Technische Universität München
Institut für Informatik
Lehrstuhl für Effiziente Algorithmen
Dr. Michal Mnuk

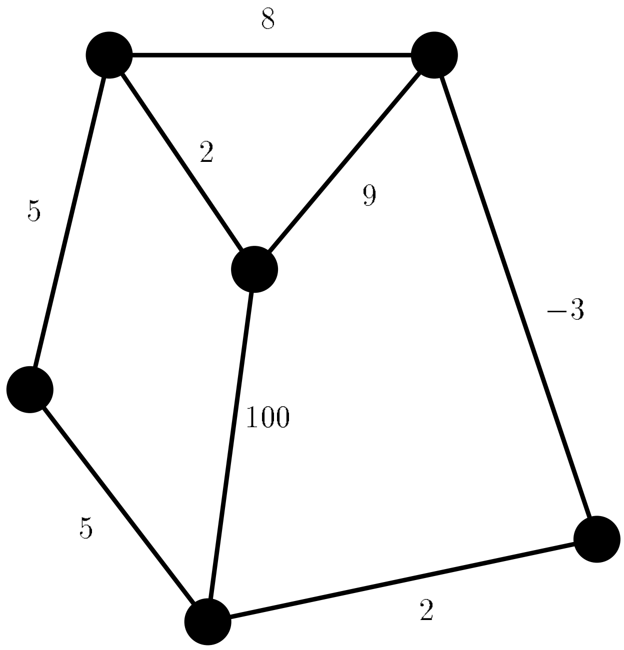
SS 1998
Übungsblatt 9
22. Juli 1998

Grundlegende Algorithmen

Abgabe bis 29. Juni 1998 in der Übung

Aufgabe 26

a) Wenden Sie den Algorithmus von Kruskal auf den folgenden Graphen mit gewichteten Kanten an, um einen MST zu konstruieren. Ist dieser MST eindeutig?

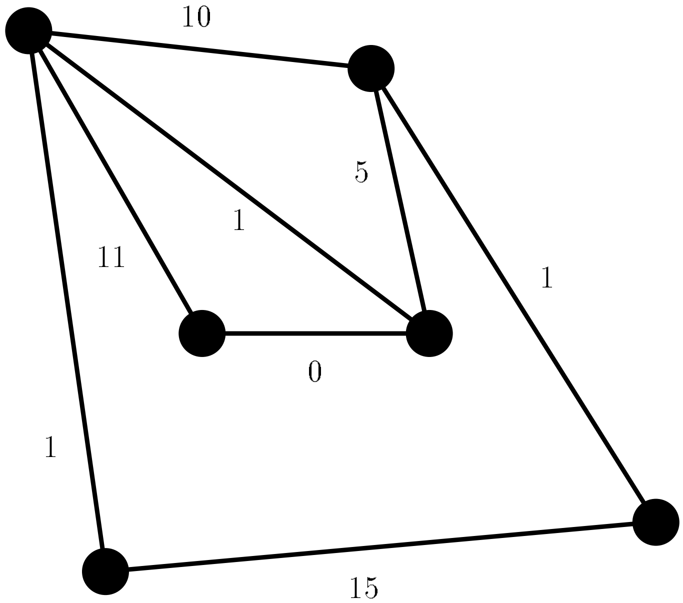


b) Der Algorithmus von Kruskal, wie Sie ihn in der Vorlesung besprochen haben, ist eigentlich erst eine Vorstufe zu einem konkreten Algorithmus (gewählt ausgedrückt stellt er eine Klasse von Algorithmen dar). Man muß nämlich festlegen, wie genau man den Wald, der während des Ablaufs entsteht, als Datenstruktur repräsentiert, und wie man entscheidet, ob eine Kante einen Kreis schließt. Im obigen Beispiel haben Sie dies intuitiv richtig „von Hand“ erledigt.
Finden Sie eine geeignete Datenstruktur, so daß die asymptotische Worst–Case–Laufzeit für einen Graphen mit n Knoten und $m = m(n)$ Kanten $O(m(n) + n \log n)$ beträgt, wobei Sie voraussetzen dürfen, daß die Kanten bereits nach ihrem Gewicht sortiert vorliegen.

[**Hinweis:** Betrachten Sie zwei Knoten zur „Zeit“ $j < m$ als *äquivalent*, wenn sie im gleichen Teilbaum des Waldes liegen, der durch die bisher hinzugenommenen Kanten gebildet wird.]

Aufgabe 27

a) Wenden Sie den Algorithmus von Dijkstra auf den folgenden Graphen mit gewichteten Kanten an, um für alle Knoten den Abstand von s zu berechnen.



b) Auch hier ist es so, daß man genauer spezifizieren muß, wie man jeweils den Knoten im Rand der bereits abgearbeiteten Menge V', welcher minimalen vorläufigen Abstand hat, effizient vorrätig hält und aktualisiert. Der Aufwand kann sich lohnen, wenn $m(n) \notin \Theta(n^2)$. (Beachten Sie, daß $m(n) > n - 1$, wenn der Graph zusammenhängend ist).
Dies geschieht mit der Datenstruktur der Priority–Queue, die sich im Prinzip wie ein Heap verhält (`delete_min`, `insert`, `size`) aber zusätzlich die Operation `decrease_key` ermöglicht, die den Schlüssel eines bereits in der Priority–Queue befindlichen Elements erniedrigen kann.
Wir simulieren die Priority–Queue mit einem modifizierten Heap. Die Operation `decrease_key` geschieht nun so, daß man an der entsprechenden Position den Schlüssel verkleinert und ein `reheap` durchführt.
Zeigen Sie: Die asymptotische Worst–Case–Laufzeit des Dijkstra–Algorithmus, der einen modifizierten Heap verwendet, ist $O(m(n) \log n)$ und bestimmen Sie ein $m(n)$, so daß diese Schranke für die Worst–Case–Laufzeit nicht in $\Omega(n^2)$ liegt, also echt besser ist als quadratisch.
[**Bemerkung:** Um dieses Verhalten ($O(m(n)+n \log n)$) zu verbessern, wurden die Fibonacci–Heaps entwickelt.]

Aufgabe 28

Sei G ein beliebiger zusammenhängender ungerichteter Graph. Wir hatten eingesehen, daß es unsinnig ist, die kürzesten (u, v)–Wege bestimmen zu wollen, wenn es negative Kreise gibt. Geben Sie ein einfache Erweiterung des Floyd–Warshall–Algorithmus an, der an den Werten der Matrixeinträge erkennt, ob ein negativer Kreis existiert.