

Termination Analysis of Model Transformations by Petri Nets

Dániel Varró, Szilvia Varró-Gyapay

(Budapest Univ. of Technology and Economics)

Hartmut Ehrig, Ulrike Prange, Gabi Taentzer

(TU Berlin)



Motivation

- Context: Analysis of model transformations
 - Model transformations are software artifacts → „buggy”
 - Goal: Guarantee that a transformation does not introduce defects into the target design
- Properties of interest
 - Termination
 - Confluence / Uniqueness
 - Preservation of semantics / Property preservation
- Practical relevance
 - QVT-related transformation languages are declarative
 - Compilation to GT rules (Rensink et al. – GTVMT 2006)
 - Own experience with MTF (IBM Alphaworks):
termination is not always straightforward

Budapest University of Technology and Economics TU Berlin 3

What is termination?

GT rule: DPO

- **Definition of Termination:**
 - A transformation sequence $G \Rightarrow^* H$ is *terminating* if no GT rule in the GTS is applicable to H any more.
 - A GTS is *terminating* if all transformation sequences are terminating for any start graph G
- **Causes of nontermination:**
 - arbitrarily large graphs / infinite graphs
 - cycles in the transformation
- **Problem:**
 - Termination of a GTS is *undecidable* in general
 - Restriction: GT rules specialized for model transformations
- **Common idea of proving termination:**
Show that some measure function is monotonely decreasing during execution

RHS „contains” at least one NAC

Termination Analysis of Model Transformations by Petri Nets ICGT 2006, Natal, Brazil

Budapest University of Technology and Economics TU Berlin 4

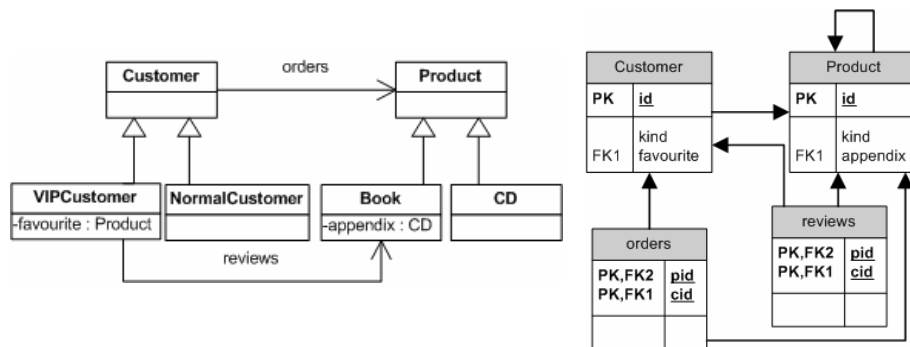
The Approach

- *A sufficient criteria* for the termination of a GTS is presented
 - **Yes**, the GTS is terminating
 - **Don't know**, the GTS may be non-terminating
- **Our approach:**
 1. Derive a Place/Transition Net (PN) abstraction of a GTS
 2. Prove that the PN is simulating the GTS
 3. Analyze the PN by algebraic techniques
 4. Prove that the PN runs out of tokens
 - in finitely many steps
 - regardless of the initial marking
 5. Conclude that the GTS is terminating due to simulation

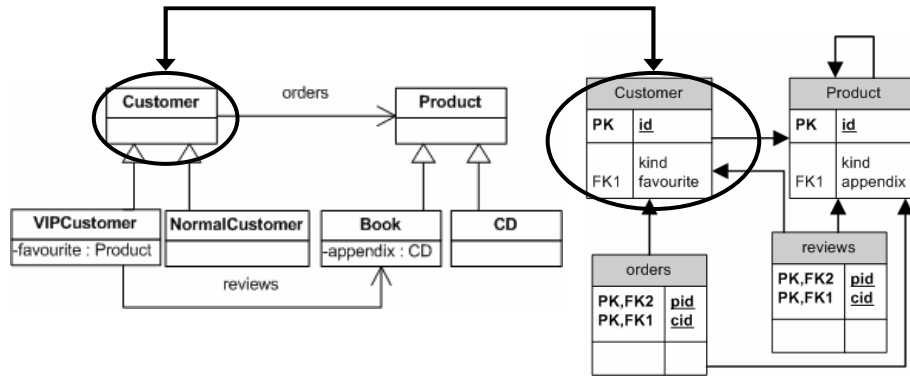
Termination Analysis of Model Transformations by Petri Nets ICGT 2006, Natal, Brazil

The Object-Relational Mapping as a sample model transformation problem

Object-Relational Mapping

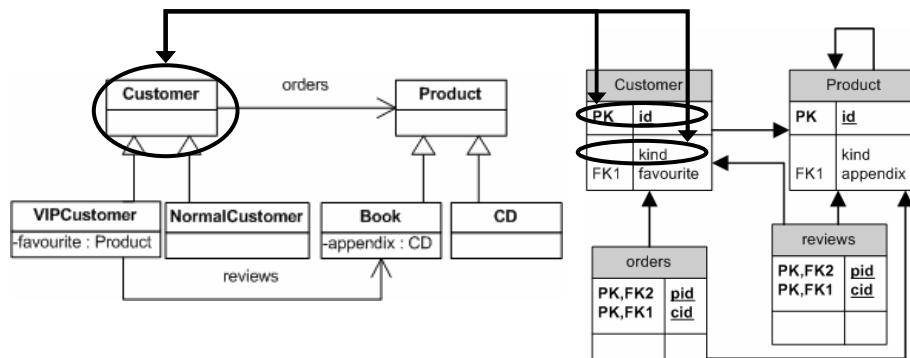


Informal rules of the transformation



Each top-level class is projected into a DB table

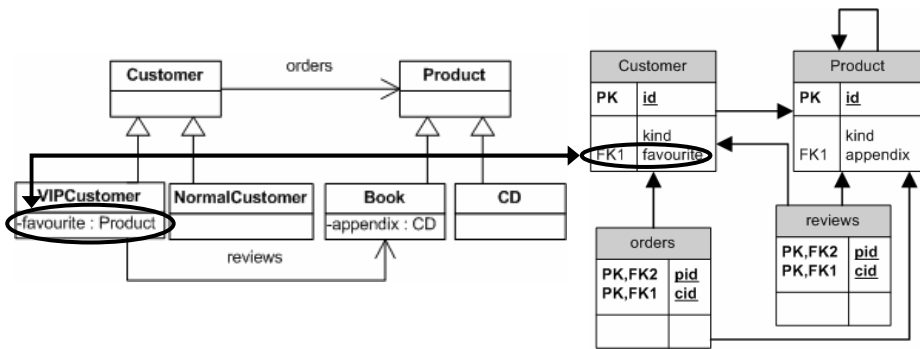
Informal rules of the transformation



Two additional columns are derived for each top-level class

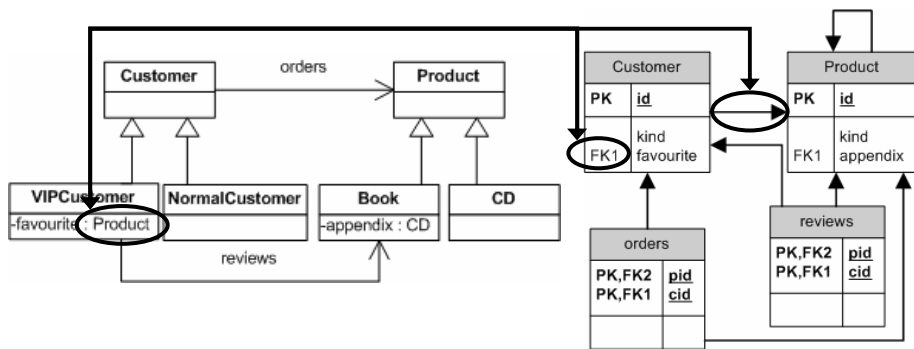
- a unique identifier (primary key),
- the type information of instances

Informal rules of the transformation



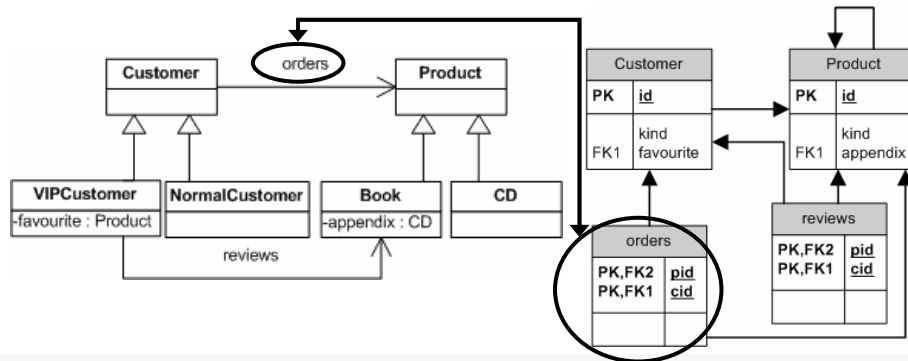
Each attribute of a class will appear as columns in the table related to the top-level ancestor of the class
 (The type of an attribute is restricted to user defined classes)

Informal rules of the transformation



The structural consistency of valid instances in columns is maintained by foreign key constraints

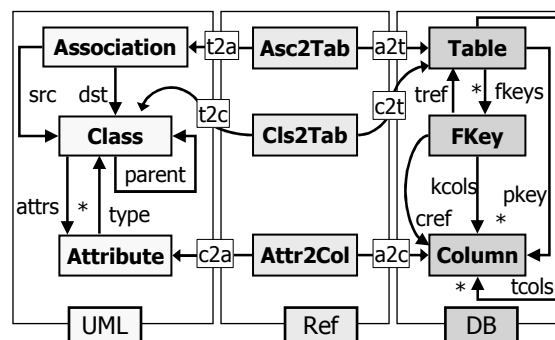
Informal rules of the transformation



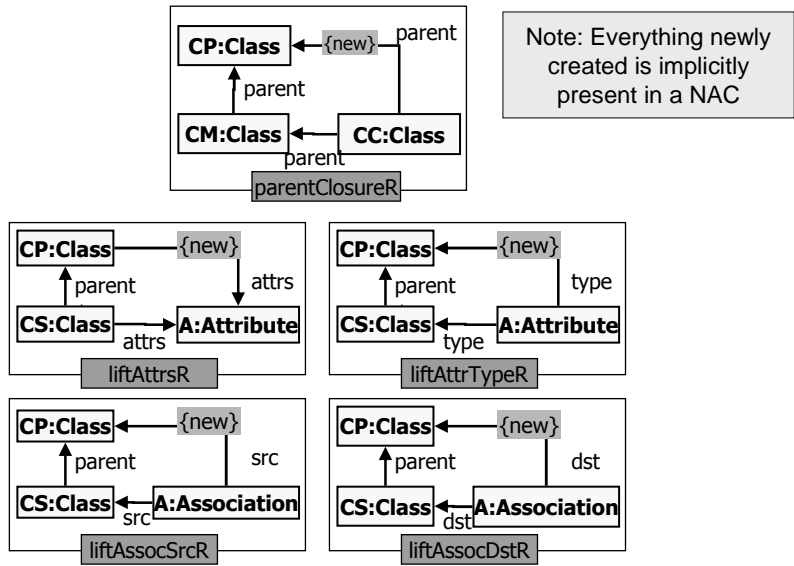
Each association is projected into a table with two columns

- pointing to the tables related to the source and the target classes
- consistency insured by foreign key constraints

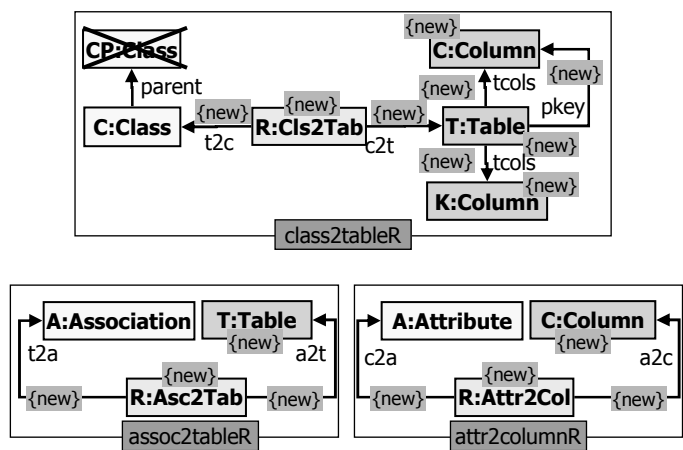
Metamodels of the problem

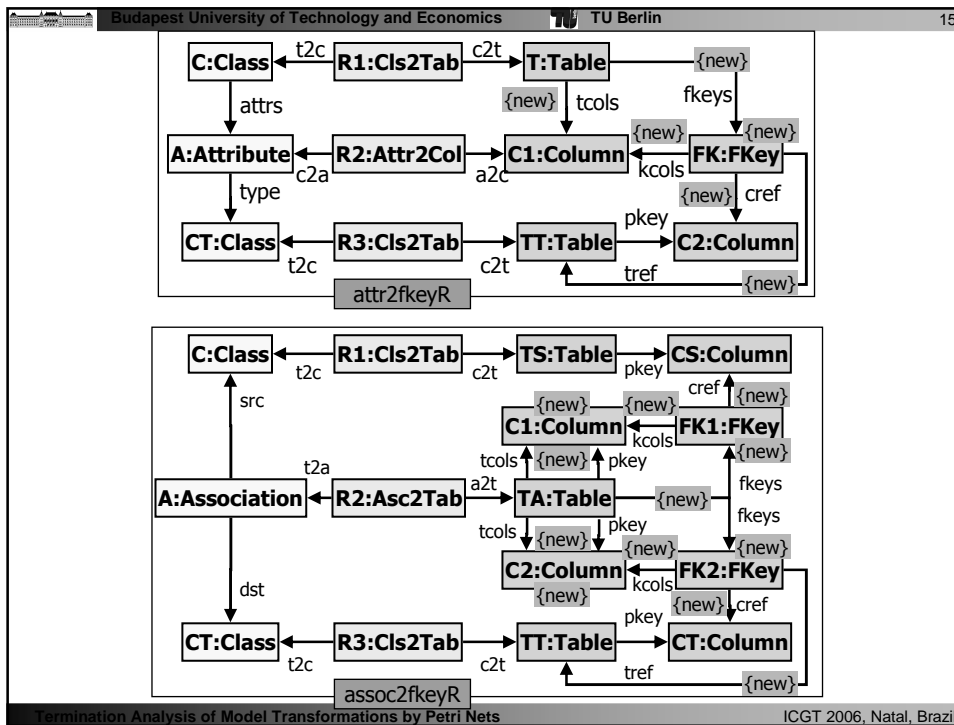


Preprocessing rules



Model transformation rules





Budapest University of Technology and Economics TU Berlin 16

Overview of Petri nets

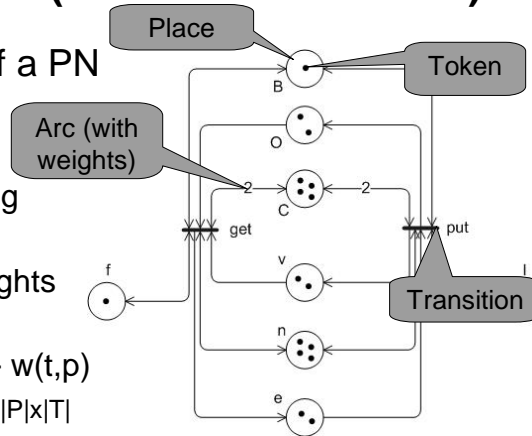
Termination Analysis of Model Transformations by Petri Nets ICGT 2006, Natal, Brazil

What is a Petri net? (Place/Transition net)

■ Firing a transition of a PN

- Remove tokens from incoming places
- Add tokens to outgoing places
- As defined by arc weights
- Formally:

$$M(p)' = M(p) - w(p,t) + w(t,p)$$



■ Incidence matrix: $W^{|P| \times |T|}$

describes the token flow when firing a transition

- $W_{i,j} = w(t_i, p_j) - w(p_j, t_i)$

W	f	B	O	C	v	n	e	l
get	0	0	-1	0	1	0	-1	0
put	0	0	1	0	-1	0	1	0

State equation of Petri nets

- Transition sequence: $s = M_0 \langle t_1, t_2, t_1, \dots, t_k \rangle M_k$
- Transition firing (Parikh) vector: σ
 count the number of occurrences of individual transitions
- State equation: $M_k = M_0 + W \cdot \sigma$

(Partial) Repetitiveness

- **Repetitiveness:** A Petri net is *partially repetitive* if
 - there exists a marking M_0 and
 - a firing sequence s from M_0 such that
 - some transition occurs infinitely many times in s .
- **Theorem:**
 - A Petri net with the incidence matrix W is partially repetitive

↑

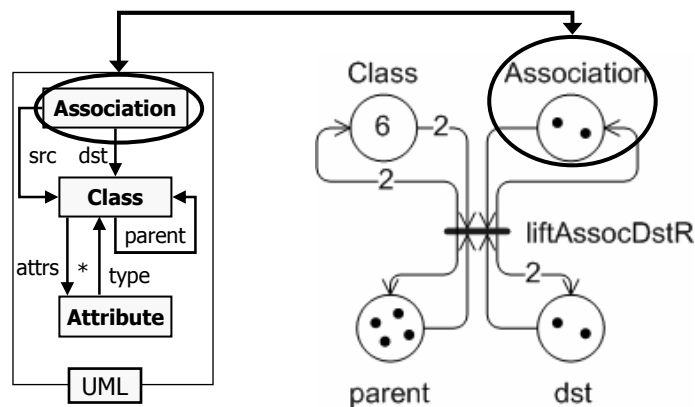
 - there exists a Parikh-vector $\sigma \geq 0$, $\sigma \neq 0$ such that $W^T \cdot \sigma \geq 0$.
- **Corollary:**
Non-repetitive P/T net has only finite firing sequences
→ terminating

A Petri net abstraction of GTSSs

Cardinality P/T nets

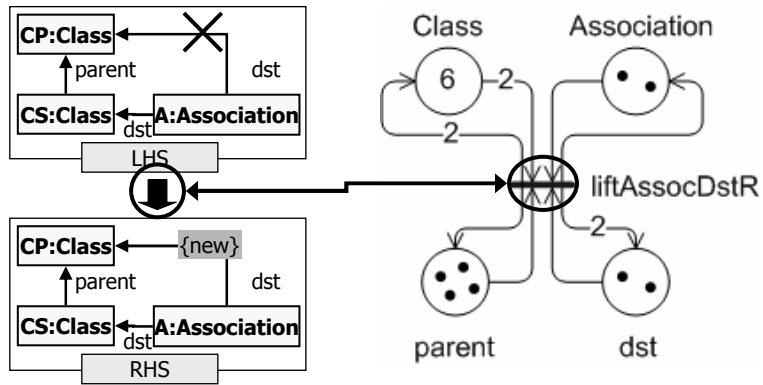
- General idea:
 - Abstract from the concrete graph structure
 - Keep track only the number of graph elements of a certain type
 - No negative conditions for the moment
- Cardinality P/T net of a GTS: $PN = F(GTS)$
 - types \Rightarrow places: one place for each node and edge in the type graph
 - instances \Rightarrow tokens: there are as many tokens in a place as the number of the instances from the corresponding type in the model graph
 - rules \Rightarrow transitions: one transition for each graph transformation rule
 - input places: for each item (node or edge) in the LHS there is an arc from their type place to the transition
 - output places: for each item (node or edge) in the RHS there is an arc from the transition to their type place

Example: Cardinality P/T net



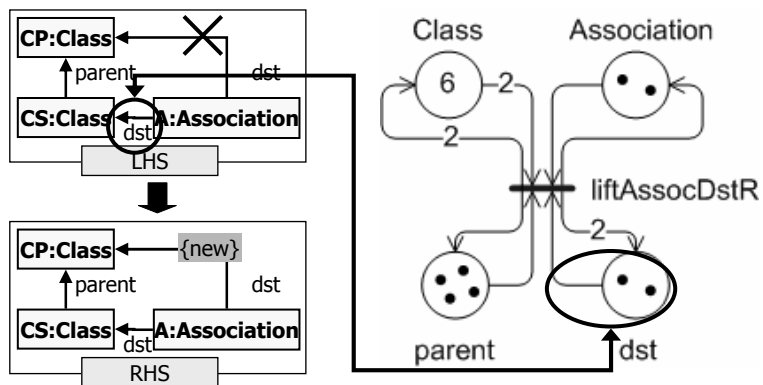
Types \Rightarrow Places:
 one place for each node and edge in the type graph

Example: Cardinality P/T net



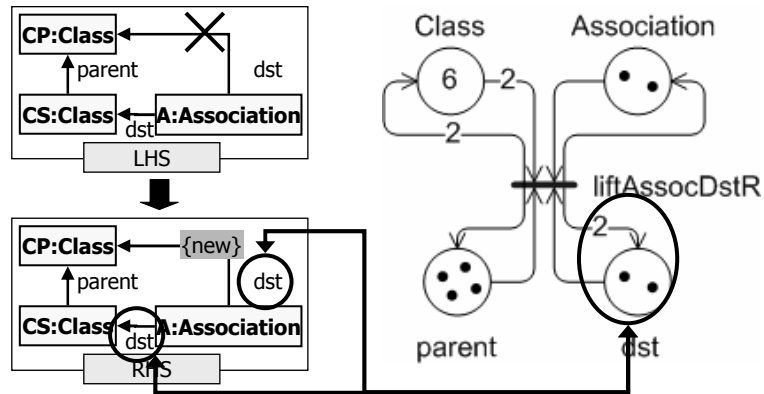
Rules \Rightarrow Transitions:
one transition for each graph transformation rule

Example: Cardinality P/T net



Input places: for each item (node or edge) in the LHS
there is an arc from their type place to the transition

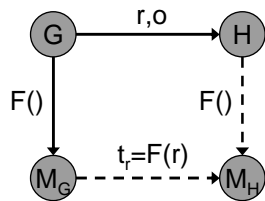
Example: Cardinality P/T net



Output places: for each item (node or edge) in the RHS, there is an arc from the transition to their type place

Termination analysis of GTS by P/T nets

- **Theorem: P/T net simulates the GTS.**



- **Theorem: Termination analysis of GTS.**
 - Given a graph transformation system *GTS* without NAC,
 - if its cardinality P/T net $PN = F(GTS)$ is non-repetitive \rightarrow
 - then *GTS* is terminating.

Termination analysis of GTS by P/T nets

- **Theorem: P/T net simulates the GTS.**
 - Whenever a rewriting step is executed in the GTS on an instance graph,
 - then the corresponding transition can always be fired in the corresponding marking in the P/T net, and
 - the result marking is an abstraction of the result graph.
- **Not a bisimulation:** no information about the connection of the certain nodes and edges
- **Theorem: Termination analysis of GTS.**
 - Given a graph transformation system *GTS without NAC*,
 - if its cardinality P/T net $PN = F(GTS)$ is non-repetitive →
 - then *GTS* is terminating.

Termination analysis of GTS with NAC

Problem with the current solution

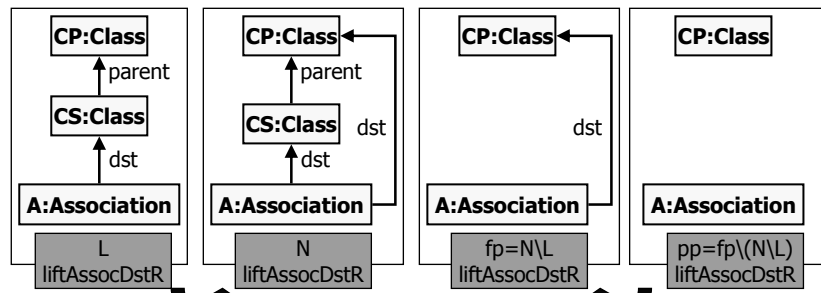
- **Problem:**
(almost) all MT rules have NACs which prohibit the application of a rule on the same matching
- **First idea:**
Since the use of NACs may restrict the application of a rule to an instance graph, the cardinality P/T net also simulates the GTS in case of NACs as well.
 - Unfortunately: the corresponding P/T net is typically partially repetitive \Rightarrow many false alarms
- **Second idea:** P/T nets with inhibitor arcs
 - Unfortunately: analysis techniques of P/T nets rarely support inhibitor arcs \Rightarrow no automated analysis

Main idea: Permission places

- **Permission places**
 - One permission place for each NAC
 - Count how many times the GT rule can be applied to the instance graph such that the corresponding NAC is satisfied
- **Problem:** it is not possible to calculate the exact number of rule applications,
 - the number of tokens can be unbounded
 - which have to be put into a permission place after firing a transition (e.g. new matchings are created).
- **We define an *overapproximation* of the potential number of rule applications**

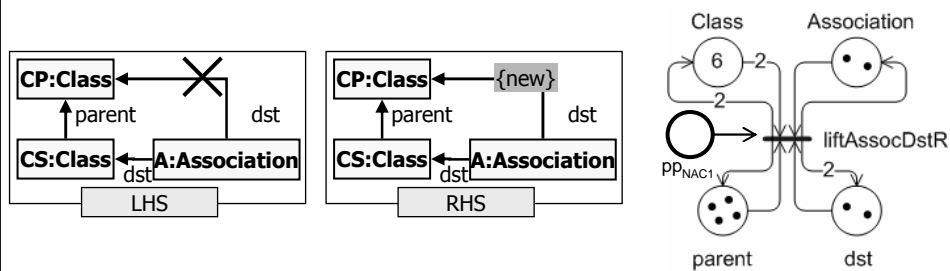
What do we count? Permission patterns

- Forbidden pattern:** fp^i (for a NAC N^i of a rule r)
 - the *context* of $n^i : L \rightarrow N^i$
 - the smallest subgraph of N^i that contains $N^i \setminus L$
- Permission pattern:** pp^i (Number of permissions)
 - the *boundary* of $n^i : L \rightarrow N^i$: defined as $fp^i \setminus (N^i \setminus L)$
 - „an LHS pattern having a NAC with the forbidden pattern ($fp^i \setminus (N^i \setminus L) \rightarrow fp^i$)



P/T net with permission places I.

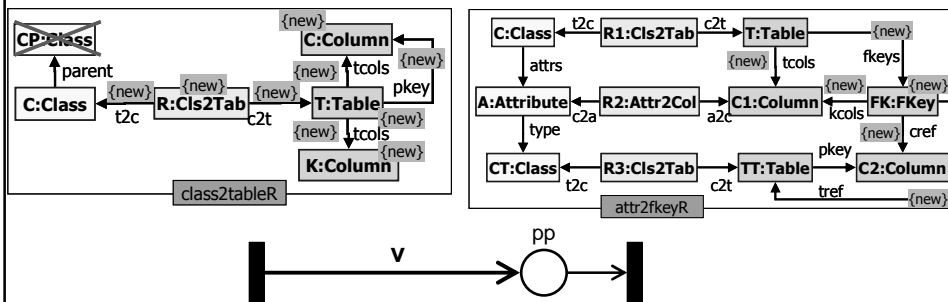
- Start graph**
 - The initial marking of permission places enable the firing of a transition
 - as many times as the permission pattern of GT rule can be matched
- Removing permissions (typical to MTs)**
 - If a GT rule is applied to a matching, and
 - the rule cannot be applied on the same matching twice due to a NAC
 - one token is taken from the permission place derived from the NAC



P/T net with permission places II.

Creating permissions

- If a GT rule r_1 generates a permission for another GT rule r_2 ,
- a finite (but unknown) number of tokens is put into all the corresponding permission places of the second rule r_2 ,
- by introducing arcs with variable weight v_k
- which depends on the actual graph: $W(G) = W(v_k)$



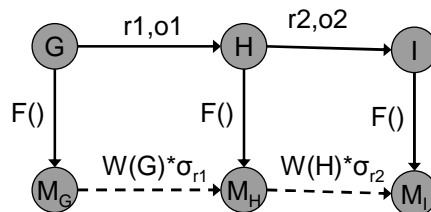
Further notes

Note 1: Permission ≠ Causality

Instead of regarding the causality between two rules based upon the RHS of rule r_1 and the LHS of r_2 , we define causality between the effects of a rule r_1 and the permission pattern of r_2 .

Note 2: Simulation after variable substitution

For each step of the P/T net, we can substitute the variables of arcs with proper values to simulate the original GTS in a step-wise way (defines $W(v)$).



Main results 1.

- Generalization of partial repetitiveness of PNs
 - Observe the state space without using W
 - W may change may change finitely in each step (only markings are involved, not W itself)
 - PN runs out of tokens due to local decreases

Theorem 1. *If for all infinite sequences $\{M_i\} = M_0, M_1, \dots$ of n -dimensional (state) vectors of nonnegative integer values with $M_j - M_{j-1} < \infty$ for all j*

$$(1) \quad \forall i, \forall j : j > i, M_i \not\equiv \underline{0} \Rightarrow \exists k : M_j[k] - M_i[k] < 0, \text{ and}$$

$$(2) \quad \forall i, \forall j : j > i, M_i \equiv \underline{0} \Rightarrow M_j \equiv \underline{0}$$

then $M \equiv \underline{0}$ in finitely many steps, i.e. $\exists s : M_s \equiv \underline{0}$ (where $M_j[k]$ denote component k in vector M_j).

Main results 2.

- Theorem: Termination analysis: GTS with NAC
 - Termination oracle: solve quadratic inequalities $W(\underline{v}) \cdot \underline{\sigma} \geq \underline{0}$ using the incidence matrix $W(\underline{v})$ with variables (e.g. GAMS toolkit)
 - If no solution found \rightarrow PN runs out of tokens \rightarrow GTS is terminating

Theorem 3 (Termination). *Let $W(\underline{v})$ be the incidence matrix of a cardinality P/T net $PN = \mathcal{F}_{pp}(GTS)$ derived as the abstraction of a GTS.*

If $\exists \underline{\sigma} \exists \underline{v} \quad W(\underline{v}) \cdot \underline{\sigma} \geq \underline{0}$ has no solutions with $\underline{v} \geq 1, \underline{\sigma} \geq \underline{0}, \underline{\sigma} \neq \underline{0}$ (thus $\forall \underline{\sigma} \forall \underline{v} \quad \exists k : (W(\underline{v}) \cdot \underline{\sigma})[k] < 0$), then GTS is terminating.

Conclusions

- Contributions:
 - A novel PN abstraction for analyzing GTSs
 - Simulation of GTS by abstracted PN
 - Generalization of partial repetitiveness for PNs
 - Sufficient criteria for the termination of GTS for MTs
- Future / Ongoing work
 - Implementation of the GTS2PN transformation
 - Solving the optimal trajectory problem
(see the talk at the PNGT workshop)

Thanks for your kind attention!

Analysis of the Example

- Derive the cardinality P/T net

	UML	Ref	DB	Permission places / NAC
	A C a s i A t t s a t s d t y p o s t r s r p a c s r c t s e r	t t c A C A a c a 2 2 2 2 2 2 2 2 2 a c a T T C t t c	f t k F t k p c c c T k C r e k o r o a e o e y e l e l b y l f s y s f s	p l i l l c c a l a a a l l a a a a r A t s s 2 2 s t t s C t T S D t t 2 2 2 l l t y r s b b t c f f o r p c t 1 2 b l k k
parentClosureR	0 0 0 0 0 0 0 1	0 0 0 0 0 0 0 0 0	0 0 0 0 0 0 0 0 0	-1 0 0 0 0 0 0 0 0 0
liftAttrR	0 0 0 0 0 1 0 0	0 0 0 0 0 0 0 0 0	0 0 0 0 0 0 0 0 0	0 -1 0 0 0 0 0 0 0 0
liftAttrTypeR	0 0 0 0 0 0 1 0	0 0 0 0 0 0 0 0 0	0 0 0 0 0 0 0 0 0	0 0 -1 0 0 0 0 0 0 0
liftAssocSrcR	0 0 0 1 0 0 0 0	0 0 0 0 0 0 0 0 0	0 0 0 0 0 0 0 0 0	0 0 0 -1 0 0 0 0 0 0
liftAssocDstR	0 0 0 0 1 0 0 0	0 0 0 0 0 0 0 0 0	0 0 0 0 0 0 0 0 0	0 0 0 0 -1 0 0 0 0 0
class2tableR	0 0 0 0 0 0 0 0	0 1 0 0 1 0 0 1 0	1 0 2 0 0 1 2 0 0	0 0 0 0 0 0 -1 0 0 ∞
assoc2tableR	0 0 0 0 0 0 0 0	1 0 0 1 0 0 1 0 0	1 0 0 0 0 0 0 0 0	0 0 0 0 0 0 0 -1 0 ∞
attr2columnR	0 0 0 0 0 0 0 0	0 0 1 0 0 1 0 0 1	0 0 1 0 0 0 0 0 0	0 0 0 0 0 0 0 0 -1 ∞
attr2keyR	0 0 0 0 0 0 0 0	0 0 0 0 0 0 0 0 0	0 1 0 1 1 0 1 1 1	0 0 0 0 0 0 0 0 0 -1 0
assoc2keyR	0 0 0 0 0 0 0 0	0 0 0 0 0 0 0 0 0	0 2 2 1 2 2 2 2 2	0 0 0 0 0 0 0 0 0 0 -1

- Solve the inequalities $W^T \cdot \sigma \geq 0$ with a symbolic optimization toolkit (e.g. GAMS)
 - The GTS is terminating

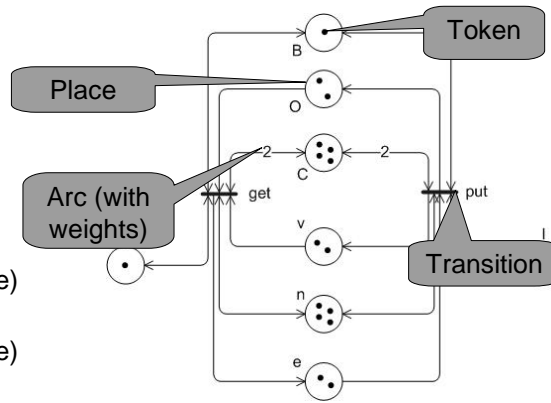
Incoming and outgoing arcs for permission places

- NACs into incoming arcs.
 - If everything included in the NAC N_i exists or is created by the RHS
 - an incoming arc is generated in the P/T net with arc weight 1
- Rule actions into outgoing arcs. For each pair of rules (r_1, r_2) , an outgoing arc with infinite weight is generated if
 - at least one graph item o is deleted by r_2 such that there exists a graph item $o_2 \in N_1 \setminus L_1$, and $type(o) = type(o_2)$ or
 - at least one graph item o is created by r_2 such that there exists a graph item $o_2 \in pp'_1 \setminus (N_1 \setminus L_1)$, and $type(o) = type(o_2)$.

What is a Petri net? (Place/Transition net)

Structure of PN

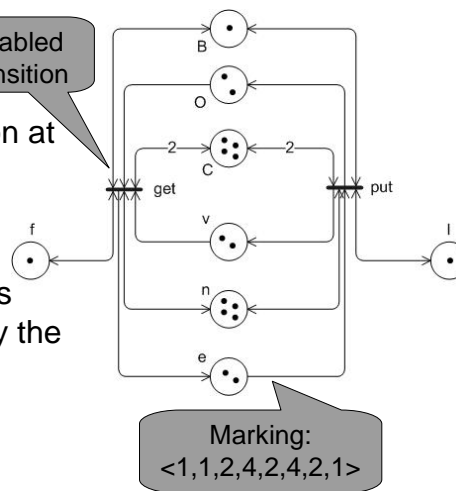
- Places
- Tokens
- Transitions
- Arcs (with weights)
 - Incoming arcs (of a transition / a place)
 - Outgoing arcs (of a transition / a place)



What is a Petri net? (Place/Transition net)

Behavior of PN

- **Marking of a PN:** $M(p)$
the current token distribution at each place
- **Enabled transition:**
all incoming places of the transition contain at least as many tokens as required by the weight of the incoming arc

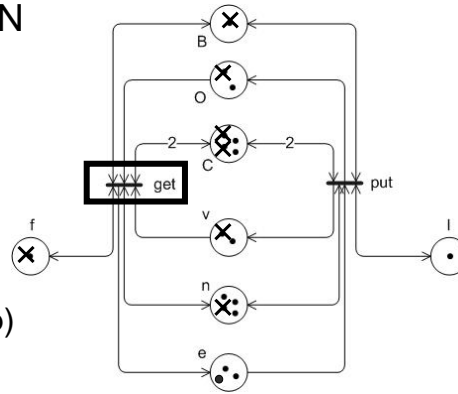


What is a Petri net? (Place/Transition net)

■ Firing a transition of a PN

- Remove tokens from incoming places
- Add tokens to outgoing places
- As defined by arc weights
- Formally:

$$M(p)' = M(p) - w(p,t) + w(t,p)$$



■ Incidence matrix: $W^{|P| \times |T|}$

describes the token flow when firing a transition

- $W_{i,j} = w(t_i, p_j) - w(p_j, t_i)$

W	f	B	O	C	v	n	e	l
get	0	0	-1	0	1	0	-1	0
put	0	0	1	0	-1	0	1	0