7 Random Walks

Wir beginnen mit einem motivierenden Beispiel.

7.1 Ein randomisierter Algorithmus für 2-SAT

Bekanntlich ist 3-SAT ein **NP**-vollständiges Problem. Hingegen ist 2-SAT auch deterministisch in Polynomialzeit lösbar. Hierbei kommen also in jeder Klausel nur *zwei* Literale vor. Wir wollen nun einen randomisierten Algorithmus für 2-SAT betrachten.

Im Folgenden sei F eine erfüllbare Formel mit n Variablen (*nicht* Vorkommen von Literalen). Wie man den nachfolgenden Algorithmus modifizieren sollte, damit auch nicht erfüllbare Formeln sinnvoll behandelt werden, wird in Punkt 7.4 kurz diskutiert.

7.1 Algorithmus.

$B \leftarrow \langle$*zufällige Belegung aller* $x_1, \dots, x_n$ *mit Werten*$\rangle$
while $F(B) =$ *false* **do**
  $k \leftarrow \langle$*von* B *nicht erfüllte Klausel in* $F\rangle$
  $x_i \leftarrow \langle$*zufällig gewählte Variable in* $k\rangle$
  $B(x_i) \leftarrow$ **not** $B(x_i)$
od

7.2 Für die Analyse des Algorithmus sei A eine fixierte Variablenbelegung, die F erfüllt. (Im allgemeinen kann es mehrere erfüllende Belegungen geben.) Mit j werde die Anzahl Variablen x bezeichnet, für die $B(x) = A(x)$ ist.

Betrachtet man den linearen Pfad P_{n+1}, dessen Knoten die möglichen Werte $0, 1, \dots, n$ von j sind, so entspricht die jeweils aktuelle Belegung B des Algorithmus einem Knoten. Bei jedem Schleifendurchlauf wird j entweder um 1 erhöht oder um 1 erniedrigt, d. h. jede Veränderung von B entspricht dem Schritt zu einem der beiden Nachbarknoten.

Im schlimmsten Fall beginnt der Algorithmus am Knoten $j = 0$ und endet spätestens dann, wenn zum ersten mal $j = n$ erreicht wird. (Das Ende kann auch schon vorher erreicht sein, wenn es außer A noch andere erfüllende Belegungen gibt.)

7.3 Eine Erhöhung der Knotennummer findet mindestens mit Wahrscheinlichkeit $1/2$ statt. Die Frage nach der Laufzeit des Algorithmus ist gleichbedeutend mit der Frage nach der Anzahl Schritte vom Startknoten bis zum Finden einer erfüllenden Belegung. Diese kann nach oben abgeschätzt werden durch die Anzahl Schritte von Knoten 0 zu Knoten n.

Das ist eine typische Fragestellung bei *Random Walks*: Was ist der Erwartungswert für die Anzahl Schritte, um von einem bestimmten Startknoten zu einem bestimmten Zielknoten zu gelangen?

Bevor wir genauer auf Random Walks eingehen, teilen wir schon ein mal mit die Antwort auf die oben angeschnittenen Fragen mit:

7.4 Der Erwartungswert für die Anzahl Schritte von Knoten 0 nach Knoten n entlang eines einzelnen Pfades ist $\leqslant n^2$ (vgl. die Argumentation in Beispiel 7.13).

Wegen der Markov-Ungleichung ist daher die Wahrscheinlichkeit, dass ein Random Walk der Länge $2n^2$ *nicht* zum Ziel führt höchstens $1/2$. Das motiviert die folgende Variante des Algorithmus:

$B \leftarrow$ ⟨*zufällige Belegung aller* $x_1, \dots, x_n$ *mit Werten*⟩
$m \leftarrow 0$ ⟨*Zähler für die Anzahl Versuche*⟩
while $F(B) =$ *false* **and** $m < 2n^2$ **do**
  $k \leftarrow$ ⟨*von* B *nicht erfüllte Klausel in* F⟩
  $x_i \leftarrow$ ⟨*zufällig gewählte Variable in k*⟩
  $B(x_i) \leftarrow$ **not** $B(x_i)$
  $m \leftarrow m + 1$
od
if $F(B) =$ *true* **then**
  return ⟨F *erfüllbar durch* B⟩
else
  return ⟨F *nicht erfüllbar*⟩
fi

7.2 Random Walks

7.5 In diesem Kapitel bezeichne $G = (V, E)$ einen endlichen, zusammenhängenden, ungerichteten Graphen mit $|V| = n \geqslant 2$ Knoten und $|E| = m$ Kanten. Für einen Knoten u bezeichnen wir mit $\Gamma(u) = \{v \mid (u, v) \in E\}$ die Menge der „Nachbarn“, so dass $d(u) = |\Gamma(u)|$ der Grad von u ist.

Wir stellen uns vor, dass ein Objekt, Agent, Random Walker, Irrläufer, oder wie auch immer man es oder ihn nennen möchte, auf die folgende Art „auf dem Graphen herumläuft“:

- Zu jedem Zeitpunkt befindet er sich an einem Knoten des Graphen, und
- jeder Schritt besteht darin, dass sich der Random Walker von seinem aktuellen Knoten zufällig zu einem anderen bewegt, zu dem eine Kante führt, wobei jede ausgehende Kante mit gleicher Wahrscheinlichkeit gewählt wird.

7.6 Definition

- Mit m_{uv} wird der Erwartungswert bezeichnet für die Anzahl Schritte, die benötigt wird, um von Knoten u erstmals zu Knoten v zu gelangen.
- Die *Wechselzeit* $C_{uv} = m_{uv} + m_{vu}$ ist die erwartete Anzahl Schritte für einen Random Walk, der von u nach u führt und unterwegs mindestens einmal v besucht. ◇

Das folgende Beispiel zeigt, dass man z. B. bei m_{uv} mit intuitiv erscheinenden Annahmen vorsichtig sein muss.

7.7 Beispiel. Es bezeichne $L_n = (V, E)$ den Graphen mit Knotenmenge $V = \{1, \dots, n\}$, bei dem die ersten $n/2$ Knoten eine Clique bilden, an die ein Pfad mit den anderen $n/2$ Knoten „angeklebt“ ist, also $E = \{((u, v) \mid 1 \leqslant u, v \leqslant n/2\} \cup \{(u, u+1) \mid n/2 \leqslant u < n\}$.

Es bezeichne nun u den Knoten $n/2$ und v den Knoten n (siehe Abbildung 7.1 für ein Beispiel).

Wie wir weiter unten sehen werden, ist $m_{uv} \in \Theta(n^3)$ während $m_{vu} \in \Theta(n^2)$ ist. Die beiden Werte können also „weit“ auseinander liegen. (Man versuche, sich schon einmal klar zu machen, woher der Unterschied kommt!)

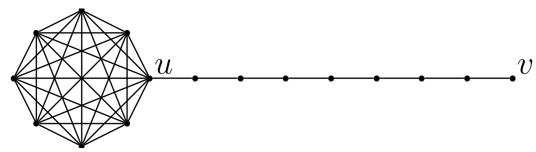


Abbildung 7.1: Der Graph L_{16}.

7.3 Widerstandsnetzwerke

7.8 Jedem ungerichteten zusammenhängenden Graphen G ohne Schlingen entspricht ein Netzwerk $N(G)$ elektrischer Widerstände, indem man jede Kante von G durch einen Widerstand von $1\ \Omega$ ersetzt.

Für zwei verschiedene Knoten u und v in $N(G)$ kann man den *effektiven Widerstand* R_{uv} zwischen ihnen bestimmen. Das ist (natürlich) der Quotient U_{uv}/I_{uv} aus einer zwischen u und v angelegten Spannung und dem dann fließenden Strom.

Für die einfache Reihen- bzw. Parallelschaltung von Widerständen R_k hat man vermutlich schon früher gelernt, dass der sich ergebende Gesamtwiderstand $R = \sum_k R_k$ bzw. $R = 1/(\sum_k 1/R_k)$ ist. In allgemeineren Fällen ist die Bestimmung etwas schwieriger. Was ist z. B. der effektive Widerstand zwischen raumdiagonal gegenüberliegenden Ecken eines Würfels?

Der folgende Satz zeigt, warum wir uns hier für Widerstandsnetzwerke interessieren:

7.9 Satz. *Für beliebige Knoten* u *und* v *in* G *gilt:* $C_{uv} = 2mR_{uv}$.

Als vorbereitenden Schritt zeigen wir einen Zusammenhang von m_{uv} mit einem elektrischen Sachverhalt:

7.10 Lemma. Für zwei Knoten u und v von G bezeichne φ_{uv} die Spannung bei u relativ zu v, die man misst, wenn man in $N(G)$ jedem Knoten x ein Strom von $d(x)$ Ampere zugeführt wird und der gesamte Strom (von insgesamt $2m$ Ampere) an Knoten v abgeführt wird. Dann ist $m_{uv} = \varphi_{uv}$.

In Abbildung 7.2 sind für einen Beispielgraphen alle Ströme und φ_{uv} dargestellt. Die Stromstärke ist durch die Pfeillänge wiedergegeben; zufließende Ströme sind blau und der abfließende rot dargestellt.

7.11 Beweis. Da alle Widerstände $1\ \Omega$ sind, ist der von einem Knoten $u \in V \setminus \{v\}$ zu einem Nachbarknoten $w \in \Gamma(u)$ fließende Strom gerade $\varphi_{uv} - \varphi_{wv}$. Nach der Kirchhoffschen Regel muss für jeden Knoten u der dort zufließende Strom gleich dem von dort abfließenden Strom sein. Also ist

$$d(u) = \sum_{w \in \Gamma(u)} (\varphi_{uv} - \varphi_{wv}) \qquad \text{bzw.} \qquad d(u) + \sum_{w \in \Gamma(u)} \varphi_{wv} = d(u)\varphi_{uv} \ .$$

Andererseits gilt wegen der Linearität des Erwartungswertes für $u \in V \setminus \{v\}$:

$$m_{uv} = \sum_{w \in \Gamma(u)} (1 + m_{wv})/d(u) \qquad \text{bzw.} \qquad d(u) + \sum_{w \in \Gamma(u)} m_{wv} = d(u)m_{uv} \ .$$

Wie man sieht, liegt hier zweimal das gleiche lineare Gleichungssystem vor. Es hat auch eine eindeutige Lösung. (Wer das genauer dargelegt haben will, werfe einen Blick auf die Vorlesungsfolien oder lese die schöne Arbeit von Doyle und Snell (2000).) Also gilt $m_{uv} = \varphi_{uv}$. ∎

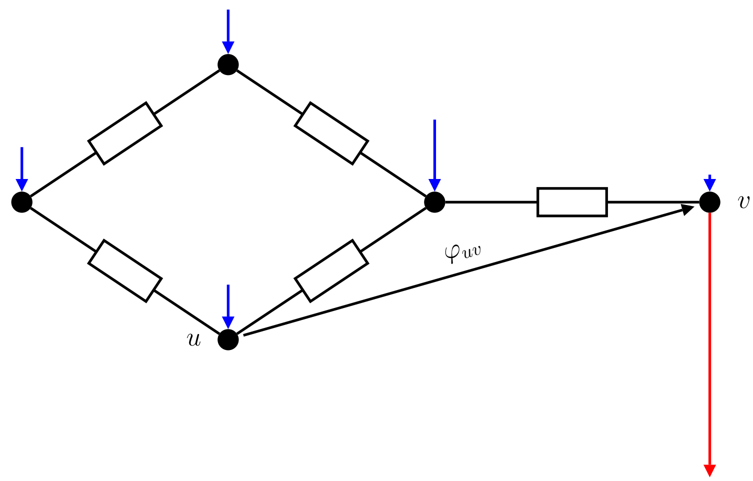


Abbildung 7.2: Ströme für die Messung von φ_{uv}.

7.12 Beweis. (von Satz 7.9) Aus dem vorangegangenen Lemma wissen wir bereits, dass $m_{uv} = \varphi_{uv}$ ist (Abb. 7.3 (a)). Analog ist $m_{vu} = \varphi_{vu}$ die Spannung bei v relativ zu u, wenn dem Widerstandsnetzwerk an jedem Knoten x ein Strom von $d(x)$ Ampere zugeführt wird und der gesamte Strom (von insgesamt $2m$ Ampere) an Knoten u abgeführt wird (Abb. 7.3 (b)). Umdrehen aller Vorzeichen ergibt, dass $m_{vu} = \varphi_{vu}$ auch die Spannung bei u relativ zu v ist, wenn aus dem Widerstandsnetzwerk an jedem Knoten x ein Strom von $d(x)$ Ampere abgeführt wird und der gesamte Strom (von insgesamt $2m$ Ampere) an Knoten u zugeführt wird (Abb. 7.3 (c)).

Widerstandsnetzwerke sind linear. Also ist $C_{uv} = m_{uv} + m_{vu} = \varphi_{uv} + \varphi_{vu}$ die Spannung bei u relativ zu v, wenn an jedem Knoten x jeweils $d(x)$ Ampere zu- und abgeführt werden (also kein Strom zu- oder abgeführt wird) und außerdem an Knoten u $2m$ Ampere zu- und an Knoten v $2m$ Ampere abgeführt werden (Abb. 7.3 (d)).

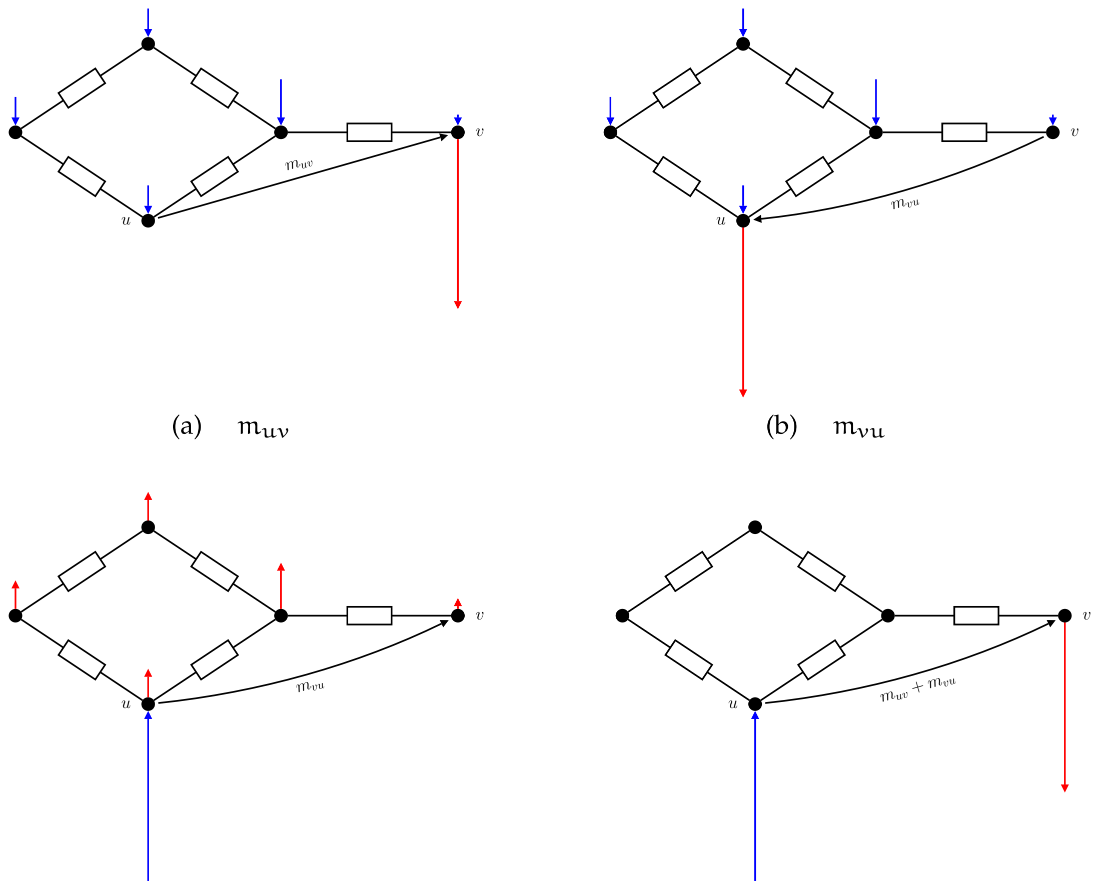
Nach dem Ohmschen Gesetz ist diese Spannung aber gerade $2mR_{uv}$. ∎

7.13 Beispiel. Mit Hilfe der Charakterisierung von m_{xy} mittels φ_{xy} kann man z. B. für den Graphen L_n aus Beispiel 7.7 die Werte m_{uv} und m_{vu} bestimmen.

m_{uv}: An jedem der ersten $n/2$ Knoten werden $n/2$ Ampere injiziert. Dieser Gesamtstrom von $\Theta(n^2)$ Ampere fließt über u und weitere $n/2$ Knoten nach v und verursacht an allen Widerständen unterwegs jeweils einen Spannungsabfall von $\Theta(n^2)$ Volt. Diese summieren sich also zu $\Theta(n^3)$.

m_{vu}: An jedem der Knoten $1 + n/2, 2 + n/2, \dots, n - 1$ werden 2 Ampere injiziert, die zu Knoten u fließen. Also fließen durch den Widerstand zwischen Knoten $n - i$ und $n - i - 1$ ($0 \leqslant i \leqslant n/2 + 1$) gerade $2(i + 1)$ Ampere. Die entsprechenden Spannungsabfälle summieren sich also zu $\Theta(n^2)$.

7.14 Korollar. Für jeden Graphen mit n Knoten und beliebige Knoten u und v gilt: $C_{uv} < n^3$.



(a) m_{uv} (b) m_{vu}

(c) m_{vu} bei umgekehrten Vorzeichen (d) Summe von (a) und (c)

Abbildung 7.3: Die Beweisschritte am Beispiel eines Graphen mit 5 Knoten

7.15 **Beweis.** Ein ungerichteter Graph mit n Knoten hat höchstens $m \leqslant n(n-1)/2$ Kanten. Der maximale effektive Widerstand R_{uv} ist nach oben beschränkt durch die Länge der kürzesten Wege von u nach v. Reihenschaltung ist der schlimmste Fall (man überlege sich das). Dann ist $R_{uv} \leqslant n-1$ und folglich insgesamt $C_{uv} = 2mR_{uv} < n^3$. ∎

7.4 Randomisierte Algorithmen für Zusammenhangstests

7.16 Wir betrachten nun das Problem, für einen vorgegebenen ungerichteten Graphen und zwei seiner Knoten festzustellen, ob sie zu der gleichen Zusammenhangskomponente gehören.

Dieses Problem wird üblicherweise mit USTCON bezeichnet (engl. *undirected* s–t *connectivity*). Die Aufgabe besteht mit anderen Worten darin, für zwei beliebige Knoten s und t eines ungerichteten Graphen festzustellen, ob sie durch einen Pfad miteinander verbunden sind.

Dieses Problem taucht im Zusammenhang mit mehreren Problemen aus der Graphentheorie auf. Ohne genauer darauf einzugehen, sei auch noch mitgeteilt, dass es auch in der Komplexitätstheorie von Bedeutung ist. (USTCON ist vollständig für **SL**. Für diese Klasse hat Reingold (2005) gezeigt: **SL** $=$ L.)

7.17 ALGORITHMUS. Wir beschreiben einen randomisierten Algorithmus A, der USTCON mit logarithmischem Platz (und in polynomialer Zeit) mit einseitigem Fehler „löst“: Es wird ein Random Walk auf dem eingegebenen Graphen simuliert, der bei s startet, und zwar maximal $2n^3$ Schritte. Wenn man dabei auf Knoten t trifft, wird sofort YES ausgegeben. Geschieht das nicht, wird am Ende NO ausgegeben.

Dass Algorithmus 7.17 logarithmischen Platz und polynomiale Zeit benötigt, sollte klar sein.

7.18 LEMMA. Die Wahrscheinlichkeit, dass A aus Algorithmus 7.17 fälschlicherweise NO ausgibt, ist höchstens $1/2$.

7.19 BEWEIS. Es ist nur der Fall zu betrachten, dass s und t in der gleichen Zusammenhangskomponente liegen. Wegen Korollar 7.14 ist offensichtlich der Erwartungswert $m_{st} \leqslant n^3$. Nach der Markov-Ungleichung aus Satz 4.10 ist daher die Wahrscheinlichkeit, dass ein Random Walk mehr als doppelt solange benötigt, um von s nach t zu gelangen, höchstens $1/2$. ■

7.20 Zum Abschluss dieses Kapitels wollen wir noch die Variante des obigen Problems für gerichtete Graphen betrachten. Sie wird üblicherweise mit STCON bezeichnet (engl. s–t *connectivity*). Die Aufgabe besteht darin, für zwei beliebige Knoten s und t eines gerichteten Graphen mit anderen Worten darin festzustellen, ob sie durch einen Pfad miteinander verbunden sind.

Die offensichtliche Anpassung von Algorithmus 7.17 an den gerichteten Fall ist ungeeignet: Der zu untersuchende Graph kann „Sackgassen“ enthalten, aus denen man mit dem Random Walk nicht mehr herauskommt.

Dies ist aber sozusagen das einzige Problem, dem man entkommt, indem man auf geeignete, noch genau zu spezifizierende Weise gelegentlich an den Ausgangspunkt s zurückspringt, wenn man noch nicht t erreicht hat.

7.21 ALGORITHMUS. Es werden abwechselnd immer wieder die beiden folgenden Phasen ausgeführt, bis der Algorithmus in einer von ihnen anhält:

1. Ausgehend von s wird in G ein Random Walk der Länge maximal $n-1$ durchgeführt. Wird dabei t erreicht, hält der Algorithmus sofort mit der Ausgabe YES an.

2. Es werden $\log n^n = n \log n$ Zufallsbits „gewürfelt“. Wenn sie alle 0 sind, hält der Algorithmus sofort mit der Ausgabe NO an.

7.22 Lemma.

a) Algorithmus 7.21 kann so implementiert werden, dass der Platzbedarf stets kleiner gleich $O(\log n)$ ist.
b) Wenn kein Pfad von s nach t existiert, gibt der Algorithmus nie YES aus, also stets NO, sofern er hält.
c) Wenn ein Pfad von s nach t existiert, gibt der Algorithmus mit einer Wahrscheinlichkeit größer gleich $1/2$ die Antwort YES aus.

7.23 Beweis.

a) In Phase 1 des Algorithmus muss man im wesentlichen speichern, wie lang der bisherige Random Walk war und bei welchem Knoten man sich gerade befindet. Dafür reichen $O(\log n)$ Bits.

   Phase 2 kann implementiert werden, indem man zählt, wieviele Bits bislang gewürfelt wurden, und ob alle gleich 0 waren. Für den Zähler genügen $\log(n \log n) \in O(\log n)$ Bits.
b) Wenn kein Pfad existiert, wird kein Random Walk von s nach t führen, also wird auch nie YES ausgegeben.
c) Angenommen, es gibt einen Pfad von s nach t. Da es insgesamt im Graphen höchstens n^n Pfade der Länge $n-1$ geben kann, ist die Wahrscheinlichkeit, dass in Phase 1 ein Pfad von s nach t gefunden wird, mindestens n^{-n}. Die Wahrscheinlichkeit, dass in Phase 2 die falsche Antwort gegeben wird, ist $(1-n^{-n})n^{-n} \leqslant n^{-n}$.

   Es sei X die Zufallsvariable, die angibt, in welchem Durchlauf die richtige Antwort gegeben wird, und p die Wahrscheinlichkeit, dass überhaupt die richtige Antwort gegeben wird. Dann ist

$$p = \mathbf{P}(X \geqslant 1) = \mathbf{P}(X = 1) + \mathbf{P}(X \geqslant 2) \geqslant n^{-n} + (1-n^{-n})^2 p \geqslant n^{-n} + (1-2n^{-n})p \ .$$

   Auflösen nach p ergibt sofort $p \geqslant 1/2$.

∎

Zusammenfassung

1. Es gibt Zusammenhänge zwischen Random Walks und elektrischen Widerstandsnetzwerken.

2. Random Walks kann man benutzen, um in Graphen zwei Knoten auf Verbundenheit zu testen.

Literatur

Doyle, Peter G. und J. Laurie Snell (2000). *Random walks and electric networks*. arXiv preprint available at the URL: `http://arxiv.org/abs/math.PR/0001057` (siehe S. 66).

Reingold, Omer (2005). „Undirected ST-connectivity in log-space“. In: *Proceedings of the 37th Annual ACM Symposium on Theory of Computing (STOC)*. Hrsg. von Harold N. Gabow und Ronald Fagin. ACM, S. 376–385. ISBN: 1-58113-960-8 (siehe S. 69).

