Jun.-Prof. Dr. Bernhard Beckert
Ulrich Koch

UNIVERSITÄT KOBLENZ · LANDAU

Theoretische Informatik II
Wintersemester 2007/2008
12. Aufgabenblatt

Ausgabe: 10. 02. 2008 Besprechung: 13. 02. 2008

---

1 Lambda-Kalkül

Geben Sie einen λ-Term *isprime* an, so dass für alle $n \in \mathbb{N}_0$ gilt:

$$isprime\ \overline{n} \equiv \begin{cases} true, & \text{falls } n \text{ prim ist} \\ false & \text{sonst} \end{cases}$$

Sie dürfen alle λ-Terme verwenden, die bisher in der Vorlesung oder in der Übung definiert wurden — auch relationale Operatoren, logische Operatoren und den Fixpunkt-Kombinator Y. Zusätzlich dürfen Sie den λ-Term *isdiv* verwenden, für den gilt:

$$isdiv\ \overline{k}\ \overline{n} \equiv \begin{cases} true, & \text{falls } k \text{ ein Teiler von } n \text{ ist} \\ false & \text{sonst} \end{cases}$$

Lösung:

$isprime : \mathbb{N}_0 \to \{true, false\}$ lässt sich folgendermaßen definieren:

$$isprime(n) = \begin{cases} false, & n < 2 \\ check(n, n-1) & \text{sonst} \end{cases}$$

Hier wird $check : \mathbb{N}_0^2 \to \{true, false\}$ als Hilfsfunktion verwendet. $check(n, m)$ liefert genau dann *true*, wenn keine der Zahlen $2, \ldots, m$ ein Teiler von n ist:

$$check(n, m) = \begin{cases} true, & m < 2 \\ \neg isdiv(m, n) \wedge check(n, m-1) & \text{sonst} \end{cases}$$

Dadurch ergibt sich:

$$isprime =_{def} \lambda n\,.\,if\ n < \bar{2}\ then\ false\ else\ check\ n\ (pred\ n)$$

Dabei:

$$check =_{def} Y\Big(\lambda f\,.\,\lambda n\,.\,\lambda m\,.\,if\ m < \bar{2}\ then\ true\ else\ \neg(isDiv\ m\ n) \wedge \big(f\ n\ (pred\ m)\big)\Big)$$

2 Komplexität

Das Problem SET PACKING ist wie folgt definiert:

> Gegeben: $C = \{S_1, \ldots, S_m\}$, wobei jedes S_i eine endliche Menge ist, sowie eine Zahl $l \geq 1$.
>
> Gefragt: Gibt es eine l-elementige Teilmenge D von C, so dass die Elemente von D paarweise disjunkt sind?

Beweisen Sie, dass SET PACKING $\mathcal{NP}$-vollständig ist, indem Sie

1. zeigen, dass es in $\mathcal{NP}$ liegt, und
2. CLIQUE (das Problem, ob ein Graph G eine Clique der Größe k hat) polynomiell auf SET PACKING reduzieren.

Lösung:

1. $C = \{S_1, \ldots, S_m\}$ und l sind gegeben. Die NTM rät eine Teilmenge $D = \{S_{i_1}, \ldots, S_{i_l}\}$ von C und prüft dann folgendermaßen, ob die Elemente von D paarweise disjunkt sind:

```
for j := 1 ... l − 1 do
   for k := j + 1 ... l do
      if S_{i_j} ∩ S_{i_k} ≠ ∅ then
         return false
      end
   end
end;
return true
```

   Um den Zeitaufwand für diesen Algorithmus abzuschätzen, müssen wir wissen, in welcher Zeit der Test $S_{i_j} \cap S_{i_k} \neq \emptyset$ abläuft. Dafür ist wichtig, wie die Eingabe (C, l) des Algorithmus codiert wird: Die Elemente der S_i werden als natürliche Zahlen zur Basis 2 (ohne führende Nullen) dargestellt, l ebenso. Jeder Test $S_{i_j} \cap S_{i_k} \neq \emptyset$ hat einen Zeitaufwand in $O(|S_{i_j}| \cdot |S_{i_k}| \cdot \log x)$, wobei x die größte Zahl in $S_{i_j} \cup S_{i_k}$ ist. l und die $|S_{i_j}|$ können von Fall zu Fall sehr unterschiedlich sein. In jedem Fall gilt aber abhängig von der Länge n der Eingabe: $\forall i_j(|S_{i_j}| = O(n))$. Der Aufwand für jeden Test ist deshalb in $O(n^2 \log z)$, wobei z die größte Zahl ist, die in einem S_{i_j} vorkommt. Da der gesamte Algorithmus den Test $O(l^2)$-mal ausführt und $l \leq m \leq n$ vorausgesetzt werden kann[1], ist der Gesamtaufwand in $O(n^4 \log z)$, wegen $\log z = O(n)$ in $O(n^5)$.

2. (G, k), wobei G die Darstellung eines Graphen ist, wird wie auf dem 11. Aufgabenblatt codiert, nämlich als $m'\#E'\#k'$, wobei gilt:

   - m' ist die Binärdarstellung von m ohne führende Nullen.
   - $E' \in \{0,1\}^*$ ist eine binäre $(m \times m)$-Matrix, die an der Stelle i, j genau dann eine 1 enthält, wenn v_i und v_j benachbart sind.

[1] Für $l > m$ wäre die Aufgabe trivial und damit in polynomieller Zeit zu lösen.

- k' ist die Binärdarstellung von k ohne führende Nullen.

Die reduzierende Funktion f bildet Wörter, die nicht die Form (G, k) haben, auf ε ab. (G, k) dagegen wird auf (C, k) abgebildet. Dabei gilt:

Sei $G = (V, E), V = \{v_1, \ldots, v_m\}$.

Konstruiere $C = \{S_1, \ldots, S_m\}$ wie folgt: $S_i = \{(v_i, v_j), (v_j, v_i) \mid (v_i, v_j) \notin E\}$

Für alle $i, j \in \{1, \ldots, m\}$ gilt: S_i und S_j haben genau dann ein gemeinsames Element, wenn es *keine* Kante zwischen v_i und v_j gibt. Also wird für eine Clique der Größe k eine Teilmenge $D = \{S_{i_1}, \ldots, S_{i_k}\}$ konstruiert, deren Elemente paarweise disjunkt sind.

Damit ist (C, k) genau dann eine Instanz von SET PACKING, wenn (G, k) eine Instanz von CLIQUE ist.

Zum Zeitaufwand von f: Jede der konstruierten Mengen S_i hat $O(m)$ Elemente (Kanten). Diese Kanten können tatsächlich mit einem Zeitaufwand in $O(m)$ berechnet werden, da der Test $(v_i, v_j) \notin E$ in konstanter Zeit abläuft[2]. Die Länge der Darstellung einer Kante ist in $O(\log m)$, weil m die größte Zahl ist, die in einer Kante vorkommt (s. 1.). Damit ist der Gesamtzeitaufwand für die Erzeugung und Ausgabe aller S_i in $O(m^2 \log m)$. Außerdem muss k' von der Eingabe in die Ausgabe kopiert werden; der Aufwand dafür ist gleich $|k'| = O(\log k)$. Damit ist der Zeitaufwand für die Konstruktion von (C, k) in $O(m^2 \log m + \log k)$.

Die Länge n der Eingabe (G, k) ist in $O(\log m + m^2 + \log k) = O(m^2 + \log k)$ (s. 11. Aufgabenblatt). Damit ist der Zeitaufwand von f in $O(n^2)$.

[2]Begründung: E' wird linear codiert, und der Lesekopf der NTM wird bei jeder Änderung von j nur um ein Feld bewegt (vgl. 11. Aufgabenblatt).