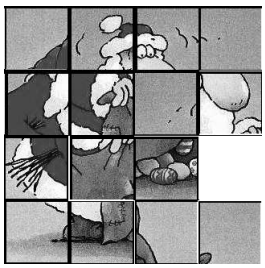
Prof. Dr. R. Schrader
Birgit Engels [engels@zpr.uni-koeln.de]

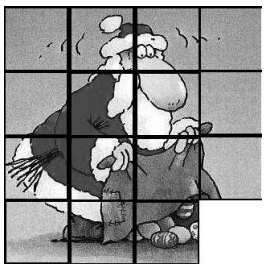
WS 2006/2007

Theoretische Informatik

7. Übung, Abgabe Mittwoch, 06.12.2006

Aktuelle Informationen bezüglich der Vorlesung und der Übungen finden sich unter :
http://www.zaik.uni-koeln.de/AFS/teachings/courses/ThInf/index.html und
http://www.zaik.uni-koeln.de/AFS/teachings/courses/ThInf/uebungen.html





Aufgabe 27 (Speicherplatzkomplexität — Schiebepuzzle — 6 Punkte):
Betrachten Sie folgende Art von Puzzle:

> **(n,n)-Schiebepuzzle:** Auf einem quadratischen Feld der Größe $n \times n$ seien $n^2 - 1$ (ebenfalls quadratische, der Einfachheit halber von 1 bis n^2-1 durchnummerierte) 1×1-Puzzleteile in beliebiger Ordnung gegeben. Aufgabe ist es nun die Puzzleteile durch wiederholtes Verschieben von angrenzenden Puzzleteilen auf das verbleidende leere Feld, die Ordnung der Puzzleteile auf dem Feld (mit zeilenweise monoton wachsender Nummerierung) wieder herzustellen, bzw. festzustellen, dass dies nicht möglich ist.

Argumentieren Sie, warum **(n,n)-Schiebepuzzle** in $PSPACE$ liegt.
Hinweis: Betrachten Sie den *worst case* für eine (n,n)-Schiebepuzzle-Lösung — Wie viele "Zustände" (also verschiedene Anordnungen der Puzzleteile auf dem Puzzlefeld) kann das Puzzle zwischen Ausgangszustand und Lösung maximal annehmen? Müssen diese alle gleichzeitig erzeugt werden? Wie kann ein Lösungsweg (eine Abfolge von Zuständen bzw. Konfigurationen) einer Schiebe-Puzzle-Instanz (Gegebenes Schiebepuzzle in Ausgangsposition) von einer NTM mit polynomiellem *Speicherplatz*-Aufwand gefunden werden? Wie ist dies auch mittels einer DTM realisierbar? (Wenden Sie die Idee aus dem $NPSPACE = PSPACE$-Beweis aus der Vorlesung an.)

Aufgabe 28 (Interaktive Beweissysteme — "Zero-Knowledge"-Beweis — 8 Punkte):
In der Vorlesung wurde das coGraphenisomorphie-Problem vorgestellt, wobei für 2 gegebene Graphen zu entscheiden ist, ob diese zueinander *nicht* isomorph sind. Ganz analog definieren wir:

> **Graphenisomorphie (GI):** Gegeben 2 Graphen $G_1 = (V, E_1), G_2 = (V, E_2)$, $V = \{1, \ldots, n\}$ entscheide, ob die beiden Graphen zueinander isomorph sind. (D.h. $\exists$ Permutation der Knoten $\pi : V \to V$ mit $\pi(G_1) = G_2$.)

Bitte Wenden!

Betrachten Sie folgendes Protokoll P, welches n-mal ausgeführt werden soll:

1 N wählt eine zufällige Permutation ρ der Knoten $V = \{1, \ldots, n\}$ und schickt den entstehenden Graphen $H = \rho(G_1)$ an V.

2 V wählt zufällig $i \in \{1, 2\}$ und schickt i an N.

3 N schickt eine Permutation μ von $V = \{1, \ldots, n\}$ an V. (N kann hier versuchen zu "betrügen" und irgendeine Permutation schicken.)

4 V überprüft $\mu(G_i) = H$.

V akzeptiert nach n Runden als bewiesen, dass beide Graphen G_1 und G_2 isomorph sind, falls in jeder Runde $\mu(G_i) = H$ gilt !

a) Wie hoch ist die Wahrscheinlichkeit, dass V den Isomorphie-Nachweis akzeptiert, falls G_1 und G_2 isomorph sind ? Wie hoch ist die Wahrscheinlichkeit, dass V den Isomorphie-Nachweis verwirft, falls G_1 und G_2 nicht isomorph sind ?

b) Nach a) ist P ein *Interaktiver Nachweis* von **GI**. Betrachten wir Punkt 3 des Protokolls genauer: N muss hier eine Permutation μ mit $\mu(G_i) = H$ bestimmen. Wie schnell ist dies (nach momentaner Kenntnis) möglich, falls

- N $\pi : V \to V$ mit $\pi(G_1) = G_2$ kennt?
- N π nicht kennt?

Wie können wir (bzw. V) also N^*, der π kennen soll von einem Betrüger N unterscheiden, der zwar G_1 und G_2, nicht aber π kennt ?

c) In b) haben wir vermutet, dass V unterscheiden kann, ob N π kennt oder nicht, ohne selbst π zu kennen. — Kann V durch das Protokoll (falls sich also N und V genau an P halten) π herausfinden oder sonstige Hinweise erhalten ? Begründen Sie Ihre Behauptung.
Hinweis: Zeigen Sie, dass sich die Informationen, die V von N erhält aus dem Blickwinkel von V nicht von zufälligen Informationen der gleichen Art unterscheiden!

Ein Protokoll bzw. ein Interaktiver Nachweis für die die Behauptung aus c) gilt, nennt man "Zero-Knowledge"-Beweis.

Aufgabe 29 (IP — Abgeschlossenheit unter polynomieller Reduktion — 4 Punkte):
Zeigen Sie, dass die Komplexitätsklasse IP gegenüber polynomieller Reduktion abgeschlossen ist:

$$L' \leq_{pol} L, L \in IP \Rightarrow L' \in IP.$$

Hinweis:

$$L' \leq_{pol} L \Leftrightarrow \exists f_{pol} : x \in L' \Leftrightarrow f(x) \in L.$$