Aufwand

- Begrenzender Schlüsselfaktor
- Es macht kaum einen Unterschied, ob eine Berechnung zu lange dauert oder gar nicht endet
- Gesucht sind Methoden zur Aufwandsermittlung und -analyse
- Versuch in CE-S: Gleichungsanwendungen brauchen Zeit

CE-S-Ansatz zur Aufwandsermittlung

Zeitaufwand als maximale Zahl von Gleichungsanwendungen, die nötig sind, um eine Operation für bestimmte Argumente auszuwerten in Abhängigkeit von der Länge der eingegebenen Zeichenketten und der Größe der eingegebenen Zahlen.

Schreibweise: $T^{op}(n)$ bzw. $T^{op}(n_1, \ldots, n_k)$

k: Anzahl der Zeichenkettenargumente und der Zahlenargumente von op

Vorgehensweise (1)

- Auswirkung der spezifischen Gleichungen auf die Aufwandsfunktion

Beispiel

$$\boxed{insort(x, \lambda) = x} \quad \rightsquigarrow \quad \boxed{T^{insort}(0) = 1}$$

$$\boxed{insort(x, yv) = \left.\begin{array}{l} \text{if } x \leq y \text{ then } xyv \\ \text{else } y\,insort(x, v) \end{array}\right\}} \quad \rightsquigarrow$$

$$\boxed{\begin{array}{ll} T^{insort}(n+1) & =^{?} 1 + \begin{cases} 0 \text{ falls } x \leq y \\ T^{insort}(n) \text{ sonst} \end{cases} \\ & =_{Schlechtester\ Fall} 1 + T^{insort}(n) \end{array}}$$

Vorgehensweise (2)

- Aufstellen einer nicht rekursiven Aufwandshypothese (Beweis meist durch vollständige Induktion)

  Zum Beispiel:

  $$T^{insort}(n) = n + 1 \text{ für alle } n \in \mathbb{N}$$

- Statt einer exakten Aufwandsaussage kann auch nach oben abgeschätzt werden (was oft einfacher ist).

Vorgehensweise (3)

- Beim Beweis von Aufwandsabschätzungen werden oft Eigenschaften der Operationen benötigt (Beweis mit Hilfe von Gleichwertigkeit oft mit vollständiger Induktion über Wörter).

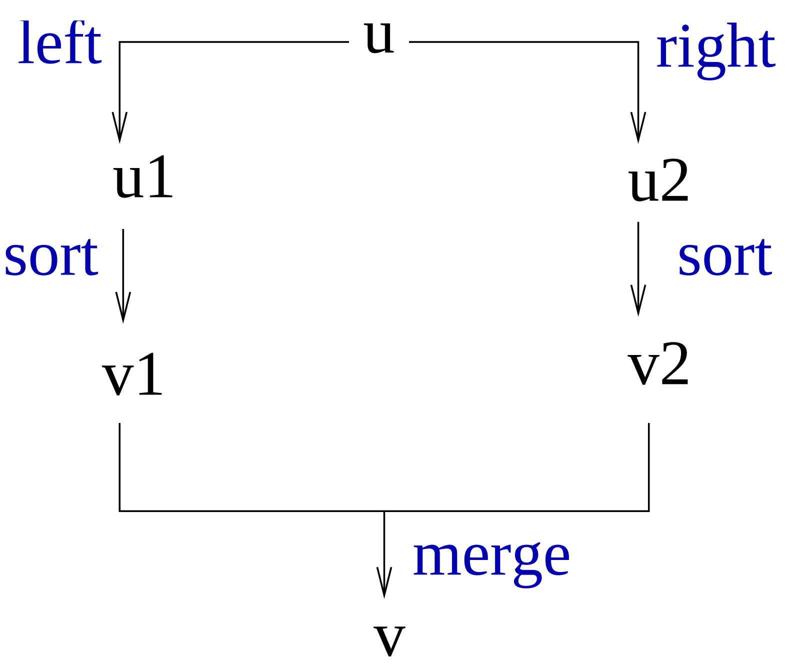
Zum Beispiel:

$$T^{sort}(n) = \frac{(n+2)(n+1)}{2}$$

falls $T^{insort}(n) = n + 1$ und $length(sort(u)) = length(u)$.

Beispiel: Sortieren durch Mischen

Idee: Sortiere linke und rechte Hälfte eines Wortes und mische die sortierten Hälften zusammen.



mergesort

opns: $\mathit{left}, \mathit{right}, \mathit{sort}: A^* \to A^*,\ \mathit{merge}: A^* \times A^* \to A^*$

vars: $x, y \in A,\ u, v, w \in A^*$

eqns: $\mathit{left}(\lambda) = \lambda\ ,\ \mathit{left}(x) = x\ ,\ \mathit{left}(xvy) = x\, \mathit{left}(v)$

$\mathit{right}(\lambda) = \lambda\ ,\ \mathit{right}(x) = \lambda\ ,\ \mathit{right}(xvy) = \mathit{right}(v)y$

$\mathit{merge}(\lambda, v) = v\ ,\ \mathit{merge}(u, \lambda) = u$

$$\mathit{merge}(xu, yv) = \begin{array}{l} \text{if } x \leq y \text{ then } x\, \mathit{merge}(u, yv) \\ \text{else } y\, \mathit{merge}(xu, v) \end{array}$$

$\mathit{sort}(\lambda) = \lambda,\ \mathit{sort}(x) = x$

$$\mathit{sort}(w) = \begin{array}{l} \mathit{merge}(\mathit{sort}(\mathit{left}(w)), \mathit{sort}(\mathit{right}(w))) \\ \text{falls } \mathit{length}(w) \geq 2 \end{array}$$

Aufwand von sort

Benötigte Eigenschaften:

1. $T^{left}(n) = T^{right}(n) = \begin{cases} \frac{n}{2} + 1 \text{ falls } n \text{ gerade,} \\ \frac{n+1}{2} \text{ sonst.} \end{cases}$

2. $T^{merge}(m,n) = \begin{cases} 1 \text{ falls } \min(m,n) = 0, \\ m + n \text{ sonst.} \end{cases}$

3. $length(sort(w)) = length(w)$.

4. $length(left(w)) = \begin{cases} \frac{length(w)}{2} & \text{falls } length(w) \text{ gerade,} \\ \frac{length(w)+1}{2} & \text{sonst.} \end{cases}$

5. $length(right(w)) = \begin{cases} \frac{length(w)}{2} & \text{falls } length(w) \text{ gerade,} \\ \frac{length(w)-1}{2} & \text{sonst.} \end{cases}$

6. $length(left(w)) + length(right(w)) = length(w)$

Aufwand von sort

1. Auswirkungen der Gleichungen auf den Aufwand:

$$T^{sort}(0) = T^{sort}(1) = 1$$

Für $n \geq 2$:

$$T^{sort}(n) = 2 + 2n + \begin{cases} 2 * T^{sort}(\frac{n}{2}) + 1 \text{ falls } n \text{ gerade,} \\ T^{sort}(\frac{n+1}{2}) + T^{sort}(\frac{n-1}{2}) \text{ sonst.} \end{cases}$$

2. Abschätzung nach oben:

$T^{sort}(n) \leq 5n * k$ für $k \geq 1$ und $2^{k-1} < n \leq 2^k$.