Der Satz von Savitch

Peter Faymonville

12. August 2010

Zusammenfassung

Der Satz von Savitch beschreibt einen Algorithmus, um Erreichbarkeit in gerichteten Graphen zwischen zwei gegebenen Knoten mit beschränkter Pfadlänge zu entscheiden. Er basiert auf der Erkenntnis, dass man jeden Pfad in zwei gleich lange Unterpfade teilen kann, es also stets einen Mittelknoten gibt. Dadurch lässt sich das Problem in quadratisch-logarithmischem Platz lösen. Als Konsequenz ergibt sich, dass $\mathsf{NL} \subseteq \mathsf{L}^2$ und $\mathsf{NPSPACE} = \mathsf{PSPACE}$.

Inhaltsverzeichnis

1 Einführung

Der Satz von Savitch gibt uns einen intuitiven Algorithmus zur Entscheidung des Erreichbarkeitsproblems für gerichtete Graphen. Genauer lässt sich die Frage, ob es zwischen zwei Knoten einen Pfad mit Maximallänge k gibt, in quadratisch-logarithmischen Platz lösen. Da dieses Problem NL-vollständig ist, gibt uns dies zunächst eine wichtige Erkenntnis für die logarithmischen Platzklassen: $\mathsf{L} \subseteq \mathsf{NL} \subseteq \mathsf{L}^2$. Im weiteren hat es aber Auswirkungen auf die gesamte Platzhierarchie oberhalb von NL: Indem man den gegebenen Algorithmus die Erreichbarkeit von akzeptierenden Endzuständen im Konfigurationsgraph von nichtdeterministischen Turingmaschinen testen lässt, fallen die nichtdeterministischen Platzklassen (oberhalb von NL) mit den deterministischen Platzklassen zusammen. Der Nichtdeterminismus ist weniger mächtig als in der Zeithierarchie, da der benötigte Platz der deterministierten Maschine nur quadratisch mehr Platz benötigt. Daher gilt zum Beispiel $\mathsf{NPSPACE} = \mathsf{PSPACE}$.

2 Savitch's Algorithmus

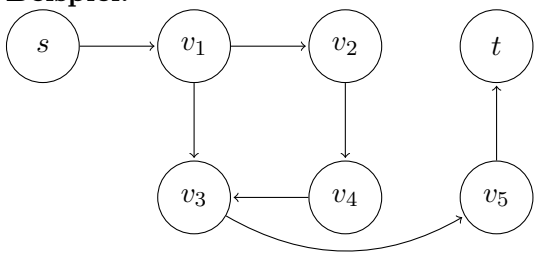
Hinter diesem Algorithmus steckt eine einfach anmutende Kernidee: Man kann für jeden Pfad zwischen zwei Knoten einen Mittelknoten finden, der (annähernd) gleich weit von beiden entfernt ist. Für die beiden resultierenden Teilpfade kann man dasselbe tun, bis man durch fortgesetzte Anwendung dieses rekursiven Prinzips nur noch die Existenz direkter Kanten zwischen den Knoten testen muss.

Definition 2.1. REACHMAX $(s, t, k) \Leftrightarrow \exists$ *Pfad von* s *nach* t *mit Maximallänge* 2^k

Wir berechnen wir das Prädikat REACHMAX mit Hilfe des folgenden Algorithmus:

```
1: function REACHMAX(s, t, k): Boolean
2:     if  k < 0  then return false
3:     else if  (s,t) ∈ E then return true
4:     else
5:         for v ∈ V − {s,t}  do
6:             if reachMax(s,v,k − 1) then
7:                 if reachMax(v,t,k − 1) then
8:                     return true
9:     return false
```

Beispiel:



Wir betrachten nun die Ausführung, die zum Ziel führt, indem der Pfad von s nach t über v_1, v_3, v_5 für die Pfadlänge $2^2 = 4$ betrachtet wird. Alle anderen möglichen Mittelpunkte, die in der for-Schleife betrachtet werden, führen nicht zum Ziel.

REACHMAX $(s, t, 2)? \rightarrow (s, v_3, 1)?, (v_3, t, 1)?$
$\rightarrow (s, v_1, 0)?, (v_1, v_3, 0)?, (v_3, t, 1)?$
$\rightarrow (s, v_1, 0)!, (v_1, v_3, 0)!, (v_3, t, 1)?$
$\rightarrow (s, v_1, 0)!, (v_1, v_3, 0)!, (v_3, v_5, 0)?, (v_5, t, 0)?$
$\rightarrow (s, v_1, 0)!, (v_1, v_3, 0)!, (v_3, v_5, 0)!, (v_5, t, 0)!$

2.1 Platzkomplexität

Theorem 1. REACH $\in \mathsf{SPACE}(O(log^2 n))$

Wir zeigen, dass der beschriebene Algorithmus als determistische 2-Bandturingmaschine realisiert werden kann.

- Repräsentation von G liegt auf dem Eingabeband
- Arbeitsband 1: Aktuell betrachtetes v (foreach)
- Arbeitsband 2: Rekursionsstack

Analyse des benötigten Bandplatzes:

- Rekursionstiefe: $log\ n$ Aufrufe (Halbierung der Distanz)
- Größe des Aufrufs: $3 \times log\ n$ (Kodierung s,t,k)

Damit ist gezeigt, dass die Berechnung $O(log^2 n)$ Platz benötigt.

3 Konsequenzen

Wir wissen bereits, dass das Erreichbarkeitsproblem für gerichtete Graphen NL-vollständig ist. Daher gilt:

$$\mathsf{NL} \subseteq \mathsf{SPACE}(O(log^2 n))$$

$$\mathsf{L} \subseteq \mathsf{NL} \subseteq \mathsf{L}^2$$

3.1 Das Verhältnis von NSPACE und SPACE

Der Algorithmus lässt sich auch dazu einsetzen, dass Verhältnis von deterministischem und nichtdeterministischem Platz zu charakterisieren. Hierfür zeigen wir, dass jede nichtdeterministische Turingmaschine M in quadratischem Platzverbrauch determinisiert werden kann. Hierzu berechnen wir für eine konkrete Eingabe w nach Bedarf den Konfigurationsgraphen von M und testen mithilfe des Algorithmus die Erreichbarkeit von akzeptierenden Endzuständen.

Die Analyse der NDTM M und eine Kardinalitätsabschätzung des Konfigurationsgraphens liefern dann die nötigen Argumente für den Beweis, dass

Theorem 2. $\mathsf{NSPACE}(f(n)) \subseteq \mathsf{SPACE}(f^2(n))$

gilt.

Die NDTM M akzeptiert eine Sprache L und ist charakterisiert durch:

- Eingabeband, Ausgabeband
- s Arbeitsbänder
- Zustandsraum Q
- c Elemente des Bandalphabets
- n Länge der Eingabe w

Die Eigenschaften des Konfigurationsgraph für ein konkretes w sind wie folgt:

- Graph ist azyklisch
- $|Q|$ Zustände
- $c^{s+2 \cdot f(n)}$ Bandkonfigurationen, wenn $L \in \mathsf{NSPACE}(f(n))$
- $n \cdot f(n)^{s+1}$ Bandkopfpositionen

Daraus ergibt sich die folgende obere Schranke für die Anzahl der Konfigurationen:

$$|Q| \cdot c^{s+2 \cdot f(n)} \cdot n \cdot f(n)^{s+1}$$

Wir beweisen nun:

$$\mathsf{NSPACE}(f(n)) \subseteq \mathsf{SPACE}(f^2(n))$$

für eine ordentliche[1] Komplexitätsfunktion $f(n) \geq O(log\ n)$.

1. Gegeben eine NDTM M, die eine Sprache $L \in \mathsf{NSPACE}(f(n))$ akzeptiert.
2. Modifiziere Savitch's Algorithmus:
3. Beim Test auf $(s,t) \in E$, benutze Transitionsrelation von M, die auf dem Eingabeband liegt.
4. Der Konfigurationsgraph hat $k^{log\, n+f(n)}$ Knoten, für ein konstantes k.
5. Daher reicht $O(f^2(n))$ Platz, um die Erreichbarkeit von Endzuständen zu entscheiden.

Dies bedeutet insbesondere für die polynomielle Platzklasse NPSPACE, dass sie äquivalent zu PSPACE ist, da das Quadrat einer polynomiellen Funktion ebenso ein Polynom ist.

$$\mathsf{NPSPACE} = \mathsf{PSPACE}$$

[1] monoton steigend, berechenbar von einer TM in $O(|n| + f(n))$ Zeit, $O(f(n))$ Platz