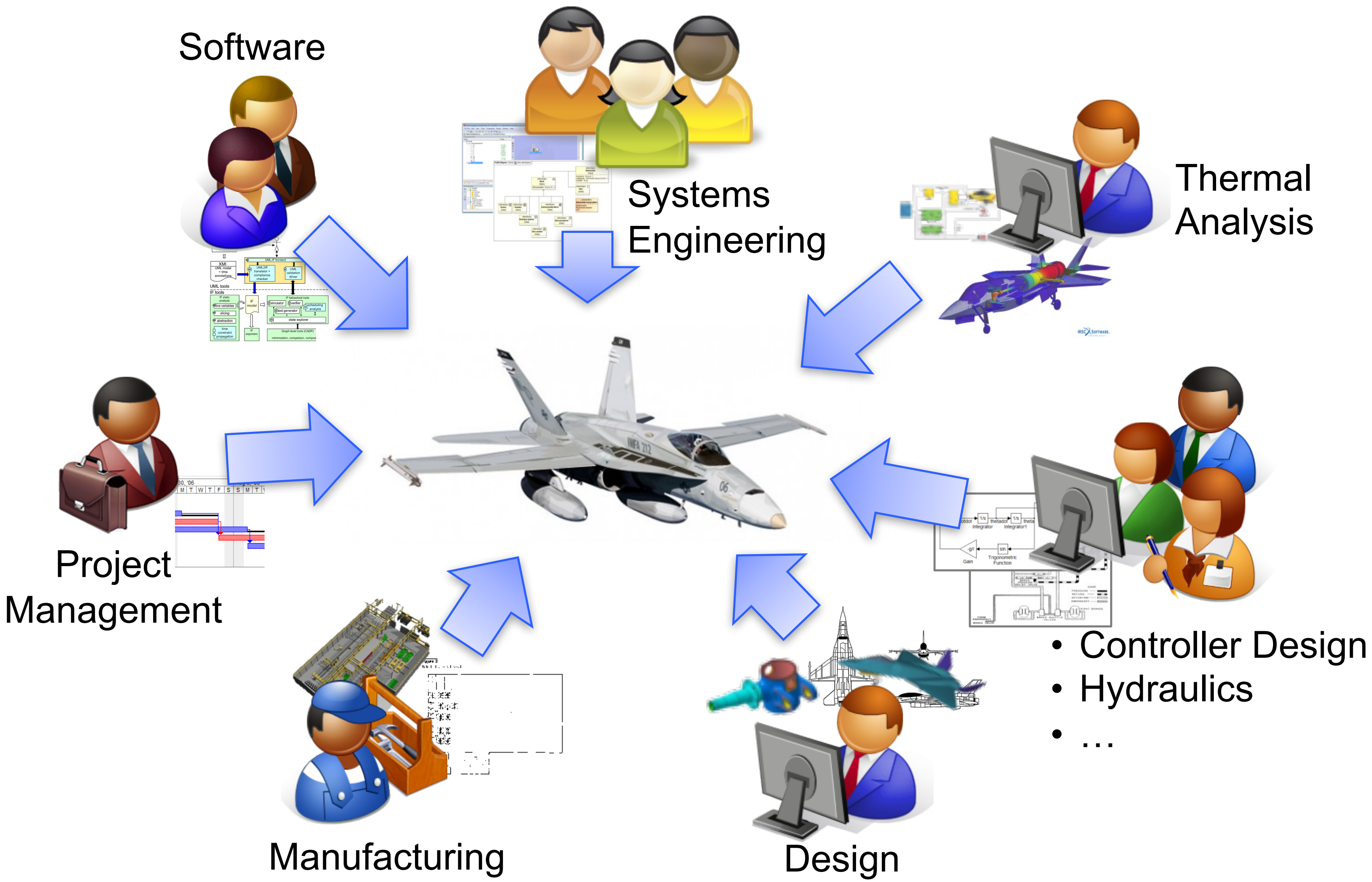
A Bayesian Learning Approach to Inconsistency Identification in MBSE

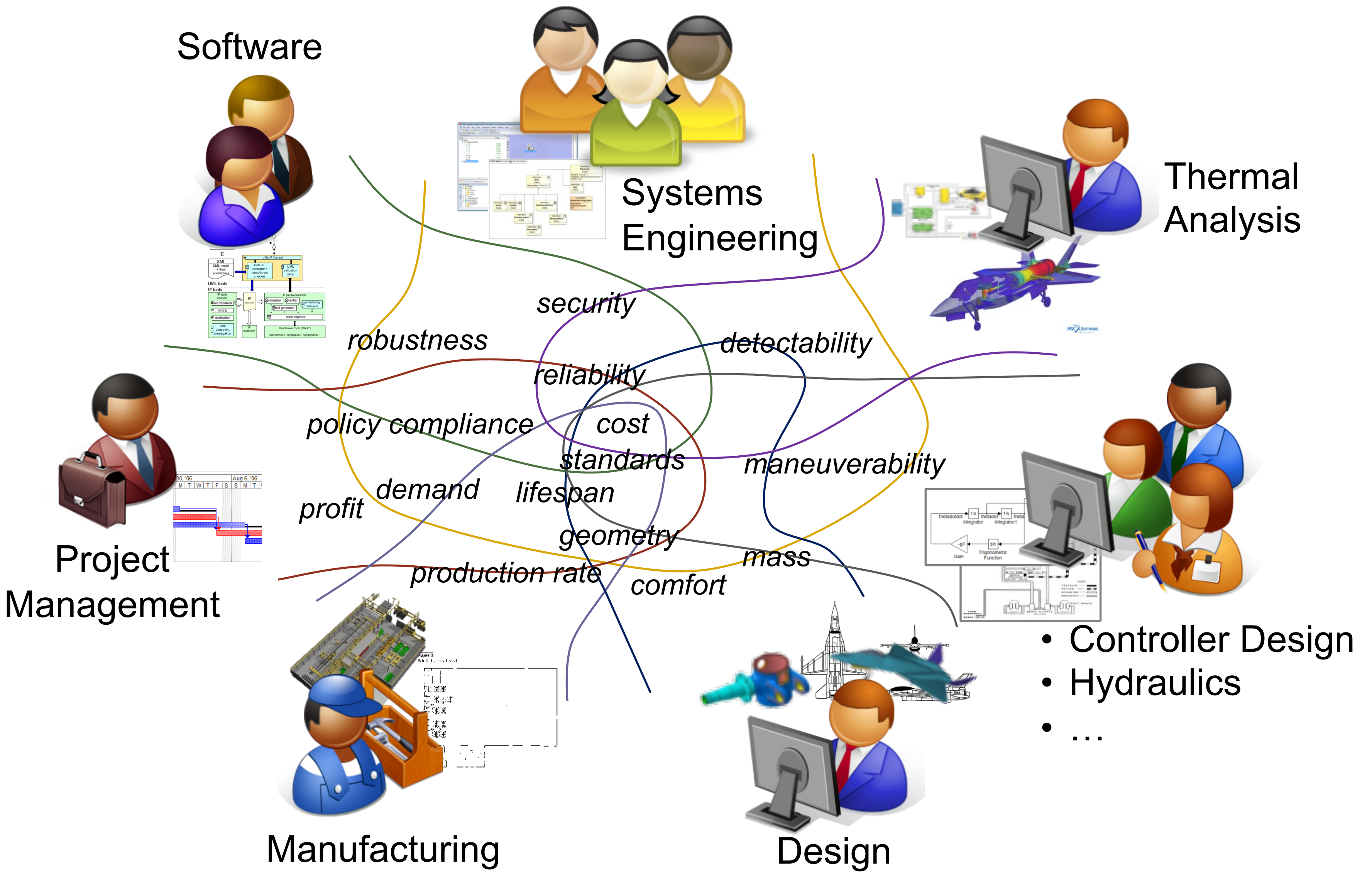
Sebastian J. I. Herzig

August 1st, 2016

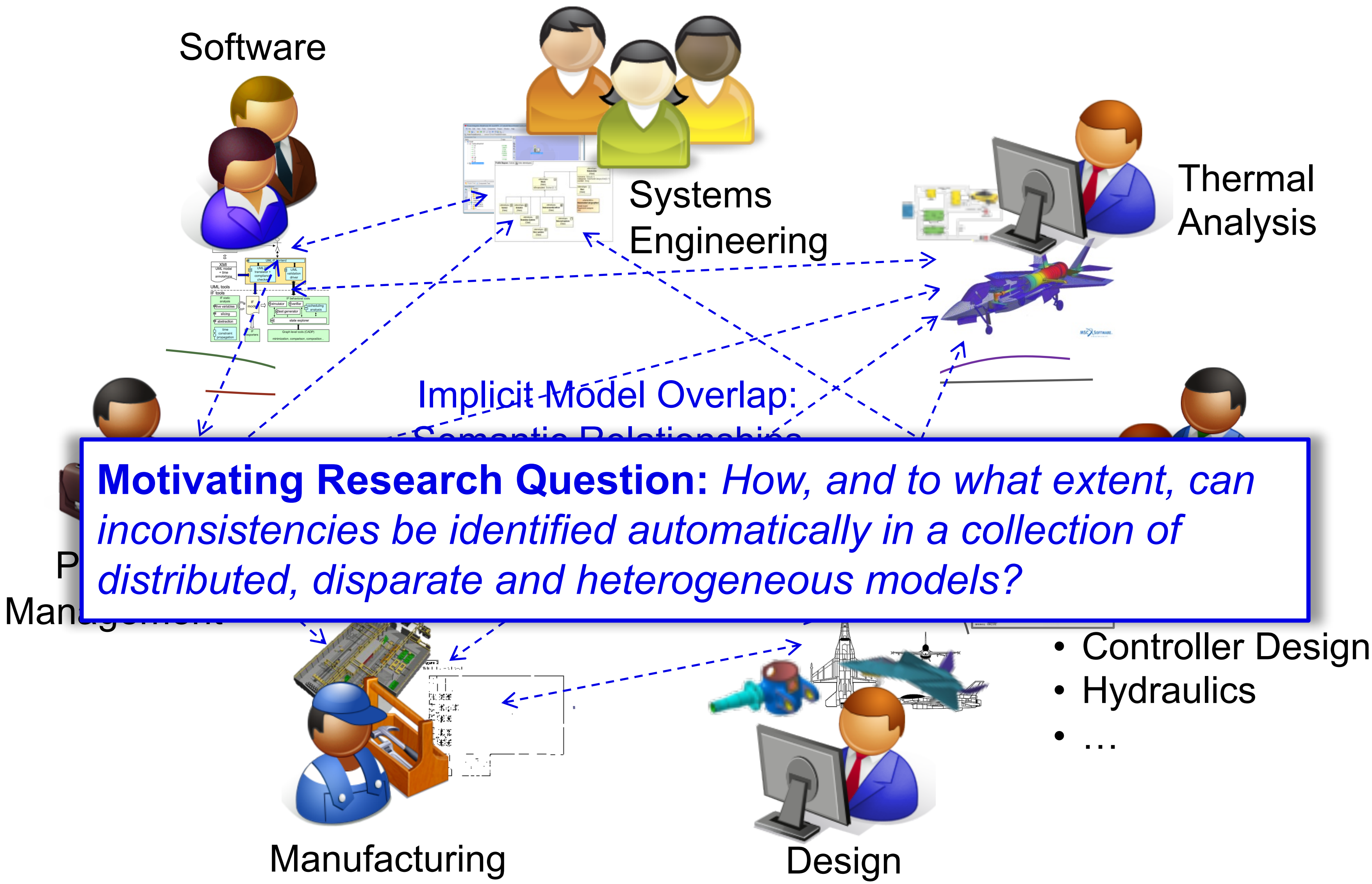
Collaboratively Developing Complex Systems



Overlapping Concerns & Dependencies



Overlapping Concerns & Dependencies



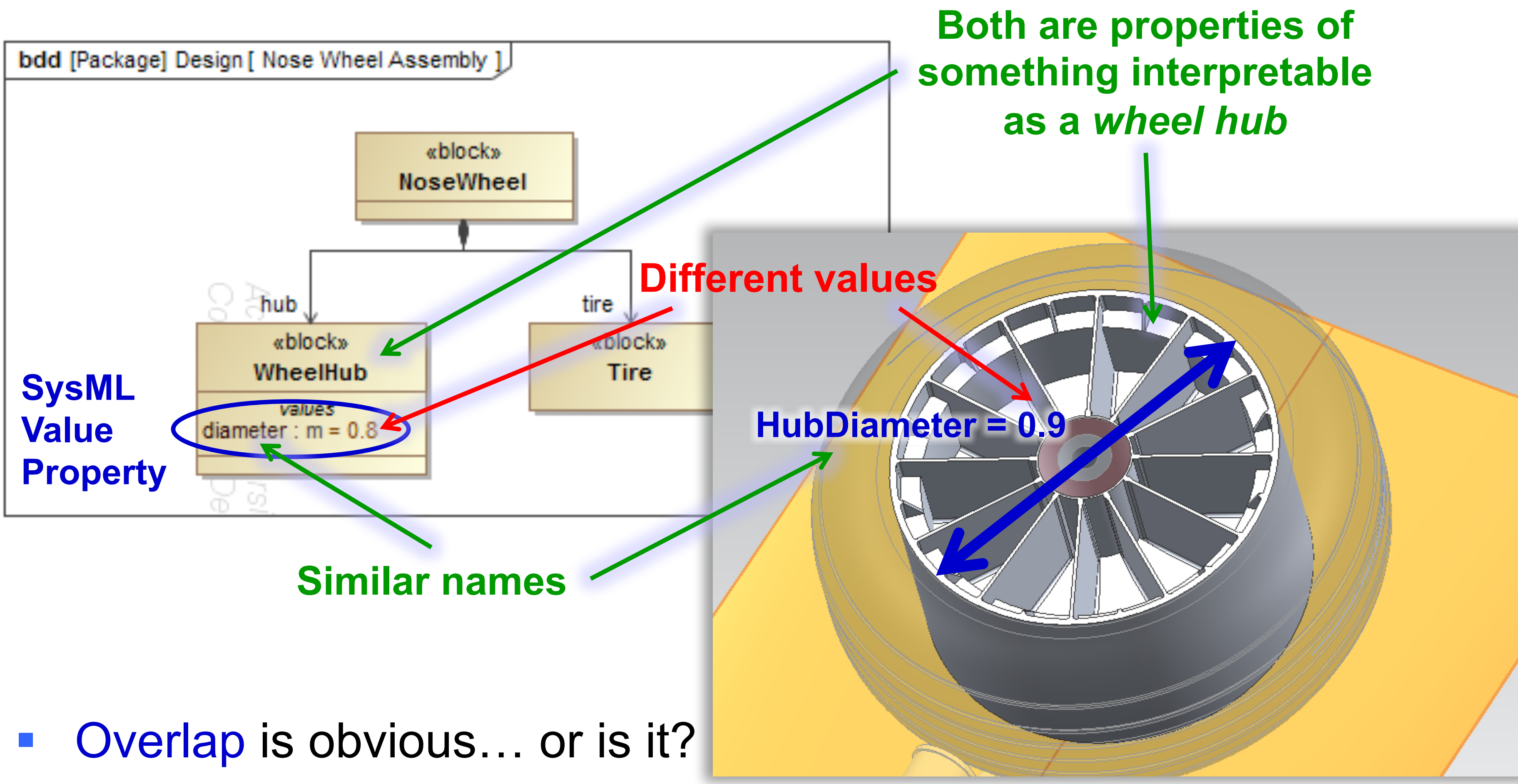
Motivating Research Question: *How, and to what extent, can inconsistencies be identified automatically in a collection of distributed, disparate and heterogeneous models?*

Outline

- Introduction
- Foundations & Methods
- Evaluation & Case Study
  - Characterization of Approach
  - Comparison to Deterministic Approach
- Conclusion

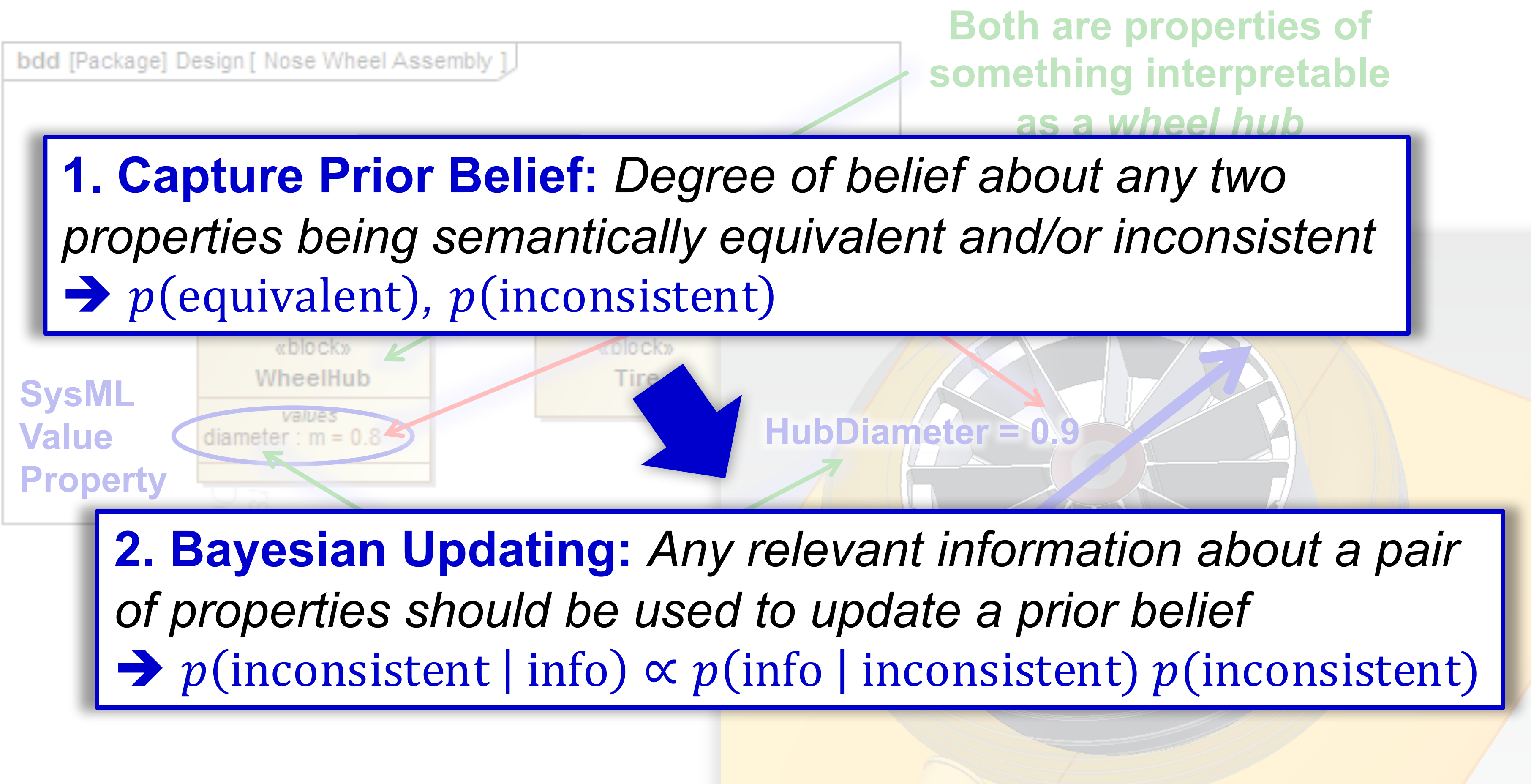
Example Problem

Are the Models Inconsistent?



- Overlap is obvious… or is it?
- Humans tend to measure similarity through identifying features
- Can a computer infer the overlap in a similar fashion?

A Bayesian Approach



1. Capture Prior Belief: *Degree of belief about any two properties being semantically equivalent and/or inconsistent*
➔ $p(\text{equivalent}),\ p(\text{inconsistent})$

2. Bayesian Updating: *Any relevant information about a pair of properties should be used to update a prior belief*
➔ $p(\text{inconsistent} \mid \text{info}) \propto p(\text{info} \mid \text{inconsistent})\, p(\text{inconsistent})$

(Herzig and Paredis: "Bayesian Reasoning over Models", MoDeVVa @ MoDELS 2014)

Setting up the Probabilistic Reasoning Problem

- Define sample space Ω:

  Here: $\Omega = \{\omega_i \mid \omega_i \text{ is a pair of properties}\}$

- Define random variables $X : \Omega \rightarrow E$:

Random Variable	Possible Values	Outcomes that map to this value
Equiv.	*Equivalent*	Semantically **equivalent properties**
	Not	
Incon.	*In*	
	Not	
NameSim.		
	Not Similar	Pairs of properties with **dissimilar names**
Values	*Equa*	**s**
	Not Eq	**ues**

E.g., say that known *(i.e., assumed with certainty):*
1. Names are similar
2. Values are not equal

➔ Update prior on inconsistency

$$P(\mathrm{I} \mid \mathrm{N}, \neg\mathrm{V}) = \frac{P(\mathrm{N}, \neg\mathrm{V} \mid \mathrm{I})P(\mathrm{I})}{P(\mathrm{N}, \neg\mathrm{V})} = \frac{P(\mathrm{I}, \mathrm{N}, \neg\mathrm{V})}{P(\mathrm{N}, \neg\mathrm{V})}$$

Bayesian Networks

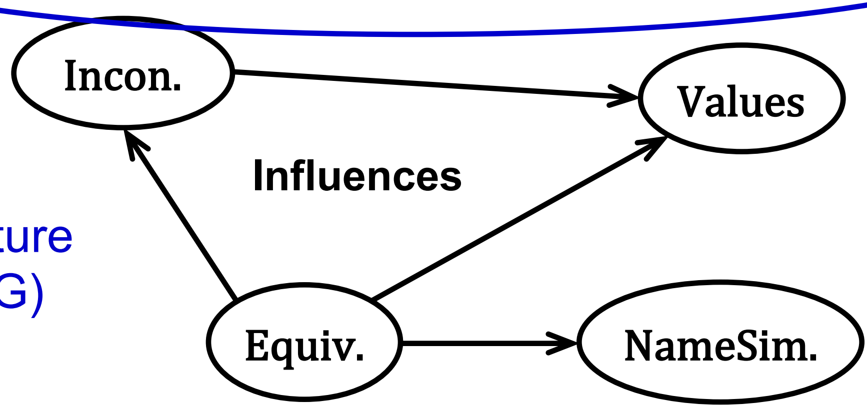
- *Markov condition*: each variable is independent of its non-descendants given values of its parent variables

$$P(X_1 = x_1, X_2 = x_2, \ldots) = \prod_i P(X_i = x_i \mid \mathrm{pa}(X_i))$$

- Bayesian networks are DAGs that fulfill this condition and are compact representations of joint probability distributions

$P(\text{Incon.} = \text{Inconsistent} \mid \text{Equiv.} = \text{Equivalent}) = 0.1$
$P(\text{Incon.} = \text{Inconsistent} \mid \text{Equiv.} = \text{Not Equivalent}) = 0.001$
…

Network Parameters



Structure (DAG)

➔ Here: only 10 instead of $2^4 = 16$ probability values required
➔ Difference much larger for bigger networks…

Bayesian Learning

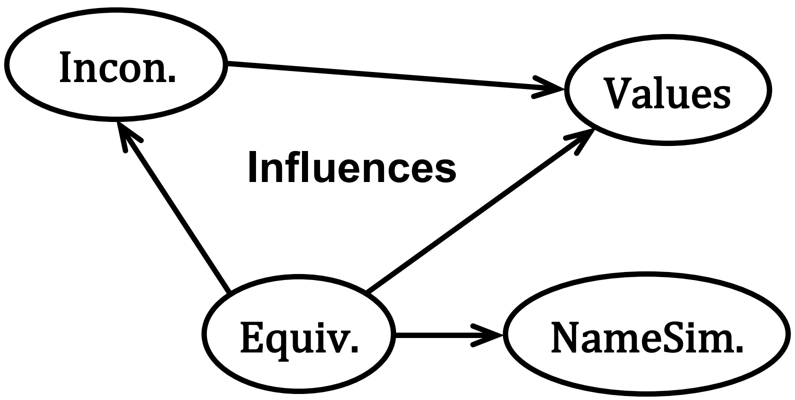
Learning Network Parameters

1. Priors:

$P(\text{Incon.} = \text{Inconsistent} \mid \text{Equiv.} = \text{Equivalent}) = 0.1$ ← $\mathbf{Dir(100, 900)}$

$P(\text{Incon.} = \text{Inconsistent} \mid \text{Equiv.} = \text{Not Equivalent}) = 0.001$ ← $\mathbf{Dir(1, 999)}$

…



2. Set of Data Cases:

Incon.	**Equiv.**	**Values**	**NameSim.**
Not Inc.	*Not Eq.*	*Similar*	*Not Sim.*
Incon.	*Equiv.*	*Not Sim.*	*Similar*
…	…	…	…

3. Use Statistical Count to Update Prior Distributions

$$P(X_i = j \mid \text{pa}(X_i) = k, \mathbf{D}) = \frac{m_{ijk} + \alpha_{ijk}}{\sum_j (m_{ijk} + \alpha_{ijk})}$$

Parameters of Dirichlet priors

Specific configuration of values for parent variables

Models as a Source for Information (Evidence)

- Define sample space Ω:

  Here: $\Omega = \{\omega_i \mid \omega_i \text{ is a pair of properties}\}$

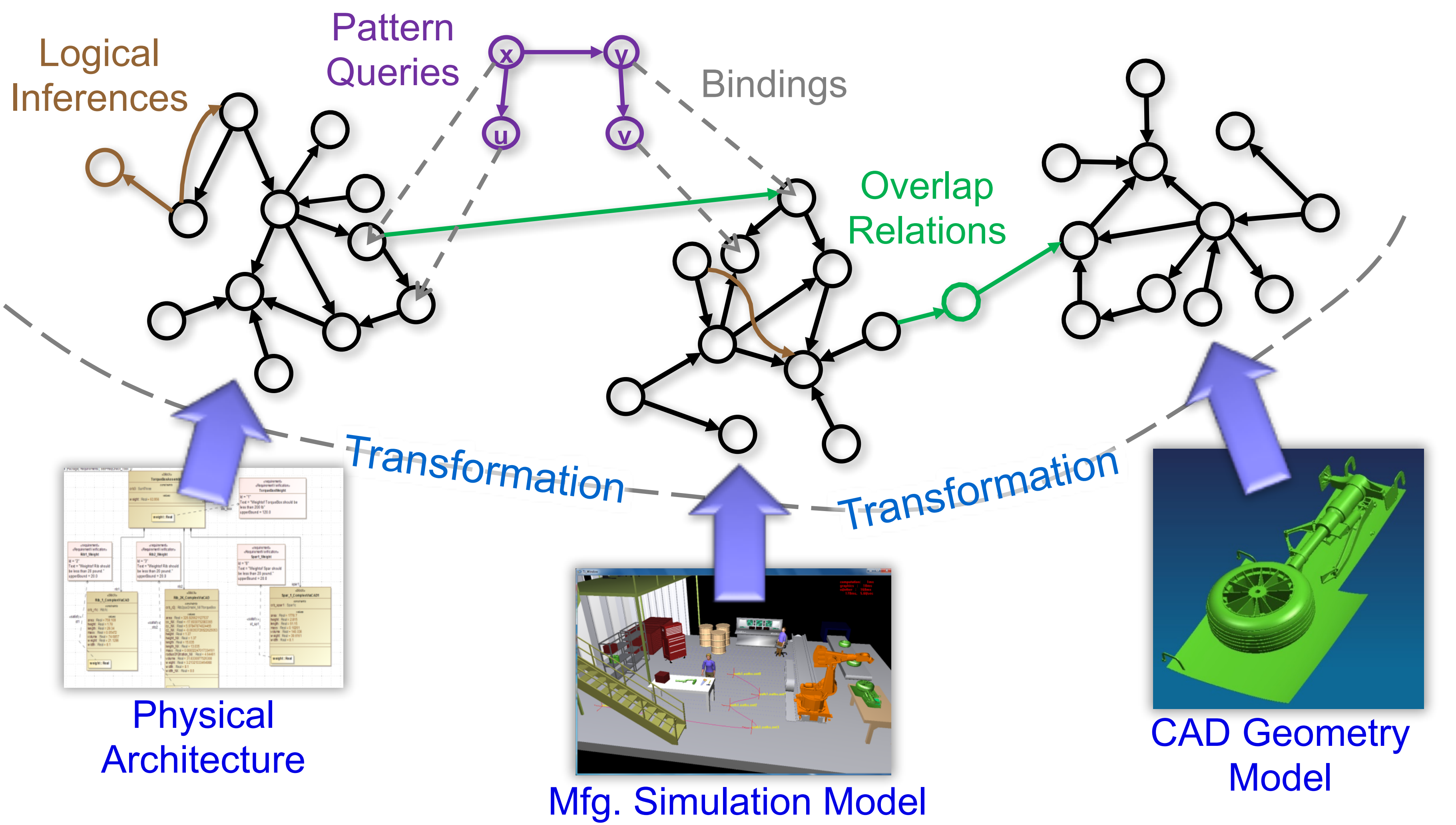
- Define random variables $X : \Omega \rightarrow E$:

Random Variable	Possible Values	Outcomes that map to this value
Equiv.	*Equivalent*	Semantically **equivalent properties**
	Not Equivalent	Semantically **different properties**
Incon.	*Inconsistent*	Pai…
	Not Inconsistent	Pairs … **nt**
NameSim.	*Similar*	Pa…
	Not Similar	Pair… **s**
Values	*Equal*	Pairs of properties with **equal values**
	Not Equal	Pairs of properties with **non-equal values**

How can *models* be used as a source of information, from which some of these can be determined computationally?

A Common Representational Formalism

Representing Propositional Information & Knowledge by Graphs



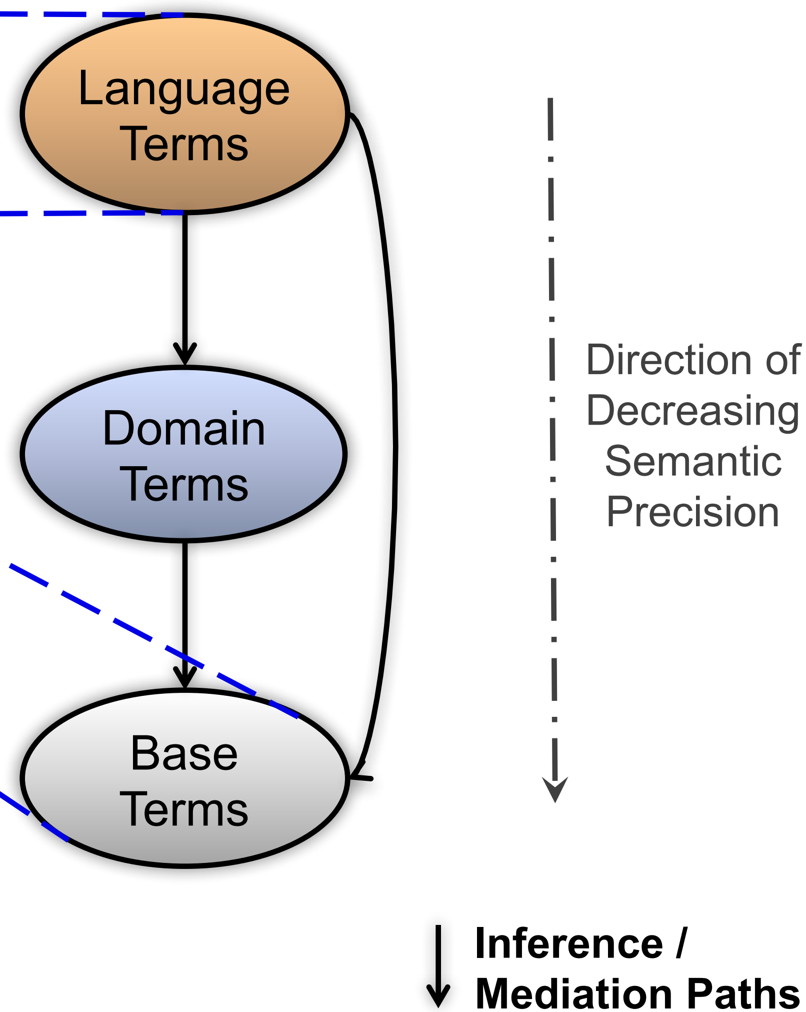
(Herzig and Paredis: "A Conceptual Basis for Inconsistency Management in MBSE", CIRP Design 2014)

Loosely Relating Heterogeneous Models through Semantic Abstraction (Mediation)

E.g., SysML:
ValueProperty, **PartProperty**, Block, Requirement, ...

mediated to **mediated to**

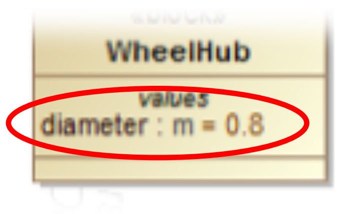
Base:
Entity, **Property**, Relation, type, Constraint, value, ...

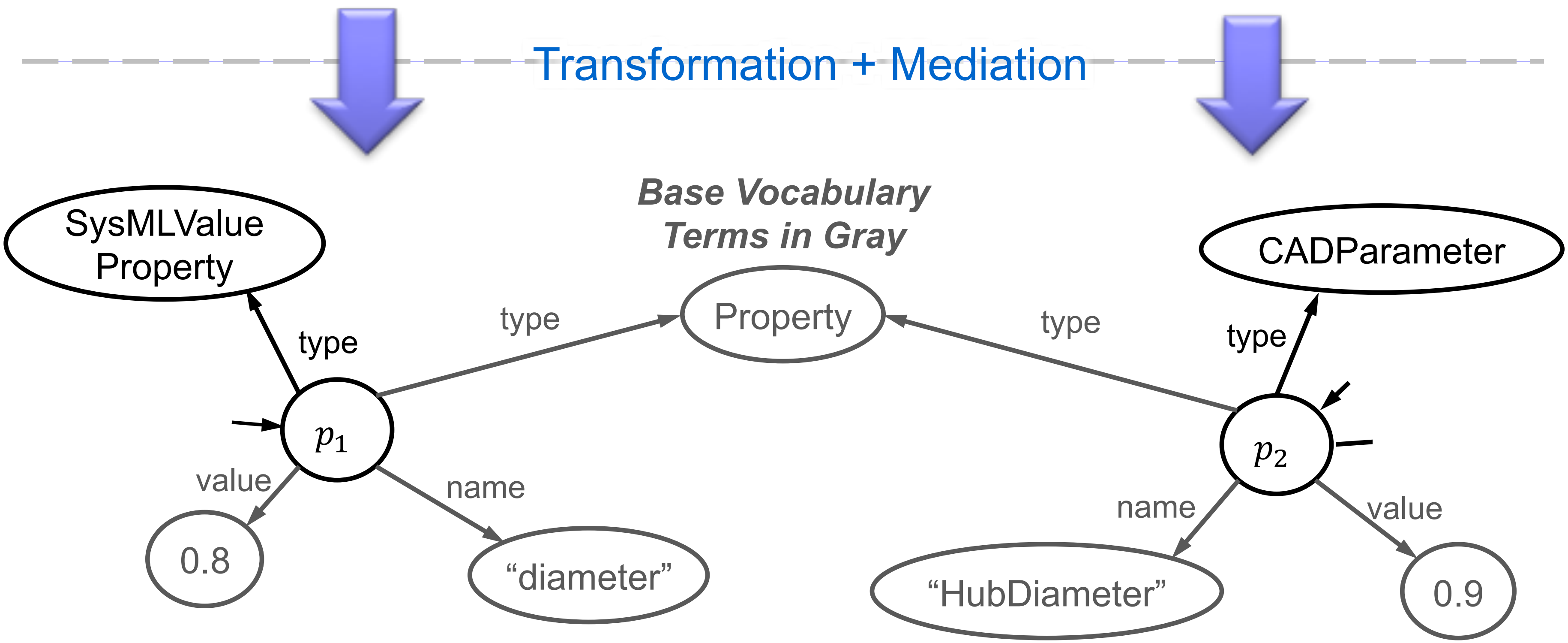


- This “base vocabulary” provides a basic interface between models
- Enables higher level reasoning

(Partly result of joint work with Technische Universität München under grant SFB768)

Back to the Example Problem…

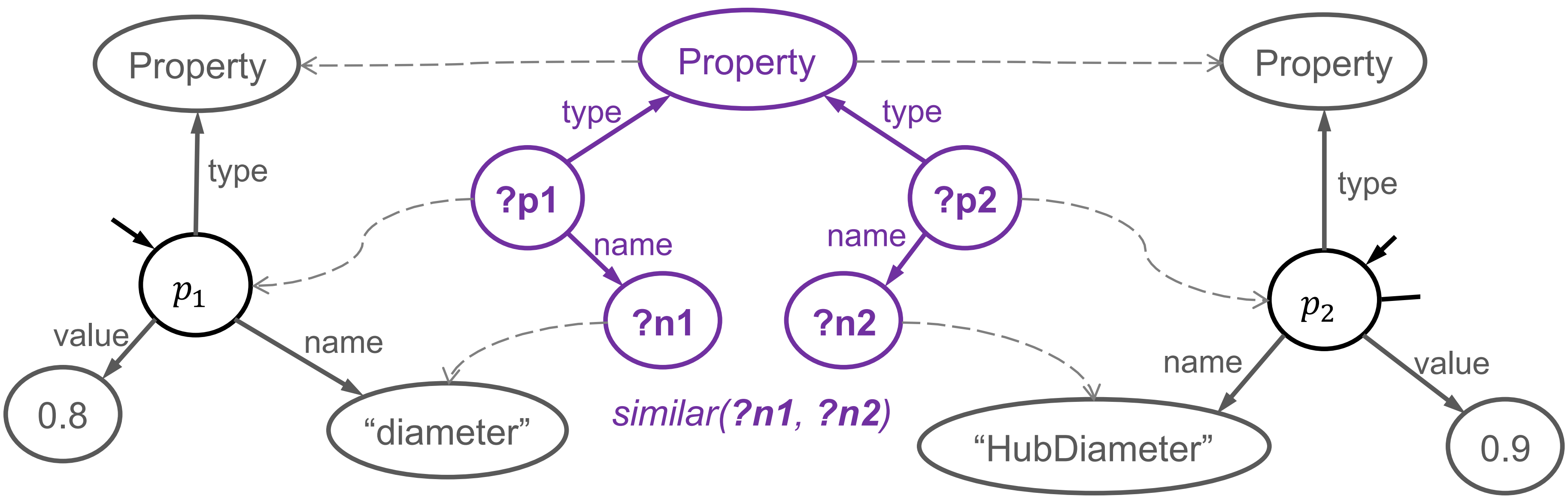




Extracting Evidence from the Graph-Based Model

- **Models = one source for information**
- Use graph pattern matching to collect information about outcomes
  - ➔ Identify members of pre-images of random variables
  - ➔ For example: $(p_1, p_2) \in$ **NameSim.**$^{-1}$(similar)

"Pairs of properties with similar names"



Practically Defining the Reasoning Problem

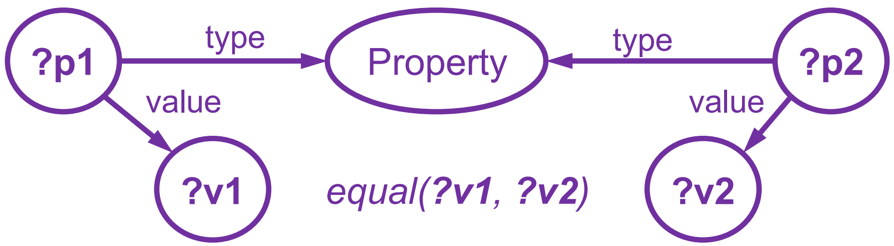
- Define sample space Ω, *and a base pattern*:

  Ω := Set of pairs of properties

- Define random variables, *and associated patterns*:

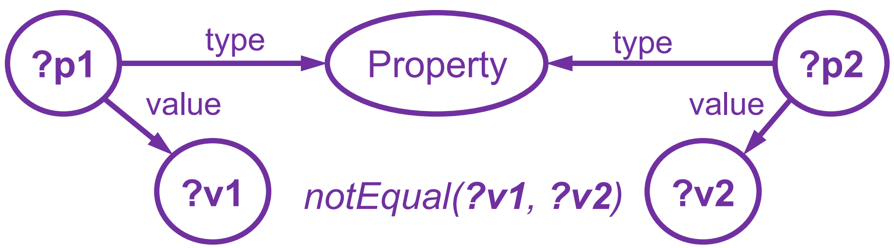
Values = {

Equal ←

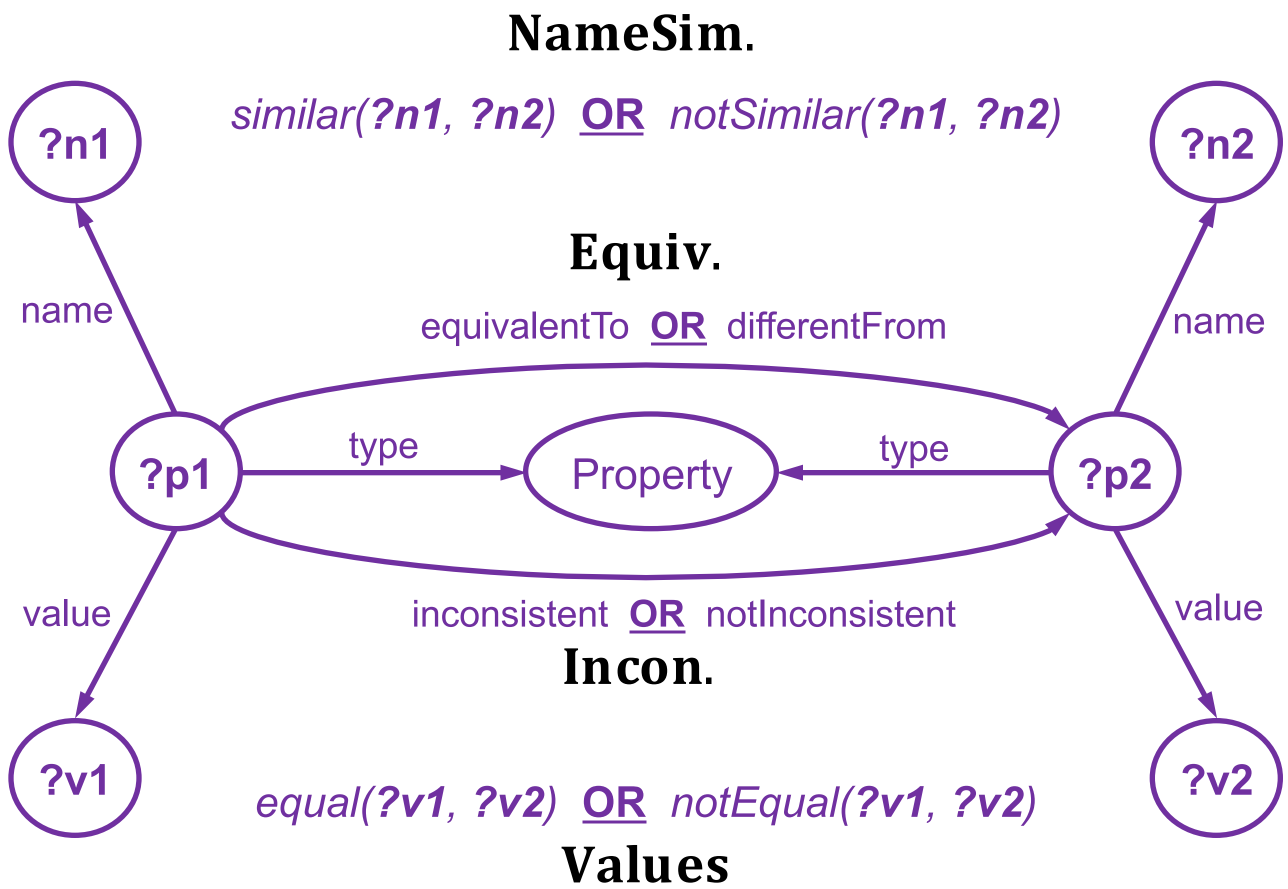


Important: mutual exclusion

Not Equal ←



Overlaying the Different Patterns



For each context defined by a match to the base pattern, variable bindings are shared across all "sub"-patterns

Should all Inferred Statements be Considered?

- Say a set of models defines 200 properties…
  → $200^2 = 40{,}000$ pairs
  → **40,000 *possible* inconsistencies**

- Not all of the possible inconsistencies are *actual* inconsistencies: which inconsistencies should be reported?
  → Rank order, and / or group, by probability
  → Heuristic: how probable is "probable enough" for an inconsistency to be reported? I.e., what is an appropriate **cutoff value**?

$$\textbf{IF } P(\text{Incon} \mid \text{Evidence}) \geq \text{cutoff } \textbf{ THEN } \textit{Report}$$

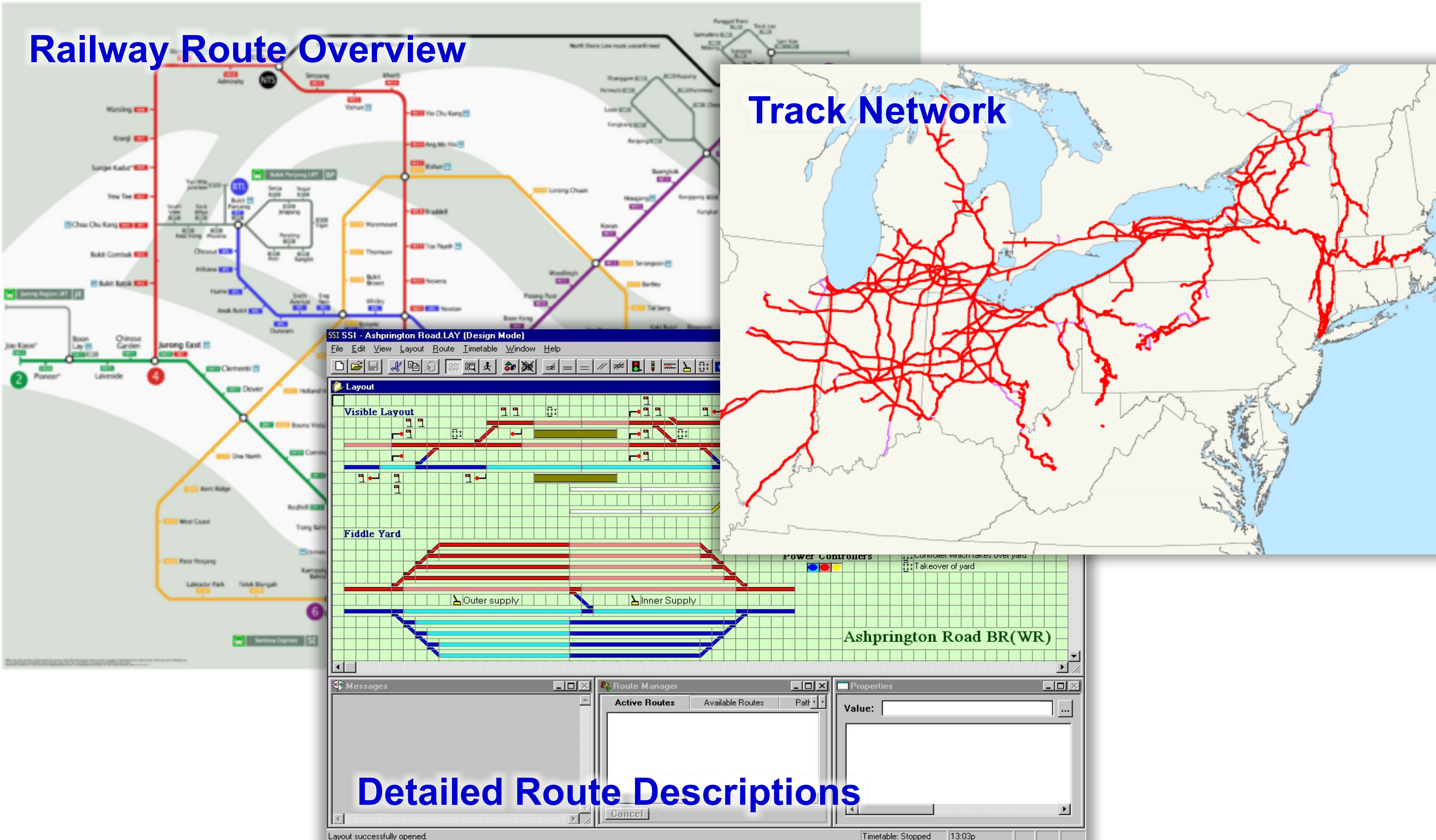
- When to *classify* something as inconsistent is a question of *value*:

$$\mathrm{E}[V_{c=i}] = \mathrm{E}[V_{mission}] - \underbrace{\left(C_{test} + C_{ver}\left(P_{TP,c=i} + P_{FP,c=i}\right) + C_{ver}P_{FN,c=i}\right)}_{\text{Incurred cost, per sample}}$$

Outline

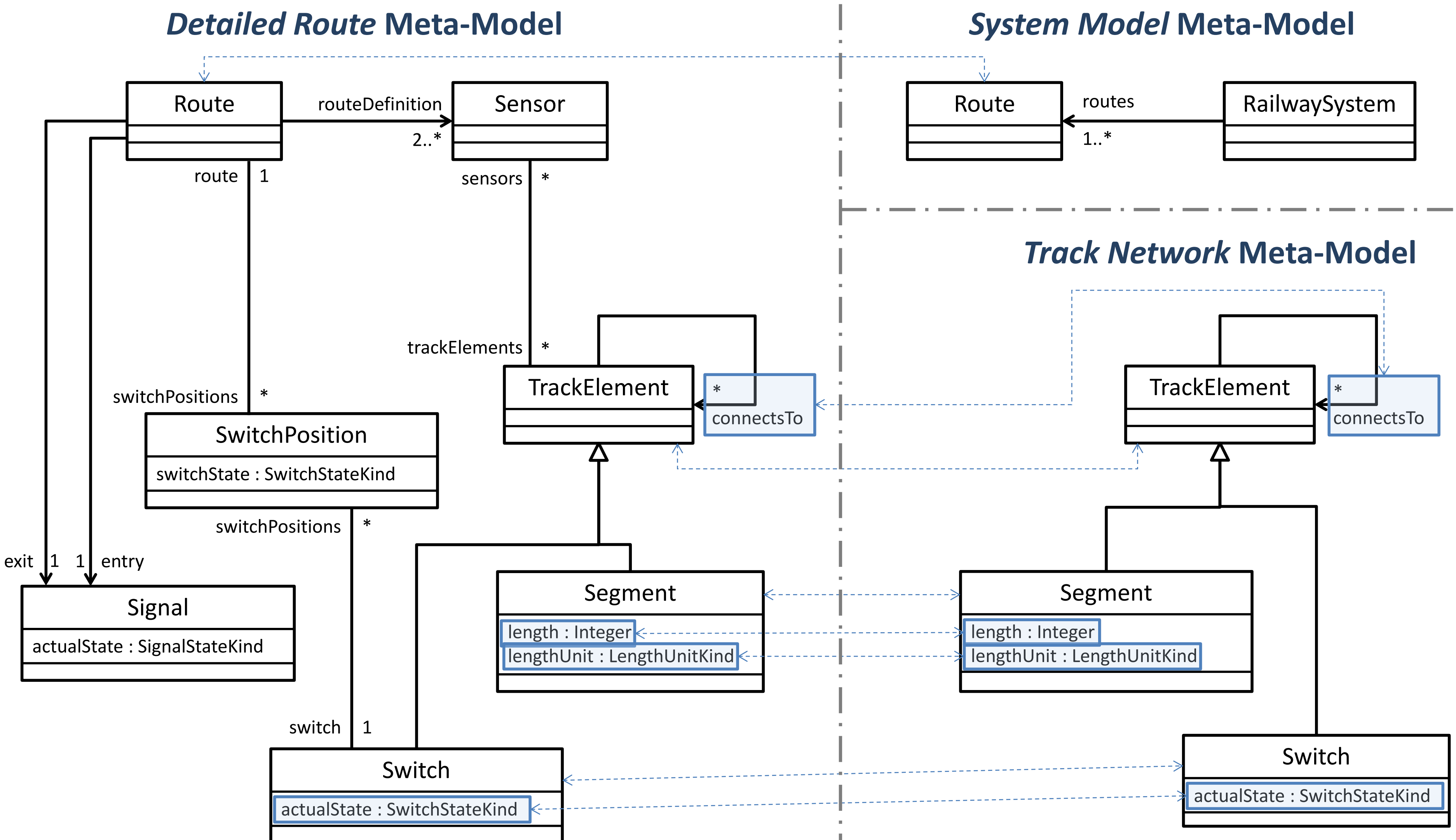
- Introduction
- Foundations & Methods
- Evaluation & Case Study
  - Characterization of Approach
  - Comparison to Deterministic Approach
- Conclusion

Case Study: Railway Systems



*Source: Wikimedia commons

Case Study: Railway Systems
Meta-Models & Meta Correspondences

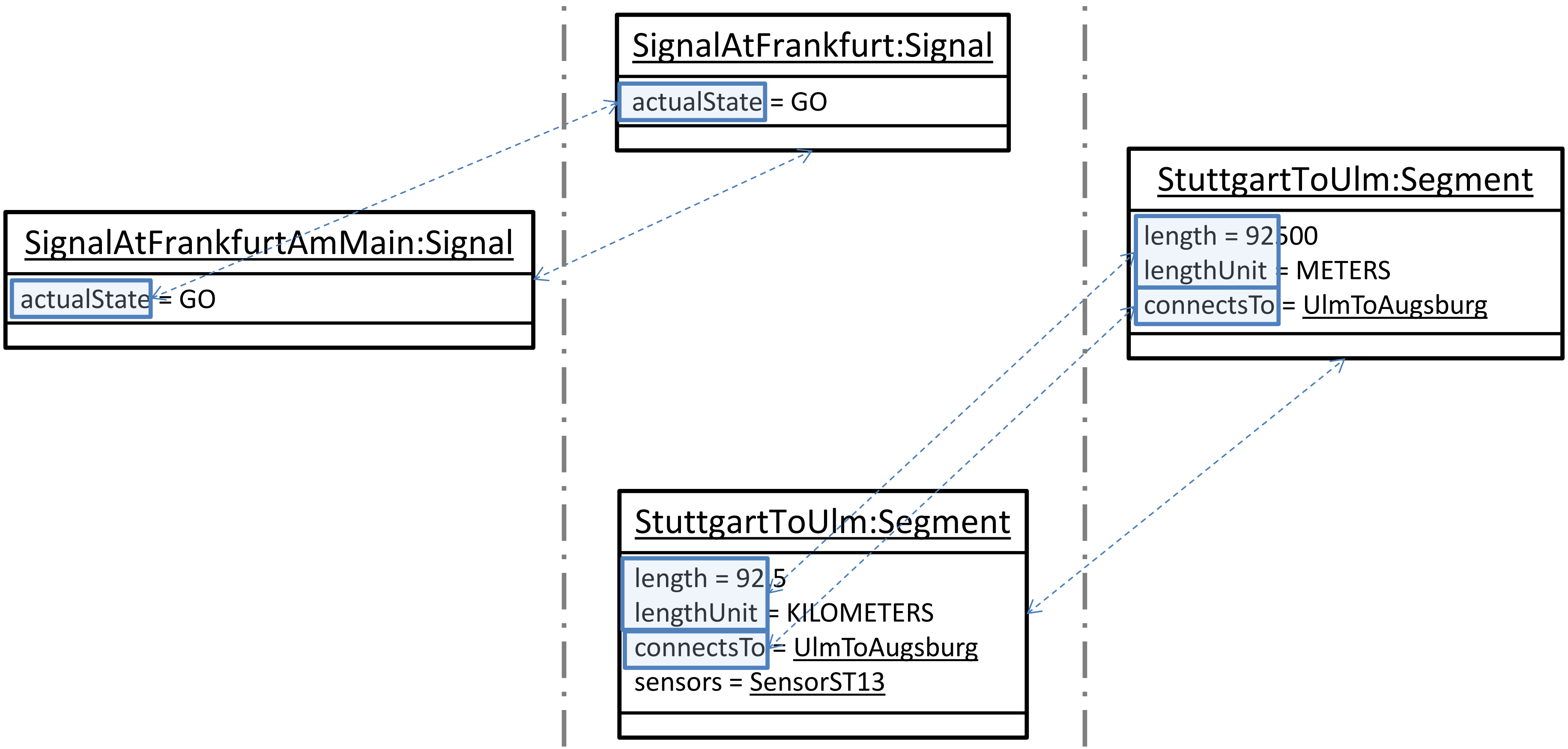


*Meta-models adapted from MOGENTES EU FP7 project

Case Study: Railway Systems

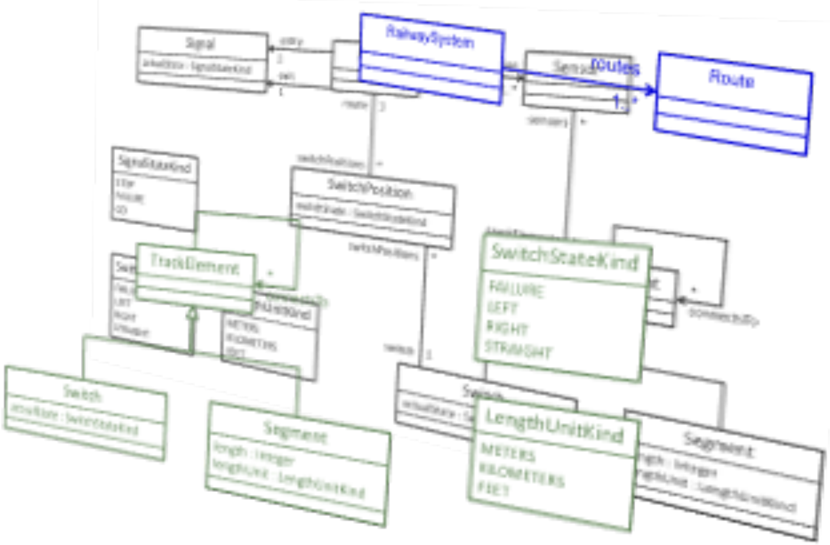
Sample Instance & Model Overlap

Detailed Route Model A | **Detailed Route Model B** | **Track Network Model**



Experiment Setup

- Automatic generation of instance models
  - WordNet® for sensible names
  - Injected with inconsistencies and imperfections at random
  - Up to ~12,500 graph triples (memory limit reached (16GB))
- Measured true and false positives and negatives for various cutoff probabilities
- RDF as graph-based formalism
- Value-based classification metric:

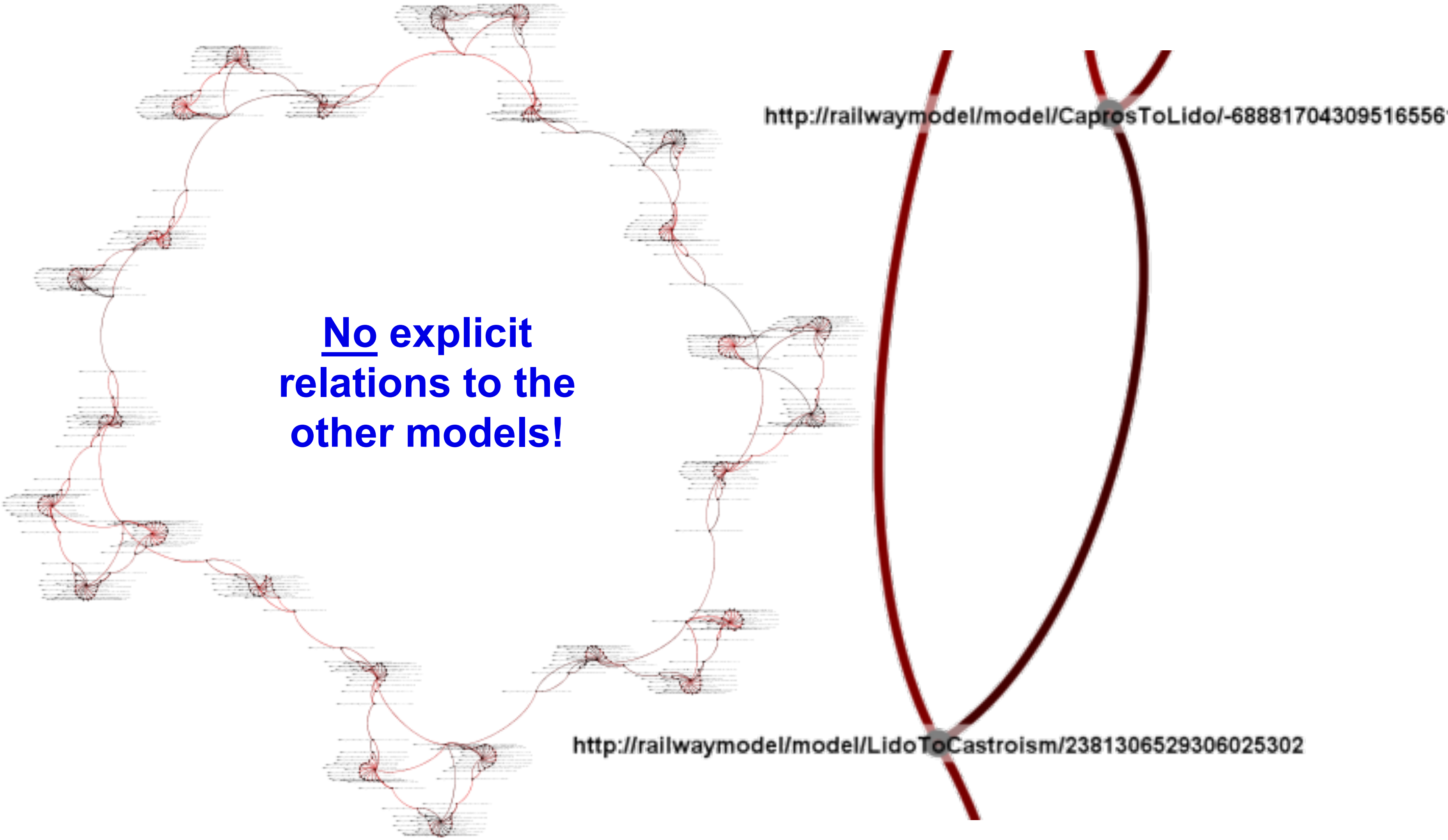


$$\mathrm{E}[V_{c=i}] = \mathrm{E}[V_{mission}] - \underbrace{\left(C_{test} + C_{ver}\left(P_{TP,c=i} + P_{FP,c=i}\right) + C_{ver}P_{FN,c=i}\right)}_{\text{Incurred cost by classifier, per sample}}$$

Incurred cost by classifier, per sample

*ICE Train: https://farm4.staticflickr.com/3326/3223619342_638d736155_z.jpg
**Meta-models adapted from MOGENTES EU FP7 project

Example Track Network Model as RDF Graph

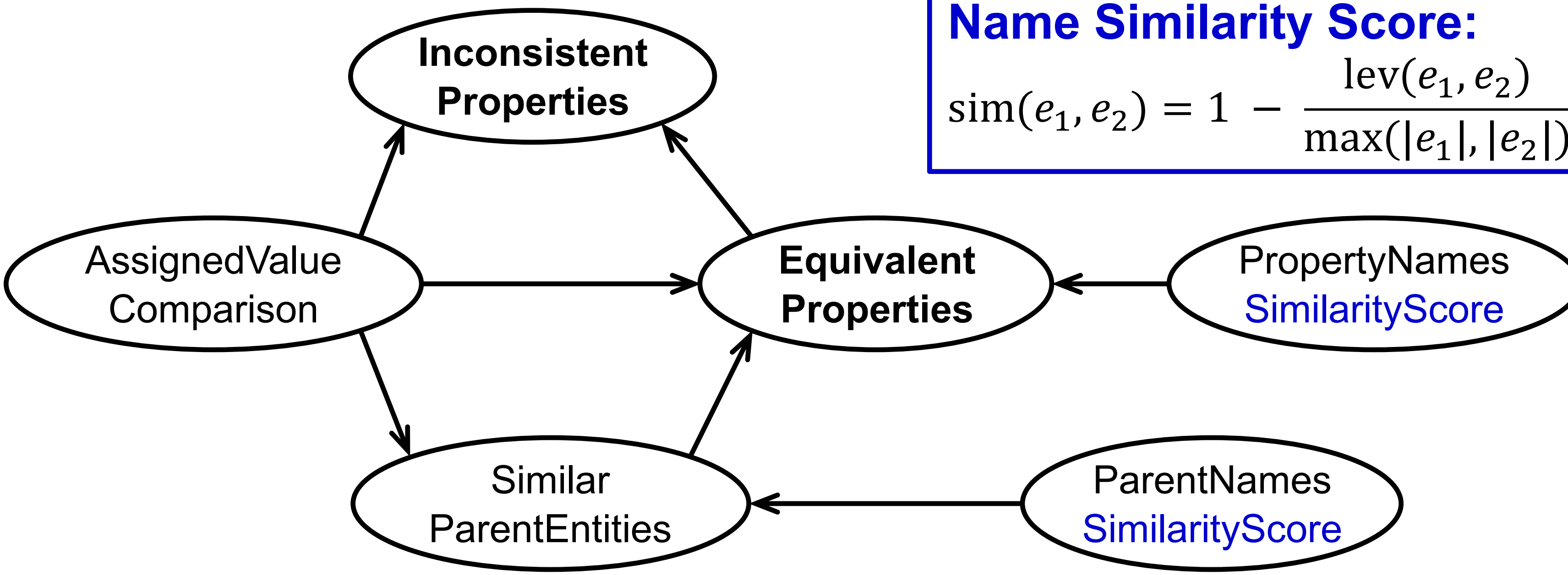


<u>No</u> explicit relations to the other models!

Defining the Reasoning Problem

Name Similarity Score:

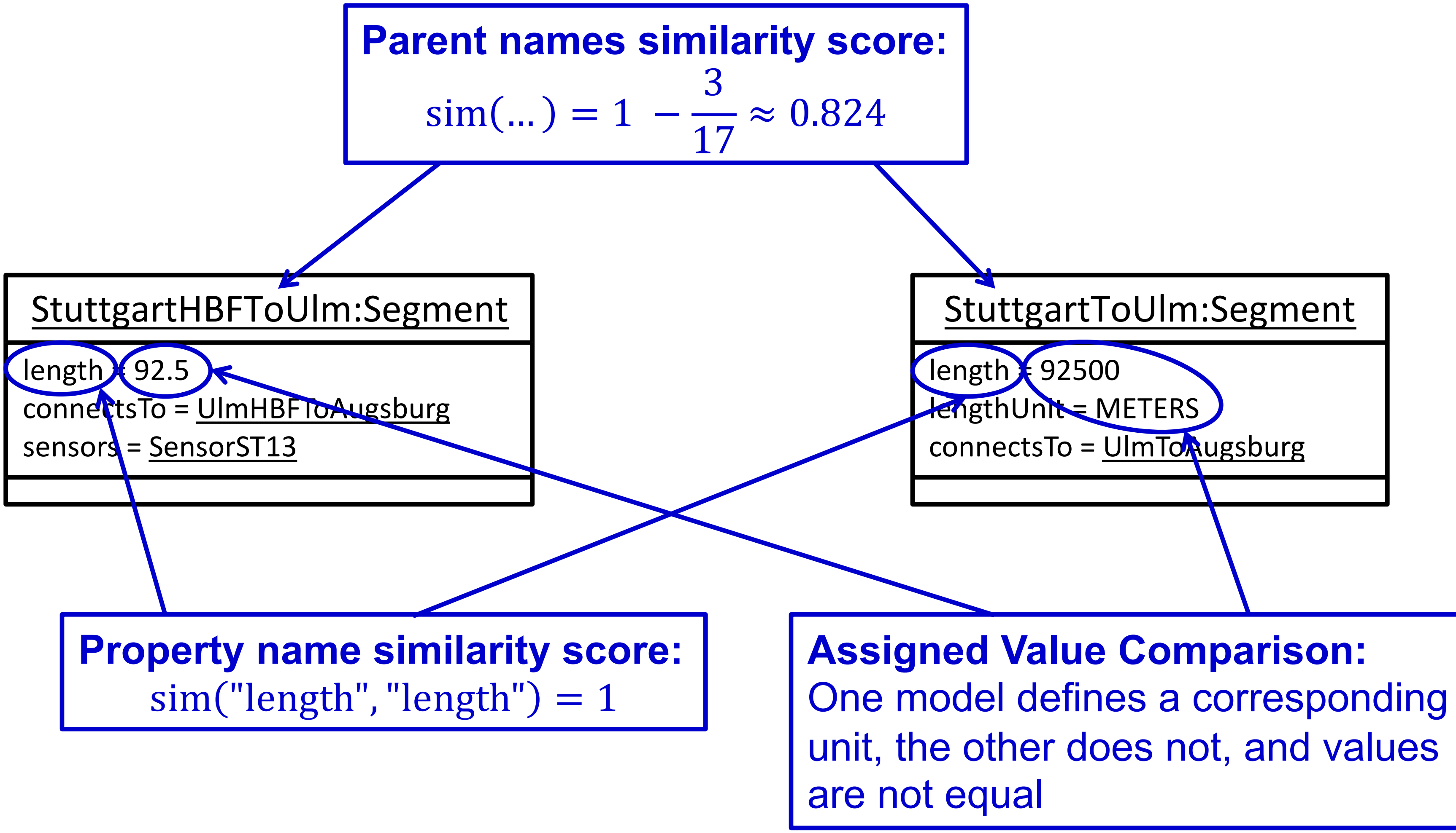
$$\text{sim}(e_1, e_2) = 1 - \frac{\text{lev}(e_1, e_2)}{\max(|e_1|, |e_2|)}$$



- **Goal:** identify inconsistent, overlapping pairs of properties
- Bayesian network with 6 **discrete** random variables and subjective probabilities for Bayesian network parameters
- Associated patterns (*20 in total*) only use terms from **base vocabulary**
- Expected cost incurred by a FN = 10,000 * cost of verifying a TP/FP (* 0.05 for case of semantic equivalences)

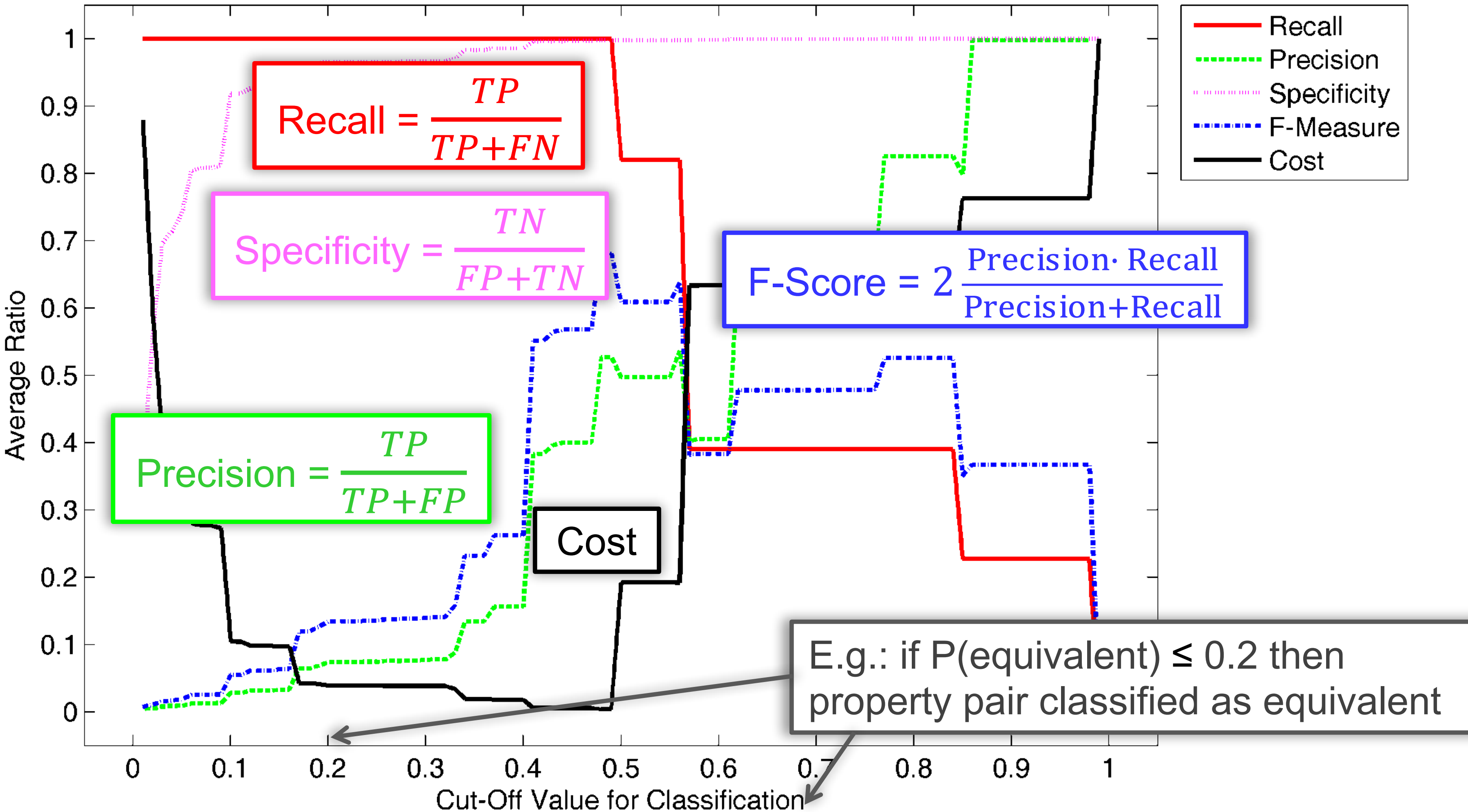
Associated Patterns

➔ **20 patterns part of reasoning knowledge**

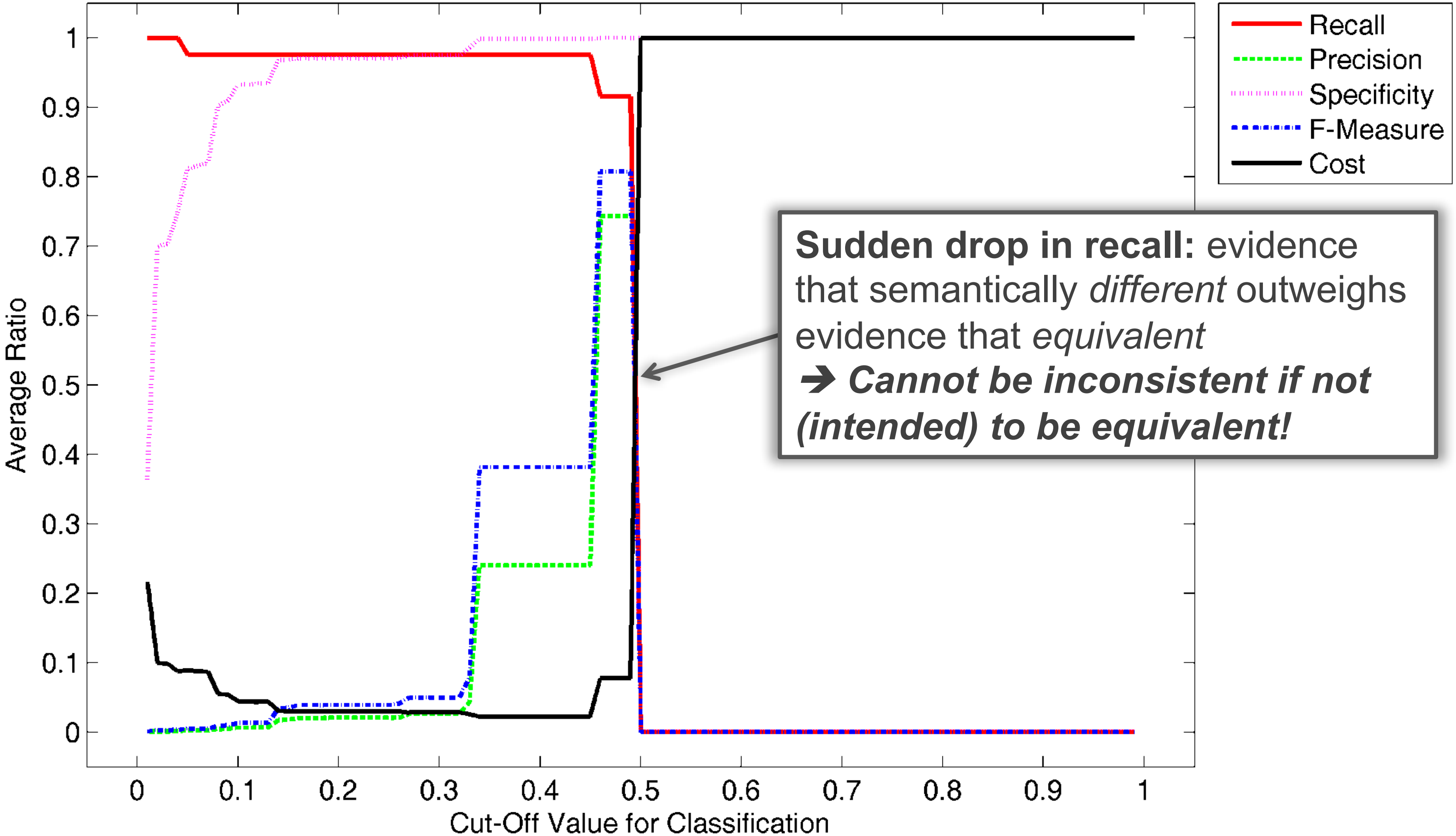


Initial Measurements

(Intended) Semantic Equivalence of Properties



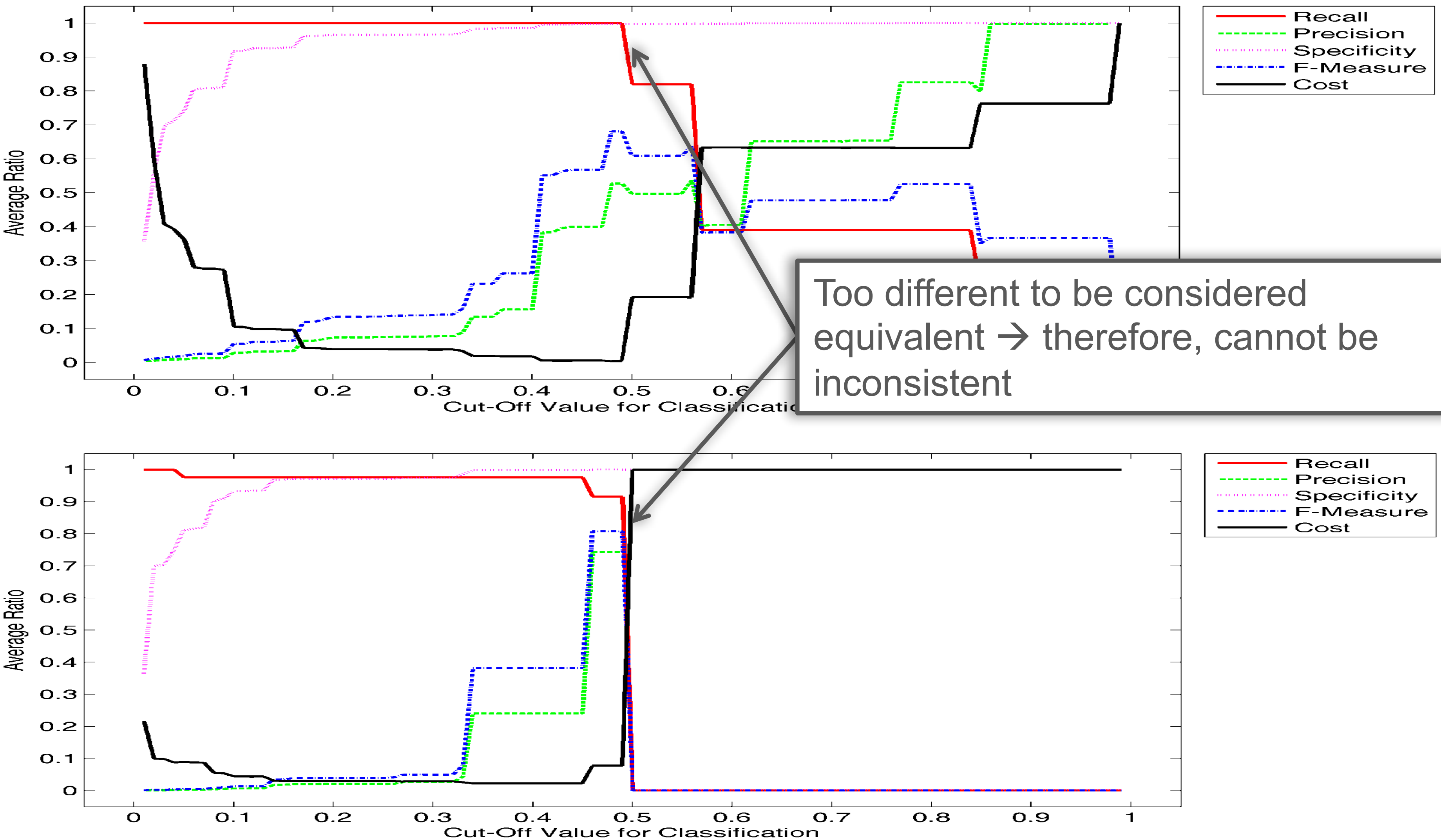
Initial Measurements
Inconsistency of Properties



Sudden drop in recall: evidence that semantically *different* outweighs evidence that *equivalent*
➔ ***Cannot be inconsistent if not (intended) to be equivalent!***

Initial Measurements

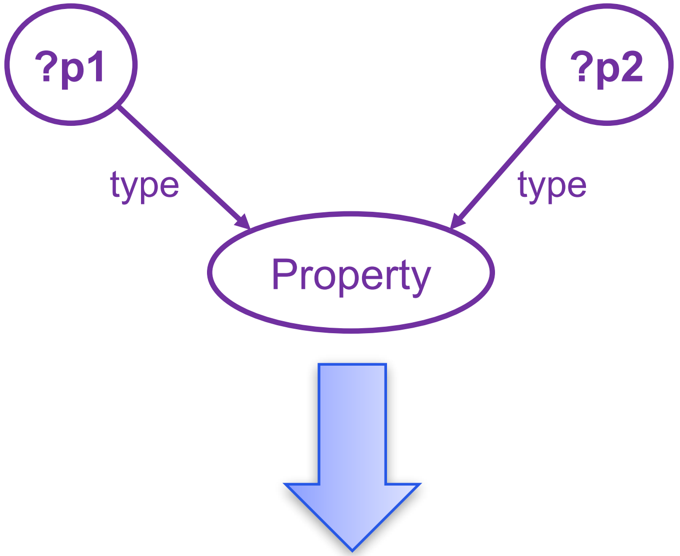
Insights



Comparison to Deterministic, Rule-Based Approach

- SPARQL pattern queries over same set of generated models
  - Detect semantic overlap
  - Identify inconsistencies

Dependent



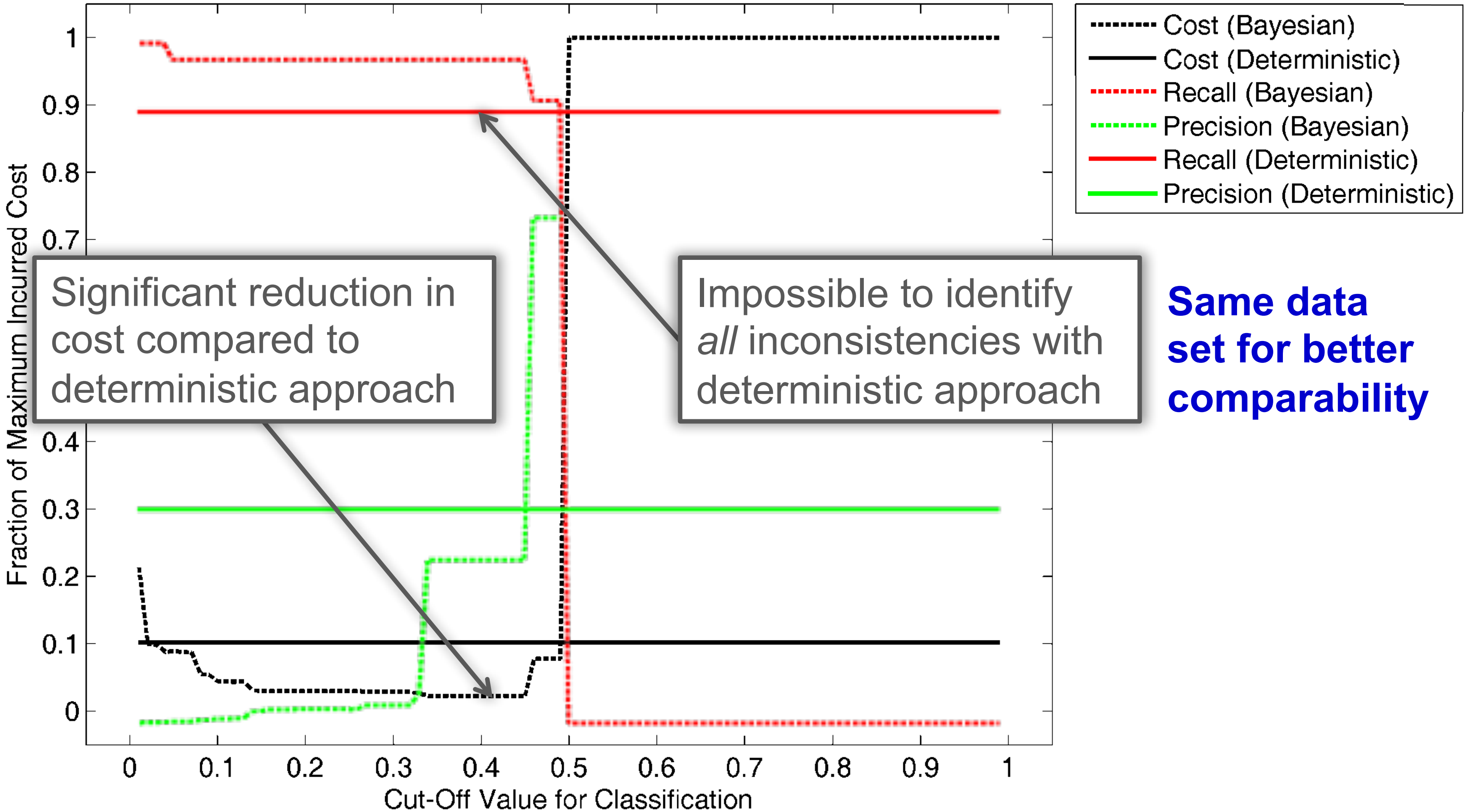

```
PREFIX base: <http://.../ns/base#>
SELECT ?p1 ?p2
WHERE {
   ?p1  base:type  base:Property .
   ?p2  base:type  base:Property .
}
```

- SPARQL a viable candidate due to its high expressivity, and W3C standard for querying RDF data

- Comparison:
  - Measured true and false positives and negatives
  - Calculated & compared incurred cost

Comparison to Deterministic Approach

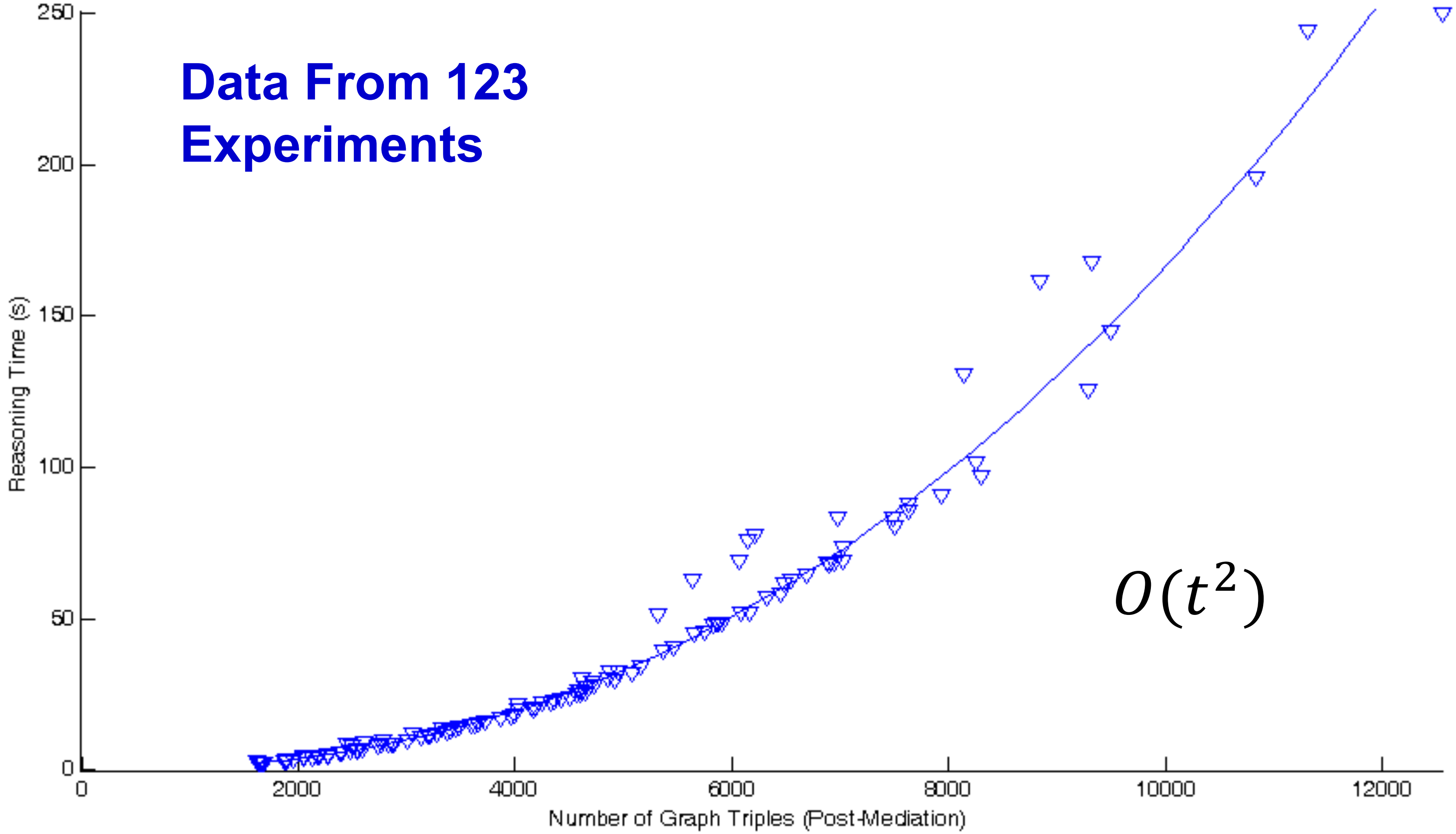
Measurements & Insights



Same data set for better comparability

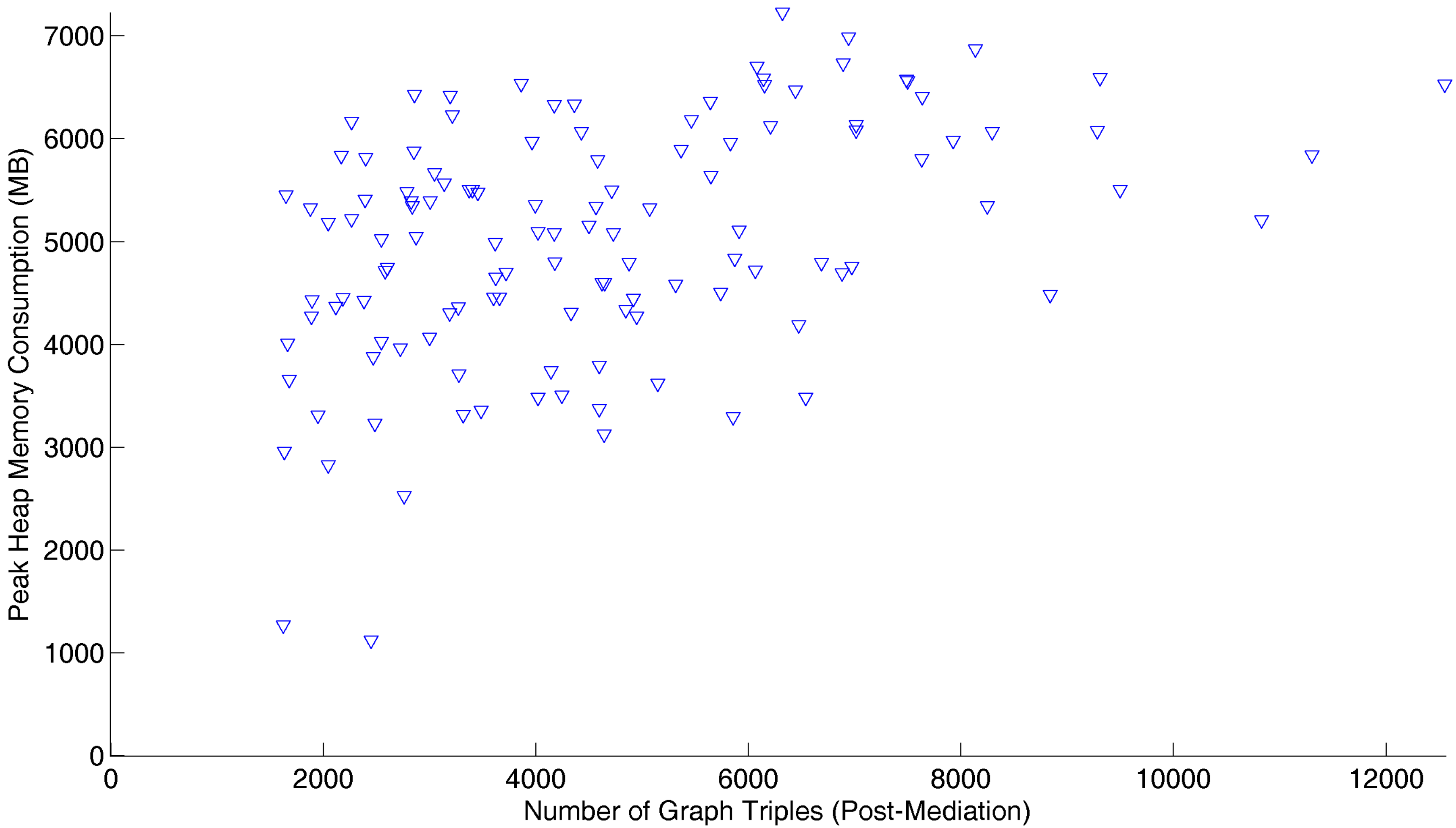
Efficiency: Performance of Algorithm

CPU Time (User + System Time)



Efficiency: Performance of Algorithm

Peak Heap Memory Usage



Outline

- Introduction
- Foundations & Methods
- Evaluation & Case Study
  - Characterization of Approach
  - Comparison to Deterministic Approach
- Conclusion

Limitations & Future Work

- Characterization & Evaluation → Performed under controlled (laboratory) conditions
- Limited Reasoning Scope → Currently very local
- Acquisition and Maintenance of Reasoning Knowledge → What is a good process for this in practice? Can data mining help identify patterns?
- Scalability → For very large graphs, can be slow – explore parallelization of, e.g., graph searches
- Application Beyond Inconsistency Identification → Explore other applications of developed inexact reasoning method

Conclusion

- Development of an approach, (incremental) algorithm, and proof-of-concept software tools for probabilistic reasoning over models
- Rigorous evaluation of approach
- Characterization & evaluation demonstrated:
  - Able to detect inconsistencies that otherwise overlooked
  - Even small sets of inconsistency identification knowledge effective
  - Bayesian learning improves performance
  - Better performance / more valuable than deterministic approach
  - Reasonable computational cost

➔ Bayesian approach significant improvement over status quo methods!

Publications

Herzig, S. J. I., Qamar, A., Paredis, C. J. J.: *A Conceptual Framework for Consistency Management in Model-Based Systems Engineering*. ASME International Design Engineering Technical Conference & Computers and Information in Engineering Conference (IDETC/CIE) 2011

Herzig, S. J. I., Rouquette, N. F., Forrest, S., Jenkins, S. J.: *Integrating Analytical Models with Descriptive System Models: Implementation of the OMG SyML Standard for the Tool-Specific Case of MapleSim and MagicDraw*. Conference on Systems Engineering Research 2013

Qamar, A., **Herzig, S. J. I.**, Paredis, C. J. J.: *A Domain-Specific Language for Dependency Management in Model-Based Systems Engineering*. Workshop on Multi-Paradigm Modeling (MPM) at MoDELS 2013

Herzig, S. J. I., Paredis, C. J. J.: *A Conceptual Basis for Inconsistency Management in Model-Based Systems Engineering*. CIRP Design 2014

Herzig, S. J. I., Qamar, A., Paredis, C. J. J.: *An Approach to Identifying Inconsistencies in Model-Based Systems Engineering*. Conference on Systems Engineering Research 2014

Herzig, S. J. I., Qamar, A., Paredis, C. J. J.: *Inconsistency Management in Model-Based Systems Engineering*. 2014 Global Product Data Interoperability Summit

Herzig, S. J. I., Kruse, B., Ciccozzi, F., Denil, J., Salay, R., Varro, D.: *Towards an Approach for Orchestrating Design Space Exploration Problems to Fix Multi-Paradigm Inconsistencies*. Workshop on Multi-Paradigm Modeling (MPM) at MoDELS 2014

Herzig, S. J. I., Paredis, C. J. J.: *Bayesian Reasoning over Models*. Workshop on Model-Driven Engineering, Verification, and Validation (MoDeVVa) at MoDELS 2014

Qamar, A., **Herzig, S. J. I.**, Paredis, C. J. J., Toerngren, M.: *Analyzing Semantic Relationships between Multi-Formalism Models for Inconsistency Management*. IEEE Systems Conference 2015 (accepted)

Feldmann, S., **Herzig, S. J. I.**, Kernschmidt, K., Wolfenstetter, T., Kammerl, D., Qamar, A., Lindemann, U., Krcmar, H., Paredis, C. J. J., Vogel-Heuser, B.: *Towards Effective Management of Inconsistencies in Model-Based Engineering of Automated Production Systems*. IFAC Symposium on Information Control in Manufacturing (INCOM) 2015 (accepted)

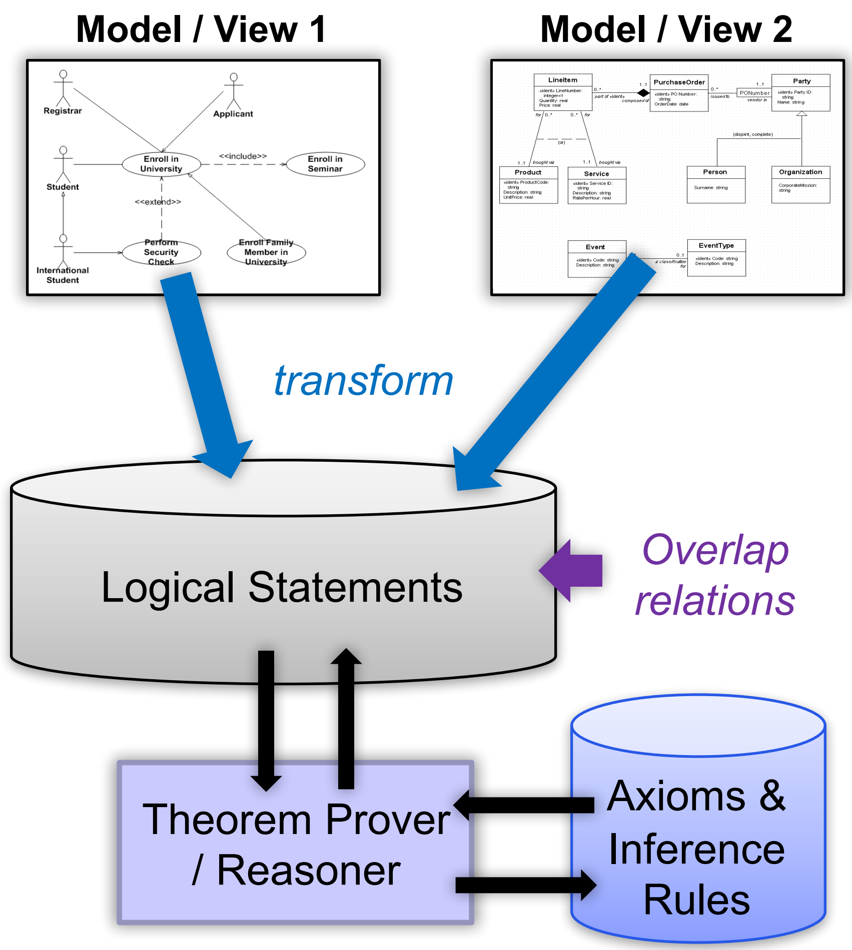
Feldmann, S., **Herzig, S. J. I.**, Kernschmidt, K., Wolfenstetter, T., Kammerl, D., Qamar, A., Lindemann, U., Krcmar, H., Paredis, C. J. J., Vogel-Heuser, B.: *A Comparison of Inconsistency Management Approaches Using a Mechatronic Manufacturing System Design Case Study*. IEEE International Conference on Automation Science and Engineering (CASE) 2015 (submitted)

Herzig, S. J. I., Paredis, C. J. J.: *Probabilistic Inexact Reasoning over Graph-Based Models (tentative)*. Journal of Artifical Intelligence (in progress)

BACKUP SLIDES

Literature: Theorem Proving Based Approaches

- Core idea: translate models to logical statements and construct proof for determining whether or not consistent with formal system
  - Syntactic transformations
  - Semantic satisfiability test
- Logically correct conclusions (within bounds of formal system)
- Practical issues:
  - Requires complete and consistent definition of formal system
  - Generally semi- or un-decidable
  - Some manual translation necessary



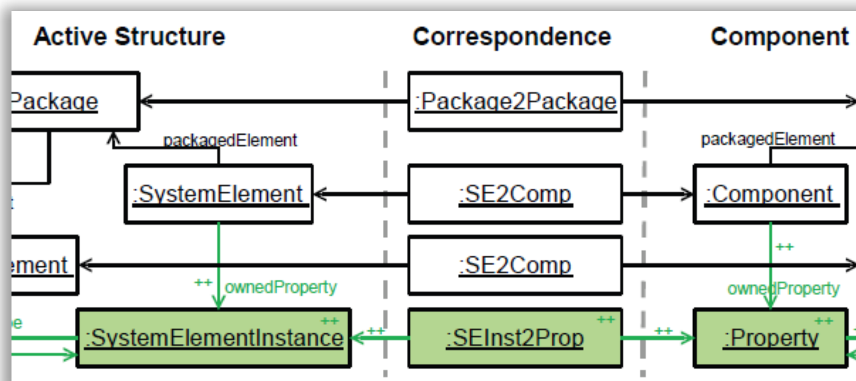
(e.g., using FOL: Finkelstein et al., 1994, using DL: Van der Straeten et al., 2006)

Literature: Rule-Based Approaches

- Aspect of model heterogeneity addressed in two main ways:
  - n*(n-1) transformations
  - Transformation to central model
- Identification of inconsistencies by searching for sufficient conditions (e.g., using patterns)
- Semantic overlap detection:
  - Syntactic matching
  - Tagging with ontology elements
  - Human inspection
  - Similarity analysis

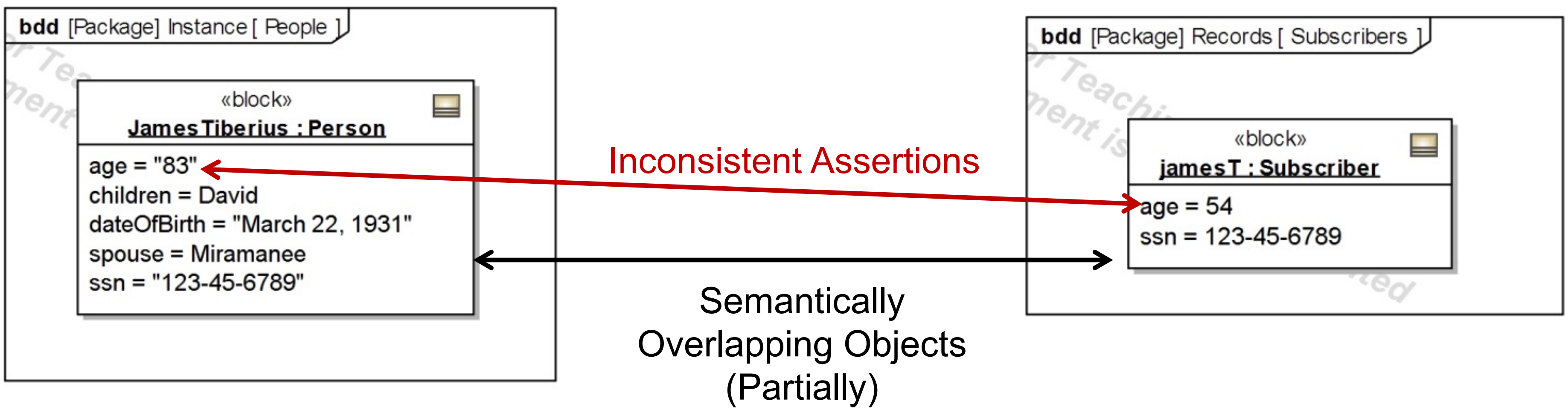
Rule1: *applies to <Electrical Drive> requires(this).contains(<Power Electronics>)*

Mechatronic Design Ontology & Consistency Rules (Hehenberger et al., 2010)



Using TGG Transformation Rules between Domain-Specific Models and a Holistic System Model (Gausemeier, 2009 – 2014)

Example Inconsistency



- Are we really talking about the same person in the models?
  - *If so*, age is inconsistent
  - *If not*, then SSN assignment is erroneous

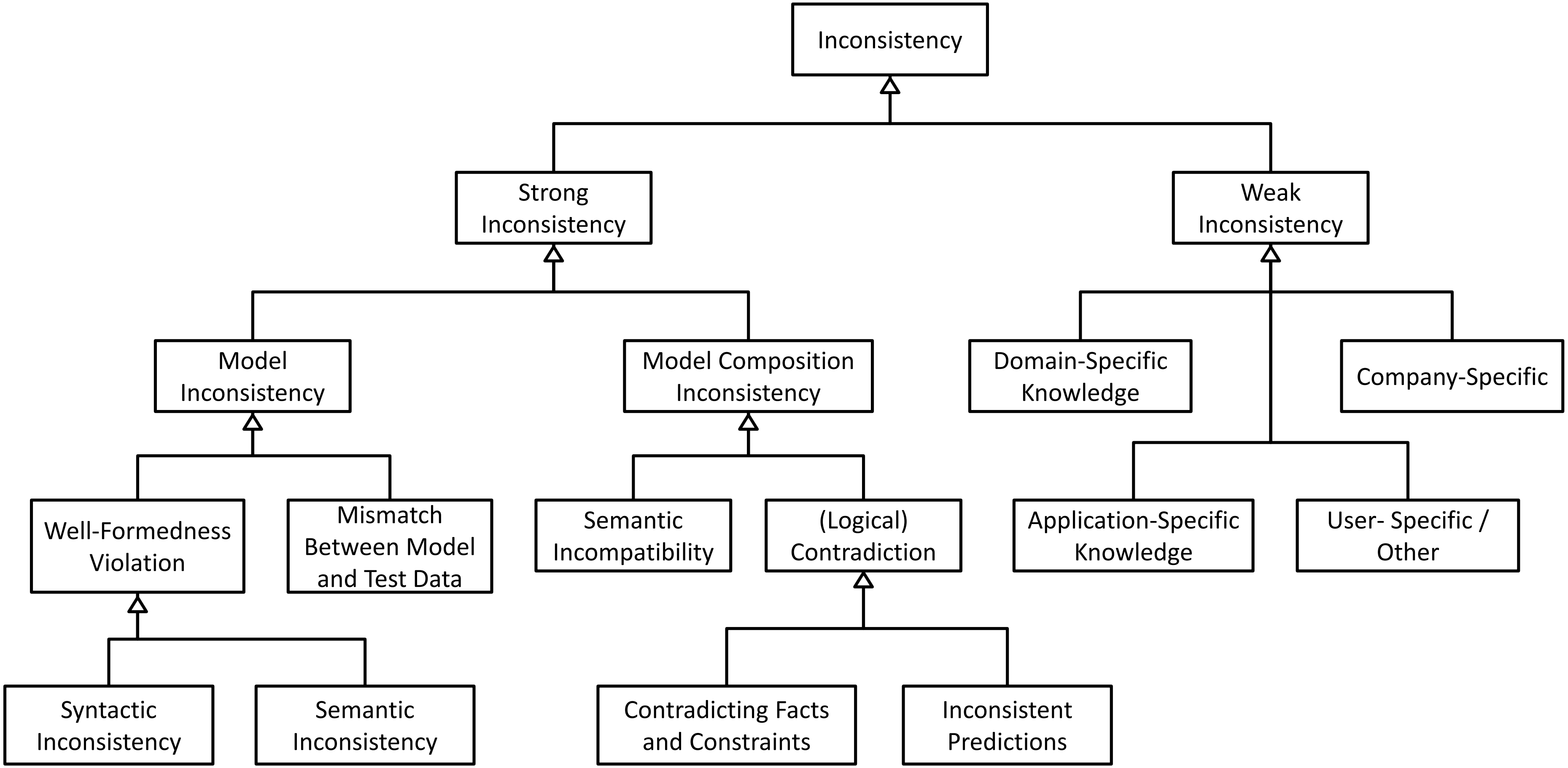
Types of Inconsistencies in Models

- Violation of Well-Formedness Rules
  - Grammar rules
  - Structural semantics
- Inconsistencies in Redundant Information
  - Multiple redundant specifications
  - Multiple redundant analyses
  - Analyses based on parts of specifications
- Not following Guidelines / Best Practices
  - Style guides / naming conventions
  - DFMA / ergonomics guidelines
  - Heuristics / expert knowledge
  - Domain knowledge
  - Adherence to process (precedence relationships)

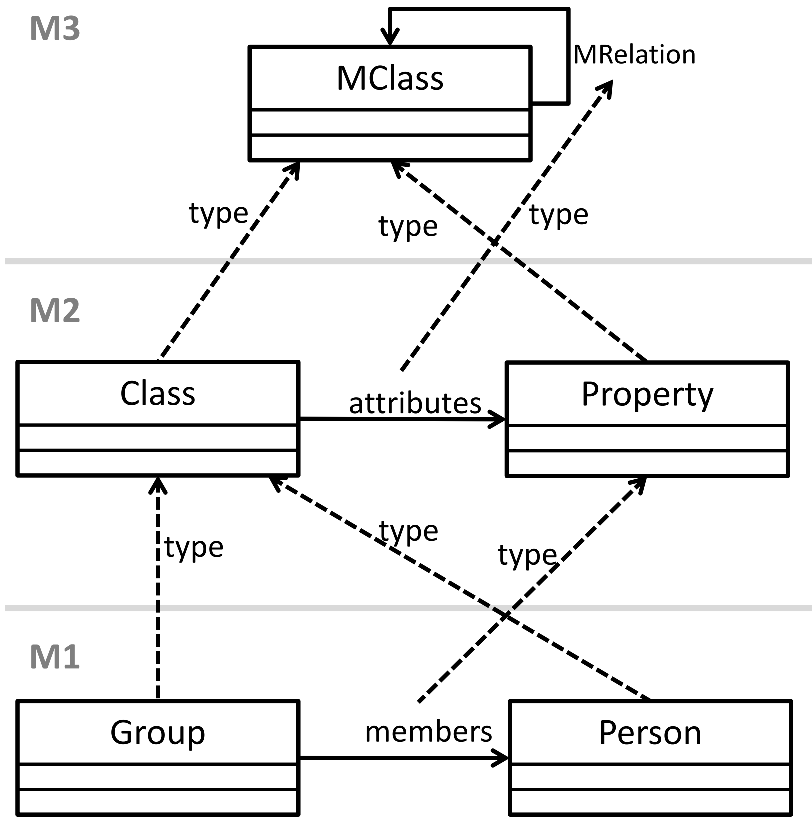
"true" inconsistencies (Violation of Well-Formedness Rules; Inconsistencies in Redundant Information)

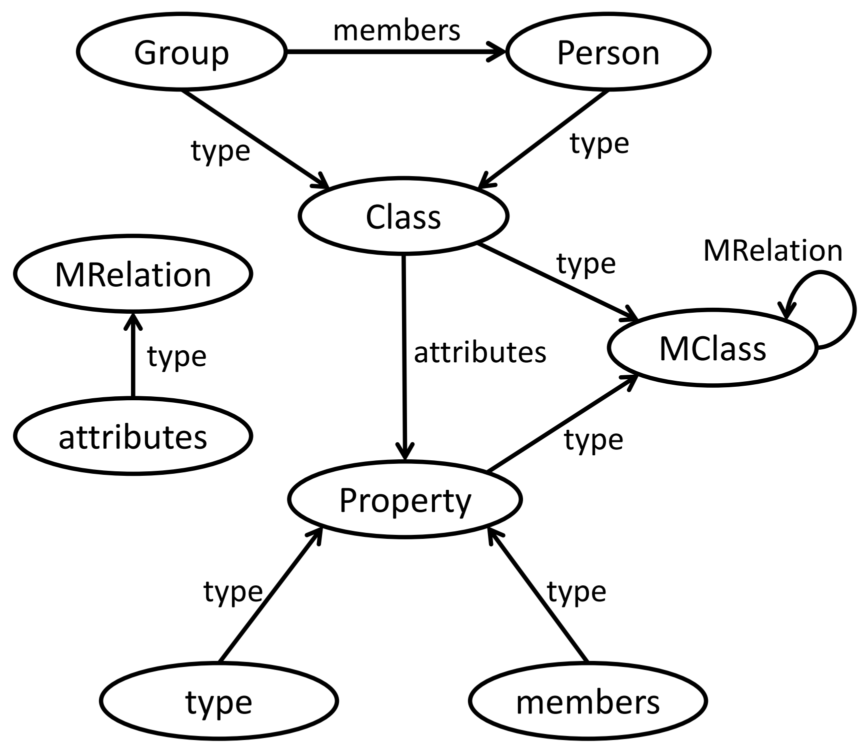
"weak" inconsistencies (Not following Guidelines / Best Practices)

Classification



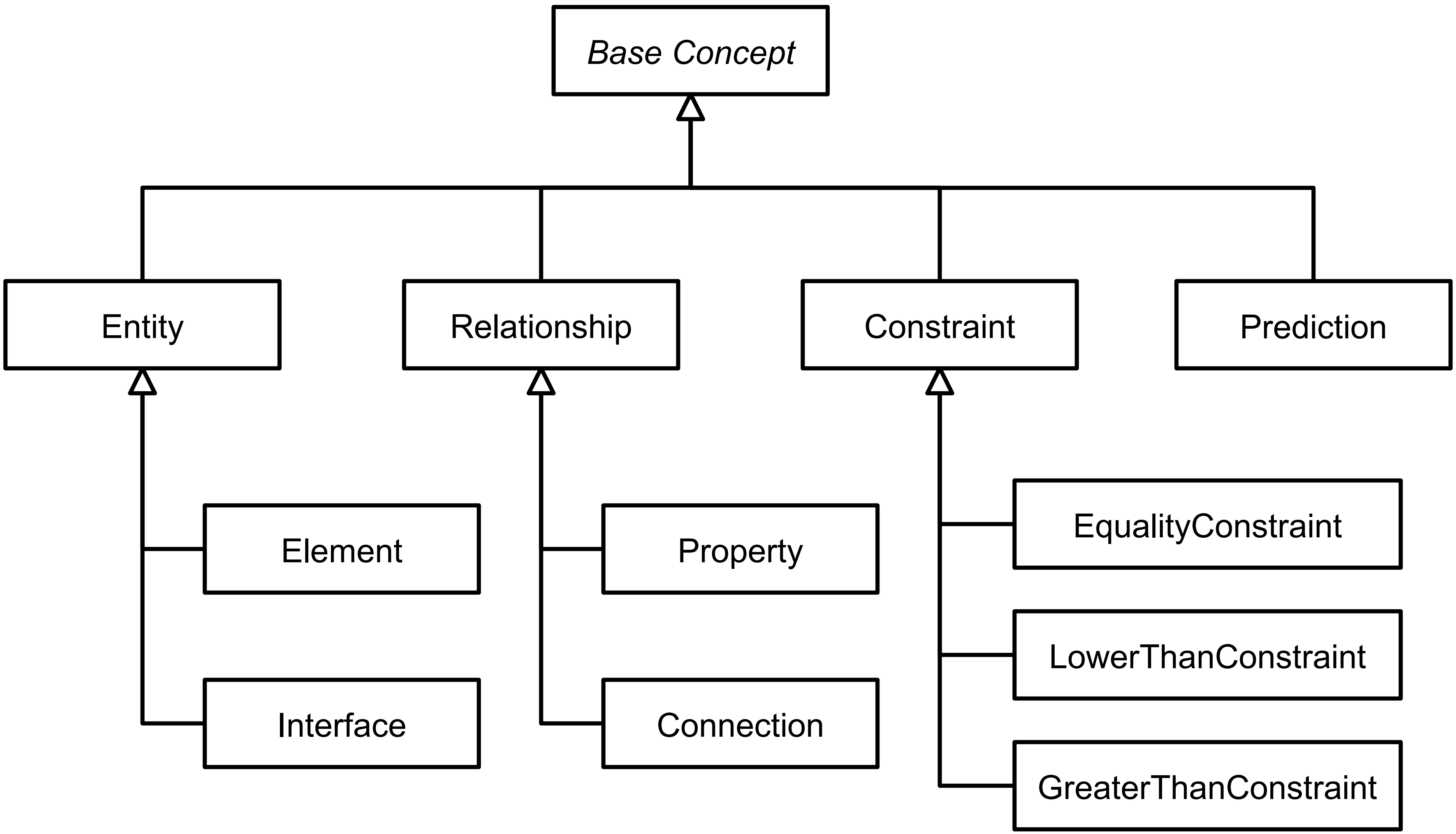
Sample Transformation





- Every term (denoting properties, classes, etc.) is defined by a vertex
- The *name* is the term from the source language
- Relationships are edges with mappings to vertices (definitions)

Base Vocabulary



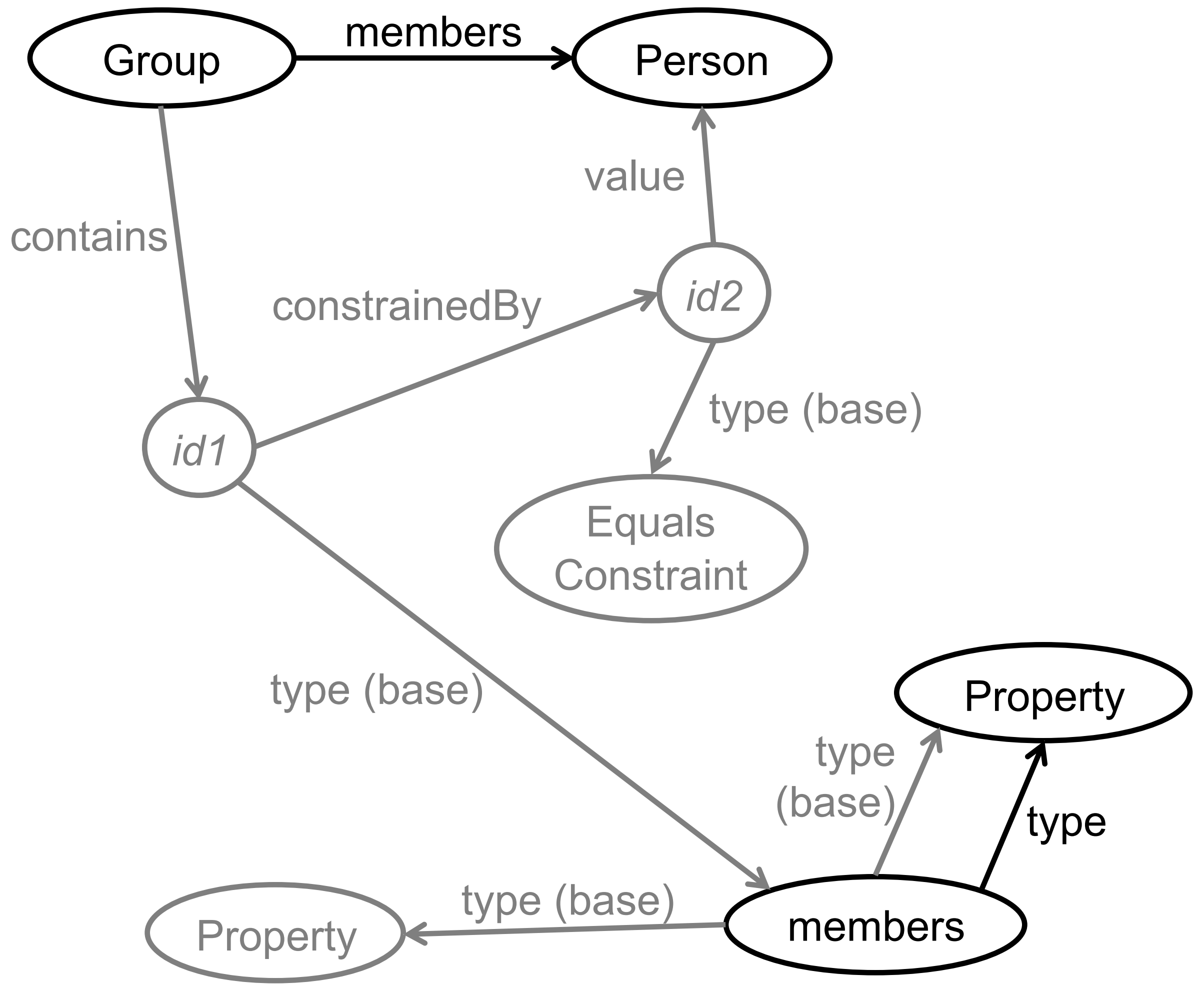
Base Vocabulary

Property	Domain	Range	Description
type	BaseConcept	BaseConcept	Instance relation; relation between a general concept and individual instances.
contains	BaseConcept	BaseConcept	(Weak) containment relation (e.g., part-whole or object-property).
containedIn	BaseConcept	BaseConcept	Inverse of above.
generalizationOf	BaseConcept	BaseConcept	Hyponymous relation; denotes (sub-)class membership.
specializationOf	BaseConcept	BaseConcept	Inverse of above.
domain	Relation	BaseConcept	The source domain of a relation.
range	Relation	BaseConcept	The target domain of a relation.
equivalentTo	BaseConcept	BaseConcept	Semantic equivalence of base concept; synonymy; identifies that both source and target elements have the same meaning.
differentFrom	BaseConcept	BaseConcept	Inverse of above.

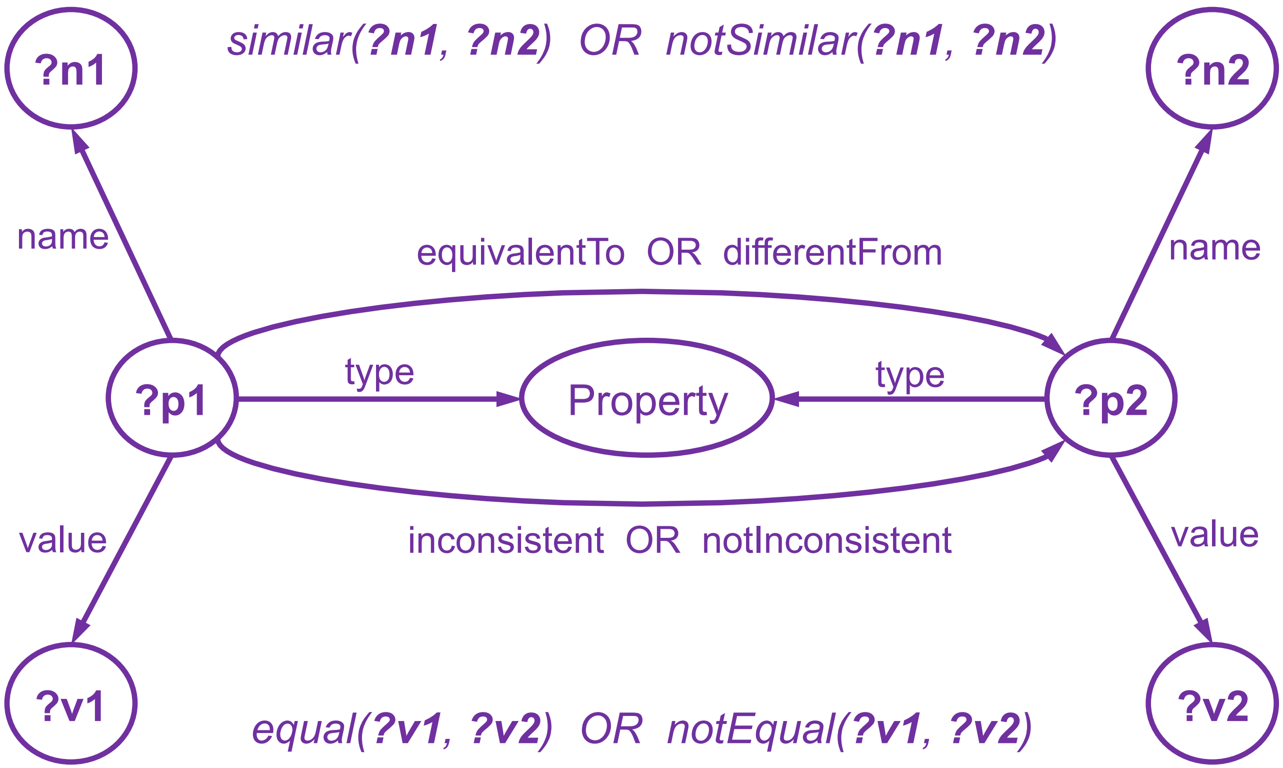
Base Vocabulary

Property	Domain	Range	Description
name	BaseConcept	Element	Identifier object.
value	Constraint	Element	Value associated with a constraint.
unitType	Constraint	Entity	Unit type associated with a constraint value.
constrainedBy	BaseConcept	Constraint	An applied constraint.

Sample Mediation



Partial Pattern Matching & Mutual Exclusion



Sample Mediation Rule

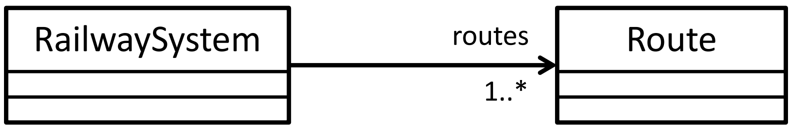
Apache Jena Datalog Implementation Syntax

```
[ p1:
    (?x rdf:type railway:RailwaySystem)
    (?x ?prop ?y)
    (?prop rdf:type railway_rs:routes)
    uriConcat(?prop, '/constr', ?c)
    →
    (?x base:contains ?prop)
    (?prop rdf:type base:Property)
    (?prop base:type railway_rs:routes)
    (?prop base:constrainedBy ?c)
    (?c rdf:type base:EqualsConstraint)
    (?c base:value ?y)
]
```

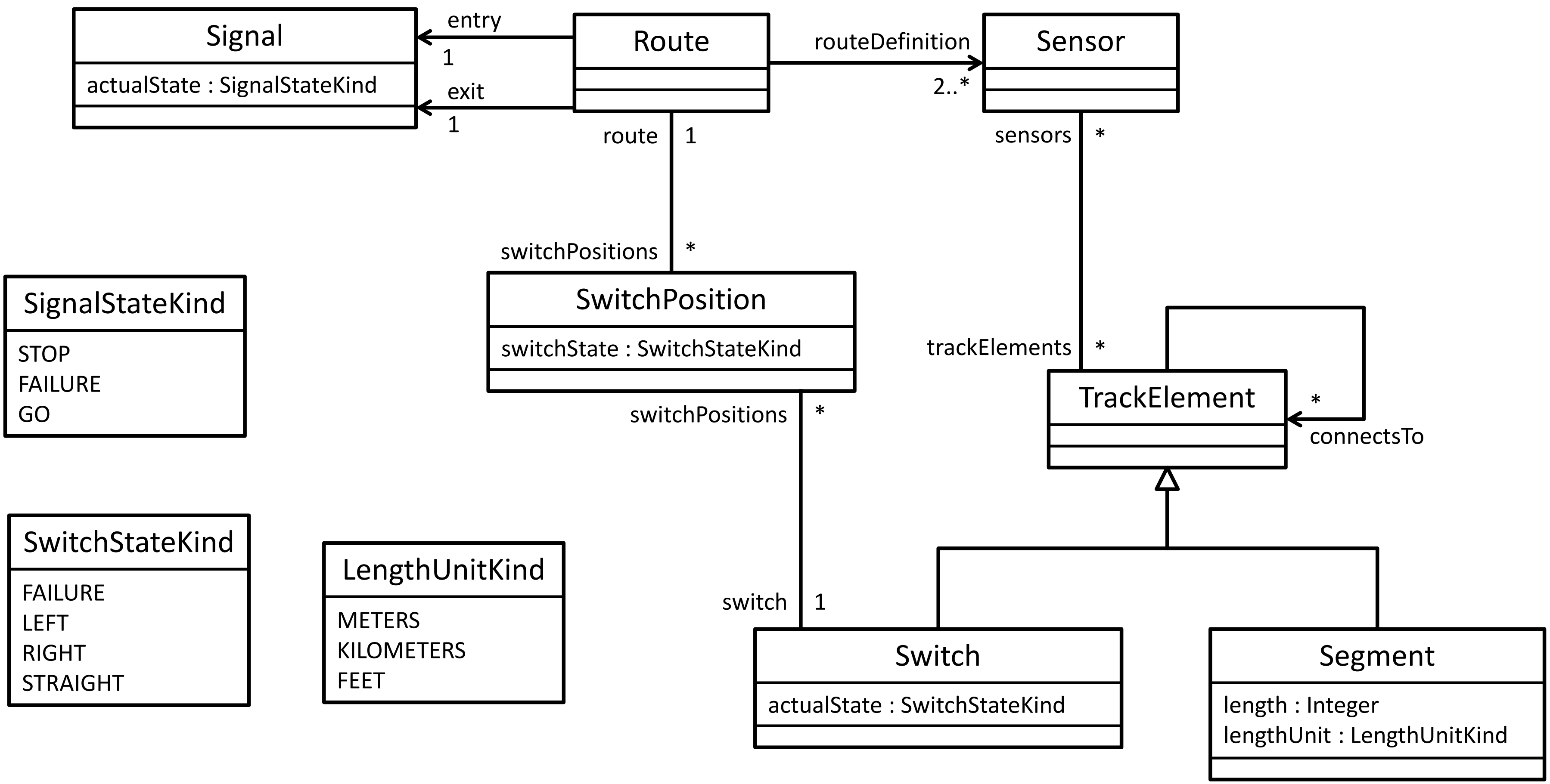
Antecedent

Consequent

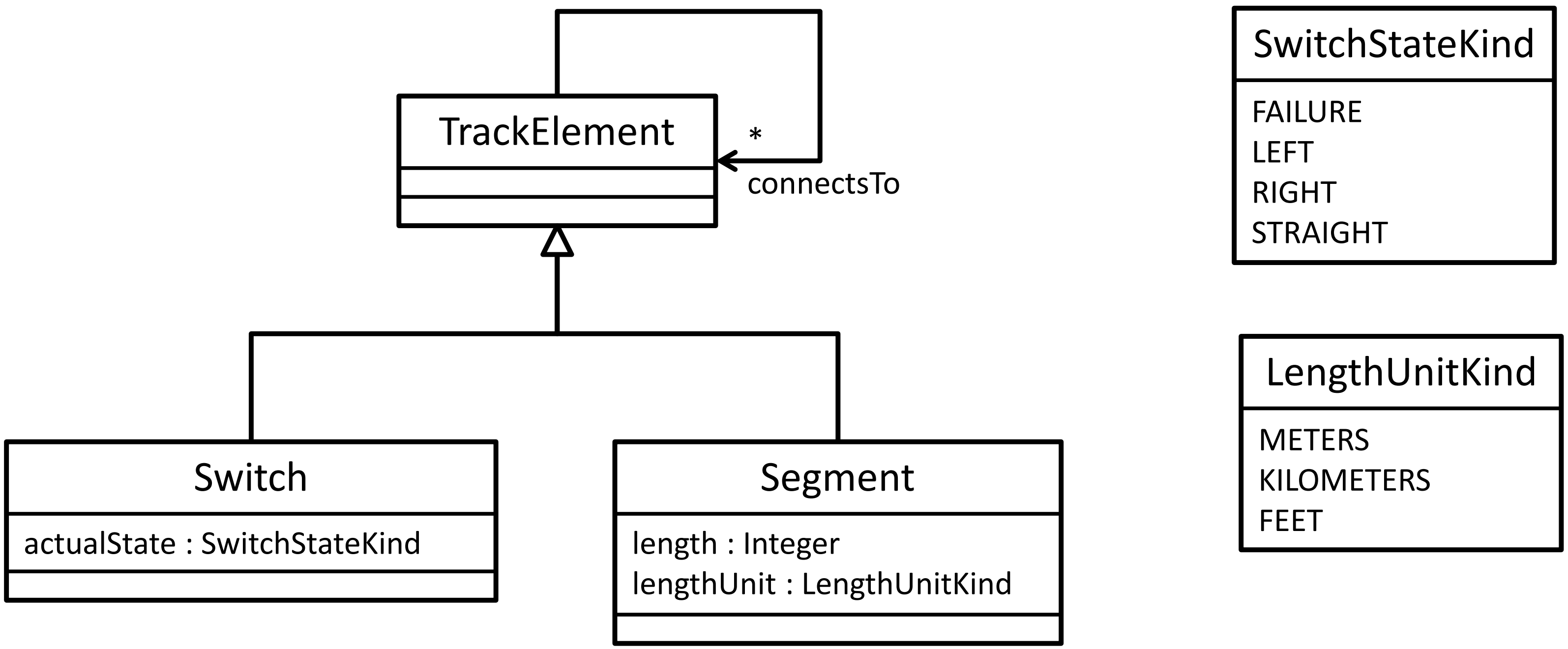
Case Study: “System Model” Meta-Model



Case Study: “Route Model” Meta-Model

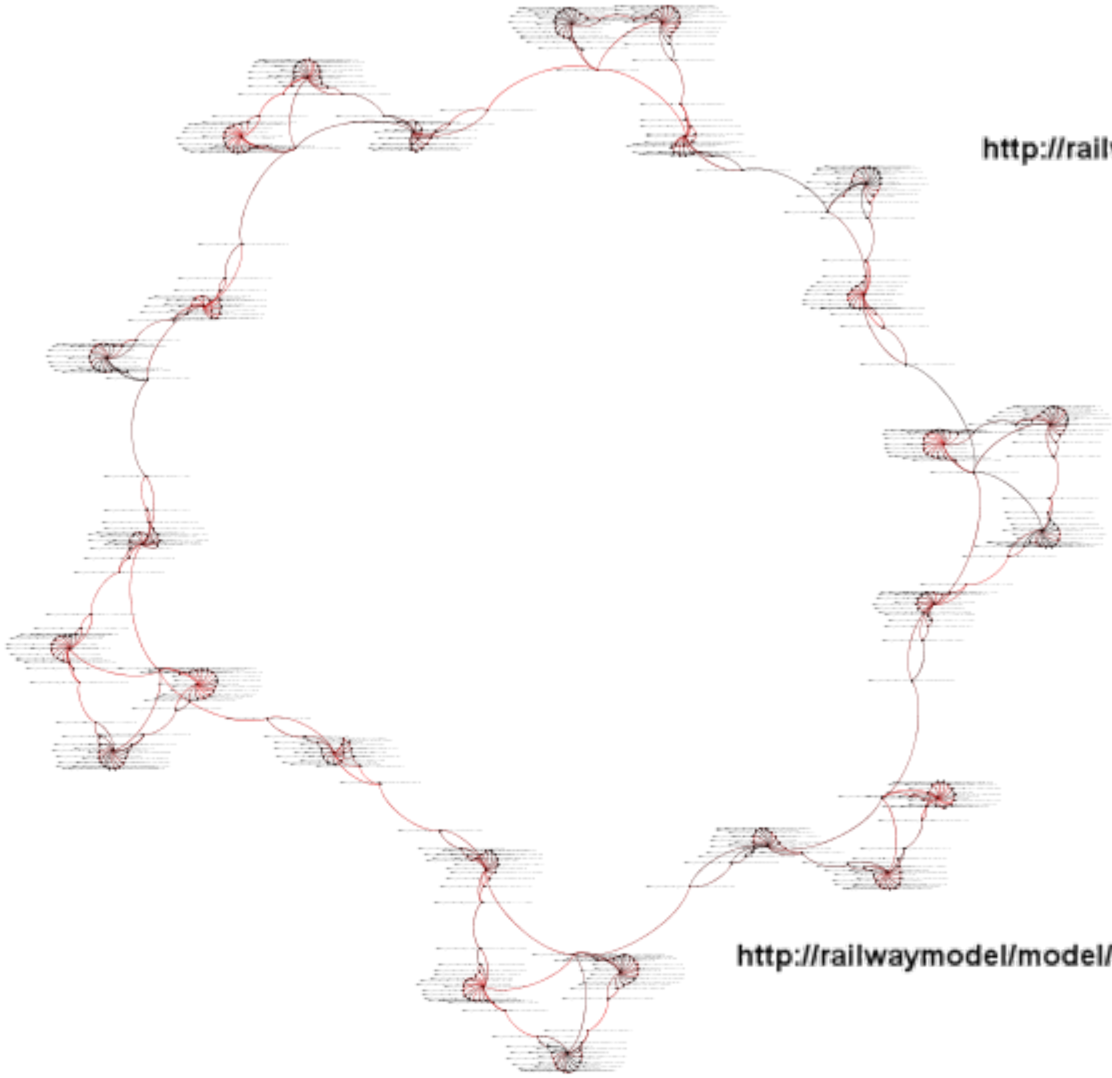


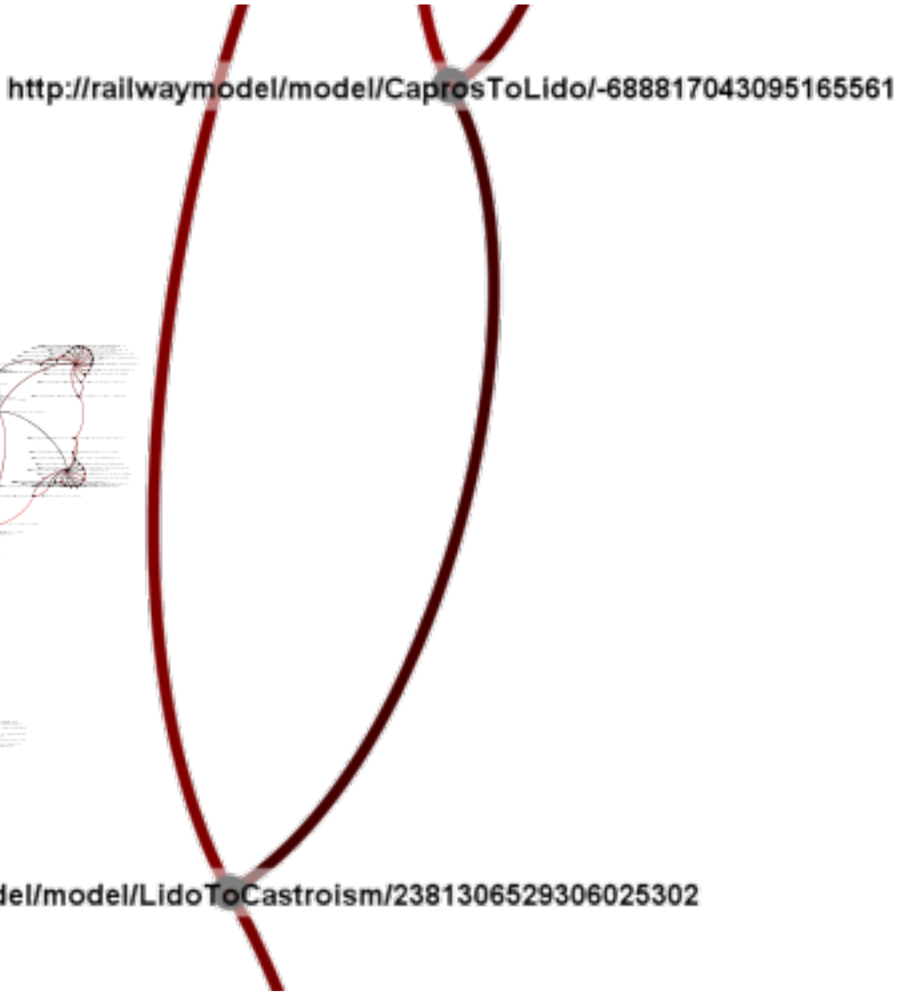
Case Study: “Track Network Model” Meta-Model



RDF Example

Excerpt – Showing Only Track Elements and Sensors

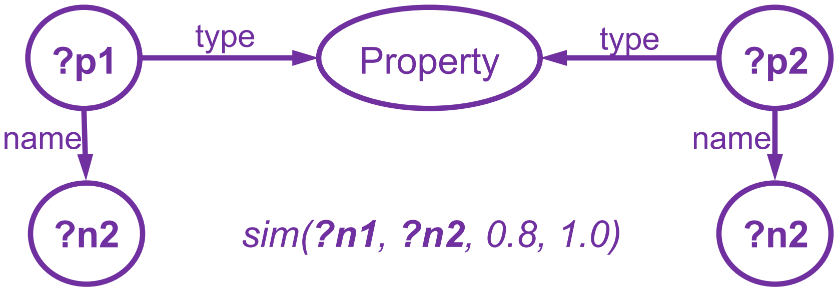




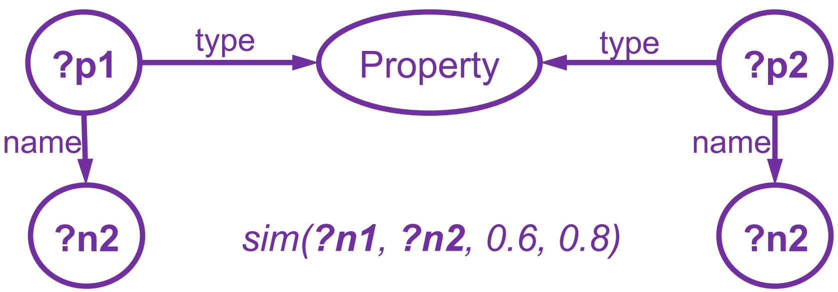
Patterns from Evaluation Example (Excerpt)

PropertyNames SimilarityScore =

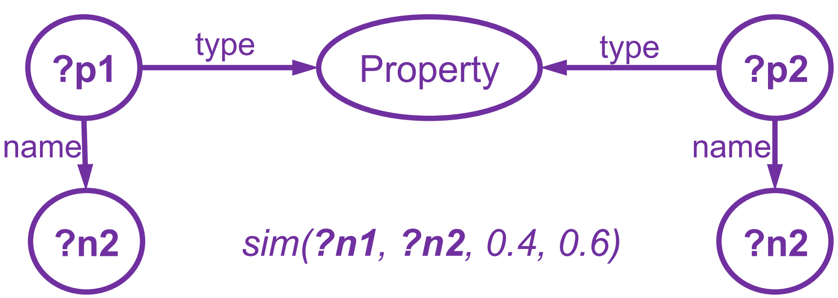
… is between 0.8 and 1.0 ←



… is between 0.6 and 0.8 ←



… is between 0.4 and 0.6 ←



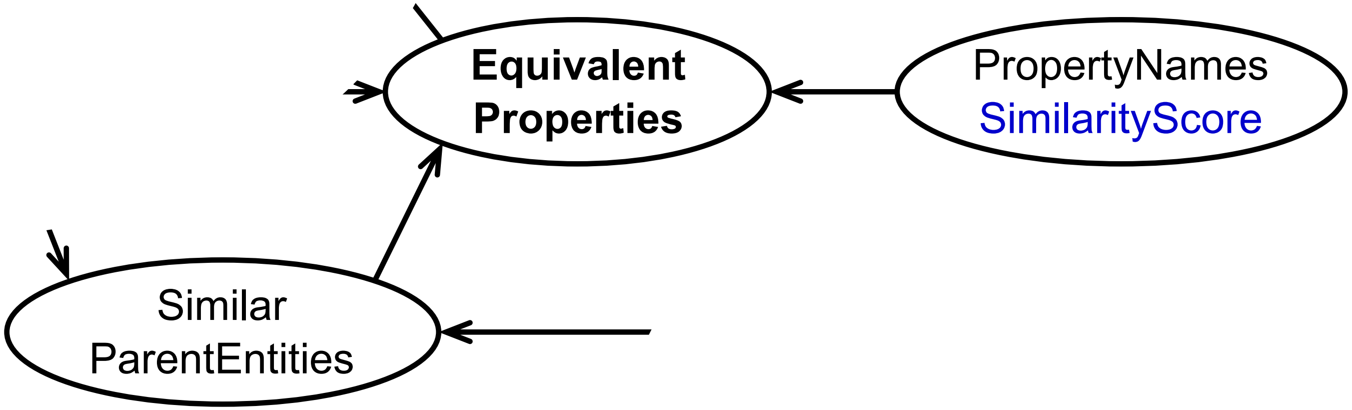
…

Network Parameters

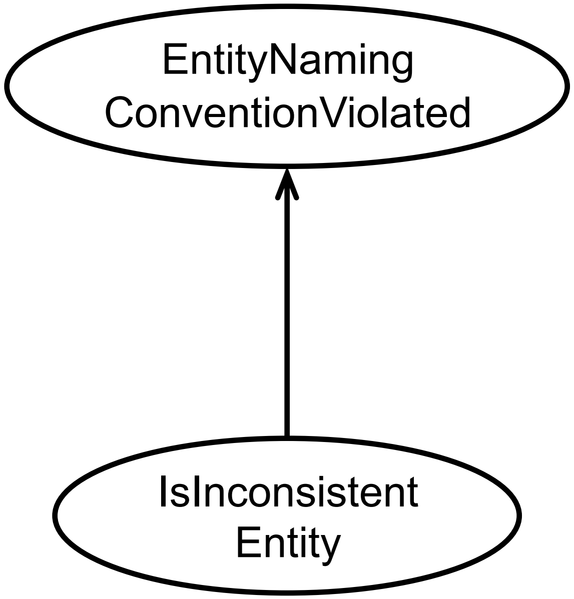
$$P(\text{Equiv. Prop.} \mid \text{SimilarPar.},\ \text{NameSimilarit} = \text{between } 0.8 \text{ and } 1.0) = 0.9$$

$$P(\text{Equiv. Prop.} \mid \text{SimilarPar.},\ \text{NameSimilarit} = \text{between } 0.6 \text{ and } 0.8) = 0.6$$

...



Alternative Pattern – Style Guide

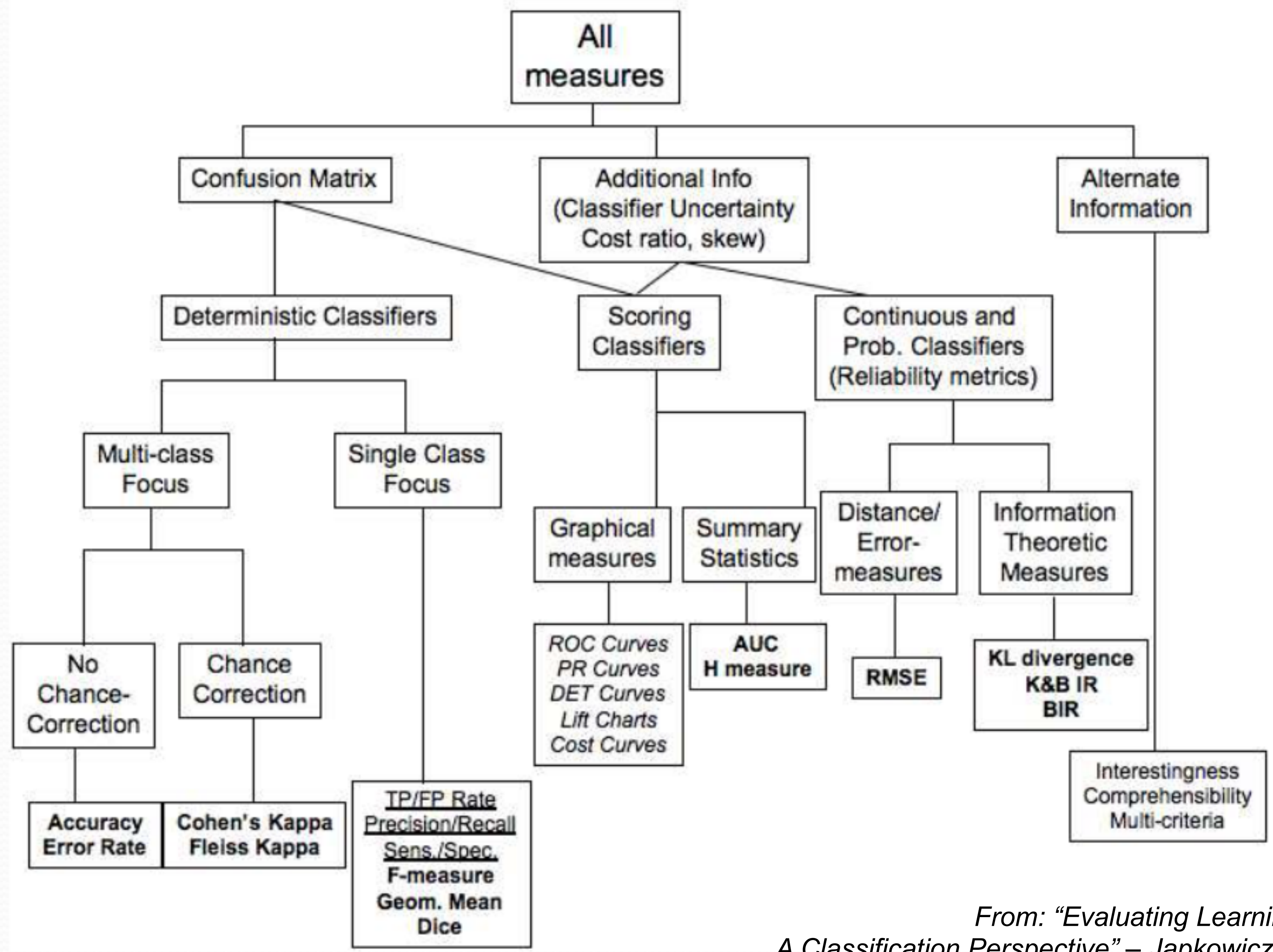


E.g., pattern for ‘not violated’:

(?x base:name ?n) regex(?n, “^[A-Z-\']+[a-z0-9-\']*(([A-Z0-9-\']|[a-z0-9-\'])*)*$”)

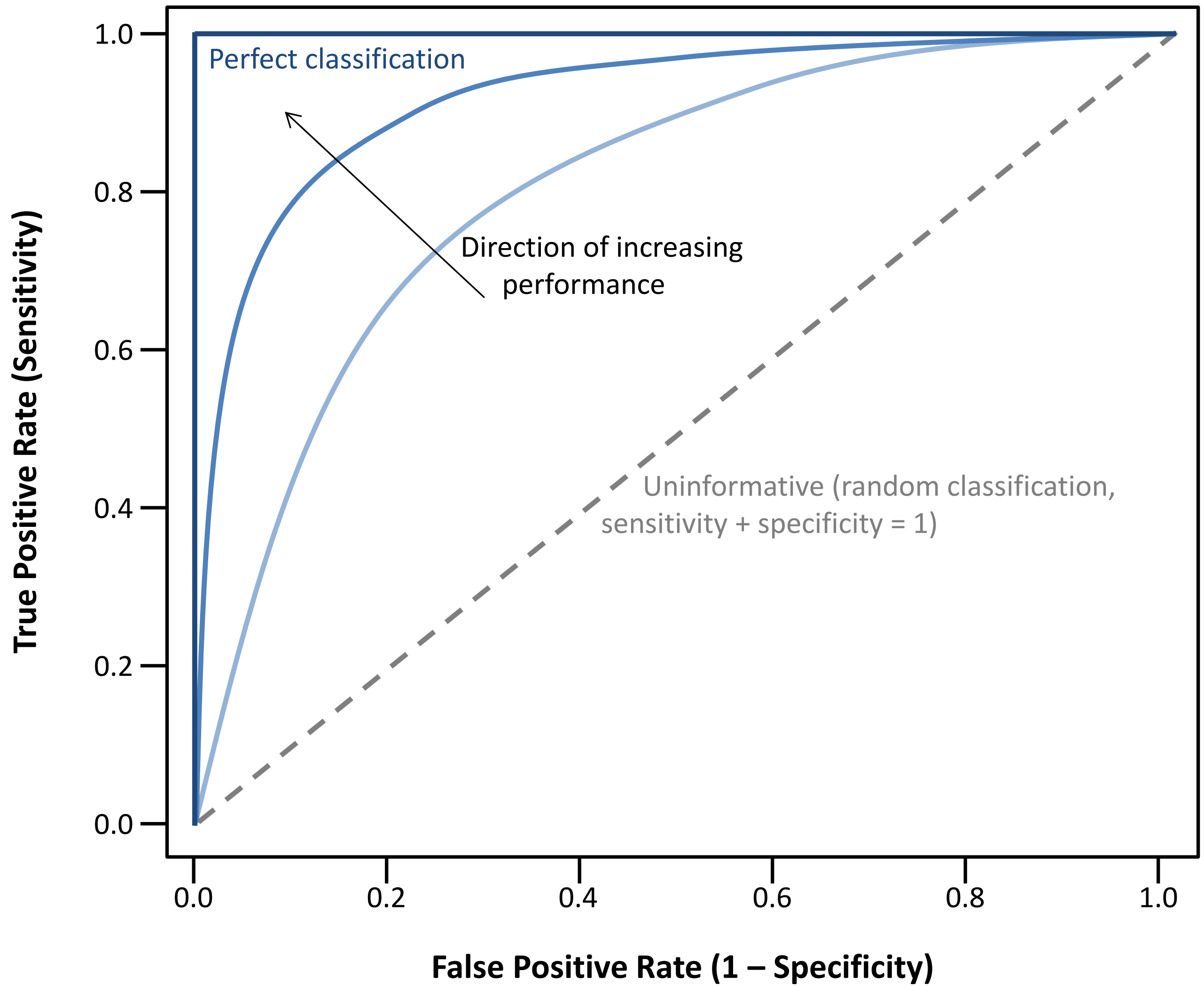
I.e., UpperCamelCase, no special characters other than “-” and “'” allowed, no leading number

Evaluation Measures for Learning Algorithms

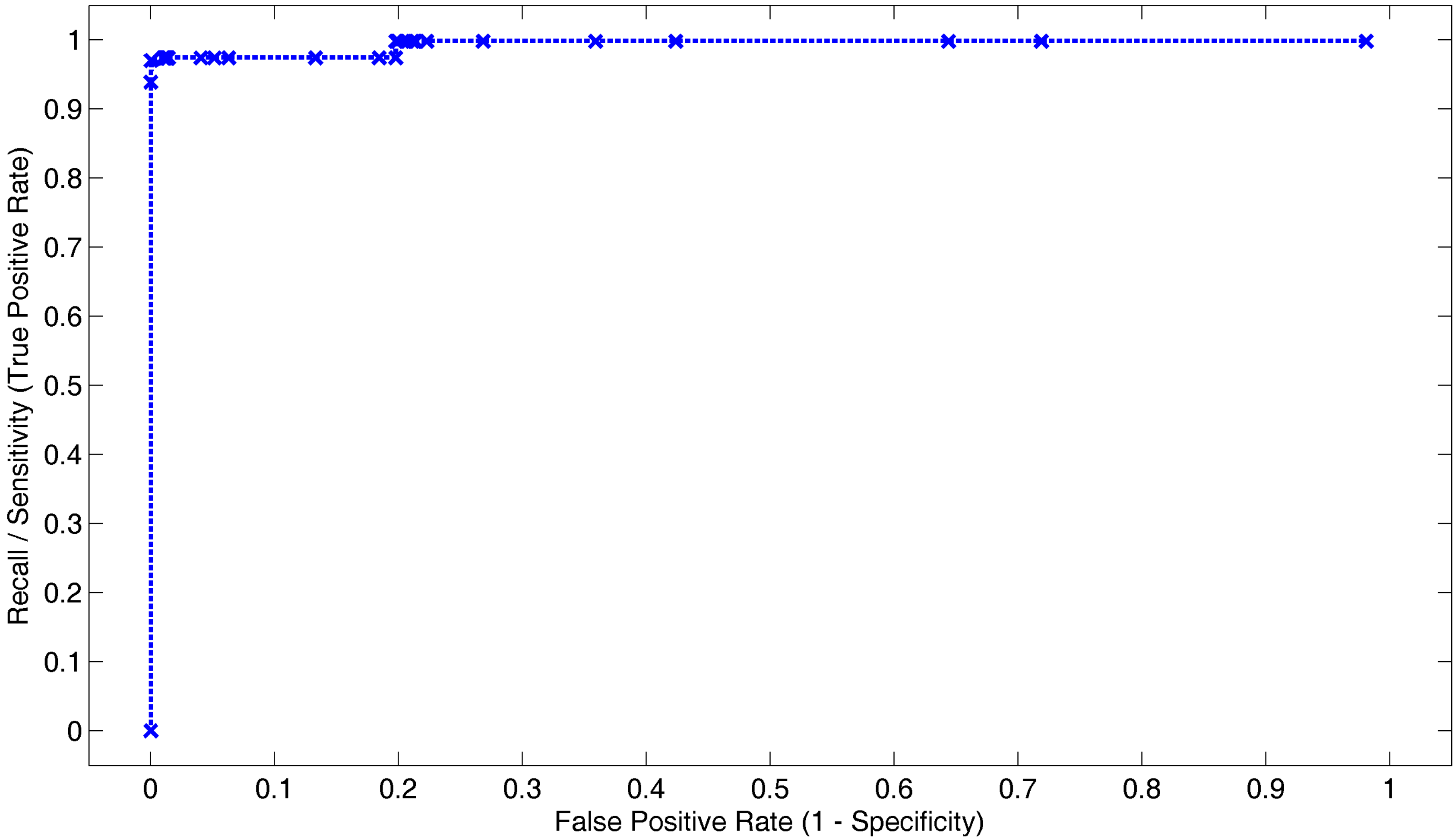


From: "Evaluating Learning Algorithms: A Classification Perspective" – Japkowicz & Shah, 2011

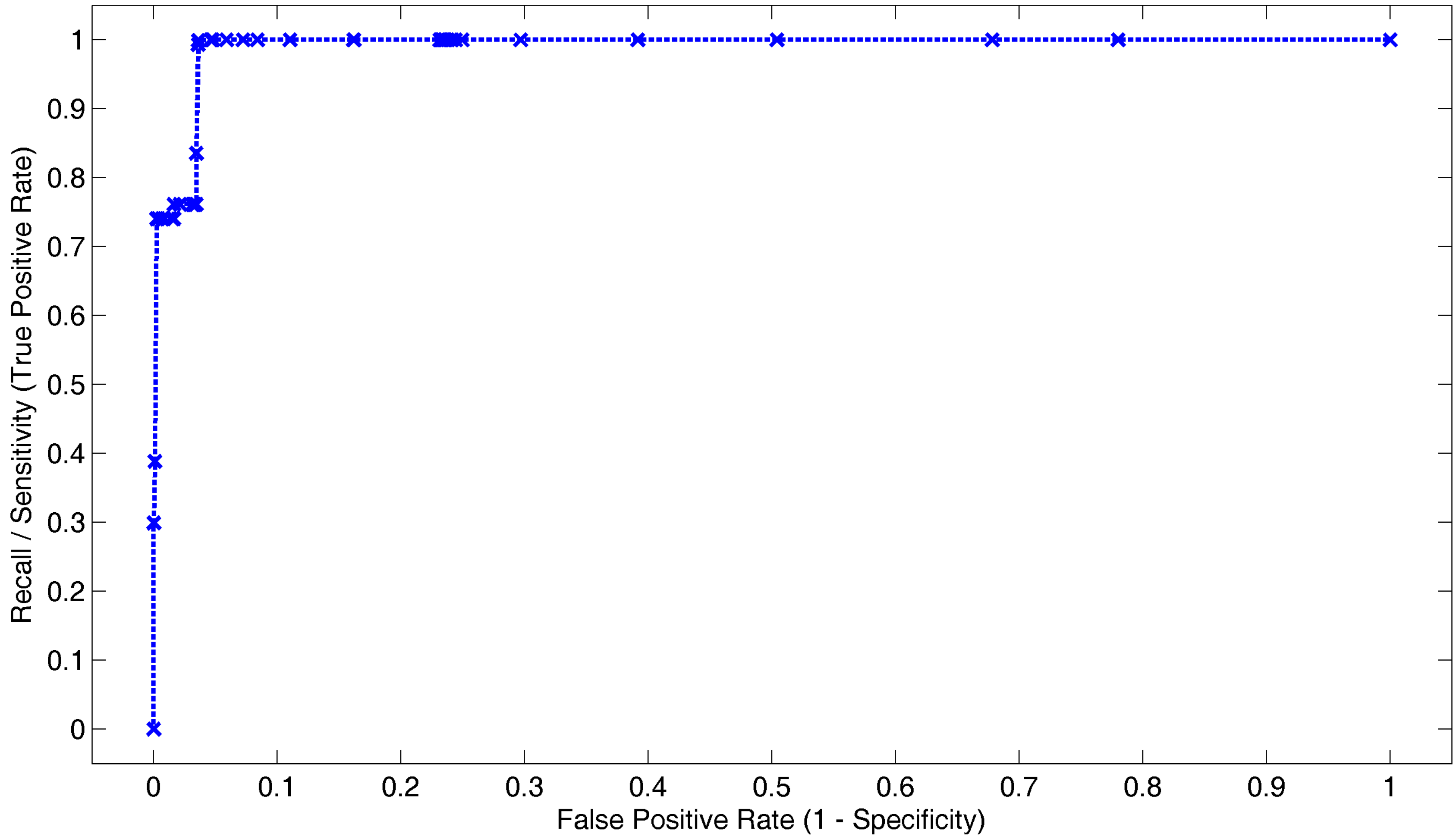
ROC Curves


Perfect classification
Direction of increasing performance
Uninformative (random classification, sensitivity + specificity = 1)
True Positive Rate (Sensitivity)
False Positive Rate (1 – Specificity)
0.0
0.2
0.4
0.6
0.8
1.0

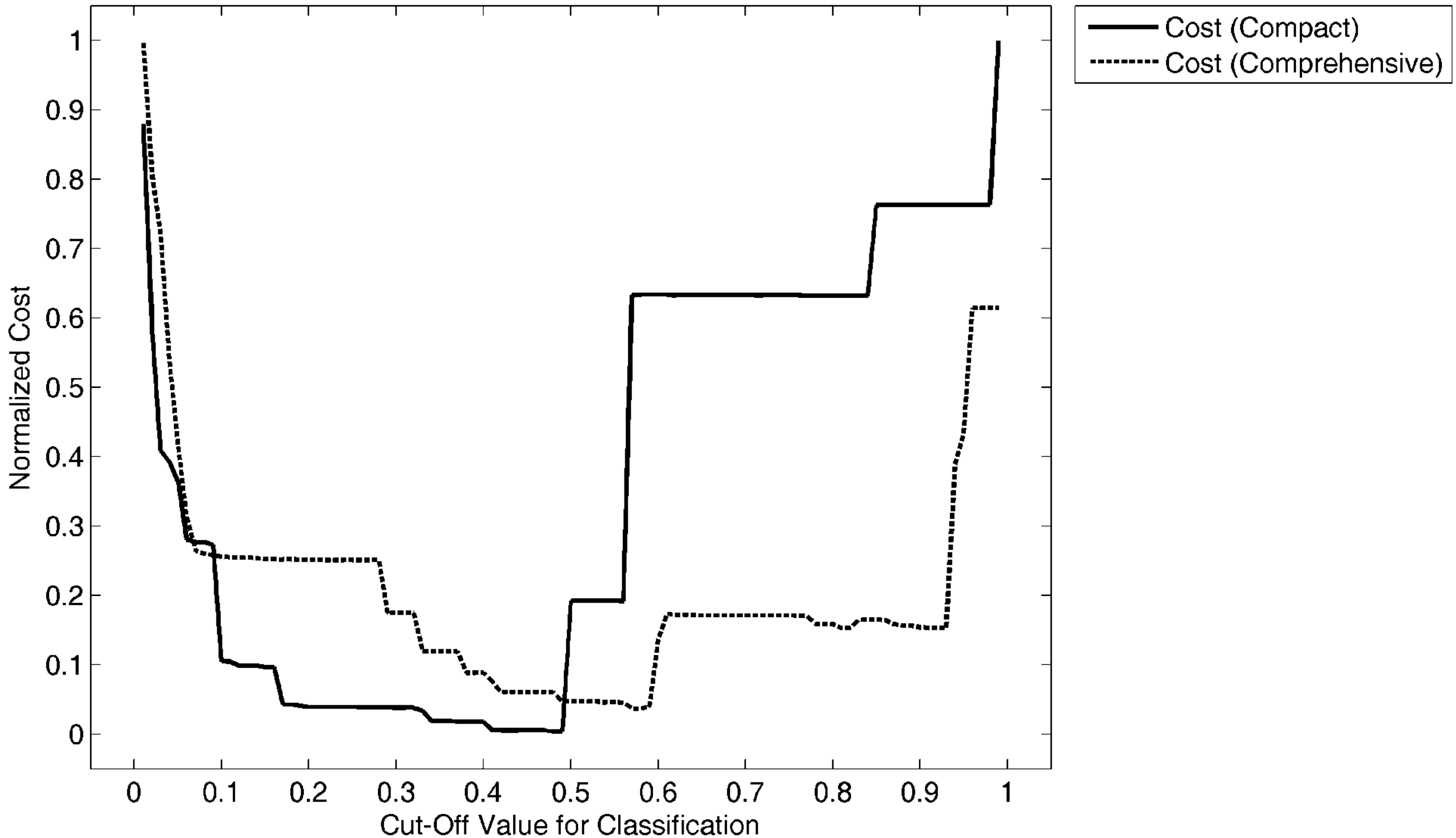
ROC: Inconsistency Classifier



ROC: Equivalence Classifier

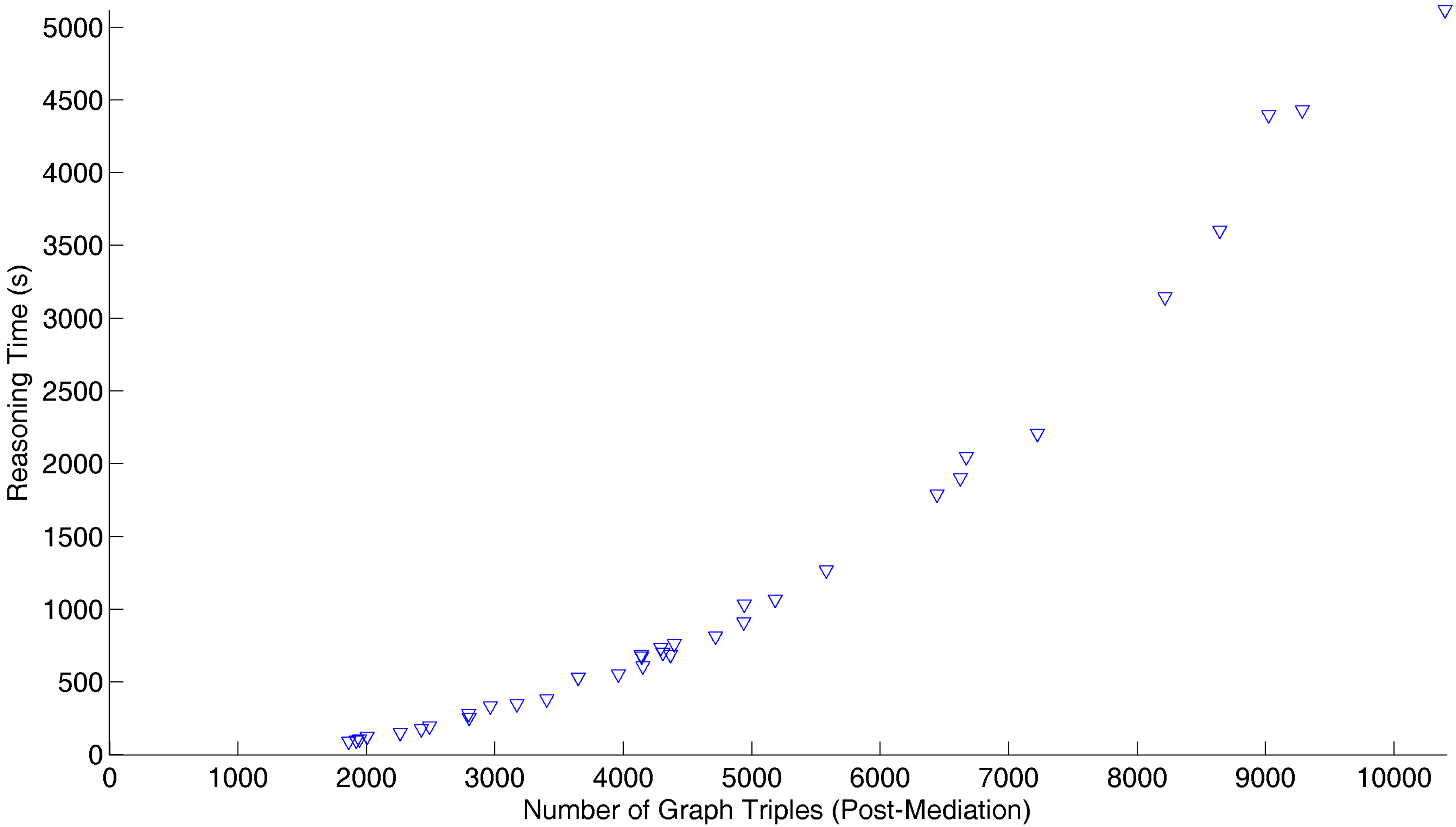


Comparing Classifiers: Semantic Equivalence



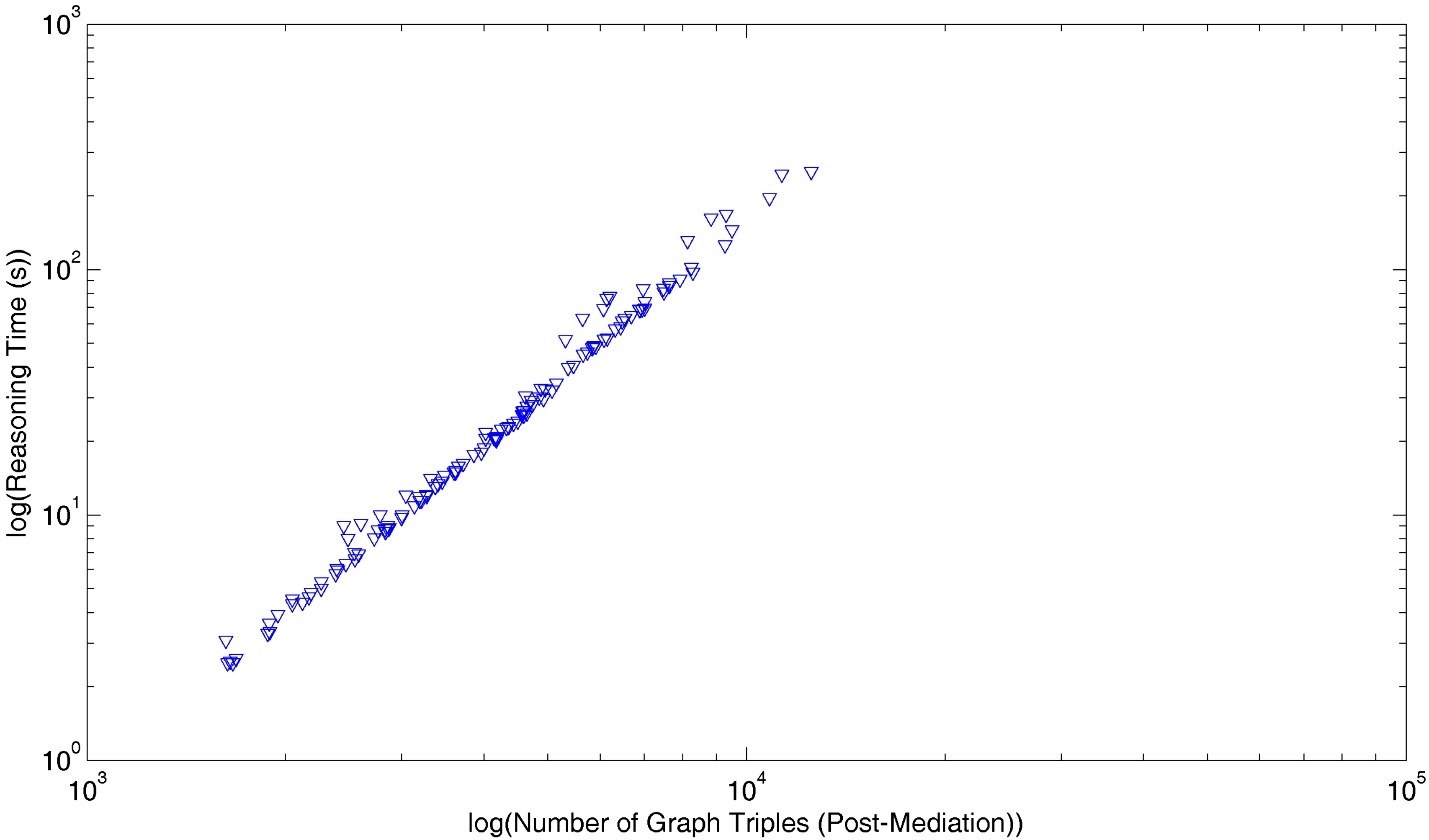
Efficiency: Performance of Algorithm

CPU Time (User + System Time) – Larger Network



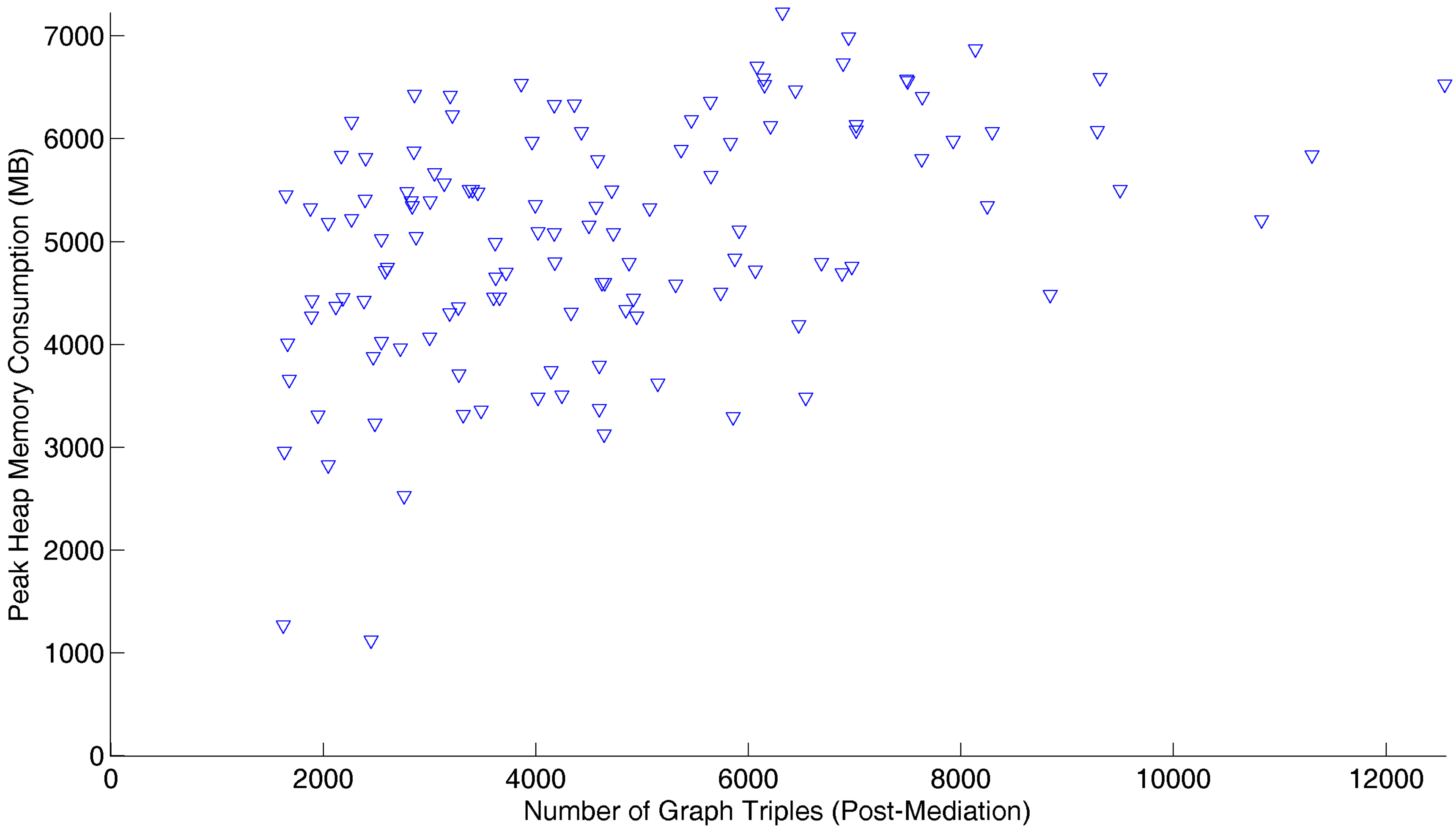
Efficiency: Performance of Algorithm

CPU Time (User + System Time) – Log-log Plot



Efficiency: Performance of Algorithm

Peak Heap Memory Usage

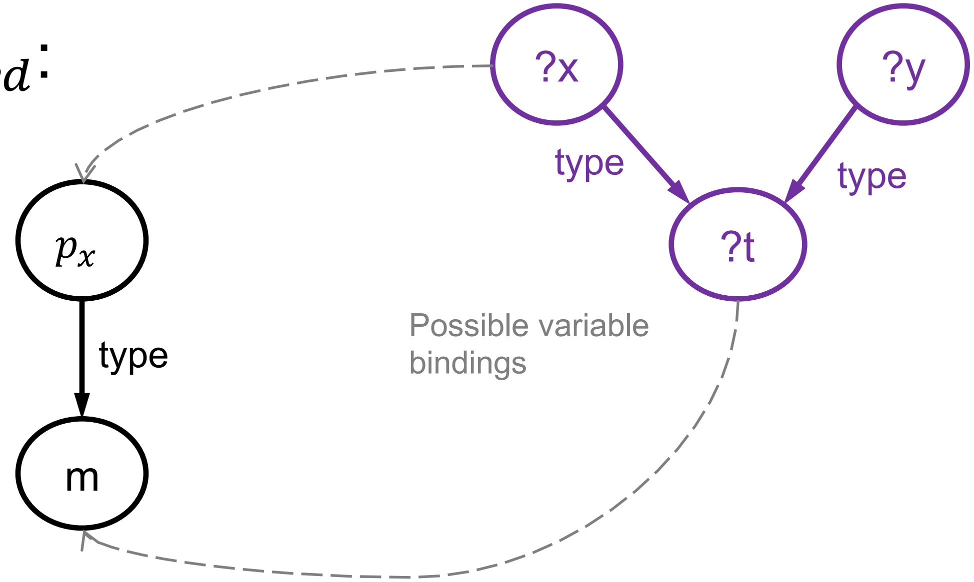


An Incremental Algorithm

1. Retrieve a list of graph triples added to graph (T_{added})
2. For each graph triple in the list, attempt to, at least *partially*, match one of the known patterns

➔ If no partial match to any of the patterns was found, go to next triple

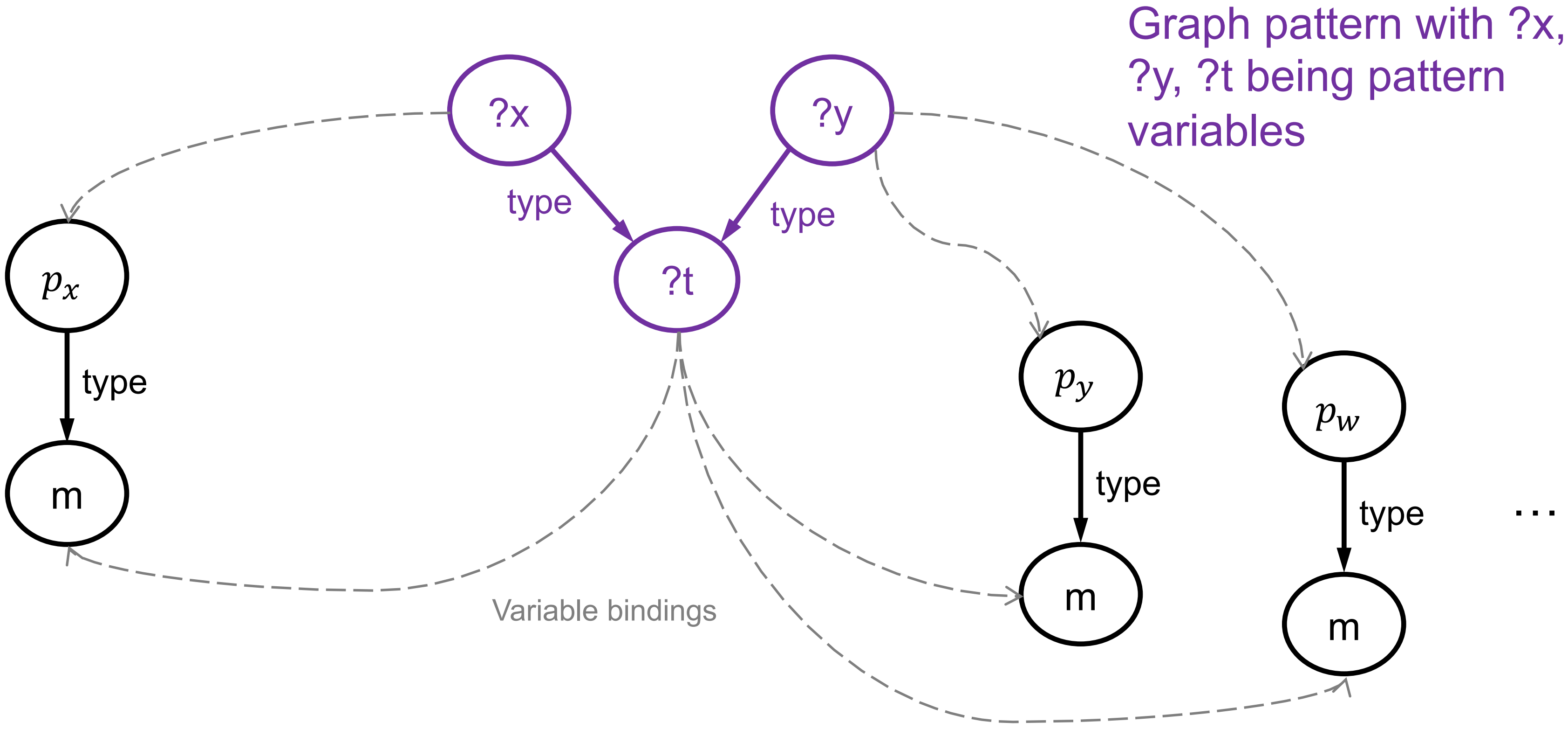
$\forall\, T_{added}$:



Graph pattern with ?x, ?y, ?t being pattern variables

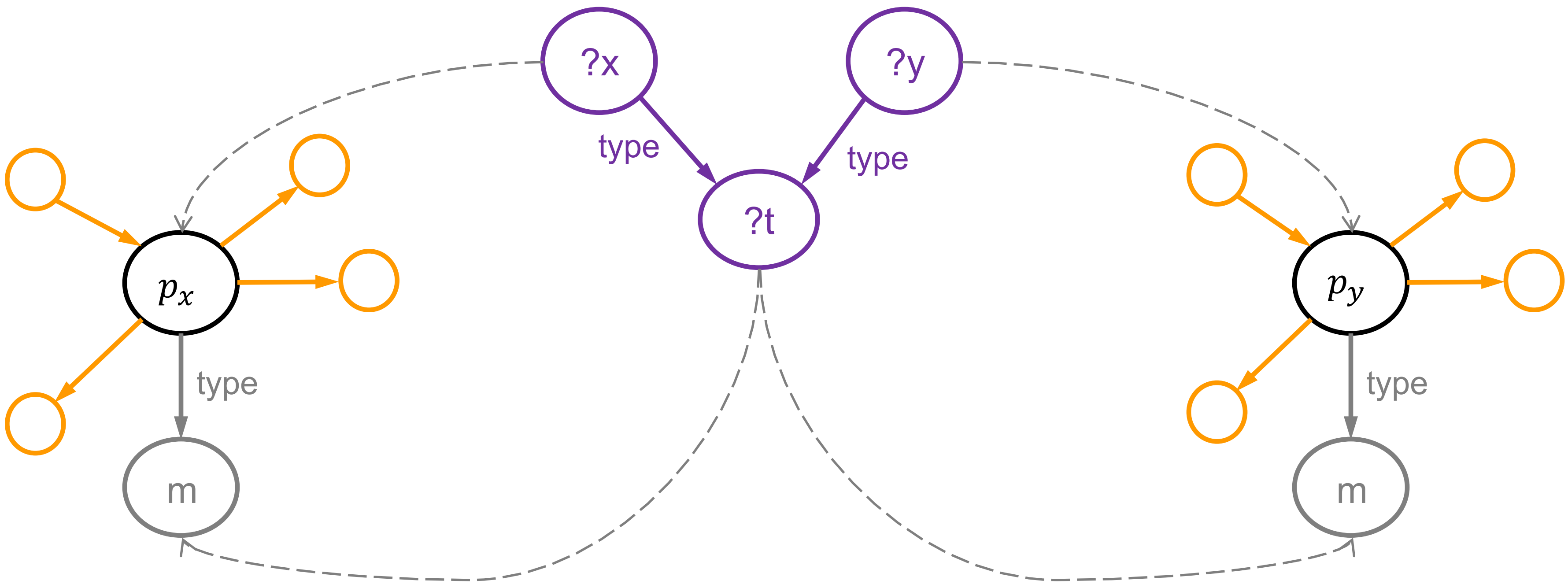
An Incremental Algorithm

3. If a partial match was found, attempt a *full* pattern match over the entire graph (old triples + added triples), *with the original triple as an anchor* ➔ potentially many results



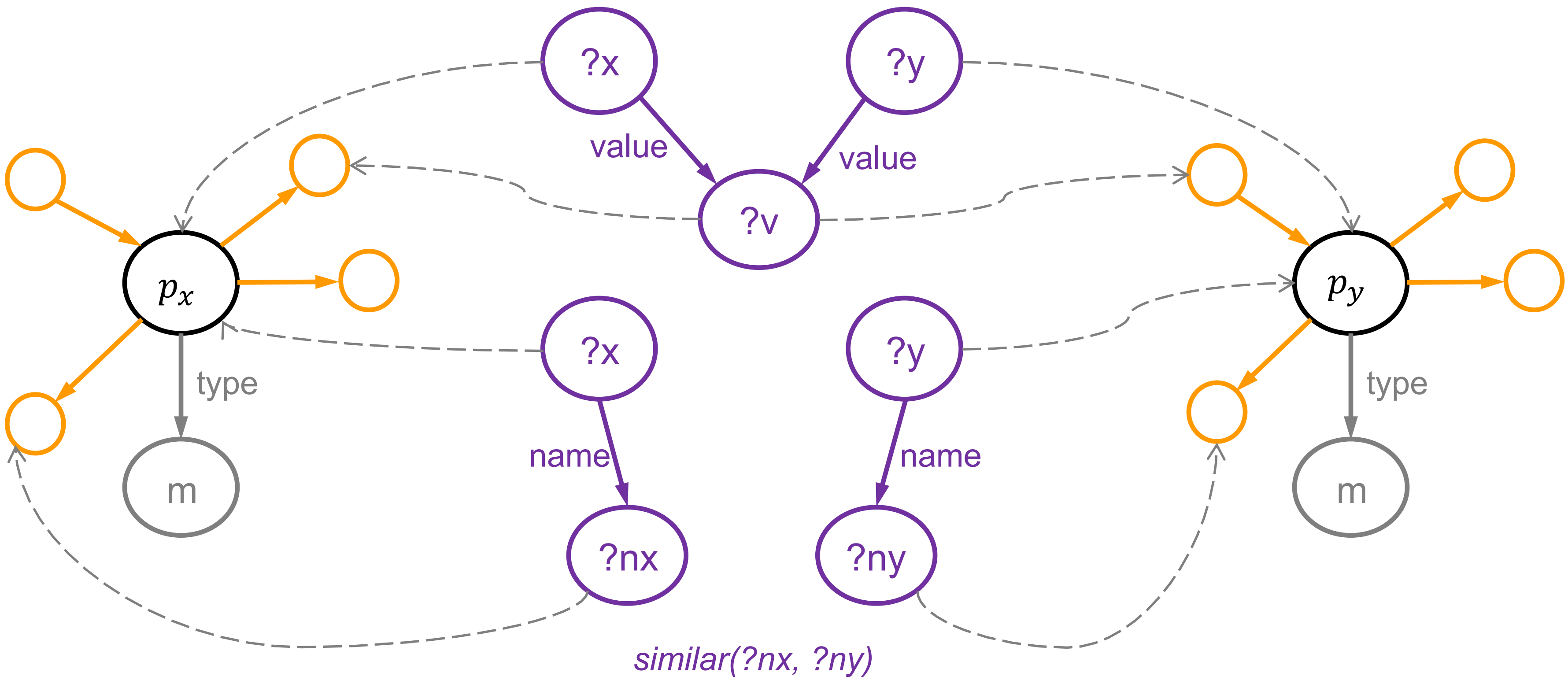
An Incremental Algorithm

4. For *all* matches, retrieve nearest neighbors (as graph triples) of those nodes that are bound to variables shared across all patterns (here: ?x and ?y)



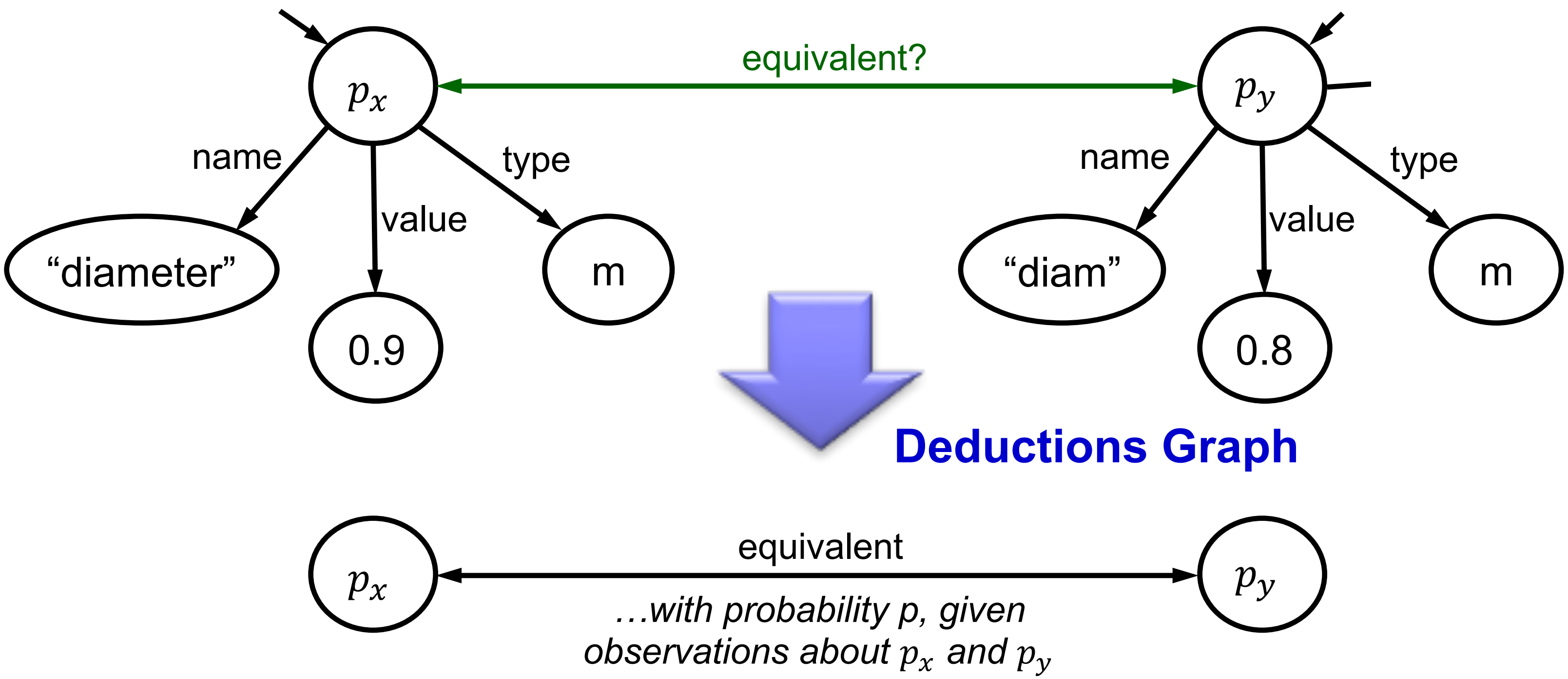
An Incremental Algorithm

5. For each of the triples in the list, use it as a hook to find matches *in the overall graph* to all *other* patterns ➔ store matches in a map with common variable bindings as index

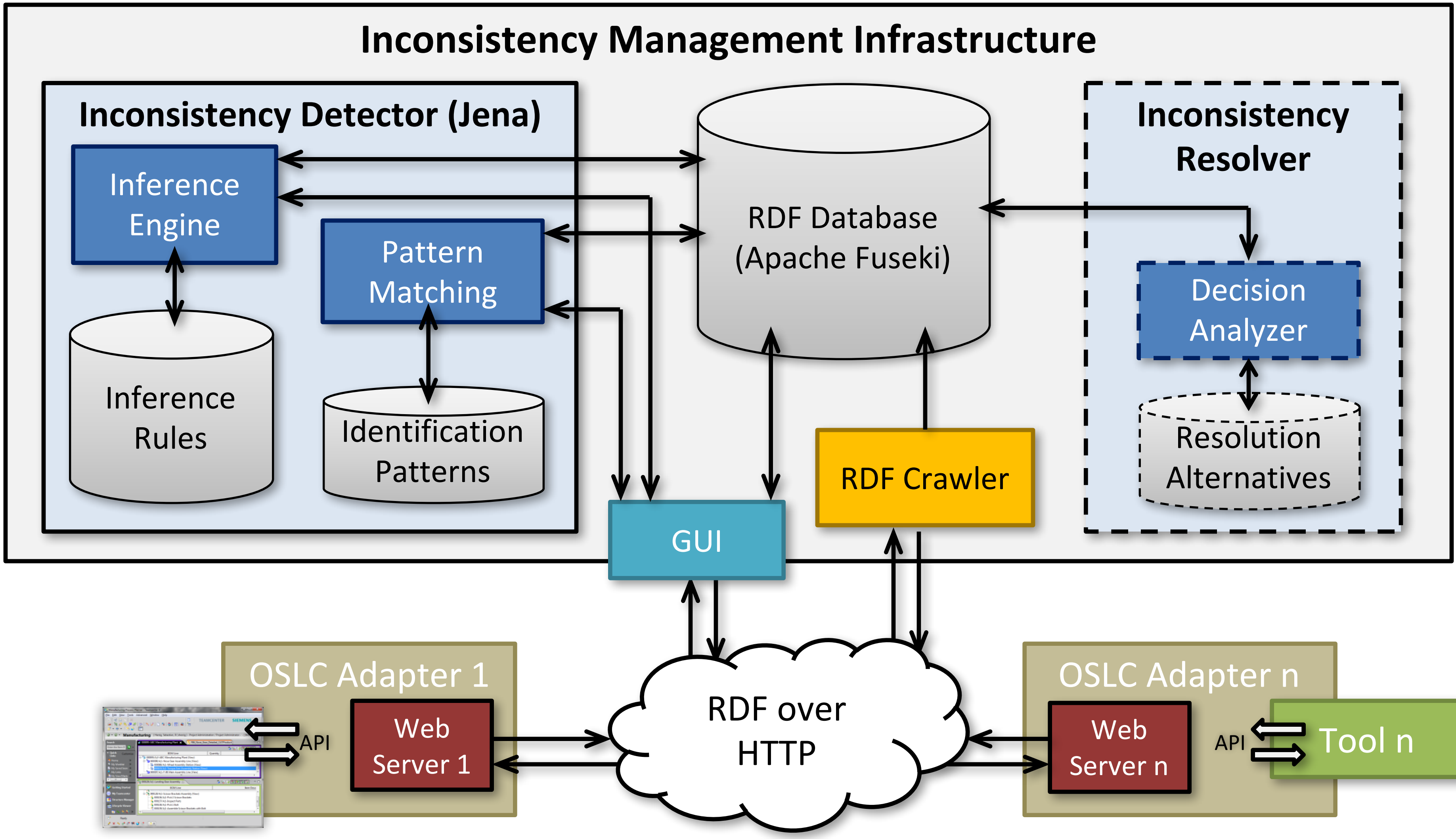


An Incremental Algorithm

6. Iterate over the items in the map and use Bayesian network inference to infer probability of events not observed
7. Continue with the next added triple by repeating from 2.

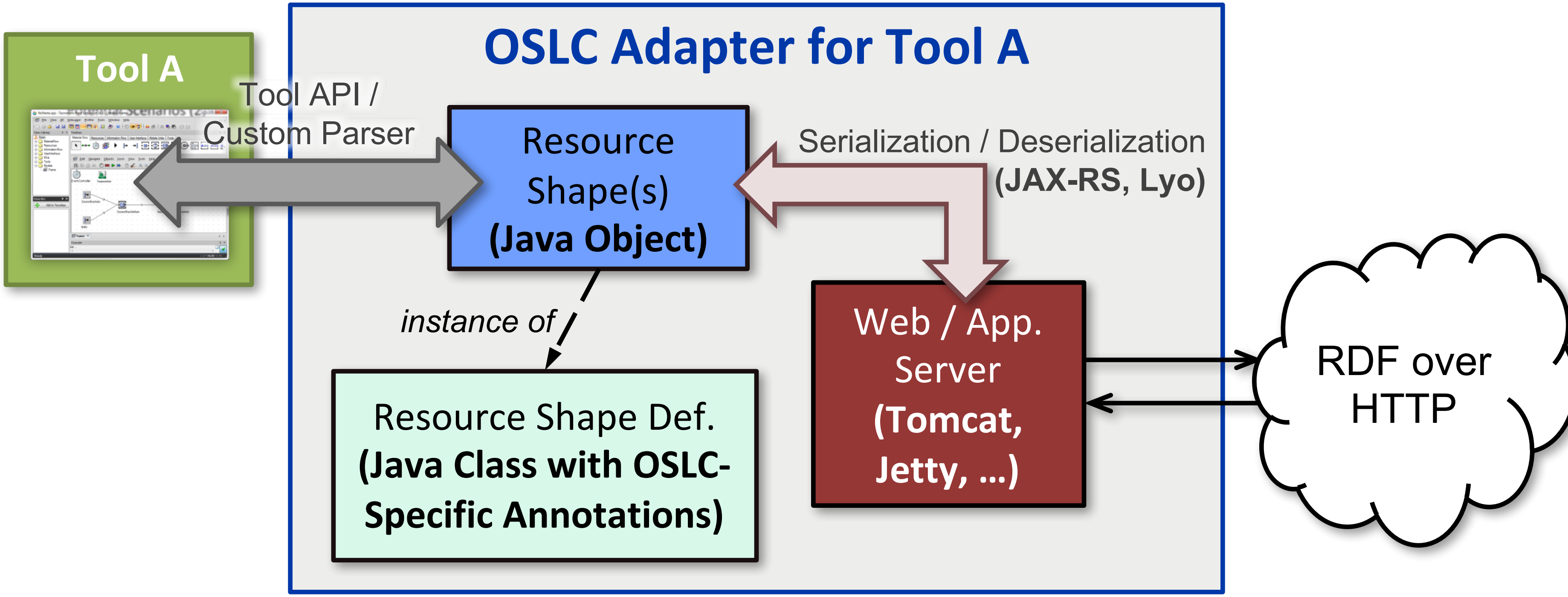


ConSystent



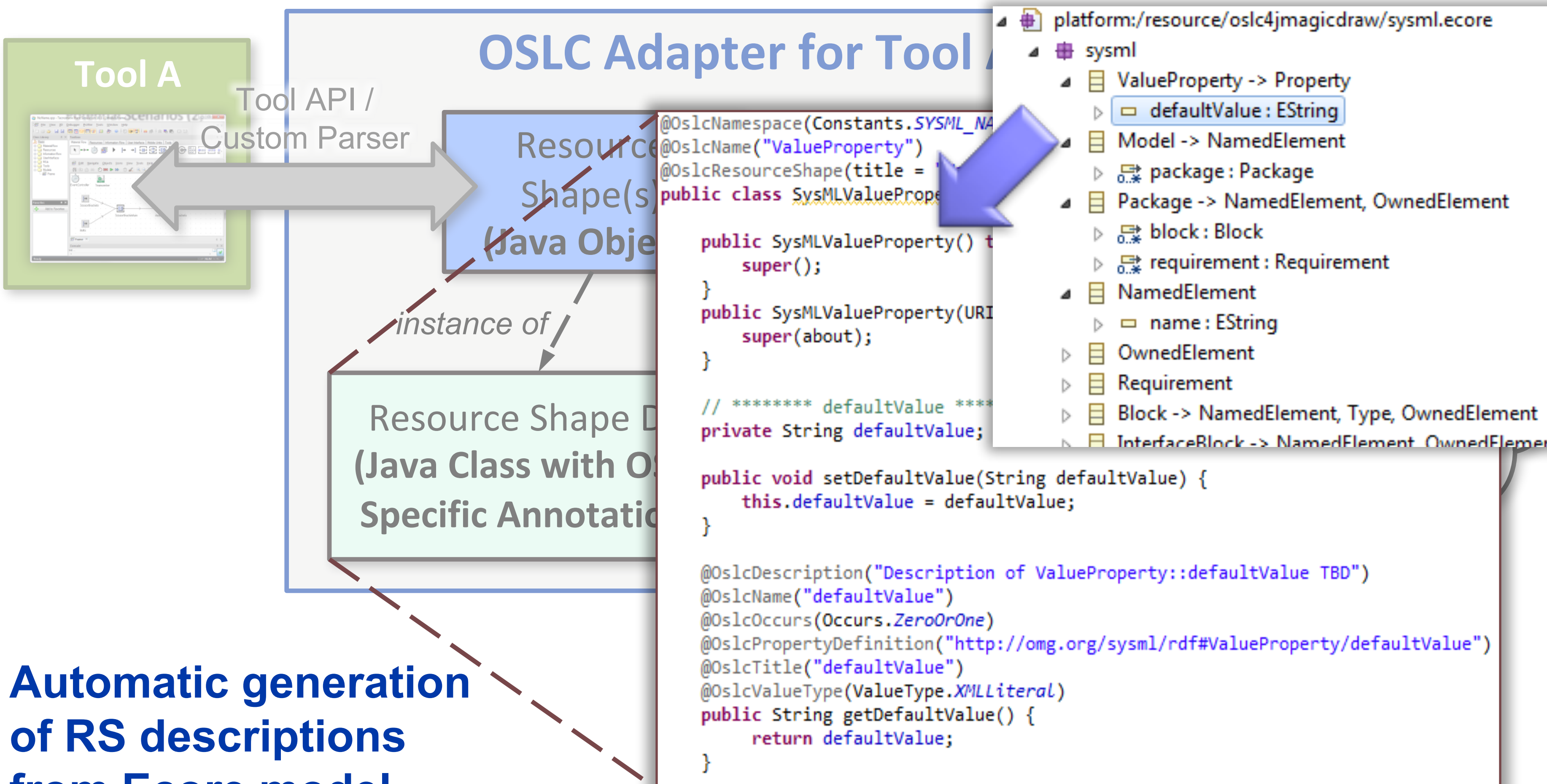
Inconsistency Management Infrastructure
Inconsistency Detector (Jena)
Inference Engine
Pattern Matching
Inference Rules
Identification Patterns
RDF Database (Apache Fuseki)
Inconsistency Resolver
Decision Analyzer
Resolution Alternatives
GUI
RDF Crawler
OSLC Adapter 1
Web Server 1
API
RDF over HTTP
OSLC Adapter n
Web Server n
API
Tool n

OSLC Tool Adapters in Java in a Nutshell



Work with Axel Reichwein, Koneksys LLC

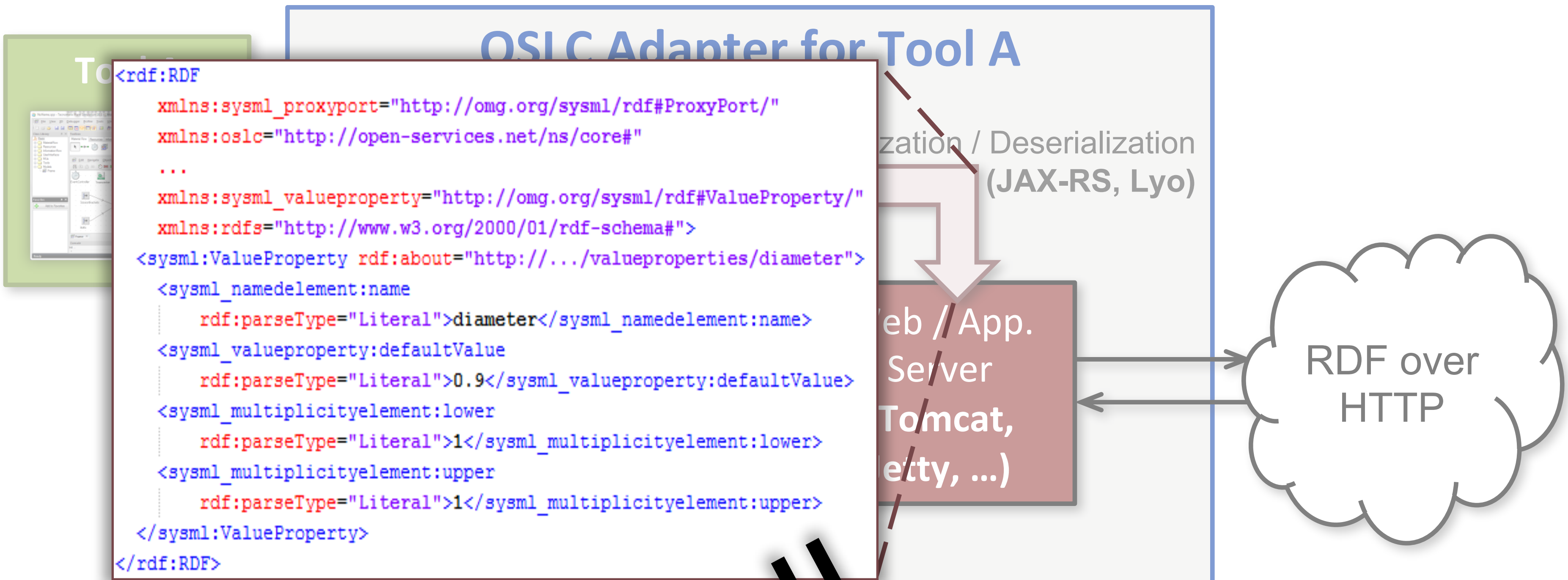
OSLC Tool Adapters in Java in a Nutshell

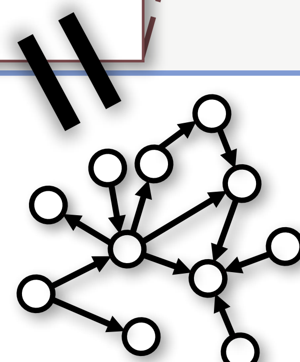


Automatic generation of RS descriptions from Ecore model

Work with Axel Reichwein, Koneksys LLC

OSLC Tool Adapters in Java in a Nutshell



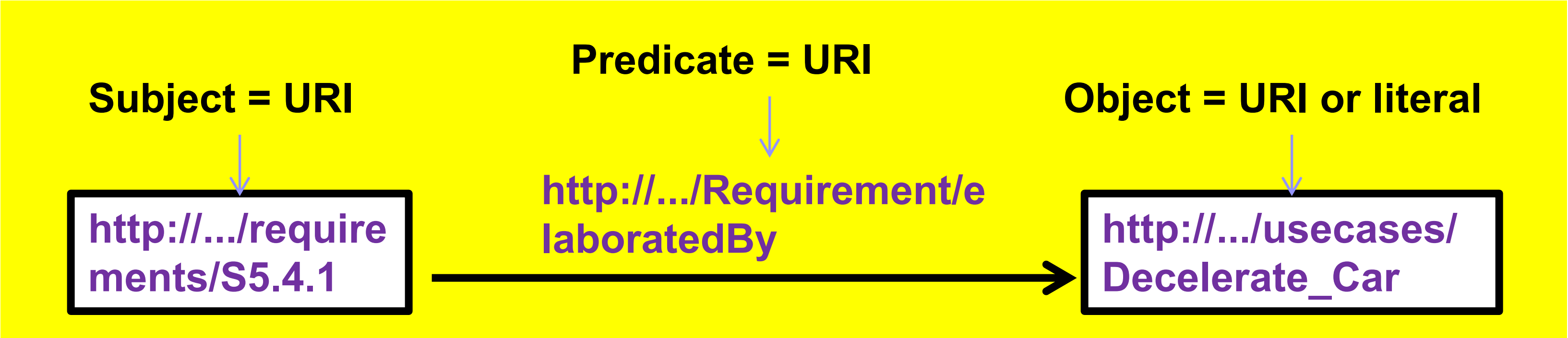


Serialization of resource shape in various formats (e.g., RDF/XML, TTL, HTML)

Work with Axel Reichwein, Koneksys LLC

Resource Description Framework (RDF)

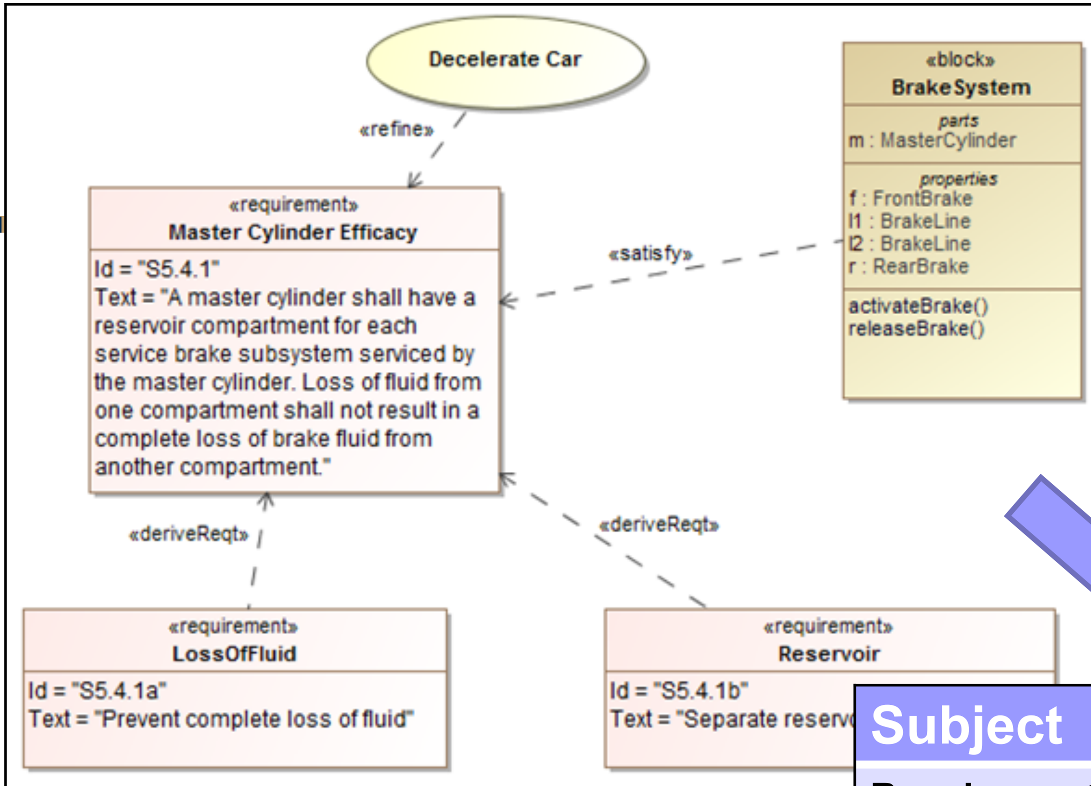
- Statements about resources in the form of **subject-predicate-object expressions (triples)**



- World Wide Web Consortium (W3C) standard for data interchange on the Web
- Used for Semantic Web applications (e.g. semantic reasoning)
- Variety of syntax notations and data serialization formats

(Axel Reichwein, Koneksys LLC)

RDF Example



Subject	Predicate	Object
Requirement „Master Cylinder Efficacy“	elaboratedBy	**Use Case** „Decelerate Car“
Requirement „Master Cylinder Efficacy“	satisfiedBy	**Block** „Brake System“
Requirement „Master Cylinder Efficacy“	derivedRqt	**Requirement** „Loss of Fluid“
Requirement „Master Cylinder Efficacy“	derivedRqt	**Requirement** „Reservoir“

RDF = set of simple **subject-predicate-object** statements (triples)

(Axel Reichwein, Koneksys LLC)

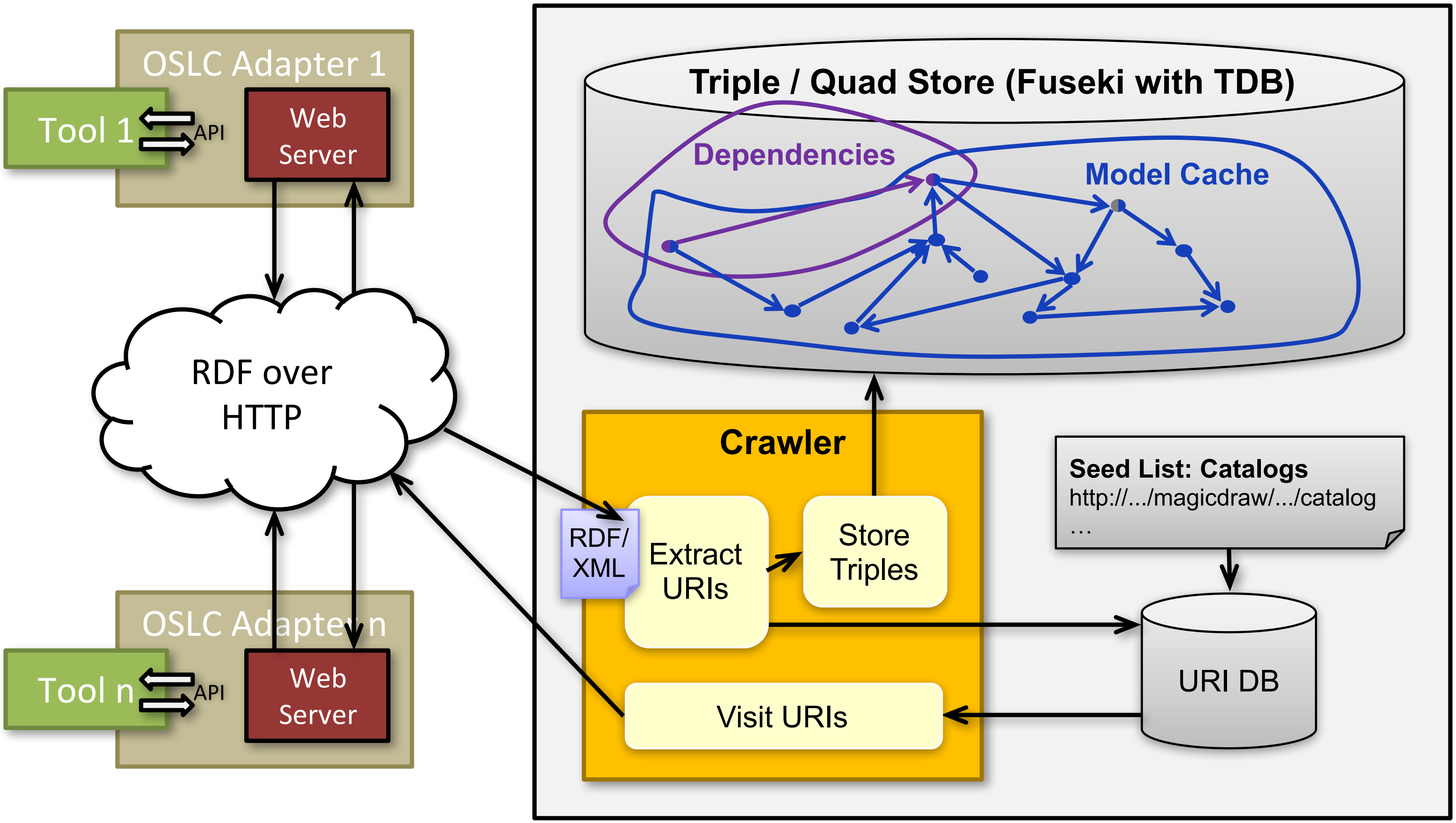
OSLC Vocabularies

Domain	Status
ALM/PLM Interoperabiltiy	Draft
Architecture Management	2.0
Asset Management	2.0
Automation	2.0
Change Management	2.0
Estimation and Measurement	Draft
Performance Monitoring	2.0
Quality Management	2.0
Reconciliation	2.0
Requirements Management	2.0
MBSE / SysML	Draft

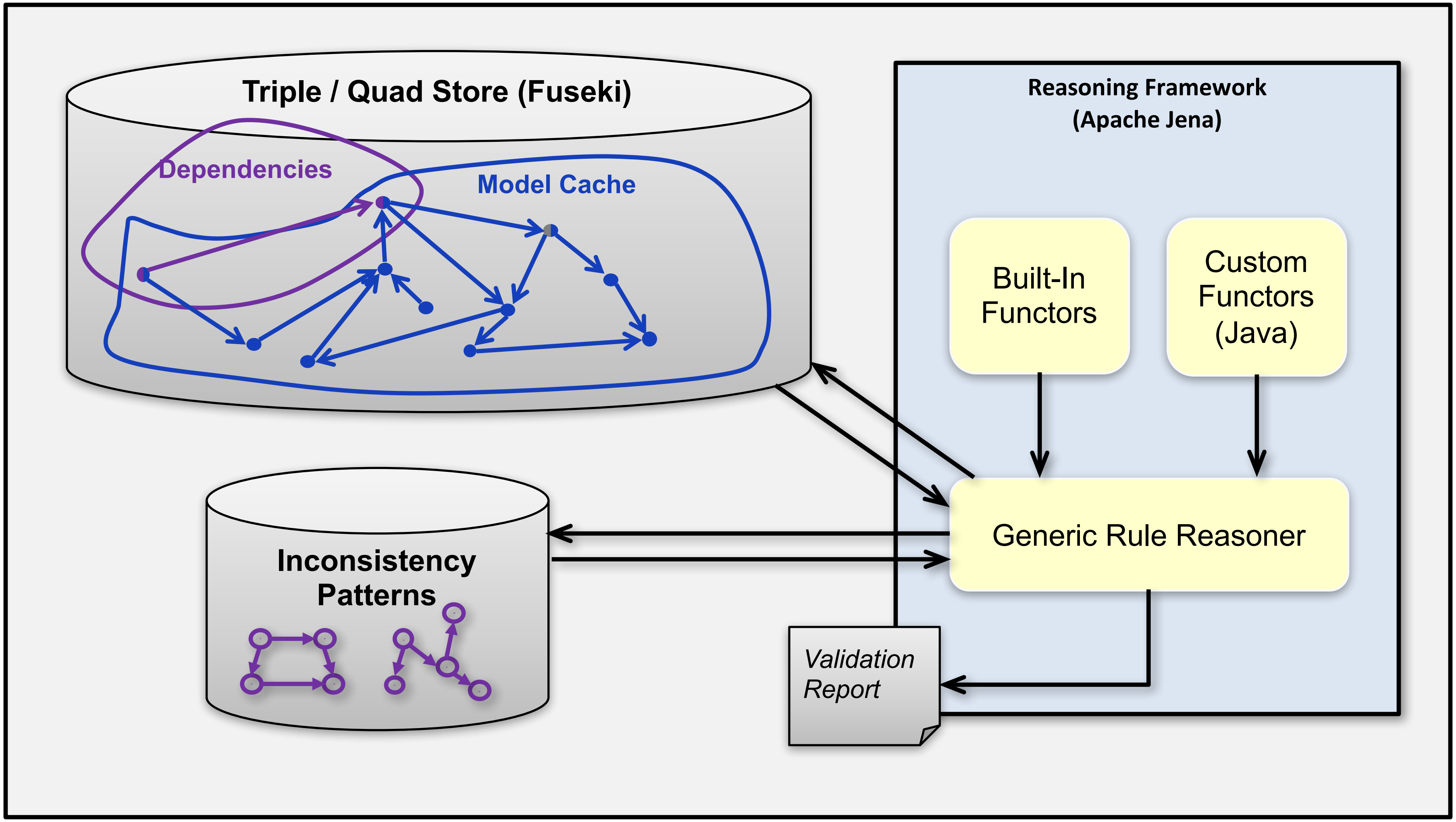
Standardized OSLC resource types for various domains

(Axel Reichwein, Koneksys LLC)

Crawler & Triple / Quad Store

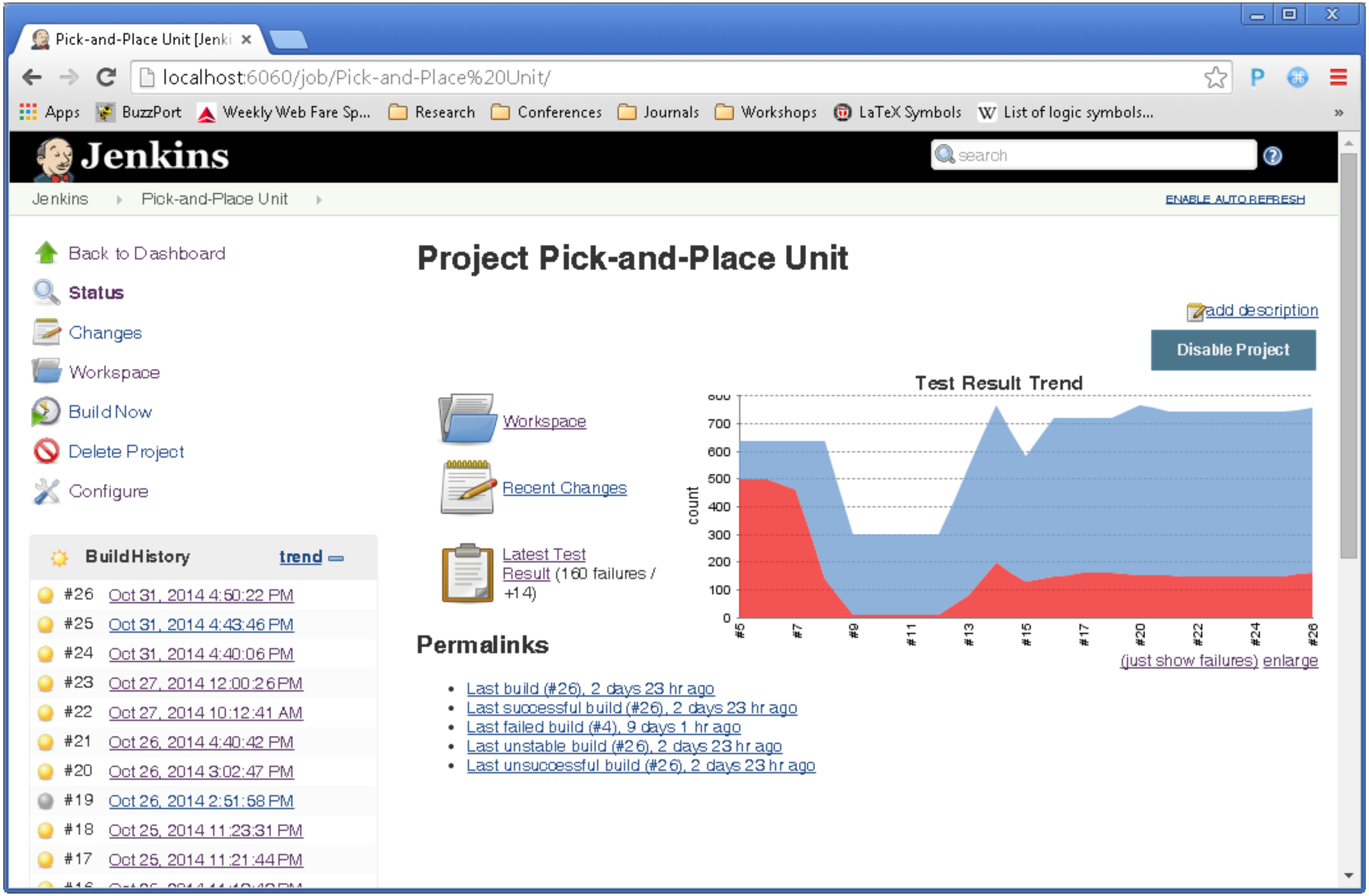


Inconsistency Reasoner



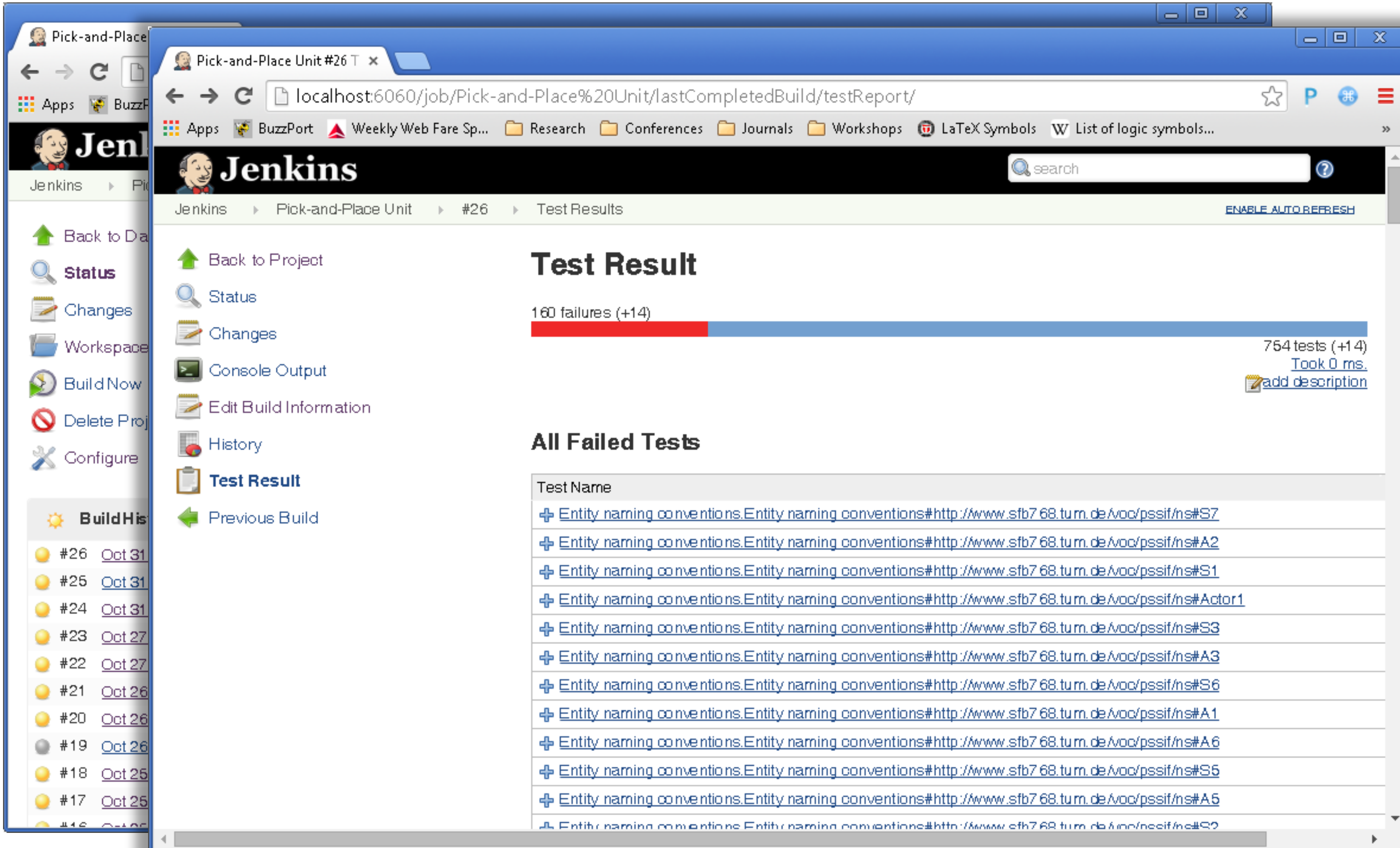
Presenting Identified Inconsistencies in Jenkins CI

A "Build" = Inconsistency Check = Sets of Test Cases

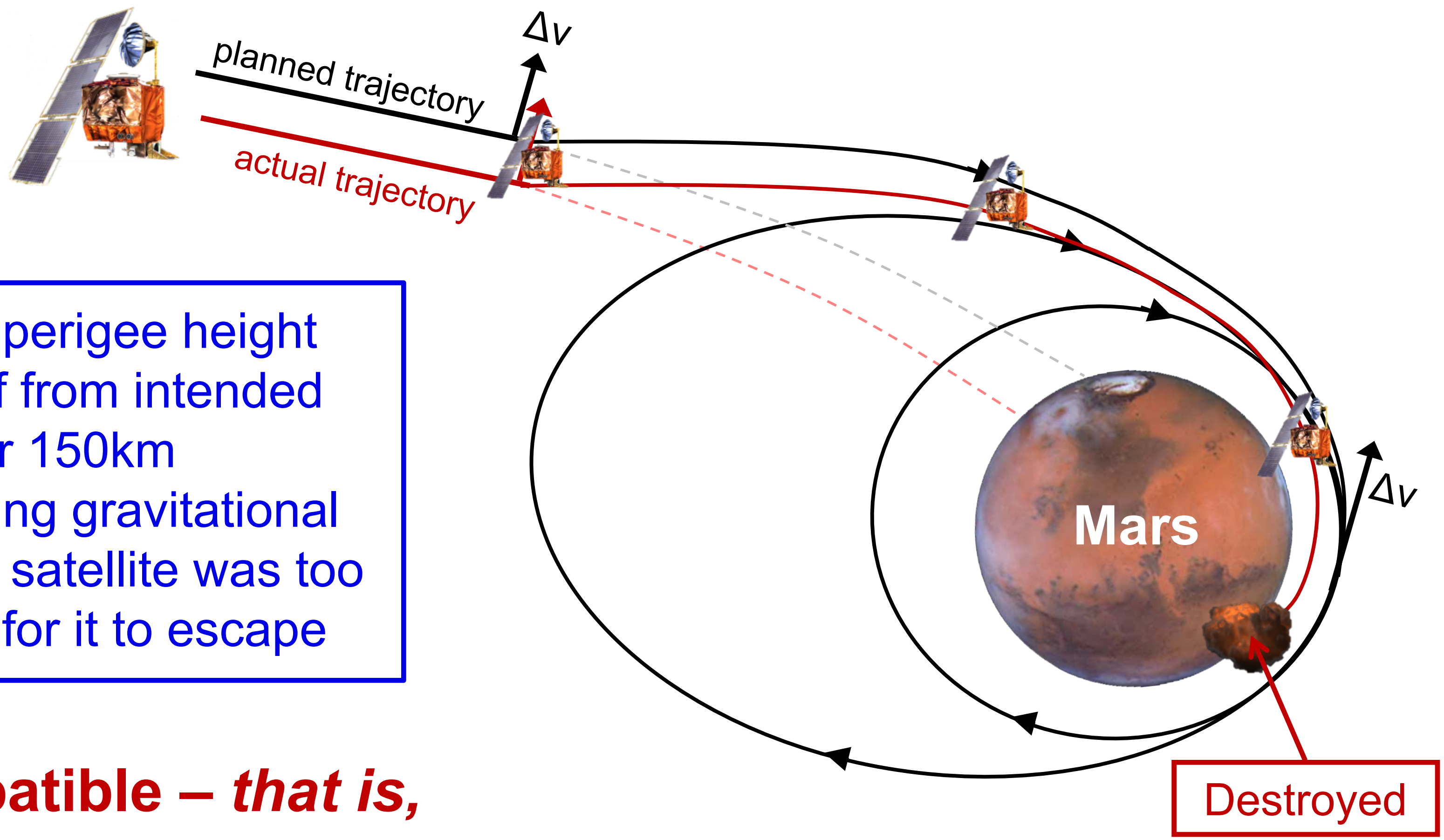


Presenting Identified Inconsistencies in Jenkins CI

A "Build" = Inconsistency Check = Sets of Test Cases



The Mars Climate Orbiter



- Actual perigee height was off from intended by over 150km
- Resulting gravitational pull on satellite was too strong for it to escape

Incompatible – *that is, inconsistent* – units led to mission failure!

(Source: NASA / JPL)

Task 1

Characterize & Classify Inconsistencies and Semantic Relationships

- Characterize and classify kinds and types of inconsistencies in sets of engineering models and semantic relationships that these are strongly connected to → answers RQ 1

- Re-consider existing taxonomies and ontologies in literature
  - E.g., Mens *et al*., 2005: structural & behavioral inconsistencies vs. horizontal & vertical abstraction; Schaetz *et al.* 2003: syntactic & semantic vs. variant & invariant inconsistencies
  - Semantic relationships (language literature): synonymy (“same as”), hyponymy (“is a”), meronymy (“part-whole”), …

- Refine by relating to design & development of physical systems: e.g., consider relationship between specification, prediction and realization

Task 2
Bayesian Framework for Inconsistency Management

- Task 2.1: Extend current framework
  - Investigate how stochastic inference influences the detection and resolution of inconsistencies
  - Investigate role of humans → e.g., elicitation of beliefs, feedback
  - Integrate identified activities into current framework
- Task 2.2 + 2.3: Formulate conceptual approach and develop method for performing Bayesian inference in graphs
  - Develop method for observing evidence in data
  - Investigate what should be captured as result of inference (all possible events? MAP? Associated probabilities?)
  - Conceptualize a formal language for capturing Bayesian belief networks
- Task 2.4: Develop learning algorithms
  - Investigate what statistics should be computed to inform modeler
  - Given new observations, how can a belief be updated?

Task 3

Identify Measures for Evaluating Effectiveness of Approach

- Guiding question: *what constitutes an effective inconsistency management approach?*

  → Accuracy, completeness, correctness, degree of automation, ...

- Completeness and correctness
  - Completeness: fraction of inconsistencies correctly identified and actual number of inconsistencies, expressiveness of inconsistency detection rules, …
  - Accuracy & correctness: fraction of false positives-identified, …

- Comparison to status quo: how does the proposed approach compare to manually managing inconsistencies or using logical inference?

- Scalability: does the proposed approach scale? Training data size?

Task 4

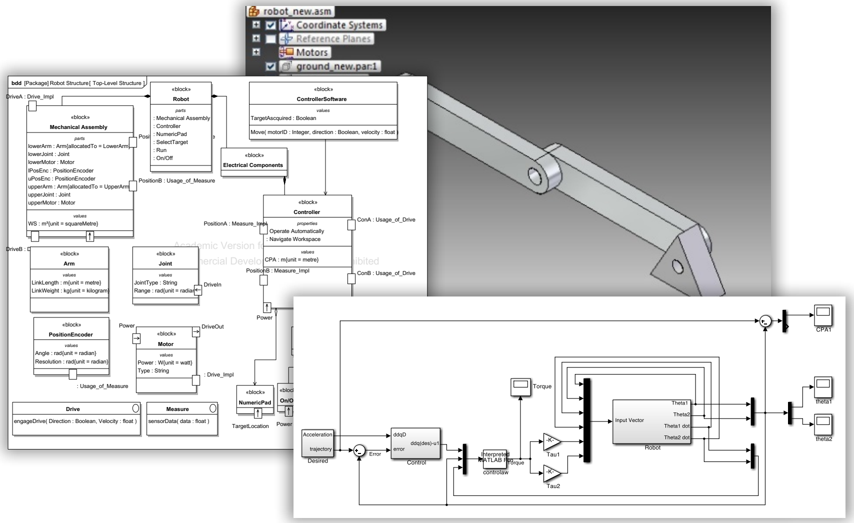
Extend Proof-of-Concept Implementation

- Task 4.1: Develop concrete implementations of languages conceptualized in task 2
  - Capture Bayesian belief networks
  - Capture events and probabilities (e.g., posterior beliefs) in RDF
- Extend POC implementation by adding software components supporting the Bayesian framework developed in task 2
  - Task 4.2: Refine and extend current architecture
  - Task 4.3: Implement components, e.g., Bayesian inference engine and graphical user interface for defining belief networks and eliciting beliefs
- Before re-inventing the wheel, analyze and evaluate suitability of existing Bayesian reasoning and machine learning frameworks → e.g., Weka, JavaBayes, Banjo and Apache Mahout (Task 4.0)

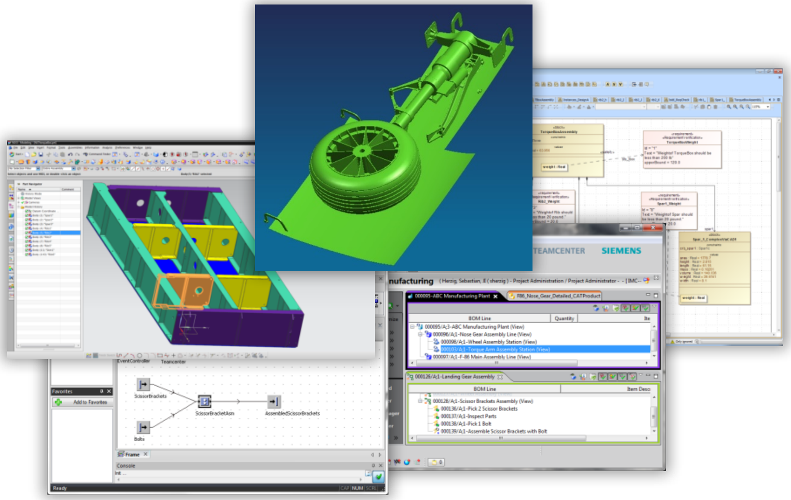
Task 5

Evaluate Effectiveness of & Validate Approach

- Aim: develop case studies to evaluate effectiveness of approach



Two-Degree of Freedom Robot (Qamar)



Torque Box & Landing Gear (Boeing)

- For each case study, the following is to be developed:
  - Concrete inconsistency scenario
  - Accompanying models
  - Inconsistency management rules

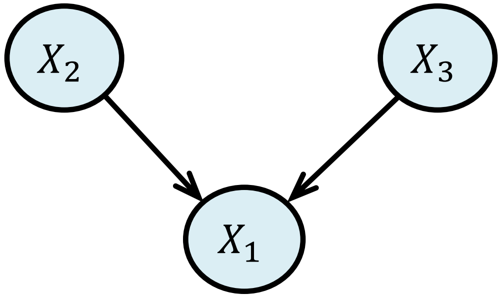
⟹ Measure properties identified in task 3 to evaluate effectiveness of approach

Terminology

- Now assume a Bayesian network with n nodes / variables X_i
  - Assume each variable X_i can have different number of states / values q_i
  - Also assume that each may have a different number of *configurations* of parents $\mathrm{pa}(X_i)$
- Goal: estimate / update belief about parameters θ_{ijk}:

$$\theta_{ijk} = P(X_i = j \mid \mathrm{pa}(X_i) = k) \qquad \forall i,j,k\colon \theta_{i,j,k} \geq 0 \qquad \sum_j \theta_{i,j,k} = 1$$

- Example:



(Assume binary: $X_i = 1$ or $X_i = 2$)

$$\theta_{111} = P(X_1 = 1 \mid \mathrm{pa}(X_1) = 1)$$
$$\theta_{112} = P(X_1 = 1 \mid \mathrm{pa}(X_1) = 2)$$

$$\mathrm{pa}(X_1) = 1\colon P(X_1 = \cdots \mid X_2 = 1, X_3 = 1)$$
$$\mathrm{pa}(X_1) = 2\colon P(X_1 = \cdots \mid X_2 = 1, X_3 = 2)$$
$$\mathrm{pa}(X_1) = 3\colon P(X_1 = \cdots \mid X_2 = 2, X_3 = 1)$$

… *(configuration = one part. permutation)*

Closed Form

- Using elementary properties of Bayesian networks:

$$P(X_1 = x_1, X_2 = x_2, \ldots \mid \mathbf{D}) = \prod_i P(X_i = x_i \mid \mathrm{pa}(X_i), \mathbf{D})$$

- Finally, it can be shown that:

$$\boxed{P(X_i = j \mid \mathrm{pa}(X_i) = k, \mathbf{D}) = \frac{m_{ijk} + \alpha_{ijk}}{\sum_j (m_{ijk} + \alpha_{ijk})}}$$

Training Data Cases

- Assume a set of data

X_1	X_2	X_3
1	1	2
2	1	1
…	…	…

- Let each data case D_i be a vector of values – e.g.,:

$$D_2 = (X_1 = 2, X_2 = 1, X_3 = 1)$$

- Data set $\mathbf{D} = \{D_1, D_2, \ldots, D_m\}$ of m data cases

Maximum Likelihood Estimate

- Similar to case with one variable:

$$L(\theta \mid \mathbf{D}) = P(\mathbf{D} \mid \theta) = \prod_{l=1}^{m} P(D_l \mid \theta)$$

  – Easier to work with log-likelihood:

$$\log L(\theta \mid \mathbf{D}) = \sum_{l} \log P(D_l \mid \theta)$$
$$= \sum_{i,j,k} m_{ijk} \log \theta_{ijk}$$

Where m_{ijk} := number of data cases where $X_i = j$ and $\mathrm{pa}(X_i) = k$

- It can be shown that likelihood is maximized for:

$$\theta_{ijk}{}^{*} = \frac{m_{ijk}}{\sum_j m_{ijk}}$$

Therefore, reasonable estimates require a lot of training data cases!

Accounting for Prior Belief

- Bayesian estimation: view θ as a vector of random variables θ_{ijk}

$$p(\theta \mid \mathbf{D}) \propto p(\theta) \cdot L(\theta \mid \mathbf{D}) = p(\theta) \prod_{i,j,k} {\theta_{ijk}}^{m_{ijk}}$$

- Some important assumptions about prior:
  - Global independence w.r.t. variables in Bayesian network:

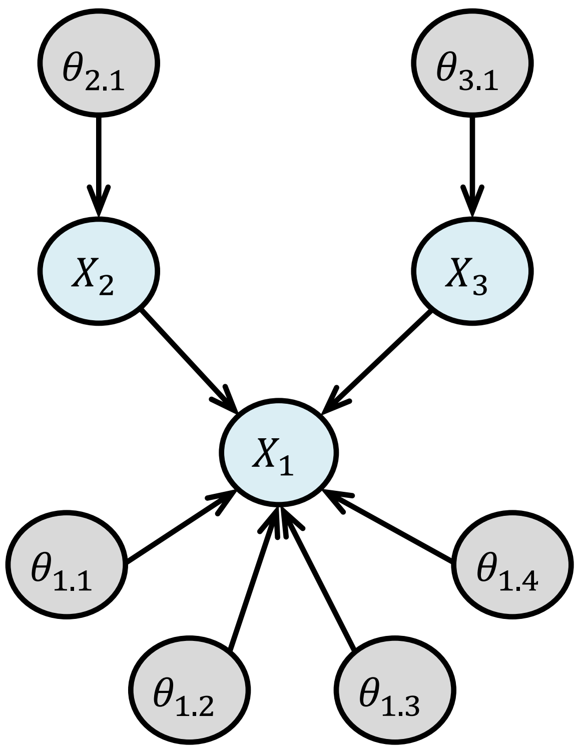
$$p(\theta) = \prod_{i} p(\theta_{i..})$$

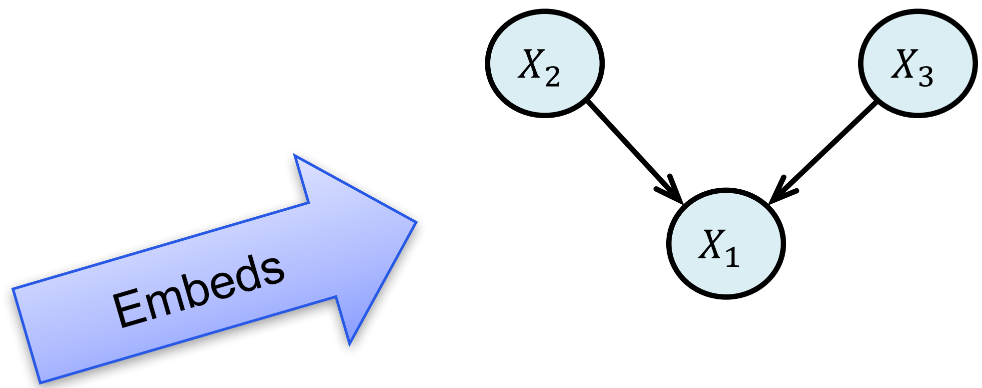
  - Local independence w.r.t "parent configuration" / values of parents:

$$p(\theta_{i..}) = \prod_{k} p(\theta_{i.k})$$

A Quick Look at Augmented Bayesian Networks

- Sometimes these assumptions are made explicit using augmented Bayesian networks, which embed the BN of interest





- Note that, together, global and local independence imply independence among parameters:

$$p(\theta) = \prod_{i,k} p(\theta_{i.k})$$

Choosing a Prior

- Since we are assuming discrete random variables, it is practical to assume Dirichlet priors:

$$p(\theta_{i.k}) \propto \prod_{j} \theta_{ijk}{}^{\alpha_{ijk}-1}$$

- We then get (using the independence of parameters assumption):

$$p(\theta) \propto \prod_{i,k} \prod_{j} \theta_{ijk}{}^{\alpha_{ijk}-1}$$

- This result can be inserted into the posterior to give:

$$p(\theta \mid \mathbf{D}) \propto p(\theta) \cdot L(\theta \mid \mathbf{D}) = p(\theta) \prod_{i,j,k} \theta_{ijk}{}^{m_{ijk}} = \prod_{i,k} \prod_{j} \theta_{ijk}{}^{m_{ijk}+\alpha_{ijk}-1}$$

Closed Form

- Using elementary properties of Bayesian networks:

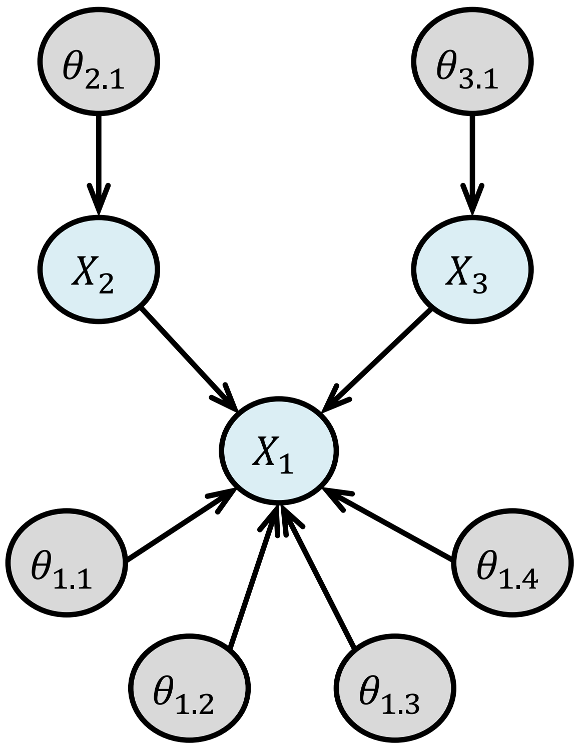
$$P(X_1 = x_1, X_2 = x_2, \ldots \mid \mathbf{D}) = \prod_i P(X_i = x_i \mid \mathrm{pa}(X_i), \mathbf{D})$$

- Finally, it can be shown that:

$$P(X_i = j \mid \mathrm{pa}(X_i) = k, \mathbf{D}) = \frac{m_{ijk} + \alpha_{ijk}}{\sum_j (m_{ijk} + \alpha_{ijk})}$$

A Note About Specifying Priors

- We need to be careful about the use of mixed sample sizes when specifying priors
- Say we use Beta $B(\alpha_{1,x.z}, \alpha_{2,x.z})$ distributions for all $\theta_{x.z}$:



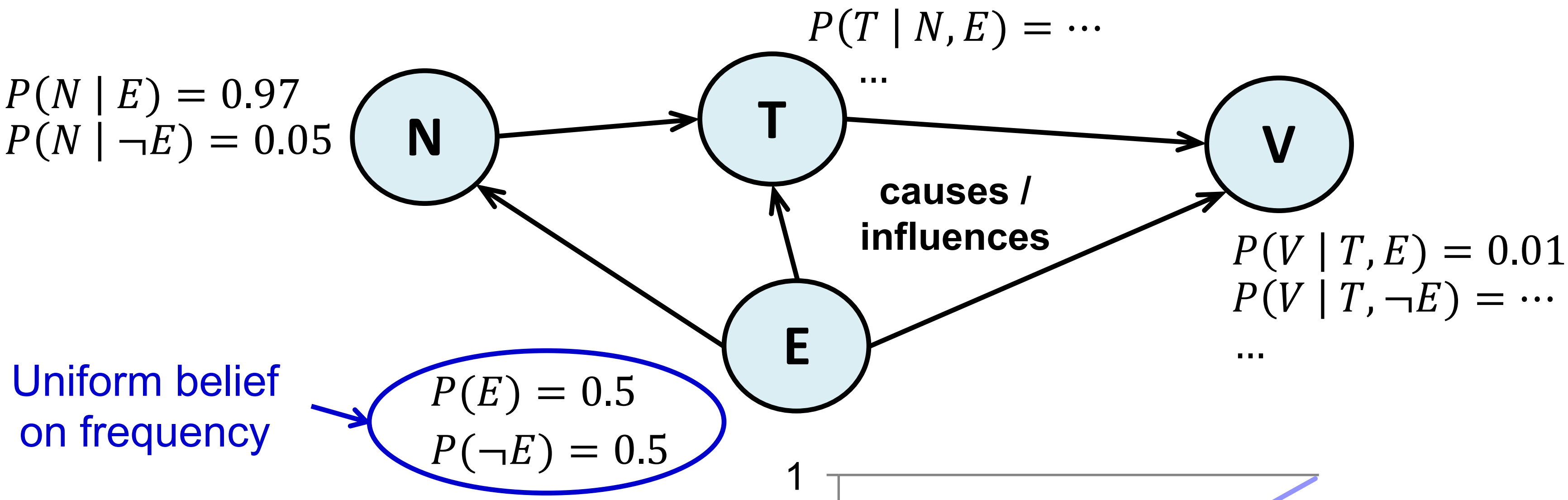
- Say the following priors were elicited:

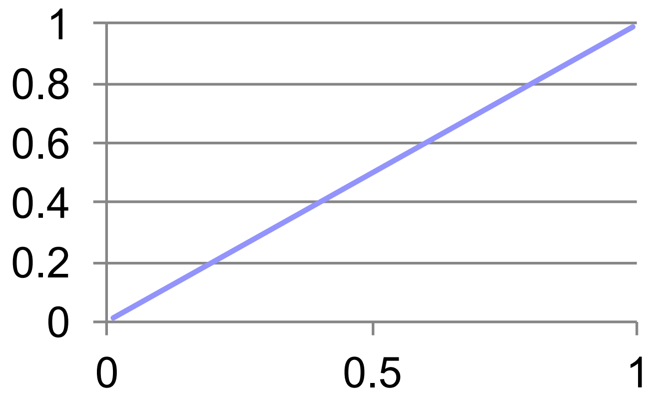
		Eq. Sample Sizes	
$\boldsymbol{X_2}$:	$\theta_{2.1}$: $B(1,2)$	3	
$\boldsymbol{X_3}$:	$\theta_{3.1}$: $B(4,6)$	10	
$\boldsymbol{X_1}$:	$\theta_{1.1}$: $B(7,10)$	17	Total: 26
	$\theta_{1.2}$: $B(1,1)$	2	
	$\theta_{1.3}$: $B(3,1)$	4	
	$\theta_{1.4}$: $B(1,2)$	3	

- Assumes different number of samples for each variable ➔ inconsistent number of observations

Updating Beliefs with Data Cases

- Beliefs are "guesses" by a human, based on expert knowledge
- Say a human has only a very vague belief (~ uninformed)





CDF of Beta(1,1)

Data Cases

- Assume a set of data

N	T	E	V
similar	*incompatible*	*equivalent*	*equal*
dissimilar	*compatible*	*different*	*not_equal*
…	…	…	

- Let each data case D_i be a vector of values – e.g.,:

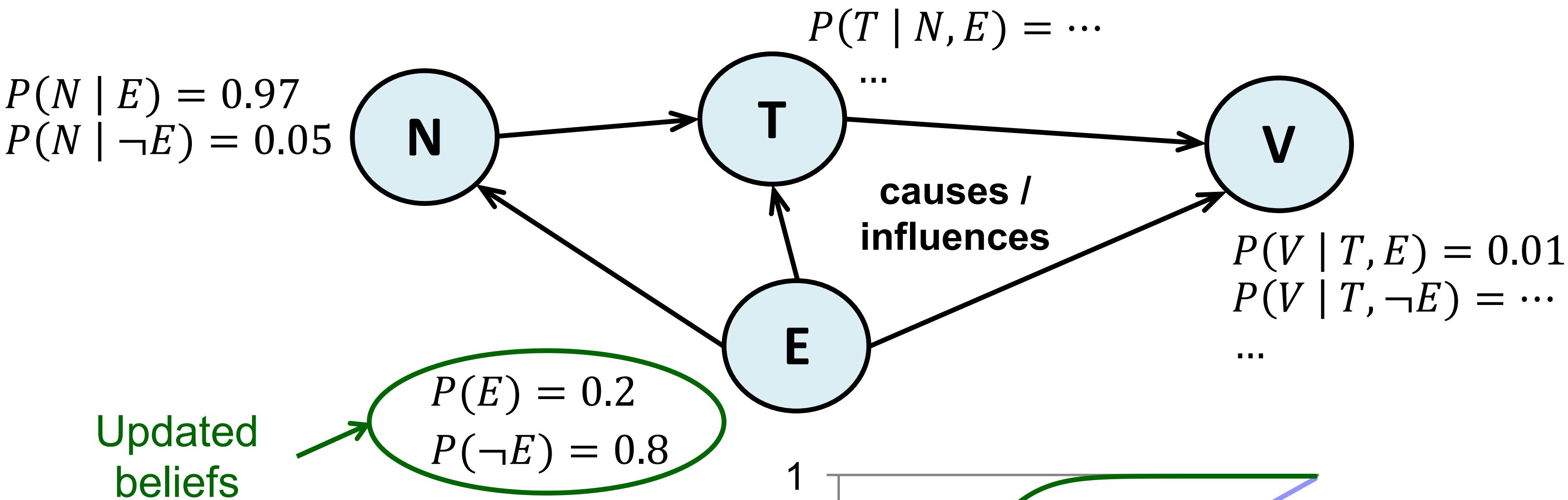
$$D_2 = (\mathbf{N} = \text{dissimilar}, \mathbf{T} = \text{compatible}, \mathbf{E} = \text{different}, \mathbf{V} = \text{not}_{\downarrow}\text{equal})$$

- Data set $\mathbf{D} = \{D_1, D_2, \ldots, D_m\}$ of m data cases, collected up-front and / or incrementally

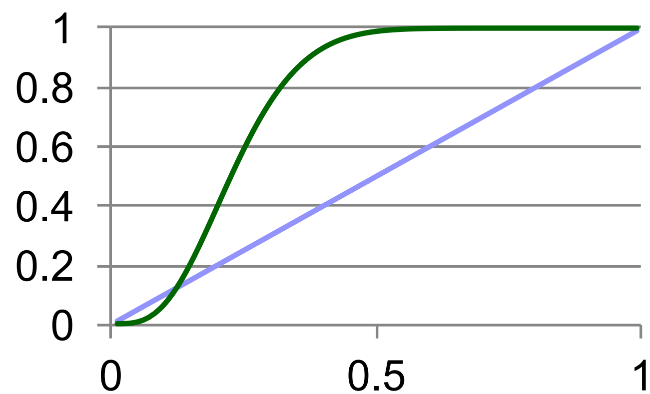
$$P(X_i = j \mid \text{parents}(X_i) = k, \mathbf{D}) = \frac{m_{ijk} + \alpha_{ijk}}{\sum_j (m_{ijk} + \alpha_{ijk})}$$

Updating Beliefs with Data Cases

- Beliefs are “guesses” by a human, based on expert knowledge
- Say a human has only a very vague belief (~ uninformed)



Similar to Bayesian spam filtering systems: human identifies false positives and false negatives, system updates beliefs



Updated CDF after considering data cases

Outline

- Introduction
  - Motivation & Context
  - Open Issues in State of the Art & Research Objectives
- Foundations & Methods
  - Characteristics of Inconsistencies
  - A Bayesian Reasoning Approach to Identifying Inconsistencies
- Evaluation & Case Study
  - Characterization of Approach
  - Comparison to Deterministic Approach
- Contributions
- Limitations & Future Work