Foundations of the UML

Lecture 12: A Logic for MSCs

Joost-Pieter Katoen

Lehrstuhl für Informatik 2
Software Modeling and Verification Group

http://moves.rwth-aachen.de/i2/370

7. Dezember 2009

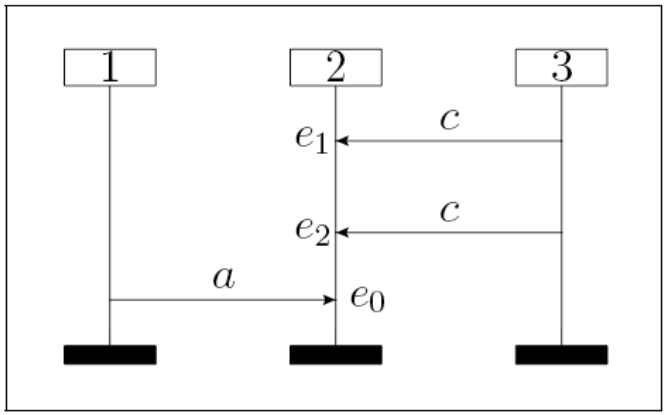
RWTH AACHEN UNIVERSITY

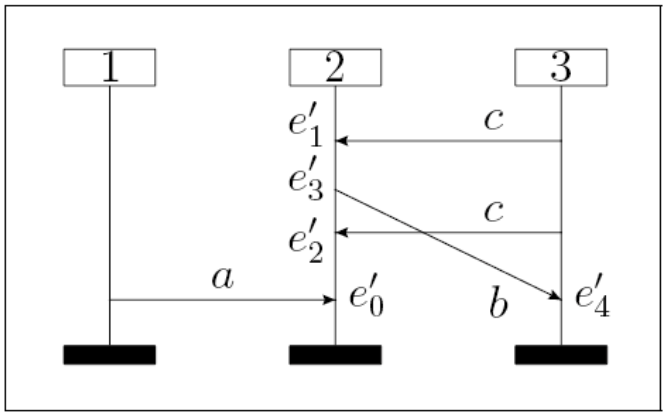
A logic for MSCs

- This lecture will be devoted to a logic that is interpreted over MSCs
- The logic is used to express properties of MSCs
  - does a given MSC M satisfy the logical formula φ?
- And to characterise a set of MSCs by means of a logical formula
  - all MSCs that satisfy the formula φ
- Our logic is a variant of propositional dynamic logic (PDL) [Fischer & Ladner, 1979]
  - combines easy-to-grasp concepts such as regular expressions and Boolean operators
- We consider syntax, semantics, examples and the membership problem.

RWTH AACHEN UNIVERSITY

Informal examples





1. The (unique) maximal event of M is labeled by $?(2, 1, a)$ Yes. No.
2. The maximal event on process 2 is labeled by $?(2, 1, a)$ Yes. Yes.
3. No two consecutive events are labeled with $?(2, 3, c)$ No. Yes.
4. The number of send events at process 3 is odd. No. No.

RWTH AACHEN UNIVERSITY

Local formulas

Definition (Syntax of local formulas)

For communication action $\sigma \in Act$ and path expression α, the grammar of local formulas is given by:

$$\varphi \ ::= \ true \ \mid \ \sigma \ \mid \ \neg\varphi \ \mid \ \varphi \vee \varphi \ \mid \ \langle\alpha\rangle\varphi \ \mid \ \langle\alpha\rangle^{-1}\varphi$$

Definition (Derived operators)

$$\begin{aligned} false &= \neg true \\ \varphi_1 \wedge \varphi_2 &= \neg(\neg\varphi_1 \vee \neg\varphi_2) \\ \varphi_1 \rightarrow \varphi_2 &= \neg\varphi_1 \vee \varphi_2 \\ [\alpha]\varphi &= \neg\langle\alpha\rangle\neg\varphi \\ [\alpha]^{-1}\varphi &= \neg\langle\alpha\rangle^{-1}\neg\varphi \end{aligned}$$

Path expressions

Definition (Syntax of path expressions)

For local formula φ, the grammar of path expressions is given by:

$$\alpha \ ::= \ \{\varphi\} \ | \ \mathsf{proc} \ | \ \mathsf{msg} \ | \ \alpha;\alpha \ | \ \alpha + \alpha \ | \ \alpha^*$$

Definition (Syntax of PDL formulas)

For local formula φ, the grammar of PDL formulas is given by:

$$\Phi \ ::= \ \exists\varphi \ | \ \neg\Phi \ | \ \Phi \vee \Phi$$

and we define $\forall\varphi \ := \ \neg\exists\neg\varphi$.

RWTH AACHEN UNIVERSITY

Intuitive meaning of local formulas

- Local formulas are interpreted over MSC events
- Event e satisfies $\underbrace{!(p,q,a)}_{\sigma}$ iff e is labeled with action $\underbrace{!(p,q,a)}_{\sigma}$
- Path expression α defines a binary relation between events:
  1. $\{\varphi\}$ is the set of pairs (e,e') such that e satisfies φ
  2. $(e,e') \models \mathsf{proc}$ iff e and e' reside at the same process and e' is a direct successor of e wrt. $<_p$
  3. $(e,e') \models \mathsf{msg}$ iff e' is the matching event of e, i.e., $e' = m(e)$

RWTH AACHEN UNIVERSITY

Intuitive meaning of local formulas

- Event e satisfies $\langle\alpha\rangle\varphi$ iff there is an event e' such that (e, e') satisfies α and e' satisfies φ
- The interpretation of $\langle\alpha\rangle^{-1}\varphi$ is dual, i.e., e satisfies it iff there is an event e' such that (e', e) satisfies α and e' satisfies φ
- The composition $\alpha;\beta$ defines the set of pairs (e, e') for which there exists event e'' such that $(e, e'') \models \alpha$ and $(e'', e') \models \beta$
- $\alpha + \beta$ denotes the union of the relations α and β
- α^* denotes the reflexive and transitive closure of the relation α

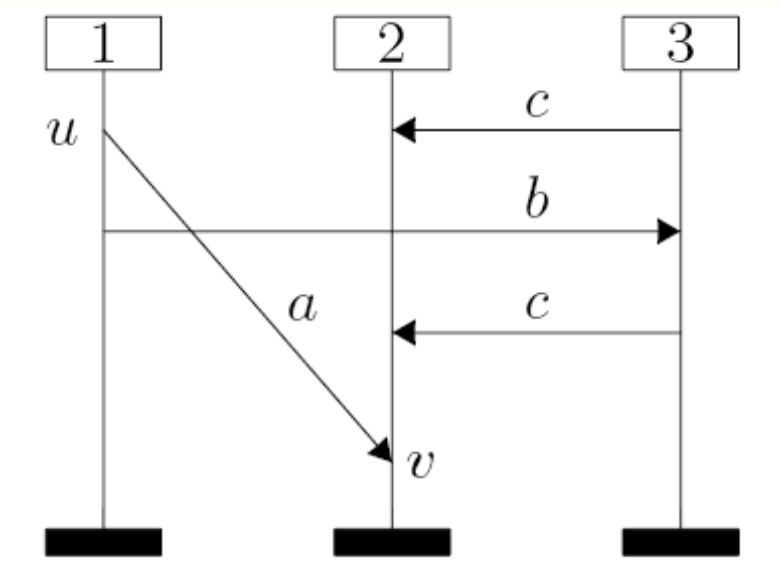
RWTH AACHEN UNIVERSITY

Intuition of PDL formulas

- MSC M satisfies $\exists\varphi$ if it has some event e satisfying φ
- MSC M satisfies $\exists\langle\alpha\rangle\varphi$ if from some event e in M, there exists an α-labeled path from e to an event e', say, satisfying φ
- MSC M satisfies $\exists[\alpha]\varphi$ if from some event e in M, any event that can be reached via an α-labeled path satisfies φ

RWTH AACHEN UNIVERSITY

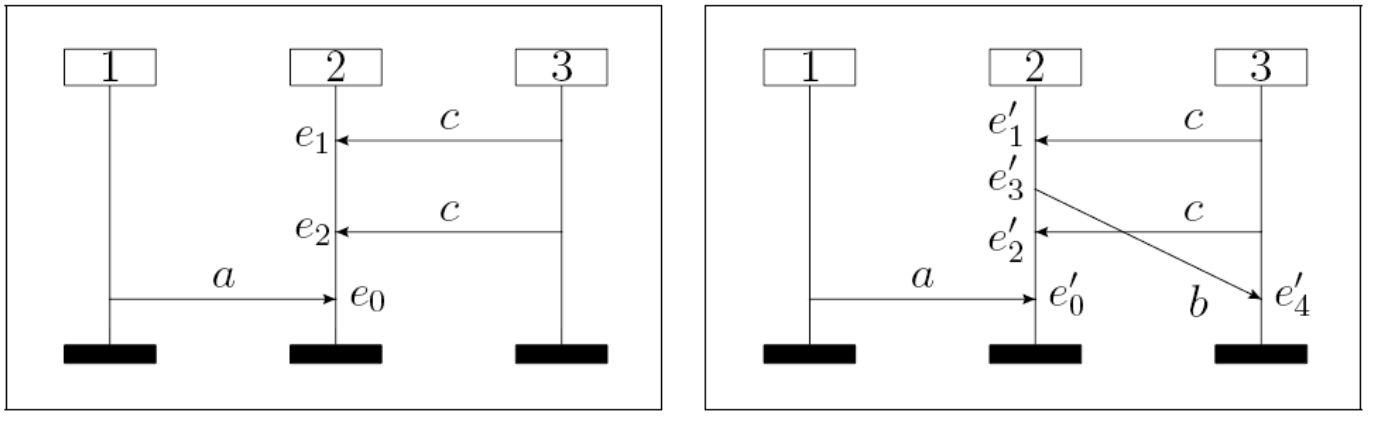
Example



1. $u \models !(1, 2, a)$ — u is labeled with the action $!(1, 2, a)$
2. $u \models [\mathsf{proc}]^{-1}\, \mathit{false}$ — u is the first event on the process line
3. $u \models \langle(\mathsf{proc} + \mathsf{msg})^*\rangle ?(2, 1, a)$ — event u happens before the event v

RWTH AACHEN UNIVERSITY

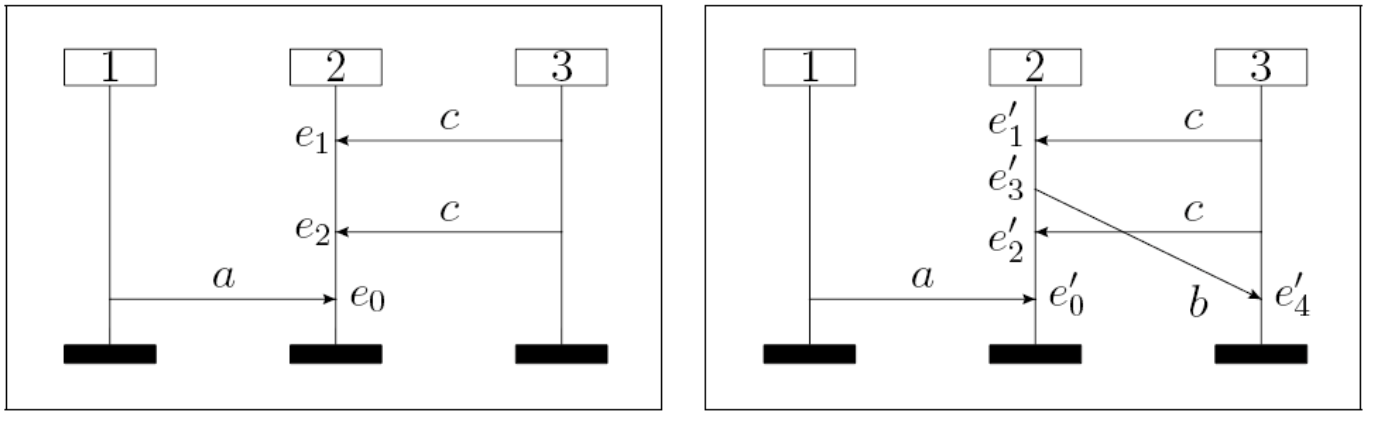
Example (1)



- The (unique) maximal event of M is labeled by $?(2, 1, a)$ Yes. No.
- $\forall\,(\langle(\mathsf{proc} + \mathsf{msg})^*\rangle([\mathsf{proc}]\,\mathit{false} \wedge ?(2, 1, a)))$ Yes. No.

RWTH AACHEN UNIVERSITY

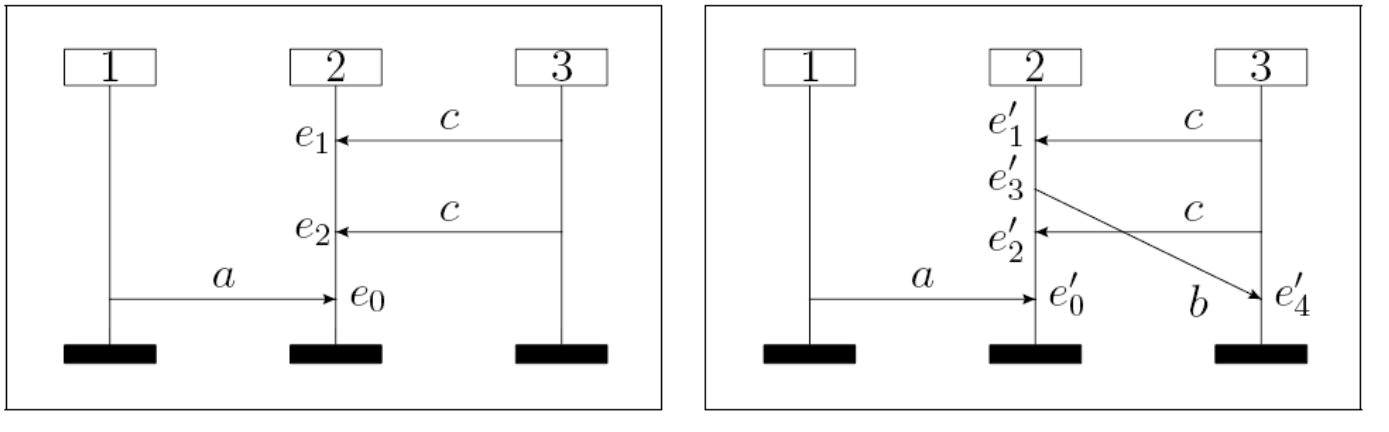
Example (2)



- The maximal event on process 2 is labeled by $?(2, 1, a)$ Yes. Yes.
- $\exists\,([\mathsf{proc}]\,\mathit{false} \wedge ?(2, 1, a))$ Yes. Yes.

RWTH AACHEN UNIVERSITY

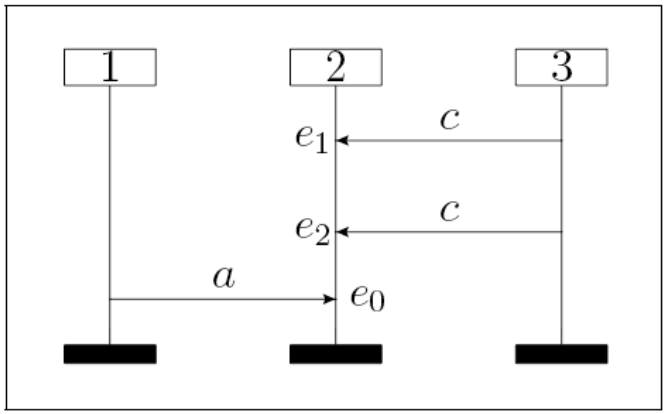
Example (3)

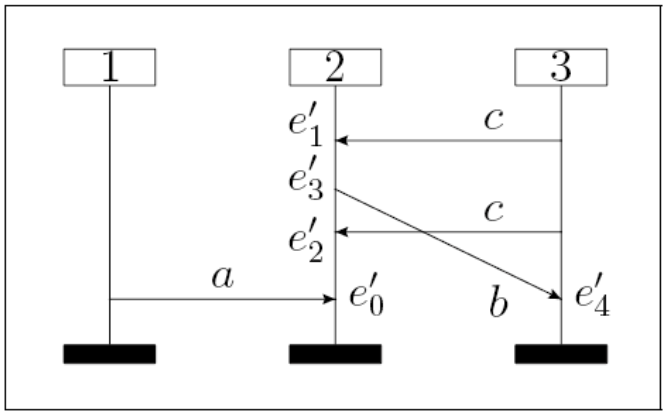


- No two consecutive events are labeled with $?(2, 3, c)$ No. Yes.
- $\forall\,([?(2, 3, c); \mathsf{proc}; ?(2, 3, c)]\, \mathit{false})$ No. Yes.

RWTH AACHEN UNIVERSITY

Example (4)





- The number of send events at process 3 is odd. No. No.
- See next slide

Example

MSC M has an even number of messages sent from process 1 to 2:

$$\forall\Big(\quad \underbrace{[\mathsf{proc}]^{-1}\, \mathit{false}}_{\text{minimal event on process}} \quad \rightarrow\ \langle\alpha\rangle \quad \underbrace{[\mathsf{proc}]\, \mathit{false}}_{\text{maximal event on process}} \quad \Big)$$

$$\alpha \;=\; ((\{?_2\};\mathsf{proc})^*;\{!_1\};\mathsf{proc};(\{?_2\};\mathsf{proc})^*;\{!_1\};\mathsf{proc};(\{?_2\};\mathsf{proc})^*)^*$$

where $!_1$ abbreviates $!(1,2,a)$ and $?_2$ stands for $?(2,1,a)$, and

RWTH AACHEN UNIVERSITY

Semantics of local formulas (1)

Definition (Semantics of simple local formulas)

Let $M = (\mathcal{P}, E, \mathcal{C}, l, m, <) \in \mathbb{M}$ be an MSC and $e \in E$.

Relation $\models$ is defined such that $((M, e), \varphi) \in \models$ iff local formula φ holds in event e of MSC M.

$$
\begin{array}{rcl}
M, e \models \mathit{true} & & \text{for all } e \in E \\
M, e \models \sigma & \text{iff} & l(e) = \sigma \\
M, e \models \neg\varphi & \text{iff} & \text{not } M, e \models \varphi \\
M, e \models \varphi_1 \vee \varphi_2 & \text{iff} & M, e \models \varphi_1 \text{ or } M, e \models \varphi_2
\end{array}
$$

Notes:

1. If M is clear from context we write $e \models \varphi$ instead of $M, e \models \varphi$.
2. It remains to define the semantics for local formulas with path expressions.

Semantics of local formulas (2)

Definition (Semantics of local formulas with forward path expressions)

Let $M = (\mathcal{P}, E, \mathcal{C}, l, m, <) \in \mathbb{M}$ be an MSC and $e \in E$.

$((M, e), \varphi) \in \models$ iff local formula φ holds in event e of MSC M.

$$
\begin{aligned}
e \models \langle \{\psi\} \rangle \varphi \quad &\text{iff} \quad e \models \psi \text{ and } e \models \varphi \\
e \models \langle \mathsf{proc} \rangle \varphi \quad &\text{iff} \quad \exists p \in \mathcal{P}, e' \in E.\, e \lessdot_p e' \text{ and } e' \models \varphi \\
e \models \langle \mathsf{msg} \rangle \varphi \quad &\text{iff} \quad \exists e' \in E.\, e' = m(e) \text{ and } e' \models \varphi \\
e \models \langle \alpha_1 ; \alpha_2 \rangle \varphi \quad &\text{iff} \quad e \models \langle \alpha_1 \rangle \langle \alpha_2 \rangle \varphi \\
e \models \langle \alpha_1 + \alpha_2 \rangle \varphi \quad &\text{iff} \quad e \models \langle \alpha_1 \rangle \varphi \text{ or } e \models \langle \alpha_2 \rangle \varphi \\
e \models \langle \alpha^* \rangle \varphi \quad &\text{iff} \quad \exists n \in \mathbb{N}.\, e \models (\langle \alpha \rangle)^n \varphi
\end{aligned}
$$

Where $e \lessdot_p e'$ iff $e <_p e'$ and $\neg(\exists e''.\, e <_p e'' <_p e')$, i.e., e' is a direct successor of e under $<_p$.

Semantics of local formulas (3)

Definition (Semantics of local formulas with backward path expressions)

Let $M = (\mathcal{P}, E, \mathcal{C}, l, m, <) \in \mathbb{M}$ be an MSC and $e \in E$.

$((M, e), \varphi) \in \models$ iff local formula φ holds in event e of MSC M.

$$
\begin{aligned}
e \models \langle \{\psi\} \rangle^{-1} \varphi \quad &\text{iff} \quad e \models \psi \text{ and } e \models \varphi \\
e \models \langle \mathsf{proc} \rangle^{-1} \varphi \quad &\text{iff} \quad \exists p \in \mathcal{P}, e' \in E.\, e' <_p e \text{ and } e' \models \varphi \\
e \models \langle \mathsf{msg} \rangle^{-1} \varphi \quad &\text{iff} \quad \exists e' \in E.\, e' = m^{-1}(e) \text{ and } e' \models \varphi \\
e \models \langle \alpha_1 ; \alpha_2 \rangle^{-1} \varphi \quad &\text{iff} \quad e \models \langle \alpha_1 \rangle^{-1} \langle \alpha_2 \rangle^{-1} \varphi \\
e \models \langle \alpha_1 + \alpha_2 \rangle^{-1} \varphi \quad &\text{iff} \quad e \models \langle \alpha_1 \rangle^{-1} \varphi \text{ or } e \models \langle \alpha_2 \rangle^{-1} \varphi \\
e \models \langle \alpha^* \rangle^{-1} \varphi \quad &\text{iff} \quad \exists n \in \mathbb{N}.\, e \models \left(\langle \alpha \rangle^{-1} \right)^n \varphi
\end{aligned}
$$

RWTH AACHEN UNIVERSITY

Semantics of PDL formulas

Definition (Semantics of PDL formulas)

Let $M = (\mathcal{P}, E, \mathcal{C}, l, m, <) \in \mathbb{M}$ be an MSC.

$(M, \Phi) \in \models$ iff PDL formula Φ holds in MSC M.

$$
\begin{array}{rcl}
M \models \exists\varphi & \text{iff} & \exists e \in E.\, M, e \models \varphi \\
M \models \neg\Phi & \text{iff} & \text{not } M \models \Phi \\
M \models \Phi_1 \vee \Phi_2 & \text{iff} & M \models \Phi_1 \text{ or } M \models \Phi_2
\end{array}
$$

Thus:

$M \models \forall\varphi$ iff $M \models \neg\exists\neg\varphi$ iff $\neg\,(\exists e \in E.\, M, e \models \neg\varphi)$ iff $\forall e \in E.\, M, e \models \varphi$.

RWTH AACHEN UNIVERSITY

Examples revisited

Some example formulas with formal semantics on black board.

RWTH AACHEN UNIVERSITY

Membership problem for MSCs

Theorem:

Given a PDL formula Φ and an MSC M, the decision problem "does $M \models \Phi$?" can be solved in $\mathcal{O}(|E| \cdot |\Phi|^2)$ time where $|E|$ denotes the number of events in M and $|\Phi|$ the length of Φ.

Proof

Let Φ be the given PDL formula. In subformulae $\langle\alpha\rangle\varphi$ and $\langle\alpha\rangle^{-1}\varphi$ of Φ, we consider α as regular expression over some finite alphabet $\{\,\mathsf{proc}, \mathsf{msg}, \{\varphi_1\}, \ldots, \{\varphi_n\}\,\}$ with local formulae φ_i. Any such expression can be transformed into a corresponding finite automaton of linear size. We proceed by inductively labeling events of the given MSC with states of the finite automata. This state information is then used to discover whether or not an event of M satisfies a subformula $\langle\alpha\rangle\varphi$ and $\langle\alpha\rangle^{-1}\varphi$ which yields labelings in $\{\,0, 1\,\}$. Boolean combinations and $\exists\varphi$ are then handled in a straightforward manner.

Algorithm for PDL membership problem (1)

LOCAL FORMULA CHECK:

```
1  V = {0, .. , n-1}
2
3  boolean[] Sat(LocalFormula f) {
4      boolean[] sat = new boolean[n];
5      switch(f) {
6      case Not(f1):
7          boolean[] sat1 = Sat(f1);
8          for (int i = 0; i < n; i++)
9              sat[i] = !sat1[i];
10         break;
11     case Or(f1, f2):
12         boolean[] sat1 = Sat(f1);
13         boolean[] sat2 = Sat(f2);
14         for (int i = 0; i < n; i++)
15             sat[i] = sat1[i] || sat2[i];
16         break;
17     case Event(..):
18         for (int i = 0; i < n; i++)
19             sat[i] = (V[i].event.equals(f));
20         break;
```

RWTH AACHEN UNIVERSITY

Algorithm for PDL membership problem (2)

```
21      case <p1> f2:
22          boolean[][] trans1 = Trans(p1);
23          boolean[] sat2 = Sat(f2);
24          for (int i = 0; i < n; i++) {
25              sat[i] = false;
26              for (int j = 0; j < n; j++)
27                  if(trans[i][j])
28                  sat[i] = sat2[j];
29          }
30          break;
31      case <p1>^-1 f2:
32          boolean[][] trans1 = TransBack(p1);
33          boolean[] sat2 = Sat(f2);
34          for (int i = 0; i < n; i++) {
35              sat[i] = false;
36              for (int j = 0; j < n; j++)
37                  if(trans[i][j])
38                  sat[i] = sat2[j];
39          }
40          break;
41      }
42  }
```

RWTH AACHEN UNIVERSITY

Algorithm for PDL membership problem (3)

FORWARD PATH EXPRESSION CHECK:

```
1  boolean[][] Trans(PathFormula p) {
2      boolean[][] trans = new boolean[n][n];
3      switch(p) {
4      case (p1; p2):
5          boolean[][] trans1 = Trans(p1);
6          boolean[][] trans2 = Trans(p2);
7          for (int i = 0; i < n; i++)
8              for (int k = 0; k < n; k++) {
9                  trans[i][k] = false;
10                 for (int j = 0; j < n; j++)
11                     if(trans1[i][j] && trans1[j][k])
12                         trans[i][k] = true;
13             }
14         break;
15     case p1 + p2:
16         boolean[][] trans1 = Trans(p1);
17         boolean[][] trans2 = Trans(p2);
18         for (int i = 0; i < n; i++)
19             for (int j = 0; j < n; j++)
20                 trans[i][j] = trans1[i][j] || trans2[i][j];
21         break;
```

RWTH AACHEN UNIVERSITY

Algorithm for PDL membership problem (4)

```
22      case p1*:
23          boolean[][] trans1 = Trans(p1);
24          for (int i = 0; i < n; i++)
25              for (int j = 0; j < n; j++)
26                  star[i][j] = (i==j);
27          while (true) {
28              for (int i = 0; i < n; i++)
29                  for (int j = 0; j < n; j++)
30                      if (trans1[i][j])
31                          for (int k = 0; k < n; k++)
32                              if (!trans[i][k] && trans1[j][k]) {
33                                  trans[i][k] = true;
34                                  continue;
35                              }
36              break;
37          }
38          break;
39      }
40  }
```

RWTH AACHEN UNIVERSITY

Satisfiability problem for MSCs

Theorem: [Bollig et al., 2007]

Given a PDL formula Φ, the decision problem "does $M \models \Phi$ holds for some MSC M?" is undecidable.

RWTH AACHEN UNIVERSITY

Other PDL decision problems

Theorem: [Alur et al., 2001, Bollig et al., 2007]

Let Φ be a PDL formula. Then:

1. The decision problem "does there exist a CFM $\mathcal{A}$ such that for any MSC $M \in L(\mathcal{A})$ we have $M \models \Phi$" is undecidable.
2. The decision problem "does there exist a CFM $\mathcal{A}$ such that for some existentially B-bounded MSC $M \in L(\mathcal{A})$ we have $M \models \Phi$" is decidable in PSPACE.
3. The decision problem "for MSG G, is there an MSC $M \in L(G)$ such that $M \models \Phi$" is NP-complete.

RWTH AACHEN UNIVERSITY