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

```
procedure RESERVE(P, R, k)
    for all p∈P do
        R_rest = weiseResourcenZu(p, R) // gibt alle noch nicht zugewiesenen Ressourcen zurück
        P = P − p
        for all q∈P do
            b = istVerteilungMöglich(q, R_rest)
            if b == true then return true
            end if
        end for
    end forreturn false
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 $3-$ SAT **auf** SET SPLITTING **reduzieren**

Aufgabe 2.2.1.a Zeigen, dass SET SPLITTING$\in$ *NP* gilt

Mit einem Verfifikationsalgorithmus, welcher die Eingabe $\langle S_1, S_2, C \rangle$ bekommt, kann man in polynomieller Laufzeit prüfen, ob eine Aufteilung in Klassen (also ein set splitting), korrekt ist.

Dazu geht man zunächst jedes $c \in C$ durch und nimmt ein Element e_1 aus c. Für dieses Element prüft man nun ob es ein $e_2 \in c$ gibt, bei dem gilt $class(e_1) \neq class(e_2)$ (also ob e_1 in einer anderen Klasse ist, als e_2). Gibt es ein solches e_2, ist die Aufteilung der Teilmenge c schon mal gültig und verifiziert. Zu prüfen sind noch die restlichen Teilmengen.

Die Laufzeit ist polynomiell, da man $|C|$ viele Teilmengen durch geht und pro Teilmenge maximal $|S|$ Elemente. Es gibt maximal $|S|$ viele Teilmengen (jede Teilmenge mit je einem Element), damit wäre die Laufzeit in $\mathcal{O}(|S|^2)$.

Aufgabe 2.2.1.b Reduktion angeben

Aufgabe 2.2.2