

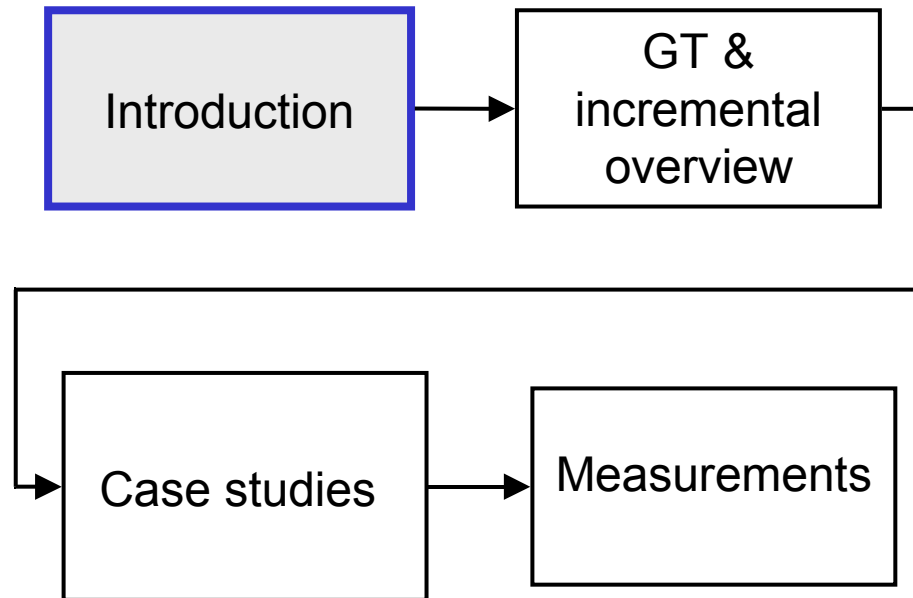
M Ű E G Y E T E M 1 7 8 2

# A Benchmark Evaluation for Incremental Pattern Matching in Graph Transformation

Gábor Bergmann, **Ákos Horváth**,  
István Ráth, Dániel Varró



# Talk Overview



# Benchmarking

## ■ Aim:

- systematic and reproducible measurements
- on performance
- under different and precisely defined circumstances

## ■ Overall goal:

- help transformation engineers in selecting tools
- Serve as reference for future research

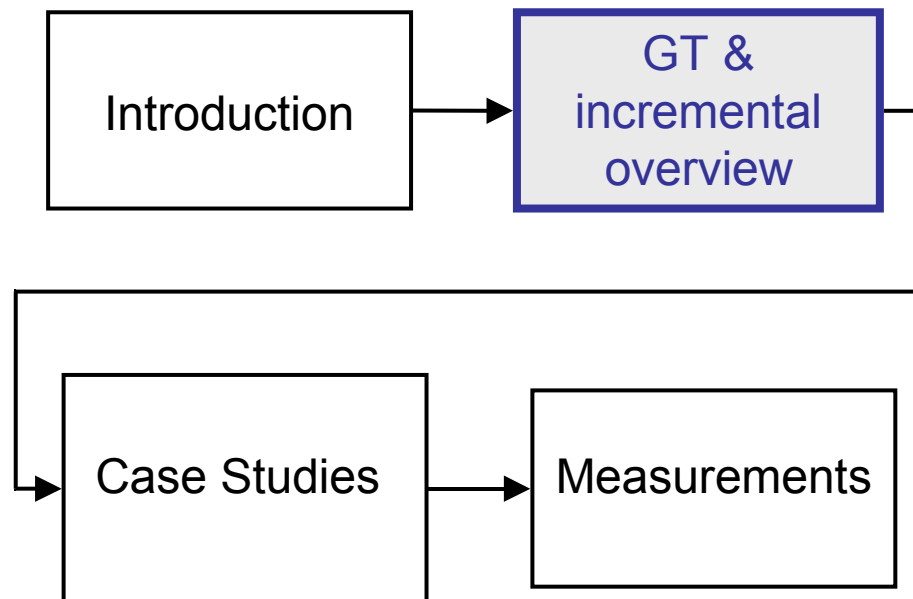
## ■ Popular approach in different fields

- AI
- relational databases
- rule-based expert systems

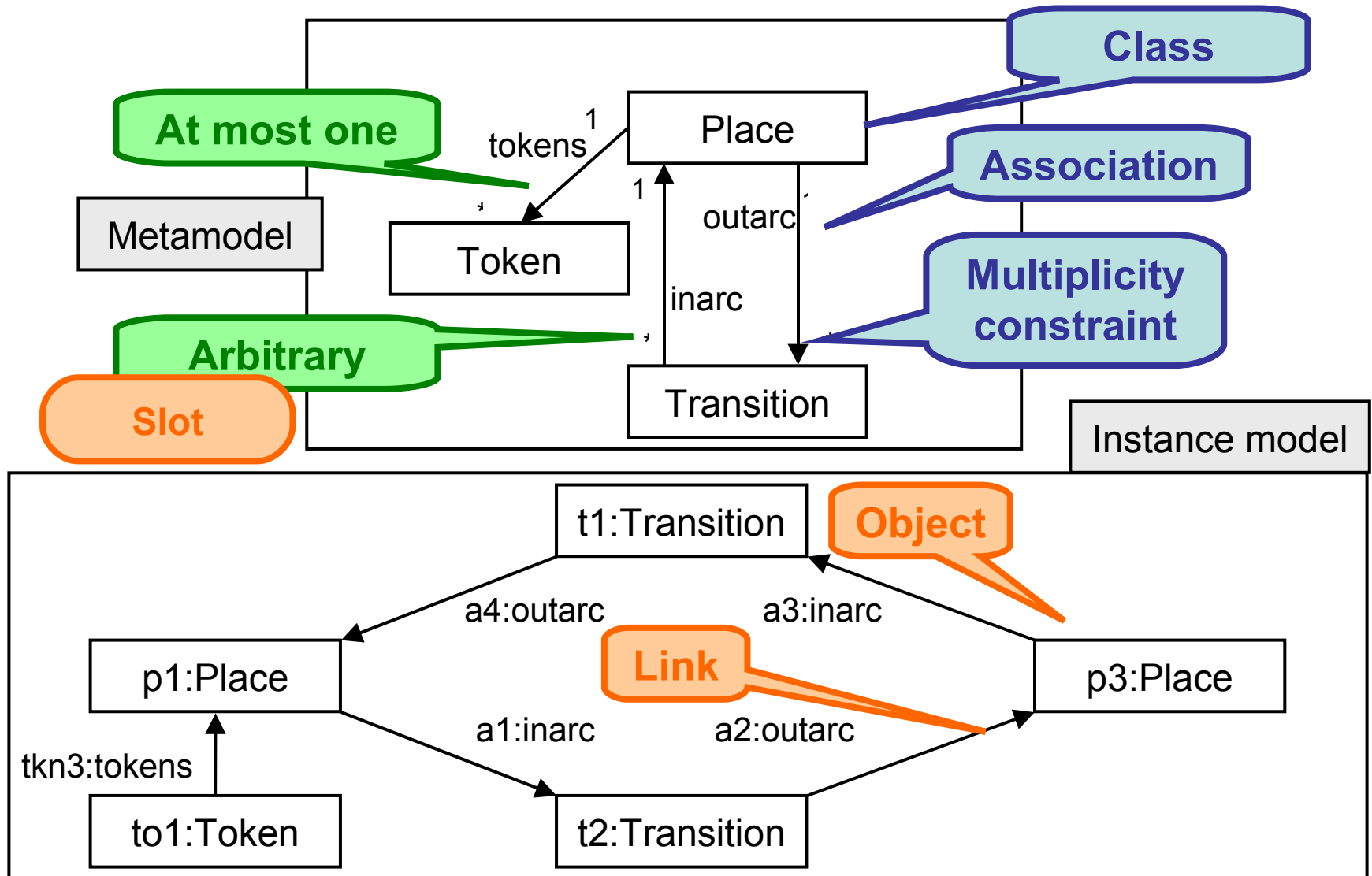
# Benchmarking in graph transformation

- Specification examples for GT
  - Goal: assessing expressiveness
  - UML-to-XMI, object-relational mapping, UML-to-EJB, etc.
- „Generic” Performance benchmarks for GT
  - Varró benchmark
  - R. Geiß and M. Kroll: On Improvements of the Varró Benchmark for Graph Transformation Tools
  - (Active Tool Contest, Grabats '08, ...)
- In our paper:  
**Benchmarks for graph transformation with incremental pattern matching**
  - simulation
  - synchronization

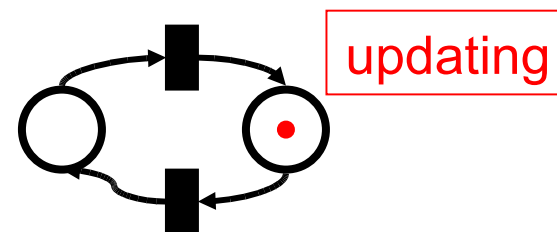
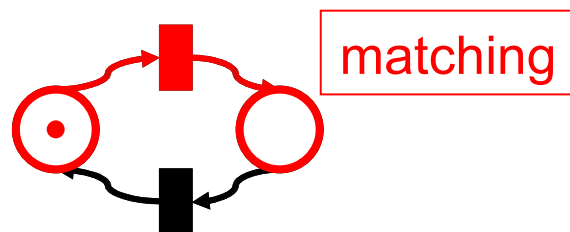
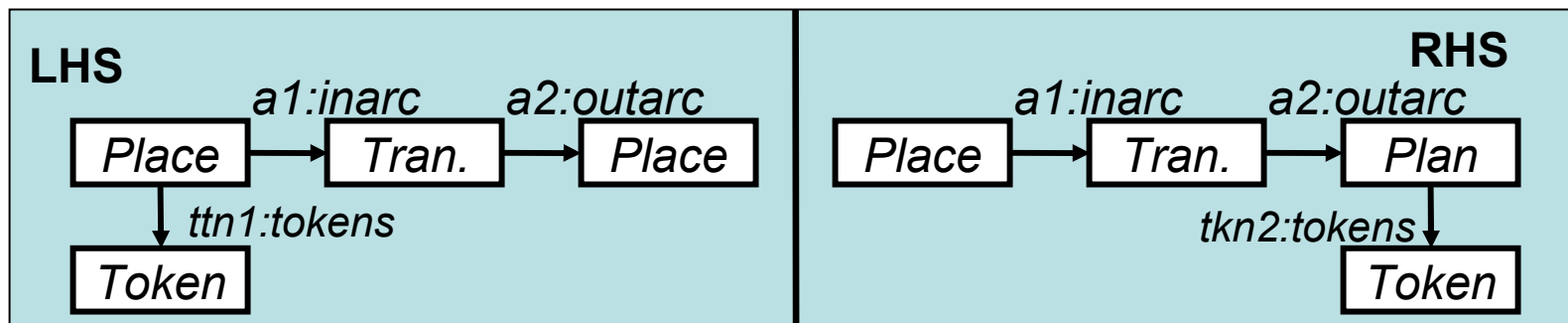
# Talk overview



# Metamodeling



# Graph Transformation



## Phases of GT matching

- Pattern Matching phase
- Updating phase: delete+ create

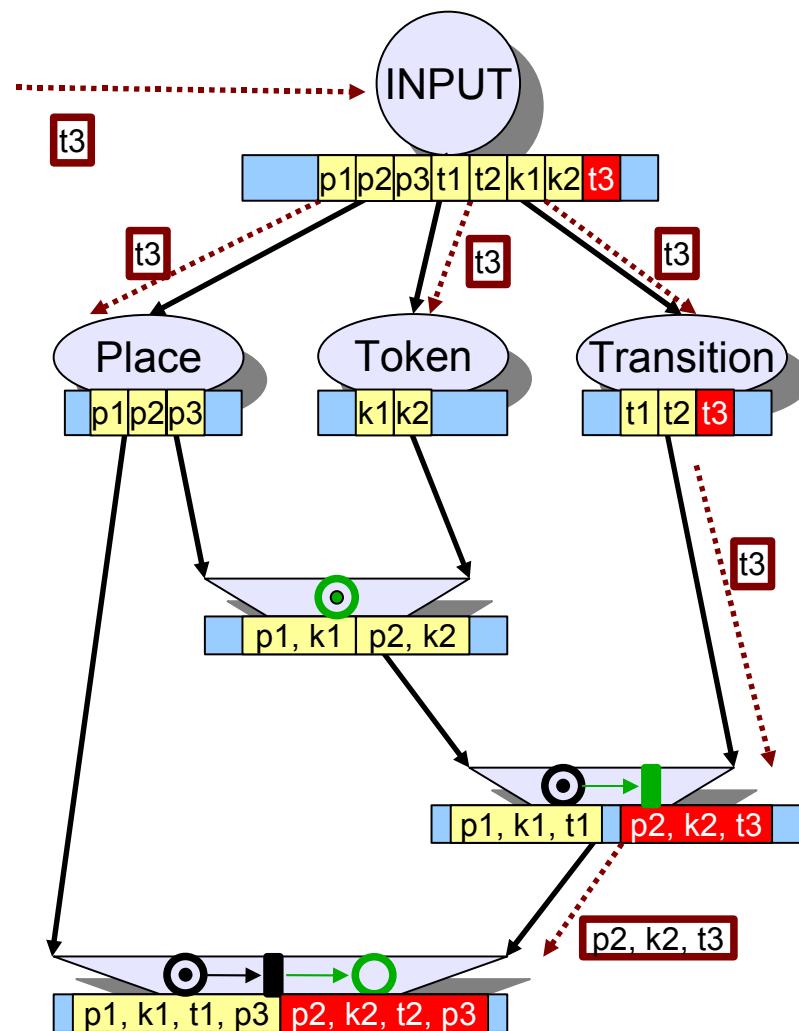
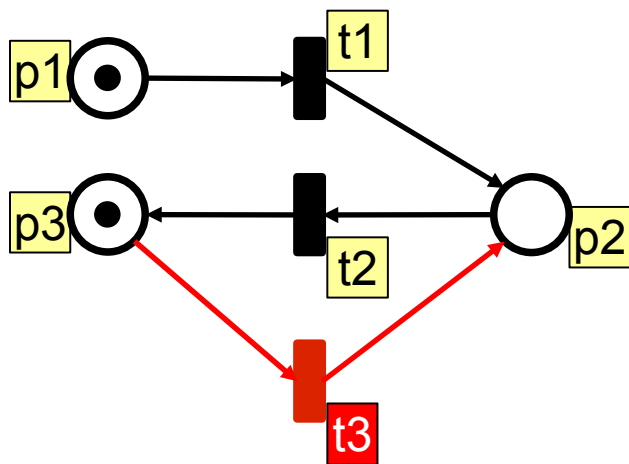
Pattern Matching is the **most critical issue** from **performance** viewpoint

# Incremental Pattern Matching

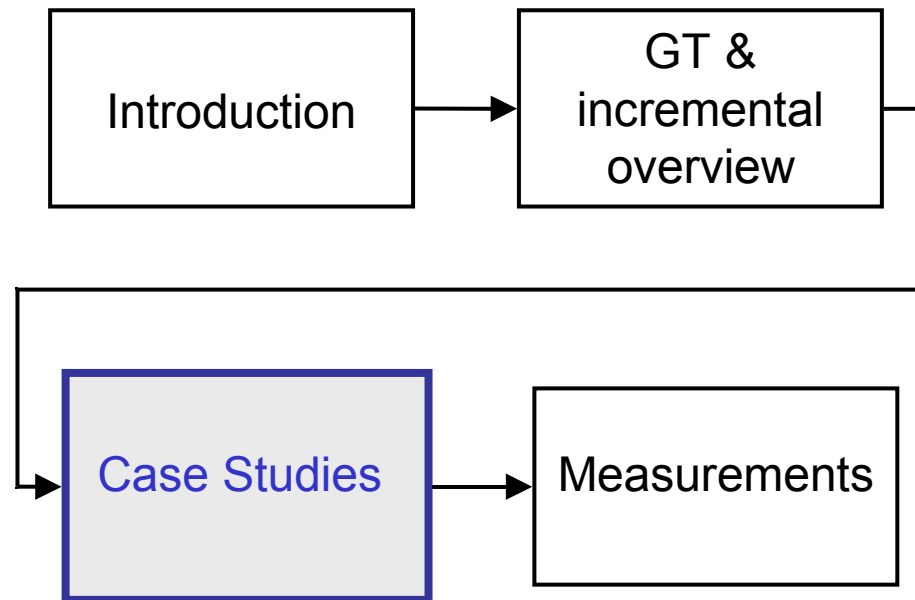
- Precondition of GT rules → pattern matching
- Goal
  - **Store matching sets**
  - Incremental update
  - Fast response
- Good performance expected when:
  - frequent pattern matching
  - Small updates
- Possible application domain
  - E.g. synchronization, constraints, model simulation, etc.
- An adapted RETE algorithm

# Incremental Pattern Matching Example

- RETE net
  - nodes: intermediate matchings
  - edge: update propagation
- Example
  - input: Petri net
  - pattern: firable transition
  - update: new transition



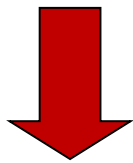
# Talk overview



# Test Cases

## ■ Model Simulation

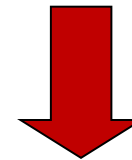
- Visual languages
- Static graph structure
- Small local model manipulation
- **ALAP execution**



Petri Net simulation

## ■ Synchronization

- Match reusability
- Unidirectional
- Complex rules
- Batch like execution

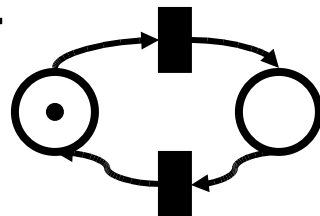


ORM synchronization

# Petri net Simulation

## ■ Test Case generation

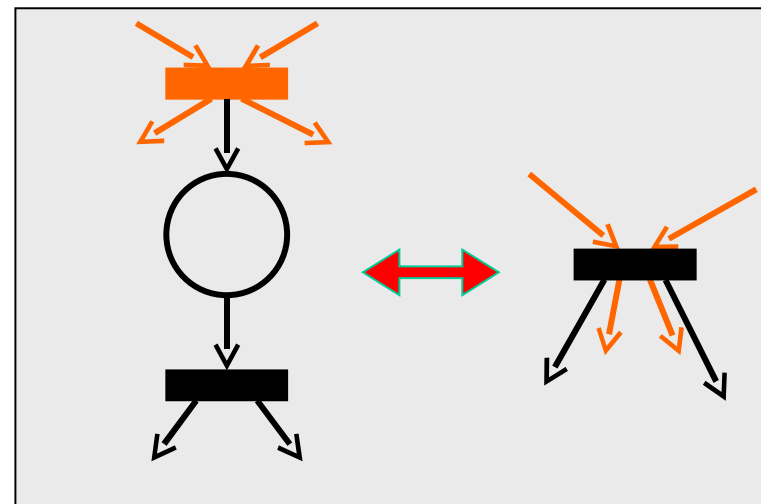
- Inverse property preservation reduction operation
- Random weighted select
- Few tokens
- Transition/Place  $\sim 1.1$
- Starting from:



## ■ Execution

- Random fireable transition selection
- 1000 / 1000000 iterations
- Measured:  
Total execution time

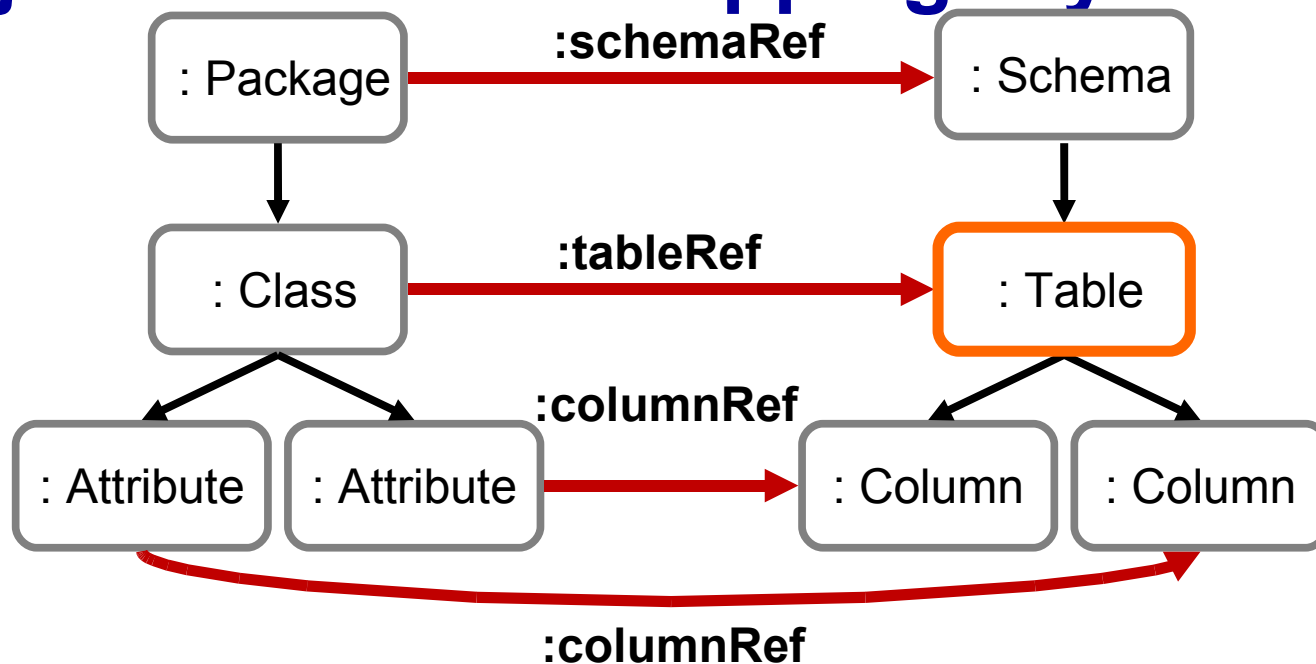
## ■ Sequential Transition reduction



## ■ Characteristics

Paradigm Features	Petri Net
LHS size	small
fan-out	small
matchings	PD
transformation sequence length	Small/ Long

# Object-Relational Mapping: Synchronization



## orphanTable

NEG

C: Class

: tableRef

OR

A: Association

: tableRef

{DEL}

T : Table

## Sync order

1. Orphan schema delete
2. Orphan Table → class delete
3. Orphan Table → assoc delete
4. Orphan Column → attr. Delete
5. Renaming
6. new Classes/Assocs/Columns created

# Object-Relational Mapping: Synchronization

## ■ Test Case generation

- Fully connected graph
- N classes
- $N(N-1)$  directed association
- K attributes

## ■ Source modification

- 1/3 classes deleted
- 1/5 associations deleted
- $\frac{1}{2}$  attributes renamed
- 1 new class added and the fully connected graph is rebuilt

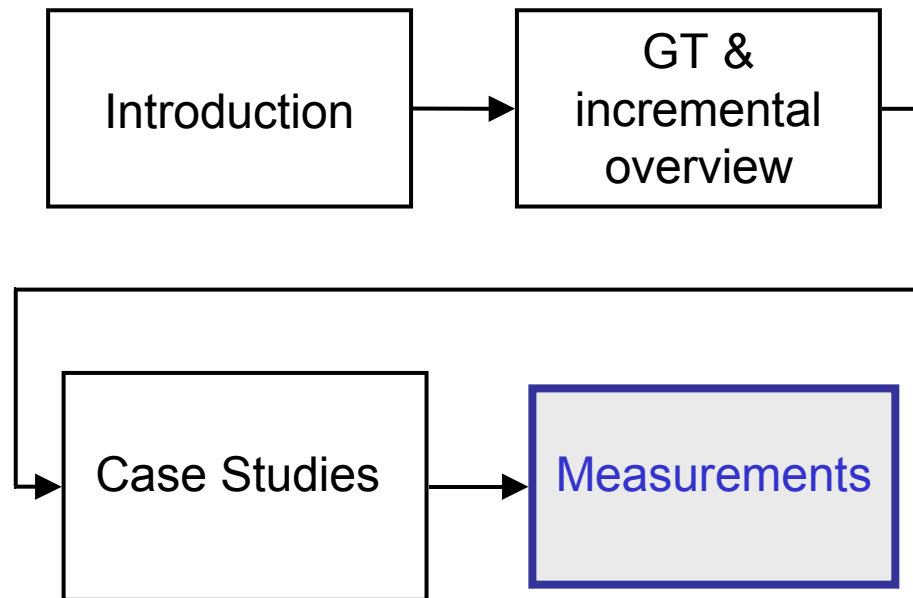
## ■ Execution

- Phases
  1. Generation
  2. Build
  3. Modification
  4. Synchronization
- Measured:  
Synchronization phase

## ■ Characteristic

Paradigm Features	ORM Syn.
LHS size	large
fan-out	medium
matchings	PD
transformation sequence length	PD

# Talk overview



# Environment

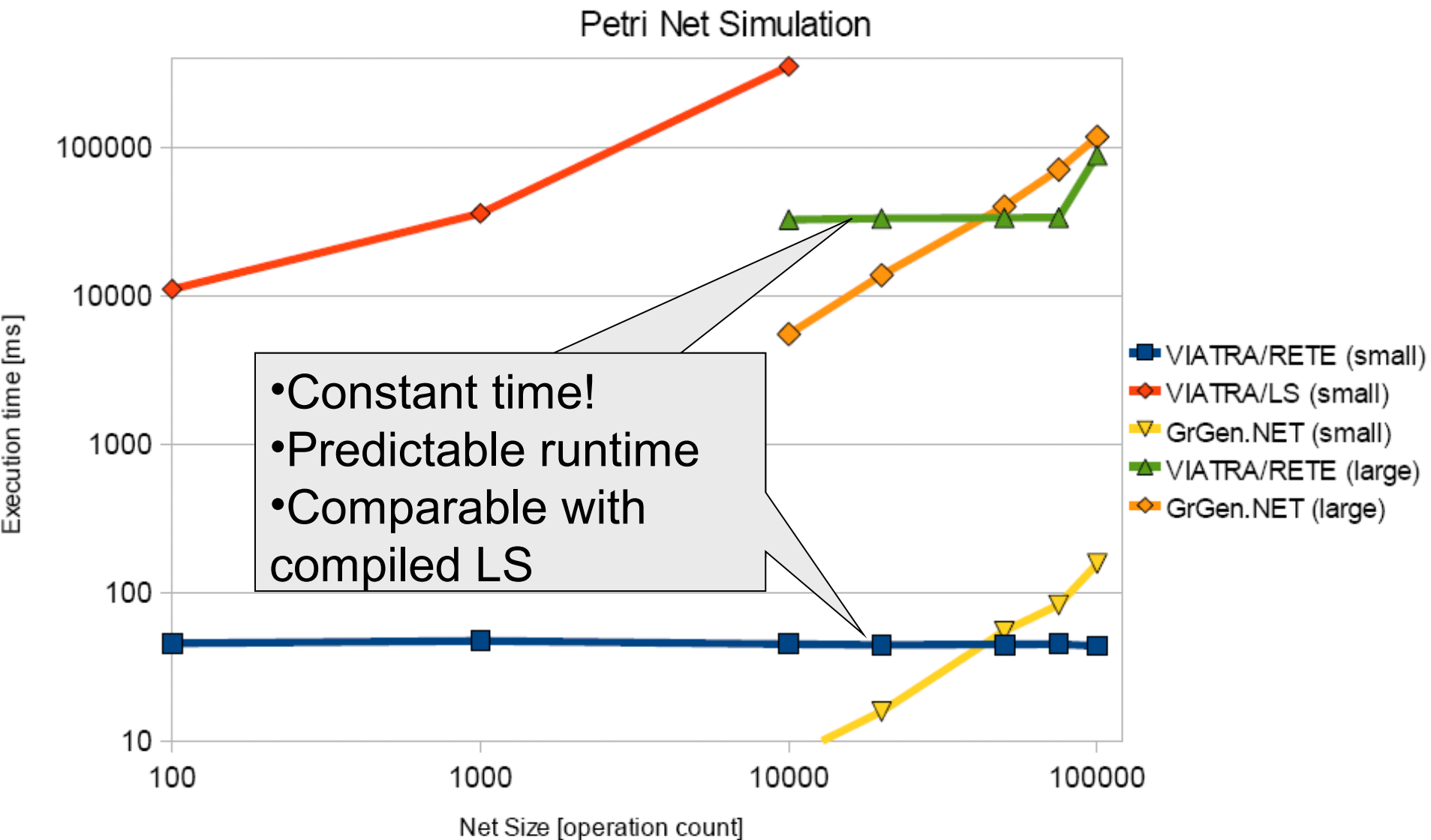
## ■ Hardware and OS

- 2 GHz Core2
- 2048 MB RAM
- Linux kernel of version 2.6.22
- Sun JVM 1.6.0\_05 for VIATRA
- .Net 3.0 on Windows Vista

## ■ Tool related

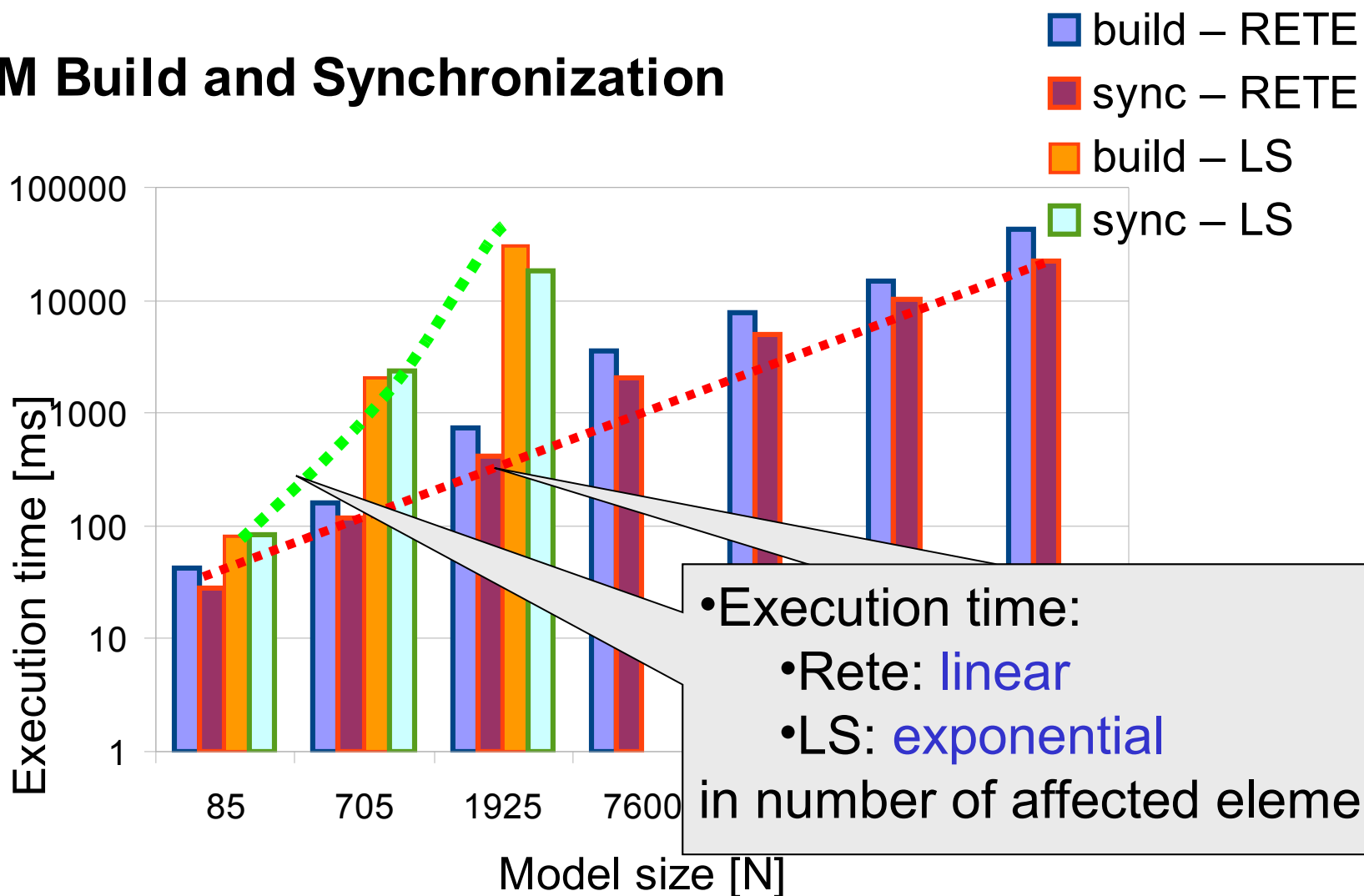
- VIATRA with and GrGen without GUI
- Standard services of the default distribution

# Results of the Petri Net Simulation



# Result of the ORM Synchronization

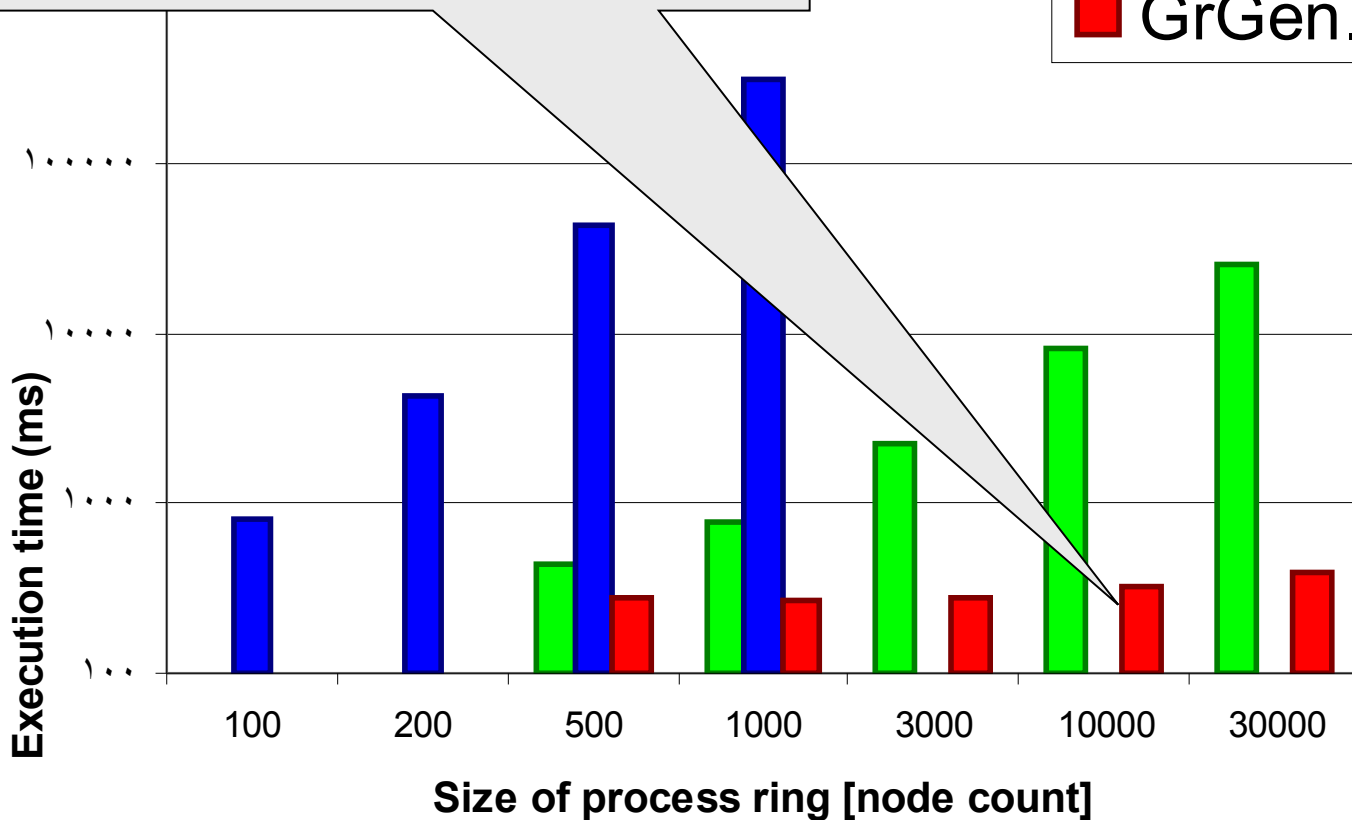
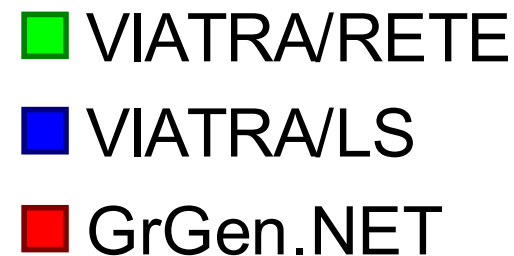
## ORM Build and Synchronization



# Varró: STS Benchmark

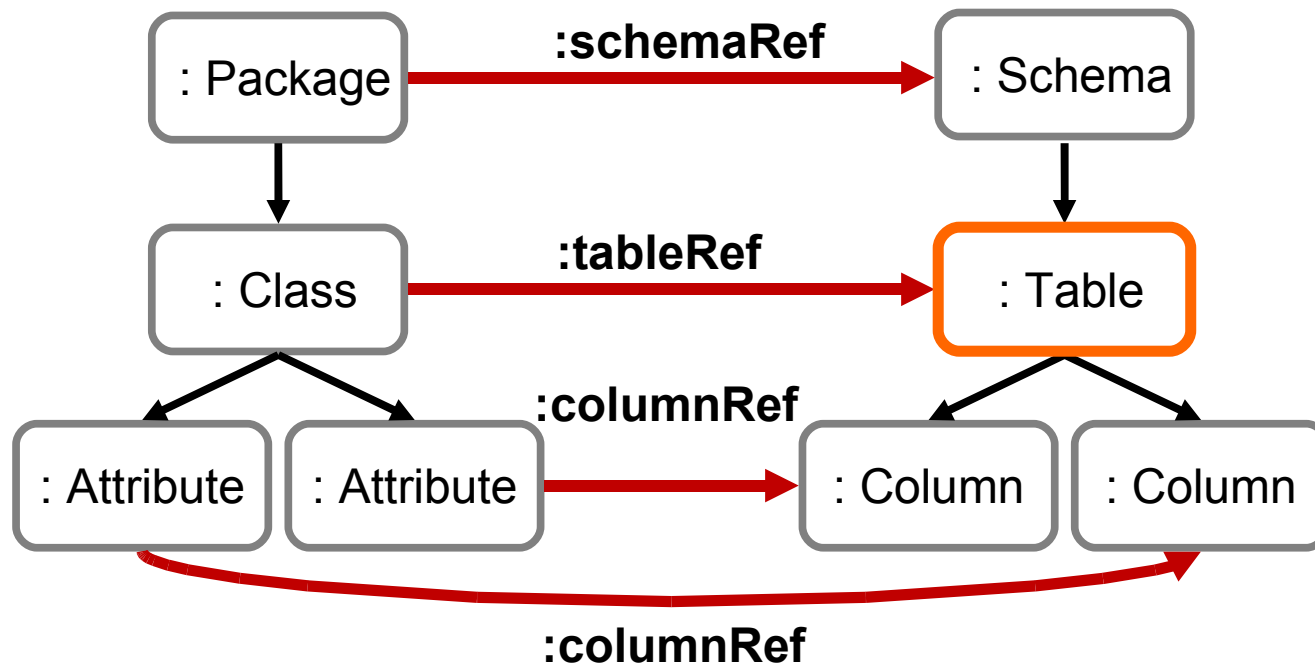
- Non reusable model elements
- Suites better for LS

bench



# Summary

- Rete based incremental PM
  - Increased memory consumption
  - Fix overhead
  - **Predictable linear scaling** for practical problems
  - Comparable performance with fastest LSs in some cases.
- Contribution to Benchmarking of GTs
  - 2 new benchmarks for assessing incremental PM
  - Larger models
  - **ALAP execution**
- Different matching strategies for different
  - Execution phases
  - Patterns in a transformation?
- In overall the **performance of GT tools are satisfying.**
- Benchmark example descriptions, specifications, and measurement results available at:
  - <http://wiki.eclipse.org/VIATRA2/Benchmarks>



### orphanTable

NEG

C: Class : tableRef

OR

A: Association : tableRef

{DEL}

T: Table

