

Wissensrepräsentation

Christopher Habel, Carola Eschenbach
Universität Hamburg, FB Informatik
AB Wissens- und Sprachverarbeitung (WSV)

Sommersemester 2003

Wissensrepräsentation

Christopher Habel, Carola Eschenbach
Sommersemester 2003

Sitzung 26: Aktionen und Instruktionen

- Situationskalkül
- Spezifikationen von Aktionen
- Erweiterung: Golog: Programmiersprache für Agenten
- Instruktionsbasiertes Handeln

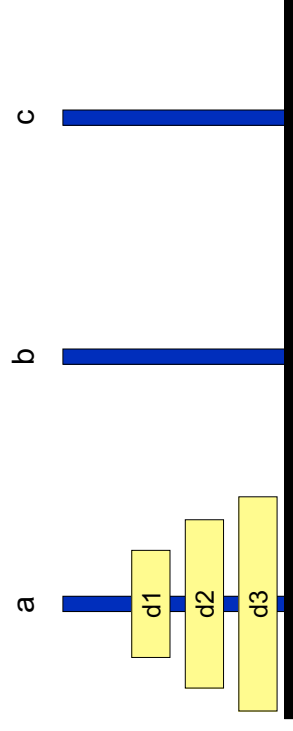
Literatur

Levesque, Hector J., Raymond Reiter, Yves Lespérance, Fangzhen Lin & Richard B. Scherl (1997). GOLOG: A logic programming language for dynamic domains. *Journal of Logic Programming* 31. 59–84.

Brachman, Ronald J. & Hector J. Levesque (to appear). *Knowledge Representation and Reasoning*. Chapter 14.

Reiter, Raymond (1991). The frame problem in the situation calculus: A simple solution (sometimes) and a completeness result for goal regression. In V. Lifschitz (ed.) *Artificial Intelligence and Mathematical Theory of Computation* (pp. 359–380). Academic Press: Boston.

Aktionen für den Turm von Hanoi



Welche Axiome des Nachfolgezustands gelten für die Fluents?

free

Effektaxiome

- $$\forall x, y, D, \text{Sit} [\neg \exists E [\text{on}(D, E, \text{Sit})] \wedge y \neq x \Rightarrow \text{free}(x, \text{do}(\text{move}(x, y, D), \text{Sit}))]$$
- $$\forall y, x, D, \text{Sit} [\neg \text{free}(x, \text{do}(\text{move}(y, x, D), \text{Sit}))]$$

Axiom des Nachfolgezustands

- $$\forall x, \text{Sit}, a [\text{free}(x, \text{do}(a, \text{Sit})) \Leftrightarrow \exists y, D [(a = \text{move}(x, y, D) \wedge \neg \exists E [\text{on}(D, E, \text{Sit})] \wedge y \neq x)] \vee (\text{free}(x, \text{Sit}) \wedge \neg \exists y, D [a = \text{move}(y, x, D)])]$$

on

Effektaxiome

- $$\forall x, y, D, E, \text{Sit} [\text{ontop}(E, y, \text{Sit}) \wedge y \neq x \Rightarrow \text{on}(D, E, \text{do}(\text{move}(x, y, D), \text{Sit}))]$$
- $$\forall x, y, D, E, \text{Sit} [\text{on}(D, E, \text{Sit}) \wedge y \neq x \Rightarrow \neg \text{on}(D, E, \text{do}(\text{move}(x, y, D), \text{Sit}))]$$

Axiom des Nachfolgezustands

- $$\forall D, E, \text{Sit} [\text{on}(D, E, \text{do}(a, \text{Sit})) \Leftrightarrow \exists x, y [a = \text{move}(x, y, D) \wedge \text{ontop}(E, y, \text{Sit}) \wedge y \neq x] \vee \text{on}(D, E, \text{Sit}) \wedge \neg \exists x, y [a = \text{move}(x, y, D) \wedge \text{on}(D, E, \text{Sit}) \wedge y \neq x]]$$

free

Effektaxiome

- $$\forall x, y, D, \text{Sit} [\neg \exists E [\text{on}(D, E, \text{Sit})] \wedge y \neq x \Rightarrow \text{free}(x, \text{do}(\text{move}(x, y, D), \text{Sit}))]$$
- $$\forall y, x, D, \text{Sit} [\neg \text{free}(x, \text{do}(\text{move}(y, x, D), \text{Sit}))]$$

Axiom des Nachfolgezustands

- $$\forall x, \text{Sit}, a [\text{free}(x, \text{do}(a, \text{Sit})) \Leftrightarrow \exists y, D [(a = \text{move}(x, y, D) \wedge \neg \exists E [\text{on}(D, E, \text{Sit})] \wedge y \neq x)] \vee (\text{free}(x, \text{Sit}) \wedge \neg \exists y, D [a = \text{move}(y, x, D)])]$$

ontop

Effektaxiome

- $$\forall y, x, D, \text{Sit} [\text{ontop}(D, x, \text{do}(\text{move}(y, x, D), \text{Sit}))]$$
- $$\forall x, y, D, E, \text{Sit} [\text{on}(E, D, \text{Sit}) \wedge y \neq x \Rightarrow \text{ontop}(D, x, \text{do}(\text{move}(x, y, E), \text{Sit}))]$$
- $$\forall x, y, D, \text{Sit} [y \neq x \Rightarrow \neg \text{ontop}(D, x, \text{do}(\text{move}(x, y, D), \text{Sit}))]$$
- $$\forall y, x, D, E, \text{Sit} [\text{ontop}(D, x, \text{Sit}) \wedge x \neq y \Rightarrow \neg \text{ontop}(D, x, \text{do}(\text{move}(y, x, E), \text{Sit}))]$$

Axiom des Nachfolgezustands

- $$\forall x, D, \text{Sit}, a [\text{ontop}(D, x, \text{do}(a, \text{Sit})) \Leftrightarrow \exists y [a = \text{move}(y, x, D)] \vee \exists y, E [a = \text{move}(x, y, E) \wedge \text{on}(E, D, \text{Sit}) \wedge y \neq x] \vee \text{ontop}(D, x, \text{Sit}) \wedge \neg \exists y [a = \text{move}(x, y, D) \wedge y \neq x] \wedge \neg \exists y, E [a = \text{move}(y, x, E) \wedge \text{ontop}(D, x, \text{Sit}) \wedge x \neq y]]$$

Planung im Situationskalkül

Gegeben

- Zielspezifikation: Eine Formel $G(s)$ mit s als einziger freien Variable
- Startsituation: eine Situation s_0 (mit vollständiger Spezifikation)

Gesucht

- Eine Folge $\langle a_1, \dots, a_n \rangle$ von primitiven Aktionen, so dass
- $\mathcal{KB} \models G(\text{do}(\langle a_1, \dots, a_n \rangle, s_0)) \wedge \text{Legal}(\langle a_1, \dots, a_n \rangle, s_0)$

Verfahren

- KI-Suchverfahren (Bei Bewertungsmöglichkeit: A^*)
- (Resolutions-)Beweis mit Antwortextraktion
- Goal-Regression (Reiter 1991)

Goal-Regression

Ziel

- Bestimme eine Folge $\langle a_1, \dots, a_n \rangle$ von primitiven Aktionen, so dass
- $\mathcal{KB} \models G(\text{do}(\langle a_1, \dots, a_n \rangle, s_0)) \wedge \text{Legal}(\langle a_1, \dots, a_n \rangle, s_0)$

Vorgehen

- Systematische Ersetzung von Teil-Ausdrücken der Form $\text{do}(\langle a, s \rangle)$ in den auftretenden Fluents
 - unter Verwendung der Axiome des Nachfolgezustands und der Vorbedingungsaxiome,
 - so dass $G^*(s_0)$ nur s_0 als Situationsausdruck enthält
 - und obige Folgerungsbeziehung genau dann besteht, wenn
 - $\mathcal{KB} \setminus (\text{Pre} \cup \text{SSt}) \models G^*(s_0)$
- Erlaubt auch die Bestimmung von Vorbedingungen von Aktionsfolgen

Ch. Habel / C. Eschenbach: Wissensrepräsentation, SoSe 2003

26 – 9

Aufzugs-Planung im Situationskalkül

Gegeben

- Ziel: Alle Anforderungen erfüllt, Parken im Erdgeschoss mit offener Tür.
- $\neg \exists n [\text{on}(n, s)] \wedge \text{currentFloor}(0, s) \wedge \neg \text{d_closed}(s)$
- Startsituation s_0

Gesucht

- Eine Folge $\langle a_1, \dots, a_n \rangle$ von primitiven Aktionen, so dass
- $\mathcal{KB} \models G(\text{do}(\langle a_1, \dots, a_n \rangle, s_0)) \wedge \text{Legal}(\langle a_1, \dots, a_n \rangle, s_0)$

Lösung

$\langle \text{close, down}(3), \text{open, turnoff}(3), \text{close, up}(5), \text{open, turnoff}(5), \text{close, down}(0), \text{open} \rangle$
 $s = \text{do}(\text{open, do}(\text{down}(0), \text{do}(\text{close, do}(\text{turnoff}(5), \text{do}(\text{open, do}(\text{up}(5), \text{do}(\text{close, do}(\text{turnoff}(3), \text{do}(\text{open, do}(\text{down}(3), \text{do}(\text{close, } s_0))))))))))$

Ch. Habel / C. Eschenbach: Wissensrepräsentation, SoSe 2003

26 – 10

Aufzugsbeispiel: Mögliche Aktionen

	cFl	on	d_cl	Poss(a)
s0	4	3, 5	f	close
s1	do(close, s0)	4	3, 5	t open, down(0)...(3), up(5)
s2	do(down(3), s1)	3	3, 5	t open, down(0)...(2), up(4)...(5)
s3	do(open, s2)	3	3, 5	f turnoff, close
s4	do(turnoff(3), s3)	3	5	f close
s5	do(close, s4)	3	5	t open, down(0)...(2), up(4)...(5)
s6	do(up(5), s5)	5	5	t open, down(0)...(4)
s7	do(open, s6)	5	5	f turnoff, close
s8	do(turnoff(5), s7)	5	f	f close
s9	do(close, s8)	5	t	t open, down(0)...(4)
s10	do(down(0), s9)	0	t	t open, up(1)...(5)
s11	do(open, s10)	0	f	f close

Ch. Habel / C. Eschenbach: Wissensrepräsentation, SoSe 2003

26 – 11

Hanoi-Planung im Situationskalkül

Gegeben

- Ziel: Alle Scheiben auf Stab b
- $\text{free}(a) \wedge \text{free}(c)$
- Startsituation s_0 : alle Scheiben auf Stab a

Gesucht

- Eine Folge $\langle a_1, \dots, a_n \rangle$ von primitiven Aktionen, so dass
- $\mathcal{KB} \models G(\text{do}(\langle a_1, \dots, a_n \rangle, s_0)) \wedge \text{Legal}(\langle a_1, \dots, a_n \rangle, s_0)$

Lösung

$\langle \text{move}(a, b, d1), \text{move}(a, c, d2), \text{move}(b, c, d1), \text{move}(a, b, d3), \text{move}(c, a, d1), \text{move}(c, b, d2), \text{move}(a, b, d1) \rangle$
 $s = \text{do}(\text{move}(a, b, d1), \text{do}(\text{move}(c, b, d2), \text{do}(\text{move}(c, a, d1), \text{do}(\text{move}(a, b, d3), \text{do}(\text{move}(b, c, d1), \text{do}(\text{move}(a, c, d2), \text{do}(\text{move}(a, b, d1), s_0))))))$

Ch. Habel / C. Eschenbach: Wissensrepräsentation, SoSe 2003

26 – 12

Instruktionen für einen Agenten

Als Ziel

- Spezifikation des durch den Agenten zu erreichenden Zustandes

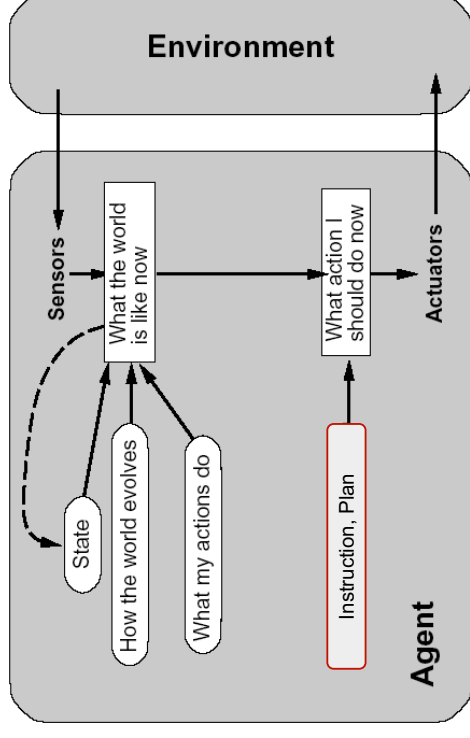
Als Aktionssequenz

- Resultat der Planung
- durch den Agenten in dieser Form auszuführen

Als komplexe Aktion

- mit Kontrollstruktur
 - z.B. bedingten Aktionen
- Agenten-Programm
 - GOLOG

Model-based reflex agent (with instructions)



Komplexe Aktionen

Sequenz

- Fahre in die dritte Etage und öffne dann die Tür.

Bedingte Aktion

- Wenn keine Anforderung besteht, warte im Erdgeschoss mit offener Tür.

Iteration

- Solange ein Anforderung besteht, bediene Anforderungen.

Auswahl

- Wähle eine bestehende Anforderung zur Bedienung aus.

➔ Kontrollstrukturen in Programmiersprachen

- Unterschied: Primitive Aktionen

Komplexe Aktionen in GOLOG

Erweiterung des Situationskalküls

- Es sei
 - PrA eine Menge von Symbolen für primitive Aktionen
 - F eine Menge von Symbolen für Fluents
- Die Menge der GOLOG-Programme $Golog(PrA, F)$ ist dann die kleinste Menge, für die gilt
 - $PrA \subseteq Golog(PrA, F)$
 - Wenn $A, B \in Golog(PrA, F)$, $\Phi \in F$ und x eine Variable, dann sind auch $[A ; B]$, $[A | B]$, $if(\Phi, A)$, $while(\Phi, A)$, $?(A)$, $[\pi x. A] \in Golog(PrA, F)$

Ausführung komplexer Aktionen

Fluents für Tests

- in den Kontrollstrukturen **if**, **while**, **?** werde Tests vorgenommen, deren Resultat situationsabhängig sein kann
- Fluents erlauben diesen Situationsbezug
- Damit im 'Programm' implizit gelassen werden kann, auf welche Situation der Fluent angewendet wird:
 - Φ steht für ein Fluent mit unterdrückten Situationsargument (**d_open**, **on(3)**, **currentFloor(4)**)
 - $\Phi(s)$ steht für den entsprechenden Fluent mit 'restauriertem' Situationsargument **s** (**d_open(s)**, **on(3, s)**, **currentFloor(4, s)**)

Ch. Habel / C. Eschenbach: Wissensrepräsentation, SoSe 2003

26 – 17

Ausführung komplexer Aktionen

- ist eine Sequenz primitiver Aktionen, angewendet auf **s0**

Vorbedingungen

- ergeben sich aus den Vorbedingungen der eingebetteten primitiven Aktionen

Zielsituationen

- sind nicht eindeutig (mehrere Sequenzen primitiver Aktionen können einer komplexen Aktion entsprechen)

Do(A, s, s')

- Formel im Situationskalkül
- Intendierte Bedeutung: die Ausführung von **A** kann von **s** zu **s'** führen
- **s'** gibt eine Sequenz von **primitiven Aktionen**, angewendet auf **s** an

Ch. Habel / C. Eschenbach: Wissensrepräsentation, SoSe 2003

26 – 18

Ausführung komplexer Aktionen

- ist eine Sequenz primitiver Aktionen, angewendet auf **s0**

Vorbedingungen

- ergeben sich aus den Vorbedingungen der eingebetteten primitiven Aktionen

Zielsituationen

- sind nicht eindeutig (mehrere Sequenzen primitiver Aktionen können einer komplexen Aktion entsprechen)

Do(A, s, s')

- Formel im Situationskalkül
- Intendierte Bedeutung: die Ausführung von **A** kann von **s** zu **s'** führen
- **s'** gibt eine Sequenz von **primitiven Aktionen**, angewendet auf **s** an

Ch. Habel / C. Eschenbach: Wissensrepräsentation, SoSe 2003

26 – 18

Bedingungen für Do(A, s, s')

- 'rekursive Definition'

Primitive Aktionen

$$\text{Do}(a, s, s') \Leftrightarrow \text{Poss}(a, s) \wedge s' = \text{do}(a, s)$$

Sequenzen

$$\text{Do}([A;B], s, s') \Leftrightarrow \exists s'' [\text{Do}(A, s, s'') \wedge \text{Do}(B, s'', s')]$$

Wahl, nichtdeterministische Verzweigung

$$\text{Do}([A \mid B], s, s') \Leftrightarrow \text{Do}(A, s, s') \vee \text{Do}(B, s, s')$$

Wahl eines Wertes, nichtdeterministisch

(**x** kommt in **A** frei vor)

$$\text{Do}([\text{rx}. A], s, s') \Leftrightarrow \exists x [\text{Do}(A, s, s')]$$

Ch. Habel / C. Eschenbach: Wissensrepräsentation, SoSe 2003

26 – 19

Bedingungen für Do(A, s, s')

Tests

$$\text{Do}(\text{?}(\Phi), s, s') \Leftrightarrow (\Phi(s) \wedge s = s')$$

Bedingte Aktionen

$$\text{Do}(\text{if}(\Phi, A, B), s, s') \Leftrightarrow (\Phi(s) \wedge \text{Do}(A, s, s')) \vee (\neg\Phi(s) \wedge \text{Do}(B, s, s'))$$

Iteration, While-Schleife

$$\text{Do}(\text{while}(\Phi, A), s, s') \Leftrightarrow (\neg\Phi(s) \wedge s = s') \vee (\Phi(s) \wedge \text{Do}([A; \text{while}(\Phi, A)], s, s'))$$

- hier ist die allgemeine Terminationsproblematik zu beachten

Ch. Habel / C. Eschenbach: Wissensrepräsentation, SoSe 2003

26 – 20

GOLOG-Programme

Primitive Aktionen und Fluents

- sind anwendungsbezogen zu spezifizieren
- durch Vorbedingungen und Nachfolgezustandsaxiome

Die Programm-Semantik

- (= was das Ausführen der Programme bewirkt)
- ist von diesen Spezifikationen abhängig

Zwei Aspekte der Ausführung von A beginnend bei s_0

- gibt es (überhaupt) eine Sequenz von primitiven Aktionen $\langle a_1, \dots, a_n \rangle$, so dass $\mathcal{KB} \models \text{Do}(A, s_0, \text{do}(\langle a_1, \dots, a_n \rangle s_0))$
- Führe die Sequenz $\langle a_1, \dots, a_n \rangle$ durch einen Roboter oder Simulator aus

Aufzug-Kontrolle

Definitionen komplexer Aktionen

$\text{goFloor}(n) \stackrel{\text{def}}{=} \text{?(currentFloor}(n)) \#$
 $\text{((?(d_closed) \# close) ; (up}(n) \# \text{down}(n)))}$
 $\text{serve}(n) \stackrel{\text{def}}{=} \text{goFloor}(n) ; \text{?(?-d_closed) \# open} ;$
 $\text{turnoff}(n)$
 $\text{serveAfloor} \stackrel{\text{def}}{=} [\pi n. \text{?(on}(n)) ; \text{serve}(n)]$
 $\text{park} \stackrel{\text{def}}{=} \text{goFloor}(0) ; \text{?(?-d_closed) \# open}$
 $\text{control} \stackrel{\text{def}}{=} \text{while}(\exists n \text{ on}(n), \text{serveAfloor}) ; \text{park}$

Aufzugsbeispiel: Mögliche Aktionen

	cFl	on	d_cl	Poss(a)	
s0	4	3, 5	f	close	
s1	do(close, s0)	4	3, 5	t	open, down(0)...(3), up(5)
s2	do(down(3), s1)	3	3, 5	t	open, down(0)...(2), up(4)...(5)
s3	do(open, s2)	3	3, 5	f	turnoff, close
s4	do(turnoff(3), s3)	3	5	f	close
s5	do(close, s4)	3	5	t	open, down(0)...(2), up(4)...(5)
s6	do(up(5), s5)	5	5	t	open, down(0)...(4)
s7	do(open, s6)	5	5	f	turnoff, close
s8	do(turnoff(5), s7)	5	5	f	close
s9	do(close, s8)	5	5	t	open, down(0)...(4)
s10	do(down(0), s9)	0	5	t	open, up(1)...(5)
s11	do(open, s10)	0	5	f	close

$\text{goFloor}(n) \stackrel{\text{def}}{=} \text{?(currentFloor}(n)) \#$ $\text{((?(d_closed) \# close) ; (up}(n) \# \text{down}(n)))}$

	cFl	on	d_cl	Do(goFloor(n), s, s')	
s0	4	3, 5	f		
s1	do(close, s0)	4	3, 5	t	
s2	do(down(3), s1)	3	3, 5	t	3
s3	do(open, s2)	3	3, 5	f	
s4	do(turnoff(3), s3)	3	5	f	
s5	do(close, s4)	3	5	t	
s6	do(up(5), s5)	5	5	t	5
s7	do(open, s6)	5	5	f	
s8	do(turnoff(5), s7)	5	5	f	
s9	do(close, s8)	5	5	t	
s10	do(down(0), s9)	0	5	t	0
s11	do(open, s10)	0	5	f	

park =_{def} goFloor(0) ; (?(- d_closed) # open)

	cFl	on	d_cl	Do(park, s, s')
s0	4	3, 5	f	
s1	4	3, 5	t	
s2	3	3, 5	t	
s3	3	3, 5	f	
s4	3	5	f	
s5	3	5	t	
s6	5	5	t	
s7	5	5	f	
s8	5		f	
s9	5		t	
s10	0		t	t
s11	0		f	t

Ch. Habel / C. Eschenbach: Wissensrepräsentation, SoSe 2003

26 – 25

serve(n) =_{def} goFloor(n) ; (?(- d_closed) # open) ; turnoff(n)

	cFl	on	d_cl	Do(serve(n), s, s')
s0	4	3, 5	f	
s1	4	3, 5	t	
s2	3	3, 5	t	
s3	3	3, 5	f	3
s4	3	5	f	3
s5	3	5	t	
s6	5	5	t	
s7	5	5	f	5
s8	5		f	5
s9	5		t	
s10	0		t	
s11	0		f	

Ch. Habel / C. Eschenbach: Wissensrepräsentation, SoSe 2003

26 – 26

while(En on(n), serveAfloor)

	cFl	on	d_cl	Do(while(En on(n), serveAfloor), s, s')
s0	4	3, 5	f	
s1	4	3, 5	t	
s2	3	3, 5	t	
s3	3	3, 5	f	
s4	3	5	f	
s5	3	5	t	
s6	5	5	t	
s7	5	5	f	
s8	5		f	
s9	5		t	
s10	0		t	
s11	0		f	

Ch. Habel / C. Eschenbach: Wissensrepräsentation, SoSe 2003

26 – 27

control =_{def} while(En on(n), serveAfloor) ; park

	cFl	on	d_cl	Do(control, s, s')
s0	4	3, 5	f	
s1	4	3, 5	t	
s2	3	3, 5	t	
s3	3	3, 5	f	
s4	3	5	f	
s5	3	5	t	
s6	5	5	t	
s7	5	5	f	
s8	5		f	
s9	5		t	
s10	0		t	
s11	0		f	

Ch. Habel / C. Eschenbach: Wissensrepräsentation, SoSe 2003

26 – 28

$\mathcal{KB} \models \text{Do}(\text{control}, s_0, s)$?

*Init*_{EL}

$\forall m [\text{on}(m, s_0) \Leftrightarrow m = 3 \vee m = 5]$

$\forall m [\text{currentFloor}(m, s_0) \Leftrightarrow m = 4]$

$\neg d_closed(s_0)$

gibt es eine Sequenz $\langle a_1, \dots, a_n \rangle$, so dass

$\mathcal{KB} \models \text{Do}(\text{control}, s_0, \text{do}(\langle a_1, \dots, a_n \rangle, s_0))$

Ja: $\langle \text{close}, \text{down}(3), \text{open}, \text{turnoff}(3), \text{close}, \text{up}(5), \text{open}, \text{turnoff}(5), \text{close}, \text{down}(0), \text{open} \rangle$

Golog

aufbauend auf

- Situationskalkül
- Spezifikation von Vorbedingung und Effekte primitiver Aktionen

Ergänzung um

- Kontrollstrukturen
- mit Spezifikation der Semantik

Semantik eines Golog-Programms

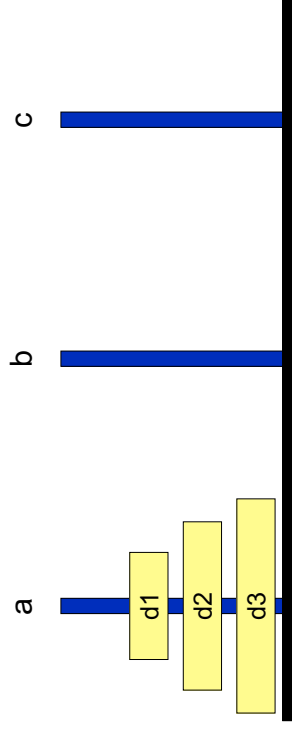
- über Menge der Sequenzen von primitiven Aktionen, die (nachweislich) die Kontrollstrukturen realisieren

Nutzen

- abstraktere Plan-Spezifikation (Instruktion, Planung?)

Golog-Programm für den Turm von Hanoi

Gruppen- diskussion



Schreiben Sie ein Golog-Programm für den Turm von Hanoi