Berechnbarkeit und Komplexität
Heimübung 1

Eli Kogan-Wang (7251030)
David Noah Stamm (7249709)
Bogdan Rerich (7248483)
Jan Schreiber()

20. Oktober 2022

Aufgabe 1

Der Algorithmus BIPARTITE-CHECK-BREADTH-FIRST-SEARCH nimmt einen Graphen G und einen Startknoten $s \in V$.

Algorithm 1 BIPARTITE-CHECK-BREADTH-FIRST-SEARCH(G, s)

```
for jeden Knoten u in V\{s} do
    color[u] ← WHITE
    π[u] ← NIL
color[s] ← GRAY
π[s] ← NIL
Q ← {}
Enqueue(Q, s)
while Q ≠ ∅ do
    u ← Dequeue(Q)
    if color[u] = GRAY then
        otherColor ← BLACK
    else
        otherColor ← GRAY
    for jeden Knoten v in A(u) do                ▷ A(u): Nachbarn von u
        if color[v] = WHITE then
            color[v] ← otherColor
            π[v] ← u
            Enqueue(Q, v)
        else if color[v] ≠ otherColor then
            return FALSE
Return TRUE
```

Wir reduzieren das Problem der Bipartitheitsprüfung auf den der 2-Färbbarkeit.

Mithilfe der Breitensuche könenn wir die 2-Färbung auf G versuchen und bei einem Konflikt abbrechen.
Ein Graph ist genau dann 2-Färbbar, wenn er bipartit ist.
Unser Algorithmus ist damit korrekt, weil true rückgibt, genau dann wenn der Graph 2-Färbbar ist. Und weil er false rückgibt, wenn der Graph nicht 2-Färbbar ist.
Die Laufzeit einer Breitensuche ist aus DuA mit $O(|V| + |E|)$ bekannt. Da wir hierbei nur um Elementare Operationen erweitert, weswegen die Laufzeit für diesen Algorthimus erhalten bleibt.

Aufgabe 2

$$
\begin{aligned}
010:& \\
&q_0 \triangleright 010 \\
&\triangleright q_0 010 \\
&\triangleright \sqcup q_1 10 \\
&\triangleright \sqcup 1 q_1 0 \\
&\triangleright \sqcup 10 q_1 \sqcup \\
&\triangleright \sqcup 1 q_3 0 \sqcup \\
&\triangleright \sqcup q_5 1 \sqcup \sqcup \\
&\triangleright q_5 \sqcup 1 \sqcup \sqcup \\
&\triangleright \sqcup q_0 1 \sqcup \sqcup \\
&\triangleright \sqcup \sqcup q_2 \sqcup \sqcup \\
&\triangleright \sqcup q_4 \sqcup \sqcup \sqcup \\
&\triangleright q_6 \sqcup \sqcup \sqcup \sqcup \\
q_6 = q_{\text{accept}}&
\end{aligned}
$$

1011 :

$q_0 \triangleright 1011$

$\triangleright q_0 1011$

$\triangleright \sqcup q_2 011$

$\triangleright \sqcup 0 q_2 11$

$\triangleright \sqcup 01 q_2 1$

$\triangleright \sqcup 011 q_2 \sqcup$

$\triangleright \sqcup 01 q_4 1 \sqcup$

$\triangleright \sqcup 0 q_5 1 \sqcup \sqcup$

$\triangleright \sqcup q_5 01 \sqcup \sqcup$

$\triangleright q_5 \sqcup 01 \sqcup \sqcup$

$\triangleright \sqcup q_0 01 \sqcup \sqcup$

$\triangleright \sqcup \sqcup q_1 1 \sqcup \sqcup$

$\triangleright \sqcup \sqcup 1 q_1 \sqcup \sqcup$

$\triangleright \sqcup \sqcup q_3 1 \sqcup \sqcup$

$\triangleright \sqcup q_7 \sqcup 1 \sqcup \sqcup$

$q_7 = q_{\text{reject}}$

0110 :

$q_0 \triangleright 0110$

$\triangleright q_0 0110$

$\triangleright \sqcup q_1 110$

$\triangleright \sqcup 1 q_1 10$

$\triangleright \sqcup 11 q_1 0$

$\triangleright \sqcup 110 q_1 \sqcup$

$\triangleright \sqcup 11 q_3 0 \sqcup$

$\triangleright \sqcup 1 q_5 1 \sqcup \sqcup$

$\triangleright \sqcup q_5 11 \sqcup \sqcup$

$\triangleright q_5 \sqcup 11 \sqcup \sqcup$

$\triangleright \sqcup q_0 11 \sqcup \sqcup$

$\triangleright \sqcup \sqcup q_2 1 \sqcup \sqcup$

$\triangleright \sqcup \sqcup 1 q_2 \sqcup \sqcup$

$\triangleright \sqcup \sqcup q_4 1 \sqcup \sqcup$

$\triangleright \sqcup q_5 \sqcup \sqcup \sqcup \sqcup$

$\triangleright \sqcup \sqcup q_0 \sqcup \sqcup \sqcup$

$\triangleright \sqcup \sqcup \sqcup q_6 \sqcup \sqcup$

$q_6 = q_{\text{accept}}$

Aufgabe 3

informelle Beschreibung:

DTM bei Eingabe z $\in 0, 1, \#$
Die DTM kann in 2 Phasen eingeteilt werden:

Phase 1 (pre prossessing):
Teste, ob genau eine "#in der Eingabe vorkommt. Andernfalls verwerfe dieses. (Der Lesekopf wird wieder auf den Anfang des Wortes gesetzt).

Phase 2:
Falls die Eingabe von der Form w#x mit $w, x \in \{0, 1\}$ ist, teste ob w=x gilt, indem jeweils Buchstabenweise w mit x verglichen wird. Ersetzte zu Vergleichende Buchstaben mit X. Falls nun Wort vollständig makiert ist, bis auf #, so gilt w=x und die Eingabe ist in L

Formal:
Σ = {0, 1,#}, Γ = {0, 1,# ▷, ⊔, X}
Q = {q0, . . . , q10}, q0 = s, q9 = qaccept, q10 = qreject
δ definiert durch:

δ	0	1	⊔	▷	#	X
q0	(q0,0,R)	(q0,1,R)	(q10,⊔,R)	(q0,▷,R)	(q1,#,R)	(q10,X,R)
q1	(q1,0,R)	(q1,1,R)	(q2,⊔,L)	(q10,▷,R)	(q10,#,R)	(q10,X,R)
q2	(q2,0,L)	(q2,1,L)	(q10,⊔,L)	(q3,▷,R)	(q2,#,L)	(q2,X,L)
q3	(q4,X,R)	(q6,X,R)	(q9,⊔,R)	(q10,▷,R)	(q3,#,R)	(q3,X,R)
q4	(q4,0,R)	(q4,1,R)	(q10,⊔,R)	(q10,▷,R)	(q5,#,R)	(q4,X,R)
q5	(q2,X,R)	(q10,1,R)	(q8,⊔,R)	(q10,▷,R)	(q5,#,R)	(q5,X,R)
q6	(q6,0,R)	(q6,1,R)	(q10,⊔,R)	(q10,▷,R)	(q7,#,R)	(q6,X,R)
q7	(q10,0,R)	(q2,X,R)	(q8,⊔,R)	(q10,▷,R)	(q7,#,R)	(q7,X,R)
q8	(q10,0,R)	(q10,1,R)	(q10,⊔,R)	(q9,▷,R)	(q8,#,L)	(q8,X,R)

Aufgabe 4

Annahme: δ ist zu jedem Zeitpunkt wohldefiniert. (d.h. das Beispielsweise ein Übergang, welcher links aus der Turingmaschine herausführen würde ist nicht in δ möglich.
Zu zeigen:

$$\forall \text{DTM}_{\text{Abgewandelt}}\ M \exists \text{DTM}_{\text{Klassisch}}\ M' : L(M) = L(M')$$

Gegeben sei $M = (\Sigma, \Gamma, Q, \delta)$ wie in der Aufgabe.
Wir beschreiben den normalen DTM $M' = (\Sigma, \Gamma, Q', \delta')$. Mit:

$$Q' = (Q \backslash \{q_{\text{accept}}, q_{\text{reject}}\}) \times \{L, \emptyset\} \cup \{q_{\text{accept}}, q_{\text{reject}}\}$$

$$\begin{aligned}
\delta'((q, L), s) &= ((q, \emptyset), s, L) \\
\delta'((q, \emptyset), s) &= \text{Sei } (q', s', d) = \delta(q, s) \text{ in} \\
&\quad \begin{cases} (q_{\text{accept}}, s', d) \text{ falls } q' = q_{\text{accept}} \\ (q_{\text{reject}}, s', d) \text{ falls } q' = q_{\text{reject}} \\ ((q', L), s', L) \text{ falls } d = 2L \\ ((q', \emptyset), s', R) \text{ falls } d = R \end{cases}
\end{aligned}$$

Wobei der neue Startzustand $q'_{\text{start}} = (q_{\text{start}}, \emptyset)$ ist.
Unsere Turing Maschine ersetzt einen 2L Schritt durch einen L Schritt und einen im Zustand vermerkten L Schritt, der im nächsten Übergang ausgeführt wird.

Beweis:
Sei $w \in \Sigma*$.
Nun seien $\begin{matrix} K_0 = (q_{\text{start}} \triangleright w) \\ K'_0 = (q'_{\text{start}} \triangleright w) \end{matrix}$ und $\begin{matrix} K_{i+1} \text{ Nachfolgekonfiguration von } K_i = (...q...) \\ K'_{i+1} \text{ Nachfolgekonfiguration von } K'_i(...q'...) \end{matrix}$
wenn $\begin{matrix} q_{\text{accept}} \neq q \neq q_{\text{reject}} \\ q_{\text{accept}} \neq q' \neq q_{\text{reject}} \end{matrix}$ und $\begin{matrix} K_{i+1} = K_i \\ K'_{i+1} = K'_i \end{matrix}$ sonst.
Nun ist $g((\alpha(q', L)\beta)) = 0$, $g(K) = 1$ sonst. Und $K'' = (K'_i | g(K_i) = 1 \forall i)$. Nun ist $\Psi((\alpha q \beta)) = \begin{cases} (\alpha q \beta) & \text{wenn } q = q_{\text{accept}} \\ (\alpha q \beta) & \text{wenn } q = q_{\text{reject}} \\ (\alpha(q, \emptyset)\beta) & \text{sonst} \end{cases}$
Jetzt ist es vollkommen trivial, dass $\Psi(K_i) = K''_i$ für alle i.
Damit ist $L(M) = L(M')$. $\square$

Wir haben im Beweis gezeigt, dass für alle Wörter, die Konfigurationsfolgen der Turingmaschinen, insofern man die hinzugefügten Zustände ignoriert, gleich sind.

Insbesondere sind dann die Zustände q_{accept} und q_{reject} gleich auftretend bei M und M'.