

---

## Programmanalyse

...speziell *Datenflussanalyse*

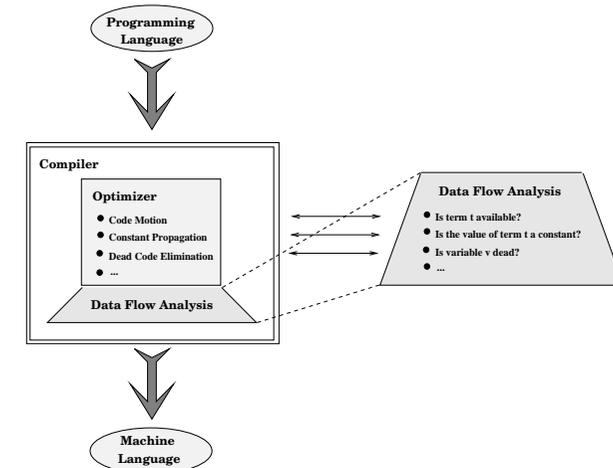
Typische Fragen sind...

- Welchen *Wert* hat eine Variable an einer Programmstelle?  
~> Konstantenausbreitung und Faltung
- Steht der Wert eines Ausdrucks an einer Programmstelle *verfügbar*?  
~> (Partielle) Redundanzelimination
- Ist eine Variable *tot* an einer Programmstelle?  
~> Elimination (partiell) toten Codes

---

## Hintergrund

...(Programm-) Analyse zur (Programm-) Optimierung



---

## In der Folge

*Zentrale Fragen...*

Grundlegendes ebenso...

- Was heißt *Optimalität*  
...in Analyse und in Optimierung?

...wie (scheinbar) Nebensächliches:

- Was ist eine *angemessene* Programmrepräsentation?

---

## Ausblick

Genauer werden wir unterscheiden...

- Intraprozedurale,
- interprozedurale,
- parallele,
- konditionale,
- ...

Datenflussanalyse.

## Ausblick

Ingredienzien *intraprozeduraler* Datenflussanalyse:

- (Lokale) *abstrakte Semantik*
  1. Ein *Datenflussanalyseverband*  $\hat{\mathcal{C}} = (\mathcal{C}, \sqcap, \sqcup, \sqsubseteq, \perp, \top)$
  2. Ein *Datenflussanalysefunktional*  $\llbracket \cdot \rrbracket : E \rightarrow (\mathcal{C} \rightarrow \mathcal{C})$
  3. Anfangsinformation/-zusicherung  $c_s \in \mathcal{C}$
- *Globalisierungsstrategien*
  1. "Meet over all Paths"-Ansatz (*MOP*)
  2. Maximaler Fixpunktansatz (*MaxFP*)
- *Generischer Fixpunktalgorithmus*

## Ausblick

Hauptresultate:

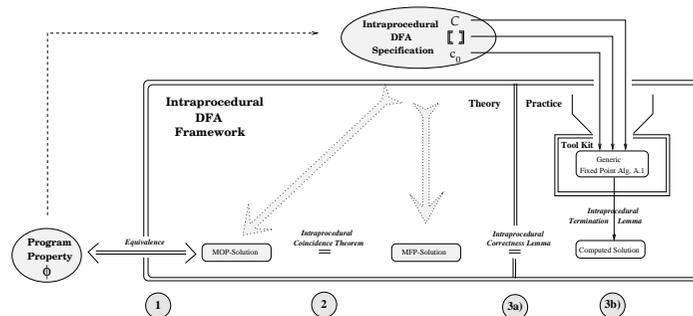
- *Sicherheits- (Korrektheits-) Theorem*
- *Koinzidenz- (Vollständigkeits-) Theorem*

Sowie:

- *Effektivitäts- (Terminierungs-) Theorem*

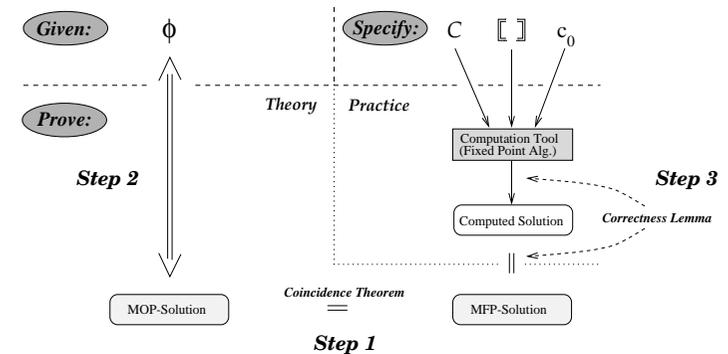
## Ausblick: Intraprozedurale Datenflussanalyse (1)

...die (detaillierte) Werkzeugkistensicht:



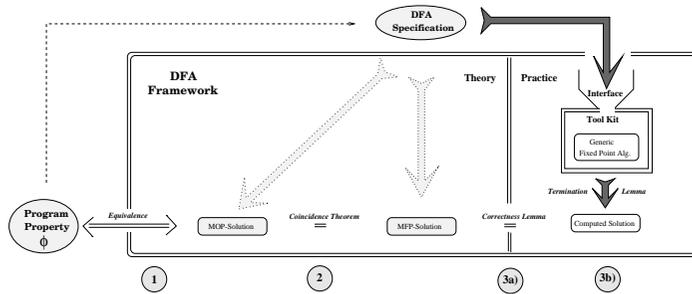
## Ausblick: Intraprozedurale Datenflussanalyse (2)

...bei genauerem Hineinsehen:



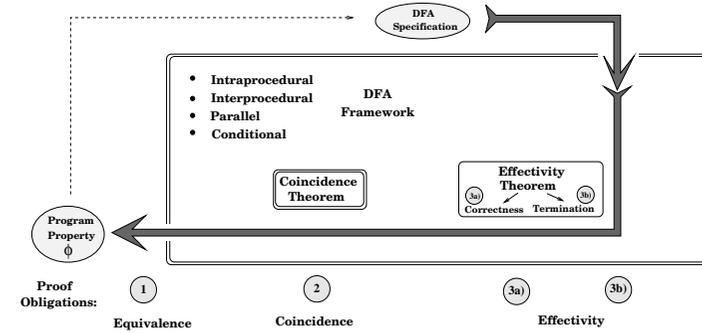
## Ausblick: DFA-Frameworks / DFA-Toolkits (1)

...aus größerer Ferne und Konzentration auf das Wesentliche:



## Ausblick: DFA-Frameworks / DFA-Toolkits (2)

...das generelle Muster, die Werkzeugkistensicht:



## Ziel

Optimale Programmpoptimierung...

...weiße Schimmel in der Informatik?

## Ohne Fleiß kein Preis!

In der Sprechweise der optimierenden Übersetzung...

...ohne Analyse keine Optimierung!

## Zurück zum Anfang: Zur Programm-analyse

...speziell *Datenflussanalyse*

Üblich ist...

- die Repräsentation von Programmen durch (nichtdeterministische) *Flussgraphen*

## Flussgraph

Ein (nichtdeterministischer) *Flussgraph* ist ein Quadrupel  $G = (N, E, s, e)$  mit

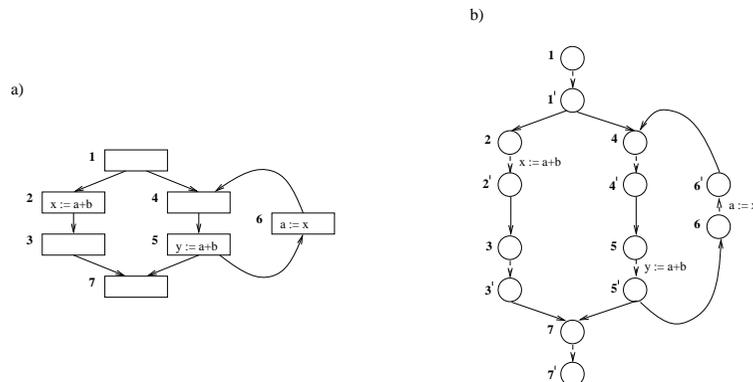
- Knotenmenge (engl. *Nodes*)  $N$
- Kantenmenge (engl. *Edges*)  $E \subseteq N \times N$
- ausgezeichnetem Startknoten  $s$  ohne Vorgänger und
- ausgezeichnetem Endknoten  $e$  ohne Nachfolger

Knoten repräsentieren Programmpunkte, Kanten repräsentieren die Verzweigungsstruktur. Elementare Programmanweisungen (Zuweisungen, Tests) können wahlweise durch Knoten oder Kanten repräsentiert werden.

↪ Darstellungsvarianten: Knoten- vs. kantenbenannte Flussgraphen

## Veranschaulichung

Knoten- vs. kantenbenannte Flussgraphen  
(hier mit Einzelanweisungsbenennung)



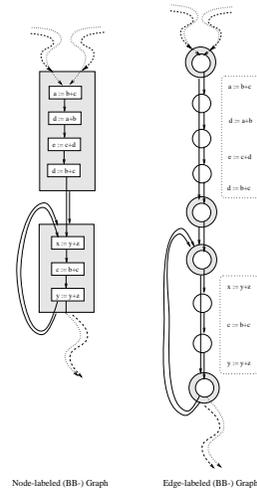
## Flussgraphen

Darstellungsvarianten...

- Knotenbenannte Graphen
  - Einzelanweisungsgraphen (SI-Graphen)
  - Basisblockgraphen (BB-Graphen)
- Kantenbenannte Graphen
  - Einzelanweisungsgraphen (SI-Graphen)
  - Basisblockgraphen (BB-Graphen)

In der Folge werden wir bevorzugt kantenbenannte SI-Graphen betrachten.

## Knoten- vs. kantenbenannte Flussgraphen



## Bezeichnungen

Sei  $G = (N, E, s, e)$  ein Flussgraph, seien  $m, n$  zwei Knoten aus  $N$ . Dann bezeichne:

- $P_G[m, n]$ : ...die Menge aller Pfade von  $m$  nach  $n$
- $P_G[m, n[$ : ...die Menge aller Pfade von  $m$  zu einem Vorgänger von  $n$
- $P_G]m, n]$ : ...die Menge aller Pfade von einem Nachfolger von  $m$  nach  $n$
- $P_G]m, n[$ : ...die Menge aller Pfade von einem Nachfolger von  $m$  zu einem Vorgänger von  $n$

*Bem.:* Wenn  $G$  aus dem Kontext eindeutig hervorgeht, schreiben wir einfacher auch  $P$  statt  $P_G$ .

## Datenflussanalysespezifikation

- (Lokale) abstrakte Semantik
  1. Ein Datenflussanalyseverband  $\tilde{\mathcal{C}} = (\mathcal{C}, \sqcap, \sqcup, \sqsubseteq, \perp, \top)$
  2. Ein Datenflussanalysefunktional  $\llbracket \cdot \rrbracket : E \rightarrow (\mathcal{C} \rightarrow \mathcal{C})$
- Eine Anfangsinformation/-zusicherung:  $c_s \in \mathcal{C}$

## Globalisierung einer lokalen abstrakten Semantik

Zwei Strategien:

- "Meet over all Paths"-Ansatz (*MOP*)  
 $\rightsquigarrow$  spezifizierende Lösung
- Maximaler Fixpunktansatz (*MaxFP*)  
 $\rightsquigarrow$  berechenbare Lösung

---

## Der *MOP* -Ansatz

*Zentral:* Ausdehnung der lokalen abstrakten Semantik auf Pfade

$$\llbracket p \rrbracket =_{df} \begin{cases} Id_{\mathcal{C}} & \text{falls } q < 1 \\ \llbracket \langle e_2, \dots, e_q \rangle \rrbracket \circ \llbracket e_1 \rrbracket & \text{sonst} \end{cases}$$

---

## Die *MOP* -Lösung

$$\forall c_s \in \mathcal{C} \forall n \in N. MOP_{c_s}(n) = \sqcap \{ \llbracket p \rrbracket (c_s) \mid p \in \mathbf{P}[s, n] \}$$

---

## Der *MaxFP* -Ansatz

*Zentral:* Das *MaxFP* -Gleichungssystem:

$$\mathbf{inf}(n) = \begin{cases} c_s & \text{falls } n = s \\ \sqcap \{ \llbracket (m, n) \rrbracket (\mathbf{inf}(m)) \mid m \in \text{pred}(n) \} & \text{sonst} \end{cases}$$

---

## Die *MaxFP* -Lösung

$$\forall c_s \in \mathcal{C} \forall n \in N. MaxFP_{(\llbracket \cdot \rrbracket, c_s)}(n) =_{df} \mathbf{inf}_{c_s}^*(n)$$

wobei  $\mathbf{inf}_{c_s}^*$  die größte Lösung des *MaxFP* -Gleichungssystems bezeichnet.

---

## Generischer Fixpunktalgorithmus 1(2)

**Eingabe:** (1) Ein Flussgraph  $G=(N,E,s,e)$ , (2) eine (lokale) abstrakte Semantik bestehend aus einem Datenflussanalyseverband  $\mathcal{C}$ , einem Datenflussanalysefunktional  $\llbracket \ ] : E \rightarrow (\mathcal{C} \rightarrow \mathcal{C})$ , und (3) einer Anfangsinformation  $c_s \in \mathcal{C}$ .

**Ausgabe:** Unter den Voraussetzungen des Effektivitätstheorems (später!) die *MaxFP*-solution. Abhängig von den Eigenschaften des Datenflussanalysefunktionals gilt dann:

(1)  $\llbracket \ ]$  ist *distributiv*: Variable *inf* enthält für jeden Knoten die stärkste Nachbedingung bezüglich der Anfangsinformation  $c_s$ .

(2)  $\llbracket \ ]$  ist *monoton*: Variable *inf* enthält für jeden Knoten eine sichere (d.h. untere) Approximation der stärksten Nachbedingung bezüglich der Anfangsinformation  $c_s$ .

**Bemerkung:** Die Variable *workset* steuert den iterativen Prozess. Ihre Elemente sind Knoten aus  $G$ , deren Annotation jüngst aktualisiert worden ist.

---

## Generischer Fixpunktalgorithmus 2(2)

( Prolog: Initialisierung von *inf* and *workset* )

```
FORALL  $n \in N \setminus \{s\}$  DO  $inf[n] := \top$  OD;  
 $inf[s] := c_s$ ;  
 $workset := \{s\}$ ;
```

( Hauptprozess: Iterative Fixpunktberechnung )

```
WHILE  $workset \neq \emptyset$  DO  
  CHOOSE  $m \in workset$ ;  
   $workset := workset \setminus \{m\}$ ;  
  ( Aktualisiere die Nachfolgerumgebung von Knoten  $m$  )  
  FORALL  $n \in succ(m)$  DO  
     $meet := \llbracket (m, n) \rrbracket (inf[m]) \sqcap inf[n]$ ;  
    IF  $inf[n] \sqsupseteq meet$   
    THEN  
       $inf[n] := meet$ ;  
       $workset := workset \cup \{n\}$   
    FI  
  OD  
ESOOHC  
OD.
```

---

## Hauptresultate

Zusammenhang von...

- *MOP* - und *MaxFP* -Lösung
  - Korrektheit
  - Vollständigkeit
- *MaxFP* -Lösung und generischem Algorithmus
  - Terminierung mit *MaxFP* -Lösung

---

## Korrektheit: Sicherheitstheorem

**Theorem** [Sicherheit (Safety)]

Die *MaxFP*-Lösung ist eine sichere (konservative). d.h. untere Approximation der *MOP*-Lösung, d.h.,

$$\forall c_s \in \mathcal{C} \forall n \in N. MaxFP_{c_s}(n) \sqsubseteq MOP_{c_s}(n)$$

falls das Datenflussanalysefunktional  $\llbracket \ ]$  monoton ist.

---

## Vollständigkeit (und Korrektheit): Ko- inzidenztheorem

**Theorem** [Koinzidenz (Coincidence)]

Die *MaxFP*-solution stimmt mit der *MOP*-Lösung überein, d.h.,

$$\forall c_s \in \mathcal{C} \forall n \in \mathbb{N}. \text{MaxFP}_{c_s}(n) = \text{MOP}_{c_s}(n)$$

falls das Datenflussanalysefunktional  $\llbracket \cdot \rrbracket$  distributiv ist.

---

## Terminierung: Effektivitätstheorem

**Theorem** [Effektivität]

Der generische Fixpunktalgorithmus terminiert mit der *MaxFP*-Lösung, falls das Datenflussanalysefunktional monoton ist und der Verband die absteigende Kettenbedingung erfüllt.

---

## Nachzutragende Definitionen

...sind:

- Absteigende (aufsteigende) Kettenbedingung
- Monotonie und Distributivität von Datenflussanalysefunktionalen

---

## Auf-/absteigende Kettenbedingung

**Definition** [Ab-/aufsteigende Kettenbedingung]

Ein Verband  $\hat{\mathcal{C}} = (\mathcal{C}, \sqcap, \sqcup, \sqsubseteq, \sqsupseteq, \perp, \top)$  erfüllt

1. die *absteigende Kettenbedingung*, falls jede absteigende Kette stationär wird, d.h. für jede Kette  $p_1 \sqsupseteq p_2 \sqsupseteq \dots \sqsupseteq p_n \sqsupseteq \dots$  gibt es einen Index  $m \geq 1$  so dass  $x_m = x_{m+j}$  für alle  $j \in \mathbb{N}$  gilt
2. die *aufsteigende Kettenbedingung*, falls jede aufsteigende Kette stationär wird, d.h. für jede Kette  $p_1 \sqsubseteq p_2 \sqsubseteq \dots \sqsubseteq p_n \sqsubseteq \dots$  gibt es einen Index  $m \geq 1$  so dass  $x_m = x_{m+j}$  für alle  $j \in \mathbb{N}$  gilt

---

## Monotonie, Distributivität, Additivität

...von Funktionen auf (Datenflussanalyse-) Verbänden.

**Definition** [Monotonie, Distributivität, Additivität]

Sei  $\mathcal{C} = (\mathcal{C}, \sqcap, \sqcup, \sqsubseteq, \perp, \top)$  ein vollständiger Verband und  $f : \mathcal{C} \rightarrow \mathcal{C}$  eine Funktion auf  $\mathcal{C}$ . Dann heißt  $f$

1. *monoton* gdw  $\forall c, c' \in \mathcal{C}. c \sqsubseteq c' \Rightarrow f(c) \sqsubseteq f(c')$   
(Erhalt der Ordnung der Elemente)
2. *distributiv* gdw  $\forall C' \subseteq \mathcal{C}. f(\sqcap C') = \sqcap \{f(c) \mid c \in C'\}$   
(Erhalt der größten unteren Schranken)
3. *additiv* gdw  $\forall C' \subseteq \mathcal{C}. f(\sqcup C') = \sqcup \{f(c) \mid c \in C'\}$   
(Erhalt der kleinsten oberen Schranken)

---

## Zur Erinnerung: Oft nützlich

...ist folgende äquivalente Charakterisierung der Monotonie:

**Lemma**

Sei  $\mathcal{C} = (\mathcal{C}, \sqcap, \sqcup, \sqsubseteq, \perp, \top)$  ein vollständiger Verband und  $f : \mathcal{C} \rightarrow \mathcal{C}$  eine Funktion auf  $\mathcal{C}$ . Dann gilt:

$$f \text{ ist monoton} \iff \forall C' \subseteq \mathcal{C}. f(\sqcap C') \sqsubseteq \sqcap \{f(c) \mid c \in C'\}$$

---

## Monotonie und Distributivität

...von Datenflussanalysefunktionalen.

**Definition**

Ein Datenflussanalysefunktional  $\llbracket \cdot \rrbracket : E \rightarrow (\mathcal{C} \rightarrow \mathcal{C})$  heißt *monoton (distributiv)* gdw  $\forall e \in E. \llbracket e \rrbracket$  ist monoton (distributiv).

---

## Beispiel: Verfügbare Ausdrücke

...ein typisches distributives DFA-Problem.

• **Abstrakte Semantik für verfügbare Ausdrücke:**

1. *Datenflussanalyseverband:*  
 $(\mathcal{C}, \sqcap, \sqcup, \sqsubseteq, \perp, \top) =_{df} (\mathcal{B}, \wedge, \vee, \leq, false, true)$
2. *Datenflussanalysefunktional:*  $\llbracket \cdot \rrbracket_{av} : E \rightarrow (\mathcal{B} \rightarrow \mathcal{B})$  definiert durch

$$\forall e \in E. \llbracket e \rrbracket_{av} =_{df} \begin{cases} Cst_{true} & \text{falls } Comp_e \wedge Transp_e \\ Id_{\mathcal{B}} & \text{falls } \neg Comp_e \wedge Transp_e \\ Cst_{false} & \text{sonst} \end{cases}$$

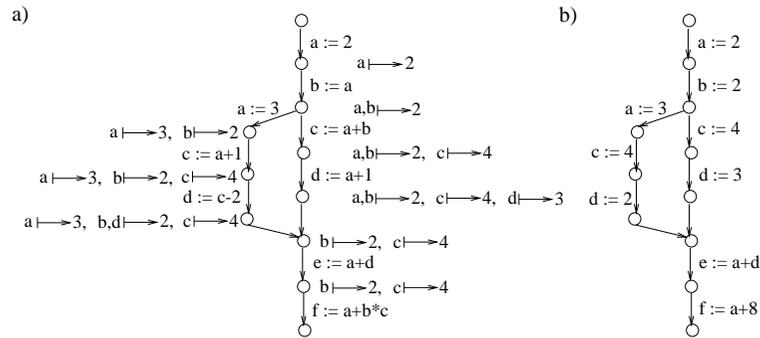
wobei

$$\widehat{\mathcal{B}} =_{df} (\mathcal{B}, \wedge, \vee, \leq, false, true)$$

den Verband der Wahrheitswerte bezeichnet mit  $false \leq true$  und dem logischen "und" und "oder" als Schnitt- bzw. Vereinigungsoperation  $\sqcap$  and  $\sqcup$ .

## Beispiel: Einfache Konstanten

Ein typisches monotones (nicht distributives) DFA-Problem...



## Abstrakte Semantik für einfache Konstanten

### • Abstrakte Semantik für einfache Konstanten:

#### 1. Datenflussanalyseverband:

$$(\mathcal{C}, \sqcap, \sqcup, \sqsubseteq, \perp, \top) =_{df} (\Sigma, \sqcap, \sqcup, \sqsubseteq, \sigma_{\perp}, \sigma_{\top})$$

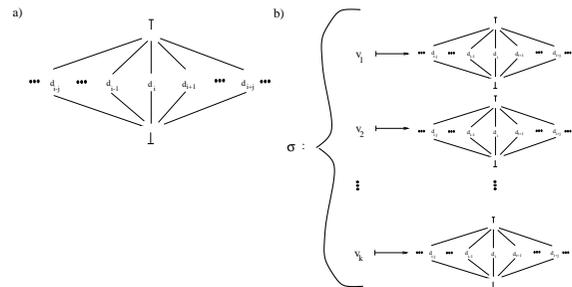
#### 2. Datenflussanalysefunktional:

$$\llbracket \cdot \rrbracket_{sc} : E \rightarrow (\Sigma \rightarrow \Sigma) \text{ definiert durch}$$

$$\forall e \in E. \llbracket e \rrbracket_{sc} =_{df} \theta_e$$

## Datenflussanalyseverband für einfache Konstanten

Der "kanonische" Verband für Konstantenausbreitung/-faltung:



## Die Semantik von Termen

Die *Semantik* von Termen  $t \in \mathbf{T}$  ist gegeben durch die *Evaluationsfunktion*

$$\mathcal{E} : \mathbf{T} \rightarrow (\Sigma \rightarrow \mathbf{D})$$

die induktiv definiert ist durch:

$$\forall t \in \mathbf{T} \forall \sigma \in \Sigma. \mathcal{E}(t)(\sigma) =_{df} \begin{cases} \sigma(x) & \text{falls } t = x \in \mathbf{V} \\ I_0(c) & \text{falls } t = c \in \mathbf{C} \\ I_0(op)(\mathcal{E}(t_1)(\sigma), \dots, \mathcal{E}(t_r)(\sigma)) & \text{falls } t = op(t_1, \dots, t_r) \end{cases}$$

---

## Nachzutragende Begriffe und Definitionen

...um die Definition der Termsemantik abzuschließen:

- Termsyntax
- Interpretation
- Zustand

---

## Die Syntax von Termen (1)

Sei

- $V$  eine Menge von Variablen und
- $Op$  eine Menge von  $n$ -stelligen Operatoren,  $n \geq 0$ , sowie  $C \subseteq Op$  die Menge der 0-stelligen Operatoren, der sog. *Konstanten* in  $Op$ .

---

## Die Syntax von Termen (2)

Dann legen wir fest:

1. Jede Variable  $v \in V$  und jede Konstante  $c \in C$  ist ein Term.
2. Ist  $op \in Op$  ein  $n$ -stelliger Operator,  $n \geq 1$ , und sind  $t_1, \dots, t_n$  Terme, dann ist auch  $op(t_1, \dots, t_n)$  ein Term.
3. Es gibt keine weiteren Terme außer den nach den obigen beiden Regeln konstruierbaren.

Die Menge aller Terme bezeichnen wir mit  $T$ .

---

## Interpretation

Sei  $D'$  ein geeigneter Datenbereich (z.B. die Menge der ganzen Zahlen), seien  $\perp$  und  $\top$  zwei ausgezeichnete Elemente mit  $\perp, \top \notin D'$  und sei  $D =_{df} D' \cup \{\perp, \top\}$ .

Eine *Interpretation* über  $T$  und  $D$  ist ein Paar  $I \equiv (D, I_0)$ , wobei

- $I_0$  eine Funktion ist, die mit jedem 0-stelligen Operator  $c \in Op$  ein Datum  $I_0(c) \in D'$  und mit jedem  $n$ -stelligen Operator  $op \in Op$ ,  $n \geq 1$ , eine totale Funktion  $I_0(op) : D^n \rightarrow D$  assoziiert, die als *strikt* angenommen wird (d.h.  $I_0(op)(d_1, \dots, d_n) = \perp$ , wann immer es ein  $j \in \{1, \dots, n\}$  gibt mit  $d_j = \perp$ )

---

## Menge der Zustände

$$\Sigma =_{df} \{ \sigma \mid \sigma : \mathbf{V} \rightarrow \mathbf{D} \}$$

...bezeichnet die Menge der *Zustände*, d.h. die Menge der Abbildungen  $\sigma$  von der Menge der Programmvariablen  $\mathbf{V}$  auf einen geeigneten (hier nicht näher spezifizierten) Datenbereich  $\mathbf{D}$ .

Insbesondere

- $\sigma_{\perp}$ : ...bezeichnet den wie folgt definierten *total undefinierten* Zustand aus  $\Sigma$ :  $\forall v \in \mathbf{V}. \sigma_{\perp}(v) = \perp$

---

## Zustandstransformationsfunktion

Die *Zustandstransformationsfunktion*

$$\theta_t : \Sigma \rightarrow \Sigma, \quad t \equiv x := t$$

ist definiert durch:

$$\forall \sigma \in \Sigma \quad \forall y \in \mathbf{V}. \theta_t(\sigma)(y) =_{df} \begin{cases} \mathcal{E}(t)(\sigma) & \text{falls } y = x \\ \sigma(y) & \text{sonst} \end{cases}$$

---

## Der funktionale *MaxFP* -Ansatz

*Zentral*: Das "funktionale" *MaxFP* -Gleichungssystem:

$$\llbracket n \rrbracket = \begin{cases} Id_{\mathcal{C}} & \text{falls } n = s \\ \bigcap \{ \llbracket (n, m) \rrbracket \circ \llbracket m \rrbracket \mid m \in \text{pred}(n) \} & \text{sonst} \end{cases}$$

In der Folge bezeichne das Funktional

$$\llbracket \rrbracket : N \rightarrow (\mathcal{C} \rightarrow \mathcal{C})$$

die größte Lösung des obigen Gleichungssystems.

---

## Zusammenhang von *MaxFP* - und funktionalem *MaxFP* -Ansatz

**Theorem** [Äquivalenz]

$$\forall n \in N \quad \forall c_s \in \mathcal{C}. \text{MaxFP}_{(G, \llbracket \rrbracket)}(n)(c_s) = \llbracket n \rrbracket(c_s)$$

---

## Ausblick

Die funktionale Perspektive auf den *MaxFP* -Ansatz liefert den Schlüssel zu

- interprozeduraler (d.h. von Programmen mit Prozeduren)
- paralleler (d.h. von Programmen mit Parallelität)

Datenflussanalyse.

---

## Vorschau auf die weiteren Vorlesungstermine...

- Mo, 26.11.2007: Vorlesung von 16:15 Uhr bis 17:45 Uhr im Hörsaal 14, TU-Hauptgebäude
- Mo, 03.12.2007: Vorlesung von 16:15 Uhr bis 17:45 Uhr im Hörsaal 14, TU-Hauptgebäude
- Mo, 10.12.2007: Vorlesung von 16:15 Uhr bis 17:45 Uhr im Hörsaal 14, TU-Hauptgebäude
- *Mo, 17.12./24.12./31.12.2007: Keine Vorlesung(en)! (Ferialzeit)*