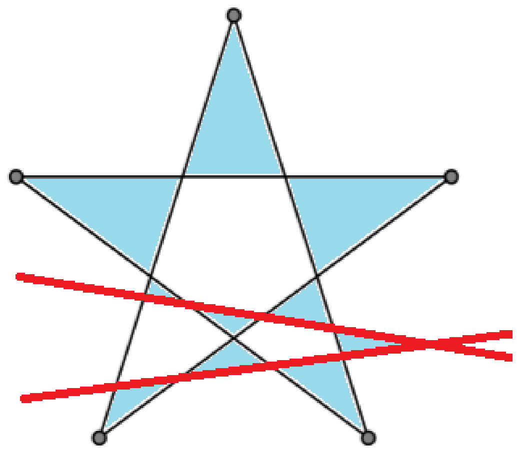
Εθνικό Μετσόβιο Πολυτεχνείο
Σχολή Ηλεκτρολόγων Μηχανικών & Μηχανικών Υπολογιστών
Προηγμένα Θέματα Αλγορίθμων (Ροή Λ) – 8ο Εξάμηνο 2018-2019
Αβραμίδης Κλεάνθης – 03115117

1η Σειρά Ασκήσεων – Απαντήσεις

- **Άσκηση 1**

Η λύση του προβλήματος φαίνεται γραφικά στο παρακάτω σχήμα:



- **Άσκηση 2**

Αποδεικνύουμε αρχικά πως ο αλγόριθμος της εκφώνησης δίνει λύση στο Vertex Cover: Υποθέτουμε ότι δε δίνει λύση. Τότε θα υπάρχει ακμή $e(u, v)$ που δε θα καλύπτεται, συνεπώς κανένας από τους κόμβους u, v δε θα είναι επιλεγμένος, θα είναι δηλαδή φύλλα. Αυτό είναι άτοπο, καθώς δε μπορούν 2 φύλλα να συνδέονται μεταξύ τους.

Αποδεικνύουμε τώρα πως αυτός ο αλγόριθμος είναι 2-προσεγγιστικός, χρησιμοποιώντας την έννοια του maximal matching: Γνωρίζουμε πως το σύνολο κόμβων της OPT λύσης δε μπορεί να είναι μικρότερης πληθικότητας από το σύνολο των ανεξάρτητων ακμών στο maximal matching (δε θα έφταναν οι κόμβοι για να καλύψουν όλες τις ανεξάρτητες ακμές – χρειαζόμαστε 1 ανά τέτοια ακμή). Ισχύει λοιπόν $OPT = |C^*| \geq |MM|$.

Παράγουμε τώρα maximal matching στο DFS Tree με την εξής διαδικασία: Ξεκινάμε από τη ρίζα του δέντρου και επιλέγουμε μία από τις προσπίπτουσες ακμές και τη διαγράφουμε. Επαναλαμβάνουμε στα δέντρα που προκύπτουν μέχρι να εξαλειφθούν όλα ή να καταλήξουν σε μοναδικό κόμβο (φύλλο). Καταφέρνουμε έτσι να καλύψουμε κάθε εσωτερικό κόμβο του γραφήματος ενώ τα όποια φύλλα μένουν ακάλυπτα δεν μπορούν έτσι κι αλλιώς να συνδέονται. Συνεπώς δε χάνουμε κάποια ακμή και το matching είναι maximal. Βλέπουμε πως *στην καλύτερη περίπτωση* κάθε ακμή που επιλέξαμε θα καλύπτει 2 κόμβους ενώ θα έχουν μείνει ακάλυπτα όλα τα φύλλα, που είναι αχρείαστα. Συνεπώς $|MM| \geq \frac{|V| - φύλλα}{2} = \frac{SOL}{2}$. Σε συνδυασμό με την παραπάνω σχέση: $OPT \geq |MM| \geq \frac{SOL}{2} \rightarrow \boldsymbol{\frac{SOL}{OPT} \leq 2}$. Άρα πράγματι ο αλγόριθμος είναι 2-προσεγγιστικός για το πρόβλημα Vertex Cover.

- **Άσκηση 3**

Θεωρώντας OPT την βέλτιστη λύση του προβλήματος Set Cover (δηλ. τον ελάχιστο αριθμό επιλογών σετ) γνωρίζουμε πως για το σετ μέγιστης πληθικότητας θα ισχύει $|S_m| \geq \frac{N}{OPT}$ όπου N ο συνολικός αριθμός των διαθέσιμων στοιχείων. Αυτό ισχύει καθώς σε αντίθετη περίπτωση δε θα αρκούσαν OPT σύνολα για να συγκεντρωθούν τα N στοιχεία, όπως απαιτείται από το πρόβλημα. Με βάση την εκφώνηση, σε κάθε επιλογή επιλέγουμε σετ για το οποίο ισχύει:

$$\frac{|S_m|}{|S_{sel}|} \leq \rho \;\rightarrow\; |S_{sel}| \geq \frac{|S_m|}{\rho} \geq \frac{N}{\rho * OPT}$$

Μετά την πρώτη επιλογή θα έχουν μείνει $N_1 = N - |S_{sel}| \leq N - \frac{N}{\rho * OPT} = N\left(1 - \frac{1}{\rho * OPT}\right)$ στοιχεία διαθέσιμα. Μετά την δεύτερη επιλογή θα έχουν μείνει κατ' αναλογία $N_2 \leq N_1 - \frac{N_1}{\rho*(OPT-1)} \leq N_1 - \frac{N_1}{\rho*OPT} = N_1\left(1 - \frac{1}{\rho*OPT}\right) = N\left(1 - \frac{1}{\rho*OPT}\right)^2$ στοιχεία. Γενικά, μετά από k επαναλήψεις θα έχουν μείνει $N_k \leq N\left(1 - \frac{1}{\rho*OPT}\right)^k \leq Ne^{\frac{-k}{\rho*OPT}}$ στοιχεία. Για την επίλυση του προβλήματος με $SOL = k$ επιλογές, θα πρέπει να ισχύει το εξής:

$$N_k \leq 1 \rightarrow Ne^{\frac{-k}{\rho*OPT}} \leq 1 \rightarrow \ln\left(e^{\frac{-k}{\rho*OPT}}\right) \geq \ln\left(\frac{1}{N}\right) \;\rightarrow\; \frac{-k}{\rho * OPT} \geq -lnN \rightarrow \boldsymbol{\frac{SOL}{OPT}} \leq \mathbf{ln}(\boldsymbol{N^{\rho}})$$

Για το Weighted Set Cover, θεωρούμε OPT το συνολικό βάρος των σετ που επιλέγει ο βέλτιστος αλγόριθμος. Γνωρίζουμε συνεπώς ότι όλα τα N διαθέσιμα στοιχεία καλύπτονται από σετ αθροιστικού βάρους OPT. O κλασικός greedy αλγόριθμος για το συγκεκριμένο πρόβλημα θα επιλέγει κάθε φορά το σετ S με ελάχιστο ratio $r_S = \frac{w(S)}{|S/C|}$ όπου C το σύνολο των μέχρι στιγμής καλυμμένων στοιχείων. Δείχνουμε συνοπτικά ότι $r_S \leq \frac{OPT}{N}$:

$$r_S = \min_{\mathrm{i:S_i}}(r) = \frac{w(S)}{|S/C|} \leq \frac{\sum_{i\epsilon OPT} w(S_i)}{\sum_{i\epsilon OPT}|S/C|} = \frac{OPT}{N}$$

Κατ' αναλογία στην k επιλογή θα ισχύει $r_S \leq \frac{OPT}{N_k}$ όπου $N_k = N - |C_k|$ τα στοιχεία που παραμένουν ακάλυπτα. Στην περίπτωση τώρα που η επιλογή σετ δε γίνεται με ακρίβεια, αλλά ρ-προσεγγιστικά, θα ισχύει $\frac{r_{sel}}{r_S} \leq \rho \rightarrow r_{sel} \leq \frac{\rho OPT}{N}$. Ακολουθώντας την παραπάνω συλλογιστική, σε k επαναλήψεις θα έχουν μείνει

$$N_k = N_{k-1} - |S/C_{k-1}| = N_{k-1} - \frac{w(S)}{r_S} \leq N_{k-1} - \frac{w(S) * N_{k-1}}{OPT * \rho} = N_{k-1}\left(1 - \frac{w(S)}{OPT * \rho}\right)$$

Από την τελευταία σχέση μπορούμε αναδρομικά να λάβουμε:

$$N_k \leq N * \prod_{i=1}^{k-1}\left(1 - \frac{w(S)}{OPT * \rho}\right) \leq N * \prod_{i=1}^{k-1} \exp\left(\frac{-w(S)}{OPT * \rho}\right) = N * \exp\left(\frac{-1}{OPT * \rho}\sum_{i=1}^{k-1} w(S)\right)$$

Εφαρμόζοντας φυσικό λογάριθμο και στα 2 μέλη της ανίσωσης έχουμε τελικά:

$$\ln(N_k) \;\leq\; \ln(N) - \frac{1}{OPT * \rho}\sum_{i=1}^{k-1} w(S) \;\rightarrow\; \sum_{i=1}^{k-1} w(S) \;\leq\; \ln\left(\frac{N}{N_k}\right) * OPT * \rho \;\leq\; \rho\ln(N) * OPT$$

αφού γενικά $N_k \geq 1$. Για την επίλυση του προβλήματος με k επιλογές έχουμε $SOL = \sum_{i=1}^{k-1} w(S)$ οπότε:

$$SOL \leq \rho\ln(N) * OPT \;\rightarrow\; \boldsymbol{\frac{SOL}{OPT}} \leq \mathbf{ln}(\boldsymbol{N^{\rho}})$$

(β) *Tight Example for Cardinality Set Cover greedy APX Algorithm:* Έστω σύνολο $S = S_a \cup S_b$ όπου:

$$S_a = \{a_1, a_2, \dots, a_{2^{n-1}}\}, \qquad S_b = \{b_1, b_2, \dots, b_{2^{n-1}}\}$$

Θεωρούμε επίσης για $i = 1,2,\dots,n$ τα σύνολα:

$$S_1 = \{a_{2^{n-1}}, a_{2^{n-1}+1}, \dots, a_{2^n-1}\} \cup \{b_{2^{n-1}}, b_{2^{n-1}+1}, \dots, b_{2^n-1}\}$$

$$S_2 = \{a_{2^{n-2}}, a_{2^{n-2}+1}, \dots, a_{2^{n-1}-1}\} \cup \{b_{2^{n-2}}, b_{2^{n-2}+1}, \dots, b_{2^{n-1}-1}\}$$

$$\dots$$

$$S_i = \left\{a_{2^{n-i}}, a_{2^{n-i}+1}, \dots, a_{2^{n-i+1}-1}\right\} \cup \left\{b_{2^{n-i}}, b_{2^{n-i}+1}, \dots, b_{2^{n-i+1}-1}\right\}$$

Παρατηρούμε πως $|S_a| = |S_b| = 2^n - 1$ ενώ $|S_1| = 2 * (2^n - 2^{n-1}) = 2^n$, $|S_2| = 2 * (2^{n-1} - 2^{n-2}) = 2^{n-1}$ και γενικά $|S_i| = 2^{n-i+1}$. Συνεπώς, ο greedy αλγόριθμος θα επιλέξει πρώτα το S_1. Στη συνέχεια, θεωρώντας μόνο τα ακάλυπτα στοιχεία θα έχουμε $|S_a| = |S_b| = 2^n - 1 - |S_1|/2 = 2^{n-1} - 1$ οπότε ο αλγόριθμος θα επιλέξει το S_2. Η ίδια διαδικασία ακολουθείται μέχρι τέλους, οπότε ο αλγόριθμος θα τερματίσει όταν $|S_a| = |S_b| = \emptyset$ έχοντας επιλέξει $\log(2^n) = n$ σύνολα (προφανώς $OPT = 2$). Βλέπουμε λοιπόν πως ο αλγόριθμος έχει logN προσεγγισιμότητα.

- ## Άσκηση 4

Έστω σύνολο S με N στοιχεία και f το μέγιστο πλήθος διαθέσιμων υποσυνόλων του S που μπορεί να ανήκει ένα στοιχείο. Προτείνουμε τον εξής αλγόριθμο: Διαλέγουμε ένα ακάλυπτο στοιχείο και επιλέγουμε όλα τα σετ (το πολύ f) στα οποία ανήκει. Επαναλαμβάνουμε μέχρις ότου να μην υπάρχουν ακάλυπτα στοιχεία, οπότε και έχουμε λύση για το πρόβλημα Set Cover. Αποδεικνύουμε την f-προσεγγισιμότητα του αλγορίθμου:

Θεωρούμε OPT τη βέλτιστη λύση του προβλήματος, δηλαδή OPT σύνολα περιέχουν όλα τα N στοιχεία. Στον αλγόριθμό μας, για κάθε στοιχείο θα επιλέξουμε το πολύ f σύνολα. Μέσα σε αυτά θα περιέχεται σίγουρα ένα σύνολο που ανήκει στα OPT καθώς σε αντίθετη περίπτωση θα είχαμε βρει στοιχείο που δεν καλύπτεται από τη βέλτιστη λύση. Συνεπώς θα χρειαστούμε το πολύ OPT επιλογές μέχρι να λύσουμε το πρόβλημα. Θα ισχύει:

$$SOL \leq \sum_{i=1}^{OPT} f = f * OPT \rightarrow \boldsymbol{\frac{SOL}{OPT} \leq f}$$

Tight Example: Θεωρούμε N στοιχεία και τα σύνολα: $S = \{a_1, a_2, \dots, a_n\}$, $f - 1$ μονοσύνολα $\{a_1\}$. Ο αλγόριθμος θα επιστρέψει όλα τα f σύνολα, ενώ $OPT = 1$. Συνεπώς ο αλγόριθμος είναι όντως f-προσεγγιστικός.

Για το Weighted Set Cover, επεκτείνουμε τον degree-weighted αλγόριθμο για το Weighted Vertex Cover: Θεωρούμε υπερσύνολο U, k σύνολα $S_1, \dots, S_k$ και N στοιχεία, καθένα από τα οποία απαντάται σε το πολύ f από αυτά τα σύνολα. Θεωρούμε επίσης συνάρτηση w: $S \rightarrow \mathbb{N}^+$ ανάθεσης βαρών στα σύνολα, τέτοιων ώστε για κάθε διαθέσιμο σύνολο S να ισχύει $\mathrm{t}(S) = c|S|$ (degree-weighted function). Για την συγκεκριμένη συνάρτηση θα ισχύει $t\left(\sum_{i=0}^{k} S_i\right) \leq f * OPT$ όπου OPT το optimal Set Cover του συγκεκριμένου προβλήματος. Απόδειξη:

Έστω U ένα optimal Set Cover. Αφού το U καλύπτει όλα τα στοιχεία θα ισχύει $\sum_{S \in U} |S| \geq N \rightarrow t(U) \geq cN$. Ισχύει ακόμη από υπόθεση ότι $\sum_{i=0}^{k} |S_i| \leq fN \rightarrow\ t\left(\sum_{i=0}^{k} |S_i|\right) \leq c * fN \leq f * t(U) = f * OPT$.

Εφαρμόζουμε τον ακόλουθο αλγόριθμο για την εύρεση weighted Set Cover: Για τα μη κενά σύνολα υπολογίζουμε την ποσότητα $\mathrm{c} = \min\{w(S)/|S|\}$ και ορίζουμε την degree-weighted function $t(S) = c|S|$. Θα αποσυνθέσουμε τα βάρη των συνόλων μέσω της t θεωρώντας:

$$D = \{\, S \mid S = \emptyset \,\}, \qquad W = \{\, S \mid w(S) = t(S) \,\}, \qquad U' = U - (D \cup K)$$

Παράλληλα αφαιρούμε τα στοιχεία που επιλέχθηκαν από τα εναπομείναντα σύνολα. Επαναλαμβάνουμε τον αλγόριθμο μέχρις ότου $U' = \emptyset$ (έστω m επαναλήψεις). Έχουμε χωρίσει πλέον το U στα σύνολα $\bigcup_{i=0}^{m-1} W_i$ και $\bigcup_{i=0}^{m} D_i$. Το Set Cover αποτελείται από την ένωση όλων των W συνόλων που υπολογίστηκαν από τον αλγόριθμο: $C = \bigcup_{i=0}^{m-1} W_i$. Θα αποδείξουμε την ορθότητα του αλγορίθμου καθώς και την f-προσεγγισιμότητα:

Έστω ότι C δεν είναι Set Cover. Θα υπάρχει τότε στοιχείο που δεν έχει καλυφθεί και σύνολο S που το περιέχει. Αυτό το σύνολο θα βρίσκεται υποχρεωτικά στο $U - C = \bigcup_{i=0}^{m} D_i$, άτοπο αφού κάθε D_i περιέχει μόνο κενά σύνολα. Συνεπώς ο παραπάνω αλγόριθμος επιλύει πράγματι το πρόβλημα Set Cover. Έστω τώρα C^* ένα optimal Set Cover καθώς και ένα σύνολο $S \in C$. Θα ισχύει

$$S \in W_i \rightarrow w(S) = \sum_{j \leq i} t_j(S) \quad \& \quad S \in D_i \rightarrow w(S) \geq \sum_{j < i} t_j(S)$$

Λόγω της φύσης του αλγόριθμου (τεχνική επαναληπτικής βελτίωσης για το σύνολο C) το $C^* \cap U_i$ θα είναι Set Cover για το U_i. Συνεπώς, όπως αποδείξαμε πριν, $t_i(C \cap U_i) \leq f * t_i(C^* \cap U_i)$. Τελικά συνάγουμε:

$$SOL = w(C) = \sum_{i=0}^{m-1} t_i(C \cap U_i) \leq f * \sum_{i=0}^{m-1} t_i(C^* \cap U_i) \leq f * w(C^*) = f * OPT \ \rightarrow \boldsymbol{\frac{SOL}{OPT} \leq f}$$

- **Άσκηση 5**

(α) *Ο πρώτος αλγόριθμος:* Αρχικά βρίσκουμε ένα MST στο γράφημα, για το οποίο γνωρίζουμε ότι έχει κόστος το πολύ ίσο με το βέλτιστο μονοπάτι Euler: $cost(MST) \leq OPT$. Στη συνέχεια διπλασιάζουμε τις ακμές του, οπότε $cost_{new}(MST) = 2 * cost(MST) \leq 2OPT$. Ακολούθως αφαιρούμε το ελάχιστου κόστους $path(s,t)$, οπότε ισχύει:

$$cost(SOL) \leq 2OPT - cost\big(path(s,t)\big) \leq 2OPT - c_{s,t}$$

όπου $c_{s,t} = cost\Big(\min\big(path(s,t)\big)\Big)$. Βρίσκουμε αντίστοιχο μονοπάτι Euler και εκτελούμε short-cutting που δεν αυξάνει το κόστος (ιδιότητα της τριγωνικής ανισότητας). Συνεπώς $\boldsymbol{SOL_1 \leq 2OPT - c_{s,t}}$.

Ο δεύτερος αλγόριθμος: Βρίσκουμε και πάλι MST και υπολογίζουμε ένα perfect matching για τους κόμβους περιττού βαθμού του γραφήματος. Γνωρίζουμε πως $cost(PM) \leq \frac{OPT}{2}$. Ακόμη θα ισχύει $OPT \leq OPT_{s-t} + c(s,t)$ διότι το βέλτιστο μονοπάτι Hamilton θα φράσσεται από το αντίστοιχο βέλτιστο s-t μονοπάτι συν το κόστος του s-t path στο MST. Συμπληρώνουμε τέλος το μονοπάτι ενώνοντας τους κόμβους άρτιου βαθμού χωρίς διπλότυπα (short-cutting). Η ενέργεια αυτή δε θα αυξήσει το κόστος πάλι λόγω της τριγωνικής ανισότητας. Συνεπώς:

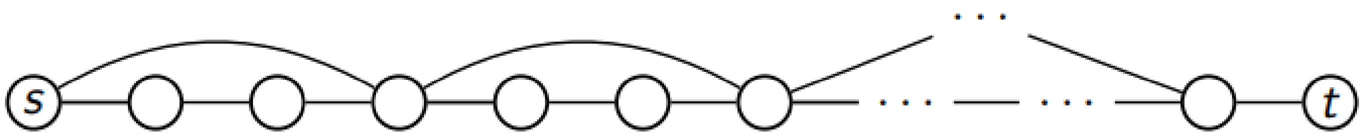
$$SOL_2 \leq \frac{1}{2}OPT + OPT_{s-t} \leq \frac{1}{2}\big(OPT_{s-t} + c(s,t)\big) + OPT_{s-t} \rightarrow \boldsymbol{SOL_2 \leq \frac{3OPT_{s-t} + c(s,t)}{2}}$$

Η τελική μας λύση είναι $SOL = \min\{SOL_1, SOL_2\}$. Για το λόγο προσεγγισιμότητας διακρίνουμε 2 περιπτώσεις:

1. $SOL_1 \leq SOL_2 \rightarrow 2OPT - c_{s,t} \leq \frac{3OPT + c_{s,t}}{2} \rightarrow \frac{OPT}{2} \leq \frac{3c_{s,t}}{2} \rightarrow c_{s,t} \geq \frac{OPT}{3}$
   $SOL = SOL_1 \leq 2OPT - c_{s,t} = 2OPT - \frac{OPT}{3} = \frac{5}{3}OPT$
2. $SOL_1 \geq SOL_2 \rightarrow 2OPT - c_{s,t} \geq \frac{3OPT + c_{s,t}}{2} \rightarrow \frac{OPT}{2} \geq \frac{3c_{s,t}}{2} \rightarrow c_{s,t} \leq \frac{OPT}{3}$
   $SOL = SOL_2 \leq \frac{3OPT + c_{s,t}}{2} \leq \frac{3OPT + OPT/3}{2} = \frac{5}{3}OPT$

Σε κάθε περίπτωση λοιπόν θα ισχύει $\boldsymbol{\frac{SOL}{OPT} \leq \frac{5}{3}}$, οπότε ο συνολικός αλγόριθμος θα είναι 5/3-προσεγγιστικός.

(β) *Tight Example:* Θα δείξουμε tight example για το οποίο και οι δύο επιμέρους αλγόριθμοι, συνεπώς και ο συνολικός, εμφανίζουν τον ζητούμενο λόγο προσέγγισης 5/3:



Θεωρούμε το παραπάνω πλήρες γράφημα με ακμές μοναδιαίου κόστους. Όσες ακμές δε φαίνονται έχουν κόστος ανάλογο του μήκους ενός μονοπατιού από τις ήδη υπάρχουσες. Υποθέτοντας n τριάδες, το βέλτιστο s-t μονοπάτι Euler θα είναι η ευθεία με κόστος $3n$. Ένα ελάχιστο συνδετικό δέντρο για το γράφημα:

Σύμφωνα με τον πρώτο αλγόριθμο θα διπλασιάσουμε τις ακμές του γραφήματος και θα αφαιρέσουμε το min s-t path του MST που εν προκειμένω αποτελείται από τις εξωτερικές ακμές. Το Euler path θα έχει την εξής μορφή:

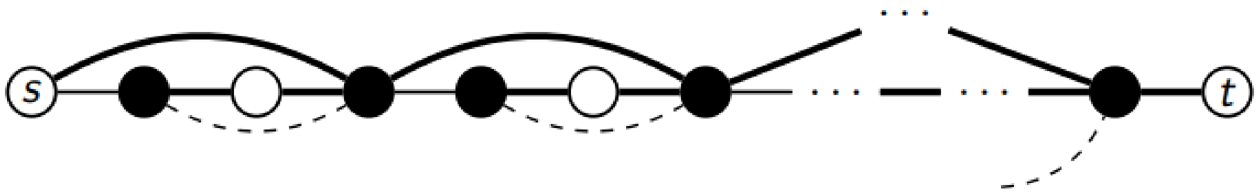
$$1 \equiv \mathrm{s} \rightarrow 4 \rightarrow 3 \rightarrow 2 \rightarrow 3 \rightarrow 4 \rightarrow 7 \rightarrow 6 \rightarrow 5 \rightarrow 6 \rightarrow 7 \rightarrow \cdots \rightarrow t$$

Στον κόμβο 2 θα εκτελεστεί short-cutting: θα συνδεθούν 2 και 7, αγνοώντας έτσι τους 3,4. Η ακμή 2-7 θα έχει ωστόσο κόστος 3. Συνολικά το κόστος θα έχει την εξής μορφή:

$$cost = 1 + 1 + 1 + 3 + 1 + 1 + 3 + \cdots + 3 = \cdots = 5(n-1) + 4 = 5n - 1$$

Συνεπώς προσεγγίζουμε ασυμπτωτικά τον ζητούμενο λόγο 5/3.

Σύμφωνα με τον δεύτερο αλγόριθμο, αφού δημιουργήσουμε το παραπάνω MST θα πρέπει να μαρκάρουμε τους κόμβους περιττού βαθμού (εκτός των s,t) και να βρούμε ένα perfect matching:



Έχουμε κόστος 2 για τις διακεκομμένες ακμές και συνολικό κόστος $5n$ για το μονοπάτι. Εκτελώντας short-cutting όπως προηγουμένως για να αποφύγουμε επανάληψη κόμβου, καταλήγουμε και πάλι σε κόστος $5n-1$, δηλαδή σε λόγο προσέγγισης 5/3. Συνεπώς έχουμε λόγο 5/3 και για τους 2 επιμέρους αλγόριθμους. Τελικά επιλέγουμε $SOL = \min\{SOL_1, SOL_2\} = \frac{5}{3}OPT$, δηλαδή και ο συνολικός αλγόριθμος θα έχει τον ζητούμενο λόγο.

- **Άσκηση 6**

Γενική μορφή tableau και αλγόριθμος Simplex: Κάθε στήλη αναφέρεται σε μία μεταβλητή του προγράμματος, η οποία δηλώνεται στην κορυφή, εκτός της τελευταίας στήλης που αναφέρεται στο δεξί μέλος των constraint functions. Από την άλλη, κάθε γραμμή αναφέρεται σε ένα constraint, εκτός της τελευταίας που αναφέρεται στην αντικειμενική συνάρτηση. Αριστερά των γραμμών δηλώνονται οι βασικές εφικτές μεταβλητές ενώ κάτω δεξιά δηλώνεται το κόστος της μέχρι τώρα λύσης. Με πράσινο δηλώνεται η εισερχόμενη βασική μεταβλητή και η αντίστοιχη στήλη περιστροφής και με κόκκινο η εξερχόμενη βασική μεταβλητή και η αντίστοιχη γραμμή περιστροφής. Περιλαμβάνεται επίσης μία επιπλέον στήλη όπου αναφέρεται το θ-ratio κάθε γραμμής. Τέλος, σημειώνουμε πως η σύμβασή μας θέλει το max problem ως standard form: ($\min c^T x \to -\max -c^T x$).

(α1) Παρουσιάζουμε διαδοχικά τα Simplex tableau για την πρώτη εκτέλεση του αλγορίθμου:

vars	x_1	x_2	x_3	x_4	x_5	x_6	x_7	b	***ratio***
x_5	1/2	−11/2	−5/2	9	1	0	0	0	0
x_6	1/2	−3/2	−1/2	1	0	1	0	0	0
x_7	1	0	0	0	0	0	1	1	1
f	−10	57	9	24	0	0	0	0	**0**

vars	x_1	x_2	x_3	x_4	x_5	x_6	x_7	b	***ratio***
x_1	1	−11	−5	18	2	0	0	0	–
x_6	0	4	2	−8	−1	1	0	0	0
x_7	0	11	5	−18	−2	0	1	1	1/11
f	0	−53	−41	204	20	0	0	0	**0**

vars	x_1	x_2	x_3	x_4	x_5	x_6	x_7	b	***ratio***
x_1	1	0	1/2	−4	−3/4	11/4	0	0	0
x_2	0	1	1/2	−2	−1/4	1/4	0	0	0
x_7	0	0	−1/2	4	3/4	−11/4	1	1	–
f	0	0	−29/2	98	27/4	53/4	0	0	**0**

vars	x_1	x_2	x_3	x_4	x_5	x_6	x_7	b	***ratio***
x_3	2	0	1	−8	−3/2	11/2	0	0	–
x_2	−1	1	0	2	1/2	−5/2	0	0	0
x_7	1	0	0	0	0	0	1	1	–
f	29	0	0	−18	−15	93	0	0	**0**

vars	x_1	x_2	x_3	x_4	x_5	x_6	x_7	b	***ratio***
x_3	−2	4	1	0	1/2	−9/2	0	0	0
x_4	−1/2	1/2	0	1	1/4	−5/4	0	0	0
x_7	−1	0	0	0	0	0	1	1	–
f	20	9	0	0	−21/2	141/2	0	0	**0**

$vars$	x_1	x_2	x_3	x_4	x_5	x_6	x_7	b	$ratio$
x_5	-4	8	2	0	1	-9	0	0	$-$
x_4	$1/2$	$-3/2$	$-1/2$	1	0	1	0	0	0
x_7	1	0	0	0	0	0	1	1	$-$
f	-22	93	21	0	0	-24	0	0	$\mathbf{0}$

$vars$	x_1	x_2	x_3	x_4	x_5	x_6	x_7	b	$ratio$
x_5	$1/2$	$-11/2$	$-5/2$	9	1	0	0	0	0
x_6	$1/2$	$-3/2$	$-1/2$	1	0	1	0	0	0
x_7	1	0	0	0	0	0	1	1	1
f	-10	57	9	24	0	0	0	0	$\mathbf{0}$

Παρατηρούμε πως έχουμε επανέλθει στη μορφή του πρώτου tableau, δηλαδή ο αλγόριθμος έπεσε σε cycling. Αυτό συμβαίνει επειδή οι περιορισμοί του συγκεκριμένου προβλήματος είναι πλεονάζοντες (degeneracy). Υπό αυτές τις συνθήκες, ο αλγόριθμος τελικά αποτυγχάνει να επιστρέψει βέλτιστη λύση για το πρόβλημα και η μέθοδος Simplex απαιτεί συγκεκριμένες τροποποιήσεις ώστε να επιστρέψει λύση, η οποία λόγω αυτής της ιδιότητας θα περιέχει και μηδενικές τιμές για βασικές μεταβλητές.

(α2) Ο νέος κανόνας που εισάγεται αποτελεί ουσιαστικά τον κανόνα του Bland και μεταβάλλει την εκτέλεση του αλγορίθμου μόνο στην τελευταία από τις παραπάνω επαναλήψεις, επιτρέποντας στον αλγόριθμο να τερματίσει:

$vars$	x_1	x_2	x_3	x_4	x_5	x_6	x_7	b	$ratio$
x_5	-4	8	2	0	1	-9	0	0	$-$
x_4	$1/2$	$-3/2$	$-1/2$	1	0	1	0	0	0
x_7	1	0	0	0	0	0	1	1	$-$
f	-22	93	21	0	0	-24	0	0	$\mathbf{0}$

$vars$	x_1	x_2	x_3	x_4	x_5	x_6	x_7	b	$ratio$
x_5	0	-20	-2	8	1	-1	0	0	$-$
x_1	1	-3	-1	2	0	2	0	0	$-$
x_7	0	3	1	-2	0	-2	1	1	1
f	0	27	-1	44	0	20	0	0	$\mathbf{0}$

$vars$	x_1	x_2	x_3	x_4	x_5	x_6	x_7	b	$ratio$
x_5	0	-14	0	4	1	-5	2	2	$-$
x_1	1	0	0	0	0	0	1	1	$-$
x_3	0	3	1	-2	0	-2	1	1	$-$
f	0	30	0	42	0	18	1	1	$\mathbf{1}$

Ο αλγόριθμος Simplex τερματίζει καθώς η τελευταία γραμμή έχει αποκλειστικά μη αρνητικές καταχωρήσεις. Άρα το γραμμικό πρόγραμμα είναι επιλύσιμο και έχει βέλτιστη λύση τη $(\boldsymbol{x_1}, \boldsymbol{x_2}, \boldsymbol{x_3}, \boldsymbol{x_4}) = (\mathbf{1}, \mathbf{0}, \mathbf{1}, \mathbf{0})$ με το κόστος να είναι -1 (αφού μιλάμε για εξαρχής πρόβλημα ελαχιστοποίησης).

(β) Το dual του δοθέντος γραμμικού προγράμματος είναι το ακόλουθο:

$$\begin{aligned} &\max y_3 \\ s.t. \quad &0.5y_1 + 0.5y_2 + y_3 \leq -10 \\ &-5.5y_1 - 1.5y_2 \leq 57 \\ &-2.5y_1 - 0.5y_2 \leq 9 \\ &9y_1 + y_2 \leq 24 \end{aligned}$$

Οι 2 complementary slackness συνθήκες, δεδομένης της λύσης $(x_1, x_2, x_3, x_4) = (1,0,1,0)$ για το primal, είναι:

$$\begin{cases} 0.5y_1 + 0.5y_2 + y_3 = -10 \\ -2.5y_1 - 0.5y_2 = 9 \end{cases}$$

Εξαρχής δε μπορούμε να επιλύσουμε το σύστημα. Παρατηρούμε όμως στο primal πως ο πρώτος περιορισμός δεν είναι tight: $0.5 * 1 - 2.5 * 1 = -2 < 0$. Συνεπώς θα ισχύει $y_1 = 0$ και τώρα το σύστημα λύνεται, δίνοντας $y_2 = -18$, $y_3 = -1$. Τελικά, η λύση του dual γραμμικού προγράμματος είναι η $(\boldsymbol{y_1}, \boldsymbol{y_2}, \boldsymbol{y_3}) = (\mathbf{0}, \mathbf{-18}, \mathbf{-1})$.

- ## Άσκηση 7

Θεωρούμε κυρτό σύνολο $P \subseteq \mathbb{R}^n$ και κυρτή συνάρτηση $f: \mathbb{R}^n \to \mathbb{R}$. Θέλουμε να δείξουμε $f(x) \leq f(y)\ \forall y \in P$. Θεωρούμε σημείο x^* που είναι βέλτιστο σε μια γειτονιά του $N_\varepsilon(x^*)$. Θεωρούμε επίσης σημείο $y \in P$ εκτός της $N_\varepsilon(x^*)$. Επειδή P κυρτό, θα υπάρχει ευθύγραμμο τμήμα με άκρα τα x^*, y και θα δίνεται από την παραμετρική σχέση $tx^* + (1-t)y\ \ \forall t \in [0,1]$. Για $t \to 1$ παρατηρούμε πως το τμήμα αυτό θα συγκεντρώνεται στην $N_\varepsilon(x^*)$. Επομένως $f(x^*) \leq f(tx^* + (1-t)y)$. Επειδή όμως f κυρτή, θα ισχύει τελικά:

$$f(x^*) \leq f(tx^* + (1-t)y) \leq tf(x^*) + (t-1)f(y) \ \to\ (t-1)\big(f(x^*) + f(y)\big) \geq 0 \ \to\ f(x^*) \leq f(y)$$

Επειδή επιλέξαμε τυχαίο $y \in P$, αποδείξαμε το ζητούμενο και το x^* θα είναι βέλτιστο σε όλο το P.

Εφαρμογή στην ορθότητα του Simplex: Η συνθήκη τερματισμού του αλγόριθμου Simplex είναι η μη δυνατότητα εύρεσης καλύτερης γειτονικής κορυφής, κάτι που συνεπάγεται πως βρισκόμαστε σε τοπικά βέλτιστη λύση. Από το παραπάνω όμως προκύπτει πως η λύση είναι και γενικά βέλτιστη, άρα ο Simplex επιστρέφει βέλτιστη λύση.

- ## Άσκηση 8

(α) Καταρχάς, αν x είναι κορυφή, μπορούμε να πάρουμε $\mathrm{u} = x$. Υποθέτουμε λοιπόν πως δεν είναι κορυφή. Θα υπάρχει τότε $y \neq 0$ τέτοιο ώστε $x + y,\ x - y \in P$. Δηλαδή, εφόσον το x δεν είναι κορυφή, θα μπορούσαμε να μετακινηθούμε είτε στην κατεύθυνση $+y$ είτε στην $-y$ προκειμένου να βελτιώσουμε τη λύση μας. Συγκεκριμένα, θα προτιμήσουμε την κατεύθυνση $+y$ αν $c^T(x+y) \leq c^T x \ \to\ c^T y \leq 0$ και αντίστοιχα την $-y$ αν $c^T(x-y) \leq c^T x \ \to\ c^T(-y) \leq 0$. Θεωρώντας λοιπόν $z \in \{y, -y\}$ ισχύει γενικά $c^T z \leq 0$. Σημειώνουμε επίσης ότι:

$$x + y, x - y \in P \ \ \to\ \ A(x+y) = A(x-y) = b \ \ \to\ \ Az = 0$$

Έστω ότι μετακινούμαστε κατά την κατεύθυνση $x + az,\ a > 0$. Παρατηρούμε αρχικά πως

$$A(x + az) = Ax + aAz = Ax = b$$

$$c^T(x + az) = c^T x + ac^T z \leq c^T x$$

Δηλαδή, η κατεύθυνσή μας βρίσκεται εξ ολοκλήρου στο διαθέσιμο πεδίο που ορίζεται από την $Ax = b$ ($x + az \in P$) και βελτιώνει την υπάρχουσα λύση, ωστόσο πιθανώς να μην συναντήσουμε ποτέ ένα όριο του P. Εξετάζουμε αυτό το ενδεχόμενο διακρίνοντας 2 περιπτώσεις:

1. $\exists j: z_j < 0$: Καθώς το α αυξάνει, το z_j μειώνεται μέχρις ότου να μην είναι πλέον επιλύσιμο ($x + az < 0$). Διαλέγουμε $\alpha = \min\limits_{j:\, z_j < 0} x_j / z_j = x_k / z_k$ το μεγαλύτερο δυνατό a τέτοιο ώστε $x + az \geq 0$. Από αυτό και από τα παραπάνω συνάγουμε ότι $x + az$ έχει επιπλέον μηδενικούς συντελεστές από το x. Συνεπώς το αντικαθιστούμε. Η διαδικασία αυτή δε μπορεί να επαναληφθεί πάνω από n φορές, αφού το πρόβλημα έχει n συντελεστές. Συνεπώς καταλήγουμε πάντα σε μια κορυφή του P.
2. $\forall j: z_j \geq 0$: Υποθέτουμε $c^T z < 0$ (η περίπτωση $c^T z = 0$ δεν βελτιώνει την αντικειμενική συνάρτηση) και $x + az$ επιλύσιμο για $\alpha \geq 0$. Ισχύει $x + az \geq x \geq 0$ όμως $c^T(x + az) = c^T x + ac^T z \to -\infty$ όταν $\alpha \to \infty$. Δηλαδή το πρόβλημα είναι μη φραγμένο, που είναι όμως άτοπο για την υπόθεσή μας.

(β) Για το πρώτο ζητούμενο, έχουμε ότι $P \neq \emptyset$, άρα θα υπάρχουν σημεία x_i που σύμφωνα με το (α) θα αντιστοιχούν σε κορυφές του P σύμφωνα με τη σχέση $c^T u \leq c^T x_i$. Για το δεύτερο ζητούμενο, έστω ότι η βέλτιστη λύση του $\min\{c^T x : x \in P\}$ δεν αποτελεί κορυφή του P. Σύμφωνα με το (α) θα υπάρχει κορυφή που θα κοστίζει όχι παραπάνω από αυτή τη λύση, άρα θα είναι και αυτή βέλτιστη.

- ## Άσκηση 9

Θεωρούμε το εξής primal-dual pair:

$$\begin{cases} \max\ 0^T x \\ Ax \leq b \end{cases} \quad \& \quad \begin{cases} \min\ b^T y \\ A^T y = 0 \\ y \geq 0 \end{cases}$$

Παρατηρούμε καταρχάς πως αν $primal\ feasible$, τότε θα έχει optimal κόστος 0. Αφού $y = 0$ ικανοποιεί το dual, το dual είτε θα είναι μη φραγμένο, είτε θα έχει optimal σημείο. Υποθέτουμε λοιπόν πως $primal\ feasible$ με κόστος 0. Σε αυτή την περίπτωση $dual\ feasible$ με κόστος 0 λόγω δυϊσμού. Άρα $\nexists y \geq 0 : b^T y < 0$ (συμβιβαστό σύστημα). Αντίστροφα, αν το σύστημα είναι συμβιβαστό, τότε για κάθε y θα ισχύει είτε $A^T y \neq 0$ είτε $b^T y \geq 0$. Στην πρώτη περίπτωση $dual\ infeasible$ άρα $primal\ unbounded$, άτοπο. Στη δεύτερη περίπτωση το dual σίγουρα θα φράσσεται από κάτω, άρα θα έχει optimal σημείο. Αυτό συνεπάγεται ωστόσο πως $primal\ feasible$. Αποδείξαμε λοιπόν το εξής:

$$primal\ επιλύσιμο \leftrightarrow dual\ συμβιβαστό \quad δηλ. \quad \boldsymbol{primal\ μη\ επιλύσιμο} \leftrightarrow \boldsymbol{dual\ μη\ συμβιβαστό}$$

✓ ***Βιβλιογραφία***

1. V.V. Vazirani. Approximation Algorithms: Chapters 2, 3
2. A. Levitin. Introduction to the Design and Analysis of Algorithms: Chapter 10.1
3. Michael X. Goemans, Linear Programming
4. Διαφάνειες μαθήματος στο CoreLab «Προσεγγιστικοί Αλγόριθμοι»
5. Σχήματα Άσκησης 5: https://resources.mpi-inf.mpg.de/conferences/adfocs-15/material/David-Lect2.pdf?fbclid=IwAR1pE8TZ-Jl9Zea4kECeVhG13FukPvoF7Uwhi61sGbxhplCLWMKv3rPHjJU

ΤΕΛΟΣ ΑΝΑΦΟΡΑΣ