The Semantics of Partial Model Transformations

Michalis Famelis, Rick Salay, and Marsha Chechik

University of Toronto

June 3rd, 2012,

Models in Software Engineering Workshop at ICSE

Introduction: Uncertainty

Uncertainty: pervasive in SE

Models with uncertainty:

- Represent choice among many possibilities
- Can be refined to many different classical models

Our goal:

Handle models with uncertainty in MDE
without having to remove uncertainty.

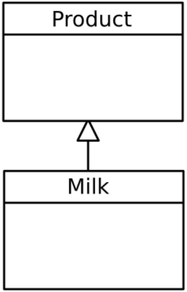
In this talk: Transformations of models with uncertainty

Introduction: Transformations

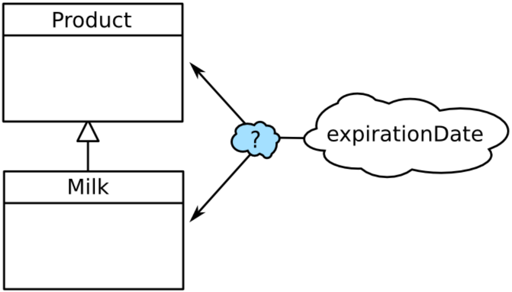
Existing model (graph) transformations:

- Unambiguous model is assumed as input.
- When model contains uncertainty:
  - either first remove uncertainty
    - Premature commitment.
    - Reduced quality.
  - or transform *all* alternatives.
    - Hard to maintain.

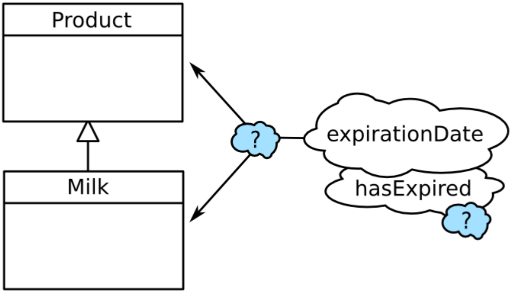
Motivating Example



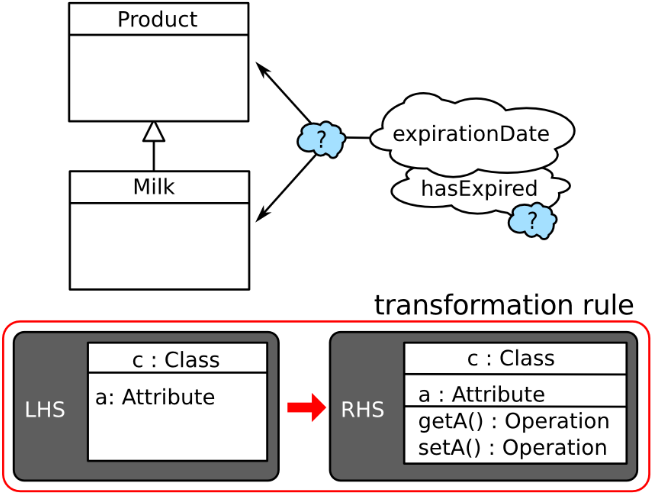
Motivating Example



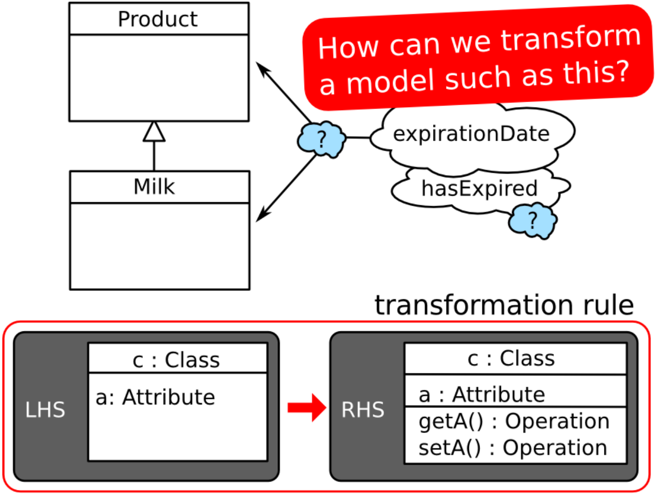
Motivating Example



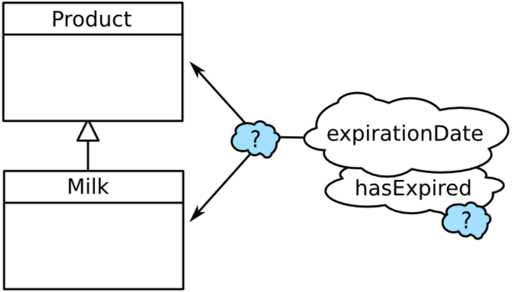
Motivating Example



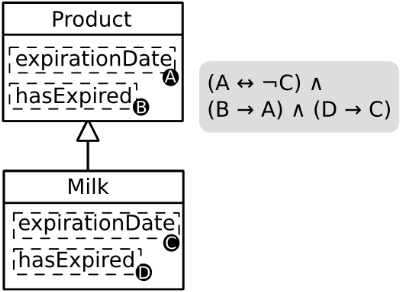
Motivating Example



Partial Models

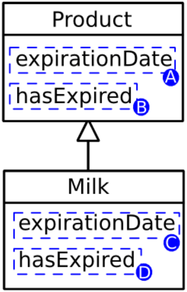


Partial Models



Partial Models:
- Explicate uncertainty
- Syntactic annotations [FASE'12]

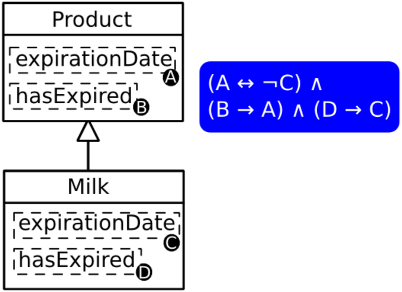
Partial Models



$(A \leftrightarrow \neg C) \wedge$
$(B \rightarrow A) \wedge (D \rightarrow C)$

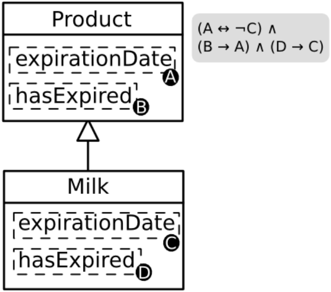
Partial Models:
- Explicate uncertainty
- Syntactic annotations [FASE'12]
- Optional / mandatory elements

Partial Models



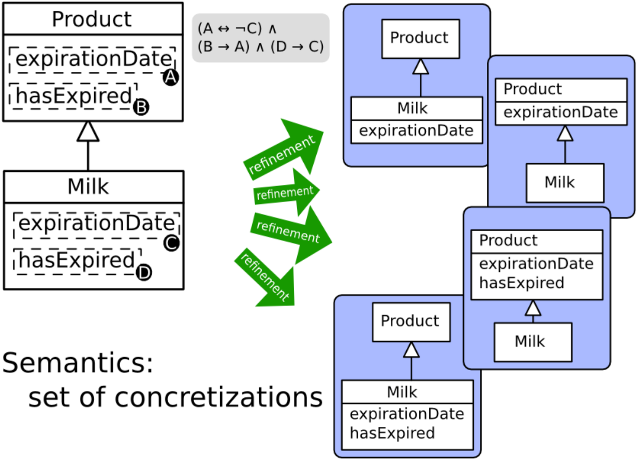
Partial Models:
- Explicate uncertainty
- Syntactic annotations [FASE'12]
- Optional / mandatory elements
- May formula → allowable configurations

Semantics of Partial Models



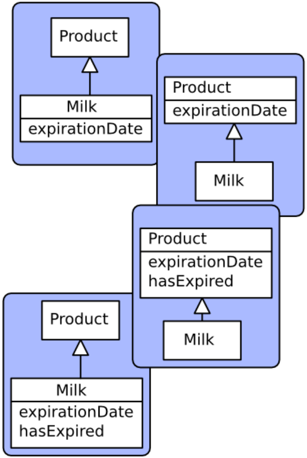
Uncertainty: set of possibilities

Semantics of Partial Models



Semantics:
set of concretizations

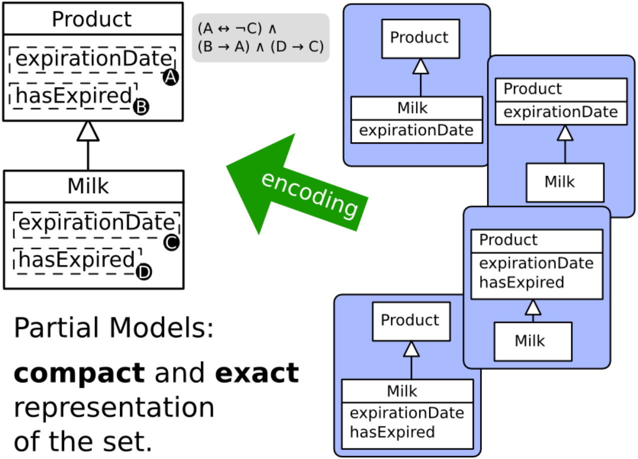
Semantics of Partial Models



Partial Models:

compact and **exact** representation of the set.

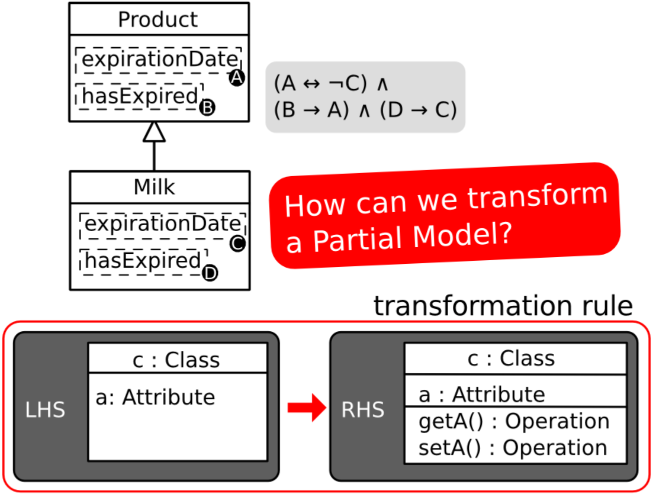
Semantics of Partial Models



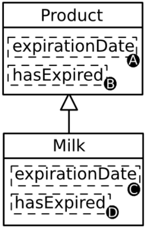
Partial Models:

compact and **exact** representation of the set.

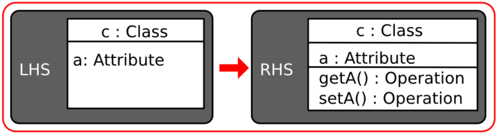
Goal of This Work



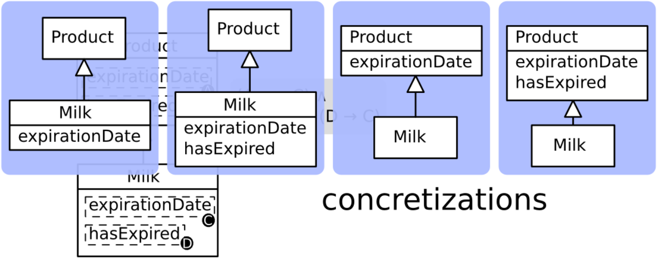
Intuition

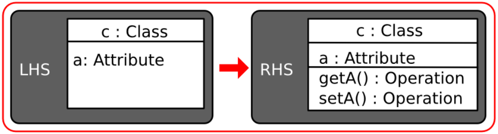


$(A \leftrightarrow \neg C) \land$
$(B \rightarrow A) \land (D \rightarrow C)$

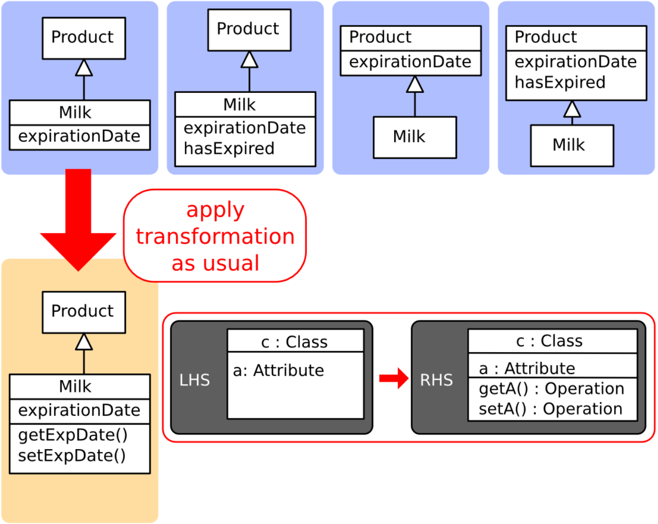


Intuition

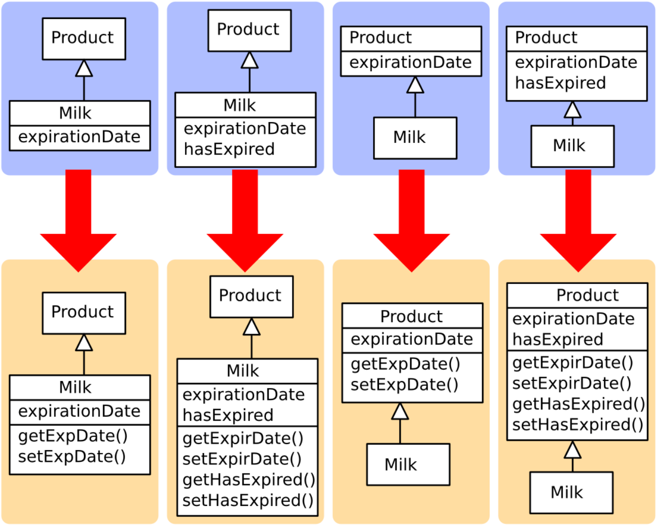

Product
Milk
expirationDate
Product
Milk
expirationDate
hasExpired
Product
expirationDate
Milk
Product
expirationDate
hasExpired
Milk
Milk
expirationDate
hasExpired
concretizations


LHS
c : Class
a: Attribute
RHS
c : Class
a : Attribute
getA() : Operation
setA() : Operation

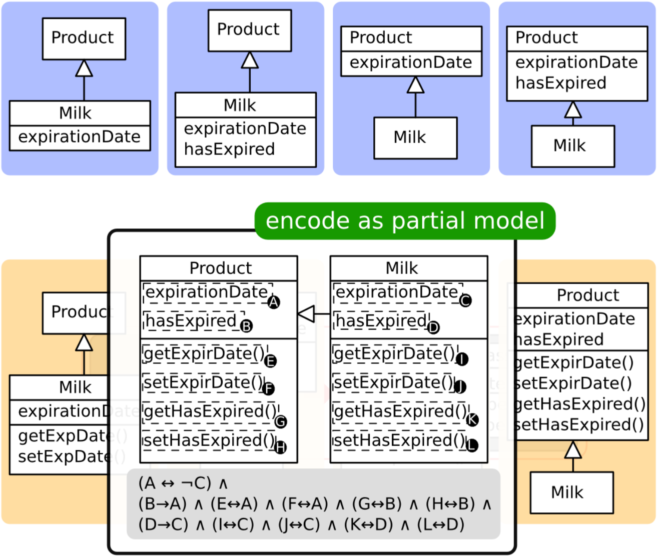
Intuition


Product
Milk
expirationDate
Product
Milk
expirationDate
hasExpired
Product
expirationDate
Milk
Product
expirationDate
hasExpired
Milk
apply transformation as usual
Product
Milk
expirationDate
getExpDate()
setExpDate()
LHS
c : Class
a: Attribute
RHS
c : Class
a : Attribute
getA() : Operation
setA() : Operation

Intuition


Product
Milk
expirationDate
Product
Milk
expirationDate
hasExpired
Product
expirationDate
Milk
Product
expirationDate
hasExpired
Milk
Product
Milk
expirationDate
getExpDate()
setExpDate()
Product
Milk
expirationDate
hasExpired
getExpirDate()
setExpirDate()
getHasExpired()
setHasExpired()
Product
expirationDate
getExpDate()
setExpDate()
Milk
Product
expirationDate
hasExpired
getExpirDate()
setExpirDate()
getHasExpired()
setHasExpired()
Milk

Intuition


Product
Milk
expirationDate
Product
Milk
expirationDate
hasExpired
Product
expirationDate
Milk
Product
expirationDate
hasExpired
Milk
encode as partial model
Product
expirationDate A
hasExpired B
getExpirDate() E
setExpirDate() F
getHasExpired() G
setHasExpired() H
Milk
expirationDate C
hasExpired D
getExpirDate() I
setExpirDate() J
getHasExpired() K
setHasExpired() L
(A ↔ ¬C) ∧
(B→A) ∧ (E↔A) ∧ (F↔A) ∧ (G↔B) ∧ (H↔B) ∧
(D→C) ∧ (I↔C) ∧ (J↔C) ∧ (K↔D) ∧ (L↔D)

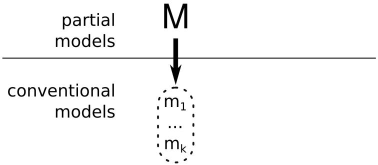
Our approach

Summarizing the intuition:

Applying a transformation to a partial model M

Our approach

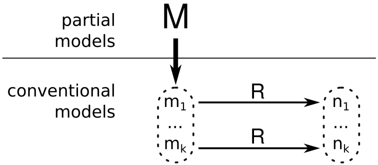
Summarizing the intuition:



Applying a transformation to a partial model M
should be the same as if
we had created all its concretizations,

Our approach

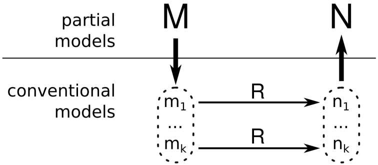
Summarizing the intuition:



Applying a transformation to a partial model M
should be the same as if
we had created all its concretizations,
applied the transformation to each separately,

Our approach

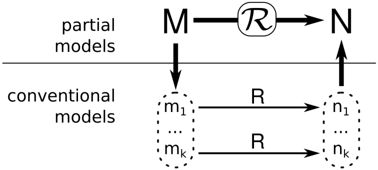
Summarizing the intuition:



Applying a transformation to a partial model M should be the same as if we had created all its concretizations, applied the transformation to each separately, and encoded the result as a partial model.

Our approach

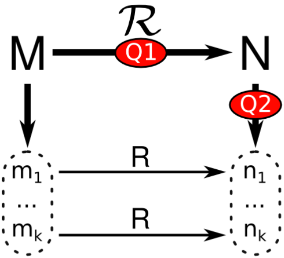
Summarizing the intuition:



Correctness Criterion

Applying a transformation to a partial model M should be the same as if we had created all its concretizations, applied the transformation to each separately, and encoded the result as a partial model.

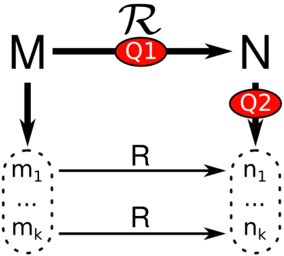
Lifting Transformations



Q1: How do we transform M directly to N?

Q2: Are the concretizations of N exactly the models $n_1 \dots n_k$?

Applying Rules to Partial Models



Q1: How do we transform M directly to N?

- *Lifted semantics* of transformations, using logic.

Q2: Are the concretizations of N exactly the models $n_1 \dots n_k$?

Transfer Predicates

Represent $M \overset{R\ *}{\Longrightarrow} N$ as:

$$\Phi_N = \mathcal{R}(R, M, N) \wedge \Phi_M$$

$\mathcal{R}$ is a conjunction $\phi_1 \wedge \phi_2 \wedge \ldots$

- One subformula at each application point:

$$(\Phi_{LHS} \rightarrow \Phi_{RHS}) \wedge (\neg\Phi_{LHS} \rightarrow \Phi_{NE})$$

Transfer Predicates

Represent $M \overset{R\ *}{\Longrightarrow} N$ as:

$$\Phi_N = \mathcal{R}(R, M, N) \wedge \Phi_M$$

$\mathcal{R}$ is a conjunction $\phi_1 \wedge \phi_2 \wedge \ldots$

- One subformula at each application point:

$$(\Phi_{LHS} \rightarrow \Phi_{RHS}) \wedge (\neg\Phi_{LHS} \rightarrow \Phi_{NE})$$

"Left hand side" of the rule; matching pattern.

Transfer Predicates

Represent $M \overset{R\ *}{\Longrightarrow} N$ as:

$$\Phi_N = \mathcal{R}(R, M, N) \wedge \Phi_M$$

$\mathcal{R}$ is a conjunction $\phi_1 \wedge \phi_2 \wedge \ldots$

- One subformula at each application point:

$$(\Phi_{\mathrm{LHS}} \rightarrow \Phi_{\mathrm{RHS}}) \wedge (\neg\Phi_{\mathrm{LHS}} \rightarrow \Phi_{\mathrm{NE}})$$

"Right hand side" of the rule; rule side effect.

Transfer Predicates

Represent $M \overset{R}{\Longrightarrow}{}^{*} N$ as:

$$\Phi_N = \mathcal{R}(R, M, N) \wedge \Phi_M$$

$\mathcal{R}$ is a conjunction $\phi_1 \wedge \phi_2 \wedge \ldots$

- One subformula at each application point:

$$(\Phi_{LHS} \rightarrow \Phi_{RHS}) \wedge (\neg\Phi_{LHS} \rightarrow \Phi_{NE})$$

If the rule matches, apply it.

Transfer Predicates

Represent $M \overset{R\ *}{\Longrightarrow} N$ as:

$$\Phi_N = \mathcal{R}(R, M, N) \wedge \Phi_M$$

$\mathcal{R}$ is a conjunction $\phi_1 \wedge \phi_2 \wedge \ldots$

- One subformula at each application point:

$$(\Phi_{\mathrm{LHS}} \rightarrow \Phi_{\mathrm{RHS}}) \wedge (\neg \Phi_{\mathrm{LHS}} \rightarrow \Phi_{\mathrm{NE}})$$

"Left hand side" of the rule; matching pattern.

Transfer Predicates

Represent $M \stackrel{R\,*}{\Longrightarrow} N$ as:

$$\Phi_N = \mathcal{R}(R, M, N) \wedge \Phi_M$$

$\mathcal{R}$ is a conjunction $\phi_1 \wedge \phi_2 \wedge \ldots$

- One subformula at each application point:

$$(\Phi_{LHS} \rightarrow \Phi_{RHS}) \wedge (\neg\Phi_{LHS} \rightarrow \Phi_{NE})$$

Rule did not match;
No-Op.

Transfer Predicates

Represent $M \overset{R\ *}{\Longrightarrow} N$ as:

$$\Phi_N = \mathcal{R}(R, M, N) \wedge \Phi_M$$

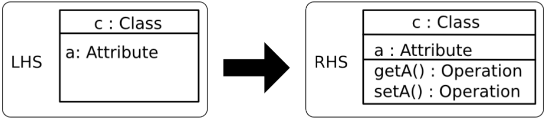
$\mathcal{R}$ is a conjunction $\phi_1 \wedge \phi_2 \wedge \ldots$

- One subformula at each application point:

$$(\Phi_{LHS} \rightarrow \Phi_{RHS}) \wedge (\neg\Phi_{LHS} \rightarrow \Phi_{NE})$$

If the rule does not match, do nothing.

Example 1/2



$$(\mathbf{\Phi}_{LHS} \rightarrow \mathbf{\Phi}_{RHS}) \wedge (\neg \mathbf{\Phi}_{LHS} \rightarrow \mathbf{\Phi}_{NE})$$

- $\mathbf{\Phi}_{LHS} = c \wedge a \wedge \neg g \wedge \neg s$
- $\mathbf{\Phi}_{RHS} = (c' \leftrightarrow c) \wedge (a' \leftrightarrow a) \wedge (g' \leftrightarrow a) \wedge (s' \leftrightarrow a)$
- $\mathbf{\Phi}_{NE} = (x' \leftrightarrow x)$

Example 2/2

After matching, grounding and simplifying:

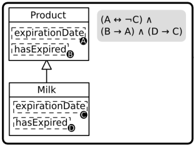
$$\begin{aligned}\mathcal{R}(\mathsf{R},\mathsf{M},\mathsf{N}) \quad &= (\texttt{Product}' \leftrightarrow \texttt{Product}) \wedge (\texttt{Milk}' \leftrightarrow \texttt{Milk}) \wedge \\ &(A' \leftrightarrow A) \wedge (B' \leftrightarrow B) \wedge (C' \leftrightarrow C) \wedge \\ &(D' \leftrightarrow D) \wedge (E' \leftrightarrow A) \wedge (F' \leftrightarrow A) \wedge \\ &(G' \leftrightarrow B) \wedge (H' \leftrightarrow B) \wedge (I' \leftrightarrow C) \wedge \\ &(J' \leftrightarrow C) \wedge (K' \leftrightarrow D) \wedge (L' \leftrightarrow D) \wedge \\ &(\texttt{gen\_Milk\_Product}' \leftrightarrow \texttt{gen\_Milk\_Product})\end{aligned}$$

$$\Phi_M \quad \wedge \quad \mathcal{R}(\mathsf{R},\ \mathsf{M}\ ,\ \ \mathsf{N}\ ) \quad = \quad \Phi_N$$

Example 2/2

After matching, grounding and simplifying:

$$\begin{aligned}\mathcal{R}(\mathrm{R},\mathrm{M},\mathrm{N}) \quad = & (\texttt{Product}' \leftrightarrow \texttt{Product}) \wedge (\texttt{Milk}' \leftrightarrow \texttt{Milk}) \wedge \\ & (A' \leftrightarrow A) \wedge (B' \leftrightarrow B) \wedge (C' \leftrightarrow C) \wedge \\ & (D' \leftrightarrow D) \wedge (E' \leftrightarrow A) \wedge (F' \leftrightarrow A) \wedge \\ & (G' \leftrightarrow B) \wedge (H' \leftrightarrow B) \wedge (I' \leftrightarrow C) \wedge \\ & (J' \leftrightarrow C) \wedge (K' \leftrightarrow D) \wedge (L' \leftrightarrow D) \wedge \\ & (\texttt{gen\_Milk\_Product}' \leftrightarrow \texttt{gen\_Milk\_Product})\end{aligned}$$

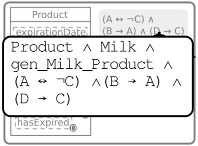


$$\wedge \quad \mathcal{R}(\mathrm{R},\mathrm{M},\mathrm{N}) \quad = \quad \Phi_{\mathrm{N}}$$

Example 2/2

After matching, grounding and simplifying:

$$\begin{aligned}\mathcal{R}(\mathsf{R},\mathsf{M},\mathsf{N}) \quad & = (\texttt{Product}' \leftrightarrow \texttt{Product}) \wedge (\texttt{Milk}' \leftrightarrow \texttt{Milk}) \wedge \\ & (A' \leftrightarrow A) \wedge (B' \leftrightarrow B) \wedge (C' \leftrightarrow C) \wedge \\ & (D' \leftrightarrow D) \wedge (E' \leftrightarrow A) \wedge (F' \leftrightarrow A) \wedge \\ & (G' \leftrightarrow B) \wedge (H' \leftrightarrow B) \wedge (I' \leftrightarrow C) \wedge \\ & (J' \leftrightarrow C) \wedge (K' \leftrightarrow D) \wedge (L' \leftrightarrow D) \wedge \\ & (\texttt{gen\_Milk\_Product}' \leftrightarrow \texttt{gen\_Milk\_Product})\end{aligned}$$

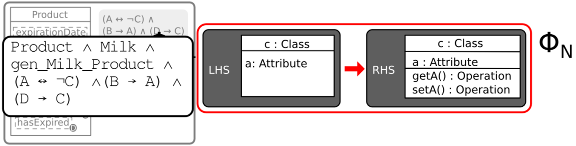


$$\wedge \quad \mathcal{R}(\mathsf{R},\mathsf{M},\mathsf{N}) \quad = \quad \Phi_{\mathsf{N}}$$

Example 2/2

After matching, grounding and simplifying:

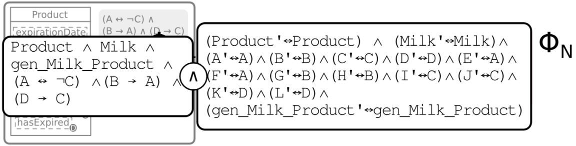
$$\begin{aligned}\mathcal{R}(\mathsf{R},\mathsf{M},\mathsf{N}) \quad &= (\texttt{Product}' \leftrightarrow \texttt{Product}) \wedge (\texttt{Milk}' \leftrightarrow \texttt{Milk}) \wedge \\ &(A' \leftrightarrow A) \wedge (B' \leftrightarrow B) \wedge (C' \leftrightarrow C) \wedge \\ &(D' \leftrightarrow D) \wedge (E' \leftrightarrow A) \wedge (F' \leftrightarrow A) \wedge \\ &(G' \leftrightarrow B) \wedge (H' \leftrightarrow B) \wedge (I' \leftrightarrow C) \wedge \\ &(J' \leftrightarrow C) \wedge (K' \leftrightarrow D) \wedge (L' \leftrightarrow D) \wedge \\ &(\texttt{gen\_Milk\_Product}' \leftrightarrow \texttt{gen\_Milk\_Product})\end{aligned}$$



Example 2/2

After matching, grounding and simplifying:

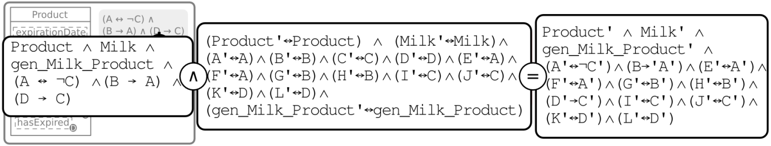
$$\begin{aligned}\mathcal{R}(\mathsf{R},\mathsf{M},\mathsf{N}) \quad &= (\mathtt{Product}' \leftrightarrow \mathtt{Product}) \wedge (\mathtt{Milk}' \leftrightarrow \mathtt{Milk})\wedge \\ &(A' \leftrightarrow A) \wedge (B' \leftrightarrow B) \wedge (C' \leftrightarrow C)\wedge \\ &(D' \leftrightarrow D) \wedge (E' \leftrightarrow A) \wedge (F' \leftrightarrow A)\wedge \\ &(G' \leftrightarrow B) \wedge (H' \leftrightarrow B) \wedge (I' \leftrightarrow C)\wedge \\ &(J' \leftrightarrow C) \wedge (K' \leftrightarrow D) \wedge (L' \leftrightarrow D)\wedge \\ &(\mathtt{gen\_Milk\_Product}' \leftrightarrow \mathtt{gen\_Milk\_Product})\end{aligned}$$



Example 2/2

After matching, grounding and simplifying:

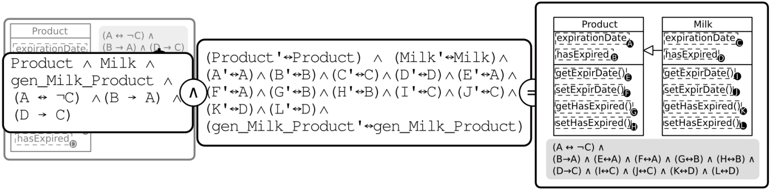
$$\begin{aligned}\mathcal{R}(\mathsf{R},\mathsf{M},\mathsf{N}) &= (\texttt{Product}' \leftrightarrow \texttt{Product}) \wedge (\texttt{Milk}' \leftrightarrow \texttt{Milk}) \wedge \\ & (A' \leftrightarrow A) \wedge (B' \leftrightarrow B) \wedge (C' \leftrightarrow C) \wedge \\ & (D' \leftrightarrow D) \wedge (E' \leftrightarrow A) \wedge (F' \leftrightarrow A) \wedge \\ & (G' \leftrightarrow B) \wedge (H' \leftrightarrow B) \wedge (I' \leftrightarrow C) \wedge \\ & (J' \leftrightarrow C) \wedge (K' \leftrightarrow D) \wedge (L' \leftrightarrow D) \wedge \\ & (\texttt{gen\_Milk\_Product}' \leftrightarrow \texttt{gen\_Milk\_Product})\end{aligned}$$



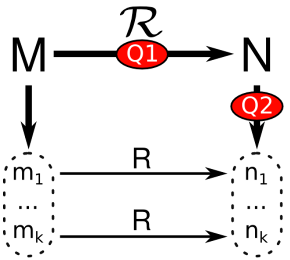
Example 2/2

After matching, grounding and simplifying:

$$
\begin{aligned}
\mathcal{R}(\mathsf{R},\mathsf{M},\mathsf{N}) \quad = & (\texttt{Product}' \leftrightarrow \texttt{Product}) \wedge (\texttt{Milk}' \leftrightarrow \texttt{Milk}) \wedge \\
& (A' \leftrightarrow A) \wedge (B' \leftrightarrow B) \wedge (C' \leftrightarrow C) \wedge \\
& (D' \leftrightarrow D) \wedge (E' \leftrightarrow A) \wedge (F' \leftrightarrow A) \wedge \\
& (G' \leftrightarrow B) \wedge (H' \leftrightarrow B) \wedge (I' \leftrightarrow C) \wedge \\
& (J' \leftrightarrow C) \wedge (K' \leftrightarrow D) \wedge (L' \leftrightarrow D) \wedge \\
& (\texttt{gen\_Milk\_Product}' \leftrightarrow \texttt{gen\_Milk\_Product})
\end{aligned}
$$



Testing Rule Application



Q1: How do we transform M directly to N?

Q2: Are the concretizations of N exactly the models $n_1 \ldots n_k$?

– Check equivalence of encodings using SAT.

Checking Using a SAT Solver

$$\Phi_M \wedge \mathcal{R}(R,M,N) = \Phi_N$$

Checking Using a SAT Solver

$$\Phi_M \wedge \mathcal{R}(R,M,N) = \Phi_N$$

$$\Phi_M \xrightarrow{m_i} \Phi_{m1}$$

Checking Using a SAT Solver

$$\Phi_M \wedge \mathcal{R}(R,M,N) = \Phi_N$$

$\Phi_M \xrightarrow{m_i}$

Φ_{m1}

...

Φ_{mk}

Checking Using a SAT Solver

$$\Phi_M \wedge \mathcal{R}(R,M,N) = \Phi_N$$

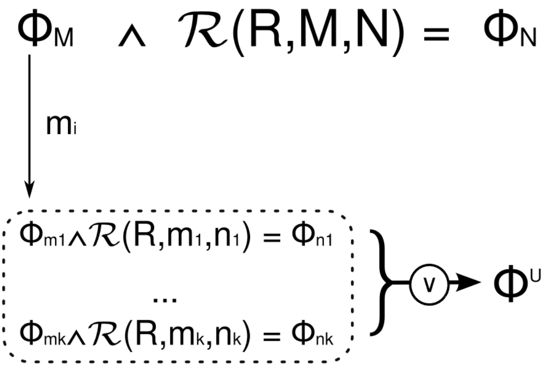
$$\Big\downarrow m_i$$

$$\Phi_{m1} \wedge \mathcal{R}(R,m_1,n_1) = \Phi_{n1}$$

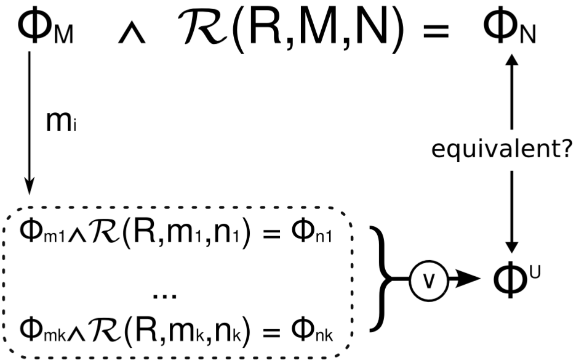
$$\ldots$$

$$\Phi_{mk} \wedge \mathcal{R}(R,m_k,n_k) = \Phi_{nk}$$

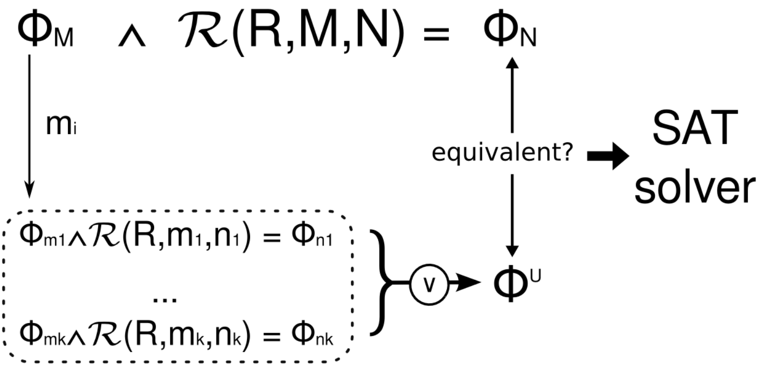
Checking Using a SAT Solver



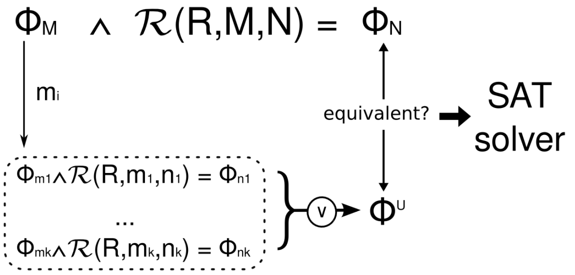
Checking Using a SAT Solver



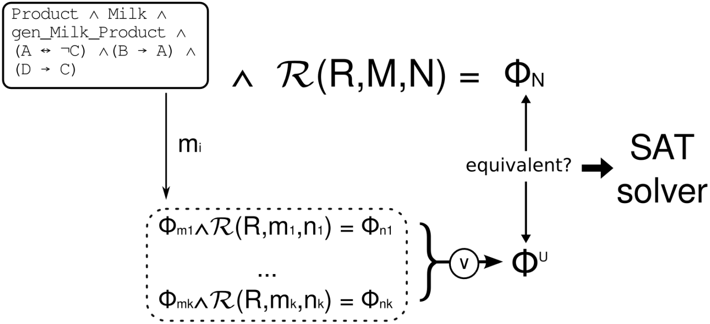
Checking Using a SAT Solver



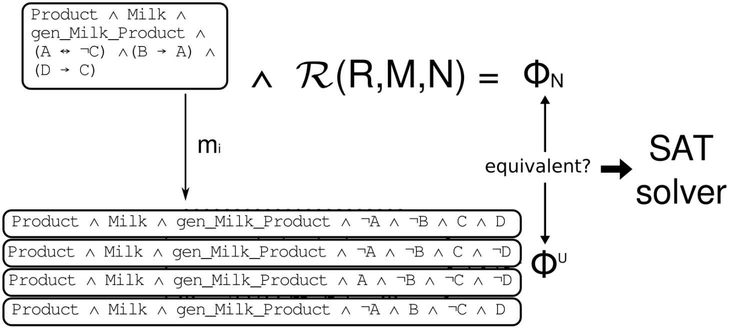
Checking Example



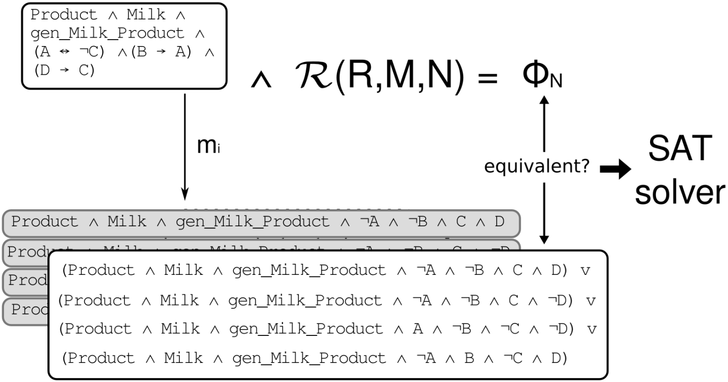
Checking Example



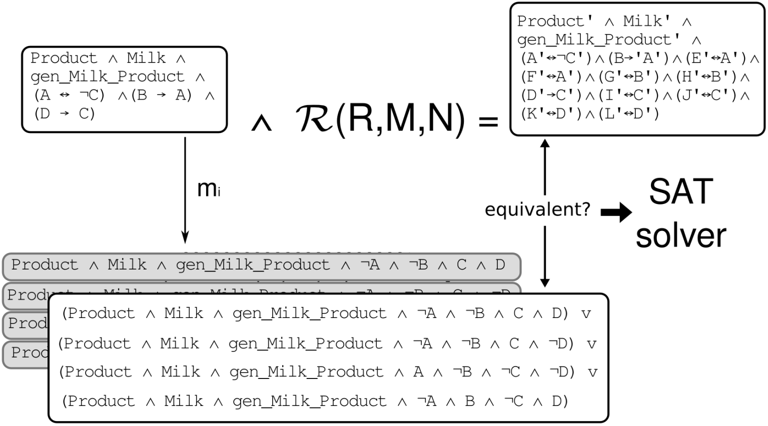
Checking Example



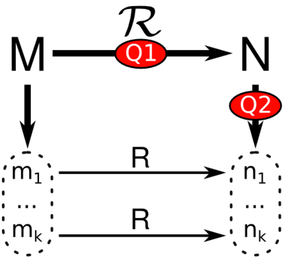
Checking Example



Checking Example



Summary



Q1: How do we transform M directly to N?

– *Lifted semantics* of transformations, using logic.

Q2: Are the concretizations of N exactly the models $n_1 \ldots n_k$?

– Check equivalence of encodings using SAT.

Conclusion

Transforming models that contain uncertainty.

- Represent uncertainty using Partial Models.
- Lift transformation rules from classical to Partial Models.
- Check Correctness Criterion for the lifted transformation .

Next steps:

- *Compositionally* test Correctness Criterion.
- Systematically create Transfer Predicates using FOL.
- Handle expanding/contracting model vocabularies.
- Partial Models as an Adhesive HLR Category?

Overall Picture

Overall research goal [MoDeVVa'11]:

- **Handling uncertainty...**
  - Partial models: *sets of possibilities*.
  - Syntactic "partiality" annotations.
  - Other kinds of partiality ("MAVO") [FASE'12].

- **...throughout the software lifecycle.**
  - Partial models *as first-class artifacts.*
    - (1) Reasoning [ICSE'12]
    - (2) Refinement [VOLT'12]
    - (3) Transformation

Questions?

Bibliography I

M. Famelis, Shoham Ben-David, Marsha Chechik, and Rick Salay.
"Partial Models: A Position Paper".
In *Proceedings of MoDeVVa'11*, pages 1–6, 2011.

Michalis Famelis, Marsha Chechik, and Rick Salay.
"Partial Models: Towards Modeling and Reasoning with Uncertainty".
In *Proceedings of ICSE'12*, 2012.

R. Salay, M. Chechik, and J. Gorzny.
"Towards a Methodology for Verifying Partial Model Refinements".
In *Proceedings of VOLT'12*, 2012.

R. Salay, M. Famelis, and M. Chechik.
"Language Independent Refinement using Partial Modeling".
In *Proceedings of FASE'12*, 2012.