Efficient Manipulation of Logical Formulas as Decision Diagrams

Milán Mondok, Vince Molnár

Budapest University of Technology and Economics, Hungary

Supported by the **UNKP-23-3-I-BME-8** New National Excellence Program of the Ministry for Culture and Innovation from the source of the National Research, Development and Innovation Fund

MŰEGYETEM 1782

BME FACULTY OF ELECTRICAL ENGINEERING AND INFORMATICS

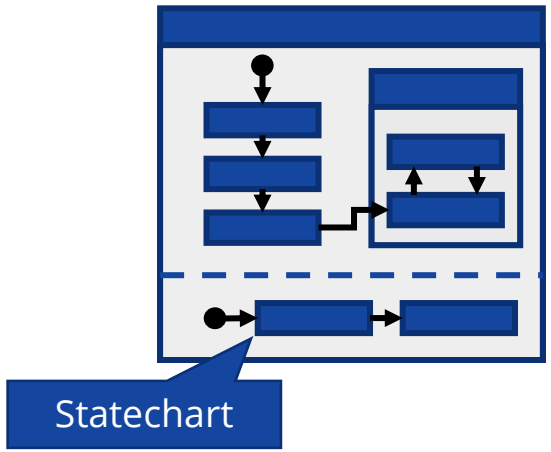
ftsrg Critical Systems Research Group

Motivation

- Heterogeneous system models
  - Mix of multiple high-level languages
  - SysMLv2: new OMG systems modeling language

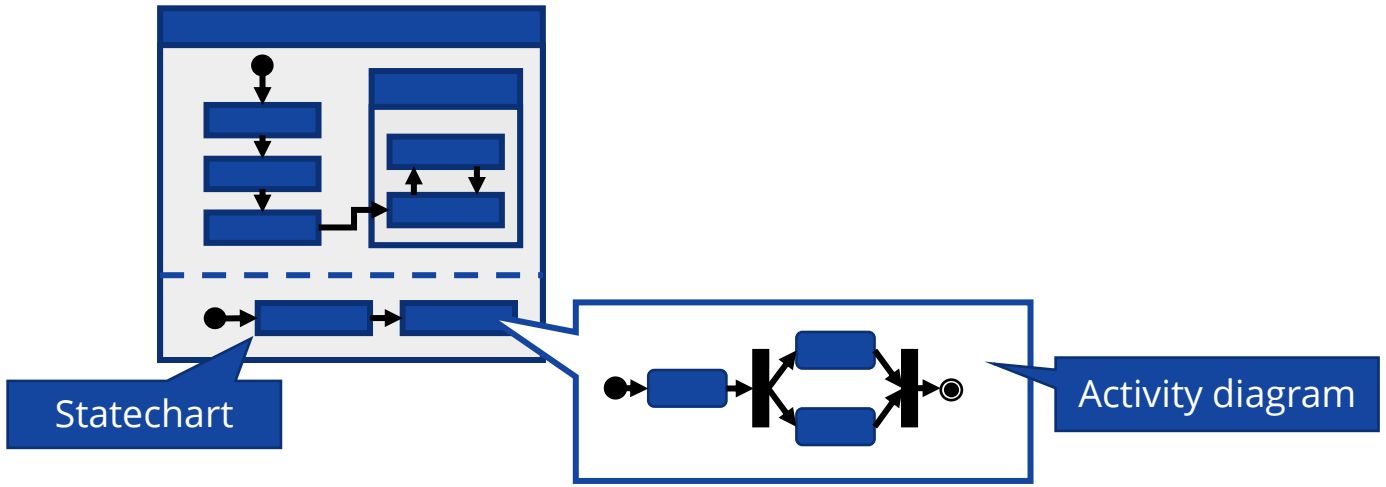
Motivation

- Heterogeneous system models
  - Mix of multiple high-level languages
  - SysMLv2: new OMG systems modeling language



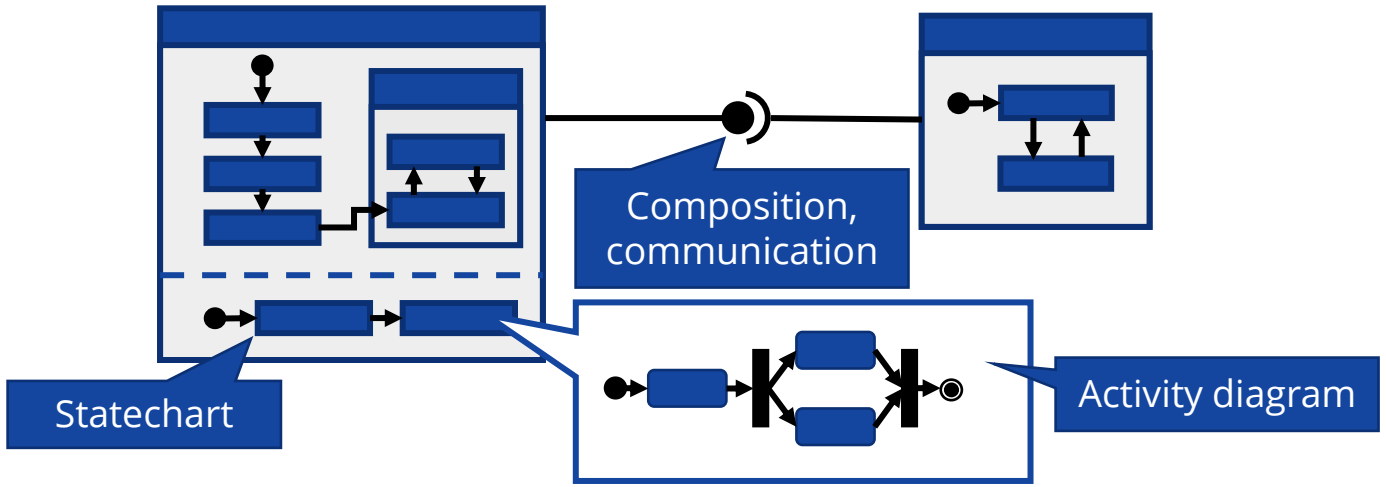
Motivation

- Heterogeneous system models
  - Mix of multiple high-level languages
  - SysMLv2: new OMG systems modeling language



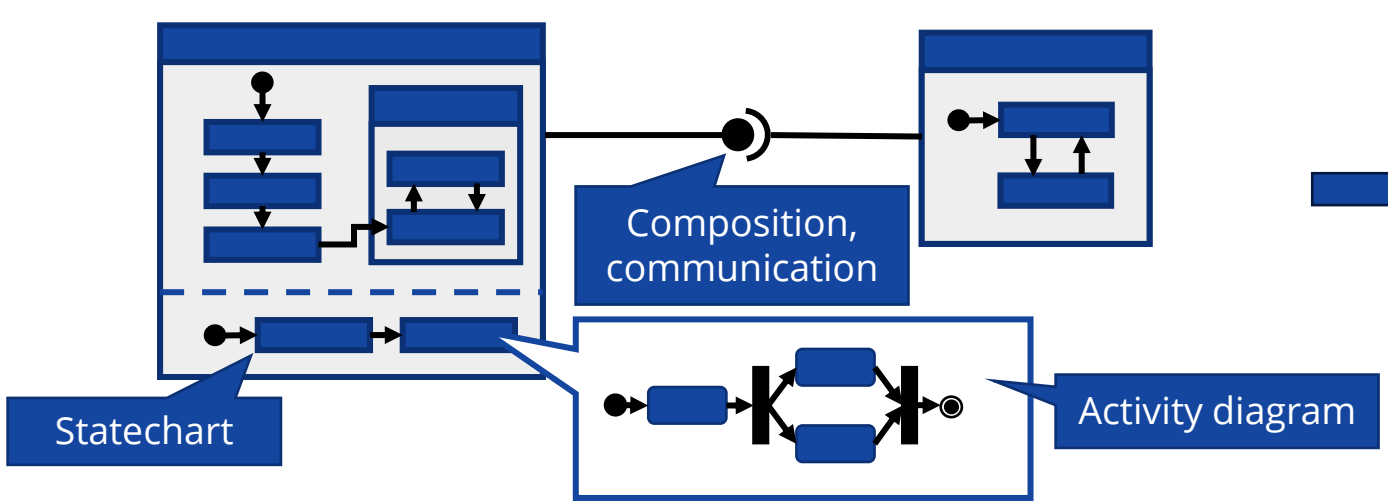
Motivation

- Heterogeneous system models
  - Mix of multiple high-level languages
  - SysMLv2: new OMG systems modeling language



Motivation

- Heterogeneous system models
  - Mix of multiple high-level languages
  - SysMLv2: new OMG systems modeling language



Initial states

I: x = 0

Transition relation

T: (x < 5 ∧ x′ = x + 1) ∨ (x >= 5 ∧ x′ = x)

SMT formulas

→ Transformed to an SMT-based representation

Model-Based Testing of Asynchronously Communicating Distributed Controllers**, Bence Graics et al**, FACS 2023

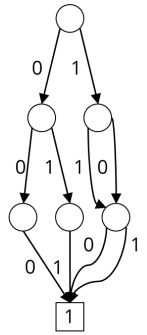
Motivation

- Async behavior
  - Several asynchronously coupled components
    - Communication via e.g. message queues

Motivation

- Async behavior
  - Several asynchronously coupled components
    - Communication via e.g. message queues
  - Decision-diagram-based model checkers
    - (Generalized) Saturation algorithm
    - Proved efficient for Petri Nets

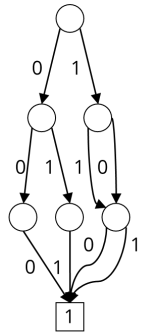
Extensions and generalization of the saturation algorithm in model checking, **Vince Molnár**, PhD Thesis, 2019



Motivation

- Async behavior
  - Several asynchronously coupled components
    - Communication via e.g. message queues
  - Decision-diagram-based model checkers
    - (Generalized) Saturation algorithm
    - Proved efficient for Petri Nets

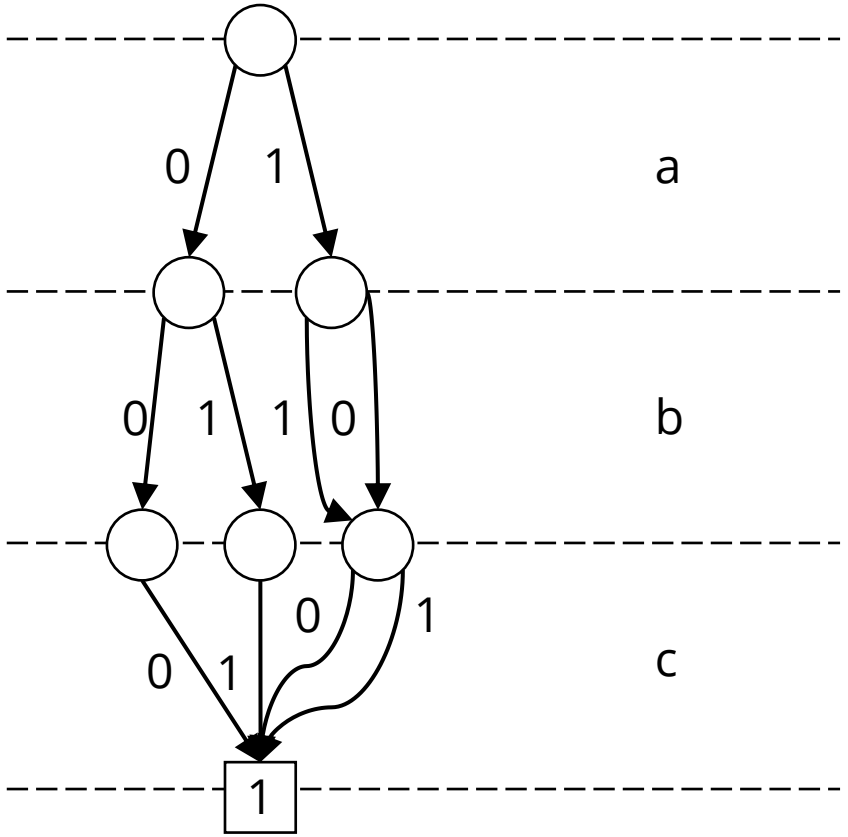
Extensions and generalization of the saturation algorithm in model checking, **Vince Molnár**, PhD Thesis, 2019



Goal: Exploit the advantages of **decision-diagram**-based algorithms (e.g., saturation) on **SMT-based** model representations

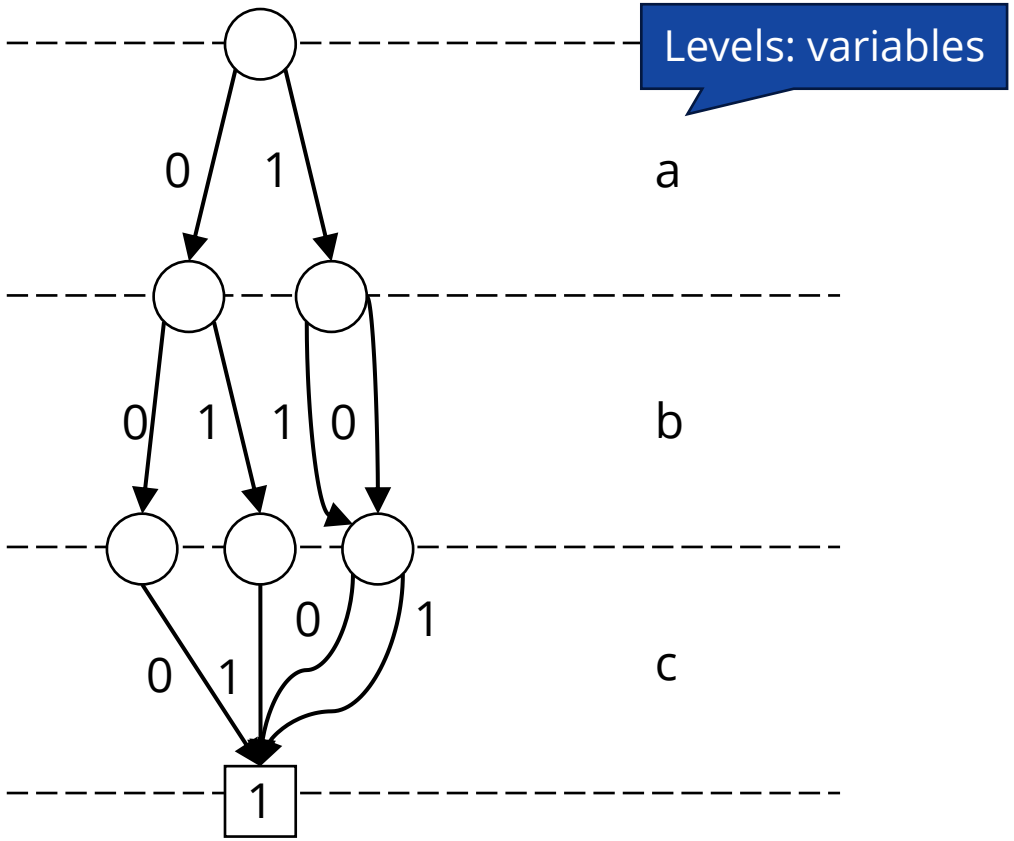
Decision diagrams

- Compact representation of a set of vectors



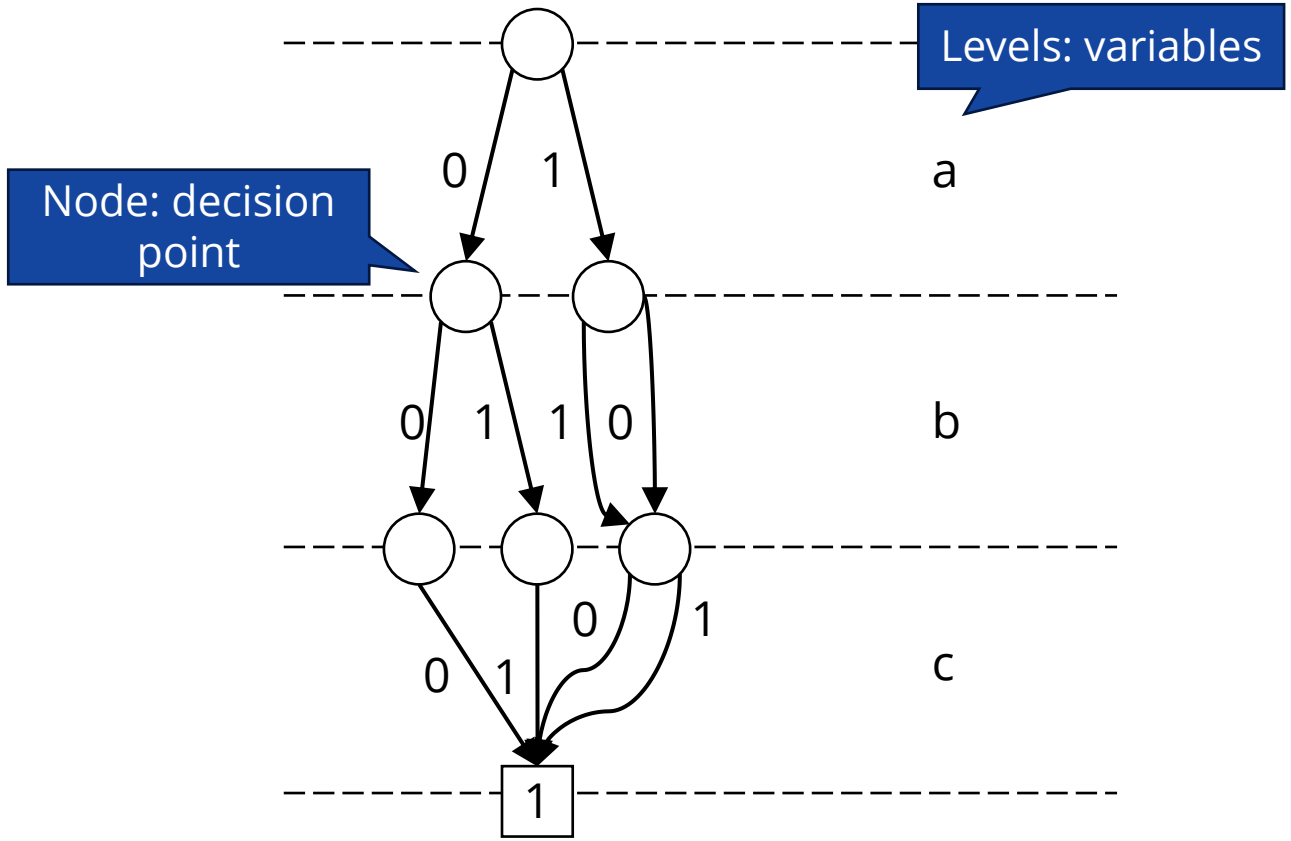
Decision diagrams

- Compact representation of a set of vectors



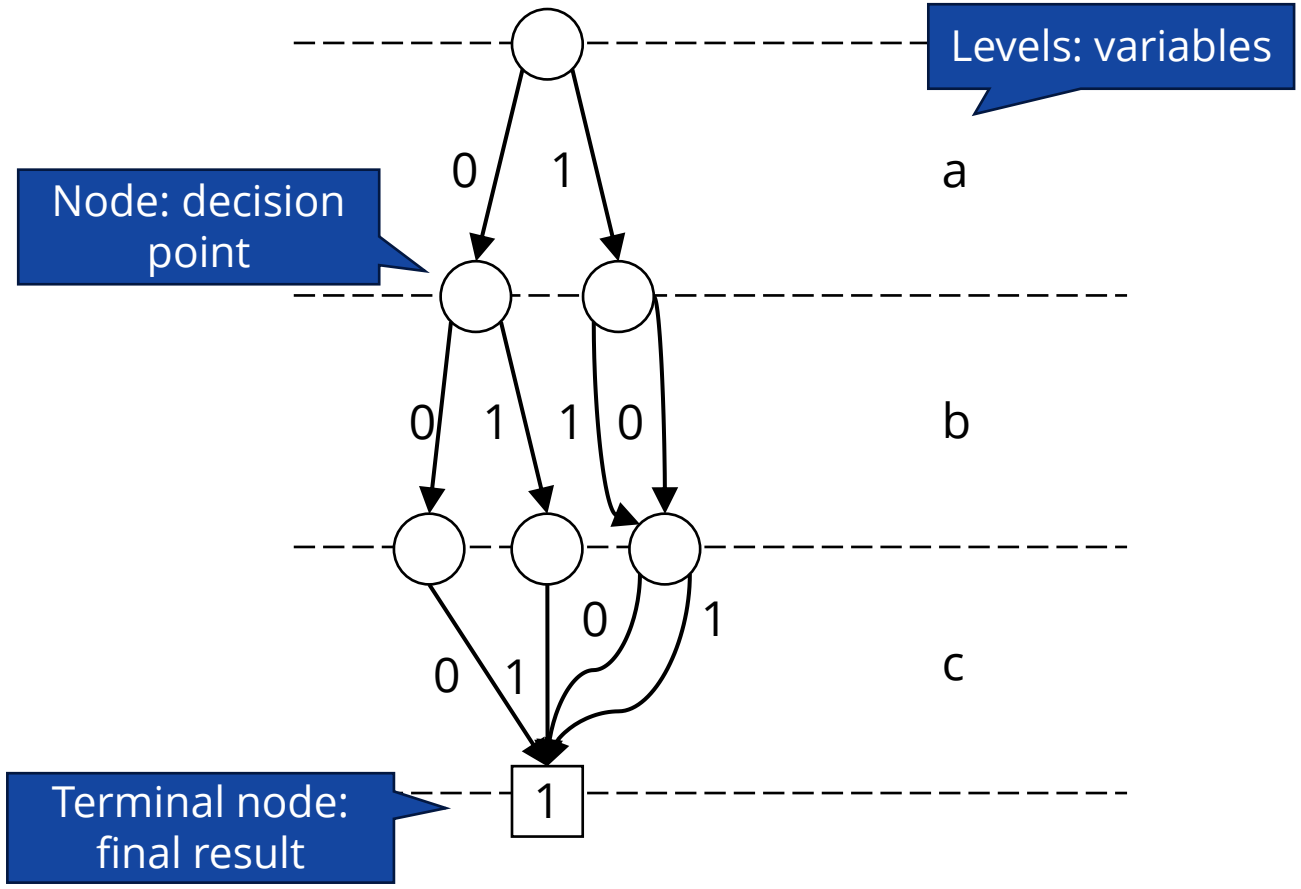
Decision diagrams

- Compact representation of a set of vectors



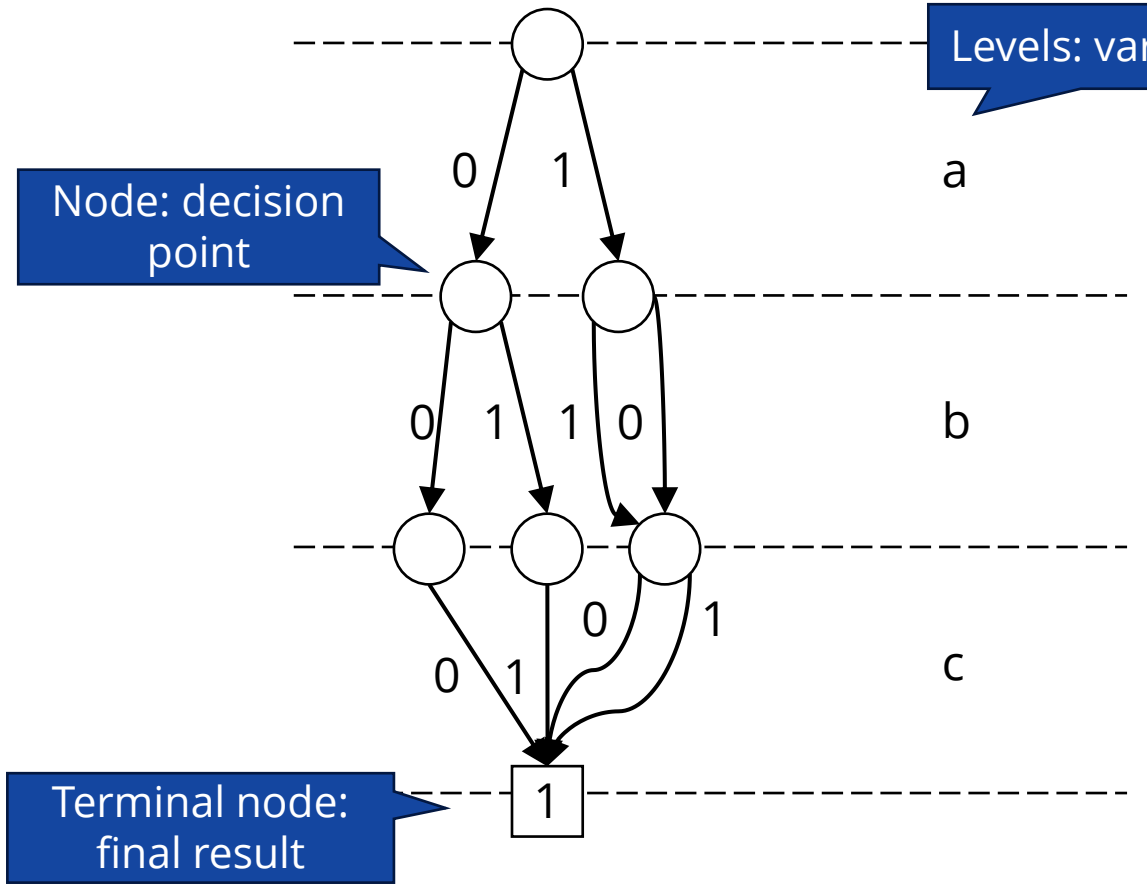
Decision diagrams

- Compact representation of a set of vectors



Decision diagrams

- Compact representation of a set of vectors



Usage for model checking:

- Encode states with k levels
- Encode transitions with 2k levels
- Model step: relational product

  → Calculate fixed point

level i: x
level i+1: x'

Decision diagram from SMT formula?

- One possible way: enumerate all solutions first

Decision diagram from SMT formula?

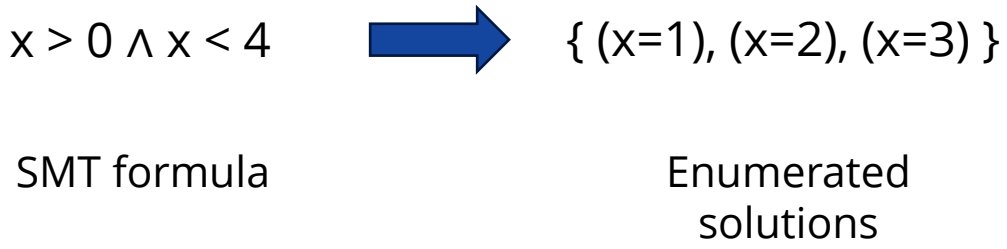
- One possible way: enumerate all solutions first

$x > 0 \wedge x < 4$

SMT formula

Decision diagram from SMT formula?

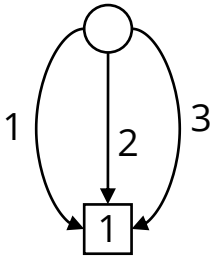
- One possible way: enumerate all solutions first



Decision diagram from SMT formula?

- One possible way: enumerate all solutions first

$x > 0 \wedge x < 4$ → { (x=1), (x=2), (x=3) } →



SMT formula

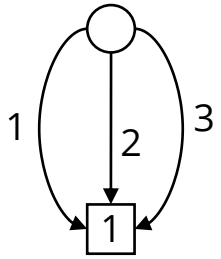
Enumerated solutions

Decision diagram

Decision diagram from SMT formula?

- One possible way: enumerate all solutions first

$x > 0 \wedge x < 4$ → { (x=1), (x=2), (x=3) } →



SMT formula

Enumerated solutions

Decision diagram

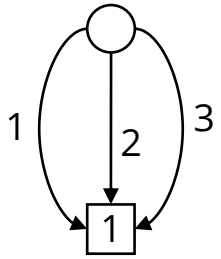
- Problem: formula might have too many solutions → can't enumerate
  - Too many variables
  - Transition relation might be infinitely large

For example, x' = x + 1

Decision diagram from SMT formula?

- One possible way: enumerate all solutions first

$x > 0 \wedge x < 4$ → { (x=1), (x=2), (x=3) } →



SMT formula

Enumerated solutions

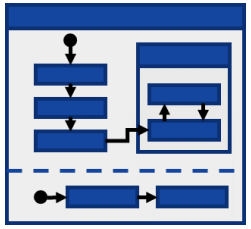
Decision diagram

- Problem: formula might have too many solutions → can't enumerate
  - Too many variables
  - Transition relation might be infinitely large

For example, x' = x + 1

How to represent **general** transition relations given as **SMT formulas**?

Overview

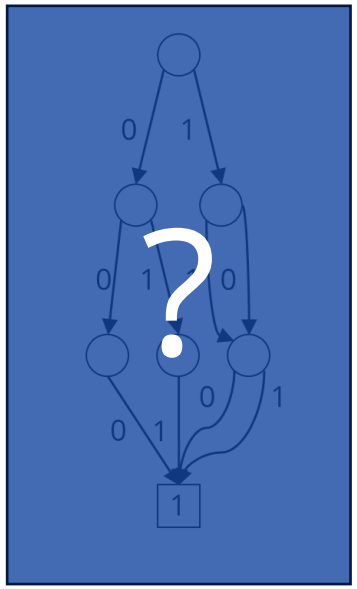


Input model

I: x = 0

T: (x < 5 ∧ x′ = x + 1)
∨ (x >= 5 ∧ x′ = x)

SMT formulas

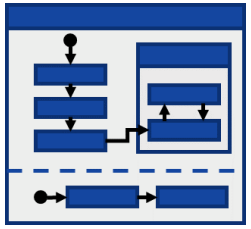


Decision diagrams

Symbolic model checker

✓ / X

Overview

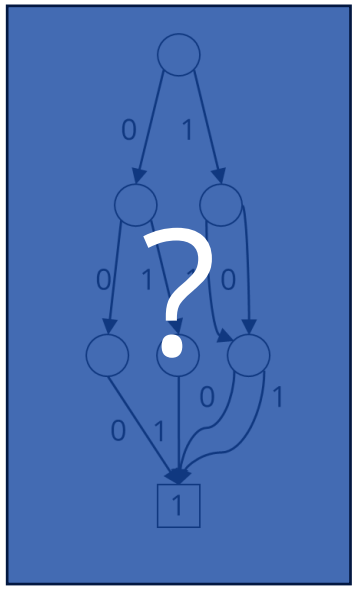


Input model

I: x = 0

T: (x < 5 ∧ x′ = x + 1)
∨ (x >= 5 ∧ x′ = x)

SMT formulas



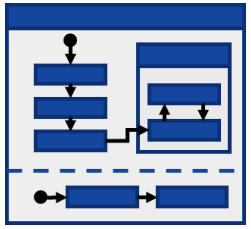
Decision diagrams

Symbolic model checker

✓ / X

Goal: Exploit the advantages of **decision-diagram**-based algorithms (e.g., saturation) on **SMT-based model** representations

Overview

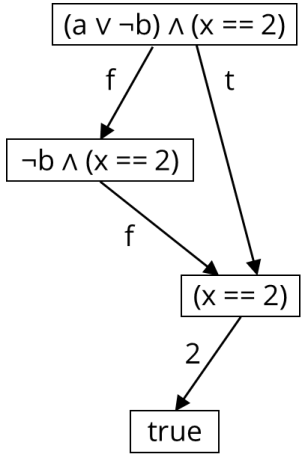


Input model

I: x = 0

T: (x < 5 ∧ x′ = x + 1)
∨ (x >= 5 ∧ x′ = x)

SMT formulas



Substitution diagrams

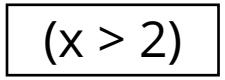
Symbolic model checker

✓ / X

Goal: Exploit the advantages of **decision-diagram**-based algorithms (e.g., saturation) on **SMT-based model** representations

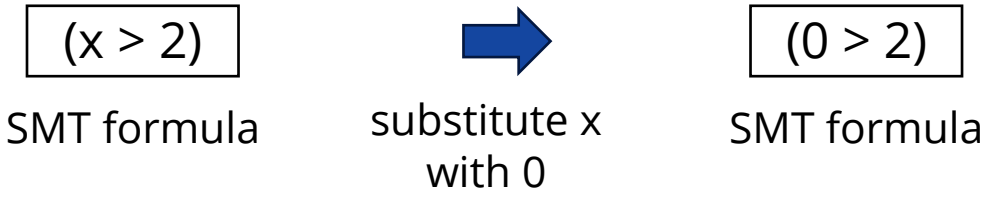
Substitution diagram

Substitution diagram

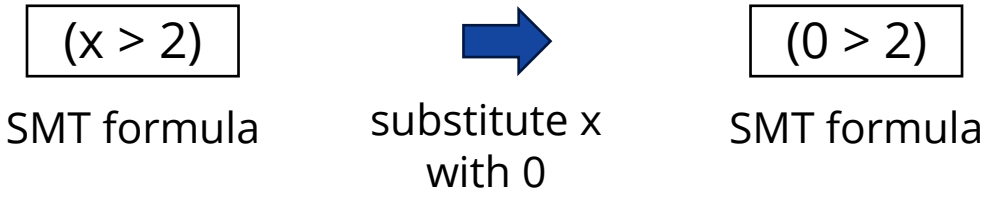


SMT formula

Substitution diagram



Substitution diagram



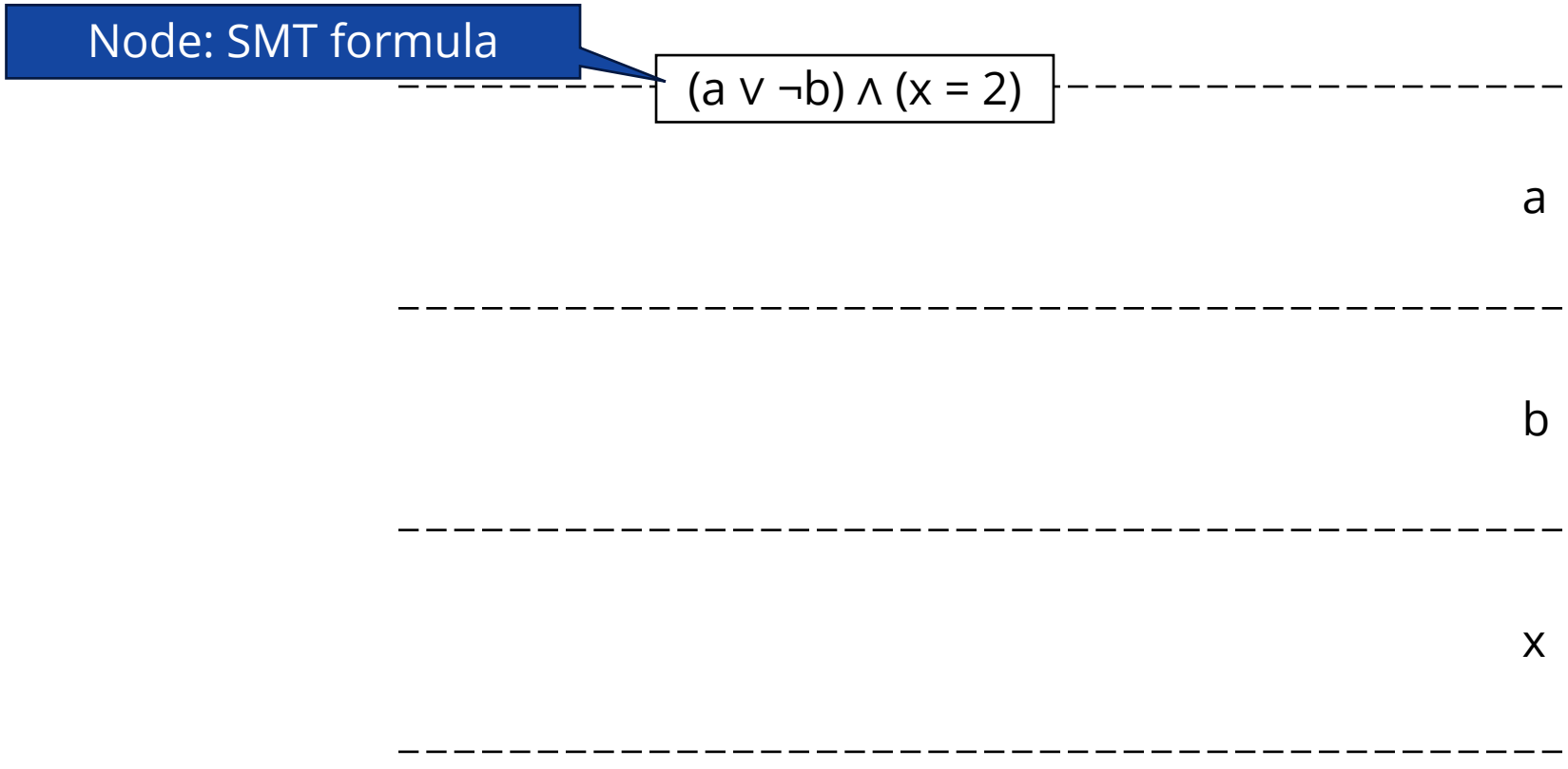
Observation: SMT formulas and the **variable substitution** operation span a structure that is similar to decision diagrams

SMT formulas → Nodes

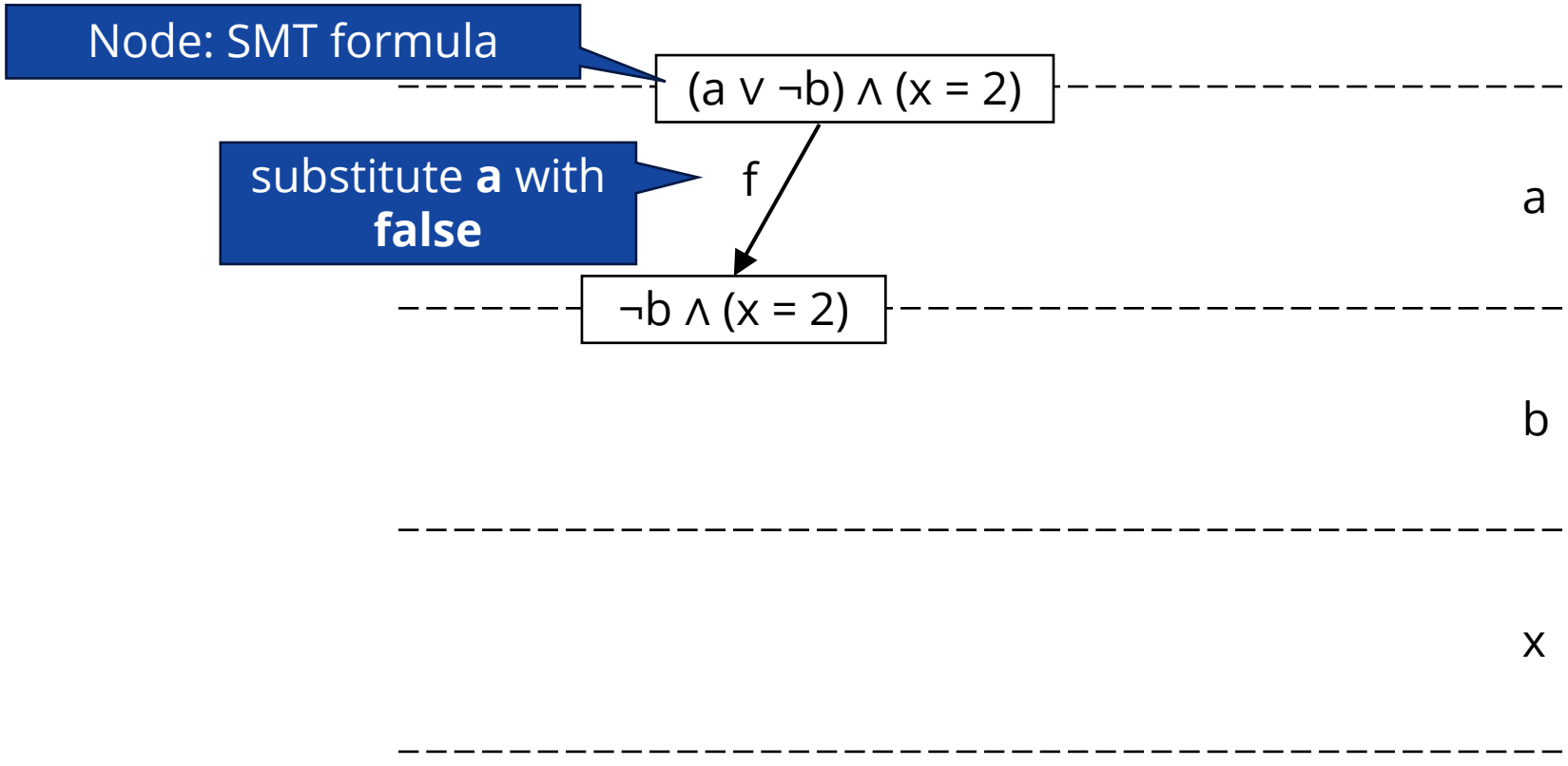
Variable substitution → Edges

Controlling SAT/SMT solvers with decision diagrams to support abstraction-based model checking
Almási Nóra, BME VIK TDK 2020

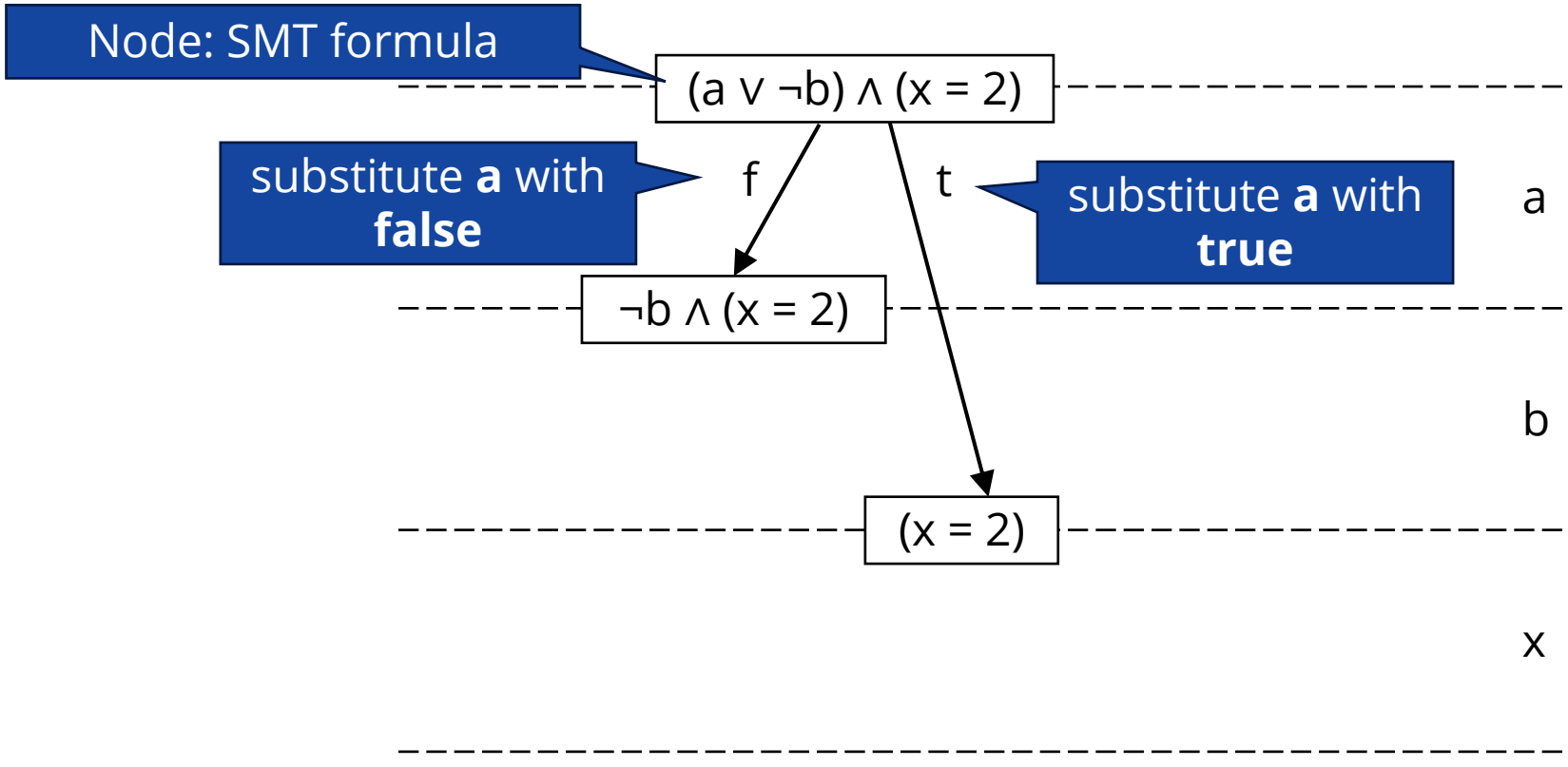
Substitution diagram



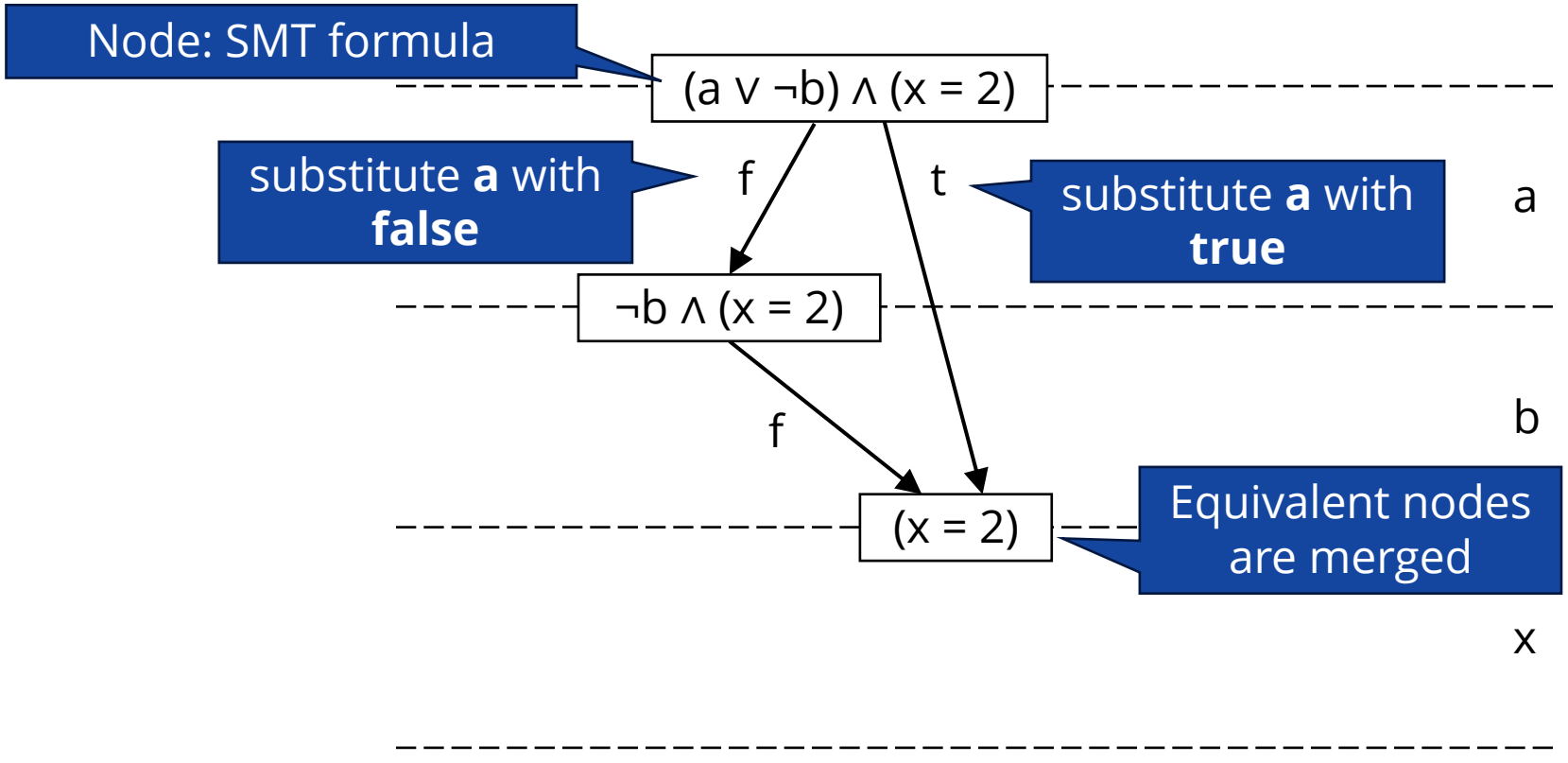
Substitution diagram



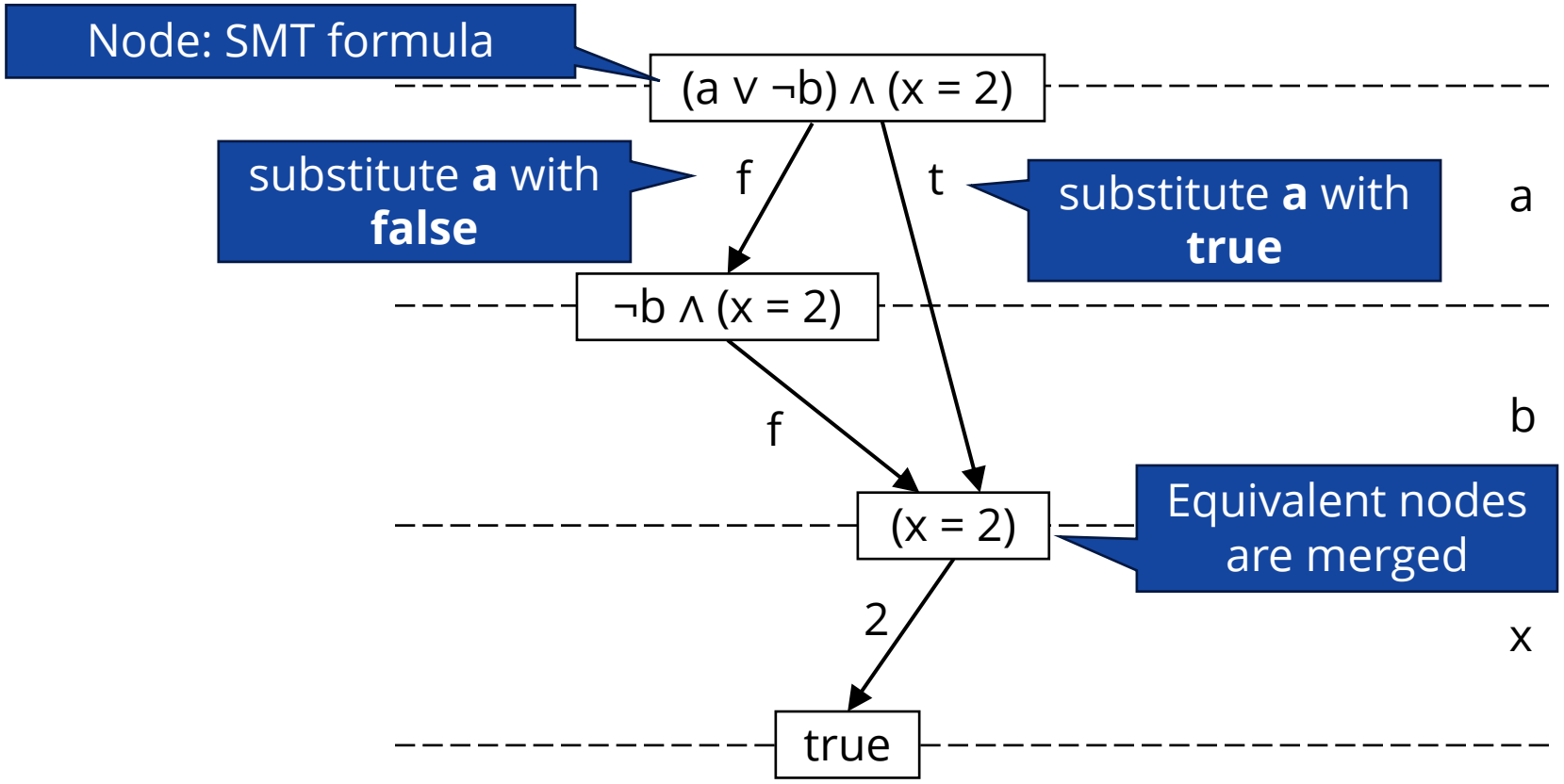
Substitution diagram



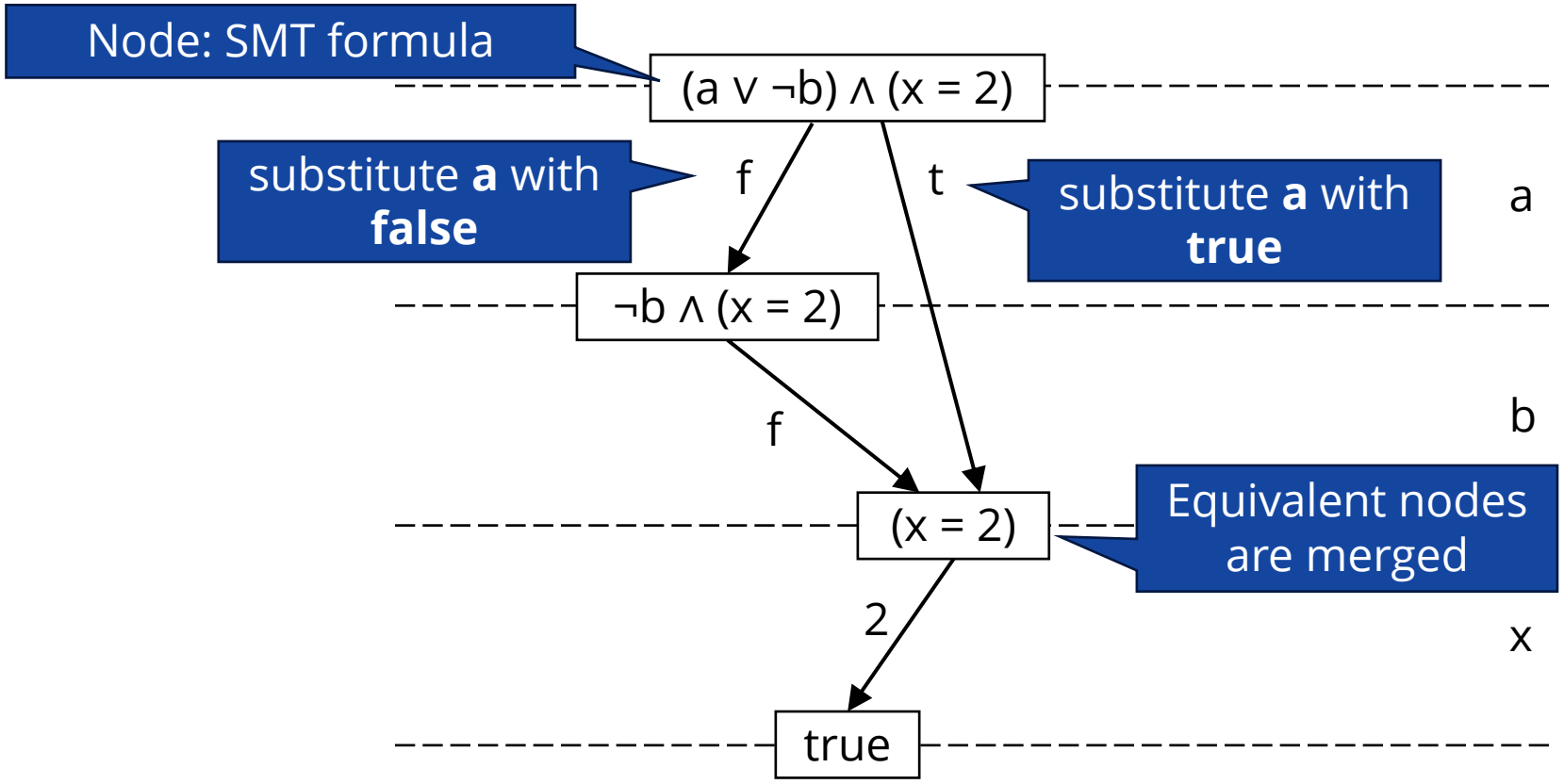
Substitution diagram



Substitution diagram



Substitution diagram



Lazy evaluation: presence of edges and children evaluated only when queried!

syntactically or with an **SMT-solver**

Model checking with substitution diagrams

Model checking with substitution diagrams

Initial states

$I: (x = 0) \wedge (y = 1)$

Model checking with substitution diagrams

Initial states

$I: (x = 0) \wedge (y = 1)$

$T: (x' = x + 1) \wedge (y' = y)$

Transition relation

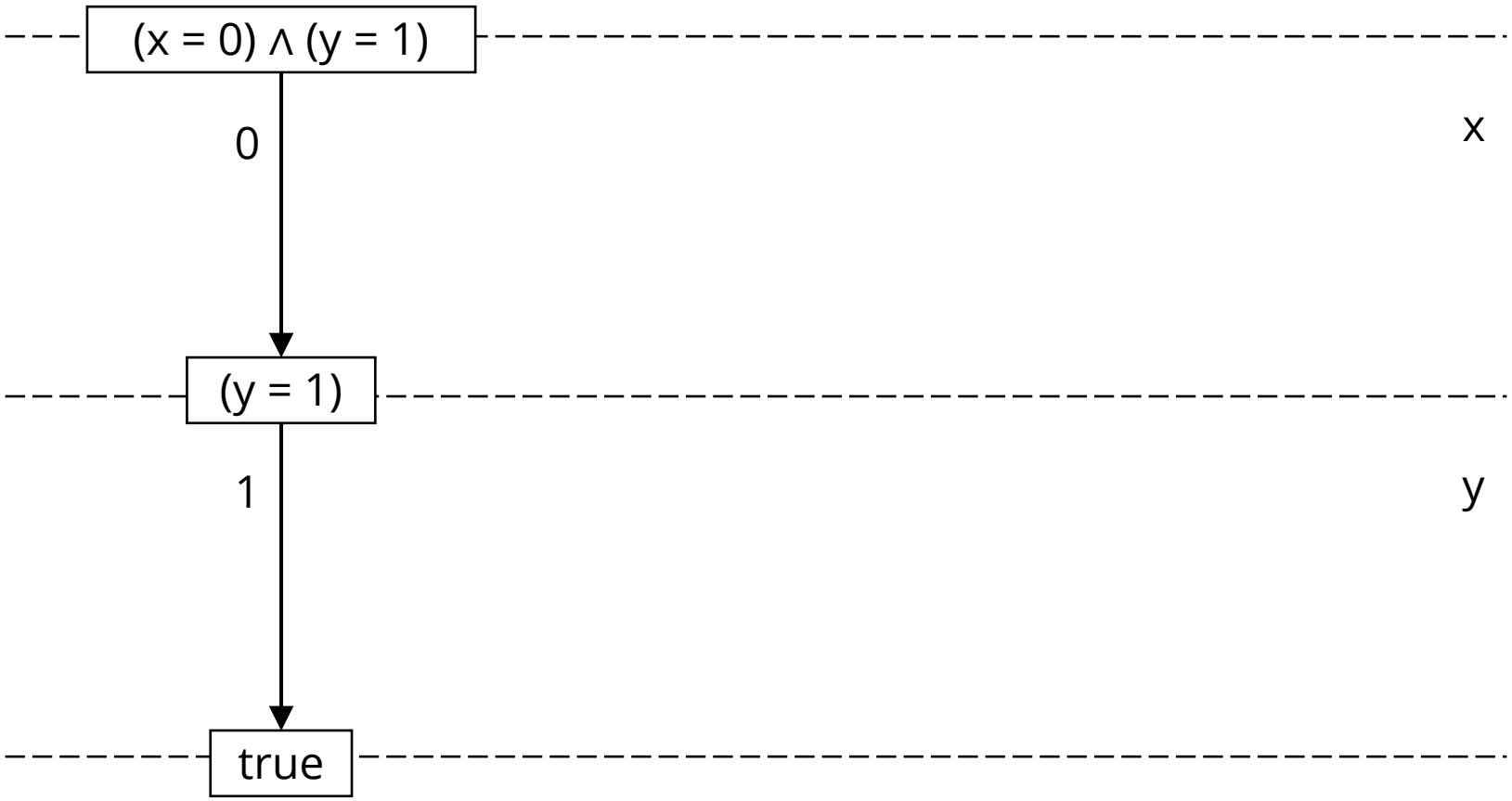
Model checking with substitution diagrams

Initial states

I: $(x = 0) \wedge (y = 1)$

T: $(x' = x + 1) \wedge (y' = y)$

Transition relation



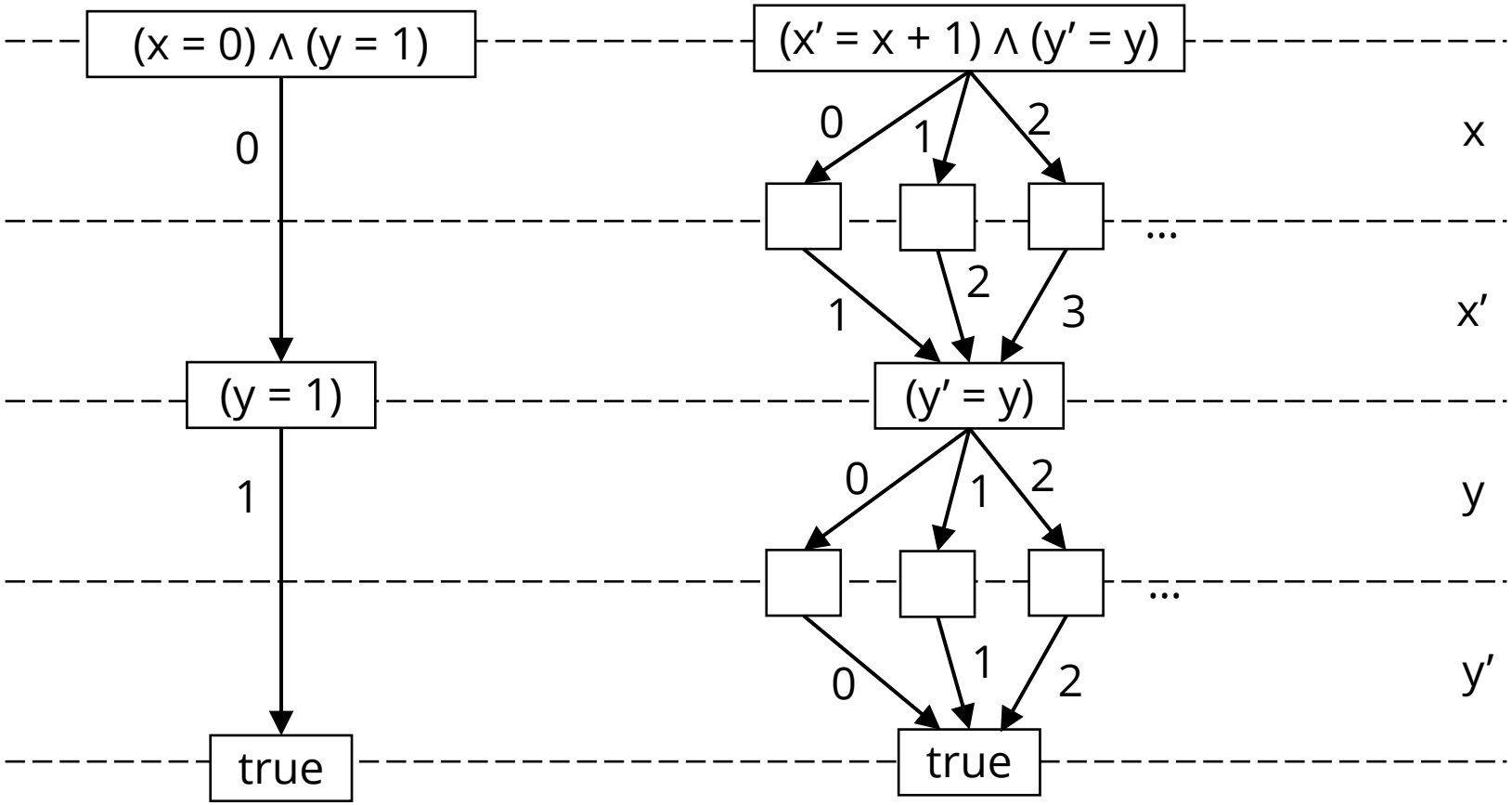
Model checking with substitution diagrams

Initial states

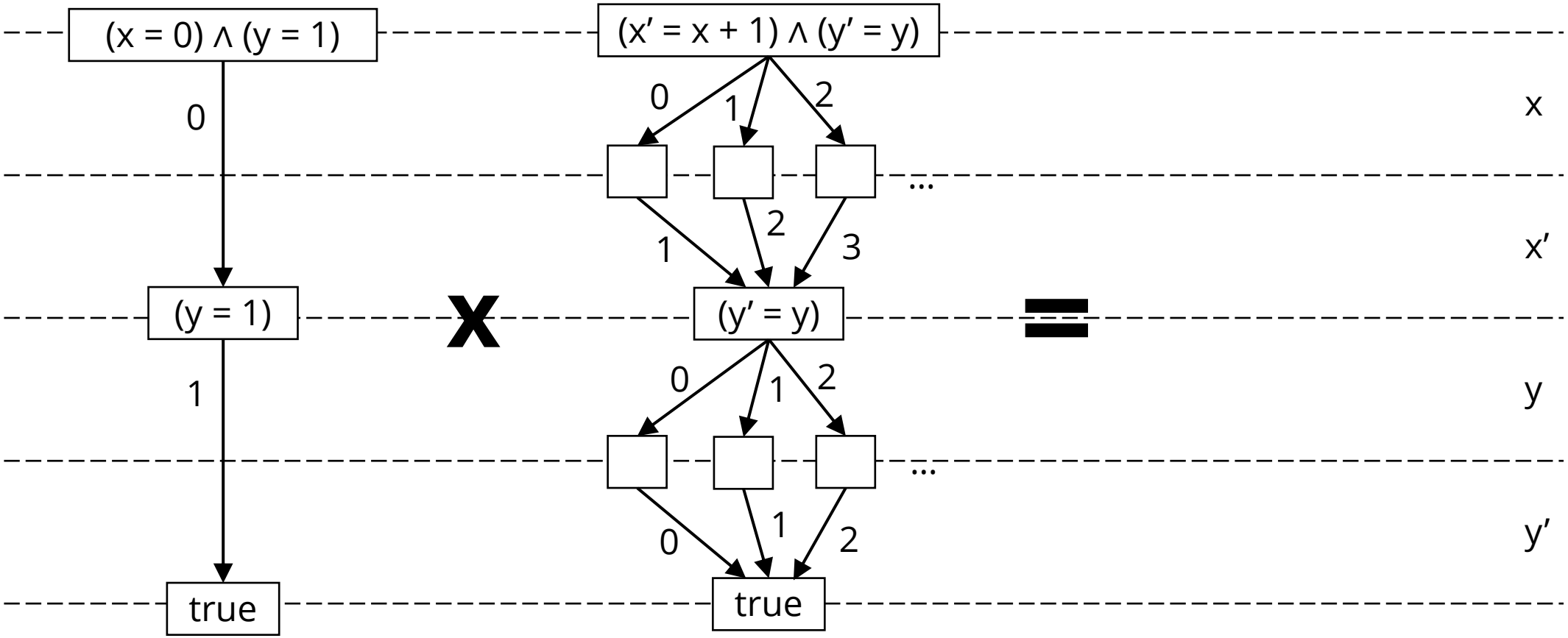
I: (x = 0) ∧ (y = 1)

T: (x′ = x + 1) ∧ (y′ = y)

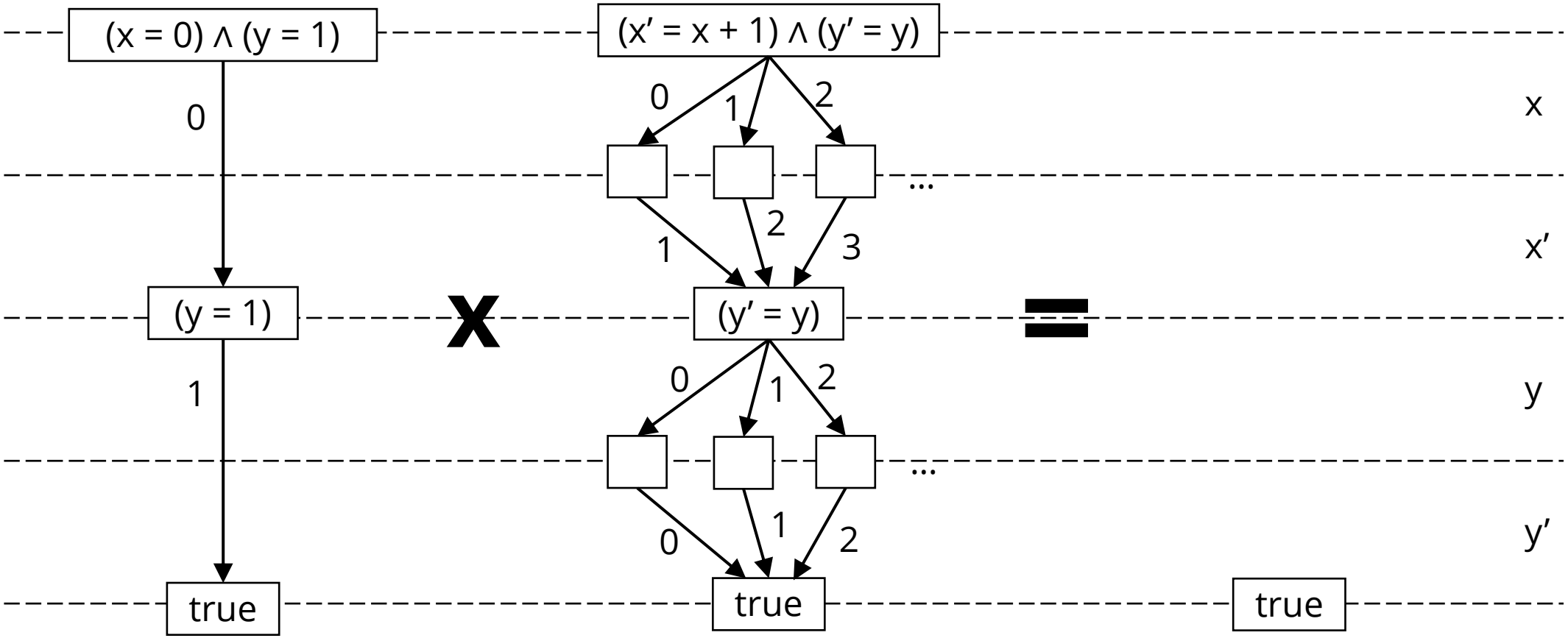
Transition relation



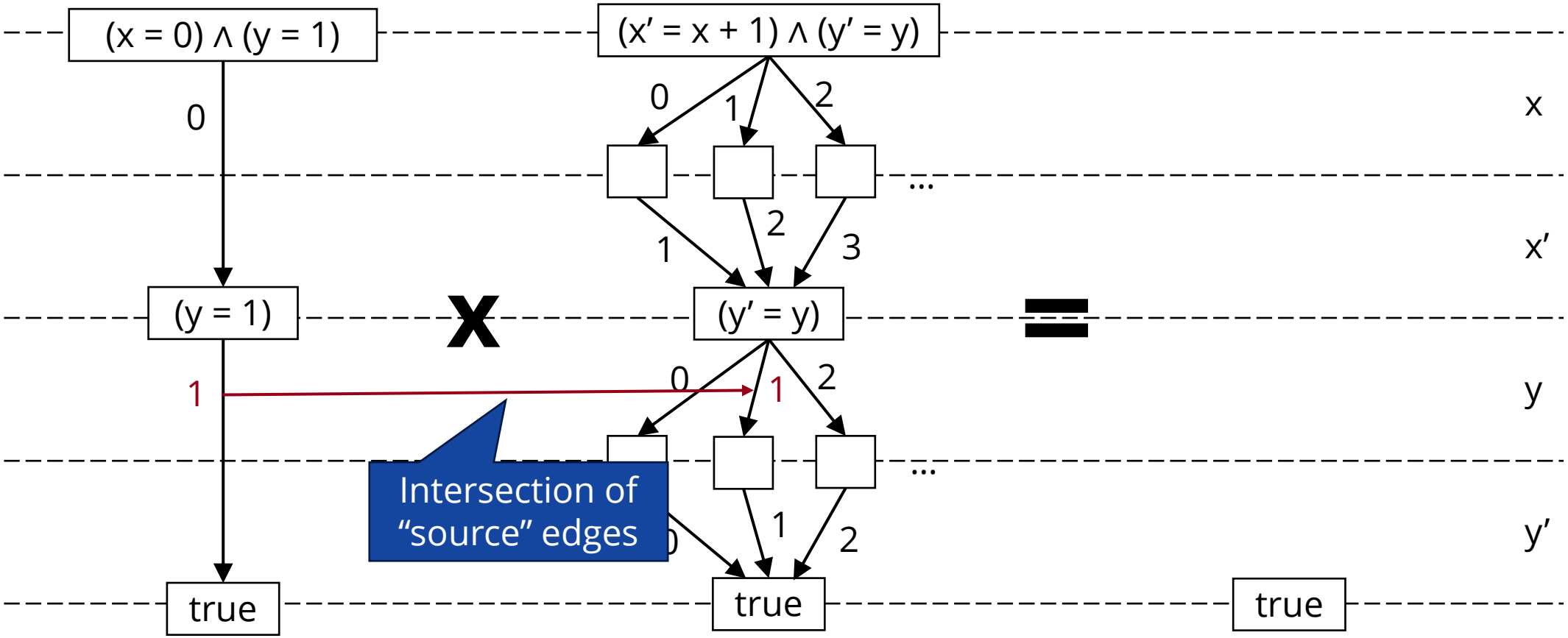
Relational product: model step



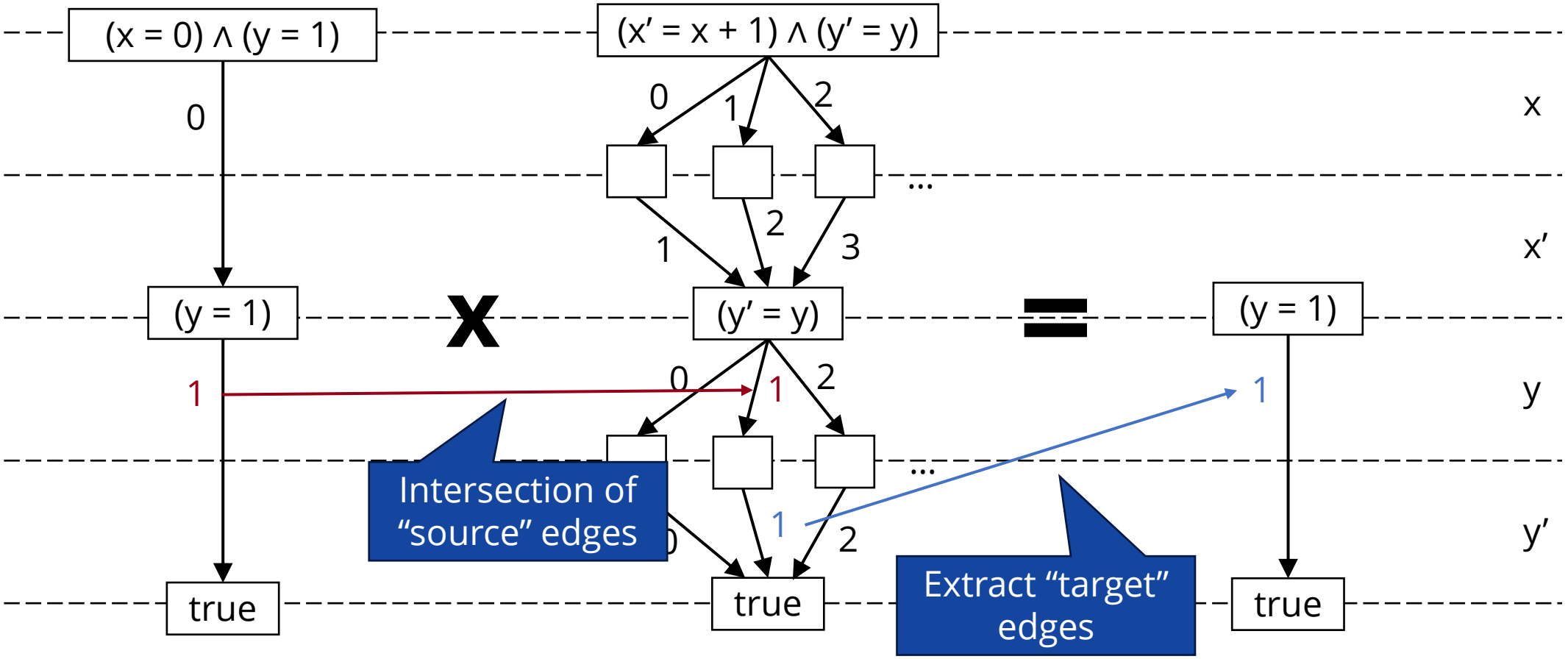
Relational product: model step



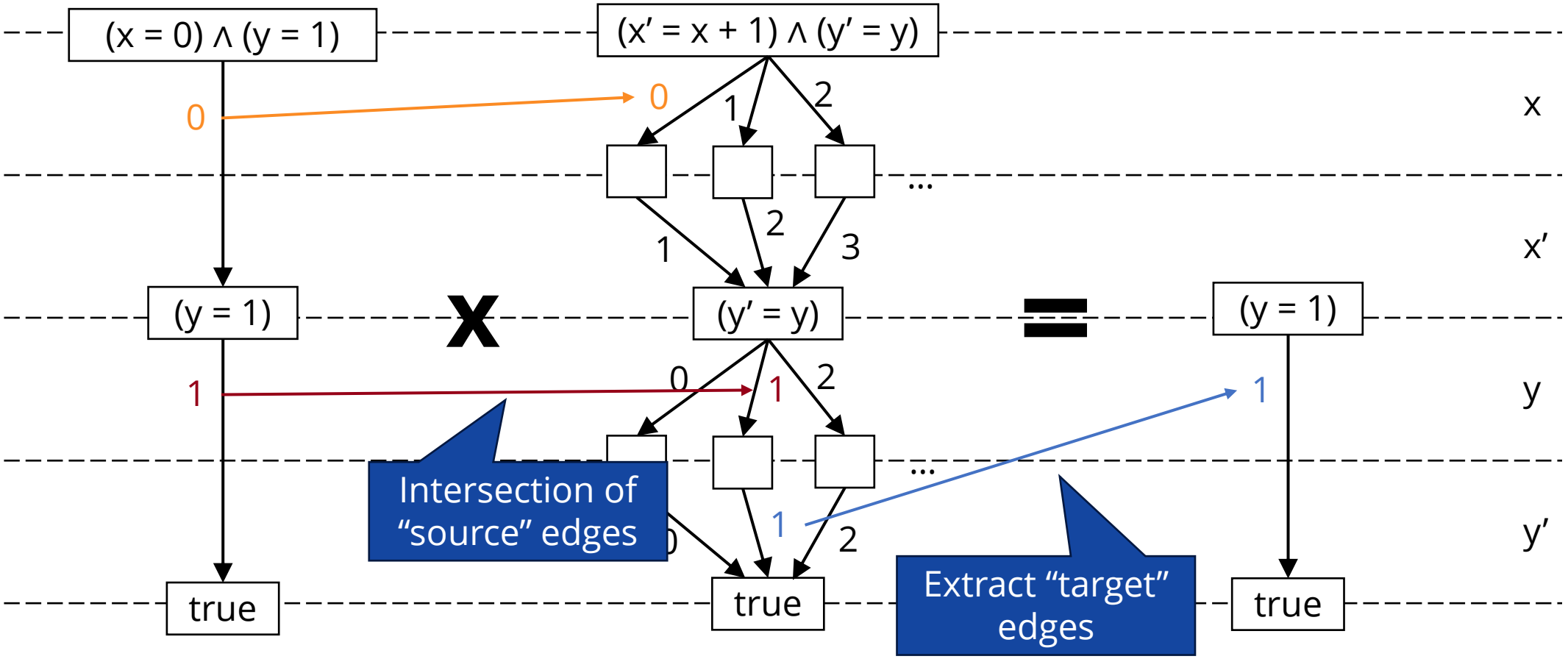
Relational product: model step



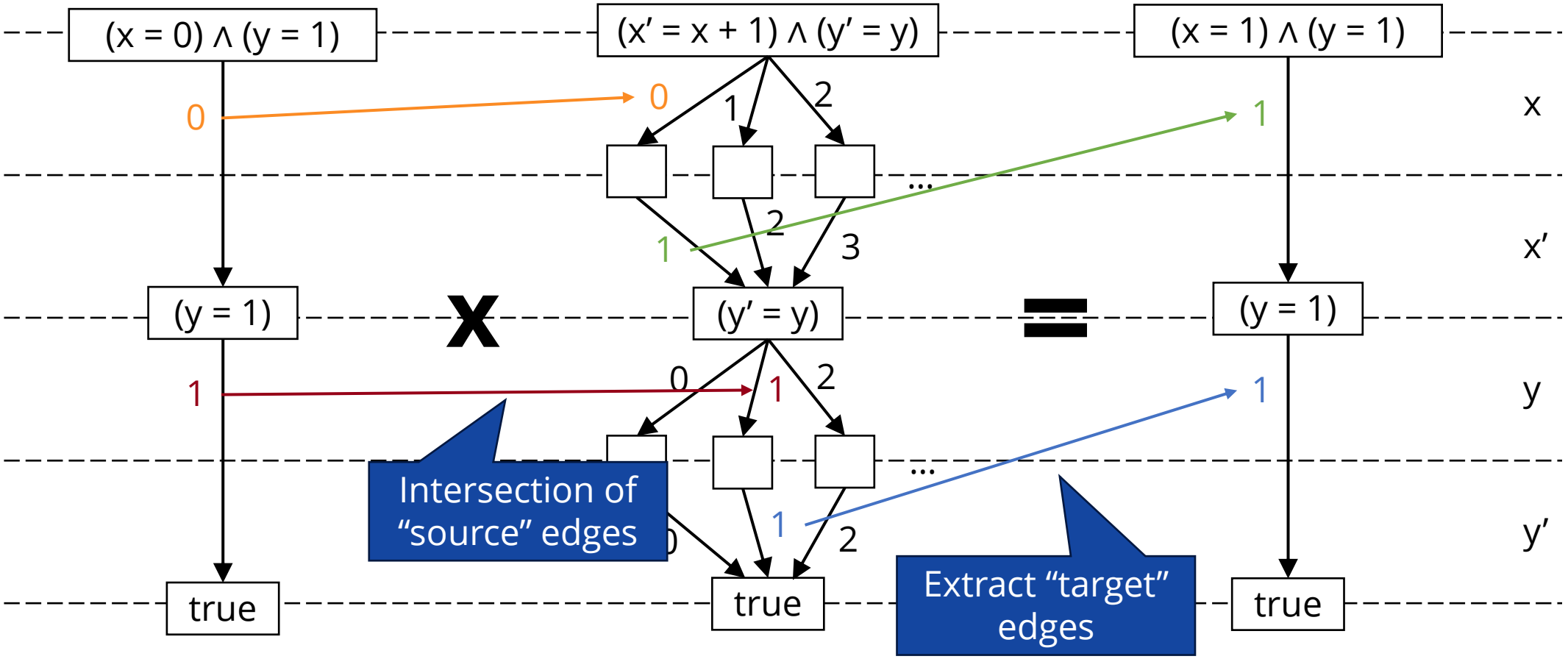
Relational product: model step



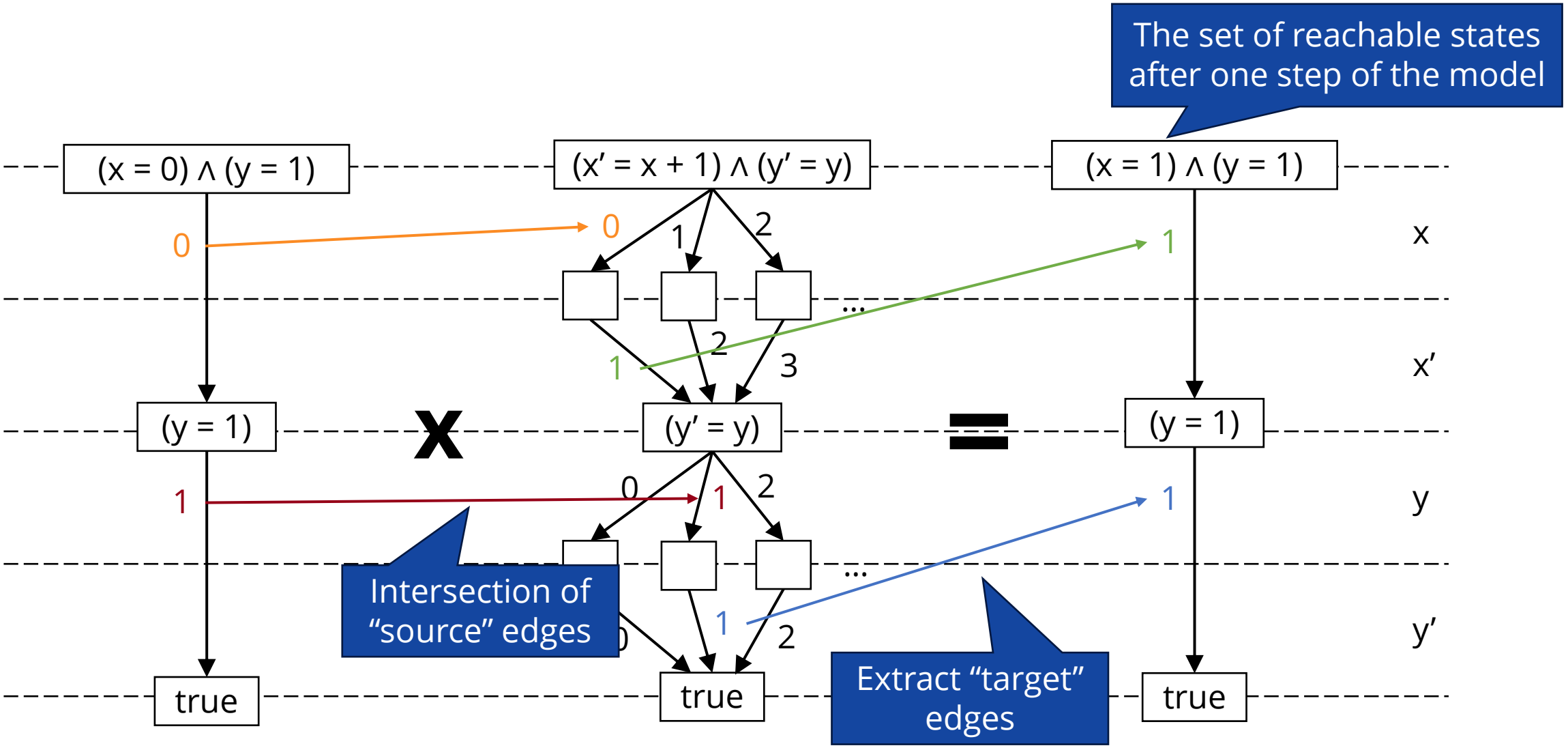
Relational product: model step



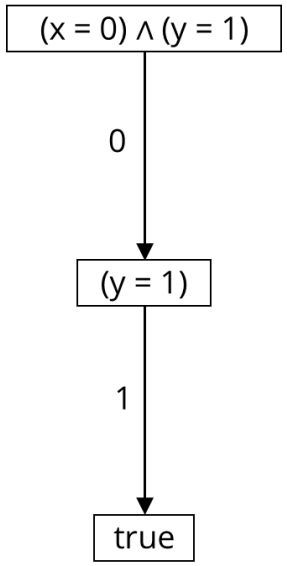
Relational product: model step



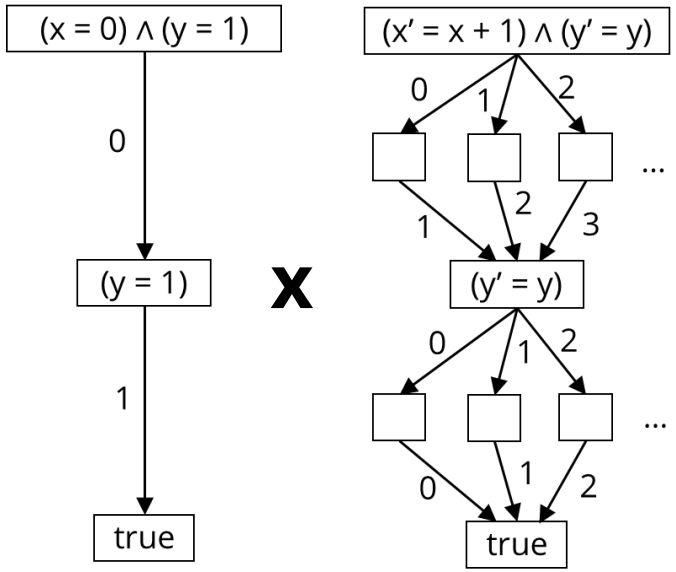
Relational product: model step



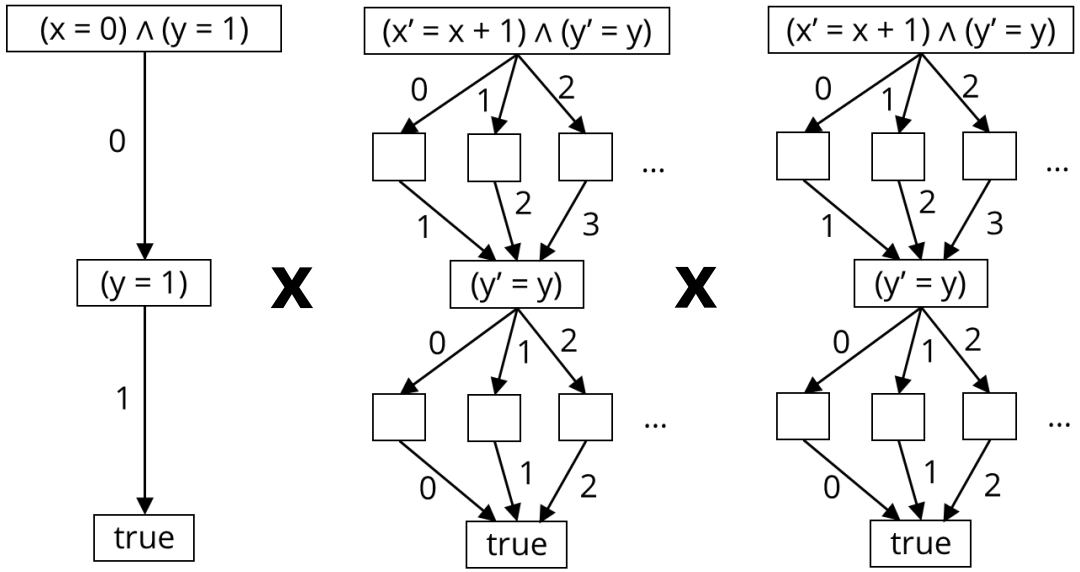
Fixed point calculation



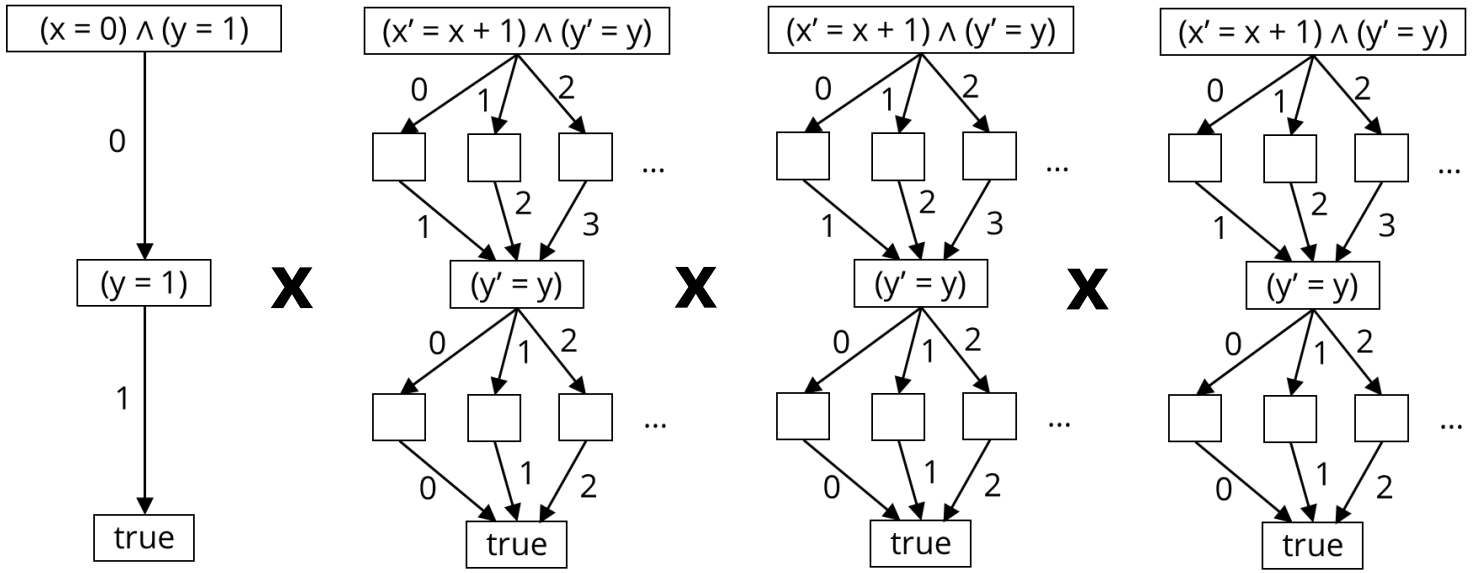
Fixed point calculation



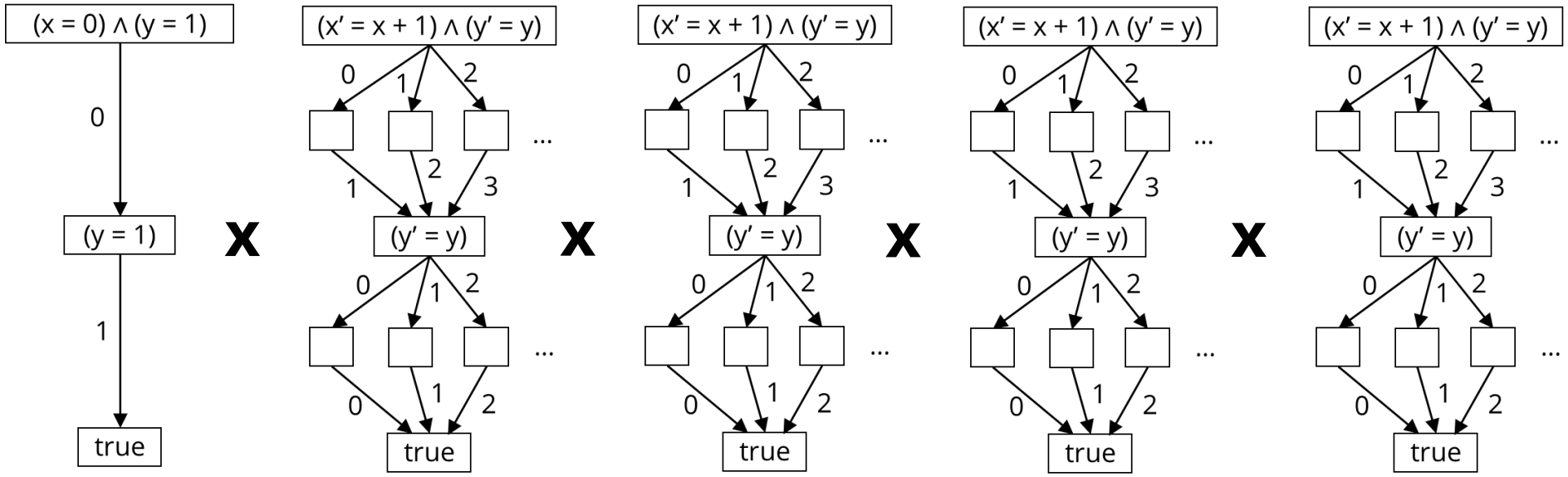
Fixed point calculation



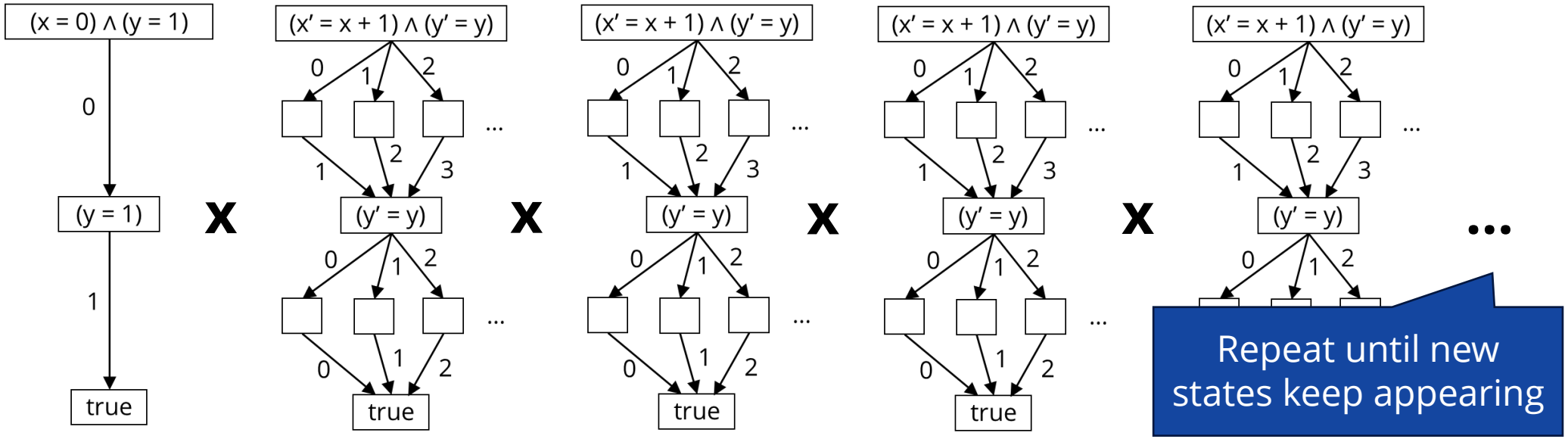
Fixed point calculation



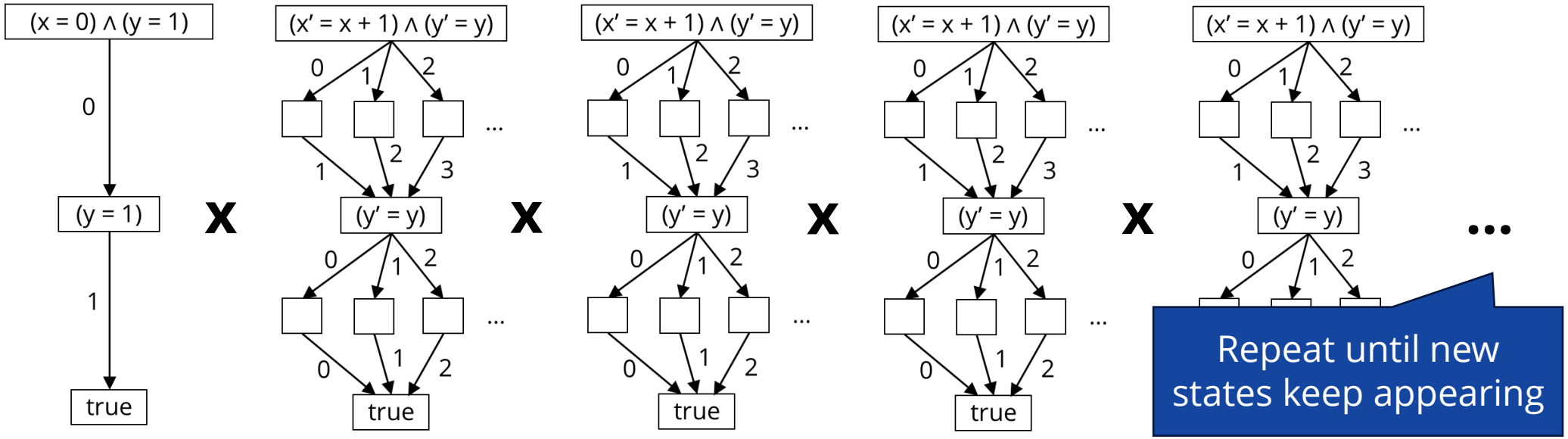
Fixed point calculation



Fixed point calculation



Fixed point calculation



Many possible algorithms: BFS, Saturation

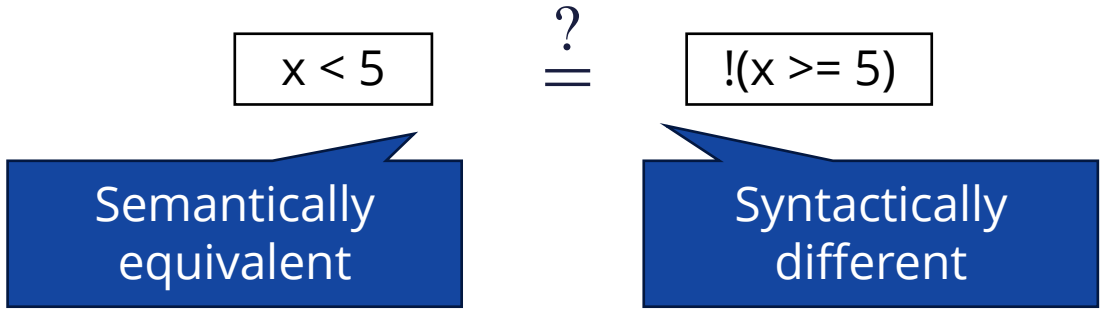
Syntactic vs semantic equivalence

Syntactic vs semantic equivalence

- When should we merge two nodes?

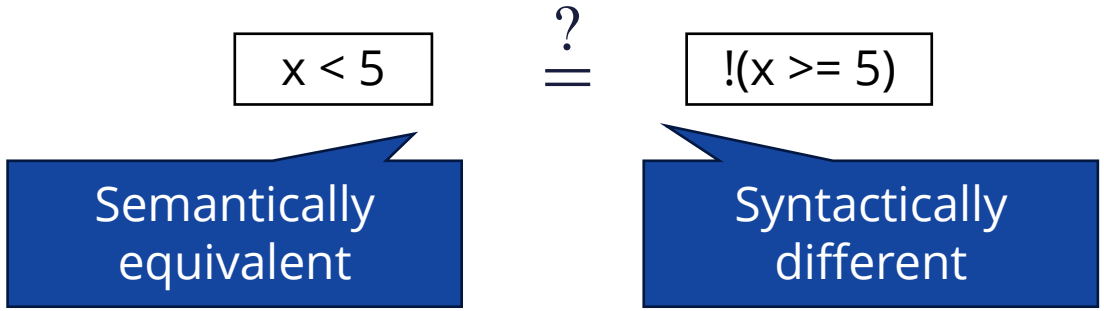
Syntactic vs semantic equivalence

- When should we merge two nodes?



Syntactic vs semantic equivalence

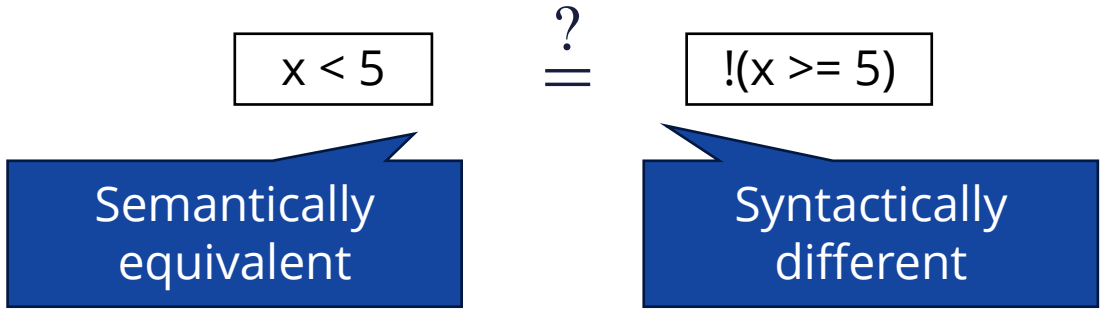
- When should we merge two nodes?



- Decision diagrams: semantic equivalence — More precise → smaller diagrams

Syntactic vs semantic equivalence

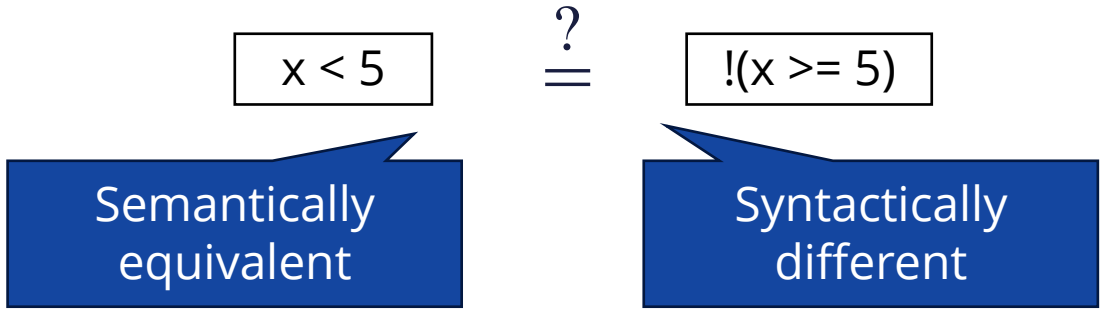
- When should we merge two nodes?



- Decision diagrams: semantic equivalence — More precise → smaller diagrams
- Would be too costly for SMT formulas: solver required

Syntactic vs semantic equivalence

- When should we merge two nodes?



- Decision diagrams: semantic equivalence — More precise → smaller diagrams
- Would be too costly for SMT formulas: solver required
- Compromise: syntactic equivalence + normal form transformation

Syntactic vs semantic equivalence

- Compromise: syntactic equivalence + normal form transformation

Syntactic vs semantic equivalence

- Compromise: syntactic equivalence + normal form transformation
  - Replaces appearances of substituted variable with a constant

Syntactic vs semantic equivalence

- Compromise: syntactic equivalence + normal form transformation
  - Replaces appearances of substituted variable with a constant
  - Removes unnecessary operands

Syntactic vs semantic equivalence

- Compromise: syntactic equivalence + normal form transformation
  - Replaces appearances of substituted variable with a constant
  - Removes unnecessary operands

  - Replaces operations expressible using other operations

Syntactic vs semantic equivalence

- Compromise: syntactic equivalence + normal form transformation
  - Replaces appearances of substituted variable with a constant
  - Removes unnecessary operands

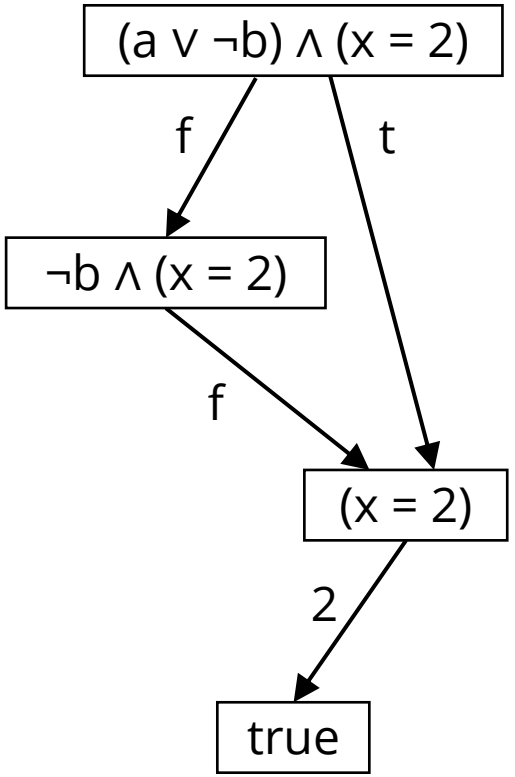
  - Replaces operations expressible using other operations

  - Entirely syntactic, no solver used → lightweight

Summary

New data structure: substitution diagram

- Decision diagram structure from SMT formulas
- Lazy evaluation
- Syntactically equivalent nodes are merged
  - Lightweight normal form transformation
- Implemented in the Theta model checker
  - Reachability analysis
  - github.com/ftsrg/theta



Enables the use of **decision-diagram**-based (e.g., saturation) algorithms on **SMT-based** model representations

Evaluation

- How good is our normal form transformation?

Evaluation

- How good is our normal form transformation?
  - Compare node count of decision diagrams vs substitution diagrams

Evaluation

- How good is our normal form transformation?
  - Compare node count of decision diagrams vs substitution diagrams

  → 10000 randomly generated transition systems

Evaluation

- How good is our normal form transformation?
  - Compare node count of decision diagrams vs substitution diagrams

  → 10000 randomly generated transition systems
  → 10000 SMT formulas

Evaluation

- How good is our normal form transformation?
  - Compare node count of decision diagrams vs substitution diagrams

→ 10000 randomly generated transition systems
→ 10000 SMT formulas
→ **3789 satisfiable formulas**

Evaluation

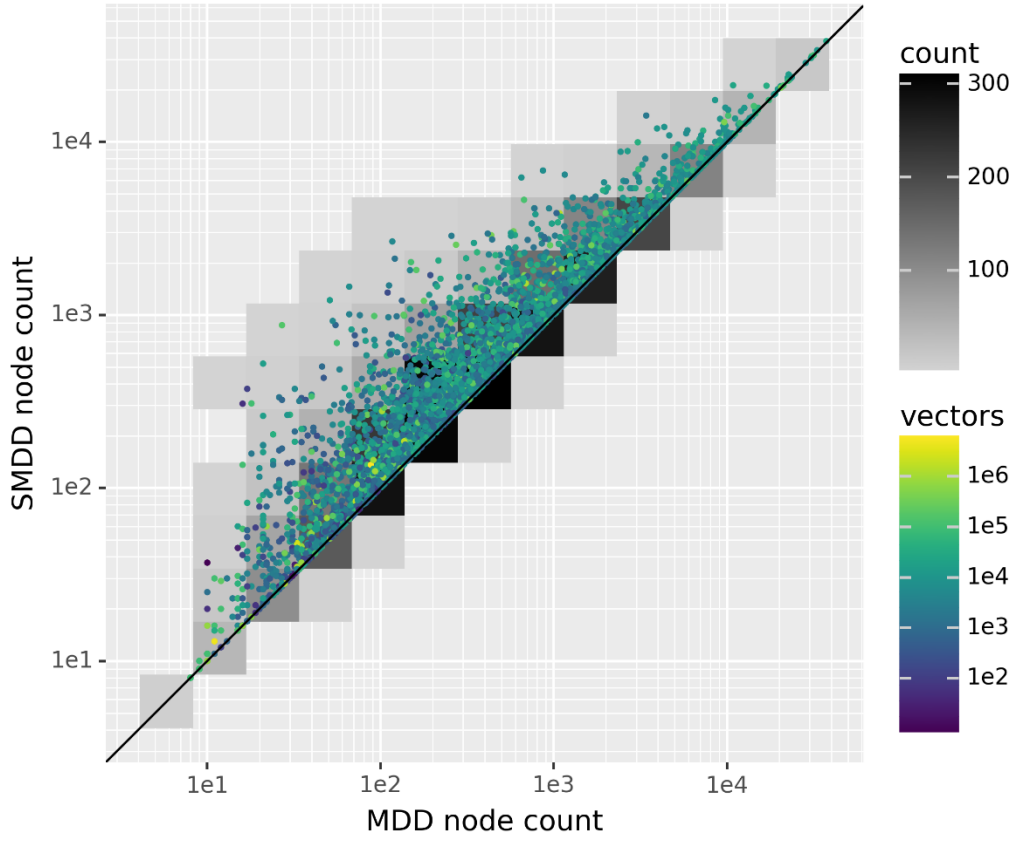
- How good is our normal form transformation?
  - Compare node count of decision diagrams vs substitution diagrams

→ 10000 randomly generated transition systems
→ 10000 SMT formulas
→ **3789 satisfiable formulas**

→ Build decision and substitution diagrams from these formulas and calculate node counts

Evaluation

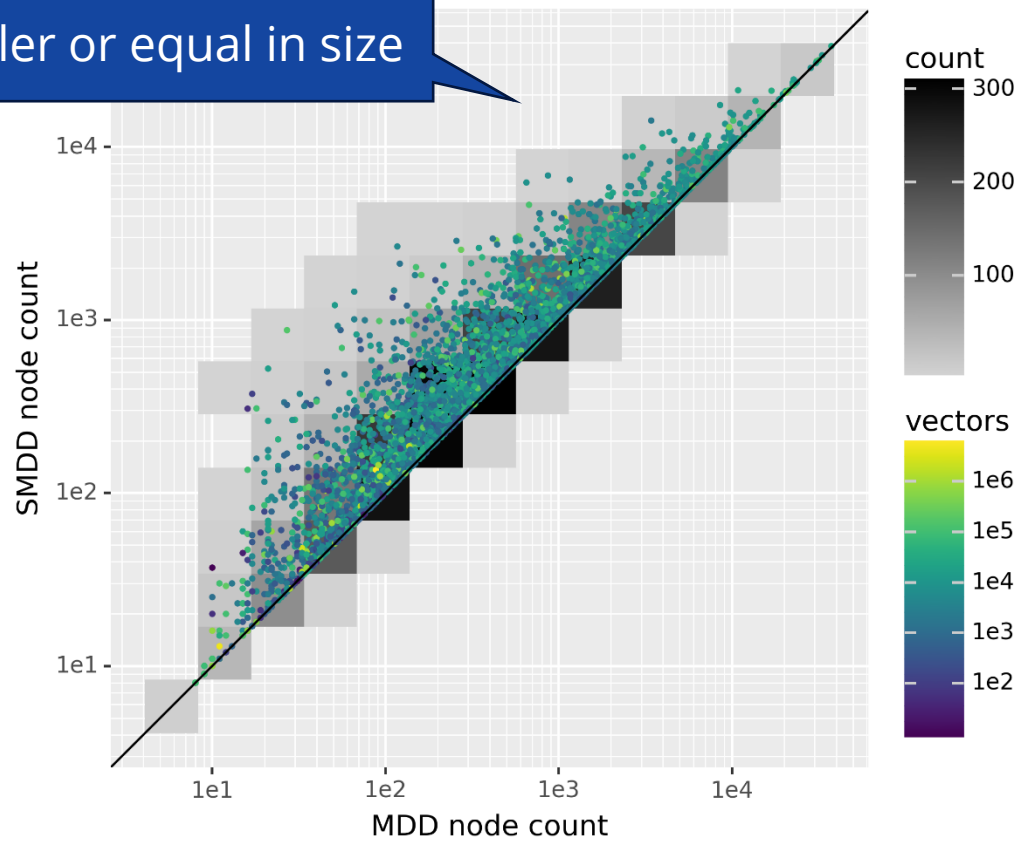
- How good is our normal form transformation?
  - Node count comparison on **3789** randomly generated SMT formulas



Evaluation

- How good is our normal form transformation?
  - Node count comparison on **3789** randomly generated SMT formulas

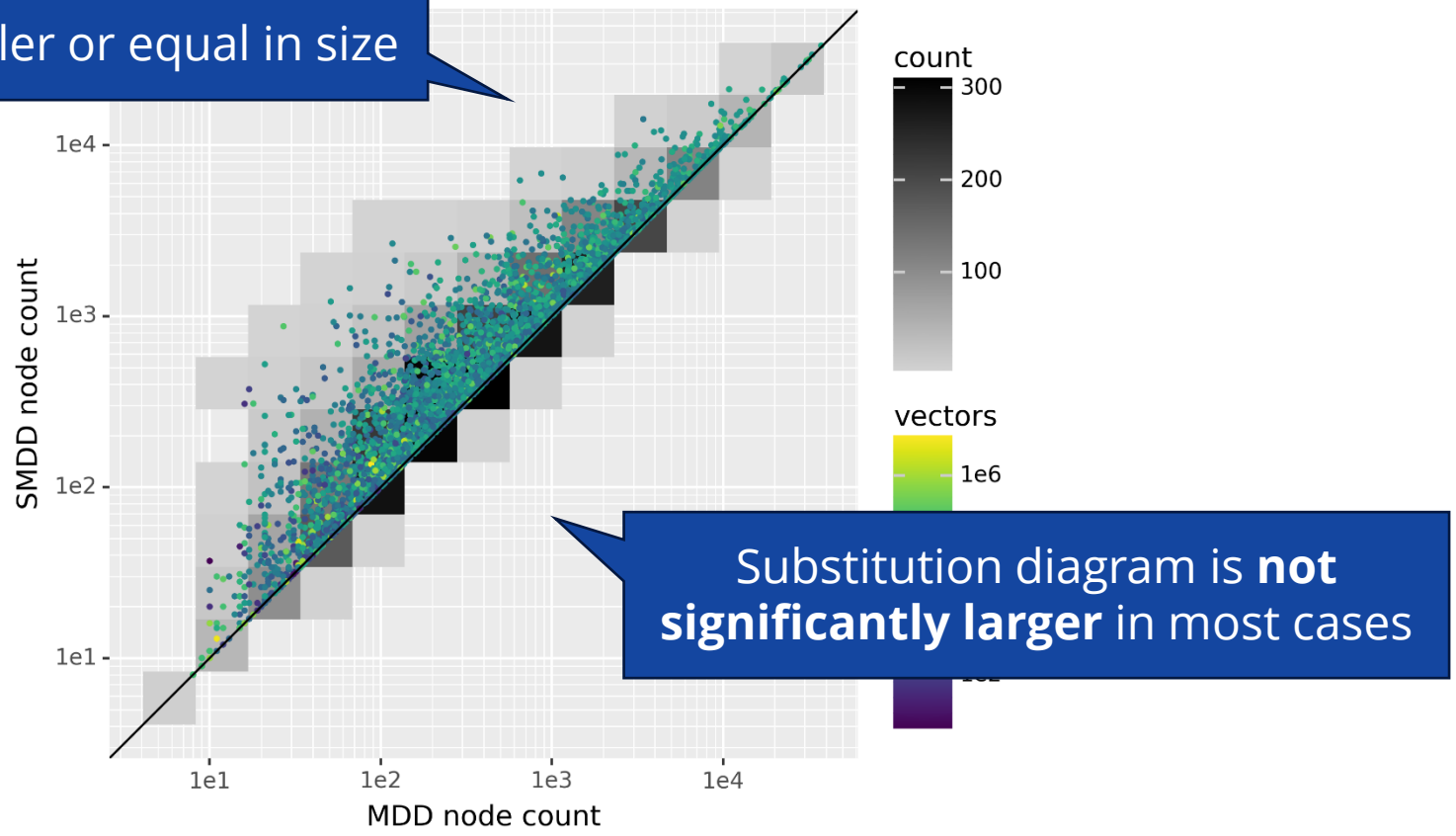
MDD is always smaller or equal in size



Evaluation

- How good is our normal form transformation?
  - Node count comparison on **3789** randomly generated SMT formulas

MDD is always smaller or equal in size



Substitution diagram is **not significantly larger** in most cases