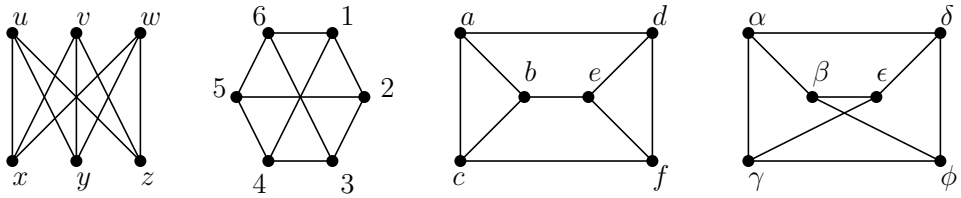
Universität zu Köln
Mathematisches Institut
Prof. Dr. F. Vallentin
Dr. A. Gundert

Einführung in die Theoretische Informatik

Wintersemester 2016/17

— Aufgabenblatt 11 —

Aufgabe 11.1 Gegeben sind die folgenden vier Graphen:



G_1 G_2 G_3 G_4

Geben Sie an (und begründen Sie), welche davon isomorph sind und welche nicht.

Aufgabe 11.2 Die Klasse IP^* sei analog zur Klasse IP definiert, mit der Ausnahme, dass der Verifizierer eine polynomiell zeit-beschränkte *deterministische* Turing-Maschine ist. Zeigen Sie:

$$\mathrm{IP}^* = \mathrm{NP}.$$

Aufgabe 11.3 Definiere $\mathrm{PSPACE} = \bigcup_{k\in\mathbb{N}} \mathrm{DSPACE}(n^k)$, wobei für $T : \mathbb{N} \to \mathbb{N}$ eine Sprache $L \subseteq \Sigma^*$ in der Klasse $\mathrm{DSPACE}(T(n))$ ist, falls es eine Turing-Maschine gibt, die für Eingaben w der Länge n einen Platzbedarf von $O(T(n))$ hat und w genau dann akzeptiert, wenn $w \in L$.
Zeigen Sie:

$$\mathrm{IP} \subseteq \mathrm{PSPACE}.$$

Aufgabe 11.4 (10 Punkte) Für einen Graphen $G = (V, E)$ heißt eine bijektive Abbildung $\sigma : V \to V$ *Automorphismus*, falls $\{u, v\} \in E$ genau dann, wenn $\{\sigma(u), \sigma(v)\} \in E$. Die Identität ist immer ein trivialer Automorphismus. Zeigen Sie, dass für die Probleme

$$\mathrm{GA} = \{\langle G\rangle : G \text{ Graph, es existiert ein nicht-trivialer Automorphismus von } G\}$$

und

$$\text{1-GI} = \{\langle G, H\rangle : G, H \text{ Graphen, es existiert genau ein Isomorphismus zwischen } G \text{ und } H\}$$

gilt:

$$\mathrm{GA} \in \mathrm{P} \iff \text{1-GI} \in \mathrm{P}.$$

Abgabe: Bis Mittwoch, 25. Januar 2017 um 12 Uhr im Schließfach im Studierendenarbeitsraum im MI (Raum 3.01). Bitte Namen und Matrikelnummer auf die Abgabe schreiben.

Intuitively, what should we require from an efficient theorem-proving procedure?

(1) That it should be possible to "prove" a true theorem.

(2) That it should not be possible to "prove" a false theorem.

(3) That communicating the "proof" should be efficient. Namely, regardless of how much time it takes to come up with the proof, its correctness should be efficiently verified.

The NP formalization of the concept of an efficient proof system captures one way of communicating a proof. [...] we will generalize the NP proof system to capture a more general way of communicating a proof. The verifier will be a probabilistic polynomial time (in the length of the common input) machine that is able to exchange messages (strings) with the prover. At the same time that we introduce probability into the proof system, we relax the classical notion of a "proof." Our verifier may erroneously be convinced of the truth of a proposition with a very small probability of error (less than n^{-k} for each positive constant k and all sufficiently large input-sizes n).

Shafi Goldwasser, Silvio Micali, Charles Rackoff, *The knowledge complexity of interactive proof systems*, SIAM J. Comput. **18** (1989), 186–208.