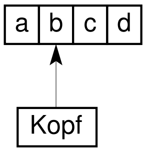
Technische Universität Chemnitz Sommersemester 2002
Professur Theoretische Informatik
Prof. Dr. A. Goerdt, Dipl.-Inf. F. Schädlich, Dipl.-Inf. A. Lanka

9. Übung „Theoretische Informatik II“
Lösungsvorschläge

Aufgabe 1

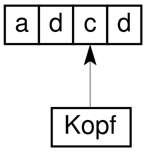
Wir möchten den Platzbedarf unserer gegebenen Turingmaschine um den Faktor C verkleinern. Dazu bilden wir auf dem Eingabeband Blöcke der Länge C. Jeden dieser Blöcke wollen wir als ein neues Zeichen des Bandalphabetes kodieren. Wir sehen ein, daß sich dadurch der Platzbedarf der Turingmaschine um den Faktor C verkleinert.

Es bleibt noch zu untersuchen, wie groß das Bandalphabet unserer modifizierten Turingmaschine dadurch wird. Es muß schließlich unabhängig von der Eingabelänge sein. Sehen wir uns beispielhaft einen solchen Block an.



Dieser Block wird nun so kodiert:
aKbcd

Das K gibt dabei die Kopfposition an. Es steht immer *vor* dem Zeichen, über dem der Kopf sich gerade befindet. Angenommen wir befinden uns im Zustand z und $(z', d, R) \in \delta(z, b)$ ist Teil der Überführungsfunktion. Das führt uns zu dem neuen Block



dem das Zeichen adKcd entspricht. Entsprechend wird die Überführungsfunktion angepaßt. Aus $(z', d, R) \in \delta(z, b)$ wird dann $(z', \text{adKcd}, N) \in \delta(z, \text{aKbcd})$.

Wieviele mögliche Blöcke gibt es nun? Für jede der C Positionen gibt es $|\Gamma|$ viele Möglichkeiten. Daher gibt es $|\Gamma|^C$ viele verschiedene Blöcke. Wir kodieren jedoch die Kopfposition mit in unsere neuen Zeichen. Es gibt C mögliche Positionen und zusätzlich noch die Alternative, daß der Kopf sich überhaupt nicht im betrachteten Block befindet. Insgesamt haben wir also $(C + 1) \cdot |\Gamma|^C$ viele neue Zeichen (das sind zwar viele, aber nur konstant viele), für die die Überführungsfunktion entsprechend angepaßt werden muß. Unsere neue TM M' erhält die gleiche Eingabe wie M, faßt sie zu Blöcken zusammen, rechnet mit der neuen Überführungsfunktion und macht schließlich aus den Blöcken wieder „richtige“ Zeichen.

Der Satz gilt auch, wenn wir statt des Platzes die Zeit reduzieren wollen. Dazu fassen wir die Eingabe nach obigem Prinzip zu Blöcken, diesmal der Länge $3 \cdot C$ (wir sehen später warum), zusammen. Danach simulieren wie die Turingmaschine auf allen möglichen Blöcken

$((C+1)\cdot|\Gamma|^C)$ und ermitteln die „Ergebnisblöcke" nach jeweils $3\cdot C$ Schritten der originalen TM M. Nun bauen wir uns eine neue Überführungsfunktion, die uns für jeden Block sofort das Ergebnis nach $3\cdot C$ Schritten auf das Band schreibt. Es kann natürlich sein, daß die TM während ihrer $3\cdot C$ Schritte den Block verläßt und in einen Nachbarblock wechselt. Allerdings kann sie wegen der großen Blocklänge nur in maximal zwei Blöcken arbeiten. Wenn wir das Ergebnis der Simulation auf das Band schreiben wollen brauchen wir daher maximal drei Schritte:

1. Den aktuellen Block lesen und „merken" (zum Beispiel im Zustand) und gegebenenfalls in den Nachbarblock wechseln.
2. Den Nachbarblock lesen und mit dem gemerkten zusammen entscheiden, wie dieser Block verändert wird. Das Ergebnis schreiben und in den Ausgangsblock wechseln.
3. Die Änderungen in diesem Block ausführen und wieder in den Nachbarblock gehen, falls sich die neue Kopfposition dort befindet.

Wir können also $3\cdot C$ Schritte simulieren und verbrauchen dafür höchstens 3 neue Schritte und haben somit eine Beschleunigung um den Faktor C.

Aufgabe 2

Um die Lesbarkeit der folgenden Programme zu erleichtern, werden folgende Anweisungen verwendet, die alle LOOP-berechenbar sind:

x := c;	(= x:=y+c; y ist eine noch nicht verwendete Variable, d. h. y=0)
x := y;	(= x:=y+0;)
x := y ± z;	(= LOOP x:=y; LOOP z DO x:=x±1; END)
x := y ∗ z;	(= x:=0; LOOP z DO x:=x+y; END)

a) IF $x \geq y$ THEN A ELSE B END

a := 1;
b := 1;
y := y - x; $x \geq y \;\Rightarrow\; y - x = 0 \;\; (A)$, sonst $y - x > 0$ (B)
LOOP y DO r := 0 END;
LOOP y DO a := 0; b := 1 END;
LOOP a DO A END;
LOOP b DO B END;

b) z := x^y

z := 1; LOOP y DO z := z ∗ x END;

c) z := MAX(x,y)

IF $x \geq y$ THEN z := x ELSE z := y END;

d) z := MIN(x,y)

IF $x \geq y$ THEN z := y ELSE z := x END;

e) z := x DIV y

```
z := 0;
LOOP x DO
    IF x ≥ y THEN
        x := x - y;
        z := z + 1
    END;
END;
```

f) z := x MOD y

```
LOOP x DO
    IF x ≥ y THEN x := x - y END;
END; z := x;
```

g) y := FIB(x)

```
x := x - 1;
v := 1;
y := 1;
LOOP x DO
    z := y;
    y := v + y;
    v := z;
END;
```

h) y := FAK(x)

```
y := 1;
LOOP x DO
    y := y * x;
    x := x -1;
END;
```

i) z := BIN(x,y)

```
a := FAK(x);
b := x-y;
c := FAK(b);
d := FAK(y);
e := b+d;
z := a DIV e;
```