Heutiges Thema

Von Verifikation zu Analyse...

- *Worst-Case Execution Time*-Analyse als erstes Beispiel
- Nachträge zu mathematischen Grundlagen

...nach

- Hanne Riis Nielson, Flemming Nielson. *Semantics with Applications – A Formal Introduction*, Wiley, 1992.

Worst-Case Execution Time (WCET)-Analyse

Motivation:

- In vielen Anwendungsbereichen sind Aussagen über die Ausführungszeit erforderlich.
- Der Nachweis totaler Korrektheit garantiert zwar Terminierung, sagt aber nichts über den Ressourcen-, speziell den Zeitbedarf aus.

In der Folge:

- Erweiterung und Adaptierung des Beweissystems für totale Korrektheit, um solche Aussagen zu ermöglichen.

Die grundlegende Idee (1)

...zur Zuordnung von Ausführungszeiten:

- *Leere Anweisung*
  ...Ausführungszeit in $\mathcal{O}(1)$, d.h. Ausführungszeit ist beschränkt durch eine Konstante.
- *Zuweisung*
  ...Ausführungszeit in $\mathcal{O}(1)$.
- (*Sequentielle*) *Komposition*
  ...Ausführungszeit entspricht, bis auf einen konstanten Faktor, der Summe der Ausführungszeiten der Komponenten.

Die grundlegende Idee (2)

- *Fallunterscheidung*
  ...Ausführungszeit entspricht, bis auf einen konstanten Faktor, der gößeren der Ausführungszeiten der beiden Zweige.
- (*while*)-*Schleife*
  ...Ausführungszeit der Schleife entspricht, bis auf einen konstanten Faktor, der Summe der wiederholten Ausführungszeiten des Rumpfes der Schleife.

Bemerkung: Verfeinerungen sind offenbar möglich.

Formalisierung

...dieser grundlegenden Idee in 3 Schritten:

1. Angabe einer Semantik, die die Auswertungszeit arithmetischer und Boolescher Ausdrücke beschreibt.
2. Erweiterung und Adaption der natürlichen Semantik von WHILE zur Bestimmung der Ausführungszeit eines Programms.
3. Erweiterung und Adaption des Beweissystems für totale Korrektheit zum Nachweis über die Größenordnung der Ausführungszeit von Programmen.

Erster Schritt

Festlegung von Semantikfunktionen

- $[\![\,.\,]\!]_{TA}$: **Aexpr** $\to \mathbb{Z}$ und
- $[\![\,.\,]\!]_{TB}$: **Bexpr** $\to \mathbb{Z}$

zur Beschreibung der Auswertungszeit arithmetischer und Boolescher Ausdrücke (in Zeiteinheiten einer abstrakten Maschine).

Semantik zur Ausführungszeit der Auswertung arithmetischer Ausdrücke

$[\![\,.\,]\!]_{TA}$: **Aexpr** $\to \mathbb{Z}$ induktiv definiert durch

- $[\![n]\!]_{TA} =_{df} \mathbf{1}$
- $[\![x]\!]_{TA} =_{df} \mathbf{1}$
- $[\![a_1 + a_2]\!]_{TA} =_{df} [\![a_1]\!]_{TA} + [\![a_2]\!]_{TA} + \mathbf{1}$
- $[\![a_1 * a_2]\!]_{TA} =_{df} [\![a_1]\!]_{TA} + [\![a_2]\!]_{TA} + \mathbf{1}$
- $[\![a_1 - a_2]\!]_{TA} =_{df} [\![a_1]\!]_{TA} + [\![a_2]\!]_{TA} + \mathbf{1}$
- $[\![a_1 / a_2]\!]_{TA} =_{df} [\![a_1]\!]_{TA} + [\![a_2]\!]_{TA} + \mathbf{1}$
- ... (andere Operatoren analog, ggf. auch mit operationsspezifischen Kosten)

Anmerkungen zu $[\![\,.\,]\!]_{TA}$ und $[\![\,.\,]\!]_{TB}$

Die Semantikfunktionen

- $[\![\,.\,]\!]_{TA}$ und $[\![\,.\,]\!]_{TB}$

...beschreiben intuitiv die Anzahl der Zeiteinheiten, die eine (hier nicht spezifizierte) abstrakte Maschine zur Auswertung arithmetischer und Boolescher Ausdrücke benötigt.

Semantik zur Ausführungszeit der Auswertung Boolescher Ausdrücke

$[\![\,.\,]\!]_{TB}$: **Bexpr** $\rightarrow \mathbb{Z}$ induktiv definiert durch

- $[\![\, true \,]\!]_{TB} =_{df} \mathbf{1}$
- $[\![\, false \,]\!]_{TB} =_{df} \mathbf{1}$
- $[\![\, a_1 = a_2 \,]\!]_{TB} =_{df} [\![\, a_1 \,]\!]_{TA} + [\![\, a_2 \,]\!]_{TA} + \mathbf{1}$
- $[\![\, a_1 < a_2 \,]\!]_{TB} =_{df} [\![\, a_1 \,]\!]_{TA} + [\![\, a_2 \,]\!]_{TA} + \mathbf{1}$
- ... (andere Relatoren (z.B. $\leq$, ...) analog)
- $[\![\, \neg b \,]\!]_{TB} =_{df} [\![\, b \,]\!]_{TB} + \mathbf{1}$
- $[\![\, b_1 \wedge b_2 \,]\!]_{TB} =_{df} [\![\, b_1 \,]\!]_{TB} + [\![\, b_2 \,]\!]_{TB} + \mathbf{1}$
- $[\![\, b_1 \vee b_2 \,]\!]_{TB} =_{df} [\![\, b_1 \,]\!]_{TB} + [\![\, b_2 \,]\!]_{TB} + \mathbf{1}$

Zweiter Schritt

Erweiterung und Adaption der

- natürlichen Semantik von WHILE

zur Bestimmung der Ausführungszeit von Programmen.

Idee

Übergang zu Transitionen der Form

$$\langle \pi, \sigma \rangle \ \rightarrow^{t} \ \sigma'$$

mit der Bedeutung, dass π angesetzt auf σ nach t Zeiteinheiten in σ' terminiert.

Natürliche Semantik erweitert um den Ausführungszeitaspekt (1)

...für das Beispiel von WHILE:

[skip$_{tns}$] $\dfrac{—}{\langle skip,\sigma\rangle \ \rightarrow^{1} \ \sigma}$

[abort$_{tns}$] $\dfrac{—}{\langle abort,\sigma\rangle \ \rightarrow^{1} \ error}$

[ass$_{tns}$] $\dfrac{—}{\langle x{:=}t,\sigma\rangle \ \rightarrow^{[\![\, t \,]\!]_{TA}+1} \ \sigma[[\![\, t \,]\!]_A(\sigma)/x]}$

[comp$_{tns}$] $\dfrac{\langle \pi_1,\sigma\rangle \ \rightarrow^{t_1} \ \sigma', \langle \pi_2,\sigma'\rangle \ \rightarrow^{t_2} \ \sigma''}{\langle \pi_1;\pi_2,\sigma\rangle \ \rightarrow^{t_1+t_2} \ \sigma''}$

Natürliche Semantik erweitert um den Ausführungszeitaspekt (2)

$$[\text{if}^{tt}_{tns}] \quad \frac{\langle \pi_1, \sigma \rangle \to^t \sigma'}{\langle \text{if } b \text{ then } \pi_1 \text{ else } \pi_2 \text{ fi}, \sigma \rangle \to^{[\![b]\!]_{TB}+t+1} \sigma'} \quad [\![b]\!]_B(\sigma) = \mathtt{tt}$$

$$[\text{if}^{ff}_{tns}] \quad \frac{\langle \pi_2, \sigma \rangle \to^t \sigma'}{\langle \text{if } b \text{ then } \pi_1 \text{ else } \pi_2 \text{ fi}, \sigma \rangle \to^{[\![b]\!]_{TB}+t+1} \sigma'} \quad [\![b]\!]_B(\sigma) = \mathtt{ff}$$

$$[\text{while}^{tt}_{tns}] \quad \frac{\langle \pi, \sigma \rangle \to^t \sigma', \langle \text{while } b \text{ do } \pi \text{ od}, \sigma' \rangle \to^{t'} \sigma''}{\langle \text{while } b \text{ do } \pi \text{ od}, \sigma \rangle \to^{[\![b]\!]_{TB}+t+t'+2} \sigma''} \quad [\![b]\!]_B(\sigma) = \mathtt{tt}$$

$$[\text{while}^{ff}_{tns}] \quad \frac{\text{---}}{\langle \text{while } b \text{ do } \pi \text{ od}, \sigma \rangle \to^{[\![b]\!]_{TB}+3} \sigma} \quad [\![b]\!]_B(\sigma) = \mathtt{ff}$$

Dritter Schritt

Erweiterung und Adaption der

- des Beweiskalküls für totale Korrektheit

um den Ausführungszeitaspekt von Programmen.

Idee (1)

Übergang zu Korrektheitsformeln der Form

$$\{p\} \ \pi \ \{e \Downarrow q\}$$

wobei

- p und q Prädikate (wie bisher!) und
- $e \in$ **Aexp** ein arithmetischer Ausdruck ist.

Idee (2)

Die Korrektheitsformel

$$\{p\} \ \pi \ \{e \Downarrow q\}$$

ist gültig gdw. für jeden Anfangszustand σ gilt: ist die Vorbedingung p in σ erfüllt, **dann** terminiert die zugehörige Berechnung von π angesetzt auf σ regulär mit einem Endzustand σ' **und** die Nachbedingung q ist in σ' erfüllt, und die benötigte Ausführungszeit ist in $\mathcal{O}(e)$.

Axiomatische Semantik zum Ausführungszeitaspekt (1)

[skip$_e$] $\dfrac{\overline{}}{\{p\}\ skip\ \{1 \Downarrow p\}}$

[ass$_e$] $\dfrac{\overline{}}{\{p[t \backslash x]\}\ x{:=}t\ \{1 \Downarrow p\}}$

[comp$_e$] $\dfrac{\{p \wedge e_2'{=}u\}\ \pi_1\ \{e_1 \Downarrow r \wedge e_2{\le}u\},\ \{r\}\ \pi_2\ \{e_2 \Downarrow q\}}{\{p\}\ \pi_1;\pi_2\ \{e_1{+}e_2' \Downarrow q\}}$

wobei u frische logische Variable ist

[ite$_e$] $\dfrac{\{p \wedge b\}\ \pi_1\ \{e \Downarrow q\},\ \{p \wedge \neg b\}\ \pi_2\ \{e \Downarrow q\}}{\{p\}\ \text{if } b \text{ then } \pi_1 \text{ else } \pi_2 \text{ fi}\ \{e \Downarrow q\}}$

[cons$_e$] $\dfrac{\{p'\}\ \pi\ \{e' \Downarrow q'\}}{\{p\}\ \pi\ \{e \Downarrow q\}}$

wobei (für eine natürliche Zahl k) $p \Rightarrow p' \wedge e' \le k * e$ und $q' \Rightarrow q$

Axiomatische Semantik zum Ausführungszeitaspekt (2)

[while$_e$] $\dfrac{\{p(z{+}1) \wedge e'{=}u\}\ \pi\ \{e_1 \Downarrow p(z) \wedge e{\le}u\}}{\{\exists z.\ p(z)\}\ \text{while } b \text{ do } \pi \text{ od}\ \{e \Downarrow p(0)\}}$

wobei $p(z+1) \Rightarrow b \wedge e \geq e_1 + e',\ p(0) {\Rightarrow} \neg b \wedge 1 \leq e$
u eine frische logische Variable ist und
z Werte aus den natürlichen Zahlen annimmt (d.h. $z \geq 0$)

Beispiele (1)

Die Korrektheitsformel

```
                                             ||
{x=3} y:=1; while x/=1 do y:=y*x; x:=x-1 od {1 \/ True}
```

beschreibt, dass die Ausführungszeit des Fakultätsprogramms angesetzt auf einen Zustand, in dem x den Wert 3 hat, von der Grössenordnung von 1 ist, also durch eine Konstante beschränkt ist.

Beispiele (2)

Die Korrektheitsformel

```
                                             ||
{x>0} y:=1; while x/=1 do y:=y*x; x:=x-1 od {x \/ True}
```

beschreibt, dass die Ausführungszeit des Fakultätsprogramms angesetzt auf einen Zustand, in dem x einen Wert größer als 0 hat, von der Grössenordnung von x ist, also linear beschränkt ist.

Nachträge

Mathematische Grundlagen im Zusammenhang mit der...

1. Definition abstrakter Semantiken für Programmanalysen
2. Definition der denotationellen Semantik von **WHILE** im Detail

Wichtig insbesondere...

- Mengen, Relationen, Verbände
- Partielle und vollständige partielle Ordnungen
- Schranken, Fixpunkte und Fixpunkttheoreme

Mengen und Relationen 1(2)

Sei M eine Menge und R eine Relation auf M, d.h. $R \subseteq M \times M$.

Dann heißt R...

- *reflexiv* gdw. $\forall\, m \in M.\ m\,R\,m$
- *transitiv* gdw. $\forall\, m, n, p \in M.\ m\,R\,n \ \wedge\ n\,R\,p \ \Rightarrow\ m\,R\,p$
- *antisymmetrisch* gdw. $\forall\, m, n \in M.\ m\,R\,n \wedge n\,R\,m \ \Rightarrow\ m = n$

Darüberhinaus... (in der Folge allerdings weniger wichtig)

- *symmetrisch* gdw. $\forall\, m, n \in M.\ m\,R\,n \iff n\,R\,m$
- *total* gdw. $\forall\, m, n \in M.\ m\,R\,n \ \vee\ n\,R\,m$

Mengen und Relationen 2(2)

Eine Relation R auf M heißt

- *Quasiordnung* gdw. R ist reflexiv und transitiv
- *partielle Ordnung* gdw. R ist reflexiv, transitiv und antisymmetrisch

Zur Vollständigkeit sei ergänzt...

- *Äquivalenzrelation* gdw. R ist reflexiv, transitiv und symmetrisch

...eine partielle Ordnung ist also eine antisymmetrische Quasiordnung, eine Äquivalenzrelation eine symmetrische Quasiordnung.

Schranken, kleinste, größte Elemente

Sei $(Q, \sqsubseteq)$ eine Quasiordnung, sei $q \in Q$ und $Q' \subseteq Q$.

Dann heißt q...

- *obere* (*untere*) *Schranke* von Q', in Zeichen: $Q' \sqsubseteq q$ ($q \sqsubseteq Q'$), wenn für alle $q' \in Q'$ gilt: $q' \sqsubseteq q$ ($q \sqsubseteq q'$)
- *kleinste obere* (*größte untere*) *Schranke* von Q', wenn q obere (untere) Schranke von Q' ist und für jede andere obere (untere) Schranke $\widehat{q}$ von Q' gilt: $q \sqsubseteq \widehat{q}$ ($\widehat{q} \sqsubseteq q$)
- *größtes* (*kleinstes*) Element von Q, wenn gilt: $Q \sqsubseteq q$ ($q \sqsubseteq Q$)

Eindeutigkeit von Schranken

- In partiellen Ordnungen sind kleinste obere und größte untere Schranken eindeutig bestimmt, wenn sie existieren.
- Existenz (und damit Eindeutigkeit) vorausgesetzt, wird die kleinste obere (größte untere) Schranke einer Menge $P' \subseteq P$ der Grundmenge einer partiellen Ordnung $(P, \sqsubseteq)$ mit $\bigsqcup P'$ ($\bigsqcap P'$) bezeichnet. Man spricht dann auch vom *Supremum* und *Infimum* von P'.
- Analog für kleinste und größte Elemente. Existenz vorausgesetzt, werden sie üblicherweise mit $\perp$ und $\top$ bezeichnet.

Verbände und vollständige Verbände

Sei $(P, \sqsubseteq)$ eine partielle Ordnung.

Dann heißt $(P, \sqsubseteq)$...

- *Verband*, wenn jede *endliche* Teilmenge P' von P eine kleinste obere und eine größte untere Schranke in P besitzt
- *vollständiger Verband*, wenn *jede* Teilmenge P' von P eine kleinste obere und eine größte untere Schranke in P besitzt

...(vollständige) Verbände sind also spezielle partielle Ordnungen.

Vollständige partielle Ordnungen

...ein etwas schwächerer, aber in der Informatik oft ausreichender und daher angemessenerer Begriff.

Sei $(P, \sqsubseteq)$ eine partielle Ordnung.

Dann heißt $(P, \sqsubseteq)$...

- *vollständig*, kurz *CPO* (von engl. complete partial order), wenn jede aufsteigende Kette $K \subseteq P$ eine kleinste obere Schranke in P besitzt.

Es gilt:

- Eine CPO $(C, \sqsubseteq)$ (genauer wäre: "kettenvollständige partielle Ordnung (engl. chain complete partial order (CCPO)") besitzt stets ein kleinstes Element, eindeutig bestimmt als Supremum der leeren Kette und üblicherweise mit $\perp$ bezeichnet: $\perp =_{df} \bigsqcup \emptyset$.

Ketten

Sei $(P, \sqsubseteq)$ eine partielle Ordnung.

Eine Teilmenge $K \subseteq P$ heißt...

- *Kette* in P, wenn die Elemente in K total geordnet sind. Für $K = \{k_0 \sqsubseteq k_1 \sqsubseteq k_2 \sqsubseteq \ldots\}$ $(\{k_0 \sqsupseteq k_1 \sqsupseteq k_2 \sqsupseteq \ldots\})$ spricht man auch genauer von einer *aufsteigenden* (*absteigenden*) Kette in P.

Eine Kette K heißt...

- *endlich*, wenn K endlich ist, sonst *unendlich*.

Kettenendlichkeit, endliche Elemente

Eine partielle Ordnung $(P, \sqsubseteq)$ heißt

- *kettenendlich* gdw. P enthält keine unendlichen Ketten

Ein Element $p \in P$ heißt

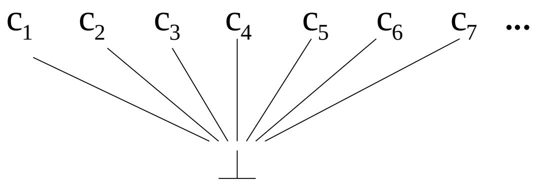
- *endlich* gdw. die Menge $Q =_{df} \{q \in P \mid q \sqsubseteq p\}$ keine unendliche Kette enthält
- *endlich relativ zu* $r \in P$ gdw. die Menge $Q =_{df} \{q \in P \mid r \sqsubseteq q \sqsubseteq p\}$ keine unendliche Kette enthält

(Standard-) CPO-Konstruktionen 1(4)

Flache CPOs...

Sei $(C, \sqsubseteq)$ eine CPO. Dann heißt $(C, \sqsubseteq)$...

- *flach*, wenn für alle $c, d \in C$ gilt: $c \sqsubseteq d \Leftrightarrow c = \bot \ \vee \ c = d$



(Standard-) CPO-Konstruktionen 2(4)

Produktkonstruktionen...

Seien $(P_1, \sqsubseteq_1), (P_2, \sqsubseteq_2), \ldots, (P_n, \sqsubseteq_n)$ CPOs. Dann sind auch...

- das *nichtstrikte* (*direkte*) *Produkt* $(\times P_i, \sqsubseteq)$ mit
  - $(\times P_i, \sqsubseteq) = (P_1 \times P_2 \times \ldots \times P_n, \sqsubseteq)$ mit $\forall (p_1, p_2, \ldots, p_n), (q_1, q_2, \ldots, q_n) \in \times P_i.\ (p_1, p_2, \ldots, p_n) \sqsubseteq (q_1, q_2, \ldots, q_n) \Rightarrow \forall i \in \{1, \ldots, n\}.\ p_i \sqsubseteq_i q_i$
- und das *strikte* (*direkte*) *Produkt* (*smash Produkt*) mit
  - $(\otimes P_i, \sqsubseteq) = (P_1 \otimes P_2 \otimes \ldots \otimes P_n, \sqsubseteq)$, wobei $\sqsubseteq$ wie oben definiert ist, jedoch zusätzlich gesetzt wird:
  $$(p_1, p_2, \ldots, p_n) = \bot \Rightarrow \exists i \in \{1, \ldots, n\}.\ p_i = \bot_i$$

  CPOs.

(Standard-) CPO-Konstruktionen 3(4)

Summenkonstruktion...

Seien $(P_1, \sqsubseteq_1), (P_2, \sqsubseteq_2), \ldots, (P_n, \sqsubseteq_n)$ CPOs. Dann ist auch...

- die *direkte Summe* $(\bigoplus P_i, \sqsubseteq)$ mit...
  - $(\bigoplus P_i, \sqsubseteq) = (P_1 \dot\cup P_2 \;\dot\cup\; \ldots \;\dot\cup\; P_n, \sqsubseteq)$ disjunkte Vereinigung der P_i, $i \in \{1, \ldots, n\}$ und $\forall\, p, q \in \bigoplus P_i.\ p \sqsubseteq q \Rightarrow \exists\, i \in \{1, \ldots, n\}.\ p, q \in P_i \;\wedge\; p \sqsubseteq_i q$ und der Identifikation der kleinsten Elemente der $(P_i, \sqsubseteq_i)$, $i \in \{1, \ldots, n\}$, d.h. $\bot =_{df} \bot_i$, $i \in \{1, \ldots, n\}$

  eine CPO.

(Standard-) CPO-Konstruktionen 4(4)

Funktionenraum...

Seien $(C, \sqsubseteq_C)$ und $(D, \sqsubseteq_D)$ zwei CPOs und $[C \to D] =_{df} \{f : C \to D \mid f \text{ stetig}\}$ die Menge der stetigen Funktionen von C nach D.

Dann ist auch...

- der *stetige Funktionenraum* $([C \to D], \sqsubseteq)$ eine CPO mit
  - $\forall\, f, g \in [C \to D].\ f \sqsubseteq g \Longleftrightarrow \forall\, c \in C.\ f(c) \sqsubseteq_D g(c)$

Funktionen auf CPOs / Eigenschaften

Seien $(C, \sqsubseteq_C)$ und $(D, \sqsubseteq_D)$ zwei CPOs und sei $f : C \to D$ eine Funktion von C nach D.

Dann heißt f...

- *monoton* gdw. $\forall\, c, c' \in C.\ c \sqsubseteq_C c' \Rightarrow f(c) \sqsubseteq_D f(c')$
  (*Erhalt der Ordnung der Elemente*)
- *stetig* gdw. $\forall\, C' \subseteq C.\ f(\bigsqcup_C C') =_D \bigsqcup_D f(C')$
  (*Erhalt der kleinsten oberen Schranken*)

Sei $(C, \sqsubseteq)$ eine CPO und sei $f : C \to C$ eine Funktion auf C.

Dann heißt f...

- *inflationär* (*vergrößernd*) gdw. $\forall\, c \in C.\ c \sqsubseteq f(c)$

Funktionen auf CPOs / Resultate

Mit den vorigen Bezeichnungen gilt...

Lemma
f ist monoton gdw. $\forall\, C' \subseteq C.\ f(\bigsqcup_C C') \sqsupseteq_D \bigsqcup_D f(C')$

Korollar
Eine stetige Funktion ist stets monoton, d.h. f stetig $\Rightarrow$ f monoton.

(Kleinste und größte) Fixpunkte 1(2)

Sei $(C, \sqsubseteq)$ eine CPO, $f : C \to C$ eine Funktion auf C und sei c ein Element von C, also $c \in C$.

Dann heißt c ...

- *Fixpunkt* von f gdw. $f(c) = c$

Ein Fixpunkt c von f heißt...

- *kleinster Fixpunkt* von f gdw. $\forall\, d \in C.\ f(d) = d \Rightarrow c \sqsubseteq d$
- *größter Fixpunkt* von f gdw. $\forall\, d \in C.\ f(d) = d \Rightarrow d \sqsubseteq c$

(Kleinste und größte) Fixpunkte 2(2)

Seien $d, c_d \in C$. Dann heißt c_d ...

- *bedingter kleinster Fixpunkt* von f bezüglich d gdw. c_d ist der kleinste Fixpunkt von C mit $d \sqsubseteq c_d$, d.h. für alle anderen Fixpunkte x von f mit $d \sqsubseteq x$ gilt: $c_d \sqsubseteq x$.

Bezeichnungen:
Der kleinste bzw. größte Fixpunkt einer Funktion f wird oft mit μf bzw. νf bezeichnet.

Fixpunktsatz

Theorem (Knaster/Tarski, Kleene)
Sei $(C, \sqsubseteq)$ eine CPO und sei $f : C \to C$ eine stetige Funktion auf C.

Dann hat f einen kleinsten Fixpunkt μf und dieser Fixpunkt ergibt sich als kleinste obere Schranke der Kette (sog. *Kleene-*Kette) $\{\perp, f(\perp), f^2(\perp), \ldots\}$, d.h.

$$\mu f = \bigsqcup_{i \in \mathbb{N}_0} f^i(\perp) = \bigsqcup \{\perp, f(\perp), f^2(\perp), \ldots\}$$

Beweis des Fixpunktsatzes 1(4)

Zu zeigen: μf...

1. existiert
2. ist Fixpunkt
3. ist kleinster Fixpunkt

Beweis des Fixpunktsatzes 2(4)

1. *Existenz*

   - Es gilt $f^0 \perp = \perp$ und $\perp \sqsubseteq c$ für alle $c \in C$.
   - Durch vollständige Induktion lässt sich damit zeigen: $f^n \perp \sqsubseteq f^n c$ für alle $c \in C$.
   - Somit gilt $f^n \perp \sqsubseteq f^m \perp$ für alle n, m mit $n \leq m$. Somit ist $\{f^n \perp \mid n \geq 0\}$ eine (nichtleere) Kette in C.
   - Damit folgt die Existenz von $\bigsqcup_{i \in \mathbb{N}_0} f^i(\perp)$ aus der CPO-Eigenschaft von $(C, \sqsubseteq)$.

Beweis des Fixpunktsatzes 3(4)

2. *Fixpunkteigenschaft*

$$
\begin{array}{rcl}
 & & f(\bigsqcup_{i \in \mathbb{N}_0} f^i(\perp)) \\
(f \text{ stetig}) & = & \bigsqcup_{i \in \mathbb{N}_0} f(f^n \perp) \\
 & = & \bigsqcup_{i \in \mathbb{N}_1} f^n \perp \\
(K \text{ Kette} \Rightarrow \bigsqcup K = \perp \sqcup \bigsqcup K) & = & \bigsqcup_{i \in \mathbb{N}_1} f^n \perp \ \sqcup \ \perp \\
(f^0 = \perp) & = & \bigsqcup_{i \in \mathbb{N}_0} f^n \perp \\
 & = & \bigsqcup_{i \in \mathbb{N}_0} f^i(\perp)
\end{array}
$$

Beweis des Fixpunktsatzes 4(4)

3. *Kleinster Fixpunkt*

   - Sei c beliebig gewählter Fixpunkt von f. Dann gilt $\perp \sqsubseteq c$ und somit auch $f^n \perp \sqsubseteq f^n c$ für alle $n \geq 0$.
   - Folglich gilt $f^n \perp \sqsubseteq c$ wg. der Wahl von c als Fixpunkt von f.
   - Somit gilt auch, dass c eine obere Schranke von $\{f^i(\perp) \mid i \in \mathbb{N}_0\}$ ist.
   - Da $\bigsqcup_{i \in \mathbb{N}_0} f^i(\perp)$ nach Definition die kleinste obere Schranke dieser Kette ist, gilt wie gewünscht $\bigsqcup_{i \in \mathbb{N}_0} f^i(\perp) \sqsubseteq c$.

Bedingte Fixpunkte

Theorem (Endliche Fixpunkte)
Sei $(C, \sqsubseteq)$ eine CPO, sei $f : C \to C$ eine stetige, inflationäre Funktion auf C und sei $d \in C$.

Dann hat f einen kleinsten bedingten Fixpunkt μf_d und dieser Fixpunkt ergibt sich als kleinste obere Schranke der Kette $\{d, f(d), f^2(d), \ldots\}$, d.h.

$$\mu f_d = \bigsqcup_{i \in \mathbb{N}_0} f^i(d) = \bigsqcup \{d, f(d), f^2(d), \ldots\}$$

Endliche Fixpunkte

Theorem (Endliche Fixpunkte)
Sei $(C, \sqsubseteq)$ eine CPO und sei $f : C \to C$ eine stetige Funktion auf C.

Dann gilt: Sind in der Kleene-Kette von f zwei aufeinanderfolgende Glieder gleich, etwa $f^i(\bot) = f^{i+1}(\bot)$, so gilt $\mu f = f^i(\bot)$.

Existenz endlicher Fixpunkte

Hinreichende Bedingungen für die Existenz endlicher Fixpunkte sind...

- Endlichkeit von Definitions- und Wertebereich von f
- f ist von der Form $f(c) = c \sqcup g(c)$ für monotones g über kettenendlichem Wertebereich

Vorschau auf die nächsten Vorlesungstermine...

- Di, 05.12.2006, Vorlesung von 17:45 Uhr bis 19:15 Uhr, Bibliothek E185/1
- Di, 12.12.2006, Vorlesung von 17:45 Uhr bis 19:15 Uhr, Bibliothek E185/1
- *Di, 19.12.2006: Keine Vorlesung!* (*Ferialzeit*)
- *Di, 26.12.2006: Keine Vorlesung!* (*Ferialzeit*)

Vorschau auf die weiteren Vorlesungstermine...

- *Di, 02.01.2007: Keine Vorlesung!* (*Ferialzeit*)
- Di, 09.01.2007, Vorlesung von 17:45 Uhr bis 19:15 Uhr, Bibliothek E185/1
- Di, 16.01.2007, Vorlesung von 17:45 Uhr bis 19:15 Uhr, Bibliothek E185/1
- Di, 23.01.2007, Vorlesung von 17:45 Uhr bis 19:15 Uhr, Bibliothek E185/1
- Di, 30.01.2007, Vorlesung von 17:45 Uhr bis 19:15 Uhr, Bibliothek E185/1