Bernhard Dietrich 6256800
Lars Fernhomberg 6256030
Sebastian Kniesburges 6257120
Thomas Brinkmann 6254798
Yascha Cebeci 6254677

Aufgabe 29

Sei $L = \left\{ \langle M \rangle \mid \exists z \in \{0,1\}^* : M \text{ gestartet mit } z \text{ hält und } M \text{ gestartet mit } z0 \text{ hält nicht} \right\}$

Zeigen Sie durch Reduktion, dass L nicht entscheidbar ist.

Ansatz: $H \leq L$

$x \in H \Rightarrow f(x) \in L$

$x \notin H \Rightarrow f(x) \notin L$

$f := \langle M \rangle x \rightarrow f(\langle M' \rangle x)$

- $\langle M \rangle x \in H \Rightarrow M \Rightarrow M$ gestartet mit x hält.
- $\langle M' \rangle x$: vergleicht die Eingabe z mit x, stellenweise von links nach rechts und löscht dabei Ziffernweise die Eingabe z.
  Falls sich z an irgendeiner Stelle bis zur letzten Stelle von x unterscheidet, dann verwirft $\langle M' \rangle x$ die Eingabe. Falls die ersten Stellen von z komplett mit x übereinstimmen, dann wird geschaut welche Ziffern auf dem Band übrig geblieben sind.
  Sollte es nur die 0 sein, so entspricht $z = x0$ und die DTM $\langle M' \rangle x$ geht in eine Endlosschleife.
  Sind andere Ziffern übrig, so verwirft M' die Eingabe.
  Stehen keine Ziffern mehr auf dem Band, so ist $x = z$.
  Jetzt schreibt $\langle M' \rangle x$ x aufs Band und simuliert M mit der Eingabe x.

⇨ H ist reduzierbar auf L mit der Funktion f.

⇨ L ist nicht entscheidbar.

Aufgabe 30

Eine Mehrband-NTM ist das nichtdeterministische Analogon zu einer Mehrband-DTM. Eine Primzahl p in unärer Darstellung hat die Form $p_{un} = 1^p$. Beispiel: $p = 5, p_{un} = 11\,111$.

1. Beschreiben Sie informal eine 2-Band NTM, die die Sprache
   $L_1 := \left\{ 1^p \mid p \text{ ist keine Primzahl} \right\}$ in linearer Zeit akzeptiert.

   Eine 2-Band NTM hat auf dem ersten Band die Eingabe $x \in \Sigma^*$.
   - Die NTM überprüft ob die Eingabe x ausschließlich aus Einsen der Form 1^n besteht.
   - Anschließend rät sie auf dem 2. Band einen potentiellen Teiler der Form $1^i \mid i < n, i \in \mathbb{N}$.
     Sie fängt bei einer 11 an und entscheidet dann ob sie die nächste 1 dazuschreibt solange $i < n$ oder aufhört und den Teiler somit festlegt.
   - Ein Algorithmus löscht nun immer i Einsen auf einmal von der Eingabe, solange bis alle Einsen gelöscht sind, oder nicht 11 bis i-1 Einsen noch in der Eingabe stehen.
     Sollte keine Eins mehr auf dem Band stehen so ist 1^i ein Teiler von 1^n.
     Sollten noch 1 bis $i-1$ Einsen übrig bleiben so ist 1^i kein Teiler von 1^n.
   - Laufzeit:
     Der erste Schritt des Überprüfens benötigt $O(n)$; die NTM rät in maximal linearer Zeit $O(n)$.
     Der Algorithmus löscht i Einsen von der n langen Eingabe. Somit muss er sich einmal durch die ganze Eingabe durcharbeiten, dies geschieht ebenfalls in linearer Zeit $O(n)$.

   ⇒ Also ist die Gesamt-Laufzeit: $O(n)$

2. Zeigen Sie, dass L_1 in P liegt.

   Da eine DMT nicht wie eine NTM direkt einen Teiler „raten“ kann musst sie alle möglichen potentiellen Teiler durchprobieren um L_1 zu akzeptieren.

   $\Rightarrow$ Bei einer Zahl n mit 1^n als Eingabe gibt es somit 1^2 bis 1^{n-1}, da 1 alles ganzzahlig teilen würde.
   Somit muss die DTM im Worst-Case $n-2$ mal testen, bevor sie akzeptieren kann. Daher muss sie $n-2$ mal $O(n)$ Zeit aufwenden: $(n-2)\cdot n = n^2 - 2n = O(n^2) \Rightarrow L$ liegt in P.

3. Zeigen Sie, dass die Sprache $L_2 := \{bin(p) \mid p \text{ ist keine Primzahl}\}$ in NP liegt.

   Ein Polynomialzeit-Verifizierer A für L_2 kann wie folgt arbeiten. Zunächst wird ein Teiler t geraten. Dann wird verifiziert, ob $t \mid x$, in diesem Fall wird akzeptiert, ansonsten verworfen.
   Da x in Binärdarstellung wird polynomielle Zeit benötigt.

Aufgabe 31

Zwei ungerichtete Graphen $G = (V_1, E_1)$ und $H = (V_2, E_2)$ heißen *isomorph*, wenn es eine bijektive Abbildung $\phi : V_1 \to V_2$ gibt, so dass für alle $v, u \in V_1$ gilt:

$$\{v,u\} \in E_1 \Leftrightarrow \{\phi(u), \phi(v)\} \in E_2$$

Zeigen Sie, dass die Sprache ISO in NP liegt.

$$ISO := \{(G,H) \mid G, H \text{ sind isomorphe Graphen}\}$$

Die Sprache ISO liegt in NP, da es eine NTM M gibt, welche ISO in polynomieller Zeit entscheidet. Dabei geht M folgendermaßen vor: M rät eine Reihenfolge der Knoten aus E_2. Durch eine vorher festgelegte Reihenfolge der Knoten aus E_1 entsteht somit eine Abbildung ϕ der Knoten aus E_1 zu den Knoten aus E_2.
Anschließend wird diese Abbildung als Isomorphismus verifiziert. $\{v,u\} \in E_1 \Leftrightarrow \{\phi(u), \phi(v)\} \in E_2$, d.h. es wird geprüft, ob die Kante in G ihre Entsprechung in H besitzt.
Jeder Knoten kann maximal $|V|-1$ Kanten besitzen. Da jeder Knoten überprüft wird ergibt sich eine Laufzeit von $|V| \cdot (|V|-1) = O(|V|^2)$. Also ist die Laufzeit polynomiell, somit ISO in NP.

Bernhard Dietrich 6256800
Lars Fernhomberg 6256030
Sebastian Kniesburges 6257120
Thomas Brinkmann 6254798
Yascha Cebeci 6254677

Aufgabe 32

Geben Sie jeweils einen Verifizierer für die folgenden zwei Probleme an. Begründen Sie die Korrektheit und geben Sie die Laufzeit Ihrer Algorithmen an.

- **Degree Constrained Spanning Tree (DCST):** Sei ein ungerichteter Graph $G = (V, E)$ und eine ganze Zahl $k \leq |V|$ gegeben. Frage: Gibt es einen Spannbaum für G mit maximalem Grad k (d.h. jeder Knoten hat maximal k Nachbarn)?

  Zur Lösung des Problems hat eine NTM einen Teilgraphen geraten, der ein potentieller Spannbaum ist. Zunächst wird verifiziert ob ein Spannbaum vorliegt. Dies geschieht mittels Tiefensuche ($O(|V| + |E|)$).

  Anschließend überprüfen wir die Knotengrade auf Beschränktheit mit k ($O(|V|)$).
  Also Verifikation mit Laufzeit der Tiefensuche.

- **Bottleneck Traveling Salesman:** Sei C eine Menge von n Städten mit euklidischen Entfernungen $d(c_i, c_j) \in \mathbb{R}, \forall c_i, c_j \in C$ und $k \in \mathbb{R}$. Gibt es eine Tour durch alle Städte, deren Entfernungen von Stadt zu Stadt nicht länger ist als k, d.h. gibt es eine Permutation $(c_{\pi(1)}, \ldots, c_{\pi(n)})$ von C, so dass $d(c_{\pi(i)}, c_{\pi(i+1)}) \leq k$ und $d(c_{\pi(n)}, c_{\pi(1)}) \leq k$.

  Zur Lösung des Problems hat eine NTM eine Permutation $(c_{\pi(1)}, \ldots, c_{\pi(n)})$ von C geraten, die eine potentielle Tour ist. Zunächst wird verifiziert ob die Permutation ein Euler-Kreis ist, d.h. dass jede der n Städte besucht wird.
  Dazu wird die Existenz jeder Kante $\{c_{\pi(i)}, c_{\pi(i+1)}\}$ und die Kante $\{c_{\pi(n)}, c_{\pi(1)}\}$ überprüft ($O(|E|)$).
  Anschließend überprüfen wir ob die jeweilige euklidische Entfernung zwischen den Städten k nicht überschreitet. Hierbei wird die euklidische Entfernung jeder einzelnen Kante überprüft, also auch hier $O(|E|)$. Somit arbeitet der Verifizierer in $O(|E|)$.