Verteilte ASM: Concurrency, reactivity, time

Verteilte ASM (DASM) Berechnungsmodell:

Asynchrones Berechnungsmodell
Autonom operierende Agenten
• Eine endliche Menge autonomer ASM-Agenten, jeder mit eigenem Programm.
• Agenten interagieren durch lesen und schreiben von gemeinsamen Lokationen globaler Maschinenzustände.
• Potentielle Konflikte werden durch das zugrundeliegende semantische Modell aufgelöst, nach Maßgabe der Definition partiell geordneter Läufe.

Grundlagen:Ordnungen, CPO's, Beweistechniken

Eigenschaften binärer Relationen

- X Menge
- $\rho \subseteq X \times X$ binäre Relation
- Eigenschaften

(P1)	$x \rho x$	(reflexiv)
(P2)	$(x \rho y \wedge y \rho x) \rightarrow x = y$	(antisymmetrisch)
(P3)	$(x \rho y \wedge y \rho z) \rightarrow x \rho z$	(transitiv)
(P4)	$(x \rho y \vee y \rho x)$	(linear)

Quasi-Ordnungen

- $\lesssim \subseteq X \times X$ Quasi-Ordnung gdw $\lesssim$ reflexiv und transitiv.
- Kern:
$$\approx = \lesssim \cap \lesssim^{-1}$$
- Strikter Anteil : $< = \lesssim \setminus \approx$
- $Y \subseteq X$ links geschlossen (bzgl. $\lesssim$) gdw
$$(\forall y \in Y : (\forall x \in X : x \lesssim y \rightarrow x \in Y))$$
- Schreibweise: Quasi-Ordnung $(X, \lesssim)$

Partial-Ordnungen

- $\leq \subseteq X \times X$ Partial-Ordnung gdw $\leq$ reflexiv, antisymmetrisch und transitiv.
- Kern: Es gilt
$$\mathrm{id}_X = \leq \cap \leq^{-1}$$
- Strikter Anteil: $< = \leq \setminus \mathrm{id}_X$
- Oft: $<$ Partial-Ordnung gdw $<$ irreflexiv, transitiv.
- Schreibweise: Partial-Ordnung $(X, \leq)$

Wohlfundierte Ordnungen

- Partial-Ordnung $\leq \subseteq X \times X$ wohlfundiert gdw

$$(\forall Y \subseteq X : Y \neq \emptyset \rightarrow (\exists y \in Y : y \text{ minimal in } Y \text{ bzgl. } \leq))$$

- Quasi-Ordnung $\lesssim$ wohlfundiert gdw strikter Anteil von $\lesssim$ wohlfundiert.
- Initiales Segment: $Y \subseteq X$, links-geschlossen
- Initiale Sektion von x: $\text{sec}(x) = \{y : y < x\}$

Supremum

- Sei $(X, \leq)$ Partial-Ordnung und $Y \subseteq X$
- $S \subseteq X$ Kette gdw Elemente von S werden linear geordnet durch $\leq$.
- y obere Schranke von Y gdw

$$\forall y' \in Y : y' \leq y$$

- Supremum: y Supremum von Y gdw

$$\forall y' \in X : ((y' \text{ obere Schranke von } Y) \rightarrow y \leq y')$$

- Analog: untere Schranke, Infimum $\inf(Y)$

CPO

- Partial-Ordnung $(D, \sqsubseteq)$ complete partial ordering (CPO) gdw
  - $\exists$ kleinstes Element $\perp$ von D (bzgl. $\sqsubseteq$)
  - Jede Kette S hat ein Supremum $\sup(S)$.

Beispiel

Beispiel 4.1

- $(\mathcal{P}(X), \subseteq)$ *ist CPO.*
- $(D, \sqsubseteq)$ *ist CPO mit*
  - $D = X \nrightarrow Y$*: Menge aller partieller Funktionen* f *mit* $\text{dom}(f) \subseteq X$ *und* $\text{cod}(f) \subseteq Y$.
  - *Sei* $f, g \in X \nrightarrow Y$.

$$f \sqsubseteq g \text{ gdw } \text{dom}(f) \subseteq \text{dom}(g) \wedge (\forall x \in \text{dom}(f) : f(x) = g(x))$$

monoton, stetig

- $(D, \sqsubseteq)$, $(E, \sqsubseteq')$ CPOs
- $f : D \to E$ monoton gdw

$$(\forall d, d' \in D : d \sqsubseteq d' \to f(d) \sqsubseteq' f(d'))$$

- $f : D \to E$ stetig gdw f monoton und

$$(\forall S \subseteq D : S \text{ Kette } \to f(\sup(S)) = \sup(f(S)))$$

- $X \subseteq D$ zulässig gdw

$$(\forall S \subseteq X : S \text{ Kette } \to \sup(S) \in X)$$

Fixpunkt

- $(D, \sqsubseteq)$ CPO, $f : D \to D$
- $d \in D$ Fixpunkt von f gdw

$$f(d) = d$$

- $d \in D$ kleinster Fixpunkt von f gdw d Fixpunkt von f und

$$(\forall d' \in D : d' \text{ Fixpunkt } \to d \sqsubseteq d')$$

Fixpunkt-Satz

Fixpunkt-Satz: $(D, \sqsubseteq)$ CPO, $f : D \to D$ stetig, dann hat f einen kleinsten Fixpunkt μf. Es gilt

$$\mu f = \sup\{f^i(\bot) : i \in \mathbb{N}\}$$

Beweis: (Skizze)

- $\sup\{f^i(\bot) : i \in \mathbb{N}\}$ Fixpunkt:

$$\begin{array}{rcl} f(\sup\{f^i(\bot) : i \in \mathbb{N}\}) & = & \sup\{f^{i+1}(\bot) : i \in \mathbb{N}\} \\ & & \text{(stetig)} \\ & = & \sup\{\sup\{f^{i+1}(\bot) : i \in \mathbb{N}\}, \bot\} \\ & = & \sup\{f^i(\bot) : i \in \mathbb{N}\} \end{array}$$

Fixpunkt-Satz

Fixpunkt-Satz: $(D, \sqsubseteq)$ CPO, $f : D \to D$ stetig, dann hat f einen kleinsten Fixpunkt μf. Es gilt

$$\mu f = \sup\{f^i(\bot) : i \in \mathbb{N}\}$$

Beweis: (Fortsetzung)

- $\sup\{f^i(\bot) : i \in \mathbb{N}\}$ kleinster Fixpunkt:
  1. d' Fixpunkt von f
  2. $\bot \sqsubseteq d'$
  3. f monoton, d' FP: $f(\bot) \sqsubseteq f(d') = d'$
  4. Induktion: $\forall i \in \mathbb{N} : f^i(\bot) \sqsubseteq f^i(d') = d'$
  5. $\sup\{f^i(\bot) : i \in \mathbb{N}\} \sqsubseteq d'$

Induktion über $\mathbb{N}$

Induktionsprinzip:

$$(\forall X \subseteq \mathbb{N} : ((0 \in X \wedge (\forall x \in X : x \in X \rightarrow x + 1 \in X))) \rightarrow X = \mathbb{N})$$

Korrektheit:

1. Ann. nein, also $\exists X \subseteq \mathbb{N} : \mathbb{N} \setminus X \neq \emptyset$
2. Sei y minimal in $\mathbb{N} \setminus X$ (bzgl. $<$).
3. $y \neq 0$
4. $y - 1 \in X \wedge y \notin X$
5. Widerspruch

Induktion über $\mathbb{N}$ (Alternative)

Induktionsprinzip:

$$(\forall X \subseteq \mathbb{N} : (\forall x \in \mathbb{N} : \mathrm{sec}(x) \subseteq X \rightarrow x \in X) \rightarrow X = \mathbb{N})$$

Korrektheit:

1. Ann. nein, also $\exists X \subseteq \mathbb{N} : \mathbb{N} \setminus X \neq \emptyset$
2. Sei y minimal in $\mathbb{N} \setminus X$ (bzgl. $<$).
3. $\mathrm{sec}(y) \subseteq X, y \notin X$
4. Widerspruch

Wohlfundierte Induktion

Induktionsprinzip: Sei $(Z, \leq)$ wohlfundierte Partial-Ordnung.

$$(\forall X \subseteq Z : (\forall x \in Z : \mathrm{sec}(x) \subseteq X \rightarrow x \in X) \rightarrow X = Z)$$

Korrektheit:

1. Ann. nein, also $Z \setminus X \neq \emptyset$
2. Sei z minimal in $Z \setminus X$ (bzgl. $\leq$).
3. $\mathrm{sec}(z) \subseteq X, z \notin X$
4. Widerspruch

FP-Induktion

Induktionsprinzip: Sei $(D, \sqsubseteq)$ CPO, $f : D \rightarrow D$ stetig.

$$(\forall X \subseteq D, \text{zulässig} : (\perp \in X \wedge (\forall y : y \in X \rightarrow f(y) \in X)) \rightarrow \mu f \in X)$$

Korrektheit: Sei $X \subseteq D$ zulässig.

$$\begin{array}{lll} \mu f \in X & \Leftrightarrow & \sup\{f^i(\perp) : i \in \mathbb{N}\} \in X \qquad \text{(FP-Satz)} \\ & \Leftarrow & \forall i \in \mathbb{N} : f^i(\perp) \in X \qquad (X \text{ zulässig}) \\ & \Leftarrow & \perp \in X \wedge (\forall n \in \mathbb{N} : f^n(\perp) \in X \rightarrow f(f^n(\perp)) \in X) \\ & & \qquad \text{(Induktion } \mathbb{N}) \\ & \Leftarrow & \perp \in X \wedge (\forall y \in X \rightarrow f(y) \in X) \qquad \text{(Gen.)} \end{array}$$

Aufgabe

Übung 4.1 *Sei* $(D, \sqsubseteq)$ *CPO mit*

- $X = Y = \mathbb{N}$
- $D = X \rightarrow\!\!\!\!\!\!\cdot\;\; Y$*: Menge aller partieller Funktionen* f *mit* $\mathrm{dom}(f) \subseteq X$ *und* $\mathrm{cod}(f) \subseteq Y$*.*
- *Sei* $f, g \in X \rightarrow\!\!\!\!\!\!\cdot\;\; Y$*.*

$$f \sqsubseteq g \text{ gdw } \mathrm{dom}(f) \subseteq \mathrm{dom}(g) \wedge (\forall x \in \mathrm{dom}(f) : f(x) = g(x))$$

Betrachte

$$F : \begin{array}{lcl} D & \rightarrow & \mathbb{N} \times \mathbb{N} \\ g & \mapsto & \begin{cases} \{(0,1)\} & g = \emptyset \\ \{(x, x \cdot g(x-1)) : x - 1 \in \mathrm{dom}(g)\} & \text{sonst} \end{cases} \end{array}$$

Aufgabe

Zeigen Sie:

1. $\forall g \in D : F(g) \in D$, d.h. $F : D \rightarrow D$
2. $F : D \rightarrow D$ stetig
3. $\forall n \in \mathbb{N} : \mu F(n) = n!$

Bemerkung:

- μF kann aufgefasst werden als Semantik einer Funktionsdefinition

$$\begin{array}{l} \text{function Fac}(n : \mathbb{N}_\perp) : \mathbb{N}_\perp =_{\text{def}} \\ \quad \text{if } n = 0 \text{ then } 1 \\ \quad \text{else } n \cdot \text{Fac}(n-1) \end{array}$$

- Stichwort: 'derived functions' in ASM

Aufgabe

Übung 4.2 *Zeigen Sie: Sei* $G = (V, E)$ *ein unendlicher gerichteter Graph mit*

- *G besitzt endlich viele Wurzeln (Knoten, ohne eingehende Kanten).*
- *Jeder Knoten besitzt endlichen Aus-Grad.*
- *Jeder Knoten ist erreichbar von einer Wurzel.*

Dann existiert ein unendlicher Pfad, der bei einer Wurzel beginnt.

Verteilte ASM

Definition 4.1 *Eine DASM A über eine Signatur (Vokabular)* Σ *ist gegeben durch:*

- *Ein verteiltes Programm* Π_A *über* Σ*.*
- *Eine nicht-leere Menge* I_A *initialer Zustände*
  Ein initialer Zustand legt eine mögliche Interpretation von Σ *über eine potentiell unendlichen Basismenge* X *fest.*

A enthält in seiner Signatur ein dynamisches Relationssymbol AGENT, die als endliche Menge autonom operierender Agenten interpretiert wird.

- *Das Verhalten eines Agenten a in Zustand S von A ist durch* $program_S(a)$ *festgelegt.*
- *Ein Agent kann terminiert werden durch die Festlegung von* $program_S(a) := undef$ *(Darstellung eines ungültigen Programms).*

Partiell geordnete Läufe

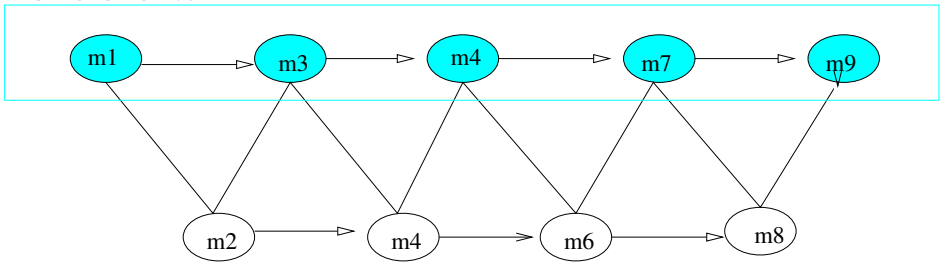
Ein Lauf einer verteilten ASM A ist durch ein Tripel $\varrho \rightleftharpoons (M, \lambda, \sigma)$ mit folgenden Eigenschaften gegeben:

- 1. M ist eine partiell geordnete Menge von Zügen (moves), wobei jeder Zug nur endlich viele Vorgänger hat.
- 2. λ ist eine Funktion auf M, die jedem Zug einen Agenten zuordnet, so dass die Züge eines einzelnen Agenten stets linear geordnet sind.
- 3. σ assoziiert einen Zustand von A mit jedem endlichen initialen Segment Y von M, hierbei ist $\sigma(Y)$ das "Ergebnis der Durchführung aller Züge" in Y. $\sigma(Y)$ ist ein initialer Zustand falls Y leer ist.
- 4. Die Kohärenz Bedingung ist erfüllt:
  Ist *max* eine Menge maximaler Elemente in einem endlichen initialen Segment X von M und $Y = X \setminus max$, dann ist $\lambda(x)$ ein Agent in $\sigma(Y)$ für $x \in max$ und man erhält $\sigma(X)$ aus $\sigma(Y)$ durch Feuern von $\{\lambda(x) : x \in max\}$ (ihre Programme) in $\sigma(Y)$.

Bemerkung, Beispiel

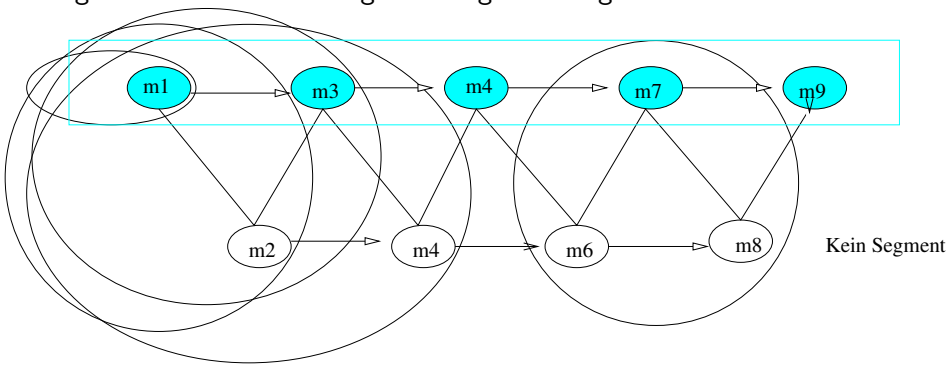
Die Agenten von A modellieren die konkurrierenden Kontrollthreads in der Ausführung von Π_A.
Ein Lauf kann als der gemeinsame Anteil der Historie der gleichen Berechnung aus der Sicht mehrerer Beobachter.

Die Rolle von λ:



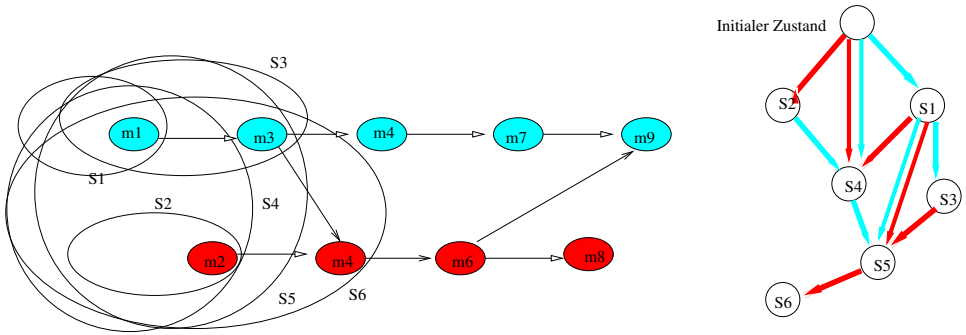
Bemerkung, Beispiel

Die Rolle von σ: Momentaufnahmen sind die initialen Segmente der partiell geordneten Menge M. Jedem initialen Segment wir ein Zustand von A (Interpretation von Σ) zugeordnet, der die Wirkung der Programme der im Segment vorkommenden Agenten wiedergibt.
$\rightsquigarrow$"Ergebnis der Durchführung aller Züge" im Segment.



Kohärenz Bedingung, Beispiel

Ist *max* eine Menge maximaler Elemente in einem endlichen initialen Segment X von M und $Y = X \setminus max$, dann ist $\lambda(x)$ ein Agent in $\sigma(Y)$ für $x \in max$ und man erhält $\sigma(X)$ aus $\sigma(Y)$ durch Feuern von $\{\lambda(x) : x \in max\}$ (ihre Programme) in $\sigma(Y)$.



Folgerungen aus der Kohärenz Bedingung

Lemma 4.1 *Alle Linearisierungen eines initialen Segments eines Laufes ϱ haben den selben Endzustand.*

Lemma 4.2 *Eine Eigenschaft P gilt genau dann in allen erreichbaren Zuständen eines Laufes ϱ, wenn sie in jedem erreichbaren Zustand von jeder Linearisierung von ϱ gilt.*

Einfaches Beispiel

Beispiel 4.2 *Seien $\{door, window\}$ aussagenlogische Konstanten in der Signatur mit natürlicher Bedeutung:*
door = true bedeute Tür offen und analog für Fenster.

Das Programm bestehe aus zwei Agenten, einen Tür-Manager d und einen Fenster-Manager w mit Programmen:

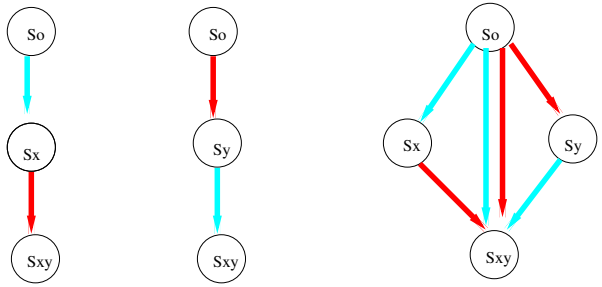
$program_d = door := true$ // move x
$program_w = window := true$ // move y

Im initialen Zustand S_0 seien Tür und Fenster geschlossen, d und w seien in der Agentenmenge.

Welche sind die möglichen Läufe?

Einfaches Beispiel (Fort.)

Seien $\varrho_1 = (\{x, y\}, x < y), id, \sigma), \varrho_2 = (\{x, y\}, y < x), id, \sigma)$,
$\varrho_3 = (\{x, y\}, <>), id, \sigma)$ (Gröbste Partialordnung)

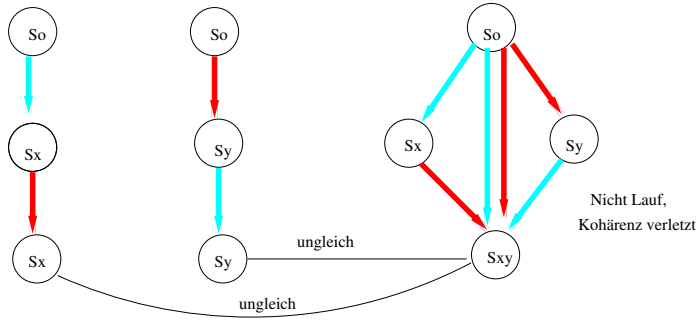


Variante von Einfaches Beispiel

Das Programm bestehe aus zwei Agenten, einen Tür-Manager d und einen Fenster-Manager w mit Programmen:

$program_d = if \neg window\ then\ door := true$ // move x
$program_w = if \neg door\ then\ window := true$ // move y

Im initialen Zustand S_0 seien Tür und Fenster geschlossen, d und w seien in der Agentenmenge. Wie sehen nun Läufe aus? Gleiche ϱ's



Mehr Variationen

Übung 4.3 *Betrachte folgende Agentenpaare*
$x, y \in \mathbb{N}$ $(x = 2, y = 1$ *im Anfangszustand)*

1. $a = x := x + 1$ *und* $b = x := x + 1$
2. $a = x := x + 1$ *und* $b = x := x - 1$
3. $a = x := y$ *und* $b = y := x$

Welche Läufe sind mit zweielementigen partiell geordneten Mengen möglich?
Versuche alle Läufe zu charakterisieren.

Mehr Variationen

Betrachte folgende Agenten mit üblicher Interpretation:

1. $Program_d =$ *if* $\neg window$ *then* $door := true$ //move x
2. $Program_w =$ *if* $\neg door$ *then* $window := true$ //move y
3. $Program_l =$ *if* $\neg light \wedge (\neg door \vee \neg window)$ *then* //move z
   $light := true$
   $door := false$
   $window := false$

Welche Endzustände sind möglich, wenn im Anfangszustand die drei Konstanten false sind?

Noch zum Üben

Consumer-producer problem: Assume a single producer agent and two or more consumer agents operating concurrently on a global shared structure. This data structure is linearly organized and the producer adds items at the one end side while the consumers can remove items at the opposite end of the data structure. For manipulating the data structure, assume operations *insert* and *remove* as introduced below.

$insert : \quad Item \times \quad ItemList \rightarrow \quad ItemList$
$remove : \quad ItemList \rightarrow \quad (Item \times \quad ItemList)$

(1) Which kind of potential conflicts do you see?
(2) How does the semantic model of partially ordered runs resolve such conflicts?

Umgebung

Reaktive Systeme zeichnen sich durch ihre Interaktion mit der Umgebung aus. Modelliert kann dies mit Hilfe eines Umgebungsagenten. Läufe können nun (mit λ) diesen Agenten enthalten, λ muss dann auch die Updatemenge der Umgebung in den entsprechenden Zug festlegen.
Die Kohärenzbedingung muss dann auch für solche Läufe gelten.

Für extern kontrollierte Funktionen führt dies sicherlich nicht zu Inkonsistenzen bei den Updatemengen, das Verhalten der internen Agenten kann natürlich beeinflusst werden. Bei shared Funktionen können bei der simultanen Ausführung der Züge eines internen Agenten und der Umgebung inkonsistente Updatemengen entstehen.

Oft werden noch Einschränkungen (Annahmen) der Umgebung gemacht. Hier gibt es viele Möglichkeiten: Umgebung wird nur beobachtet oder Umgebung erfüllt Integritätsbedingungen.

Zeit

Die Beschreibung von Real Time Verhalten muss explizit Zeitaspekte berücksichtigen. Dies kann mit Hilfe von Timer (siehe SDL), globale Systemzeit oder lokale Systemzeiten erfolgen.

- Reaktionen können instantan sein (das Feuern der Regeln der Agenten erfordert keine Zeit)
- Aktionen benötigen Zeit

Bei der globalen Zeitbetrachtung geht man davon aus, dass eine linear geordnete Domäne *TIME* vorliegt, etwa mit Deklarationen

$$\mathit{domain}\ \ (\mathit{TIME}, \leq),\quad (\mathit{TIME}, \leq)\ \subset (\mathbb{R}, \leq)$$

Hierbei wird die Zeit von einer diskreten Systemuhr gemessen:

$$\mathit{monitored}\quad \mathit{now} :\rightarrow \mathit{TIME}$$

Geldautomat

Beispiel 4.3 *Abstrakte Modellierung eines Geldautomaten: Drei Agenten sind im Modell: ATM Manager, Autenti*