Lösungsstrategien für NP-schwere Probleme der Kombinatorischen Optimierung

— Übungsblatt 2 —

Walter Stieben (4stieben@inf) Tim Reipschläger (4reipsch@inf) Louis Kobras (4kobras@inf) Hauke Stieler (4stieler@inf)

Abgabe am: 25. April 2016

Aufgabe 2.1

Aufgabe 2.1.1

Aufgabe 2.1.2 Spezialfall $k = 2$

Beim Spezialfall für $k = 2$ kann man tatsächlich einen Algorithmus angeben, dessen Laufzeit polynomiell in der Eingabe ist.

Algorithm 1 reserve

```
1:  procedure RESERVE(P, R, k)
2:      for all p∈P do
3:          R_rest = weiseResourcenZu(p, R) // gibt alle noch nicht zugewiesenen Ressourcen zurück
4:          P = P − p
5:          for all q∈P do
6:              b = istVerteilungMöglich(q, R_rest)
7:              if b == true then return true
8:              end if
9:          end for
10:     end forreturn false
11: end procedure
```

Die Laufzeit beträgt dabei $\mathcal{O}(|P| \cdot n + |P|^2 \cdot n)$, wobei $n = |R|$ ist.

Man Geht in der äußeren Schleife alle $p \in P$ durch , also $|P|$ mal. Dort verteilt man zunächst maximal n viele Ressourcen, ergo n viele Schritte. Nun wird p aus P entfernt, was mit einer schlauen Implementierung (z.B. als Liste) in konstanter Zeit machbar ist.
Danach beginnt die innere Schleife, welche alle $q \in P$ durch geht und somit $|P| - 1$ mal durchläuft. Dort wird dann geprüft ob eine weitere Verteilung der restlichen Ressourcen auf q möglich ist. Auch da benötigt man maximal n viele Schritte.
Man erhält also eine Laufzeit von $\mathcal{O}(|P| \cdot (n + |P| \cdot n))$, was mit $\mathcal{O}(|P| \cdot n + |P|^2 \cdot n)$ äquivalent ist (hier erkennt man aber schön das Polynom).

Aufgabe 2.1.3

Aufgabe 2.1.4

Aufgabe 2.2

Aufgabe 2.2.1

Aufgabe 2.2.2