4.20

MENJABE *produeix una gran varietat de menús de menjars diferents. Malauradament, només poden produir els seus menús en quantitats limitades, de manera que solen quedar-se sense els més populars, deixant clients insatisfets. Per minimitzar aquest problema,* MENJABE *vol implementar un sofisticat sistema de distribució de dinars. Els clients haurien d'enviar un missatge de text amb les seves opcions de menús acceptables abans de l'hora de dinar. A continuació, fan una assignació de dinars als clients.* MENJABE *té decidit compensar amb un val de 5 euros als clients als qui no pot assignar cap de les seves opcions.* MENJABE *vol minimitzar la quantitat de vals que dona.*

a) *Doneu un algorisme eficient per a assignar menús als clients en un dia. En general, en un dia determinat,* MENJABE *sap que ha produït m tipus de menús i la quantitat de cadascun d'ells* $q_1, \ldots, q_m$. *A més els n clients envien un text amb les seves preferències, el client i indica un conjunt* A_i *de menús acceptables. L'algorisme ha d'assignar a cada client una de les seves opcions o un val de 5 euros, de manera que es minimitzi el nombre de vals.*

Solució

Resolem el problema via càlcul de fluxos màxims. Definim la següent xarxa $G = (V, E, c)$ amb capacitats $c : E \to Z^+ \cup \{0\}$, on el conjunt de vèrtexos es defineix com

$$V = \{s, t, C_1, \ldots, C_n, M_1, \ldots, M_m\},$$

essent s el node font, t el node embornal, $\{C_1, \ldots, C_n\}$ els nodes que representen els n clients, i $\{M_1, \ldots, M_m\}$ els nodes que representen els m menús. El conjunt d'arestes E es defineix com

$$E = \{(s, C_i)\}_{1 \leq i \leq n} \bigcup \{(M_j, t)\}_{1 \leq j \leq m} \bigcup \{(C_i, M_j) \mid M_j \in A_i\}.$$

A les arestes de la font s als nodes C_i fixem capacitat 1. A les arestes dels nodes M_j a l'embornal t fixem capacitat q_j que correspon a la quantitat de menús de tipus M_j que MENJABE pot servir. A totes les arestes (C_i, M_j) fixem també capacitat 1.

Un cop construida la xarxa, apliquem un algorisme de max-flow de manera que si $f : E \to \mathbb{N}$ és el flux màxim calculat per l'algorisme, podem definir l'assignació òptima per a MENJABE

$$assig : \{C_1, \ldots, C_n\} \to \{M_1, \ldots, M_m\} \cup \{val\}$$

de la següent forma:

- $f(s, C_i) = 1 \implies \exists j : M_j \in A_i : f(C_i, M_j) = 1$
  En aquest cas definim $assig(C_i) = M_j$ amb cost per a MENJABE $cost(C_i) = 0$.
- $f(s, C_i) = 0 \implies \forall j : M_j \in A_i : f(C_i, M_j) = 0$
  En aquest cas $assig(C_i) = val$ amb cost per a MENJABE $cost(C_i) = 5$.
- $f(M_j, t) \leq q_j \implies$ mai se serveixen més menús dels que es disposa.

L'algorisme de max-flow maximitza

$$\big|\{C_i \mid f(s, C_i) = 1\}\big|,$$

(el nombre de clients amb flow 1 des de la font) i per tant maximitza

$$\big|\{C_i \mid cost(C_i) = 0\}\big|,$$

(el nombre de clients que produeixen cost 0) i això vol dir que minimitza

$$\big|\{C_i \mid cost(C_i) = 5\}\big|$$

(el nombre de clients que produeixen cost 5 perquè se'ls proporciona un val) i minimitza per tant el nombre de vals que s'han de repartir. Així queda demostrada la correctesa.

Quin algorisme seria més adequat entre Ford-Fulkerson i Edmons-Karp? Notem que el nombre de nodes és $|V| = n+m+2$, i el nombre d'arestes és $|E| = n+\sum_{i=1}^{n}|A_i|+m = O(nm)$.

- Ford-Fulkerson: $O(|E||f^*|) = O(n^2m)$, perquè $|f^*| \leq n$
- Edmons-Karp: $O(|V||E|^2) = O(n^3m^2+n^2m^3)$

Per tant, en aquest cas el més eficient seria el Ford-Fulkerson i caldria afegir el temps necessari per calcular l'assignació a partir del fluxe. En total, $T(n,m) = O(n^2m)$.

b) *Un dels directius de* MENJABE *proposa minimitzar el diners gastats en vals oferint un descompte en el preu del menú als clients que optin per una opció de menú sorpresa. En aquest cas els clients poden rebre qualsevol dels menús disponibles amb un descompte de 3 euros en el menú servit. En aquest cas, els clients envien un email indicant que volen, o bé un menú sorpresa, o bé la llista de les seves preferències. De nou, si no els poden servir un menú dintre de les seves preferències o, en el cas dels menús sorpresa no se'ls pots servir cap menú, rebran el val de 5 euros. Doneu un algorisme eficient per a determinar l'assignació menys costosa per a* MENJABE.

Solució

Construïm una xarxa com la de l'apartat a), considerant només els clients que tenen una llista de preferències (i no menú sorpresa). Calculem el flux màxim $f^*_{sorpresa}$, minimitzant així

$$\big|\{C_i \mid A_i \neq \{\text{sorpresa}\} \wedge assig(C_i) = val\}\big|.$$

Ara calculem:

- $Q = \sum_{i=1}^{m} q_j - |f^*_{\text{sorpresa}}|$ dels menús que no han estat assignats encara, i
- $S = |\{C_i \mid A_i = \{\text{sorpresa}\}\}|$ el nombre de clients que desitgen menú sorpresa.

Si $Q \geq S$, aleshores assignem un menú sorpresa a tots els clients que volien menú sorpresa. Cost mínim per $\{C_i \mid A_i = \{\text{sorpresa}\}\}$: $3S$.

Si $Q < S$, aleshores assignem tots els menús sorpresa possibles (en total Q), i assignem un val a la resta (en total $S-Q$), i aleshores el cost mínim per $\{C_i \mid A_i = \{\text{sorpresa}\}\}$: $3S+5(S-Q)$.

No es poden barrejar els clients C_i amb $A_i \subseteq \{M_1,\ldots,M_m\}$ (clients amb conjunt de menús acceptables) amb els clients C_j amb $A_j = \{\text{menú\_sorpresa}\}$. Si a la xarxa tinguéssim al mateix temps aquests dos tipus de clients, el client "sorpresa" podria acceptar qualsevol menú. Indirectament, aleshores maximitzar el flux no implicaria minimitzar el cost.

L'eficiència de l'algorisme proposat segueix l'anàlisi de l'apartat a), afegint-hi l'assignació de menús sopresa ($O(n)$ addicional). En total, $T(n,m) = O(n^2m)$.