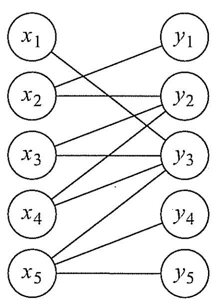
Κεφάλαιο 7

Ροή δικτύου

Σε αυτό το κεφάλαιο θα εστιάσουμε σε ένα πλούσιο σύνολο αλγοριθμικών προβλημάτων τα οποία, κατά μία έννοια, πηγάζουν από ένα από τα αρχικά προβλήματα που διατυπώσαμε στην αρχή του βιβλίου: το *Διμερές Ταίριασμα*.

Ας θυμηθούμε τη διαμόρφωση του προβλήματος για το Διμερές Ταίριασμα. Το *διμερές γράφημα* (bipartite graph) $G = (V, E)$ είναι ένα μη κατευθυνόμενο γράφημα του οποίου το σύνολο κόμβων μπορεί να διαμεριστεί ως $V = X \cup Y$, με την ιδιότητα ότι κάθε ακμή $e \in E$ έχει μόνο ένα άκρο στο X και το άλλο άκρο στο Y. Συχνά σχεδιάζουμε τα διμερή γραφήματα με τη μορφή που φαίνεται στην Εικόνα 7.1, με τους κόμβους του X σε μια στήλη στα αριστερά, τους κόμβους του Y σε μια στήλη στα δεξιά, και την κάθε ακμή να διασταυρώνει από την αριστερή προς τη δεξιά στήλη.

Έχουμε ήδη συναντήσει την έννοια του *ταιριάσματος* (matching) σε αρκετά σημεία αυτού του βιβλίου: Χρησιμοποιήσαμε αυτόν τον όρο για να περιγράψουμε συλλογές ζευγαριών από ένα σύνολο, με την ιδιότητα ότι κανένα από τα στοιχεία του συνόλου δεν εμφανίζεται σε περισσότερα από ένα ζευγάρια. (Σκεφθείτε τα αρσενικά (X) που ταιριάζουν με τα θηλυκά (Y) στο πρόβλημα του Ευσταθούς Ταιριάσματος, ή το ταίριασμα χαρακτήρων στο πρόβλημα Ευθυγράμμισης Ακολουθίας.) Στην περίπτωση του γραφήματος οι ακμές συνιστούν ζευγάρια κόμβων, και κατά συνέπεια λέμε ότι το *ταίριασμα* σε ένα γράφημα $G = (V, E)$ είναι ένα σύνολο ακμών $M \subseteq E$ που έχει την ιδιότητα ότι ο κάθε κόμβος εμφανίζεται το πολύ σε μία ακμή του M. Ένα σύνολο ακμών M αποτελεί *τέλειο ταίριασμα* (perfect matching) όταν ο κάθε κόμβος εμφανίζεται σε ακριβώς μία ακμή του M.



Εικόνα 7.1 Ένα διμερές γράφημα.

Τα ταιριάσματα σε διμερή γραφήματα μπορούν να μοντελοποιήσουν καταστάσεις στις οποίες τα αντικείμενα *αντιστοιχίζονται* σε άλλα αντικείμενα. Συναντήσαμε κάποιες τέτοιες περιπτώσεις σε προηγούμενες αναλύσεις σχετικά με τα γραφήματα και τα διμερή γραφήματα. Ένα φυσικό παράδειγμα παρουσιάζεται όταν οι κόμβοι του συνόλου X αντιπροσωπεύουν εργασίες, οι κόμβοι του συνόλου Y αντιπροσωπεύουν μηχανές, και η ακμή (x_i, y_j) υποδηλώνει ότι η μηχανή y_j έχει τη δυνατότητα εκτέλεσης της εργασίας x_i. Στην περίπτωση αυτή το τέλειο ταίριασμα είναι η αντιστοίχιση της κάθε εργασίας σε μια μηχανή που μπορεί να την εκτελέσει, με την ιδιότητα ότι στην κάθε μηχανή αντιστοιχίζεται ακριβώς μία εργασία. Τα διμερή γραφήματα μπορούν να αναπαραστήσουν πολλές ακόμα σχέσεις που παρουσιάζονται μεταξύ δύο διακριτών συνόλων αντικειμένων, όπως τις σχέσεις μεταξύ πελατών και καταστημάτων, κατοικιών και γειτονικών σταθμών πυροσβεστικής, και πολλές άλλες.

Ένα από τα παλαιότερα προβλήματα στους συνδυαστικούς αλγορίθμους είναι το πρόβλημα του προσδιορισμού του μεγέθους του μεγαλύτερου ταιριάσματος σε ένα διμερές γράφημα G. (Ως ειδική περίπτωση, σημειώστε ότι το G διαθέτει τέλειο ταίριασμα αν και μόνο αν $|X| = |Y|$ και υπάρχει ένα ταίριασμα με μέγεθος $|X|$.) Αυτό το πρόβλημα αποδεικνύεται ότι λύνεται από έναν αλγόριθμο που εκτελείται σε πολυωνυμικό χρόνο, όμως η ανάπτυξη αυτού του αλγορίθμου απαιτεί ιδέες που είναι θεμελιωδώς διαφορετικές από τις τεχνικές τις οποίες έχουμε συναντήσει μέχρι τώρα.

Αντί να αναπτύξουμε αμέσως τον αλγόριθμο, θα ξεκινήσουμε με τη διατύπωση μιας γενικής κατηγορίας προβλημάτων — τα προβλήματα *ροής δικτύου* (network flow) — τα οποία περιλαμβάνουν το πρόβλημα του Διμερούς Ταιριάσματος ως ειδική περίπτωση. Στη συνέχεια θα αναπτύξουμε έναν αλγόριθμο πολυωνυμικού χρόνου για ένα γενικό πρόβλημα, το *πρόβλημα Μέγιστης Ροής* (Maximum-Flow Problem), και θα δείξουμε πώς αυτό το πρόβλημα παρέχει επίσης έναν αποδοτικό αλγόριθμο και για το πρόβλημα του Διμερούς Ταιριάσματος. Αν και το αρχικό κίνητρο για τα προβλήματα ροής δικτύου προέρχεται από το ζήτημα της κίνησης σε ένα δίκτυο, θα δούμε ότι τα προβλήματα αυτά έχουν εφαρμογή σε ένα εντυπωσιακά μεγάλο εύρος τομέων και οδηγούν σε αποδοτικούς αλγορίθμους όχι μόνο για το πρόβλημα του Διμερούς Ταιριάσματος, αλλά και για πάμπολλα άλλα προβλήματα.

7.1 Το πρόβλημα της Μέγιστης Ροής και ο αλγόριθμος Ford-Fulkerson

Το πρόβλημα

Τα γραφήματα χρησιμοποιούνται συχνά για τη μοντελοποίηση των *δικτύων μεταφοράς* (transportation networks) — δηλαδή δικτύων στα οποία οι ακμές μεταφέρουν κάποιο είδος κίνησης και οι κόμβοι λειτουργούν ως "μεταγωγείς" για τη μεταφορά της κίνησης στις διάφορες ακμές. Σκεφθείτε, για παράδειγμα, ένα οδικό δίκτυο εθνικών οδών όπου οι ακμές είναι οι εθνικοί οδοί και οι κόμβοι είναι κυκλοφοριακοί κόμβοι (έξοδοι) των εθνικών οδών· ή ένα δίκτυο υπολογιστών όπου οι ακμές είναι συνδέσεις που μπορούν να μεταφέρουν πακέτα και οι κόμβοι είναι μεταγωγείς· ή ένα δίκτυο μεταφοράς υγρών όπου οι ακμές είναι σωληνώσεις και οι κόμβοι είναι συνδέσεις μεταξύ των σωληνώσεων. Τα μοντέλα δικτύου αυτού του τύπου έχουν αρκετά συστατικά: *χωρητικότητες* (capacities) στις ακμές, οι οποίες δείχνουν πόση ποσότητα μπορούν να μεταφέρουν· κόμβους *προέλευσης* (source) στο γράφημα, οι οποίοι παράγουν κίνηση· κόμβους *απόληξης* (sink) (ή προορισμού) στο γράφημα, οι οποίοι μπορούν να "απορροφήσουν" την κίνηση που καταφθάνει· και τέλος, την ίδια την κίνηση που διαδίδεται μέσω των ακμών.

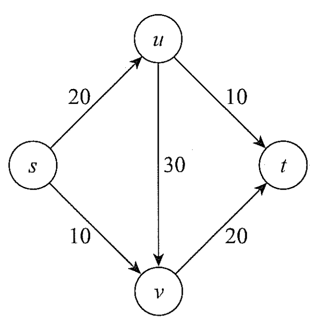
Δίκτυα Ροής Θα εξετάσουμε γραφήματα αυτής της μορφής, και θα αναφερόμαστε στην κίνηση ως *ροή* (flow) — μια αφηρημένη οντότητα που παράγεται στους κόμβους προέλευσης, μεταδίδεται μέσω των ακμών, και απορροφάται από τους κόμβους απόληξης. Τυπικά, θα λέμε ότι το *δίκτυο ροής* (flow network) είναι ένα κατευθυνόμενο γράφημα $G = (V, E)$ με τα ακόλουθα χαρακτηριστικά.

- Με κάθε ακμή e είναι συσχετισμένη μια *χωρητικότητα*, η οποία είναι ένας μη αρνητικός αριθμός τον οποίο συμβολίζουμε με το c_e.
- Υπάρχει ένας μόνο κόμβος *προέλευσης* $s \in V$.
- Υπάρχει ένας μόνο κόμβος *απόληξης* $t \in V$.

Οι υπόλοιποι κόμβοι, εκτός των s και t, θα ονομάζονται *εσωτερικοί* (internal) κόμβοι.

Θα κάνουμε τρεις παραδοχές σχετικά με τα δίκτυα ροής με τα οποία θα ασχοληθούμε: πρώτον, ότι δεν υπάρχει καμία ακμή που να εισέρχεται στην προέλευση s ή να εξέρχεται από την απόληξη t· δεύτερον, ότι υπάρχει τουλάχιστον μία ακμή προσκείμενη σε κάθε κόμβο· και τρίτον ότι όλες οι χωρητικότητες είναι ακέραιοι αριθμοί. Αυτές οι παραδοχές μας διευκολύνουν να εξετάζουμε τα ζητήματα πιο ξεκάθαρα, και αν και εξαλείφουν κάποιες παθολογικές περιπτώσεις διατηρούν ουσιαστικά όλα τα ζητήματα τα οποία θέλουμε να μελετήσουμε.

Η Εικόνα 7.2 δείχνει ένα δίκτυο ροής με τέσσερις κόμβους και πέντε ακμές, όπου οι τιμές της χωρητικότητας εμφανίζονται δίπλα στις ακμές.



Εικόνα 7.2 Ένα δίκτυο ροής με προέλευση s και απόληξη t. Οι αριθμοί δίπλα στις ακμές είναι οι χωρητικότητες.

Ορισμός της ροής Θα ορίσουμε τώρα τι σημαίνει για το δίκτυό μας η μεταφορά κίνησης, ή ροής. Λέμε ότι η *ροή s-t* (*s-t* flow) είναι μια συνάρτηση f που αντιστοιχίζει κάθε ακμή e σε έναν μη αρνητικό πραγματικό αριθμό f: $E \rightarrow \mathbf{R}^{+}$· η τιμή $f(e)$ αντιπροσωπεύει διαισθητικά την ποσότητα ροής που μεταφέρεται από την ακμή e. Η ροή f θα πρέπει να ικανοποιεί τις παρακάτω δύο ιδιότητες[1].

(i) (*Συνθήκες χωρητικότητας*) Για κάθε ακμή $e \in E$, έχουμε $0 \le f(e) \le c_e$.

(ii) (*Συνθήκες διατήρησης*) Για κάθε κόμβο v εκτός των s και t, έχουμε

$$\sum_{e \text{ προς το } v} f(e) = \sum_{e \text{ από το } v} f(e).$$

Εδώ το $\sum_{e \text{ προς το } v} f(e)$ αθροίζει τις τιμές ροής $f(e)$ για όλες τις ακμές που εισέρχονται στον κόμβο v, ενώ το $\sum_{e \text{ από το } v} f(e)$ είναι το άθροισμα των τιμών ροής για όλες τις ακμές που εξέρχονται από τον κόμβο v.

Άρα η ροή μιας ακμής δεν μπορεί να υπερβαίνει τη χωρητικότητα της ακμής. Για όλους τους άλλους κόμβους εκτός της προέλευσης και της απόληξης, η ποσότητα της ροής που εισέρχεται θα πρέπει να είναι ίση με την εξερχόμενη ροή. Η προέλευση δεν έχει εισερχόμενους κόμβους (με βάση την παραδοχή μας), αλλά μπορεί να έχει ροή που εξέρχεται από αυτή· με άλλα λόγια, μπορεί να παράγει ροή. Συμμετρικά, η απόληξη μπορεί να έχει μόνο εισερχόμενη ροή, όμως δεν έχει ακμές που εξέρχονται από αυτή. Η *τιμή* μιας ροής f, που συμβολίζεται ως $v(f)$, ορίζεται ως η ποσότητα της ροής που παράγεται στην προέλευση:

$$v(f) = \sum_{e \text{ από το } s} f(e).$$

Για να κάνουμε πιο συμπαγή τη σημειογραφία μας, ορίζουμε ότι $f^{out}(v) = \sum_{e \text{ από το } v} f(e)$ και $f^{in}(v) = \sum_{e \text{ προς το } v} f(e)$. Μπορούμε να το επεκτείνουμε αυτό σε σύνολα κόμβων· αν $S \subseteq V$, ορίζουμε ότι $f^{out}(S) = \sum_{e \text{ από το } S} f(e)$ και $f^{in}(S) = \sum_{e \text{ προς το } S} f(e)$. Με αυτή την ορολογία, η συνθήκη διατήρησης για τους κόμβους $v \neq s, t$ γίνεται $f^{in}(v) = f^{out}(v)$, και μπορούμε να γράψουμε ότι $v(f) = f^{out}(s)$.

Το πρόβλημα της Μέγιστης Ροής Με δεδομένο ένα δίκτυο ροής, ένας φυσικός στόχος είναι η διευθέτηση της κίνησης έτσι ώστε να γίνεται όσο το δυνατόν πιο αποδοτική χρήση της διαθέσιμης χωρητικότητας. Έτσι το βασικό αλγοριθμικό πρόβλημα που θα εξετάσουμε είναι το ακόλουθο: Με δεδομένο ένα δίκτυο ροής, βρείτε μια ροή με τη μέγιστη δυνατή τιμή.

Καθώς θα μελετάμε το σχεδιασμό αλγορίθμων για αυτό το πρόβλημα, είναι χρήσιμο να εξετάσουμε με ποιον τρόπο η δομή του δικτύου ροής θέτει άνω όρια ως προς τη μέγιστη τιμή μιας ροής *s-t*. Ένα βασικό "εμπόδιο" ως προς την ύπαρξη μεγάλων ροών είναι το εξής: Υποθέστε ότι διαιρούμε τους κόμβους του γραφήματος σε δύο σύνολα, A και B, έτσι ώστε $s \in A$ και $t \in B$. Έτσι, διαισθητικά, οποιαδήποτε ροή από το s

---

[1] Αυτή η έννοια της ροής που ορίσαμε μοντελοποιεί την κίνηση καθώς προχωρά μέσω του δικτύου με σταθερό ρυθμό. Έχουμε μία μόνο μεταβλητή $f(e)$ που συμβολίζει την ποσότητα της ροής στην ακμή e. Δεν μοντελοποιούμε την κίνηση *σε ριπές*, όπου η ροή μεταβάλλεται με το χρόνο.

προς το t θα πρέπει σε κάποιο σημείο να περνάει από το σύνολο A στο σύνολο B, άρα θα χρησιμοποιεί κάποια από τις χωρητικότητες ακμής από το A προς το B. Αυτό μας δείχνει ότι οποιαδήποτε τέτοια "διαίρεση" του γραφήματος θέτει ένα όριο ως προς τη μέγιστη δυνατή τιμή της ροής. Ο αλγόριθμος μέγιστης ροής που θα αναπτύξουμε εδώ θα είναι συνυφασμένος με μια απόδειξη ότι η τιμή της μέγιστης ροής είναι ίση με την ελάχιστη χωρητικότητα οποιασδήποτε τέτοιας διαίρεσης, που ονομάζεται *ελάχιστη αποκοπή* (minimum cut). Ως πρόσθετο πλεονέκτημα, ο αλγόριθμός μας θα υπολογίζει επίσης την ελάχιστη αποκοπή. Θα δούμε ότι το πρόβλημα της εύρεσης αποκοπών με ελάχιστη χωρητικότητα σε ένα δίκτυο ροής είναι εξίσου πολύτιμο, από την οπτική γωνία των εφαρμογών, με το πρόβλημα της εύρεσης της μέγιστης ροής.

Σχεδιασμός του αλγορίθμου

Υποθέστε ότι θέλουμε να βρούμε τη μέγιστη ροή σε ένα δίκτυο. Πώς θα πρέπει να προσπαθήσουμε να το πετύχουμε; Χρειάζονται κάποιες δοκιμές για να αποφασίσουμε ότι μια προσέγγιση όπως ο δυναμικός προγραμματισμός δεν φαίνεται να δουλεύει — τουλάχιστον, δεν υπάρχει κάποιος γνωστός αλγόριθμος για το πρόβλημα της Μέγιστης Ροής ο οποίος να μπορεί να θεωρηθεί ότι ανήκει στη μέθοδο του δυναμικού προγραμματισμού. Αφού δεν υπάρχουν άλλες ιδέες, θα μπορούσαμε να επιστρέψουμε στις απλές άπληστες προσεγγίσεις και να δούμε σε ποιο σημείο αποτυγχάνουν.

Υποθέστε ότι ξεκινάμε με μηδενική ροή: $f(e) = 0$ για όλα τα e. Είναι προφανές ότι αυτό είναι συνεπές με τις συνθήκες χωρητικότητας και διατήρησης· το πρόβλημα είναι ότι η τιμή είναι 0. Θα προσπαθήσουμε τώρα να αυξήσουμε την τιμή του f "προωθώντας" ροή κατά μήκος μιας διαδρομής από το s προς το t, μέχρι τα όρια τα οποία θέτουν οι χωρητικότητες των ακμών. Έτσι, στην Εικόνα 7.3 θα μπορούσαμε να επιλέξουμε τη διαδρομή που αποτελείται από τις ακμές $\{(s, u), (u, v), (v, t)\}$ και να αυξήσουμε τη ροή σε καθεμία από αυτές τις ακμές σε 20, αφήνοντας $f(e) = 0$ για τις άλλες δύο ακμές. Με αυτόν τον τρόπο εξακολουθούμε να μην παραβιάζουμε τις συνθήκες χωρητικότητας — αφού ορίσαμε τη ροή σε ένα ύψος το οποίο επιτρέπουν οι χωρητικότητες των ακμών — και τις συνθήκες διατήρησης — αφού όταν αυξάνουμε τη ροή σε μια ακμή που μπαίνει σε έναν εσωτερικό κόμβο, την αυξάνουμε επίσης και για την ακμή που βγαίνει από τον κόμβο. Τώρα η τιμή της ροής μας είναι 20, και μπορούμε να ρωτήσουμε: Είναι αυτή η μέγιστη δυνατή ροή για το γράφημα που φαίνεται στην εικόνα; Αν το σκεφτούμε, θα δούμε ότι η απάντηση είναι αρνητική επειδή μπορούμε να κατασκευάσουμε μια ροή με τιμή 30. Το πρόβλημα είναι ότι εξακολουθούμε να είμαστε κολλημένοι — δεν υπάρχει καμία διαδρομή s-t στην οποία να μπορούμε να προωθήσουμε άμεσα ροή χωρίς να υπερβούμε κάποια χωρητικότητα — και παρόλα αυτά δεν έχουμε τη μέγιστη ροή. Αυτό που χρειαζόμαστε είναι ένας πιο γενικός τρόπος για την προώθηση ροής από το s προς το t, έτσι ώστε, σε μια περίπτωση σαν αυτή, να έχουμε τρόπο αύξησης της τιμής της τρέχουσας ροής.

Ουσιαστικά, θα θέλαμε να εκτελέσουμε την ακόλουθη λειτουργία, που φαίνεται με διακεκομμένη γραμμή στην Εικόνα 7.3(γ). Προωθούμε 10 μονάδες ροής κατά μήκος της ακμής (s, v)· αυτό έχει αποτέλεσμα να εισέρχεται υπερβολικά πολλή ροή στον κόμβο v. Έτσι "αναιρούμε" 10 μονάδες ροής από την ακμή (u, v)· αυτό επαναφέρει σε ισχύ τη συνθήκη διατήρησης για τον κόμβο v, οδηγεί όμως σε έξοδο πολύ λίγης ροής από τον κόμβο u. Έτσι τελικά προωθούμε 10 μονάδες ροής κατά μήκος της ακμής (u, t), επαναφέροντας σε ισχύ τη συνθήκη διατήρησης στον κόμβο u. Έχουμε τώρα μια έγκυρη ροή της οποίας η τιμή είναι 30. Δείτε σχετικά την Εικόνα 7.3, όπου οι έντονες ακμές μεταφέρουν ροή πριν από τη λειτουργία αυτή και οι διακεκομμένες ακμές σχηματίζουν αυτό το νέο είδος επαύξησης.

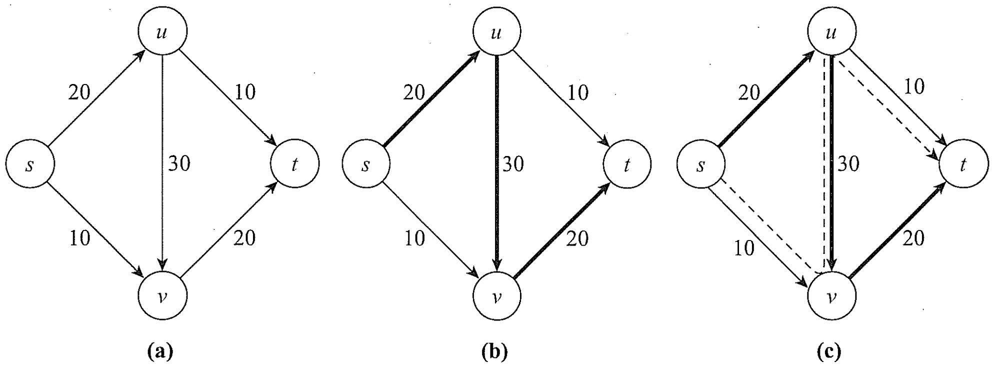
Αυτός είναι ένας γενικότερος τρόπος για την προώθηση ροής. Μπορούμε να προωθούμε *προς τα εμπρός (ευθύδρομα)* σε ακμές που έχουν περίσσευμα χωρητικότητας και να προωθούμε *προς τα πίσω (ανάδρομα)* σε ακμές που μεταφέρουν ήδη ροή, έτσι ώστε να την εκτρέψουμε σε διαφορετική κατεύθυνση. Μπορούμε να ορίσουμε τώρα το *υπολειπόμενο γράφημα* (residual graph), το οποίο παρέχει ένα συστηματικό τρόπο για την αναζήτηση ευθύδρομων και ανάδρομων λειτουργιών όπως αυτή.

Το υπολειπόμενο γράφημα Με δεδομένο ένα γράφημα ροής G και μια ροή f στο G, ορίζουμε το υπολειπόμενο γράφημα G_f του G ως προς τη ροή f με τον εξής τρόπο. (Στην Εικόνα 7.4 μπορείτε να δείτε το υπολειπόμενο γράφημα για τη ροή της Εικόνας 7.3 μετά την προώθηση 20 μονάδων ροής κατά μήκος της διαδρομής s, u, v, t.)

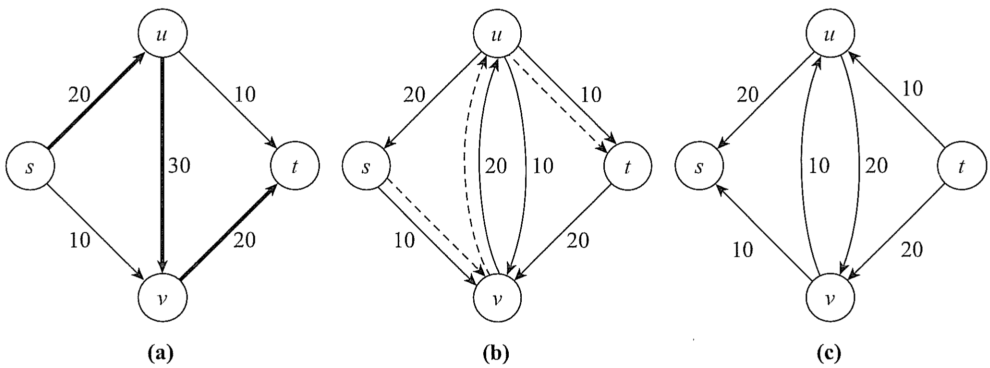
- Το σύνολο κόμβων του G_f είναι ίδιο με εκείνο του G.
- Για κάθε ακμή $e = (u, v)$ του G στην οποία $f(e) < c_e$, υπάρχουν $c_e - f(e)$ "υπολειπόμενες" μονάδες χωρητικότητες για τις οποίες θα μπορούσαμε να προσπαθήσουμε να προωθήσουμε ροή προς τα εμπρός. Έτσι συμπεριλαμβάνουμε την ακμή $e = (u, v)$ στο G_f με χωρητικότητα $c_e - f(e)$. Θα αποκαλούμε αυτές τις ακμές ευθύδρομες (forward edges).
- Για κάθε ακμή $e = (u, v)$ του G στην οποία $f(e) > 0$, υπάρχουν $f(e)$ μονάδες ροής τις οποίες μπορούμε να "αναιρέσουμε" αν το θέλουμε, προωθώντας τη ροή προς τα πίσω. Έτσι συμπεριλαμβάνουμε την ακμή $e' = (v, u)$ στο G_f με χωρητικότητα $f(e)$. Σημειώστε ότι η ακμή e' έχει ίδια άκρα με την e αλλά με αντίστροφη διεύθυνση· θα αποκαλούμε αυτές τις ακμές ανάδρομες (backward edges).

Αυτό ολοκληρώνει τον ορισμό για το υπολειπόμενο γράφημα G_f. Σημειώστε ότι η κάθε ακμή e του G μπορεί να δημιουργήσει μία ή δύο ακμές στο γράφημα G_f: Αν $0 < f(e) < c_e$ τότε θα συμπεριληφθεί στο G_f και μια ευθύδρομη ακμή και μια ανάδρομη ακμή. Άρα το G_f έχει το πολύ διπλάσιες ακμές από το G. Θα χρησιμοποιούμε μερικές φορές τον όρο *υπολειπόμενη χωρητικότητα* (residual capacity) όταν αναφερόμαστε στη χωρητικότητα μιας ακμής στο υπολειπόμενο γράφημα, έτσι ώστε να τη διακρίνουμε από τη χωρητικότητα της αντίστοιχης ακμής στο αρχικό δίκτυο ροής G.

Διαδρομές επαύξησης σε ένα υπολειπόμενο γράφημα Τώρα μπορούμε να διατυπώσουμε με ακρίβεια τον τρόπο με τον οποίο προωθούμε ροή από το s προς το t στο γράφημα G_f. Έστω ότι η P είναι μια απλή διαδρομή s-t στο G_f — με άλλα λόγια, η P δεν επισκέπτεται κανέναν κόμβο περισσότερες από μία φορές. Ορίζουμε ότι η `bottleneck` (P, f) είναι η ελάχιστη υπολειπόμενη χωρητικότητα οποιασδήποτε ακμής της P, σε σχέση με τη ροή f. Ορίζουμε τώρα την ακόλουθη λειτουργία augment(f, P), η οποία παράγει μια νέα ροή f' στο G.



Εικόνα 7.3 (α) Το δίκτυο της Εικόνας 7.2. (β) Προώθηση 20 μονάδων ροής κατά μήκος της διαδρομής s, u, v, t. (γ) Το νέο είδος της διαδρομής επαύξησης με χρήση της ακμής (u, v) προς τα πίσω.



Εικόνα 7.4 (α) Το γράφημα G με τη διαδρομή s, u, v, t που χρησιμοποιήθηκε για την προώθηση 20 μονάδων ροής. (β) Το υπολειπόμενο γράφημα για την προκύπτουσα ροή f, με την υπολειπόμενη χωρητικότητα δίπλα σε κάθε ακμή. Η διακεκομμένη γραμμή είναι η νέα διαδρομή επαύξησης. (γ) Το υπολειπόμενο γράφημα μετά την προώθηση 10 πρόσθετων μονάδων ροής κατά μήκος της νέας διαδρομής επαύξησης s, v, u, t.

```
augment(f, P)
  Έστω ότι b = bottleneck(P, f)
  For όλες τις ακμές (u, v) ∈ P
    If e = (u, v) είναι ευθύδρομη ακμή then
      αύξησε το f(e) στο G κατά b
    Else (το (u, v) είναι ανάδρομη ακμή, και έστω e=(v, u))
      μείωσε το f(e) στο G κατά b
    Endif
  Endfor
  Return(f)
```

Η εκτέλεση αυτής της λειτουργίας ήταν ο μόνος λόγος για τον ορισμό του υπολειπόμενου γραφήματος· για να δείξουμε τη σημασία της λειτουργίας `augment` (που σημαίνει "επαυξάνω"), σκεφθείτε ότι πολλοί συχνά αναφέρονται σε οποιαδήποτε διαδρομή s-t στο υπολειπόμενο γράφημα ως *διαδρομή επαύξησης* (augmenting path).

Το αποτέλεσμα της `augment`(f, P) είναι μια νέα ροή f' στο G, η οποία λαμβάνεται με αύξηση και μείωση των τιμών ροής στις ακμές της διαδρομής P. Ας επιβεβαιώσουμε καταρχήν ότι η f' είναι πράγματι μια ροή.

(7.1) *Η f' είναι μια ροή στο G.*

Απόδειξη. Θα πρέπει να επαληθεύσουμε τις συνθήκες χωρητικότητας και διατήρησης.

Αφού η f' διαφέρει από την f μόνο στις ακμές της διαδρομής P, χρειάζεται να ελέγξουμε τις συνθήκες χωρητικότητας μόνο σε αυτές τις ακμές. Έτσι, έστω ότι η (u,v) είναι μια ακμή στην P. Άτυπα μπορούμε να πούμε ότι η συνθήκη χωρητικότητας εξακολουθεί να ισχύει, επειδή αν η $e = (u, v)$ είναι ευθύδρομη ακμή αποφύγαμε ρητά να αυξήσουμε τη ροή της e σε τιμή πάνω από c_e· ενώ αν η (u, v) είναι μια ανάδρομη ακμή που προκύπτει από την ακμή $e = (v, u) \in E$, αποφύγαμε ρητά να μειώσουμε τη ροή της e σε τιμή μικρότερη από 0. Πιο συγκεκριμένα, σημειώστε ότι η `bottleneck`(P,f) δεν είναι μεγαλύτερη από την υπολειπόμενη χωρητικότητα της (u, v). Αν η $e = (u, v)$ είναι μια ευθύδρομη ακμή, τότε η υπολειπόμενη χωρητικότητά της είναι $c_e - f(e)$· έτσι έχουμε

$$0 \le f(e) \le f'(e) = f(e) + \texttt{bottleneck}(P,f) \le f(e) + (c_e - f(e)) = c_e,$$

άρα εξακολουθεί να ισχύει η συνθήκη χωρητικότητας. Αν η (u, v) είναι μια ανάδρομη ακμή που προκύπτει από την ακμή $e = (v, u) \in E$, τότε η υπολειπόμενη χωρητικότητά της είναι $f(e)$ και έτσι έχουμε

$$c_e \ge f(e) \ge f'(e) = f(e) - \texttt{bottleneck}(P,f) \ge f(e) - f(e) = 0$$

άρα ισχύει και πάλι η συνθήκη χωρητικότητας.

Θα πρέπει επίσης να ελέγξουμε τη συνθήκη διατήρησης για κάθε εσωτερικό κόμβο που βρίσκεται στη διαδρομή P. Έστω ότι ο v είναι ένας τέτοιος κόμβος· μπορούμε να επαληθεύσουμε ότι η μεταβολή στην ποσότητα ροής που εισέρχεται στον κόμβο v είναι ίδια με τη μεταβολή στην ποσότητα ροής που εξέρχεται από τον κόμβο v· αφού η f ικανοποιούσε τη συνθήκη διατήρησης για τον κόμβο v, θα πρέπει να ισχύει το ίδιο και για την f'. Τεχνικά, υπάρχουν τέσσερις περιπτώσεις που πρέπει να ελέγξουμε, ανάλογα με τον αν η ακμή της διαδρομής P που εισέρχεται στον κόμβο v είναι ευθύδρομη ακμή ή ανάδρομη ακμή και αν η ακμή της P που εξέρχεται από τον κόμβο v είναι ευθύδρομη ακμή ή ανάδρομη ακμή. Παρόλα αυτά, ο χειρισμός αυτών των περιπτώσεων είναι εύκολος και τον αφήνουμε ως άσκηση για τους αναγνώστες. ∎

Αυτή η λειτουργία επαύξησης αποτυπώνει τη διαδικασία ευθύδρομης και ανάδρομης προώθησης ροής που αναφέραμε προηγουμένως. Ας εξετάσουμε τώρα τον παρακάτω αλγόριθμο για τον υπολογισμό μιας ροής *s-t* στο *G*.

```
Max-Flow
  Αρχικά f(e) = 0 για όλα τα e του G
  While υπάρχει μια διαδρομή s-t στο υπολειπόμενο γράφημα G_f
    Έστω ότι η P είναι μια απλή διαδρομή s-t στο G_f
    f' = augment(f, P)
    Ενημέρωσε το f σε f'
    Ενημέρωσε το υπολειπόμενο γράφημα G_f σε G_f'
  Endwhile
  Return f
```

Θα ονομάσουμε αυτόν τον αλγόριθμο *Αλγόριθμο Ford-Fulkerson* (Ford-Fulkerson Algorithm), από τα ονόματα των δύο ερευνητών που τον ανέπτυξαν το 1956. Στην Εικόνα 7.4 θα δείτε μια εκτέλεση του αλγορίθμου. Ο αλγόριθμος Ford-Fulkerson είναι πραγματικά αρκετά απλός. Αυτό που δεν είναι καθόλου ξεκάθαρο είναι το αν ο κεντρικός βρόχος `While` τερματίζει, και το αν η ροή που επιστρέφεται είναι μια μέγιστη ροή. Οι απαντήσεις και στις δύο αυτές ερωτήσεις αποδεικνύεται ότι είναι αρκετά έξυπνες.

Ανάλυση του αλγορίθμου: Τερματισμός και χρόνος εκτέλεσης

Θα εξετάσουμε τώρα, με επαγωγή ως προς το πλήθος επαναλήψεων του βρόχου `While`, κάποιες από τις ιδιότητες που διατηρεί ο αλγόριθμός μας με βάση την παραδοχή μας ότι όλες οι χωρητικότητες είναι ακέραιοι αριθμοί.

(7.2) *Σε κάθε ενδιάμεσο στάδιο του αλγορίθμου Ford-Fulkerson, οι τιμές της ροής* $\{f(e)\}$ *και οι υπολειπόμενες χωρητικότητες στο* G_f *είναι ακέραιοι αριθμοί.*

Απόδειξη. Αυτό είναι προφανώς αληθές πριν από την έναρξη των επαναλήψεων στο βρόχο `While`. Ας υποθέσουμε τώρα ότι είναι αληθές μετά από j επαναλήψεις. Τότε, αφού όλες οι υπολειπόμενες χωρητικότητες στο G_f είναι ακέραιες, η τιμή `bottleneck`(P, f) για τη διαδρομή επαύξησης που βρέθηκε στην επανάληψη $j + 1$ θα είναι ακέραια. Άρα η ροή f' θα έχει ακέραιες τιμές, και το ίδιο θα ισχύει και για τις χωρητικότητες του νέου υπολειπόμενου γραφήματος. ■

Μπορούμε να χρησιμοποιήσουμε αυτή την ιδιότητα για να αποδείξουμε ότι ο αλγόριθμος Ford-Fulkerson τερματίζει. Όπως και σε προηγούμενα σημεία αυτού του βιβλίου, θα ψάξουμε για ένα μέτρο της *προόδου* που υποδηλώνει τερματισμό.

Καταρχήν θα δείξουμε ότι η τιμή της ροής είναι γνησίως αύξουσα όταν εφαρμόζουμε επαύξηση.

(7.3) *Έστω ότι* f *είναι μια ροή στο* G*, και έστω ότι* P *είναι μια απλή διαδρομή s-t στο* G_f*. Τότε* $\nu(f') = \nu(f) +$ `bottleneck`(P,f)*· και αφού* `bottleneck`$(P,f) > 0$*, έχουμε* $\nu(f') > \nu(f)$.

Απόδειξη. Η πρώτη ακμή e στη διαδρομή P θα πρέπει να είναι μια ακμή που εξέρχεται από το s στο υπολειπόμενο γράφημα G_f και αφού η διαδρομή είναι απλή, δεν έχουμε άλλη επίσκεψη στο s. Αφού το G δεν έχει ακμές που εισέρχονται στον κόμβο s, η ακμή e θα πρέπει να είναι ευθύδρομη ακμή. Αυξάνουμε τη ροή αυτής της ακμής κατά `bottleneck`(P,f) και δεν αλλάζουμε τη ροή καμίας άλλης ακμής που συνδέεται με το s. Κατά συνέπεια η τιμή της f' υπερβαίνει την τιμή της f κατά `bottleneck` (P,f). ■

Χρειαζόμαστε μία ακόμα παρατήρηση για να αποδείξουμε τον τερματισμό: Χρειαζόμαστε να μπορέσουμε να θέσουμε ένα άνω όριο ως προς τη μέγιστη δυνατή τιμή της ροής. Να ένα άνω όριο: Αν όλες οι ακμές που εξέρχονται από το s μπορούσαν να κορεστούν πλήρως με ροή, η τιμή της ροής θα ήταν $\sum_{e \text{ από το } s} c_e$ Έστω ότι το C συμβολίζει αυτό το άθροισμα. Έτσι έχουμε $\nu(f) \leq C$ για όλες τις ροές s-t f. (Το C μπορεί να αποτελεί εκπληκτικά μεγάλη υπερεκτίμηση ως προς τη μέγιστη τιμή της ροής στο G, αλλά είναι βολικό ως πεπερασμένο και εύκολα διατυπωνόμενο όριο.) Χρησιμοποιώντας την (7.3), μπορούμε τώρα να αποδείξουμε τον τερματισμό.

(7.4) *Υποθέστε, όπως παραπάνω, ότι όλες οι χωρητικότητες σε ένα δίκτυο ροής G είναι ακέραιες. Στην περίπτωση αυτή ο αλγόριθμος Ford-Fulkerson τερματίζει μετά από το πολύ C επαναλήψεις του βρόχου* `While`.

Απόδειξη. Είδαμε προηγουμένως ότι καμία ροή στο G δεν μπορεί να έχει τιμή μεγαλύτερη από C, εξαιτίας της συνθήκης χωρητικότητας στις ακμές που εξέρχονται από το s. Τώρα, με βάση την (7.3), η τιμή της ροής που διατηρείται από τον αλγόριθμο Ford-Fulkerson αυξάνεται σε κάθε επανάληψη· έτσι με βάση την (7.2) θα αυξάνεται κατά τουλάχιστον 1 σε κάθε επανάληψη. Αφού ξεκινά από την τιμή 0 και δεν μπορεί να γίνει μεγαλύτερη από C, ο βρόχος `While` στον αλγόριθμο Ford-Fulkerson μπορεί να εκτελείται για το πολύ C επαναλήψεις. ■

Θα εξετάσουμε τώρα το χρόνο εκτέλεσης για τον αλγόριθμο Ford-Fulkerson. Έστω ότι το n συμβολίζει το πλήθος των κόμβων στο G και το m συμβολίζει το πλήθος των ακμών στο G. Έχουμε υποθέσει ότι όλοι οι κόμβοι έχουν τουλάχιστον μία προσκείμενη ακμή, έτσι $m \geq n/2$, άρα μπορούμε να χρησιμοποιήσουμε τη σχέση $O(m+n) = O(m)$ για να απλουστεύσουμε τα όριά μας.

(7.5) *Υποθέστε, όπως προηγουμένως, ότι όλες οι χωρητικότητες στο δίκτυο ροής G είναι ακέραιοι αριθμοί. Στην περίπτωση αυτή ο αλγόριθμος Ford-Fulkerson μπορεί να υλοποιηθεί έτσι ώστε να εκτελείται σε χρόνο $O(mC)$.*

Απόδειξη. Γνωρίζουμε από την (7.4) ότι ο αλγόριθμος τερματίζει σε C το πολύ επαναλήψεις του βρόχου `While`. Έτσι θα εξετάσουμε την ποσότητα εργασίας του αλγορίθμου που περιλαμβάνεται σε μία επανάληψη όταν η τρέχουσα ροή είναι f.

Το υπολειπόμενο γράφημα G_f έχει το πολύ $2m$ ακμές, αφού η κάθε ακμή του G δημιουργεί μέχρι δύο ακμές στο υπολειπόμενο γράφημα. Θα διατηρούμε το G_f χρησιμοποιώντας μια αναπαράσταση λίστας γειτονικότητας· θα έχουμε δύο συνδεδεμένες λίστες για κάθε κόμβο v, μία με τις ακμές που εισέρχονται στο v και μία με τις ακμές που εξέρχονται από το v. Για να βρούμε μια διαδρομή s-t στο G_f μπορούμε να χρησιμοποιήσουμε είτε αναζήτηση πρώτα κατά πλάτος είτε αναζήτηση πρώτα κατά βάθος, που εκτελούνται σε χρόνο $O(m + n)$· αφού όμως $m \geq n/2$, το $O(m + n)$ είναι ίδιο με το $O(m)$. Η διαδικασία `augment(f,P)` απαιτεί χρόνο $O(n)$, αφού η διαδρομή P έχει το πολύ $n - 1$ ακμές. Με δεδομένη τη νέα ροή f', μπορούμε να δημιουργήσουμε το νέο υπολειπόμενο γράφημα σε χρόνο $O(m)$: Για κάθε ακμή e του G κατασκευάζουμε τις σωστές ευθύδρομες και ανάδρομες ακμές στο $G_{f'}$. ∎

Μια κάπως πιο αποδοτική εκδοχή του αλγορίθμου θα διατηρούσε τις συνδεδεμένες λίστες των ακμών στο υπολειπόμενο γράφημα G_f ως τμήμα της διαδικασίας `augment` που τροποποιεί τη ροή f μέσω επαύξησης.

7.2 Μέγιστες ροές και ελάχιστες αποκοπές σε ένα δίκτυο

Σε ολόκληρη αυτή την ενότητα θα συνεχίσουμε τώρα με την ανάλυση του αλγορίθμου Ford-Fulkerson. Στην πορεία δεν θα μάθουμε μόνο πολλά πράγματα σχετικά με τον αλγόριθμο, αλλά θα δούμε επίσης και ότι η ανάλυση του αλγορίθμου μας παρέχει σημαντικές γνώσεις σχετικά με το ίδιο το πρόβλημα της Μέγιστης Ροής.

Ανάλυση του αλγορίθμου: Ροές και αποκοπές

Ο επόμενος στόχος μας είναι να δείξουμε ότι η ροή που επιστρέφεται από τον αλγόριθμο Ford-Fulkerson έχει τη μέγιστη δυνατή τιμή από οποιαδήποτε ροή στο G. Για να προχωρήσουμε προς αυτόν το στόχο, θα επιστρέψουμε σε ένα ζήτημα το οποίο θέσαμε στην Ενότητα 7.1: τον τρόπο με τον οποίο η δομή του δικτύου ροής θέτει άνω όρια ως προς τη μέγιστη τιμή μιας ροής s-t. Έχουμε ήδη δει ένα άνω όριο: η τιμή $v(f)$ για οποιαδήποτε ροή s-t f είναι το πολύ ίση με $C = \sum_{e \text{ από το } s} c_e$. Κάποιες φορές αυτό το όριο είναι χρήσιμο, όμως μερικές φορές είναι πολύ αδύναμο. Θα χρησιμοποιήσουμε τώρα την έννοια της *αποκοπής* (cut) για να αναπτύξουμε έναν πολύ πιο γενικό τρόπο καθορισμού άνω ορίων ως προς την τιμή της μέγιστης ροής.

Θεωρήστε ότι διαιρούμε τους κόμβους του γραφήματος σε δύο σύνολα, A και B, έτσι ώστε $s \in A$ και $t \in B$. Όπως είδαμε στην Ενότητα 7.1, οποιαδήποτε τέτοια διαίρεση θέτει ένα άνω όριο ως προς τη μέγιστη δυνατή τιμή ροής, αφού όλη η ροή θα πρέπει σε κάποιο σημείο να "περάσει" από το A στο B. Για να το διατυπώσουμε τυπικά, λέμε ότι μια *αποκοπή s-t* είναι μια διαμέριση (A, B) του συνόλου κορυφών V έτσι ώστε $s \in A$ και $t \in B$. Η *χωρητικότητα* (capacity) μιας αποκοπής (A, B), την οποία θα συμβολίζουμε με το $c(A, B)$, είναι απλώς το άθροισμα των χωρητικοτήτων για όλες τις ακμές που εξέρχονται από το A: $c(A, B) = \sum_{e \text{ από το } A} c_e$.

Οι αποκοπές αποδεικνύεται ότι παρέχουν πολύ φυσικά άνω όρια ως προς τις τιμές των ροών, όπως είδαμε από την προηγούμενη διαισθητική παρατήρηση. Θα το κάνουμε αυτό πιο ακριβές μέσα από μια σειρά από ισχυρισμούς.

(7.6) *Έστω ότι η f είναι κάποια ροή s-t και (A, B) είναι κάποια αποκοπή s-t. Τότε* $\nu(f) = f^{out}(A) - f^{in}(A)$.

Αυτή η πρόταση είναι στην πραγματικότητα πολύ πιο ισχυρή από ένα απλό άνω όριο. Μας λέει ότι, αν παρακολουθούμε την ποσότητα ροής f που στέλνεται διαμέσου μιας αποκοπής, μπορούμε να *μετρήσουμε* επακριβώς την τιμή της ροής: Είναι η συνολική ποσότητα που βγαίνει από το A μείον την ποσότητα που "επιστρέφει" στο A. Διαισθητικά αυτό φαίνεται λογικό, αν και η απόδειξη απαιτεί κάποιο χειρισμό αθροισμάτων.

Απόδειξη. Από τον ορισμό έχουμε $\nu(f) = f^{out}(s)$. Με βάση τις παραδοχές μας έχουμε $f^{in}(s) = 0$, αφού η προέλευση s δεν έχει εισερχόμενες ακμές, άρα μπορούμε να γράψουμε $\nu(f) = f^{out}(s) - f^{in}(s)$. Αφού όλες οι άλλες κορυφές ν του A εκτός από την s είναι εσωτερικές, γνωρίζουμε ότι $f^{out}(\nu) - f^{in}(\nu) = 0$ για όλες αυτές τις κορυφές. Έτσι

$$\nu(f) = \sum_{\nu \in A} (f^{out}(\nu) - f^{in}(\nu)),$$

αφού ο μόνος όρος αυτού του αθροίσματος που είναι μη μηδενικός είναι εκείνος όπου το ν τίθεται σε s.

Ας προσπαθήσουμε να αναδιατυπώσουμε τώρα το άθροισμα στο δεξιό σκέλος της εξίσωσης. Αν μια ακμή e έχει και τα δύο άκρα της στο A, τότε η $f(e)$ εμφανίζεται στο άθροισμα μία φορά με πρόσημο "+" και μία φορά με πρόσημο "–", και έτσι αυτοί οι δύο όροι αλληλοαναιρούνται. Αν η e έχει μόνο την αρχή της στο A, τότε η $f(e)$ εμφανίζεται μόνο μία φορά στο άθροισμα με πρόσημο "+". Αν η e έχει μόνο το τέλος της στο A, τότε η $f(e)$ εμφανίζεται επίσης μόνο μία φορά στο άθροισμα, με πρόσημο "–". Τέλος, αν η e δεν έχει κανένα άκρο της στο A, τότε το $f(e)$ δεν εμφανίζεται καθόλου στο άθροισμα. Με βάση όλα αυτά, έχουμε

$$\sum_{\nu \in A} f^{out}(\nu) - f^{in}(\nu) = \sum_{e \text{ από το } A} f(e) - \sum_{e \text{ προς το } A} f(e) = f^{out}(A) - f^{in}(A)$$

Συνδυάζοντας αυτές τις δύο εξισώσεις έχουμε την (7.6). ■

Αν $A=\{s\}$, τότε $f^{out}(A) = f^{out}(s)$ και $f^{in}(A) = 0$ αφού με βάση την παραδοχή μας δεν υπάρχουν ακμές που να εισέρχονται στην προέλευση. Έτσι η πρόταση αυτή για το σύνολο $A = \{s\}$ είναι ακριβώς ο ορισμός της τιμής ροής $\nu(f)$.

Σημειώστε ότι, αν το (A, B) είναι αποκοπή, τότε οι ακμές που εισέρχονται στο B είναι ακριβώς οι ακμές που εξέρχονται από το A. Παρομοίως, οι ακμές που εξέρχονται από το B είναι ακριβώς οι ακμές που εισέρχονται στο A. Έτσι έχουμε $f^{out}(A) = f^{in}(B)$ και $f^{in}(A) = f^{out}(B)$ από τον ορισμό αυτών των παραστάσεων. Άρα μπορούμε να αναδιατυπώσουμε την εξίσωση (7.6) με τον ακόλουθο τρόπο.

(7.7) *Έστω f κάποια ροή s-t και (A, B) κάποια αποκοπή s-t. Τότε* $v(f) = f^{in}(B) - f^{out}(B)$.

Αν θέσουμε $A = V - \{t\}$ και $B = \{t\}$ στην εξίσωση (7.7), έχουμε ότι $v(f) = f^{in}(B) - f^{out}(B) = f^{in}(t) - f^{out}(t)$. Με βάση την παραδοχή μας ότι η απόληξη t δεν έχει εξερχόμενες ακμές, έχουμε ότι $f^{out}(t) = 0$. Αυτό μας δείχνει ότι θα μπορούσαμε εξίσου καλά να έχουμε εξαρχής ορίσει την *τιμή* της ροής με βάση την απόληξη t: είναι ίση με $f^{in}(t)$, δηλαδή η ποσότητα ροής που φτάνει στην απόληξη.

Μια πολύ χρήσιμη συνέπεια της (7.6) είναι το ακόλουθο άνω όριο.

(7.8) *Έστω f κάποια ροή s-t και (A, B) κάποια αποκοπή s-t. Τότε* $v(f) \leq c(A, B)$.

Απόδειξη.

$$\begin{aligned} v(f) &= f^{out}(A) - f^{in}(A) \\ &\leq f^{out}(A) \\ &= \sum_{e \text{ από το } A} f(e) \\ &\leq \sum_{e \text{ από το } A} c_e \\ &= c(A, B) \end{aligned}$$

Εδώ η πρώτη γραμμή είναι απλώς η εξίσωση (7.6)· προχωράμε από την πρώτη γραμμή στη δεύτερη επειδή $f^{in}(A) \geq 0$, και προχωράμε από την τρίτη γραμμή στην τέταρτη με εφαρμογή των συνθηκών χωρητικότητας σε καθέναν από τους όρους του αθροίσματος. ■

Κατά μία έννοια η (7.8) φαίνεται πιο "ασθενής" από την (7.6), επειδή είναι απλώς μια ανισότητα αντί για ισότητα. Παρόλα αυτά, θα μας φανεί εξαιρετικά χρήσιμη επειδή η δεξιά πλευρά της είναι ανεξάρτητη από οποιαδήποτε συγκεκριμένη ροή f. Αυτό που μας λέει η (7.8) είναι ότι *στην τιμή κάθε ροής τίθεται ένα άνω όριο με βάση τη χωρητικότητα κάθε αποκοπής*. Με άλλα λόγια, αν συναντήσουμε κάποια αποκοπή s-t στο G με τιμή c^*, γνωρίζουμε αμέσως ότι δεν μπορεί να υπάρχει στο G κάποια ροή s-t με τιμή μεγαλύτερη από c^*. Αντιστρόφως, αν συναντήσουμε στο G κάποια ροή s-t με τιμή v^*, γνωρίζουμε αμέσως από την (7.8) ότι δεν μπορεί να υπάρχει στο G κάποια αποκοπή s-t με τιμή μικρότερη από v^*.

Ανάλυση του αλγορίθμου: Η μέγιστη ροή ισούται με την ελάχιστη αποκοπή

Έστω ότι το $\bar{f}$ συμβολίζει τη ροή που επιστρέφεται από τον αλγόριθμο Ford-Fulkerson. Θέλουμε να δείξουμε ότι η $\bar{f}$ έχει τη μέγιστη δυνατή τιμή από οποιαδήποτε ροή στο G, και θα το επιτύχουμε αυτό με τη μέθοδο που αναφέραμε παραπάνω: θα βρούμε μια αποκοπή s-t (A^*, B^*) για την οποία $v(\bar{f}) = c(A^*, B^*)$. Αυτό αμέσως αποδεικνύει ότι η $\bar{f}$ έχει τη μέγιστη τιμή ροής και ότι η (A^*, B^*) έχει την ελάχιστη χωρητικότητα από οποιαδήποτε αποκοπή *s-t*.

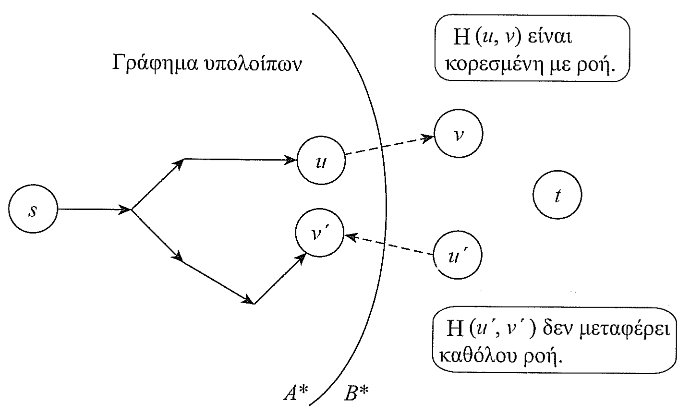
Ο αλγόριθμος Ford-Fulkerson τερματίζει όταν η ροή *f* δεν έχει διαδρομή *s-t* στο υπολειπόμενο γράφημα G_f. Αποδεικνύεται ότι αυτή είναι η μόνη ιδιότητα που απαιτείται προκειμένου να αποδείξουμε ότι επιστρέφει τη μέγιστη τιμή.

(7.9) *Αν η f είναι μια ροή s-t τέτοια ώστε να μην υπάρχει διαδρομή s-t στο υπολειπόμενο γράφημα G_f, τότε υπάρχει μια αποκοπή s-t (A^*, B^*) στο* G *για την οποία $v(f) = c\,(A^*, B^*)$. Κατά συνέπεια, η f έχει τη μέγιστη τιμή από όλες τις ροές του G και η (A^*, B^*) έχει την ελάχιστη χωρητικότητα από όλες τις αποκοπές s-t του G.*

Απόδειξη. Αυτή η πρόταση ισχυρίζεται την ύπαρξη μιας αποκοπής που ικανοποιεί μια επιθυμητή ιδιότητα· έτσι θα πρέπει τώρα να προσδιορίσουμε μια τέτοια αποκοπή. Για να το επιτύχουμε αυτό, έστω ότι το A^* συμβολίζει το σύνολο όλων των κόμβων *v* του *G* για τους οποίους υπάρχει μια διαδρομή *s-v* στο G_f. Έστω ότι το B^* συμβολίζει το σύνολο όλων των υπόλοιπων κόμβων: $B^* = V - A^*$.

Καταρχήν θα δείξουμε ότι το (A^*, B^*) είναι πράγματι μια αποκοπή *s-t*. Είναι σαφές ότι πρόκειται για μια διαμέριση του *V*. Η προέλευση *s* ανήκει στο A^*, επειδή υπάρχει πάντα μια διαδρομή από το *s* μέχρι το *s*. Επιπρόσθετα, $t \notin A^*$ με βάση την παραδοχή ότι δεν υπάρχει διαδρομή *s-t* στο υπολειπόμενο γράφημα· άρα $t \in B^*$ όπως ήταν το επιθυμητό.

Στη συνέχεια, υποθέστε ότι $e = (u, v)$ είναι μια ακμή του *G* για την οποία $u \in A^*$ και $v \in B^*$, όπως φαίνεται στην Εικόνα 7.5. Ισχυριζόμαστε ότι $f(e) = c_e$. Αν δεν συνέβαινε αυτό το *e* θα ήταν μια ευθύδρομη ακμή στο υπολειπόμενο γράφημα G_f, και αφού $u \in A^*$ υπάρχει μια διαδρομή *s-u* στο G_f προσαρτώντας την *e* σε αυτή τη διαδρομή θα παίρναμε μια διαδρομή *s-v* στο G_f, γεγονός που έρχεται σε αντίφαση με την υπόθεσή μας ότι $v \in B^*$.



Εικόνα 7.5 Η αποκοπή (A^*, B^*) για την απόδειξη της (7.9).

Υποθέστε τώρα ότι $e' = (u', v')$ είναι μια ακμή του *G* για την οποία $u' \in B^*$ και $v' \in A^*$. Ισχυριζόμαστε ότι $f(e') = 0$. Αν δεν συνέβαινε αυτό, η e' θα προκαλούσε μια ανάδρομη ακμή $e'' = (v', u')$ στο υπολειπόμενο γράφημα G_f, και αφού $v' \in A^*$ θα υπάρχει μια διαδρομή s-v' στο G_f η προσάρτηση της e'' σε αυτή τη διαδρομή θα μας έδινε μια διαδρομή s-u' στο G_f, γεγονός που έρχεται σε αντίφαση με την υπόθεσή μας ότι $u' \in B^*$.

Έτσι όλες οι ακμές που εξέρχονται από το A^* είναι πλήρως κορεσμένες με ροή, ενώ όλες οι ακμές που εισέρχονται στο A^* μένουν τελείως αχρησιμοποίητες. Μπορούμε τώρα να χρησιμοποιήσουμε την (7.6) για να φτάσουμε στο επιθυμητό συμπέρασμα:

$$\begin{aligned} v(f) &= f^{out}(A^*) - f^{in}(A^*) \\ &= \sum_{e \text{ από το } A^*} f(e) - \sum_{e \text{ προς το } A^*} f(e) \\ &= \sum_{e \text{ από το } A^*} c_e - 0 \\ &= c(A^*, B^*). \quad \blacksquare \end{aligned}$$

Σημειώστε με ποιον τρόπο οι δύο τύποι υπολειπόμενων ακμών — οι ευθύδρομες ακμές και οι ανάδρομες ακμές — είναι κρίσιμες για την ανάλυση των δύο όρων στην παράσταση (7.6).

Με δεδομένο ότι ο αλγόριθμος Ford-Fulkerson τερματίζει όταν δεν υπάρχει καμία διαδρομή s-t στο υπολειπόμενο γράφημα, η εξίσωση (7.6) υποδηλώνει άμεσα ότι είναι βέλτιστος.

(7.10) *Η ροή $\bar{f}$ που επιστρέφεται από τον αλγόριθμο Ford-Fulkerson είναι μια μέγιστη ροή.*

Παρατηρούμε επίσης ότι ο αλγόριθμός μας μπορεί εύκολα να επεκταθεί έτσι ώστε να υπολογίζει μια ελάχιστη αποκοπή s-t (A^*, B^*), με τον ακόλουθο τρόπο.

(7.11) *Με δεδομένη μια ροή f με μέγιστη τιμή, μπορούμε να υπολογίσουμε μια αποκοπή s-t με ελάχιστη χωρητικότητα σε χρόνο $O(m)$.*

Απόδειξη. Ακολουθούμε απλώς τη δομή της απόδειξης της πρότασης (7.9). Κατασκευάζουμε το υπολειπόμενο γράφημα G_f και εκτελούμε αναζήτηση πρώτα κατά πλάτος ή πρώτα κατά βάθος για να προσδιορίσουμε το σύνολο A^* με όλους τους κόμβους τους οποίους μπορεί να φτάσει ο κόμβος s. Μετά ορίζουμε $B^* = V - A^*$ και επιστρέφουμε την αποκοπή (A^*, B^*). $\blacksquare$

Σημειώστε ότι μπορεί να υπάρχουν πολλές αποκοπές ελάχιστης χωρητικότητας σε ένα γράφημα G· η διαδικασία απόδειξης της (7.11) απλώς βρίσκει μια συγκεκριμένη από αυτές τις αποκοπές, ξεκινώντας από μια μέγιστη ροή $\bar{f}$.

Μέσα από την ανάλυση του αλγορίθμου έχουμε εντοπίσει, ως πρόσθετο πλεονέκτημα, το ακόλουθο εντυπωσιακό γεγονός.

(7.12) *Σε όλα τα δίκτυα ροής υπάρχει μια ροή f και μια αποκοπή (A, B) έτσι ώστε $v(f) = c(A, B)$.*

Η ουσία είναι ότι η f στην (7.12) πρέπει να είναι μια μέγιστη ροή s-t· επειδή αν υπήρχε μια ροή f' με μεγαλύτερη τιμή, η τιμή της f' θα υπερέβαινε τη χωρητικότητα της αποκοπής (A, B) και αυτό θα ερχόταν σε αντίφαση με την (7.8). Παρομοίως, αυτό δείχνει ότι η (A, B) στην (7.12) είναι μια *ελάχιστη αποκοπή* — καμία άλλη αποκοπή δεν μπορεί να έχει μικρότερη χωρητικότητα — επειδή αν υπήρχε μια αποκοπή (A', B') με μικρότερη χωρητικότητα αυτή θα ήταν μικρότερη από την τιμή της f, γεγονός που και πάλι έρχεται σε αντίφαση με την (7.8). Λόγω αυτών των συνεπειών η (7.12) συχνά αναφέρεται ως *Θεώρημα Μέγιστης Ροής και Ελάχιστης Αποκοπής* (Max-Flow Min-Cut Theorem), και διατυπώνεται με τον ακόλουθο τρόπο.

(7.13) *Σε όλα τα δίκτυα ροής η μέγιστη τιμή μιας ροής s-t είναι ίση με την ελάχιστη χωρητικότητα μιας αποκοπής s-t.*

Πρόσθετη ανάλυση: Ροές με ακέραιες τιμές

Μεταξύ των πολλών συμπερασμάτων που απορρέουν από την ανάλυση του αλγορίθμου Ford-Fulkerson, ένα ιδιαίτερα σημαντικό συμπέρασμα είναι το ακόλουθο. Με την (7.2) διατηρήσαμε συνεχώς μια ροή με ακέραιες τιμές, και με την (7.9) καταλήξαμε σε μια μέγιστη ροή. Έτσι έχουμε

(7.14) *Αν όλες οι χωρητικότητες στο δίκτυο ροής είναι ακέραιες, τότε υπάρχει μια μέγιστη ροή f για την οποία όλες οι τιμές ροής $f(e)$ είναι ακέραιες.*

Σημειώστε ότι η πρόταση (7.14) δεν ισχυρίζεται ότι *όλες* οι μέγιστες ροές έχουν ακέραιες τιμές, αλλά μόνο ότι *κάποια* μέγιστη ροή έχει αυτή την ιδιότητα. Περιέργως, αν και η (7.14) δεν κάνει καμία αναφορά στον αλγόριθμο Ford-Fulkerson, η αλγοριθμική μας προσέγγιση παρέχει μάλλον τον ευκολότερο τρόπο για την απόδειξή της.

Πραγματικοί αριθμοί ως χωρητικότητες; Τέλος, πριν προχωρήσουμε παραπέρα, μπορούμε να διερωτηθούμε πόσο κρίσιμη είναι η παραδοχή μας σχετικά με τις ακέραιες χωρητικότητες (αν παραβλέψουμε τις (7.4), (7.5), και (7.14) που προφανώς τη χρειάζονται). Καταρχήν παρατηρούμε ότι με το να επιτρέπουμε χωρητικότητες που είναι ρητοί αριθμοί το πρόβλημα δεν γίνεται πιο γενικό, επειδή μπορούμε να προσδιορίσουμε το ελάχιστο κοινό πολλαπλάσιο όλων των χωρητικοτήτων, να τις πολλαπλασιάσουμε όλες με αυτή την τιμή, και να πάρουμε έτσι ένα ισοδύναμο πρόβλημα με ακέραιες χωρητικότητες.

Τι γίνεται όμως αν έχουμε πραγματικούς αριθμούς ως χωρητικότητες; Σε ποιο σημείο της απόδειξης βασιστήκαμε στο ότι οι χωρητικότητες είναι ακέραιες; Στην πραγματικότητα βασιστήκαμε σε αυτό με πολύ κρίσιμο τρόπο: Χρησιμοποιήσαμε την (7.2) για να αποδείξουμε, στην (7.4), ότι η τιμή της ροής αυξάνεται τουλάχιστον κατά 1 σε κάθε βήμα. Με πραγματικούς αριθμούς ως χωρητικότητες θα πρέπει να μας απασχολεί το γεγονός ότι η τιμή της ροής μας συνεχώς αυξάνεται, αλλά με βήματα τα οποία γίνονται με αυθαίρετο τρόπο όλο και μικρότερα· έτσι δεν έχουμε καμία εγγύηση ότι ο αριθμός των επαναλήψεων στο βρόχο θα είναι πεπερασμένος. Και αυτό αποδεικνύεται εξαιρετικά μεγάλο πρόβλημα για τον ακόλουθο λόγο: *Αν γίνονται "παθολογικές" επιλογές για τη διαδρομή επαύξησης, ο αλγόριθμος Ford-Fulkerson με πραγματικές χωρητικότητες μπορεί να εκτελείται για πάντα.*

Παρόλα αυτά, μπορούμε και πάλι να αποδείξουμε ότι το Θεώρημα Μέγιστης Ροής και Ελάχιστης Αποκοπής (7.12) εξακολουθεί να ισχύει ακόμα και αν οι χωρητικότητες μπορεί να είναι πραγματικοί αριθμοί. Σημειώστε ότι στην (7.9) υποθέσαμε μόνο ότι η ροή f δεν έχει διαδρομή s-t στο υπολειπόμενο γράφημα G_f, για να συμπεράνουμε ότι υπάρχει μια αποκοπή s-t με ίση τιμή. Προφανώς, για οποιαδήποτε ροή f με μέγιστη τιμή το υπολειπόμενο γράφημα δεν έχει διαδρομή s-t· διαφορετικά θα υπήρχε τρόπος να αυξήσουμε την τιμή της ροής. Έτσι μπορούμε να αποδείξουμε την (7.12) στην περίπτωση των χωρητικοτήτων με πραγματικές τιμές, αποδεικνύοντας απλώς ότι για κάθε δίκτυο ροής υπάρχει μια μέγιστη ροή.

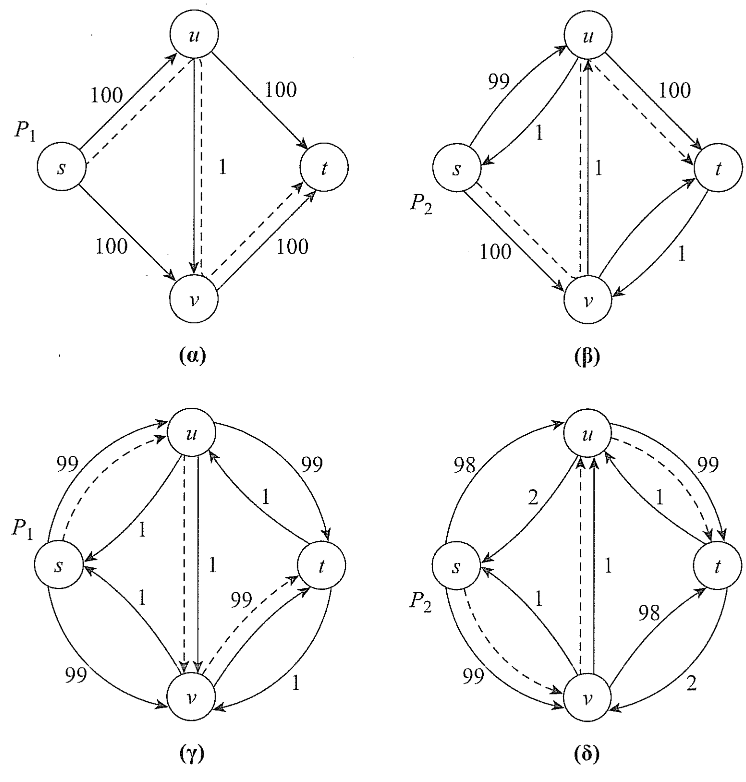
Φυσικά, οι χωρητικότητες στις πρακτικές εφαρμογές των δικτύων ροής θα είναι ακέραιοι ή ρητοί αριθμοί. Παρόλα αυτά, το πρόβλημα των "παθολογικών" επιλογών σε σχέση με τις διαδρομές επαύξησης μπορεί να παρουσιαστεί ακόμα και με ακέραιες χωρητικότητες: μπορεί να κάνει τον αλγόριθμο Ford-Fulkerson να απαιτεί τεράστιο αριθμό επαναλήψεων. Στην επόμενη ενότητα θα εξετάσουμε πώς να επιλέγουμε τις διαδρομές επαύξησης έτσι ώστε να αποφεύγεται η δυνητικά κακή συμπεριφορά του αλγορίθμου.

7.3 Επιλογή καλών διαδρομών επαύξησης

Στην προηγούμενη ενότητα είδαμε ότι ο οποιοσδήποτε τρόπος επιλογής μιας διαδρομής επαύξησης αυξάνει την τιμή της ροής, και αυτό οδηγεί σε ένα όριο C ως προς το πλήθος των επαυξήσεων, όπου $C = \sum_{e \text{ από το } s} c_e$. Όταν το C δεν είναι πολύ μεγάλο, αυτό είναι ένα εύλογο όριο· δυστυχώς, όμως, είναι πολύ ασθενές όταν το C είναι μεγάλο.

Για να πάρουμε μια αίσθηση για το πόσο κακό μπορεί να είναι αυτό το όριο, ας εξετάσουμε το παράδειγμα γραφήματος της Εικόνας 7.2· όμως αυτή τη φορά θα υποθέσουμε ότι οι χωρητικότητες είναι οι ακόλουθες: Οι ακμές (s, v), (s, u), (v, t), και (u, t) έχουν χωρητικότητα 100 και η ακμή (u, v) έχει χωρητικότητα 1, όπως φαίνεται στην Εικόνα 7.6. Είναι εύκολο να δούμε ότι η μέγιστη ροή έχει τιμή 200, και έχει $f(e) = 100$ για τις ακμές (s, v), (s, u), (v, t) και (u, t) ενώ έχει τιμή 0 στην ακμή (u, v). Αυτή η ροή μπορεί να επιτευχθεί με μια ακολουθία δύο επαυξήσεων, με χρήση των διαδρομών των κόμβων s, u, t και της διαδρομής s, v, t. Ας δούμε όμως πόσο άσχημα μπορεί να λειτουργήσει ο αλγόριθμος Ford-Fulkerson όταν γίνονται "παθολογικές" επιλογές για τις διαδρομές επαύξησης. Υποθέστε ότι ξεκινάμε με τη διαδρομή επαύξησης P_1 από τους κόμβους s, u, v, t με αυτή τη σειρά (όπως φαίνεται στην Εικόνα 7.6). Για αυτή τη διαδρομή έχουμε `bottleneck`$(P_1, f) = 1$. Μετά από αυτή την επαύξηση έχουμε $f(e) = 1$ στην ακμή $e = (u, v)$, έτσι η αντίστροφη ακμή υπάρχει στο υπολειπόμενο γράφημα. Για την επόμενη διαδρομή επαύξησης επιλέγουμε τη διαδρομή P_2 με τους κόμβους s, v, u, t

με αυτή τη σειρά. Σε αυτή τη δεύτερη επαύξηση έχουμε επίσης bottleneck $(P_2, f) = 1$. Μετά από αυτή τη δεύτερη επαύξηση έχουμε $f(e) = 0$ για την ακμή $e = (u, v)$, έτσι η ακμή υπάρχει και πάλι στο υπολειπόμενο γράφημα. Υποθέστε ότι εναλλασσόμαστε μεταξύ των επιλογών P_1 και P_2 για την επαύξηση. Σε αυτή την περίπτωση η κάθε επαύξηση θα έχει ελάχιστη υπολειπόμενη χωρητικότητα ίση με 1, και θα χρειαστούν 200 επαυξήσεις για να πάρουμε την επιθυμητή ροή με τιμή 200. Αυτό είναι ακριβώς το όριο που αποδείξαμε στην (7.4), αφού σε αυτό το παράδειγμα έχουμε $C = 200$.



Εικόνα 7.6 Οι εικόνες (α) έως (δ) δείχνουν τέσσερις επαναλήψεις του αλγορίθμου Ford-Fulkerson όταν χρησιμοποιείται κακή επιλογή ως προς τη διαδρομή επαύξησης: Οι επαυξήσεις εναλλάσσονται μεταξύ της διαδρομής P_1 με τους κόμβους s, u, v, t με αυτή τη σειρά, και της διαδρομής P_2 με τους κόμβους s, v, u, t με αυτή τη σειρά.

Σχεδιασμός ενός ταχύτερου αλγορίθμου ροής

Ο στόχος αυτής της ενότητας είναι να δείξουμε ότι μπορούμε να βελτιώσουμε σημαντικά αυτό το όριο με καλύτερη επιλογή των διαδρομών. Έχει γίνει πάρα πολλή δουλειά πάνω στο θέμα της εύρεσης καλών τρόπων επιλογής διαδρομών επαύξησης στο πρόβλημα της Μέγιστης Ροής, έτσι ώστε να ελαχιστοποιείται ο αριθμός των επαναλήψεων. Εδώ θα εστιάσουμε σε μια από τις πιο φυσικές προσεγγίσεις, και θα αναφέρουμε τις άλλες προσεγγίσεις στο τέλος της ενότητας. Θυμηθείτε ότι η επαύξηση αυξάνει την τιμή της μέγιστης ροής κατά την (ελάχιστη) υπολειπόμενη χωρητικότητα της επιλεγμένης διαδρομής· έτσι, αν επιλέγουμε διαδρομές με μεγάλη υπολειπόμενη χωρητικότητα θα κάνουμε μεγάλη πρόοδο. Μια φυσική ιδέα είναι να επιλέγουμε τη διαδρομή που έχει τη μεγαλύτερη υπολειπόμενη χωρητικότητα. Όμως η ανάγκη για εύρεση τέτοιων διαδρομών μπορεί να επιβραδύνει σημαντικά την κάθε επανάληψη. Για να αποφύγουμε αυτή την επιβράδυνση δεν θα προσπαθήσουμε να επιλέξουμε τη διαδρομή που έχει την *απολύτως* μεγαλύτερη υπολειπόμενη χωρητικότητα. Αντί γι' αυτό, θα διατηρούμε μια *παράμετρο κλιμάκωσης* (scaling parameter) Δ και θα αναζητούμε για διαδρομές που έχουν υπολειπόμενη χωρητικότητα τουλάχιστον ίση με Δ.

Έστω ότι το $G_f(\Delta)$ είναι το υποσύνολο του υπολειπόμενου γραφήματος που αποτελείται μόνο από ακμές με υπολειπόμενη χωρητικότητα τουλάχιστον ίση με Δ. Θα δουλέψουμε με τιμές του Δ που είναι δυνάμεις του 2. Ο αλγόριθμος είναι ο ακόλουθος.

```
Scaling Max-Flow
  Αρχικά f(e) = 0 για όλα τα e του G
  Αρχικά θέσε το Δ ίσο με τη μεγαλύτερη δύναμη του 2 που δεν είναι
        μεγαλύτερη από τη μέγιστη χωρητικότητα των ακμών που εξέρχονται
        από το s: Δ ≤ max_{e από το s} c_e
    While Δ ≥ 1
       While υπάρχει διαδρομή s-t στο γράφημα G_f(Δ)
          Έστω ότι η P είναι μια απλή διαδρομή s-t στο G_f(Δ)
          f'=augment(f, P)
          Ενημέρωσε την f σε f' και ενημέρωσε το G_f(Δ)
       Endwhile
       Δ = Δ/2
    Endwhile
Return f
```

Ανάλυση του αλγορίθμου

Καταρχήν παρατηρήστε ότι ο νέος αλγόριθμος Μέγιστης Ροής με Κλιμάκωση (Scaling Max-Flow) είναι πραγματικά απλώς μια υλοποίηση του αρχικού αλγορίθμου Ford-Fulkerson. Οι νέοι βρόχοι, η τιμή Δ, και το περιορισμένο υπολειπόμενο γράφημα G_f (Δ) χρησιμοποιούνται μόνο για να καθοδηγήσουν την επιλογή της υπολειπόμενης διαδρομής — με στόχο να χρησιμοποιούνται για όσο το δυνατόν περισσότερο ακμές με μεγάλη υπολειπόμενη χωρητικότητα. Κατά συνέπεια όλες οι ιδιότητες που αποδείξαμε σχετικά με τον πρωτότυπο αλγόριθμο Μέγιστης Ροής ισχύουν επίσης και για αυτή τη νέα εκδοχή: η ροή εξακολουθεί να έχει ακέραια τιμή κατά την πορεία του αλγορίθμου, και έτσι όλες οι υπολειπόμενες χωρητικότητες έχουν ακέραιες τιμές.

(7.15) *Αν οι χωρητικότητες έχουν ακέραιες τιμές, τότε καθόλη την εκτέλεση του αλγορίθμου Μέγιστης Ροής με Κλιμάκωση η ροή και οι υπολειπόμενες χωρητικότητες θα παραμένουν ακέραιες. Αυτό σημαίνει ότι, όταν $\Delta = 1$, το $G_f(\Delta)$ θα είναι ίδιο με το G_f και έτσι όταν ο αλγόριθμος τερματίζει η ροή f θα έχει μέγιστη τιμή.*

Θα εξετάσουμε τώρα το χρόνο εκτέλεσης. Θα ονομάσουμε την επανάληψη του εξωτερικού βρόχου `While` — με σταθερή τιμή του Δ — ως *φάση κλιμάκωσης του* Δ (Δ scaling phase). Μπορούμε εύκολα να βρούμε ένα άνω όριο ως προς το πλήθος των διαφορετικών φάσεων κλιμάκωσης του Δ, σε σχέση με την τιμή $C = \sum_{e \text{ από το } s} c_e$ που χρησιμοποιήσαμε επίσης και στην προηγούμενη ενότητα. Η αρχική τιμή του Δ είναι το πολύ ίση με C, υποδιπλασιάζεται συνεχώς, και ποτέ δεν γίνεται μικρότερη από 1. Έτσι

(7.16) *Το πλήθος των επαναλήψεων στον εξωτερικό βρόχο* `While` *είναι το πολύ* $1 + \lceil \log_2 C \rceil$.

Το δυσκολότερο κομμάτι είναι να βρούμε ένα όριο ως προς το πλήθος των επαυξήσεων που πραγματοποιούνται σε κάθε φάση κλιμάκωσης. Η ιδέα εδώ είναι ότι χρησιμοποιούμε διαδρομές που επαυξάνουν κατά πολύ τη ροή, και έτσι θα πρέπει να υπάρχουν σχετικά λίγες επαυξήσεις. Κατά τη φάση κλιμάκωσης Δ χρησιμοποιούμε μόνο ακμές με υπολειπόμενη χωρητικότητα τουλάχιστον ίση με Δ. Από την (7.3) έχουμε

(7.17) *Κατά τη φάση κλιμάκωσης Δ, η κάθε επαύξηση αυξάνει την τιμή της ροής κατά τουλάχιστον Δ.*

Η ιδέα-κλειδί είναι ότι στο τέλος της φάσης κλιμάκωσης Δ η ροή f δεν μπορεί να είναι πολύ μακριά από τη μέγιστη δυνατή τιμή.

(7.18) *Έστω ότι f είναι η ροή στο τέλος της φάσης κλιμάκωσης Δ. Υπάρχει στο G μια αποκοπή s-t (A, B) για την οποία $c(A, B) \le \nu(f) + m\Delta$, όπου το m είναι το πλήθος των ακμών στο γράφημα G. Κατά συνέπεια, η μέγιστη ροή του δικτύου έχει τιμή το πολύ ίση με $\nu(f) + m\Delta$.*

Απόδειξη. Αυτή η απόδειξη είναι ανάλογη με την απόδειξή μας για την πρόταση (7.9), η οποία έδειξε ότι η ροή που επιστρέφεται από τον αρχικό αλγόριθμο Μέγιστης Ροής έχει μέγιστη τιμή.

Όπως και σε εκείνη την απόδειξη, πρέπει να προσδιορίσουμε μια αποκοπή (A, B) που να έχει την επιθυμητή ιδιότητα. Έστω ότι το A συμβολίζει το σύνολο όλων των κόμβων v του G για τους οποίους υπάρχει μια διαδρομή s-v στο $G_f(\Delta)$. Έστω ότι το B συμβολίζει το σύνολο όλων των υπόλοιπων κόμβων: $B = V - A$. Μπορούμε να δούμε ότι το (A, B) είναι πράγματι μια αποκοπή s-t, αφού διαφορετικά η φάση δεν θα είχε τελειώσει.

Θεωρήστε τώρα μια ακμή $e = (u, v)$ του G για την οποία $u \in A$ και $v \in B$. Ισχυριζόμαστε ότι $c_e < f(e) + \Delta$. Αν δεν συνέβαινε αυτό, τότε το e θα ήταν μια ευθύδρομη ακμή στο γράφημα $G_f(\Delta)$ και, αφού $u \in A$, θα υπάρχει μια διαδρομή s-u στο $G_f(\Delta)$· προσαρτώντας την e σε αυτή τη διαδρομή θα παίρναμε μια διαδρομή s-v στο $G_f(\Delta)$, κάτι που έρχεται σε αντίφαση με την παραδοχή μας ότι $v \in B$. Παρομοίως, ισχυριζόμαστε ότι για οποιαδήποτε ακμή $e' = (u', v')$ του G για την οποία $u' \in B$ και $v' \in A$ έχουμε $f(e') < \Delta$. Πράγματι, αν $f(e') \ge \Delta$ τότε το e' θα δημιουργούσε μια ανάδρομη ακμή $e'' = (v', u')$ στο γράφημα $G_f(\Delta)$, και αφού $v' \in A$ θα υπάρχει μια διαδρομή s-v' στο $G_f(\Delta)$· προσαρτώντας την e'' σε αυτή τη διαδρομή θα παίρναμε μια διαδρομή s-u' στο $G_f(\Delta)$, κάτι που έρχεται σε αντίφαση με την παραδοχή μας ότι $u' \in B$.

Έτσι όλες οι ακμές e που εξέρχονται από το A είναι σχεδόν κορεσμένες — ικανοποιούν τη σχέση $c_e < f(e) + \Delta$ — και όλες οι ακμές που εισέρχονται στο A είναι σχεδόν κενές — ικανοποιούν τη σχέση $f(e) < \Delta$. Μπορούμε τώρα να χρησιμοποιήσουμε την (7.6) για να φτάσουμε στο επιθυμητό συμπέρασμα:

$$
\begin{aligned}
\nu(f) &= \sum_{e \text{ από το } A^*} f(e) - \sum_{e \text{ προς το } A^*} f(e) \\
&\geq \sum_{e \text{ από το } A^*} (c_e - \Delta) - \sum_{e \text{ προς το } A^*} \Delta \\
&= \sum_{e \text{ από το } A^*} c_e - \sum_{e \text{ από το } A^*} \Delta - \sum_{e \text{ προς το } A^*} \Delta \\
&\geq \mathrm{c(A,B)} - \mathrm{m}\Delta
\end{aligned}
$$

Εδώ η πρώτη ανισότητα προκύπτει από τα όριά μας ως προς τις τιμές ροής για ακμές κατά μήκος της αποκοπής, και η δεύτερη ανισότητα προκύπτει από το απλό γεγονός ότι το γράφημα περιέχει μόνο m ακμές συνολικά.

Με βάση την (7.8), η τιμή της μέγιστης ροής έχει ως όριο τη χωρητικότητα οποιασδήποτε αποκοπής. Χρησιμοποιώντας την αποκοπή (A, B) μπορούμε να βρούμε το όριο το οποίο αναφέρεται στη δεύτερη πρόταση της (7.18). ■

(7.19) *Ο αριθμός των επαυξήσεων σε μια φάση κλιμάκωσης είναι το πολύ ίσος με 2m.*

Απόδειξη. Η πρόταση αυτή είναι προφανώς αληθής για την πρώτη φάση κλιμάκωσης: μπορούμε να χρησιμοποιήσουμε την καθεμία από τις ακμές που εξέρχονται από τον κόμβο s μόνο για το πολύ μία επαύξηση σε αυτή τη φάση. Θεωρήστε τώρα μια μεταγενέστερη φάση κλιμάκωσης Δ, και έστω ότι f_p είναι η ροή στο τέλος της *προηγούμενης* φάσης κλιμάκωσης. Σε εκείνη τη φάση χρησιμοποιήσαμε ως παράμετρό μας το $\Delta' = 2\Delta$. Με βάση την (7.18), η μέγιστη ροή f^* έχει τιμή το πολύ ίση με $\nu(f^*) \leq \nu(f_p) + m\Delta' = \nu(f_p) + 2m\Delta$. Στη φάση κλιμάκωσης Δ η κάθε επαύξηση αυξάνει την τιμή της ροής κατά τουλάχιστον Δ, και έτσι μπορούν να γίνουν το πολύ $2m$ επαυξήσεις. ■

Η κάθε επαύξηση απαιτεί $O(m)$ χρόνο, συμπεριλαμβανομένου του χρόνου που απαιτείται για τη διαμόρφωση του γραφήματος και την εύρεση της κατάλληλης διαδρομής. Έχουμε το πολύ $1 + \lceil \log_2 C \rceil$ φάσεις κλιμάκωσης και το πολύ $2m$ επαυξήσεις σε κάθε φάση κλιμάκωσης. Έτσι έχουμε το ακόλουθο αποτέλεσμα.

(7.20) *Ο αλγόριθμος Μέγιστης Ροής με Κλιμάκωση (Scaling Max-Flow) σε ένα γράφημα με m ακμές και ακέραιες χωρητικότητες βρίσκει μια μέγιστη ροή το πολύ σε* $2m(1 + \lceil \log_2 C \rceil)$ *επαυξήσεις. Ο αλγόριθμος αυτός μπορεί να υλοποιηθεί έτσι ώστε να εκτελείται το πολύ σε* $O(m^2 \log_2 C)$ *χρόνο.*

Όταν το C είναι μεγάλο, αυτό το χρονικό όριο είναι πολύ καλύτερο από το όριο $O(mC)$ που ισχύει για μια τυχαία υλοποίηση του αλγορίθμου Ford-Fulkerson. Στο παράδειγμα που είδαμε στην αρχή της ενότητας είχαμε χωρητικότητες με μέγεθος 100, όμως θα μπορούσαμε εξίσου καλά να έχουμε χωρητικότητες με μέγεθος 2^{100}· στην περίπτωση αυτή ο γενικός αλγόριθμος Ford-Fulkerson θα χρειαζόταν χρόνο ανάλογο του 2^{100}, ενώ ο αλγόριθμος κλιμάκωσης θα χρειαζόταν χρόνο ανάλογο του $\log_2(2^{100}) = 100$. Μπορούμε να δούμε τη διαφορά με τον ακόλουθο τρόπο: Ο γενικός αλγόριθμος Ford-Fulkerson απαιτεί χρόνο ανάλογο προς το *μέγεθος* των χωρητικοτήτων, ενώ ο αλγόριθμος κλιμάκωσης απαιτεί χρόνο ανάλογο προς το πλήθος των *δυαδικών ψηφίων* που απαιτούνται για τον προσδιορισμό των χωρητικοτήτων στην είσοδο του προβλήματος. Κατά συνέπεια ο αλγόριθμος κλιμάκωσης εκτελείται σε χρόνο πολυωνυμικό ως προς το μέγεθος της εισόδου (δηλαδή, το πλήθος των ακμών και η αριθμητική αναπαράσταση των χωρητικοτήτων), και έτσι ικανοποιεί τον παραδοσιακό στόχο μας να επιτύχουμε έναν αλγόριθμο πολυωνυμικού χρόνου. Οι κακές υλοποιήσεις του αλγορίθμου Ford-Fulkerson, οι οποίες μπορεί να απαιτούν σχεδόν C επαναλήψεις, δεν ικανοποιούν αυτό το κριτήριο πολυωνυμικότητας. (Θυμηθείτε ότι στην Ενότητα 6.4 χρησιμοποιήσαμε τον όρο *ψευδοπολυωνυμικός* για να περιγράψουμε τέτοιους αλγορίθμους, οι οποίοι είναι πολυωνυμικοί ως προς το μέγεθος των αριθμών εισόδου αλλά όχι ως προς το πλήθος των ψηφίων που απαιτούνται για την αναπαράστασή τους.)

Επεκτάσεις: Ισχυρά πολυωνυμικοί αλγόριθμοι

Θα μπορούσαμε άραγε να ζητήσουμε κάτι καλύτερο από αυτό που εγγυάται ο αλγόριθμος κλιμάκωσης; Ας δούμε ένα πράγμα το οποίο θα μπορούσαμε να θέλουμε: Το παράδειγμα γραφήματός μας (Εικόνα 7.6) είχε τέσσερις κόμβους και πέντε ακμές, έτσι θα ήταν καλό να μπορούμε να χρησιμοποιήσουμε έναν αριθμό επαναλήψεων που είναι πολυωνυμικός ως προς τους αριθμούς 4 και 5, και είναι τελείως ανεξάρτητος από τις τιμές των χωρητικοτήτων. Ένας τέτοιος αλγόριθμος, ο οποίος είναι πολυωνυμικός μόνο ως προς τα $|V|$ και $|E|$ και δουλεύει με αριθμούς που έχουν πολυωνυμικό πλήθος ψηφίων, ονομάζεται *ισχυρά πολυωνυμικός αλγόριθμος* (strongly polynomial algorithm). Στην πραγματικότητα υπάρχει μια απλή και φυσική υλοποίηση του αλγορίθμου Ford-Fulkerson που οδηγεί σε ένα τέτοιο ισχυρά πολυωνυμικό όριο: η κάθε επανάληψη χρησιμοποιεί τη διαδρομή επαύξησης με το μικρότερο αριθμό ακμών. Ο Dinitz, και ανεξάρτητα οι Edmonds και Karp, απέδειξαν ότι με αυτή την επιλογή ο αλγόριθμος τερματίζεται σε το πολύ $O(mn)$ επαναλήψεις. Στην πραγματικότητα αυτοί ήταν οι πρώτοι πολυωνυμικού αλγόριθμοι για το πρόβλημα της Μέγιστης Ροής. Από τότε έχει γίνει πολύ μεγάλη έρευνα πάνω στο θέμα της βελτίωσης του χρόνου εκτέλεσης για τους αλγορίθμους μέγιστης ροής. Σήμερα υπάρχουν αλγόριθμοι που επιτυγχάνουν χρόνους εκτέλεσης $O(mn \log n)$, $O(n^3)$, και $O(\min(n^{2/3}, m^{1/2})m \log n \log U)$, όπου το τελευταίο όριο υποθέτει ότι όλες οι χωρητικότητες είναι ακέραιες και το πολύ ίσες με U. Στην επόμενη ενότητα θα εξετάσουμε έναν ισχυρά πολυωνυμικό αλγόριθμο μέγιστης ροής που βασίζεται σε μια διαφορετική αρχή.

*7.4 Ο αλγόριθμος Μέγιστης Ροής με Προροή-Προώθηση

Από το ξεκίνημα η ανάλυση που κάναμε στο πρόβλημα της Μέγιστης Ροής εστιάστηκε γύρω από την ιδέα των διαδρομών επαύξησης στο υπολειπόμενο γράφημα. Υπάρχουν, όμως, και κάποιες πολύ ισχυρές τεχνικές για το πρόβλημα της μέγιστης ροής, οι οποίες δεν βασίζονται ρητά στις διαδρομές επαύξησης. Σε αυτή την ενότητα θα μελετήσουμε μια τέτοια τεχνική, τον αλγόριθμο Προροής-Προώθησης (Preflow-Push Algorithm).

Σχεδιασμός του αλγορίθμου

Οι αλγόριθμοι που βασίζονται σε διαδρομές επαύξησης διατηρούν μια ροή f και χρησιμοποιούν τη διαδικασία augment για να αυξήσουν την τιμή της ροής. Σε αντιδιαστολή, ο αλγόριθμος Προροής-Προώθησης αυξάνει ουσιαστικά τη ροή σε βάση ακμής προς ακμή. Η τροποποίηση της ροής σε μια ακμή θα παραβιάζει τυπικά τη συνθήκη διατήρησης, και έτσι ο αλγόριθμος θα πρέπει κατά τη λειτουργία του να διατηρεί κάτι που δεν συμπεριφέρεται εξίσου καλά με μια ροή — κάτι που δεν υπακούει στη συνθήκη διατήρησης.

Προροές Λέμε ότι η *προροή s-t* (*s-t* preflow, και για συντομία απλώς *προροή*) είναι μια συνάρτηση f που αντιστοιχίζει την κάθε ακμή e σε ένα μη αρνητικό πραγματικό αριθμό $f: E \to \mathbf{R}^+$. Η προροή f θα πρέπει να ικανοποιεί τις συνθήκες χωρητικότητας:

(i) Για κάθε $e \in E$, έχουμε $0 \le f(e) \le c_e$.

Στη θέση των συνθηκών διατήρησης απαιτούμε μόνο ανισότητες. Κάθε κόμβος εκτός του s πρέπει να έχει εισερχόμενη ροή που είναι τουλάχιστον ίση με την εξερχόμενη.

(ii) Για κάθε κόμβο v εκτός της προέλευσης s, έχουμε

$$\sum_{e \text{ προς το } v} f(e) \ge \sum_{e \text{ από το } v} f(e).$$

Θα ονομάζουμε τη διαφορά

$$e_f(v) = \sum_{e \text{ προς το } v} f(e) - \sum_{e \text{ από το } v} f(e).$$

πλεόνασμα (excess) της προροής στον κόμβο v. Παρατηρήστε ότι η προροή όπου όλοι οι κόμβοι εκτός του s και του t έχουν μηδενικό πλεόνασμα αποτελεί ροή, και η τιμή της ροής είναι ακριβώς $e_f(t) = -e_f(s)$. Μπορούμε και πάλι να ορίσουμε την έννοια του υπολειπόμενο γραφήματος G_f για μια προροή f, ακριβώς όπως κάναμε και για μια ροή. Ο αλγόριθμος θα "προωθεί" ροή στις ακμές του υπολειπόμενου γραφήματος (χρησιμοποιώντας και ευθύδρομες και ανάδρομες ακμές).

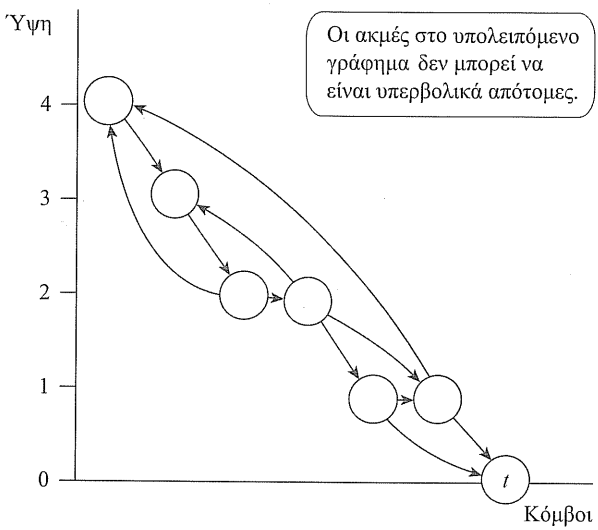
Προροές και Ετικέτες Ο αλγόριθμος Προροής-Προώθησης θα διατηρεί μια προροή και θα δουλεύει για να τη μετατρέψει σε ροή. Ο αλγόριθμος βασίζεται στη φυσική διαισθητική ιδέα ότι η ροή βρίσκει με φυσικό τρόπο το δρόμο της " κατηφορικά". Τα "υψώματα" για αυτή τη διαισθητική ιδέα θα είναι ετικέτες $h(v)$ τις οποίες ο αλγόριθμος θα ορίζει και θα διατηρεί για κάθε κόμβο, όπως φαίνεται στην Εικόνα 7.7. Θα προωθούμε ροή από κόμβους με υψηλότερες ετικέτες σε κόμβους με χαμηλότερες ετικέτες, ακολουθώντας τη διαισθητική ιδέα ότι η ροή προχωρά "κατηφορικά". Για να το κάνουμε αυτό πιο ακριβές, η *απόδοση ετικετών* (labeling) είναι μια συνάρτηση $h : V \to \mathbf{Z}_{\ge 0}$ από τους κόμβους στους μη αρνητικούς ακέραιους. Θα αναφερόμαστε επίσης στις ετικέτες με τον όρο *ύψος* των κόμβων. Θα λέμε ότι μια απόδοση ετικετών h και μια προροή *s-t* είναι *συμβατές* αν

(i) (*Συνθήκες προέλευσης και απόληξης*) $h(t) = 0$ και $h(s) = n$,

(ii) (*Συνθήκες ρύθμισης κλίσης*) Για όλες τις ακμές $(v, w) \in E_f$ στο υπολοιπόμενο γράφημα, έχουμε $h(v) \le h(w) + 1$.

Διαισθητικά, η διαφορά ύψους n μεταξύ της προέλευσης και της απόληξης έχει στόχο να εξασφαλίζει ότι η ροή ξεκινά από αρκετό ύψος ώστε να ρέει από το s προς την απόληξη t, ενώ η συνθήκη ρύθμισης κλίσης θα βοηθά να γίνεται αρκετά σταδιακά η κάθοδος έτσι ώστε να φτάσει στην απόληξη.

Η βασική ιδιότητα μιας συμβατής προροής και απόδοσης ετικετών είναι ότι δεν μπορεί να υπάρχει καμία διαδρομή s-t στο υπολοιπόμενο γράφημα.



Εικόνα 7.7 Ένα υπολοιπόμενο γράφημα και μια συμβατή απόδοση ετικετών. Καμία ακμή στο υπολοιπόμενο γράφημα δεν μπορεί να είναι υπερβολικά "απότομη" — η αρχή της μπορεί να απέχει από το τέλος της το πολύ μία μονάδα ύψους. Ο κόμβος προέλευσης s θα πρέπει να έχει $h(s) = n$, και δεν είναι σχεδιασμένος στην εικόνα.

(7.21) *Αν η προροή s-t f είναι συμβατή με μια απόδοση ετικετών h, τότε δεν υπάρχει καμία διαδρομή s-t στο υπολοιπόμενο γράφημα G_f.*

Απόδειξη. Θα αποδείξουμε αυτή την πρόταση με εις άτοπον απαγωγή. Έστω ότι P είναι μια απλή διαδρομή s-t στο υπολοιπόμενο γράφημα G. Υποθέστε ότι οι κόμβοι κατά μήκος της διαδρομής P είναι οι $s, v_1, ..., v_k = t$. Από τον ορισμό της απόδοσης ετικετών που είναι συμβατή με μια προροή f έχουμε ότι $h(s) = n$. Η ακμή (s, v_1) υπάρχει στο υπολοιπόμενο γράφημα, και έτσι $h(v_1) \ge h(s) - 1 = n - 1$. Χρησιμοποιώντας επαγωγή στο i, και με βάση τη συνθήκη ρύθμισης κλίσης για την ακμή (v_{i-1}, v_i), βρίσκουμε ότι για όλες τις ακμές v_i στη διαδρομή P το ύψος είναι τουλάχιστον $h(v_i) \ge n - i$. Σημειώστε ότι ο τελευταίος κόμβος αυτής της διαδρομής είναι ο $v_k = t$· έτσι βρίσκουμε ότι $h(t) \ge n - k$. Όμως εξ ορισμού $h(t) = 0$, και $k < n$ αφού η διαδρομή P είναι απλή. Αυτή η αντίφαση αποδεικνύει τον ισχυρισμό μας. ■

Θυμηθείτε από την (7.9) ότι αν δεν υπάρχει καμία διαδρομή s-t στο υπολοιπόμενο γράφημα G_f για μια *ροή* f, τότε η ροή έχει μέγιστη τιμή. Αυτό μας οδηγεί στο ακόλουθο συμπέρασμα.

(7.22) *Αν η ροή s-t f είναι συμβατή με μια απόδοση ετικετών h, τότε η f είναι ροή με μέγιστη τιμή.*

Σημειώστε ότι η πρόταση (7.21) εφαρμόζεται σε προροές, ενώ η πρόταση (7.22) είναι πιο περιοριστική αφού εφαρμόζεται μόνο σε ροές. Έτσι ο αλγόριθμος Προροής-Προώθησης θα διατηρεί μια προροή f και μια απόδοση ετικετών h που είναι συμβατή με την f, και θα τροποποιεί τα f και h έτσι ώστε να κάνει την f ροή. Όταν η f γίνει πραγματικά ροή, μπορούμε να χρησιμοποιήσουμε την πρόταση (7.22) για να συμπεράνουμε ότι είναι μια μέγιστη ροή. Με βάση αυτά, μπορούμε να θεωρήσουμε ότι ο αλγόριθμος Προροής-Προώθησης είναι κατά κάποιον τρόπο ορθογώνιος με τον αλγόριθμο Ford-Fulkerson. Ο αλγόριθμος Ford-Fulkerson διατηρεί μια εφικτή ροή και την τροποποιεί σταδιακά έτσι ώστε να την κάνει βέλτιστη. Από την άλλη πλευρά, ο αλγόριθμος Προροής-Προώθησης διατηρεί μια συνθήκη που υπονοεί ότι θα ήταν βέλτιστη μια προροή f *αν ήταν εφικτή ροή*, και ο αλγόριθμος μετατρέπει σταδιακά την προροή f σε ροή.

Για να ξεκινήσουμε τον αλγόριθμο, θα χρειαστεί να ορίσουμε μια αρχική προροή f και μια απόδοση ετικετών h που να είναι συμβατή. Ως αρχική απόδοση ετικετών θα χρησιμοποιήσουμε $h(v) = 0$ για όλα τα $v \neq s$ και $h(s) = n$. Για να δημιουργήσουμε μια προροή f που να είναι συμβατή με αυτή την απόδοση ετικετών, πρέπει να εξασφαλίσουμε ότι στο υπολοιπόμενο γράφημα δεν υπάρχουν ακμές που να φεύγουν από το s (επειδή αυτές οι ακμές δεν ικανοποιούν τη συνθήκη ρύθμισης κλίσης). Έτσι ορίζουμε την αρχική προροή ως $f(e) = c_e$ για όλες τις ακμές $e = (s, v)$ που εξέρχονται από την προέλευση, και $f(e) = 0$ για όλες τις άλλες ακμές.

(7.23) *Η αρχική προροή f και απόδοση ετικετών h είναι συμβατές.*

Προώθηση και ανανέωση ετικετών Θα εξετάσουμε τώρα τα βήματα που κάνει ο αλγόριθμος για να μετατρέψει την προροή f σε μια εφικτή ροή, διατηρώντας την παράλληλα συμβατή με κάποια απόδοση ετικετών h. Ας εξετάσουμε οποιονδήποτε κόμβο v που διαθέτει πλεόνασμα — με άλλα λόγια, $e_f(v) > 0$. Αν υπάρχει στο υπολοιπόμενο γράφημα G_f μια ακμή e που φεύγει από τον κόμβο v και πηγαίνει σε έναν κόμβο w με χαμηλότερο ύψος (σημειώστε ότι το $h(w)$ είναι το πολύ κατά 1 μικρότερο από το $h(v)$, εξαιτίας της συνθήκης ρύθμισης κλίσης), τότε μπορούμε να τροποποιήσουμε την f προωθώντας κάποιο από το πλεόνασμα ροής από το v προς το w. Θα την ονομάζουμε αυτή λειτουργία *προώθησης* (push).

```
push(f, h, v, w)
  Είναι εφαρμόσιμη αν e_f(v) > 0, h(w) < h(v), και (v, w) ∈ E_f
  If e = (v, w) είναι ευθύδρομη ακμή then
    θέσε δ = min(e_f(v), c_e-f(e)) και
    αύξησε το f(e) κατά δ
  If (v, w) είναι ανάδρομη ακμή then
```

```
    θέσε e = (w, v), δ = min(e_f(v), f(e)) και
    μείωσε το f(e) κατά δ
  Return(f, h)
```

Αν δεν μπορούμε να προωθήσουμε το πλεόνασμα του v σε καμία από τις ακμές που φεύγουν από το v, τότε θα πρέπει να αλλάξουμε το ύψος του v. Θα την ονομάσουμε αυτή λειτουργία *ανανέωσης ετικετών* (relabel).

```
relabel(f, h, v)
  Είναι εφαρμόσιμη αν e_f(v) > 0 και
     για όλες τις ακμές (v, w) ∈ E_f έχουμε h(w) ≥ h(v)
  Αύξησε το h(v) κατά 1
  Return(f, h)
```

Ο πλήρης αλγόριθμος Προροής-Προώθησης Έτσι, για να συνοψίσουμε, ο αλγόριθμος Προροής-Προώθησης είναι ο ακόλουθος.

```
Preflow-Push
  Αρχικά h(v) = 0 για όλα τα v ≠ s και h(s) = n, και
  f(e) = c_e για όλα τα e = (s, v) και f(e) = 0 για τις υπόλοιπες ακμές
  While υπάρχει κόμβος v ≠ t με πλεόνασμα e_f(v) > 0
    Έστω ότι v είναι ένας κόμβος με πλεόνασμα
    If υπάρχει w όπου μπορεί να εφαρμοστεί η push(f, h, v, w) then
       push(f, h, v, w)
    Else
       relabel(f, h, v)
  Endwhile
  Return(f)
```

Ανάλυση του αλγορίθμου

Όπως συνήθως, αυτός ο αλγόριθμος είναι κάπως "ελλιπώς προσδιορισμένος". Για μια υλοποίηση αυτού του αλγορίθμου θα πρέπει να προσδιορίσουμε ποιον από τους κόμβους με πλεόνασμα θα επιλέξουμε, και πώς θα επιλέγουμε με αποδοτικό τρόπο μια ακμή στην οποία θα κάνουμε προώθηση. Παρόλα αυτά, είναι ξεκάθαρο ότι η κάθε επανάληψη αυτού του αλγορίθμου μπορεί να υλοποιηθεί σε πολυωνυμικό χρόνο. (Θα μελετήσουμε στη συνέχεια με ποιον τρόπο μπορούμε να τον υλοποιήσουμε επαρκώς αποδοτικά.) Επιπρόσθετα, δεν είναι δύσκολο να δούμε ότι η προροή f και η απόδοση ετικετών h είναι συμβατές σε ολόκληρη την πορεία του αλγορίθμου. Αν ο αλγόριθμος τερματίσει — κάτι που δεν είναι καθόλου προφανές με βάση την περιγραφή του — τότε δεν θα υπάρχει κανένας άλλος κόμβος εκτός από τον t με θετικό πλεόνασμα, και έτσι η προροή f θα είναι στην πραγματικότητα ροή. Από την (7.22) γνωρίζουμε ότι στην περίπτωση αυτή κατά τον τερματισμό η f θα είναι μια μέγιστη ροή.

Ας συνοψίσουμε μερικές απλές παρατηρήσεις σχετικά με τον αλγόριθμο.

(7.24) *Κατά την εκτέλεση του αλγορίθμου Προροής-Προώθησης:*

(i) οι ετικέτες είναι μη αρνητικοί ακέραιοι·

(ii) η f είναι μια προροή, και αν οι χωρητικότητες είναι ακέραιες, τότε η προροή f είναι ακέραια· και

(iii) η προροή f και η απόδοση ετικετών h είναι συμβατές.

Αν ο αλγόριθμος επιστρέψει μια προροή f, τότε η f είναι μια ροή με μέγιστη τιμή.

Απόδειξη. Από την (7.23) έχουμε ότι η αρχική προροή f και απόδοση ετικετών h είναι συμβατές. Θα δείξουμε με επαγωγή ως προς το πλήθος των λειτουργιών `push` and `relabel` ότι τα f και h ικανοποιούν τις ιδιότητες της πρότασης αυτής. Η λειτουργία `push` τροποποιεί την προροή f, αλλά τα όρια ως προς το δ εξασφαλίζουν ότι η επιστρεφόμενη f ικανοποιεί τους περιορισμούς ως προς τη χωρητικότητα, και ότι τα πλεονάσματα παραμένουν όλα μη αρνητικά, άρα η f είναι προροή. Για να δείξουμε ότι η προροή f και η απόδοση ετικετών h είναι συμβατές, παρατηρήστε ότι η `push`(f, h, v, w) μπορεί να προσθέσει στο υπολοιπόμενο γράφημα μία ακμή, την αντίστροφη ακμή (v, w), και αυτή η ακμή ικανοποιεί τη συνθήκη ρύθμισης κλίσης. Η λειτουργία `relabel` αυξάνει την ετικέτα του v, και έτσι αυξάνει την "κλίση" όλων των ακμών που φεύγουν από το v. Παρόλα αυτά, εφαρμόζεται μόνο όταν καμία από τις ακμές που φεύγουν από το v στο υπολοιπόμενο γράφημα δεν πηγαίνει "κατηφορικά", και έτσι η προροή f και η απόδοση ετικετών h είναι συμβατές μετά την ανανέωση ετικετών.

Ο αλγόριθμος τερματίζει αν κανένας άλλος κόμβος εκτός από τον s ή τον t δεν έχει πλεόνασμα. Στην περίπτωση αυτή η f είναι εξ ορισμού ροή· και αφού η προροή f και η απόδοση ετικετών h παραμένουν συμβατές σε όλη την πορεία του αλγορίθμου, η (7.22) μας δείχνει ότι η f είναι μια ροή με μέγιστη τιμή. ■

Θα εξετάσουμε τώρα τον αριθμό των λειτουργιών `push` και `relabel`. Καταρχήν θα αποδείξουμε ένα όριο ως προς τις λειτουργίες `relabel`, και αυτό θα μας βοηθήσει να αποδείξουμε ένα όριο ως προς το μέγιστο δυνατό αριθμό λειτουργιών `push`. Ο αλγόριθμος δεν αλλάζει ποτέ την ετικέτα του s (αφού η προέλευση δεν έχει ποτέ θετικό πλεόνασμα). Όλοι οι άλλοι κόμβοι v ξεκινούν με $h(v) = 0$, και οι ετικέτες τους αλλάζουν κατά 1 σε κάθε μεταβολή. Έτσι χρειάζεται απλώς να δώσουμε ένα όριο ως προς το πόσο μεγάλη μπορεί να είναι μια ετικέτα. Εξετάζουμε έναν κόμβο v για τη λειτουργία `relabel` μόνο όταν ο κόμβος v έχει πλεόνασμα. Η μόνη πηγή ροής στο δίκτυο είναι η προέλευση s· έτσι, διαισθητικά, το πλεόνασμα του v θα πρέπει να προέρχεται από τον κόμβο s. Η ακόλουθη συνέπεια αυτού του γεγονότος θα αποδειχθεί θεμελιώδης για τον καθορισμό ορίου ως προς τις ετικέτες.

(7.25) *Έστω ότι η f είναι μια προροή. Αν ο κόμβος v έχει πλεόνασμα, τότε υπάρχει στο G_f μια διαδρομή από το v μέχρι την προέλευση s.*

Απόδειξη. Έστω ότι το A συμβολίζει όλους τους κόμβους w για τους οποίους υπάρχει στο υπολοιπόμενο γράφημα G_f μια διαδρομή από το w έως το s, και έστω ότι $B = V - A$. Χρειάζεται μόνο να δείξουμε ότι όλοι οι κόμβοι με πλεόνασμα ανήκουν στο A.

Παρατηρήστε ότι $s \in A$. Επιπρόσθετα, καμία ακμή $e = (x, y)$ που φεύγει από το A δεν μπορεί να έχει θετική ροή, αφού μια ακμή με $f(e) > 0$ θα προκαλούσε μια αντίστροφη ακμή (y, x) στο υπολοιπόμενο γράφημα, και τότε το y θα συμπεριλαμβανόταν στο A. Ας εξετάσουμε τώρα το σύνολο του πλεονάσματος στο σύνολο B, και θυμηθείτε ότι όλοι οι κόμβοι του B έχουν μη αρνητικό πλεόνασμα αφού $s \notin B$.

$$0 \leq \sum_{v \in B} e_f(v) = \sum_{v \in B} (f^{in}(v) - f^{out}(v))$$

Ας αναδιατυπώσουμε το άθροισμα στα δεξιά με τον ακόλουθο τρόπο. Αν μια ακμή e έχει και τις δύο ακμές της στο B, τότε το $f(e)$ εμφανίζεται στο άθροισμα μία φορά με πρόσημο "+" και μία φορά με πρόσημο "–", και έτσι αυτοί οι δύο όροι αλληλοαναιρούνται. Αν η e έχει μόνο το τέλος της στο B τότε η e ξεκινά από το A, και είδαμε προηγουμένως ότι όλες οι ακμές που φεύγουν από το A έχουν $f(e) = 0$. Αν η e έχει μόνο την αρχή της στο B, τότε το $f(e)$ εμφανίζεται μόνο μία φορά στο άθροισμα, με πρόσημο "–". Έτσι έχουμε

$$0 \leq \sum_{v \in B} e_f(v) = -f^{out}(B).$$

Επειδή, οι ροές είναι μη αρνητικές, έχουμε ότι το άθροισμα του πλεονάσματος στο B είναι μηδενικό· αφού το κάθε μεμονωμένο πλεόνασμα στο B είναι μη αρνητικό, θα πρέπει όλα να είναι ίσα με 0. ■

Είμαστε πλέον έτοιμοι για να αποδείξουμε ότι οι ετικέτες δεν αλλάζουν υπερβολικά πολύ. Θυμηθείτε ότι το n συμβολίζει το πλήθος των κόμβων του V.

(7.26) *Σε ολόκληρη την πορεία του αλγορίθμου, όλοι οι κόμβοι έχουν* $h(v) \leq 2n - 1$.

Απόδειξη. Οι αρχικές ετικέτες $h(t) = 0$ και $h(s) = n$ δεν αλλάζουν κατά την πορεία του αλγορίθμου. Ας εξετάσουμε κάποιον άλλον κόμβο $v \neq s, t$. Ο αλγόριθμος τροποποιεί την ετικέτα του v μόνο όταν εφαρμόζει τη λειτουργία `relabel`, έτσι έστω ότι f και h είναι η προροή και η απόδοση ετικετών που επιστρέφονται από μια λειτουργία `relabel`(f, h, v). Με βάση την (7.25) υπάρχει στο υπολοιπόμενο γράφημα G_f μια διαδρομή P από το v έως το s. Έστω ότι το $|P|$ συμβολίζει το πλήθος ακμών της διαδρομής P, και σημειώστε ότι $|P| \leq n - 1$. Η συνθήκη ρύθμισης της κλίσης μάς λέει ότι τα ύψη των κόμβων μπορούν να μειώνονται το πολύ κατά 1 κατά μήκος κάθε ακμής της διαδρομής P, και έτσι $h(v) - h(s) \leq |P|$, που αποδεικνύει τον ισχυρισμό μας. ■

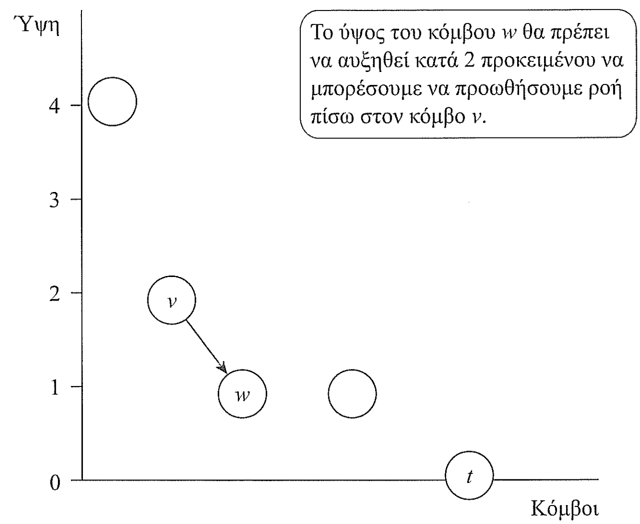
Οι ετικέτες είναι διαρκώς αυξανόμενες κατά την πορεία του αλγορίθμου, έτσι αυτή η πρόταση μας δίνει αμέσως ένα όριο ως προς το πλήθος των λειτουργιών ανανέωσης ετικετών.

(7.27) *Κατά την πορεία του αλγορίθμου αλλάζει η ετικέτα του κάθε κόμβου το πολύ* $2n - 1$ *φορές, και το συνολικό πλήθος των λειτουργιών ανανέωσης ετικετών είναι λιγότερο από* $2n^2$.

Τώρα θα βρούμε ένα όριο για τον αριθμό των λειτουργιών `push`. Θα διακρίνουμε δύο είδη λειτουργιών `push`. Μια λειτουργία `push`(f, h, v, w) είναι *κορεστική* (saturating) εάν είτε η $e = (v, w)$ είναι ευθύδρομη ακμή στο E_f και $\delta = c_e - f(e)$, είτε η (v, w) είναι ανάδρομη ακμή με $e = (w, v)$ και $\delta = f(e)$. Με άλλα λόγια, η λειτουργία προώθησης είναι κορεστική εάν, μετά την προώθηση, η ακμή (v, w) δεν υπάρχει πλέον στο υπολοιπόμενο γράφημα. Όλες οι άλλες λειτουργίες push θα αναφέρονται ως *μη κορεστικές*.

(7.28) *Κατά την πορεία του αλγορίθμου, το πλήθος των κορεστικών λειτουργιών push είναι το πολύ 2nm.*

Απόδειξη. Θεωρήστε μια ακμή (v, w) του υπολοιπόμενου γραφήματος. Μετά από μια κορεστική λειτουργία push(f, h, v, w) έχουμε $h(v) = h(w) + 1$ και η ακμή (v, w) δεν υπάρχει πλέον στο υπολοιπόμενο γράφημα G_f, όπως φαίνεται στην Εικόνα 7.8. Για να μπορέσουμε να εκτελέσουμε ξανά προώθηση σε αυτή την ακμή, θα πρέπει πρώτα να προωθήσουμε το w προς το v έτσι ώστε να κάνουμε την ακμή (v, w) να εμφανιστεί στο υπολοιπόμενο γράφημα. Όμως για να μπορέσουμε να προωθήσουμε από το w προς το v θα πρέπει πρώτα η ετικέτα του w να αυξηθεί κατά τουλάχιστον 2 (έτσι ώστε το w να βρίσκεται πάνω από το v). Η ετικέτα του w μπορεί να αυξηθεί κατά 2 το πολύ n-1 φορές, έτσι η κορεστική προώθηση από το v προς το w μπορεί να προκύψει το πολύ n φορές. Η κάθε ακμή $e \in E$ μπορεί να δημιουργήσει δύο ακμές στο υπολοιπόμενο γράφημα, έτσι συνολικά έχουμε το πολύ $2nm$ κορεστικές προωθήσεις. ■



Εικόνα 7.8 Μετά από μια κορεστική προώθηση (f, h, v, w), το ύψος του v υπερβαίνει το ύψος του w κατά 1.

Το δυσκολότερο τμήμα της ανάλυσης είναι η απόδειξη ενός ορίου για το πλήθος των μη κορεστικών προωθήσεων, και αυτό θα αποτελεί επίσης το περιοριστικό σημείο για το θεωρητικό όριο ως προς το χρόνο εκτέλεσης.

(7.29) *Κατά την πορεία του αλγορίθμου, το πλήθος των μη κορεστικών λειτουργιών* push *είναι το πολύ* $2n^2m$.

Απόδειξη. Για την απόδειξη αυτή θα χρησιμοποιήσουμε μια *μέθοδο συνάρτησης δυναμικού* (potential function method). Για μια προροή f και μια συμβατή απόδοση ετικετών h ορίζουμε ότι

$$\Phi(f, h) = \sum_{v: e_f(v) > 0} h_v$$

είναι το άθροισμα του ύψους όλων των κόμβων με θετικό πλεόνασμα. (Η Φ αναφέρεται συχνά ως *δυναμικό*, επειδή μοιάζει με τη "δυναμική ενέργεια" όλων των κόμβων που έχουν θετικό πλεόνασμα.)

Στην αρχική προροή και απόδοση ετικετών όλοι οι κόμβοι με θετικό πλεόνασμα έχουν ύψος 0, έτσι $\Phi(f, h) = 0$. Το $\Phi(f, h)$ παραμένει μη αρνητικό σε ολόκληρη την πορεία του αλγορίθμου. Μια μη κορεστική λειτουργία push(f, h, v, w) μειώνει το $\Phi(f, h)$ κατά τουλάχιστον 1, επειδή μετά την προώθηση ο κόμβος v δεν θα έχει καθόλου πλεόνασμα και ο κόμβος w, ο μόνος κόμβος που αποκτά νέο πλεόνασμα από τη λειτουργία, βρίσκεται σε ύψος κατά 1 μικρότερο από το ύψος του v. Όμως η κάθε κορεστική λειτουργία push και η κάθε λειτουργία relabel μπορεί να αυξήσει το $\Phi(f, h)$. Μια λειτουργία relabel αυξάνει το $\Phi(f, h)$ κατά ακριβώς 1. Πραγματοποιούνται το πολύ $2n^2$ λειτουργίες relabel, έτσι η συνολική αύξηση του $\Phi(f, h)$ εξαιτίας των λειτουργιών relabel είναι $2n^2$. Οι κορεστικές λειτουργίες push(f, h, v, w) δεν αλλάζουν τις ετικέτες, όμως μπορεί να αυξήσουν το $\Phi(f, h)$ επειδή ο κόμβος w μπορεί ξαφνικά να αποκτήσει θετικό πλεόνασμα μετά από την προώθηση. Αυτό θα αύξανε το $\Phi(f, h)$ κατά το ύψος του w, που είναι το πολύ ίσο με $2n - 1$. Πραγματοποιούνται το πολύ $2nm$ κορεστικές λειτουργίες push, έτσι η συνολική αύξηση στο $\Phi(f, h)$ εξαιτίας αυτών των λειτουργιών push είναι το πολύ $2mn(2n - 1)$. Έτσι, το $\Phi(f, h)$ μπορεί να αυξηθεί κατά το πολύ $4mn^2$ στην πορεία του αλγορίθμου.

Όμως αφού η Φ παραμένει συνεχώς μη αρνητική και μειώνεται κατά τουλάχιστον 1 σε κάθε μη κορεστική λειτουργία push, προκύπτει ότι μπορούν να πραγματοποιούνται το πολύ $4mn^2$ μη κορεστικές λειτουργίες push. ■

Επεκτάσεις: Μια βελτιωμένη εκδοχή του αλγορίθμου

Έχει γίνει πάρα πολλή έρευνα πάνω στο θέμα του καθορισμού κανόνων επιλογής κόμβων για τον αλγόριθμο Προροής-Προώθησης έτσι ώστε να βελτιωθεί ο χρόνος εκτέλεσης στη χειρότερη περίπτωση. Εδώ θα δούμε έναν απλό κανόνα που οδηγεί σε ένα βελτιωμένο όριο $O(n^3)$ ως προς το πλήθος των μη κορεστικών λειτουργιών push.

(7.30) *Αν σε κάθε βήμα επιλέγουμε τον κόμβο με πλεόνασμα που βρίσκεται στο μεγαλύτερο ύψος, τότε το πλήθος των μη κορεστικών λειτουργιών* push *κατά την πορεία του αλγορίθμου είναι* $4n^3$.

Απόδειξη. Θεωρήστε το μέγιστο ύψος $H = \max_{v: e_f(v) > 0} h(v)$ οποιουδήποτε κόμβου με πλεόνασμα καθώς προχωρά ο αλγόριθμος. Η ανάλυση θα χρησιμοποιήσει αυτό το μέγιστο ύψος H στη θέση της συνάρτησης δυναμικού Φ του προηγούμενου ορίου $O(n^2m)$.

Αυτό το μέγιστο ύψος H μπορεί μόνο να αυξάνεται εξαιτίας της ανανέωσης ετικετών (καθώς η ροή προωθείται πάντα σε κόμβους χαμηλότερου ύψους), και έτσι η συνολική αύξηση του H κατά την πορεία του αλγορίθμου είναι το πολύ ίση με $2n^2$, με βάση την (7.26). Το H ξεκινά από το 0 και παραμένει μη αρνητικό, έτσι το πλήθος φορών που αλλάζει το H είναι το πολύ ίσο με $4n^2$.

Ας εξετάσουμε τώρα τη συμπεριφορά του αλγορίθμου σε μία χρονική φάση στην οποία το H παραμένει σταθερό. Ισχυριζόμαστε ότι ο κάθε κόμβος μπορεί να έχει το πολύ μία μη κορεστική λειτουργία push κατά τη διάρκεια αυτής της φάσης. Πράγματι, κατά τη διάρκεια αυτής της φάσης η ροή προωθείται από κόμβους με ύψος H σε κόμβους με ύψους $H - 1$˙ και μετά από μια μη κορεστική λειτουργία push από τον κόμβο v, αυτός θα πρέπει να λάβει ροή από έναν κόμβο με ύψος $H + 1$ πριν να μπορέσουμε να εκτελέσουμε ξανά προώθηση από αυτόν.

Αφού υπάρχουν το πολύ n μη κορεστικές λειτουργίες push μεταξύ κάθε αλλαγής του H, και αφού το H αλλάζει το πολύ $4n^2$ φορές, ο συνολικός αριθμός των μη κορεστικών λειτουργιών push είναι το πολύ $4n^3$. ∎

Ως επακόλουθο της (7.30), είναι ενδιαφέρον να σημειώσουμε ότι πειραματικά το υπολογιστικά χρονοβόρο σημείο της μεθόδου είναι το πλήθος των λειτουργιών ανανέωσης ετικετών, και επιτυγχάνονται καλύτεροι πειραματικοί χρόνοι εκτέλεσης από παραλλαγές που δουλεύουν με αύξηση των ετικετών κατά περισσότερο από ένα. Αυτό το θέμα θα το εξετάσουμε περαιτέρω σε μερικές από τις ασκήσεις.

Υλοποίηση του αλγορίθμου Προροής-Προώθησης

Τέλος, θα πρέπει να εξετάσουμε σε συντομία με ποιον τρόπο μπορεί να υλοποιηθεί αποδοτικά αυτός ο αλγόριθμος. Η διατήρηση μερικών απλών δομών δεδομένων θα μας επιτρέψει να υλοποιήσουμε αποτελεσματικά την κάθε μία από τις λειτουργίες του αλγορίθμου σε σταθερό χρόνο, και έτσι συνολικά να υλοποιήσουμε τον αλγόριθμο σε χρόνο $O(mn)$ συν το πλήθος των μη κορεστικών λειτουργιών push. Έτσι ο γενικός αλγόριθμος θα εκτελείται σε χρόνο $O(mn^2)$, ενώ η εκδοχή του που επιλέγει πάντα τον κόμβο με το μέγιστο ύψος θα εκτελείται σε χρόνο $O(n^3)$.

Μπορούμε να διατηρούμε όλους τους κόμβους με πλεόνασμα σε μια απλή λίστα, και έτσι θα έχουμε τη δυνατότητα να επιλέγουμε έναν τέτοιον κόμβο σε σταθερό χρόνο. Θα πρέπει όμως να είμαστε κάπως προσεκτικοί για να μπορούμε να επιλέγουμε σε σταθερό χρόνο έναν κόμβο με μέγιστο ύψος H. Για να το επιτύχουμε αυτό, θα διατηρούμε μια συνδεδεμένη λίστα με όλους τους κόμβους που έχουν πλεόνασμα σε όλα τα πιθανά ύψη. Σημειώστε ότι, όταν αλλάζει η ετικέτα ενός κόμβου v ή ο κόμβος συνεχίζει να έχει θετικό πλεόνασμα μετά από μια λειτουργία push, αυτός παραμένει ένας κόμβος με το μέγιστο ύψος H. Έτσι χρειάζεται να επιλέγουμε νέο κόμβο μετά από μια λειτουργία push μόνο όταν ο τρέχων κόμβος v δεν έχει πλέον θετικό πλεόνασμα. Αν ο κόμβος v βρισκόταν σε ύψος H, τότε ο νέος κόμβος μέγιστου ύψους θα βρίσκεται επίσης σε ύψος H — ή, αν κανένας από τους κόμβους σε ύψος H δεν έχει πλεόνασμα, τότε το μέγιστο ύψος θα είναι $H - 1$ αφού η προηγούμενη λειτουργία push από τον κόμβο v προώθησε ροή σε έναν κόμβο με ύψος $H - 1$.

Υποθέστε τώρα ότι έχουμε επιλέξει έναν κόμβο v και θέλουμε να επιλέξουμε μια ακμή (v, w) στην οποία θα εφαρμόσουμε τη λειτουργία push(f, h, v, w) (ή τη λειτουργία relabel(f, h, v) αν δεν υπάρχει τέτοιος κόμβος w). Για να μπορούμε να επιλέξουμε γρήγορα μια ακμή, θα χρησιμοποιήσουμε την αναπαράσταση σε λίστες γειτονικότητας

(7.33) *Ο χρόνος εκτέλεσης του αλγορίθμου Προροής-Προώθησης που υλοποιείται με τις παραπάνω δομές δεδομένων είναι $O(mn)$ συν $O(1)$ για κάθε μη κορεστική λειτουργία* push. *Ειδικότερα, ο γενικός αλγόριθμος Προροής-Προώθησης εκτελείται σε χρόνο $O(n^2m)$, ενώ η εκδοχή του όπου επιλέγουμε πάντα τον κόμβο με το μέγιστο ύψος εκτελείται σε χρόνο $O(n^3)$.*

Απόδειξη. Η αρχική ροή και απόδοση ετικετών πραγματοποιούνται σε χρόνο $O(m)$. Και οι δύο λειτουργίες push and relabel μπορούν να υλοποιηθούν σε χρόνο $O(1)$, μετά την επιλογή της λειτουργίας. Ας θεωρήσουμε έναν κόμβο v. Γνωρίζουμε ότι για τον v μπορεί να αλλάξει η ετικέτα του το πολύ $2n$ φορές κατά την πορεία του αλγορίθμου. Θα εξετάσουμε το συνολικό χρόνο που δαπανά ο αλγόριθμος για να βρει τη σωστή ακμή στην οποία θα προωθήσει ακμή από τον κόμβο v, μεταξύ των δύο αποδόσεων ετικέτας στον κόμβο v. Αν ο κόμβος v έχει d_v γειτονικές ακμές, τότε από την (7.32) βλέπουμε ότι δαπανάμε χρόνο $O(d_v)$ για να προχωρήσουμε το δείκτη current(v) μεταξύ των διαδοχικών αποδόσεων ετικέτας στον v. Έτσι ο συνολικός χρόνος που δαπανάται για τη μετακίνηση των δεικτών current κατά την πορεία του αλγορίθμου είναι $O(\sum_{v \in V} nd_v) = O(mn)$, όπως ισχυριστήκαμε. ■

7.5 Μια πρώτη εφαρμογή: Το πρόβλημα του Διμερούς Ταιριάσματος

Αφού αναπτύξαμε κάποιους ισχυρούς αλγορίθμους για το πρόβλημα της Μέγιστης Ροής, θα στραφούμε τώρα στο έργο της ανάπτυξης εφαρμογών με μέγιστες ροές και ελάχιστες αποκοπές σε γραφήματα. Θα ξεκινήσουμε με δύο πολύ βασικές εφαρμογές. Καταρχήν σε αυτή την ενότητα θα μελετήσουμε το πρόβλημα του Διμερούς Ταιριάσματος (Bipartite Matching Problem), το οποίο αναφέραμε στην αρχή αυτού του κεφαλαίου. Στην επόμενη ενότητα θα μελετήσουμε το πιο γενικό *πρόβλημα των Ασύνδετων Διαδρομών* (Disjoint Paths Problem).

Το πρόβλημα

Ένας από τους αρχικούς στόχους μας κατά την ανάλυση του προβλήματος Μέγιστης Ροής ήταν η δυνατότητα επίλυσης του προβλήματος Διμερούς Ταιριάσματος, και τώρα θα δείξουμε πώς γίνεται αυτό. Θυμηθείτε ότι το *διμερές γράφημα* (bipartite graph) $G = (V, E)$ είναι ένα μη κατευθυνόμενο γράφημα του οποίου το σύνολο κόμβων μπορεί να διαμεριστεί ως $V = X \cup Y$, με την ιδιότητα ότι η κάθε ακμή $e \in E$ έχει ένα άκρο στο X και το άλλο άκρο στο Y. Το *ταίριασμα* (matching) M στο G είναι ένα υποσύνολο των ακμών $M \subseteq E$ τέτοιο ώστε ο κάθε κόμβος να εμφανίζεται το πολύ σε μία ακμή του M. Το πρόβλημα του Διμερούς Ταιριάσματος (Bipartite Matching Problem) είναι η εύρεση ενός ταιριάσματος στο G που να έχει το μεγαλύτερο δυνατό μέγεθος.

(7.33) *Ο χρόνος εκτέλεσης του αλγορίθμου Προροής-Προώθησης που υλοποιείται με τις παραπάνω δομές δεδομένων είναι $O(mn)$ συν $O(1)$ για κάθε μη κορεστική λειτουργία* `push`. *Ειδικότερα, ο γενικός αλγόριθμος Προροής-Προώθησης εκτελείται σε χρόνο $O(n^2m)$, ενώ η εκδοχή του όπου επιλέγουμε πάντα τον κόμβο με το μέγιστο ύψος εκτελείται σε χρόνο $O(n^3)$.*

Απόδειξη. Η αρχική ροή και απόδοση ετικετών πραγματοποιούνται σε χρόνο $O(m)$. Και οι δύο λειτουργίες `push` and `relabel` μπορούν να υλοποιηθούν σε χρόνο $O(1)$, μετά την επιλογή της λειτουργίας. Ας θεωρήσουμε έναν κόμβο v. Γνωρίζουμε ότι για τον v μπορεί να αλλάξει η ετικέτα του το πολύ $2n$ φορές κατά την πορεία του αλγορίθμου. Θα εξετάσουμε το συνολικό χρόνο που δαπανά ο αλγόριθμος για να βρει τη σωστή ακμή στην οποία θα προωθήσει ακμή από τον κόμβο v, μεταξύ των δύο αποδόσεων ετικέτας στον κόμβο v. Αν ο κόμβος v έχει d_v γειτονικές ακμές, τότε από την (7.32) βλέπουμε ότι δαπανάμε χρόνο $O(d_v)$ για να προχωρήσουμε το δείκτη `current`(v) μεταξύ των διαδοχικών αποδόσεων ετικέτας στον v. Έτσι ο συνολικός χρόνος που δαπανάται για τη μετακίνηση των δεικτών `current` κατά την πορεία του αλγορίθμου είναι $O(\sum_{v \in V} nd_v) = O(mn)$, όπως ισχυριστήκαμε. ■

7.5 Μια πρώτη εφαρμογή: Το πρόβλημα του Διμερούς Ταιριάσματος

Αφού αναπτύξαμε κάποιους ισχυρούς αλγορίθμους για το πρόβλημα της Μέγιστης Ροής, θα στραφούμε τώρα στο έργο της ανάπτυξης εφαρμογών με μέγιστες ροές και ελάχιστες αποκοπές σε γραφήματα. Θα ξεκινήσουμε με δύο πολύ βασικές εφαρμογές. Καταρχήν σε αυτή την ενότητα θα μελετήσουμε το πρόβλημα του Διμερούς Ταιριάσματος (Bipartite Matching Problem), το οποίο αναφέραμε στην αρχή αυτού του κεφαλαίου. Στην επόμενη ενότητα θα μελετήσουμε το πιο γενικό *πρόβλημα των Ασύνδετων Διαδρομών* (Disjoint Paths Problem).

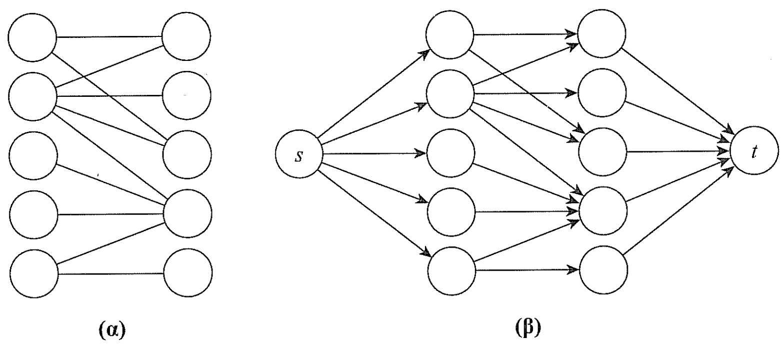
Το πρόβλημα

Ένας από τους αρχικούς στόχους μας κατά την ανάλυση του προβλήματος Μέγιστης Ροής ήταν η δυνατότητα επίλυσης του προβλήματος Διμερούς Ταιριάσματος, και τώρα θα δείξουμε πώς γίνεται αυτό. Θυμηθείτε ότι το *διμερές γράφημα* (bipartite graph) $G = (V, E)$ είναι ένα μη κατευθυνόμενο γράφημα του οποίου το σύνολο κόμβων μπορεί να διαμεριστεί ως $V = X \cup Y$, με την ιδιότητα ότι η κάθε ακμή $e \in E$ έχει ένα άκρο στο X και το άλλο άκρο στο Y. Το *ταίριασμα* (matching) M στο G είναι ένα υποσύνολο των ακμών $M \subseteq E$ τέτοιο ώστε ο κάθε κόμβος να εμφανίζεται το πολύ σε μία ακμή του M. Το πρόβλημα του Διμερούς Ταιριάσματος (Bipartite Matching Problem) είναι η εύρεση ενός ταιριάσματος στο G που να έχει το μεγαλύτερο δυνατό μέγεθος.

Σχεδιασμός του αλγορίθμου

Το γράφημα που ορίζει ένα πρόβλημα ταιριάσματος είναι μη κατευθυνόμενο, ενώ τα δίκτυα ροής είναι κατευθυνόμενα· όμως στην πραγματικότητα δεν είναι δύσκολο να χρησιμοποιήσουμε έναν αλγόριθμο του προβλήματος Μέγιστης Ροής για να βρούμε ένα μέγιστο ταίριασμα.

Ξεκινώντας με το γράφημα G ενός προβλήματος Διμερούς Ταιριάσματος, κατασκευάζουμε ένα δίκτυο ροής G' με τον τρόπο που φαίνεται στην Εικόνα 7.9. Καταρχήν κατευθύνουμε όλες τις ακμές του G από το X προς το Y. Στη συνέχεια προσθέτουμε έναν κόμβο s και μια ακμή (s, x) από το s προς όλους τους κόμβους του X. Προσθέτουμε μετά έναν κόμβο t και μια ακμή (y, t) από κάθε κόμβο του Y προς το t. Τέλος, αποδίδουμε σε κάθε ακμή του G' χωρητικότητα ίση με 1.



Εικόνα 7.9 (α) Ένα διμερές γράφημα. (β) Το αντίστοιχο δίκτυο ροής, όπου όλες οι χωρητικότητες είναι ίσες με 1.

Υπολογίζουμε τώρα μια μέγιστη ροή s-t σε αυτό το δίκτυο G'. Θα διαπιστώσουμε ότι η τιμή αυτής της μέγιστης ροής είναι ίση με το μέγεθος του μέγιστου ταιριάσματος στο G. Επιπρόσθετα, στην ανάλυσή μας θα δούμε πώς μπορούμε να χρησιμοποιήσουμε την ίδια τη ροή για να ανακτήσουμε το ταίριασμα.

Ανάλυση του αλγορίθμου

Η ανάλυση βασίζεται στο να δείξουμε ότι οι ροές με ακέραιες τιμές στο G' κωδικοποιούν με επαρκώς διαφανή τρόπο τα ταιριάσματα στο G. Καταρχήν υποθέστε ότι υπάρχει στο G ένα ταίριασμα που αποτελείται από k ακμές (x_{i1}, y_{i1}), ..., (x_{ik}, y_{ik}). Εξετάστε τώρα τη ροή f που στέλνει μια μονάδα κατά μήκος κάθε διαδρομής της μορφής s, x_{ij}, y_{ij}, t — με άλλα λόγια, $f(e) = 1$ για όλες τις ακμές σε αυτές τις διαδρομές. Μπορούμε εύκολα να επαληθεύσουμε ότι πράγματι ικανοποιούνται τα κριτήρια χωρητικότητας και διατήρησης και ότι η f είναι μια ροή s-t με τιμή k.

Αντίστροφα, υποθέστε ότι υπάρχει στο G' μια ροή f' με τιμή k. Από το θεώρημα ακέραιων τιμών για τις μέγιστες ροές (7.14) γνωρίζουμε ότι υπάρχει μια ακέραια ροή f με τιμή k· και αφού όλες οι χωρητικότητες είναι 1, αυτό σημαίνει ότι το $f(e)$ θα είναι ίσο είτε με 0 είτε με 1 για την κάθε ακμή e. Εξετάστε τώρα το σύνολο M' των ακμών της μορφής (x, y) για τις οποίες η τιμή της ροής είναι 1.

Οι ακόλουθες είναι μερικές απλές παρατηρήσεις σχετικά με αυτό το σύνολο M'.

(7.34) *Το M' περιέχει k ακμές.*

Απόδειξη. Για να το αποδείξουμε αυτό, θεωρήστε την αποκοπή (A, B) στο G' με $A = \{s\} \cup X$. Η τιμή της ροής είναι η συνολική ροή που εξέρχεται από το A μείον τη συνολική ροή που εισέρχεται στο A. Ο πρώτος από αυτούς τους όρους είναι απλώς ο πληθάριθμος του M', αφού αυτές είναι οι ακμές που εξέρχονται από το A και μεταφέρουν ροή, και η καθεμία από αυτές μεταφέρει μία μονάδα ροής. Ο δεύτερος από αυτούς τους όρους είναι 0, αφού δεν υπάρχουν ακμές που να εισέρχονται στο A. Άρα το M' περιέχει k ακμές. ■

(7.35) *Ο κάθε κόμβος του X αποτελεί αρχή για το πολύ μία ακμή στο M'.*

Απόδειξη. Για να το αποδείξουμε αυτό, υποθέστε ότι ο κόμβος $x \in X$ αποτελεί αρχή για τουλάχιστον δύο ακμές στο M'. Αφού η ροή μας έχει ακέραιες τιμές, αυτό σημαίνει ότι από το x εξέρχονται τουλάχιστον δύο μονάδες ροής. Με βάση την αρχή της διατήρησης της ροής θα πρέπει να εισέρχονται στο x τουλάχιστον δύο μονάδες ροής — όμως αυτό δεν είναι δυνατό, επειδή στο x εισέρχεται μόνο μία ακμή με χωρητικότητα 1. Άρα το x αποτελεί αρχή για το πολύ μία ακμή στο M'. ■

Με την ίδια συλλογιστική μπορούμε να δείξουμε ότι

(7.36) *Ο κάθε κόμβος του Y αποτελεί τέλος για το πολύ μία ακμή στο M'.*

Συνδυάζοντας όλα τα παραπάνω βλέπουμε ότι, αν θεωρήσουμε το M' ένα σύνολο ακμών στο αρχικό διμερές γράφημα G, παίρνουμε ένα ταίριασμα με μέγεθος k. Συνοψίζοντας, έχουμε αποδείξει το ακόλουθο γεγονός.

(7.37) *Το μέγεθος του μέγιστου ταιριάσματος στο G είναι ίσο με την τιμή της μέγιστης ροής στο G'· και οι ακμές σε ένα τέτοιο ταίριασμα του G είναι οι ακμές που μεταφέρουν ροή από το* Χ *στο* Υ *για το γράφημα G'.*

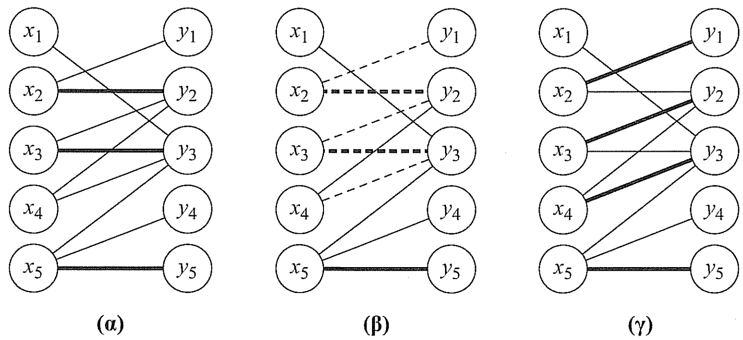
Σημειώστε τον κρίσιμο τρόπο με τον οποίο το θεώρημα ακέραιων τιμών (7.14) χρησιμοποιήθηκε στην ανάλυση: χρειαζόμασταν να γνωρίζουμε αν υπάρχει στο G' μια μέγιστη ροή που να παίρνει μόνο τιμές 0 και 1.

Εύρεση ορίων ως προς το χρόνο εκτέλεσης. Ας εξετάσουμε τώρα πόσο γρήγορα μπορούμε να υπολογίσουμε ένα μέγιστο ταίριασμα στο G. Έστω ότι $n = |X| = |Y|$, και έστω ότι m είναι το πλήθος ακμών στο G. Θα υποθέσουμε σιωπηρά ότι υπάρχει τουλάχιστον μία ακμή που συνδέεται με κάθε κόμβο στο αρχικό πρόβλημα, και έτσι $m \geq n/2$. Ο χρόνος που απαιτείται για τον υπολογισμό ενός μέγιστου ταιριάσματος είναι κατά κύριο λόγο ο χρόνος που απαιτείται για τον υπολογισμό μιας ακέραιης μέγιστης ροής στο G', αφού η μετατροπή της σε ταίριασμα στο G είναι απλή υπόθεση. Για αυτό το πρόβλημα ροής έχουμε ότι $C = \sum_{e \text{ από το } s} c_e = |\mathrm{X}| = n$, επειδή το s έχει μια ακμή χωρητικότητας 1 προς κάθε κόμβο του X. Έτσι, αν χρησιμοποιήσουμε το χρονικό όριο $O(mC)$ από την (7.5), βρίσκουμε το εξής.

(7.38) *Ο αλγόριθμος Ford-Fulkerson μπορεί να χρησιμοποιηθεί για την εύρεση ενός μέγιστου ταιριάσματος σε ένα διμερές γράφημα σε χρόνο $O(mn)$.*

Είναι ενδιαφέρον ότι, αν χρησιμοποιούσαμε εδώ τα "καλύτερα" όρια $O(m^2 \log_2 C)$ ή $O(n^3)$ που βρήκαμε στις προηγούμενες ενότητες, θα παίρναμε για αυτό το πρόβλημα τους υποδεέστερους χρόνους εκτέλεσης $O(m^2 \log n)$ ή $O(n^3)$. Δεν υπάρχει τίποτα το περίεργο σε αυτό. Τα όρια εκείνα ήταν σχεδιασμένα έτσι ώστε να είναι καλά για *όλα* τα στιγμιότυπα, ακόμα και όταν το C είναι πολύ μεγάλο σε σχέση με το m και το n. Όμως για το πρόβλημα του Διμερούς Ταιριάσματος έχουμε $C = n$, και έτσι δεν είναι απαραίτητο το κόστος για την πρόσθετη περιπλοκότητα.

Αξίζει τον κόπο να εξετάσουμε τι σημαίνουν οι διαδρομές επαύξησης στο δίκτυο G'. Θεωρήστε το ταίριασμα M που αποτελείται από τις ακμές (x_2, y_2), (x_3, y_3), και (x_5, y_5) στο διμερές γράφημα της Εικόνας 7.1· δείτε επίσης την Εικόνα 7.10. Έστω ότι f είναι η αντίστοιχη ροή στο G'. Αυτό το ταίριασμα δεν είναι μέγιστο, έτσι η f δεν είναι μέγιστη ροή s-t άρα υπάρχει μια διαδρομή επαύξησης στο υπολοιπόμενο γράφημα G'_f. Μια τέτοια διαδρομή επαύξησης επισημαίνεται στην Εικόνα 7.10(β). Σημειώστε ότι οι ακμές (x_2, y_2) και (x_3, y_3) χρησιμοποιούνται "ανάδρομα", ενώ όλες οι άλλες ακμές "ευθύδρομα". Όλες οι διαδρομές επαύξησης θα πρέπει να εναλλάσσουν μεταξύ ακμών που χρησιμοποιούνται ανάδρομα και ευθύδρομα, αφού όλες οι ακμές στο γράφημα G' πηγαίνουν από το X στο Y. Έτσι οι διαδρομές επαύξησης ονομάζονται επίσης και *διαδρομές εναλλαγής* (alternating paths) στα πλαίσια της εύρεσης ενός μέγιστου ταιριάσματος. Η επίδραση αυτής της επαύξησης είναι να βγάλει από το ταίριασμα τις διαδρομές που πηγαίνουν ανάδρομα και να τις αντικαταστήσει με ακμές που πηγαίνουν ευθύδρομα. Αφού η διαδρομή επαύξησης πηγαίνει από το s στο t, θα υπάρχει μία επιπλέον ευθύδρομη ακμή από τις ανάδρομες ακμές· έτσι το μέγεθος του ταιριάσματος αυξάνεται κατά ένα.



Εικόνα 7.10 (α) Ένα διμερές γράφημα, με ένα ταίριασμα M. (β) Η διαδρομή επαύξησης στο αντίστοιχο υπολοιπόμενο γράφημα. (γ) Το ταίριασμα που επιτυγχάνεται από την επαύξηση.

Επεκτάσεις: Η δομή των διμερών γραφημάτων που δεν έχουν τέλειο ταίριασμα

Αλγοριθμικά έχουμε ήδη δει πώς μπορούμε να βρούμε τέλεια ταιριάσματα: Χρησιμοποιούμε τον παραπάνω αλγόριθμο για να βρούμε ένα μέγιστο ταίριασμα και μετά ελέγχουμε αν αυτό το ταίριασμα είναι τέλειο.

Ας διερωτηθούμε όμως μια ελαφρώς λιγότερο αλγοριθμική ερώτηση. Δεν έχουν όλα τα διμερή γραφήματα τέλεια ταιριάσματα. Ποια θα είναι η μορφή ενός διμερούς γραφήματος που δεν έχει τέλειο ταίριασμα; Υπάρχει κάποιος εύκολος τρόπος για να δούμε ότι ένα διμερές γράφημα δεν έχει τέλειο ταίριασμα — ή τουλάχιστον ένας εύκολος τρόπος για να πείσουμε κάποιον τρίτο, μετά την εκτέλεση του αλγορίθμου, ότι το γράφημα δεν έχει τέλειο ταίριασμα; Πιο συγκεκριμένα, θα ήταν ωραία αν ο αλγόριθμος, όταν έφτανε στο συμπέρασμα ότι δεν υπάρχει τέλειο ταίριασμα, μπορούσε να παράγει ένα σύντομο "πιστοποιητικό" αυτού του γεγονότος. Αυτό το πιστοποιητικό θα επέτρεπε να πείσουμε κάποιον στα γρήγορα ότι δεν υπάρχει τέλειο ταίριασμα, χωρίς να χρειάζεται να κοιτάξει ολόκληρη την πορεία εκτέλεσης του αλγορίθμου.

Ένας τρόπος για να κατανοήσουμε την ιδέα ενός τέτοιου πιστοποιητικού είναι ο ακόλουθος. Μπορούμε να αποφασίσουμε αν το γράφημα G έχει τέλειο ταίριασμα ελέγχοντας αν η μέγιστη ροή σε ένα σχετιζόμενο γράφημα G' έχει τιμή τουλάχιστον n. Από το θεώρημα Μέγιστης Ροής και Ελάχιστης Αποκοπής γνωρίζουμε ότι θα υπάρχει μια αποκοπή s-t με χωρητικότητα μικρότερη από n αν η τιμή της μέγιστης ροής στο G' είναι μικρότερη από n. Έτσι, κατά μία έννοια, μια αποκοπή με χωρητικότητα μικρότερη από n παρέχει ένα τέτοιο πιστοποιητικό. Θέλουμε όμως ένα πιστοποιητικό που θα έχει φυσική σημασία στα πλαίσια του αρχικού γραφήματος G.

Με τι μπορεί να μοιάζει ένα τέτοιο πιστοποιητικό; Για παράδειγμα, αν υπάρχουν κόμβοι $x_1, x_2 \in X$ που έχουν μόνο μία προσκείμενη ακμή ο καθένας τους και το άλλο άκρο της ακμής είναι ο ίδιος κόμβος y, τότε είναι σαφές ότι το γράφημα δεν διαθέτει τέλειο ταίριασμα: και το x_1 και το x_2 θα πρέπει να ταιριάξουν με τον ίδιο κόμβο y. Γενικότερα, θεωρήστε ένα υποσύνολο κόμβων $A \subseteq X$, και έστω ότι το $\Gamma(A) \subseteq Y$ συμβολίζει το σύνολο όλων των κόμβων που είναι "γειτονικοί" με τους κόμβους του A. Αν το γράφημα διαθέτει τέλειο ταίριασμα τότε ο κάθε κόμβος του A θα πρέπει να ταιριάζει με διαφορετικό κόμβο του $\Gamma(A)$, έτσι το $\Gamma(A)$ θα πρέπει να είναι τουλάχιστον εξίσου μεγάλο με το A. Αυτό μας δίνει το ακόλουθο γεγονός.

(7.39) *Αν ένα διμερές γράφημα $G = (V, E)$ με διαμέριση X και Y διαθέτει τέλειο ταίριασμα, τότε για όλα τα $A \subseteq X$ θα πρέπει να έχουμε $|\Gamma(A)| \geq |A|$.*

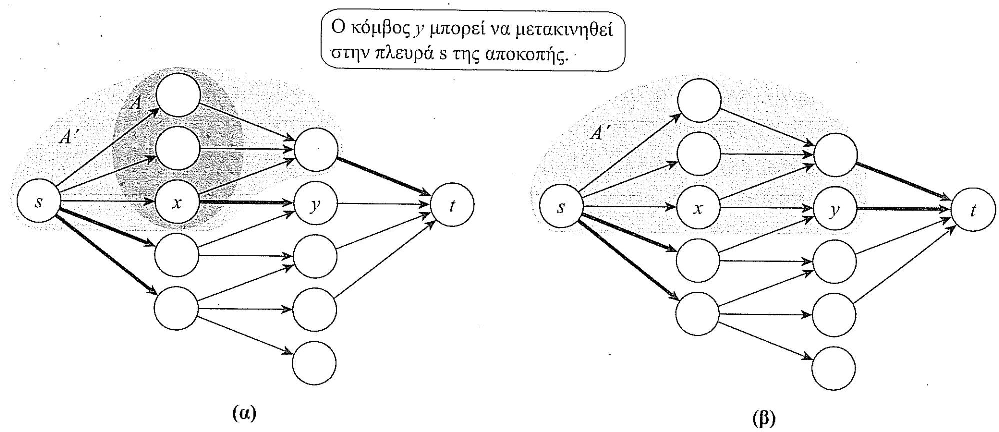
Η πρόταση αυτή μας παρέχει έναν τύπο πιστοποιητικού που αποδεικνύει ότι ένα γράφημα δεν διαθέτει τέλειο ταίριασμα: ένα σύνολο $A \subseteq X$ τέτοιο ώστε $|\Gamma(A)| < |A|$. Είναι όμως επίσης αληθές και το αντίστροφο της (7.39); Αληθεύει ότι όταν δεν υπάρχει τέλειο ταίριασμα, υπάρχει ένα τέτοιο σύνολο A που να το αποδεικνύει; Η απάντηση αποδεικνύεται ότι είναι καταφατική, υπό τον όρο ότι θα προσθέσουμε την προφανή συνθήκη $|X| = |Y|$ (χωρίς την οποία προφανώς δεν θα είχαμε ένα τέλειο ταίριασμα). Αυτή η πρόταση είναι γνωστή στη βιβλιογραφία ως *Θεώρημα του Hall* (Hall's Theorem), αν και εκδοχές της έχουν ανακαλυφθεί ανεξάρτητα από πολλούς άλλους — ίσως ο πρώτος να ήταν ο König — στις αρχές του 20ου αιώνα. Η απόδειξη αυτής της πρότασης μας παρέχει επίσης έναν τρόπο εύρεσης ενός τέτοιου υποσυνόλου A σε πολυωνυμικό χρόνο.

(7.40) *Υποθέστε ότι το διμερές γράφημα $G = (V, E)$ έχει διαμέριση X και Y τέτοια ώστε $|X| = |Y|$. Τότε το γράφημα G είτε διαθέτει τέλειο ταίριασμα, ή υπάρχει ένα υποσύνολο $A \subseteq X$ τέτοιο ώστε $|\Gamma(A)| < |A|$. Το τέλειο ταίριασμα ή το κατάλληλο υποσύνολο A μπορούν να βρεθούν σε χρόνο $O(mn)$.*

Απόδειξη. Θα χρησιμοποιήσουμε το ίδιο γράφημα G' όπως στην (7.37). Υποθέστε ότι $|X| = |Y| = n$. Από την (7.37) γνωρίζουμε ότι το γράφημα G διαθέτει τέλειο ταίριασμα εάν και μόνο εάν η τιμή της μέγιστης ροής του G' είναι n.

Πρέπει να δείξουμε ότι, αν η τιμή της μέγιστης ροής είναι μικρότερη από n, τότε υπάρχει ένα υποσύνολο A τέτοιο ώστε $|\Gamma(A)| < |A|$, όπως ισχυρίζεται η πρότασή μας. Από το θεώρημα Μέγιστης Ροής και Ελάχιστης Αποκοπής (7.12) γνωρίζουμε ότι, αν η τιμή της μέγιστης ροής είναι μικρότερη από n, τότε υπάρχει στο G' μια αποκοπή (A', B') με χωρητικότητα μικρότερη από n. Τώρα το σύνολο A' περιλαμβάνει το s και μπορεί να περιέχει κόμβους και από το X και από το Y, όπως φαίνεται στην Εικόνα 7.11. Ισχυριζόμαστε ότι το σύνολο $A = X \cap A'$ διαθέτει την επιθυμητή ιδιότητα. Αυτό θα αποδείξει και τα δύο μέρη της πρότασης, αφού έχουμε δει στην (7.11) ότι η ελάχιστη αποκοπή (A', B') μπορεί επίσης να βρεθεί με εκτέλεση του αλγορίθμου Ford-Fulkerson.

Καταρχήν ισχυριζόμαστε ότι μπορούμε να τροποποιήσουμε την ελάχιστη αποκοπή (A', B') έτσι ώστε να εξασφαλίσουμε ότι $\Gamma(A) \subseteq A'$, όπου $A = X \cap A'$ όπως και προηγουμένως. Για να το επιτύχουμε αυτό, θεωρήστε έναν κόμβο $y \in \Gamma(A)$ ο οποίος ανήκει στο B', όπως φαίνεται στην Εικόνα 7.11(α). Ισχυριζόμαστε ότι με τη μετακίνηση του y από το B' στο A' δεν αυξάνουμε τη χωρητικότητα της αποκοπής. Τι συμβαίνει όταν μετακινούμε τον κόμβο y από το B' στο A'; Η ακμή (y, t) διασχίζει πλέον την αποκοπή, αυξάνοντας τη χωρητικότητα κατά ένα. Όμως προηγουμένως υπήρχε *τουλάχιστον* μία ακμή (x, y) με $x \in A$, αφού $y \in \Gamma(A)$˙ όλες αυτές οι ακμές από το A και το y διέτρεχαν την αποκοπή, όμως αυτό δεν συμβαίνει πια. Έτσι συνολικά η χωρητικότητα της αποκοπής δεν μπορεί να αυξάνεται. (Σημειώστε ότι δεν χρειάζεται να ασχοληθούμε με τους κόμβους $x \in X$ που δεν υπάρχουν στο A. Τα δύο άκρα της ακμής (x, y) θα βρίσκονται σε διαφορετικές πλευρές της αποκοπής, όμως αυτή η ακμή δεν αυξάνει τη χωρητικότητα της αποκοπής επειδή πηγαίνει από το B' στο A'.)



Εικόνα 7.11 (α) Μια ελάχιστη αποκοπή για απόδειξη της πρότασης (7.40). (β) Η ίδια αποκοπή μετά τη μετακίνηση του κόμβου y στην πλευρά A'. Οι ακμές που διασχίζουν την αποκοπή εμφανίζονται με πιο παχιές γραμμές.

Θα εξετάσουμε τώρα τη χωρητικότητα αυτής της ελάχιστης αποκοπής (A', B') που έχει $\Gamma(A) \subseteq A'$ όπως φαίνεται στην Εικόνα 7.11(β). Αφού όλοι οι γείτονες του A ανήκουν στο A', βλέπουμε ότι οι μόνες ακμές που εξέρχονται από το A' είναι είτε ακμές που φεύγουν από την προέλευση s, είτε ακμές που εισέρχονται στην απόληξη t. Έτσι η χωρητικότητα της αποκοπής είναι ακριβώς

$$c(A', B') = |X \cap B'| + |Y \cap A'|.$$

Παρατηρήστε ότι $|X \cap B'| = n - |A|$ και $|Y \cap A'| \geq |\Gamma(A)|$. Η παραδοχή ότι $c(A', B') < n$ μας δείχνει ότι

$$n - |A| + |\Gamma(A)| \leq |X \cap B'| + |Y \cap A'| = c(A', B') < n.$$

Συγκρίνοντας τον πρώτο και τον τελευταίο όρο βρίσκουμε τη ζητούμενη ανισότητα $|A| > |\Gamma(A)|$. ■

7.6 Ασύνδετες διαδρομές σε κατευθυνόμενα και μη κατευθυνόμενα γραφήματα

Στην Ενότητα 7.1 περιγράψαμε τη ροή f ως ένα είδος "κίνησης" σε ένα δίκτυο. Όμως ο πραγματικός μας ορισμός για τη ροή έχει μια πολύ πιο στατική "αίσθηση": για κάθε ακμή e, προσδιορίζουμε απλώς έναν αριθμό $f(e)$ που δηλώνει την ποσότητα ροής που περνάει από την e. Ας δούμε μήπως μπορούμε να αποκαταστήσουμε κάπως την πιο δυναμική εικόνα που σχετίζεται με την κίνηση, και να προσπαθήσουμε να τυποποιήσουμε την έννοια των μονάδων ροής που "ταξιδεύουν" από την προέλευση στην απόληξη. Από αυτή την πιο δυναμική θεώρηση των ροών θα καταλήξουμε στο ονομαζόμενο *Πρόβλημα των Ασύνδετων Διαδρομών s-t* (*s-t* Disjoint Paths Problem).

Το πρόβλημα

Για να ορίσουμε με ακρίβεια αυτό το πρόβλημα, θα ασχοληθούμε με δύο ζητήματα. Πρώτον, θα διατυπώσουμε με ακρίβεια τη διαισθητική αντιστοιχία μεταξύ των μονάδων ροής που ταξιδεύουν κατά μήκος διαδρομών και της έννοιας της ροής που έχουμε μελετήσει μέχρι τώρα. Δεύτερον, θα επεκτείνουμε το πρόβλημα των Ασύνδετων Διαδρομών σε *μη κατευθυνόμενα* γραφήματα. Θα δούμε ότι, αν και το πρόβλημα της Μέγιστης Ροής είχε οριστεί για ένα κατευθυνόμενο γράφημα, μπορεί να χρησιμοποιηθεί με φυσικό τρόπο για την αντιμετώπιση συναφών προβλημάτων σε μη κατευθυνόμενα γραφήματα.

Λέμε ότι ένα σύνολο διαδρομών είναι *ασύνδετο ως προς τις ακμές* (edge-disjoint) αν τα σύνολα των ακμών τους είναι ξένα, δηλαδή αν δεν υπάρχουν δύο διαδρομές που να έχουν κοινή ακμή, αν και πολλές διαδρομές μπορούν να διέρχονται από κάποιους κοινούς κόμβους. Με δεδομένο ένα κατευθυνόμενο γράφημα $G = (V, E)$ με δύο διακεκριμένους κόμβους s, $t \in V$, το πρόβλημα των *Κατευθυνόμενων Διαδρομών με Ασύνδετες Ακμές* (Directed Edge-Disjoint Paths Problem) συνίσταται στο να βρούμε το μέγιστο αριθμό διαδρομών s-t στο G που είναι ασύνδετες ως προς τις ακμές. Το πρόβλημα των *Μη Κατευθυνόμενων Διαδρομών με Ασύνδετες Ακμές* (Undirected Edge-Disjoint Paths Problem) είναι να βρούμε το μέγιστο αριθμό διαδρομών s-t στο μη κατευθυνόμενο γράφημα G που είναι ασύνδετες ως προς τις ακμές. Το σχετιζόμενο πρόβλημα της εύρεσης διαδρομών που είναι όχι μόνο ασύνδετες ως προς τις ακμές, αλλά είναι ασύνδετες και ως προς τους κόμβους (προφανώς, με εξαίρεση του κόμβους s και t) θα εξεταστεί στις ασκήσεις αυτού του κεφαλαίου.

Σχεδιασμός του αλγορίθμου

Και η κατευθυνόμενη και η μη κατευθυνόμενη εκδοχή του προβλήματος μπορούν να επιλυθούν με πολύ φυσικό τρόπο με τη χρήση ροών. Ας ξεκινήσουμε με το κατευθυνόμενο πρόβλημα. Με δεδομένο το γράφημα $G = (V, E)$, με τους δύο διακεκριμένους κόμβους s και t, ορίζουμε ένα δίκτυο ροής στο οποίο οι s και t είναι αντίστοιχα η προέλευση και η απόληξη, και υπάρχει χωρητικότητα 1 σε κάθε ακμή. Υποθέστε τώρα ότι υπάρχουν k διαδρομές s-t που είναι ασύνδετες ως προς τις ακμές. Μπορούμε να κάνουμε την κάθε μία από αυτές τις διαδρομές να μεταφέρει μία μονάδα ροής. Ορίζουμε τη ροή να είναι $f(e) = 1$ σε κάθε ακμή e για οποιαδήποτε από τις διαδρομές και $f(e') = 0$ για όλες τις άλλες ακμές, και αυτό μας ορίζει μια εφικτή ροή με τιμή k.

(7.41) *Αν υπάρχουν k ασύνδετες ως προς τις ακμές διαδρομές από το s προς το t σε ένα κατευθυνόμενο γράφημα G, τότε η τιμή της μέγιστης ροής s-t στο G είναι τουλάχιστον ίση με k.*

Υποθέστε ότι θα μπορούσαμε επίσης να αποδείξουμε το αντίστροφο της πρότασης (7.41): Αν υπάρχει μια ροή με τιμή k, τότε υπάρχουν k ασύνδετες ως προς τις ακμές διαδρομές s-t. Τότε θα μπορούσαμε απλώς να υπολογίσουμε μια μέγιστη ροή s-t στο G και να δηλώσουμε (σωστά) ότι αυτό είναι το μέγιστο πλήθος διαδρομών s-t που είναι ασύνδετες ως προς τις ακμές.

Θα προχωρήσουμε τώρα στην απόδειξη αυτής της αντίστροφης πρότασης, επιβεβαιώνοντας ότι αυτή η προσέγγιση με τη χρήση ροών μας δίνει πράγματι τη σωστή απάντηση. Η ανάλυσή μας θα μας δώσει επίσης έναν τρόπο για την εξαγωγή k ασύνδετων ως προς τις ακμές διαδρομών από μια ακέραιη ροή που στέλνει k μονάδες από το s στο t. Έτσι ο υπολογισμός μιας μέγιστης ροής στο G δεν θα μας δώσει μόνο το μέγιστο *πλήθος* διαδρομών που είναι ασύνδετες ως προς τις ακμές, αλλά θα μας δώσει και τις ίδιες τις διαδρομές.

Ανάλυση του αλγορίθμου

Η απόδειξη της αντίστροφης κατεύθυνσης της (7.41) είναι η καρδιά της ανάλυσής μας, αφού θα καταδείξει αμέσως το βέλτιστο της χρήσης αλγορίθμων που βασίζονται σε ροές για την εύρεση ασύνδετων διαδρομών.

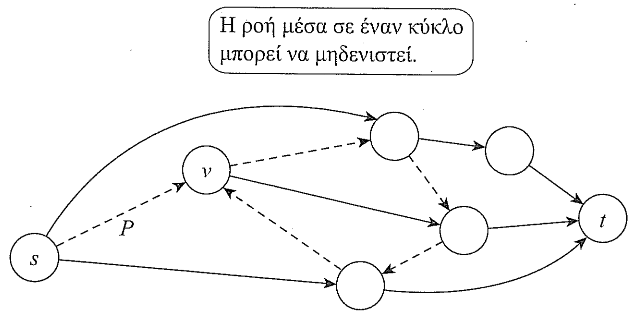
Για να το αποδείξουμε αυτό, θα εξετάσουμε μια ροή που έχει τιμή τουλάχιστον k και θα κατασκευάσουμε k διαδρομές που είναι ασύνδετες ως προς τις ακμές. Από την (7.14) γνωρίζουμε ότι υπάρχει μια μέγιστη ροή f με ακέραιες τιμές ροής. Αφού όλες οι ακμές έχουν όριο χωρητικότητας 1, και αφού η ροή έχει ακέραιες τιμές, η κάθε ακμή που μεταφέρει ροή στην f έχει ακριβώς μία μονάδα ροής. Έτσι χρειάζεται μόνο να αποδείξουμε το ακόλουθο γεγονός.

(7.42) *Αν η f είναι μια ροή 0-1 με τιμή ν, τότε το σύνολο των ακμών που έχουν τιμή ροής $f(e) = 1$ περιλαμβάνει ένα σύνολο από ν ασύνδετες ως προς τις ακμές διαδρομές.*

Απόδειξη. Θα το αποδείξουμε αυτό με επαγωγή ως προς το πλήθος ακμών της f που μεταφέρουν ροή. Αν $\nu = 0$ δεν υπάρχει τίποτα για να αποδείξουμε. Διαφορετικά, θα πρέπει να υπάρχει μια ακμή (s, u) που μεταφέρει μία μονάδα ροής. Τώρα "ακολουθούμε" μια διαδρομή από ακμές που πρέπει επίσης να μεταφέρουν ροή: Αφού η ακμή (s, u) μεταφέρει μια μονάδα ροής, προκύπτει από τη συνθήκη διατήρησης ότι υπάρχει κάποια ακμή (u, v) που μεταφέρει μία μονάδα ροής, και μετά πρέπει να υπάρχει κάποια ακμή (v, w) που μεταφέρει μία μονάδα ροής, κ.ο.κ.. Αν συνεχίσουμε με αυτόν τον τρόπο, τελικά θα συμβεί κάποιο από τα εξής δύο ενδεχόμενα. Είτε θα φτάσουμε στο t, ή θα φτάσουμε για δεύτερη φορά σε έναν κόμβο v.

Αν συμβεί η πρώτη περίπτωση — βρούμε μια διαδρομή P από το s στο t — τότε χρησιμοποιούμε αυτή τη διαδρομή ως μία από τις ν διαδρομές μας. Έστω ότι f' είναι η ροή που παίρνουμε αν μειώσουμε σε τιμή 0 τις τιμές ροής στις ακμές της διαδρομής P. Αυτή η νέα ροή f' έχει τιμή $\nu - 1$, και έχει λιγότερες ακμές που μεταφέρουν ροή. Εφαρμόζοντας την υπόθεση της επαγωγής στη ροή f', παίρνουμε $\nu - 1$ διαδρομές που είναι ασύνδετες ως προς τις ακμές· και αυτό, μαζί με τη διαδρομή P, μας δίνει τις ν διαδρομές που ισχυριστήκαμε.

Αν η διαδρομή P φτάσει για δεύτερη φορά σε έναν κόμβο v, τότε έχουμε μια κατάσταση όπως αυτή που φαίνεται στην Εικόνα 7.12. (Οι ακμές στην εικόνα μεταφέρουν όλες μία μονάδα ροής, και οι διακεκομμένες γραμμές δείχνουν τη διαδρομή που έχουμε διατρέξει μέχρι στιγμής, η οποία μόλις έχει φτάσει για δεύτερη φορά σε έναν κόμβο v.) Στην περίπτωση αυτή μπορούμε να σημειώσουμε πρόοδο με διαφορετικό τρόπο.



Εικόνα 7.12 Οι ακμές στην εικόνα μεταφέρουν όλες μία μονάδα ροής. Η διαδρομή P με τις διακεκομμένες ακμές είναι μία πιθανή διαδρομή για την απόδειξη της πρότασης (7.42).

Εξετάστε τον κύκλο C με τις ακμές που επισκεφθήκαμε μεταξύ της πρώτης και της δεύτερης εμφάνισης του v. Μπορούμε να πάρουμε μια νέα ροή f' από την f μειώνοντας σε 0 όλες τις ακμές κατά μήκος του κύκλου C. Αυτή η νέα ροή f' έχει τιμή ν, έχει όμως λιγότερες ακμές που μεταφέρουν ροή. Εφαρμόζοντας την υπόθεση της επαγωγής στην f', παίρνουμε τις ν ασύνδετες ως προς τις ακμές διαδρομές που ισχυριστήκαμε. ■

Μπορούμε να συνοψίσουμε τις (7.41) και (7.42) στο ακόλουθο αποτέλεσμα.

(7.43) *Υπάρχουν k ασύνδετες ως προς τις ακμές διαδρομές σε ένα κατευθυνόμενο γράφημα G από το s στο t εάν και μόνο εάν η μέγιστη τιμή ροής s-t στο G είναι τουλάχιστον k.*

Παρατηρήστε επίσης με ποιον τρόπο η απόδειξη της (7.42) μας δίνει μια πραγματική διαδικασία για την κατασκευή των k διαδρομών, όταν είναι δεδομένη μια μέγιστη ροή με ακέραιες τιμές για το γράφημα G. Αυτή η διαδικασία αναφέρεται κάποιες φορές ως *διάσπαση* (path decomposition) της ροής σε διαδρομές, επειδή "διασπά" τη ροή σε ένα συστατικό σύνολο διαδρομών. Έτσι έχουμε δείξει ότι ο αλγόριθμός μας που βασίζεται σε ροές βρίσκει το μέγιστο πλήθος διαδρομών s-t που είναι ασύνδετες ως προς τις ακμές και παρέχει επίσης έναν τρόπο για τη δόμηση των πραγματικών διαδρομών.

Εύρεση ορίου ως προς το χρόνο εκτέλεσης. Για αυτό το πρόβλημα ροής έχουμε $C = \sum_{e \text{ από το } s} c_e \leq |V| = n$, αφού υπάρχουν το πολύ $|V|$ ακμές που εξέρχονται από τον κόμβο s και η καθεμία από αυτές έχει χωρητικότητα 1. Έτσι, χρησιμοποιώντας το όριο $O(mC)$ που βρήκαμε στην (7.5), μπορούμε να πάρουμε μια ακέραια μέγιστη ροή σε χρόνο $O(mn)$.

Η διαδικασία διάσπασης ροής σε διαδρομές που είδαμε στην απόδειξη της (7.42), η οποία παράγει τις ίδιες τις διαδρομές, μπορεί επίσης να εκτελεστεί σε χρόνο $O(mn)$. Για να το δείξουμε αυτό, παρατηρήστε ότι, με λίγη προσοχή, η διαδικασία αυτή μπορεί να παραγάγει μία διαδρομή από το s έως το t εκτελώντας σταθερή ποσότητα εργασίας ανά ακμή του γραφήματος, και έτσι απαιτεί χρόνο $O(m)$. Αφού μπορεί να υπάρχουν το πολύ n-1 ασύνδετες ως προς τις ακμές διαδρομές από το s έως το t (η καθεμία τους πρέπει να χρησιμοποιεί διαφορετική εξερχόμενη ακμή από το s), απαιτείται χρόνος $O(mn)$ για την παραγωγή όλων των διαδρομών.

Συνοψίζοντας, έχουμε δείξει ότι

(7.44) *Ο αλγόριθμος Ford-Fulkerson μπορεί να χρησιμοποιηθεί για την εύρεση ενός μέγιστου συνόλου ασύνδετων ως προς τις ακμές διαδρομών s-t σε ένα κατευθυνόμενο γράφημα G, σε χρόνο O(mn).*

Μια εκδοχή του θεωρήματος Μέγιστης Ροής και Ελάχιστης Αποκοπής για ασύνδετες διαδρομές. Το θεώρημα Μέγιστης Ροής και Ελάχιστης Αποκοπής (7.13) μπορεί να χρησιμοποιηθεί για να μας δώσει τον ακόλουθο χαρακτηρισμό ως προς το μέγιστο πλήθος διαδρομών s-t που είναι ασύνδετες ως προς τις ακμές. Λέμε ότι ένα σύνολο ακμών $F \subseteq E$ *διαχωρίζει* το s από το t εάν, μετά την κατάργηση των ακμών F από το γράφημα G, δεν παραμένει στο γράφημα καμία διαδρομή s-t.

(7.45) *Σε όλα τα κατευθυνόμενα γραφήματα με κόμβους s και t, το μέγιστο πλήθος των διαδρομών s-t που είναι ασύνδετες ως προς τις ακμές είναι ίσο με το ελάχιστο πλήθος ακμών των οποίων η κατάργηση διαχωρίζει το s από το t.*

Απόδειξη. Αν η κατάργηση ενός συνόλου ακμών $F \subseteq E$ διαχωρίζει το s από το t, τότε η κάθε διαδρομή s-t θα πρέπει να χρησιμοποιεί τουλάχιστον μία ακμή από το F, και έτσι το πλήθος των ασύνδετων ως προς τις ακμές διαδρομών s-t είναι το πολύ $|F|$.

Για να αποδείξουμε το αντίστροφο θα χρησιμοποιήσουμε το θεώρημα Μέγιστης Ροής και Ελάχιστης Αποκοπής (7.13). Από την (7.43) γνωρίζουμε ότι το μέγιστο πλήθος διαδρομών που είναι ασύνδετες ως προς τις ακμές είναι η τιμή v της μέγιστης ροής s-t. Όμως το θεώρημα (7.13) λέει ότι υπάρχει μια αποκοπή s-t (A, B) με χωρητικότητα v. Έστω ότι F είναι το σύνολο των ακμών που πηγαίνουν από το A στο B. Η κάθε ακμή έχει χωρητικότητα 1, έτσι $|F| = v$ και, από τον ορισμό της αποκοπής s-t, η κατάργηση αυτών των v ακμών από το G διαχωρίζει το s από το t. ■

Αυτό το αποτέλεσμα μπορεί να θεωρηθεί μια φυσική ειδική περίπτωση του θεωρήματος Μέγιστης Ροής και Ελάχιστης Αποκοπής, όπου όλες οι χωρητικότητες ακμών είναι ίσες με 1. Στην πραγματικότητα αυτή η ειδική περίπτωση είχε αποδειχθεί από τον Menger το 1927, πολύ πριν από τη διατύπωση και απόδειξη του πλήρους θεωρήματος Μέγιστης Ροής και Ελάχιστης Αποκοπής· γι' αυτόν το λόγο αναφέρεται συχνά ως *θεώρημα του Menger*. Αν το καλοσκεφτούμε, η απόδειξη του θεωρήματος του Hall (7.40) για τα διμερή ταιριάσματα περιλαμβάνει επίσης μια αναγωγή σε γράφημα με ακμές μοναδιαίας χωρητικότητας, και έτσι μπορεί να αποδειχθεί με βάση το θεώρημα του Menger αντί με βάση το γενικό θεώρημα Μέγιστης Ροής και Ελάχιστης Αποκοπής. Με άλλα λόγια, το θεώρημα του Hall είναι στην πραγματικότητα μια ειδική περίπτωση του θεωρήματος του Menger, που με τη σειρά του είναι μια ειδική περίπτωση του θεωρήματος Μέγιστης Ροής και Ελάχιστης Αποκοπής. Και η ιστορία ακολουθεί αυτή τη διαδοχή, αφού ανακαλύφθηκαν με αυτή τη σειρά, σε απόσταση λίγων δεκαετιών[2].

Επεκτάσεις: Ασύνδετες διαδρομές σε μη κατευθυνόμενα γραφήματα

Τέλος, θα εξετάσουμε το πρόβλημα των ασύνδετων διαδρομών σε ένα μη κατευθυνόμενο γράφημα G. Παρά το ότι το γράφημά μας G είναι τώρα μη κατευθυνόμενο, μπορούμε να χρησιμοποιήσουμε τον αλγόριθμο μέγιστης ροής για να βρούμε στο G διαδρομές που είναι ασύνδετες ως προς τις ακμές. Η ιδέα είναι αρκετά απλή: Αντικαθιστούμε την κάθε μη κατευθυνόμενη ακμή (u, v) του G με δύο κατευθυνόμενες ακμές (u, v) και (v, u), και με αυτόν τον τρόπο δημιουργούμε μια κατευθυνόμενη εκδοχή G' του G. (Μπορούμε να διαγράψουμε τις ακμές που εισέρχονται στο s ή εξέρχονται από το t, αφού δεν μας είναι χρήσιμες.) Τώρα θέλουμε να χρησιμοποιήσουμε τον αλγόριθμο Ford-Fulkerson στο κατευθυνόμενο γράφημα που προέκυψε. Υπάρχει, όμως, ένα σημαντικό ζήτημα το οποίο θα πρέπει πρώτα να αντιμετωπίσουμε. Παρατηρήστε ότι δύο διαδρομές P_1 και P_2 μπορεί να είναι ασύνδετες ως προς τις ακμές στο κατευθυνόμενο γράφημα αλλά παρόλα αυτά να μοιράζονται μια κοινή ακμή στο μη κατευθυνόμενο γράφημα G: Αυτό συμβαίνει όταν η P_1 χρησιμοποιεί την κατευθυνόμενη ακμή (u, v) ενώ η P_2 χρησιμοποιεί την ακμή (v, u). Δεν είναι όμως δύσκολο να δείξουμε ότι σε οποιοδήποτε δίκτυο υπάρχει μια μέγιστη ροή που χρησιμοποιεί το πολύ *τη μία* από το ζευγάρι ακμών αντίθετης κατεύθυνσης.

(7.46) *Σε οποιοδήποτε δίκτυο ροής, υπάρχει μια μέγιστη ροή f όπου για όλες τις ακμές αντίθετης κατεύθυνσης $e = (u, v)$ και $e' = (v, u)$ έχουμε είτε $f(e) = 0$, ή $f(e') = 0$. Αν οι χωρητικότητες του δικτύου ροής είναι ακέραιες, τότε υπάρχει επίσης μια τέτοια ακέραια μέγιστη ροή.*

Απόδειξη. Θα πάρουμε μια μέγιστη ροή f και θα την τροποποιήσουμε έτσι ώστε να ικανοποιεί τη ζητούμενη συνθήκη. Υποθέστε ότι οι $e = (u, v)$ και $e' = (v, u)$ είναι αντίθετες κατευθυνόμενες ακμές και ότι $f(e) \neq 0$, $f(e') \neq 0$. Έστω ότι δ είναι η μικρότερη από αυτές τις τιμές· τροποποιούμε τώρα την f μειώνοντας την τιμή ροής και στην e και στην e' κατά δ. Η ροή f' που προκύπτει είναι εφικτή, έχει ίδια τιμή με την f, και η τιμή της σε μία από τις e και e' είναι 0. ■

Μπορούμε τώρα να χρησιμοποιήσουμε τον αλγόριθμο Ford-Fulkerson και τη διαδικασία διάσπασης ροής σε διαδρομές από την (7.42) για να πάρουμε ασύνδετες ως προς τις ακμές διαδρομές στο μη κατευθυνόμενο γράφημα G.

---

[2] Στην πραγματικότητα, σε μια ενδιαφέρουσα ιστορική ανασκόπηση που γράφτηκε το 1981, ο Menger αναφέρει την εκδοχή του για το πώς εξήγησε καταρχήν το θεώρημά του στον König, έναν από εκείνους που είχαν ανακαλύψει ανεξάρτητα το θεώρημα του Hall. Πιθανώς να πιστεύετε ότι ο König, ο οποίος είχε μελετήσει πολύ αυτά τα προβλήματα, θα αντιλαμβανόταν αμέσως γιατί η γενίκευση του Menger ως προς το θεώρημά του είναι αληθής, ή ότι ίσως να τη θεώρησε κιόλας προφανή. Όμως στην πραγματικότητα συνέβη το αντίθετο· ο König δεν πίστεψε ότι μπορεί να είναι σωστή, και έμεινε όλη τη νύχτα ξάγρυπνος αναζητώντας κάποιο αντιπαράδειγμα. Την επόμενη ημέρα, εξαντλημένος, αναζήτησε τον Menger και του ζήτησε την απόδειξη.

(7.47) *Υπάρχουν k ασύνδετες ως προς τις ακμές διαδρομές από το s έως το t σε ένα μη κατευθυνόμενο γράφημα G εάν και μόνο εάν η μέγιστη τιμή μιας ροής s-t στην κατευθυνόμενη εκδοχή G' του γραφήματος G είναι τουλάχιστον ίση με k. Επιπρόσθετα, ο αλγόριθμος Ford-Fulkerson μπορεί να χρησιμοποιηθεί για την εύρεση ενός μέγιστου συνόλου ασύνδετων διαδρομών s-t για το μη κατευθυνόμενο γράφημα G σε χρόνο* O(*mn*).

Το ανάλογο της (7.45) για τα μη κατευθυνόμενα γραφήματα είναι επίσης αληθές, αφού σε οποιαδήποτε αποκοπή *s-t* το πολύ μία από τις δύο ακμές αντίθετης κατεύθυνσης μπορεί να διασχίζει την αποκοπή από την πλευρά του *s* προς την πλευρά του *t* (επειδή αν η μία διασχίζει, τότε η άλλη πρέπει να πηγαίνει από την πλευρά του *t* στην πλευρά του *s*).

(7.48) *Σε όλα τα μη κατευθυνόμενα γραφήματα με κόμβους s και t, το μέγιστο πλήθος διαδρομών s-t που είναι ασύνδετες ως προς τις ακμές είναι ίσο με το ελάχιστο πλήθος ακμών των οποίων η κατάργηση διαχωρίζει το s από το t.*

7.7 Επεκτάσεις στο πρόβλημα Μέγιστης Ροής

Μεγάλο μέρος της ισχύος του προβλήματος Μέγιστης Ροής δεν έχει ουσιαστικά καμία σχέση με το ότι μοντελοποιεί την κίνηση σε ένα δίκτυο. Αντίθετα, πηγάζει από το γεγονός ότι πολλά προβλήματα που έχουν μια σύνθετη συνδυαστική συνιστώσα αναζήτησης μπορούν να λυθούν σε πολυωνυμικό χρόνο επειδή μπορούν να αναχθούν στο πρόβλημα της εύρεσης της μέγιστης ροής ή της ελάχιστης αποκοπής σε ένα κατευθυνόμενο γράφημα.

Το διμερές ταίριασμα είναι μια φυσική πρώτη εφαρμογή πάνω σε αυτό το πνεύμα· στις επόμενες ενότητες θα εξερευνήσουμε κάποιες ακόμα εφαρμογές. Για αρχή, θα παραμείνουμε με την εικόνα ότι η ροή είναι ένα αφηρημένο είδος "κίνησης", και θα ψάξουμε για πιο γενικές συνθήκες τις οποίες μπορούμε να επιβάλουμε σε αυτή την κίνηση. Αυτές οι γενικότερες συνθήκες θα αποδειχθούν χρήσιμες σε κάποιες από τις επόμενες εφαρμογές..

Ειδικότερα, θα εστιάσουμε σε δύο γενικεύσεις της μέγιστης ροής. Θα δούμε ότι και οι δύο μπορούν να αναχθούν στο βασικό πρόβλημα Μέγιστης Ροής.

Το πρόβλημα: Κυκλοφορίες με ζήτηση

Μια απλοποιητική παραδοχή στην αρχική μας διατύπωση του προβλήματος Μέγιστης Ροής είναι ότι είχαμε μόνο μία προέλευση *s* και μία απόληξη *t*. Υποθέστε τώρα ότι μπορεί να υπάρχει ένα σύνολο *S* από προελεύσεις που παράγουν ροή και ένα σύνολο *T* από απολήξεις που μπορούν να απορροφούν ροή. Όπως και προηγουμένως, σε κάθε ακμή υπάρχει μια ακέραια χωρητικότητα.

Όταν υπάρχουν πολλές προελεύσεις και απολήξεις, είναι κάπως ασαφής ο τρόπος με τον οποίο θα αποφασίσουμε ποια προέλευση ή απόληξη θα πρέπει να προτιμήσουμε σε ένα πρόβλημα μεγιστοποίησης. Έτσι, αντί να μεγιστοποιήσουμε την τιμή της ροής, θα εξετάσουμε το πρόβλημα όπου οι προελεύσεις έχουν σταθερές τιμές *παροχής* (supply) και οι απολήξεις έχουν σταθερές τιμές *ζήτησης* (demand), και ο στόχος μας είναι η δρομολόγηση ροής από τους κόμβους με τη διαθέσιμη παροχή σε εκείνους με τη δεδομένη ζήτηση. Φανταστείτε, για παράδειγμα, ότι το δίκτυο αναπαριστάνει ένα σύστημα οδικών αρτηριών ή σιδηροδρομικών γραμμών στις οποίες θέλουμε να δρομολογήσουμε προϊόντα από εργοστάσια (που δίνουν παροχή) σε καταστήματα μεταπώλησης (που έχουν ζήτηση). Σε αυτόν τον τύπο προβλήματος δεν ψάχνουμε να μεγιστοποιήσουμε κάποια συγκεκριμένη τιμή· αντίθετα, θέλουμε απλώς να ικανοποιήσουμε όλη τη ζήτηση χρησιμοποιώντας τη διαθέσιμη παροχή.

Μας δίνεται λοιπόν ένα δίκτυο ροής $G = (V, E)$ με χωρητικότητες στις ακμές. Τώρα με κάθε κόμβο $v \in V$ είναι συσχετισμένη μια *ζήτηση* d_v. Αν $d_v > 0$, αυτό σημαίνει ότι ο κόμβος v έχει μια *ζήτηση* d_v για ροή· ο κόμβος είναι μια απόληξη, και επιθυμεί να λάβει d_v περισσότερες μονάδες ροής από όσες θα στείλει. Αν $d_v < 0$, αυτό δείχνει ότι ο κόμβος v έχει μια *παροχή* $-d_v$· ο κόμβος είναι προέλευση, και επιθυμεί να στείλει $-d_v$ περισσότερες μονάδες ροής από όσες θα λάβει. Αν $d_v = 0$, τότε ο κόμβος v δεν είναι ούτε προέλευση ούτε απόληξη. Θα υποθέσουμε ότι όλες οι χωρητικότητες και η ζήτηση είναι ακέραιες.

Θα χρησιμοποιούμε το S για να συμβολίσουμε το σύνολο όλων των κόμβων με αρνητική ζήτηση και το T για να συμβολίσουμε το σύνολο όλων των κόμβων με θετική ζήτηση. Αν και οι κόμβοι v του συνόλου S θέλουν να στείλουν περισσότερη ροή από όση θα λάβουν, δεν υπάρχει πρόβλημα αν υπάρχει σε αυτούς και εισερχόμενη ροή· θα πρέπει απλώς να υπερκαλύπτεται από τη ροή που φεύγει από τον κόμβο v στις εξερχόμενες ακμές. Το ίδιο (κατά την αντίθετη κατεύθυνση) ισχύει και για το σύνολο T.

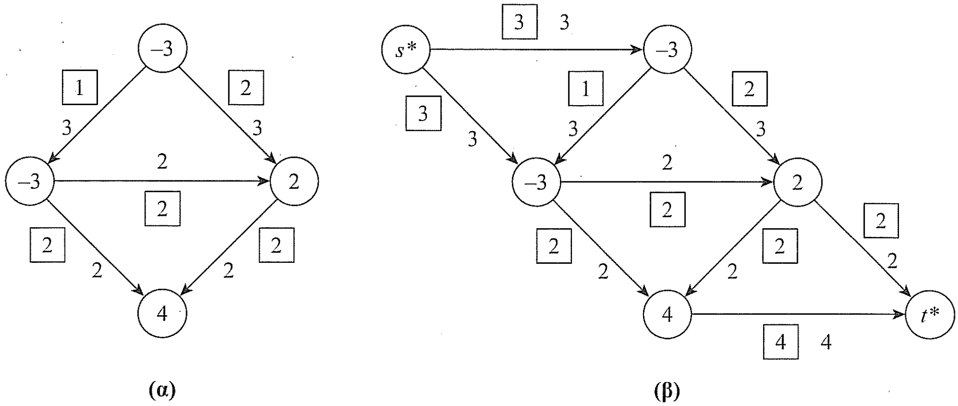
Σε αυτή τη διευθέτηση, λέμε ότι η *κυκλοφορία* με ζήτηση $\{d_v\}$ είναι μια συνάρτηση f που αποδίδει ένα μη αρνητικό πραγματικό αριθμό σε κάθε ακμή και ικανοποιεί τις ακόλουθες δύο συνθήκες.

(i) (*Συνθήκες χωρητικότητας*) Για κάθε $e \in E$, έχουμε $0 \le f(e) \le c_e$.

(ii) (*Συνθήκες ζήτησης*) Για κάθε $v \in V$, έχουμε $v, f^{in}(v) - f^{out}(v) = d_v$.

Στην περίπτωση αυτή, αντί να εξετάζουμε ένα πρόβλημα μεγιστοποίησης, μας απασχολεί ένα *πρόβλημα υλοποιησιμότητας:* Θέλουμε να γνωρίζουμε αν *υπάρχει* μια κυκλοφορία που να ικανοποιεί τις συνθήκες (i) και (ii).

Για παράδειγμα, εξετάστε το στιγμιότυπο της Εικόνας 7.13(α). Δύο από τους κόμβους είναι προελεύσεις, με ζήτηση –3 και –3· δύο άλλοι από τους κόμβους είναι απολήξεις, με ζήτηση 2 και 4. Οι τιμές ροής στην εικόνα συνιστούν μια εφικτή κυκλοφορία, και δείχνουν ότι μπορεί να ικανοποιηθεί όλη η ζήτηση χωρίς να παραβιαστούν οι χωρητικότητες.



Εικόνα 7.13 (α) Ένα παράδειγμα του προβλήματος Κυκλοφορίας μαζί με μια λύση του. Οι αριθμοί μέσα στους κόμβους είναι η ζήτηση· οι αριθμοί δίπλα στις ακμές είναι οι χωρητικότητες και οι τιμές ροής, με τις τιμές ροής μέσα σε πλαίσια. (β) Το αποτέλεσμα της αναγωγής αυτού του στιγμιοτύπου σε ένα ισοδύναμο στιγμιότυπο Μέγιστης Ροής.

Αν εξετάσουμε μια τυχαία περίπτωση του προβλήματος Κυκλοφορίας, η ακόλουθη είναι μια απλή συνθήκη που θα πρέπει να ικανοποιείται προκειμένου να υπάρχει κάποια εφικτή κυκλοφορία: η συνολική παροχή θα πρέπει να είναι ίση με τη συνολική ζήτηση.

(7.49) *Αν υπάρχει μια εφικτή κυκλοφορία με ζήτηση* $\{d_v\}$, *τότε* $\sum_v d_v = 0$.

Απόδειξη. Υποθέστε ότι υπάρχει μια εφικτή κυκλοφορία για αυτή την περίπτωση του προβλήματος. Τότε $\sum_v d_v = \sum_v f^{in}(v) - f^{out}(v)$. Όμως σε αυτή την παράσταση η τιμή $f(e)$ για την κάθε ακμή $e = (u, v)$ μετριέται ακριβώς δύο φορές: μία στην $f^{out}(u)$ και μία στην $f^{in}(v)$. Αυτοί οι δύο όροι αλληλοαναιρούνται, και αφού αυτό ισχύει για όλες τις τιμές $f(e)$ το συνολικό άθροισμα είναι 0. ■

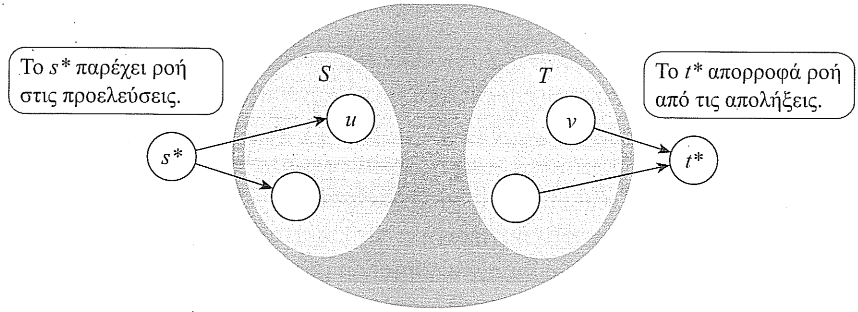
Χάρη στην (7.49) γνωρίζουμε ότι

$$\sum_{v:d_v>0} d_v = \sum_{v:d_v<0} -d_v .$$

Έστω ότι το D συμβολίζει αυτή την κοινή τιμή.

Σχεδιασμός και ανάλυση ενός αλγορίθμου για κυκλοφορίες

Αποδεικνύεται ότι μπορούμε να αναγάγουμε το πρόβλημα της εύρεσης μιας εφικτή κυκλοφορίας με ζήτηση $\{d_v\}$ στο πρόβλημα της εύρεσης μιας μέγιστης ροής s-t σε ένα διαφορετικό δίκτυο, όπως φαίνεται στην Εικόνα 7.14.



Εικόνα 7.14 Αναγωγή του προβλήματος Κυκλοφορίας σε πρόβλημα Μέγιστης Ροής.

Η αναγωγή μοιάζει πολύ με εκείνη που χρησιμοποιήσαμε για το πρόβλημα του Διμερούς Ταιριάσματος: προσαρτούμε μια "υπερπροέλευση" s^* σε κάθε κόμβο του S και μια "υπεραπόληξη" t^* σε κάθε κόμβο του T. Πιο συγκεκριμένα, δημιουργούμε ένα γράφημα G' με βάση το G με προσθήκη των νέων κόμβων s^* και t^* στο G. Για κάθε κόμβο $v \in T$ — δηλαδή, για κάθε κόμβο v με $d_v > 0$ — προσθέτουμε μια ακμή (v, t^*) με χωρητικότητα d_v. Για κάθε κόμβο $u \in S$ — δηλαδή, για κάθε κόμβο με $d_u < 0$ — προσθέτουμε μια ακμή (s^*, u) με χωρητικότητα $-d_u$. Μεταφέρουμε χωρίς μεταβολή την υπόλοιπη δομή του G στο G'.

Σε αυτό το γράφημα G' θα αναζητήσουμε μια μέγιστη ροή s^*-t^*. Διαισθητικά, μπορούμε να σκεφτούμε αυτή την αναγωγή ως προσθήκη ενός κόμβου s^* ο οποίος "προμηθεύει" τις προελεύσεις με την πρόσθετη ροή τους και ενός κόμβου t^* ο οποίος "απορροφά" την πρόσθετη ροή από τις απολήξεις. Για παράδειγμα, η Εικόνα 7.13(β) δείχνει το αποτέλεσμα της εφαρμογής αυτής της αναγωγής στο στιγμιότυπο της Εικόνας 7.13(α).

Σημειώστε ότι δεν μπορεί να υπάρχει στο G' μια ροή s^*-t^* με τιμή μεγαλύτερη από D, αφού η αποκοπή (A, B) με $A = \{s^*\}$ έχει μόνο χωρητικότητα D. Στο σημείο αυτό, αν υπάρχει στο G μια εφικτή κυκλοφορία f με ζήτηση $\{d_v\}$, τότε με την αποστολή μιας τιμής ροής $-d_v$ σε κάθε ακμή (s^*, v) και μιας τιμής ροής d_v σε κάθε ακμή (v, t^*) παίρνουμε στο G' μια ροή s^*-t^* με τιμή D, και έτσι αυτή είναι μια μέγιστη ροή. Αντιστρόφως, υποθέστε ότι υπάρχει στο G' μια (μέγιστη) ροή s^*-t^* με τιμή D. Αυτό σημαίνει ότι όλες οι ακμές που εξέρχονται από το s^* και όλες οι ακμές που εισέρχονται στο t^* είναι πλήρως κορεσμένες με ροή. Έτσι, αν διαγράψουμε αυτές τις ακμές, παίρνουμε στο G μια κυκλοφορία f με $f^{in}(v) - f^{out}(v) = d_v$ για κάθε κόμβο v. Επιπρόσθετα, αν υπάρχει μια ροή με τιμή D στο G' τότε υπάρχει μια τέτοια ροή που να έχει ακέραιες τιμές.

Συνοψίζοντας, έχουμε αποδείξει το εξής.

(7.50) *Υπάρχει στο G μια εφικτή κυκλοφορία με ζήτηση $\{d_v\}$ εάν και μόνο εάν η μέγιστη ροή s^*-t^* στο G' έχει τιμή D. Αν όλες οι χωρητικότητες και η ζήτηση στο G είναι ακέραιες και υπάρχει κάποια εφικτή κυκλοφορία, τότε υπάρχει εφικτή κυκλοφορία με ακέραιες τιμές.*

Στο τέλος της Ενότητας 7.5 χρησιμοποιήσαμε το θεώρημα Μέγιστης Ροής και Ελάχιστης Αποκοπής για να εξαγάγουμε το χαρακτηρισμό (7.40) για τα διμερή γραφήματα που δεν έχουν τέλειο ταίριασμα. Μπορούμε να δώσουμε έναν ανάλογο χαρακτηρισμό για τα γραφήματα που δεν έχουν εφικτή κυκλοφορία. Ο χαρακτηρισμός αυτός χρησιμοποιεί την έννοια της *αποκοπής* (cut), προσαρμοσμένη στη συγκεκριμένη περίπτωση. Στα πλαίσια των προβλημάτων κυκλοφορίας με ζήτηση, η αποκοπή (A, B) είναι οποιαδήποτε διαμέριση του συνόλου κόμβων V σε δύο σύνολα, χωρίς κανέναν περιορισμό ως προς την πλευρά όπου θα τοποθετηθούν οι προελεύσεις και οι απολήξεις. Δίνουμε εδώ το χαρακτηρισμό χωρίς απόδειξη.

(7.51) *Το γράφημα G διαθέτει εφικτή κυκλοφορία με ζήτηση $\{d_v\}$ εάν και μόνο εάν για όλες τις αποκοπές (A, B) ισχύει*

$$\sum_{v \in B} d_v \le c(A,B).$$

Είναι σημαντικό να επισημάνουμε ότι το δίκτυό μας έχει μόνο ένα "είδος" ροής. Αν και η ροή παρέχεται από πολλές προελεύσεις και απορροφάται από πολλές απολήξεις, δεν μπορούμε να θέσουμε περιορισμούς ως προς το ποια προέλευση θα παρέχει τη ροή σε κάθε απόληξη· θα πρέπει να αφήσουμε τον αλγόριθμό μας να το αποφασίσει αυτό. Ένα δυσκολότερο πρόβλημα είναι το *πρόβλημα Ροής Πολλών Αγαθών*· σε αυτό για κάθε i η απόληξη t_i θα πρέπει να προμηθεύεται ροή που προέρχεται από την προέλευση s_i. Θα αναλύσουμε περισσότερο αυτό το ζήτημα στο Κεφάλαιο 11.

Το πρόβλημα: Κυκλοφορίες με ζήτηση και κάτω όρια

Τέλος, ας γενικεύσουμε λίγο το προηγούμενο πρόβλημα. Σε πολλές εφαρμογές δεν θέλουμε μόνο να ικανοποιούμε ζήτηση σε διάφορους κόμβους· θέλουμε επίσης να εξαναγκάζουμε τη ροή να χρησιμοποιεί συγκεκριμένες ακμές. Αυτό μπορεί να επιβληθεί με την επιβολή *κάτω ορίων* για τις ακμές, μαζί με τα συνηθισμένα άνω όρια που επιβάλλονται από τις χωρητικότητες των ακμών.

Θεωρήστε ένα δίκτυο ροής $G = (V, E)$ με μια *χωρητικότητα* c_e και ένα *κάτω όριο* ℓ_e σε κάθε ακμή e. Θα υποθέσουμε ότι $0 \le \ell_e \le c_e$ για κάθε e. Όπως και προηγουμένως, ο κάθε κόμβος v θα έχει επίσης μια ζήτηση d_v, η οποία μπορεί να είναι είτε θετική είτε αρνητική. Θα θεωρήσουμε ότι όλη η ζήτηση, οι χωρητικότητες, και τα κάτω όρια είναι ακέραιοι αριθμοί.

Οι παραπάνω ποσότητες έχουν το ίδιο νόημα όπως και προηγουμένως, και εδώ το κάτω όριο ℓ_e σημαίνει ότι η τιμή της ροής στην ακμή e θα πρέπει να είναι *τουλάχιστον* ℓ_e. Έτσι η κυκλοφορία στο δίκτυο ροής μας θα πρέπει να ικανοποιεί τις ακόλουθες δύο συνθήκες.

(i) (*Συνθήκες χωρητικότητας*) Για κάθε $e \in E$, έχουμε $\ell_e \le f(e) \le c_e$.

(ii) (*Συνθήκες ζήτησης*) Για κάθε $v \in V$, έχουμε $f^{in}(v) - f^{out}(v) = d_v$.

Όπως και προηγουμένως, θέλουμε να αποφασίσουμε αν υπάρχει κάποια εφικτή *κυκλοφορία* — δηλαδή κυκλοφορία που ικανοποιεί τις συνθήκες.

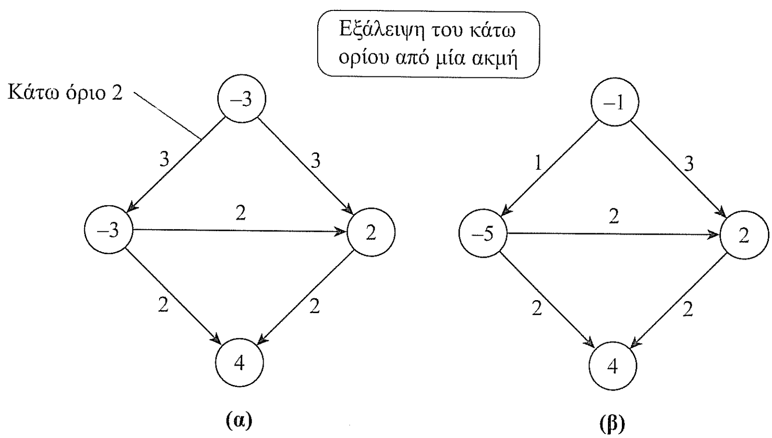
Σχεδιασμός και ανάλυση ενός αλγορίθμου με κάτω όρια

Η στρατηγική μας θα είναι να το αναγάγουμε αυτό σε ένα πρόβλημα εύρεσης κυκλοφορίας με ζήτηση αλλά χωρίς κάτω όρια. (Είδαμε ήδη ότι αυτό το τελευταίο πρόβλημα μπορεί με τη σειρά του να αναχθεί στο τυπικό πρόβλημα Μέγιστης Ροής.) Η ιδέα είναι η ακόλουθη. Γνωρίζουμε ότι σε κάθε ακμή e θα πρέπει να στείλουμε τουλάχιστον ℓ_e μονάδες ροής. Υποθέστε λοιπόν ότι ορίζουμε μια αρχική κυκλοφορία f_0 όπου απλώς $f_0(e) = \ell_e$. Η f_0 ικανοποιεί όλες τις συνθήκες χωρητικότητας (και τα κάτω και τα άνω όρια)· όμως ενδεχομένως να μην ικανοποιεί όλες τις συνθήκες ζήτησης. Ειδικότερα

$$f_0^{in}(v) - f_0^{out}(v) = \sum_{e \text{ προς το } v} \ell_e - \sum_{e \text{ από το } v} \ell_e$$

Ας συμβολίσουμε αυτή την ποσότητα με το L_v. Αν $L_v = d_v$ έχουμε ικανοποιήσει τη συνθήκη ζήτησης για τον κόμβο v· αν όμως δεν συμβαίνει αυτό, τότε θα πρέπει να υπερθέσουμε μια κυκλοφορία f_1 "πάνω" από την f_0 η οποία θα εξαλείψει την εναπομένουσα "ανισορροπία" στον κόμβο v. Έτσι χρειαζόμαστε $f_1^{in}(v) - f_1^{out}(v) = d_v - L_v$. Και πόση χωρητικότητα διαθέτουμε, με την οποία θα το επιτύχουμε αυτό; Αφού έχουμε ήδη στείλει ℓ_e μονάδες ροής σε κάθε ακμή e, μας μένουν $c_e - \ell_e$ με τις οποίες μπορούμε να δουλέψουμε.

Αυτοί οι συλλογισμοί οδηγούν άμεσα στην ακόλουθη κατασκευή. Έστω ότι το γράφημα G' έχει τους ίδιους κόμβους και ακμές, με χωρητικότητες και ζήτηση αλλά χωρίς κάτω όρια. Η χωρητικότητα της ακμής e θα είναι $c_e - \ell_e$. Η ζήτηση του κόμβου v θα είναι $d_v - L_v$.



Εικόνα 7.15 (α) Ένα στιγμιότυπο προβλήματος Κυκλοφορίας με κάτω όρια: οι αριθμοί μέσα στους κόμβους είναι η ζήτηση, και οι αριθμοί στις ακμές είναι οι χωρητικότητες. Έχουμε επίσης θέσει ένα κάτω όριο 2 σε μία από τις ακμές. (β) Το αποτέλεσμα της αναγωγής αυτού του στιγμιότυπου σε ένα ισοδύναμο στιγμιότυπο του προβλήματος Κυκλοφορίας χωρίς κάτω όρια.

Για παράδειγμα, εξετάστε το πρόβλημα που φαίνεται στην Εικόνα 7.15(α). Αυτό το πρόβλημα είναι ίδιο με εκείνο που είδαμε στην Εικόνα 7.13, με τη διαφορά ότι έχουμε τώρα καθορίσει για μία από τις ακμές κάτω όριο 2. Στην Εικόνα 7.15(β) έχουμε εξαλείψει αυτό το κάτω όριο στέλνοντας δύο μονάδες ροής μέσω της ακμής. Αυτό ελαττώνει το άνω όριο της ακμής και τροποποιεί επίσης τη ζήτηση στα δύο άκρα της ακμής. Κατά την πορεία γίνεται φανερό ότι δεν υπάρχει δυνατή κυκλοφορία, επειδή μετά από την εφαρμογή αυτής της κατασκευής υπάρχει ένας κόμβος με ζήτηση –5 αλλά με μόνο τέσσερις μονάδες χωρητικότητας στις εξερχόμενες ακμές του.

Ισχυριζόμαστε τώρα ότι αυτή η γενική διαδικασία κατασκευής παράγει ένα ισοδύναμο πρόβλημα με ζήτηση αλλά χωρίς κάτω όρια· έτσι μπορούμε να χρησιμοποιήσουμε τον αλγόριθμό μας και για αυτό το πρόβλημα.

(7.52) *Υπάρχει στο G μια εφικτή κυκλοφορία εάν και μόνο εάν υπάρχει μια εφικτή κυκλοφορία στο G'. Αν στο G όλη η ζήτηση, οι χωρητικότητες, και τα κάτω όρια είναι ακέραιοι αριθμοί και υπάρχει εφικτή κυκλοφορία, τότε υπάρχει μια εφικτή κυκλοφορία με ακέραιες τιμές.*

Απόδειξη. Υποθέστε καταρχήν ότι υπάρχει μια κυκλοφορία f' στο G'. Ορίζουμε στο G μια κυκλοφορία f με $f(e) = f'(e) + \ell_e$. Έτσι η f ικανοποιεί τις συνθήκες χωρητικότητας στο G, και

$$f^{in}(v) - f^{out}(v) = \sum_{e \text{ προς το } v} (\ell_e + f'(e)) - \sum_{e \text{ από το } v} (\ell_e + f'(e)) = L_v + (d_v - L_v) = d_v,$$

άρα ικανοποιεί επίσης και τις συνθήκες ζήτησης στο G.

Αντιστρόφως, υποθέστε ότι υπάρχει στο G μια κυκλοφορία f· ορίζουμε τότε στο G' μια κυκλοφορία f' με $f'(e) = f(e) - \ell_e$. Η f' ικανοποιεί τις συνθήκες χωρητικότητας στο G', και

$$(f')^{in}(v) - (f')^{out}(v) = \sum_{e \text{ προς το } v} (f(e) - l_e) - \sum_{e \text{ από το } v} (f(e) - l_e) = d_v - L_v$$

άρα ικανοποιεί επίσης και τις συνθήκες ζήτησης στο G'. ■

7.8 Σχεδιασμός διεξαγωγής έρευνας

Πολλά προβλήματα που παρουσιάζονται στις εφαρμογές μπορούν στην πραγματικότητα να λυθούν αποδοτικά με αναγωγή σε προβλήματα Μέγιστης Ροής· όμως είναι συχνά δύσκολο να ανακαλύψει κανείς πότε είναι εφικτή μια τέτοια αναγωγή. Στις επόμενες ενότητες θα δώσουμε αρκετά ενδεικτικά παραδείγματα τέτοιων προβλημάτων. Ο στόχος είναι να δούμε με τι μοιάζουν αυτές οι αναγωγές και να καταδείξουμε κάποιες συνηθισμένες χρήσεις των ροών και των αποκοπών στο σχεδιασμό αποδοτικών συνδυαστικών αλγορίθμων. Ένα βασικό σημείο που θα δούμε να ανακύπτει είναι το ακόλουθο: Κάποιες φορές η λύση που θέλουμε περιλαμβάνει τον υπολογισμό μιας μέγιστης ροής, ενώ κάποιες άλλες φορές περιλαμβάνει τον υπολογισμό μιας ελάχιστης αποκοπής· και οι ροές και οι αποκοπές είναι πολύ χρήσιμα αλγοριθμικά εργαλεία.

Θα ξεκινήσουμε με μια πολύ βασική εφαρμογή που θα ονομάσουμε *σχεδιασμό διεξαγωγής έρευνας*, η οποία είναι μια απλοποιημένη εκδοχή μιας εργασίας που αντιμετωπίζουν πολλές εταιρείες οι οποίες θέλουν να μετρήσουν την ικανοποίηση των πελατών τους. Γενικότερα, το πρόβλημα αυτό δείχνει με ποιον τρόπο η κατασκευή που χρησιμοποιήσαμε για την επίλυση του προβλήματος Διμερούς Ταιριάσματος ανακύπτει με φυσικό τρόπο σε οποιαδήποτε περίπτωση όπου θέλουμε να εξισορροπήσουμε προσεκτικά αποφάσεις ως προς ένα σύνολο επιλογών — στην περίπτωση αυτή, το σχεδιασμό ερωτηματολογίων με εξισορρόπηση των συναφών ερωτήσεων ως προς τον πληθυσμό των καταναλωτών.

Το πρόβλημα

Ένα σημαντικό ζήτημα στον ταχύτατα αναπτυσσόμενο τομέα της *εξόρυξης δεδομένων* (data mining) είναι η μελέτη των μοτίβων προτιμήσεων των πελατών. Φανταστείτε μια εταιρεία που πουλά k προϊόντα και διαθέτει μια βάση δεδομένων η οποία περιέχει το ιστορικό αγορών για ένα μεγάλο αριθμό πελατών. (Όσοι από εσάς διαθέτετε "Κάρτες Μέλους" πιθανώς να μπορείτε να μαντέψετε πώς συλλέγονται αυτά τα δεδομένα.) Η εταιρεία θέλει να διεξαγάγει μια έρευνα στέλνοντας προσαρμοσμένα ερωτηματολόγια σε μια συγκεκριμένη ομάδα n πελατών της, έτσι ώστε να προσπαθήσει να διαπιστώσει με ποια προϊόντα είναι συνολικά ευχαριστημένοι οι πελάτες.

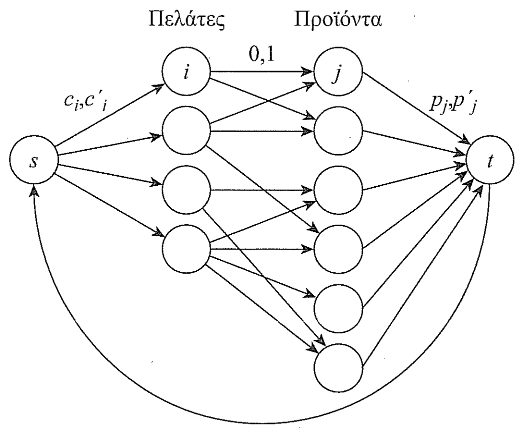
Οι γενικές κατευθυντήριες γραμμές για το σχεδιασμό του ερωτηματολογίου είναι οι ακόλουθες.

- Ο κάθε πελάτης θα λάβει ερωτήσεις για ένα συγκεκριμένο υποσύνολο προϊόντων.
- Οι πελάτες μπορούν να ερωτηθούν μόνο σχετικά με προϊόντα που έχουν αγοράσει.
- Για να είναι πληροφοριακό το ερωτηματολόγιο, αλλά και να μην είναι τόσο μεγάλο που να αποθαρρύνει τη συμμετοχή, ο κάθε πελάτης i θα πρέπει να ερωτάται για έναν αριθμό προϊόντων μεταξύ c_i and c'_i.
- Τέλος, για να συλλεγούν επαρκή δεδομένα για το καθένα από τα προϊόντα, θα πρέπει να ερωτηθούν για το κάθε προϊόν j από p_j έως p'_j διαφορετικοί πελάτες.

Για να το διατυπώσουμε πιο τυπικά, η είσοδος για το πρόβλημα του *Σχεδιασμού Διεξαγωγής Έρευνας* αποτελείται από ένα διμερές γράφημα G του οποίου οι κόμβοι είναι οι πελάτες και τα προϊόντα, και υπάρχει ακμή μεταξύ του πελάτη i και του προϊόντος j αν ο πελάτης έχει κάποτε αγοράσει το προϊόν j. Επιπρόσθετα, για κάθε πελάτη $i = 1, ..., n$ έχουμε όρια $c_i \leq c'_i$ ως προς το πλήθος των προϊόντων για τα οποία θα μπορεί να ερωτηθεί ο πελάτης· για κάθε προϊόν $j = 1, ..., k$ έχουμε όρια $p_j \leq p'_j$ ως προς το πλήθος διαφορετικών πελατών που θα πρέπει να ρωτήσουμε. Το πρόβλημα είναι να αποφασίσουμε αν υπάρχει τρόπος σχεδιασμού ενός ερωτηματολογίου για κάθε πελάτη έτσι ώστε να ικανοποιούνται όλες αυτές οι προϋποθέσεις.

Σχεδιασμός του αλγορίθμου

Θα λύσουμε αυτό το πρόβλημα ανάγοντάς το σε ένα πρόβλημα κυκλοφορίας για ένα δίκτυο ροής G' με ζήτηση και κάτω όρια, όπως φαίνεται στην Εικόνα 7.16. Για να κατασκευάσουμε το γράφημα G' από το G, προσανατολίζουμε τις ακμές του G από τους πελάτες προς τα προϊόντα, προσθέτουμε κόμβους s και t με ακμές (s, i) για κάθε πελάτη $i = 1, ..., n$, ακμές (j, t) για κάθε προϊόν $j = 1, ..., k$, και μια ακμή (t, s). Η κυκλοφορία σε αυτό το δίκτυο θα αντιστοιχεί στον τρόπο με τον οποίο γίνονται οι ερωτήσεις. Η ροή στην ακμή (s, i) είναι το πλήθος των προϊόντων που συμπεριλαμβάνονται στο ερωτηματολόγιο για τον πελάτη i, έτσι αυτή η ακμή θα έχει χωρητικότητα c_i' και κάτω όριο c_i. Η ροή στην ακμή (j, t) θα αντιστοιχεί στο πλήθος των πελατών που ερωτώνται σχετικά με το προϊόν j, έτσι αυτή η ακμή θα έχει χωρητικότητα p'_j και κάτω όριο p_j. Η κάθε ακμή (i, j) που συνδέει έναν πελάτη με ένα προϊόν το οποίο έχει αγοράσει θα έχει χωρητικότητα 1 και κάτω όριο 0. Η ροή που μεταφέρεται από την ακμή (t, s) αντιστοιχεί στο συνολικό πλήθος ερωτήσεων που πραγματοποιήθηκαν. Μπορούμε να αποδώσουμε σε αυτή την ακμή χωρητικότητα $\sum_i c'_i$ και κάτω όριο $\sum_i c_i$. Όλοι οι κόμβοι έχουν ζήτηση ίση με 0.



Εικόνα 7.16 Το πρόβλημα του Σχεδιασμού Διεξαγωγής Έρευνας μπορεί να αναχθεί στο πρόβλημα της εύρεσης μιας εφικτής κυκλοφορίας: Η ροή μεταφέρεται από τους πελάτες (με όρια χωρητικότητας τα οποία δείχνουν πόσες ερωτήσεις μπορούν να πραγματοποιηθούν) στα προϊόντα (με όρια χωρητικότητας τα οποία δείχνουν πόσες ερωτήσεις μπορούν να γίνουν για κάθε προϊόν).

Ο αλγόριθμός μας είναι απλώς η κατασκευή του δικτύου G' και ο έλεγχος για την ύπαρξη εφικτής κυκλοφορίας. Μπορούμε να διατυπώσουμε τώρα έναν ισχυρισμό ο οποίος δείχνει την ορθότητα αυτού του αλγορίθμου.

Ανάλυση του αλγορίθμου

(7.53) *Το γράφημα G' που μόλις κατασκευάσαμε διαθέτει εφικτή κυκλοφορία εάν και μόνο εάν υπάρχει ένας εφικτός τρόπος για το σχεδιασμό της έρευνας.*

Απόδειξη. Η παραπάνω κατασκευή μάς δείχνει άμεσα έναν τρόπο για τη μετατροπή του σχεδιασμού διεξαγωγής της έρευνας σε μια αντίστοιχη ροή. Η ακμή (i, j) θα μεταφέρει μία μονάδα ροής αν ο πελάτης i ερωτάται σχετικά με το προϊόν j στην έρευνα, ενώ διαφορετικά δεν θα μεταφέρει ροή. Η ροή στις ακμές (s, i) είναι το πλήθος ερωτήσεων που έγιναν στον πελάτη i, η ροή στην ακμή (j, t) είναι το πλήθος πελατών που ερωτήθηκαν σχετικά με το προϊόν j, και τέλος η ροή στην ακμή (t, s) είναι το συνολικό πλήθος ερωτήσεων που πραγματοποιήθηκαν. Αυτή η ροή ικανοποιεί τη μηδενική ζήτηση, δηλαδή ισχύει η διατήρηση ροής σε όλους τους κόμβους. Αν η έρευνα ικανοποιεί αυτούς τους κανόνες, τότε η αντίστοιχη ροή ικανοποιεί τις χωρητικότητες και τα κάτω όρια.

Αντιστρόφως, αν το πρόβλημα Κυκλοφορίας είναι εφικτό, τότε με βάση την (7.52) υπάρχει μια εφικτή κυκλοφορία με ακέραιες τιμές, και μια τέτοια κυκλοφορία με ακέραιες τιμές αντιστοιχεί φυσικά σε έναν εφικτό σχεδιασμό διεξαγωγής έρευνας. Ο πελάτης i θα ερωτηθεί για το προϊόν j εάν και μόνο εάν η ακμή (i, j) μεταφέρει μία μονάδα ροής. ■

7.9 Χρονοπρογραμματισμός αεροπορικών δρομολογίων

Τα υπολογιστικά προβλήματα τα οποία αντιμετωπίζουν οι μεγάλες αεροπορικές εταιρείες στις ΗΠΑ είναι τόσο πολύπλοκα που σχεδόν δεν χωρούν στη φαντασία. Θα πρέπει καθημερινά να καταστρώνουν προγράμματα για χιλιάδες δρομολόγια τα οποία θα είναι αποδοτικά ως προς τη χρήση του εξοπλισμού, την ανάθεση πληρωμάτων, την ικανοποίηση των πελατών, και μυριάδες άλλους παράγοντες — και όλα αυτά ενόψει απρόβλεπτων παραγόντων όπως ο καιρός και οι βλάβες. Δεν προκαλεί λοιπόν έκπληξη το γεγονός ότι οι εταιρείες αυτές είναι από τους μεγαλύτερους χρήστες αλγοριθμικών τεχνικών μεγάλης ισχύος.

Η κάλυψη αυτών των υπολογιστικών προβλημάτων σε οποιοδήποτε ρεαλιστικό επίπεδο λεπτομέρειας θα μας οδηγούσε υπερβολικά μακριά. Αντί γι' αυτό, θα εξετάσουμε ένα "απλοϊκό" πρόβλημα το οποίο καλύπτει, με πολύ ξεκάθαρο τρόπο, μερικά από τα ζητήματα κατανομής πόρων που προκύπτουν σε ένα τέτοιο περιβάλλον. Επίσης, όπως συμβαίνει συχνά σε αυτό το βιβλίο, αυτό το "απλοϊκό" πρόβλημα θα μας φανεί πολύ χρησιμότερο για τους σκοπούς μας σε σχέση με το "πραγματικό" πρόβλημα, επειδή η λύση του απλοϊκού προβλήματος περιλαμβάνει μια πολύ γενική τεχνική που μπορεί να εφαρμοστεί σε μεγάλο εύρος περιστάσεων.

Το πρόβλημα

Υποθέστε ότι είστε υπεύθυνοι για τη διαχείριση ενός στόλου αεροσκαφών και θέλετε να κατασκευάσετε ένα χρονοδιάγραμμα πτήσεων για αυτά. Το ακόλουθο είναι ένα πολύ απλό μοντέλο για την περίπτωση αυτή. Η έρευνα αγοράς έχει προσδιορίσει ένα σύνολο από m συγκεκριμένα τμήματα πτήσεων που θα ήταν πολύ προσοδοφόρα αν μπορούσατε να τα εξυπηρετήσετε· το κάθε τμήμα πτήσης j προσδιορίζεται από τέσσερις παραμέτρους: το αεροδρόμιο προέλευσης, το αεροδρόμιο προορισμού, το χρόνο αναχώρησης, και το χρόνο άφιξης. Η Εικόνα 7.17(α) δείχνει ένα απλό παράδειγμα που αποτελείται από έξι τμήματα πτήσεων τα οποία θα θέλατε να εξυπηρετήσετε με τα αεροπλάνα σας μέσα σε μία ημέρα:

(1) BOS (Βοστόνη αναχώρηση 6 πμ) – DCA (Ουάσιγκτον DC άφιξη 7 πμ)

(2) PHL (Φιλαδέλφεια αναχώρηση 7 πμ) – PIT (Πίτσμπουργκ άφιξη 8 πμ)

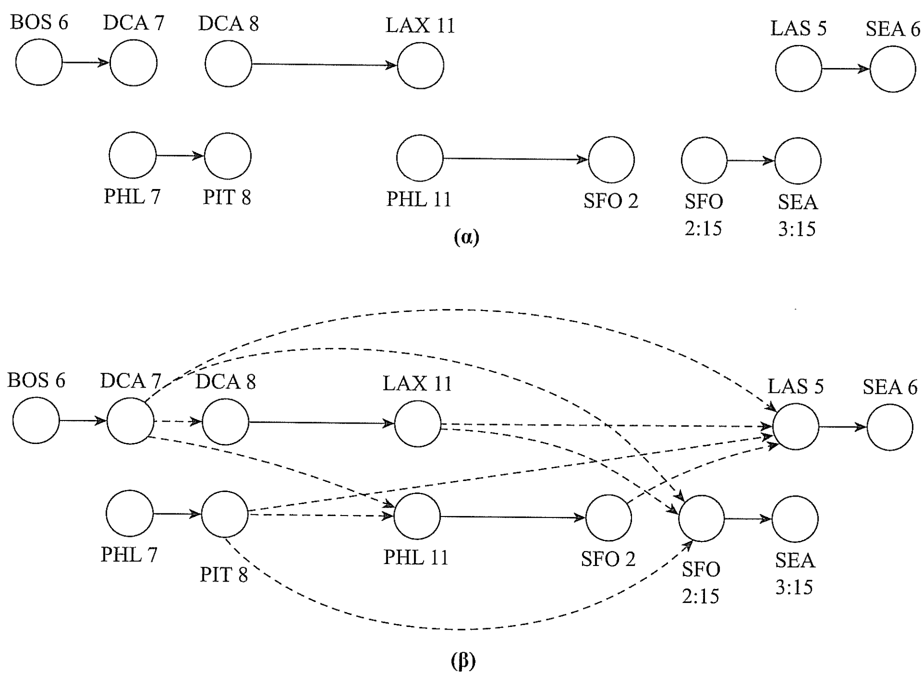
(3) DCA (Ουάσιγκτον DC αναχώρηση 8 πμ) – LAX (Λος Άντζελες άφιξη 11 πμ)

(4) PHL (Φιλαδέλφεια αναχώρηση 11 πμ) – SFO (Σαν Φρανσίσκο άφιξη 2 μμ)

(5) SFO (Σαν Φρανσίσκο αναχώρηση 2:15 μμ) – SEA (Σηάτλ άφιξη 3:15 μμ)

(6) LAS (Λας Βέγκας αναχώρηση 5 μμ) – SEA (Σηάτλ άφιξη 6 μμ)

Παρατηρήστε ότι το κάθε τμήμα πτήσης αναφέρει, εκτός από τα αεροδρόμια, και τις ώρες στις οποίες θέλετε να γίνει η πτήση.



Εικόνα 7.17 (α) Ένα μικρό στιγμιότυπο του απλού προβλήματος Χρονοπρογραμματισμού Αεροπορικών Δρομολογίων. (β) Ένα εκτεταμένο γράφημα το οποίο δείχνει ποιες πτήσεις είναι προσπελάσιμες από άλλες πτήσεις.

Μπορούμε να χρησιμοποιήσουμε το ίδιο αεροπλάνο για το τμήμα πτήσης i και στη συνέχεια για το τμήμα πτήσης j εάν

(α) ο προορισμός της πτήσης i είναι ίδιος με την προέλευση της πτήσης j, και υπάρχει επαρκής χρόνος μεταξύ των δύο πτήσεων για την πραγματοποίηση συντήρησης στο αεροπλάνο· ή

(β) μπορούμε να προσθέσουμε μεταξύ των δύο πτήσεων μια άλλη πτήση που να μεταφέρει το αεροπλάνο από τον προορισμό της πτήσης j στην προέλευση της πτήσης i με επαρκή χρόνο μεταξύ των πτήσεων.

Για παράδειγμα, αν υποθέσουμε ότι απαιτείται μία ώρα ως ενδιάμεσος χρόνος συντήρησης, θα μπορούσαμε να χρησιμοποιήσουμε το ίδιο αεροπλάνο για τις πτήσεις (1), (3), και (6) αφήνοντας το αεροπλάνο στην Ουάσιγκτον μεταξύ των πτήσεων (1) και (3) και μετά εισάγοντας την πτήση

LAX (Λος Άντζελες αναχώρηση 12 πμ) - LAS (Λας Βέγκας άφιξη 1 μμ)

μεταξύ των πτήσεων (3) και (6).

Διατύπωση του προβλήματος Μπορούμε να μοντελοποιήσουμε πολύ γενικά αυτή την κατάσταση, ανεξάρτητα από συγκεκριμένους κανόνες σχετικά με τους χρόνους συντήρησης και τα ενδιάμεσα τμήματα πτήσεων, με τον εξής τρόπο: Θα πούμε απλώς ότι η πτήση j είναι *προσπελάσιμη* από την πτήση i εάν είναι δυνατό να χρησιμοποιήσουμε το ίδιο αεροπλάνο για την πτήση i και στη συνέχεια και για την πτήση j. Έτσι, σύμφωνα με τους συγκεκριμένους κανόνες (α) και (β) που είδαμε παραπάνω, μπορούμε εύκολα να προσδιορίσουμε για κάθε ζευγάρι i, j εάν η πτήση j είναι προσπελάσιμη από την πτήση i. (Προφανώς, θα μπορούσε κανείς εύκολα να φανταστεί πιο πολύπλοκους κανόνες σχετικά με την προσπελασιμότητα. Για παράδειγμα, η διάρκεια του χρόνου συντήρησης που απαιτείται στο (α) μπορεί να εξαρτάται από το αεροδρόμιο· ή στο (β) μπορεί να απαιτείται ότι το τμήμα πτήσης που θα εισαχθεί θα είναι από μόνο του επαρκώς προσοδοφόρο.) Όμως το θέμα είναι ότι με τον ορισμό μας μπορούμε να χειριστούμε οποιοδήποτε σύνολο κανόνων: η είσοδος του προβλήματος δεν θα περιλαμβάνει μόνο τα τμήματα πτήσεων, αλλά και έναν προσδιορισμό των ζευγαριών (i, j) για τα οποία η μεταγενέστερη πτήση j είναι προσπελάσιμη από μια προγενέστερη πτήση i. Αυτά τα ζευγάρια μπορούν να σχηματίζουν ένα αυθαίρετο κατευθυνόμενο ακυκλικό γράφημα.

Ο στόχος σε αυτό το πρόβλημα είναι να προσδιορίσουμε αν είναι δυνατό να εξυπηρετήσουμε και τις m πτήσεις της αρχικής λίστας μας χρησιμοποιώντας το πολύ k αεροπλάνα συνολικά. Για να μπορέσουμε να το επιτύχουμε αυτό, θα πρέπει να βρούμε έναν τρόπο ώστε να επαναχρησιμοποιούμε με αποδοτικό τρόπο τα αεροπλάνα για πολλαπλές πτήσεις.

Για παράδειγμα, ας επανέλθουμε στο παράδειγμα της Εικόνας 7.17 και ας υποθέσουμε ότι διαθέτουμε $k = 2$ αεροπλάνα. Αν χρησιμοποιήσουμε το ένα από τα αεροπλάνα για τις πτήσεις (1), (3), και (6) όπως προτείναμε προηγουμένως, δεν θα μπορούμε να εξυπηρετήσουμε όλες τις πτήσεις (2), (4), και (5) με το άλλο αεροπλάνο — επειδή δεν υπάρχει επαρκής διαθέσιμος χρόνος για συντήρηση στο Σαν Φρανσίσκο μεταξύ των πτήσεων (4) και (5). Υπάρχει, όμως, τρόπος να εξυπηρετήσουμε και τις έξι πτήσεις με δύο αεροπλάνα, μέσω μιας διαφορετικής λύσης: Το ένα αεροπλάνο εξυπηρετεί τις πτήσεις (1), (3), και (5) (με παρεμβολή μιας πτήσης LAX - SFO), ενώ το άλλο εξυπηρετεί τις πτήσεις (2), (4), και (6) (με παρεμβολή μιας πτήσης PIT - PHL και μιας πτήσης SFO - LAS).

Σχεδιασμός του αλγορίθμου

Θα εξετάσουμε τώρα έναν αποδοτικό αλγόριθμο ο οποίος μπορεί να λύσει οποιαδήποτε στιγμιότυπα του προβλήματος Χρονοπρογραμματισμού Αεροπορικών Δρομολογίων, και ο οποίος βασίζεται στη ροή δικτύου. Θα δούμε ότι οι τεχνικές ροής προσαρμόζονται με πολύ φυσικό τρόπο σε αυτό το πρόβλημα.

Η λύση βασίζεται στην ακόλουθη ιδέα. Οι μονάδες ροής θα αντιστοιχούν σε αεροπλάνα. Θα έχουμε μία ακμή για κάθε πτήση, με άνω και κάτω όριο 1 σε αυτές τις ακμές έτσι ώστε να απαιτούμε ότι ακριβώς μία μονάδα ροής θα διέρχεται μέσω της κάθε ακμής. Με άλλα λόγια, η κάθε πτήση θα πρέπει να εξυπηρετείται από ένα από τα αεροπλάνα. Αν (u_i, v_i) είναι η ακμή που αντιπροσωπεύει την πτήση i και (u_j, v_j) είναι η ακμή που αντιπροσωπεύει την πτήση j, και εάν η πτήση j είναι προσπελάσιμη από την πτήση i, τότε μπορούμε να έχουμε μια ακμή από το v_i στο u_j με χωρητικότητα 1· με αυτόν τον τρόπο, μια μονάδα ροής μπορεί να διασχίσει την ακμή (u_i, v_i) και μετά να μετακινηθεί απευθείας στην ακμή (u_j, v_j). Μια τέτοια κατασκευή ακμών φαίνεται στην Εικόνα 7.17(β).

Επεκτείνουμε αυτή την κατασκευή σε δίκτυο ροής συμπεριλαμβάνοντας μια προέλευση και μια απόληξη· ας δούμε τώρα λεπτομερώς ολόκληρη την κατασκευή. Το σύνολο κόμβων του υποκείμενου γραφήματος G ορίζεται με τον εξής τρόπο.

- Για κάθε πτήση i, το γράφημα G θα έχει τους δύο κόμβους u_i και v_i.
- Το G θα διαθέτει επίσης διακριτούς κόμβους προέλευσης s και απόληξης t.

Το σύνολο ακμών του G ορίζεται με τον εξής τρόπο.

- Για κάθε i, υπάρχει μια ακμή (u_i, v_i) με κάτω όριο 1 και χωρητικότητα 1. (*Θα πρέπει να εξυπηρετούνται όλες οι πτήσεις της λίστας.*)
- Για κάθε i και j τέτοια ώστε η πτήση j να είναι προσπελάσιμη από την πτήση i, υπάρχει μια ακμή (v_i, u_j) με κάτω όριο 0 και χωρητικότητα 1. (*Το ίδιο αεροπλάνο μπορεί να εκτελέσει τις πτήσεις i και j.*)
- Για κάθε i, υπάρχει μια ακμή (s, u_i) με κάτω όριο 0 και χωρητικότητα 1. (*Οποιοδήποτε αεροπλάνο μπορεί να ξεκινήσει την ημέρα με την πτήση i.*)
- Για κάθε j, υπάρχει μια ακμή (v_j, t) με κάτω όριο 0 και χωρητικότητα 1. (*Οποιοδήποτε αεροπλάνο μπορεί να τελειώσει την ημέρα με την πτήση j.*)
- Υπάρχει μια ακμή (s, t) με κάτω όριο 0 και χωρητικότητα k. *(Αν διαθέτουμε επιπλέον αεροπλάνα, δεν χρειάζεται να τα χρησιμοποιήσουμε για καμία από τις πτήσεις.)*

Τέλος, ο κόμβος s θα έχει ζήτηση $-k$ και ο κόμβος t θα έχει ζήτηση k. Όλοι οι άλλοι κόμβοι θα έχουν μηδενική ζήτηση.

Ο αλγόριθμός μας είναι η κατασκευή του δικτύου G και η αναζήτηση για μια εφικτή λύση στο δίκτυο αυτό. Θα αποδείξουμε τώρα την ορθότητα αυτού του αλγορίθμου.

Ανάλυση του αλγορίθμου

(7.54) *Υπάρχει τρόπος για την πραγματοποίηση όλων των πτήσεων με το πολύ k αεροπλάνα εάν και μόνο εάν υπάρχει μια εφικτή κυκλοφορία στο δίκτυο G.*

Απόδειξη. Καταρχήν, υποθέστε ότι υπάρχει τρόπος για την πραγματοποίηση όλων των πτήσεων με $k' \leq k$ αεροπλάνα. Το σύνολο των πτήσεων που πραγματοποιούνται από το κάθε μεμονωμένο αεροπλάνο ορίζει μια διαδρομή P στο δίκτυο G, και στέλνουμε μία μονάδα ροής σε κάθε τέτοια διαδρομή P. Για να ικανοποιηθεί η συνολική ζήτηση στους κόμβους s και t, στέλνουμε $k - k'$ μονάδες ροής στην ακμή (s, t). Η κυκλοφορία που προκύπτει ικανοποιεί όλες τις συνθήκες ζήτησης, χωρητικότητας, και κάτω ορίων.

Αντιστρόφως, ας εξετάσουμε μια εφικτή κυκλοφορία στο δίκτυο G. Από την (7.52) γνωρίζουμε ότι υπάρχει μια εφικτή κυκλοφορία με ακέραιες τιμές. Υποθέστε ότι k' μονάδες ροής στέλνονται σε ακμές εκτός της (s, t). Αφού όλες οι άλλες ακμές έχουν όριο χωρητικότητας 1, και αφού η κυκλοφορία έχει ακέραιες τιμές, η κάθε τέτοια ακμή που μεταφέρει ροή θα έχει ακριβώς μία μονάδα ροής.

Για να το μετατρέψουμε αυτό σε χρονοδιάγραμμα θα χρησιμοποιήσουμε την ίδια συλλογιστική που είδαμε και στην απόδειξη της (7.42), όπου μετατρέψαμε μια ροή σε μια συλλογή διαδρομών. Στην πραγματικότητα, εδώ η κατάσταση είναι ευκολότερη επειδή το γράφημα δεν έχει κύκλους. Θεωρήστε μία ακμή (s, u_i) που μεταφέρει μία μονάδα ροής. Από την αρχή διατήρησης προκύπτει ότι η (u_i, v_i) μεταφέρει μία μονάδα ροής και ότι θα πρέπει να υπάρχει μία μοναδική ακμή από τον κόμβο v_i που να μεταφέρει μία μονάδα ροής. Αν συνεχίσουμε με αυτόν τον τρόπο, θα κατασκευάσουμε μια διαδρομή P από το s στο t όπου η κάθε ακμή αυτής της διαδρομής μεταφέρει μία μονάδα ροής. Μπορούμε να εφαρμόσουμε αυτή τη συλλογιστική σε οποιαδήποτε ακμή της μορφής (s, u_j) που μεταφέρει μία μονάδα ροής· με αυτόν τον τρόπο θα πάρουμε k' διαδρομές από το s στο t, η κάθε μία από τις οποίες μεταφέρει μία μονάδα ροής. Στο σημείο αυτό, για κάθε διαδρομή P που κατασκευάσαμε κατ' αυτόν τον τρόπο, μπορούμε να αντιστοιχίσουμε ένα αεροπλάνο το οποίο θα πραγματοποιεί όλες τις πτήσεις που περιλαμβάνονται στη διαδρομή αυτή. ■

Επεκτάσεις: Μοντελοποίηση άλλων πτυχών του προβλήματος

Στην πραγματική ζωή, ο χρονοπρογραμματισμός αεροπορικών δρομολογίων καταναλώνει ατελείωτες ώρες CPU. Αναφέραμε, όμως, στην αρχή της ενότητας ότι η διατύπωσή μας είναι στην πραγματικότητα ένα απλοϊκό πρόβλημα: παραβλέπει αρκετούς προφανείς παράγοντες που θα πρέπει να ληφθούν υπόψη σε τέτοιες εφαρμογές. Καταρχήν, παραβλέπει το γεγονός ότι ένα δεδομένο αεροπλάνο μπορεί να πετάξει μόνο για ένα συγκεκριμένο αριθμό ωρών, και μετά θα πρέπει να αποσυρθεί προσωρινά για να υποβληθεί σε πιο εκτενή συντήρηση. Δεύτερον, καταστρώνουμε ένα βέλτιστο χρονοδιάγραμμα για μία μόνο ημέρα (ή τουλάχιστον για ένα χρονικό εύρος) παραβλέποντας ότι υπάρχει το χτες και το αύριο· στην πραγματικότητα θα θέλουμε στο τέλος της ημέρας N αυτά τα αεροπλάνα να είναι στις βέλτιστες θέσεις τους για την αρχή της ημέρας N+1. Τρίτον, όλα αυτά τα αεροπλάνα θα πρέπει να επανδρώνονται με πληρώματα, και παρά το ότι τα πληρώματα επαναχρησιμοποιούνται σε πολλές πτήσεις ισχύει για αυτά ένα διαφορετικό σύνολο περιορισμών, αφού οι άνθρωποι και τα αεροπλάνα έχουν διαφορετικό ρυθμό κόπωσης. Και όλα αυτά τα ζητήματα δεν επιχειρούν καν να αντιμετωπίσουν το γεγονός ότι η εξυπηρέτηση οποιουδήποτε μεμονωμένου τμήματος πτήσης δεν αποτελεί αυστηρό περιορισμό· αντίθετα, ο πραγματικός στόχος είναι η βελτιστοποίηση των εσόδων, και έτσι μπορούμε να επιλέξουμε μεταξύ πολλών δυνατών πτήσεων για να τις συμπεριλάβουμε στο πρόγραμμά μας (για να μην αναφέρουμε το σχεδιασμό μιας καλής τιμολογιακής πολιτικής για τους επιβάτες) έτσι ώστε να επιτύχουμε αυτόν το στόχο.

Τελικά, το μήνυμα είναι πιθανότατα το εξής: Οι τεχνικές ροής είναι χρήσιμες για την επίλυση προβλημάτων τέτοιου τύπου, και χρησιμοποιούνται ευρέως στην πράξη. Πράγματι, η λύση που δώσαμε παραπάνω είναι μια γενική προσέγγιση ως προς την αποδοτική επαναχρησιμοποίηση ενός περιορισμένου συνόλου πόρων, και μπορεί να εφαρμοστεί σε πολλές περιπτώσεις. Ταυτόχρονα, η λειτουργία μιας αεροπορικής εταιρείας με αποδοτικό τρόπο στην πραγματική ζωή είναι ένα πολύ δύσκολο πρόβλημα.

7.10 Τμηματοποίηση εικόνας

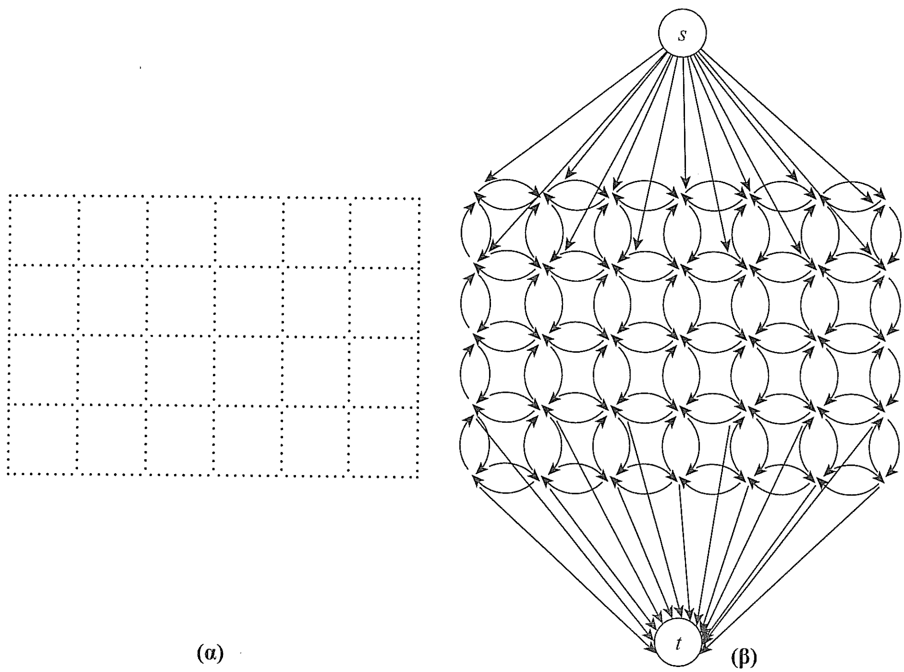
Ένα κεντρικό πρόβλημα στην επεξεργασία εικόνων είναι η *τμηματοποίηση* (segmentation) μιας εικόνας σε διάφορες λογικά συνεκτικές περιοχές. Για παράδειγμα, μπορεί να έχετε μια φωτογραφία τριών ανθρώπων οι οποίοι στέκονται μπροστά από ένα σύνθετο φόντο. Ένας φυσικός αλλά δύσκολος στόχος είναι ο προσδιορισμός των τριών αυτών ατόμων ως λογικών συνεκτικών αντικειμένων στη σκηνή.

Το πρόβλημα

Ένα από τα βασικότερα προβλήματα που εντάσσονται σε αυτό το πλαίσιο είναι το πρόβλημα του διαχωρισμού προσκηνίου και φόντου: Θέλουμε να δώσουμε σε κάθε εικονοστοιχείο μια ετικέτα που θα χαρακτηρίζει ότι ανήκει είτε στο προσκήνιο είτε στο φόντο της εικόνας. Αποδεικνύεται ότι ένα πολύ φυσικό μοντέλο για αυτό το ζήτημα οδηγεί σε ένα πρόβλημα που μπορεί να επιλυθεί αποδοτικά με υπολογισμό ελάχιστης αποκοπής.

Έστω ότι V είναι το σύνολο των *εικονοστοιχείων* (pixel) στην εικόνα την οποία αναλύουμε. Χαρακτηρίζουμε κάποια ζευγάρια εικονοστοιχείων ως *γειτονικά*, και χρησιμοποιούμε το E για να συμβολίσουμε το σύνολο όλων των ζευγαριών γειτονικών εικονοστοιχείων. Με αυτόν τον τρόπο παίρνουμε ένα *μη κατευθυνόμενο* γράφημα $G = (V, E)$. Θα είμαστε εσκεμμένα ασαφείς ως προς το τι εννοούμε ως "εικονοστοιχείο", ή τι εννοούμε ως "γειτονική" σχέση. Στην πραγματικότητα, οποιοδήποτε γράφημα G θα δίνει ένα πρόβλημα που μπορεί να επιλυθεί με αποδοτικό τρόπο, έτσι είμαστε ελεύθεροι να καθορίσουμε αυτές τις έννοιες με όποιον τρόπο θέλουμε. Προφανώς, είναι φυσικό να θεωρούμε ότι τα εικονοστοιχεία σχηματίζουν ένα πλέγμα κουκκίδων, και ότι οι γείτονες ενός εικονοστοιχείου είναι εκείνα τα εικονοστοιχεία που βρίσκονται ακριβώς δίπλα σε αυτό στο πλέγμα, όπως φαίνεται στην Εικόνα 7.18(α).

Για κάθε εικονοστοιχείο i, έχουμε μια *πιθανότητα* a_i να ανήκει στο προσκήνιο και μια *πιθανότητα* b_i να ανήκει στο φόντο. Για τους σκοπούς της μελέτης μας, θα θεωρήσουμε ότι αυτές οι τιμές πιθανοτήτων είναι αυθαίρετοι μη αρνητικοί αριθμοί που παρέχονται ως τμήμα του προβλήματος, και προσδιορίζουν πόσο επιθυμητό είναι να έχουμε το εικονοστοιχείο i στο φόντο ή στο προσκήνιο. Πέρα από αυτό, δεν είναι σημαντικό να καθορίσουμε ποιες φυσικές ιδιότητες της εικόνας μετρούν ή με ποιον τρόπο έχουν υπολογιστεί.



Εικόνα 7.18 (α) Ένα γράφημα εικονοστοιχείων. (β) Ένα σκιαγράφημα του αντίστοιχου δικτύου ροής. Δεν έχουν σχεδιαστεί όλες οι ακμές από την προέλευση ή προς την απόληξη.

Αν λειτουργούσαμε απομονωμένα, θα θέλαμε να χαρακτηρίσουμε το εικονοστοιχείο i ότι ανήκει στο προσκήνιο αν $a_i > b_i$ και ότι διαφορετικά ανήκει στο φόντο. Οι αποφάσεις, όμως, που παίρνουμε σχετικά με τους γείτονες του i θα πρέπει να επηρεάσουν την απόφασή μας για το i. Για παράδειγμα, αν πολλοί γείτονες του i έχουν χαρακτηριστεί ως "φόντο", θα πρέπει να έχουμε την τάση να χαρακτηρίσουμε και το i ως "φόντο"· αυτό θα κάνει το χαρακτηρισμό πιο "ομαλό", ελαχιστοποιώντας το πλήθος των ορίων προσκηνίου/φόντου. Έτσι, για κάθε ζευγάρι γειτονικών εικονοστοιχείων (i, j), υπάρχει μια *ποινή διαχωρισμού* (separation penalty) $p_{ij} \geq 0$ για την τοποθέτηση του ενός από τα i και j στο προσκήνιο και του άλλου στο φόντο.

Μπορούμε τώρα να προσδιορίσουμε με ακρίβεια το *Πρόβλημα Τμηματοποίησης*, με βάση τις παραμέτρους της πιθανότητας και του διαχωρισμού: Είναι η εύρεση μιας διαμέρισης του συνόλου εικονοστοιχείων στα σύνολα A και B (προσκήνιο και φόντο, αντίστοιχα), έτσι ώστε να μεγιστοποιείται η ποσότητα

$$q(A,B) = \sum_{i \in A} a_i + \sum_{j \in B} b_j - \sum_{\substack{(i,j) \in E \\ |A \cap \{i,j\}| = 1}} p_{ij}$$

Έτσι επιβραβευόμαστε όταν έχουμε υψηλές τιμές πιθανότητας και τιμωρούμαστε όταν έχουμε γειτονικά ζευγάρια (i, j) με το ένα εικονοστοιχείο στο σύνολο A και το άλλο στο B. Έτσι το πρόβλημα είναι ο υπολογισμός ενός *βέλτιστου χαρακτηρισμού* (optimal labelling) — μιας διαμέρισης (A, B) που να μεγιστοποιεί το $q(A, B)$.

Σχεδιασμός και ανάλυση του αλγορίθμου

Παρατηρούμε αμέσως ότι υπάρχει μια σαφής ομοιότητα μεταξύ του προβλήματος της ελάχιστης αποκοπής και του προβλήματος εύρεσης ενός βέλτιστου χαρακτηρισμού. Υπάρχουν, όμως, και κάποιες σημαντικές διαφορές. Καταρχήν, θέλουμε να μεγιστοποιήσουμε μια αντικειμενική συνάρτηση, αντί να την ελαχιστοποιήσουμε. Δεύτερο, δεν υπάρχει προέλευση και απόληξη στο πρόβλημα του χαρακτηρισμού· επιπρόσθετα, θα πρέπει να χειριστούμε τιμές a_i και b_i στους κόμβους. Τρίτον, έχουμε ένα μη κατευθυνόμενο γράφημα G, ενώ στο πρόβλημα της ελάχιστης αποκοπής θέλαμε να δουλέψουμε με ένα κατευθυνόμενο γράφημα. Ας αντιμετωπίσουμε αυτά τα προβλήματα ένα-ένα.

Για να αντιμετωπίσουμε το ζήτημα ότι το Πρόβλημα Τμηματοποίησης είναι ένα πρόβλημα μεγιστοποίησης, κάνουμε την εξής παρατήρηση. Έστω ότι $Q = \sum_i (a_i + b_i)$. Το άθροισμα $\sum_{i \in A} a_i + \sum_{j \in B} b_j$ είναι το ίδιο με το άθροισμα $Q - \sum_{i \in A} b_i - \sum_{j \in B} a_j$, έτσι μπορούμε να γράψουμε

$$q(A, B) = Q - \sum_{i \in A} b_i - \sum_{j \in B} a_j - \sum_{\substack{(i,j) \in E \\ |A \cap \{i,j\}| = 1}} p_{ij}$$

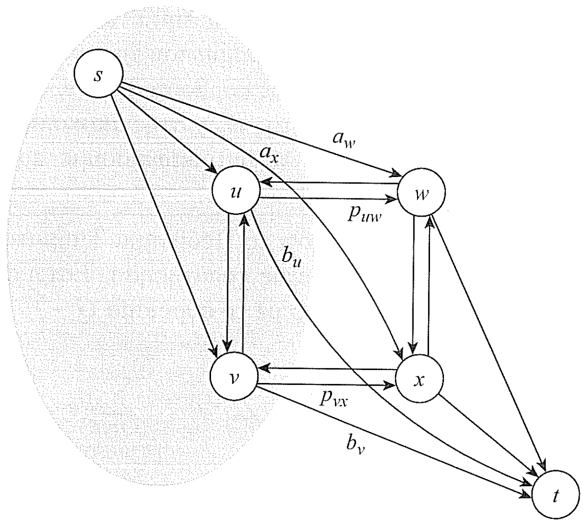
Έτσι βλέπουμε ότι η μεγιστοποίηση του $q(A, B)$ είναι ισοδύναμη με την ελαχιστοποίηση της ποσότητας

$$q'(A, B) = \sum_{i \in A} b_i + \sum_{j \in B} a_j + \sum_{\substack{(i,j) \in E \\ |A \cap \{i,j\}| = 1}} p_{ij}$$

Όσον αφορά το θέμα της έλλειψης της προέλευσης και της απόληξης, δουλεύουμε σε αναλογία με τις μεθόδους που χρησιμοποιήσαμε σε προηγούμενες ενότητες: Δημιουργούμε μια νέα "υπερπροέλευση" s που αντιπροσωπεύει το προσκήνιο και μια νέα "υπεραπόληξη" t που αντιπροσωπεύει το φόντο. Αυτό μας δίνει επίσης έναν τρόπο για να χειριστούμε τις τιμές a_i και b_i που ανήκουν στους κόμβους (ενώ οι ελάχιστες αποκοπές μπορούν να χειριστούν μόνο αριθμούς που σχετίζονται με ακμές). Ειδικότερα, θα συνδέσουμε το καθένα από τα s και t με όλα τα εικονοστοιχεία, και θα χρησιμοποιήσουμε τα a_i και b_i για να ορίσουμε κατάλληλες χωρητικότητες στις ακμές μεταξύ του εικονοστοιχείου i και της προέλευσης ή της απόληξης αντίστοιχα.

Τέλος, για να αντιμετωπίσουμε το ζήτημα των μη κατευθυνόμενων ακμών, θα μοντελοποιήσουμε το κάθε γειτονικό ζευγάρι (i, j) με *δύο* κατευθυνόμενες ακμές, (i, j) και (j, i), όπως κάναμε και στο μη κατευθυνόμενο πρόβλημα των Ασύνδετων Διαδρομών. Θα δούμε ότι η λύση αυτή λειτουργεί πολύ καλά και στην περίπτωση αυτή, αφού σε οποιαδήποτε αποκοπή s-t το πολύ η μία από αυτές τις δύο ακμές αντίθετης κατεύθυνσης μπορεί να διασχίζει από την πλευρά του s προς την πλευρά του t στην αποκοπή (επειδή αν συμβαίνει αυτό για τη μία ακμή, τότε η άλλη θα πρέπει να πηγαίνει από την πλευρά του t στην πλευρά του s).

Πιο συγκεκριμένα, ορίζουμε το ακόλουθο δίκτυο ροής $G' = (V', E')$ που φαίνεται στην Εικόνα 7.18(β). Το σύνολο κόμβων V' αποτελείται από το σύνολο V των εικονοστοιχείων μαζί με τους δύο πρόσθετους κόμβους s και t. Για κάθε γειτονικό ζευγάρι εικονοστοιχείων i και j, προσθέτουμε κατευθυνόμενες ακμές (i, j) και (j, i), όπου η καθεμία έχει χωρητικότητα p_{ij}. Για κάθε εικονοστοιχείο i, προσθέτουμε μία ακμή (s, i) με χωρητικότητα a_i και μία ακμή (i, t) με χωρητικότητα b_i.



Εικόνα 7.19 Μια αποκοπή s-t για ένα γράφημα που κατασκευάστηκε από τέσσερα εικονοστοιχεία. Παρατηρήστε με ποιον τρόπο οι τρεις τύποι όρων στην παράσταση για το $q'(A, B)$ αποτυπώνονται στην αποκοπή.

Τώρα μια αποκοπή s-t (A, B) αντιστοιχεί σε μια διαμέριση των εικονοστοιχείων στα σύνολα A και B. Ας εξετάσουμε με ποιον τρόπο σχετίζεται η χωρητικότητα της αποκοπής $c(A, B)$ με την ποσότητα $q'(A, B)$ την οποία προσπαθούμε να ελαχιστοποιήσουμε. Μπορούμε να ομαδοποιήσουμε τις ακμές που διασχίζουν την αποκοπή (A, B) σε τρεις φυσικές κατηγορίες.

- Ακμές (s, j), όπου $j \in B$· αυτές οι ακμές συνεισφέρουν κατά a_j στη χωρητικότητα της αποκοπής.
- Ακμές (i, t), όπου $i \in A$· αυτές οι ακμές συνεισφέρουν κατά b_i στη χωρητικότητα της αποκοπής.
- Ακμές (i, j) όπου $i \in A$ και $j \in B$· αυτές οι ακμές συνεισφέρουν κατά p_{ij} στη χωρητικότητα της αποκοπής.

Η Εικόνα 7.19 δείχνει πώς φαίνονται αυτά τα τρία είδη ακμών σε σχέση με την αποκοπή, για ένα παράδειγμα με τέσσερα εικονοστοιχεία.

Αν προσθέσουμε τις συνεισφορές αυτών των τριών ειδών ακμών, παίρνουμε

$$c(A, B) = \sum_{i \in A} b_i + \sum_{j \in B} a_j + \sum_{\substack{(i,j) \in E \\ |A \cap \{i,j\}| = 1}} p_{ij} = q'(A, B)$$

Έτσι τα πάντα ταιριάζουν μεταξύ τους. Το δίκτυο ροής είναι διαμορφωμένο έτσι ώστε η χωρητικότητα της αποκοπής (A, B) να μετρά επακριβώς την ποσότητα $q'(A, B)$: Τα τρία είδη ακμών που διασχίζουν την αποκοπή (A, B), όπως τα έχουμε ήδη ορίσει (ακμές από την προέλευση, ακμές προς την απόληξη, και ακμές που δεν περιλαμβάνουν ούτε την προέλευση ούτε την απόληξη) αντιστοιχούν στα τρία είδη όρων στην παράσταση για το $q'(A, B)$.

Έτσι, αν θέλουμε να ελαχιστοποιήσουμε το $q'(A, B)$ — αφού δείξαμε προηγουμένως ότι αυτό είναι ισοδύναμο με τη μεγιστοποίηση του $q(A,B)$ — θα πρέπει απλώς να βρούμε μια αποκοπή με ελάχιστη χωρητικότητα. Και αυτό το τελευταίο πρόβλημα, προφανώς, είναι κάτι που ξέρουμε πώς να λύνουμε με αποδοτικό τρόπο.

Κατά συνέπεια, μέσω της επίλυσης αυτού του προβλήματος ελάχιστης αποκοπής έχουμε ένα βέλτιστο αλγόριθμο στο μοντέλο μας για το διαχωρισμό προσκηνίου και φόντου.

(7.55) *Η επίλυση του προβλήματος Τμηματοποίησης μπορεί να επιτευχθεί από έναν αλγόριθμο ελάχιστης αποκοπής στο γράφημα G' που κατασκευάσαμε παραπάνω. Για μια ελάχιστη αποκοπή (A', B'), η διαμέριση (A, B) που λαμβάνεται με διαγραφή των s^* και t^* μεγιστοποιεί την τιμή τμηματοποίησης $q(A, B)$.*

7.11 Επιλογή έργων

Οι μεγάλες (και οι μικρές) εταιρείες αντιμετωπίζουν συνεχώς το πρόβλημα της εξισορρόπησης μεταξύ των έργων που μπορούν να παραγάγουν έσοδα και των εξόδων που απαιτούνται για τις ενέργειες οι οποίες θα μπορούν να υποστηρίξουν αυτά τα έργα. Υποθέστε, για παράδειγμα, ότι ο γίγαντας των τηλεπικοινωνιών CluNet εξετάζει τα υπέρ και τα κατά ενός έργου για την προσφορά ενός νέου τύπου υπηρεσίας γρήγορης πρόσβασης για τους πελάτες της των αστικών περιοχών. Η έρευνα αγοράς έδειξε ότι η υπηρεσία θα δώσει σημαντικά έσοδα, αλλά αυτά θα πρέπει να αντισταθμίσουν κάποια δαπανηρά προκαταρκτικά έργα τα οποία απαιτούνται προκειμένου να γίνει δυνατή αυτή η υπηρεσία: αύξηση της χωρητικότητας οπτικών ινών στον πυρήνα του δικτύου τους και αγορά δρομολογητών υψηλής ταχύτητας νέας γενιάς.

Αυτό που κάνει το συγκεκριμένο είδος αποφάσεων ιδιαίτερα δύσκολο είναι ότι υπάρχουν περίπλοκοι τρόποι αλληλεπιδράσεων: απομονωμένα, τα έσοδα από την υπηρεσία πρόσβασης υψηλής ταχύτητας μπορεί να μην είναι αρκετά για να δικαιολογήσουν την ανανέωση των δρομολογητών· *παρόλα αυτά*, όταν η εταιρεία ανανεώσει τους δρομολογητές της, θα είναι σε θέση να διεκδικήσει ένα πρόσθετο προσοδοφόρο έργο με τους εταιρικούς πελάτες της· και πιθανώς αυτό το πρόσθετο έργο να αλλάξει τις ισορροπίες. Και μάλιστα αυτές οι αλληλεπιδράσεις σχηματίζουν αλυσίδες: το έργο για τους εταιρικούς πελάτες απαιτεί στην πραγματικότητα και ένα ακόμα έξοδο, όμως αυτό με τη σειρά του θα επιτρέψει δύο άλλα προσοδοφόρα έργα — και ούτω καθεξής. Τελικά το ερώτημα είναι: Ποια έργα θα πρέπει να επιδιώξει η εταιρεία, και ποια θα πρέπει να απορρίψει; Αυτό είναι ένα βασικό πρόβλημα εξισορρόπησης του προκαλούμενου κόστους ως προς τις κερδοφόρες ευκαιρίες που γίνονται δυνατές.

Το πρόβλημα

Το ακόλουθο είναι ένα πολύ γενικό πλαίσιο εργασίας για τη μοντελοποίηση ενός τέτοιου συνόλου αποφάσεων. Υπάρχει ένα υποκείμενο σύνολο *έργων* P, και για κάθε έργο $i \in P$ υπάρχει ένα συσχετισμένο *έσοδο* p_i, το οποίο μπορεί να είναι είτε θετικό είτε αρνητικό. (Με άλλα λόγια, η κάθε προσοδοφόρα ευκαιρία και το κάθε δαπανηρό βήμα κατασκευής υποδομής του προηγούμενου παραδείγματός μας θα αναφέρονται ως ξεχωριστά έργα.) Κάποια έργα είναι προαπαιτούμενα για άλλα έργα, και το μοντελοποιούμε αυτό με ένα κατευθυνόμενο ακυκλικό γράφημα $G = (P, E)$. Οι κόμβοι του G είναι τα έργα, η ακμή (i, j) δείχνει ότι το έργο i μπορεί να επιλεγεί μόνο αν έχει επιλεγεί επίσης και το έργο j. Σημειώστε ότι ένα έργο i μπορεί να έχει πολλά προαπαιτούμενα, και ότι μπορούν να υπάρχουν πολλά έργα που να έχουν το έργο j ως ένα από τα προαπαιτούμενά τους. Ένα σύνολο έργων $A \subseteq P$ είναι *εφικτό* αν τα προαπαιτούμενα για όλα τα έργα του A ανήκουν επίσης στο A: για κάθε $i \in A$ και κάθε ακμή $(i, j) \in E$, έχουμε επίσης $j \in A$. Θα αναφερόμαστε στις απαιτήσεις αυτής τις μορφής με τον όρο *περιορισμοί προτεραιότητας* (precedence constraints). Το κέρδος για ένα σύνολο έργων ορίζεται ως

$$\texttt{profit}(A) = \sum_{i \in A} p_i$$

Το *Πρόβλημα Επιλογής Έργων* (Project Selection Problem) είναι να επιλέξουμε ένα εφικτό σύνολο έργων με μέγιστο κέρδος.

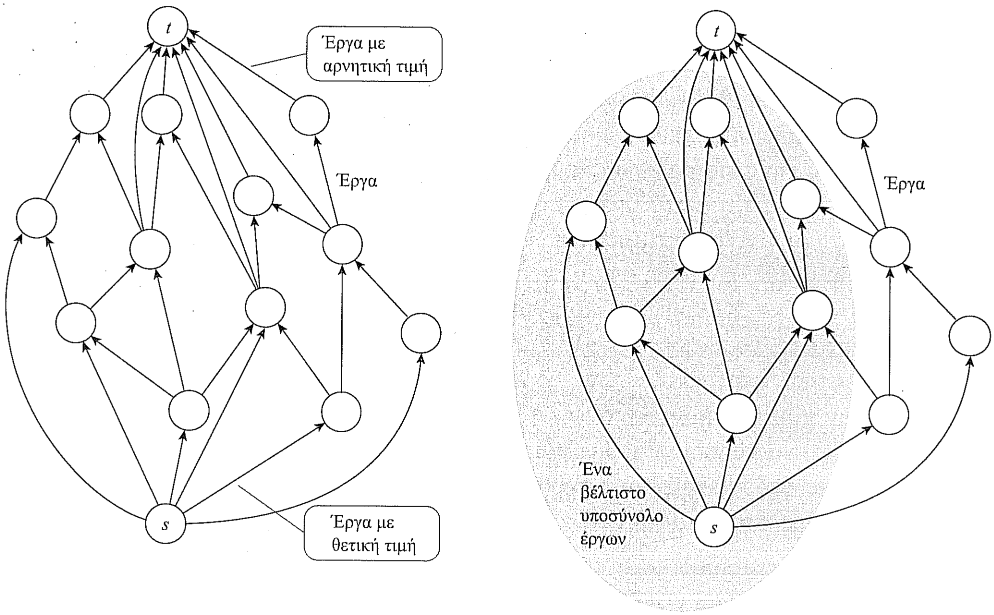
Αυτό το πρόβλημα, ήδη από τις αρχές της δεκαετίας του 1960, αποτέλεσε σημαντικό θέμα μελέτης στη βιβλιογραφία της μεταλλευτικής· εκεί ονομαζόταν *Πρόβλημα Μεταλλείων Ανοικτού Ορύγματος* (Open-Pit Mining Problem).[3] Τα μεταλλεία ανοικτού ορύγματος είναι μια επιφανειακή μεταλλευτική δραστηριότητα όπου κομμάτια της γης αποσπώνται από την επιφάνεια για να εξαχθεί το μετάλλευμα που υπάρχει μέσα σε αυτά. Πριν να ξεκινήσει η μεταλλευτική δραστηριότητα, ολόκληρη η περιοχή διαιρείται σε P *τμήματα*, και γίνεται εκτίμηση της καθαρής αξίας p_i για το κάθε τμήμα: Αυτή είναι η αξία του μεταλλεύματος μείον το κόστος επεξεργασίας, αν εξετάσουμε αυτό το τμήμα απομονωμένα. Κάποιες από αυτές τις καθαρές αξίες θα είναι θετικές, ενώ άλλες θα είναι αρνητικές. Το πλήρες σύνολο τμημάτων έχει περιορισμούς προτεραιότητας οι οποίοι ουσιαστικά αποτρέπουν την εξόρυξη ενός τμήματος πριν να γίνει η εξόρυξη των τμημάτων που βρίσκονται πάνω από αυτό. Το πρόβλημα Μεταλλείων Ανοικτού Ορύγματος είναι να προσδιορίσουμε ποιο είναι το πιο προσοδοφόρο σύνολο τμημάτων για εξόρυξη, με δεδομένους τους περιορισμούς προτεραιότητας. Αυτό το πρόβλημα εντάσσεται επίσης στο πλαίσιο επιλογής έργων — το κάθε τμήμα αντιστοιχεί σε ένα ξεχωριστό έργο.

Σχεδιασμός του αλγορίθμου

Θα δείξουμε τώρα ότι το πρόβλημα Επιλογής Έργων μπορεί να λυθεί με αναγωγή του σε υπολογισμό ελάχιστης αποκοπής για ένα εκτεταμένο γράφημα G', το οποίο ορίζεται με ανάλογο τρόπο όπως στην Ενότητα 7.10 για την τμηματοποίηση εικόνων. Η ιδέα είναι να κατασκευάσουμε το G' από το G με έναν τέτοιον τρόπο ώστε η "πλευρά της ελάχιστης αποκοπής που περιλαμβάνει την προέλευση" στο G' να αντιστοιχεί σε ένα βέλτιστο σύνολο έργων προς επιλογή.

Για να σχηματίσουμε το γράφημα G', προσθέτουμε στο γράφημα G μια νέα προέλευση s και μια νέα απόληξη t όπως φαίνεται στην Εικόνα 7.20. Για κάθε κόμβο $i \in P$ με $p_i > 0$ προσθέτουμε μια ακμή (s, i) με χωρητικότητα p_i. Για κάθε κόμβο $i \in P$ με $p_i < 0$ προσθέτουμε μια ακμή (i, t) με χωρητικότητα $-p_i$. Θα ορίσουμε αργότερα τις χωρητικότητες για τις ακμές του G. Μπορούμε όμως ήδη να δούμε ότι η χωρητικότητα της αποκοπής $(\{s\}, P \cup \{t\})$ είναι $C = \sum_{i \in P: p_i > 0} p_i$, έτσι η μέγιστη ροή στο δίκτυο αυτό είναι το πολύ ίση με C.

[3] Σε αντιδιαστολή με τον τομέα της εξόρυξης δεδομένων, ο οποίος αποτέλεσε το κίνητρο για πολλά από τα προβλήματα που εξετάσαμε προηγουμένως, εδώ μιλάμε για πραγματική εξόρυξη, όπου σκάβουμε για να εξορύξουμε υλικά από τη γη.



Εικόνα 7.20 Το γράφημα ροής που χρησιμοποιείται για την επίλυση του προβλήματος Επιλογής Έργων. Στα δεξιά φαίνεται μια πιθανή αποκοπή με ελάχιστη χωρητικότητα.

Θέλουμε να εξασφαλίσουμε ότι, αν το (A', B') είναι μια ελάχιστη αποκοπή σε αυτό το γράφημα, τότε το $A = A' - \{s\}$ ικανοποιεί τους περιορισμούς προτεραιότητας· δηλαδή, αν ο κόμβος $i \in A$ έχει μια ακμή $(i, j) \in E$, τότε θα πρέπει να έχουμε $j \in A$. Ιδεατά, ο πιο καθαρός τρόπος για να το εξασφαλίσουμε αυτό είναι να δώσουμε σε καθεμία από τις ακμές του G χωρητικότητα με τιμή ∞. Δεν έχουμε προηγουμένως ορίσει τυπικά τι σημαίνει η άπειρη χωρητικότητα, όμως δεν υπάρχει πρόβλημα σε αυτό: είναι απλώς μια ακμή για την οποία η συνθήκη χωρητικότητας δεν θέτει κανένα άνω όριο. Οι αλγόριθμοι των προηγούμενων ενοτήτων, καθώς και το θεώρημα Μέγιστης Ροής και Ελάχιστης Αποκοπής, μπορούν να επεκταθούν για να χειρίζονται άπειρες χωρητικότητες. Παρόλα αυτά, μπορούμε επίσης να αποφύγουμε την εισαγωγή της έννοιας των άπειρων χωρητικοτήτων αποδίδοντας απλώς σε καθεμία από αυτές τις ακμές μια χωρητικότητα που να είναι "πρακτικά άπειρη". Στα πλαίσια του προβλήματός μας, η απόδοση χωρητικότητας $C + 1$ σε καθεμία από αυτές τις ακμές θα επιτύγχανε αυτόν το στόχο: Η μέγιστη δυνατή ροή στο G' είναι το πολύ ίση με C, και έτσι καμία ελάχιστη αποκοπή δεν μπορεί να περιέχει ακμή με χωρητικότητα μεγαλύτερη από C. Στην περιγραφή που ακολουθεί δεν έχει σημασία ποια από τις δύο αυτές προσεγγίσεις θα επιλέξουμε.

Μπορούμε τώρα να διατυπώσουμε τον αλγόριθμο: Υπολογίζουμε μια ελάχιστη αποκοπή (A', B') στο G', και δηλώνουμε ότι το $A' - \{s\}$ είναι ένα βέλτιστο σύνολο έργων. Θα αποδείξουμε τώρα ότι αυτός ο αλγόριθμος δίνει πραγματικά μια βέλτιστη λύση.

Ανάλυση του αλγορίθμου

Θεωρήστε καταρχήν ένα σύνολο έργων A που ικανοποιεί τους περιορισμούς προτεραιότητας. Έστω ότι $A' = A \cup \{s\}$ και $B' = (P - A) \cup \{t\}$, και εξετάστε την αποκοπή s-t (A', B'). Αν το σύνολο A ικανοποιεί τους περιορισμούς προτεραιότητας, τότε καμία ακμή $(i, j) \in E$ δεν διασχίζει αυτή την αποκοπή, όπως φαίνεται στην Εικόνα 7.20. Η χωρητικότητα της αποκοπής μπορεί να εκφραστεί με τον ακόλουθο τρόπο.

(7.56) *Η χωρητικότητα της αποκοπής (A', B'), όπως ορίζεται από ένα σύνολο έργων A που ικανοποιεί τους περιορισμούς προτεραιότητας, είναι $c(A', B') = C - \sum_{i \in A} p_i$.*

Απόδειξη. Οι ακμές του G' μπορούν να χωριστούν σε τρεις κατηγορίες: αυτές που αντιστοιχούν στο σύνολο ακμών E του G, αυτές που ξεκινούν από την προέλευση s, και αυτές που καταλήγουν στην απόληξη t. Αφού το A ικανοποιεί τους περιορισμούς προτεραιότητας, οι ακμές του E δεν διασχίζουν την αποκοπή (A', B'), και έτσι δεν συνεισφέρουν στη χωρητικότητά της. Οι ακμές που καταλήγουν στην απόληξη t συνεισφέρουν κατά

$$\sum_{i \in A \text{ και } p_i < 0} -p_i$$

στη χωρητικότητα της αποκοπής, ενώ οι ακμές που ξεκινούν από την προέλευση s συνεισφέρουν κατά

$$\sum_{i \notin A \text{ και } p_i > 0} p_i \,.$$

Χρησιμοποιώντας τον ορισμό του C, μπορούμε να ξαναγράψουμε την τελευταία ποσότητα ως $C - \sum_{i \in A \text{ και } p_i > 0} p_i$. Η χωρητικότητα της αποκοπής (A', B') είναι το άθροισμα αυτών των δύο όρων, το οποίο είναι

$$\sum_{i \in A \text{ και } p_i < 0} (-p_i) + \left(C - \sum_{i \in A \text{ και } p_i > 0} p_i \right) = C - \sum_{i \in A} p_i$$

όπως ισχυριστήκαμε. ∎

Θυμηθείτε τώρα ότι οι ακμές του G έχουν χωρητικότητα μεγαλύτερη από $C = \sum_{i \in P : p_i > 0} p_i$, και έτσι αυτές οι ακμές δεν μπορούν να διασχίζουν μια αποκοπή με μέγιστη χωρητικότητα C. Αυτό μας δείχνει ότι οι αποκοπές αυτές ορίζουν εφικτά σύνολα έργων.

(7.57) *Αν η (A', B') είναι μια αποκοπή με μέγιστη χωρητικότητα C, τότε το σύνολο $A = A' - \{s\}$ ικανοποιεί τους περιορισμούς προτεραιότητας.*

Μπορούμε πλέον να αποδείξουμε το κύριο ζήτημα της κατασκευής μας, ότι δηλαδή η ελάχιστη αποκοπή στο G' προσδιορίζει το βέλτιστο σύνολο έργων. Συνδυάζοντας τα δύο προηγούμενα συμπεράσματα, βλέπουμε ότι οι αποκοπές (A', B') που έχουν χωρητικότητα το πολύ ίση με C αποτελούν μια αντιστοιχία ένα προς ένα με τα εφικτά σύνολα έργων $A = A' - \{s\}$. Η χωρητικότητα μιας τέτοιας αποκοπής (A', B') είναι $c(A', B') = C - \texttt{profit}(A)$.

Η τιμή χωρητικότητας C είναι μια σταθερά, ανεξάρτητη από την αποκοπή (A', B'), έτσι η αποκοπή με την ελάχιστη χωρητικότητα αντιστοιχεί στο σύνολο έργων A με το μέγιστο κέρδος. Άρα έχουμε αποδείξει το ακόλουθο.

(7.58) *Αν η (A', B') είναι μια ελάχιστη αποκοπή στο G', τότε το σύνολο $A = A' - \{s\}$ αποτελεί μια βέλτιστη λύση για το πρόβλημα της Επιλογής Έργων.*

7.12 Αποκλεισμός στο μπέιζμπολ

Στο ραδιοφωνικό σταθμό ο παραγωγός τους είπε, "Το διαβάσατε αυτό για τον Αϊνστάιν στην εφημερίδα την προηγούμενη εβδομάδα; ... Κάποιος δημοσιογράφος του ζήτησε να υπολογίσει τις μαθηματικές πιθανότητες για το πρωτάθλημα. Ξέρετε, η μία ομάδα κερδίζει τόσους από τους υπόλοιπους αγώνες της, οι άλλες ομάδες κερδίζουν τόσους ή τόσους αγώνες. Ποιες είναι οι μυριάδες πιθανότητες; Ποιος έχει το πλεονέκτημα;"

"Και τι ξέρει αυτός;"

"Απ' ό,τι φαίνεται όχι πολλά. Διάλεξε ότι οι Dodgers θα αποκλείσουν τους Giants την προηγούμενη Παρασκευή."

—Don DeLillo, Underworld

Το πρόβλημα

Υποθέστε ότι είστε δημοσιογράφοι στα *Αλγοριθμικά Αθλητικά Νέα* και ότι τον προηγούμενο Σεπτέμβριο προέκυψε η ακόλουθη κατάσταση. Υπάρχουν τέσσερις ομάδες μπέιζμπολ που προσπαθούν να τερματίσουν στην πρώτη θέση στην Ανατολική Περιφέρεια του Αμερικανικού πρωταθλήματος· ας τις ονομάσουμε Νέα Υόρκη, Βαλτιμόρη, Τορόντο, και Βοστόνη. Τη δεδομένη στιγμή η κάθε μία από τις ομάδες είχε τον ακόλουθο αριθμό νικών:

Νέα Υόρκη: 92, Βαλτιμόρη: 91, Τορόντο: 91, Βοστόνη: 90.

Μένουν πέντε αγώνες για την ολοκλήρωση του πρωταθλήματος: σε αυτούς περιλαμβάνονται όλα τα δυνατά ζευγάρια μεταξύ των τεσσάρων παραπάνω ομάδων, εκτός από τον αγώνα Νέας Υόρκης εναντίον Βοστόνης.

Το ερώτημα είναι: Μπορεί η Βοστόνη να τερματίσει με τουλάχιστον τόσες νίκες όπως και οποιαδήποτε άλλη ομάδα αυτής της περιφέρειας (δηλαδή, να τερματίσει στην πρώτη θέση, ακόμα και σε ισοβαθμία);

Αν το σκεφτείτε, θα δείτε ότι η απάντηση είναι αρνητική. Μια απόδειξη είναι η ακόλουθη. Είναι ξεκάθαρο ότι η Βοστόνη θα πρέπει να κερδίσει και τα δύο εναπομείναντα παιχνίδια της ενώ η Νέα Υόρκη θα πρέπει να τα χάσει και τα δύο. Όμως αυτό σημαίνει ότι η Βαλτιμόρη και το Τορόντο θα κερδίσουν και οι δύο τη Νέα Υόρκη· έτσι ο νικητής του αγώνα Βαλτιμόρη-Τορόντο θα καταλήξει να έχει περισσότερες νίκες από τη Βοστόνη.

Ας δούμε έναν άλλον τρόπο απόδειξης που αποφεύγει αυτό το είδος ανάλυσης περιπτώσεων. Η Βοστόνη μπορεί να τερματίσει το πολύ με 92 νίκες. Αθροιστικά, οι υπόλοιπες τρεις ομάδες έχουν αυτή τη στιγμή 274 νίκες, και οι τρεις εναπομείναντες αγώνες μεταξύ τους θα δώσουν ακριβώς τρεις ακόμα νίκες, οπότε το τελικό σύνολο νικών θα είναι 277. Όμως οι 277 νίκες για τρεις ομάδες σημαίνουν ότι η μία από αυτές θα πρέπει να τερματίσει με περισσότερες από 92 νίκες.

Έτσι μπορεί να αρχίσετε να αναρωτιέστε: (i) Υπάρχει κάποιος αποδοτικός αλγόριθμος που να προσδιορίζει αν μια ομάδα δεν μπορεί πλέον να κατακτήσει την πρώτη θέση; Και (ii) όταν μια ομάδα έχει πλέον χάσει την πρώτη θέση, υπάρχει ένα "μεσοσταθμικό" επιχείρημα όπως το προηγούμενο που να το αποδεικνύει αυτό;

Με πιο συγκεκριμένη σημειογραφία, υποθέστε ότι έχουμε ένα σύνολο S από ομάδες, και για κάθε $x \in S$ ο τρέχων αριθμός νικών του είναι w_x. Επίσης, για δύο ομάδες $x, y \in S$, απομένουν ακόμα g_{xy} παιχνίδια μεταξύ τους. Τέλος, μας δίνεται μια συγκεκριμένη ομάδα z.

Θα χρησιμοποιήσουμε τεχνικές μέγιστης ροής για να επιτύχουμε τα ακόλουθα δύο πράγματα. Πρώτον, θα δώσουμε έναν αποδοτικό αλγόριθμο που να αποφασίζει αν η ομάδα z έχει χάσει την πρώτη θέση — ή, για να το θέσουμε διαφορετικά, αν είναι δυνατό να επιλέξουμε τα αποτελέσματα των υπόλοιπων αγώνων με τέτοιον τρόπο ώστε η ομάδα z να τερματίσει με τουλάχιστον ίσες νίκες με οποιαδήποτε άλλη ομάδα του συνόλου S. Δεύτερο, θα αποδείξουμε το ακόλουθο ξεκάθαρο θεώρημα χαρακτηρισμού για τον αποκλεισμό στο μπέιζμπολ — ουσιαστικά, ότι υπάρχει πάντα μια σύντομη "απόδειξη" όταν μια ομάδα έχει αποκλειστεί.

(7.59) *Υποθέστε ότι η ομάδα z έχει πράγματι αποκλειστεί. Τότε υπάρχει μια "απόδειξη" αυτού του γεγονότος με την ακόλουθη μορφή:*

- *η ομάδα z μπορεί να τερματίσει το πολύ με m νίκες.*
- *Υπάρχει ένα σύνολο ομάδων $T \subseteq S$ τέτοιο ώστε*

$$\sum_{x \in T} w_x + \sum_{x,y \in T} g_{xy} > m|T|.$$

(Και έτσι μία από τις ομάδες του T θα πρέπει να τερματίσει με περισσότερες από m νίκες.)

Ως ένα δεύτερο, πιο σύνθετο παράδειγμα του τρόπου με τον οποίο δουλεύει το μεσοσταθμικό επιχείρημα της (7.59), σκεφτείτε την ακόλουθη περίπτωση. Υποθέστε ότι έχουμε τις ίδιες τέσσερις ομάδες όπως και προηγουμένως, όμως τώρα ο τρέχων αριθμός νικών είναι

Νέα Υόρκη: 90, Βαλτιμόρη: 88, Τορόντο: 87, Βοστόνη: 79.

Οι εναπομείναντες αγώνες είναι οι ακόλουθοι. Η Βοστόνη έχει ακόμα να παίξει τέσσερις αγώνες με την καθεμία από τις τρεις άλλες ομάδες. Η Βαλτιμόρη έχει έναν ακόμα αγώνα με τη Νέα Υόρκη και το Τορόντο. Και τέλος, η Νέα Υόρκη και το Τορόντο έχουν ακόμα να παίξουν έξι αγώνες μεταξύ τους. Είναι ξεκάθαρο ότι τα πράγματα δεν φαίνονται ρόδινα για τη Βοστόνη, όμως έχει πραγματικά αποκλειστεί;

Η απάντηση είναι ναι· η Βοστόνη έχει αποκλειστεί. Για να το δείτε αυτό, σημειώστε καταρχήν ότι η Βοστόνη μπορεί να τερματίσει το πολύ με 91 νίκες· εξετάστε τώρα το σύνολο ομάδων T = {Νέα Υόρκη, Τορόντο}. Η Νέα Υόρκη και το Τορόντο έχουν μαζί 177 νίκες· οι έξι εναπομείναντες αγώνες τους θα μας δώσουν σύνολο νικών 183· και $\frac{183}{2} > 91$. Αυτό σημαίνει ότι η μία από αυτές τις ομάδες θα πρέπει να τερματίσει με περισσότερες από 91 νίκες, και έτσι η Βοστόνη δεν μπορεί να τερματίσει πρώτη. Είναι ενδιαφέρον ότι στη συγκεκριμένη περίπτωση το σύνολο και των τριών ομάδων που προηγούνται της Βοστόνης δεν μπορεί να μας δώσει μια παρόμοια απόδειξη: Αν πάρουμε και τις τρεις ομάδες μαζί έχουμε ένα σύνολο 265 νικών με 8 εναπομείναντα παιχνίδια μεταξύ τους· αυτό μας δίνει άθροισμα 273, και $\frac{273}{3} = 91$ — που δεν αρκεί από μόνο του για να μας αποδείξει ότι η Βοστόνη δεν μπορεί να τερματίσει πρώτη σε πολλαπλή ισοβαθμία. Έτσι είναι κρίσιμο για το μεσοσταθμικό επιχείρημα να επιλέξουμε το σύνολο T που αποτελείται μόνο από τη Νέα Υόρκη και το Τορόντο, και να παραλείψουμε τη Βαλτιμόρη.

Σχεδιασμός και ανάλυση του αλγορίθμου

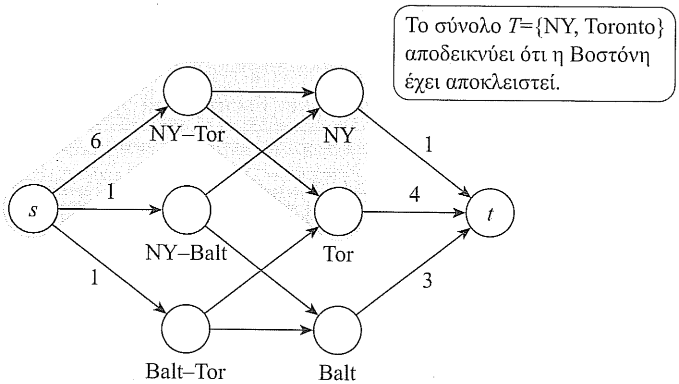
Θα ξεκινήσουμε με την κατασκευή ενός δικτύου ροής που να παρέχει έναν αποδοτικό αλγόριθμο για τον προσδιορισμό του αποκλεισμού του z. Στη συνέχεια, εξετάζοντας την ελάχιστη αποκοπή σε αυτό το δίκτυο, θα αποδείξουμε την (7.59).

Είναι προφανές ότι, αν υπάρχει κάποιος τρόπος να τερματίσει η ομάδα z στην πρώτη θέση, θα πρέπει η ομάδα z να κερδίσει όλους τους υπόλοιπους αγώνες της. Ας υποθέσουμε ότι αυτό της δίνει συνολικά m νίκες. Θέλουμε τώρα να κατανείμουμε προσεκτικά τις νίκες στους υπόλοιπους αγώνες έτσι ώστε καμία από τις άλλες ομάδες να μην καταλήξει με περισσότερες από m νίκες. Αυτού του είδους η εκχώρηση νικών μπορεί να επιλυθεί με υπολογισμό μέγιστης ροής, μέσω της ακόλουθης βασικής ιδέας. Έχουμε την προέλευση s από την οποία "πηγάζουν" όλες οι νίκες. Η νίκη i μπορεί να περνά μέσα από μία από τις δύο ομάδες που παίζουν στον αγώνα i. Επιβάλλουμε στη συνέχεια έναν περιορισμό χωρητικότητας που λέει ότι μπορούν να διέλθουν μέχρι $m - w_x$ νίκες από την ομάδα x.

Πιο συγκεκριμένα, κατασκευάζουμε το ακόλουθο δίκτυο ροής G, όπως φαίνεται στην Εικόνα 7.21. Καταρχήν, έστω ότι $S' = S - \{z\}$, και έστω ότι $g^* = \sum_{x,y \in S'} g_{xy}$ — το συνολικό πλήθος αγώνων που απομένουν μεταξύ όλων των ζευγαριών ομάδων του S'. Συμπεριλαμβάνουμε τους κόμβους s και t, έναν κόμβο v_x για κάθε ομάδα $x \in S'$, και έναν κόμβο u_{xy} για κάθε ζευγάρι $x, y \in S'$ για το οποίο απομένει μη μηδενικός αριθμός αγώνων. Έχουμε τις ακόλουθες ακμές.

- Ακμές (s, u_{xy}) (η νίκες πηγάζουν από το s);

- Ακμές (u_{xy}, v_x) και (u_{xy}, v_y) (μόνο το x ή το y μπορεί να κερδίσει σε έναν αγώνα μεταξύ αυτών των δύο ομάδων)· και
- Ακμές (v_x, t) (οι νίκες απορροφώνται από το t).



Εικόνα 7.21 Το δίκτυο ροής για το δεύτερο παράδειγμα. Όπως δείχνει η ελάχιστη αποκοπή, δεν υπάρχει ροή με τιμή g^*, και έτσι η Βοστόνη έχει αποκλειστεί.

Ας εξετάσουμε τι χωρητικότητες θέλουμε να τοποθετήσουμε σε αυτές τις ακμές. Θέλουμε g_{xy} νίκες να ρέουν από το s στο u_{xy} κατά τον κορεσμό, έτσι δίνουμε στο (s, u_{xy}) χωρητικότητα g_{xy}. Θέλουμε να εξασφαλίσουμε ότι η ομάδα x δεν μπορεί να κερδίσει περισσότερους από $m - w_x$ αγώνες, έτσι δίνουμε στην ακμή (v_x, t) χωρητικότητα $m - w_x$. Τέλος, μια ακμή της μορφής (u_{xy}, v_y) θα πρέπει να έχει *τουλάχιστον* g_{xy} μονάδες χωρητικότητας, έτσι ώστε να μπορεί να μεταφέρει όλες τις νίκες από το u_{xy} στο v_x· στην πραγματικότητα, η ανάλυσή μας θα είναι πιο ξεκάθαρη αν της δώσουμε *άπειρη* χωρητικότητα. (Σημειώστε ότι η κατασκευή μας δουλεύει ακόμα και αν της δώσουμε μόνο g_{xy} μονάδες χωρητικότητας, όμως η απόδειξη της (7.59) θα γίνει κάπως πιο περίπλοκη.)

Εάν τώρα υπάρχει μια ροή με τιμή g^*, τότε είναι δυνατό τα αποτελέσματα των υπόλοιπων αγώνων να δώσουν μια κατάσταση στην οποία καμία ομάδα δεν έχει περισσότερες από m νίκες· έτσι, αν η ομάδα z κερδίσει όλα τα υπόλοιπα παιχνίδια της, θα μπορεί να επιτύχει τουλάχιστον ισοβαθμία για την πρώτη θέση. Αντιστρόφως, αν υπάρχουν αποτελέσματα των υπόλοιπων παιχνιδιών με τα οποία η ομάδα z επιτυγχάνει τουλάχιστον ισοβαθμία στην πρώτη θέση, μπορούμε να χρησιμοποιήσουμε τα αποτελέσματα αυτά για να ορίσουμε μια ροή με τιμή g^*. Για παράδειγμα, στην Εικόνα 7.21, η οποία βασίζεται στο δεύτερο παράδειγμά μας, η σημειωμένη αποκοπή δείχνει ότι η μέγιστη ροή έχει τιμή το πολύ ίση με 7, ενώ $g^* = 6 + 1 + 1 = 8$.

Συνοψίζοντας, έχουμε

(7.60) *Η ομάδα z έχει αποκλειστεί εάν και μόνο εάν η μέγιστη ροή στο G έχει τιμή απολύτως μικρότερη από g^*. Έτσι μπορούμε να ελέγξουμε σε πολυωνυμικό χρόνο αν η ομάδα z έχει αποκλειστεί.*

Χαρακτηρισμός εάν μια ομάδα έχει αποκλειστεί

Η ροή δικτύου που κατασκευάσαμε μπορεί επίσης να χρησιμοποιηθεί για να αποδείξουμε την (7.59). Η ιδέα είναι ότι το θεώρημα Μέγιστης Ροής και Ελάχιστης Αποκοπής παρέχει έναν όμορφο χαρακτηρισμό "εάν και μόνο εάν" για την ύπαρξη της ροής, και αν ερμηνεύσουμε αυτόν το χαρακτηρισμό στα πλαίσια της εφαρμογής μας βρίσκουμε και για αυτή έναν εξίσου όμορφο χαρακτηρισμό. Αυτό μας δείχνει ένα γενικό τρόπο με τον οποίο μπορεί κανείς να παραγάγει θεωρήματα χαρακτηρισμού για προβλήματα που ανάγονται σε ροή δικτύου.

Απόδειξη της (7.59). Υποθέστε ότι η ομάδα z έχει αποκλειστεί από την πρώτη θέση. Τότε η μέγιστη ροή s-t στο G έχει τιμή $g' < g^*$· έτσι υπάρχει μια αποκοπή s-t (A, B) με χωρητικότητα g', και η (A, B) είναι ελάχιστη αποκοπή. Έστω ότι το T είναι το σύνολο των ομάδων x για τις οποίες $v_x \in A$. Θα δείξουμε τώρα ότι το T μπορεί να χρησιμοποιηθεί για το "μεσοσταθμικό επιχείρημα" (ή "επιχείρημα μέσου όρου") της (7.59).

Καταρχήν, θεωρήστε τον κόμβο u_{xy} και υποθέστε ότι το ένα από τα x και y δεν ανήκει στο T αλλά $u_{xy} \in A$. Τότε η ακμή (u_{xy}, v_x) θα διασχίζει από το A στο B, και έτσι η (A, B) θα έχει άπειρη χωρητικότητα. Αυτό έρχεται σε αντίφαση με την υπόθεση ότι η (A, B) είναι μια ελάχιστη αποκοπή με χωρητικότητα μικρότερη από g^*. Έτσι, αν το ένα από τα x και y δεν ανήκει στο T, τότε $u_{xy} \in B$. Από την άλλη πλευρά, υποθέστε ότι και το x και το y ανήκουν στο T αλλά $u_{xy} \in B$. Εξετάστε την αποκοπή (A', B') που θα παίρναμε αν προσθέταμε τη u_{xy} στο σύνολο A και τη διαγράφαμε από το σύνολο B. Η χωρητικότητα της (A', B') είναι απλώς η χωρητικότητα της (A, B) μείον τη χωρητικότητα g_{xy} της ακμής (s, u_{xy}) — επειδή αυτή η ακμή (s, u_{xy}) διέσχιζε προηγουμένως από το A στο B αλλά τώρα δεν διασχίζει από το A' στο B'. Όμως αφού $g_{xy} > 0$, αυτό σημαίνει ότι η (A', B') έχει μικρότερη χωρητικότητα από την αποκοπή (A, B), γεγονός που και πάλι έρχεται σε αντίφαση με την υπόθεσή μας ότι η (A, B) αποτελεί ελάχιστη αποκοπή. Έτσι, αν και το x και το y ανήκουν στο T, τότε $u_{xy} \in A$.

Έχουμε λοιπόν καταλήξει στο ακόλουθο συμπέρασμα, με βάση το γεγονός ότι η (A, B) είναι μια ελάχιστη αποκοπή: $u_{xy} \in A$ εάν και μόνο εάν και το x και το $y \in T$.

Πρέπει τώρα να διατυπώσουμε τη χωρητικότητα της ελάχιστης αποκοπής $c(A, B)$ ως προς τις συνιστώσες χωρητικότητες ακμών. Από το συμπέρασμα της προηγούμενης παραγράφου, γνωρίζουμε ότι οι ακμές που διασχίζουν από το A στο B έχουν μία από τις παρακάτω δύο μορφές:

- ακμές της μορφής (v_x, t), όπου $x \in T$, και
- ακμές της μορφής (s, u_{xy}), όπου τουλάχιστον το ένα από τα x ή y δεν ανήκει στο T (με άλλα λόγια, $\{x, y\} \not\subset T$).

Έτσι έχουμε

$$\begin{aligned} c(A, B) &= \sum_{x \in T} (m - w_x) + \sum_{\{x,y\} \not\subset T} g_{xy} \\ &= m|T| - \sum_{x \in T} w_x + (g^* - \sum_{x,y \in T} g_{xy}). \end{aligned}$$

Αφού γνωρίζουμε ότι $c(A, B) = g' < g^*$, η τελευταία ανισότητα μας δείχνει ότι

$$m|T| - \sum_{x \in T} w_x - \sum_{x,y \in T} g_{xy} < 0,$$

και έτσι

$$\sum_{x \in T} w_x + \sum_{x,y \in T} g_{xy} > m|T|. \quad \blacksquare$$

Για παράδειγμα, αν εφαρμόσουμε το επιχείρημα της απόδειξης της (7.59) στο στιγμιότυπο της Εικόνας 7.21, βλέπουμε ότι οι κόμβοι για τη Νέα Υόρκη και το Τορόντο βρίσκονται στην "πλευρά της ελάχιστης αποκοπής που περιλαμβάνει την προέλευση" και, όπως είδαμε προηγουμένως, αυτές οι δύο ομάδες συνιστούν πραγματικά απόδειξη για το ότι η Βοστόνη έχει αποκλειστεί.

*7.13 Μια πρόσθετη κατεύθυνση: Προσθήκη κόστους στο πρόβλημα Ταιριάσματος

Ας επιστρέψουμε στο πρώτο πρόβλημα που αναλύσαμε σε αυτό το κεφάλαιο, το πρόβλημα του Διμερούς Ταιριάσματος. Τα τέλεια ταιριάσματα σε ένα διμερές γράφημα μας έδωσαν έναν τρόπο μοντελοποίησης για το πρόβλημα του ζευγαρώματος ενός είδους αντικειμένων με ένα άλλο — για παράδειγμα, εργασίες με μηχανές. Όμως σε πολλές περιπτώσεις υπάρχει μεγάλος αριθμός δυνατών τέλειων ταιριασμάτων για το ίδιο σύνολο αντικειμένων, και θα θέλαμε έναν τρόπο για να εκφράσουμε την ιδέα ότι κάποια τέλεια ταιριάσματα μπορεί να είναι "καλύτερα" από άλλα.

Το πρόβλημα

Ένας φυσικός τρόπος για τη διατύπωση ενός προβλήματος που βασίζεται σε αυτή την ιδέα είναι η εισαγωγή της έννοιας του *κόστους*. Πιθανόν να προκύπτει κάποιο κόστος από την εκτέλεση μιας δεδομένης εργασίας σε μια δεδομένη μηχανή, και θα θέλαμε να ταιριάξουμε τις εργασίες και τις μηχανές με έναν τρόπο που να ελαχιστοποιεί το συνολικό κόστος. Ή μπορεί να υπάρχουν n πυροσβεστικά οχήματα που θα πρέπει να σταλούν σε n διαφορετικά σπίτια· το κάθε σπίτι βρίσκεται σε μια δεδομένη απόσταση από το σταθμό της πυροσβεστικής, και θα θέλαμε να βρούμε ένα ταίριασμα που να ελαχιστοποιεί τη μέση απόσταση την οποία διανύει το κάθε όχημα μέχρι το αντίστοιχο σπίτι. Με λίγα λόγια, είναι πολύ χρήσιμο να έχουμε έναν αλγόριθμο που να βρίσκει ένα τέλειο ταίριασμα *με το ελάχιστο συνολικό κόστος*.

Με τυπικούς όρους, θεωρούμε ένα διμερές γράφημα $G = (V, E)$ του οποίου το σύνολο κόμβων, όπως συνήθως, διαμερίζεται ως $V = X \cup Y$ έτσι ώστε η κάθε ακμή $e \in E$ να έχει ένα άκρο στο X και το άλλο άκρο στο Y. Επιπρόσθετα, η κάθε ακμή e έχει ένα μη αρνητικό κόστος c_e. Για ένα ταίριασμα M, λέμε ότι το κόστος του ταιριάσματος είναι το συνολικό κόστος όλων των ακμών του M, δηλαδή $cost(M) = \sum_{e \in M} c_e$. Το πρόβλημα του *Τέλειου Ταιριάσματος Ελάχιστου Κόστους* (Minimum-Cost Perfect Matching) υποθέτει ότι $|X| = |Y| = n$, και ο στόχος είναι να βρούμε ένα τέλειο ταίριασμα με ελάχιστο κόστος.

Σχεδιασμός και ανάλυση του αλγορίθμου

Θα περιγράψουμε τώρα έναν αποδοτικό αλγόριθμο για την επίλυση αυτού του προβλήματος, ο οποίος βασίζεται στην ιδέα των διαδρομών επαύξησης αλλά είναι προσαρμοσμένος έτσι ώστε να συνυπολογίζει και το κόστος. Έτσι, ο αλγόριθμος κατασκευάζει επαναληπτικά ταιριάσματα με χρήση i ακμών, για κάθε τιμή του i από 1 έως n. Θα δείξουμε ότι όταν ο αλγόριθμος καταλήξει σε ένα ταίριασμα με μέγεθος n, αυτό είναι ένα τέλειο ταίριασμα με ελάχιστο κόστος. Η δομή υψηλού επιπέδου του αλγορίθμου είναι αρκετά απλή. Αν έχουμε ένα ταίριασμα ελάχιστου κόστους με μέγεθος i, αναζητούμε μια διαδρομή επαύξησης για να παραγάγουμε ένα ταίριασμα με μέγεθος $i + 1$· και αντί να ψάξουμε απλώς για μια οποιαδήποτε διαδρομή επαύξησης (όπως ήταν επαρκές στην περίπτωση χωρίς το κόστος), χρησιμοποιούμε τη φθηνότερη διαδρομή επαύξησης έτσι ώστε το μεγαλύτερο ταίριασμα να έχει επίσης ελάχιστο κόστος.

Θυμηθείτε τη δομή του υπολοιπόμενου γραφήματος, το οποίο χρησιμοποιήσαμε για την εύρεση των διαδρομών επαύξησης. Έστω ότι το M είναι ένα ταίριασμα. Προσθέτουμε δύο νέους κόμβους s και t στο γράφημα. Προσθέτουμε ακμές (s, x) για όλους τους κόμβους $x \in X$ που δεν είναι ταιριασμένοι, καθώς και ακμές (y, t) για όλους τους κόμβους $y \in Y$ που δεν είναι ταιριασμένοι. Η ακμή $e = (x, y) \in E$ είναι προσανατολισμένη από το x προς το y εάν η e δεν ανήκει στο ταίριασμα M, ή από το y προς το x εάν $e \in M$. Θα χρησιμοποιούμε το G_M για να συμβολίσουμε το υπολοιπόμενο γράφημα. Σημειώστε ότι όλες οι ακμές που πηγαίνουν από το Y προς το X ανήκουν στο ταίριασμα M, ενώ οι ακμές που πηγαίνουν από το X προς το Y δεν ανήκουν. Οποιαδήποτε κατευθυνόμενη διαδρομή s-t P του γραφήματος G_M αντιστοιχεί σε ένα ταίριασμα κατά ένα μεγαλύτερο από το M με εναλλαγή των ακμών κατά μήκος της διαδρομής P· δηλαδή, οι ακμές της P από το X προς το Y προστίθενται στο M και όλες οι ακμές της διαδρομής P που πηγαίνουν από το Y προς το X διαγράφονται από το M. Όπως και προηγουμένως, θα αποκαλούμε τη διαδρομή P στο G_M ως *διαδρομή επαύξησης* (augmenting path), και θα λέμε ότι *επαυξάνουμε* το ταίριασμα M χρησιμοποιώντας τη διαδρομή P.

Θα θέλαμε τώρα το ταίριασμα που προκύπτει να έχει όσο το δυνατόν μικρότερο κόστος. Για να το επιτύχουμε αυτό, θα αναζητούμε μια φθηνή διαδρομή επαύξησης αναφορικά με τα ακόλουθα φυσικά κόστη. Οι ακμές που φεύγουν από το s ή που εισέρχονται στο t θα έχουν κόστος 0· μια ακμή e που πηγαίνει από το X προς το Y θα έχει κόστος c_e (επειδή η συμπερίληψη αυτής της ακμής στη διαδρομή σημαίνει ότι προσθέτουμε την ακμή στο M)· ενώ μια ακμή e που πηγαίνει από το Y προς το X θα έχει κόστος $-c_e$ (επειδή η συμπερίληψη αυτής της ακμής στη διαδρομή σημαίνει ότι διαγράφουμε την ακμή από το M). Θα χρησιμοποιούμε το $cost(P)$ για να συμβολίσουμε το κόστος μιας διαδρομής P στο G_M. Η ακόλουθη πρόταση συνοψίζει αυτή την κατασκευή.

(7.61) *Έστω ότι M είναι ένα ταίριασμα και P είναι μια διαδρομή στο G_M από το s έως το t. Έστω ότι M' είναι το ταίριασμα που παίρνουμε από το M μέσω της επαύξησης με τη διαδρομή P. Τότε $|M'| = |M| + 1$ και $cost(M') = cost(M) + cost(P)$.*

Με δεδομένη αυτή την πρόταση, μπορούμε με πολύ φυσικό τρόπο να προτείνουμε έναν αλγόριθμο για την εύρεση ενός ταιριάσματος με ελάχιστο κόστος. Βρίσκουμε επαναληπτικά διαδρομές ελάχιστου κόστους στο G_M, και χρησιμοποιούμε τις διαδρομές για να επαυξήσουμε τα ταιριάσματα. Πώς όμως θα μπορούμε να είμαστε σίγουροι ότι το τέλειο ταίριασμα που βρίσκουμε έχει ελάχιστο κόστος; Ή, ακόμα χειρότερα, έχει καν αυτός ο αλγόριθμος νόημα; Μπορούμε να βρούμε διαδρομές ελάχιστου κόστους μόνο αν γνωρίζουμε ότι το γράφημα G_M δεν έχει αρνητικούς κύκλους.

Ανάλυση αρνητικών κύκλων. Στην πραγματικότητα, η κατανόηση του ρόλου των αρνητικών κύκλων στο γράφημα G_M είναι το κλειδί για την ανάλυση του αλγορίθμου. Εξετάστε καταρχήν την περίπτωση όπου το M είναι ένα τέλειο ταίριασμα. Παρατηρήστε ότι στην περίπτωση αυτή στο γράφημα G_M ο κόμβος s δεν έχει καθόλου εξερχόμενες ακμές και ο κόμβος t δεν έχει καθόλου εισερχόμενες ακμές (επειδή το ταίριασμα είναι τέλειο), και έτσι κανένας κύκλος στο G_M δεν περιέχει το s ή το t.

(7.62) *Έστω ότι το M είναι ένα τέλειο ταίριασμα. Αν υπάρχει ένας κατευθυνόμενος κύκλος C αρνητικού κόστους στο G_M, τότε το M δεν είναι ταίριασμα ελάχιστου κόστους.*

Απόδειξη. Για να το δείξουμε αυτό, θα χρησιμοποιήσουμε τον κύκλο C για επαύξηση, με τον ίδιο τρόπο με τον οποίο χρησιμοποιήσαμε κατευθυνόμενες διαδρομές για να πάρουμε μεγαλύτερα ταιριάσματα. Η επαύξηση του M με τον κύκλο C περιλαμβάνει την εναλλαγή των ακμών κατά μήκος του C στο ταίριασμα M. Το νέο τέλειο ταίριασμα M' που προκύπτει έχει $cost(M') = cost(M) + cost(C)$· όμως $cost(C) < 0$, και έτσι το M δεν είναι ταίριασμα ελάχιστου κόστους. ■

Ακόμα πιο σημαντικό είναι ότι είναι αληθές και το αντίστροφο αυτής της πρότασης· έτσι στην πραγματικότητα ένα τέλειο ταίριασμα M έχει ελάχιστο κόστος ακριβώς όταν δεν υπάρχουν αρνητικοί κύκλοι στο γράφημα G_M.

(7.63) *Έστω ότι το M είναι ένα τέλειο ταίριασμα. Αν δεν υπάρχουν κατευθυνόμενοι κύκλοι C αρνητικού κόστους στο G_M, τότε το M είναι τέλειο ταίριασμα ελάχιστου κόστους.*

Απόδειξη. Υποθέστε ότι η πρόταση αυτή δεν είναι αληθής, και έστω ότι το M' είναι ένα τέλειο ταίριασμα με μικρότερο κόστος. Θεωρήστε το σύνολο των ακμών που βρίσκονται στο ένα από τα M και M' αλλά όχι και στα δύο. Προσέξτε ότι αυτό το σύνολο ακμών αντιστοιχεί σε ένα σύνολο ασύνδετων ως προς τους κόμβους κατευθυνόμενων κύκλων στο G_M. Το κόστος του συνόλου των κατευθυνόμενων κύκλων είναι ακριβώς $cost(M') - cost(M)$. Αν υποθέσουμε ότι το M' έχει μικρότερο κόστος από το M, τότε τουλάχιστον ένας από αυτούς τους κύκλους έχει αρνητικό κόστος. ■

Το σχέδιό μας λοιπόν είναι να κάνουμε επαναλήψεις σε ταιριάσματα με όλο και μεγαλύτερο μέγεθος, διατηρώντας σε κάθε επανάληψη την ιδιότητα ότι το γράφημα

G_M δεν έχει αρνητικούς κύκλους. Με αυτόν τον τρόπο ο υπολογισμός μας για τη διαδρομή ελάχιστου κόστους θα είναι πάντα καλά ορισμένος, και όταν ολοκληρώσουμε τις επαναλήψεις με ένα τέλειο ταίριασμα θα μπορούμε να χρησιμοποιήσουμε την (7.63) για να συμπεράνουμε ότι αυτό έχει ελάχιστο κόστος.

Διατήρηση τιμών στους κόμβους. Είναι χρήσιμο να σκεφτόμαστε για μια αριθμητική *τιμή* $p(v)$ που σχετίζεται με κάθε κόμβο v. Αυτές οι τιμές και θα μας βοηθήσουν να κατανοήσουμε πώς λειτουργεί ο αλγόριθμος, αλλά και θα επιταχύνουν επίσης την υλοποίηση. Ένα ζήτημα το οποίο πρέπει να αντιμετωπίσουμε είναι η διατήρηση της ιδιότητας ότι το γράφημα G_M δεν έχει αρνητικούς κύκλους σε καμία επανάληψη. Πώς θα γνωρίζουμε μετά από μία επαύξηση ότι το νέο υπολοιπόμενο γράφημα εξακολουθεί να μην έχει αρνητικούς κύκλους; Οι τιμές θα μας χρησιμεύσουν στο να δώσουμε μια συμπαγή απόδειξη για αυτό το θέμα.

Για να κατανοήσετε την έννοια των τιμών, είναι χρήσιμο να έχετε στο μυαλό σας μια οικονομική ερμηνεία τους. Υποθέστε ότι το σύνολο X αντιπροσωπεύει ανθρώπους στους οποίους πρέπει να ανατεθεί ένα σύνολο εργασιών Y. Για μια ακμή $e = (x, y)$, το κόστος c_e είναι το κόστος που σχετίζεται με την ανάθεση της εργασίας y στο άτομο x. Μπορούμε τώρα να φανταστούμε την τιμή $p(x)$ ως ένα πρόσθετο μπόνους που πληρώνουμε έτσι ώστε το άτομο x να συμμετάσχει σε αυτό το σύστημα, κάτι σαν "μπόνους εγγραφής". Με αυτή τη συλλογιστική, το κόστος για την ανάθεση της εργασίας y στο άτομο x γίνεται $p(x) + c_e$. Από την άλλη πλευρά, φανταζόμαστε την τιμή $p(y)$ για τους κόμβους $y \in Y$ ως επιβράβευση, ή προστιθέμενη αξία που κερδίζεται από την εκτέλεση της εργασίας y (ανεξάρτητα από το ποιο άτομο από το X την πραγματοποιεί). Έτσι το "καθαρό κόστος" για την ανάθεση της εργασίας y στο άτομο x γίνεται $p(x) + c_e - p(y)$: αυτό είναι το κόστος για την πρόσληψη του x με μπόνους $p(x)$, την ανάθεση σε αυτόν της εργασίας y για κόστος c_e, και μετά την είσπραξη της ανταμοιβής $p(y)$. Θα το αποκαλούμε αυτό *ανηγμένο κόστος* μιας ακμής $e = (x, y)$ και θα το συμβολίζουμε με $c_e^p = p(x) + c_e - p(y)$. Παρόλα αυτά, είναι σημαντικό να θυμάστε ότι μόνο τα κόστη c_e αποτελούν τμήμα της περιγραφής του προβλήματος· οι τιμές (μπόνους και ανταμοιβές) είναι ένας τρόπος για να κατανοήσουμε τη λύση.

Ειδικότερα, λέμε ότι ένα σύνολο αριθμών $\{p(v) : v \in V\}$ σχηματίζει ένα σύνολο *συμβατών τιμών* αναφορικά με ένα ταίριασμα M εάν

(i) για όλους τους μη ταιριασμένους κόμβους $x \in X$ έχουμε $p(x) = 0$ (δηλαδή, οι άνθρωποι που δεν τους ζητείται να κάνουν κάποια δουλειά δεν πληρώνονται)·

(ii) για όλες τις ακμές $e = (x, y)$ έχουμε $p(x) + c_e \geq p(y)$ (δηλαδή, όλες οι ακμές έχουν μη αρνητικό ανηγμένο κόστος)· και

(iii) για όλες τις ακμές $e = (x, y) \in M$ έχουμε $p(x) + c_e = p(y)$ (όλες οι ακμές που χρησιμοποιούνται στην ανάθεση εργασιών έχουν μηδενικό ανηγμένο κόστος).

Γιατί είναι χρήσιμες αυτές οι τιμές; Διαισθητικά, οι συμβατές τιμές μάς δείχνουν ότι το ταίριασμα είναι φθηνό: Κατά μήκος των ταιριασμένων ακμών η ανταμοιβή είναι ίση με το κόστος, ενώ σε όλες τις άλλες ακμές η ανταμοιβή δεν είναι μεγαλύτερη από το κόστος. Για ένα μερικό ταίριασμα αυτό μπορεί να μη σημαίνει ότι το ταίριασμα έχει το μικρότερο δυνατό κόστος για το αντίστοιχο μέγεθος ταιριάσματος (αφού μπορεί να καλύπτουμε τις δαπανηρές εργασίες). Ισχυριζόμαστε όμως ότι, εάν το M είναι ένα ταίριασμα για το οποίο υπάρχει ένα σύνολο με συμβατές τιμές, τότε το G_M δεν έχει αρνητικούς κύκλους. Για ένα τέλειο ταίριασμα M, αυτό με βάση την (7.63) θα σημαίνει ότι το M έχει ελάχιστο κόστος.

Για να δούμε γιατί το G_M δεν μπορεί να έχει αρνητικούς κύκλους, θα επεκτείνουμε τον ορισμό του ανηγμένου κόστους στο υπολοιπόμενο γράφημα χρησιμοποιώντας την ίδια παράσταση $c_e^p = p(v) + c_e - p(w)$ για οποιαδήποτε ακμή $e = (v, w)$. Παρατηρήστε πως ο ορισμός των συμβατών τιμών υπονοεί ότι όλες οι ακμές στο υπολοιπόμενο γράφημα G_M έχουν μη αρνητικά ανηγμένα κόστη. Τώρα, προσέξτε ότι για οποιονδήποτε κύκλο C έχουμε

$$cost(C) = \sum_{e \in C} c_e = \sum_{e \in C} c_e^p ,$$

αφού όλοι οι όροι στη δεξιά πλευρά που αντιστοιχούν σε τιμές αλληλοαναιρούνται. Γνωρίζουμε ότι ο κάθε όρος της δεξιάς πλευράς είναι μη αρνητικός, και έτσι είναι προφανές ότι το $cost(C)$ είναι μη αρνητικό.

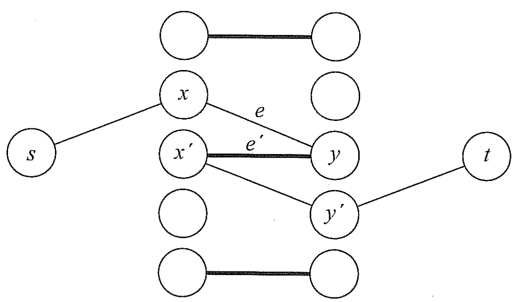
Υπάρχει και ένας δεύτερος, αλγοριθμικός λόγος για τον οποίο είναι χρήσιμο να έχουμε τιμές στους κόμβους. Όταν έχουμε ένα γράφημα με ακμές αρνητικού κόστους αλλά χωρίς αρνητικούς κύκλους, μπορούμε να υπολογίσουμε τις συντομότερες διαδρομές με τον αλγόριθμο Bellman-Ford σε χρόνο $O(mn)$. Εάν όμως στην πραγματικότητα το γράφημά μας δεν έχει ακμές αρνητικού κόστους, τότε μπορούμε να χρησιμοποιήσουμε αντί γι' αυτόν τον αλγόριθμο του Dijkstra, ο οποίος απαιτεί μόνο χρόνο $O(m \log n)$ — δηλαδή είναι ταχύτερος σχεδόν κατά έναν ολόκληρο συντελεστή n.

Στην περίπτωσή μας, η διατήρηση των τιμών μάς επιτρέπει να υπολογίσουμε τις συντομότερες διαδρομές ως προς το μη αρνητικό ανηγμένο κόστος c_e^p, φθάνοντας σε μια ισοδύναμη απάντηση. Πράγματι, υποθέστε ότι χρησιμοποιούμε τον αλγόριθμο του Dijkstra για να βρούμε το ελάχιστο κόστος $d_{p,M}(v)$ για μια κατευθυνόμενη διαδρομή από το s προς οποιονδήποτε κόμβο $v \in X \cup Y$ με βάση το κόστος c_e^p. Με δεδομένα τα ελάχιστα κόστη $d_{p,M}(y)$ για ένα μη ταιριασμένο κόμβο $y \in Y$, το (μη ανηγμένο) κόστος της διαδρομής από το s μέχρι το t μέσω του y είναι $d_{p,M}(y) + p(y)$, και έτσι βρίσκουμε το ελάχιστο κόστος σε $O(n)$ πρόσθετο χρόνο. Συνοψίζοντας, έχουμε το ακόλουθο γεγονός.

(7.64) *Έστω ότι το M είναι ένα ταίριασμα και p είναι συμβατές τιμές. Μπορούμε να χρησιμοποιήσουμε μια εκτέλεση του αλγορίθμου του Dijkstra ώστε σε* O(n) *πρόσθετο χρόνο να βρούμε τη διαδρομή ελάχιστου κόστους από το s μέχρι το t.*

Ενημέρωση των τιμών των κόμβων Εκμεταλλευτήκαμε τις τιμές για να βελτιώσουμε μία επανάληψη του αλγορίθμου. Για να μπορέσουμε να είμαστε έτοιμοι για την επόμενη επανάληψη, χρειαζόμαστε όχι μόνο τη διαδρομή ελάχιστου κόστους (για να πάρουμε το επόμενο ταίριασμα), αλλά επίσης και έναν τρόπο για να παράγουμε ένα σύνολο συμβατών τιμών ως προς το νέο ταίριασμα.

Για να πάρετε μια ιδέα για τον τρόπο με τον οποίο μπορούμε να το επιτύχουμε αυτό, θεωρήστε έναν μη ταιριασμένο κόμβο x ως προς ένα ταίριασμα M, και μια ακμή $e = (x, y)$, όπως φαίνεται στην Εικόνα 7.22. Αν το νέο ταίριασμα M' περιλαμβάνει την ακμή e (με άλλα λόγια, αν η e βρίσκεται στη διαδρομή επαύξησης που χρησιμοποιούμε για να ενημερώσουμε το ταίριασμα), τότε θα θέλουμε να αφήσουμε σε μηδενική τιμή το ανηγμένο κόστος αυτής της ακμής. Παρόλα αυτά, οι τιμές p που χρησιμοποιήσαμε με το ταίριασμα M μπορεί να προκαλούν ένα ανηγμένο κόστος $c_e^p > 0$ — δηλαδή, η αντιστοίχιση του ατόμου x στην εργασία y, στην οικονομική ερμηνεία μας, μπορεί να μη θεωρείται αρκετά φθηνή. Μπορούμε να καταλήξουμε στο μηδενικό ανηγμένο κόστος είτε με αύξηση της τιμής $p(y)$ (της ανταμοιβής του y) κατά c_e^p, ή με μείωση της τιμής $p(x)$ κατά την ίδια ποσότητα. Για να διατηρηθούν οι τιμές μη αρνητικές, θα αυξήσουμε την τιμή $p(y)$. Όμως ο κόμβος y μπορεί να ταιριάζεται στο ταίριασμα M με κάποιον άλλον κόμβο x' μέσω μιας ακμής $e'=(x', y)$, όπως φαίνεται στην Εικόνα 7.22. Η αύξηση της ανταμοιβής $p(y)$ μειώνει σε αρνητικό το ανηγμένο κόστος της ακμής e',



Εικόνα 7.22 Ένα ταίριασμα M (οι έντονες ακμές), και ένα υπολοιπόμενο γράφημα που χρησιμοποιείται για αύξηση του μεγέθους του ταιριάσματος.

και έτσι οι τιμές δεν είναι πλέον συμβατές. Για να διατηρήσουμε τη συμβατότητα, μπορούμε να αυξήσουμε κατά την ίδια ποσότητα το $p(x')$. Όμως αυτή η αλλαγή μπορεί να προκαλέσει προβλήματα σε άλλες ακμές. Μπορούμε άραγε να ενημερώσουμε όλες τις τιμές και να διατηρήσουμε το ταίριασμα και τις τιμές συμβατές σε όλες τις ακμές; Προκαλεί έκπληξη το ότι αυτό μπορούμε να το επιτύχουμε με αρκετά απλό τρόπο χρησιμοποιώντας τις αποστάσεις από το s προς όλους τους άλλους κόμβους, τις οποίες υπολόγισε ο αλγόριθμος του Dijkstra.

(7.65) *Έστω ότι M είναι ένα ταίριασμα, p είναι συμβατές τιμές, και M' είναι ένα ταίριασμα το οποίο παίρνουμε με επαύξηση κατά μήκος της διαδρομής ελάχιστου κόστους από το s μέχρι το t. Τότε το $p'(v) = d_{p,M}(v) + p(v)$ είναι ένα συμβατό σύνολο τιμών για το M'.*

Απόδειξη. Για να αποδείξουμε τη συμβατότητα, θεωρήστε καταρχήν μια ακμή $e = (x', y) \in M$. Η μόνη ακμή που εισέρχεται στο x' είναι η κατευθυνόμενη ακμή (y, x'), και έτσι $d_{p,M}(x') = d_{p,M}(y) - c_e^p$, όπου $c_e^p = p(y) + c_e - p(x')$, οπότε παίρνουμε την επιθυμητή εξίσωση για αυτές τις ακμές. Ας εξετάσουμε τώρα τις ακμές (x, y) στο $M' - M$. Αυτές οι ακμές βρίσκονται στη διαδρομή ελάχιστου κόστους από το s προς το t, και έτσι ικανοποιούν τη σχέση $d_{p,M}(y) = d_{p,M}(x) + c_e^p$ όπως είναι το επιθυμητό. Τέλος, βρίσκουμε την απαιτούμενη ανισότητα για όλες τις υπόλοιπες ακμές, αφού όλες οι ακμές $e = (x, y) \notin M$ θα πρέπει να ικανοποιούν τη σχέση $d_{p,M}(y) \leq d_{p,M}(x) + c_e^p$. ■

Τέλος, θα πρέπει να εξετάσουμε τον τρόπο με τον οποίο θα αρχικοποιήσουμε τον αλγόριθμο, έτσι ώστε να τον βάλουμε στο σωστό δρόμο. Ορίζουμε ως αρχική τιμή για το M το κενό σύνολο, ορίζουμε $p(x) = 0$ για όλα τα $x \in X$, και ορίζουμε το $p(y)$, για $y \in Y$, να είναι το ελάχιστο κόστος μιας ακμής που εισέρχεται στο y. Σημειώστε ότι οι τιμές αυτές είναι συμβατές ως προς το $M = \varnothing$.

Η ακόλουθη είναι μια σύνοψη του αλγορίθμου

```
Ξεκίνα με M ίσο με το κενό σύνολο
Όρισε p(x) = 0 για x ∈ X, και p(y) =      min       c_e για y ∈ Y
                                      e προς το y
While M δεν είναι τέλειο ταίριασμα
   Βρες μια διαδρομή s-t ελάχιστου κόστους P στο G_M με βάση την (7.64)
      και με τιμές p
   Επαύξησε κατά μήκος της P για να πάρουμε ένα νέο ταίριασμα M'
   Βρες ένα σύνολο συμβατών τιμών ως προς την M' μέσω της (7.65)
Endwhile
```

Το τελικό σύνολο συμβατών τιμών παρέχει απόδειξη ότι το G_M δεν έχει αρνητικούς κύκλους· και η (7.63) μας δείχνει ότι αυτό σημαίνει πως το ταίριασμα M έχει ελάχιστο κόστος.

(7.66) *Μπορούμε να βρούμε το τέλειο ταίριασμα ελάχιστου κόστους δαπανώντας το χρόνο που απαιτείται για n υπολογισμούς συντομότερης διαδρομής με μη αρνητικά μήκη ακμών.*

Επεκτάσεις: Μια οικονομική ερμηνεία των τιμών

Για να ολοκληρώσουμε την ανάλυση για το πρόβλημα του Τέλειου Ταιριάσματος Ελάχιστου Κόστους, θα αναπτύξουμε λίγο περισσότερο την οικονομική ερμηνεία των τιμών. Θεωρήστε το ακόλουθο σενάριο. Υποθέστε ότι το X είναι ένα σύνολο n ανθρώπων που ψάχνουν όλοι να αγοράσουν σπίτι, και Y είναι ένα σύνολο n σπιτιών τα οποία είναι υποψήφια για αγορά από όλους αυτούς τους ανθρώπους. Έστω ότι το $v(x, y)$ συμβολίζει την αξία του σπιτιού y για τον αγοραστή x. Αφού ο καθένας από τους αγοραστές θέλει ένα από τα σπίτια, θα μπορούσε κανείς να ισχυριστεί ότι η καλύτερη διευθέτηση είναι να βρούμε ένα τέλειο ταίριασμα M το οποίο να μεγιστοποιεί το $\sum_{(x,y) \in M} v(x, y)$. Μπορούμε να βρούμε ένα τέτοιο τέλειο ταίριασμα χρησιμοποιώντας τον αλγόριθμο τέλειου ταιριάσματος ελάχιστου κόστους με κόστη $c_e = -v(x,y)$ αν $e = (x,y)$.

Το ερώτημα τώρα είναι το εξής: Μπορούμε να πείσουμε αυτούς τους αγοραστές να αγοράσουν το σπίτι που τους αντιστοιχίζεται; Από την πλευρά του, ο κάθε αγοραστής x θα ήθελε να αγοράσει το σπίτι y που έχει τη μέγιστη αξία $v(x, y)$ για αυτόν. Πώς θα μπορούσαμε να τον πείσουμε να αγοράσει αντί γι' αυτό, το σπίτι που του αντιστοίχισε το ταίριασμα M το οποίο βρήκαμε; Θα χρησιμοποιήσουμε τις τιμές για να αλλάξουμε τα κίνητρα των αγοραστών. Υποθέστε ότι ορίζουμε μια τιμή $P(y)$ για κάθε σπίτι y, δηλαδή το άτομο που θα αγοράσει το σπίτι y θα πρέπει να πληρώσει $P(y)$. Με βάση αυτές τις τιμές, ο αγοραστής θα ενδιαφέρεται να αγοράσει το σπίτι με τη μέγιστη καθαρή αξία, δηλαδή το σπίτι που μεγιστοποιεί τη σχέση $v(x, y) - P(y)$. Λέμε ότι ένα τέλειο ταίριασμα M και οι τιμές σπιτιών P είναι *σε ισορροπία* αν, για όλες τις ακμές $(x, y) \in M$ και όλα τα άλλα σπίτια y', έχουμε

$$v(x, y) - P(y) \geq v(x, y') - P(y').$$

Μπορούμε όμως να βρούμε ένα τέλειο ταίριασμα και ένα σύνολο τιμών έτσι ώστε να επιτύχουμε αυτή την κατάσταση, όπου όλοι οι αγοραστές θα είναι ευχαριστημένοι; Στην πραγματικότητα, το τέλειο ταίριασμα ελάχιστου κόστους και το συσχετισμένο σύνολο συμβατών τιμών μάς παρέχουν ακριβώς αυτό το οποίο ψάχνουμε.

(7.67) *Έστω ότι M είναι ένα τέλειο ταίριασμα ελάχιστου κόστους, όπου $c_e = -v(x, y)$ για όλες τις ακμές $e = (x, y)$, και έστω ότι p είναι ένα συμβατό σύνολο τιμών. Τότε το ταίριασμα M και το σύνολο τιμών $\{P(y) = -p(y) : y \in Y\}$ είναι σε ισορροπία.*

Απόδειξη. Θεωρήστε μια ακμή $e = (x, y) \in M$, και έστω $e' = (x, y')$. Αφού το M και το p είναι συμβατά, έχουμε $p(x) + c_e = p(y)$ και $p(x) + c_e \geq p(y')$. Αφαιρώντας αυτές τις δύο ανισότητες για να εξαλείψουμε το $p(x)$, και αντικαθιστώντας τις τιμές των p και c, βρίσκουμε τη ζητούμενη ανισότητα από τον ορισμό της ισορροπίας. ■

Λυμένες ασκήσεις

Λυμένη άσκηση 1

Υποθέστε ότι σας δίνεται ένα κατευθυνόμενο γράφημα $G = (V, E)$, με μια θετική ακέραια χωρητικότητα c_e σε κάθε ακμή e, μια καθορισμένη προέλευση $s \in V$, και μια καθορισμένη απόληξη $t \in V$. Σας δίνεται επίσης μια ακέραια ροή s-t στο G, η οποία ορίζεται με μια τιμή ροής f_e σε κάθε ακμή e.

Υποθέστε τώρα ότι διαλέγουμε μια συγκεκριμένη ακμή $e \in E$ και αυξάνουμε τη χωρητικότητά της κατά μία μονάδα. Δείξτε με ποιον τρόπο μπορούμε να βρούμε μια μέγιστη ροή για το γράφημα χωρητικοτήτων που προκύπτει σε χρόνο $O(m + n)$, όπου m είναι το πλήθος ακμών του G και n είναι το πλήθος των κόμβων.

Λύση Το πρόβλημα είναι ότι ο $O(m + n)$ δεν είναι αρκετός χρόνος για να υπολογίσουμε μια νέα μέγιστη ροή από το μηδέν, έτσι πρέπει να βρούμε πώς μπορούμε να χρησιμοποιήσουμε τη ροή f που μας δίνεται. Διαισθητικά, ακόμα και μετά την προσθήκη μίας μονάδας στη χωρητικότητα της ακμής e, η ροή f δεν μπορεί να απέχει πολύ από τη μέγιστη· άλλωστε, το δίκτυο δεν έχει αλλάξει πάρα πολύ.

Στην πραγματικότητα, δεν είναι δύσκολο να δείξουμε ότι η μέγιστη τιμή ροής μπορεί να μεγαλώσει το πολύ κατά 1.

(7.68) *Θεωρήστε το δίκτυο ροής G' που προκύπτει από την προσθήκη μίας μονάδας στη χωρητικότητα της ακμής e. Η τιμή της μέγιστης ροής στο G' είναι είτε $v(f)$ είτε $v(f) + 1$.*

Απόδειξη. Η τιμή της μέγιστης ροής στο G' είναι τουλάχιστον ίση με $\nu(f)$, αφού η f εξακολουθεί να είναι μια εφικτή ροή σε αυτό το δίκτυο. Η ροή αυτή έχει επίσης ακέραιες τιμές. Έτσι είναι αρκετό να δείξουμε ότι η τιμή της μέγιστης ροής στο G' είναι το πολύ ίση με $\nu(f) + 1$.

Από το θεώρημα Μέγιστης Ροής και Ελάχιστης Αποκοπής, υπάρχει στο αρχικό δίκτυο ροής G κάποια αποκοπή s-t (A, B) με χωρητικότητα $\nu(f)$. Εξετάζουμε τώρα ποια είναι η χωρητικότητα της (A, B) στο νέο δίκτυο ροής G'. Όλες οι ακμές που διέσχιζαν την αποκοπή (A, B) έχουν στο G' ίδια χωρητικότητα όπως και στο G, με την πιθανή εξαίρεση της ακμής e — στην περίπτωση που η e διασχίζει την αποκοπή (A, B). Όμως η c_e αυξήθηκε μόνο κατά 1, και έτσι η χωρητικότητα της αποκοπής (A, B) στο νέο δίκτυο ροής G' είναι το πολύ ίση με $\nu(f) + 1$. ∎

Η (7.68) μας προτείνει ένα φυσικό αλγόριθμο. Ξεκινώντας από την εφικτή ροή f στο G', προσπαθούμε να βρούμε στο υπολοιπόμενο γράφημα G'_f μία διαδρομή επαύξησης από το s στο t. Αυτό απαιτεί χρόνο $O(m + n)$. Τώρα θα συμβεί ένα από τα εξής δύο πράγματα. Η μία περίπτωση είναι να μην καταφέρουμε να βρούμε μια διαδρομή επαύξησης, και στην περίπτωση αυτή θα γνωρίζουμε ότι η f είναι μια μέγιστη ροή. Διαφορετικά η επαύξηση θα είναι επιτυχής, παράγοντας μια ροή f' με τιμή το πολύ ίση με $\nu(f) + 1$. Στην περίπτωση αυτή γνωρίζουμε από την (7.68) ότι η f' πρέπει να είναι μέγιστη ροή. Έτσι, και στις δύο περιπτώσεις, βρίσκουμε μια μέγιστη ροή μετά από ένα μόνο υπολογισμό διαδρομής επαύξησης.

Λυμένη άσκηση 2

Βοηθάτε την ιατρική συμβουλευτική οργάνωση Γιατροί Χωρίς Σαββατοκύριακα να οργανώσει τα προγράμματα των γιατρών σε ένα μεγάλο νοσοκομείο. Έχουν ήδη σχεδόν ετοιμάσει τα προγράμματα για τις κανονικές εργάσιμες ημέρες, και θέλουν τώρα να αντιμετωπίσουν όλες τις ειδικές περιπτώσεις, και ειδικότερα να εξασφαλίσουν ότι θα υπάρχει τουλάχιστον ένας γιατρός ο οποίος θα καλύπτει τις ημέρες των αργιών.

Να πώς γίνεται αυτό. Υπάρχουν k περίοδοι αργιών (π.χ., η εβδομάδα των Χριστουγέννων, η Μεγάλη Εβδομάδα, κ.λπ.), η καθεμία από τις οποίες εκτείνεται σε αρκετές συνεχόμενες ημέρες. Έστω ότι D_j είναι το σύνολο των ημερών που συμπεριλαμβάνονται στην περίοδο αργίας j· θα αναφερόμαστε στην ένωση όλων αυτών των ημερών, $\cup_j D_j$, ως σύνολο όλων των *ημερών αργίας*.

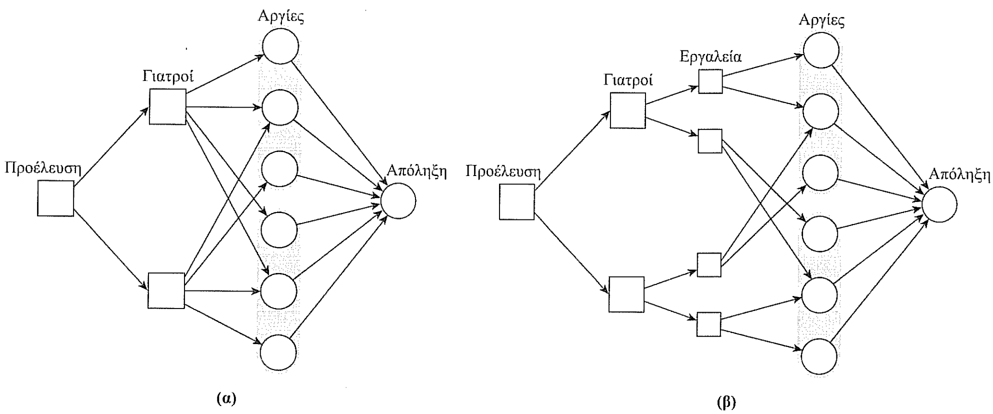
Υπάρχουν n γιατροί στο νοσοκομείο, και ο γιατρός i έχει ένα σύνολο ημερών αργίας S_i κατά τις οποίες ο γιατρός είναι διαθέσιμος για εργασία. (Αυτό το σύνολο μπορεί να περιλαμβάνει κάποιες ημέρες από μια δεδομένη περίοδο αργίας αλλά όχι τις υπόλοιπες· έτσι, για παράδειγμα, ένας γιατρός μπορεί να είναι διαθέσιμος για εργασία τη Μεγάλη Παρασκευή, το Μεγάλο Σάββατο, και την Κυριακή του Πάσχα αλλά όχι τη Μεγάλη Πέμπτη.)

Κατασκευάστε έναν αλγόριθμο πολυωνυμικού χρόνου ο οποίος θα παίρνει αυτές τις πληροφορίες και θα αποφασίζει αν είναι δυνατή η επιλογή ενός γιατρού για εργασία σε κάθε ημέρα αργίας με βάση τους ακόλουθους περιορισμούς.

- Για μια δεδομένη παράμετρο c, ο κάθε γιατρός θα πρέπει να αντιστοιχίζεται για εργασία σε το πολύ c ημέρες αργίας, και μόνο σε ημέρες κατά τις οποίες είναι διαθέσιμος.
- Για κάθε περίοδο αργίας j, ο κάθε γιατρός θα πρέπει να αντιστοιχίζεται για εργασία σε το πολύ μία ημέρα του συνόλου D_j. (Με άλλα λόγια, παρά το ότι μπορεί κάποιος γιατρός να είναι διαθέσιμος για εργασία σε πολλές ημέρες αργίας σε ολόκληρο το χρόνο, δεν θα πρέπει να επιλεγεί για εργασία σε δύο ημέρες της εβδομάδας των Χριστουγέννων, ή της εβδομάδας του Πάσχα, κ.ο.κ.)

Ο αλγόριθμος θα πρέπει είτε να επιστρέφει μια αντιστοίχιση των γιατρών που να ικανοποιεί αυτούς τους περιορισμούς, είτε να αναφέρει (σωστά) ότι δεν υπάρχει τέτοια αντιστοίχιση.

Λύση Αυτή είναι μια πολύ φυσική περίπτωση για την εφαρμογή δικτύου ροής, επειδή σε υψηλό επίπεδο προσπαθούμε να ταιριάξουμε ένα σύνολο (τους γιατρούς) με ένα άλλο σύνολο (τις ημέρες αργίας). Οι περιπλοκές πηγάζουν από την απαίτηση ότι ο κάθε γιατρός μπορεί να δουλέψει σε το πολύ μία ημέρα από κάθε περίοδο αργίας.



Εικόνα 7.23 (α) Οι γιατροί αντιστοιχίζονται σε ημέρες αργίας χωρίς περιορισμό ως προς το σε πόσες ημέρες μιας περιόδου αργίας μπορεί να δουλέψει ο γιατρός. (β) Η ροή δικτύου επεκτείνεται με "μικροεργαλεία" που αποτρέπουν την εργασία ενός γιατρού σε περισσότερες από μία ημέρες της κάθε περιόδου αργίας. Τα σκιασμένα σύνολα αντιστοιχούν στις διαφορετικές περιόδους αργίας.

Για να ξεκινήσουμε, ας δούμε πώς μπορούμε να λύσουμε αυτό το πρόβλημα χωρίς τη συγκεκριμένη απαίτηση, για την απλούστερη περίπτωση όπου ο κάθε γιατρός i έχει ένα σύνολο S_i ημερών στις οποίες μπορεί να δουλέψει, και όπου ο κάθε γιατρός μπορεί να αντιστοιχιστεί συνολικά σε το πολύ c ημέρες. Η δομή αυτή παρουσιάζεται στην Εικόνα 7.23(α). Έχουμε έναν κόμβο u_i που αντιπροσωπεύει τον κάθε γιατρό να συνδέεται με έναν κόμβο v_ℓ που αντιπροσωπεύει την κάθε ημέρα κατά την οποία μπορεί να εργαστεί ο συγκεκριμένος γιατρός· η ακμή αυτή έχει χωρητικότητα 1. Συνδέουμε κάθε κόμβο γιατρού u_i σε μια υπερπροέλευση s με μια ακμή χωρητικότητας c, και συνδέουμε κάθε κόμβο ημέρας v_ℓ σε μια υπεραπόληξη t με μια ακμή η οποία έχει άνω και κάτω όριο 1. Με αυτόν τον τρόπο οι ημέρες ανάθεσης θα μπορούν να "ρέουν" μέσω των γιατρών στις ημέρες στις οποίες μπορούν να δουλέψουν, και τα κάτω όρια στις ακμές από τις ημέρες προς την απόληξη εξασφαλίζουν ότι θα καλυφθούν όλες οι ημέρες. Τέλος, υποθέστε ότι υπάρχουν συνολικά d ημέρες αργίας· θέτουμε μια ζήτηση $+d$ στην απόληξη και $-d$ στην προέλευση, και ψάχνουμε για κάποια εφικτή κυκλοφορία. (Θυμηθείτε ότι, αφού καθορίσαμε κάτω όρια σε κάποιες ακμές, οι αντίστοιχοι αλγόριθμοι αναφέρονται σε κυκλοφορίες με ζήτηση, και όχι σε μέγιστη ροή.)

Θα πρέπει όμως τώρα να αντιμετωπίσουμε αυτόν τον πρόσθετο περιορισμό, ότι ο κάθε γιατρός μπορεί να δουλέψει το πολύ σε μία ημέρα από κάθε περίοδο αργίας. Για να το επιτύχουμε αυτό, παίρνουμε κάθε ζευγάρι (i, j) που αποτελείται από ένα γιατρό i και μία περίοδο αργίας j και προσθέτουμε ένα "μικροεργαλείο αργιών" με τον ακόλουθο τρόπο. Συμπεριλαμβάνουμε ένα νέο κόμβο w_{ij} με εισερχόμενη ακμή χωρητικότητας 1 από τον κόμβο του γιατρού u_i και εξερχόμενες ακμές χωρητικότητας 1 προς κάθε ημέρα της περιόδου αργίας j κατά την οποία ο γιατρός i είναι διαθέσιμος για εργασία. Αυτό το μικροεργαλείο χρησιμεύει στην "περικοπή" της ροής από τον κόμβο u_i στις ημέρες που σχετίζονται με κάθε περίοδο αργίας j, έτσι ώστε να μπορεί συνολικά να πηγαίνει σε αυτές το πολύ μία μονάδα ροής. Η δομή αυτή φαίνεται στην Εικόνα 7.23(β). Όπως και προηγουμένως, θέτουμε ζήτηση $+d$ στην απόληξη και $-d$ στην προέλευση και ψάχνουμε για εφικτή κυκλοφορία. Ο συνολικός χρόνος εκτέλεσης είναι ο χρόνος για την κατασκευή του γραφήματος, που είναι $O(nd)$, συν το χρόνο για έλεγχο ύπαρξης εφικτής κυκλοφορίας σε αυτό το γράφημα.

Η ορθότητα του αλγορίθμου είναι συνέπεια του ακόλουθου ισχυρισμού.

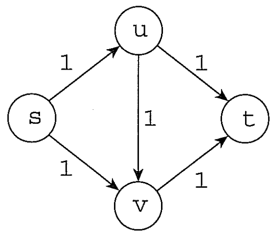
(7.69) *Υπάρχει τρόπος αντιστοίχισης των γιατρών σε ημέρες αργίας έτσι ώστε να ικανοποιούνται όλοι οι περιορισμοί εάν και μόνο εάν υπάρχει μια εφικτή κυκλοφορία στο δίκτυο ροής που έχουμε κατασκευάσει.*

Απόδειξη. Καταρχήν, αν υπάρχει τρόπος αντιστοίχισης των γιατρών στις ημέρες αργίας έτσι ώστε να ικανοποιούνται όλοι οι περιορισμοί, τότε μπορούμε να κατασκευάσουμε την ακόλουθη κυκλοφορία. Αν ο γιατρός i δουλεύει την ημέρα ℓ της περιόδου αργίας j, τότε μπορούμε να στείλουμε μια μονάδα ροής κατά μήκος της διαδρομής s, u_i, w_{ij}, v_ℓ, t· το κάνουμε αυτό για όλα τα ζευγάρια (i, ℓ). Αφού η αντιστοίχιση των γιατρών ικανοποιεί όλους τους περιορισμούς, η κυκλοφορία που προκύπτει θα ικανοποιεί όλες τις συνθήκες χωρητικότητας· και επίσης θα στέλνει d μονάδες ροής από το s στο t, έτσι θα ικανοποιεί τη ζήτηση.

Αντιστρόφως, υποθέστε ότι υπάρχει μια εφικτή κυκλοφορία. Για αυτή την πλευρά της απόδειξης, θα δείξουμε με ποιον τρόπο μπορούμε να χρησιμοποιήσουμε την κυκλοφορία για να κατασκευάσουμε ένα χρονοδιάγραμμα για όλους τους γιατρούς. Με βάση την (7.52), υπάρχει μια εφικτή κυκλοφορία στην οποία όλες οι τιμές ροής είναι ακέραιες. Κατασκευάζουμε τώρα το ακόλουθο χρονοδιάγραμμα: Αν η ακμή (w_{ij}, v_ℓ) μεταφέρει μία μονάδα ροής, τότε βάζουμε το γιατρό i να δουλέψει την ημέρα ℓ. Χάρη στις χωρητικότητες, στο χρονοδιάγραμμα που προκύπτει ο κάθε γιατρός θα μπορεί να δουλέψει το πολύ για c ημέρες, το πολύ για μία ημέρα σε κάθε περίοδο αργίας, και η κάθε ημέρα αργίας θα καλύπτεται από ένα γιατρό. ■

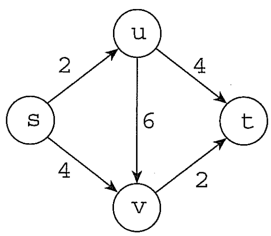
Ασκήσεις

1. **(α)** Φτιάξτε έναν κατάλογο με όλες τις ελάχιστες αποκοπές *s-t* για το δίκτυο ροής της Εικόνας 7.24. Η χωρητικότητα της κάθε ακμής φαίνεται ως ετικέτα δίπλα στην ακμή.



Εικόνα 7.24 Ποιες είναι οι ελάχιστες αποκοπές *s-t* για αυτό το δίκτυο ροής;

(β) Ποια είναι η ελάχιστη χωρητικότητα για μια αποκοπή *s-t* στο δίκτυο ροής της Εικόνας 7.25; Και πάλι, η χωρητικότητα της κάθε ακμής φαίνεται ως ετικέτα δίπλα στην ακμή.

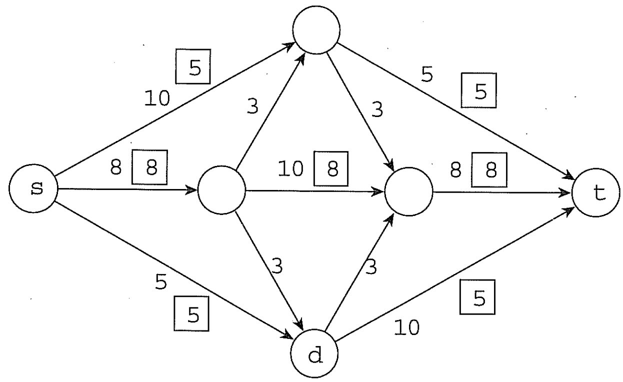


Εικόνα 7.25 Ποια είναι η ελάχιστη χωρητικότητα για μια αποκοπή *s-t* σε αυτό το δίκτυο ροής;

2. Η Εικόνα 7.26 δείχνει ένα δίκτυο ροής στο οποίο έχει υπολογιστεί μια ροή *s-t*. Η χωρητικότητα της κάθε ακμής φαίνεται ως ετικέτα δίπλα στην ακμή, και οι αριθμοί μέσα στα πλαίσια δείχνουν την ποσότητα ροής που στέλνεται σε κάθε ακμή. (Για τις ακμές που δεν έχουν αριθμούς μέσα σε πλαίσια — για παράδειγμα, για τις τέσσερις ακμές με χωρητικότητα 3 — δεν στέλνεται ροή μέσα από αυτές.)

   (α) Ποια είναι η τιμή αυτής της ροής; Είναι αυτή μια μέγιστη ροή (*s*, *t*) για το συγκεκριμένο γράφημα;

   (β) Βρείτε μια ελάχιστη αποκοπή *s-t* για το δίκτυο ροής της Εικόνας 7.26, και πείτε ποια είναι η χωρητικότητά της.

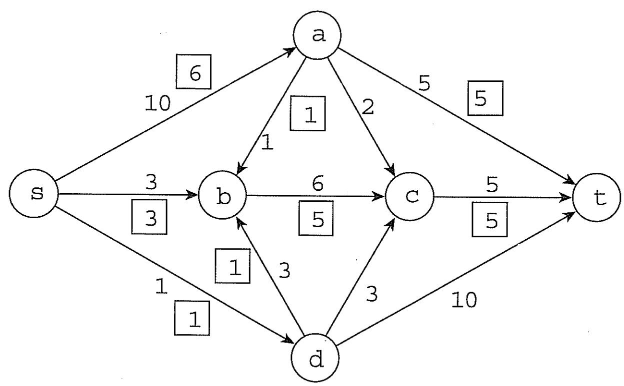


Εικόνα 7.26 Ποια είναι η τιμή της εικονιζόμενης ροής; Είναι μέγιστη ροή; Ποια είναι η ελάχιστη αποκοπή;

3. Η Εικόνα 7.27 δείχνει ένα δίκτυο ροής στο οποίο έχει υπολογιστεί μια ροή *s*-*t*. Η χωρητικότητα της κάθε ακμής παρουσιάζεται ως ετικέτα δίπλα στην ακμή, ενώ οι αριθμοί μέσα στα πλαίσια δείχνουν την ποσότητα ροής που στέλνεται μέσα από κάθε ακμή. (Για τις ακμές που δεν έχουν αριθμούς μέσα σε πλαίσια δεν στέλνεται ροή μέσα από αυτές.)

(α) Ποια είναι η τιμή αυτής της ροής; Είναι αυτή μια μέγιστη ροή (*s*,*t*) για το συγκεκριμένο γράφημα;

(β) Βρείτε μια ελάχιστη αποκοπή *s*-*t* για το δίκτυο ροής της Εικόνας 7.27, και πείτε ποια είναι η χωρητικότητά της.



Εικόνα 7.27 Ποια είναι η τιμή της εικονιζόμενης ροής; Είναι μέγιστη ροή; Ποια είναι η ελάχιστη αποκοπή;

4. Αποφασίστε αν η ακόλουθη πρόταση είναι αληθής ή ψευδής. Αν είναι αληθής, δώστε μια σύντομη εξήγηση. Αν είναι ψευδής, δώστε ένα αντιπαράδειγμα.

Έστω ότι το G είναι ένα αυθαίρετο δίκτυο ροής, με μια προέλευση s, μια απόληξη t, και μια θετική ακέραια χωρητικότητα c_e σε κάθε ακμή e. Αν η f είναι μια μέγιστη ροή s-t στο G, τότε η f προκαλεί κορεσμό σε όλες τις ακμές που *εξέρχονται από το s και μεταφέρουν ροή (δηλαδή σε όλες τις ακμές e που εξέρχονται από το s έχουμε $f(e) = c_e$).*

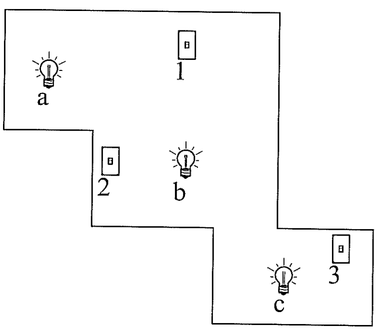
5. Αποφασίστε αν η ακόλουθη πρόταση είναι αληθής ή ψευδής. Αν είναι αληθής, δώστε μια σύντομη εξήγηση. Αν είναι ψευδής, δώστε ένα αντιπαράδειγμα.

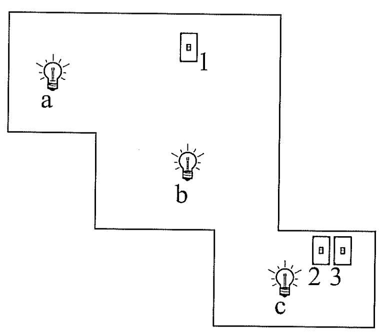
Έστω ότι το G είναι ένα αυθαίρετο δίκτυο ροής, με μια προέλευση s, μια απόληξη t, και μια θετική ακέραια χωρητικότητα c_e σε κάθε ακμή e· και έστω ότι η (A, B) είναι μια ελάχιστη αποκοπή s-t σε σχέση με αυτές τις χωρητικότητες $\{c_e : e \in E\}$. Υποθέστε τώρα ότι προσθέτουμε 1 σε όλες τις χωρητικότητες· τότε η (A, B) εξακολουθεί να είναι μια ελάχιστη αποκοπή s-t σε σχέση με αυτές τις νέες χωρητικότητες $\{1 + c_e : e \in E\}$.

6. Υποθέστε ότι είστε σύμβουλοι στην Επιτροπή Εργονομικής Αρχιτεκτονικής και ήρθαν να σας συμβουλευτούν για το παρακάτω πρόβλημα.

Ενδιαφέρονται να σχεδιάσουν σπίτια που να είναι "φιλικά προς το χρήστη", και έχουν πρόβλημα με τη διευθέτηση των εγκαταστάσεων φωτισμού και των διακοπτών στα νέα σπίτια τα οποία σχεδιάζουν. Για παράδειγμα, σκεφτείτε την περίπτωση ενός οροφοδιαμερίσματος με *n* εγκαταστάσεις φωτισμού και *n* θέσεις για τους διακόπτες στους τοίχους. Θα θέλετε να μπορείτε να καλωδιώσετε τον κάθε διακόπτη ώστε να ελέγχει μία εγκατάσταση φωτισμού, αλλά με τρόπο που να επιτρέπει σε αυτόν που χειρίζεται το διακόπτη να *βλέπει* την εγκατάσταση φωτισμού την οποία ελέγχει.

Κάποιες φορές αυτό είναι δυνατό, ενώ άλλες δεν είναι. Δείτε τις δύο απλές κατόψεις σπιτιών που φαίνονται στην Εικόνα 7.28. Υπάρχουν τρεις εγκαταστάσεις φωτισμού (με ετικέτες *a*, *b*, *c*) και τρεις διακόπτες (με ετικέτες 1, 2, 3). Στην Εικόνα 7.28(α) μπορούμε να καλωδιώσουμε τους διακόπτες και τις εγκαταστάσεις φωτισμού έτσι ώστε από τον κάθε διακόπτη να είναι ορατή η εγκατάσταση φωτισμού, αλλά αυτό δεν είναι δυνατό στην Εικόνα 7.28(β).





Εικόνα 7.28 Η κάτοψη (α) είναι εργονομική, επειδή μπορούμε να καλωδιώσουμε τους διακόπτες και τις εγκαταστάσεις φωτισμού έτσι ώστε η κάθε εγκατάσταση φωτισμού να είναι ορατή από το διακόπτη που την ελέγχει. (Αυτό μπορεί να επιτευχθεί με σύνδεση του διακόπτη 1 στο *a*, του διακόπτη 2 στο *b*, και του διακόπτη 3 στο *c*.) Η κάτοψη (β) δεν είναι εργονομική, επειδή δεν είναι δυνατή κάποια τέτοια καλωδίωση.

Ας αποκαλέσουμε μια κάτοψη, μαζί με τις n θέσεις εγκαταστάσεων φωτισμού και τις n θέσεις των διακοπτών, *εργονομική* αν μπορούμε να συνδέσουμε ένα διακόπτη σε κάθε εγκατάσταση φωτισμού έτσι ώστε η κάθε εγκατάσταση φωτισμού να είναι ορατή από το διακόπτη που την ελέγχει. Η κάτοψη αναπαριστάνεται από ένα σύνολο m οριζόντιων ή κάθετων ευθύγραμμών τμημάτων στο επίπεδο (τους τοίχους), όπου ο τοίχος i έχει άκρα (x_i, y_i), (x'_i, y'_i). Οι n διακόπτες και οι n εγκαταστάσεις φωτισμού προσδιορίζονται με συντεταγμένες στο επίπεδο. Μια εγκατάσταση φωτισμού είναι *ορατή* από ένα διακόπτη εάν το ευθύγραμμο τμήμα που τη συνδέει με αυτόν δεν τέμνει κανέναν από τους τοίχους.

Κατασκευάστε έναν αλγόριθμο που να αποφασίζει αν μια δεδομένη κάτοψη είναι εργονομική. Ο χρόνος εκτέλεσης θα πρέπει να είναι πολυωνυμικός ως προς τα m και n. Μπορείτε να υποθέσετε ότι έχετε μια ρουτίνα, με χρόνο εκτέλεσης $O(1)$, η οποία παίρνει ως είσοδο δύο ευθύγραμμα τμήματα και αποφασίζει αν τέμνονται ή όχι στο επίπεδο.

7. Φανταστείτε ένα σύνολο πελατών κινητού υπολογισμού σε μια πόλη, οι οποίοι θα πρέπει να συνδεθούν σε έναν από τους διάφορους *σταθμούς βάσης*. Θα υποθέσουμε ότι υπάρχουν n πελάτες, με τη θέση του κάθε πελάτη να προσδιορίζεται από τις συντεταγμένες του (x, y) στο επίπεδο. Υπάρχουν επίσης k σταθμοί βάσης· η θέση του κάθε σταθμού προσδιορίζεται επίσης με συντεταγμένες (x, y).

Κάθε πελάτη θέλουμε να τον συνδέσουμε με ακριβώς έναν από τους σταθμούς βάσης. Η επιλογή μας ως προς τις συνδέσεις περιορίζεται με τους ακόλουθους τρόπους. Υπάρχει μια *παράμετρος εμβέλειας* r — ο πελάτης μπορεί να συνδεθεί μόνο σε σταθμούς βάσης που βρίσκονται σε ακτίνα απόστασης r. Υπάρχει επίσης μια *παράμετρος φορτίου* L — δεν μπορούν να συνδεθούν περισσότεροι από L πελάτες σε ένα σταθμό βάσης.

Ο στόχος σας είναι να σχεδιάσετε έναν αλγόριθμο πολυωνυμικού χρόνου για το ακόλουθο πρόβλημα. Με δεδομένο ένα σύνολο πελατών και ένα σύνολο σταθμών βάσης, καθώς και τις παραμέτρους εμβέλειας και φορτίου, ο αλγόριθμος θα πρέπει να αποφασίζει αν όλοι οι πελάτες μπορούν να συνδεθούν ταυτόχρονα σε κάποιο σταθμό βάσης, χωρίς να παραβιαστούν οι συνθήκες εμβέλειας και φορτίου που αναφέραμε στην προηγούμενη παράγραφο.

8. Στατιστικά, ο ερχομός της Άνοιξης τυπικά έχει ως αποτέλεσμα την αύξηση των ατυχημάτων και την αυξημένη ανάγκη για επείγουσα ιατρική περίθαλψη, γεγονός που συχνά απαιτεί μεταγγίσεις αίματος. Ας εξετάσουμε το πρόβλημα το οποίο αντιμετωπίζει ένα νοσοκομείο που προσπαθεί να εκτιμήσει αν τα αποθέματά του σε αίμα είναι επαρκή.

Ο βασικός κανόνας για την παροχή αίματος είναι ο ακόλουθος. Στο αίμα του κάθε ανθρώπου υπάρχουν κάποια *αντιγόνα* (μπορούμε να φανταστούμε τα αντιγόνα ως ένα είδος μοριακής υπογραφής)· δεν μπορεί να λάβει κανείς αίμα με ένα συγκεκριμένο αντιγόνο αν δεν υπάρχει αυτό το αντιγόνο στο αίμα του. Πιο συγκεκριμένα, αυτή η αρχή βρίσκεται πίσω από το διαχωρισμό του αίματος σε τέσσερις *ομάδες αίματος*: A, B, AB, και O. Το αίμα της ομάδας A έχει το αντιγόνο A, το αίμα της ομάδας B έχει το αντιγόνο B, το αίμα της ομάδας AB έχει και τα δύο αντιγόνα, και το αίμα της ομάδας O δεν έχει κανένα. Έτσι, οι ασθενείς με ομάδα αίματος A μπορούν σε μια μετάγγιση να λάβουν αίμα ομάδας A ή O, οι ασθενείς με ομάδα αίματος B μπορούν να λάβουν μόνο αίμα B ή O, οι ασθενείς με ομάδα αίματος O μπορούν να λάβουν μόνο αίμα ομάδας O, και οι ασθενείς με ομάδα αίματος AB μπορούν να λάβουν αίμα οποιασδήποτε ομάδας.[4]

(α) Έστω ότι τα s_O, s_A, s_B, και s_{AB} συμβολίζουν το απόθεμα σε μονάδες για τις τέσσερις διαφορετικές ομάδες αίματος. Υποθέστε ότι το νοσοκομείο γνωρίζει την αναμενόμενη ζήτηση για κάθε ομάδα αίματος d_O, d_A, d_B, και d_{AB} για την επόμενη εβδομάδα. Κατασκευάστε έναν αλγόριθμο πολυωνυμικού χρόνου ο οποίος θα εκτιμά αν το διαθέσιμο αίμα είναι επαρκές για την αναμενόμενη ζήτηση.

Ομάδα αίματος	**Απόθεμα**	**Ζήτηση**
O	50	45
A	36	42
B	11	8
AB	8	3

(β) Εξετάστε το ακόλουθο παράδειγμα. Κατά την επόμενη εβδομάδα αναμένουν να χρειαστούν το πολύ 100 μονάδες αίματος. Η τυπική κατανομή των ομάδων αίματος για τους ασθενείς στις ΗΠΑ είναι περίπου 45 τοις εκατό ομάδα O, 42 τοις εκατό ομάδα A, 10 τοις εκατό ομάδα B, και 3 τοις εκατό ομάδα AB. Το νοσοκομείο θέλει να γνωρίζει αν το απόθεμα αίματος που διαθέτει θα είναι επαρκές αν καταφθάσουν 100 ασθενείς με την αναμενόμενη κατανομή μονάδων αίματος. Υπάρχουν συνολικά διαθέσιμες 105 μονάδες αίματος. Ο παραπάνω πίνακας δείχνει τη ζήτηση αυτή και το διαθέσιμο απόθεμα.

Είναι αρκετές οι διαθέσιμες 105 μονάδες αίματος για την εξυπηρέτηση των 100 ζητούμενων μονάδων; Βρείτε μια εκχώρηση που να ικανοποιεί το μέγιστο δυνατό αριθμό ασθενών. Χρησιμοποιήστε ένα επιχείρημα που βασίζεται στην αποκοπή ελάχιστης χωρητικότητας για να δείξετε γιατί δεν μπορούν να λάβουν αίμα όλοι οι ασθενείς. Επίσης, δώστε μια εξήγηση αυτού του γεγονότος που να είναι κατανοητή από τους υπεύθυνους του νοσοκομείου, οι οποίοι δεν έχουν μελετήσει αλγορίθμους. (Έτσι, για παράδειγμα, αυτή η εξήγηση δεν θα πρέπει να περιλαμβάνει τους όρους *ροή*, *αποκοπή*, ή *γράφημα* με την έννοια με την οποία τους χρησιμοποιούμε σε αυτό το βιβλίο.)

---

[4] Ο Αυστριακός επιστήμονας Karl Landsteiner πήρε το 1930 το βραβείο Νόμπελ για την ανακάλυψη των ομάδων αίματος A, B, O, και AB.

9. Ζητήματα ροής δικτύου ανακύπτουν όταν αντιμετωπίζουμε φυσικές καταστροφές ή άλλες κρίσεις, αφού τα μεγάλα αναπάντεχα συμβάντα συχνά απαιτούν τη μετακίνηση και εκκένωση μεγάλων αριθμών ανθρώπων σε μικρό χρονικό διάστημα.

Θεωρήστε το ακόλουθο σενάριο. Εξαιτίας μιας μεγάλης πλημμύρας στην περιοχή οι νοσηλευτές έχουν προσδιορίσει ένα σύνολο n τραυματισμένων ανθρώπων, διάσπαρτων στην περιφέρεια, που θα πρέπει να μεταφερθούν εσπευσμένα σε νοσοκομεία. Υπάρχουν k νοσοκομεία στην περιοχή, και ο καθένας από τους n ανθρώπους θα πρέπει να μεταφερθεί σε ένα νοσοκομείο που βρίσκεται σε μισής ώρας απόσταση με το αυτοκίνητο από την τρέχουσα θέση του (έτσι οι διάφοροι άνθρωποι θα έχουν διαφορετικές επιλογές ως προς τα νοσοκομεία, ανάλογα με τη θέση στην οποία βρίσκονται).

Ταυτόχρονα, δεν θέλουμε να υπερφορτώσουμε κανένα από τα νοσοκομεία στέλνοντάς του υπερβολικά πολλούς ασθενείς. Οι νοσηλευτές είναι σε επαφή με κινητά τηλέφωνα, και θέλουν συλλογικά να ξέρουν αν μπορούν να επιλέξουν ένα νοσοκομείο για καθέναν από τους τραυματίες έτσι ώστε το φορτίο των νοσοκομείων να είναι *ισοσταθμισμένο*: Το κάθε νοσοκομείο δέχεται το πολύ $\lceil n/k \rceil$ ανθρώπους.

Κατασκευάστε έναν αλγόριθμο πολυωνυμικού χρόνου ο οποίος θα παίρνει τις πληροφορίες σχετικά με τη θέση των ανθρώπων και θα προσδιορίζει αν κάτι τέτοιο είναι δυνατό.

10. Υποθέστε ότι σας δίνεται ένα κατευθυνόμενο γράφημα $G = (V, E)$, με μια θετική ακέραια χωρητικότητα c_e σε καθεμία από τις ακμές e, μια προέλευση $s \in V$, και μια απόληξη $t \in V$. Σας δίνεται επίσης μια μέγιστη ροή s-t στο G, η οποία προσδιορίζεται από μια τιμή ροής f_e σε καθεμία από τις ακμές e. Η ροή f είναι *ακυκλική:* Δεν υπάρχει στο G κανένας κύκλος στον οποίο όλες οι ακμές να μεταφέρουν θετική ροή. Επίσης, η ροή f έχει ακέραιες τιμές.

Υποθέστε τώρα ότι διαλέγουμε μια συγκεκριμένη ακμή $e^* \in E$ και ελαττώνουμε τη χωρητικότητά της κατά 1 μονάδα. Δείξτε με ποιον τρόπο μπορούμε να βρούμε σε χρόνο $O(m + n)$ τη μέγιστη ροή στο γράφημα που προκύπτει, όπου m είναι ο αριθμός ακμών του G και n είναι ο αριθμός των κόμβων του.

11. Οι φίλοι σας έχουν γράψει ένα πολύ γρήγορο τμήμα κώδικα μέγιστης ροής, το οποίο βασίζεται στην επαναληπτική εύρεση διαδρομών επαύξησης όπως στην Ενότητα 7.1. Παρόλα αυτά, όταν ρίξατε μια ματιά στην έξοδο του προγράμματος αυτού, συνειδητοποιήσατε ότι δεν βρίσκει πάντα τη *μέγιστη* τιμή. Το σφάλμα ήταν πολύ εύκολο να εντοπιστεί· οι φίλοι σας δεν χρησιμοποίησαν την ιδέα των ανάδρομων ακμών στον κώδικά τους, και έτσι η υλοποίησή τους φτιάχνει μια εκδοχή του υπολοιπόμενου γραφήματος που *αυξάνει μόνο τις ευθύδρομες ακμές*. Με άλλα λόγια, ψάχνει για διαδρομές s-t σε ένα γράφημα $\tilde{G}_f$ που αποτελείται μόνο από τις ακμές e για τις οποίες $f(e) < c_e$, και τερματίζει όταν δεν υπάρχει διαδρομή επαύξησης που να αποτελείται αποκλειστικά από τέτοιες ακμές. Θα τον αποκαλέσουμε αυτό αλγόριθμο Μόνο Ευθύδρομων Ακμών (Forward-Edge-Only). (Σημειώστε ότι δεν προσπαθούμε να περιγράψουμε πώς επιλέγει αυτός ο αλγόριθμος τις ευθύδρομες ακμές· μπορεί να τις επιλέγει με όποιον τρόπο θέλει, υπό τον όρο ότι τερματίζει όταν δεν υπάρχουν άλλες διαδρομές ευθύδρομων ακμών.)

Είναι δύσκολο να πείσετε τους φίλους σας ότι θα πρέπει να ξαναγράψουν τον κώδικα. Εκτός από την εκπληκτική του ταχύτητα, ισχυρίζονται επίσης ότι στην πραγματικότητα δεν επιστρέφει ποτέ μια ροή της οποίας η τιμή να είναι μικρότερη από ένα σταθερό κλάσμα της βέλτιστης ροής. Το πιστεύετε αυτό; Η ουσία του ισχυρισμού τους είναι ακριβώς η ακόλουθη πρόταση.

Υπάρχει μια απόλυτη σταθερά $b > 1$ (ανεξάρτητη από το συγκεκριμένο δίκτυο ροής εισόδου), έτσι ώστε σε κάθε περίπτωση του προβλήματος Μέγιστης Ροής ο αλγόριθμος Μόνο Ευθύδρομων Ακμών να είναι εξασφαλισμένο ότι θα βρει μια τιμή με τιμή τουλάχιστον $1/b$ φορές την τιμή της μέγιστης ροής (ανεξάρτητα από τον τρόπο με τον οποίο επιλέγει τις διαδρομές ευθύδρομων ακμών).

Αποφασίστε αν αυτή η πρόταση είναι αληθής ή ψευδής, και δώστε μια απόδειξη είτε της πρότασης είτε της άρνησής της.

12. Θεωρήστε το ακόλουθο πρόβλημα. Σας δίνεται ένα δίκτυο ροής με ακμές μοναδιαίας χωρητικότητας: Αυτό αποτελείται από ένα γράφημα $G = (V, E)$, μια προέλευση $s \in V$, και μια απόληξη $t \in V$· και ισχύει $c_e = 1$ για όλα τα $e \in E$. Σας δίνεται επίσης μια παράμετρος k.

Ο στόχος είναι να διαγράψετε k ακμές έτσι ώστε να μειώσετε όσο το δυνατόν περισσότερο τη μέγιστη ροή s-t για το γράφημα G. Με άλλα λόγια, θα πρέπει να βρείτε ένα σύνολο ακμών $F \subseteq E$ έτσι ώστε $|F| = k$ και η μέγιστη ροή s-t στο $G' = (V, E - F)$ να είναι όσο το δυνατόν μικρότερη.

Κατασκευάστε έναν αλγόριθμο πολυωνυμικού χρόνου για την επίλυση αυτού του προβλήματος.

13. Στα τυπικά προβλήματα Μέγιστης Ροής s-t υποθέτουμε ότι οι ακμές έχουν χωρητικότητες και ότι δεν υπάρχει όριο ως προς τη ροή που επιτρέπεται να περάσει από τους κόμβους. Σε αυτό το πρόβλημα θα εξετάσουμε μια παραλλαγή των προβλημάτων Μέγιστης Ροής και Ελάχιστης Αποκοπής όπου υπάρχουν χωρητικότητες κόμβων.

Έστω ότι $G = (V, E)$ είναι ένα κατευθυνόμενο γράφημα, με προέλευση $s \in V$, απόληξη $t \in V$, και μη αρνητικές χωρητικότητες κόμβων $\{c_v \geq 0\}$ για κάθε κόμβο $v \in V$. Με δεδομένη μια ροή f σε αυτό το γράφημα, η ροή μέσω του κόμβου v ορίζεται ως $f^{\text{in}}(v)$. Λέμε ότι η ροή είναι εφικτή εάν ικανοποιεί τους συνηθισμένους περιορισμούς διατήρησης ροής και τους περιορισμούς ως προς τη χωρητικότητα των κόμβων: $f^{\text{in}}(v) \leq c_v$ για όλους τους κόμβους.

Κατασκευάστε έναν αλγόριθμο πολυωνυμικού χρόνου για την εύρεση μια μέγιστης ροής s-t σε ένα τέτοιο δίκτυο με χωρητικότητες κόμβων. Ορίστε την αποκοπή s-t για τα δίκτυα με χωρητικότητες κόμβων, και δείξτε ότι ισχύει το ανάλογο του θεωρήματος Μέγιστης Ροής και Ελάχιστης Αποκοπής.

14. Το *Πρόβλημα Διαφυγής* (Escape Problem) ορίζεται με τον ακόλουθο τρόπο. Μας δίνεται ένα κατευθυνόμενο γράφημα $G = (V, E)$ (φανταστείτε το οδικό δίκτυο). Μια συγκεκριμένη συλλογή κόμβων $X \subset V$ προσδιορίζονται ως *κατοικούμενοι κόμβοι*, και μια άλλη συλλογή $S \subset V$ προσδιορίζονται ως *ασφαλείς κόμβοι*. (Υποθέστε ότι τα X και S είναι ξένα μεταξύ τους.) Σε κατάσταση ανάγκης θέλουμε δρομολόγια εκκένωσης από τους κατοικημένους κόμβους στους ασφαλείς κόμβους. Το σύνολο των δρομολογίων εκκένωσης ορίζεται ως ένα σύνολο διαδρομών στο G έτσι ώστε (i) ο κάθε κόμβος του X να βρίσκεται στην αφετηρία μιας διαδρομής, (ii) ο τελευταίος κόμβος κάθε διαδρομής να ανήκει στο S, και (iii) οι διαδρομές να μην έχουν κοινές ακμές. Ένα τέτοιο σύνολο διαδρομών δίνει στους κατοίκους των κατοικούμενων κόμβων έναν τρόπο για να "διαφύγουν" στο S, χωρίς να προκαλέσουν υπερβολική συμφόρηση σε καμία ακμή του G.

(α) Με δεδομένα τα G, X, και S, δείξτε πώς μπορούμε να αποφασίσουμε σε πολυωνυμικό χρόνο αν υπάρχει ένα τέτοιο σύνολο δρομολογίων εκκένωσης.

(β) Υποθέστε ότι έχουμε ακριβώς το ίδιο πρόβλημα όπως στο (α), αλλά θέλουμε να επιβάλουμε μια ακόμα πιο ισχυρή συνθήκη "μη συμφόρησης" (iii). Έτσι αλλάζουμε τη συνθήκη (iii) σε "οι διαδρομές δεν έχουν καθόλου κοινούς *κόμβους*".

Με αυτή τη νέα συνθήκη, δείξτε πώς μπορούμε να αποφασίσουμε σε πολυωνυμικό χρόνο αν υπάρχει ένα τέτοιο σύνολο δρομολογίων εκκένωσης.

Δώστε επίσης ένα παράδειγμα με τα ίδια G, X, και S, στο οποίο η απάντηση να είναι θετική στην ερώτηση (α) και αρνητική στην ερώτηση (β).

15. Υποθέστε ότι εσείς και η φίλη σας η Αλίκη ζείτε, μαζί με n-2 άλλους ανθρώπους, σε ένα δημοφιλές κοινόβιο διαμέρισμα έξω από το πανεπιστήμιο, το οποίο ονομάζεται Upson Collective. Τις επόμενες n βραδιές ο καθένας από εσάς αναμένεται να μαγειρέψει μία φορά δείπνο για όλους, έτσι ώστε κάποιος να μαγειρέψει το καθένα από τα βράδια.

Όπως είναι φυσικό, όλοι έχουν διενέξεις προγράμματος σε κάποιες από τις βραδιές (π.χ. εξετάσεις, συναυλίες, κ.λπ.), έτσι η απόφαση για το ποιος θα μαγειρέψει την κάθε βραδιά γίνεται δύσκολη υπόθεση. Για να είμαστε συγκεκριμένοι, ας δώσουμε στα άτομα τις ετικέτες

$$\{p_1, ..., p_n\},$$

στις βραδιές τις ετικέτες

$$\{d_1, ..., d_n\}$$

και για το άτομο p_i υπάρχει ένα σύνολο βραδιών $S_i \subset \{d_1, ..., d_n\}$ κατά τις οποίες *δεν* μπορεί να μαγειρέψει.

Το *εφικτό πρόγραμμα δείπνων* είναι μια αντιστοίχιση του κάθε ατόμου του κοινοβίου σε διαφορετική βραδιά, έτσι ώστε το κάθε άτομο να μαγειρεύει το πολύ μία βραδιά, να υπάρχει σε κάθε βραδιά κάποιος που να μαγειρεύει, και αν το άτομο p_i μαγειρεύει τη βραδιά d_j, τότε $d_j \notin S_i$.

(α) Περιγράψτε ένα διμερές γράφημα G έτσι ώστε το G να έχει τέλειο ταίριασμα αν και μόνο αν υπάρχει εφικτό πρόγραμμα δείπνων για το κοινόβιο.

(β) Η φίλη σας η Αλίκη ανέλαβε το έργο να ετοιμάσει ένα εφικτό πρόγραμμα δείπνων. Μετά από πολλή προσπάθεια, κατασκεύασε ένα πρόγραμμα που ισχυριζόταν ότι είναι εφικτό και έφυγε για τα μαθήματα της ημέρας.

Δυστυχώς, όταν ρίξατε μια ματιά στο πρόγραμμα που ετοίμασε, παρατηρήσατε ένα μεγάλο πρόβλημα. n-2 από τα άτομα του κοινόβιου έχουν αντιστοιχιστεί σε διαφορετικές βραδιές στις οποίες τα άτομα αυτά είναι διαθέσιμα: δεν υπάρχει πρόβλημα με αυτούς. Όμως για τα άλλα δύο άτομα, τα p_i and p_j, και τις δύο άλλες βραδιές d_k and d_ℓ, ανακαλύψατε ότι κατά λάθος όρισε πως και ο p_i και ο p_j θα μαγειρέψουν τη βραδιά d_k και δεν καθόρισε κανέναν για να μαγειρέψει τη βραδιά d_ℓ.

Θέλετε να διορθώσετε το λάθος της Αλίκης, χωρίς όμως να επανυπολογίσετε τα πάντα από την αρχή. Δείξτε ότι είναι δυνατό, χρησιμοποιώντας το "σχεδόν σωστό" πρόγραμμά της, να αποφασίσουμε σε μόνο $O(n^2)$ χρόνο αν υπάρχει εφικτό πρόγραμμα δείπνων για το κοινόβιο. (Αν υπάρχει τέτοιο πρόγραμμα, θα πρέπει επίσης να το δώσετε ως έξοδο.)

16. Τις γεμάτες ευφορία πρώτες ημέρες του Παγκόσμιου Ιστού οι άνθρωποι συνήθιζαν να ισχυρίζονται ότι μεγάλο μέρος του τεράστιου δυναμικού εταιρειών όπως η Yahoo! ήταν τα "μάτια" — το απλό γεγονός ότι εκατομμύρια άνθρωποι βλέπουν τις σελίδες της κάθε μέρα. Επιπρόσθετα, ένας ιστότοπος σαν αυτόν της Yahoo! μπορεί, πείθοντας τους ανθρώπους να καταχωρίσουν προσωπικά τους δεδομένα σε αυτόν, να δείχνει σε κάθε χρήστη μια εξαιρετικά στοχευμένη διαφήμιση όποτε αυτός επισκέπτεται τον ιστότοπο, με έναν τρόπο που δεν μπορούν να ελπίζουν ότι θα επιτύχουν τα τηλεοπτικά δίκτυα ή τα περιοδικά. Έτσι αν ο χρήστης έχει πει στη Yahoo! ότι είναι ένας 20χρονος τελειόφοιτος των επιστήμης των υπολογιστών στο Πανεπιστήμιο Cornell, ο ιστότοπος μπορεί να παρουσιάζει ένα διαφημιστικό πανό για διαμερίσματα στην Ιθάκη της Νέας Υόρκης· από την άλλη πλευρά, αν ο χρήστης έχει πει ότι είναι ένας 50χρονος τραπεζικός υπάλληλος από το Greenwich του Κονέκτικατ, ο ιστότοπος μπορεί να εμφανίζει διαφημιστικό πανό με οικογενειακά αυτοκίνητα Lincoln.

Όμως η απόφαση για το ποιες διαφημίσεις θα πρέπει να εμφανίσει ο ιστότοπος σε ποιους ανθρώπους περιλαμβάνει παρασκηνιακά μερικούς πολύ σημαντικούς υπολογισμούς. Υποθέστε ότι οι διευθυντές ενός δημοφιλούς ιστότοπου έχουν προσδιορίσει k διακριτές *δημογραφικές ομάδες* G_1, G_2, ..., G_k. (Αυτές οι ομάδες μπορεί να επικαλύπτονται· για παράδειγμα, η ομάδα G_1 μπορεί να είναι οι κάτοικοι στην πολιτεία της Νέας Υόρκης και η ομάδα G_2 να είναι οι άνθρωποι που έχουν πτυχίο στους υπολογιστές.) Ο ιστότοπος έχει συμβόλαια με m διαφορετικούς *διαφημιστές*, για να δείξει έναν αριθμό αντιγράφων των διαφημίσεών τους στους χρήστες του ιστότοπου. Ας δούμε με τι μοιάζει το συμβόλαιο με το διαφημιστή i.

- Για ένα υποσύνολο $X_i \subseteq \{G_1, ..., G_k\}$ των δημογραφικών ομάδων, ο διαφημιστής i θέλει να εμφανίζονται οι διαφημίσεις του σε χρήστες που ανήκουν σε τουλάχιστον μία από τις δημογραφικές ομάδες του συνόλου X_i.
- Για έναν αριθμό r_i, ο διαφημιστής i θέλει να εμφανίζονται οι διαφημίσεις του σε τουλάχιστον r_i χρήστες κάθε λεπτό.

Θεωρήστε τώρα το πρόβλημα του σχεδιασμού μιας καλής *διαφημιστικής πολιτικής* — έναν τρόπο για να δείχνουμε μια διαφήμιση σε κάθε χρήστη του ιστότοπου. Υποθέστε ότι σε ένα δεδομένο λεπτό υπάρχουν n χρήστες που έχουν επισκεφθεί τον ιστότοπο. Αφού έχουν πληροφορίες εγγραφής για καθέναν από αυτούς τους χρήστες, γνωρίζουμε ότι ο χρήστης j (για $j = 1, 2, ..., n$) ανήκει σε ένα υποσύνολο $U_j \subseteq \{G_1, ..., G_k\}$ των δημογραφικών ομάδων. Το πρόβλημα είναι το ακόλουθο: Υπάρχει τρόπος να δείξουμε μία διαφήμιση σε κάθε χρήστη έτσι ώστε να τηρηθούν για αυτό το λεπτό τα συμβόλαια του ιστότοπου με καθέναν από τους m διαφημιστές; (Με άλλα λόγια, για κάθε $i = 1, 2, ..., m$, μπορούν τουλάχιστον r_i από τους n χρήστες, καθένας από τους οποίους ανήκει σε τουλάχιστον μία από τις δημογραφικές ομάδες του συνόλου X_i, να βλέπει μια διαφήμιση που παρέχεται από το διαφημιστή i;)

Κατασκευάστε έναν αποδοτικό αλγόριθμο που να αποφασίζει αν είναι δυνατό κάτι τέτοιο και, όταν είναι, να επιλέγει πραγματικά μια διαφήμιση την οποία θα εμφανίσει σε κάθε χρήστη.

17. Κληθήκατε να βοηθήσετε κάποιους διαχειριστές δικτύου να διαγνώσουν την έκταση της αστοχίας στο δίκτυό τους. Το δίκτυο είναι σχεδιασμένο να μεταφέρει κίνηση από έναν καθορισμένο κόμβο προέλευσης s σε έναν καθορισμένο κόμβο προορισμού t, έτσι θα το μοντελοποιήσουμε ως κατευθυνόμενο γράφημα $G = (V, E)$ στο οποίο η χωρητικότητα της κάθε ακμής είναι 1 και ο κάθε κόμβος βρίσκεται σε τουλάχιστον μία διαδρομή από το s στο t.

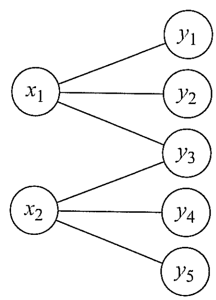
Όταν όλα λειτουργούν ομαλά στο δίκτυο, η μέγιστη ροή s-t στο γράφημα G έχει τιμή k. Παρόλα αυτά, η τρέχουσα κατάσταση (και ο λόγος για τον οποίο βρίσκεστε εκεί) είναι ότι ένας κακόβουλος εισβολέας έχει καταστρέψει μερικές από τις ακμές του δικτύου, και έτσι δεν υπάρχει πλέον καμία διαδρομή από το s στο t που να χρησιμοποιεί τις υπόλοιπες (εναπομείνασες) ακμές. Για λόγους τους οποίους δεν θα αναφέρουμε εδώ, οι διαχειριστές του δικτύου πιστεύουν ότι ο εισβολέας έχει καταστρέψει μόνο k ακμές, τον ελάχιστο αριθμό που απαιτείται για να απομονώσει το s από το t (με άλλα λόγια, το μέγεθος της ελάχιστης αποκοπής s-t)· και θα υποθέσουμε ότι έχουν δίκιο σε αυτή την πεποίθησή τους.

Οι διαχειριστές δικτύου εκτελούν στον κόμβο s ένα εργαλείο παρακολούθησης, το οποίο έχει την ακόλουθη συμπεριφορά. Αν δώσετε την εντολή *ping*(v) για ένα δεδομένο κόμβο v, η εντολή σάς λέει αν υπάρχει αυτή τη στιγμή διαδρομή από το s στο v. (Έτσι η εντολή *ping*(t) αναφέρει ότι δεν υπάρχει αυτή τη στιγμή διαδρομή· από την άλλη πλευρά, η εντολή *ping*(s) αναφέρει πάντα ότι υπάρχει μια διαδρομή από το s στον εαυτό του.) Αφού δεν είναι πρακτικά δυνατό να πάνε και να ελέγξουν όλες τις ακμές του δικτύου, θέλουν να προσδιορίσουν την έκταση της αστοχίας χρησιμοποιώντας αυτό το εργαλείο παρακολούθησης, με συνετή χρήση της εντολής *ping*.

Να λοιπόν το πρόβλημα που αντιμετωπίζετε: Κατασκευάστε έναν αλγόριθμο ο οποίος θα δίνει μια ακολουθία εντολών *ping* σε διάφορους κόμβους του δικτύου, και μετά αναφέρετε το *πλήρες* σύνολο κόμβων που δεν είναι τη δεδομένη στιγμή προσπελάσιμοι από τον κόμβο s. Θα μπορούσατε, φυσικά, να το κάνετε αυτό δίνοντας την εντολή για όλους τους κόμβους του δικτύου, όμως θέλετε να το επιτύχετε με όσο το δυνατόν λιγότερες εντολές γίνεται (με δεδομένη την παραδοχή ότι έχουν διαγραφεί μόνο k ακμές). Κατά την εκτέλεση αυτής της ακολουθίας εντολών, ο αλγόριθμός σας επιτρέπεται να αποφασίζει ποιον κόμβο θα πρέπει να ελέγξει στη συνέχεια με βάση το αποτέλεσμα προηγούμενων εντολών *ping*.

Κατασκευάστε έναν αλγόριθμο ο οποίος θα εκτελεί αυτή την εργασία με μόνο $O(k \log n)$ εντολές *ping*.

18. Εξετάζουμε το πρόβλημα του Διμερούς Ταιριάσματος σε ένα διμερές γράφημα $G = (V, E)$. Ως συνήθως, λέμε ότι το V διαμερίζεται στα σύνολα X και Y, και η κάθε ακμή έχει το ένα άκρο στο X και το άλλο στο Y.



Εικόνα 7.29 Ένα παράδειγμα Επέκτασης με Κάλυψη.

Αν το M είναι ένα ταίριασμα στο G, λέμε ότι ο κόμβος $y \in Y$ *καλύπτεται* από το M αν το y αποτελεί άκρο σε μία από τις ακμές του M.

(α) Θεωρήστε το ακόλουθο πρόβλημα. Μας δίνεται το γράφημα G και ένα ταίριασμα M στο G. Για ένα δεδομένο αριθμό k, θέλουμε να αποφασίσουμε αν υπάρχει στο G ένα ταίριασμα M' τέτοιο ώστε

(i) το M' έχει k περισσότερες ακμές από το M, *και*

(ii) όλοι οι κόμβοι $y \in Y$ που καλύπτονται στο M καλύπτονται επίσης και στο M'.

Θα το ονομάσουμε αυτό *Πρόβλημα Επέκτασης με Κάλυψη*, με είσοδο τα G, M, και k, και θα λέμε ότι το M' είναι μια *λύση* αυτού του προβλήματος.

Κατασκευάστε έναν αλγόριθμο πολυωνυμικού χρόνου που θα παίρνει ένα στιγμιότυπο του προβλήματος Επέκτασης με Κάλυψη και είτε θα επιστρέφει μια λύση M', ή θα αναφέρει (σωστά) ότι δεν υπάρχει λύση. (Θα πρέπει να συμπεριλάβετε μια ανάλυση ως προς το χρόνο εκτέλεσης, καθώς και μια σύντομη απόδειξη για την ορθότητα του αλγορίθμου.)

Σημείωση: Ίσως είναι καλό να ρίξετε μια ματιά στο (β) για να βοηθηθείτε ως προς την κατασκευή του αλγορίθμου.

Παράδειγμα. Εξετάστε την Εικόνα 7.29, και υποθέστε ότι το M είναι το ταίριασμα που αποτελείται από την ακμή (x_1, y_2). Υποθέστε ότι μας γίνεται το παραπάνω ερώτημα με $k = 1$.

Σε αυτό το στιγμιότυπο του προβλήματος Επέκτασης με Κάλυψη η απάντηση είναι καταφατική. Μπορούμε να θέσουμε ως M' το ταίριασμα που αποτελείται (για παράδειγμα) από τις δύο ακμές (x_1, y_2) και (x_2, y_4)· το M' έχει μία περισσότερη ακμή από το M, και το y_2 εξακολουθεί να καλύπτεται στο M'.

(β) Δώστε ένα παράδειγμα στιγμιοτύπου του προβλήματος Επέκτασης με Κάλυψη, το οποίο θα προσδιορίζεται από τα G, M, και k, έτσι ώστε να προκύπτει η ακόλουθη κατάσταση.

Το στιγμιότυπο έχει λύση· όμως σε οποιαδήποτε λύση M' οι ακμές του M δεν σχηματίζουν υποσύνολο των ακμών του M'.

(γ) Έστω ότι το G είναι ένα διμερές γράφημα, και έστω ότι το M είναι κάποιο ταίριασμα στο G. Θεωρήστε τις ακόλουθες δύο ποσότητες.

– K_1 είναι το μέγεθος του μεγαλύτερου ταιριάσματος M' έτσι ώστε κάθε κόμβος y που καλύπτεται από το M να καλύπτεται επίσης και από το M'.

– K_2 είναι το μέγεθος του μεγαλύτερου ταιριάσματος M'' στο G.

Είναι προφανές ότι $K_1 \leq K_2$, αφού το K_2 λαμβάνεται με εξέταση *όλων των δυνατών* ταιριασμάτων στο G.

Αποδείξτε ότι στην πραγματικότητα $K_1 = K_2$· δηλαδή, μπορούμε να λάβουμε ένα μέγιστο ταίριασμα ακόμα και αν περιοριστούμε στην κάλυψη όλων των κόμβων που καλύπτονται από το αρχικό μας ταίριασμα M.

19. Βοηθάτε περιοδικά την ιατρική συμβουλευτική εταιρία Γιατροί Χωρίς Σαββατοκύριακα για διάφορα ζητήματα χρονοπρογραμματισμού νοσοκομείων, και ήρθαν να ζητήσουν τη συμβουλή σας για ένα νέο πρόβλημα. Για καθεμία από τις επόμενες n ημέρες το νοσοκομείο έχει προσδιορίσει τον αριθμό των γιατρών που θέλει να είναι διαθέσιμοι· δηλαδή, για την ημέρα i, έχει την ζήτηση να είναι παρόντες *ακριβώς* p_i γιατροί.

Υπάρχουν k γιατροί, και από καθέναν τους έχει ζητηθεί να δώσουν μια λίστα με τις ημέρες στις οποίες είναι πρόθυμοι να δουλέψουν. Έτσι ο γιατρός j έχει δώσει ένα σύνολο L_j με τις ημέρες στις οποίες είναι διαθέσιμος για εργασία.

Το σύστημα που παράγεται από τη συμβουλευτική εταιρεία θα πρέπει να παίρνει αυτές τις λίστες και να προσπαθεί να επιστρέψει σε κάθε γιατρό j μια λίστα L'_j με τις ακόλουθες ιδιότητες.

(Α) Το L'_j είναι ένα υποσύνολο του L_j, έτσι ώστε κάθε γιατρός j να δουλεύει μόνο σε ημέρες που θεωρεί αποδεκτές.

(Β) Αν εξετάσουμε όλες τις λίστες $L'_1, ..., L'_k$, θα διαπιστώσουμε ότι ακριβώς p_i γιατροί θα είναι παρόντες κατά την ημέρα i, για $i = 1, 2, ..., n$.

(α) Περιγράψτε έναν αλγόριθμο πολυωνυμικού χρόνου που υλοποιεί αυτό το σύστημα. Ειδικότερα, δώστε έναν αλγόριθμο πολυωνυμικού χρόνου που παίρνει τους αριθμούς $p_1, p_2, ..., p_n$ και τις λίστες $L_1, ..., L_k$, και κάνει ένα από τα ακόλουθα δύο πράγματα.

- Επιστρέφει τις λίστες $L'_1, L'_2, ..., L'_k$ που ικανοποιούν τις ιδιότητες (Α) και (Β)˙ ή
- Αναφέρει (σωστά) ότι δεν υπάρχει κανένα σύνολο από λίστες $L'_1, L'_2, ..., L'_k$ που να ικανοποιεί και τις δύο ιδιότητες (Α) και (Β).

(β) Το νοσοκομείο διαπίστωσε ότι οι γιατροί τείνουν συχνά να υποβάλλουν λίστες που είναι υπερβολικά περιοριστικές, και έτσι συμβαίνει συχνά το σύστημα να αναφέρει (σωστά, αλλά αυτό δεν είναι ευχάριστο) ότι δεν υπάρχει κανένα αποδεκτό σύνολο από λίστες $L'_1, L'_2, ..., L'_k$.

Έτσι το νοσοκομείο αποφάσισε να χαλαρώσει τις απαιτήσεις με τον επόμενο τρόπο. Προστέθηκε μια νέα παράμετρος $c > 0$, και το σύστημα θα πρέπει τώρα να προσπαθεί να επιστρέψει σε κάθε γιατρό j μια λίστα L'_j που έχει τις ακόλουθες ιδιότητες.

(Α*) Το L'_j περιέχει το πολύ c ημέρες που δεν περιλαμβάνονται στη λίστα L_j.

(Β) (*Ίδιο με προηγουμένως*) Αν εξετάσουμε όλες τις λίστες $L'_1, ..., L'_k$, θα διαπιστώσουμε ότι ακριβώς p_i γιατροί θα είναι παρόντες κατά την ημέρα i, για $i = 1, 2, ..., n$.

Περιγράψτε έναν αλγόριθμο πολυωνυμικού χρόνου ο οποίος υλοποιεί αυτό το αναθεωρημένο σύστημα. Ο αλγόριθμος θα πρέπει να παίρνει τους αριθμούς $p_1, p_2, ..., p_n$, τις λίστες $L_1, ..., L_k$, και την παράμετρο $c > 0$, και να κάνει ένα από τα ακόλουθα δύο πράγματα.

- Να επιστρέφει τις λίστες $L'_1, L'_2, ..., L'_k$ που ικανοποιούν τις ιδιότητες (Α*) και (Β)˙ ή
- Να αναφέρει (σωστά) ότι δεν υπάρχει κανένα σύνολο από λίστες $L'_1, L'_2, ..., L'_k$ που να ικανοποιεί και τις δύο ιδιότητες (Α*) και (Β).

20. Οι φίλοι σας ασχολούνται με επιστημονικά ατμοσφαιρικά πειράματα μεγάλης κλίμακας. Πρέπει να πάρουν καλές μετρήσεις για ένα σύνολο S από n διαφορετικές συνθήκες της ατμόσφαιρας (όπως η στάθμη του όζοντος σε διάφορα σημεία), και έχουν ένα σύνολο από m αερόστατα τα οποία σκοπεύουν να στείλουν για να πραγματοποιήσουν αυτές τις μετρήσεις. Το κάθε αερόστατο μπορεί να πραγματοποιήσει το πολύ δύο μετρήσεις.

Δυστυχώς, δεν μπορούν όλα τα αερόστατα να μετρήσουν όλες τις συνθήκες, έτσι για κάθε αερόστατο $i = 1, \ldots, m$ έχουν ένα σύνολο S_i από συνθήκες τις οποίες μπορεί να μετρήσει το αερόστατο i. Τέλος, για να είναι τα αποτελέσματα πιο αξιόπιστα, σχεδιάζουν να κάνουν την κάθε μέτρηση με τουλάχιστον k διαφορετικά αερόστατα. (Σημειώστε ότι κανένα αερόστατο δεν θα πρέπει να μετρήσει δύο φορές την ίδια συνθήκη.) Το πρόβλημά τους είναι να βρουν ποιες συνθήκες θα πρέπει να μετρήσουν με κάθε αερόστατο.

Παράδειγμα. Υποθέστε ότι $k = 2$, ότι υπάρχουν $n = 4$ συνθήκες με ετικέτες c_1, c_2, c_3, και c_4, και ότι υπάρχουν $m = 4$ αερόστατα που μπορούν να μετρήσουν συνθήκες, υπό τον περιορισμό ότι $S_1 = S_2 = \{c_1, c_2, c_3\}$ και $S_3 = S_4 = \{c_1, c_3, c_4\}$. Στην περίπτωση αυτή ένας πιθανός τρόπος για να εξασφαλίσουμε ότι η κάθε συνθήκη θα μετρηθεί τουλάχιστον $k = 2$ φορές είναι να έχουμε

- το αερόστατο 1 να μετρά τις συνθήκες c_1, c_2,
- το αερόστατο 2 να μετρά τις συνθήκες c_2, c_3,
- το αερόστατο 3 να μετρά τις συνθήκες c_3, c_4, και
- το αερόστατο 4 να μετρά τις συνθήκες c_1, c_4.

(α) Δώστε έναν αλγόριθμο πολυωνυμικού χρόνου ο οποίος θα παίρνει ως είσοδο ένα στιγμιότυπο του προβλήματος (τις n συνθήκες, τα σύνολα S_i για καθένα από τα m αερόστατα, και την παράμετρο k) και θα αποφασίζει αν υπάρχει τρόπος να μετρηθεί η κάθε συνθήκη από k διαφορετικά αερόστατα, όπου το κάθε αερόστατο θα μετρά το πολύ δύο συνθήκες.

(β) Δείξατε στους φίλους σας τη λύση την οποία υπολόγισε ο αλγόριθμός σας από το (α), και προς μεγάλη σας έκπληξη σας είπαν "Αυτό δεν γίνεται με τίποτα — η μία από τις συνθήκες μετριέται έτσι μόνο από αερόστατα ενός μόνο υπεργολάβου." Δεν σας είχαν προηγουμένως αναφέρει τίποτα σχετικά με υπεργολάβους· αποδεικνύεται όμως ότι υπάρχει ένα ακόμα στοιχείο το οποίο ξέχασαν να σας πουν…

Το καθένα από τα αερόστατα παράγεται από έναν από τους τρεις διαφορετικούς υπεργολάβους που συμμετέχουν στο πείραμα. Μια απαίτηση του πειράματος είναι ότι δεν θα υπάρχει καμία συνθήκη για την οποία και οι k μετρήσεις προέρχονται από έναν μόνο υπεργολάβο.

Για παράδειγμα, υποθέστε ότι το αερόστατο 1 προέρχεται από τον πρώτο υπεργολάβο, τα αερόστατα 2 και 3 από το δεύτερο υπεργολάβο, και το αερόστατο 4 από τον τρίτο υπεργολάβο. Στην περίπτωση αυτή η προηγούμενη λύση μας δεν δουλεύει πια, επειδή και οι δύο μετρήσεις για τη συνθήκη c_3 πραγματοποιούνται από αερόστατα του δεύτερου υπεργολάβου. Μπορούμε όμως να χρησιμοποιήσουμε τα αερόστατα 1 και 2 για να μετρήσουμε τις συνθήκες c_1, c_2 και να χρησιμοποιήσουμε τα αερόστατα 3 και 4 για να μετρήσουμε τις συνθήκες c_3, c_4.

Εξηγήστε με ποιον τρόπο μπορείτε να τροποποιήσετε τον αλγόριθμο πολυωνυμικού χρόνου της άσκησης (α) σε ένα νέο αλγόριθμο ο οποίος θα αποφασίζει αν υπάρχει λύση που να ικανοποιεί όλες τις απαιτήσεις της άσκησης (α), συν τη νέα απαίτηση σχετικά με τους υπεργολάβους.

21. Βοηθάτε στην οργάνωση μιας τάξης στο πανεπιστήμιο, όπου έχει αποφασιστεί να δοθούν σε όλους τους φοιτητές φορητοί υπολογιστές ασύρματης σύνδεσης για το εξάμηνο. Υπάρχει λοιπόν μια συλλογή από n φορητούς υπολογιστές με δυνατότητα ασύρματης σύνδεσης· υπάρχει επίσης μια συλλογή από n *σημεία ασύρματης πρόσβασης*, στα οποία μπορεί να συνδεθεί ένας φορητός υπολογιστής όταν είναι εντός εμβέλειας.

Οι φορητοί υπολογιστές είναι αυτή τη στιγμή διάσπαρτοι σε ολόκληρο το πανεπιστήμιο· ο φορητός υπολογιστής ℓ βρίσκεται εντός εμβέλειας για ένα *σύνολο* S_ℓ από σημεία πρόσβασης. Θα υποθέσουμε ότι ο κάθε φορητός υπολογιστής βρίσκεται εντός εμβέλειας για ένα τουλάχιστον σημείο πρόσβασης (έτσι τα σύνολα S_ℓ δεν μπορεί να είναι κενά)· θα υποθέσουμε επίσης ότι το κάθε σημείο πρόσβασης p έχει τουλάχιστον ένα φορητό υπολογιστή εντός εμβέλειας.

Για να εξασφαλίσετε ότι το λογισμικό ασύρματης σύνδεσης λειτουργεί σωστά, θέλετε να βάλετε τους φορητούς υπολογιστές να δοκιμάσουν να έρθουν σε επαφή με τα σημεία πρόσβασης με τέτοιον τρόπο ώστε ο κάθε φορητός υπολογιστής και το κάθε σημείο πρόσβασης να συμμετέχει σε τουλάχιστον μία σύνδεση. Έτσι λέμε ότι το *σύνολο δοκιμής* T είναι μια συλλογή από διατεταγμένα ζεύγη της μορφής (ℓ, p), για ένα φορητό υπολογιστή ℓ και σημείο πρόσβασης p, με τις ιδιότητες ότι

(i) Αν $(\ell, p) \in T$, τότε το ℓ είναι εντός της εμβέλειας του p (δηλαδή, $p \in S_\ell$).

(ii) Ο κάθε φορητός υπολογιστής εμφανίζεται σε τουλάχιστον ένα διατεταγμένο ζεύγος του T.

(iii) Το κάθε σημείο πρόσβασης εμφανίζεται σε τουλάχιστον ένα διατεταγμένο ζεύγος του T.

Με αυτόν τον τρόπο, αν δοκιμάσουμε όλα τα ζευγάρια που προσδιορίζονται στο T, θα μπορούμε να είμαστε βέβαιοι ότι όλοι οι φορητοί υπολογιστές και όλα τα σημεία πρόσβασης έχουν λογισμικό που λειτουργεί σωστά.

Το πρόβλημα είναι το εξής: Με δεδομένα τα σύνολα S_ℓ για τον κάθε φορητό υπολογιστή (δηλαδή, ποιοι φορητοί υπολογιστές βρίσκονται εντός εμβέλειας σε ποια σημεία πρόσβασης) και έναν αριθμό k, να αποφασιστεί αν υπάρχει ένα σύνολο δοκιμής με μέγεθος το πολύ ίσο με k.

Παράδειγμα. Υποθέστε ότι $n = 3$· ο φορητός υπολογιστής 1 βρίσκεται εντός εμβέλειας για τα σημεία πρόσβασης 1 και 2· ο φορητός υπολογιστής 2 βρίσκεται εντός εμβέλειας για το σημείο πρόσβασης 2· και ο φορητός υπολογιστής 3 βρίσκεται εντός εμβέλειας για τα σημεία πρόσβασης 2 και 3. Τότε το σύνολο ζευγαριών *(φορητός 1, σημείο πρόσβασης 1), (φορητός 2, σημείο πρόσβασης 2), (φορητός 3, σημείο πρόσβασης 3)*

σχηματίζει ένα σύνολο δοκιμής με μέγεθος 3.

(α) Δώστε ένα παράδειγμα περίπτωσης του προβλήματος για το οποίο δεν υπάρχει σύνολο δοκιμής με μέγεθος n. (Θυμηθείτε ότι ο κάθε φορητός υπολογιστής βρίσκεται εντός εμβέλειας για τουλάχιστον ένα σημείο πρόσβασης, και κάθε σημείο πρόσβασης p έχει εντός εμβέλειας έναν τουλάχιστον φορητό υπολογιστή.)

(β) Δώστε έναν αλγόριθμο πολυωνυμικού χρόνου ο οποίος θα παίρνει ως είσοδο ένα στιγμιότυπο του προβλήματος (συμπεριλαμβανομένης της παραμέτρου k) και θα αποφασίζει αν υπάρχει σύνολο δοκιμής με μέγεθος το πολύ ίσο με k.

22. Έστω ότι το M είναι ένας πίνακας $n \times n$ με κάθε στοιχείο του ίσο με 0 ή 1. Έστω ότι το m_{ij} συμβολίζει το στοιχείο της γραμμής i και στήλης j. Το *διαγώνιο στοιχείο* είναι ένα στοιχείο της μορφής m_{ii} για κάποια τιμή του i.

Η *αντιμετάθεση* των γραμμών i και j στον πίνακα M δηλώνει την ακόλουθη ενέργεια: αντιμεταθέτουμε τις τιμές m_{ik} και m_{jk} για $k = 1, 2, ..., n$. Με ανάλογο τρόπο ορίζεται και η αντιμετάθεση στηλών.

Λέμε ότι ο πίνακας M είναι *αναδιατάξιμος* αν είναι δυνατόν να αντιμεταθέσουμε κάποια από τα ζευγάρια των γραμμών του πίνακα και κάποια από τα ζευγάρια των στηλών του (με οποιαδήποτε σειρά) έτσι ώστε, μετά από όλες τις αντιμεταθέσεις, όλες τα διαγώνια στοιχεία του πίνακα M να είναι ίσα με 1.

(α) Δώστε ένα παράδειγμα πίνακα M ο οποίος δεν είναι αναδιατάξιμος, αλλά στον οποίο τουλάχιστον ένα στοιχείο σε κάθε γραμμή και στήλη του είναι ίσο με 1.

(β) Κατασκευάστε έναν αλγόριθμο πολυωνυμικού χρόνου ο οποίος προσδιορίζει αν ένας πίνακας M με στοιχεία 0-1 είναι αναδιατάξιμος.

23. Υποθέστε ότι εξετάζουμε ένα δίκτυο ροής G με προέλευση s και απόληξη t, και θέλουμε να μπορέσουμε να εκφράσουμε κάτι που μοιάζει με την ακόλουθη διαισθητική έννοια: Κάποιοι κόμβοι βρίσκονται ξεκάθαρα στην "πλευρά προέλευσης" των κύριων περιορισμών χωρητικότητας· κάποιοι κόμβοι βρίσκονται ξεκάθαρα στην "πλευρά απόληξης" των κύριων περιορισμών χωρητικότητας· και κάποιοι κόμβοι βρίσκονται ενδιάμεσα. Όμως το G μπορεί να έχει πολλές ελάχιστες αποκοπές, έτσι θα πρέπει να είμαστε προσεκτικοί στον τρόπο με τον οποίο θα προσπαθήσουμε αν αποδώσουμε με ακρίβεια αυτή την ιδέα.

Ο ακόλουθος είναι ένας τρόπος διαίρεσης των κόμβων G σε τρεις τέτοιες κατηγορίες.

- Λέμε ότι ο κόμβος v είναι *στην αρχή της ροής* (upstream) αν, για όλες τις ελάχιστες αποκοπές s-t (A, B), έχουμε $v \in A$ — δηλαδή, το v βρίσκεται στην πλευρά όλων των ελάχιστων αποκοπών που περιλαμβάνει την προέλευση.

- Λέμε ότι ο κόμβος v είναι *στο τέλος της ροής* (downstream) αν, για όλες τις ελάχιστες αποκοπές s-t (A, B), έχουμε $v \in B$ — δηλαδή, το v βρίσκεται στην πλευρά απόληξης σε όλες τις ελάχιστες αποκοπές.
- Λέμε ότι ο κόμβος v είναι *κεντρικός* (central) όταν δεν είναι ούτε στην αρχή ούτε στο τέλος της ροής· υπάρχει τουλάχιστον μία ελάχιστη αποκοπή s-t (A, B) για την οποία $v \in A$, και τουλάχιστον μία ελάχιστη αποκοπή s-t (A', B') για την οποία $v \in B'$.

Κατασκευάστε έναν αλγόριθμο ο οποίος θα παίρνει ένα δίκτυο ροής G και θα κατηγοριοποιεί όλους τους κόμβους ως κόμβους αρχής ροής, τέλους ροής, ή κεντρικούς. Ο χρόνος εκτέλεσης του αλγόριθμού σας θα πρέπει να είναι κατά ένα σταθερό συντελεστή μεγαλύτερος από το χρόνο που απαιτείται για τον υπολογισμό *μίας* μέγιστης ροής.

24. Έστω ότι το $G = (V, E)$ είναι ένα κατευθυνόμενο γράφημα, με προέλευση $s \in V$, απόληξη $t \in V$, και μη αρνητικές χωρητικότητες ακμών $\{c_e\}$. Κατασκευάστε έναν αλγόριθμο πολυωνυμικού χρόνου ο οποίος θα αποφασίζει αν το G έχει μια *μοναδική* ελάχιστη αποκοπή s-t (δηλαδή, μια αποκοπή s-t με χωρητικότητα μικρότερη από εκείνη όλων των άλλων αποκοπών s-t).

25. Υποθέστε ότι ζείτε σε ένα μεγάλο διαμέρισμα μαζί με πολλούς φίλους. Κατά τη διάρκεια ενός έτους, υπάρχουν πολλές περιπτώσεις στις οποίες ένας από εσάς πληρώνει για κάποιο έξοδο το οποίο είναι κοινόχρηστο από ένα υποσύνολο του διαμερίσματος, αναμένοντας ότι όλα θα αντισταθμιστούν δίκαια μέχρι το τέλος του έτους. Για παράδειγμα, ένας από εσάς μπορεί να πληρώσει ολόκληρο το λογαριασμό τηλεφώνου για ένα δεδομένο μήνα, κάποιος άλλος μπορεί να κάνει τις αγορές φαγητού στο γειτονικό παντοπωλείο, και ένας τρίτος μπορεί να χρησιμοποιεί την πιστωτική του κάρτα για να καλύψει ολόκληρο το λογαριασμό στο τοπικό Ιταλοϊνδικό εστιατόριο Little Idli.

Σε κάθε περίπτωση, είναι πλέον το τέλος του χρόνου και είναι καιρός να ξεκαθαρίσουν οι λογαριασμοί. Υπάρχουν n άνθρωποι στο διαμέρισμα· και για κάθε διατεταγμένο ζευγάρι (i, j) υπάρχει ένα ποσό $a_{ij} \geq 0$ το οποίο ο i οφείλει στον j, το οποίο έχει συσσωρευτεί κατά τη διάρκεια του χρόνου. Θα απαιτήσουμε ότι για οποιουσδήποτε δύο ανθρώπους i και j, τουλάχιστον η μία από τις ποσότητες a_{ij} ή a_{ji} είναι ίση με 0. Αυτό μπορεί να επιτευχθεί εύκολα με τον ακόλουθο τρόπο: Αν ο i χρωστά στον j ένα θετικό ποσό χρημάτων x και ο j χρωστά στον i ένα θετικό ποσό χρημάτων $y < x$, τότε μπορούμε να αφαιρέσουμε και από τις δύο ποσότητες το y και να έχουμε $a_{ij} = x - y$ και $a_{ji} = 0$. Με βάση αυτές τις ποσότητες, ορίζουμε τώρα ότι το *υπόλοιπο* ενός ατόμου i είναι η συνολική ποσότητα χρημάτων που οι άλλοι χρωστούν στον i, μείον το άθροισμα των χρημάτων που ο i χρωστά στους υπόλοιπους. (Σημειώστε ότι το υπόλοιπο μπορεί να είναι θετικό, αρνητικό, ή μηδενικό.)

Για να επανέλθουν όλα τα υπόλοιπα στο 0, έτσι ώστε όλοι να φύγουν ευχαριστημένοι, κάποια από τα άτομα θα πρέπει να δώσουν επιταγές σε άλλα άτομα· με άλλα λόγια, για κάθε διατεταγμένο ζευγάρι (i, j), ο i θα δώσει στον j μια επιταγή για $b_{ij} > 0$ χρήματα. Θα λέμε ότι ένα σύνολο επιταγών συνιστά *συνδιαλλαγή* αν, για κάθε άτομο i, η συνολική αξία των επιταγών τις οποίες λαμβάνει ο i μείον τη συνολική αξία των επιταγών που δίνει ο i είναι ίση με το υπόλοιπο του i. Τέλος, εσείς και οι φίλοι σας πιστεύετε ότι δεν είναι σωστό ο i να δώσει στον j επιταγή όταν ο i δεν χρωστά στην πραγματικότητα χρήματα στον j, έτσι λέμε ότι η συνδιαλλαγή είναι *συνεπής* αν, όποτε ο i δίνει μια επιταγή στον j, ισχύει $a_{ij} > 0$.

Δείξτε ότι, για οποιοδήποτε σύνολο ποσοτήτων a_{ij}, υπάρχει πάντα μια συνεπής συνδιαλλαγή στην οποία δίνονται το πολύ $n - 1$ επιταγές· για να το επιτύχετε αυτό, κατασκευάστε έναν αλγόριθμο πολυωνυμικού χρόνου για τον υπολογισμό μιας τέτοιας συνδιαλλαγής.

26. Μπορεί να διαπιστώσει κανείς ότι τα κινητά τηλέφωνα λειτουργούν στις αγροτικές κοινότητες παρατηρώντας τα γιγάντια "πιάτα" που ξεπροβάλλουν συχνά μέσα από τα χωράφια και τα αγροκτήματα. Ας εξετάσουμε ένα πολύ απλοποιημένο μοντέλο του δικτύου κινητής τηλεφωνίας σε μια αραιοκατοικημένη περιοχή.

Μας δίνονται οι θέσεις n *σταθμών βάσης*, που προσδιορίζονται ως σημεία b_1, ..., b_n στο επίπεδο. Μας δίνονται επίσης οι θέσεις n κινητών τηλεφώνων, τα οποία προσδιορίζονται ως σημεία p_1, ..., p_n στο επίπεδο. Τέλος, μας δίνεται μια *παράμετρος εμβέλειας* $\Delta > 0$. Θα αποκαλούμε αυτό το σύνολο κινητών τηλεφώνων *πλήρως συνδεδεμένο* αν είναι δυνατό να αντιστοιχίσουμε το κάθε τηλέφωνο σε ένα σταθμό βάσης έτσι ώστε

- Το κάθε τηλέφωνο να έχει αντιστοιχιστεί σε διαφορετικό σταθμό βάσης, και
- Αν το τηλέφωνο στη θέση p_i έχει αντιστοιχιστεί στο σταθμό βάσης b_j, τότε η απόσταση σε ευθεία γραμμή μεταξύ των σημείων p_i και b_j είναι το πολύ ίση με Δ.

Υποθέστε ότι ο ιδιοκτήτης του κινητού τηλεφώνου p_1 αποφάσισε να πάει μια βόλτα, ταξιδεύοντας συνεχώς για συνολικά z μονάδες απόστασης προς τα ανατολικά. Καθώς μετακινείται το κινητό τηλέφωνο, μπορεί να χρειάζεται να ενημερωθεί η αντιστοίχιση τηλεφώνων σε σταθμούς βάσης (πιθανώς πολλές φορές) έτσι ώστε το σύνολο των τηλεφώνων να παραμένει πλήρως συνδεδεμένο.

Δώστε έναν αλγόριθμο πολυωνυμικού χρόνου που να αποφασίζει αν είναι δυνατό να διατηρούμε πάντα πλήρως συνδεδεμένο το σύνολο των κινητών τηλεφώνων κατά τη διάρκεια της μετακίνησης αυτού του ενός τηλεφώνου. (Θα πρέπει να υποθέσετε ότι τα υπόλοιπα κινητά τηλέφωνα δεν αλλάζουν θέση κατά τη διάρκεια αυτής της περιόδου.) Αν αυτό είναι εφικτό, ο αλγόριθμος θα πρέπει να αναφέρει μια ακολουθία αντιστοιχίσεων κινητών τηλεφώνων σε σταθμούς βάσης η οποία θα είναι επαρκής για τη διατήρηση της πλήρους σύνδεσης· αν δεν είναι εφικτό, θα πρέπει να αναφέρετε το σημείο της διαδρομής του μετακινούμενου τηλεφώνου στο οποίο δεν μπορεί να διατηρηθεί η πλήρης σύνδεση.

Αν είναι δυνατόν, θα πρέπει να κάνετε τον αλγόριθμό σας να εκτελείται σε χρόνο $O(n^3)$.

Παράδειγμα. Υποθέστε ότι έχουμε τηλέφωνα στα σημεία $p_1 = (0, 0)$ και $p_2 = (2, 1)$· έχουμε επίσης τους σταθμούς βάσης $b_1 = (1, 1)$ και $b_2 = (3, 1)$· και $\Delta = 2$.

Εξετάστε τώρα την περίπτωση στην οποία το τηλέφωνο p_1 μετακινείται ανατολικά κατά τέσσερις μονάδες απόστασης, καταλήγοντας στη θέση (4, 0). Στην περίπτωση αυτή είναι δυνατό να διατηρήσουμε την πλήρη σύνδεση των τηλεφώνων κατά τη διάρκεια της μετακίνησης. Ξεκινάμε αντιστοιχίζοντας το p_1 στο b_1 και το p_2 στο b_2, και κατά τη διάρκεια της κίνησης αντιστοιχίζουμε το p_1 στο b_2 και το p_2 στο b_1 — για παράδειγμα, όταν το p_1 περάσει το σημείο (2, 0).

27. Κάποιοι από τους φίλους σας αποφάσισαν ότι χρειάζονται πραγματικά λίγο χρόνο μπροστά από τους υπολογιστές τους, και η πρωινή οδήγηση στην Αττική Οδό έμοιαζε μοναδική λύση. Έτσι αποφάσισαν να αλληλομεταφέρονται εναλλάξ με τα αυτοκίνητά τους.

Δυστυχώς, όλοι τους σιχαίνονται την οδήγηση, και έτσι θέλουν να εξασφαλίσουν ότι η όποια συμφωνία αλληλομεταφοράς θα είναι δίκαιη και δεν θα υπερφορτώνει κανέναν τους με υπερβολική οδήγηση. Τα απλά συστήματα της κυκλικής εναλλαγής είναι εκτός συζήτησης, επειδή κανένας τους δεν πηγαίνει στη δουλειά καθημερινά και έτσι το υποσύνολο των ατόμων που βρίσκονται μέσα στο αυτοκίνητο ποικίλλει από ημέρα σε ημέρα.

Να ένας τρόπος ορισμού του *δίκαιου συστήματος*. Έστω ότι τα άτομα έχουν ετικέτες $S = \{p_1, ..., p_k\}$. Λέμε ότι η *συνολική υποχρέωση οδήγησης* του p_j για ένα σύνολο ημερών είναι το αναμενόμενο πλήθος φορών που ο p_j θα είχε οδηγήσει αν είχε επιλεγεί ομοιόμορφα στην τύχη ο οδηγός μεταξύ των ατόμων που πηγαίνουν στη δουλειά κάθε μέρα. Πιο συγκεκριμένα, υποθέστε ότι το σχέδιο αλληλομεταφοράς διαρκεί για d ημέρες και κατά την ημέρα i πηγαίνει στη δουλειά ένα υποσύνολο $S_i \subseteq S$ των ατόμων. Στην περίπτωση αυτή ο παραπάνω ορισμός της συνολικής υποχρέωσης οδήγησης Δ_j για τον p_j μπορεί να γραφτεί ως $\Delta_j = \sum_{i:p_j \in S_i} \frac{1}{|S_i|}$. Ιδεατά, θα θέλαμε να απαιτήσουμε ότι ο p_j θα οδηγήσει το πολύ Δ_j φορές· δυστυχώς, όμως, το Δ_j μπορεί να μην είναι ακέραιος.

Ας πούμε λοιπόν ότι το *χρονοδιάγραμμα οδήγησης* είναι μια επιλογή οδηγού για κάθε ημέρα — με άλλα λόγια, μια ακολουθία $p_{i1}, p_{i2}, ..., p_{id}$ με $p_{it} \in S_t$ — και ότι το *δίκαιο χρονοδιάγραμμα οδήγησης* είναι ένα χρονοδιάγραμμα στο οποίο ο p_j επιλέγεται ως οδηγός το πολύ για $\lceil \Delta_j \rceil$ ημέρες.

(α) Αποδείξτε ότι για οποιαδήποτε ακολουθία συνόλων $S_1, ..., S_d$, υπάρχει ένα δίκαιο χρονοδιάγραμμα οδήγησης.

(β) Κατασκευάστε έναν αλγόριθμο ο οποίος θα υπολογίζει ένα δίκαιο χρονοδιάγραμμα οδήγησης σε χρόνο πολυωνυμικό ως προς τα k και d.

28. Μια ομάδα φοιτητών αποφάσισε να προσθέσει μερικές λειτουργίες στο ηλεκτρονικό Σύστημα Διαχείρισης Σπουδών (Course Management System, CMS) του πανεπιστημίου Cornell, έτσι ώστε να χειρίζεται πτυχές του σχεδιασμού σπουδών που δεν καλύπτονται αυτή τη στιγμή από το πρόγραμμα. Ξεκίνησαν με μια υπομονάδα που βοηθά στο χρονοπρογραμματισμό ωρών γραφείου στην αρχή του εξαμήνου.

Το αρχικό τους πρωτότυπο δουλεύει με τον εξής τρόπο. Το χρονοδιάγραμμα ωρών γραφείου θα είναι το ίδιο από τη μία εβδομάδα στην επόμενη, έτσι είναι αρκετό να εστιαστούμε στο πρόβλημα χρονοπρογραμματισμού για μία μόνο εβδομάδα. Ο υπεύθυνος του μαθήματος εισάγει μια συλλογή από μη επικαλυπτόμενα διαστήματα της μίας ώρας $I_1, I_2, ..., I_k$ κατά τα οποία θα μπορούν οι βοηθοί να βρίσκονται στο γραφείο· το τελικό χρονοδιάγραμμα ωρών γραφείου θα αποτελείται από ένα υποσύνολο από κάποια, αλλά συνήθως όχι όλα, αυτά τα χρονικά διαστήματα. Στη συνέχεια καθένας από τους βοηθούς εισάγει το δικό του εβδομαδιαίο πρόγραμμα, δείχνοντας τις ώρες κατά τις οποίες θα είναι διαθέσιμος για τη λειτουργία του γραφείου.

Τέλος, ο υπεύθυνος του μαθήματος προσδιορίζει, με τις παραμέτρους a, b, και c, ότι θέλει ο κάθε βοηθός να βρίσκεται στο γραφείο μεταξύ a και b ώρες ανά εβδομάδα, καθώς και ότι επιθυμεί να υπάρχουν συνολικά ακριβώς c ώρες γραφείου κατά τη διάρκεια της εβδομάδας.

Το πρόβλημα, λοιπόν, είναι η αντιστοίχιση καθενός από τους βοηθούς σε κάποια από τα χρονικά διαστήματα ωρών γραφείου, έτσι ώστε ο βοηθός να είναι διαθέσιμος το αντίστοιχο χρονικό διάστημα αλλά και να πραγματοποιείται ο επιθυμητός αριθμός ωρών γραφείου. (Σε κάθε ώρα γραφείου θα πρέπει να υπάρχει μόνο ένας βοηθός στο γραφείο.)

Παράδειγμα. Υποθέστε ότι υπάρχουν πέντε δυνατά χρονικά διαστήματα για ώρες γραφείου:

I_1 = *Δευτέρα 3-4 μ.μ.*· I_2 = *Τρίτη 1-2 μ.μ.*· I_3 = *Τετάρτη 10-11 π.μ.*·
I_4 = *Τετάρτη 3-4 μ.μ.*· και I_5 = *Πέμπτη 10-11 π.μ.*.

Υπάρχουν δύο βοηθοί· ο πρώτος μπορεί να πραγματοποιεί τις ώρες γραφείου οποιαδήποτε στιγμή τις Δευτέρες και τις Τετάρτες απόγευμα, ενώ ο δεύτερος είναι διαθέσιμος για ώρες γραφείου τις Τρίτες, τις Τετάρτες, και τις Πέμπτες. (Γενικότερα, η διαθεσιμότητα των βοηθών μπορεί να είναι πιο περίπλοκη, όμως θέλουμε να διατηρήσουμε αυτό το παράδειγμα απλό.) Τέλος, ο κάθε βοηθός θα πρέπει να δουλέψει μεταξύ $a = 1$ και $b = 2$ ώρες γραφείου, και θέλουμε συνολικά ακριβώς $c = 3$ ώρες γραφείου ανά εβδομάδα.

Μια πιθανή λύση είναι να αντιστοιχίσουμε τον πρώτο βοηθό στο χρονικό διάστημα I_1 και το δεύτερο βοηθό στα χρονικά διαστήματα I_2 και I_5.

(α) Δώστε έναν αλγόριθμο πολυωνυμικού χρόνου ο οποίος θα παίρνει ως είσοδο ένα στιγμιότυπο του προβλήματος (τα χρονικά διαστήματα, τα προγράμματα των βοηθών, και τις παραμέτρους a, b, και c) και θα κάνει ένα από τα ακόλουθα δύο πράγματα:

- Θα κατασκευάζει ένα έγκυρο χρονοδιάγραμμα για τις ώρες γραφείου, προσδιορίζοντας ποιος βοηθός θα καλύψει το κάθε χρονικό διάστημα, ή
- Θα αναφέρει (σωστά) ότι δεν υπάρχει έγκυρος τρόπος χρονοπρογραμματισμού των ωρών γραφείου.

(β) Αυτή η λειτουργία χρονοπρογραμματισμού των ωρών γραφείου γίνεται πολύ δημοφιλής, και έτσι το διδακτικό προσωπικό αρχίζει να ζητά ακόμα περισσότερα. Ειδικότερα, παρατήρησαν ότι είναι καλό να έχουμε μεγαλύτερη πυκνότητα στις ώρες γραφείου στις ημέρες που είναι κοντά στις παραδόσεις εργασιών.

Έτσι αυτό που θέλουν είναι να μπορούν να προσδιορίσουν μια παράμετρο *πυκνότητας ωρών γραφείου* για κάθε ημέρα της εβδομάδας: Ο αριθμός d_i καθορίζει ότι θέλουν να έχουν τουλάχιστον d_i ώρες γραφείου σε μια δεδομένη ημέρα i της εβδομάδας.

Για παράδειγμα, υποθέστε ότι προσθέτουμε στο προηγούμενο παράδειγμά μας τον περιορισμό ότι θέλουμε τουλάχιστον μία ώρα γραφείου την Τετάρτη και τουλάχιστον μία ώρα γραφείου την Πέμπτη. Στην περίπτωση αυτή η προηγούμενη λύση μας δεν δουλεύει πια· όμως υπάρχει μια δυνατή λύση όπου αντιστοιχίζουμε τον πρώτο βοηθό στις ώρες γραφείου του χρονικού διαστήματος I_1, ενώ ο δεύτερος βοηθός αναλαμβάνει τις ώρες γραφείου στα χρονικά διαστήματα I_3 και I_5. (Μια άλλη λύση είναι να αναθέσουμε στον πρώτο βοηθό τις ώρες γραφείου στα χρονικά διαστήματα I_1 και I_4, και στο δεύτερο βοηθό τις ώρες γραφείου στο χρονικό διάστημα I_5.)

Δώστε έναν αλγόριθμο πολυωνυμικού χρόνου ο οποίος θα υπολογίζει τα χρονοδιαγράμματα ωρών γραφείου με αυτό το πιο σύνθετο σύνολο περιορισμών. Ο αλγόριθμος θα πρέπει είτε να κατασκευάζει ένα χρονοδιάγραμμα, ή να αναφέρει (σωστά) ότι δεν υπάρχει κανένα χρονοδιάγραμμα.

29. Κάποιοι από τους φίλους σας αποφοίτησαν πρόσφατα και ίδρυσαν μια μικρή εταιρεία, την οποία αυτή τη στιγμή λειτουργούν από το γκαράζ των γονιών τους. Βρίσκονται τώρα στη διαδικασία της μεταφοράς όλου του λογισμικού από ένα παλιό σύστημα σε ένα νέο, αναβαθμισμένο σύστημα· και αντιμετωπίζουν το ακόλουθο πρόβλημα.

Έχουν μια συλλογή από n εφαρμογές λογισμικού, $\{1, 2, ..., n\}$, που λειτουργούν στο παλιό σύστημα, και θα ήθελαν να μεταφέρουν κάποιες από αυτές τις εφαρμογές στο νέο σύστημα. Αν μεταφέρουν την εφαρμογή i στο νέο σύστημα, αναμένουν ένα καθαρό (χρηματικό) κέρδος $b_i \geq 0$. Οι διάφορες εφαρμογές λογισμικού αλληλεπιδρούν μεταξύ τους· αν οι εφαρμογές i και j έχουν εκτενή αλληλεπίδραση, τότε η εταιρεία θα έχει κόστος αν μεταφέρει μόνο την εφαρμογή i ή την εφαρμογή j στο νέο σύστημα, αλλά όχι και τις δύο· ας συμβολίσουμε αυτό το κόστος με το $x_{ij} \geq 0$.

Έτσι, αν η κατάσταση ήταν πραγματικά τόσο απλή, οι φίλοι σας θα μετέφεραν απλώς όλες τις n εφαρμογές, επιτυγχάνοντας συνολικό κέρδος $\sum_i b_i$. Δυστυχώς, υπάρχει ένα πρόβλημα…

Εξαιτίας κάποιων μικρών αλλά θεμελιωδών ασυμβατοτήτων μεταξύ των δύο συστημάτων, δεν υπάρχει τρόπος μεταφοράς της εφαρμογής 1 στο νέο σύστημα· θα πρέπει να παραμείνει στο παλιό σύστημα. Παρόλα αυτά, μπορεί και πάλι να είναι χρήσιμη η μεταφορά κάποιων από τις υπόλοιπες εφαρμογές, ώστε να επωφεληθούν από το αντίστοιχο κέρδος αλλά και να πληρώσουν το κόστος ως προς την αλληλεπίδραση μεταξύ εφαρμογών διαφορετικών συστημάτων.

Να λοιπόν το ερώτημα που σας έθεσαν: Ποιες από τις υπόλοιπες εφαρμογές, αν υπάρχουν τέτοιες, θα πρέπει να μεταφέρουν; Κατασκευάστε έναν αλγόριθμο πολυωνυμικού χρόνου για να βρείτε ένα σύνολο $S \subseteq \{2, 3, ..., n\}$ για το οποίο μεγιστοποιείται το άθροισμα των κερδών μείον το κόστος από τη μεταφορά των εφαρμογών του συνόλου S στο νέο σύστημα.

30. Εξετάστε μια παραλλαγή του προηγούμενου προβλήματος. Στο νέο σενάριο μπορεί δυνητικά να μεταφερθεί οποιαδήποτε εφαρμογή, όμως τώρα κάποια από τα οφέλη b_i από τη μετακίνηση στο νέο σύστημα είναι στην πραγματικότητα *αρνητικά*: Αν $b_i < 0$, τότε είναι προτιμότερο (κατά μία ποσότητα που προσδιορίζεται από το b_i) να αφήσουμε την εφαρμογή i στο παλιό σύστημα. Και πάλι, δώστε έναν αλγόριθμο πολυωνυμικού χρόνου που θα βρίσκει ένα σύνολο $S \subseteq \{1, 2, ..., n\}$ για το οποίο θα μεγιστοποιείται το άθροισμα από τα οφέλη μείον το κόστος μετακίνησης των εφαρμογών του S στο νέο σύστημα.

31. Κάποιοι από τους φίλους σας δουλεύουν στη μικρή εταιρεία υψηλής τεχνολογίας WebExodus. Ένα συνηθισμένο αστείο μεταξύ των υπαλλήλων αυτής της εταιρείας είναι ότι στο πίσω μέρος των γραφείων της εταιρείας αφιερώνεται λιγότερος χώρος για διακομιστές υψηλής ταχύτητας, παρά για άδεια κουτιά από υπολογιστές τα οποία στοιβάζονται εκεί για την περίπτωση που θα πρέπει να σταλεί κάτι στον προμηθευτή για συντήρηση.

Λίγες μέρες πριν παρέλαβαν ένα μεγάλο φορτίο με οθόνες υπολογιστών, που η καθεμία βρισκόταν στο δικό της μεγάλο κουτί· και αφού υπήρχαν πολλά διαφορετικά είδη οθονών στο φορτίο αυτό, το κουτιά δεν έχουν όλα το ίδιο μέγεθος. Κάποιοι από τους εργαζόμενους πέρασαν μερικές ώρες το πρωί προσπαθώντας να βρουν πώς θα αποθηκεύσουν όλα αυτά τα πράγματα, συνειδητοποιώντας φυσικά ότι τα κουτιά θα έπιαναν λιγότερο χώρο αν μπορούσε να είναι *ένθετα* το ένα μέσα στο άλλο.

Υποθέστε ότι το κάθε κουτί i είναι ένα ορθογώνιο παραλληλεπίπεδο με μήκη πλευρών (i_1, i_2, i_3)· υποθέστε επίσης ότι το μήκος κάθε πλευράς είναι πάντα μεταξύ μισού και ενός μέτρου. Γεωμετρικά, γνωρίζετε τι σημαίνει η ένθεση του ενός κουτιού μέσα στο άλλο: κάτι τέτοιο είναι δυνατό αν μπορείτε να περιστρέψετε το μικρότερο κουτί έτσι ώστε να χωράει μέσα στο μεγαλύτερο σε όλες τις διαστάσεις του. Με τυπικούς όρους, μπορούμε να πούμε ότι έχουμε τη δυνατότητα να *ενθέσουμε* το κουτί i με διαστάσεις (i_1, i_2, i_3) μέσα στο κουτί j με διαστάσεις (j_1, j_2, j_3) αν υπάρχει μια αντιμετάθεση a, b, c των διαστάσεων $\{1, 2, 3\}$ έτσι ώστε $i_a < j_1$ και $i_b < j_2$ και $i_c < j_3$. Προφανώς, η ένθεση είναι αναδρομική: Αν μπορούμε να ενθέσουμε το κουτί i μέσα στο j και το κουτί j μέσα στο k, τότε αν τοποθετήσουμε το κουτί i μέσα στο j και όλα αυτά μέσα στο k μόνο το κουτί k θα είναι ορατό. Λέμε ότι η *διευθέτηση ένθεσης* για ένα σύνολο n κουτιών είναι μια ακολουθία ενεργειών με τις οποίες το κουτί i τοποθετείται μέσα σε ένα άλλο κουτί j στο οποίο χωράει· και αν υπήρχαν ήδη άλλα κουτιά μέσα στο i, τότε καταλήγουν και αυτά μέσα στο κουτί j. (Παρατηρήστε επίσης το εξής: Αφού τα μήκη πλευρών του κουτιού i είναι όλα μεγαλύτερα από μισό μέτρο, και αφού τα μήκη πλευρών του j είναι όλα μικρότερα από ένα μέτρο, το κουτί i θα καταλάβει περισσότερο από το μισό της κάθε διάστασης του j, και έτσι μετά την ένθεση του κουτιού i μέσα στο κουτί j δεν μπορεί να τοποθετηθεί τίποτα άλλο μέσα στο κουτί j.) Λέμε ότι ένα κουτί k είναι *ορατό* σε μια διευθέτηση ένθεσης αν η ακολουθία ενεργειών δεν έχει αποτέλεσμα την τοποθέτηση αυτού του κουτιού μέσα σε κάποιο άλλο κουτί.

Να λοιπόν το πρόβλημα που αντιμετωπίζουν οι άνθρωποι στη WebExodus: Αφού μόνο τα ορατά κουτιά καταλαμβάνουν χώρο, πώς θα πρέπει να επιλεγεί μια διευθέτηση ένθεσης έτσι ώστε να ελαχιστοποιηθεί το *πλήθος* των ορατών κουτιών;

Δώστε έναν αλγόριθμο πολυωνυμικού χρόνου για την επίλυση αυτού του προβλήματος.

Παράδειγμα. Υποθέστε ότι υπάρχουν τρία κουτιά με διαστάσεις (.6, .6, .6), (.75, .75, .75), και (.9, .7, .7). Το πρώτο κουτί μπορεί να τοποθετηθεί είτε μέσα στο δεύτερο κουτί, είτε μέσα στο τρίτο κουτί· όμως σε οποιαδήποτε διευθέτηση ένθεσης θα είναι ορατά και το δεύτερο και το τρίτο κουτί. Έτσι ο ελάχιστος δυνατός αριθμός ορατών κουτιών είναι δύο, και μια λύση που επιτυγχάνει αυτόν τον αριθμό είναι να τοποθετήσουμε το πρώτο κουτί μέσα στο δεύτερο.

32. Με δεδομένο ένα γράφημα $G = (V, E)$ και ένα φυσικό αριθμό k, μπορούμε να ορίσουμε μια σχέση $\stackrel{G,k}{\rightarrow}$ για ζευγάρια κορυφών του G με τον εξής τρόπο. Αν $x, y \in V$, λέμε ότι $x \stackrel{G,k}{\rightarrow} y$ αν στο G υπάρχουν k αμοιβαία ασύνδετες ως προς τις ακμές διαδρομές από το x προς το y.

Είναι αληθές ότι για κάθε γράφημα G και κάθε $k \geq 0$ η σχέση $\stackrel{G,k}{\rightarrow}$ διαθέτει τη μεταβατική ιδιότητα; Με άλλα λόγια, ισχύει πάντα ότι, αν $x \stackrel{G,k}{\rightarrow} y$ και $y \stackrel{G,k}{\rightarrow} z$, τότε έχουμε $x \stackrel{G,k}{\rightarrow} z$; Δώστε μια απόδειξη ή ένα αντιπαράδειγμα.

33. Έστω ότι το $G = (V, E)$ είναι ένα κατευθυνόμενο γράφημα, και υποθέστε ότι για κάθε κόμβο v ο αριθμός των ακμών που εισέρχονται στον κόμβο v είναι ίσος με τον αριθμό των ακμών που εξέρχονται από τον κόμβο v. Με άλλα λόγια, για όλα τα v ισχύει

$$|\{(u, v) : (u, v) \in E\}| = |\{(v, w) : (v, w) \in E\}|.$$

Έστω ότι και x και y είναι δύο κόμβοι στο G, και υποθέστε ότι υπάρχουν k αμοιβαία ασύνδετες ως προς τις ακμές διαδρομές από το x προς το y. Υπό τις συνθήκες αυτές, προκύπτει ότι υπάρχουν k αμοιβαία ασύνδετες ως προς τις ακμές διαδρομές από το y προς το x; Δώστε μια απόδειξη ή ένα αντιπαράδειγμα με εξήγηση.

34. Τα *αδόμητα δίκτυα* (Ad hoc networks), τα οποία σχηματίζονται από ασύρματες συσκευές χαμηλής ισχύος, έχουν προταθεί ως λύση για καταστάσεις όπως οι φυσικές καταστροφές, όπου οι συντονιστές της προσπάθειας διάσωσης μπορεί να θέλουν να παρακολουθήσουν τις συνθήκες σε μια δυσπρόσιτη περιοχή. Η ιδέα είναι ότι θα μπορούσε μια μεγάλη συλλογή από τέτοιες ασύρματες συσκευές να ρίχνεται στην περιοχή από ένα αεροπλάνο, και μετά οι συσκευές να διευθετούνται σε ένα λειτουργικό δίκτυο.

Σημειώστε ότι μιλάμε (α) για σχετικά φθηνές συσκευές οι οποίες (β) ρίχνονται με το αεροπλάνο σε (γ) μια επικίνδυνη περιοχή· για τους λόγους (α), (β), και (γ) γίνεται απαραίτητο να συμπεριλάβουμε πρόβλεψη για την αντιμετώπιση των αστοχιών (βλαβών) σε σημαντικό αριθμό από τους κόμβους.

Θα θέλαμε, αν μία από τις συσκευές v διαγνώσει ότι διατρέχει κίνδυνο αστοχίας, να μεταδίδει μια αναπαράσταση της τρέχουσας κατάστασής της σε κάποια άλλη συσκευή του δικτύου. Η κάθε συσκευή έχει περιορισμένη εμβέλεια μετάδοσης — ας πούμε ότι μπορεί να επικοινωνήσει με άλλες συσκευές που βρίσκονται σε ακτίνα d μέτρων από αυτή. Επιπρόσθετα, αφού δεν θέλουμε να προσπαθήσει να μεταδώσει την κατάστασή της σε μια συσκευή που έχει ήδη αστοχήσει, θα πρέπει να συμπεριλάβουμε κάποιον πλεονασμό: Η συσκευή v θα πρέπει να έχει ένα σύνολο από k άλλες συσκευές με τις οποίες θα μπορεί δυνητικά να έλθει σε επαφή, όπου η καθεμία από αυτές βρίσκεται σε ακτίνα d μέτρων από αυτή. Θα το ονομάσουμε αυτό *σύνολο εφεδρείας* για τη συσκευή v.

(α) Υποθέστε ότι μας δίνεται ένα σύνολο από n ασύρματες συσκευές, με τη θέση της κάθε συσκευής να αναπαριστάνεται από συντεταγμένες (x, y). Σχεδιάστε έναν αλγόριθμο ο οποίος θα προσδιορίζει αν είναι δυνατή η επιλογή ενός συνόλου εφεδρείας για την κάθε συσκευή (δηλαδή, k άλλων συσκευών που βρίσκονται όλες σε ακτίνα d μέτρων), με την πρόσθετη ιδιότητα ότι, για κάποια παράμετρο b, καμία συσκευή δεν εμφανίζεται στα σύνολα εφεδρείας περισσότερων από b άλλων συσκευών. Ο αλγόριθμος θα πρέπει να δίνει ως έξοδο τα ίδια τα σύνολα εφεδρείας, εφόσον μπορούν να βρεθούν.

(β) Η ιδέα ότι για κάθε ζευγάρι συσκευών v και w υπάρχει μια αυστηρή διάκριση μεταξύ "εντός εμβέλειας" και "εκτός εμβέλειας" είναι μια απλοποιημένη αφαιρετική προσέγγιση. Για να είμαστε πιο ακριβείς, υπάρχει μια συνάρτηση εξασθένησης ισχύος $f(\cdot)$ η οποία προσδιορίζει ότι, για ένα ζευγάρι συσκευών που βρίσκονται σε απόσταση δ, η ισχύς του σήματος που θα μπορούν να επιτύχουν οι συσκευές αυτές κατά την ασύρματη σύνδεσή τους είναι $f(\delta)$. (Θα υποθέσουμε ότι το $f(\delta)$ μειώνεται καθώς αυξάνεται το δ.)

Θέλουμε να ενσωματώσουμε αυτή την κατάσταση στην έννοια των συνόλων εφεδρείας με τον ακόλουθο τρόπο: μεταξύ των k συσκευών που βρίσκονται στο σύνολο εφεδρείας της συσκευής v, θα πρέπει να υπάρχει τουλάχιστον μία συσκευή με την οποία θα έχει επικοινωνία με πολύ υψηλή ισχύ σήματος, τουλάχιστον μία συσκευή όπου η επικοινωνία θα γίνεται με μέτρια ισχύ σήματος, κ.ο.κ. Πιο συγκεκριμένα, έχουμε τιμές $p_1 \geq p_2 \geq \ldots \geq p_k$ έτσι ώστε, αν το σύνολο εφεδρείας για τη συσκευή v αποτελείται από συσκευές σε αποστάσεις $d_1 \leq d_2 \leq \ldots \leq d_k$, τότε θα πρέπει να έχουμε $f(d_j) \geq p_j$ για όλα τα j.

Δώστε έναν αλγόριθμο ο οποίος θα προσδιορίζει αν είναι δυνατή η επιλογή ενός συνόλου εφεδρείας για κάθε συσκευή με βάση αυτή την πιο αναλυτική συνθήκη· εξακολουθεί να ισχύει η ζήτηση ότι καμία συσκευή δεν θα πρέπει να εμφανίζεται στο σύνολο εφεδρείας περισσότερων από b άλλων συσκευών. Και πάλι, ο αλγόριθμος θα πρέπει να δίνει ως έξοδο τα ίδια τα σύνολα εφεδρείας, εφόσον υπάρχουν.

35. Σχεδιάζετε ένα αλληλεπιδραστικό εργαλείο τμηματοποίησης εικόνων το οποίο λειτουργεί με τον ακόλουθο τρόπο. Ξεκινάτε με τη διευθέτηση τμηματοποίησης εικόνας που περιγράφεται στην Ενότητα 7.10, με n εικονοστοιχεία, ένα σύνολο γειτονικών ζευγαριών, και παραμέτρους $\{a_i\}$, $\{b_i\}$, και $\{p_{ij}\}$. Θα κάνουμε δύο παραδοχές σχετικά με αυτό το πρόβλημα. Πρώτον, θα υποθέσουμε ότι η καθεμία από τις παραμέτρους $\{a_i\}$, $\{b_i\}$, και $\{p_{ij}\}$ είναι ένας μη αρνητικός ακέραιος μεταξύ 0 και d, για κάποιον αριθμό d. Δεύτερον, θα υποθέσουμε ότι η σχέση γειτονίας μεταξύ των εικονοστοιχείων έχει την ιδιότητα ότι το κάθε εικονοστοιχείο είναι γείτονας το πολύ τεσσάρων άλλων εικονοστοιχείων (έτσι στο γράφημα που προκύπτει υπάρχουν το πολύ τέσσερις ακμές που εξέρχονται από κάθε κόμβο).

Πρώτα εκτελείτε μια *αρχική τμηματοποίηση* (A_0, B_0), έτσι ώστε να μεγιστοποιηθεί η ποσότητα $q(A_0, B_0)$. Αυτό μπορεί να έχει αποτέλεσμα την αντιστοίχιση κάποιων εικονοστοιχείων στο φόντο, ενώ ο χρήστης γνωρίζει ότι θα πρέπει να ανήκουν στο προσκήνιο. Έτσι, όταν του παρουσιάζετε την τμηματοποίηση, ο χρήστης έχει τη δυνατότητα να πατήσει με το ποντίκι σε ένα συγκεκριμένο εικονοστοιχείο v_1, και να το αντιστοιχίσει έτσι στο προσκήνιο. Όμως το εργαλείο αυτό δεν μεταφέρει απλώς το εικονοστοιχείο στο προσκήνιο· αντίθετα, θα πρέπει να υπολογίσει μια τμηματοποίηση (A_1, B_1) η οποία να μεγιστοποιεί την ποσότητα $q(A_1, B_1)$ *με δεδομένο τον περιορισμό ότι το v_1 ανήκει στο προσκήνιο.* (Στην πράξη, αυτό είναι χρήσιμο για το ακόλουθο είδος λειτουργίας: Κατά την τμηματοποίηση μιας φωτογραφίας με μια ομάδα ανθρώπων, πιθανώς κάποιος από τους ανθρώπους αυτούς να κρατά μια τσάντα η οποία να χαρακτηρίστηκε κατά λάθος ως τμήμα του φόντου. Με πάτημα σε ένα μόνο εικονοστοιχείο στην τσάντα και επανυπολογισμό της βέλτιστης τμηματοποίησης με βάση τη νέα συνθήκη, συχνά ολόκληρη η τσάντα θα γίνει τμήμα του προσκηνίου.)

Στην πραγματικότητα, το σύστημα θα πρέπει να επιτρέπει στο χρήστη να πραγματοποιήσει μια ακολουθία τέτοιων πατημάτων με το ποντίκι $v_1, v_2, \ldots, v_t$· και μετά από το πάτημα v_i το σύστημα θα πρέπει να παράγει μια τμηματοποίηση (A_i, B_i) που να μεγιστοποιεί την ποσότητα $q(A_i, B_i)$ με βάση τη συνθήκη ότι όλα τα $v_1, v_2, \ldots, v_i$ ανήκουν στο προσκήνιο.

Δώστε έναν αλγόριθμο που να πραγματοποιεί αυτή τη λειτουργία έτσι ώστε η αρχική τμηματοποίηση να εκτελείται εντός ενός σταθερού συντελεστή επί το χρόνο που απαιτείται για μία μόνο μέγιστη ροή, και μετά ο χειρισμός της αλληλεπίδρασης με το χρήστη θα γίνεται σε χρόνο $O(dn)$ ανά πάτημα του ποντικιού.

(*Σημείωση:* Η Λυμένη Άσκηση 1 αυτού του κεφαλαίου αποτελεί χρήσιμο υπόβαθρο για τη συγκεκριμένη άσκηση. Επίσης, η συμμετρική λειτουργία της υποχρεωτικής αντιστοίχισης ενός εικονοστοιχείου στο φόντο μπορεί να πραγματοποιηθεί με ανάλογο τρόπο, όμως δεν αποτελεί τμήμα αυτής της άσκησης.)

36. Θα εξετάσουμε τώρα μια διαφορετική παραλλαγή του προβλήματος τμηματοποίησης εικόνας της Ενότητας 7.10. Θα αναπτύξουμε μια λύση σε ένα πρόβλημα *απόδοσης ετικετών σε εικόνα*, όπου ο στόχος είναι να αντιστοιχίσουμε σε κάθε εικονοστοιχείο μια ετικέτα με μια χονδρική εκτίμηση της απόστασης από την κάμερα (αντί για τον απλό χαρακτηρισμό *προσκήνιο/φόντο* που χρησιμοποιήσαμε στην ενότητα του βιβλίου). Οι πιθανές ετικέτες για κάθε εικονοστοιχείο θα είναι 0, 1, 2, ..., M για κάποιον ακέραιο αριθμό M.

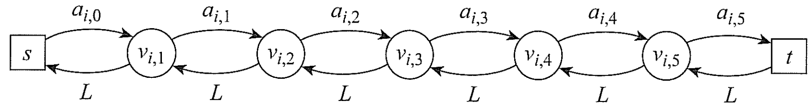
Έστω ότι το $G = (V, E)$ συμβολίζει το γράφημα του οποίου οι κόμβοι είναι τα εικονοστοιχεία και οι ακμές δηλώνουν γειτονικά ζευγάρια εικονοστοιχείων. Ο *χαρακτηρισμός* των εικονοστοιχείων είναι μια διαμέριση του V σε σύνολα A_0, A_1, ..., A_M, όπου το A_k είναι το σύνολο των εικονοστοιχείων που έχουν χαρακτηριστεί ότι έχουν απόσταση k για $k = 0, ..., M$. Θα αναζητήσουμε ένα χαρακτηρισμό με ελάχιστο *κόστος*· το κόστος θα προέρχεται από δύο είδη όρων. Σε αναλογία με το πρόβλημα του διαχωρισμού προσκηνίου/φόντου, θα έχουμε ένα *κόστος αντιστοίχισης*: για κάθε εικονοστοιχείο i και ετικέτα k, το κόστος $a_{i,k}$ είναι το κόστος για την αντιστοίχιση της ετικέτας k στο εικονοστοιχείο i. Στη συνέχεια, αν σε δύο γειτονικά εικονοστοιχεία $(i, j) \in E$ έχουν αποδοθεί διαφορετικές ετικέτες, θα υπάρχει ένα *κόστος διαχωρισμού*. Στην Ενότητα 7.10 χρησιμοποιήσαμε μια ποινή διαχωρισμού p_{ij}. Σε αυτό το πρόβλημα το κόστος διαχωρισμού θα εξαρτάται επίσης από το πόσο μακριά είναι τα δύο εικονοστοιχεία που διαχωρίζουμε· συγκεκριμένα, θα είναι ανάλογο της διαφοράς της τιμής μεταξύ των δύο ετικετών.

Έτσι το συνολικό κόστος q' για το χαρακτηρισμό ορίζεται με τον ακόλουθο τρόπο:

$$q'(A_0, ..., A_M) = \sum_{k=0}^{M} \sum_{i \in A_i} a_{i,k} + \sum_{k<l} \sum_{\substack{(i,j)\in E \\ i\in A_k, j\in A_l}} (l-k)p_{ij} .$$

Ο στόχος αυτού του προβλήματος είναι η ανάπτυξη ενός αλγορίθμου πολυωνυμικού χρόνου ο οποίος θα βρίσκει τη βέλτιστη απόδοση ετικετών με δεδομένο το γράφημα G και τις παραμέτρους κόστους $a_{i,k}$ και p_{ij}. Ο αλγόριθμος θα βασίζεται στην κατασκευή ενός δικτύου ροής, και θα σας βοηθήσουμε στο ξεκίνημα του σχεδιασμού του αλγορίθμου παρέχοντας ένα τμήμα της κατασκευής αυτής.

Το δίκτυο ροής θα έχει μια προέλευση s και μια απόληξη t. Επιπρόσθετα, για κάθε εικονοστοιχείο $i \in V$ θα έχουμε στο δίκτυο ροής κόμβους $v_{i,k}$ για k=1, ..., M, όπως φαίνεται στην Εικόνα 7.30. (Στο παράδειγμα της εικόνας M = 5.)



Εικόνα 7.30 Το σύνολο κόμβων που αντιστοιχεί σε ένα μόνο εικονοστοιχείο i της Άσκησης 36 (φαίνεται επίσης η προέλευση s και η απόληξη t).

Για ευκολία στο συμβολισμό, οι κόμβοι $v_{i,0}$ και $v_{i,M+1}$ θα αναφέρονται αντίστοιχα στα s και t, για οποιοδήποτε $i \in V$.

Προσθέτουμε τώρα ακμές $(v_{i,k}, v_{i,k+1})$ με χωρητικότητα $a_{i,k}$ για $k = 0, ..., M$· καθώς και ακμές $(v_{i,k+1}, v_{i,k})$ στην αντίθετη κατεύθυνση με πολύ μεγάλη χωρητικότητα L. Θα αναφερόμαστε σε αυτή τη συλλογή κόμβων και ακμών ως *αλυσίδα* που σχετίζεται με το εικονοστοιχείο i.

Παρατηρήστε ότι, αν κάνουμε επαρκώς μεγάλη αυτή τη μεγάλη χωρητικότητα L, δεν θα υπάρχει καμία ελάχιστη αποκοπή (A, B) στην οποία μια ακμή με χωρητικότητα L βγαίνει από το σύνολο A. (Πόσο μεγάλη πρέπει να την κάνουμε για να το επιτύχουμε αυτό;) Έτσι, για οποιαδήποτε ελάχιστη αποκοπή (A, B) και οποιοδήποτε εικονοστοιχείο i, θα υπάρχει ακριβώς μία ακμή χαμηλής χωρητικότητας από την αλυσίδα που σχετίζεται με το i η οποία θα εξέρχεται από το σύνολο A. (Θα πρέπει να ελέγξετε ότι, αν υπήρχαν δύο τέτοιες ακμές, τότε θα έπρεπε επίσης να εξέρχεται από το σύνολο A και μια ακμή μεγάλης χωρητικότητας.)

Τέλος, να το ερώτημα που θέτουμε: Χρησιμοποιήστε τους κόμβους και τις ακμές που ορίσαμε μέχρι τώρα για να ολοκληρώσετε την κατασκευή του δικτύου ροής, με την ιδιότητα ότι η απόδοση ετικετών ελάχιστου κόστους θα πρέπει να μπορεί να υπολογιστεί με αποδοτικό τρόπο από μια ελάχιστη αποκοπή s-t. Θα πρέπει να αποδείξετε ότι η κατασκευή σας διαθέτει την επιθυμητή ιδιότητα, και να δείξετε τον τρόπο με τον οποίο θα ανακτήσετε την απόδοση ετικετών ελάχιστου κόστους από την αποκοπή.

37. Στα τυπικά προβλήματα ελάχιστης αποκοπής s-t υποθέτουμε ότι όλες οι χωρητικότητες είναι μη αρνητικές· όταν επιτρέπεται ένα αυθαίρετο σύνολο από θετικές και αρνητικές χωρητικότητες, το αποτέλεσμα είναι ένα πρόβλημα που είναι πολύ δυσκολότερο υπολογιστικά. Παρόλα αυτά, όπως θα δούμε εδώ, μπορούμε να χαλαρώσουμε λίγο αυτή την απαίτηση σχετικά με τις μη αρνητικές χωρητικότητες και να εξακολουθήσουμε να έχουμε ένα πρόβλημα που μπορεί να λυθεί σε πολυωνυμικό χρόνο.

Έστω ότι το $G = (V, E)$ είναι ένα κατευθυνόμενο γράφημα, με προέλευση $s \in V$, απόληξη $t \in V$, και χωρητικότητες ακμών $\{c_e\}$. Υποθέστε ότι για κάθε ακμή e που δεν έχει ως άκρο ούτε το s ούτε το t έχουμε $c_e \geq 0$. Έτσι το c_e μπορεί να είναι αρνητικό για τις ακμές e στις οποίες τουλάχιστον το ένα άκρο τους είναι είτε το s είτε το t. Δώστε έναν αλγόριθμο πολυωνυμικού χρόνου για την εύρεση μιας ελάχιστης αποκοπής s-t σε ένα τέτοιο γράφημα. (Παρά τη νέα απαίτηση μη αρνητικών χωρητικοτήτων, και εδώ επίσης ορίζουμε ότι η τιμή μιας αποκοπής s-t (A, B) είναι το άθροισμα των χωρητικοτήτων όλων των πλευρών e για τις οποίες η αρχή του e ανήκει στο A και το τέλος του e ανήκει στο B.)

38. Δουλεύετε με μια μεγάλη βάση δεδομένων με εγγραφές εργαζομένων. Για τους σκοπούς αυτού του ερωτήματος, μπορούμε να φανταστούμε τη βάση δεδομένων σαν ένα δισδιάστατο πίνακα T με ένα σύνολο R από m γραμμές και ένα σύνολο C

από n στήλες· οι γραμμές αντιστοιχούν στους διάφορους εργαζομένους, και οι στήλες αντιστοιχούν στις διάφορες ιδιότητες (πεδία).

Για να δούμε ένα απλό παράδειγμα, μπορεί να έχουμε τις τέσσερις στήλες με ετικέτες

`όνομα, αριθμός τηλεφώνου, ημερομηνία έναρξης, όνομα διευθυντή`

και έναν πίνακα με πέντε εργαζομένους όπως φαίνεται εδώ.

όνομα	**αριθμός τηλεφώνου**	**ημερομηνία έναρξης**	**όνομα διευθυντή**
Αλίκη	3-4563	13/6/95	Γιώργος
Γιώργος	3-2341	20/1/93	Λάμπρος
Ελισάβετ	3-2345	19/12/01	Γιώργος
Ηλίας	3-9000	12/1/97	Γιώργος
Ιωάννα	3-3453	1/7/96	Γιώργος

Με δεδομένο ένα υποσύνολο S των στηλών, μπορούμε να πάρουμε ένα νέο, μικρότερο πίνακα κρατώντας μόνο τα στοιχεία που περιλαμβάνουν στήλες του συνόλου S. Θα αποκαλέσουμε αυτόν το νέο πίνακα *προβολή* του T στο S, και θα τον συμβολίζουμε με $T\,[S]$. Για παράδειγμα, αν $S =$ {`όνομα, ημερομηνία έναρξης`}, τότε η προβολή $T\,[S]$ θα είναι ο πίνακας που αποτελείται μόνο από την πρώτη και την τρίτη στήλη.

Υπάρχει μια ακόμα χρήσιμη λειτουργία στους πίνακες, η οποία είναι η *αντιμετάθεση* των στηλών. Με δεδομένη μια αντιμετάθεση p των στηλών, μπορούμε να πάρουμε ένα νέο πίνακα ίδιου μεγέθους με τον T με απλή αναδιάταξη των στηλών σύμφωνα με την αντιμετάθεση p. Θα το ονομάσουμε αυτό *αντιμετάθεση* του T κατά p, και θα το συμβολίζουμε με T_p.

Όλα αυτά χρησιμοποιούνται στη συγκεκριμένη εφαρμογή σας, με τον ακόλουθο τρόπο. Έχετε k διαφορετικά υποσύνολα των στηλών $S_1, S_2, ..., S_k$ με τα οποία θα δουλέψετε πολύ, και έτσι θέλετε να τα έχετε διαθέσιμα σε μια εύκολα προσπελάσιμη μορφή. Μια λύση θα ήταν να αποθηκεύσετε τις k προβολές $T\,[S_1]$, $T\,[S_2]$, ..., $T\,[S_k]$, όμως αυτό θα χρειαζόταν πολύ χώρο. Καθώς εξετάζατε τις διάφορες εναλλακτικές λύσεις, μάθατε ότι πιθανώς να μη χρειαστεί να εκτελέσετε ρητά την προβολή σε κάθε υποσύνολο, επειδή το υποκείμενο σύστημα βάσης δεδομένων μπορεί να χειριστεί ιδιαίτερα αποδοτικά ένα υποσύνολο των στηλών όταν (σε κάποια σειρά) τα μέλη του υποσυνόλου συνιστούν ένα *πρόθεμα* των στηλών από αριστερά προς τα δεξιά. Έτσι, στο παράδειγμά μας, τα υποσύνολα {`όνομα, αριθμός τηλεφώνου`} και {`όνομα, ημερομηνία έναρξης, αριθμός τηλεφώνου` } συνιστούν προθέματα (είναι αντίστοιχα οι πρώτες δύο και τρεις στήλες από αριστερά)· έτσι το σύστημα μπορεί να τα επεξεργαστεί πολύ πιο αποδοτικά από ένα υποσύνολο όπως το {`όνομα, ημερομηνία έναρξης`}, το οποίο δεν συνιστά πρόθεμα. (Και πάλι, σημειώστε ότι το υποσύνολο S_i δεν έχει κάποια συγκεκριμένη σειρά διάταξης των μελών του, και έτσι ενδιαφερόμαστε αν υπάρχει *κάποια* σειρά με την οποία να σχηματίζει πρόθεμα των στηλών.)

Να λοιπόν το ερώτημα: Με δεδομένη μια παράμετρο $\ell < k$, μπορούμε να βρούμε ℓ αντιμεταθέσεις των στηλών $p_1, p_2, ..., p_\ell$ έτσι ώστε για καθένα από τα υποσύνολα S_i που μας δίνονται (για $i = 1, 2, ..., k$) οι στήλες του υποσυνόλου S_i να συνιστούν πρόθεμα για τουλάχιστον έναν από τους πίνακες που προκύπτουν από τις αντιμεταθέσεις $T_{p_1}, T_{p_2}, ..., T_{p_\ell}$; Θα λέμε ότι ένα τέτοιο σύνολο αντιμεταθέσεων αποτελεί έγκυρη λύση για το πρόβλημα· αν υπάρχει έγκυρη λύση, αυτό σημαίνει ότι χρειάζεται να αποθηκεύσουμε μόνο τους ℓ πίνακες που προκύπτουν από τις αντιμεταθέσεις, αντί για όλες τις k προβολές. Κατασκευάστε έναν αλγόριθμο πολυωνυμικού χρόνου για την επίλυση αυτού του προβλήματος· στις περιπτώσεις στις οποίες υπάρχει μια έγκυρη λύση, ο αλγόριθμός σας θα πρέπει να επιστρέφει ένα κατάλληλο σύνολο ℓ αντιμεταθέσεων.

Παράδειγμα. Υποθέστε ότι για τον παραπάνω πίνακα τα υποσύνολα που μας δίνονται είναι

S_1 = {`όνομα`, `αριθμός τηλεφώνου`},

S_2 = {`όνομα`, `ημερομηνία έναρξης`},

S_3 = {`όνομα`, `όνομα διευθυντή`, `ημερομηνία έναρξης`},

και $\ell = 2$. Τότε υπάρχει έγκυρη λύση για αυτό το πρόβλημα, η οποία μπορεί να επιτευχθεί με δύο αντιμεταθέσεις

p_1 = {`όνομα`, `αριθμός τηλεφώνου`, `ημερομηνία έναρξης`, `όνομα διευθυντή`},

p_2 = {`όνομα`, `ημερομηνία έναρξης`, `όνομα διευθυντή`, `αριθμός τηλεφώνου`}.

Με αυτόν τον τρόπο το S_1 αποτελεί πρόθεμα για τον πίνακα που προκύπτει από την αντιμετάθεση T_{p_1}, ενώ τα S_2 και S_3 αποτελούν πρόθεμα για τον πίνακα που προκύπτει από την αντιμετάθεση T_{p_2}.

39. Είστε σύμβουλοι σε μια εταιρεία κοινωνικοπολιτικής στατιστικής. Η εταιρεία αυτή συλλέγει στατιστικά στοιχεία και δημοσιεύει τα δεδομένα αυτά σε ένα βιβλίο. Τα στατιστικά στοιχεία αφορούν τους πληθυσμούς διαφόρων περιοχών στον κόσμο, και καταγράφονται σε πολλαπλάσια του ενός εκατομμυρίου. Ένα παράδειγμα τέτοιων στατιστικών στοιχείων φαίνεται στον επόμενο πίνακα.

Χώρα	**A**	**B**	**Γ**	**Σύνολο**
ενήλικοι άνδρες	11.998	9.083	2.919	24.000
ενήλικες γυναίκες	12.983	10.872	3.145	27.000
παιδιά	1.019	2.045	0.936	4.000
Σύνολο	26.000	22.000	7.000	55.000

Θα υποθέσουμε εδώ, για λόγους απλότητας, ότι τα δεδομένα μας είναι τέτοια ώστε όλα τα αθροίσματα στηλών και γραμμών είναι ακέραιοι αριθμοί. Το πρόβλημα Πληθυσμιακής Στρογγυλοποίησης είναι να στρογγυλοποιήσετε όλα τα δεδομένα σε ακέραιους χωρίς να αλλάξει κανένα άθροισμα γραμμής ή στήλης. Ο κάθε δεκαδικός αριθμός μπορεί να στρογγυλοποιηθεί είτε προς τα κάτω είτε προς τα επάνω. Για παράδειγμα, μια καλή στρογγυλοποίηση για τον πίνακα δεδομένων μας θα ήταν η ακόλουθη.

Χώρα	**A**	**B**	**Γ**	**Σύνολο**
ενήλικοι άνδρες	11.000	10.000	3.000	24.000
ενήλικες γυναίκες	13.000	10.000	4.000	27.000
παιδιά	2.000	2.000	0.000	4.000
Σύνολο	26.000	22.000	7.000	55.000

(α) Εξετάστε καταρχήν την ειδική περίπτωση όπου όλα τα δεδομένα είναι μεταξύ του 0 και του 1. Έτσι έχετε έναν πίνακα με δεκαδικούς αριθμούς μεταξύ 0 και 1, και το πρόβλημά σας είναι να στρογγυλοποιήσετε τον κάθε αριθμό είτε σε 0 είτε σε 1 χωρίς να μεταβληθεί το άθροισμα των γραμμών ή των στηλών. Χρησιμοποιήστε έναν υπολογισμό ροής για να ελέγξετε αν είναι δυνατή η επιθυμητή στρογγυλοποίηση.

(β) Εξετάστε το πρόβλημα Πληθυσμιακής Στρογγυλοποίησης όπως το ορίσαμε παραπάνω, όπου τα αθροίσματα γραμμών και στηλών είναι ακέραιοι και θέλετε να στρογγυλοποιήσετε τον κάθε δεκαδικό αριθμό α είτε σε $\lfloor \alpha \rfloor$ είτε σε $\lceil \alpha \rceil$. Χρησιμοποιήστε έναν υπολογισμό ροής για να ελέγξετε αν είναι δυνατή η επιθυμητή στρογγυλοποίηση.

(γ) Αποδείξτε ότι η στρογγυλοποίηση την οποία αναζητάτε στο (α) και στο (β) υπάρχει πάντα.

40. Σε πολλούς αριθμητικούς υπολογισμούς μπορούμε να διερωτηθούμε σχετικά με τη "σταθερότητα" ή "ανθεκτικότητα" της απάντησης. Αυτό το είδος ερώτησης μπορεί να τεθεί επίσης και για τα συνδυαστικά προβλήματα· ας δούμε έναν τρόπο διατύπωσης αυτού του ερωτήματος για το πρόβλημα του Ελάχιστου Γεννητικού Δένδρου.

Υποθέστε ότι σας δίνεται ένα γράφημα $G = (V, E)$ με ένα κόστος c_e σε κάθε ακμή e. Θεωρούμε αυτά τα κόστη ποσότητες που έχουν μετρηθεί πειραματικά, και έτσι υπόκεινται σε πιθανά σφάλματα μέτρησης. Έτσι το ελάχιστο γεννητικό δένδρο που υπολογίζεται για το G μπορεί να μην είναι στην πραγματικότητα το "πραγματικό" ελάχιστο γεννητικό δένδρο.

Με δεδομένες τις παραμέτρους σφάλματος $\varepsilon > 0$ και $k > 0$ και μια συγκεκριμένη ακμή $e' = (u, v)$, θέλετε να μπορείτε να διατυπώσετε έναν ισχυρισμό της ακόλουθης μορφής.

(*) *Ακόμα και αν το κόστος όλων των ακμών αλλάξει το πολύ κατά* ε *(είτε αυξηθεί είτε ελαττωθεί), και το κόστος* k *από τις ακμές* εκτός της e' *αλλάξει ακόμα περισσότερο σε αυθαίρετα διαφορετικές τιμές, η ακμή* e' *δεν θα ανήκει και πάλι σε κανένα ελάχιστο γεννητικό δένδρο για το* G.

Μια τέτοια ιδιότητα παρέχει ένα είδος εξασφάλισης ότι η e' δεν είναι πιθανό να ανήκει σε κανένα ελάχιστο γεννητικό δένδρο, ακόμα και να υποθέσουμε την ύπαρξη σημαντικών σφαλμάτων μέτρησης.

Δώστε έναν αλγόριθμο πολυωνυμικού χρόνου ο οποίος θα παίρνει τα G, e', ε, και k, και θα αποφασίζει αν ισχύει η ιδιότητα (*) για την ακμή e'.

41. Υποθέστε ότι διαχειρίζεστε μια συλλογή επεξεργαστών και πρέπει να χρονοπρογραμματίσετε μια ακολουθία εργασιών.

Οι εργασίες έχουν τα ακόλουθα χαρακτηριστικά. Η κάθε εργασία j έχει ένα χρόνο άφιξης a_j όπου γίνεται για πρώτη φορά διαθέσιμη για επεξεργασία, ένα μήκος ℓ_j το οποίο δείχνει πόσο χρόνο επεξεργασίας χρειάζεται, και μια προθεσμία d_j μέχρι την οποία θα πρέπει να έχει ολοκληρωθεί. (Θα υποθέσουμε ότι $0 < \ell_j \leq d_j - a_j$.) Η κάθε εργασία μπορεί να εκτελεστεί σε οποιονδήποτε από τους επεξεργαστές, αλλά θα εκτελείται κάθε φορά μόνο σε έναν επεξεργαστή. Μπορεί επίσης να διακοπεί και να συνεχιστεί από το σημείο διακοπής της (πιθανώς μετά από κάποια καθυστέρηση) σε κάποιον άλλον επεξεργαστή.

Επιπρόσθετα, η συλλογή των επεξεργαστών δεν είναι εντελώς στατική: Έχετε μια συνολική δεξαμενή από k πιθανούς επεξεργαστές· όμως για κάθε επεξεργαστή i υπάρχει ένα χρονικό διάστημα $[t_i, t_i']$ κατά το οποίο είναι διαθέσιμος — όλες τις άλλες στιγμές δεν είναι διαθέσιμος.

Με δεδομένα όλα αυτά τα δεδομένα σχετικά με τις απαιτήσεις των εργασιών και τη διαθεσιμότητα των επεξεργαστών, θέλετε να αποφασίσετε αν οι εργασίες μπορούν να ολοκληρωθούν όλες ή όχι. Κατασκευάστε έναν αλγόριθμο πολυωνυμικού χρόνου ο οποίος είτε θα παράγει ένα χρονοδιάγραμμα που θα ολοκληρώνει όλες τις εργασίες εντός προθεσμίας, είτε θα αναφέρει (σωστά) ότι δεν υπάρχει τέτοιο χρονοδιάγραμμα. Μπορείτε να υποθέσετε ότι οι παράμετροι του προβλήματος είναι ακέραιες.

Παράδειγμα. Υποθέστε ότι έχουμε δύο εργασίες J_1 και J_2. Η εργασία J_1 καταφθάνει τη χρονική στιγμή 0, έχει προθεσμία τη χρονική στιγμή 4, και έχει μήκος 3. Η εργασία J_2 καταφθάνει τη χρονική στιγμή 1, έχει προθεσμία τη χρονική στιγμή 3, και έχει μήκος 2. Διαθέτουμε επίσης δύο επεξεργαστές P_1 και P_2. Ο P_1 είναι διαθέσιμος μεταξύ των χρονικών στιγμών 0 και 4· ο P_2 είναι διαθέσιμος μεταξύ των χρονικών στιγμών 2 και 3. Σε αυτή την περίπτωση, υπάρχει ένα χρονοδιάγραμμα που ολοκληρώνει και τις δύο εργασίες.

- Κατά τη χρονική στιγμή 0 ξεκινάμε την εργασία J_1 στον επεξεργαστή P_1.
- Κατά τη χρονική στιγμή 1 διακόπτουμε την εργασία J_1 και ξεκινάμε την εργασία J_2 στον επεξεργαστή P_1.

- Κατά τη χρονική στιγμή 2 συνεχίζουμε την εργασία J_1 στον επεξεργαστή P_2. (Η εργασία J_2 συνεχίζει να εκτελείται στον επεξεργαστή P_1.)
- Κατά τη χρονική στιγμή 3 η εργασία J_2 ολοκληρώνεται εντός προθεσμίας. Ο επεξεργαστής P_2 παύει να είναι διαθέσιμος, έτσι μετακινούμε ξανά την εργασία J_1 στον επεξεργαστή P_1 για να ολοκληρώσει εκεί την απομένουσα μία χρονική μονάδα επεξεργασίας.
- Κατά τη χρονική στιγμή 4 η εργασία J_1 ολοκληρώνεται στον επεξεργαστή P_1.

Παρατηρήστε ότι δεν υπάρχει λύση που να μην περιλαμβάνει διακοπή και μετακίνηση εργασιών.

42. Δώστε έναν αλγόριθμο πολυωνυμικού χρόνου για το ακόλουθο πρόβλημα ελαχιστοποίησης που είναι ανάλογο του προβλήματος Μέγιστης Ροής. Σας δίνεται ένα κατευθυνόμενο γράφημα $G = (V, E)$, με προέλευση $s \in V$ και απόληξη $t \in V$, καθώς και αριθμοί (χωρητικότητες) $\ell(v, w)$ για κάθε ακμή $(v, w) \in E$. Ορίζουμε με το συνηθισμένο τρόπο μια ροή f και την τιμή της ροής, απαιτώντας ότι όλοι οι κόμβοι εκτός του s και του t ικανοποιούν τις συνθήκες διατήρησης ροής. Παρόλα αυτά, οι αριθμοί που δίνονται είναι κάτω όρια για τη ροή στις ακμές — δηλαδή, απαιτείται ότι $f(v, w) \geq \ell(v, w)$ για κάθε ακμή $(v, w) \in E$, και δεν υπάρχει άνω όριο ως προς την τιμή της ροής στις ακμές.

(α) Κατασκευάστε έναν αλγόριθμο πολυωνυμικού χρόνου που να βρίσκει μια εφικτή ροή με την ελάχιστη δυνατή τιμή.

(β) Αποδείξτε το ανάλογο του θεωρήματος Μέγιστης Ροής και Ελάχιστης Αποκοπής για αυτό το πρόβλημα (δηλαδή, ισχύει ότι ελάχιστη ροή = μέγιστη αποκοπή;).

43. Προσπαθείτε να λύσετε ένα πρόβλημα κυκλοφορίας, όμως δεν είναι εφικτό. Το πρόβλημα έχει ζήτηση, αλλά δεν έχει όρια χωρητικότητας στις ακμές. Πιο τυπικά, υπάρχει ένα γράφημα $G = (V, E)$ και ζήτηση d_v για κάθε κόμβο v (που ικανοποιούν τη σχέση $\sum_{v \in V} d_v = 0$), και το πρόβλημα είναι να αποφασίσετε αν υπάρχει μια ροή f τέτοια ώστε $f(e) \geq 0$ και $f^{in}(v) - f^{out}(v) = d_v$ για όλους τους κόμβους $v \in V$. Σημειώστε ότι μπορούμε να λύσουμε αυτό το πρόβλημα με τον αλγόριθμο κυκλοφορίας της Ενότητας 7.7, αν θέσουμε $c_e = +\infty$ για όλες τις ακμές $e \in E$. (Εναλλακτικά, αρκεί να ορίσουμε το c_e σε έναν εξαιρετικά μεγάλο αριθμό για κάθε ακμή — για παράδειγμα, μεγαλύτερο από το σύνολο όλης της θετικής ζήτησης d_v του γραφήματος.)

Θέλετε να επιδιορθώσετε το γράφημα έτσι ώστε να κάνετε το πρόβλημα εφικτό, έτσι είναι πολύ χρήσιμο να γνωρίζετε γιατί το πρόβλημα δεν είναι εφικτό με τη μορφή που έχει αυτή τη στιγμή. Με προσεκτικότερη παρατήρηση βλέπετε ότι υπάρχει ένα υποσύνολο U των κόμβων για το οποίο δεν υπάρχει καμία ακμή που εισέρχεται στο U, και παρόλα αυτά $\sum_{v \in U} d_v > 0$. Συνειδητοποιείτε πολύ γρήγορα ότι η ύπαρξη ενός τέτοιου συνόλου υποδηλώνει αμέσως ότι δεν μπορεί να υπάρχει ροή. Το σύνολο U έχει θετική συνολική ζήτηση, και έτσι χρειάζεται εισερχόμενη ροή, όμως παρόλα αυτά το U δεν έχει καθόλου εισερχόμενες ακμές. Προσπαθώντας να εκτιμήσετε πόσο απέχει το πρόβλημα από το να είναι επιλύσιμο, αναρωτιέστε πόσο μεγάλη μπορεί να είναι η ζήτηση ενός συνόλου που δεν έχει καθόλου εισερχόμενες ακμές.

Δώστε έναν αλγόριθμο πολυωνυμικού χρόνου για την εύρεση ενός υποσυνόλου $S \subset V$ των κόμβων έτσι ώστε να μην υπάρχει καμία εισερχόμενη ακμή στο S και το $\sum_{v \in S} d_v$ να είναι όσο το δυνατόν μεγαλύτερο υπό αυτόν τον περιορισμό.

44. Υποθέστε ότι σας δίνεται ένα κατευθυνόμενο δίκτυο $G = (V, E)$ με έναν κόμβο ρίζας r και ένα σύνολο *τερματικών* $T \subseteq V$. Θέλουμε να "αποσυνδέσουμε" όσο το δυνατόν περισσότερα τερματικά από το r, αποκόπτοντας όμως σχετικά λίγες ακμές.

Ας διατυπώσουμε με ακρίβεια αυτή την αντιστάθμιση. Για ένα σύνολο ακμών $F \subseteq E$, έστω ότι το $q(F)$ συμβολίζει το σύνολο των κόμβων $v \in T$ για τους οποίους δεν υπάρχει καμία διαδρομή r-v στο υπογράφημα $(V, E - F)$. Δώστε έναν αλγόριθμο πολυωνυμικού χρόνου ο οποίος βρίσκει ένα σύνολο F των ακμών που να μεγιστοποιεί την ποσότητα $q(F) - |F|$. (Σημειώστε ότι μια επιλογή είναι να θέσουμε το F ίσο με το κενό σύνολο.)

45. Θεωρήστε τον ακόλουθο ορισμό. Μας δίνεται ένα σύνολο από n χώρες που διενεργούν εμπόριο μεταξύ τους. Για κάθε χώρα i, έχουμε την τιμή s_i του εμπορικού πλεονάσματός της· ο αριθμός αυτός μπορεί να είναι είτε θετικός είτε αρνητικός, με τους αρνητικούς αριθμούς να δηλώνουν έλλειμμα. Για κάθε ζευγάρι χωρών i, j έχουμε τη συνολική αξία e_{ij} όλων των εξαγωγών από τη χώρα i στη χώρα j· ο αριθμός αυτός είναι πάντοτε μη αρνητικός. Λέμε ότι ένα υποσύνολο S των χωρών είναι *ανεξάρτητο* αν το άθροισμα του εμπορικού πλεονάσματος των χωρών του συνόλου S μείον τη συνολική αξία όλων των εξαγωγών από χώρες του συνόλου S σε χώρες που δεν ανήκουν στο σύνολο S είναι μη αρνητικό.

Δώστε έναν αλγόριθμο πολυωνυμικού χρόνου ο οποίος θα παίρνει τα δεδομένα αυτά για ένα σύνολο n χωρών και θα αποφασίζει αν το σύνολο αυτό περιέχει ένα μη κενό ανεξάρτητο υποσύνολο που δεν είναι ίσο με το πλήρες σύνολο.

46. Στην κοινωνιολογία μελετάμε συχνά ένα γράφημα G στο οποίο οι κόμβοι αντιπροσωπεύουν ανθρώπους και οι ακμές αντιπροσωπεύουν τις σχέσεις φιλίας μεταξύ των ανθρώπων. Ας υποθέσουμε για τους σκοπούς αυτής της άσκησης ότι η φιλία είναι συμμετρική, και έτσι μπορούμε να θεωρήσουμε ότι έχουμε ένα μη κατευθυνόμενο γράφημα.

Υποθέστε τώρα ότι θέλουμε να μελετήσουμε αυτό το γράφημα G, ψάχνοντας για μια "στενά συνδεδεμένη" ομάδα ατόμων. Ένας τρόπος για να τυποποιήσουμε αυτή την έννοια είναι ο ακόλουθος. Για ένα υποσύνολο S των κόμβων, έστω ότι το $e(S)$ συμβολίζει τον αριθμό των ακμών του S — δηλαδή, τον αριθμό των ακμών που έχουν και τα δύο άκρα εντός του S. Ορίζουμε τη *συνοχή* του S ως $e(S)/|S|$. Ένα φυσικό ζήτημα προς αναζήτηση θα ήταν να βρούμε ένα σύνολο ανθρώπων S που επιτυγχάνουν τη μέγιστη συνοχή.

(α) Δώστε έναν αλγόριθμο πολυωνυμικού χρόνου ο οποίος θα παίρνει ως είσοδο ένα ρητό αριθμό α και θα αποφασίζει αν υπάρχει ένα σύνολο S με συνοχή τουλάχιστον ίση με α.

(β) Δώστε έναν αλγόριθμο πολυωνυμικού χρόνου ο οποίος θα βρίσκει ένα σύνολο S των κόμβων που επιτυγχάνει τη μέγιστη συνοχή.

47. Ο στόχος αυτού του προβλήματος είναι να προτείνει παραλλαγές του αλγορίθμου Προροής-Προώθησης οι οποίες επιταχύνουν το χρόνο εκτέλεσης στην πράξη χωρίς να καταστρέφουν την πολυπλοκότητα χειρότερης περίπτωσης. Θυμηθείτε ότι ο αλγόριθμος διατηρεί τη σχέση $h(v) \leq h(w) + 1$ για όλες τις ακμές (v, w) του υπολοιπόμενου γραφήματος για την τρέχουσα προροή. Αποδείξαμε ότι αν η f είναι ροή (και όχι απλώς προροή) με αυτή τη σχέση, τότε είναι μια μέγιστη ροή. Τα ύψη είναι γνησίως αύξοντα, και η ανάλυση ως προς το χρόνο εκτέλεσης εξαρτιόταν από το όριο ως προς το πλήθος φορών που μπορεί να αυξηθεί το ύψος των κόμβων. Η πράξη έχει δείξει ότι ο αλγόριθμος είναι σχεδόν πάντα πολύ ταχύτερος από αυτό που προτείνει η χειρότερη περίπτωση, και ότι στην πράξη το σημείο χρονικού περιορισμού του αλγορίθμου είναι η ανανέωση ετικετών για τους κόμβους (και όχι οι μη κορεστικές προωθήσεις που οδηγούσαν στη χειρότερη περίπτωσή για τη θεωρητική ανάλυση). Ο στόχος του προβλήματος αυτού είναι να μειωθεί το πλήθος των ανανεωμένων ετικετών με αύξηση των υψών κατά περισσότερο από ένα. Υποθέστε ότι έχετε ένα γράφημα G με n κόμβους, m ακμές, χωρητικότητες c, προέλευση s, και απόληξη t.

(α) Ο αλγόριθμος Προροής-Προώθησης, όπως περιγράψαμε στην Ενότητα 7.4, ξεκινά ρυθμίζοντας τη ροή ίση με τη χωρητικότητα c_e σε όλες τις ακμές e που ξεκινούν από την προέλευση, καθορίζει ροή 0 σε όλες τις άλλες ακμές, ορίζει $h(s) = n$, και ορίζει $h(v) = 0$ για όλους τους άλλους κόμβους $v \in V$. Δώστε μια διαδικασία χρόνου $O(m)$ για την απόδοση αρχικών τιμών ως προς τα ύψη των κόμβων η οποία να είναι καλύτερη από αυτή που κατασκευάσαμε στην Ενότητα 7.4. Η μέθοδός σας θα πρέπει να ορίζει το ύψος κάθε κόμβου v σε όσο το δυνατόν υψηλότερη τιμή με δεδομένη την αρχική ροή.

(β) Σε αυτό το κομμάτι θα προσθέσουμε στον αλγόριθμο Προροής-Προώθησης ένα νέο βήμα, που ονομάζεται *ανανέωση ετικετών χάσματος* (gap relabeling), το οποίο θα αυξάνει τις ετικέτες σε πολλούς κόμβους κατά περισσότερο από ένα τη φορά. Θεωρήστε μια προροή f και ύψη h τα οποία ικανοποιούν τη σχέση του αλγορίθμου. Το *χάσμα* στα ύψη είναι ένας ακέραιος $0 < h < n$ τέτοιος ώστε κανένας κόμβος να μην έχει ύψος ακριβώς h. Υποθέστε ότι το h είναι μια τιμή χάσματος, και έστω ότι το A είναι το σύνολο όλων των κόμβων v με ύψη $n > h(v) > h$. Η ανανέωση ετικετών χάσματος είναι η διαδικασία αλλαγής των υψών σε όλους τους κόμβους του A έτσι ώστε να είναι ίσα με n. Αποδείξτε ότι ο αλγόριθμος Προροής-Προώθησης με ανανέωση ετικετών χάσματος είναι ένας έγκυρος αλγόριθμος μέγιστης ροής. Παρατηρήστε ότι το μόνο νέο πράγμα που χρειάζεται να αποδείξετε είναι ότι η διαδικασία ανανέωσης ετικετών χάσματος διατηρεί την παραπάνω σχέση, δηλ. $h(v) \leq h(w) + 1$ για όλες τις ακμές (v, w) του υπολοιπόμενου γραφήματος.

(γ) Στην Ενότητα 7.4 αποδείξαμε ότι $h(v) \leq 2n - 1$ σε όλη την πορεία του αλγορίθμου. Εδώ θα έχουμε μια παραλλαγή όπου $h(v) \leq n$ σε όλη την πορεία του αλγορίθμου. Η ιδέα είναι ότι "παγώνουμε" όλους τους κόμβους όταν φτάνουν σε ύψος n· δηλαδή, οι κόμβοι σε ύψος n δεν θεωρούνται πλέον ενεργοί, και έτσι δεν χρησιμοποιούνται για τις λειτουργίες `push` και `relabel`. Με αυτόν τον τρόπο στο τέλος του αλγορίθμου έχουμε μια προροή και μια συνάρτηση ύψους που ικανοποιεί την παραπάνω σχέση, με όλο το πλεόνασμα να βρίσκεται στο ύψος n. Έστω ότι B είναι το σύνολο των κόμβων v έτσι ώστε να υπάρχει μια διαδρομή από το v προς το t στο υπολοιπόμενο γράφημα της τρέχουσας προροής. Έστω ότι $A = V - B$. Αποδείξτε ότι στο τέλος του αλγορίθμου το (A, B) είναι μια αποκοπή s-t με ελάχιστη χωρητικότητα.

(δ) Ο αλγόριθμος της άσκησης (γ) υπολογίζει μια ελάχιστη αποκοπή s-t, δεν καταφέρνει όμως να βρει μια μέγιστη ροή (επειδή τελειώνει με μια προροή που έχει πλεόνασμα). Δώστε έναν αλγόριθμο ο οποίος θα παίρνει την προροή f από το τέλος του αλγορίθμου της άσκησης (γ) και θα τη μετατρέπει σε μέγιστη ροή σε χρόνο το πολύ $O(mn)$. (*Υπόδειξη:* Εξετάστε τους κόμβους με πλεόνασμα, και προσπαθήστε να στείλετε το πλεόνασμα πίσω στο s χρησιμοποιώντας μόνο ακμές από τις οποίες ήρθε η ροή.)

48. Στην Ενότητα 7.4 μελετήσαμε τον αλγόριθμο Προροής-Προώθησης και αναλύσαμε ένα συγκεκριμένο κανόνα επιλογής για την εξέταση κορυφών. Εδώ θα εξερευνήσουμε ένα διαφορετικό κανόνα επιλογής. Θα εξετάσουμε επίσης παραλλαγές του αλγορίθμου που τερματίζονται πρόωρα (και βρίσκουν μια αποκοπή που είναι κοντά στην ελάχιστη δυνατή).

(α) Έστω ότι το f είναι οποιαδήποτε προροή. Αφού το f δεν είναι κατανάγκην μια έγκυρη ροή, μπορεί η τιμή $f^{out}(s)$ να είναι σημαντικά υψηλότερη από την τιμή της μέγιστης ροής στο G. Δείξτε, όμως, ότι το $f^{in}(t)$ αποτελεί κάτω όριο ως προς την τιμή της μέγιστης ροής.

(β) Θεωρήστε μια προροή f και μια συμβατή απόδοση ετικετών h. Θυμηθείτε ότι το σύνολο $A = \{v$: Υπάρχει μια διαδρομή s-v στο υπολοιπόμενο γράφημα $G_f\}$ και το σύνολο $B = V - A$ ορίζουν μια έγκυρη αποκοπή s-t για οποιαδήποτε προροή f που έχει μια συμβατή απόδοση ετικετών h. Δείξτε ότι η χωρητικότητα της αποκοπής (A, B) είναι ίση με $c(A,B) = \sum_{v \in B} e_f(v)$.

Ο συνδυασμός των (α) και (β) μας επιτρέπει να τερματίσουμε πρόωρα τον αλγόριθμο και να επιστρέψουμε το (A, B) ως προσεγγιστική αποκοπή ελάχιστης χωρητικότητας, αν υποθέσουμε ότι το $c(A, B) - f^{in}(t)$ είναι επαρκώς μικρό. Στη συνέχεια θα εξετάσουμε μια υλοποίηση η οποία προσπαθεί να μειώσει αυτή την τιμή προωθώντας ροή από τους κόμβους που έχουν μεγάλο πλεόνασμα.

(γ) Η κλιμακούμενη εκδοχή του αλγορίθμου Προροής-Προώθησης διατηρεί μια παράμετρο κλιμάκωσης Δ. Ορίζουμε αρχικά το Δ σε μια μεγάλη δύναμη του 2. Σε κάθε βήμα ο αλγόριθμος επιλέγει έναν κόμβο που έχει πλεόνασμα τουλάχιστον Δ και όσο το δυνατόν μικρότερο ύψος. Όταν κανένας άλλος κόμβος (εκτός από τον t) δεν έχει πλεόνασμα τουλάχιστον Δ, διαιρούμε το Δ δια 2 και συνεχίζουμε. Παρατηρήστε ότι αυτή είναι μια έγκυρη υλοποίηση του γενικού αλγορίθμου Προροής-Προθώθησης. Ο αλγόριθμος εκτελείται σε φάσεις. Η κάθε φάση διαρκεί όσο παραμένει αμετάβλητο το Δ. Σημειώστε ότι το Δ ξεκινά από τη μέγιστη χωρητικότητα και ο αλγόριθμος τερματίζεται όταν $\Delta = 1$. Έτσι υπάρχουν το πολύ $O(\log C)$ φάσεις κλιμάκωσης. Δείξτε πώς μπορεί να υλοποιηθεί αυτή η παραλλαγή του αλγορίθμου έτσι ώστε ο χρόνος εκτέλεσης να έχει όριο $O(mn + n \log C + K)$ αν ο αλγόριθμος έχει K μη κορεστικές λειτουργίες push.

(δ) Δείξτε ότι το πλήθος των μη κορεστικών λειτουργιών push στον παραπάνω αλγόριθμο είναι το πολύ ίσο με $O(n^2 \log C)$. Θυμηθείτε ότι το $O(\log C)$ αποτελεί όριο ως προς το πλήθος των φάσεων κλιμάκωσης. Για να βρείτε το πλήθος των μη κορεστικών λειτουργιών push σε κάθε φάση κλιμάκωσης, εξετάστε τη συνάρτηση δυναμικού $\Phi = \sum_{v \in V} h(v)\, e_f(v) / \Delta$. Ποια είναι η επίδραση μιας μη κορεστικής προώθησης στη συνάρτηση Φ; Ποια ή ποιες λειτουργίες μπορούν να προκαλέσουν την αύξηση της Φ;

49. Θεωρήστε ένα πρόβλημα αντιστοίχισης όπου έχουμε ένα σύνολο από n σταθμούς εξυπηρέτησης και υπάρχει ένα σύνολο από k αιτήσεις για εξυπηρέτηση. Υποθέστε, για παράδειγμα, ότι οι σταθμοί είναι κυψέλες κινητής τηλεφωνίας και οι αιτήσεις είναι κινητά τηλέφωνα. Η κάθε αίτηση μπορεί να εξυπηρετηθεί από ένα δεδομένο σύνολο σταθμών. Το πρόβλημα μέχρι το σημείο αυτό μπορεί να αναπαρασταθεί από ένα διμερές γράφημα G: στη μία πλευρά είναι οι σταθμοί, στην άλλη οι πελάτες, και υπάρχει μια ακμή (x, y) μεταξύ του πελάτη x και του σταθμού y εάν ο πελάτης x μπορεί να εξυπηρετηθεί από το σταθμό y. Υποθέστε ότι ο κάθε σταθμός μπορεί να εξυπηρετήσει το πολύ έναν πελάτη. Χρησιμοποιώντας έναν υπολογισμό μέγιστης ροής, μπορούμε να αποφασίσουμε αν μπορούν να εξυπηρετηθούν όλοι οι πελάτες, ή μπορούμε να πάρουμε μια αντιστοίχιση ενός υποσυνόλου των πελατών σε σταθμούς έτσι ώστε να μεγιστοποιείται ο αριθμός των πελατών που εξυπηρετούνται.

Εδώ θα εξετάσουμε μια εκδοχή του προβλήματος με μια πρόσθετη περιπλοκότητα. Ο κάθε πελάτης προσφέρει διαφορετική ποσότητα χρημάτων για την υπηρεσία. Έστω ότι U είναι το σύνολο των πελατών, και υποθέστε ότι ο πελάτης $x \in U$ είναι πρόθυμος να πληρώσει $v_x \geq 0$ για να εξυπηρετηθεί. Τώρα ο στόχος είναι η εύρεση ενός υποσυνόλου $X \subset U$ το οποίο να μεγιστοποιεί την ποσότητα $\sum_{x \in X} v_x$ όταν γίνεται η αντιστοίχιση των πελατών του συνόλου X σε σταθμούς.

Θεωρήστε την ακόλουθη άπληστη προσέγγιση. Επεξεργαζόμαστε τους πελάτες κατά φθίνουσα αξία (επιλέγοντας αυθαίρετα σε περίπτωση ισοβαθμίας). Κατά την εξέταση του πελάτη x ο αλγόριθμος είτε "υπόσχεται" εξυπηρέτηση του x είτε απορρίπτει τον x με την ακόλουθη άπληστη μέθοδο. Έστω ότι X είναι το σύνολο των πελατών στους οποίους έχουμε μέχρι στιγμής υποσχεθεί εξυπηρέτηση. Προσθέτουμε τον x στο σύνολο X εάν και μόνο εάν υπάρχει τρόπος αντιστοίχισης του $X \cup \{x\}$ στους σταθμούς, ενώ διαφορετικά απορρίπτουμε τον x. Σημειώστε ότι δεν γίνεται μετέπειτα επανεξέταση των πελατών που έχουμε απορρίψει. (Αυτό θεωρείται πλεονέκτημα: Αν χρειάζεται να απορρίψουμε έναν πελάτη που πληρώνει πολλά, τουλάχιστον του το λέμε εγκαίρως.) Παρόλα αυτά, δεν αντιστοιχίζουμε με άπληστο τρόπο τους πελάτες που έχουμε αποδεχθεί σε σταθμούς: η αντιστοίχιση παγιώνεται αφού παγιωθεί το σύνολο των πελατών που έχουμε αποδεχθεί. Παρέχει αυτή η άπληστη προσέγγιση ένα βέλτιστο σύνολο πελατών; Αποδείξτε το ή δώστε ένα αντιπαράδειγμα.

50. Θεωρήστε το ακόλουθο πρόβλημα χρονοπρογραμματισμού. Υπάρχουν m μηχανές, καθεμία από τις οποίες μπορεί να επεξεργαστεί εργασίες, με μία εργασία τη φορά. Το πρόβλημα είναι η αντιστοίχιση εργασιών σε μηχανές (η κάθε εργασία πρέπει να αντιστοιχιστεί σε ακριβώς μία μηχανή) και η διάταξη των εργασιών στις μηχανές έτσι ώστε να ελαχιστοποιείται μια συνάρτηση κόστους. Οι μηχανές λειτουργούν σε διαφορετικές ταχύτητες, αλλά οι εργασίες είναι ταυτόσημες ως προς τις ανάγκες επεξεργασίας τους. Για να το θέσουμε πιο τυπικά, η κάθε μηχανή i έχει μια παράμετρο ℓ_i, και η κάθε εργασία απαιτεί χρόνο ℓ_i αν αντιστοιχιστεί στη μηχανή i.

Υπάρχουν n εργασίες. Οι εργασίες έχουν ίδιες ανάγκες επεξεργασίας, όμως δεν είναι όλες εξίσου επείγουσες. Για κάθε εργασία j μας δίνεται μια συνάρτηση κόστους $c_j(t)$, το οποίο είναι το κόστος της ολοκλήρωσης της εργασίας j σε χρόνο t. Υποθέτουμε ότι τα κόστη είναι μη αρνητικά, και αυξάνονται καθώς αυξάνει το t.

Το χρονοδιάγραμμα αποτελείται από μια αντιστοίχιση εργασιών σε μηχανές, και για κάθε μηχανή το χρονοδιάγραμμα δίνει τη σειρά με την οποία θα εκτελεστούν οι εργασίες. Η εργασία που έχει αντιστοιχιστεί ως πρώτη εργασία στη μηχανή i θα ολοκληρωθεί σε χρόνο ℓ_i, η δεύτερη εργασία θα ολοκληρωθεί σε χρόνο $2\ell_i$, κ.ο.κ. Για ένα χρονοδιάγραμμα S, έστω ότι το $t_S(j)$ συμβολίζει το χρόνο ολοκλήρωσης της εργασίας j σε αυτό το χρονοδιάγραμμα. Το κόστος του χρονοδιαγράμματος είναι $cost(S) = \sum_j c_j(t_S(j))$.

Δώστε έναν αλγόριθμο πολυωνυμικού χρόνου για την εύρεση ενός χρονοδιαγράμματος με ελάχιστο κόστος.

51. Κάποιοι από τους φίλους σας έχουν κουραστεί με το παιχνίδι "Έξι Βαθμίδες του Kevin Bacon" (άλλωστε, λένε, δεν είναι απλώς μια αναζήτηση πρώτα κατά βάθος;) και αποφάσισαν να εφεύρουν ένα παιχνίδι το οποίο να παρέχει κάπως μεγαλύτερη πρόκληση, από αλγοριθμική σκοπιά. Το παιχνίδι αυτό παίζεται με τον εξής τρόπο.

Ξεκινάτε με ένα σύνολο X από n γυναίκες ηθοποιούς και ένα σύνολο Y από n άνδρες ηθοποιούς, και δύο παίκτες P_0 και P_1. Ο παίκτης P_0 ονομάζει μια ηθοποιό $x_1 \in X$, ο παίκτης P_1 ονομάζει έναν ηθοποιό y_1 ο οποίος έχει εμφανιστεί σε ταινία μαζί με την ηθοποιό x_1, ο παίκτης P_0 ονομάζει μια ηθοποιό x_2 που έχει εμφανιστεί σε ταινία μαζί με τον ηθοποιό y_1, κ.ο.κ. Έτσι, οι P_0 και P_1 παράγουν συλλογικά μια ακολουθία x_1, y_1, x_2, y_2, ... έτσι ώστε κάθε ηθοποιός της ακολουθίας να έχει συμπρωταγωνιστήσει με τον ή την ηθοποιό που βρίσκεται ακριβώς πριν από αυτόν. Ο παίκτης P_i ($i = 0$, 1) χάνει αν είναι η σειρά του να παίξει και δεν μπορεί να ονομάσει ένα μέλος του συνόλου του που να μην το έχει ονομάσει προηγουμένως.

Υποθέστε ότι σας δίνεται ένα συγκεκριμένο ζευγάρι τέτοιων συνόλων X και Y, μαζί με πλήρεις πληροφορίες σχετικά με το ποιος έχει εμφανιστεί μαζί με ποιον σε κάποια ταινία. Σε αυτή την περίπτωση, η *στρατηγική* για τον P_i είναι ένας αλγόριθμος που να παίρνει την τρέχουσα ακολουθία x_1, y_1, x_2, y_2, ... και να παράγει μια έγκυρη επόμενη κίνηση για τον παίκτη P_i (υποθέτοντας ότι είναι η σειρά του παίκτη P_i να παίξει). Δώστε έναν αλγόριθμο πολυωνυμικού χρόνου ο οποίος θα αποφασίζει ποιος από τους δύο παίκτες μπορεί να εξαναγκάσει τη νίκη σε κάποια συγκεκριμένη φάση του παιχνιδιού.

Σημειώσεις και πρόσθετα αναγνώσματα

Η ροή δικτύου αναδείχθηκε σε ενιαίο θέμα μέσα από τη δουλειά των Ford και Fulkerson (1962). Αποτελεί πλέον από μόνη της ένα ερευνητικό πεδίο, και θα μπορούσε κανείς εύκολα να αφιερώσει ένα ολόκληρο μάθημα πάνω στο θέμα αυτό· δείτε, για παράδειγμα, τις επισκοπήσεις των Goldberg, Tardos, και Tarjan (1990) και το βιβλίο των Ahuja, Magnanti, και Orlin (1993).

Ο Schrijver (2002) παρέχει μια ενδιαφέρουσα ιστορική αναδρομή για την πρώιμη εργασία των Ford και Fulkerson πάνω στο πρόβλημα της ροής. Για να δώσει πρόσθετα επιχειρήματα σε όσους από εμάς πάντα ένιωθαν ότι το πρόβλημα της Ελάχιστης Αποκοπής είχε μια ελαφρώς καταστροφική χροιά, η επισκόπηση αυτή παραθέτει μια πρόσφατα αποχαρακτηρισμένη αναφορά της αεροπορίας των ΗΠΑ η οποία δείχνει ότι στην αρχική εφαρμογή που αποτέλεσε κίνητρο για τις ελάχιστες αποκοπές το δίκτυο ήταν ένας χάρτης των σιδηροδρομικών γραμμών στη Σοβιετική Ένωση, και ο στόχος ήταν η αποδιοργάνωση των μεταφορών μέσω αυτού του δικτύου.

Όπως αναφέραμε και στο κείμενο, οι διατυπώσεις των προβλημάτων Διμερούς Ταιριάσματος και Ασύνδετων Διαδρομών προηγούνται κατά αρκετές δεκαετίες έναντι του προβλήματος της Μέγιστης Ροής· μόνο μέσα από την ανάπτυξη των δικτύων ροής όλα αυτά τα προβλήματα τοποθετήθηκαν σε ένα κοινό μεθοδολογικό υπόβαθρο. Η πλούσια δομή των ταιριασμάτων σε διμερή γραφήματα έχει πολλούς ανεξάρτητους εξερευνητές· οι P. Hall (1935) και König (1916) είναι πιθανότατα οι δύο που αναφέρονται πιο συχνά. Το πρόβλημα της εύρεσης διαδρομών ασύνδετων ως προς τις ακμές από μια προέλευση προς έναν προορισμό είναι ισοδύναμο με το πρόβλημα της Μέγιστης Ροής όταν όλες οι χωρητικότητες είναι ίσες με 1· αυτή η ειδική περίπτωση λύθηκε (σε μια ουσιαστικά ισοδύναμη μορφή) από τον Menger (1927).

Ο αλγόριθμος Μέγιστης Ροής με Προροή-Προώθηση οφείλεται στον Goldberg (1986), και η αποδοτική υλοποίησή του οφείλεται στους Goldberg και Tarjan (1986).

Σε έναν ιστότοπο που διατηρεί ο Andrew Goldberg μπορεί κανείς να βρει κώδικα υψηλών επιδόσεων για αυτόν και άλλους αλγορίθμους ροής δικτύου.

Ο αλγόριθμος για την τμηματοποίηση εικόνας με χρήση ελάχιστων αποκοπών οφείλεται στους Greig, Porteous, και Seheult (1989), και η χρήση των ελάχιστων αποκοπών έχει γίνει ενεργό θέμα στην έρευνα ως προς την υπολογιστική όραση — για παράδειγμα, δείτε τις επισκοπήσεις των Veksler (1999) και Kolmogorov και Zabih (2004)· θα μελετήσουμε κάποιες πρόσθετες επεκτάσεις αυτής της προσέγγισης στο Κεφάλαιο 12. Ο Wayne (2001) παρουσιάζει πρόσθετα αποτελέσματα πάνω στο πρόβλημα του αποκλεισμού στο μπέιζμπολ, και αποδίδει στον Alan Hoffman την αρχική διάδοση αυτού του παραδείγματος στη δεκαετία του 1960. Πολλές ακόμα εφαρμογές των ροών σε δίκτυα και των αποκοπών αναλύονται στο βιβλίο των Ahuja, Magnanti, και Orlin (1993).

Το πρόβλημα της εύρεσης ενός τέλειου ταιριάσματος με ελάχιστο κόστος αποτελεί ειδική περίπτωση του *προβλήματος Ροής Ελάχιστου Κόστους* (Minimum-Cost Flow Problem), το οποίο είναι εκτός του αντικειμένου αυτού του βιβλίου. Υπάρχουν αρκετοί ισοδύναμοι τρόποι διατύπωσης του προβλήματος Ροής Ελάχιστου Κόστους· σε μία από τις διατυπώσεις μας δίνεται ένα δίκτυο ροής με χωρητικότητες c_e και κόστη C_e στις ακμές· το *κόστος* μιας ροής f είναι ίσο με το άθροισμα του κόστους των ακμών σταθμισμένων ως προς την ποσότητα ροής που μεταφέρουν, $\sum_e C_e f(e)$, και ο στόχος είναι η παραγωγή μιας μέγιστης ροής με το ελάχιστο συνολικό κόστος. Το πρόβλημα Ροής Ελάχιστου Κόστους μπορεί να επιλυθεί σε πολυωνυμικό χρόνο, και έχει επίσης πολλές εφαρμογές· οι Cook και συνεργάτες (1998) και οι Ahuja, Magnanti, και Orlin (1993) αναλύουν αλγορίθμους για το πρόβλημα αυτό.

Αν και η ροή δικτύου μοντελοποιεί προβλήματα δρομολόγησης που μπορούν να αναχθούν στην εργασία της κατασκευής διαδρομών από μία μόνο προέλευση σε μία μόνο απόληξη, υπάρχει μια γενικότερη, και δυσκολότερη, κατηγορία προβλημάτων δρομολόγησης όπου οι διαδρομές θα πρέπει να κατασκευαστούν ταυτόχρονα μεταξύ διαφορετικών ζευγαριών αποστολέων και παραληπτών. Η σχέση μεταξύ αυτών των κατηγοριών προβλημάτων είναι αρκετά λεπτή· θα μελετήσουμε αυτό το ζήτημα, μαζί με τους αλγορίθμους για κάποιους από τους δυσκολότερους τύπους προβλημάτων δρομολόγησης, στο Κεφάλαιο 11.

Σημειώσεις πάνω στις Ασκήσεις Η Άσκηση 8 βασίζεται σε ένα πρόβλημα το οποίο μάθαμε από τον Bob Bland· η Άσκηση 16 βασίζεται σε συζητήσεις με τον Udi Manber· η Άσκηση 25 βασίζεται σε συζητήσεις με τον Jordan Erenrich· η Άσκηση 35 βασίζεται σε συζητήσεις με τους Yuri Boykov, Olga Veksler, και Ramin Zabih· η Άσκηση 36 βασίζεται σε αποτελέσματα των Hiroshi Ishikawa και Davi Geiger, και των Boykov, Veksler, και Zabih· η Άσκηση 38 βασίζεται σε ένα πρόβλημα το οποίο μάθαμε από τον Al Demers· και η Άσκηση 46 βασίζεται σε ένα αποτέλεσμα των J. Picard και H. Ratliff.