- **AVVERTENZA**: Di seguito trovate alcuni appunti, poco ordinati e poco formali, che uso come traccia durante le lezioni. **Non sono assolutamente da considerarsi sostitutivi del materiale didattico.**
- **Riferimenti**: Algorithmic Game Theory, Nisan et al. Pagine 385 - 391. Per la Sezione 2.1 facciamo riferimento a: https://www.cs.cmu.edu/ anupamg/adv-approx/lecture5.pdf , una dispensa di Anupam Gupta (le uniche parti che non abbiamo trattato sono la Sezione 1.2, il Paragrafo "Another randomized rounding twist" e il Paragrafo "Choosing maximal independent sets").

1 Facility Location Game

Consideriamo il problema del *facility location* in cui un insieme N di clienti deve accedere ad un certo servizio. Il servizio può essere erogato in diverse facility e F è l'insieme di tutte le facility. L'erogazione del servizio ha i seguenti costi: ciascun cliente $j \in N$ può servirsi presso un centro $i \in F$, che sia *aperto*, con un costo pari a c_{ij}, mentre per ogni centro $i \in F$ il costo di apertura è pari a f_i.

Il problema del facility location può essere risolto attraverso il seguente problema di PLI:

$$\begin{aligned} \min \textstyle\sum_{i\in F} f_i y_i + \sum_{i\in F}\sum_{j\in N} d_{ij}x_{ij} & \\ \textstyle\sum_{i\in F} x_{ij} \geq 1,\ j \in N & \\ x_{ij} \leq y_i,\ i \in F, j \in N & \qquad (P) \\ x_{ij}, y_i \in \{0,1\}, i \in F, j \in N & \end{aligned}$$

(Osserviamo come in una soluzione ottima il vincolo $\sum_{i\in F} x_{ij} \geq 1$ sarà soddisfatto all'uguaglianza. D'altro canto, scrivere il vincolo con il $\geq$ ci premette di associare al vincolo una variabile duale vincolata in segno, cosa che rende alcune delle prossime considerazioni più semplici).

Supponiamo di aver risolto in modo esatto il precedente problema, ovvero di aver trovato una soluzione ottima per il precedente PLI (torneremo poi su come fare). Siamo ora interessati a come *ripartire* tra gli utenti di N il costo della soluzione ottima (ovvero il costo di servire la grande coalizione). Per questo formuliamo il seguente gioco cooperativo (in forma di costo!):

- i giocatori sono i clienti;
- per ciascuna coalizione $S \subseteq N$ il valore $c(S)$ della coalizione è pari a quanto costerebbe attivare il servizio per i soli utenti di S, ovvero è pari al valore della soluzione ottima del seguente problema di PLI:

$$\begin{aligned} c(S) = \min \textstyle\sum_{i\in F} f_i y_i + \sum_{i\in F}\sum_{j\in S} d_{ij}x_{ij} & \\ \textstyle\sum_{i\in F} x_{ij} \geq 1,\ j \in S & \end{aligned}$$

$$x_{ij} \leq y_i,\ i \in F, j \in S \qquad (P_S)$$
$$x_{ij}, y_i \in \{0,1\}, i \in F, j \in S;$$

- Al solito, siamo interessati a trovare una ripartizione del costo $c(N)$ della grande coalizione tra i giocatori che sia stabile rispetto tutte le coalizioni, ovvero a trovare un vettore nel nucleo del gioco. Ma prima di fare questo, dobbiamo verificare che la nostra funzione di costo sia *sub-additiva* (siamo in forma di costo, per cui la super-additività è rimpiazzata dalla sub-additività!). Nel seguito sfruttiamo la seguente osservazione: fissato $S \subseteq N$ il problema si riduce a *quali centri aprire* per servire S: una volta scelto l'insieme dei centri da aprire, ciascun utente $j \in N$ si rivolgerà al centro, tra quelli aperti, per cui è minimo il costo di servizio d_{ij}.

  Notiamo che la funzione v così definita è sub-additiva. Infatti, prese due coalizioni disgiunte S e T, siano rispettivamente $F(S)$ e $F(T)$ i centri aperti nelle soluzioni ottime del precedente programma risolto rispetto S e T. È immediato vedere che servendo ciascun utente j della coalizione $S \cup T$ presso il centro $i \in F(S) \cup F(T)$ che minimizza d_{ij}, sarebbe possibile attivare il servizio per la coalizione $S \cup T$ con un costo $\bar{c} \leq c(S) + c(T)$. Naturalmente il costo ottimo $c(S \cup T)$ di attivazione del servizio per la coalizione $S \cup T$ – che possiamo ottenere risolvendo il precedente programma rispetto $S \cup T$ – è tale che $c(S \cup T) \leq \bar{c} \leq c(S) + c(T)$. Per cui $c(S \cup T) \leq c(S) + c(T)$.

- La funzione di costo è sub-additiva. La domanda successiva è se il nucleo del facility location game è sempre non vuoto, come capita per esempio per il modello del mercato a utilità trasferibile.

- **Esempio 1** Consideriamo il caso in cui $N = \{A, B, C\}$ e $F = \{1, 2\}$ con $f_1 = f_2 = 2$, $d_{A,1} = 2$, $d_{A,2} = 5$, $d_{B,1} = 2$, $d_{B,2} = 1$, $d_{C,1} = 4$ e $d_{C,2} = 1$. Possiamo calcolare $c(\emptyset) = 0$; $c(\{A\}) = 4; c(\{B\}) = 3; c(\{C\}) = 3; c(\{A, B\}) = 6; c(\{A, C\}) = 7; c(\{B, C\}) = 4; c(N) = 8$. È immediato verificare che la allocazione $\alpha = (4, 2, 2)$ è nel nucleo del gioco che è quindi non vuoto.

- **Esempio 2** Consideriamo ora che una nuova facility sia disponibile, per cui $F = \{1, 2, 3\}$, con $f_3 = 3$, $d_{A,3} = 1$, $d_{B,3} = 5$, $d_{C,3} = 1$, mentre gli altri costi rimangono invariati. Possiamo calcolare ora $c(\emptyset) = 0$; $c(\{A\}) = 4; c(\{B\}) = 3; c(\{C\}) = 3; c(\{A, B\}) = 6; c(\{A, C\}) = 5; c(\{B, C\}) = 4; c(N) = 8$. È immediato verificare che il nucleo è ora vuoto poiché una eventuale allocazione nel nucleo dovrebbe soddisfare $\alpha_a + \alpha_b \leq 6$; $\alpha_a + \alpha_c \leq 5$; $\alpha_b + \alpha_c \leq 4$; quindi sommando le tre disequazioni $2(\alpha_a + \alpha_b + \alpha_c) \leq 15$, che è in contraddizione con $\alpha_a + \alpha_b + \alpha_c = c(N) = 8$.

- Segue quindi che il nucleo del facility location game potrebbe essere vuoto. Abbiamo visto come il valore di Shapley fornisce un modo ragionevole di ripartire tra gli utenti di N il costo della grande coalizione anche quando il nucleo è vuoto. Per il problema del facility location, calcolare il valore di Shapley ha un grosso svantaggio, quello di richiedere la risoluzione di un problema di PLI per ogni coalizione, che dal punto di

vista computazionale è molto oneroso. Nel seguito introduciamo quindi un concetto diverso, quello di nucleo γ-approssimato, che oltre ad essere molto interessante di suo, ha anche il vantaggio di "aggirare" le difficoltà computazionali.

2 Nucleo γ-approssimato

- Un vettore $\alpha \in \mathcal{R}_+^N$ è nel *nucleo* γ-approssimato (spesso chiamato γ-nucleo) di un gioco cooperativo (N, c) se soddisfa le seguenti proprietà :
  - (i) $\sum_{j \in S} \alpha_j \leq c(S)$, per ogni $S \subseteq N$;
  - (ii) $\sum_{j \in N} \alpha_j \geq \gamma \cdot c(N)$.

  È immediato verificare che il vettore $\alpha = (3.5, 2.5, 1.5)$ è nel $\frac{15}{16}$-core per il gioco dell'Esempio 2.

- Si osservi che dalla definizione segue che il γ-nucleo è sempre vuoto per $\gamma > 1$ e che il nucleo coincide con l'1-nucleo. Naturalmente nei casi in cui il nucleo è vuoto siamo interessati a capire qual è il più grande valore di γ per cui il γ-nucleo è non vuoto: indichiamo tale valore con γ^*. Se consideriamo nuovamente l'Esempio 2, dalle considerazioni fatte in precedenza, segue che che in quel caso $\gamma^* = \frac{15}{16}$. Banalmente per l'Esempio 1 abbiamo invece $\gamma^* = 1$.

- Si noti che γ^* si può individuare risolvendo il seguente problema di programmazione lineare:

  $$\gamma^* = \max \tfrac{1}{c(N)} \textstyle\sum_{j \in N} \alpha_j$$
  $$\textstyle\sum_{j \in S} \alpha_j \leq c(S), \text{ per ogni } S \subseteq N.$$

  ma ovviamente per risolvere questo programma dovremmo innanzitutto calcolare $c(S)$ per ogni coalizione $S \subseteq N$, che era lo stesso problema che avevamo per calcolare il valore di Shapley. Seguiremo quindi una (sorprendente) strada diversa, che passa attraverso la formulazione di PLI del problema di facility location analizzata in precedenza.

- Ricordiamo che il *rilassamento lineare* del problema (P) è il seguente problema di programmazione lineare P_R, in cui appunto rilassiamo il vincolo che le variabili x e y siano binarie e chiediamo semplicemente che siano non negative:

  $$\begin{aligned} z(P_r) = \min & \textstyle\sum_{i \in F} f_i y_i + \sum_{i \in F} \sum_{j \in N} d_{ij} x_{ij} \\ & \textstyle\sum_{i \in F} x_{ij} \geq 1,\ j \in N \\ & x_{ij} \leq y_i,\ i \in F, j \in N \qquad\qquad (P_R) \\ & x_{ij}, y_i \geq 0, i \in F, j \in N. \end{aligned}$$

  Naturalmente, vale $z(P_R) \leq z(P)$.

- Consideriamo inoltre il duale D del problema P_R:

$$
\begin{aligned}
w(D) = \max & \textstyle\sum_{j\in N} \alpha_j \\
& \textstyle\sum_{j\in N} \beta_{ij} \le f_i,\ i \in F; \qquad\qquad (D)\\
& \alpha_j \le \beta_{ij} + d_{ij},\ i \in F, j \in N \\
& \alpha_j, \beta_{ij} \ge 0, i \in F, j \in N,
\end{aligned}
$$

- Quindi valgono le seguenti relazioni: $w(D) = z(P_R) \le z(P)$.

- Consideriamo l'Esempio 1. In questo caso, $w(D) = z(P_R) = z(P)$. Infatti la soluzione $(\overline{x}, \overline{y})$ ottima per P, di valore 8, è la seguente: $\overline{x}_{1A} = \overline{x}_{2B} = \overline{x}_{2C} = \overline{y}_1 = \overline{y}_2 = 1$; tutte le altre componenti di x a valore 0. Questa soluzione è ottima anche per P_R: questo lo certifichiamo esibendo una soluzione $(\overline{\alpha}, \overline{\beta})$ di D che ha lo stesso valore 8 e quindi è ottima: infatti, dovendo valere $w(\overline{\alpha}, \overline{\beta}) \le w(D) = z(P_R) \le z(P) \le z(\overline{x}, \overline{y})$, se $w(\overline{\alpha}, \overline{\beta}) = z(\overline{x}, \overline{y})$ allora vale anche $w(\overline{\alpha}, \overline{\beta}) = w(D) = z(P_R) = z(P) = z(\overline{x}, \overline{y})$. La soluzione $(\overline{\alpha}, \overline{\beta})$ è la seguente: $\overline{\alpha}_A = 4; \overline{\alpha}_B = 2; \overline{\alpha}_C = 2; \overline{\beta}_{1A} = 2, \overline{\beta}_{1B} = 0, \overline{\beta}_{1C} = 0, \overline{\beta}_{2A} = 0, \overline{\beta}_{2B} = 1, \overline{\beta}_{2C} = 1$ è ammissibile per D e ha valore 8.

- Consideriamo ora l'Esempio 2. In questo caso, $w(D) = z(P_R) < z(P)$. La soluzione $(\overline{x}, \overline{y})$ vista per il primo esempio rimane ottima e conserva valore 8 (si noti che tuttavia esistono altre soluzioni ottime, naturalmente di valore ancora 8). Tuttavia la soluzione non è più ottima per P_R perché la seguente soluzione è ammissibile per P_R e ha valore $\frac{15}{2}$: $\tilde{x}_{1A} = \tilde{x}_{3A} = \tilde{x}_{1B} = \tilde{x}_{2B} = \tilde{x}_{2C} = \tilde{x}_{3C} = \frac{1}{2}, \tilde{y}_1 = \tilde{y}_2 = \tilde{y}_3 = \frac{1}{2}$; tutte le altre componenti di x sono a valore 0. In realtà $(\tilde{x}, \tilde{y})$ è ottima per P_R, al solito possiamo certificarlo esibendo una soluzione ottima per D che ha lo stesso valore $\frac{15}{2}$: $\tilde{\alpha}_A = 3.5; \tilde{\alpha}_B = 2.5; \tilde{\alpha}_C = 1.5; \tilde{\beta}_{1A} = 1.5, \tilde{\beta}_{1B} = 0.5, \tilde{\beta}_{1C} = 0, \tilde{\beta}_{2A} = 0, \tilde{\beta}_{2B} = 1.5, \tilde{\beta}_{2C} = 0.5, \tilde{\beta}_{3A} = 2.5, \tilde{\beta}_{3B} = 0, \tilde{\beta}_{3C} = 0.5$: è facile verificare che $(\tilde{\alpha}, \tilde{\beta})$ è ammissibile per D_0 e ha valore $\frac{15}{2}$.

- Osserviamo che nei due esempi precedenti vale la seguente relazione $\gamma^* = \frac{z(P_R)}{z(P)}$. Mostreremo nel seguito come questo risultato, ovvero il fatto che γ^* sia uguale all'integrality gap della formulazione (P), valga sempre.

- Ricordiamo innanzitutto cosa è l'*integrality gap* della formulazione (P). Innanzitutto per comodità riscriviamo la formulazione:

$$
\begin{aligned}
z(P) = \min & \textstyle\sum_{i\in F} f_i y_i + \sum_{i\in F}\sum_{j\in N} d_{ij} x_{ij} \\
& \textstyle\sum_{i\in F} x_{ij} \ge 1,\ j \in N \\
& x_{ij} \le y_i,\ i \in F, j \in N \qquad\qquad (P)\\
& x_{ij}, y_i \in \{0,1\}, i \in F, j \in N.
\end{aligned}
$$

  Ricordiamo che, per definizione, $z(P) = c(N)$.

- Mostriamo ora che quello che è successo nei due precedenti esempi è sempre vero: per il Facility Location Game, γ^* è uguale all'integrality gap della formulazione P. Innanzitutto ricordiamo che per calcolare γ^* dobbiamo risolvere il seguente problema di programmazione lineare D':

  $\gamma^* \cdot c(N) = \max \sum_{j \in N} \alpha_j$ $\qquad (D')$

  $\sum_{j \in S} \alpha_j \leq c(S)$, per ogni $S \subseteq N$.D'

- Per mostrare che $\gamma^* = IG(P)$ è sufficiente mostrare che i problemi D e D' sono di fatto equivalenti. Infatti, per definizione: $IG(P) = \frac{z(P_R)}{z(P)} = \frac{w(D)}{z(P)}$ e $\gamma^* = \frac{w(D')}{c(N)} = \frac{w(D')}{z(P)}$.

- Mostriamo quindi che i problemi D e D' sono di fatto equivalenti. In particolare mostreremo come, se (α, β) è una soluzione per D, allora α è una soluzione per D' e che, se viceversa α è una soluzione per D', allora esiste β per cui (α, β) è una soluzione per D. Dato che D e D' hanno la stessa funzione obiettivo $\sum_{j \in N} \alpha_j$ segue che D e D' sono di fatto equivalenti, e quindi $w(D) = w(D')$.

  Sia dunque (α, β) è una soluzione per D. Sia $S \subseteq N$: per prima cosa mostriamo che $\sum_{j \in S} \alpha_j \leq c(S)$. Premettiamo il seguente fatto: sia $i \in F$ una fissata facility e $S(i) \subseteq N$ un fissato insieme di clienti. Se sommiamo il vincolo $\sum_{j \in N} \beta_{ij} \leq f_i$ e tutti i vincoli $\alpha_j \leq \beta_{ij} + d_{ij}$ per ogni $j \in S(i)$, otteniamo: $\sum_{j \in S(i)} \alpha_j + \sum_{j \in N \setminus S(i)} \beta_{ij} \leq f_i + \sum_{j \in S(i)} d_{ij}$ e quindi $\sum_{j \in S(i)} \alpha_j \leq f_i + \sum_{j \in S(i)} d_{ij}$ (poiché $\beta \geq 0$).

  Si consideri ora l'insieme F_S di facility aperte in una soluzione ottima al problema di Facility Location con facility F e clienti S, ovvero l'insieme di facility aperte in una soluzione ottima al problema P_S. Ogni facility $i \in F_S$ servirà un insieme di clienti $S(i) \subseteq S$, con $\bigcup_{i \in F_s} S(i) = S$ e $S(i) \cap S(i') = \emptyset$, per $i, i' \in F_S, i \neq i'$. (Si osservi che, banalmente, ogni cliente $j \in S$ si servirà presso la facility $i \in F_s$ a distanza minima.) Se adesso sommiamo la disuguaglianza appena ottenuta per tutte le coppie $(i, S(i))$, $i \in F_S$, otteniamo: $\sum_{j \in S} \alpha_j \leq \sum_{i \in F_s} f_i + \sum_{i \in F_s} \sum_{j \in S(i)} d_{ij} = \sum_{i \in F_s} f_i + \sum_{j \in S} \min_{i \in F_S} d_{ij} = c(S)$, come volevasi dimostrare.

  Mostriamo ora come viceversa, se α è tale che $\sum_{j \in S} \alpha_j \leq c(S)$, per ogni $S \in N$, allora esiste $\beta \geq 0$ tale che (α, β) è ammissibile per D. Come abbiamo fatto nell'esempio precedente, per ogni $i \in F$ e $j \in N$, poniamo $\beta_{ij} = \max\{0, \alpha_j - d_{ij}\}$. Per verificare che (α, β) è ammissibile per D resta dunque da dimostrare che $\sum_{j \in N} \beta_{ij} \leq f_i$, per ogni $i \in F$. Supponiamo per assurdo che esista $i \in F$ tale che $\sum_{j \in N} \beta_{ij} > f_i$. Sia $S \subseteq N$ l'insieme dei clienti per cui $\beta_{ij} > 0$; segue che $\sum_{j \in S} (\alpha_j - d_{ij}) > f_i$, ovvero $\sum_{j \in S} \alpha_j > f_i + \sum_{j \in S} d_{ij}$: il che non è possibile perché vorrebbe dire che il costo che α imputi ai giocatori di S è maggiore di quanto pagherebbero aprendo la sola facility i, ovvero: $\sum_{j \in S} \alpha_j > f_i + \sum_{j \in S} d_{ij} \geq c(S)$, il che è una contraddizione.

- Possiamo quindi concludere che per il Facility Location Game, γ^* è uguale all'integrality gap della formulazione P. Vedremo nella prossima sezione, come nell'ipotesi in cui

i costi d_{ij} definiscano una metrica, possiamo dimostrare che l'integrality gap è sempre al più 3. Ovvero, per una qualunque istanza del facility location game con costi metrici, possiamo imputare un costo α_j a ciascun giocatore in modo da garantire la stabilità rispetto ogni coalizione (ovvero, $\sum_{j\in S}\alpha_j \leq c(S)$) e recuperare almeno $\frac{1}{3}$ del costo $c(N)$ della grande coalizione (ovvero, $\sum_{j\in N}\alpha_j \geq \frac{1}{3}c(N)$).

2.1 Algoritmi approssimati per Facility Location

- Il trattamento di questa sezione è svolto nella dispensa di Anupam Gupta indicata nei riferimenti in cui sono richiamati alcuni risultati fondamentali sulla programmazione lineare, e presentati diversi algoritmi (deterministici e randomizzati) 4 e 3-approssimati per il problema del facility location, nell'ipotesi in cui i costi d_{ij} definiscano una metrica. Questi algoritmi restituiscono sempre, con complessità polinomiale, *sia* una soluzione al problema di facility location (ovvero l'insieme F' di facility da aprire e, di conseguenza, la connessione di ogni cliente alla facility più vicina tra quelle aperte) *sia* una soluzione (α, β) per il duale del rilassamento lineare di P_0 tale che $\sum_{i\in F'} f_i + \sum_{j\in N} \min_{i\in F'} d_{ij} \leq 3\sum_{j\in N}\alpha_j$. In particolare, segue da quello che abbiamo dimostrato in precedenza che per ogni coalizione $S \subseteq N$ vale $\sum_{j\in S}\alpha_j \leq c(S)$, e quindi il vettore α può essere interpretato come un vettore di costi stabile rispetto le coalizioni che ci permette di recuperare sempre almeno $\frac{1}{3}$ del costo della soluzione individuata.

- Quando diciamo che su un grafo $G(V, E)$ è definita una metrica $d : E \mapsto R_+$, intendiamo che il grafo è completo, ovvero $E = \binom{V}{2}$, e che per ogni tripla di vertici $i, h, j \in V$ vale: $d_{ih} + d_{hj} \geq d_{ij}$. Se d definisce una metrica, allora segue anche che per ogni quadrupla di vertici $i, h, k, j \in V$ vale: $d_{ih} + d_{hk} + d_{kj} \geq d_{ij}$ etc.

  Per il problema di facility location il grafo che ha senso considerare è un grafo bipartito completo $G(F \cup N, E)$, in cui cioè ogni facility è connessa a ogni cliente (gli spigoli tra coppie di facility o coppie di clienti possiamo aggiungerli o no, ma non sono interessanti). In questo caso se diciamo che $d : E \mapsto R_+$ definisce una metrica, per ogni quadrupla $i, i' \in F$ e $j', j \in V$ vale: $d_{ij'} + d_{i'j'} + d_{i'j} \geq d_{ij}$.

- Svolgendo l'algoritmo in Figura 3 per l'esempio 2, la prima cosa da fare è trovare una soluzione ottima per il rilassamento lineare di P_0 e il suo duale. Abbiamo già visto che le soluzioni ottime per i due problemi sono rispettivamente:

  - $x_{1A} = x_{3A} = x_{2B} = x_{2C} = x_{1C} = x_{3C} = \frac{1}{2}, y_1 = y_2 = y_3 = \frac{1}{2}$, con tutte le altre componenti di x a 0;
  - $\alpha_A = 3.5, \alpha_B = 2.5, \alpha_C = 1.5,\ \beta_{1A} = 1.5, \beta_{1B} = 0, \beta_{1C} = 2.5, \beta_{2A} = 0.5, \beta_{2B} = 1.5, \beta_{2C} = 0, \beta_{3A} = 0, \beta_{3B} = 0.5, \beta_{3C} = 0.5$.

  A questo punto al passo 2 dell'algoritmo dobbiamo scegliere un cliente j con α_j minimo: quindi scegliamo C. A questo punto dobbiamo scegliere una facility di

costo minimo tra quelle in N_C, quindi apriamo la facility 2. Osserviamo quindi che N_C interseca sia N_A che N_B, quindi l'algoritmo termina aprendo la sola facility 2 e connettendo tutti i clienti a 2. Il costo della soluzione individuata è 9, il costo della soluzione duale è 7.5 e effettivamente $9 \leq 4 \cdot 7.5$.

- Nella dimostrazione del Claim 2.3 sarebbe opportuno distinguere 2 casi: il caso dei clienti j assegnati a un impianto i tale che $x_{ij} > 0$ e il caso dei clienti j' assegnati a un impianto i tale che $x_{ij'} = 0$. Per i primi, dalle condizioni di complementarità segue che $\alpha_j = \beta_{ij} + d_{ij}$ e quindi, poiché $\beta_{ij} \geq 0$, segue che $d_{ij} \leq \alpha_j$. Per gli altri, vale $d_{ij'} \leq 3\alpha'_j$ i dettagli sono illustrati in modo chiaro sulla dispensa.

- Diamo qualche dettaglio in più sull'analisi dei costi dell'algoritmo randomizzato descritto a Pag. 5. La differenza tra questo algoritmo e quello descritto in Figura 3 sono due: 1) ad ogni iterazione il cliente j è scelto come quello che minimizza la quantità $\alpha_j + D_j$, dove $D_j = \sum_{i \in N_j} x_{ij} d_{ij}$; 2) una volta scelto j la facility i viene scelta in modo non deterministico ma secondo la distribuzione di probabilità x_{ij}. (Si noti che sulla dispensa c'è un errore: la definizione data al quart'ultimo rigo di Pag. 5 è quella di D_j e non di $D_{j'}$).

  Il costo atteso di apertura delle facility è spiegato in modo chiaro sulla dispensa ed è minore o uguale di $OPT(P)$.

  Per quanto riguarda il costo atteso di connessione del generico cliente j' procediamo come segue. Il cliente j' è stato assegnato durante una qualche iterazione a un impianto $i \in N_j$, dove j è il cliente selezionato all'inizio dell'iterazione: si noti che potrebbe anche valere $j = j'$. Il costo di connessione di j' è pari a $\sum_{i \in N_j} x_{ij} d_{ij'}$ e vogliamo dimostrare che questo costo è $\leq 2\alpha_{j'} + D_{j'}$. Sia $i' \in N_j \cap N_{j'}$: si noti che potrebbe anche valere $i = i'$. Per la disuguaglianza triangolare abbiamo che $d_{ij'} \leq d_{ij} + d_{i'j} + d_{i'j'}$. Quindi $\sum_{i \in N_j} x_{ij} d_{ij'} \leq \sum_{i \in N_j} x_{ij}(d_{ij} + d_{i'j} + d_{i'j'}) = \sum_{i \in N_j} x_{ij} d_{ij} + \sum_{i \in N_j} x_{ij} d_{i'j} + \sum_{i \in N_j} x_{ij} d_{i'j'} = \sum_{i \in N_j} x_{ij} d_{ij} + d_{i'j} \sum_{i \in N_j} x_{ij} + d_{i'j'} \sum_{i \in N_j} x_{ij} = \sum_{i \in N_j} x_{ij} d_{ij} + d_{i'j} + d_{i'j'} = D_j + d_{i'j} + d_{i'j'} \leq D_j + \alpha_j + \alpha'_j \leq 2\alpha_{j'} + D_{j'}$, dove la penultima disequazione segue dal fatto che $x_{i'j}$ e $x_{i'j'}$ sono entrambi positivi (infatti $i' \in N_j \cap N_{j'}$) e quindi dalle condizioni di complementarità segue che $d_{i'j} \leq \alpha_j$ e $d_{i'j'} \leq \alpha'_j$, e l'ultima disuguaglianza segue dal modo in cui si sceglie j.

  Combinando il costo di apertura delle facility e quello di connessione come descritto sulle dispense otteniamo che la soluzione restituita dall'algoritmo in valore atteso costa al più tre volte la soluzione duale.

2.2 Algoritmo primale-duale

- Consideriamo ora l'algoritmo *primale-duale*. La sua descrizione nella dispensa di Anupam Gupta è molto chiara, ma è opportuno modificare leggermente, nella seconda parte, l'assegnazione definitiva dei clienti alle facility aperte. Sia quindi F^a è l'insieme delle facility aperte definitivamente dall'algoritmo.

Consideriamo il generico cliente j:

- se al termine dell'algoritmo esiste una facility $i \in F^a$ tale che l'arco $\{i, j\}$ è tight, assegniamo j ad i (rompendo le situazioni di parità in modo arbitrario) e diciamo che j è *direttamente connesso a* i;
- viceversa, se j non è collegato con un arco tight ad alcuna facility aperta, vuol dire che la facility i' cui j era collegato al termine della prima fase dell'algoritmo è stata chiusa per un conflitto con un'altra facility i che invece è aperta: assegniamo j a i e diciamo che j è *indirettamente connesso a* i.

Per il resto l'algoritmo si comporta come descritto sulla dispensa. Diamo comunque qualche ulteriore dettaglio sull'analisi dell'algoritmo. Per ogni $i \in F^a$, sia S_i^I l'insieme dei clienti direttamente assegnati ad i e S_i^D l'insieme dei clienti indirettamente assegnati ad i.

Allora il costo della soluzione primale (intera) individuata dall'algoritmo è pari a:

$$\sum_{i \in F} f_i y_i + \sum_{i \in F} \sum_{j \in N} d_{ij} x_{ij} = \sum_{i \in F^a} f_i + \sum_{i \in F^a} \sum_{j \in N} d_{ij} x_{ij} = \sum_{i \in F^a} f_i + \sum_{i \in F^a} \sum_{j \in S_i^D} d_{ij} + \sum_{i \in F^a} \sum_{j \in S_i^I} d_{ij} =$$

$$= \sum_{i \in F^a} (f_i + \sum_{j \in S_i^D} d_{ij}) + \sum_{i \in F^a} \sum_{j \in S_i^I} d_{ij}.$$

Il costo della soluzione duale individuata dall'algoritmo è pari a:

$$\sum_{j \in N} \alpha_j = \sum_{i \in F^a} \sum_{j \in S_i^D \cup S_i^I} \alpha_j = \sum_{i \in F^a} \sum_{j \in S_i^D} \alpha_j + \sum_{i \in F^a} \sum_{j \in S_i^I} \alpha_j.$$

La strategia della dimostrazione è quella di mostrare che, per ogni $i \in F^a$ vale:

$$f_i + \sum_{j \in S_i^D} d_{ij} = \sum_{j \in S_i^D} \alpha_j$$

e che per ogni cliente j assegnato indirettamente a una facility i vale $d_{ij} \leq 3\alpha_j$ e che quindi vale:

$$\sum_{i \in F^a} \sum_{j \in S_i^I} d_{ij} \leq 3 \sum_{i \in F^a} \sum_{j \in S_i^I} \alpha_j.$$

Il fatto che per ogni cliente j assegnato indirettamente a una facility i valga $d_{ij} \leq 3\alpha_j$ può essere dimostrato con i soliti argomenti ed è comunque ben spiegato sulla dispensa. Per quanto riguarda la prima uguaglianza osserviamo che per ogni facility $i \in F^a$ vale:

$$f_i = \sum_{j \in N} \beta_{ij}.$$

È ora importante osservare che la risoluzione di tutti i conflitti tra le facility (nella seconda parte dell'algoritmo) assicura che la seguente proprietà è soddisfatta: sia $i \in F^a$, se $\beta_{ij} > 0$ allora j è direttamente assegnato a i, ovvero $j \in S_i$. Vale quindi:

$$f_i = \sum_{j \in N} \beta_{ij} = \sum_{j \in N : \beta_{ij} > 0} \beta_{ij} = \sum_{j \in S_i^D} \beta_{ij} = \sum_{j \in S_i^D} (\alpha_j - d_{ij})$$

che è quanto dovevamo dimostrare (l'ultima uguaglianza segue dal fatto che l'arco $\{i, j\}$ è tight).

- Svolgiamo l'algoritmo primale-duale per la seguente l'istanza del problema di Facility Location con insieme dei clienti $N = \{A, B, C, D\}$, insieme delle facility $F = \{1, 2, 3, 4\}$, costi di set-up $f_1 = 3, f_2 = 4, f_3 = 1, f_4 = 8$ e costi di connessione $d_{A,1} = 2, d_{A,2} = 3, d_{A,3} = 7, d_{A,4} = 7, d_{B,1} = 7, d_{B,2} = 2, d_{B,3} = 5, d_{B,4} = 7, d_{C,1} = 5, d_{C,2} = 7, d_{C,3} = 5, d_{C,4} = 7, d_{D,1} = 7, d_{D,2} = 7, d_{D,3} = 6, d_{D,4} = 7$.

  Innanzitutto è facile verificare che i costi di connessione soddisfano l'ipotesi metrica, quindi l'uso dell'algoritmo primale-duale è appropriato.

  Svolgiamo dunque l'algoritmo. Esso fa crescere ordinatamente le variabili $\alpha_j, j \in \{A, B, C, D\}$ e le variabili $\beta_{ij}, i \in \{1, 2, 3, 4\}, j \in \{A, B, C, D\}$. Immaginiamo che il suo svolgimento segua un clock esterno, rappresentato da una variabile temporale t: all'inizio $t = 0$ e tutte le variabili valgono 0, poi $t = \varepsilon$ e tutte le α_j valgono ε etc.

  Per $t \in (0, 2)$, le variabili α crescono uniformemente, le variabili β rimangono a 0, nessun arco è tight, nessuna facility è temporaneamente aperta.

  All'istante $t = 2$, $\alpha_A = \alpha_B = \alpha_C = \alpha_D = 2$, le variabili β sono a o, gli archi $\{1, A\}$ e $\{2, B\}$ sono tight, nessuna facility è temporaneamente aperta.

  Per $t \in (2, 3)$, le variabili α e le variabili β_{A1} e β_{B2} crescono uniformemente, le altre variabili β rimangono a 0, gli archi $\{1, A\}$ e $\{2, B\}$ sono tight, nessuna facility è temporaneamente aperta.

  All'istante $t = 3$, $\alpha_A = \alpha_B = \alpha_C = \alpha_D = 3$, $\beta_{A1} = \beta_{B2} = 1$, le altre variabili β sono a o, gli archi $\{1, A\}$, $\{2, B\}$ e $\{2, A\}$ sono tight, nessuna facility è temporaneamente aperta.

  Per $t \in (3, 4.5)$, le variabili α e le variabili β_{A1}, β_{B2} e β_{A2} crescono uniformemente, le altre variabili β rimangono a 0, $\{1, A\}$, $\{2, B\}$ e $\{2, A\}$ sono tight, nessuna facility è temporaneamente aperta.

  All'istante $t = 4.5$, $\alpha_A = \alpha_B = \alpha_C = \alpha_D = 4.5$, $\beta_{A1} = \beta_{B2} = 2.5$, $\beta_{A2} = 1.5$ e le altre variabili β sono a o, gli archi $\{1, A\}$, $\{2, B\}$ e $\{2, A\}$ sono tight, la facility 2 è temporaneamente aperta e i clienti A e B sono connessi (e la facility 2 è appunto il testimone di connessione). Da questo momento α_A, α_B e tutte le variabili β_{ij} con $j \in \{A, B\}$ non crescono oltre.

  All'istante $t = 5$, $\alpha_A = \alpha_B = 4.5, \alpha_C = \alpha_D = 5$, $\beta_{A1} = \beta_{B2} = 2.5$, $\beta_{A2} = 1.5$ e le altre variabili β sono a o, gli archi $\{1, A\}$, $\{2, B\}$, $\{2, A\}$, $\{1, C\}$ e $\{3, C\}$ sono tight, la facility 2 è temporaneamente aperta e i clienti A e B sono connessi (e la facility 2 è appunto il testimone di connessione). Da questo momento α_A, α_B e tutte le variabili β_{ij} con $j \in \{A, B\}$ non crescono oltre.

All'istante $t = 5.5$, $\alpha_A = \alpha_B = 4.5, \alpha_C = \alpha_D = 5.5$, $\beta_{A1} = \beta_{B2} = 2.5$, $\beta_{A2} = 1.5$, $\beta_{C1} = \beta_{C3} = 0.5$ e le altre variabili β sono a o, gli archi $\{1, A\}$, $\{2, B\}$, $\{2, A\}$, $\{1, C\}$ e $\{3, C\}$ sono tight, la facility 1 e la facility 2 sono temporaneamente aperte e i clienti A, B e C sono connessi (e la facility 1 è il testimone di connessione per C). Da questo momento $\alpha_A, \alpha_B, \alpha_C$ e tutte le variabili β_{ij} con $j \in \{A, B, C\}$ non crescono oltre.

All'istante $t = 6$, $\alpha_A = \alpha_B = 4.5, \alpha_C = 5.5, \alpha_D = 6$, $\beta_{A1} = \beta_{B2} = 2.5$, $\beta_{A2} = 1.5$, $\beta_{C1} = \beta_{C3} = 0.5$ e le altre variabili β sono a o, gli archi $\{1, A\}$, $\{2, B\}$, $\{2, A\}$, $\{1, C\}$, $\{3, C\}$ e $\{3, D\}$ sono tight, la facility 1 e la facility 2 sono temporaneamente aperte e i clienti A, B e C sono connessi.

All'istante $t = 6.5$, $\alpha_A = \alpha_B = 4.5, \alpha_C = 5.5, \alpha_D = 6.5$, $\beta_{A1} = \beta_{B2} = 2.5$, $\beta_{A2} = 1.5$, $\beta_{C1} = \beta_{C3} = 0.5$, $\beta_{D3} = 0.5$ e le altre variabili β sono a o, gli archi $\{1, A\}$, $\{2, B\}$, $\{2, A\}$, $\{1, C\}$, $\{3, C\}$ e $\{3, D\}$ sono tight, la facility 1, la facility 2 e la facility 3 sono temporaneamente aperte e i clienti A, B, C, e D sono connessi (e la facility 3 è il testimone di connessione per D). Questo termina la fase 1 dell'algoritmo.

Nella fase 2, l'insieme delle facility temporaneamente aperte è $F_t = \{1, 2, 3\}$. La coppia di facility 1 e 2 è però in conflitto perché β_{A1} e $\beta_{A2} > 0$; analogamente, la coppia di facility 1 e 3 è però in conflitto perché β_{C1} e $\beta_{C3} > 0$.

Consideriamo quindi ordinatamente le facility, nell'ordine in cui le abbiamo temporaneamente aperte. La prima facility che abbiamo aperto è 2, che quindi apriamo in modo definitivo. Allo stesso tempo, chiudiamo in modo definitivo la facility 1 che è in conflitto con 2. A questo punto rimane la sola facility 3 che apriamo in modo definitivo. L'insieme di facility da aprire è quindi $I = \{2, 3\}$.

Per assegnare i clienti alle facility aperte, utilizziamo la regola descritta in precedenza. Osserviamo che gli archi $\{2, B\}$, $\{2, A\}$, $\{3, C\}$ e $\{3, D\}$ sono tight, quindi assegniamo direttamente A e B a 2 e C e D a 3. Il costo di questa soluzione è pari a 21 e poiché anche il costo della soluzione duale è pari a 21, segue che la soluzione individuata è molto meglio che 3-approssimata: è ottima! (Si noti che l'analisi che abbiamo svolto in precedenza mostra che infatti *tutte* le volte che tutti i clienti sono connessi direttamente segue che la soluzione individuata dall'algoritmo ha lo stesso costo della soluzione del duale ed è quindi ottima).

- Svolgendo l'algoritmo primale-duale per l'esempio 2, al termine della prima fase abbiamo temporaneamente aperto le facility 2 e 3 e costruito la seguente soluzione ammissibile per il problema duale: $\alpha_A = 3, \alpha_B = 2, \alpha_C = 2$, $\beta_{1A} = 1, \beta_{2B} = 1, \beta_{2C} = 1, \beta_{3A} = 2, \beta_{3C} = 1$. Infine, gli archi tight sono: $\{A, 2\}, \{A, 3\}, \{B, 2\}, \{C, 2\}, \{C, 3\}$.

  Nella seconda fase, 2 e 3 sono in conflitto. La facility aperta per prima è la facility 2 per cui l'algoritmo termina aprendo la sola facility 2. I clienti connessi direttamente a 2 sono A e B, mentre C è connesso indirettamente. Il costo della soluzione primale è 10, il costo della soluzione duale è 7 e quindi il rapporto di approssimazione è pari a $\frac{10}{7} < 3$.

2.2.1 Esercizi sull'algoritmo primale-duale

- **Esercizio 1.** Applichiamo l'algoritmo primale-duale per l'istanza di Facility Location con insieme dei clienti $N = \{A, B, C, D\}$, insieme delle facility $F = \{1, 2, 3\}$, costi di set-up $f_1 = 3, f_2 = 4, f_3 = 5$ e costi di connessione $d_{A,1} = 2, d_{A,2} = 8, d_{A,3} = 7, d_{B,1} = 3, d_{B,2} = 3, d_{B,3} = 8, d_{C,1} = 8, d_{C,2} = 4, d_{C,3} = 7, d_{D,1} = 7, d_{D,2} = 7, d_{D,3} = 1$.

  Innanzitutto è facile verificare che i costi di connessione soddisfano l'ipotesi metrica, quindi l'uso dell'algoritmo primale-duale è appropriato.

  Svolgiamo dunque l'algoritmo. Esso fa crescere ordinatamente le variabili $\alpha_j, j \in \{A, B, C, D\}$ e le variabili $\beta_{ij}, i \in \{1, 2, 3, 4\}, j \in \{A, B, C, D\}$. Immaginiamo che il suo svolgimento segua un clock esterno, rappresentato da una variabile temporale t: all'inizio $t = 0$ e tutte le variabili valgono 0, poi $t = \varepsilon$ e tutte le α_j valgono ε etc.

  Per $t \in (0, 1)$, le variabili α crescono uniformemente, le variabili β rimangono a 0, nessun arco è tight, nessuna facility è temporaneamente aperta.

  All'istante $t = 1$, $\alpha_A = \alpha_B = \alpha_C = \alpha_D = 1$, le variabili β sono a o, l'arco $\{D, 3\}$ è tight, nessuna facility è temporaneamente aperta.

  Per $t \in (1, 2)$, le variabili α e la variabile β_{D3} crescono uniformemente, le altre variabili rimangono a 0, l'arco $\{D, 3\}$ è tight, nessuna facility è temporaneamente aperta.

  All'istante $t = 2$, $\alpha_A = \alpha_B = \alpha_C = \alpha_D = 2$, $\beta_{D3} = 1$, le altre variabili sono a 0, gli archi, $\{A, 1\}, \{D, 3\}$ sono tight, nessuna facility è temporaneamente aperta.

  Per $t \in (2, 3)$, le variabili α e le variabili β_{D3} e β_{A1} crescono uniformemente, le altre variabili rimangono a 0, gli archi, $\{A, 1\}, \{D, 3\}$ sono tight, nessuna facility è temporaneamente aperta.

  All'istante $t = 3$, $\alpha_A = \alpha_B = \alpha_C = \alpha_D = 3$, $\beta_{D3} = 2$, $\beta_{A1} = 1$ le altre variabili sono a o, gli archi, $\{A, 1\}, \{D, 3\}, \{B, 1\}, \{B, 2\}$ sono tight, nessuna facility è temporaneamente aperta.

  Per $t \in (3, 4)$, le variabili α e le variabili β_{D3}, β_{A1}, β_{B1}, β_{B2} crescono uniformemente, le altre variabili rimangono a 0, gli archi, $\{A, 1\}, \{D, 3\}, \{B, 1\}, \{B, 2\}$ sono tight, nessuna facility è temporaneamente aperta.

  All'istante $t = 4$, $\alpha_A = \alpha_B = \alpha_C = \alpha_D = 4$, $\beta_{D3} = 3$, $\beta_{A1} = 2$, $\beta_{B1} = 1$, $\beta_{B2} = 1$ le altre variabili sono a o, gli archi, $\{A, 1\}, \{D, 3\}, \{B, 1\}, \{B, 2\}, \{C, 2\}$ sono tight, la facility 1 è temporaneamente aperta e i clienti A e B sono connessi.

  Per $t \in (4, 6)$, le variabili α_B, α_C e le variabili β_{D3}, β_{C2} crescono uniformemente, le altre variabili rimangono ferme, gli archi, $\{A, 1\}, \{D, 3\}, \{B, 1\}, \{B, 2\}, \{C, 2\}$ sono tight, la facility 1 è temporaneamente aperta e i clienti A e B sono connessi.

  All'istante $t = 6$, $\alpha_A = \alpha_B = 4$, $\alpha_C = \alpha_D = 6$, $\beta_{D3} = 5$, $\beta_{A1} = 2$, $\beta_{B1} = 1$, $\beta_{B2} = 1$, $\beta_{C2} = 2$ le altre variabili sono a o, gli archi, $\{A, 1\}, \{D, 3\}, \{B, 1\}, \{B, 2\}, \{C, 2\}$

sono tight, le facility 1 e 3 sono temporaneamente aperte e i clienti A, B e D sono connessi.

Per $t \in (6, 7)$, le variabili α_C e β_{C2} crescono uniformemente, le altre variabili rimangono ferme, gli archi, $\{A, 1\}, \{D, 3\}, \{B, 1\}, \{B, 2\}, \{C, 2\}$ sono tight, le facility 1 e 3 sono temporaneamente aperte e i clienti A, B e D sono connessi.

All'istante $t = 7$, $\alpha_A = \alpha_B = 4$, $\alpha_C = 7$, $\alpha_D = 6$, $\beta_{D3} = 5$, $\beta_{A1} = 2$, $\beta_{B1} = 1$, $\beta_{B2} = 1$, $\beta_{C2} = 3$ le altre variabili sono a o, gli archi, $\{A, 1\}, \{D, 3\}, \{B, 1\}, \{B, 2\}, \{C, 2\}$ sono tight, le facility 1, 2 e 3 sono temporaneamente aperta e i clienti A, B, C e D sono connessi. Questo termina la fase 1 dell'algoritmo.

Nella fase 2, l'insieme delle facility temporaneamente aperte è $F_t = \{1, 2, 3\}$. La coppia di facility 1 e 2 è però in conflitto perché β_{B1} e $\beta_{B2} > 0$.

Consideriamo quindi ordinatamente le facility, nell'ordine in cui le abbiamo temporaneamente aperte. La prima facility che abbiamo aperto è 1, che quindi apriamo in modo definitivo. Allo stesso tempo, chiudiamo in modo definitivo la facility 2 che è in conflitto con 1. A questo punto rimane la sola facility 3 che apriamo in modo definitivo. L'insieme di facility da aprire è quindi $I = \{1, 3\}$.

Osserviamo ora che gli archi $\{A, 1\}, \{D, 3\}, \{B, 1\}, \{B, 2\}, \{C, 2\}$ sono tight, quindi assegniamo direttamente A e B a 1 e D a 3. Per quanto riguarda il cliente C lo assegniamo alla facility 1 "responsabile" della chiusura della facility 2 cui era connesso. Il costo di questa soluzione è pari a 22 e poiché il costo della soluzione duale è pari a 22, segue che la soluzione individuata è $\frac{22}{21}$-approssimata.

- **Esercizio 2.** Applichiamo l'algoritmo primale duale per l'istanza di Facility Location riportata nella sezione 2.2.1 della dispensa di Anupam Gupta. Per evitare che si manifestino delle situazioni di parità che potrebbero generare perplessità, supponiamo diversamente dalla dispensa che il costo delle facility $2, 3, \ldots n$ è pari a $n + 2 - \varepsilon$.

  Innanzitutto è facile verificare che i costi di connessione soddisfano l'ipotesi metrica, quindi l'uso dell'algoritmo primale-duale è appropriato.

  Svolgiamo quindi l'algoritmo primale-duale. Al primo passo aumentiamo tutti gli α_j, per $j = 1..2n$, fino al valore 1: a questo punto tutti gli archi di costo 1 sono tight. Aumentiamo quindi tutti gli α_j fino al valore 2 e per mantenere l'ammissibilità duale portiamo tutti i valori $\beta_{i,j}$ degli archi tight al valore 1. A questo punto possiamo aprire (temporaneamente) la prima facility per la quale le variabili β coprono il costo di apertura e possiamo connettere a questa facility tutti i clienti connessi ad essa connessi da archi tight: quindi i clienti $1, 2, \ldots, n + 1$. L'algoritmo prosegue aumentando gli $\alpha(j)$ di tutti i clienti non ancora connessi ad alcuna facility temporaneamente aperta: quindi aumentiamo gli α_j per $j = n + 2..2n$ fino al valore $3 - \varepsilon$ e per mantenere l'ammissibilità duale portiamo tutti i valori $\beta_{i,j}$ degli archi (tight) $\beta_{i,n+i}$, per $i = 2..n$, al valore $2 - \varepsilon$. A questo punto abbiamo coperto il costo di apertura di tutte le facility, che vengono tutte temporaneamente aperte e

ognuno dei cliente $j = n+2..2n$ viene connesso alla facility $j-n$. Termina quindi la prima fase dell'algoritmo primale duale che ha individuato: come soluzione intera al problema primale quella in cui apriamo tutte le facility con i primi $n+1$ clienti connessi alla prima facility e i clienti $j = n+2..2n$ connesso alla facility $j-n$: il costo di questa soluzione è pari a $(n+1)+(n-1)(n+2-\varepsilon)+(n+1)+(n-1)$; come soluzione duale abbiamo invece una soluzione di costo $2(n+1)+(3-\varepsilon)(n-1)$. Si noti che il rapporto tra il costo della soluzione primale e quello della soluzione duale è circa n: questo non deve sorprenderci perché non abbiamo ancora svolto la seconda fase dell'algoritmo.

Nella seconda fase costruiamo il grafo dei conflitti tra le facility e osserviamo che ogni coppia di facility è in conflitto perché ciascuno dei clienti $1, 2, \ldots, n$ ha "offerto" (attraverso le variabili β_{ij}) del denaro a ogni facility. Apriremo quindi solo la prima facility, aperta per prima nella prima fase, e collegheremo a questa facility direttamente i clienti $1, 2, \ldots, n+1$ e indirettamente i clienti $j = n+2..2n$: il costo di questa soluzione è pari a $2(n+1)+3(n-1)$, quindi differisce dal costo duale solo per un fattore (additivo) $\varepsilon(n-1)$ e quindi possiamo dire che la soluzione individuata dall'algoritmo è ottima per $n \to 0$.

- **Esercizio 3.** Consideriamo ora un problema di facility location con n clienti e 2 facility. Il costo di apertura della prima facility è pari a ε mentre il costo di apertura della seconda è pari a $(n+1)\varepsilon$. Il costo di connessione alla seconda facility è 1 per ciascun cliente, mentre il costo di connessione alla prima facility è 1 per il primo cliente e 3 per tutti gli altri clienti.

  Al solito è facile verificare che i costi di connessione soddisfano l'ipotesi metrica, quindi l'uso dell'algoritmo primale-duale è appropriato. Svolgendo l'algoritmo, nella prima fase apriamo la prima facility all'istante $1+\varepsilon$ e la seconda all'istante $1+\varepsilon+\frac{\varepsilon}{n-1}$. Nella seconda fase, tuttavia, chiudiamo la seconda facility perché il cliente 1 ha contribuito ai costi di apertura di entrambe le facility, che sono quindi in conflitto, e la prima facility è stata aperta prima. L'algoritmo termina aprendo la sola prima facility, con un costo di $1+\varepsilon+1+3(n-1)\ldots$ e questa soluzione costa circa 3 volte la soluzione che apre la sola seconda facility, che costa $(n+1)\varepsilon+n$, che è ottima.

  Questo esempio mostra che la nostra analisi del fattore di approssimazione dell'algoritmo primale-duale è *tight*, poiché ci sono dei casi in cui la soluzione individuata dista dalla soluzione ottima proprio per un fattore 3.

- Chiudiamo la discussione sull'algoritmo primale con alcune considerazioni sulle condizioni di complementarità. Se individuiamo una coppia di soluzioni ottime per il problema primale rilassato e per il problema duale, esse naturalmente devono soddisfare le condizioni di complementarità, che in questo caso sono:

  - $y_i > 0 \to \sum_{j \in N} \beta_{ij} = f_i$
  - $x_{ij} > 0 \to \alpha_j = \beta_{ij} + d_{ij}$
  - $\alpha_j > 0 \to \sum_{i \in F} x_{ij} = 1$

- $\beta_{ij} > 0 \rightarrow x_{ij} = y_i$

L'algoritmo primale individua invece una la soluzione *intera* del primale e una soluzione del duale e, dal momento che in generale esisterà un integraility gap, non possiamo attenderci che questa coppia di soluzioni soddisfi le condizioni di complementarità. Tuttavia, è facile verificare che l'algoritmo (appositamente!) costruisce una coppia di soluzioni che soddisfano 3 delle 4 condizioni di complementarità e violano la condizione:

$$x_{ij} > 0 \rightarrow \alpha_j = \beta_{ij} + d_{ij}$$

per i soli clienti j che sono assegnati in modo indiretto a una facility i. Avendo tuttavia dimostrato che, in questo caso $d_{ij} \leq 3\alpha(j)$ segue che la condizione di complementarità è soddisfatta in modo *approssimato*! Infatti, vale:

$$x_{ij} > 0 \rightarrow \frac{1}{3}\beta_{ij} + d_{ij} \leq \alpha_j \leq \beta_{ij} + d_{ij}$$

.