Algorithmenentwurf HA 5

Lukas Brandt: 7011823, Clemens Damke: 7011488, Lukas Giesel: 7011495

19. Mai 2016

Aufgabe 9

a. Algorithmus

Algorithm 1 Maximales bipartites Matching

Require: $(L \cup R, E, w)$ ist bipartiter Graph mit linker Seite L, rechter Seite R und Gewichtsfunktion $w : L \to \mathbb{R}$.

function MaxBipartiteMatching(L, R, E, w)
  $U \leftarrow \{\}$ ▷ $\mathcal{O}(1)$
  $L \leftarrow$ SortDescending$(L$ by $w)$ ▷ $\mathcal{O}(|L| \log |L|)$
  for all $l \in L$ **do** ▷ $\mathcal{O}(|L|)$
    if IsMatching$(U \cup \{l\}, R, E)$ **then** ▷ $\mathcal{O}(|L| \cdot x)$
      $U \leftarrow U \cup \{l\}$ ▷ $\mathcal{O}(|L| \cdot 1)$
  return U

function IsMatching(L, R, E)
  if $|L| = 0$ **then**
    return true
  $l \leftarrow$ FirstElement(L) ▷ $\mathcal{O}(1)$
  for all $r \in R$ **do** ▷ $\mathcal{O}(|R|)$
    if $\{l, r\} \in E \land$ IsMatching$(L \setminus \{l\}, R \setminus \{r\}, E)$ **then** ▷ $\mathcal{O}(|R|!)$
      return true
  return false

Nicht effizient.

1. Aufgabe 10

$$
\begin{aligned}
M &:= (E = J, U) \\
U &:= \{\, B \subseteq E \mid \exists \text{Bijektion}\, \pi : B \to \{1, \ldots, |B|\} : \forall j \in B : \pi(j) \leq d_j \,\} \\
p(B) &:= \sum_{j \in B} p_j,\ \forall B \in U
\end{aligned}
$$

Gesucht ist ein $B \in U$ mit maximalem $p(B)$, da alle Jobs in B rechtzeitig geschafft werden und somit keine Strafe verursachen. Das Maximieren der nicht anfallenden Strafe minimiert somit die anfallende Strafe. Als Gewichtsfunktion kann also $w(j) := p_j, \forall j \in E$ verwendet werden.

z. z. M ist Matroid:

1 $\emptyset \in U$. Per Definition von U wahr.

2 $\forall\, A \subseteq B \in U$:

$$\pi_B := \text{Eine, gemäß Definition aus } U\text{, zu } B \text{ passende Bijektion.}$$

$$\pi_A := \pi_B \setminus \{\, (j, \pi_B(j)) \mid j \in B \setminus A \,\}$$

$$\text{Da } \pi_A \subseteq \pi_B\text{: } \forall j \in A :\ \pi_A(j) = \pi_B(j) \leq d_j$$

$$\implies A \in U$$

3 Austauscheigenschaft: $\forall A, B \in U, |A| < |B|$:

$$\pi_A := \text{Eine, gemäß Definition aus } U\text{, zu } A \text{ passende Bijektion.}$$

$$\pi_B := \text{Eine, gemäß Definition aus } U\text{, zu } B \text{ passende Bijektion.}$$

$$pos_{BA} := \pi_A \circ \pi_B^{-1}$$

$$f(i) := \begin{cases} \pi_B^{-1}(i) : & \pi_B^{-1}(i) \notin A \\ f(pos_{BA}(i)) : & sonst \end{cases}$$

$$x := f(|A| + 1)$$

$$C := A \cup \{x\}$$

$f(|A|+1)$ findet gemäß dem Schubfachprinzip immer ein x aus $B \backslash A$, da die ersten $|A|+1$ Elemente von B probiert werden.

$$g(\pi, i) := \begin{cases} \pi \cup \{(\pi_B^{-1}(i), i)\} : & \pi_B^{-1}(i) \notin A \\ g(\pi \cup \{(\pi_B^{-1}(i),\ i)\} \setminus \{(\pi_B^{-1}(i),\ pos_{BA}(i))\},\ pos_{BA}(i)) : & sonst \end{cases}$$

$$\pi_C := g(\pi_A, |A| + 1)$$

g erweitert π_A so, dass x an der Stelle $\pi_B(x)$ einsortiert wird. Wenn dort in π_A bereits ein anderes Element y liegt, wird y nach $\pi_B(y)$ geschoben. Sollte dort wiederum ein Element liegen wird es nach dem selben Prinzip weitergeschoben.

Gemäß Konstruktion wird x dabei so gewählt, dass $\forall$ geschobenen $y : \pi_B(y) \leq |A| + 1$. Das Schieben findet zwangsläufig ein Ende, sobald ein Element aus A nach $|A| + 1$ geschoben wird, da π_A dort nichts einsortiert haben kann und somit nichts geschoben werden muss.

Da alle Zuordnungen von π_C entweder aus π_A oder π_B stammen, ist π_C eine Sortierung von C, die keine Strafe verursacht.

$$\implies C \in U$$

$\blacksquare$