

Übung 12 – Algorithmen II

Moritz Laupichler, Nikolai Maas – {moritz.laupichler, nikolai.maas}@kit.edu
https://algo2.iti.kit.edu/AlgorithmenII_WS23.php

Institut für Theoretische Informatik - Algorithm Engineering

```
    result = current_weight;
    return true;
}

for( EdgeID eid = graph.edgeBegin( current ); eid != graph.edgeEnd( current ); ++eid ){
    const Edge & edge = graph.getEdge( eid );
    COUNTING( statistic_data.inc( DijkstraStatisticData::TOUCHED_EDGES ); )
    if( edge.forward ){
        COUNTING( statistic_data.inc( DijkstraStatisticData::RELAXED_EDGES ); )
        weight new_weight = edge.weight + current_weight;
        GUARANTEE( new_weight >= current_weight, std::runtime_error, "Weight overflow detected." );
        if( !priority_queue.isReached( edge.target ) ){
            COUNTING( statistic_data.inc( DijkstraStatisticData::SUCCESSFULLY_RELAXED_EDGES ); )
            COUNTING( statistic_data.inc( DijkstraStatisticData::REACHED_NODES ); )
            priority_queue.push( edge.target, new_weight );
        } else {
            if( priority_queue.getCurrentKey( edge.target ) > new_weight ){
                COUNTING( statistic_data.inc( DijkstraStatisticData::SUCCESSFULLY_RELAXED_NODES ); )
                priority_queue.decreaseKey( edge.target, new_weight );
            }
        }
    }
}
```

- Besprechung der Evaluation
- Online-Algorithmen

- Information kommt **nach und nach** an
- Entscheidungen / Berechnungen mit **beschränkter Information**
 - normalerweise **supoptimale Lösungen**

Beispiele

- *Ski Rental* Problem
- Job-Scheduling / Memory-Paging
- *Streaming* Algorithmen
- Suche in Labyrinthen / Routing in Kommunikationsnetzen
- ...

- Ein Online Algorithmus ALG hat einen **kompetitiven Faktor**

$$c = \sup_I \frac{ALG(I)}{OPT(I)}$$

mit OPT optimaler **Offline** Algorithmus, I Probleminstanz
hier für Minimierungsprobleme

Einige Eigenschaften

- $c = \infty$ möglich \rightarrow Online Algorithmus beliebig schlecht
- $c = \text{const.}$
(hoffentlich) nicht von Problemgröße abhängig,
aber evtl. abhängig von weiteren Problemparametern

Generische Problembeschreibung

- kostenpflichtige Nutzung einer Leistung
 - einmalige Kosten K für unbeschränkte Nutzung **oder**
 - Kosten $k < K$ für jede Nutzung
- Anzahl Nutzungen n unbekannt
⇒ **Wann soll man K für unbeschränkte Nutzung zahlen?**

Nomenklatur

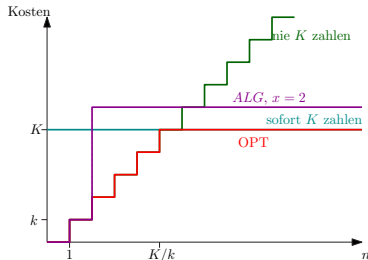
- Instanz I durch Anzahl Nutzungen n bestimmt
- $OPT(I)$, $ALG(I) \hat{=}$ Gesamtkosten

■ allgemein gilt $c = \sup_n \frac{ALG(n)}{OPT(n)}$

■ $ALG(n) = \begin{cases} n \cdot k & n < x \\ (x-1) \cdot k + K & \text{sonst} \end{cases}$ $n < x$
zahle K vor x -ter Nutzung

■ $OPT(n) = \begin{cases} n \cdot k & n < \frac{K}{k} \\ K & \text{sonst} \end{cases}$

$$\Rightarrow c = \begin{cases} \frac{K-k}{xk} + 1 & x \leq \frac{K}{k} \\ \frac{(x-1)k}{K} + 1 & \text{sonst} \end{cases}$$



Sonderfall: Bezahle K , nutze nie!

$$\Rightarrow ALG(n) = K$$

$$\Rightarrow c = \infty$$

entspricht $x = 0$

$$\text{Es gilt } c = \begin{cases} \frac{K-k}{xk} + 1 & x \leq \frac{K}{k} \\ \frac{(x-1)k}{K} + 1 & \text{sonst} \end{cases}$$

Frage: Für welches x ist c minimal?

\Rightarrow für $x = \frac{K}{k}$ ist c minimal mit $c = 2 - \frac{k}{K}$

Zahlenbeispiel: $K = 100, k = 10 \Rightarrow c = 1.9$

- Randomisierung erreicht häufig besseren kompetitiven Faktor
- Annahme hier: Eingabeinstanz fest (*oblivious/weak adversary*)
- Betrachte $C = \frac{ALG(I)}{OPT(I)}$ als **Zufallsvariable über mehrere Durchläufe**

Analyse: Was ist der erwartete kompetitive Faktor $c := \mathbb{E}(C)$?

Strategie: Wähle per Zufall, wann K bezahlt wird (Zufallsvariable X)

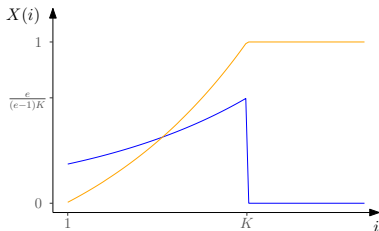
- Zur Vereinfachung $k = 1$

- Optimale Verteilung:

$$X(i) = \left(1 - \frac{1}{K}\right)^{K-i} \cdot \frac{1}{K(1 - (1 - \frac{1}{K})^K)}$$

für $i \leq K$, ansonsten 0

- Für $K \rightarrow \infty$ ergibt sich $e^{i/K} \cdot \frac{1}{(e-1)K}$



Kompetitiver Faktor: $c = e/(e-1) \approx 1.58$

Doubling

- Strategie für Online/Offline Approximationsalgorithmen

Idee

- schätze Lösung konservativ ab
- prüfe, ob Problem gelöst
- falls nicht erfolgreich, vergrößere Schätzung geometrisch
z.B. verdoppeln, verdreifachen, ...

Problembeschreibung

- Unbekannter Zielwert $U > 1$
- Bieter gibt eine Reihe an Geboten (b_i) ab bis $b_k \geq U$
- Bieter muss Summe der Gebote $\sum_{i=0}^k b_i$ bezahlen
⇒ Wie kann der Bieter möglichst geschickt vorgehen?

Nomenklatur

- Lösung durch Reihe (b_i) beschrieben
- $OPT(I)$, $ALG(I) \hat{=}$ summierten Kosten

Optimaler Offline Algorithmus

- kennt den Zielwert U
- folglich bietet er direkt $b_0 = U$
 $\Rightarrow OPT = U$

Kompetitiver Faktor

$$c = \sup_{k,U} \left\{ \frac{b_0 + b_1 + \dots + b_k}{U} : b_{k-1} < U \leq b_k \right\}$$

- Strategien sind z.B. (a) $b_i = i \cdot x$, (b) $b_i = a \cdot x^i$

Analyse: (a) $b_i = i \cdot x$ für $x > 0$

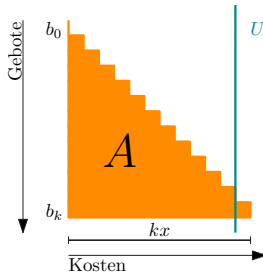
■ **worst case:** $b_{k-1} = (k-1) \cdot x = U - \varepsilon$ für $k > 0, \varepsilon \rightarrow 0$

$$\Rightarrow \text{Anzahl Gebote: } k+1 = \left(\frac{U-\varepsilon}{x} + 1\right) + 1 \leq \frac{U}{x} + 2$$

$$\Rightarrow \text{ALG} = \sum_{i=0}^k i \cdot x = \frac{1}{2}(k+1)k \cdot x$$

$$\Rightarrow \frac{\text{ALG}}{\text{OPT}} \leq \frac{1}{2U} \left(\frac{U}{x} + 2\right) \left(\frac{U}{x} + 1\right) \cdot x = \frac{U^2 + 2x^2}{2Ux} + 3$$

$$\Rightarrow c = \infty$$



$$\text{Fläche } A \approx \frac{1}{2} kx \cdot (k+1)$$

Analyse: (b) $b_i = a \cdot x^i$ für $x > 1$

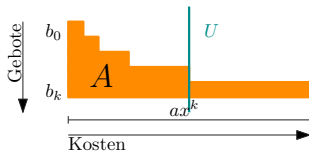
■ **worst case:** $b_{k-1} = a \cdot x^{k-1} = U - \varepsilon$ für $k > 0, \varepsilon \rightarrow 0$

$$\Rightarrow \text{Anzahl Gebote: } k + 1 = \left(\log_x \left(\frac{U - \varepsilon}{a} \right) + 1 \right) + 1 \leq \log_x \frac{U}{a} + 2$$

$$\Rightarrow \text{ALG} = \sum_{i=0}^k a \cdot x^i = \frac{a \cdot (x^{k+1} - 1)}{x - 1} \leq \frac{a \cdot (x^{\log_x \frac{U}{a} + 2} - 1)}{x - 1} = \frac{a \cdot (\frac{U}{a} x^2 - 1)}{x - 1}$$

$$\Rightarrow \frac{\text{ALG}}{\text{OPT}} \leq \frac{a \cdot (\frac{U}{a} x^2 - 1)}{U(x - 1)} = \frac{x^2}{x - 1} - \frac{a}{(x - 1)U}$$

$$\Rightarrow c \leq \frac{x^2}{x - 1} \quad \text{minimal für } x = 2 \Rightarrow c \leq 4$$



$$\text{Fläche } A \approx \frac{a \cdot (x^{k+1} - 1)}{x - 1}$$

Wissenswert nicht erschöpfend!

- kompetitiver Faktor c
- *Doubling* Technik
- *Ski Rental* Problem

Typische Fragestellungen

- Gegeben sei Algorithmus X . Wie groß ist sein kompetitiver Faktor?
- Geben Sie einen Algorithmus mit kompetitivem Faktor c an.

Ende!



Feierabend!