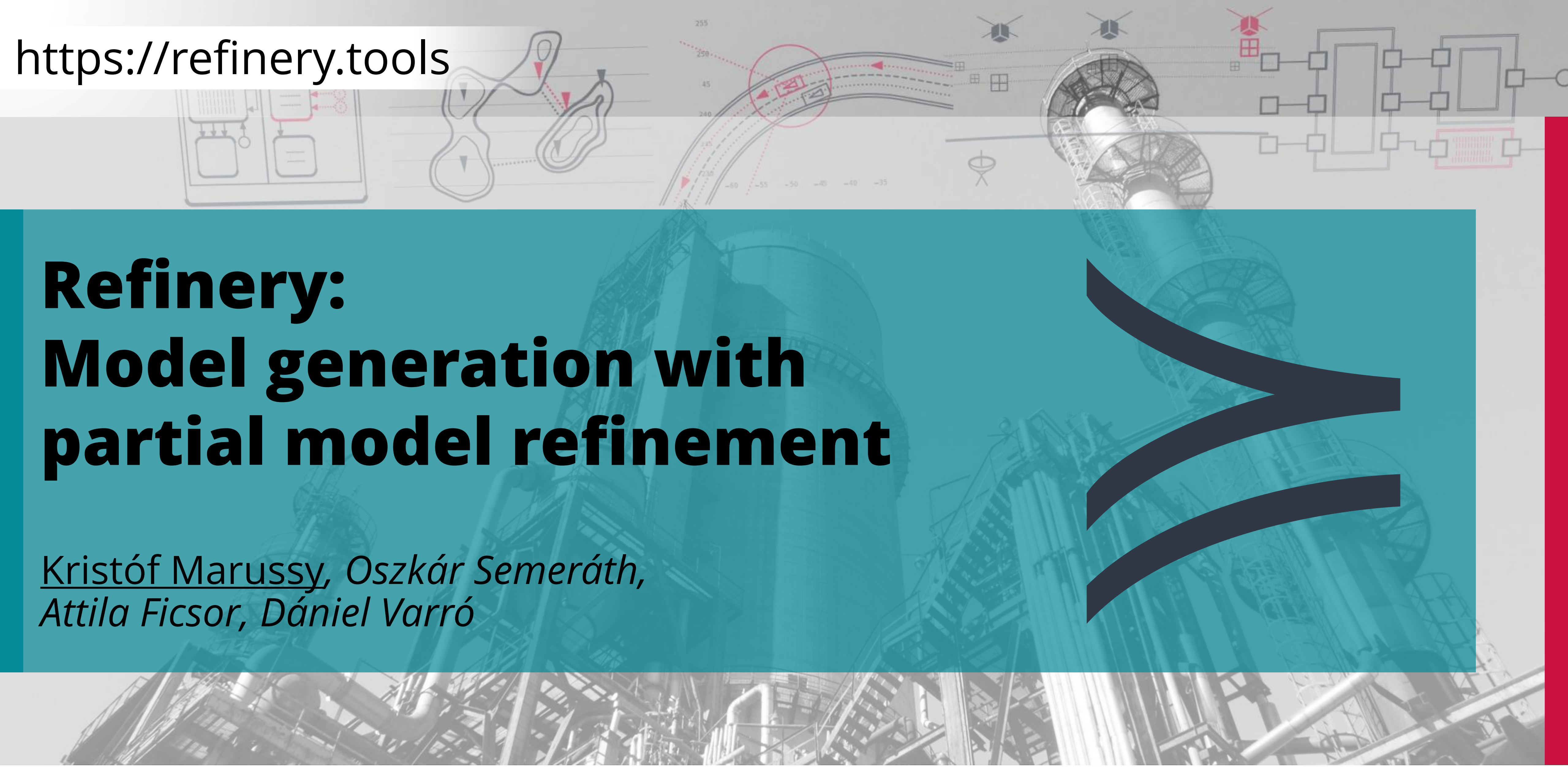
https://refinery.tools



Refinery: Model generation with partial model refinement

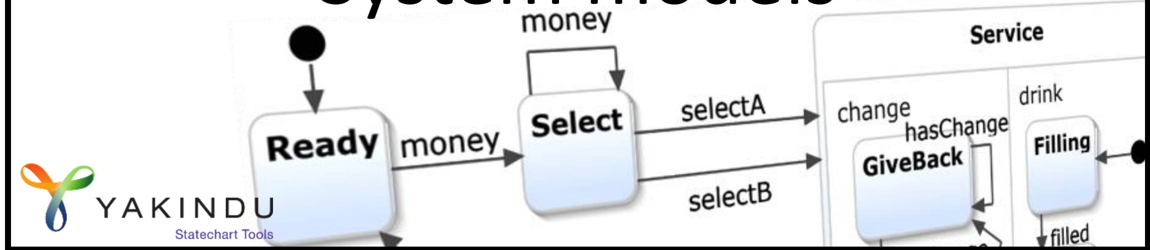
Kristóf Marussy, *Oszkár Semeráth, Attila Ficsor, Dániel Varró*

MŰEGYETEM 1782
ftsrg Critical Systems Research Group
McGill UNIVERSITY
LINKÖPING UNIVERSITY
WASP WALLENBERG AI, AUTONOMOUS SYSTEMS AND SOFTWARE PROGRAM
amazon research awards
Office of Naval Research Science & Technology

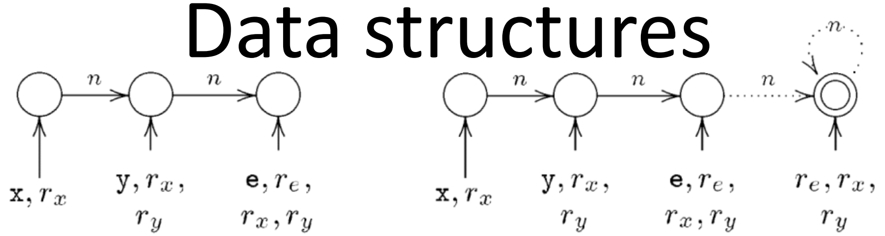
Modeling with Graphs

- Graph based models are widely used in software engineering

System models



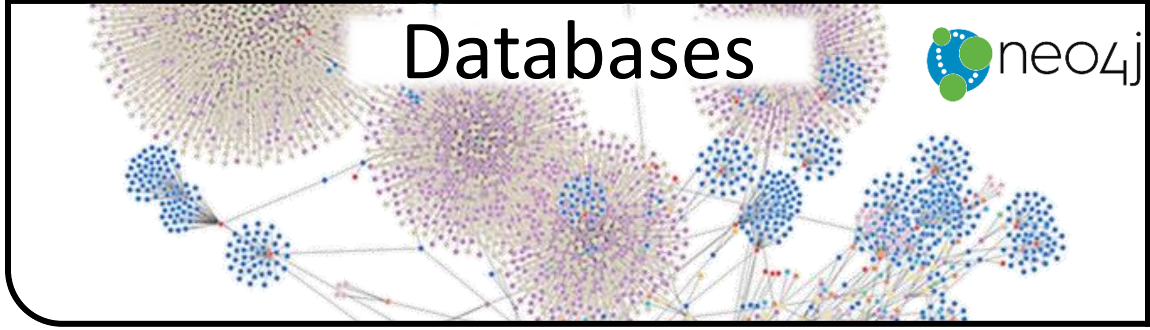
Data structures



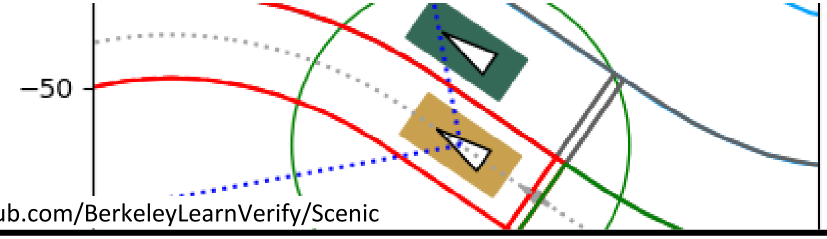
Sagiv, M., Reps, T., & Wilhelm, R. (2002). Parametric shape analysis via 3-valued logic.

Databases

neo4j



Test environments



https://github.com/BerkeleyLearnVerify/Scenic

- Testing, benchmarking or design space exploration scenarios

Generating **(consistent | realistic | diverse | scalable)** models

Hands-on demo

- Code examples available at https://refinery.tools/learn/tutorials/project/
- Watch out for numbered code examples!

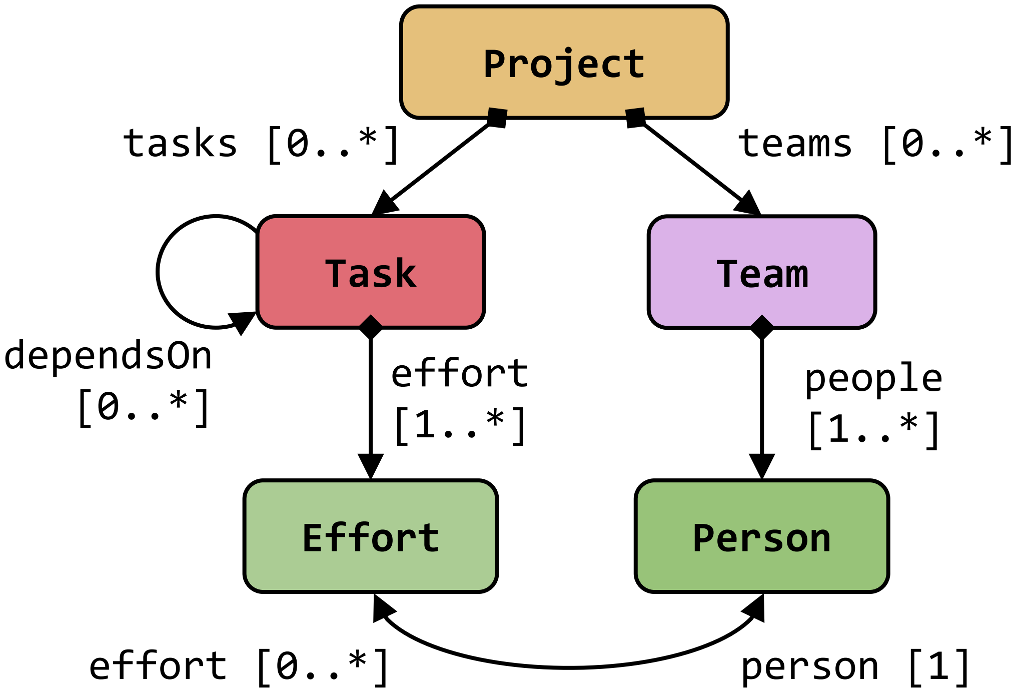
→ Example 1

Graph Structure: Project

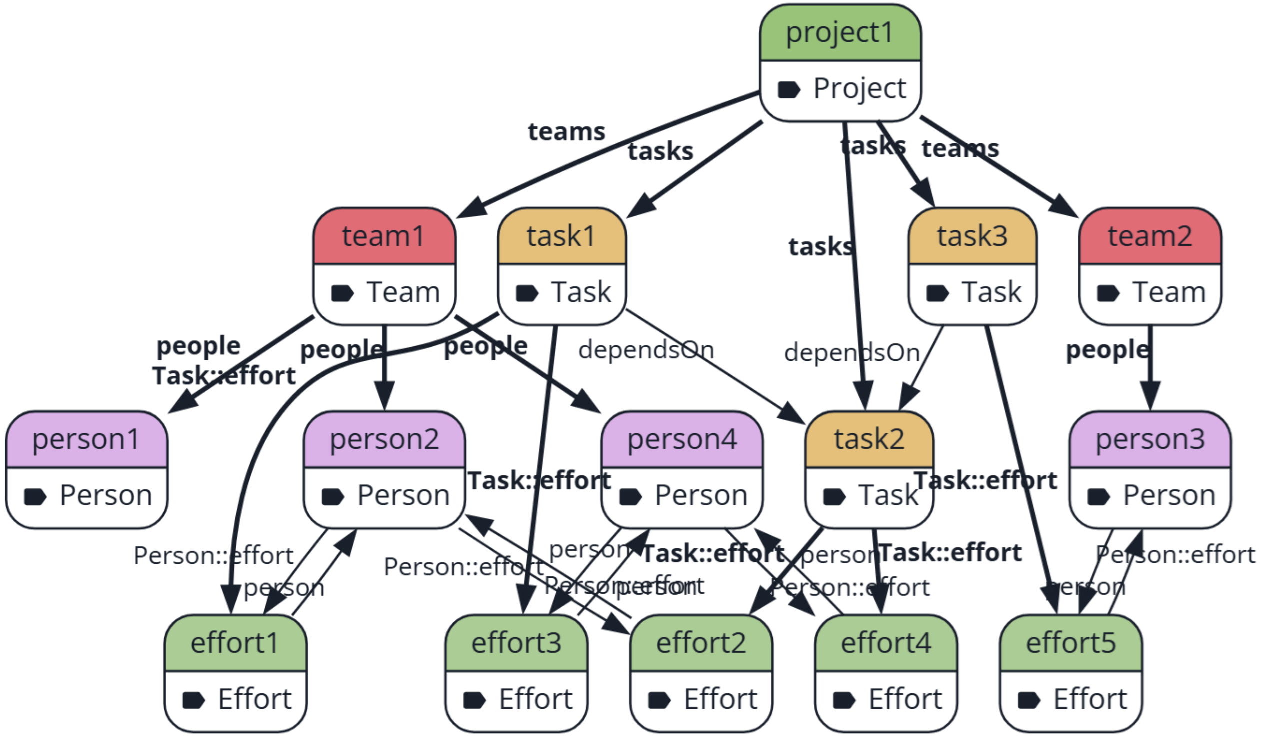
- Typical modeling workflow: **metamodel → instance model**
- **Example:** Tasks, people, and teams in a project

→ Example 2

Metamodel



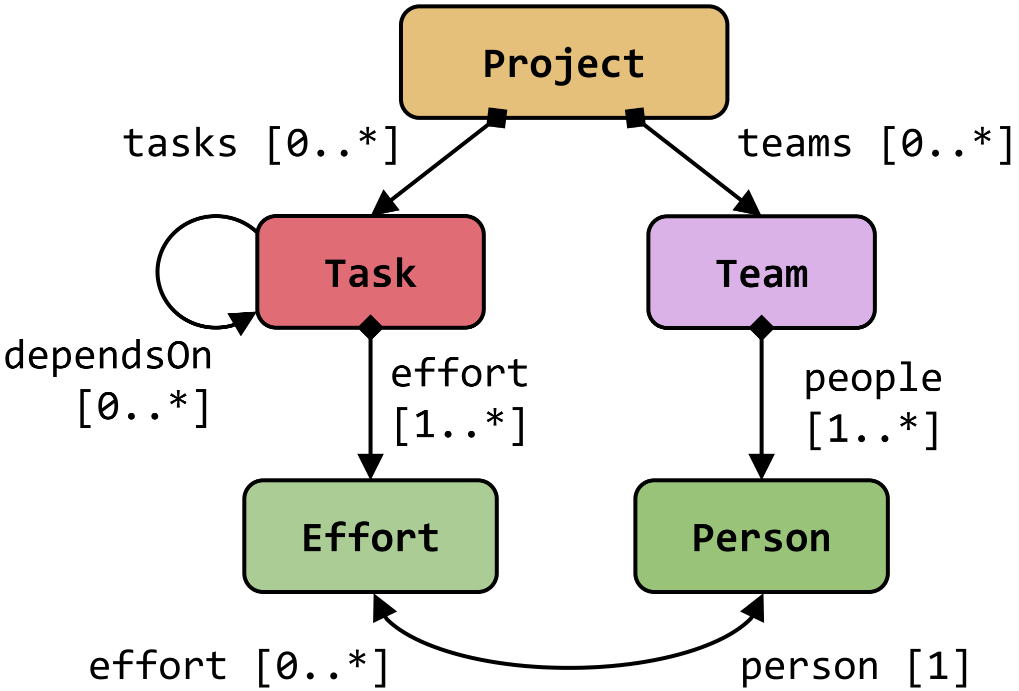
Instance Model



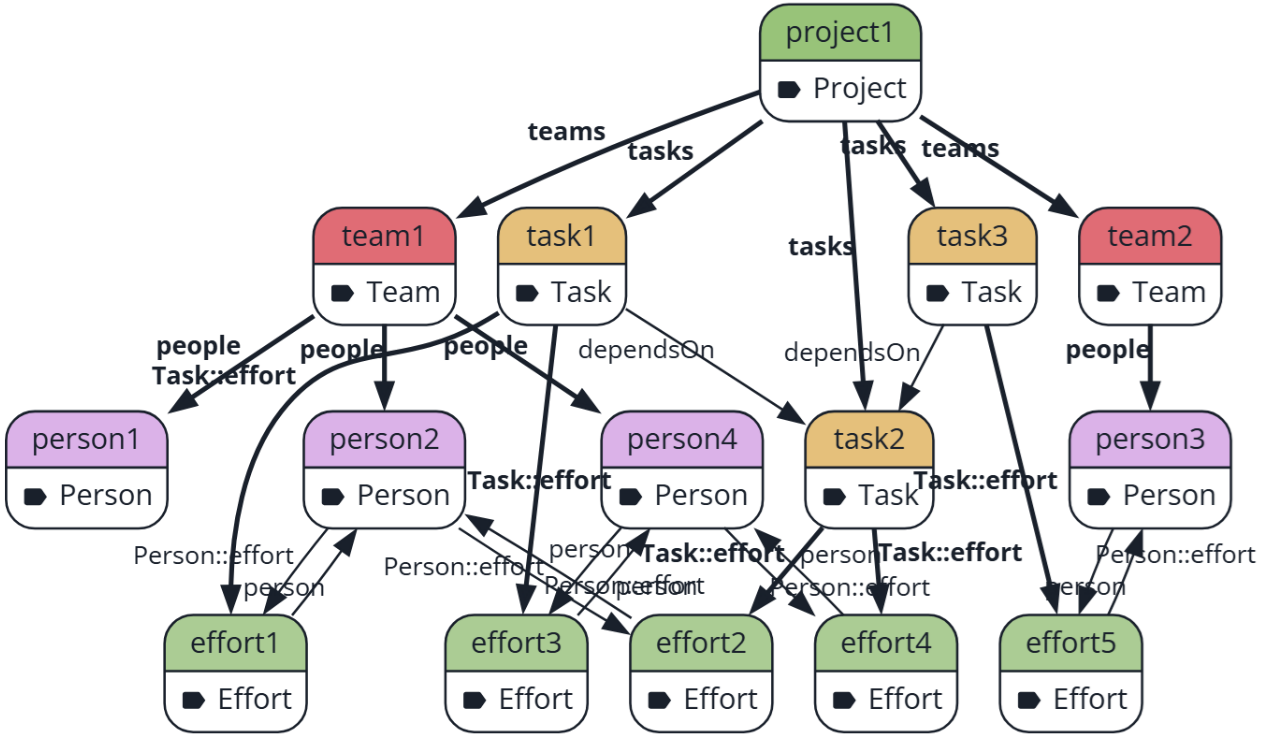
Graph Structure: Project

- Typical modeling workflow: **metamodel → instance model**
- **Example:** Tasks, people, and teams in a project

Metamodel



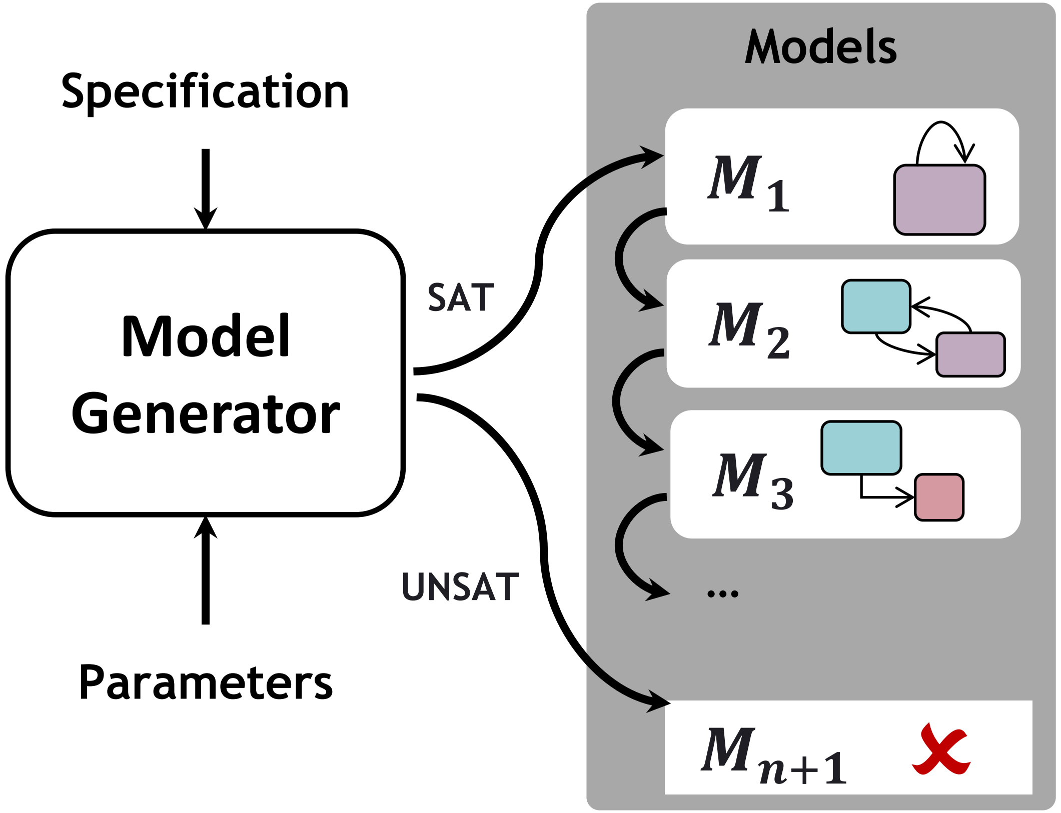
Instance Model



Consistent Graph Generation

Architecture of a generator:

- **Input:** Problem Specification
  Defines the structure of the models
  Defines the consistency constraints
- **Input:** Search Parameters
  Configures the generation process
- **Output:** Models
  Sequence of consistent models
- **Output:** Inconsistency
  Proving that there are no such consistent model



Overview of Refinery Demo

Domain specification (metamodel)

- Define classes (nodes) and relations (edges)

4-valued partial model specification

- Seed partial model to extend (with reasoning)

Constraint specification

- Graph query language (inspired by Datalog / VIATRA Query)

Graph generation parameters

- Bounds on how many nodes an instance model needs to contain

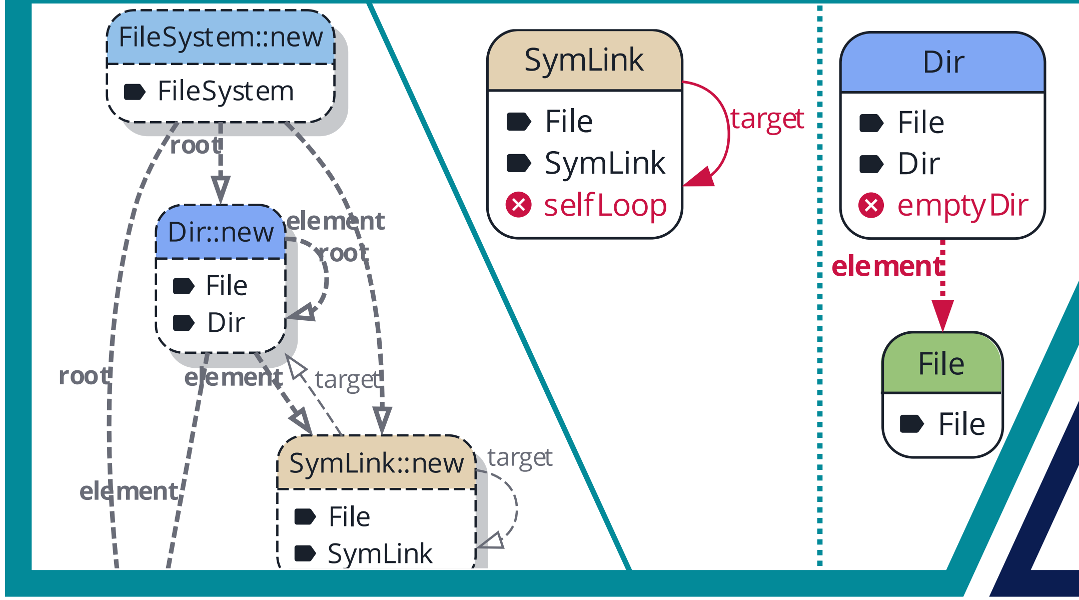
Domain-specific partial models

4-valued logic for reasoning about graph models

Models and Partial Models

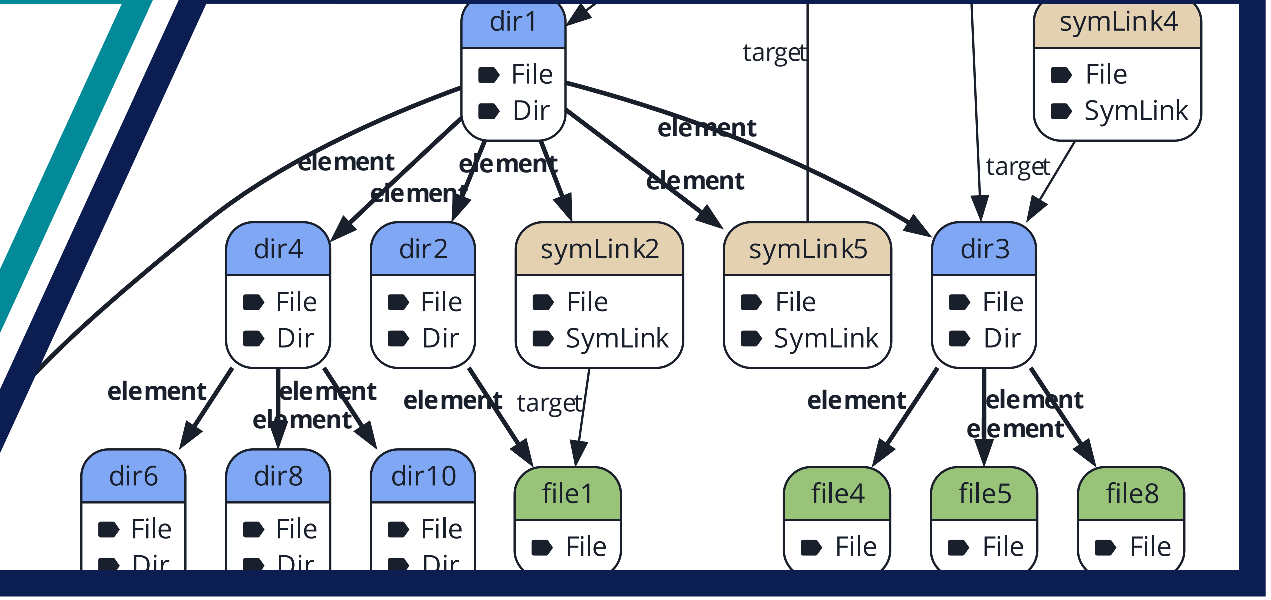
Input

Abstract models
(Metamodel + Constraints)



Output

Concrete models
(Labelled graphs)



Model generation: exploration process that gradually reduces uncertainty

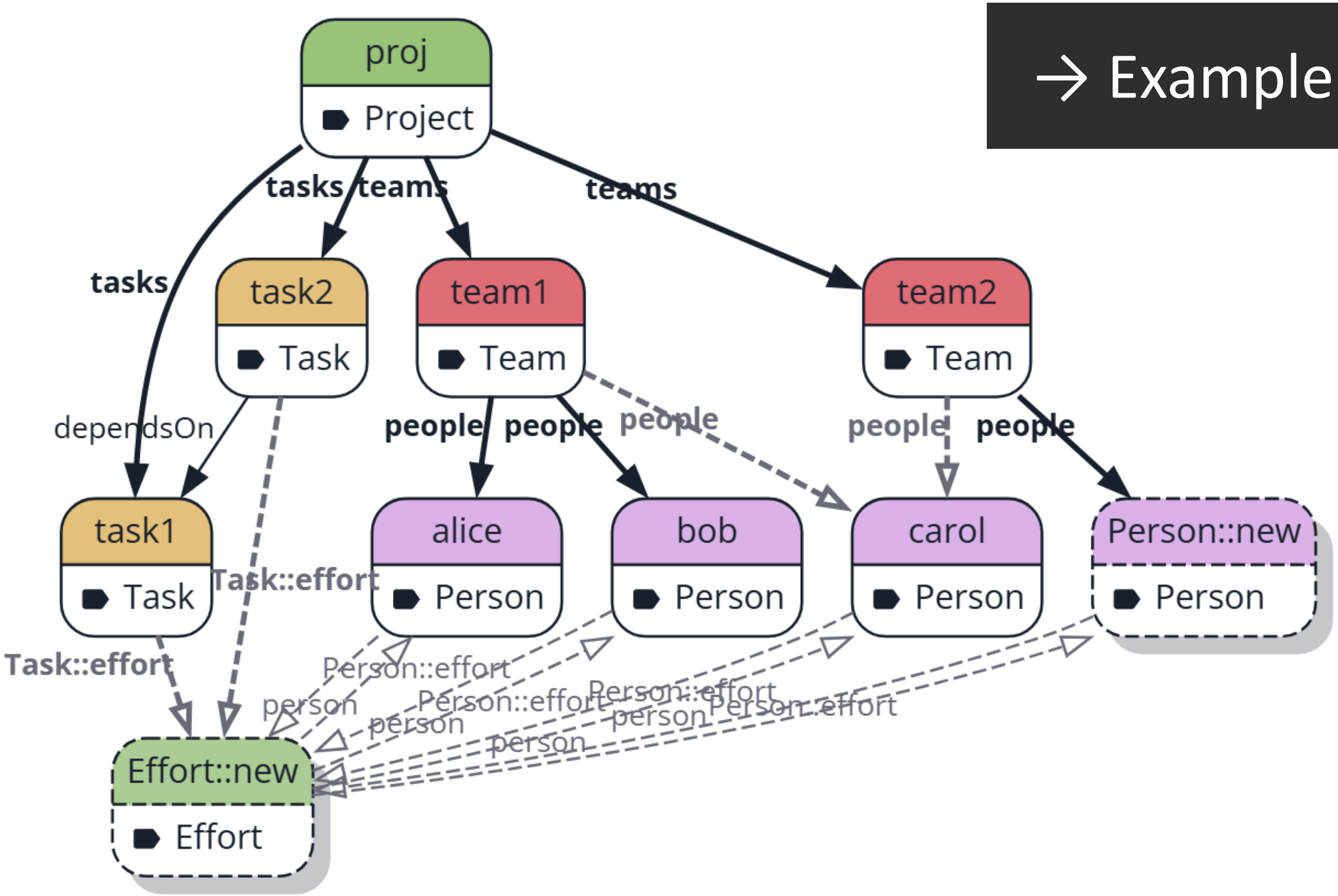
Partial Modeling with 4-valued Logic

→ Example 3

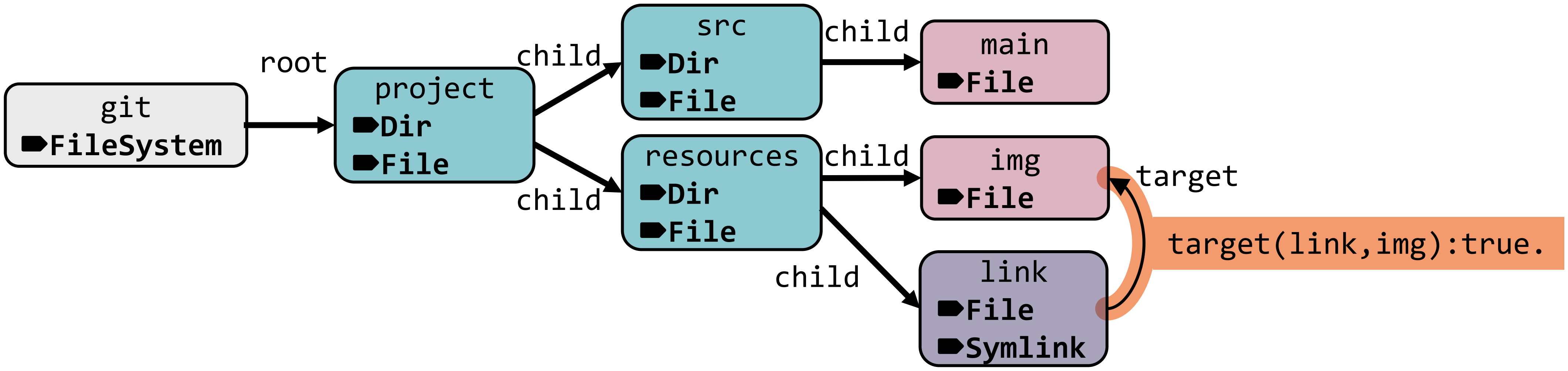
- Represent all potential extensions with **uncertainty**
- Logic abstraction:

  ■**TRUE** | **False** |
  □**Unknown** | ⊗**Error**

  – 4-valued **exists**: added or removed
  – 4-valued **equals**: merging or splitting
- **Refinement:** reduces uncertainty → concrete models



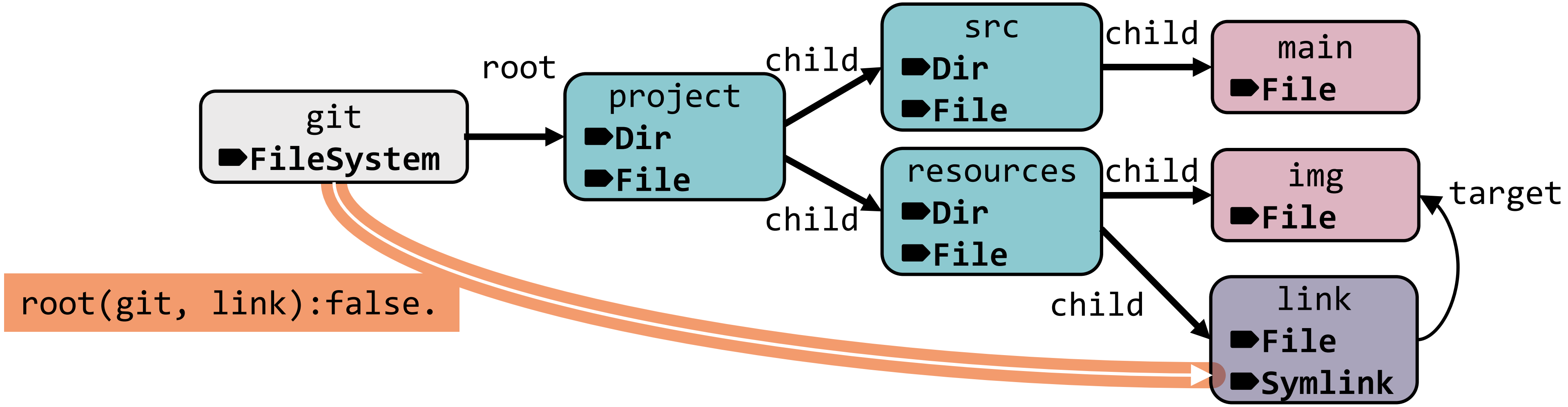
Partial modeling with 4-valued logic



target(link,img):true.

- Represent all potential extension with uncertainty
- Logic abstraction: **■TRUE | False | □Unknown | ⊗Error**

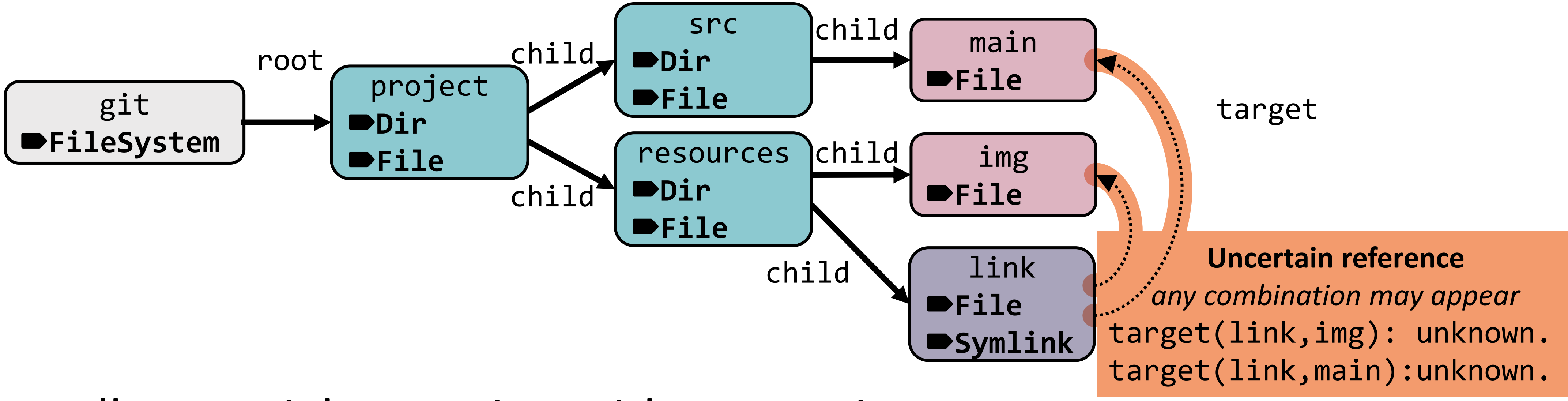
Partial modeling with 4-valued logic



root(git, link):false.

- Represent all potential extension with uncertainty
- Logic abstraction: ■TRUE | False | □Unknown | ⊗Error

Partial modeling with 4-valued logic



git
➡FileSystem

root

project
➡Dir
➡File

child

child

src
➡Dir
➡File

resources
➡Dir
➡File

child

child

child

child

main
➡File

img
➡File

link
➡File
➡Symlink

target

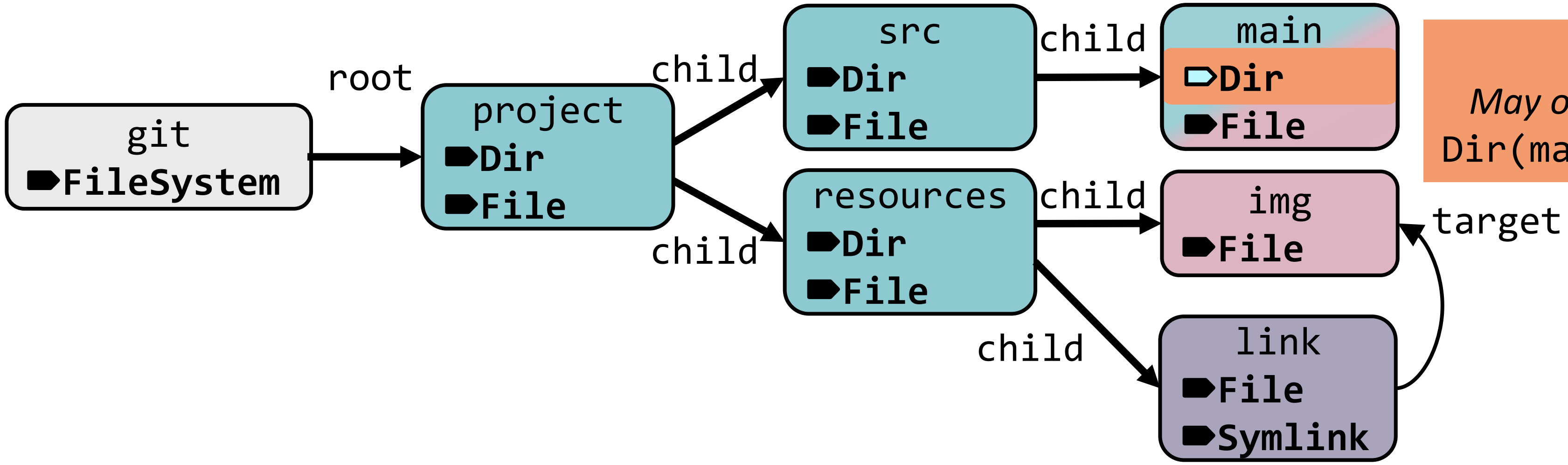
Uncertain reference

any combination may appear

```
target(link,img): unknown.
target(link,main):unknown.
```

- Represent all potential extension with uncertainty
- Logic abstraction: **➡TRUE | False | ⇨Unknown | ⊗Error**

Partial modeling with 4-valued logic



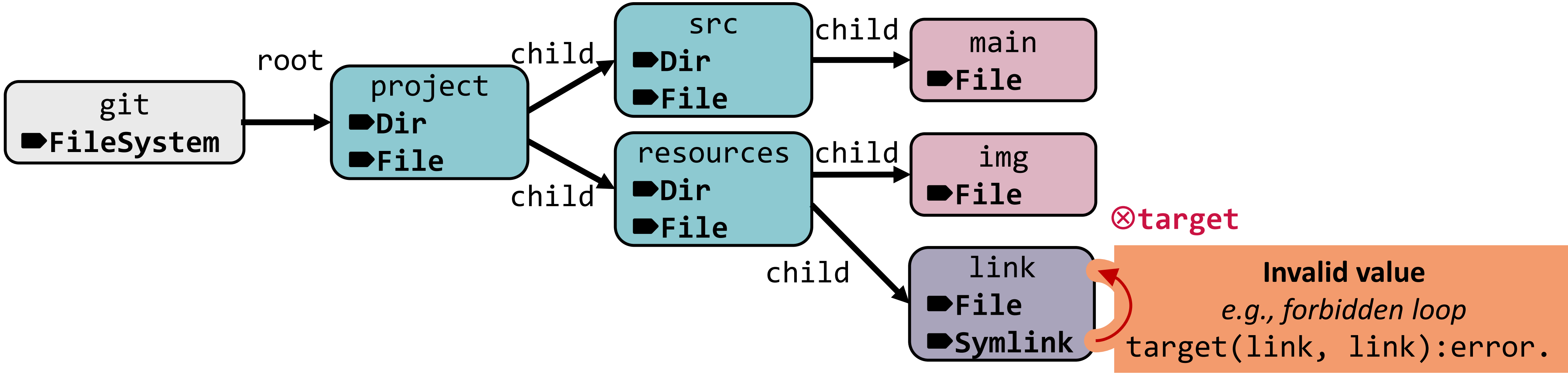
Uncertain type

May or may not have the type

`Dir(main): unknown.`

- Represent all potential extension with uncertainty
- Logic abstraction: ■**TRUE** | **False** | □**Unknown** | ⊗**Error**

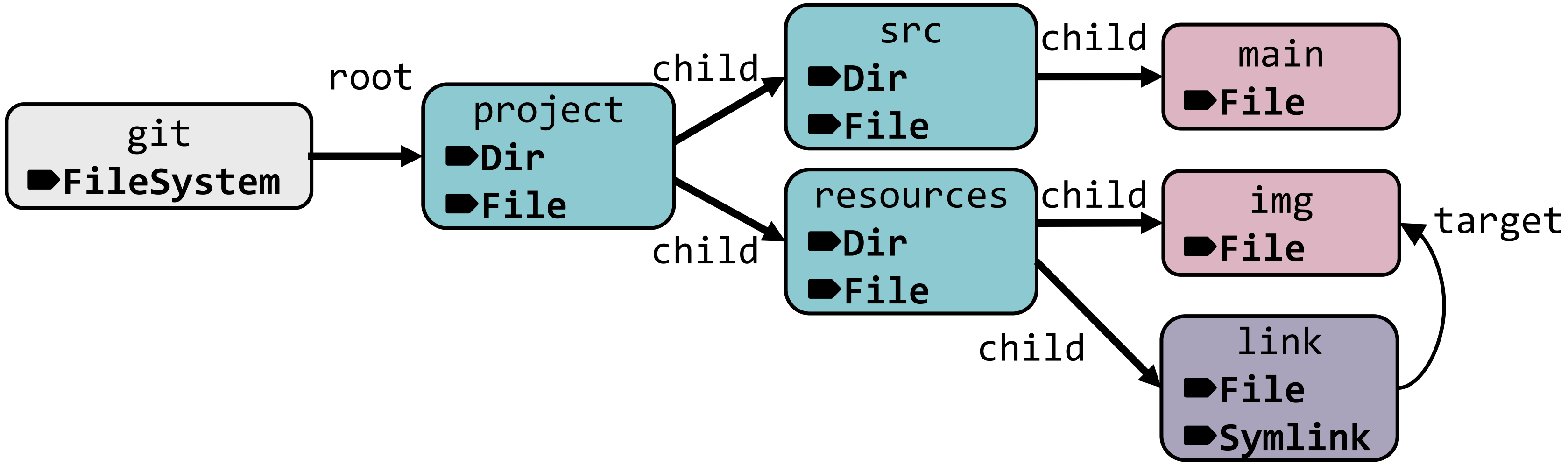
Partial modeling with 4-valued logic



Invalid value
e.g., forbidden loop
`target(link, link):error.`

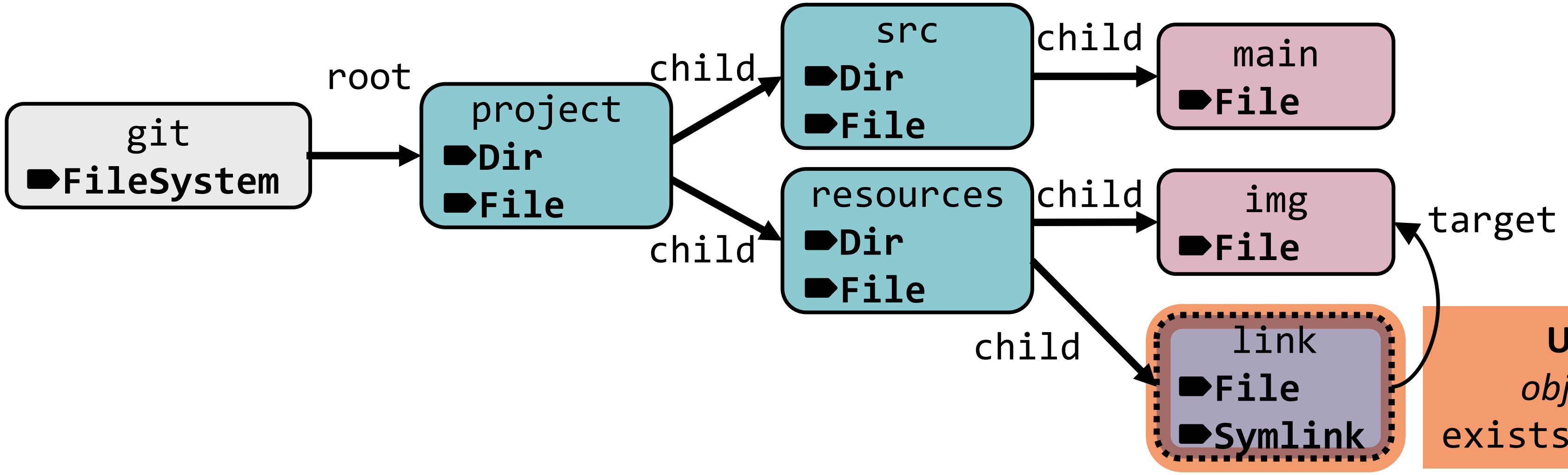
- Represent all potential extension with uncertainty
- Logic abstraction: **■TRUE | False | □Unknown | ⊗Error**

Partial modeling with 4-valued logic



- Represent all potential extension with uncertainty
- Logic abstraction: ▶**TRUE** | **False** | ▷**Unknown** | ⊗**Error**
  - 4-valued **exists**: added or removed
  - 4-valued **equals**: merging or splitting

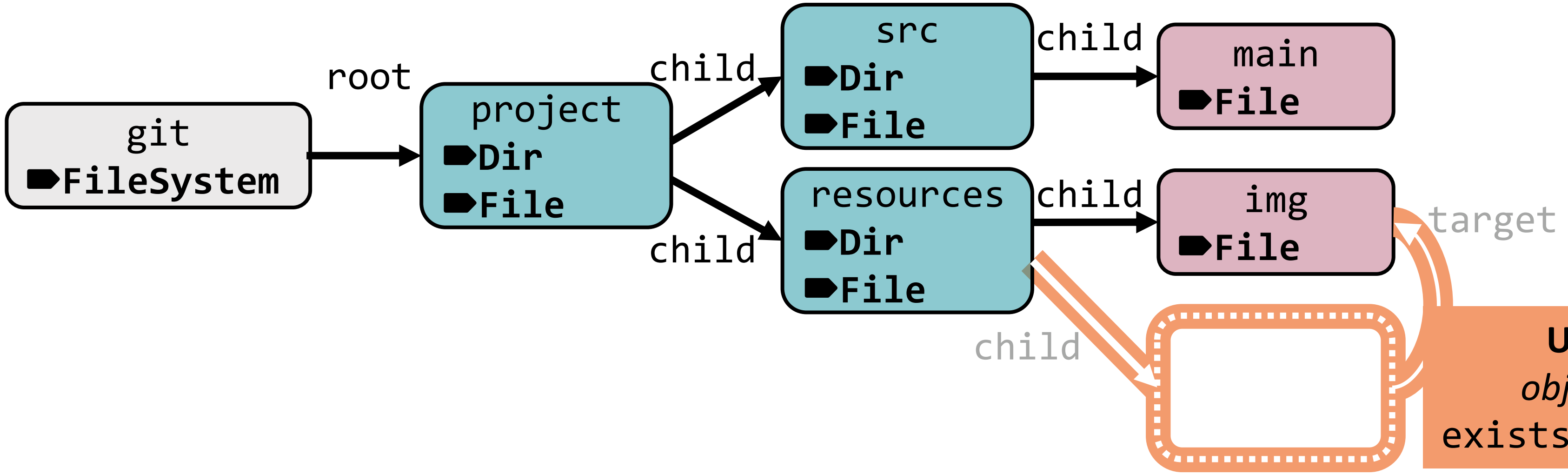
Partial modeling with 4-valued logic



Uncertain existence
object may be removed
`exists(link):unknown.`

- Represent all potential extension with uncertainty
- Logic abstraction: ■**TRUE** | **False** | □**Unknown** | ⊗**Error**
  - 4-valued **`exists`**: added or removed
  - 4-valued **`equals`**: merging or splitting

Partial modeling with 4-valued logic



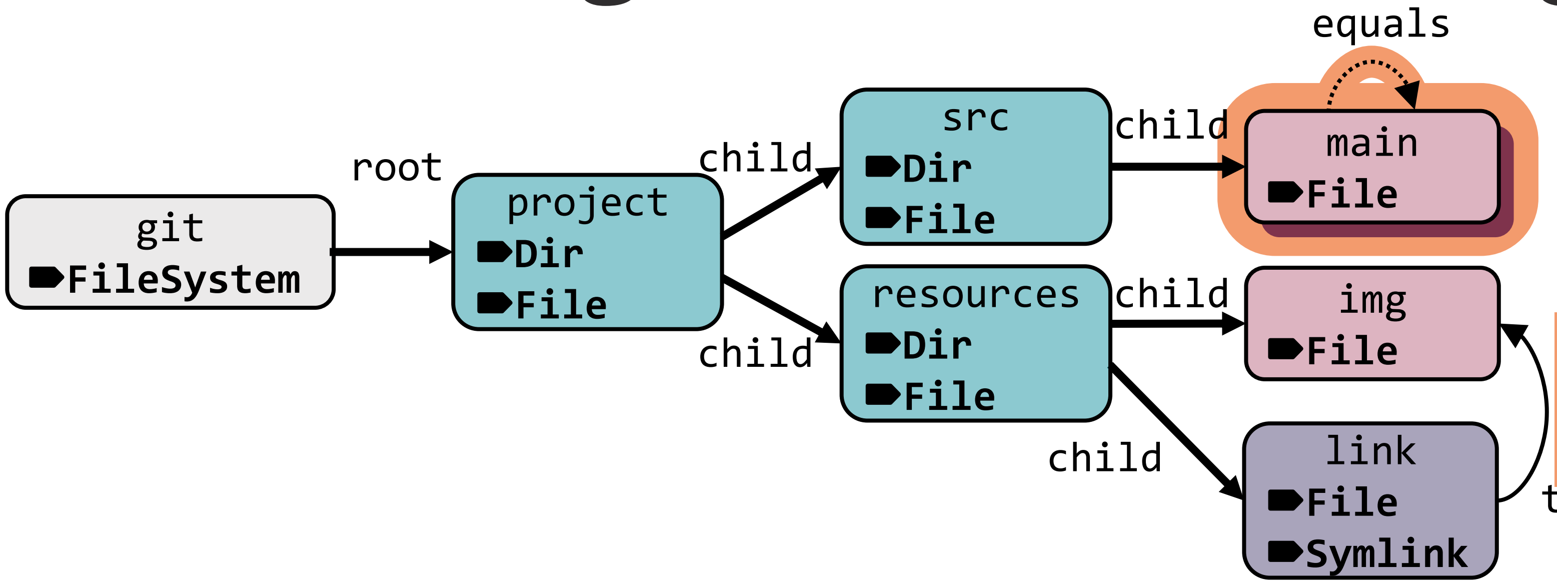
Uncertain existence

object may be removed

`exists(link):unknown.`

- Represent all potential extension with uncertainty
- Logic abstraction: ■**TRUE** | **False** | □**Unknown** | ⊗**Error**
  - 4-valued **`exists`**: added or removed
  - 4-valued **`equals`**: merging or splitting

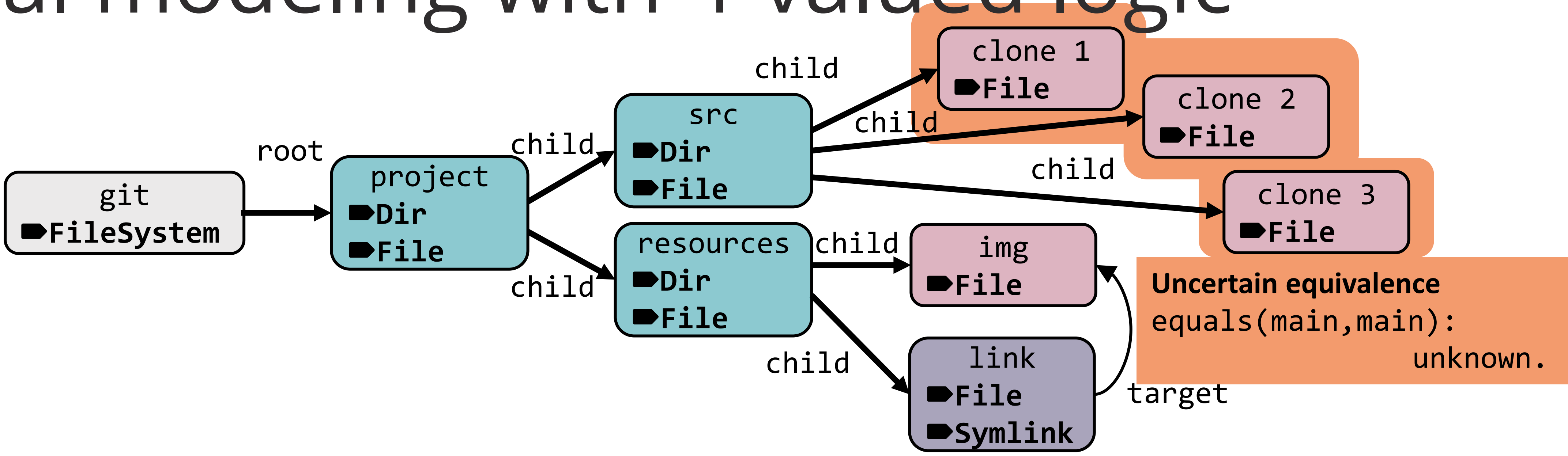
Partial modeling with 4-valued logic



Uncertain equivalence
`equals(main,main): unknown.`

- Represent all potential extension with uncertainty
- Logic abstraction: **TRUE** | **False** | **Unknown** | ⊗**Error**
  - 4-valued **exists**: added or removed
  - 4-valued **equals**: merging or splitting

Partial modeling with 4-valued logic



git
■FileSystem

root

project
■Dir
■File

child

child

src
■Dir
■File

resources
■Dir
■File

child

child

child

child

child

child

clone 1
■File

clone 2
■File

clone 3
■File

img
■File

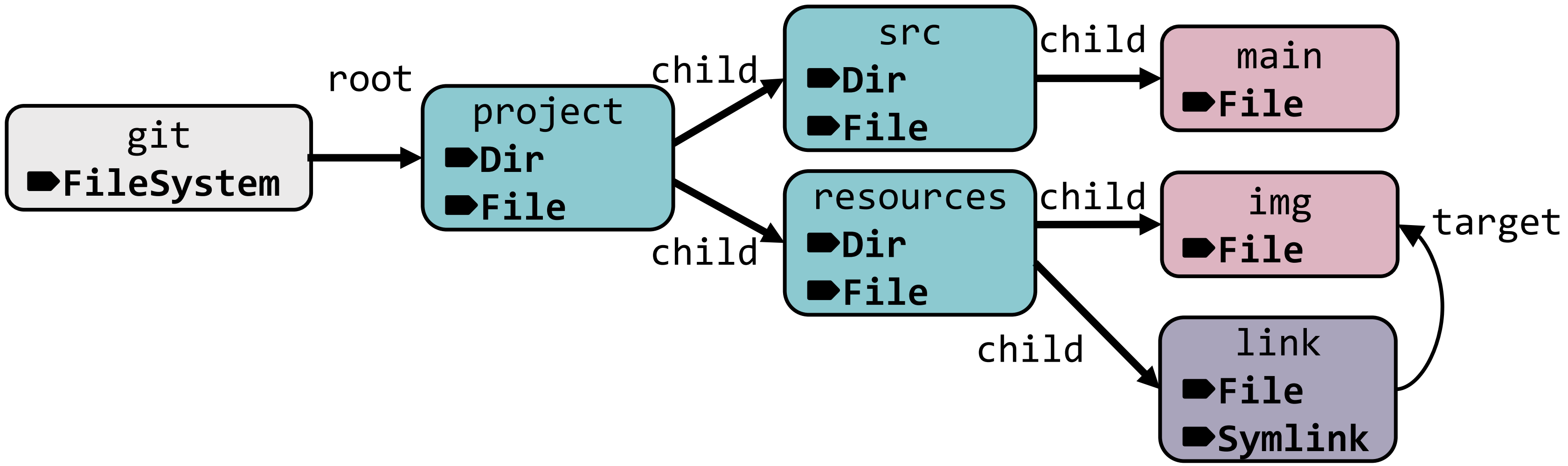
link
■File
■Symlink

target

Uncertain equivalence
`equals(main,main): unknown.`

- Represent all potential extension with uncertainty
- Logic abstraction: ■**TRUE** | **False** | □**Unknown** | ⊗**Error**
  - 4-valued **exists**: added or removed
  - 4-valued **equals**: merging or splitting

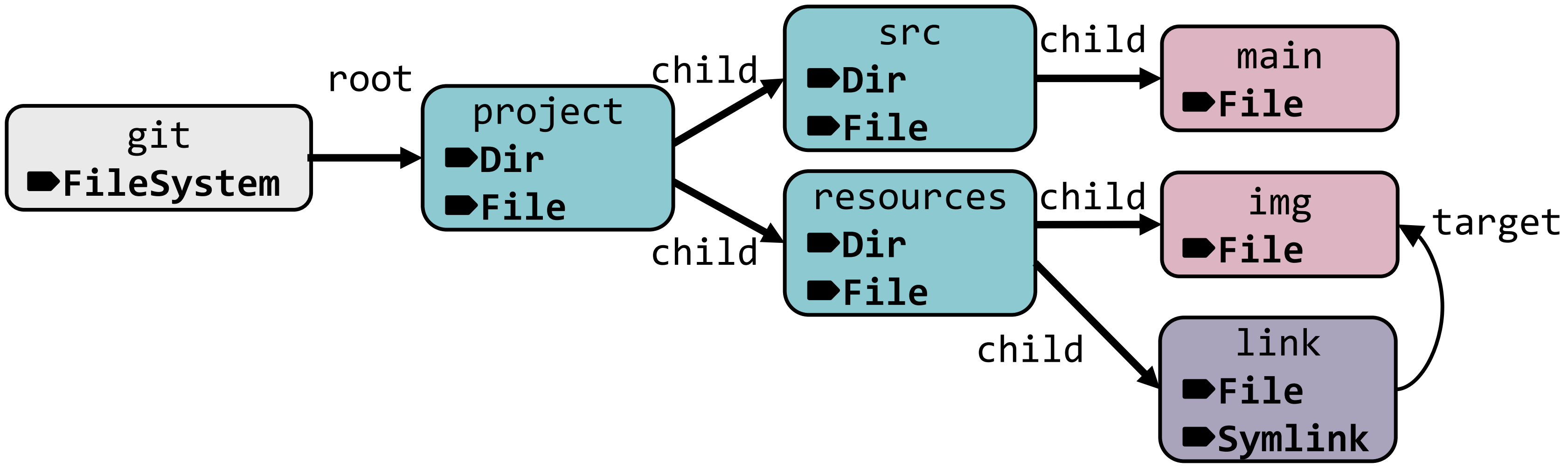
Partial modeling with 4-valued logic



- Represent all potential extension with uncertainty
- Logic abstraction: ■**TRUE** | **False** | □**Unknown** | ⊗**Error**
  - 4-valued **exists**: added or removed
  - 4-valued **equals**: merging or splitting

Model type systems as partial models → Demo

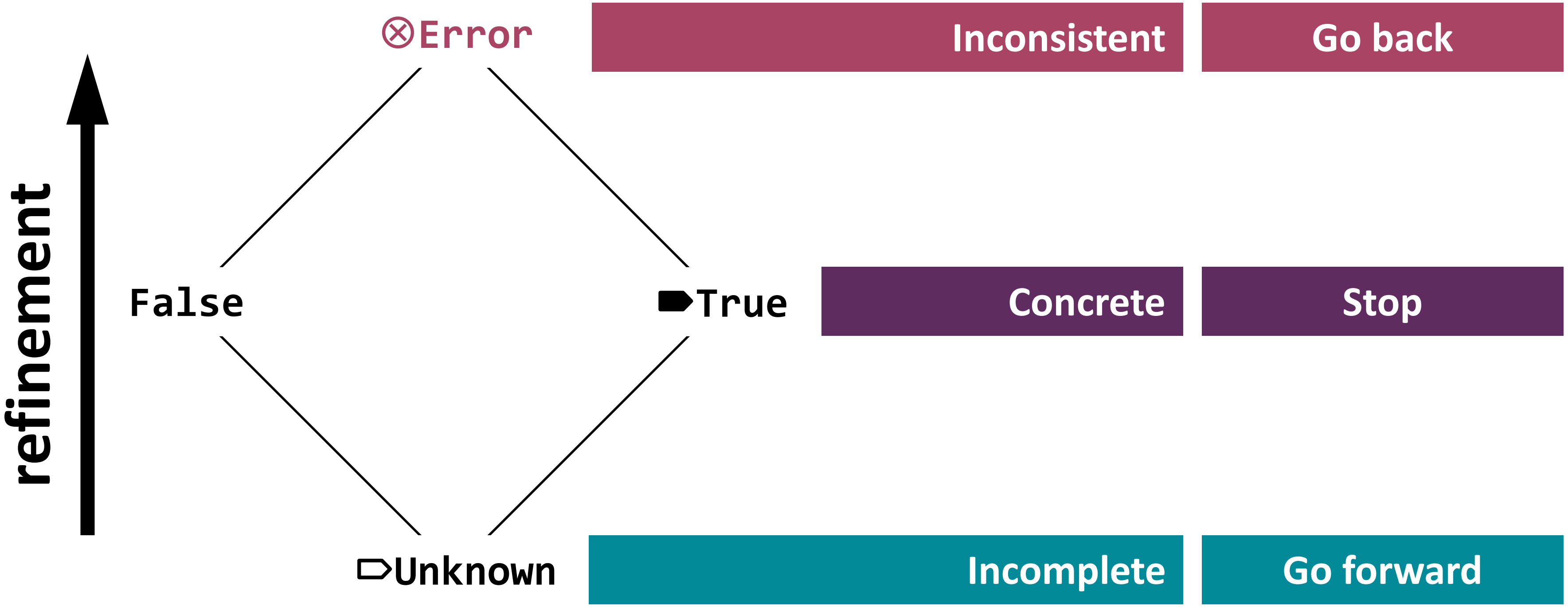
Partial modeling with 4-valued logic



- Represent all potential extension with uncertainty
- Logic abstraction: ■**TRUE** | **False** | □**Unknown** | ⊗**Error**
  - 4-valued **exists**: added or removed
  - 4-valued **equals**: merging or splitting
- **Refinement**: reduces uncertainty → concrete models

Refinement: 4-valued Logic

- Model generation is executed with respect to model refinement



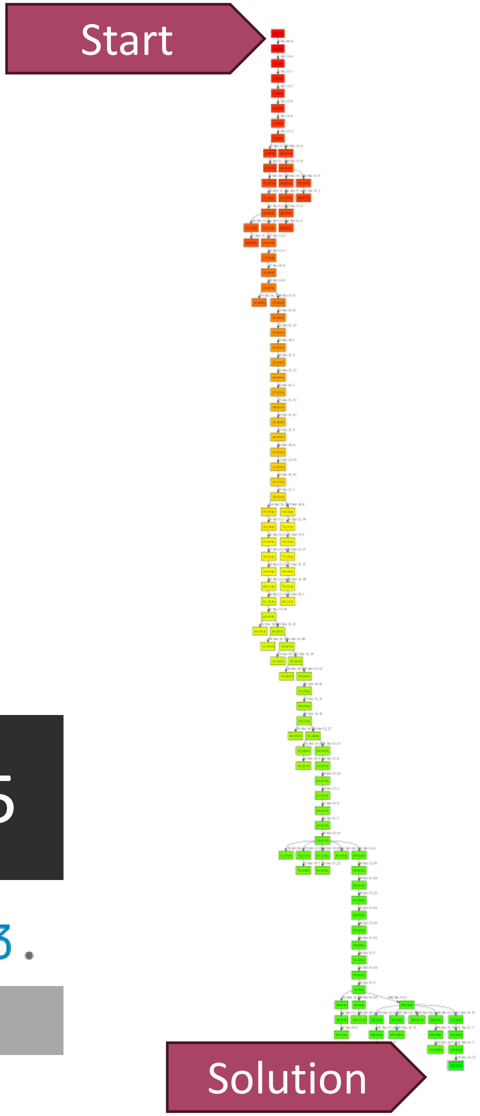
E.g.: $\texttt{person(\_,\_):}\textbf{unknown} \xrightarrow{\texttt{+true}} \texttt{person(\_,\_):}\textbf{true}$

$\texttt{person(\_,\_):}\textbf{true} \xrightarrow{\texttt{+false}} \texttt{person(\_,\_):}\textbf{error}$

Constraint specification

Using graph queries

Search Parameters for Model Generation



- Constraints are continuously reevaluated
- Automatically searching for valid models by applying refinements
- Search is parametrized
  - Number of different solutions
  - Difference between the solutions (non-isomorphic)
  - Random seed
- Scope: *"size of the models"*

→ Examples 4-5

```
scope node = 30..50, Person += 10, Task += 5, Project = 1, Group = 3.
```

# of nodes

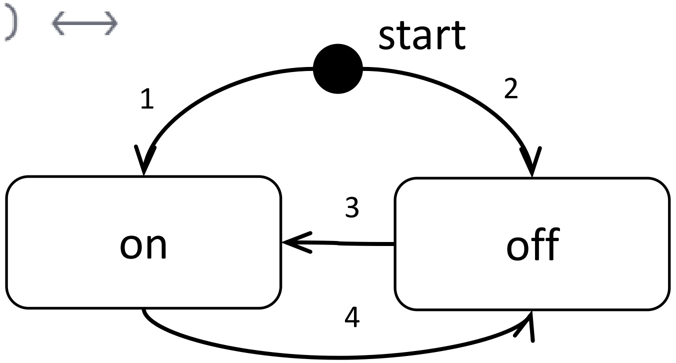
# of new objects

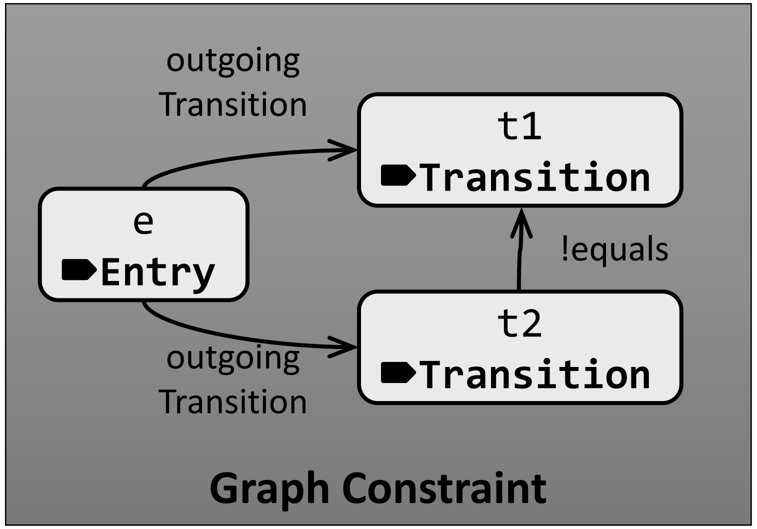
nodes by type

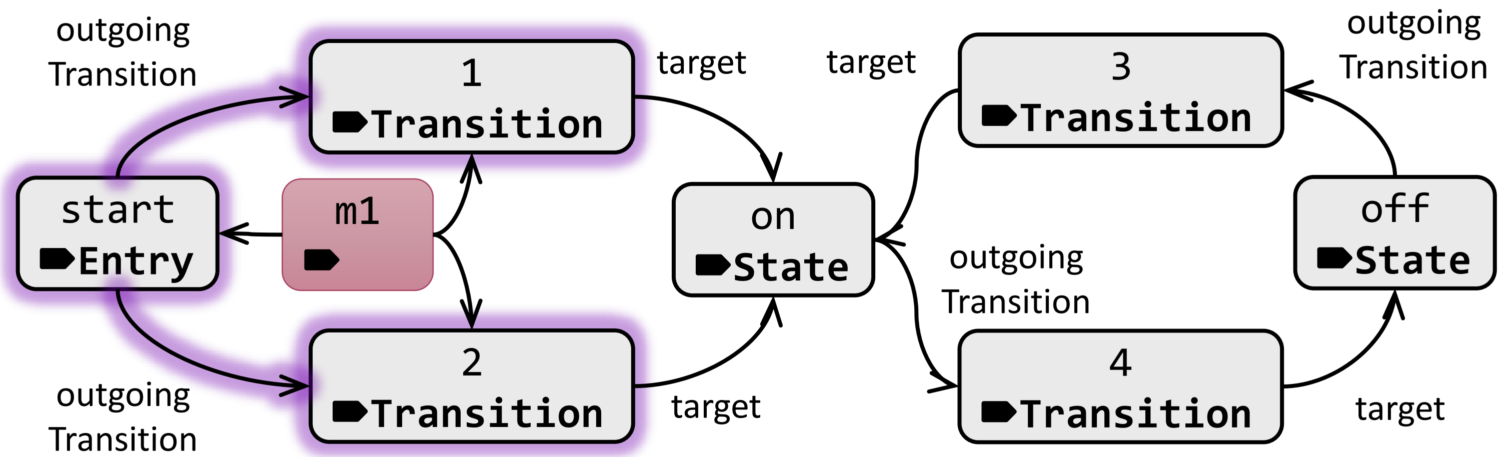
Graph Constraint Evaluation

→ Examples 6-7

```
error multipleTransitionFromEntry(Entry e, Transition t1, Transition t2) ⟷
    outgoingTransition(e, t1),
    outgoingTransition(e, t2),
    t1 ≠ t2.
```



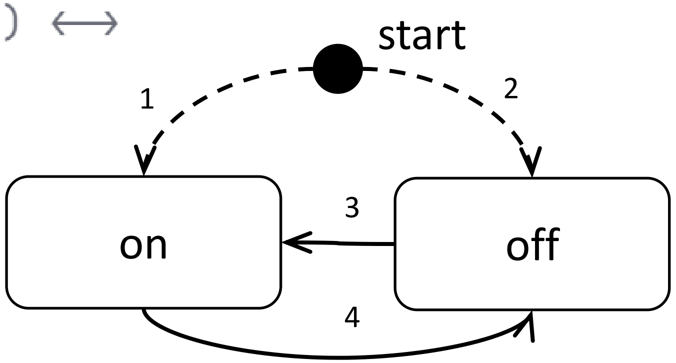


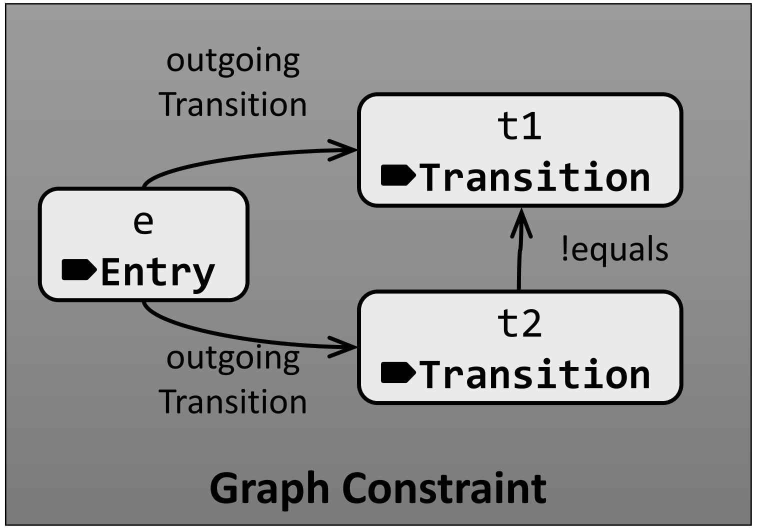


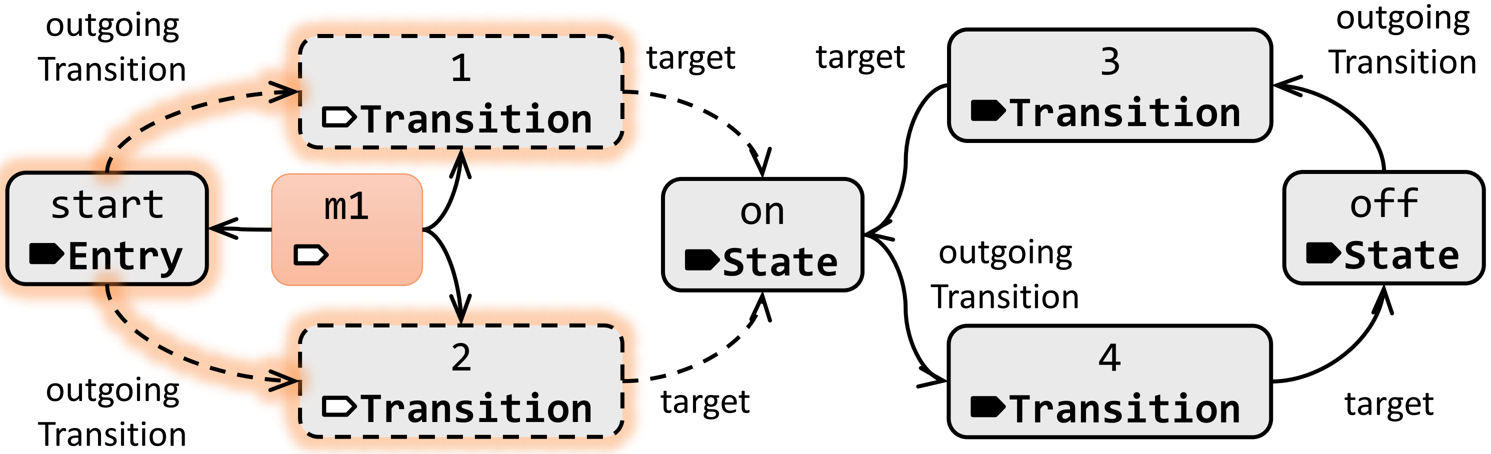
Each match of the query is a certain constraint violation (an error)

Partial Graph Constraint Evaluation

```
error multipleTransitionFromEntry(Entry e, Transition t1, Transition t2) ⟷
    outgoingTransition(e, t1),
    outgoingTransition(e, t2),
    t1 ≠ t2.
```



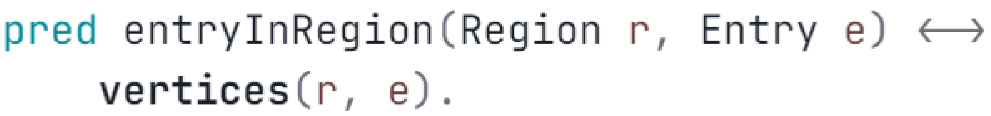




A may-match of a query is a *potential* error (which may disappear)

Predicates vs Constraints

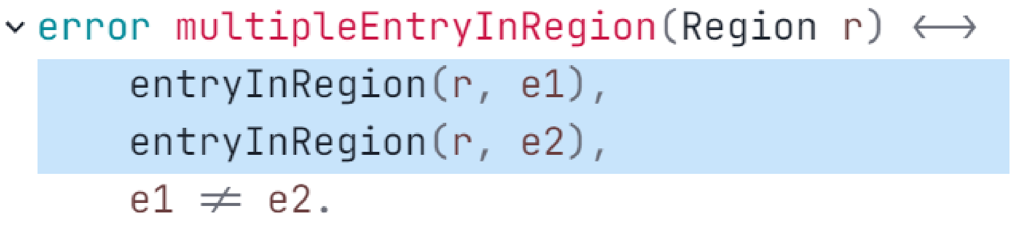
→ Examples 8-9



```
pred entryInRegion(Region r, Entry e) ⟷
    vertices(r, e).
```

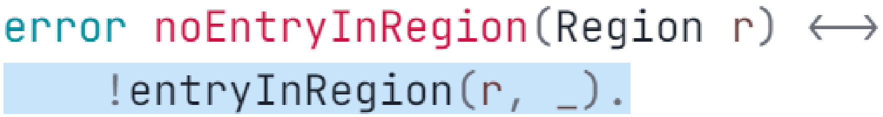
Predicates

- A graph query / predicate
- Composable: Reusable in other predicates or constraints
- Positive condition

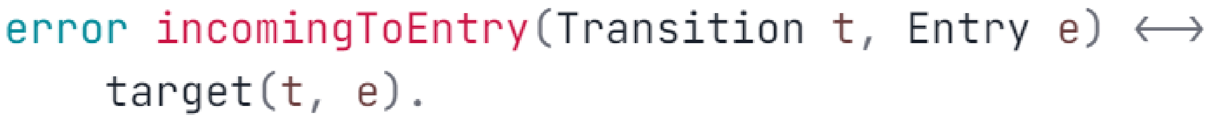


```
error multipleEntryInRegion(Region r) ⟷
    entryInRegion(r, e1),
    entryInRegion(r, e2),
    e1 ≠ e2.
```

- Negative condition



```
error noEntryInRegion(Region r) ⟷
    !entryInRegion(r, _).
```



```
error incomingToEntry(Transition t, Entry e) ⟷
    target(t, e).
```

Constraints (Error patterns)

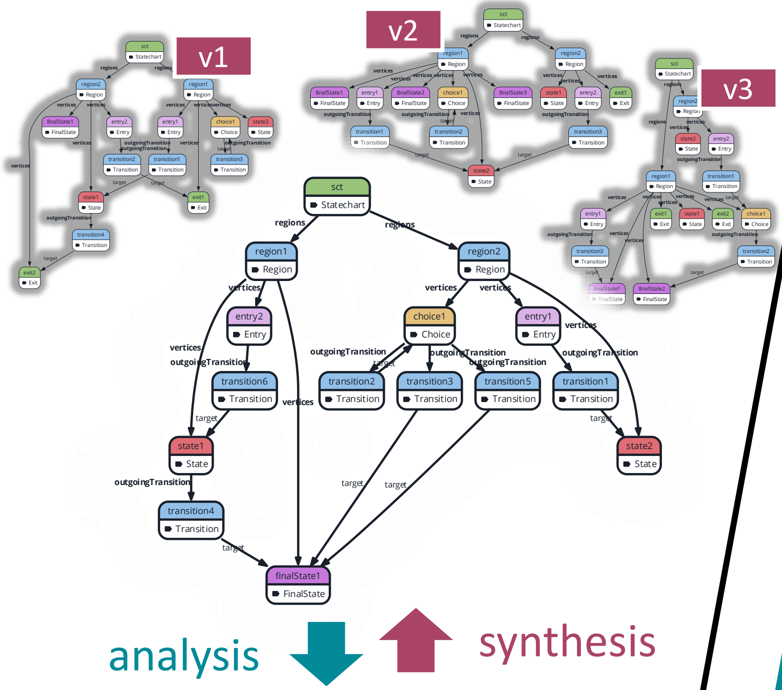
- Capture the violating cases of a domain constraint
- Each match is an error (inconsistency)
- Predicates vs. Types
  - 1-parameter predicate: special *node* type
  - 2-parameter predicate: special *edge* type

Refinery elsewhere

Applications & appearances

Graph analysis and synthesis

- Powerful mathematical **analysis techniques** for models
- Novel graph-based logic solver for the **automated synthesis** of design alternatives
- **Precision + Scalability**
- **Goal:** solve problems with complex structure



☒ ☑ validation

$$$ cost

◢◢ performance

%% coverage

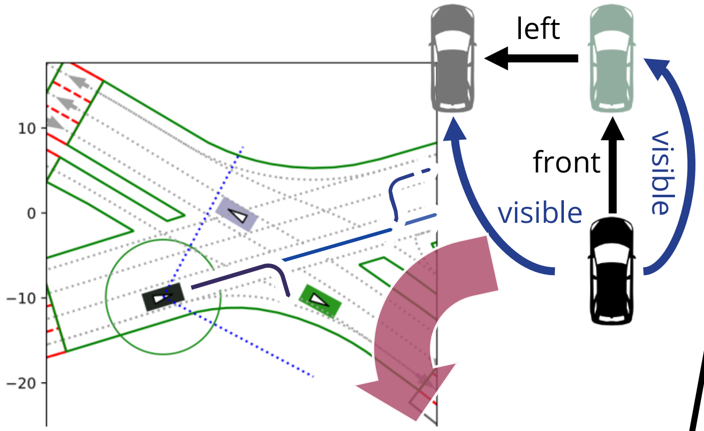
Recent Results

- Research project

VERIFIABLE AI/ML TECHNIQUES FOR PNT APPLICATIONS

Verification/Testing of AI/ML Applications

- AI applications are **data-oriented** systems
- **Complex, dynamic** environment
- Novel **generation** + Advanced **simulators**
  → Diverse **tests**
- Systematic testing of **AI applications**



Recent Results

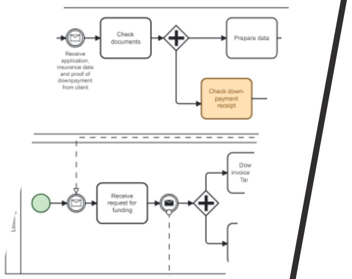
- R&D project with Knorr-Bremse
- Research project with USA Navy

Office of Naval Research Global

Advancing DLT applications

BLOCKCHAIN Koalíció

- **Hungarian Blockchain Coalition**
  - Prof. Pataricza – member of the board
  - I. Kocsis: Education WG lead, L. Gönczy: FinTech WG
- **Supporting the EMAP project (PM/NAV)**
  - "Even-based Data-sharing Platform" pilot
  - Employer data provisions: event-based, single-channel
  - Blockchain-based implementation in preparation
- **CBDC research cooperation with MNB**
  - Mapping out: blockchain ↔ Central Bank Digital Currency
  - Payment, car leasing, energy support, industrial cooperation
  - Currently: "ecosystem" research
- **EDGE-Skills: data veracity in EU data spaces**
  - Blockchain-backed Verifiable Credentials


Check documents
Prepare data
Receive application, insurance data and proof of downpayment from client
Check down-payment receipt
Receive request for funding
Dow invoice Ta

Recent results

Energy price support CBDC prototype: BIS Rosalind finalist

Fabric ↔ Ethereum CBDC bridge in Hyperledger Cacti

Smart gas meters and readings – in production

Refinery@MODELS2024

- Friday 14:30 (FAME) – *Refinery hands-on session*
- Sunday 16:00 (Super Mario Bros)
  T9: Refinery: Logic-based partial modeling
- Wednesday 15:24 (HS7 – Applications 1)
  Ulf Kargén, Dániel Varró. Towards Automated Test Scenario Generation for Assuring COLREGs Compliance of Autonomous Surface Vehicles
  - Find inconsistencies in maritime traffic rules with partial modeling
- Thursday 15:45 (HS1 – MDE&AI)
  José Antonio Hernández López, Máté Földiák, Dániel Varró. Text2VQL: Teaching a Model Query Language to Open-Source Language Models with ChatGPT
  - Generate graph models to verify graph queries generated by ChatGPT
- Thursday 15:45 (HS7 – Applications 2)
  Noor Al-Gburi, András Földvári, Kristóf Marussy, Oszkár Semeráth, Imre Kocsis. Requirement-Driven Generation of Distributed Ledger Architectures
  - Generate architectures for consortial blockchain systems

Further Information

Specification language

- K. Marussy, O. Semeráth, A. Babikian, D. Varró: A Specification Language for Consistent Model Generation based on Partial Models. J. Object Technol. 19(3): 3:1-22 (2020)

Consistent graph generation techniques

- O. Semeráth, A. Nagy, D. Varró: A graph solver for the automated generation of consistent domain-specific models. ICSE 2018: 969-980
- K. Marussy, O. Semeráth, D. Varró: Automated Generation of Consistent Graph Models With Multiplicity Reasoning. IEEE Trans. Software Eng. 48(5): 1610-1629 (2022)
- A.. Babikian, O. Semeráth, A. Li, K. Marussy, D. Varró: Automated generation of consistent models using qualitative abstractions and exploration strategies. Softw. Syst. Model. 21(5): 1763-1787 (2022)

Diverse and realistic graph generation

- O. Semeráth, R. Farkas, G. Bergmann, D. Varró: Diversity of graph models and graph generators in mutation testing. Int. J. Softw. Tools Technol. Transf. 22(1): 57-78 (2020)
- O. Semeráth, A. Babikian, B. Chen, C. Li, K. Marussy, G. Szárnyas, D. Varró: Automated generation of consistent, diverse and structurally realistic graph models. Softw. Syst. Model. 20(5): 1713-1734 (2021)

Correctness proofs

- D. Varró, O. Semeráth, G. Szárnyas, Á. Horváth: Towards the Automated Generation of Consistent, Diverse, Scalable and Realistic Graph Models. Graph Transformation, Specifications, and Nets 2018: 285-312

Summary

- **Logic reasoning** and **model generation** over graphs
- **Web-based editor:**
  - **Live editing** and **feedback**
  - Support for partial models and graph constraints
- **Containerized execution:**
  - Continuously deployed at https://refinery.services
  - Available as **Docker image:** https://refinery.tools/learn/docker/
- **Open-source project:** https://refinery.tools

MŰEGYETEM 1782
ftsrg Critical Systems Research Group
McGill UNIVERSITY
LIU LINKÖPING UNIVERSITY
WASP WALLENBERG AI, AUTONOMOUS SYSTEMS AND SOFTWARE PROGRAM
amazon research awards
USN Office of Naval Research ONR Science & Technology