per tant, l'algorisme de força bruta és un p-algorisme d'aproximació. □

3.2 Algorisme de Shmoys, Tardos i Aardal

L'algorisme de Shmoys, Tardos i Aardal considera la versió contínua del problema, on pot haver-hi fàbriques parcialment obertes i ciutats parcialment assignades, en troba la solució i l'arrodoneix a una solució entera. Un cop tenim la versió contínua del problema, l'algorisme es pot dividir en tres passos principals:

1. Es troba la solució del problema continu fent servir qualsevol mètode de programació lineal, com el mètode símplex que s'estudia en la secció 3.2.2.
2. S'aplica un subalgorisme per a filtrar les solucions, obtenint solucions fraccionals tals que els costos associats siguin baixos.
3. S'arrodoneixen els resultats per aconseguir solucions enteres.

Aquest mètode va ser plantejat i demostrat per Shmoys, Tardos i Aardal a l'article (1) i obté una solució aproximada. Concretament és un 4-algorisme d'aproximació.

Cal preparar el problema reescrivint-lo i afegint les següents variables:

- $y_i \in \{0,1\}\ \forall i \in \mathcal{F}$ indicant si s'elegeix el possible punt o no.
- $x_{i,j} \in \{0,1\}\ \forall i \in \mathcal{F},\ j \in \mathcal{D}$. Aquesta variable prendrà valor 0 per a tota j si no s'elegeix i, és a dir, si $y_i = 0$ i per a tota j existeix exactament un i tal que $x_{i,j} = 1$. Aquest fet ens permet assegurar que es cobrirà tota la demanda de forma que un mateix punt de demanda només serà atès per una fàbrica.

El problema passarà a ser:

$$\min \left\{ \begin{array}{l} \displaystyle\sum_{i \in F} f_i y_i + \sum_{j \in \mathcal{D}} \sum_{i \in \mathcal{F}} c_{i,j} x_{i,j} : \\ \qquad\qquad\qquad\qquad y_i \in \{0,1\}\ \forall i \in \mathcal{F}, \\ \qquad\qquad\qquad\qquad x_{i,j} \in \{0,1\}\ \forall i \in \mathcal{F}, j \in \mathcal{D} \\ \qquad\qquad\qquad\qquad x_{i,j} \leq y_i\ \forall i \in \mathcal{F}, j \in \mathcal{D} \\ \qquad\qquad\qquad\qquad \displaystyle\sum_{i \in \mathcal{F}} x_{i,j} = 1\ \forall j \in \mathcal{D} \end{array} \right\} \tag{3.4}$$

Alleugerint les condicions en fem la versió contínua:

$$\min \left\{ \begin{array}{l} \displaystyle\sum_{i \in F} f_i y_i + \sum_{j \in \mathcal{D}} \sum_{i \in \mathcal{F}} c_{i,j} x_{i,j} : \\ \qquad\qquad\qquad\qquad y_i \geq 0\ \forall i \in \mathcal{F}, \\ \qquad\qquad\qquad\qquad x_{i,j} \geq 0\ \forall i \in \mathcal{F}, j \in \mathcal{D} \\ \qquad\qquad\qquad\qquad x_{i,j} \leq y_i\ \forall i \in \mathcal{F}, j \in \mathcal{D} \\ \qquad\qquad\qquad\qquad \displaystyle\sum_{i \in \mathcal{F}} x_{i,j} = 1\ \forall j \in \mathcal{D} \end{array} \right\} \tag{3.5}$$

3.2.1 Consideracions prèvies

En aquest apartat es veuen alguns resultats d'optimització lineal necessaris per a desenvolupar el mètode símplex. Considerarem a partir d'ara el problema lineal general $\min\{c^T x : x \in P\}$, on $P = \{x \in \mathbb{R}^n : Ax \geq b\}$, on $c \in \mathbb{R}^n$, $A \in \mathbb{R}^{n \times n}$ i $b \in \mathbb{R}$. Qualsevol problema de programació lineal es pot transformar en un problema d'aquest tipus. Per exemple, el problema $\min\{c^T x : Ax = b,\ x \geq 0\}$, que és més semblant al problema que ens ocupa en aquest capítol és equivalent al problema $\min\{c^T x : -Ax \geq -b,\ Ax \geq b,\ x \geq 0\}$.

Definició 3.3. *Un* polítop *és un conjunt* $P = \{x \in \mathbb{R}^n : Ax \geq b\}$. *Si està acotat, se'n diu* polígon. *Anomenarem:*

- Solució factible x *a tot vector de* P. *Si una solució factible* x *és la solució al problema, s'anomena* solució òptima.
- Punt extrem *a un vector* $x \in P$ *tal que no existeixen* $y, z \in P \setminus \{x\}$ *i* $\lambda \in [0,1]$ *amb* $x = \lambda y + (1-\lambda)z$.

Direm que un poliedre P conté una recta *si existeixen* $x \in P$ *i* $d \in \mathbb{R}^n$ *diferent de zero tal que* $x + \lambda d \in P$ *per a tot* $\lambda \in \mathbb{R}$

Definició 3.4. *Considerem el problema anterior,* $\min\{c^T x : x \in P\}$*:*

- *Anomenem* variables bàsiques *al conjunt de variables del sistema de desigualtats que són linealment independents. És a dir, les columnes de les quals en* A *són linealment independents.*
- *Una* solució factible bàsica (òptima) *és una solució factible (òptima) tal que les seves components són zero en totes les variables bàsiques.*
- *Si una o més variables bàsiques d'una solució són zero, la solució s'anomena* bàsica degenerada

Existeixen dos casos en què no s'assoleix la solució:

1. Si P és buit, diem que el problema *no és factible.* Si no, és *factible.*
2. Si $\inf\{c^T x : x \in P\} = \infty$, diem que el problema *és no acotat.* Si no, és *acotat.*

Altrament, es pot veure que sempre existeix la solució:

Lema 3.5. *Sigui* $x \in P$. x *és una solució factible bàsica si i només si* x *és un punt extrem.*

Demostració. Suposem x és una solució factible bàsica i no un punt extrem. Com que és bàsica, podem considerar el conjunt de variables bàsiques I. Aleshores $\forall i \in I$, $A_i^T x = b_i$ i $\forall j \notin I$, $x_j = 0$. x per hipòtesis no és un punt extrem, i per tant existeixen $y, z \in P$ diferents de x i $\lambda \in [0,1]$ tal que $x = \lambda x + (1-\lambda)y$. Trobem els valors de les components de y i z. Per a tot $j \notin I$, $y_j + (1-)z_j = x_j = 0$ i per tant $y_j = z_j = 0$. Per altra banda, $\forall i \in I$, $A_i^T(y_i + (1-)z_i = A_i^T x_i = b_i$ i com que $y, z \in P$, $A_i^T y_i = A_i^T z_i = b_i$. Finalment obtenim $A_i^T(y_i - z_i) = 0$ amb els A_i linealment independents, concloent $y = z$ contradient la hipòtesi.

Suposem ara, que P té un punt extrem x. Demostrem que aleshores x és una solució factible bàsica, per contrarrecíproc.

Suposem que $y \in P$ no és una solució factible bàsica. Considerem $I = \{i : A_i^T y = b_i\}$. Com que x no és una solució bàsica, no existeixen n vectors linealment independents en $\{A_i : i \in I\}$. Per tant, existeix un vector $d \in \mathbb{R}$ tal que és ortogonal a tots els altres. Sigui $\epsilon > 0$ prou petit i considerem els vectors $y = x + \epsilon d$ i $z = x - \epsilon d$ complint $y, z \in P$. Però aleshores $x = \frac{1}{2}y + \frac{1}{2}z$, i x no és un punt extrem. □

Lema 3.6. *Sigui $P = \{x \in \mathcal{R}^n : a_i^T \geq b_i,\ i = 1, \ldots, m\}$ no buit. Aleshores són equivalents:*

1. *P té un punt extrem.*
2. *P no conté cap recta.*
3. *Existeixen n vectors linealment independents en el conjunt $\{A_i : i = 1, \ldots, m\}$ de les columnes de A.*

Demostració. $1 \Rightarrow 3$. Per el lema 3.5, x és una solució factible bàsica i tenim n columnes linealment independents.

$3 \Rightarrow 2$. Ho demostrarem per reducció a l'absurd. Suposem que existeixen n vectors linealment independents, $A_1, \ldots, A_n$ i que P conté una recta i arribem a un absurd. Com que conté una recta, existeix $x \in P$ i $d \in \mathbb{R}^n$ tal que $x + \lambda d \in P\ \forall \in \mathbb{R}$. Aleshores tenim $A_i^T(x+\lambda d) = A_i^T x + \lambda A_i^T d \geq b_i$, perquè pertany a P. Veiem que $A_i^T d = 0\ \forall i = 1, \ldots, n$, ja que altrament escollint λ molt gran o molt petit incompliríem alguna de les desigualtats. Finalment, com que els vectors $A_1, \ldots, A_n$ són linealment independents, $d = 0$.

$2 \Rightarrow 1$. Veurem que, si P no conté cap recta, té una solució factible bàsica i utilitzant el lema 3.5 tindrem un punt extrem.

Sigui $x \in P$ i $I = \{i : A_i^T = b_i\}$. Si n dels vectors de I són linealment independents, x és una solució factible bàsica i haurem acabat. Suposem doncs que no és així. Aleshores $I\mathbb{R}^n$ i podem trobar $d \in \mathbb{R}^n$ ortogonal a I. Considerem la recta $r(\lambda) = x + \lambda d$. Totes les variables que eren bàsiques en x segueixen sent bàsiques en $r(\lambda)$, perquè $A_i^T r(\lambda) = b_i$.

Malgrat això, hem suposat que P no conté cap recta i per tant, sempre que variem suficientment com perquè una de les desigualtats deixi de complir-se s'introdueix una nova variable bàsica. És a dir, existeixen $\lambda^* \in \mathbb{R}$ i $j \notin I$ tal que $A_j^T(x + \lambda^* d) = b_j$.

Vegem que A_j no és combinació lineal del conjunt $\{A_i : i \in I\}$. Com que $j \notin I$, $A_j^T x \neq b_j$ i sabent que $A_j^T(x + \lambda^* d) = b_j$ obtenim que $A_j^T d \neq 0$. És a dir, d no és ortogonal a A_j però si als A_i, i per tant són linealment independents.

Repetint el procés acabarem tenint n variables bàsiques linealment independents, i podem concloure que existeix un punt extrem en P. □

Teorema 3.7. *Suposem P conjunt factible amb, com a mínim, un punt extrem. Aleshores o bé el cost òptim és $-\infty$ o bé existeix $z \in P$ tal que $c^T z = \inf\{c^T x : x \in P\}$.*

Demostració. Considerem el rang d'un element $x \in P$ com la quantitat de desigualtats linealment independents actives en x. Suposem que el cost òptim és $-\infty$ i que té un punt extrem.

Considerem $x \in P$ de rang $k < n$. Volem trobar $y \in P$ amb rang superior i tal que el cost de la funció sigui igual o inferior. Sigui $I = \{i : A_i^T x = b_i\}$ Com que $k < n$, els vectors A_i tenen un vector ortogonal $d \in \mathbb{R}$. Canviant si cal el signe del vector d, podem tenir $c^T d \leq 0$. Seguim per casos:

Suposem $c^T d < 0$ i considerem la semirecta $r(\lambda) = x + \lambda d$ per $\lambda \in \mathbb{R}^+$. Per a tota $i \in I$,

$$A_i^T r(\lambda) = A_i^T x + \lambda A_i^T d = A_i^T = b_i$$

Ara bé, si tota la semirecta estigués continguda en P, el cost òptim seria $-\infty$, fet que contradiria la nostra hipòtesi. Per tant, existeix $\alpha^* > 0$ tal que per a tota $\alpha > \alpha^*$, $r(\alpha) \notin P$ i sigui $y = x + \lambda^* d$. Aleshores, existeix $j \notin I$ tal que $A_j^T y = b_j$, però tenim que $c^T y = c^T x + \lambda^* c^T d < c^T x$. Com que A_j és linealment independent de $\{A_i : i \in I\}$ el rang d'y és, com a mínim, $k + 1$.

En l'altre cas, si $c^T d = 0$, podem considerar la recta $r(\lambda) = x + \lambda d$, per $\lambda \in \mathbb{R}$. Com que P no conté cap recta (usant el lema 3.6 i que P té un punt extrem), la recta no pot estar continguda en P i amb el mateix raonament que abans obtenim $c^T y = c^T x$.

Repetint el procés fins a tenir el rang màxim, obtindrem punts $w \in P$ tals que $c^T w \leq c^T x$. Siguin $w_1, \ldots, w_s$ les solucions factibles bàsiques. Escollim w^* l'òptima entre elles i aleshores tindrem que, per a tot $x \in P$, existeix i tal que $c^T w_i \leq c^T$ i hem obtingut que w^* és òptima. □

Corol·lari 3.8. *Sigui $P = \{x \in \mathbb{R}^n : Ax \geq b\}$ conjunt no buit i acotat. Aleshores existeix la solució al problema lineal* $\min\{c^T x : x \in P\}$.

Corol·lari 3.9. *Suposem $A \in \mathbb{R}^{m \times n}$, amb $rang(A) = m$. Aleshores:*

1. *Si existeix una solució factible, existeix una solució factible bàsica.*
2. *Si existeix una solució factible òptima, existeix una solució factible òptima.*

3.2.2 Mètode símplex

El mètode símplex busca un mínim local, però com que $c^T x$ és una funció convexa, és també un mínim global. El mètode consisteix en iterar solucions factibles òptimes prenent sempre la direcció de màxim descens.

Definició 3.10. *Considerem $x \in P$. Una* direcció factible *és un vector $d \in \mathbb{R}^n$ tal que existeix $\theta \in \mathbb{R}^+$ amb $x + \theta d \in P$.*

Suposem que tenim una solució factible x. Denotarem per:

- $B(1), \cdots, B(m)$ els índexs de les variables bàsiques.
- A^i la columna i d'A.
- $B = \left(A^{B(1)}, \cdots, A^{B(m)}\right)$ la matriu formada per les variables bàsiques. L'anomenarem *matriu bàsica.*
- $x_B = \left(x_{B(1)}, \cdots, x_{B(m)}\right)$, de manera que $x = (x_B, 0)$
- $c_B = \left(c_{B(1)}, \cdots, c_{B(m)}\right)$

Definirem la translació $x + \theta d$ de manera que incrementem una quantitat θ adequada una variable no bàsica j, mantenint les altres variables no bàsiques a zero i modificant les bàsiques de manera que seguim en P. És a dir, tenim que $d_j = 1$ i $d_i = 0$ per tota variable no bàsica diferent de j.

Utilitzant que $x, x + \theta d \in P$,

$$b = A(x + \theta d) = Ax + \theta Ad = b + \theta Ad$$

I obtenim que $Ad = 0$, que es pot utilitzar per trobar d_B:

$$0 = Ad = \sum_{i=1}^{m} A^i d_i = \sum_{i=1}^{m} A^{B(i)} d_{B(i)} + A^j d_j = B d_B + A^j. \tag{3.6}$$

Com que B és invertible,

$$d_B = -B^{-1} A^j$$

Definició 3.11. *El vector direcció d, on $d_j = 1$ i $d_i = 0$ $\forall i \neq j$ variable bàsica s'anomena* direcció bàsica j.

Només resta garantir $x + \theta d \geq 0$. Com que totes les variables no bàsiques es mantenen a 0, excepte la component j que augmenta, només caldrà preocupar-se de les variables bàsiques. Segons x podem trobar-nos en dos casos:

1. Si x és no degenerada, és a dir $x_B > 0$, tindrem $x_B + \theta d_B \geq 0$ i la factibilitat és manté.
2. Si x és degenerada, és possible que la variable bàsica $x_{B(i)} = 0$ i $d_{B(i)} < 0$

Estudiarem el cas no degenerat primer. Ja hem trobat la direcció d que ens garanteix el decreixement més gran. Per trobar θ tal que la disminució de la funció sigui la més gran possible estudiarem la variació de la funció objectiu, $c^T x$.

La ràtio de *variació del cost* de x és:

$$c^T x - c^T (x - \theta d) = \theta c^T d = c_j + c_B^T d_B$$

Així doncs, la variació del cost de la variable no bàsica j és $\Delta c_j = c_j + c_B^T d_B = c_j - c_B^T B^{-1} A^j$. Falta ara trobar la funció de variació del cost en el cas de les variables bàsiques. Per altra banda, la variació del cost de una variable bàsica $B(i)$ és $\Delta c_{B(i)} = c_{B(i)} - c_B^T B^{-1} A^{B(i)} = c_{B(i)} - c_B^T \left(A^{B(1)} \cdots A^{B(m)}\right)^{-1} A^{B(i)} = c_{B(i)} - c_B^T e_i = c_{B(i)} - c_{B(i)} = 0$.

Teorema 3.12. *Sigui x una solució factible i sigui Δc el vector de variació del cost. Aleshores,*

1. *Si $\Delta c \geq 0$, x és una solució factible òptima.*
2. *Si x és òptima i no degenerada, $\Delta c \geq 0$.*

Demostració. Suposem x solució factible bàsica amb $\Delta c \geq 0$, y una altra solució factible bàsica qualsevol i considerem $z = y - x$. Com que tant x com y són factibles, $Az = A(y-x) = Ay - Ax = 0$, per tant, $Bz_B + \sum_{i \in N} A^i z_i = 0$, on N és el conjunt dels índexs de les variables no bàsiques. Com que B és invertible, $z_B = -\sum_{i \in N} B^{-1} A^i d_i$. Així doncs,

$$c^T z = c^T z_b + \sum_{i \in N} c_i^T z_i = -\sum_{i \in N} c_B^T B^{-1} A^i z_i + \sum_{i \in N} c_i^T z_i = \sum_{i \in N} (c_i^T - c_B^T B^{-1} A_i) z_i = \sum_{i \in N} \Delta c_i z_i$$

Ara, com que $\forall i \in N$, $x_i = 0$, perquè són variables no bàsiques en x i $y_i \geq 0$, ja que y és solució factible, amb la definició de z veiem que $z_i \geq 0$ i per tant, $\Delta c_i^T z_i \geq 0$. Finalment podem concloure que x era òptima, ja que qualsevol altra solució factible bàsica té un cost major.

Per a demostrar la segona part suposem que $x \neq 0$ és una solució factible bàsica òptima i que per algun j, $\Delta c_j < 0$ i busquem un absurd. Com que $\Delta c_j < 0$, j és una direcció factible i prenent-la podem obtenir una nova solució factible amb cost menor, i x no és l'òptima. □

Per tant, si tenim x una solució factible bàsica no degenerada tal que $\Delta c \geq 0$ (ja que altrament ja tindríem la solució òptima) la θ que ens permet augmentar més la variació del cost és $\theta = \max\{\theta \geq 0 : x + \theta d \in P\}$, donat que ja tenim la direcció que ens permet disminuir més, i seguirem tant com es pugui en aquella direcció per disminuir més.

Cal resoldre ara $\max\{\theta \geq 0 : x + \theta d \in P\}$. Com que a l'equació 3.6 hem vist que, $\forall \theta \geq 0$, $A(x + \theta d) = b$ i només falta garantir que $x + \theta d \geq 0$. Si $d \geq 0$, òbviament tindrem que $x + \theta d \geq 0$ $\forall \theta \geq 0$, i per tant podem prendre $\theta = +\infty$ i P no és acotat.

Per altra banda, si existeix i tal que $d_i \leq 0$, per garantir que $x_i + \theta d_i \geq 0$, cal que:

$$\theta \leq -\frac{x_i}{d_i}$$

i per tant, la θ que major disminució produeix és $\Theta = \min\left\{-\frac{x_i}{d_i} : d_i < 0\right\}$. Tenint en compte que per a tota variable no bàsica i, $d_i \in \{0, 1\}$, l'anterior és equivalent a:

$$\Theta = \min_{\{i : d_{B(i)}\}} \left\{-\frac{x_i}{d_i}\right\} \tag{3.7}$$

Fet que ens ajuda a disminuir la quantitat d'operacions a fer.

Ara ja hem trobat una nova solució factible, que anomenarem $y = x + \Theta d$. Per seguir iterant trobant una nova direcció factible, caldrà canviar la matriu B, és a dir les columnes que triem com a linealment independents. Hem de triar un índex j que afegir a la matriu B, així com decidir quina columna eliminar, d'índex $B(k)$.

Triem primer k. Considerem l'índex que minimitza l'equació 3.7, l'anomenem $B(k)$. Com que $y_{B(k)} = x_{B(k)} + \Theta d_{B(k)} = x_{B(k)} - \frac{x_{B(k)}}{d_{B(k)}} d_{B(k)} = 0$, podríem treure aquest índex. L'índex que afegirem a la base serà un índex tal que $\Delta c_j < 0$,és a dir, que la direcció j sigui una direcció factible, però pot ser que existeixi més d'un j tal que $\Delta c_j < 0$. Hi ha diversos mètodes per a elegir el pivot en aquest cas:

- Escollim j tal que Δc_j sigui el més petit. Tot i que d'aquesta manera agafarem la direcció de màxim descens, el que realment importa és com de lluny anem cap aquesta direcció.

- Elegim j tal que $\Theta|\Delta c_j|$ sigui major. Malgrat que es pot demostrar que usant aquest mètode es necessiten menys iteracions, augmenta la complexitat, i el resultat no és millor que els altres mètodes.
- Triem el j més petit tal que $\Delta c_j < 0$.

Així doncs, en aquest pas canviarem la nostra matriu i els índexs de les variables bàsiques. Així si $B_1(i)$ és l'índex de la variable bàsica número i en la primera iteració, en la següent serà:

$$B_2(i) = \begin{cases} B_1(i) & \text{si } i \neq K \\ j & \text{si } i = k \end{cases}$$

Teorema 3.13. *1. Les columnes $A^{B_2(i)}$ són linealment independents. Per tant, $B_2 = \left(A^{B_2(1)}, \cdots, A^{B_2(m)}\right)$ és una matriu bàsica.*

2. El vector $y = x + \theta d$ és una solució factible bàsica associada a la matriu base B_2.

Demostració. Demostrem el primer apartat per reducció a l'absurd. Suposem que les columnes $\{A^{B_1(i)} : i = 1, \ldots, m\}$ són linealment independents però $\{A^{B_2(i)} : i = 1, \ldots, m\}$ no. Aleshores existeixen $\lambda_1, \ldots, \lambda_m$ algun diferent de zero tal que $\sum_{i=1}^{m} \lambda_i A^{B_2(i)} = 0$ i per tant, els vectors $\{B^{-1}A^{B_2(i)} : i = 1, \ldots, m\}$. Veurem, però, que són linealment independents. $\forall i \neq l$, $B^{-1}A^{B_1(i)} = e_i$, és a dir que tots tenen el valor de la component l igual a zero. Per altra banda, $B^{-1}A^{B_2(l)} = B^{-1}A^j = -d_B$, i per tant la component de l és diferent de zero, fet que implica que són linealment independents.

Sigui $y = x + \Theta d$. En la construcció de y ja hem vist que és factible. Resta veure que és bàsica. Per l'apartat anterior, les columnes $\{A^{B_2(i)} : i = 1, \ldots, m\}$ són linealment independents, i per tant y és una solució factible bàsica. □

Així doncs, una iteració del mètode símplex en el cas que tinguem una solució no degenerada és:

1. Comencem amb una solució factible x. Sabem que existeix per la proposició 3.7 . Escollim les columnes $A^{B(1)}, \cdots, A^{B(m)}$ corresponents a les variables bàsiques triades i definim que formin la matriu base B.
2. Calculem Δc_j per a tots els índexs no bàsics j. Per casos:
   (a) Si $\forall j$, $\Delta c_j \geq 0$, la solució és òptima pel teorema **??** i aturem el programa.
   (b) Altrament triem j tal que $c_j < 0$.
3. Calculem $d_B = B^{-1}A^j$.
   (a) Si cap component del vector $B^{-1}A^j$ és positiu, com hem vist abans, el cost òptim és $-\infty$ i aturem el programa.
   (b) Altrament, definim $\Theta := \min\left\{-\frac{x_{B(i)}}{d_i} : \ i = 1, \cdots, m;\ d_i > 0\right\}$. Com que per calcular el mínim prenem i variable bàsica, calculant d_B ja tenim totes les d_i necessàries.
4. Escollim k tal que $\Theta = \frac{x_{B(k)}}{d_k}$. Construïm la nova matriu bàsica substituint $A^{B(k)}$ per A^j. Calculem $y = x + \Theta d \neq x$, que és la nova solució factible bàsica. Iterem tornant al pas 2.

Veiem que en el cas en què ens trobem, el no degenerat, el mètode símplex troba la solució en un nombre finit d'iteracions.

Teorema 3.14. *Suposem que $P = \{x \in \mathbb{R}^n : Ax = b, x > 0\}$ és no buit i què tota solució factible bàsica trobada en les iteracions del mètode símplex acaba en un nombre finit d'iteracions. Aleshores el programa símplex acaba i pot fer-ho en dos punts diferents:*

1. *Si ho fa en el pas 2, tenim una solució factible bàsica òptima.*
2. *Si ho fa en el pas 3, el cost òptim és $-\infty$ i P és no acotat.*

Demostració. Demostrem primer que el programa acaba després d'un nombre finit d'iteracions. A cada iteració ens movem una quantitat finita Θ al llarg d'una direcció factible d que hem escollit de manera que disminueixi el cost de la funció objectiu i és una nova solució factible bàsica. Hi ha un nombre finit de solucions factibles bàsiques, i per tant l'algorisme ha d'acabar en algun moment.

Si acaba el el pas dos, pel lema 3.12 com que $\Delta c_j \geq 0$, la solució és òptima.

Si per altra banda acabem el tercer pas, existeix una direcció factible j tal que $\Delta c_j < 0$ i satisfà que per a tota $\theta > 0$, $x + \theta d \in P$, i per tant el cost mínim és $-\infty$. □

Cal veure ara que fer en el cas degenerat. Aplicant els mateixos passos en la iteració que en el cas no degenerat ens podem trobar en dos problemes diferents.

El primer és que, si la solució factible és degenerada, podríem tenir $\Theta = 0$, ja que podríem tenir que la variable bàsica degenerada k, és a dir tal que $x_{B(k)} = 0$, també compleix $d_{B(k)} < 0$. En aquest cas podem triar la nova $y = x$ i simplement canviar la matriu bàsica, substituint la columna $A^{B(k)}$, per A^j trobat de la mateixa manera que en el cas no degenerat, és a dir seguint algun dels mètodes per trobar j tal que Δc_j.

L'altre problema que podríem tenir és que més d'una de les variables bàsiques originals es tornen 0 en $x + \Theta d$. Com que només modifiquem una de les columnes de la matriu bàsica, la nova solució també haurà de ser degenerada.

En qualsevol cas trobant iterativament les matrius bàsiques podríem tenir cicles, és a dir, podem tornar a la matriu bàsica original i les iteracions mai acabarien. Però triant de forma correcta els índexs de les columnes a canviar podem evitar aquest error. Es pot demostrar que, utilitzant el mètode per triar j vist anteriorment consistent en elegir l'índex j més petit tal que $\Delta c_j < 0$ s'eviten els cicles.

3.2.3 Filtratge

Per simplificar la notació, considerarem un conjunt N tal que $D \subseteq N$ i $F \subseteq N$ i suposarem que $\forall i, j$ si $i \notin D$ o $j \notin F$, aleshores $x_{i,j} = 0$. Anomenarem $m = \#N$. Per altra banda, haurem d'estendre la definició de la variable de costos c que, a més de complir la desigualtat 3.2, també caldrà que tingui les propietats resultants de ser una mètrica:

- $c_{i,i} = 0 \ \forall i \in N$.
- $c_{i,j} = c_{j,i} \ \forall i, j \in N$

Donada una solució factible (x, y), amb aquesta tècnica de filtratge se'n buscarà una altra tal que els pesos associats $c_{i,j}$ que hi actuen siguin prou petits. Aquesta noció és de proximitat.

Definició 3.15. *Fixem g_j $\forall j \in D$. Una solució factible (x, y) del programa lineal és* g-pròxima *si se satisfà que, $\forall i, j$ tal que $x_{i,j} > 0$ es dóna que $c_{i,j} \leq g_j$.*

Per a tota j, considerem la permutació π de N que ordena deforma ascendent $c_{i,j}$, és a dir $c_{\pi(1),j} \leq c_{\pi(2),j} \leq \cdots \leq c_{\pi(m),j}$. Fixem $\alpha > 0$ i definim $c_j(\alpha) := c_{\pi(i'),j}$, on $i' = \min\left\{k : \sum_{i=0}^{k} c_{\pi(i),j} \geq \alpha\right\}$.

L'objectiu és trobar una nova solució fraccional (x^*, y^*) que compleixi aquestes tres condicions, que posteriorment seran necessàries per a seguir amb l'algorisme:

1. (x^*, y^*) és una solució fraccional factible g-pròxima.
2. $g_j \leq c_j(\alpha)$ $\forall j \in D$
3. $\sum_{i \in F} \alpha f_i y_i^* \leq \sum_{i \in F} f_i y_i$

Definim $\forall j \in D$, $\alpha_j = \sum_{\substack{i \in F \\ c_{i,j} \leq c_j(\alpha)}} x_{i,j}$. Com que

$$\alpha_j = \sum_{\substack{i \in F \\ c_{i,j} \leq c_j(\alpha)}} x_{i,j} = \sum_{\substack{c_{i,j} \leq c_{\pi(i'),j},\ i \in F \\ i' = min\{k : \sum_{i=1}^{k} x_{\pi(i),j} \geq \alpha\}}} x_{\pi(i),j} = \sum_{i=0}^{i'} x_{\pi(i),j} \geq \alpha$$

Aleshores, si definim (x^*, y^*) de la següent es compleixen les propietats anteriors:

$$x_{i,j}^* = \begin{cases} \frac{x_{i,j}}{\alpha_j} & \text{si } c_{i,j} \leq c_j(\alpha) \\ 0 & \text{altrament} \end{cases} \tag{3.8}$$

$$y^* = \min\left\{1, \frac{y_i}{\alpha}\right\} \tag{3.9}$$

Veiem primer que (x^*, y^*) és una solució fraccional factible, és a dir que compleix les condicions del problema 3.5.

- Clarament, $x^* \geq 0$ i $y^* \geq 0$.
- Si $x_{i,j}^* = 0$, clarament $x^* \leq y^*$. Per tant suposem que $c_{i,j} \leq c_j(\alpha)$ i $x_{i,j}^* = \frac{x_{i,j}}{\alpha_j}$. Aleshores, per una banda, $x_{i,j}^* \leq \frac{x_{i,j}}{\alpha} \leq \frac{y_i}{\alpha}$. Per l'altra, $x_{i,j}^* = \frac{x_{i,j}}{\sum_{c_{i,j} \leq c_j(\alpha)}} \leq 1$. Per tant, $x_{i,j}^* \leq \min\left\{1, \frac{y_i}{\alpha}\right\} = y_i^*$.
- $\sum_{i \in F} x_{i,j}^* = \sum_{i=1}^{i'} \frac{x_{\pi(i),j}}{\alpha_j} = \frac{1}{\alpha_j} \sum_{i=1}^{i'} x_{\pi(i),j} = \left(\sum_{i=1}^{i'} x_{\pi(i),j}\right)^{-1} \left(\sum_{i=1}^{i'} x_{\pi(i),j}\right) = 1.$

Amb aquesta tècnica de filtratge creem clústers. En la fase d'arrodoniment obrirem una de les fàbriques en cada clúster.

3.2.4 Arrodoniment

Donada una solució (x^*, y^*) g-pròxima volem trobar la solució entera més propera, que serà 3g-pròxima, sense augmentar el pes d'obertura dels centres $\sum_{i \in F} f_i y_i$. Es segueix de forma iterativa, considerant $(\overline{x}, \overline{y})$ el resultat de les iteracions.

Denotarem:

- $\overline{F} = \{i \in F : 0 < \overline{y}_i < 1\}$, el conjunt de punts parcialment escollits.
- $\overline{D} = \{j \in D : \forall i,\, \overline{x}_{i,j} > 0 \Rightarrow i \in \overline{F}\}$, els punts donats assignats només a punts de $\overline{F}$.

Només haurem de contemplar els conjunts $\overline{F}$ i $\overline{D}$ enlloc de F i D, ja que són els parcialment oberts. Inicialment, començarem amb $(\overline{x}, \overline{y}) = (x^*, y^*)$, on (x^*, y^*) és producte de l'algorisme de filtratge anterior. Volem que $\forall i \in F$, $j \in D$ tinguem $\overline{x}_{i,j}$, $\overline{y}_i \in \{0, 1\}$. Per fer-ho seguirem els passos següents:

1. Trobem $j' \in \overline{D}$, l'índex que minimitza g tal com està definida en **??**. Definim $S = \{i \in \overline{F} : \overline{x}_{i,j'} > 0\}$ el conjunt de fàbriques parcialment obertes que tenen com a assignació parcial $j' \in \overline{D}$.
2. Trobem $i' \in S$ tal que $f_{i'}$ sigui la component mínima de f. Assignem j' a i', és a dir, fixem $\overline{x}_{i',j'} = 1$.
3. Arrodonim $\overline{y}$ seguin la següent norma:

$$\overline{y}_i = \begin{cases} 1 & \text{si } i = i' \\ 0 & \text{si } i \in S \setminus \{i'\} \end{cases}$$

   D'aquesta manera tanquem totes les fàbriques que no tenen com a assignació i'.
4. Definim $T = \{j \in D : \exists i \in S \text{ tal que } \overline{x}_{i,j} > 0\}$. Com que s'ha decidit no escollir les fàbriques $i \in S \setminus \{i'\}$, cal eliminar també les assignacions parcials que tenia. Així doncs, per a cada $j \in T$, modifiquem $\overline{x}$ així:

$$\overline{x}_{i,j} = \begin{cases} 1 & \text{si } i = i' \\ 0 & \text{si } i \neq i' \end{cases}$$

5. En el cas en què el nou conjunt $\overline{D} =$, per a tota j, existeix i' tal que $\overline{x}_{i',j} > 0$ i $\overline{y}_{i'} = 1$ i fixem $\overline{x}_{i',j} = 1$ i $\overline{x}_{i,j} = 0$ per a tota $i \neq i'$. Altrament tornem al pas 1.

Lema 3.16. *A cada iteració de l'algorisme es mantenen les propietats:*

1. $(\overline{x}, \overline{y})$ *és una solució fraccional factible.*
2. $\sum_{i \in F} f_i \overline{y}_i \leq \sum_{i \in F} f_i y_i^*$
3. *Si* $\overline{x}_{i,j} > 0$, *aleshores* $c_{i,j} < g_j \ \forall i \in \overline{F}$
4. *Si* $\overline{x}_{i,j} > 0$, *aleshores* $c_{i,j} < 3g_j \ \forall i \notin \overline{F}$

Demostració. Per la secció anterior, sabem que a l'inici es compleix la propietat. Per altra banda es compleix en cada iteració, ja que només fixem $\overline{y}_i = 1$ quan abans de començar la iteració teníem $\overline{y}_i > 0$, i quan fixem $\overline{y}_i = 0$, en els següents passos canviem també $\overline{x}$ de manera que $\overline{x}_{i,j} = 0 \quad \forall j \in D$.

Suposem ara que tenim una solució factible fraccional inicial (x^*, y^*) i que operem l'algorisme. El primer canvi que tindrem en el valor de la suma $\sum_{i \in F} f_i \overline{y}_i$ és quan elegim i' tal que $f_{i'} = \min\{f_i : i \in S\}$. Però aleshores,

$$f_{i'} = \min\{f_i : i \in S\} \leq \sum_{i \in S} f_{i'} \overline{x}_{i,j} \leq \sum_{i \in S} f_i \overline{x}_{i,j} \leq \sum_{i \in S} f_i \overline{y}_i$$

On la segona igualtat és degut al fet que $(\overline{x}, \overline{y})$ és solució factible, i per definició tindrem $\sum_{i \in S} x_{i,j} = 1$. Però, com que en acabar la iteració el valor que prendrà $\sum_{i \in S} f_i \overline{y}_i$ serà $f_{i'} \overline{y}_{i'} = f_{i'}$, sempre n'estem disminuint o mantenint el valor, i es manté la segona propietat.

Pel que fa a la tercera, és degut al fet que no s'afegeixen variables a $\overline{F}$ ni fixa $x_{i,j} \in (0, 1)$.

Finalment falta veure l'última propietat. Suposem que estem en la iteració en què es fixa $\overline{x}_{i',j}$ a 1. Això significa que $j \in T$, i per tant que existeix $i \in S$ tal que $\overline{x}_{i,j} > 0$. A més també tenim $x_{i',j'} > 0$. Com que $S \subseteq \overline{F}$, aplicant la propietat anterior, $c_{i,j} \leq g_j$, $c_{i,j'} \leq g_{j'}$ i $c_{i',j'} \leq g_{j'}$. Aplicant la desigualtat triangular, tenim el resultat:

$$c_{i',j} \leq c_{i',j'} + c_{j',i} + c_{i,j} = c_{i',j'} + c_{i,j'} + c_{i,j} \leq 2g_{j'} + g_j \leq 3g_j$$

□

Lema 3.17. *Considerem el conjunt* $C = \{i : c_{i,j} \geq c_j(\alpha)\}$. *Aleshores,* $c_j(\alpha) \leq \frac{1}{1-\alpha} \sum_{i \in F} c_{i,j} x_{i,j}$

Demostració.

$$\sum_{i \in F} c_{i,j} x_{i,j} \geq \sum_{i \in C} c_{i,j} x_{i,j} \geq \sum_{i \in C} c_j(\alpha) x_{i,j} \geq c_j(\alpha) \sum_{i \in C} x_{i,j} \geq c_j(\alpha)(1 - \alpha)$$

Cal demostrar l'última desigualtat:

$$\sum_{i \in C} x_{i,j} = \sum_{i=i'}^{m} x_{\pi(i),j} = \sum_{i \in N} x_{i,j} - \sum_{i=1}^{i'-1} x_{\pi(i),j} \geq 1 - \alpha$$

On $i' = \min\{k : \sum_{i=1}^{k} x_{\pi(i),j} \geq\}$, i per tant $\sum_{i=1}^{i'-1} x_{\pi(i),j} \leq \alpha$. □

Finalment podem veure que és l'algorisme és un 4-algorisme d'aproximació:

Teorema 3.18. *L'algorisme de Shmoys, Tardos i Aardal és un 4-algorisme d'aproximació.*

Demostració. Cal calcular la funció objectiu. Com que al primer pas hem trobat la solució exacta (x, y) (encara que sigui amb el problema continu), volem veure que la solució factible entera obtinguda al final del programa té un cost de com a molt, 4 cops

el cost de (x, y).

Per una banda, utilitzant el lema 3.16 i la segona propietat demostrada a la secció 3.2.3,

$$\sum_{i \in F} f_i \overline{y}_i \leq \sum_{i \in F} f_i y_i^* \leq \frac{1}{\alpha} \sum_{i \in F} f_i y_i$$

Per altra banda, amb la propietat 2 de la secció 3.2.3 així com el 3.17,

$$3g_j \leq 3c_j(\alpha) \leq \frac{3}{1-\alpha} \sum_{i \in F} c_{i,j} x_{i,j}$$

Unint les dues trobem una cota del cost total de la funció objectiu:

$$\begin{aligned} &\sum_{i \in F} f_i \overline{y}_i + \sum_{i \in F} \sum_{j \in D} c_{i,j} \overline{x}_{i,j} \leq \frac{1}{\alpha} \sum_{i \in F} f_i y_i + \frac{3}{1-\alpha} \sum_{i \in F} \sum_{j \in D} c_{i,j} x_{i,j} \\ &\leq \max \left\{ \frac{1}{\alpha}, \frac{3}{1-\alpha} \right\} \left(\sum_{i \in F} f_i y_i + \sum_{i \in F} \sum_{j \in D} c_{i,j} x_{i,j} \right) \end{aligned}$$

Ara, l'α que compleix $\min \left\{ \max \left\{ \frac{1}{\alpha}, \frac{3}{1-\alpha} \right\} \right\}$ és $\alpha = \frac{1}{4}$. I obtenim que la solució $(\overline{x}, \overline{y})$ té un cost com a molt, 4 cops superior a l'òptim. □

3.2.5 Modificacions de l'algorisme

Posteriorment, la ràtio d'efectivitat ha estat millorada afegint petites modificacions. La primera d'elles va ser obtinguda pels mateixos Shmoys, Tardos i Aardal i consisteix en escollir α de forma aleatòria. D'aquesta manera s'aconsegueix un factor d'aproximació millor que l'anterior:

Teorema 3.19. *L'algorisme de Shmoys, Tardos i Aardal és un* $3.16 \approx \left(\frac{3}{1+e^{-3}} \right)$*-algorisme d'aproximació.*

Chudak i Shmoys van millorar també el factor resolen de forma simultània $g = v^*$, on v^* correspon a la solució dual i modificant l'algorisme d'arrodoniment, amb modificacions en l'algorisme d'arrodoniment, com la d'obrir les fàbriques amb probabilitat van aconseguir una $1.74 \approx \left(1 + \frac{1}{e}\right)$-algorisme d'aproximació.

Finalment, Sviridenk va obtenir una ràtio de 1.58. S'han obtingut molts altres algorismes que donen millors aproximacions i tenen menys complexitat, sobretot perquè no necessiten un algorisme de programació lineal com a subrutina. A la següent secció es parla de dos d'ells.

3.3 Algorismes primals-duals

Els algorismes primals-duals resolen de forma simultània el problema original i el seu dual. Abans d'explicar-los, però, és necessari presentar teòricament el dual d'un problema i alguns resultats importants per a la construcció d'aquests algorismes.

3.3.1 Dualitat i folga complementària

Introduïm primer el dual d'un problema lineal:

Definició 3.20. *Considerem el problema d'optimització lineal* $\min\{c^T x : Ax \geq b,\ x \geq 0\}$*, on* A *és una matriu* $m \times n$*,* x *un vector columna de dimensió* n*,* b *de dimensió* m*,* c^T *un vector fila de dimensió* n*. Aleshores el seu* dual *és* $\max\{y^T b : y^T A \leq c^T,\ y \geq 0$*, on* y^T *és un vector fila* m*-dimensional. Al problema original se l'anomena* primal*.*

Per simplificar la secció, en lloc de treballar amb el problema $\min\{c^T x : Ax \geq b,\ x \geq 0\}$ ho farem amb

$$\min\{c^T x : Ax = b,\ x \geq 0\} \tag{3.10}$$

Com que en la secció 3.2.1 s'ha vist que els dos són equivalents, serà suficient. El seu dual és

$$\max\{y^T b : y^T A = c^T\} \tag{3.11}$$

Veurem a continuació que el valor òptim del problema primal i dual són el mateix.

Lema 3.21. *Si* x *i* y *són solucions factibles dels problemes 3.10 i 3.11 respectivament, aleshores* $c^T x \geq y^T b$*.*

Demostració.

$$c^T x \geq (y^T A)x = y^T Ax = y^T b$$

□

Proposició 3.22. *Si* x *i* y *són solucions factibles dels problemes 3.10 i 3.11 respectivament i* $c^T x = y^T b$*, aleshores* x *i* y *són solucions òptimes dels seus respectius problemes.*

Demostració. Pel lema anterior, els valors associats al problema primal, que volem minimitzar, són sempre superiors o iguals als valors associats al problema dual, que volem maximitzar. Així, quan $c^T x = y^T b$, x i y són òptims pels respectius problemes. □

Per tant hem vist que, si existeixen dos vectors, un del primal i un del dual, amb valors de la funció objectiu iguals, aleshores són òptims pels seus respectius problemes. Ens falta demostrar la inversa, però abans cal veure el teorema de l'hiperplà separador.

Teorema 3.23. *Sigui* $C \subseteq \mathbb{R}^n$ *un conjunt convex i* $y \notin \overline{C}$*. Aleshores existeix* $a \in \mathbb{R}^n$ *tal que* $a^T y < \inf\{a^T x : x \in C\}$*.*

Demostració. Considerem $\delta = \inf\{|x-y| : x \in C\} > 0$. Com que la funció $f(x) = |x-y|$ és continua, assoleix el seu mínim sobre qualsevol conjunt tancat i acotat, és a dir, existeix $x_0 \in Fr(C)$ tal que $|x_0 - y| = \delta$. De fet, x_0 ha d'estar en $\overline{C} \cap \overline{B}_{2\delta}(y)$.

Fixem $a = x_0 - y$ i vegem que es satisfà $a^T y < \inf\{a^T x : x \in C\}$. Com que C és un conjunt convex i $x_0 \in C$, $\forall \alpha \in [0,1]$, $x_0 + \alpha(x - x_0) \in \overline{C}$. D'aquesta manera,

$$\begin{aligned} |x_0 + \alpha(x-x_0) - y|^2 &\geq |x_0 - y|^2 \\ \alpha^2|a|^2 + |x_0 - y|^2 + 2\alpha(x_0 - y)^T(x - x_0) &\geq |a|^2 \\ 2(x_0 - y)^T(x - x_0) + \alpha^2|x - x_0|^2 &\geq 0 \end{aligned}$$

I, fent tendir $\alpha \to 0^+$ obtenim que $(x_0 - y)^T(x - x_0) \geq 0$, és a dir,

$$\begin{aligned} a^T x = (x_0 - y)^T x \geq (x_0 - y)^T x_0 = (x_0 - y)^T y + (x_0 - y)^T(x_0 - y) = \\ = (x_0 - y)^T y + |x_0 - y|^2 = (x_0 - y)^T y + \delta^2 = a^T y + \delta^2 \end{aligned}$$

Per tant, $a^T y \leq a^T x - \delta^2$ $\forall x \in C$ i es tracta del mínim, com volíem demostrar. □

Teorema 3.24. *1. Si un dels dos problemes, primal o dual, té una solució factible òptima finita, també la té l'altre i els valors de les funcions objectius són iguals.*

2. Si un problema és no acotat, l'altre no té solució factible.

Demostració. El segon punt es veu a partir del lema i la proposició anterior. Si, per exemple, el primal no és acotat, $c^T x$ és tan petit com es vulgui i com que $c^T x \geq y^T b$, el problema dual no té solució factible. El contrari és simètric.

Per demostrar el primer punt suposarem que el primal té solució factible òptima i veurem que el dual també té una solució amb el mateix valor. Així doncs, suposem que el problema 3.10 té solució òptima finita amb valor z_0. Considerem el conjunt

$$C = \{(r,w) : r = tz_0 - c^T x,\ w = tb - Ax,\ x \geq 0,\ t \geq 0\} \tag{3.12}$$

Amb uns càlculs senzills es pot veure que C és un conjunt convex, de fet un *con convex*, és a dir que $\forall x \in C$ i $\forall \alpha > 0$, $\alpha x \in C$. Veurem ara que $(1,0) \notin C$ i aplicarem després l'hiperplà separador, per a finalment trobar y tal que $y^T b = c^T x$.

Suposem que $(1,0) \in C$ i arribarem a una contradicció. Sigui $x_0, y_0 \geq 0$ tal que $(t_0 z_0 - c^T x_0,\ t_0 b - A x_0) = (1,0)$. Com que $t_0 b - A x_0 = 0$, $x = \frac{x_0}{t_0}$ és solució factible del problema primal, ja que $Ax = A\frac{x_0}{t_0} = b$ i $x \geq 0$ i $\frac{r}{t_0} = z_0 - c^T x_0 \leq 0$, ja que z_0 és l'òptim del mínim.

Ara, si $w = -Ax_0 = 0$ i $x_0 \geq 0$ i $c^T x_0 = -1$ i x és una solució factible, aleshores $x + \alpha x_0$ és factible per a tot $\alpha \geq 0$ i dóna valors que disminueixen segons augmenta α. Aquest fet contradiu la hipòtesi de valor òptim i per tant no existeix x_0 i $(1,0) \notin C$.

Podem aplicar el teorema 3.23 que ens diu que existeix un hiperplà que separa $(1,0)$ i C. Així, existeixen $s \in \mathbb{R}$, $y \in \mathbb{R}^m$ i una constant $c \in \mathbb{R}$ tals que,

$$s < c = \inf\{sr + y^T w : (r,w) \in C\}$$

Per una banda, $c \geq 0$, ja que altrament existiria $(r, w) \in C$ tals que $sr + y^T w < 0$ i, com que C és un con, $\forall \alpha > 0$, $\alpha(r, w) \in C$ i per α grans violaria la desigualtat de l'hiperplà. Per altra banda, com que $(0, 0) \in C$, $c \leq 0$. Per tant $c = 0$ i $s < 0$ i sense pèrdua de generalitat podem suposar $s = -1$.

Així doncs, existeix $y \in \mathbb{R}^m$ tal que $-r + y^T w \geq 0 \; \forall (r, w) \in C$. Substituint els valors de r i w tenim que $\forall t, x \geq 0$, $(c^T - y^T A)x - tz_0 + ty^T b \geq 0$. Si fixem $x = 0$ i $t = 1$, $y^T b \geq z_0$ i y és una solució factible òptima del problema dual. □

Finalment vegem una propietat que ens relaciona les solucions òptimes duals i primals.

Proposició 3.25. *Considerem el problema primal 3.10 i dual 3.11 i siguin x i y solucions factibles dels respectius problemes. Aleshores són equivalents:*

1. *x i y són solucions òptimes dels respectius problemes.*
2. *cx=yb*
3. *$y(b$-$Ax)$=0*

Demostració. El lema 3.22 ens demostra l'equivalència entre els dos primers. Vegem l'equivalència entre 2 i 3:

$$y(b - Ax) = yb - yAx = cx - yAx = cx - cx = 0$$

□

L'última desigualtat s'anomena *folga complementària* i la utilitzarem posteriorment en la construcció dels algorismes. També existeix la versió dual d'aquesta propietat, la *folga dual complementària*, que és $(c - yA)x = 0$ i és certa en els mateixos casos que la folga complementària.

3.3.2 Algorisme de Jain i Vazirani

El primer algorisme primal-dual per resoldre el problema el van trobar Jain i Vazirani a (6). A més a més de tenir una ràtio d'aproximació millor que la de l'algorisme de Shmoys, Tardos i Ardal, també té una complexitat molt inferior als anteriors, ja que no utilitza cap algorisme lineal com a subrutina.

L'algorisme interpreta el problema a resoldre com un graf bipartit on cal connectar tots els vèrtexs de $\mathcal{D}$ a alguns vèrtexs de $\mathcal{F}$. Com que és un algorisme primal-dual, troba la solució del problema dual alhora que la de l'original.

Per començar, però, cal trobar el dual del problema. Considerem el problema de l'equació 3.5 i veiem que, prenent c, A i b adequades el podem convertir en un programa lineal estàndard del qual saben trobar-ne el dual, és a dir de la forma $\min\{c^T x : Ax \geq b, x \geq 0\}$. Aquestes són:

$$c = \begin{pmatrix} c_{1,1} \\ \vdots \\ c_{1,m} \\ c_{2,1} \\ \vdots \\ c_{m,1} \\ \vdots \\ c_{m,m} \\ f_1 \\ \vdots \\ f_m \end{pmatrix}, \qquad x = \begin{pmatrix} x_{1,1} \\ \vdots \\ x_{1,m} \\ x_{2,1} \\ \vdots \\ x_{m,1} \\ \vdots \\ x_{m,m} \\ y_1 \\ \vdots \\ y_m \end{pmatrix}, \qquad b = \begin{pmatrix} 0 \\ \vdots \\ 0 \\ 1 \\ 1 \end{pmatrix}$$

$$A = \left(\begin{array}{ccc:c} & -Id_{m^2 \times m^2} & & \begin{matrix} B_1 \\ \vdots \\ B_m \end{matrix} \\ \hdashline Id_{m\times m} & \cdots & Id_{m\times m} & 0 \end{array}\right)$$

On B_i són matrius quadrades de dimensió $m \times m$, amb la columna i formada per 1's i la resta per 0's. D'aquesta manera utilitzant el teorema 3.24, veiem que el seu dual és:

$$\max \left\{ \begin{array}{l} \displaystyle\sum_{j\in\mathcal{D}} v_j : \\ \qquad v_i \geq 0 \quad \forall i \in \mathcal{F} \\ \qquad w_{i,j} \geq 0 \quad \forall i \in \mathcal{F},\, j \in \mathcal{D} \\ \qquad v_j - w_{i,j} \leq c_{i,j} \quad \forall i \in \mathcal{F},\, j \in \mathcal{D} \\ \qquad \displaystyle\sum_{j\in\mathcal{D}} w_{i,j} \leq f_i \quad \forall i \in \mathcal{F} \end{array} \right\} \tag{3.13}$$

Trobem ara les condicions que s'hauran de mantenir al llarg de l'algorisme i el factor d'aproximació que obtindrem. Si suposem (x, y) solució factible òptima del problema primal i (v, w) del dual, fent els càlculs a partir de les equacions de la folga complementària, obtenim:

- Si $y_i = 1$, $\displaystyle\sum_{j\in\{j:\sigma(j)=i\}} w_{i,j} = f_i$
- Si $\sigma(j) = i$, aleshores $\forall i' \neq i, \quad w_{i',j} = 0$ i $v_j - w_{i,j} = c_{i,j}$

Podem interpretar els resultats com que estem atribuint el pagament de les fàbriques a les ciutats. D'aquesta manera, v_j és el preu total pagat per la ciutat $j \in \mathcal{D}$ per a ser servida i $w_{i,j}$ és la contribució de la ciutat j per a obrir la fàbrica $i \in \mathcal{F}$.

Construirem ara l'algorisme, que consta de dues fases. En la primera es troba una solució factible dual i es determina un conjunt de fàbriques temporalment obertes F_t i per tant, una solució primal, i en la segona fase s'escull un subconjunt $I \subseteq F_t$ per obrir definitivament i es troba una assignació $\sigma : \mathcal{D} \to \mathcal{F}$.

S'introdueix una noció temporal t. A $t = 0$, comencem la primera fase inicialitzant totes les variables a zero, i, per cada iteració, farem créixer la mateixa quantitat t i les variables v_j. Poden succeir diversos esdeveniments als quals cal donar resposta:

- Quan $v_j = c_{i,j}$ per algun i no obert temporalment, es començarà a augmentar $w_{i,j}$ de forma uniforme, mantenint d'aquesta manera $v_j - w_{i,j} = c_{i,j}$. L'aresta (i, j) s'anomenarà aresta ajustada.
- Es dirà que la fàbrica i està pagada quan $\sum_{j \in \mathcal{D}} w_{i,j} = f_i$. Quan això succeeixi declararem i temporalment oberta i totes les ciutats $j \in \mathcal{D}$ no connectades prèviament amb $v_j \geq c_{i,j}$ es connectaran a i. Pararem el creixement de v_j i de $w_{i',j}$ per a tota $i' \in \mathcal{F}$.
- Una ciutat $j \in \mathcal{D}$ amb $v_j = c_{i,j}$ on $i \in \mathcal{F}$ és una fàbrica temporalment oberta es declara connectada a i i pararem el creixement de v_j.

Quan diversos esdeveniments succeeixin en una mateixa iteració es processaran en ordre arbitrari. La primera fase acaba quan totes les ciutats estan connectades. Al final d'aquesta fase una ciutat pot haver pagat per obrir temporalment més d'una fàbrica, però volem que només pagui per la que finalment es connectarà. Per això en la segona fase s'escull d'entre el subconjunt de fàbriques temporalment obertes quines obrir definitivament.

Considerem F_t el conjunt de fàbriques obertes temporalment i T el conjunt de les parelles $\{i, i'\}$ de fàbriques temporalment obertes diferents tals que hi ha $j \in \mathcal{D}$ amb $w_{i,j} > 0$ i $w_{i',j} > 0$. Trobem un conjunt maximal estable X (és a dir, un subconjunt de vèrtexs del graf tal que no són adjacents i que, si afegim un altre vèrtex qualsevol, deixa de ser estable) qualsevol del graf (F_t, T). Obrirem definitivament les fàbriques en I i, per a cada ciutat $j \in \mathcal{D}$ que estava connectada a la fàbrica $i \notin X$ el connectarem a algun veí i' de i en el graf (F_t, T), és a dir, fixarem $\sigma(j) = i'$.

Finalment, $I \subseteq \mathcal{F}$ i l'assignació σ defineixen una solució entera primal, que podem definir també com $x_{i,j} = 1$ si i només si $\sigma(j) = i$ i $y_i = 1$ si i només si $i \in I$. Per altra banda, al final de la primera fase també haurem trobat una solució factible dual.

Demostrarem ara que l'algorisme explicat és un 3-algorisme d'aproximació.

Teorema 3.26. *L'algorisme primal-dual és un* 3*-algorisme d'aproximació.*

Demostració. Definirem $c_F(I) = \sum_{i \in \mathcal{I}} f_i$ i $c_s(I) = \sum_{i \in I} \sum_{j \in \mathcal{D}} x_{i,j} c_{i,j} = \sum_{j \in \mathcal{D}} c_{\sigma(j),j}$, per tant el cost de la funció objectiu és $c_F(I) + c_s(I) = \sum_{j \in \mathcal{D}} v_j$. Veurem que $c_F(I) + c_s(I) \leq OPT$.

Calculem primer c_F. Per a tota fàbrica oberta $i \in I$, tots els consumidors $j \in \mathcal{D}$ amb $w_{i,j} > 0$ estan connectats a i, ja que només fem $w_{i,j} > 0$ quan $v_j \geq c_{i,j}$ i en aquest cas en algun moment s'obrirà temporalment la fàbrica i i, si s'acaba obrint definitivament en la segona fase, s'assignarà j a i.

Una fàbrica s'obra temporalment (condició necessària per obrir-se a la segona fase) quan $\sum_{j \in \mathcal{D}} w_{i,j} = f_i$. D'aquesta manera,

$$c_F(I) = \sum_{i \in I} f_i = \sum_{i \in I} \sum_{j \in \mathcal{D}} w_{i,j} = \sum_{j \in \mathcal{D}} w_{\sigma(j),j}$$

Vegem ara que $c_s(I) \leq 3 \sum_{j \in \mathcal{D}} (v_j - w_{\sigma(j),j})$ veient que, per a tota $j \in \mathcal{D}$, $c_{\sigma(j),j} \leq 3(v_j - w_{\sigma(j),j})$. Per casos:

Si $c_{\sigma(j),j} = v_j - w_{\sigma(j),j}$, està clar. Això succeeix quan és j la ciutat que acaba pagant la fàbrica $\sigma(j)$.

En l'altre cas tindrem $c_{\sigma(j),j} > v_j$ i $w_{\sigma(j),j} = 0$. Això succeeix quan no j no estava connectada amb cap fàbrica de I en la primera fase i s'ha assignat finalment a una fàbrica veïna de les que estava connectat. És a dir, existeix una fàbrica tancada i (connectada inicialment amb j) amb $c_{i,j} = v_j - w_{i,j}$ i $j' \in \mathcal{D}$ amb $w_{i,j'} > 0$ i $w_{\sigma(j),j'} > 0$ i per tant, $c_{i,j'} = v_{j'} - w_{i,j'} < v_{j'}$ i $c_{\sigma(j),j'} = v_{j'} - w_{\sigma(j),j'} < v_{j'}$. Com que j' es connecta a $\sigma(j)$ abans que j, podem concloure $c_{\sigma(j),j} \leq c_{\sigma(j),j'} + c_{i,j'} + c_{i,j} \leq 3v_j$.

Així doncs, sumant els dos costos i usant que, com que (v, w) trobades en la primera fase és solució factible dual i per tant, $\sum_{j \in \mathcal{D}} v_j \leq OPT$ obtenim:

$$c_F(I)+c_s(I) \leq 3\sum_{j\in\mathcal{D}} v_j - 3\sum_{j\in\mathcal{D}} w_{\sigma(j),j} + \sum_{j\in\mathcal{D}} w_{\sigma(j),j} = 3\sum_{j\in\mathcal{D}} v_j - 2\sum_{j\in\mathcal{D}} w_{\sigma(j),j} \leq 3\sum_{j\in\mathcal{D}} v_j \leq 3OPT$$

□

Amb el següent exemple podem veure que, a més, no pot existir una cota inferior a la que hem donat. Considerem $\mathcal{D} = \{1, \ldots, n\}$, $\mathcal{F} = \{1, 2\}$, amb costos d'obertura $f_1 = \epsilon$ i $f_2 = \epsilon(n + 1)$ i costos de serveis

$$c_{i,j} = \begin{cases} 1 & \text{si } i = 2 \\ 1 & \text{si } i = j = 1 \\ 3 & \text{altrament} \end{cases}$$

$c_{i,j}$ compleix la desigualtat triangular i és, per tant, una mètrica, condició necessària del problema.

Quan ϵ és prou petit, la solució òptima $\mathcal{O}$ és obrir únicament la fàbrica 2, que té un cost total $c(\mathcal{O}) = \sum_{i \in \mathcal{O}} f_i + \sum_{j \in \mathcal{D}} c_{2,j} = \epsilon(n + 1) + n$, mentre que obrir només la fàbrica 1 té un cost de $c(\mathcal{I}) = f_1 + c_{1,1} + \sum_{j \in \mathcal{D}} c_{1,j} = \epsilon + 1 + 3(n - 1)$ que per ϵ petit té un valor superior i, finalment, si obríssim les dues fàbriques, com que la segona té costos de serveis inferiors o iguals a la primera per a tot j, assignaríem totes les j a 2 i el cost seria $c(\mathcal{O}) + f_1$.

Per altra banda, l'algorisme obriria temporalment la primera fàbrica i assignaria totes les ciutats a ella. Ara, si fem tendir $\epsilon \to 0$, $c(\mathcal{I}) \to 3n - 2$ i $c(\mathcal{O}) \to n$ i per tant, escollint n i ϵ adequadament podem apropar-nos tant com vulguem a tenir $c(\mathcal{I}) = 3c(\mathcal{O})$. Així doncs, no podem trobar una cota al cost de la solució trobada per l'algorisme millor que la del teorema 3.26.

3.3.3 Millora de l'algorisme de Jain i Vazirani

Amb una millora de l'algorisme proposada en l'article (5) aconseguim un 1.861-algorisme d'aproximació. En cada iteració, la modificació de l'algorisme té en compte només les fàbriques no connectades. A més, a diferència de l'algorisme original no té dos fases,

sinó que les unifica en una. Considerem doncs U el conjunt de ciutats no connectades en l'instant actual. Inicialment, a $t = 0$, fixem $U = \mathcal{D}$ i $v_j = 0 \quad \forall j \in \mathcal{D}$. Mentre tinguem $U \neq \emptyset$, incrementem v_j amb la mateixa ràtio que t fins que succeeixi algun d'aquests esdeveniments, als quals cal donar resposta:

- Si $v_j = c_{i,j}$ per $j \in U$ i $i \in \mathcal{F}$ una fàbrica oberta, connectarem j a i i eliminarem j de U.
- Si $\sum_{j \in U} \max_{j \in U}\{0, v_j - c_{i,j}\} = \sum_{j \in U} w_{i,j} = f_i$ per i una fàbrica no oberta, obrirem la fàbrica i, connectarem j a i i eliminarem j de U. Per a tota $j \in U$ amb $v_j \geq c_{i,j}$, connectarem j a i i eliminarem j de U.
- Si $v_j = c_{i,j}$ per $j \in U$ i $i \in \mathcal{F}$ no està oberta, començarem a incrementar $w_{i,j}$ al mateix ritme, per tal de mantenir $v_j - w_{i,j} = c_{i,j}$

Com en l'algorisme anterior, si diferents esdeveniments succeeixen en un mateix instant es resolen en ordre arbitrari. En canvi, en aquest algorisme v no és una solució factible dual, ja que en cada iteració de l'algorisme s'exclou un subconjunt de ciutats i s'elimina la seva aportació a totes les altres fàbriques i, per tant, pot ser que per alguna fàbrica i, $\sum_{j \in \mathcal{D}} \max\{v_j - c_{i,j}, 0\} > f_i$, contradient així una de les condicions de factibilitat. Es pot demostrar però, que existeix $\gamma \approx 1.861$ tal que $\frac{1}{\gamma} v$ sí que és solució factible, obtenint així un γ-algorisme d'aproximació.

3.3.4 Algorisme de Jain, Mahdian i Saberi

Aquest algorisme és força semblant a l'anterior, amb la diferència de què, quan sigui beneficiós, es permet la reassignació de les ciutats. L'algorisme és el següent:

Considerem, com en l'algorisme anterior, U el conjunt de ciutats no connectades en l'instant actual. Comencem inicialitzant $U = \mathcal{D}$ i $t = 0$. Incrementarem t i v_j amb la mateixa ràtio. D'aquesta manera, fins que algun esdeveniment digui el contrari tindrem $t = v_j$. Es tenen en compte els següents successos:

- Si $v_j = c_{i,j}$ per alguna ciutat no connectada, $j \in U$ i una fàbrica i no oberta, començarem a incrementar amb la mateixa ràtio $w_{i,j}$, per tal de mantenir $v_j - w_{i,j} = c_{i,j}$.
- Si $\sum_{j \in \mathcal{D}} w_{i,j} = f_i$ per i no oberta, aleshores obrirem i i aturarem el creixement de v_j per a tota j amb $w_{i,j} > 0$. A més, fixarem $w_{i',j} = \max\{c_{i,j} - c_{i',j}, 0\}$ i connectarem j a i. Per tant, també haurem d'eliminar j de U.
- Si $v_j = c_{i,j}$ per $j \in U$ i i fàbrica oberta, aturarem v_j i fixarem $w_{i',j} = \max\{c_{i,j} - c_{i',j}, 0\}$ per a tota $i' \in \mathcal{F}$. Eliminarem j de U.

El cost total és $c(I) = \sum_{j \in \mathcal{D}} v_j$. El nostre objectiu és trobar γ tal que $\sum_{j \in \mathcal{D}} v_j \leq \gamma c(S)$ per a tota estrella $S = (i, D)$, consistent en el vèrtex i i les ciutats que hi estan connectades. Per fer-ho trobarem diverses propietats que ha de complir una solució factible i construirem un problema d'optimització, anomenat *factor-revealing LP* la solució òptima del qual serà el

factor d'aproximació del nostre algorisme. Estudiem primer algunes propietats derivades de (v, w).

Considerem una estrella qualsevol $S = (i, D)$ amb $|D| = d$, per tant $D = \{1, \cdots, d\}$. Reindexem les ciutats de D de manera que $v_1 \leq v_2 \leq \cdots \leq v_d$, aquest és també l'ordre en què es connecten a l'algorisme, ja que $t = v_j$ fins que j es connecta, que deixa d'augmentar.

Considerem ara $k \in D$. k es connecta en temps v_k. sigui $t = v_k - \epsilon$, per $\epsilon > 0$ petit, és a dir l'instant just abans que es connectés a k. Definim ara la funció:

$$r_{j,k} = \begin{cases} c_{i(j,k),j} & \text{si } j \text{ es connecta a } i(j,k) \in \mathcal{F} \text{ en temps } t \\ v_k & \text{altrament, és a dir si } v_j = v_k \end{cases}$$

Sigui $j \in D$ qualsevol. Com que el cost del servei decreix si els consumidors es reconnecten (altrament no ho farien), tenim:

$$r_{j,j+1} \geq r_{j,j+2} \geq \cdots \geq r_{j,d}$$

per altra banda, podem veure que, si $1 \leq j < k \leq d$,

$$v_k \leq r_{j,k} + c_{i,j} + c_{i,k}$$

Si $r_{j,k} = v_k$, òbviament el resultat es correcte. Altrament $r_{j,k} = c_{i(j,k),j}$ i per tant, aplicant la desigualtat triangular obtenim, $r_{j,k} + c_{i,j} + c_{i,k} = c_{i(j,k),j} + c_{i,j} + c_{i,k} \geq c_{i(j,k),j} + c_{j,k} \geq c_{i(j,k),k} = v_k - w_{i(j,k),k} = v_k$, obtenint el resultat.

Finalment l'última de les propietats que volem és que, per $k \in D = \{1, \cdots, d\}$,

$$\sum_{j=1}^{k-1} \max\{r_{j,k} - c_{i,j}, 0\} + \sum_{l=k}^{d} \max\{v_k - c_{i,l}, 0\} \leq f_i \tag{3.14}$$

Això és degut al fet que la quantitat que j ofereix per obrir la fàbrica k és $\max\{r_{j,k} - c_{i,j}, 0\}$ per $j < i$ i $\max\{t - c_{i,j}, 0\}$ si $j \geq i$, per la construcció de l'algorisme. Aleshores, sumant tots els que ofereixen diners per obrir k, i tenint en compte que el cost ofert no pot ser superior al d'obertura, obtenim la desigualtat.

Ara considerem el problema d'optimització que ens permetrà obtenir la cota de l'algorisme d'aproximació.

$$\max\left\{\begin{array}{l} \dfrac{\sum\limits_{k=1}^{d} v_k}{f_i + \sum\limits_{j=1}^{d} c_{i,j}} : \begin{array}{ll} v_j \leq v_{j+1} & 1 \leq j < d \\ r_{j,k} \geq r_{j,k+1} & 1 \leq j < k < d \\ v_k \leq r_{j,k} + c_{i,j} + c_{i,k} & 1 \leq j < k \leq d \\ \sum\limits_{j=1}^{d} c_{i,j} > 0 & \forall i \in \mathcal{F} \\ \sum\limits_{j=1}^{k-1} \max\{r_{j,k} - c_{i,j}, 0\} + \sum\limits_{l=k}^{d} \max\{v_k - c_{i,l}, 0\} \leq f_i & 1 \leq k \leq d \\ v_j, c_{i,j}, f_i, r_{j,k} \geq 0 & 1 \leq j \leq k \leq d \end{array} \end{array}\right\} \tag{3.15}$$

Malgrat que aquest problema no està escrit com un problema d'optimització lineal, afegint variables es pot transformar fàcilment en un. Anem a veure que la solució òptima del factor-revealing LP és la cota que estem buscant.

Proposició 3.27. *Sigui z_d la solució del problema d'optimització 3.15 i considerem una estrella qualsevol $S = (i, D)$ amb d ciutats. Aleshores, $\sum\limits_{j \in S} v_j \leq z_d c_s$, on c_s és el cost del servei.*

Demostració. El resultat de l'algorisme, (v, w) és també solució factible del problema 3.15, ja que anteriorment hem vist que compleix totes les condicions del problema. Aleshores,

$$\sum_{k=1}^{d} v_k \leq z_d \left(f_i + \sum_{j=1}^{d} c_{i,j} \right)$$

I, si sumen a totes les estrelles obtenim el resultat desitjat. □

Corol·lari 3.28. *Sigui z_d la solució del problema d'optimització 3.15 i $\gamma = \sup\{z_d : d \in \mathbb{N}\}$. Aleshores l'algorisme de Jain Mahdian i Saberi resol el problema de localització mètric no capacitat amb un cost d'aproximació γ.*

Així doncs, tot es redueix a trobar la solució òptima del factor-revealing LP, o bé una cota superior d'aquest vàlida per a tota $d \in \mathbb{N}$:

Proposició 3.29. *Sigui z_d la solució del factor-revealin LP. Aleshores, per a tota $d \in \mathbb{N}$, $z_d \leq 1.61$*

Demostració. Podem assumir que d és suficientment gran, ja que si doblem una solució factible del problema que tractem, és a dir $((v, v), (w, w))$ obtenim que $z_{2d} \geq \dfrac{\sum\limits_{k=1}^{d} 2v_k}{2f_i + \sum\limits_{j=1}^{d} 2c_{i,j}} = z_d$.

Considerem una solució factible de factor-revealing LP i definim $x_{j,k} = \max\{r_{j,k} - c_{i,j}\}$. Aleshores, per la quarta desigualtat del problema, la desigualtat 3.14, tenim que $\forall k < k'$, prenent $d = k' + 1$

$$(k' - k + 1)v_k \leq \sum_{j=k}^{k'} c_{i,j} + f_i - \sum_{j=1}^{k-1} x_{j,k} \tag{3.16}$$

Definim l_i de la següent manera:

$$l_k = \begin{cases} p_2 d & \text{si } k \leq p_1 d \\ d & \text{si } k > p_1 d \end{cases} \tag{3.17}$$

On p_1, p_2 són dos constants que posteriorment fixarem complint $p_1 < p_2$. Ara, si considerem la desigualtat 3.16, per $k < p_2 d$ i $k' = l_k$, dividim ambdós costats entre $(l_k - k + 1)$ i sumem al llarg de k. Obtenim:

$$\sum_{k=1}^{p_2 d} v_k \leq \sum_{k=1}^{p_2 d} \sum_{j=k}^{l_i} \frac{c_{i,j}}{l_k - k + 1} + \sum_{k=1}^{p_2 d} \frac{f_i}{l_k - k + 1} - \sum_{k=1}^{p_2 d} \sum_{j=1}^{k-1} \frac{x_{j,k}}{l_k - k + 1} \tag{3.18}$$

Definim ara $y_j = x_{j,p_2 d}$. Com que $r_{j,i} \geq r_{j,i+1}$ és una de les condicions del problema, $x_{j,i} \geq x_{j,p_2 d} = y_j$ per a tota $j < i < p_2 d$ i $x_{j,i} \leq y_j$ per a tota $i > p_2 d$.

Considerem també $\xi = \sum_{k=1}^{p_2 d} \frac{1}{l_k - k + 1}$. La desigualtat anterior, 3.18 és equivalent a:

$$\sum_{k=1}^{p_2 d} v_k \leq \sum_{k=1}^{p_2 d} \sum_{j=k}^{l_k} \frac{c_{i,j}}{l_k - k + 1} + \xi f_i - \sum_{k=1}^{p_2 d} \sum_{j=1}^{k-1} \frac{c_{i,j}}{l_k - k + 1} \tag{3.19}$$

Sigui $l \leq p_2 d$ l'índex tal que $2c_{i,l} + y_l$ és mínim. Com que $v_k \leq r_{j,k} + c_{i,k} + c_{i,j}$ és una altra desigualtat del problema,

$$v_k \leq r_{l,k} + c_{i,k} + c_{i,l} \leq x_{l,k} + c_{i,k} + 2c_{i,l} \leq y_l + c_{i,k} + 2c_{i,l}$$

Sumant al llarg de $i = p_2 d + 1, \cdots, d$ la desigualtat anterior obtenim,

$$\sum_{k=p_2 d+1}^{d} v_k \leq (y_l + 2c_{i,l})(1 - p_2)d + \sum_{j=p_2 d+1}^{d} c_{i,j} \tag{3.20}$$

Sumant les desigualtats 3.19 i 3.20,

$$\sum_{k=1}^{d} v_k \leq \sum_{k=1}^{p_2 d} \sum_{j=k}^{l_k} \frac{c_{i,j}}{l_k - k + 1} + (2c_{i,l} - y_l)(1 - p_2)df_i + \sum_{j=p_2 d+1}^{d} c_{i,j} - \sum_{k=1}^{p_2 d} \sum_{j=1}^{k-1} \frac{y_j}{l_k - k + 1} + \xi f_i \tag{3.21}$$

Com que

$$\begin{aligned}
\sum_{k=1}^{p_2 d}\sum_{j=k}^{l_k}\frac{c_{i,j}}{l_k-k+1} &= \sum_{k=1}^{p_2 d}\sum_{j=k}^{p_2 d}\frac{c_{i,j}}{l_k-k+1} + \sum_{k=p_1 d+1}^{p_2 d}\sum_{j=k}^{d}\frac{c_{i,j}}{l_k-k+1} = \\
&= \sum_{k=1}^{p_2 d}\sum_{j=k}^{p_2 d}\frac{c_{i,j}}{l_k-k+1} + \sum_{k=p_1 d+1}^{p_2 d}\sum_{j=p_2 d+1}^{d}\frac{c_{i,j}}{l_k-k+1} + \sum_{k=p_1 d+1}^{p_2 d}\sum_{j=k}^{p_2 d}\frac{c_{i,j}}{l_k-k+1} = \\
&= \sum_{k=1}^{p_2 d}\sum_{j=k}^{p_2 d}\frac{c_{i,j}}{l_k-k+1} + \sum_{k=p_1 d+1}^{p_2 d}\sum_{j=p_2 d+1}^{d}\frac{c_{i,j}}{l_k-k+1} = \\
&= -\sum_{k=1}^{p_2 d}\sum_{j=1}^{k-1}\frac{c_{i,j}}{l_k-k+1} + \sum_{k=1}^{p_2 d}\sum_{j=1}^{p_2 d}\frac{c_{i,j}}{l_k-k+1} + \sum_{k=p_1 d+1}^{p_2 d}\sum_{j=p_2 d+1}^{d}\frac{c_{i,j}}{l_k-k+1} = \\
&= -\sum_{j=1}^{p_2 d}\sum_{k=j+1}^{p_2 d}\frac{c_{i,j}}{l_k-k+1} + \sum_{k=1}^{p_2 d}\sum_{j=1}^{p_2 d}\frac{c_{i,j}}{l_k-k+1} + \sum_{k=p_1 d+1}^{p_2 d}\sum_{j=p_2 d+1}^{d}\frac{c_{i,j}}{d-k+1}
\end{aligned} \tag{3.22}$$

La desigualtat 3.21 és equivalent a

$$\begin{aligned}
\sum_{k=1}^{d} v_k \leq{}& (2c_{i,l}+y_l)(1-p_2)df_i + \xi f_i - \sum_{j=1}^{p_2 d}\sum_{k=j+1}^{p_2 d}\frac{c_{i,j}+y_j}{l_k-k+1} + \sum_{j=p_2 d+1}^{d} c_{i,j} + \\
&+ \sum_{j=p_2 d+1}^{d}\sum_{k=p_1 d+1}^{p_2 d}\frac{c_{i,j}}{d-k+1} + \sum_{k=1}^{p_2 d}\sum_{j=1}^{p_2 d}\frac{c_{i,j}}{l_k-k+1} \leq \\
&\leq \xi\sum_{j=1}^{p_2 d} c_{i,j} + \sum_{j=p_2 d+1}^{d}\left(1+\sum_{k=p_1 d+1}^{p_2 d}\frac{1}{d-k+1}\right)c_{i,j} + \xi f_i + \\
&+ (2c_{i,l}+y_l)\left((1-p_2)d - \frac{1}{2}\sum_{j=1}^{p_2 d}\sum_{k=j+1}^{p_2 d}\frac{1}{l_k-k+1}\right)
\end{aligned} \tag{3.23}$$

On la desigualtat és degut al fet que, per la definició de l tenim que $2c_{i,l}+y_l \leq 2c_{i,j}+y_j \leq 2c_{i,j}+2y_j$ quan $j \leq p_2 d$. Sigui ara

$$\xi' = 1 + \sum_{k=p_1 d+1}^{p_2 d}\frac{1}{d-k+1}$$

$$\delta = (1-p_2) - \frac{1}{2d}\sum_{j=1}^{p_2 d}\sum_{k=j+1}^{p_2 d}\frac{1}{l_k-k+1}$$

Per tant, podem reescriure la desigualtat 3.23 com:

$$\sum_{k=1}^{d} v_k \leq \sum_{j=1}^{p_2 d}\xi c_{i,j} + \sum_{j=p_2 d+1}^{d}\xi' c_{i,j} + \xi f_i + \delta(2c_{i,l}+y_l)d \tag{3.24}$$

I podem aproximar ξ, ξ', δ:

$$\begin{aligned}
\xi &= \log\left(\frac{p_2(1-p_1)}{(p_2-p_1)(1-p_2)}\right) + O(1) \\
\xi' &= 1 + \log\left(\frac{1-p_1}{1-p_2}\right) + O(1) \\
\delta &= \frac{1}{2}\left(2 - p_2 - p_2 \log\left(\frac{p_2}{p_2-p_1}\right) - \log\left(\frac{1-p_1}{1-p_2}\right)\right)
\end{aligned}$$

Considerem $\gamma = \max\{\xi, \xi'\}$. Veiem que si trobéssim p_1, p_2 tals que $\delta < 0$, hauríem demostrat que l'algorisme de Jain Mahdia i Saberi és un γ-algorisme d'aproximació:

$$\begin{aligned}
\sum_{k=1}^{d} v_k &\leq \gamma\left(\sum_{j=1}^{p_2 d} c_{i,j} + \sum_{j=p_2 d+1}^{d} c_{i,j} + f_i\right) + \delta(2c_{i,l} + y_l)d = \\
&= \gamma\left(\sum_{j=1}^{d} c_{i,j} + f_i\right) + \delta(2c_{i,l} + y_l)d < \\
&< (\gamma + O(1))\left(f_i + \sum_{k=1}^{d} v_k\right)
\end{aligned} \tag{3.25}$$

Així doncs, només cal resoldre el problema d'optimització $\min\{\max\{\xi, \xi'\} : \delta < 0\}$, per tal d'aconseguir la cota més inferior possible de γ. Resolent l'equació $\xi = \xi'$ veiem que s'assoleix quan $p_1 = \frac{e-1}{e} p_2$. Aleshores la solució que busquem s'obté substituint a l'equació $\delta = 0$ i resolent-la numèricament obtenim que $p_2 \approx 0.695$. Per tant, $p_1 \approx 0.439$. Finalment, doncs, podem concloure que $\gamma < 1.61$. □

També pot ser interessant obtenir una cota d'aproximació diferent pels costos c_F associat a l'obertura de les fàbriques i c_S als costos dels serveis. Per fer-ho considerarem el *factor-revealing LP* següent per $\gamma_F \geq 1$ i $d \in \mathbb{N}$ que, com abans, es podria reformular com un programa lineal:

$$\max\left\{ \frac{\sum_{j=1}^{d} v_j - \gamma_F f_i}{\sum_{j=1}^{d} c_{i,j}} : \begin{array}{l} v_j \leq v_{j+1} \quad 1 \leq j < d \\ r_{j,k} \geq r_{j,k+1} \quad 1 \leq j < k < d \\ v_k \leq r_{j,k} + c_{i,j} + c_{i,k} \quad 1 \leq j < k \leq d \\ \sum_{j=1}^{d} c_{i,j} > 0 \quad \forall i \in \mathcal{F} \\ \sum_{j=1}^{k-1} \max\{r_{j,k} - c_{i,j}, 0\} + \sum_{l=k}^{d} \max\{v_k - c_{i,l}, 0\} \leq f_i \quad 1 \leq k \leq d \\ v_j, c_{i,j}, f_i, r_{j,k} \geq 0 \quad 1 \leq j \leq k \leq d \end{array} \right\} \tag{3.26}$$

Teorema 3.30. *Considerem $\gamma_F \geq 1$ i sigui γ_s el suprem dels valors òptims del programa d'optimització anterior entre els $d \in \mathbb{N}$. Suposem una instància del problema no capacitat de localització de les fàbriques i sigui $X* \subseteq \mathcal{F}$ una solució qualsevol. Aleshores el cost de la solució de l'algorisme en aquesta instància és, com a molt, $\gamma_F c_F(X*) + \gamma_s c_s(x*)$.*

Demostració. L'algorisme retorna nombres v_j que podem suposar sense pèrdua de generalitat amb $v_j \leq v_k$ i per tant, de forma implícita, $r_{j,k}$ per $j, k \in D$. Per a cada estrella (i, D), els nombres $f_i, c_{i,j}, v_j, v_{j,k}$ satisfan com hem vist abans les condicions del programa d'optimització 3.26 a menys que $\sum_{j=1}^{d} c_{i,j} = 0$.

Així doncs, $\frac{\sum_{j=1}^{d} v_j - \gamma_F f_i}{\sum_{j=1}^{d} c_{i,j}} \leq \gamma_s$ ja que aquest és el valor òptim.

Escollim $\sigma* : D \to X$ tal que $c_{\sigma*(j),j} = \min_{i \in X*} c_{i,j}$ i sumant al llarg de totes les parelles $(i, \{j \in D : \sigma*(j) = i\})$ amb $i \in X*$, obtenim:

$$\sum_{j \in D} v_j \leq \gamma_F \sum_{i \in X*} f_i + \gamma_s \sum_{j \in D} c_{\sigma*(j),j} = \gamma_F c_F(X*) + \gamma_s c_s(X*)$$

□

Això ens permet trobar diferents cotes de l'algorisme d'aproximació segons el valor de γ_F, fet que ens resultarà útil posteriorment, quan busquem algorismes per a problemes similars a aquest.

Teorema 3.31. *Considerem el problema d'optimització 3.26 per algun $d \in \mathbb{N}$.*

1. *Per $\gamma_F = 1$, el valor òptim és, com a molt,* 2.
2. *Per $\gamma_F = 1.61$, el valor òptim és, com a molt,* 1.61.
3. *Per $\gamma_F = 1.11$, el valor òptim és, com a molt,* 1.78.

Demostració. El segon cas es correspon a l'òptim, i ja ha estat demostrat. Per altra banda, el tercer no es demostrarà. Veiem el cas en què $\gamma_F = 1$.

$$d\left(f_i+\sum_{j=1}^{d}c_{i,j}\right)\geq\sum_{k=1}^{d}\left(\sum_{j=1}^{k-1}\max\{r_{j,k}-c_{i,j},0\}+\sum_{j=k}^{d}\max\{v_k-c_{i,j},0\}\right)\geq$$

$$\geq\sum_{k=1}^{d}\left(\sum_{j=1}^{k-1}r_{j,k}+\sum_{j=k}^{d}v_k-\sum_{j=1}^{k-1}c_{i,j}-\sum_{l=k}^{d}c_{i,l}+\sum_{j=1}^{d}c_{i,j}\right)=$$

$$=\sum_{k=1}^{d}\left(\sum_{j=1}^{k-1}r_{j,k}+\sum_{l=k}^{d}v_k\right)\geq\sum_{k=1}^{d}\left(\sum_{j=1}^{k-1}v_k-c_{i,j}-c_{i,k}+\sum_{l=k}^{d}v_k\right)=$$

$$=\sum_{k=1}^{d}\left(\sum_{j=1}^{d}v_k-\sum_{j=1}^{k-1}(c_{i,j}+c_{i,k})\right)=$$

$$=\sum_{k=1}^{d}dv_k-\sum_{j=1}^{d}(d-j)c_{i,j}-\sum_{k=1}^{d}(k-1)c_{i,k}=$$

$$=\sum_{k=1}^{d}dv_k-\sum_{j=1}^{d}(d-1)c_{i,j}$$

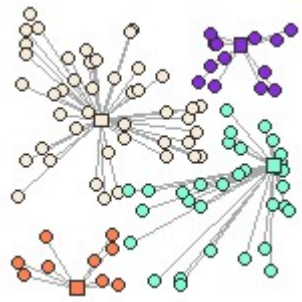
Per tant, $d\sum_{j=1}^{d}v_j\leq df_i+\sum_{j=1}^{d}c_{i,j}(2d-1)\leq df_i+2d\sum_{j=1}^{d}c_{i,j}$, fet que implica que el cost de la funció òptima és $\gamma_s\leq\frac{\sum_{k=1}^{d}v_k-f_i}{\sum_{j=1}^{d}c_{i,j}}\leq 2$ □

3.3.5 Implementació i comparativa

En aquesta secció es fa un exemple a l'atzar i es compara els diversos algorisme primals-duals estudiats. Mitjançant el programa en C *"generador.c"* es creen 100 ciutats i 40 fàbriques semi a l'atzar (per tal de poder replicar l'experiment), es calculen les distàncies entre elles i s'assignen també a l'atzar els costos d'obertura de cada una de les fàbriques. Es programen aleshores els tres programes primals-duals: el de Jain i Vazirani a *"Jain-Vazirani.c"*, la seva millora a *"JVMillora.c"* i el de Jain, Mahdian i Saberi a *"JMS.c"*. Els quatre problemes es troben a l'annex.

En cada programa s'inicialitza totes les components de v a $\min\{c_{i,j}:i\in\mathcal{F},\,j\in\mathcal{D}\}$, ja que si $v<c_{i,j}\ \forall i\in\mathcal{F},\,j\in\mathcal{D}$ no es pot produir cap esdeveniment i en cada iteració s'augmenten les components que cal augmentar (en un inici totes elles) un 0.01, que és la tolerància que hem donat a c. Es consulta després si es compleix algun dels esdeveniments i es segueixen simplement les indicacions de cada apartat. En la segona fase l'algorisme de Jain Vazirini, consistent en trobar un conjunt maximal estable, s'ha escollit la primera de les fàbriques oberta per obrir definitivament, després la següent que no tenia una ciutat en comú, i segueix iterativament fins a trobar el conjunt maximal.

S'han tractat després les dades del resultat obtingut amb R per tal de crear les imatges següents i poder comparar els diversos mètodes. El codi utilitzat en R també està en els annexos, amb el nom *"grafics.R"*.



(a) Jain, Vazirani

(b) Primera millora

(c) Jain, Mahdian, Saberi

En cada imatge es mostren les fàbriques obertes en forma quadrada i les ciutats circulars. No es mostren totes les fàbriques, sinó únicament les que s'obren. Per clarificar les imatges, s'han pintat totes les ciutats del mateix color que la fàbrica a la qual està enllaçada. Amb les imatges es poden veure les diferències entre cada procés. En el primer, es creen grans clústers i s'obra una fàbrica en cada un d'ells. Cal tenir en compte que el resultat final de l'algorisme depèn significativament del conjunt estable maximal escollit. Així, el segon i tercer algorisme són els que, com calia esperar, tenen resultats més similars. S'obren gairebé les mateixes fàbriques, però en canvi algunes d'elles no tenen assignats els mateixos consumidors, ja que es permet canviar de fàbrica quan això redueixi els costos de servei d'una ciutat.

Els costos de la funció objectiu són 2396.68, 1119.89 i 1112.62 respectivament. Per altra banda, dividint cada resultat entre el factor d'aproximació corresponent podem veure que el cost òptim és, com a mínim, 601.76 i per tant, en el pitjor cas possible la diferència entre els resultats obtinguts i l'òptim és significativa.

3.4 Cota a la ràtio d'aproximació i augmentació voraç

Al llarg d'aquest capítol s'han explicat diversos algorismes d'aproximació intentant sempre reduir al màxim el valor del factor d'aproximació. En aquesta secció es dona la millor cota possible al factor d'aproximació en el cas que els costos de servei estiguin entre 1 i 3 mitjançant l'augmentació voraç, un algorisme que, donada una solució en troba de noves amb menor cost de forma iterativa.

Donada una solució, $X \subseteq \mathcal{F}$, l'augmentació voraç considera

$$g_X(i) = c_s(X) - c_s(X \cup \{i\}) \tag{3.27}$$

i escull iterativament elements de $i \in \mathcal{F}$ que maximitzin $\frac{g_X(i)}{f_i}$ fins que $\forall i \in \mathcal{F}\ g_X(i) \leq f_i$. Trobarem ara dos resultats sobre l'augmentació voraç que s'utilitzaran posteriorment per trobar la cota del factor d'aproximació.

Proposició 3.32. *Sigui $X \neq \emptyset$ i $X^* \subseteq \mathcal{F}$, aleshores* $\sum_{i \in X^*} g_X(i) \geq c_s(X) - c_s(X^*)$

Demostració. Considerem $j \in \mathcal{D}$ i definim $\sigma(j) \in X$ tal que $c_{\sigma(j),j} = \min\{c_{i,j} : i \in X\}$ i $\sigma^*(j) \in X^*$ tal que $c_{\sigma^*(j),j} = \min\{c_{i,j} : i \in X^*\}$.

Ara, per a tot $i \in X^*$, $g_X(i) = c_s(X) - c_s(X \cup \{i\}) = \sum_{j \in \mathcal{D}, \sigma^*(j)=i} (c_{\sigma(j)j} - c_{i,j})$. Sumant obtenim el resultat. □

Aquest lema ens indica que en cada cas existeix $i \in X^*$ que maximitzi $\frac{g_X(i)}{f_i}$, justificant d'aquesta manera l'algorisme. Trobem primer una cota inferior al factor:

Proposició 3.33. *Considerem $X, X^* \neq \emptyset$, $X, X^* \subseteq \mathcal{F}$ i $Y \subseteq X$ resultat de l'algorisme d'augmentació voraç. Aleshores,*

$$c_F(Y) + c_s(Y) \leq c_F(X) + c_F(X^*) \log\left(\max\left\{1, \frac{c_s(X) - c_s(X^*)}{c_F(X^*)}\right\}\right) + c_F(X^*) + c_s(X^*)$$

Demostració. Si $c_s(X) \leq c_F(X^*) + c_s(X^*)$, aleshores $Y = X$ i per tant la desigualtat es manté. Altrament, suposem que $c_s(X) > c_F(X^*) + C_s(X^*)$ i definim $X_0, \ldots, X_k$ la successió de conjunts obtinguts amb l'augmentació voraç fins a la iteració k, la primera tal que $c_s(X_k) \leq c_F(X^*) + c_s(X^*)$. Reindexem els índexs de $\mathcal{F}$ de manera que en cada iteració afegim $\{i\}$, és a dir, $X_i \setminus X_{i-1} = \{i\}$ per $i = 1, \ldots, k$.

Per la proposició anterior, $\frac{c_s(X_{i-1}) - c_s(X_i)}{f_i} \geq \frac{c_s(X_{i-1}) - c_s(X^*)}{c_F(X^*)}$ per $i = 1, \cdots, k$. Així doncs,

$$f_i \leq c_F(X^*) \frac{c_s(X_{i-1}) - c_s(X_i}{c_s(X_{i-1}) - c_s(X^*)}$$

Ara,

$$c_F(X_k) + c_s(X_k) \leq c_s(X_k) + c_F(X^*) \sum_{i=1}^{k} \frac{c_s(X_{i-1}) - c_s(X_i)}{c_s(X_{i-1}) - c_s(X^*)} + c_F(X) \tag{3.28}$$

Observem la part esquerra de la desigualtat. Aquesta augmenta quan augmentem $c_s(X_k)$, ja que la seva derivada és $1 - \frac{c_s(X_k)}{c_s(X_{k-1}) - c_s(X^*)} = \frac{c_s(X_{k-1}) - c_x(X^*) - c_F(X^*)}{c_s(X_{k-1}) - c_s(X^*)} > 0$, ja que hem assumit que $c_s(X_i) > c_s(X^*) + c_F(X^*)$ i $c_s(X_{i-1}) > c_s(X^*)$. Podem assumir que és el màxim, és a dir, que $c_s(X_k) = c_F(X^*) + c_s(X^*)$. Utilitzant aquest fet i que $x - 1 \geq ln(X)\ \forall x > 0$,

$$\begin{aligned}
c_F(X_k) + c_s(X_k) &\leq c_s(X_k) + c_F(X^*) \sum_{i=1}^{k} \frac{c_s(X_{i-1}) - c_s(X_i)}{c_s(X_{i-1}) - c_s(X^*)} = \\
&= c_s(X_k) + c_F(X) + c_F(X^*) \sum_{i=1}^{k} \left(1 - \frac{c_s(X_i) - c_s(X^*)}{c_s(X_{i-1}) - c_s(X^*)}\right) \leq \\
&\leq c_s(X_k) + c_F(X) - c_F(X^*) \sum_{i=1}^{k} \log\left(\frac{c_s(X_k) - c_s(X^*)}{c_s(X) - c_s(X^*)}\right) \leq \\
&\leq c_s(X^*) + c_F(X) - c_F(X^*) \sum_{i=1}^{k} \log\left(\frac{c_F(X^*)}{c_s(X_k) - c_s(X^*)}\right) + c_F(X^*) = \\
&= c_F(X) + c_F(X^*) \log\left(\frac{c_s(X) - c_s(X^*)}{c_F(X^*)}\right) + c_F(X^*) + c_s(X^*)
\end{aligned}$$

□

Aquest algorisme es pot utilitzar juntament amb altres algorismes per obtenir un factor d'aproximació millor. Per exemple. l'algorisme consistent en escalar tots els costos d'obertura de les fàbriques, multiplicant-los per 1.504, aplicar l'algorisme de Jain Mahdian i Saberi, tornar al cost original els costos d'obertura i aplicar l'augmentació voraç té un factor d'aproximació de 1.52.

Estudiem ara el millor factor d'aproximació. Considerem α la solució de l'equació $\alpha + 1 = \log\left(\frac{2}{\alpha}\right)$. Per tant, $\alpha = \frac{\alpha}{\alpha+1}\log\left(\frac{2}{\alpha}\right) = \max\left\{\frac{\xi}{\xi+1}\log\left(\frac{2}{\xi}\right) : \xi > 0\right\}$,, ja que la derivada de $\frac{\xi}{\xi+1}\log\left(\frac{2}{\xi}\right)$ és $\frac{1}{(\xi+1)^2}\log\left(\frac{2}{\xi}\right) - \frac{1}{\xi+1}$, que s'anul·la en un màxim quan $\frac{1}{\xi+1}\log\left(\frac{2}{\xi}\right) = 1$. Es pot veure amb mètodes numèrics que $\alpha \approx 0.4631$.

Veurem primer que $1+\alpha$ és un factor d'aproximació en el cas que els costos de servei estan entre $[1,3]$ i posteriorment que no existeix cap cota millor.

Proposició 3.34. *El problema de localització mètric no capacitat amb $c_{i,j} \in [1,3]$ $\forall i \in \mathcal{F}, \quad j \in \mathcal{D}$ té un $(1+\alpha+\epsilon)$-algorisme d'aproximació per a tot $\epsilon > 0$*

Demostració. Fixem $\epsilon > 0$ i $k = \lceil \frac{1}{\epsilon} \rceil$. Considerem totes les solucions $X \subseteq \mathcal{F}$ de fàbriques que obrim amb $|X| \leq k$.

Calculem una nova solució Y obrint la fàbrica i que tingui cost d'obertura f_i mínim i aplicant l'augmentació voraç. Veurem que la millor solució obtinguda amb aquest mètode costa, com a molt, $(1+\alpha+\epsilon)$ cops el cost òptim.

Sigui $zeta = \frac{c_F(X^*)}{c_s(X^*)}$ i X^* una solució òptima. Si $|X^*| \leq k$, ja hauríem trobat X^*, per tant podem suposar que $|X^*| > k$. Aleshores, $c_F(\{i\}) \leq \frac{1}{k}c_F(X^*)$ i, com que els costos de servei estan entre 0 i 3, tenim que $c_s(i) \leq 3|\mathcal{D}| \leq 3c_s(X^*)$.

Utilitzant la proposició 3.33, el cost d'Y és, com a molt,

$$
\begin{aligned}
c_F(Y) + c_s(Y) &\leq c_F(X) + c_F(X^*)\log\left(\max\left\{1, \frac{c_s(X) - c_s(X^*)}{c_F(X^*)}\right\}\right) + c_F(X^*) + c_s(X^*) \leq \\
&\leq \frac{1}{k}c_F(X^*) + C_F(X^*)\log\left(\max\left\{1, \frac{2c_s(X^*)}{c_F(X^*)}\right\}\right) + c_F(X^*) + c_s(X^*) = \\
&= c_s(X^*)\left(\frac{\xi}{k} + \xi\log\left(\max\left\{1, \frac{2}{\xi}\right\}\right) + \xi + 1\right) \leq \\
&\leq c_s(X^*)\left(\frac{\xi}{k} + (1+\xi)\alpha + \xi + 1\right) = c_s(X^*)\left((1+\xi)(1+\alpha) + \frac{\xi}{k}\right) \leq \\
&\leq c_s(X^*)(1+\alpha+\epsilon)(1+\xi) = (1+\alpha+\epsilon)(c_s(X^*) + c_F(X^*))
\end{aligned}
$$

□

Per altra banda, per veure que $(1+\alpha)$ és el millor factor d'aproximació possible considerem el contrari, és a dir, que existeix un $(1+\alpha-\epsilon)$-algorisme d'aproximació per algun $\epsilon > 0$ i es planteja un problema que es demostra que és NP-complex. Per altra banda, però, es pot construir un algorisme polinòmic que resol el problema i que utilitza com a subrutina l'algorisme d'aproximació que hem suposat que existeix. Per tant, si existeix un algorisme d'aproximació amb aquest factor d'aproximació tenim $P = NP$.

3.5 Variacions del problema

En aquest capítol es treballa el problema de localització no capacitat i s'han elaborat algorismes per tal d'aproximar la solució. Existeixen, però, variacions del problema que es poden resoldre amb tècniques similars a les estudiades.

3.5.1 k-problema de localització

Aquest problema és una modificació del problema no capacitat, afegint la restricció de què no es pot escollir més de k fàbriques. Com en el problema no capacitat, podem considerar també el cas mètric. El següent teorema implica l'existència d'un 4-algorisme d'aproximació pel cas mètric.

Teorema 3.35. *Suposem que existeix $\gamma_s \in \mathbb{R}$ i un algorisme polinòmic A tal que per a tota entrada del problema mètric no capacitat troba una solució tal que $c_F(X) + c_s(X) \leq c_F(X^*) + \gamma_s c_s(X^*)$. Aleshores existeix un $2\gamma_s$-algorisme d'aproximació del k-problema de localització mètric*

En particular, pel teorema 3.31 podem concloure que utilitzant l'algorisme de Jain Mahdian i Saberi s'obté un 4-algorisme d'aproximació.

3.5.2 k-problema de mediana

Si considerem el k-problema de localització mètric amb costos d'obertura nuls, estarem minimitzant la mediana, i tindrem el k-problema de mediana. Aquest s'acostuma a resoldre mitjançant la *cerca local*, consistent en escollir inicialment una solució factible qualsevol, i en cada iteració millorar-la mitjançant intercanvis. D'aquesta manera es pot obtenir un algorisme aproximatiu de factor 5.

3.5.3 Problema capacitat dèbil

En aquesta variació del problema no capacitat s'introdueix la capacitat de cada fàbrica. Així, per cada fàbrica $i \in \mathcal{F}$ tindrem una capacitat u_i. Per minimitzar els costos podem tractar el problema com si fos el no capacitat i obrir $\left\lceil \frac{|D|}{u_i} \right\rceil$ còpies de la fàbrica i, on $D \subseteq \mathcal{D}$ són les ciutats que estan assignades a i. Aleshores es pot demostrar que amb l'algorisme de Jain Madian Saberi podem obtenir un 2.89-algorisme d'aproximació:

Teorema 3.36. *Suposem que tenim un algorisme aproximatiu A tal que $c_F(X)+c_s(X) \leq \gamma_F C_F(X^*)+\gamma_s c_s(X^*)$. Aleshores tenim un $(\gamma_F+\gamma_s)$-algorisme d'aproximació pel problema capacitat dèbil.*

Demostració. Suposem $I = (\mathcal{F}, \mathcal{D}, f, u, c)$ una entrada del problema capacitat dèbil i considerem $I' = (\mathcal{F}, \mathcal{D}, f, c')$ on $c'_{i,j} = c_{i,j} + \frac{f_i}{u_i}$ per a tot $i \in \mathcal{F}$, $j \in \mathcal{D}$. Apliquem l'algorisme A a l'entrada I' i trobem una solució $X \subseteq \mathcal{F}$ i $\sigma : \mathcal{D} \to X$. Sigui X^* i σ^* la solució òptima. Aleshores es compleix:

$$
\begin{aligned}
c_F(X) + c_s(X) &= \sum_{i \in X} \left\lceil \frac{|\{j \in \mathcal{D} : \sigma(j) = i\}|}{u_i} \right\rceil f_i + \sum_{j \in \mathcal{D}} c_{\sigma(j),j} \leq \\
&\leq \sum_{i \in X} f_i + \sum_{j \in \mathcal{D}} c'_{\sigma(j),j} \leq \gamma_F \sum_{i^*_X} f_i + \gamma_s \sum_{j \in \mathcal{D}} c'_{\sigma(j),j} \leq \\
&\leq (\gamma_F + \gamma_s) \sum_{i \in X^*} \left\lceil \frac{|\{j \in \mathcal{D} : \sigma^*(j) = i\}|}{u_i} \right\rceil f_i + \sigma_s \sum_{j \in \mathcal{D}} c_{\sigma^*(j),j} \leq \\
&\leq (\gamma_F + \gamma_s)(c_F(X^*) + c_s(X^*))
\end{aligned}
$$

□

Així doncs, utilitzant el teorema 3.31, trobem un 2.89-algorisme d'aproximació.

3.5.4 Problema de localització capacitat

Aquest problema es pot considerar una ampliació de l'anterior, on també es té en compte la demanda de cada ciutat, d_j. El millor algorisme aproximatiu que s'ha trobat per a aquests problemes és la cerca local, però n'existeixen altres, com per exemple una modificació de l'algorisme de Shmoys, Tardos i Aardal.

Bibliografia

[1] Aardal, K., Shmoys, D. D., and Tardos, E. Approximation algorithms for facility location problems (extended abstract). pp. 265–274.

[2] Bernal, R. J. C., Tornel, M. B. C., and Gómez, J. M. R. El algoritmo de weiszfeld para la resolución del problema económico de weber.

[3] Bertsimas, D., and Tsitsiklis, J. N. *Introduction to Linear Optimization*. Athena Scientific, 1197.

[4] Drezner, Z., and Hamacher, H. W. *Facility Location: Applications and Theory*. Springer, 2002.

[5] Jain, K., Mahdian, M., Markakis, E., Saberi, A., and Vazirani, V. Greedy facility location algorithms analyzed using dual fitting with factor-revealing lp. *Journal of the ACM 50* (10 2003).

[6] Jain, K., and Vazirani, V. Approximation algorithms for metric facility location and. *J. ACM 48* (2001), 274–296.

[7] Korte, B., and Vygen, J. *Combinatorial Optimization Algorithms: Theory and Algorithms*, 5a ed. Springer, 2012.

[8] Love, R. F., and Yeong, W. Y. A stopping rule for facilities location algorithms. *A I I E Transactions 13*, 4 (1981), 357–362.

[9] Luenberger, D. G., and Ye, Y. *Linear and Nonlinear Programming*, 3rd ed. Springer, 2008.

[10] Vygen, J. *Approximation Algorithms For Facility Location Problem*. Lecture Notes at University of Bonn, 2005.