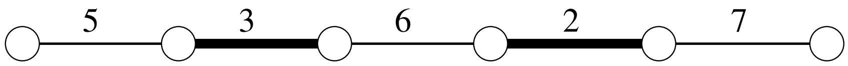
1 Gewichtete Matchings in bipartiten Graphen

Es sei ein ungerichteter Graph $G = (V, E)$ mit ganzzahligen Kantengewichten $w : E \to \mathbb{Z}$ gegeben. Ein *Matching* in G ist eine Teilmenge $M \subseteq E$, so dass keine zwei Kanten aus M einen Endpunkt gemeinsam haben. Ein Matching M heißt *Matching maximaler Kardinalität* falls es in G kein Matching M' mit $|M'| > |M|$ gibt. Das Gewicht $w(M)$ eines Matchings M ist die Summe der Gewichte aller Kanten in M. Das Gewicht $w(p)$ eines augmentierenden Pfades p bezüglich M ist die Summe der Gewichte aller Kanten aus $p \setminus M$ abzüglich der Summe der Gewichte aller Kanten aus $p \cap M$. Das Gewicht des folgenden augmentierenden Pfades wäre also $5 + 6 + 7 - 3 - 2 = 13$.



Mit dieser Definition ändert sich das Gewicht eines Matchings durch Invertieren eines augmentierenden Pfades p genau um $w(p)$.

Wir haben bereits den Algorithmus von Hopcroft und Karp kennengelernt, der in bipartiten Graphen effizient in Zeit $O(\sqrt{|V|}|E|)$ ein Matching maximaler Kardinalität berechnet. Oft gibt es jedoch in einem gegebenen Graphen verschiedene Matchings maximaler Kardinalität, von denen man nicht ein beliebiges, sondern eines mit minimalem oder maximalem Gewicht haben möchte. Im folgenden betrachten wir den Fall, dass ein Matching maximaler Kardinalität und minimalen Gewichts gesucht ist.

1.1 Inkrementelle Berechnung

Die Grundidee zur Lösung dieses Problems ist, nacheinander für $k = 1, 2, \ldots$ jeweils ein Matching M_k zu berechnen, das genau k Kanten enthält und unter allen Matchings mit k Kanten minimales Gewicht hat. Der folgende Satz zeigt, wie M_{k+1} aus M_k berechnet werden kann.

Satz 1 *Sei M_k ein Matching in G mit k Kanten, das unter allen Matchings in G, die genau k Kanten enthalten, minimales Gewicht hat. Sei p ein augmentierender Pfad in G bzgl. M_k, der unter allen augmentierenden Pfaden bzgl. M_k minimales Gewicht hat. Dann enthält $M_{k+1} := M_k \oplus p$, d.h. das durch Invertieren von p aus M_k entstehende Matching, genau $k + 1$ Kanten und hat unter allen Matchings in G, die genau $k + 1$ Kanten enthalten, minimales Gewicht.*

Beweis: Sei M' ein beliebiges Matching mit $k + 1$ Kanten, das unter allen Matchings mit $k + 1$ Kanten minimales Gewicht hat. Betrachte den Graphen $H = (V, M_k \oplus M')$, der nur die Kanten enthält, die entweder in M_k oder in M' enthalten sind (aber nicht in beiden). Alle Knoten haben in H Grad höchstens 2, und die Zusammenhangskomponenten von H können daher nur die folgende Form haben:

1. alternierende (bzgl. M_k und M') Pfade oder Kreise mit gerader Kantenanzahl

2. augmentierende Pfade bzgl. M_k (d.h. durch Invertieren eines solchen Pfades wird M_k um Eins größer bzw. M' um Eins kleiner)

3. augmentierende Pfade bzgl. M' (d.h. durch Invertieren eines solchen Pfades wird M_k um Eins kleiner bzw. M' um Eins größer)

Auf alternierenden Pfaden oder Kreisen mit gerader Kantenanzahl muss das Gewicht der Kanten aus M_k gleich dem Gewicht der Kanten aus M' sein, da sonst durch Invertieren entweder M_k oder M' billiger gemacht werden könnte (Widerspruch). Weiter muss die Anzahl augmentierender Pfade bzgl. M_k (2.) um genau Eins größer sein als die Anzahl augmentierender Pfade bzgl. M' (3.).

Enthält H keinen augmentierenden Pfad bzgl. M' (3.), so sind wir fertig: H enthält genau einen augmentierenden Pfad p bzgl. M_k, und durch Invertieren von p (oder jedes anderen augmentierenden Pfades mit minimalem Gewicht) erhalten wir aus M_k ein Matching mit $k+1$ Kanten und minimalem Gewicht.

Enthält H augmentierende Pfade bzgl. M' (3.), so muss für jedes Paar eines augmentierenden Pfades p bzgl. M_k und eines augmentierenden Pfades q bzgl. M' gelten: die Summe der Kantengewichte in $(p \cup q) \cap M_k$ ist gleich der Summe der Kantengewichte in $(p \cup q) \cap M'$. (Ansonsten könnte man durch Invertieren von p und q entweder M_k oder M' billiger machen, ein Widerspruch.) Also müssen alle augmentierenden Pfade bzgl. M_k das selbe Gewicht $c \in \mathbb{Z}$ und alle augmentierenden Pfade bzgl. M' das selbe Gewicht $-c$ haben. Auch in diesem Fall folgt sofort, dass man ein Matching mit $k + 1$ Kanten und minimalem Gewicht aus M_k durch Invertieren eines einzigen augmentierenden Pfades bzgl. M_k von Gewicht c erhalten kann, also auch durch Invertieren eines beliebigen augmentierenden Pfades mit minimalem Gewicht. □

Der Algorithmus zur Berechnung eines Matchings maximaler Kardinalität und minimalen Gewichts sieht also wie folgt aus:

- setze $M := \emptyset$
- **while** (es existiert augmentierender Pfad bzgl. M) **do**
  suche einen augmentierenden Pfad p mit minimalem Gewicht;
  invertiere p und vergrößere damit M um eine Kante;
  od
- gib M als Matching maximaler Kardinalität und minimalen Gewichts aus

1.2 Berechnung augmentierender Pfade minimalen Gewichts

Nun muss nur noch geklärt werden, wie man in einem bipartiten Graphen einen augmentierenden Pfad mit minimalem Gewicht findet. Wir wollen also unter allen alternierenden Pfaden, die von einem freien Knoten aus V_1 zu einem freien Knoten aus V_2 laufen, einen finden, dessen Gewicht minimal ist. Wir beobachten, dass jeder solche Pfad Kanten aus $E \setminus M$ nur in Richtung von V_1 nach V_2 besucht und Kanten aus M nur in Richtung von V_2 nach V_1. Wir können also die Kanten zu gerichteten Kanten machen. Ferner können wir den Kanten Kosten zuordnen derart, dass das Gewicht eines augmentierenden Pfades genau gleich der Summe der Kosten auf dem Pfad ist: dazu weisen wir jeder Kante $e \in E \setminus M$ die Kosten $c(e) = w(e)$ und jeder Kante $e \in M$ die Kosten $c(e) = -w(e)$ zu. Wenn wir nun noch zwei neue Knoten s und t einfügen, s über gerichtete Kanten mit $c(e) = 0$ mit allen freien Knoten in V_1 verbinden und alle freien Knoten in V_2 über gerichtete Kanten mit $c(e) = 0$ mit t, so lässt sich ein augmentierender Pfad mit minimalem Gewicht offensichtlich durch Suchen eines kürzesten Pfades von s nach t in diesem erweiterten Graphen G' finden. Es bleibt also nur noch zu untersuchen, wie diese Suche nach einem kürzesten Pfad von s nach t in G' erfolgen soll. (In Abschnitt 1.3 finden sich Beispiele dieser Konstruktion.)

1.2.1 Negative Gewichte und Rekalibrierung

Da Kanten in G' auch negatives Gewicht haben können, scheint es so, als ob wir den Algorithmus von Dijkstra nicht einsetzen könnten. Stattdessen wäre der Algorithmus von Bellman und Ford einsetzbar, der kürzeste Pfade in Graphen mit beliebigen Kantengewichten (aber ohne Kreise negativer Länge) in Zeit $O(|V| \cdot |E|)$ berechnet. Die sich ergebende Gesamtlaufzeit für das Finden eines Matchings maximaler Kardinalität und minimalen Gewichts wäre damit $O(|V|^2|E|)$, weil im schlimmsten Fall $|V|/2$ augmentierende Pfade gesucht werden müssen.

Mittels eines geschickten „Rekalibrierungs"-Tricks können wir jedoch die Kosten der Kanten positiv machen und dann doch den Algorithmus von Dijkstra einsetzen. Da der Algorithmus von Dijkstra bei Implementierung mittels Fibonacci-Heaps Worst-Case-Laufzeit $O(|E| + |V| \log |V|)$ hat, ergibt sich damit für das Finden eines Matchings maximaler Kardinalität und minimalen Gewichts (unter allen solchen Matchings) die Gesamtlaufzeit $O(|V|^2 \log |V| + |V| \cdot |E|)$.

Wie funktioniert nun eine solche Rekalibrierung? Betrachten wir zuerst einmal eine beliebige Funktion $F : V \to \mathbb{Z}$ und ersetzen in G' die Kosten $c(u, v)$ jeder Kante (u, v) durch $c'(u, v) := c(u, v) + F(u) - F(v)$. Die Länge jeden Pfades von s nach t ändert sich durch eine solche Rekalibrierung um genau $F(s) - F(t)$. Da diese Änderung nicht vom Pfad abhängt, bleibt ein kürzester Pfad von s nach t auch im Graph mit modifizierten Gewichten ein kürzester Pfad von s nach t. Wenn wir eine Funktion F finden, die alle $c'(u, v)$ positiv macht, so können wir anschließend einen augmentierenden Pfad minimalen Gewichts mit dem Algorithmus von Dijkstra bestimmen.

1.2.2 Wahl der Rekalibrierungsfunktion

Wir benötigen also eine Funktion F, so dass alle $c'(e)$ durch die Rekalibrierung nicht-negativ werden. Eine Möglichkeit ist, $F(u)$ gleich der Länge eines kürzesten Pfades von s nach u zu wählen: damit gilt nämlich für jede Kante (u, v) offensichtlich $F(v) \leq F(u) + c(u, v)$, woraus direkt $c'(u, v) \geq 0$ folgt. Problem dabei ist, dass wir eigentlich die kürzesten Pfade erst mit Hilfe der Rekalibrierung berechnen wollen. Die Lösung des Dilemmas ist, einfach die Informationen aus der Suche nach dem augmentierenden Pfad mit minimalem Gewicht im vorherigen Schritt auszunutzen!

Falls am Anfang, d.h. vor Suche des ersten augmentierenden Pfads, Kanten mit negativem Gewicht existieren, so addiert man einfach auf alle Kantengewichte eine genügend große positive Zahl, um alle Gewichte nicht-negativ zu machen. (Das kann man machen, weil sich dadurch das Gewicht aller Matchings maximaler Kardinalität gleich verändert.) Somit kann die Suche nach dem ersten augmentierenden Pfad bereits mit dem Algorithmus von Dijkstra realisiert werden. Der Graph G' und die Kosten der Kanten ergeben sich dabei wie oben beschrieben. Man lässt den Algorithmus von Dijkstra in G' laufen, bis die Längen der kürzesten

Pfade zu allen Knoten von G' berechnet sind (*single source*-Problem).

Wir bezeichnen mit $c(u, v)$ die aktuellen Kosten der Kanten in G' und mit $d(v)$ die berechnete Länge eines kürzesten Pfades von s nach v. Ferner sei p der kürzeste Pfad in G' von s nach t, den der Algorithmus von Dijkstra berechnet hat. Dann führt der Matching-Algorithmus die folgenden Operationen aus:

- Der p entsprechende augmentierende Pfad in G wird invertiert.
- Die erste und letzte Kante von p wird aus G' gelöscht.
- Für alle Kanten (u, v) von G', die nicht auf p liegen, werden die Kosten $c(u, v)$ durch $c(u, v) + d(u) - d(v) \geq 0$ ersetzt.
- Alle in G' verbleibenden Kanten (u, v) von p (d.h. alle Kanten außer der ersten und der letzten) werden umgedreht (d.h. aus (u, v) wird (v, u)) und erhalten neue Kosten $c(v, u) = 0$.

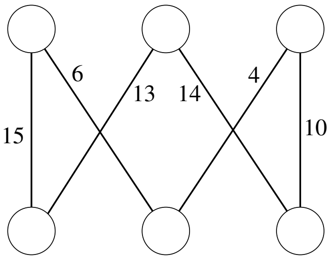
Man überzeugt sich leicht, dass durch diese Modifikationen genau ein Graph G' entsteht, der zur Berechnung des nächsten augmentierenden Pfades mit minimalem Gewicht unter Verwendung des Algorithmus von Dijkstra dienen kann. Der Prozess (Berechnung kürzester Pfade in G' einschließlich eines kürzesten Pfades p von s nach t, Invertieren des p entsprechenden augmentierenden Pfades mit minimalem Gewicht in G, Modifizieren von G' wie oben beschrieben) wird wiederholt, bis irgendwann in G' kein Pfad von s nach t mehr existiert. Dann ist das aktuelle Matching in G genau das gesuchte Matching maximaler Kardinalität, das unter allen solchen Matchings minimales Gewicht hat.

Bemerkung: Da der vorgestellte Algorithmus auch mit negativen Kantengewichten zurecht kommt, kann man auch einfach ein Matching maximaler Kardinalität berechnen, das unter allen solchen Matchings maximales (statt minimales) Gewicht hat. Dazu reicht es aus, zu Beginn des Algorithmus alle Kantengewichte zu negieren.

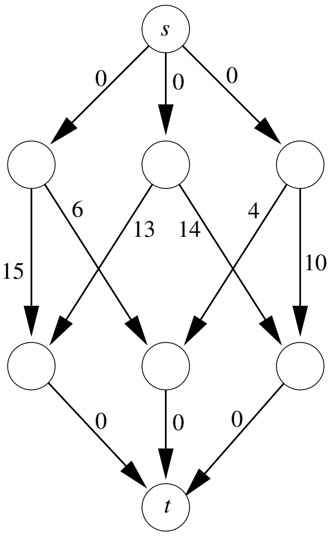
Weitere Informationen: Diese Darstellung des Algorithmus zur Berechnung von Matchings maximaler Kardinalität und minimalen bzw. maximalen Gewichts basiert auf den Seiten 81–85 aus dem Skript "Advanced Algorithms" zu einer Vorlesung von Johan Håstad. Dieses Skript ist als PostScript-Datei im WWW auf Håstad's Homepage `http://www.nada.kth.se/~johanh/` verfügbar.

1.3 Beispiellauf des Algorithmus

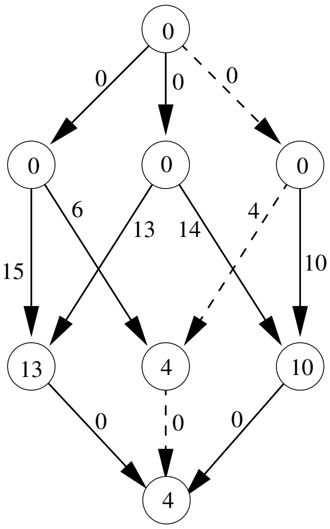
Im folgenden wollen wir den vorgestellten Algorithmus Schritt für Schritt an einem Beispiel durchrechnen.



(1) Eingabegraph G

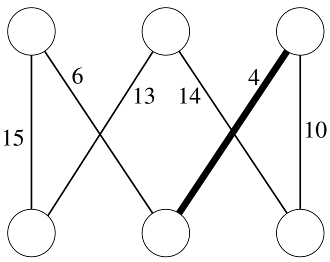


(2) Graph G'

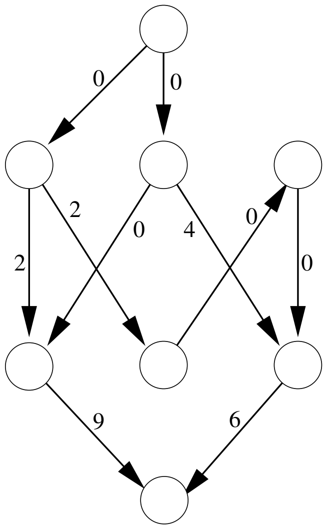


(3) kürzeste Pfade in G'

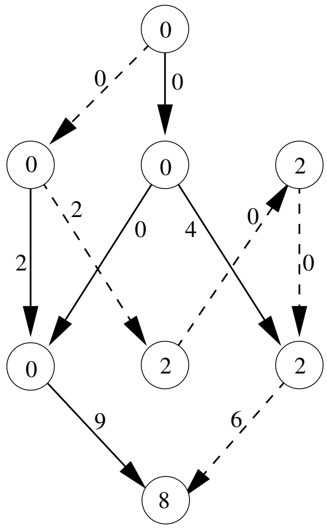
Der Eingabegraph G ist in (1) dargestellt. Die Kantengewichte von G sind alle positiv, so dass anfänglich keine Modifikation der Gewichte nötig ist. Der Graph G' ergibt sich wie in Abschnitt 1.2 beschrieben und ist in (2) dargestellt. Nach der Berechnung kürzester Pfade von s aus in G' mittels des Algorithmus von Dijkstra ergeben sich die in (3) gezeigten Abstände der Knoten von s, und der kürzeste Pfad p von s nach t ist gestrichelt dargestellt. Durch Invertieren des zugehörigen augmentierenden Pfades in G erhalten wir das in (4) dargestellte Matching M_1, das unter allen Matchings in G, die genau eine Kante enthalten, minimales Gewicht hat.



(4) Matching M_1 in G

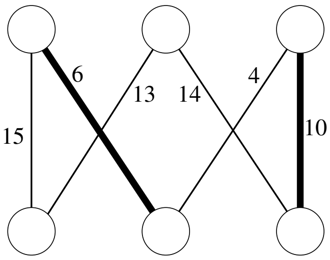


(5) neuer Graph G'

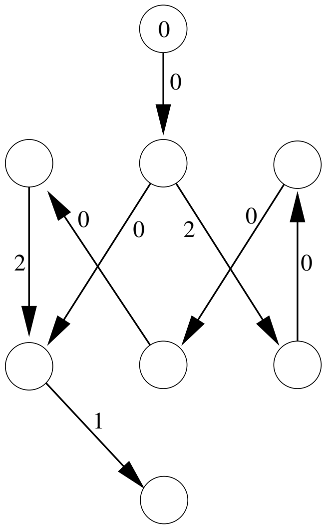


(6) kürzeste Pfade in G'

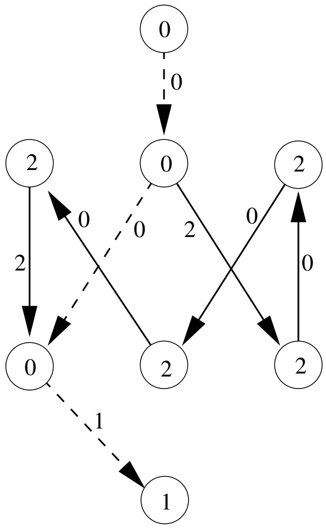
Durch Löschen der ersten und letzten Kante von p, durch Umdrehen der verbleibenden Kante von p und durch Modifikation der Gewichte entsteht aus dem alten Graphen G' in (3) der neue Graph G' in (5). In diesem Graphen wird wieder der Algorithmus von Dijkstra ausgeführt; er liefert diesmal die in (6) dargestellten Abstandswerte sowie einen neuen kürzesten Pfad p (gestrichelt dargestellt) von s nach t. Invertieren des zugehörigen augmentierenden Pfades in G liefert das in (7) dargestellte Matching M_2, das unter allen Matchings in G, die genau zwei Kanten enthalten, minimales Gewicht hat.



(7) Matching M_2 in G

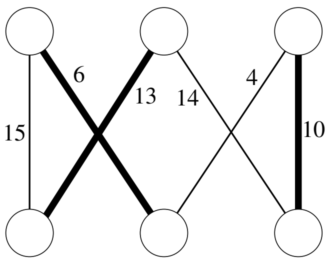


(8) neuer Graph G'



(9) kürzeste Pfade in G'

Wiederum ist der neue Graph G' in (8) abgebildet und das Ergebnis der Berechnung kürzester Pfade in G' in (9). Invertieren des augmentierenden Pfades in G, der dem gestrichelt gezeichneten, kürzesten Pfad in G' entspricht, liefert schließlich das Matching M_3 in (10). Im Graph G', der sich daraus ergibt, ist t von s aus nicht mehr erreichbar. Also hat M_3 maximale Kardinalität und unter allen Matchings maximaler Kardinalität in G minimales Gewicht, nämlich $w(M_3) = 29$.



(10) Matching M_3 in G