Eidgenössische Technische Hochschule Zürich — Ecole polytechnique fédérale de Zurich — Politecnico federale di Zurigo — Swiss Federal Institute of Technology Zurich

Institut für Theoretische Informatik *4. Dezember 2007*
Dr. B. Gärtner, Prof. Dr. J. Hromkovič

Informatik für Mathematiker und Physiker Lösung 10 HS 07

URL: `http://www.ti.inf.ethz.ch/ew/courses/Info1_07/`

Aufgabe 1 (Schnellübung)

a) Die Antwort auf alle drei Fragen lautet Ja.

(i) Das ist möglich. In C++ würde eine entsprechende Funktion (also im weiteren Sinne ein Programm) etwa wie folgt aussehen.

```
boolean alwaysTrue(){
  return true;
}
```

(ii) Das ist möglich. Das ist ziemlich genau die Aufgabe, die das Betriebssystem respektive das Programm Cygwin für uns erledigt. Wenn Sie ein Programm schreiben, führen Sie es anschliessend mit einer bestimmten Eingabe aus, und das Betriebssystem kann ihnen die Ausgabe Ihres Programmes anzeigen. Falls Ihr Programm unendlich lange läuft, so wird auch das Betriebssystem unendlich lange mit der "Simulation" beschäftigt sein, was allerdings erlaubt ist, da wir hier von Programmen sprechen und nicht von Algorithmen.

(iii) Das ist möglich. In der Vorlesung wurde besprochen, dass es möglich ist, die Menge aller Programme zu ordnen. Somit macht es durchaus Sinn vom i-ten Programm P_i zu sprechen. Das heisst, wenn wir die Eingabe i betrachten, können wir zuerst P_i generieren und dann einfach Zeichen für Zeichen mit dem anderen Teil der Eingabe, nämlich x, vergleichen. Sind die beiden gleich, geben wir JA zurück, andernfalls geben wir NEIN zurück.

b) Es gibt eine Reihe von algorithmisch unlösbaren Problemen. Einige davon wurden in der Vorlesung besprochen, andere sind aus dem Buch "Sieben Wunder der Informatik" entnommen.

- Es gibt keine Programm (und somit auch keinen Algorithmus) der die Mengen $M(\mathrm{DIAG})$ oder $M(2-\mathrm{DIAG})$ (vergleiche Übung 7) erkennt. Dies folgt aus einer Variante des Diagonalbeweises.
- Im Allgemeinen gibt keinen Algorithmus UNIV, der für jedes Programm P_i entscheidet, ob P_i die Eingabe $n \in \mathbb{N}$ akzeptiert oder nicht. Erinnern Sie sich daran, dass ein Programm laut Definition eine Eingabe n *nicht* akzeptiert, falls es entweder NEIN zurckgibt, oder unendlich lange läuft. Gegeben UNIV wäre es möglich das Problem $(\mathbb{N}, M(\mathrm{DIAG}))$ zu entscheiden. Deswegen kann es UNIV nicht geben.
- Es gibt keinen Algorithmus HALT, der für jedes Programm P_i entscheidet, ob P_i auf der Einhabe $n \in \mathbb{N}$ hält. Gegeben HALT, könnten wir uns einen Algorithmus UNIV bauen. Deswegen kann es HALT nicht geben.

- Betrachten Sie das Problem $\mathrm{Problem}(c)$ wie es auf Seiten 129ff von "Sieben Wunder der Informatik" beschrieben ist. Es geht dabei darum, als Antwort auf die Eingabe $n \in \mathbb{N}$, die Zahl c bis zur n-ten Stelle nach dem Komma anzugeben. Aus der Tatsache, dass $|\mathbb{R}| > |\mathbb{N}|$ folgt, dass es reelle Zahlen c geben muss, für die $\mathrm{Problem}(c)$ nicht algorithmisch lösbar ist. Dies ist zwar kein Entscheidungsproblem ($\mathrm{JA}/\mathrm{NEIN}$ Antwort), aber da das in der Aufgabenstellung nicht genau spezifiziert wurde, gilt diese Antwort natürlich auch. Man beachte jedoch, dass es eben nicht möglich ist, dieses Problem *exakt* anzugeben, weil es nicht möglich ist, eine Zahl c anzugeben, deren Repräsentation nicht endlich ist. Es ist gerade falsch, wenn man zum Beispiel $c = \pi$ einsetzt, weil π, wie Sie aus Serie 5 Aufgabe 51 wissen, eben endliche Darstellungen hat, die es uns erlauben, π beliebig genau zu berechnen.