Auswertungsschritt (Gleichungsanwendung)

1. Zuweisung aktueller Werte $a(x) \in T_D$ an die Variablen $x \in D$ und Substitution in Gleichung $L = R$:

$$L[a] \to R[a]$$

2. Dasselbe im Argumentterm:[1]

$$f(t_1, \ldots t_{i-1}, t_i, t_{i+1}, \ldots, t_k) \longrightarrow f(t_1, \ldots, t_{i-1}, t_i', t_{i+1}, \ldots, t_k)$$

falls $t_i \longrightarrow t_i'$

[1]Wegen Rekursion in 2. kann direkte Gleichungsanwendung gemäß 1. beliebig weit innen im Term stattfinden.

Berechnung

- Berechnung als Auswertungsschrittfolge:

$$\boxed{t \equiv t_0 \longrightarrow t_1 \longrightarrow \cdots \longrightarrow t_n \equiv t' \qquad\qquad (n = 0 : t = t')}$$

dafür kurz: $t \xrightarrow{*} t'$

Gleichwertigkeit

- Gleichwertigkeit wie Berechnung mit Rückwärtsschritt ($R = L$ statt $L = R$)

$$\boxed{t \longleftrightarrow t' \text{ falls } t \longrightarrow t' \text{ oder } t \leftarrow t'}$$

$\overset{*}{\longleftrightarrow}$ analog zu $\overset{*}{\longrightarrow}$

- für $t \overset{*}{\longleftrightarrow} t'$ auch $t = t'$
- Beachte: $\longrightarrow \subseteq \overset{*}{\longrightarrow} \subseteq \overset{*}{\longleftrightarrow} \supseteq \longleftrightarrow$

Fallunterscheidung

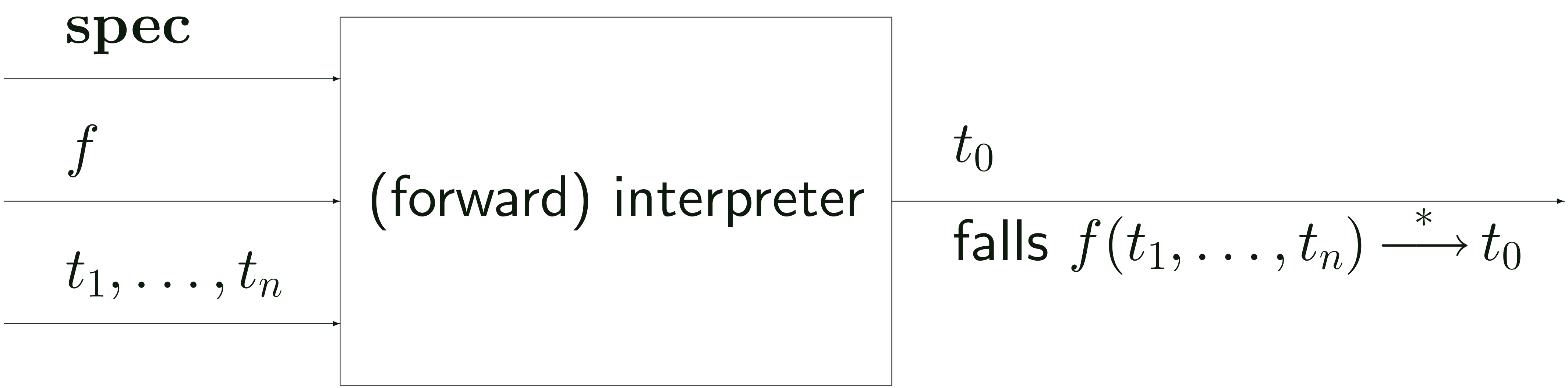
- if-then-else- als Abkürzung für 2 bedingte Gleichungen:

$$\boxed{t = \ \text{if } b \text{ then } t_1 \text{ else } t_2 \ \textsf{kurz für} \begin{cases} t = t_1 & \text{falls } b \\ t = t_2 & \text{falls } \neg b \end{cases}}$$

- Berechnungsschritte wie in Gleichungen aber bedingt:

$$\boxed{\begin{array}{ll} t[a] \longrightarrow t_1[a] & \text{falls } b[a] \overset{*}{\longleftrightarrow} T \\ t[a] \longrightarrow t_2[a] & \text{falls } b[a] \overset{*}{\longleftrightarrow} F \end{array}}$$

(Vorwärts-)Interpreter

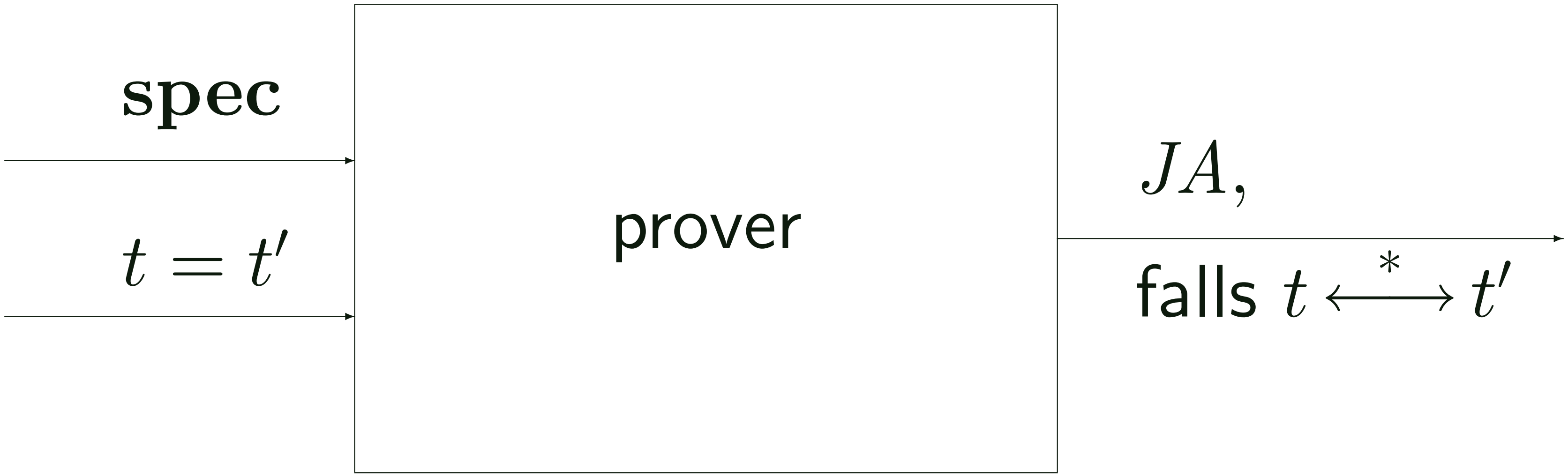


- $t_1, \ldots t_n$: Werteterme, d.h. Terme, die nur aus Grundoperationen und ohne Variablen aufgebaut sind

Beispiele für Werteterme

- $T, F, T \vee T, a \equiv b, abc \equiv abc, \ldots \in T_{BOOL}$
- $0, 1, 2, \ldots, 5 - 4, 3 + 2, length(abc), \ldots \in T_{\mathbb{N}}$
- $a, b, c, \ldots \in T_A$
- $\lambda, abc, abc \cdot abc, \ldots \in T_{A^*}$

Gleichungsbeweiser



Beispiel für eine Spezifikation in CE-S

insort

opns: $insort : A \times A^* \to A^*$, $sort : A^* \to A^*$

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

eqns: $insort(x, \lambda) = x$

$insort(x, yv) = \text{if } x \leq y \text{ then } xyv \text{ else } y\, insort(x, v)$

$sort(\lambda) = \lambda$

$sort(xu) = insort(x, sort(u))$

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

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

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

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

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 $length(sort(u)) = length(u)$.