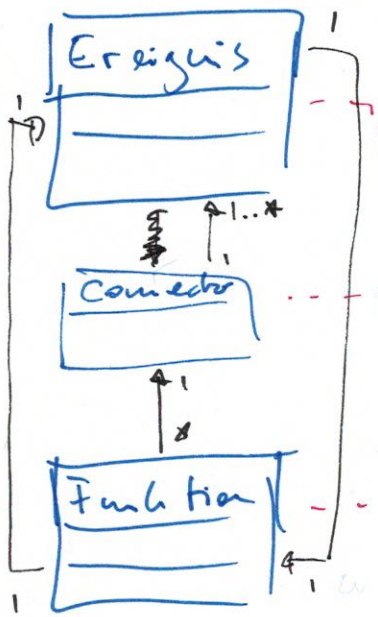
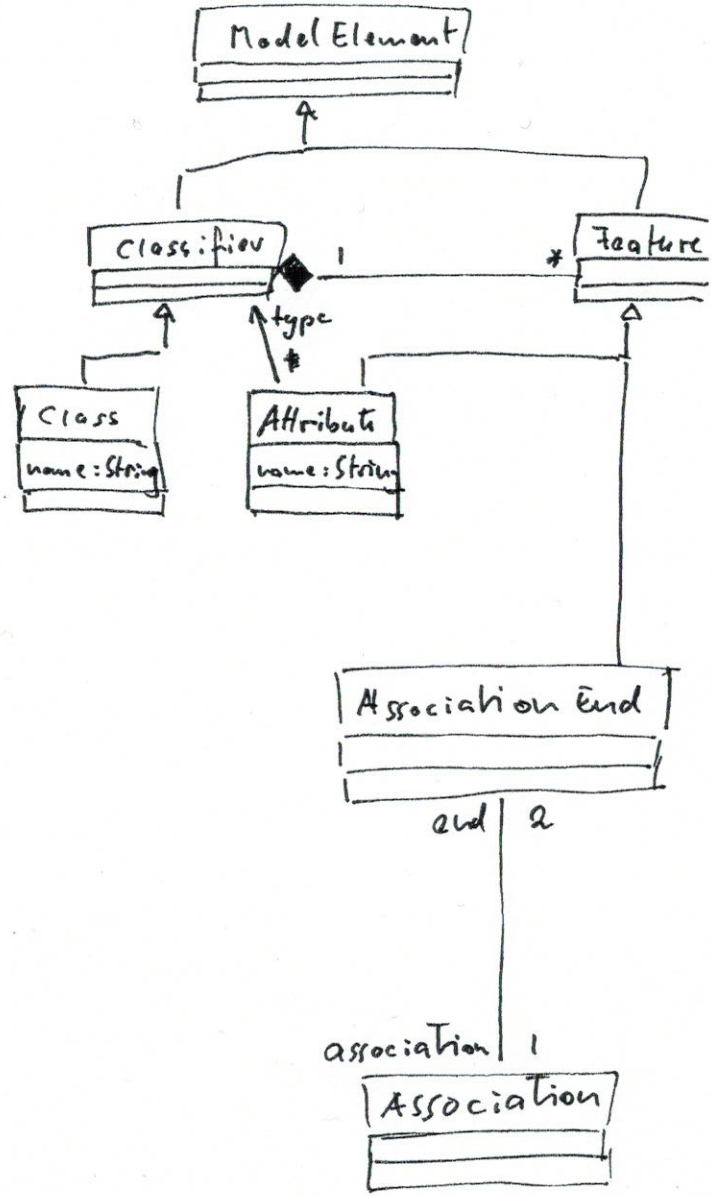
Modell
(in konkreter Syntax)

Metamodell
(in Class Diagram Syntax)

MOF (vereinfacht)



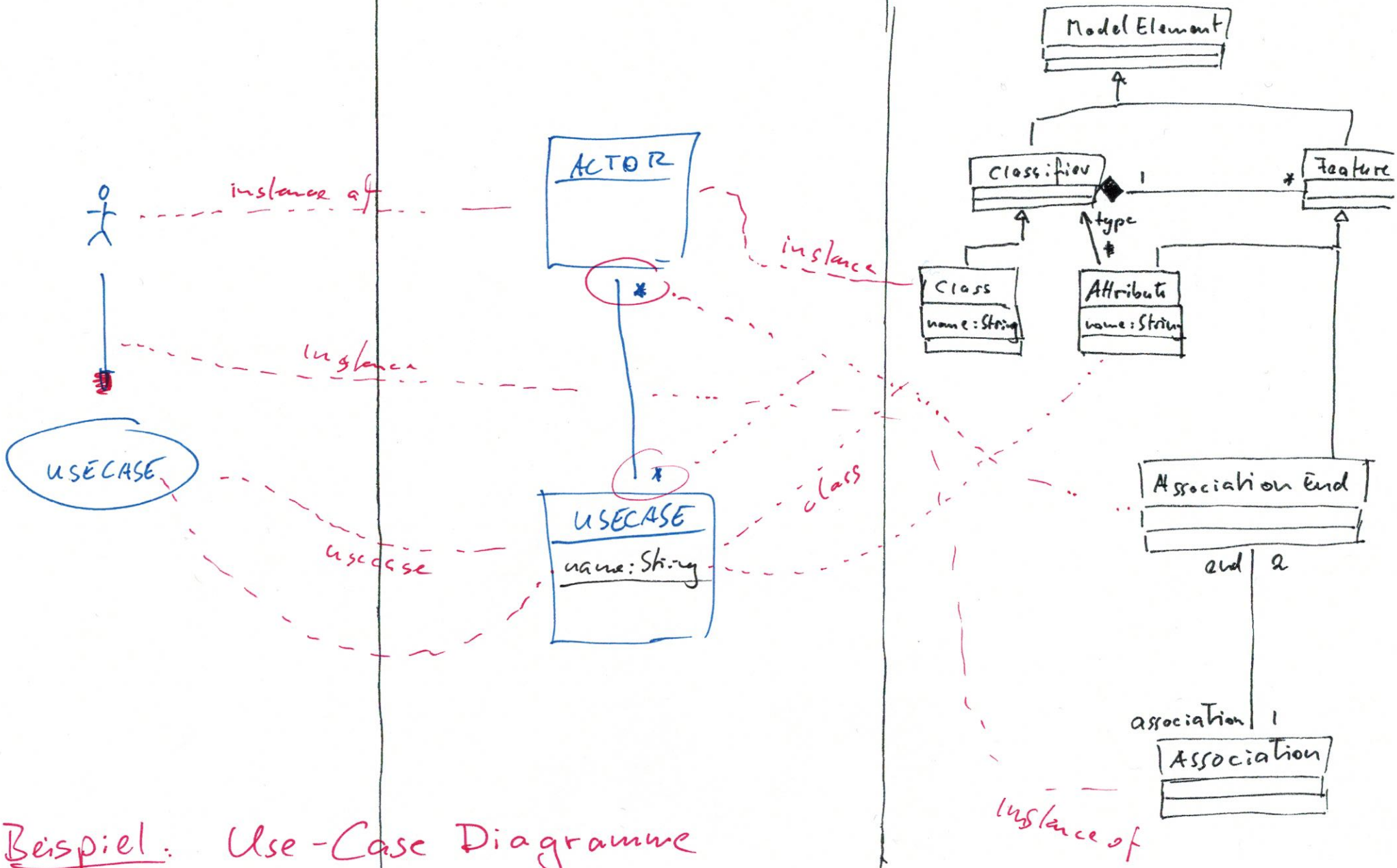


Beispiel: Ereignis Gesteuerte Prozessketten

Modell
(in konkreter Syntax)

Metamodell
(in Class Diagram Syntax)

MOF (vereinfacht)



Beispiel: Use-Case Diagramme

Beispiel: Ausdrücke und Auswertung

Variablenmenge:

$X = \{x_1, x_2\}$

$x_1 : \text{Nat}$

$x_2 : \text{State}$

> getypte Variablen

Beispiele für Belegungen

$v_1(x_1) = 101$ $\quad v_2(x_1) = 100$

$v_1(x_2) = \text{on}$ $\quad v_2(x_2) = \text{off}$

V_X — Menge aller möglichen Belegungen

Aufbau komplexer Ausdrücke

$((x_2 = \text{off}) \wedge (x_1 > 50))$ — komplexer Ausdruck

$(x_2 = \text{off})$: „$z = c$"

$(x_1 > 50)$: „$a_1\ \text{op}_r\ a_2$"

„$b_1\ \text{op}_b\ b_2$"

Komposition

Auswertung

$v(x_1) = 101$
$v(x_2) = off$

$$p_x[(x_2 = on) \wedge (x_1 > 50), v] = 0$$

$p_x[(x_2 = on), v] = 0$

$\gamma_x[x_2, v] = off$

$\gamma_x[on, v] = on$

$p_x[(x_1 > 50), v] = 1$

$\alpha_x[x_1, v] = 101$

$\alpha_x[50, v] = 50$

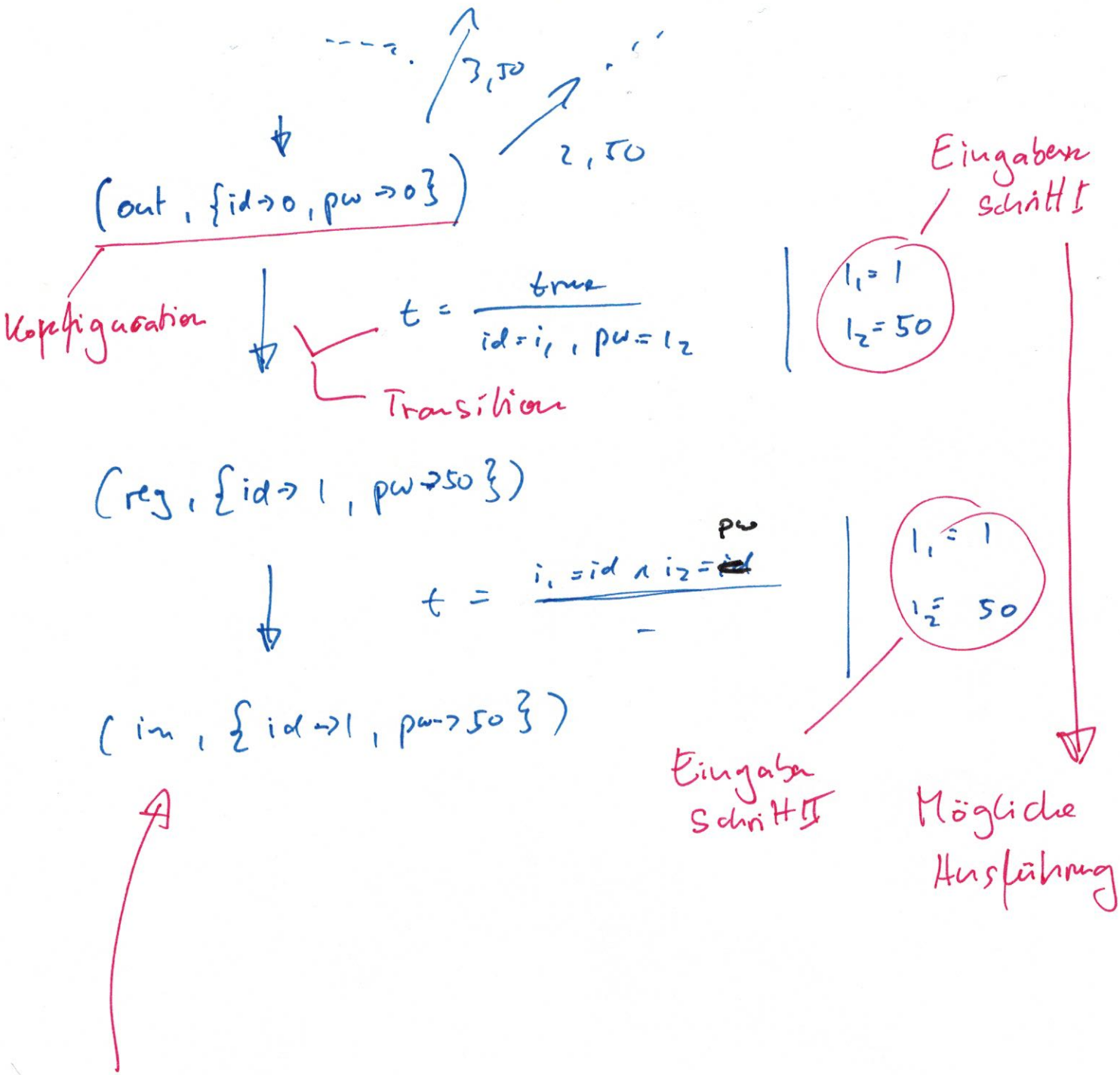
Grafische (konkrete) Syntax

$\rightarrow (i_1 \wedge .. \wedge .. i_n) \Leftrightarrow o$

Element im Blockschaltdiagramm

Ausdruck

Semantik erweiterte Endliche Automaten

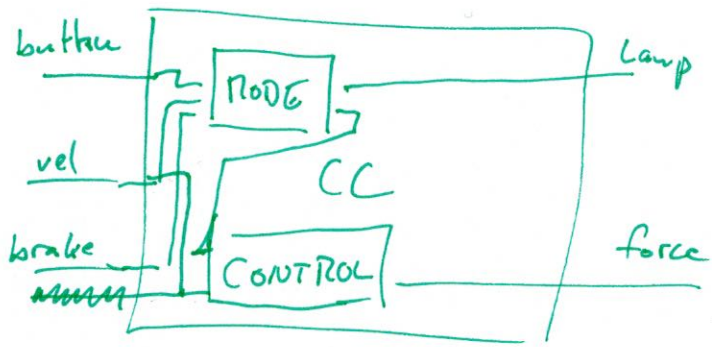


Semantik: Menge der möglichen Ausführungen
(Graph konkreter Konfigurationen)

Modellierung von Verhalten

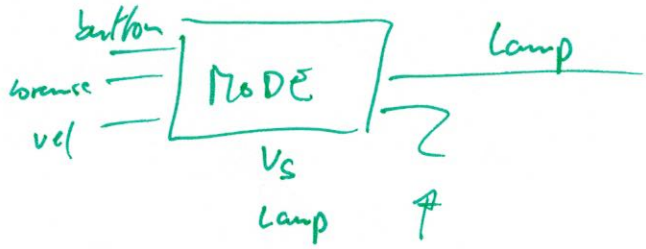
Beispiel "Cruise Control" (Tempomat)

Architektur



| sensorik | system | outputs (Aktuatorik)

Kontroll-Logik:



ignorieren wir

$v_S := 0$
(Lamp := false)

$\text{button} \wedge \text{vel} \geq 7\frac{m}{s} \wedge \overline{\text{brake}}$

$v_S := \text{vel}$; Lamp := true;

$\overline{\text{brake}} \vee \text{vel} < 7\frac{m}{s} \vee \overline{\text{button}}$

$\text{button} \wedge \text{vel} \geq 7\frac{m}{s} \wedge \overline{\text{brake}}$

brake ∨ vel < $7\frac{m}{s}$ ∨ button

Lamp := false

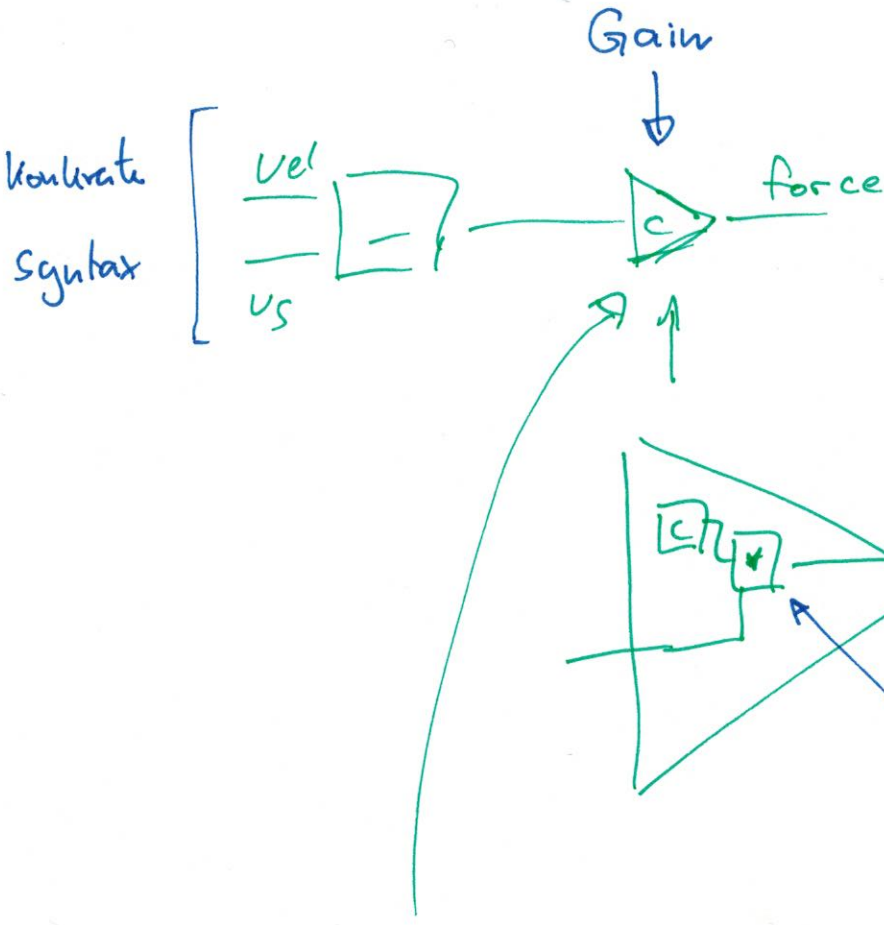
Verhalten

Ausführung

	t_1	t_2	..... → Zeit
brake	⊥	⊥	
button	⊥	T	→ Konfigurationen
vel	1	30	

$\langle q_0, v_S := 0 \rangle$ $\langle q_0, \text{Lamp} := \bot \rangle$ → $\langle q_0, -\!\!-\!\!- \rangle$ $\langle q_1, v_S := 30, \text{Lamp} := T \rangle$ .....

Regelung der Geschwindigkeit



Wir schauen uns nur einen kleinen Ausschnitt an!

$$\alpha[c \cdot x, v] =_{def} \alpha[c, v] \cdot \alpha[x, v]$$

Auswertung

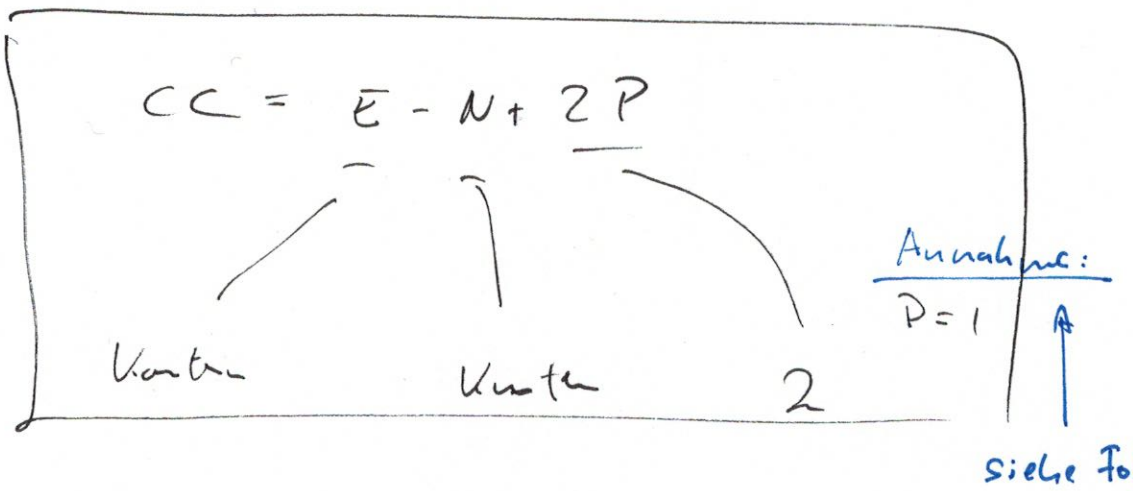
$$\alpha[5 \cdot x] = 50 \quad , \quad v(x) = 10$$

5 → $\alpha[5, v]$

10 → $\alpha[x, v]$

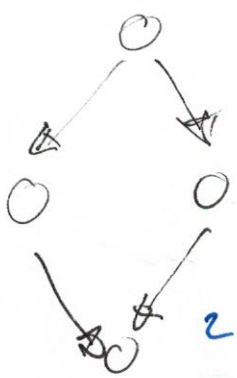
Cyclomatische Komplexität

Beispiel für unintuitive Werte bei Anwendung:



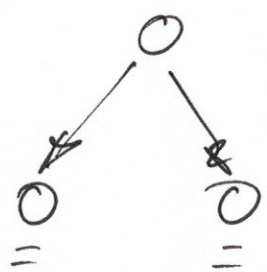
siehe Folie

P_1



2 Pfade
1 Endpunkt

P_2



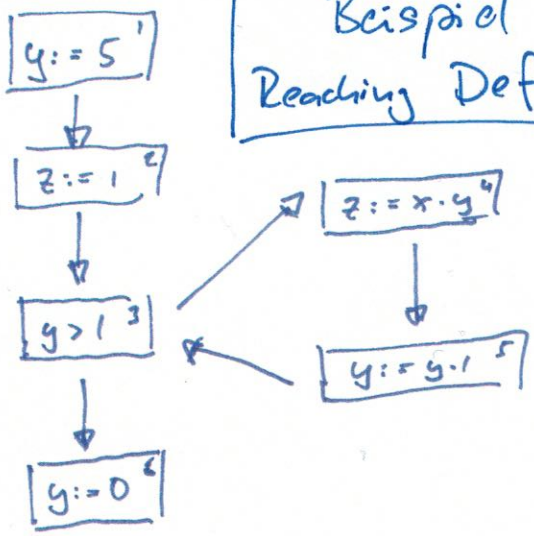
2 Pfade
2 Endpunkte

$cc(P_1) = 4 - 4 + 2 = 2$ $cc(P_2) = 3 - 3 + 2 = 1$

⇒ Wir benutzen die Metrik nur für Programme mit einem Endpunkt

While Programm:

```
[y := 5]^1
[z := 1]^2
while [y > 1]^3
  [z := x · y]^4
  [y := y - 1]^5
[y := 0]^6
```



Beispiel Reaching Def.

Label

↓	RD^{RD}_{entry}	RD_{exit}
1	$\{(x,?),(y,?),(z,?)\}$ ①	$(RD_{entry}(1) \setminus \{(x,?),\dots\}) \cup \{(y,1)\}$ ⑪
2	$RD_{exit}(1)$ ⑪	...
3	$RD_{exit}(2) \cup RD_{exit}(5)$ ⑪	...
4	...	...
5	...	...
6	...	...

① : Initial alle Variablen nicht definiert

⑪ : $RD_{exit}(l) = (RD_{entry}(l) \setminus kill(b)) \cup gen(l)$

⑪ $RD_{entry}(l) = \bigcup_{(l',l) \text{ in } flow(S_*)} RD_{exit}(l')$ für $l \notin init(S_*)$

Beispiele S.O.S.

$x := y + z$

$x := 5$ → (vgl. Auswertung Ausdrücke)

while $(x > 1)$
skip;

Semantik? S.O.S.

Regel [ASS]

→ $\langle x := 5, [x \mapsto 2] \rangle \Rightarrow [x \mapsto 5]$

→ $\langle x := y + z, [x \mapsto 2, y \mapsto 2, z \mapsto 2] \rangle = \begin{bmatrix} x \mapsto 4 \\ y \mapsto 2 \\ z \mapsto 2 \end{bmatrix}$

Konfiguration

Belegung nach Ausführung Assignment

Beispiele für weitere Regeln r.u.

Semantik von S_*

Graph von Konfigurationen

Programm S_*

S_I

$S_* \equiv$ { $x := y + 10$; { if $(x > 3)$ then $z := x - S_2$ else skip $- S_3$ } S_1

Initiale Belegung

$V_1 = [x \mapsto 7, y \mapsto 5, z \mapsto 0]$

Initiale Konfiguration: $\langle S_*, V_1 \rangle$

$\xrightarrow{\text{ASS}, \text{COMP}_2}$ $\langle S_1, V_1[x \mapsto 15] \rangle$

$\langle S_I, V_1 \rangle \Rightarrow V'$ [ASS] / $\langle S_I; S_1, V_1 \rangle \Rightarrow \langle S_1, V' \rangle$ [COMP]

$[IF_T]$

$\langle S_2, V' \rangle$ $z := x$ $\xrightarrow{[ASS]}$ $[x \mapsto 15, y \mapsto 5, z \mapsto 15]$

└ Belegung nach Ausführung von S_*

Beispiele: Vor-/Nach-Bedingung Invarianten

$S_* = S_1 ; S_2$

Vor.: $(y = 0)$

$S_1 - x := 10 + y$;
$S_2 - y := 10 - y$

} Inv. $(x > 0)$ — gilt diese Bed. in jeder Konfiguration einer Ausführung

gilt diese Nachbedingung wenn Vorbedingung gilt?

N.B.: $(x = y)$?

Inv. gilt nicht

$\langle S_* , \begin{bmatrix} x \to 0 \\ y \to 0 \end{bmatrix} \rangle \xrightarrow[\text{Comp}_2]{\text{ASS}} \langle S_2 , \begin{bmatrix} x \to 10 \\ y \to 0 \end{bmatrix} \rangle$

Ausführung I:

$\begin{bmatrix} x \mapsto 10 \\ y \mapsto 10 \end{bmatrix}$

N.B. gilt

$\langle S_* , \begin{bmatrix} x \mapsto 10 \\ y \mapsto 1 \end{bmatrix} \rangle \longrightarrow \langle S_2 , \begin{bmatrix} x \mapsto 11 \\ y \mapsto 1 \end{bmatrix} \rangle$

Ausführung II:

$\begin{bmatrix} x \mapsto 4 \\ y \mapsto 9 \end{bmatrix}$

×

N.B. nicht erfüllt