

BuK Abgabe 7 | Gruppe 17

Malte Meng (354529) , Charel Ernster (318949), Sebastian Witt (354738)

December 14, 2016

1 Aufgabe 8.1

1. (a) Die TM wird in Phasen geteilt. Man gewinnt dadurch Zeit, dass man 2 Übergänge einfügt pro Speicherzelle und es die Option gibt ohne einen Vergleich von Symbolen von v und w zum nächsten Symbole nach rechts zu laufen. Der Zähler muss ebenfalls klein sein, daher wählen wir eine Binärzahl als Zähler.

Phase 1:

Überprüfe, ob das Eingabewort die Form $v\#w$ hat mit $v, w \in \{0,1\}^*$. Falls nicht, verwerfe. Falls es zutrifft, weiter mit Phase 2.
Laufzeit $O(n)$

Phase 2:

Setze die Position auf das erste Symbol von v . Sei $v = v_1 \dots v_n$ und $w = w_1 \dots w_m$. Setze eine zweite Spur ein, um den Zähler, der zwingend mitgeschoben wird, für die Position anzugeben, an der man gerade liest. Nun arbeite wie folgt:

Es gibt mehrere Optionen für die Übergänge

- Liest man $v_i \in \{0,1\}$, vergleiche v_i mit der i . Position rechts von $\#$. Falls ein B gelesen wird
 \Rightarrow akzeptiere, ansonsten vergleiche das gelesene Symbol $w_i \in \{0,1\}$ mit v_i . Falls $v_i \neq w_i$ dann akzeptiere.
- Liest man $v_i \in \{0,1\}$, darf der Lesekopf einen Schritt nach rechts bewegt werden. Erhöhe den Zähler um 1 und wähle erneut eine Option
- Liest man $\#$, gehe $n+1$ Schritte. Verwerfe, wenn ein B gelesen wird (da dann $v = w$ gilt), ansonsten akzeptiere

Die Optionen sind nicht deterministisch gewählt worden

Laufzeit: $O(n * \log(n))$

Laufzeit insgesamt: $O(n * \log(n))$

2. (b) Nein so funktioniert das hier nicht, denn man muss jedes Symbol von v und w einzeln überprüfen und kann nicht bei Ungleichheit akzeptieren (der Schritt nach rechts darf so nicht ausgeführt werden).

2 8.2

Data: V

Nimm einen Knoten aus jeder Zusammenhangskomponente und speichere ihn in *Colored*

```
while Colored != V do
    for v in Colored do
        if (Ein Nachbarknoten besitzt die gleiche Farbe) then
            reject;
        end
        Färbte alle Nachbarknoten mit der anderen Farbe (nicht  $c(v)$ );
        Colored := Colored Union Nachbarknoten( $v$ );
    end
end
accept;
```

Graphzusammenhangproblem lösbar in $O(n^2)$ für $n = |V|^2$. Die While-Schleife läuft maximal $|V|$ mal und die For-Schleife ebenfalls maximal $|V|$ mal. Dabei werden maximal $|E|$ Kanten betrachtet. Insgesamt ergibt sich $O(|V|^2 \cdot |E|)$. Insgesamt ist der Algorithmus also durch $O(n^2 \cdot |E|)$ beschränkt.

Korrektheit:

Falls der Graph in 2 - Colorability liegt

- Der Algorithmus färbt in jeder Zusammenhangskomponente einen Knoten
- Iterativ werden deren Nachbarknoten immer weiter entgegengesetzt gefärbt
- Der Algorithmus akzeptiert den Graphen

Falls der Graph nicht in 2 - Colorability liegt

- Der Algorithmus färbt in jeder Zusammenhangskomponente einen Knoten
- Iterativ werden deren Nachbarknoten immer weiter entgegengesetzt gefärbt
- Es passiert, dass ein Knoten die gleiche Farbe wie sein Nachbar besitzt.
- Der Algorithmus verwirft den Graphen