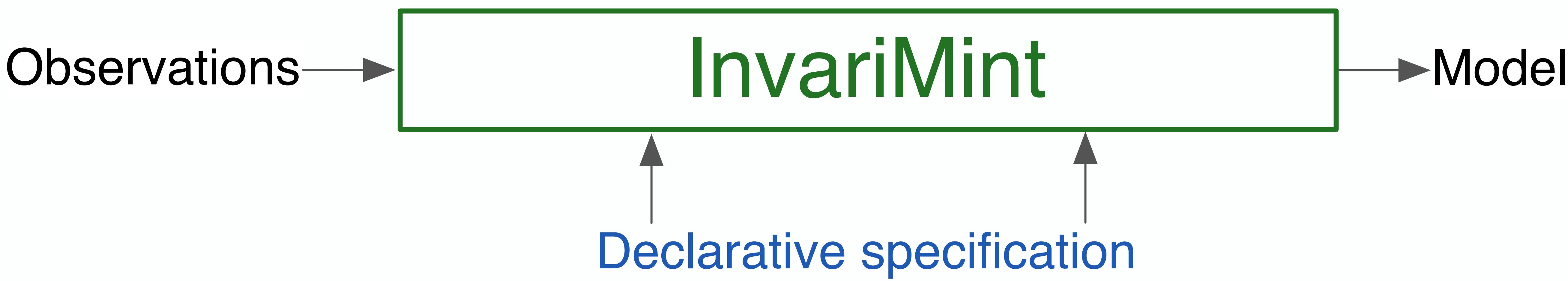
Declarative Specification of FSM-Inference Algorithms



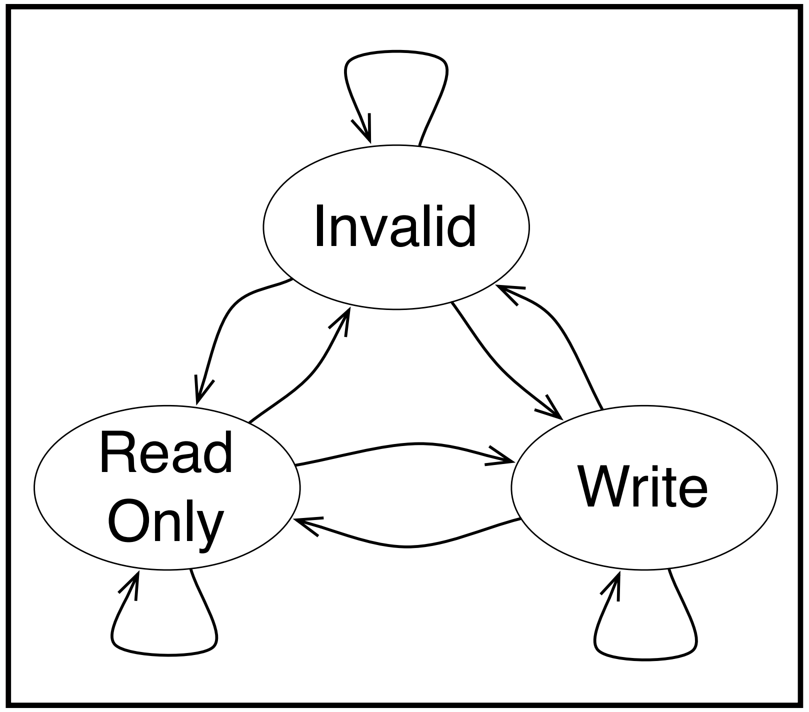
Ivan Beschastnikh
Yuriy Brun
Jenny Abrahamson
Michael D. Ernst
Arvind Krishnamurthy

University of Washington

UMass. Amherst

FSM-inference

FSM-inference algorithm



Observations

FSM Model

FSM-inference

Applications:

- Mental model validation — Cook et al. TSE 1998
- Test case generation — Mariani et al. ICSE 2007
- Anomaly detection — Dallmeier et al. ASE 2009
- Log summarization — Beschastnikh et al. FSE 2011

...

Observations

FSM-inference algorithm

Model

FSM-inference

Applications:

- Mental model validation — Cook et al. TSE 1998
- **Test case generation — Mariani et al. ICSE 2007**
- Anomaly detection — Dallmeier et al. ASE 2009
- Log summarization — Beschastnikh et al. FSE 2011

...

Observations

FSM-inference algorithm

Model

FSM-inference

Applications:

- Mental model validation — Cook et al. TSE 1998
- Test case generation — Mariani et al. ICSE 2007
- **Anomaly detection — Dallmeier et al. ASE 2009**
- Log summarization — Beschastnikh et al. FSE 2011

...

Observations

FSM-inference algorithm

Model

FSM-inference in prior work

Observations

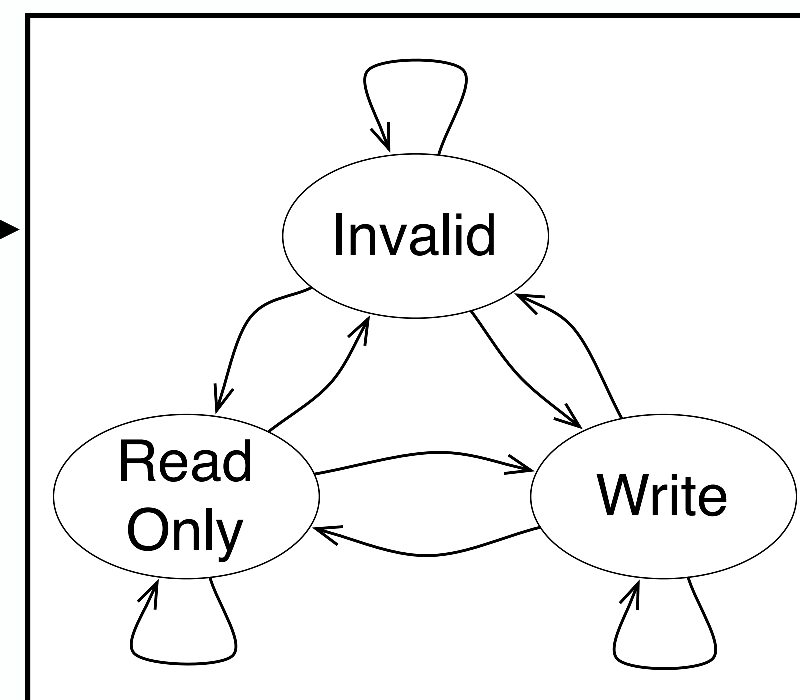
Model inference

- Specification mining
- Process discovery
- Grammar inference

Prior work:

Cook et al. TSE 1998
Mariani et al. ICSE 2007
Dallmeier et al. ASE 2009
Beschastnikh et al. FSE 2011

...



Model

FSM-inference in prior work

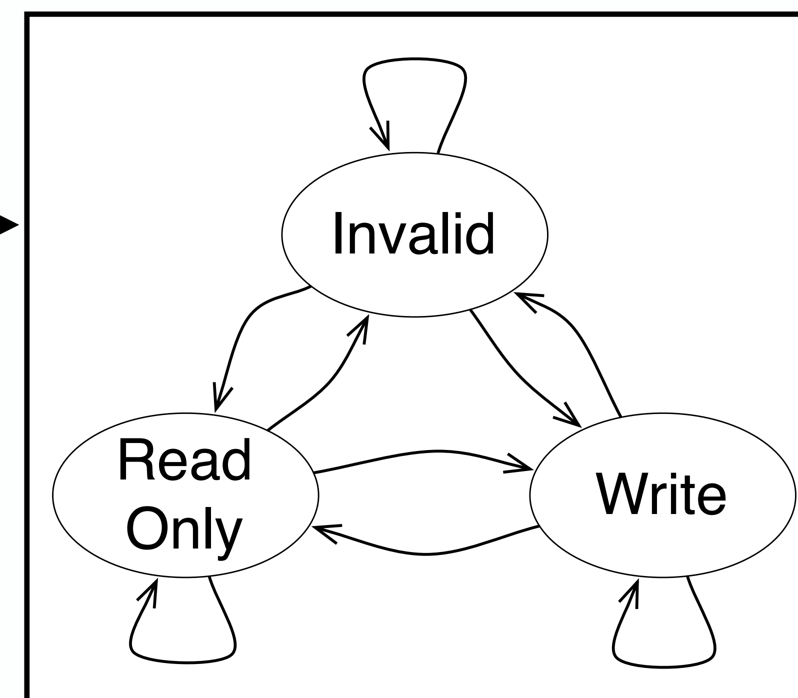
```
src : 2, dst : 0, timestamp : 8, type : prepare
src : 2, dst : 1, timestamp : 9, type : prepare
src : 0, dst : 2, timestamp : 10, type : commit
src : 1, dst : 2, timestamp : 11, type : commit
src : 2, dst : 0, timestamp : 12, type : tx_commit
src : 2, dst : 1, timestamp : 13, type : tx_commit
src : 0, dst : 2, timestamp : 14, type : ack
src : 1, dst : 2, timestamp : 15, type : ack
src : 2, dst : 0, timestamp : 16, type : prepare
src : 2, dst : 1, timestamp : 17, type : prepare
src : 0, dst : 2, timestamp : 18, type : commit
src : 1, dst : 2, timestamp : 19, type : commit
src : 2, dst : 0, timestamp : 20, type : tx_commit
src : 2, dst : 1, timestamp : 21, type : tx_commit
src : 0, dst : 2, timestamp : 22, type : ack
src : 1, dst : 2, timestamp : 23, type : ack
src : 2, dst : 0, timestamp : 0, type : prepare
src : 2, dst : 1, timestamp : 1, type : prepare
src : 0, dst : 2, timestamp : 2, type : commit
src : 1, dst : 2, timestamp : 3, type : commit
src : 2, dst : 0, timestamp : 4, type : tx_commit
src : 2, dst : 1, timestamp : 5, type : tx_commit
src : 0, dst : 2, timestamp : 6, type : ack
src : 1, dst : 2, timestamp : 7, type : ack
src : 2, dst : 0, timestamp : 8, type : prepare
src : 2, dst : 1, timestamp : 9, type : prepare
src : 0, dst : 2, timestamp : 10, type : commit
src : 1, dst : 2, timestamp : 11, type : commit
src : 2, dst : 0, timestamp : 12, type : tx_commit
src : 2, dst : 1, timestamp : 13, type : tx_commit
src : 0, dst : 2, timestamp : 14, type : ack
src : 1, dst : 2, timestamp : 15, type : ack
src : 2, dst : 0, timestamp : 16, type : prepare
src : 2, dst : 1, timestamp : 17, type : prepare
src : 0, dst : 2, timestamp : 18, type : commit
src : 1, dst : 2, timestamp : 19, type : commit
src : 2, dst : 0, timestamp : 20, type : tx_commit
src : 2, dst : 1, timestamp : 21, type : tx_commit
src : 0, dst : 2, timestamp : 22, type : ack
src : 1, dst : 2, timestamp : 23, type : ack
src : 2, dst : 0, timestamp : 0, type : prepare
src : 2, dst : 1, timestamp : 1, type : prepare
src : 0, dst : 2, timestamp : 2, type : commit
src : 1, dst : 2, timestamp : 3, type : commit
src : 2, dst : 0, timestamp : 4, type : tx_commit
src : 2, dst : 1, timestamp : 5, type : tx_commit
src : 0, dst : 2, timestamp : 6, type : ack
src : 1, dst : 2, timestamp : 7, type : ack
src : 2, dst : 0, timestamp : 8, type : prepare
src : 2, dst : 1, timestamp : 9, type : prepare
src : 0, dst : 2, timestamp : 10, type : commit
src : 1, dst : 2, timestamp : 11, type : commit
src : 2, dst : 0, timestamp : 12, type : tx_commit
src : 2, dst : 1, timestamp : 13, type : tx_commit
src : 0, dst : 2, timestamp : 14, type : ack
src : 1, dst : 2, timestamp : 15, type : ack
src : 2, dst : 0, timestamp : 16, type : prepare
src : 2, dst : 1, timestamp : 17, type : prepare
src : 0, dst : 2, timestamp : 18, type : commit
src : 1, dst : 2, timestamp : 19, type : commit
src : 2, dst : 0, timestamp : 20, type : tx_commit
src : 2, dst : 1, timestamp : 21, type : tx_commit
src : 0, dst : 2, timestamp : 22, type : ack
src : 1, dst : 2, timestamp : 23, type : ack
src : 2, dst : 0, timestamp : 0, type : prepare
src : 2, dst : 1, timestamp : 1, type : prepare
src : 0, dst : 2, timestamp : 2, type : commit
src : 1, dst : 2, timestamp : 3, type : commit
src : 2, dst : 0, timestamp : 4, type : tx_commit
src : 2, dst : 1, timestamp : 5, type : tx_commit
src : 0, dst : 2, timestamp : 6, type : ack
src : 1, dst : 2, timestamp : 7, type : ack
src : 2, dst : 0, timestamp : 8, type : prepare
src : 2, dst : 1, timestamp : 9, type : prepare
src : 0, dst : 2, timestamp : 10, type : commit
src : 1, dst : 2, timestamp : 11, type : commit
src : 2, dst : 0, timestamp : 12, type : tx_commit
src : 2, dst : 1, timestamp : 13, type : tx_commit
src : 0, dst : 2, timestamp : 14, type : ack
src : 1, dst : 2, timestamp : 15, type : ack
src : 2, dst : 0, timestamp : 16, type : prepare
src : 2, dst : 1, timestamp : 17, type : prepare
src : 0, dst : 2, timestamp : 18, type : commit
src : 1, dst : 2, timestamp : 19, type : commit
src : 2, dst : 0, timestamp : 20, type : tx_commit
src : 2, dst : 1, timestamp : 21, type : tx_commit
src : 0, dst : 2, timestamp : 22, type : ack
src : 1, dst : 2, timestamp : 23, type : ack
src : 0, dst : 2, timestamp : 2, type : commit
src : 1, dst : 2, timestamp : 3, type : commit
src : 2, dst : 0, timestamp : 4, type : tx_commit
src : 2, dst : 1, timestamp : 5, type : tx_commit
src : 0, dst : 2, timestamp : 6, type : ack
src : 1, dst : 2, timestamp : 7, type : ack
src : 2, dst : 0, timestamp : 8, type : prepare
src : 2, dst : 1, timestamp : 9, type : prepare
src : 0, dst : 2, timestamp : 10, type : commit
src : 1, dst : 2, timestamp : 11, type : commit
src : 2, dst : 0, timestamp : 12, type : tx_commit
src : 2, dst : 1, timestamp : 13, type : tx_commit
src : 0, dst : 2, timestamp : 14, type : ack
src : 1, dst : 2, timestamp : 15, type : ack
src : 2, dst : 0, timestamp : 16, type : prepare
src : 2, dst : 1, timestamp : 17, type : prepare
src : 0, dst : 2, timestamp : 18, type : commit
src : 1, dst : 2, timestamp : 19, type : commit
src : 2, dst : 0, timestamp : 20, type : tx_commit
src : 2, dst : 1, timestamp : 21, type : tx_commit
src : 0, dst : 2, timestamp : 22, type : ack
src : 1, dst : 2, timestamp : 23, type : ack
src : 2, dst : 0, timestamp : 0, type : prepare
src : 2, dst : 1, timestamp : 1, type : prepare
src : 0, dst : 2, timestamp : 2, type : commit
src : 1, dst : 2, timestamp : 3, type : commit
src : 2, dst : 0, timestamp : 4, type : tx_commit
src : 2, dst : 1, timestamp : 5, type : tx_commit
src : 0, dst : 2, timestamp : 6, type : ack
src : 1, dst : 2, timestamp : 7, type : ack
src : 2, dst : 0, timestamp : 8, type : prepare
src : 2, dst : 1, timestamp : 9, type : prepare
src : 0, dst : 2, timestamp : 10, type : commit
src : 1, dst : 2, timestamp : 11, type : commit
src : 2, dst : 0, timestamp : 12, type : tx_commit
src : 2, dst : 1, timestamp : 13, type : tx_commit
src : 0, dst : 2, timestamp : 14, type : ack
src : 1, dst : 2, timestamp : 15, type : ack
src : 2, dst : 0, timestamp : 16, type : prepare
src : 2, dst : 1, timestamp : 17, type : prepare
src : 0, dst : 2, timestamp : 18, type : commit
src : 1, dst : 2, timestamp : 19, type : commit
src : 2, dst : 0, timestamp : 20, type : tx_commit
src : 2, dst : 1, timestamp : 21, type : tx_commit
src : 0, dst : 2, timestamp : 22, type : ack
src : 1, dst : 2, timestamp : 23, type : ack
src : 2, dst : 0, timestamp : 0, type : prepare
src : 2, dst : 1, timestamp : 1, type : prepare
src : 0, dst : 2, timestamp : 2, type : commit
src : 1, dst : 2, timestamp : 3, type : commit
src : 2, dst : 0, timestamp : 4, type : tx_commit
src : 2, dst : 1, timestamp : 5, type : tx_commit
src : 0, dst : 2, timestamp : 6, type : ack
src : 1, dst : 2, timestamp : 7, type : ack
src : 2, dst : 0, timestamp : 8, type : prepare
src : 2, dst : 1, timestamp : 9, type : prepare
src : 0, dst : 2, timestamp : 10, type : commit
src : 1, dst : 2, timestamp : 11, type : commit
src : 2, dst : 0, timestamp : 12, type : tx_commit
src : 2, dst : 1, timestamp : 13, type : tx_commit
src : 0, dst : 2, timestamp : 14, type : ack
src : 1, dst : 2, timestamp : 15, type : ack
src : 2, dst : 0, timestamp : 16, type : prepare
src : 2, dst : 1, timestamp : 17, type : prepare
src : 0, dst : 2, timestamp : 18, type : commit
src : 1, dst : 2, timestamp : 19, type : commit
src : 2, dst : 0, timestamp : 20, type : tx_commit
src : 2, dst : 1, timestamp : 21, type : tx_commit
src : 0, dst : 2, timestamp : 22, type : ack
src : 1, dst : 2, timestamp : 23, type : ack
src : 2, dst : 0, timestamp : 16, type : prepare
src : 2, dst : 1, timestamp : 17, type : prepare
src : 0, dst : 2, timestamp : 18, type : commit
src : 1, dst : 2, timestamp : 19, type : commit
src : 2, dst : 0, timestamp : 20, type : tx_commit
src : 2, dst : 1, timestamp : 21, type : tx_commit
src : 0, dst : 2, timestamp : 22, type : ack
src : 1, dst : 2, timestamp : 23, type : ack
src : 1, dst : 2, timestamp : 19, type : commit
...
```

Observations

Inference algorithm

Prior work:

- Cook et al. TSE 1998
- Mariani et al. ICSE 2007
- Dallmeier et al. ASE 2009
- Beschastnikh et al. FSE 2011
- ...



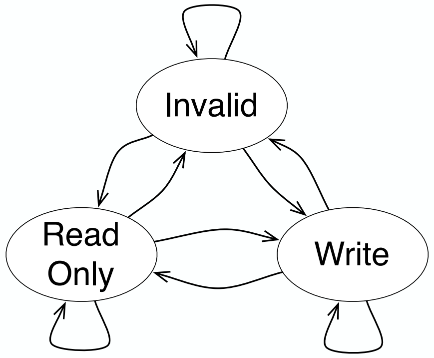
Model

FSM-inference representations

```
src : 2, dst : 0, timestamp : 0, type : prepare
src : 2, dst : 1, timestamp : 1, type : prepare
src : 0, dst : 2, timestamp : 2, type : commit
src : 1, dst : 2, timestamp : 3, type : commit
src : 2, dst : 0, timestamp : 4, type : tx_commit
src : 2, dst : 1, timestamp : 5, type : tx_commit
src : 0, dst : 2, timestamp : 6, type : ack
src : 1, dst : 2, timestamp : 7, type : ack
src : 2, dst : 0, timestamp : 8, type : prepare
src : 2, dst : 1, timestamp : 9, type : prepare
src : 0, dst : 2, timestamp : 10, type : commit
src : 1, dst : 2, timestamp : 11, type : commit
src : 2, dst : 0, timestamp : 12, type : tx_commit
src : 2, dst : 1, timestamp : 13, type : tx_commit
src : 0, dst : 2, timestamp : 14, type : ack
src : 1, dst : 2, timestamp : 15, type : ack
src : 2, dst : 0, timestamp : 16, type : prepare
src : 2, dst : 1, timestamp : 17, type : prepare
src : 0, dst : 2, timestamp : 18, type : commit
src : 1, dst : 2, timestamp : 19, type : commit
src : 2, dst : 0, timestamp : 20, type : tx_commit
src : 2, dst : 1, timestamp : 21, type : tx_commit
src : 0, dst : 2, timestamp : 22, type : ack
src : 1, dst : 2, timestamp : 23, type : ack
src : 2, dst : 0, timestamp : 0, type : prepare
src : 2, dst : 1, timestamp : 1, type : prepare
src : 0, dst : 2, timestamp : 2, type : commit
src : 1, dst : 2, timestamp : 3, type : commit
src : 2, dst : 0, timestamp : 4, type : tx_commit
```

Observations

Inference algorithm

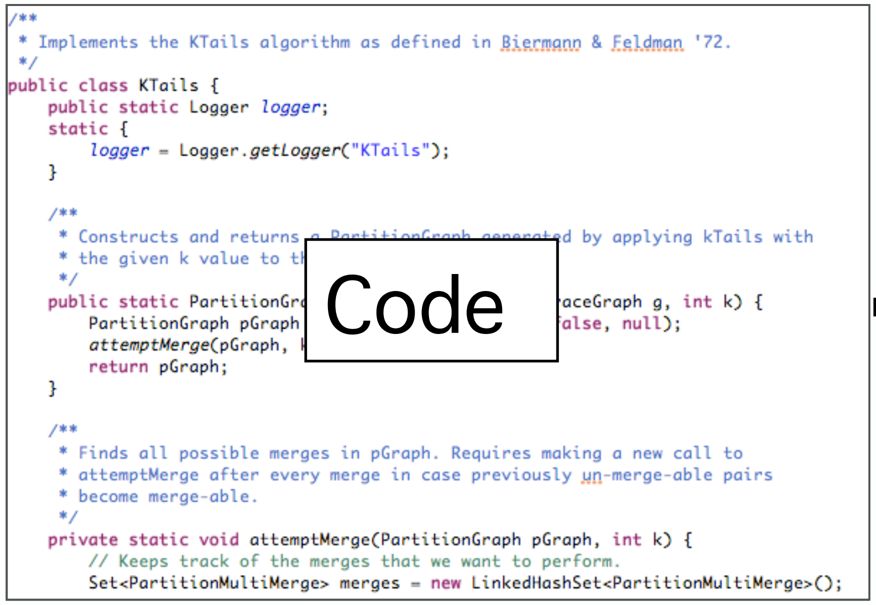


Model

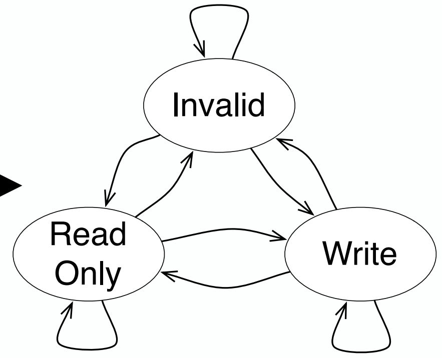
FSM-inference representations

```
src : 2, dst : 0, timestamp : 0, type : prepare
src : 2, dst : 1, timestamp : 1, type : prepare
src : 0, dst : 2, timestamp : 2, type : commit
src : 1, dst : 2, timestamp : 3, type : commit
src : 2, dst : 0, timestamp : 4, type : tx_commit
src : 2, dst : 1, timestamp : 5, type : tx_commit
src : 0, dst : 2, timestamp : 6, type : ack
src : 1, dst : 2, timestamp : 7, type : ack
src : 2, dst : 0, timestamp : 8, type : prepare
src : 2, dst : 1, timestamp : 9, type : prepare
src : 0, dst : 2, timestamp : 10, type : commit
src : 1, dst : 2, timestamp : 11, type : commit
src : 2, dst : 0, timestamp : 12, type : tx_commit
src : 2, dst : 1, timestamp : 13, type : tx_commit
src : 0, dst : 2, timestamp : 14, type : ack
src : 1, dst : 2, timestamp : 15, type : ack
src : 2, dst : 0, timestamp : 16, type : prepare
src : 2, dst : 1, timestamp : 17, type : prepare
src : 0, dst : 2, timestamp : 18, type : commit
src : 1, dst : 2, timestamp : 19, type : commit
src : 2, dst : 0, timestamp : 20, type : tx_commit
src : 2, dst : 1, timestamp : 21, type : tx_commit
src : 0, dst : 2, timestamp : 22, type : ack
src : 1, dst : 2, timestamp : 23, type : ack
src : 2, dst : 0, timestamp : 0, type : prepare
src : 2, dst : 1, timestamp : 1, type : prepare
src : 0, dst : 2, timestamp : 2, type : commit
src : 1, dst : 2, timestamp : 3, type : commit
src : 2, dst : 0, timestamp : 4, type : tx_commit
```

Observations

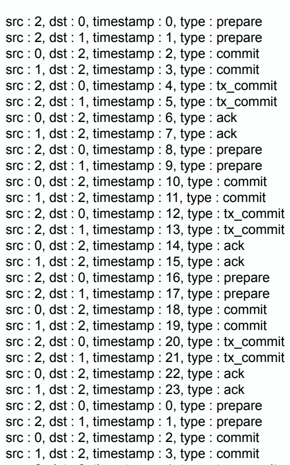



```
/**
 * Implements the KTails algorithm as defined in Biermann & Feldman '72.
 */
public class KTails {
    public static Logger logger;
    static {
        logger = Logger.getLogger("KTails");
    }

    /**
     * Constructs and returns
     * the given k value to
     */
    public static PartitionGr
        PartitionGraph pGraph
        attemptMerge(pGraph,
        return pGraph;
    }

    /**
     * Finds all possible merges in pGraph. Requires making a new call to
     * attemptMerge after every merge in case previously un-merge-able pairs
     * become merge-able.
     */
    private static void attemptMerge(PartitionGraph pGraph, int k) {
        // Keeps track of the merges that we want to perform.
        Set<PartitionMultiMerge> merges = new LinkedHashSet<PartitionMultiMerge>();
```

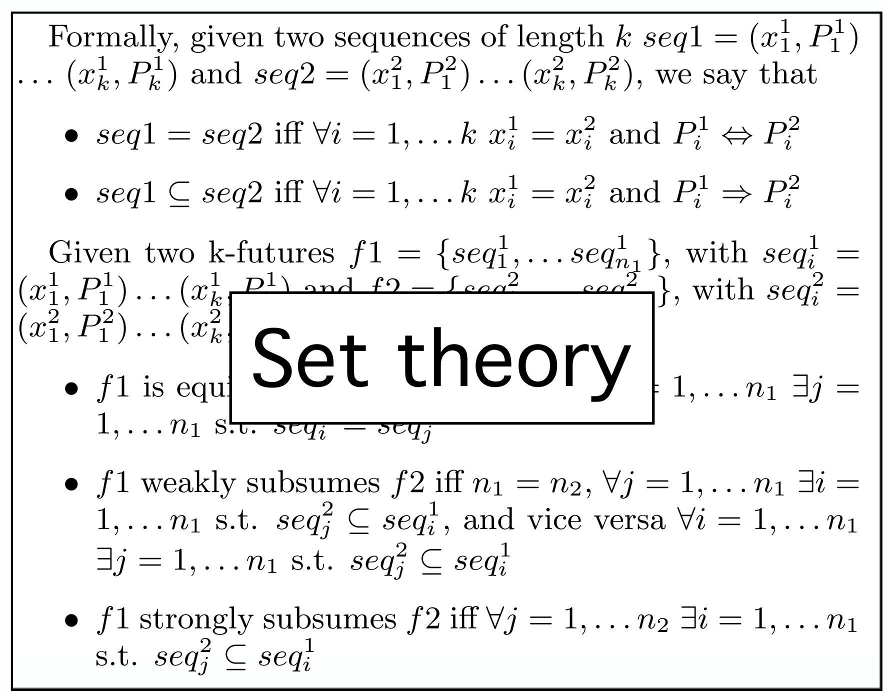


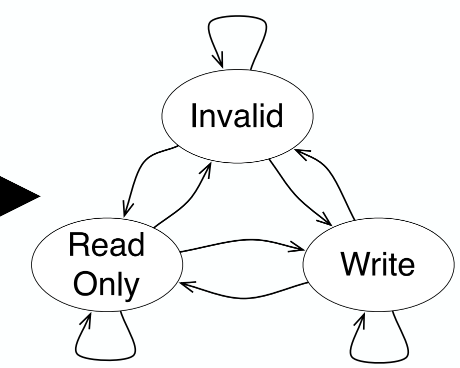
Model

FSM-inference representations



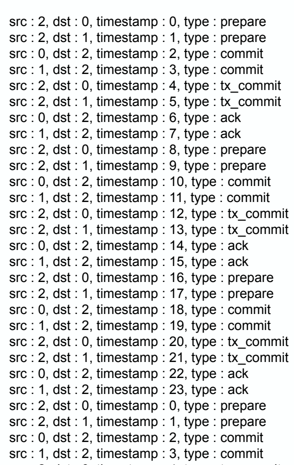
Observations



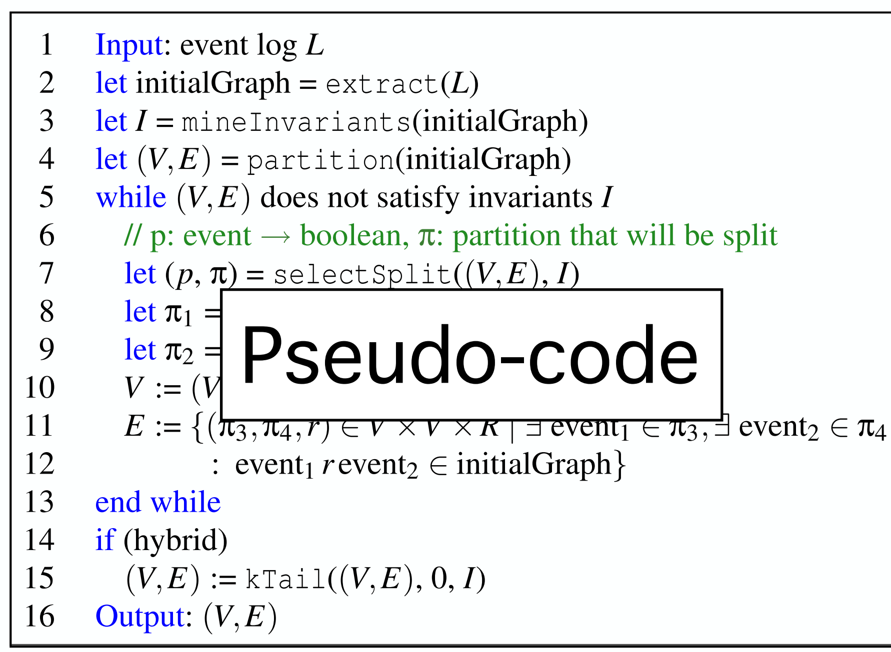


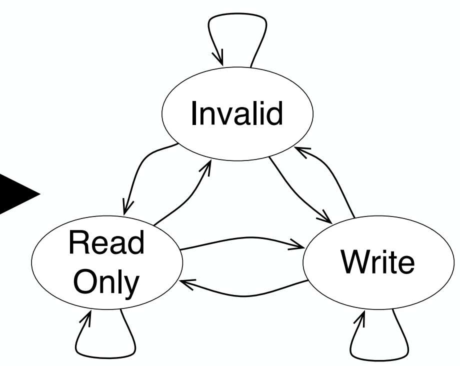
Model

FSM-inference representations



Observations

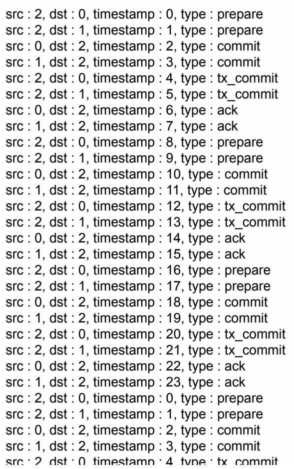




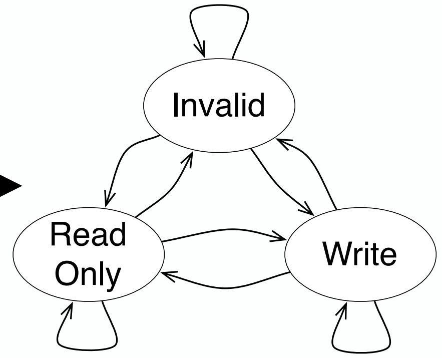
Model

FSM-inference representations

...



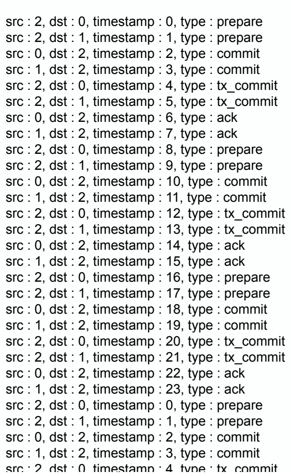
Observations



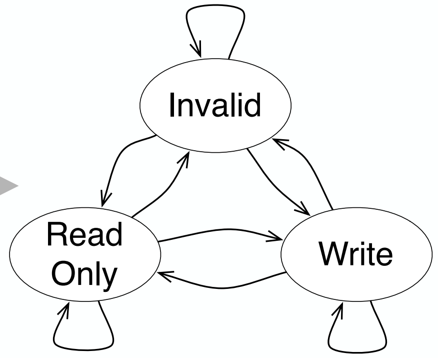
Model

FSM-inference representations

...



Observations



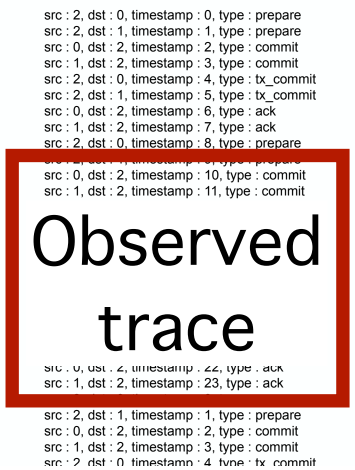
Model

Limitations:

x Transparency x Extensibility

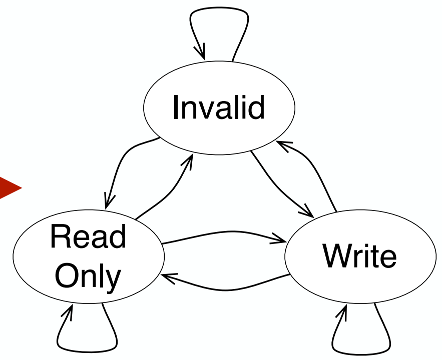
FSM-inference representations

...



Observations

Trace **admitted** by the model?



Model

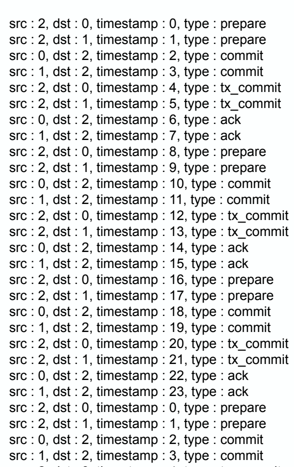
Limitations:

x Transparency

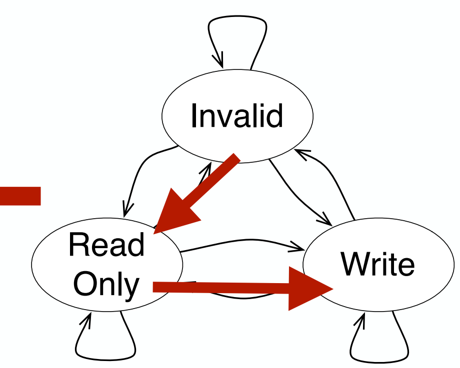
x Extensibility

FSM-inference representations

...


src : 2, dst : 0, timestamp : 0, type : prepare
src : 2, dst : 1, timestamp : 1, type : prepare
src : 0, dst : 2, timestamp : 2, type : commit
src : 1, dst : 2, timestamp : 3, type : commit
src : 2, dst : 0, timestamp : 4, type : tx_commit
src : 2, dst : 1, timestamp : 5, type : tx_commit
src : 0, dst : 2, timestamp : 6, type : ack
src : 1, dst : 2, timestamp : 7, type : ack
src : 2, dst : 0, timestamp : 8, type : prepare
src : 2, dst : 1, timestamp : 9, type : prepare
src : 0, dst : 2, timestamp : 10, type : commit
src : 1, dst : 2, timestamp : 11, type : commit
src : 2, dst : 0, timestamp : 12, type : tx_commit
src : 2, dst : 1, timestamp : 13, type : tx_commit
src : 0, dst : 2, timestamp : 14, type : ack
src : 1, dst : 2, timestamp : 15, type : ack
src : 2, dst : 0, timestamp : 16, type : prepare
src : 2, dst : 1, timestamp : 17, type : prepare
src : 0, dst : 2, timestamp : 18, type : commit
src : 1, dst : 2, timestamp : 19, type : commit
src : 2, dst : 0, timestamp : 20, type : tx_commit
src : 2, dst : 1, timestamp : 21, type : tx_commit
src : 0, dst : 2, timestamp : 22, type : ack
src : 1, dst : 2, timestamp : 23, type : ack
src : 2, dst : 0, timestamp : 0, type : prepare
src : 2, dst : 1, timestamp : 1, type : prepare
src : 0, dst : 2, timestamp : 2, type : commit
src : 1, dst : 2, timestamp : 3, type : commit
src : 2, dst : 0, timestamp : 4, type : tx_commit

Why is this execution admitted?



Observations

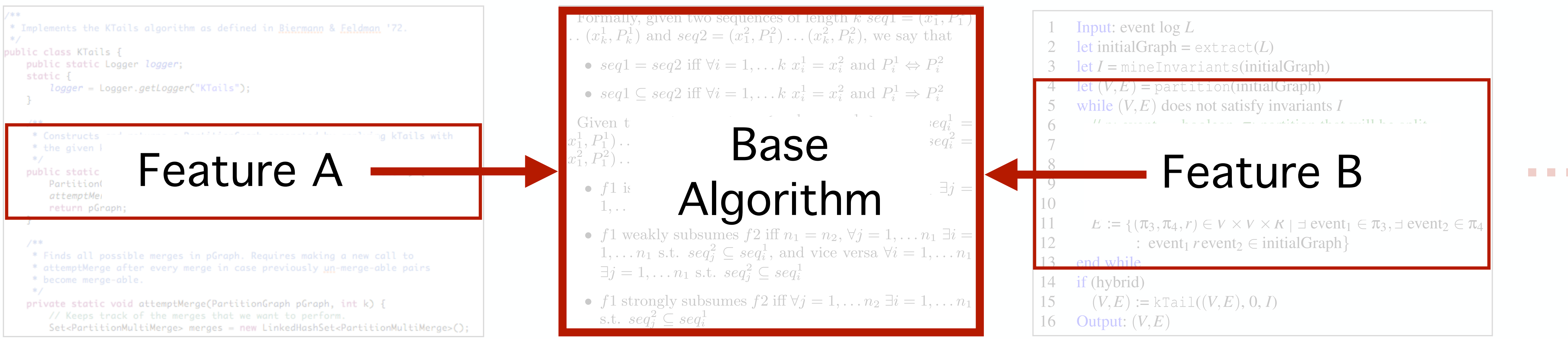
Model

Limitations:

x Transparency

x Extensibility

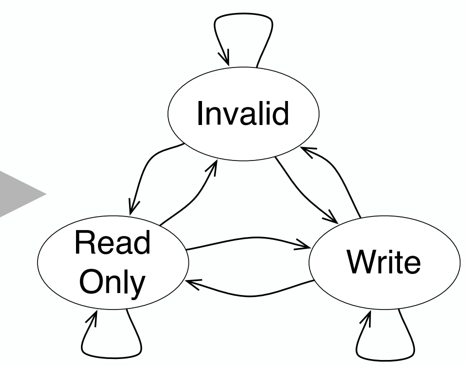
FSM-inference representations




```
src : 2, dst : 0, timestamp : 0, type : prepare
src : 2, dst : 1, timestamp : 1, type : prepare
src : 0, dst : 2, timestamp : 2, type : commit
src : 1, dst : 2, timestamp : 3, type : commit
src : 2, dst : 0, timestamp : 4, type : tx_commit
src : 2, dst : 1, timestamp : 5, type : tx_commit
src : 0, dst : 2, timestamp : 6, type : ack
src : 1, dst : 2, timestamp : 7, type : ack
src : 2, dst : 0, timestamp : 8, type : prepare
src : 2, dst : 1, timestamp : 9, type : prepare
src : 0, dst : 2, timestamp : 10, type : commit
src : 1, dst : 2, timestamp : 11, type : commit
src : 2, dst : 0, timestamp : 12, type : tx_commit
src : 2, dst : 1, timestamp : 13, type : tx_commit
src : 0, dst : 2, timestamp : 14, type : ack
src : 1, dst : 2, timestamp : 15, type : ack
src : 2, dst : 0, timestamp : 16, type : prepare
src : 2, dst : 1, timestamp : 17, type : prepare
src : 0, dst : 2, timestamp : 18, type : commit
src : 1, dst : 2, timestamp : 19, type : commit
src : 2, dst : 0, timestamp : 20, type : tx_commit
src : 2, dst : 1, timestamp : 21, type : tx_commit
src : 0, dst : 2, timestamp : 22, type : ack
src : 1, dst : 2, timestamp : 23, type : ack
src : 2, dst : 0, timestamp : 0, type : prepare
src : 2, dst : 1, timestamp : 1, type : prepare
src : 0, dst : 2, timestamp : 2, type : commit
src : 1, dst : 2, timestamp : 3, type : commit
src : 2, dst : 0, timestamp : 4, type : tx_commit
```

Observations

How to add features to an algorithm?



Model

Limitations:

x Transparency x Extensibility

FSM-inference representations

1. Low-level specifications

...

2. Monolithic architecture

Limitations:

x Transparency   x Extensibility

InvariMint: modular and declarative

Prior work:

Observations → → Model

InvariMint:

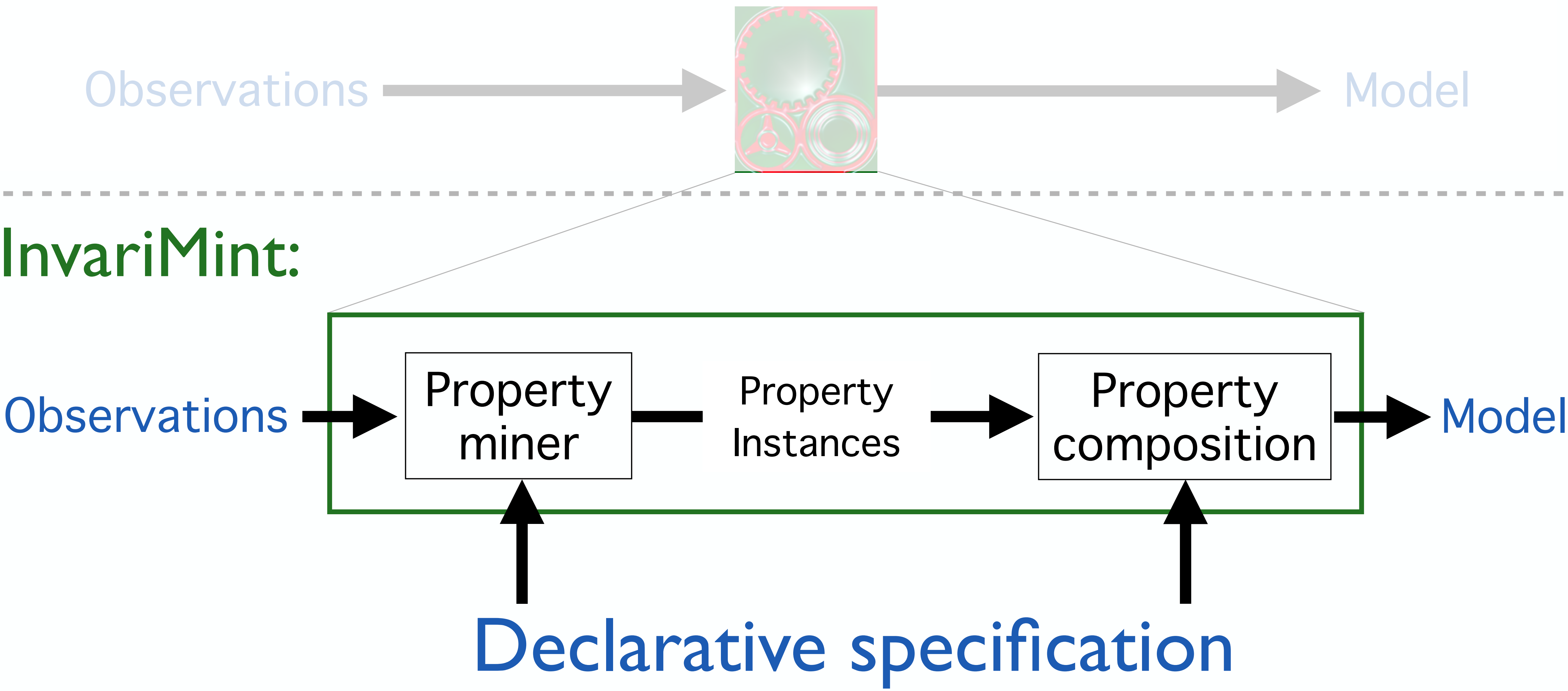
Observations → Property miner — Property Instances → Property composition → Model

Declarative specification
of FSM-inference algorithm

InvariMint: modular and declarative

Prior work:

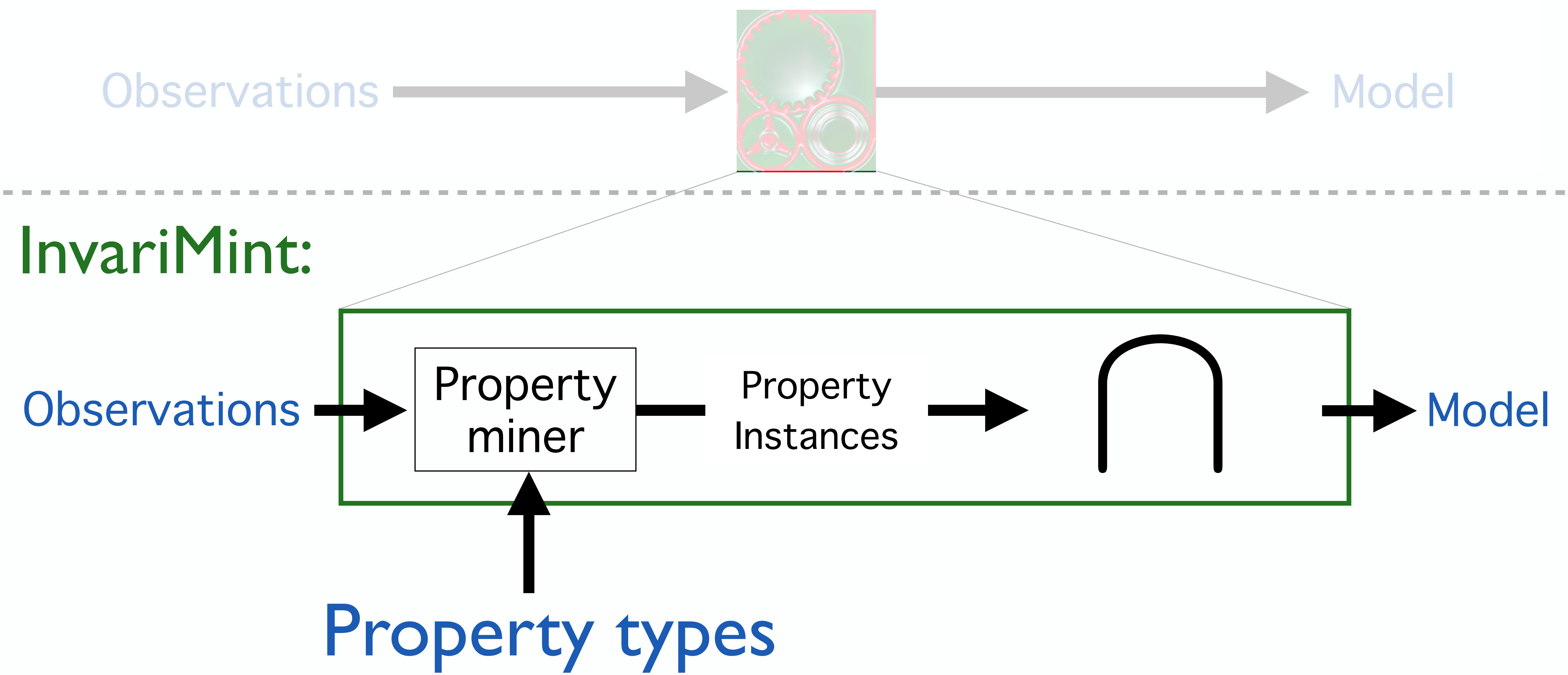
InvariMint:



✓ Transparent ✓ Extensible

InvariMint: modular and declarative

Prior work:



✓ Transparent ✓ Extensible

Outline

- FSM-inference algorithms not transparent/extensible
  - InvariMint -- modular and declarative FSM-inference
- Expressing algorithm using InvariMint
  - kTails -- a state-merging algorithm
  - Synoptic -- a state-splitting algorithm
- General specification formalism
- InvariMint evaluation

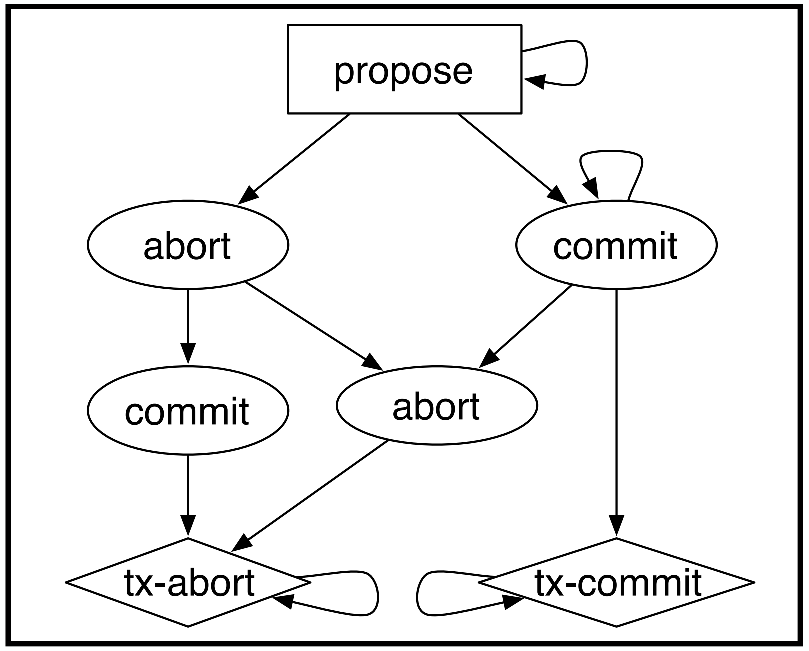
kTails and Synoptic overview

Observations

kTails

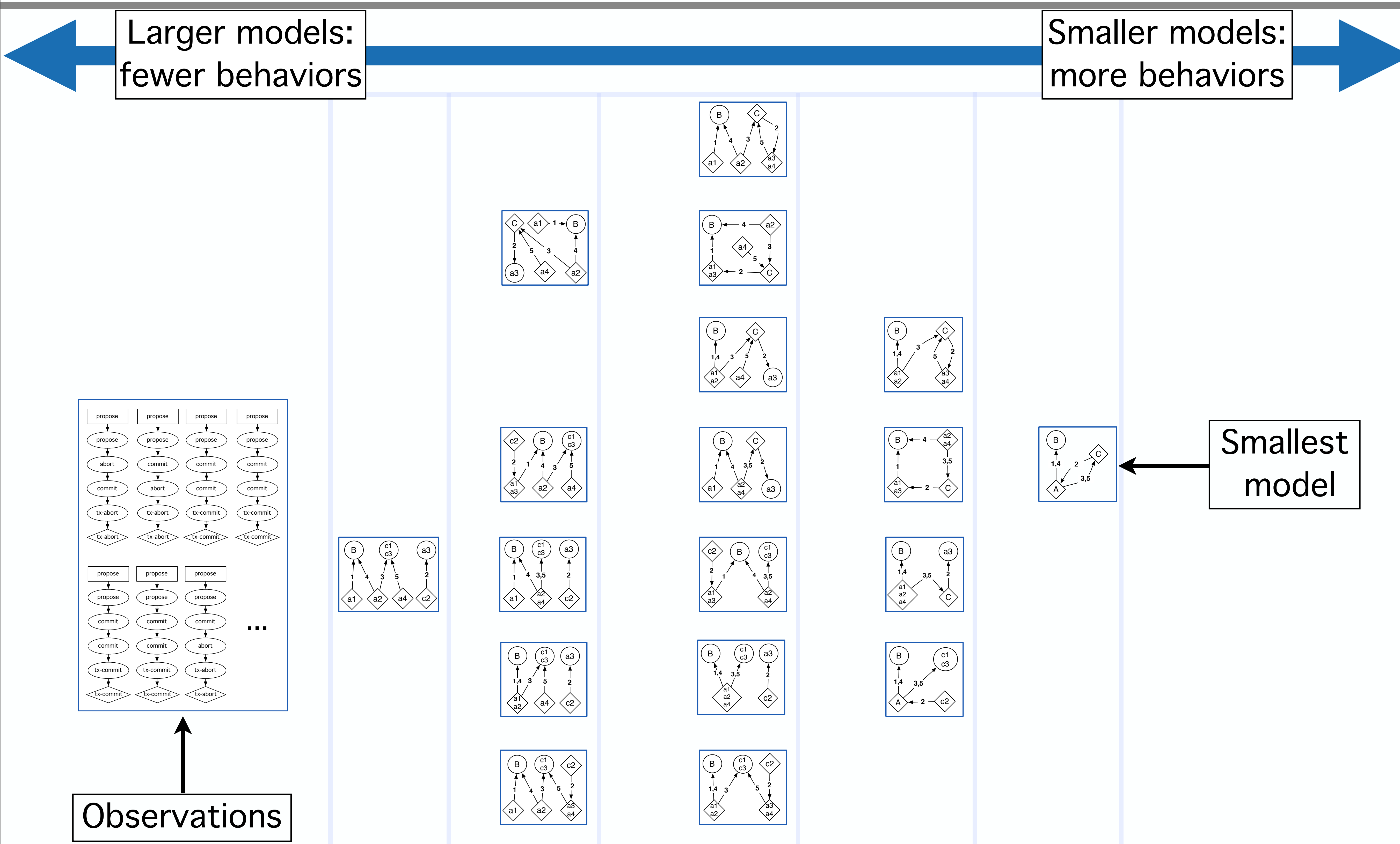
Synoptic

Biermann et al. IEEE TC 2011	Beschastnikh et al. ESEC/FSE 2011



Model

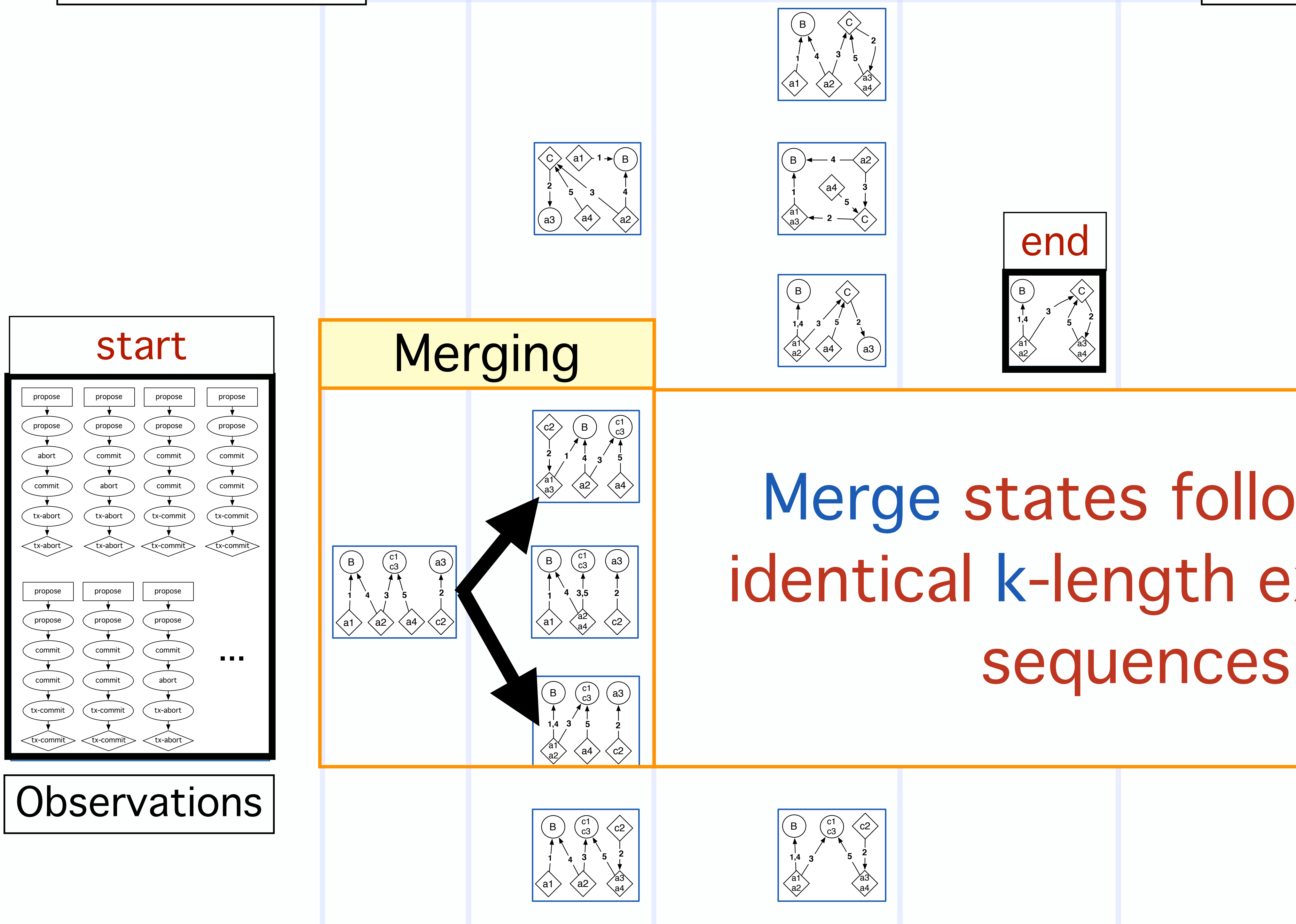
kTails and Synoptic overview



kTails(k) and Synoptic overview

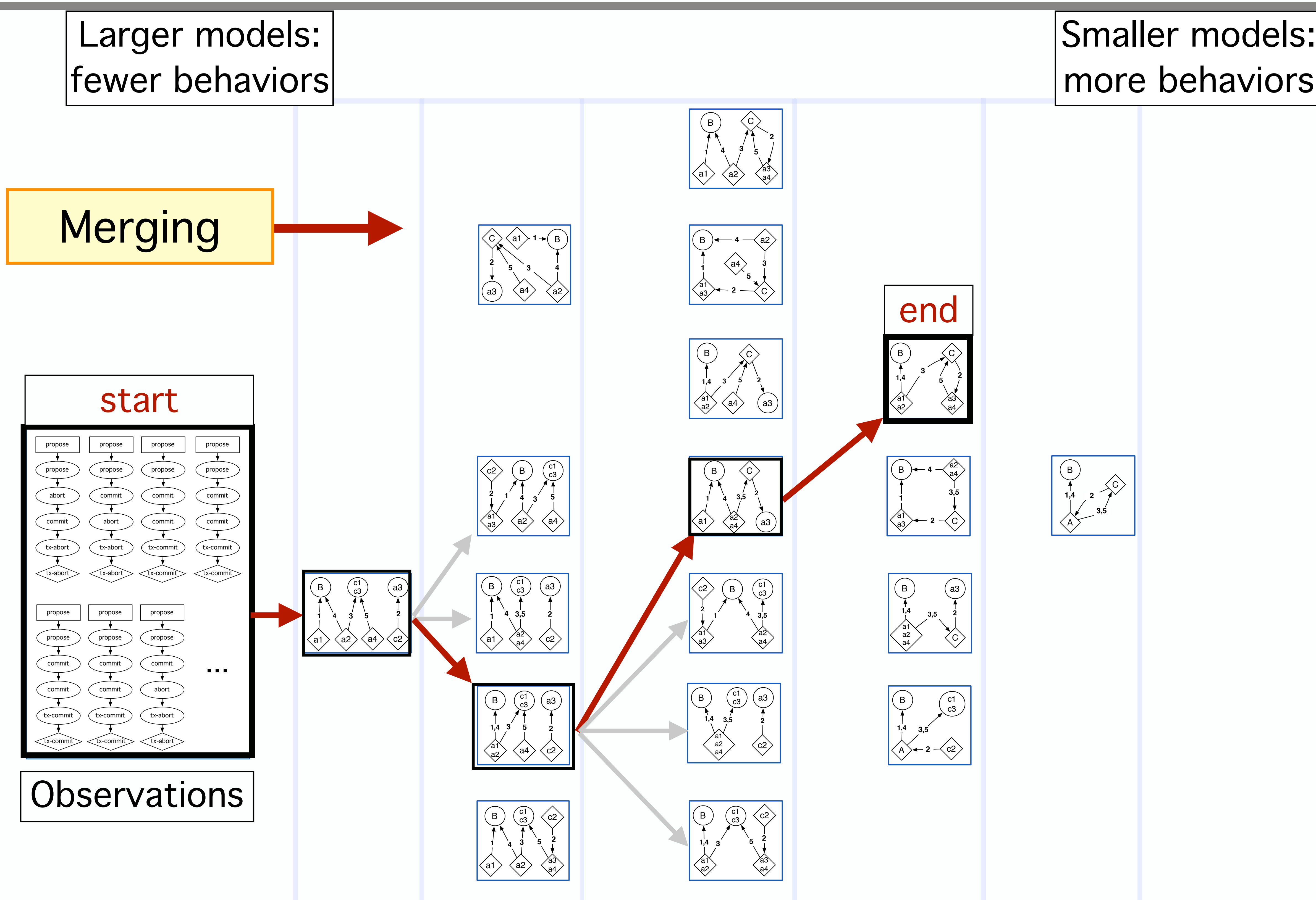
Larger models: fewer behaviors

Smaller models: more behaviors

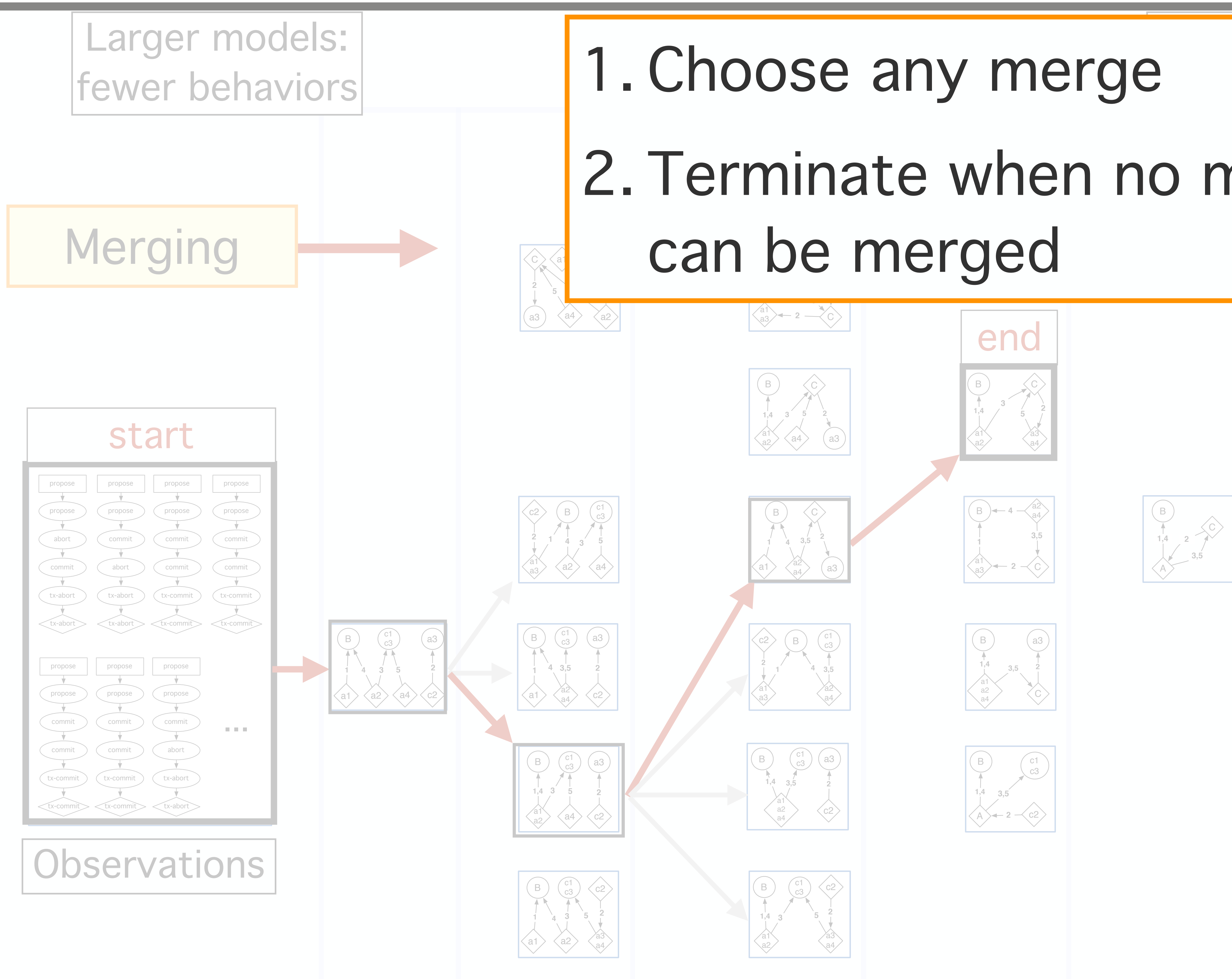


Merge states followed by identical k-length execution sequences

kTails(k) and Synoptic overview



kTails(k) and Synoptic overview



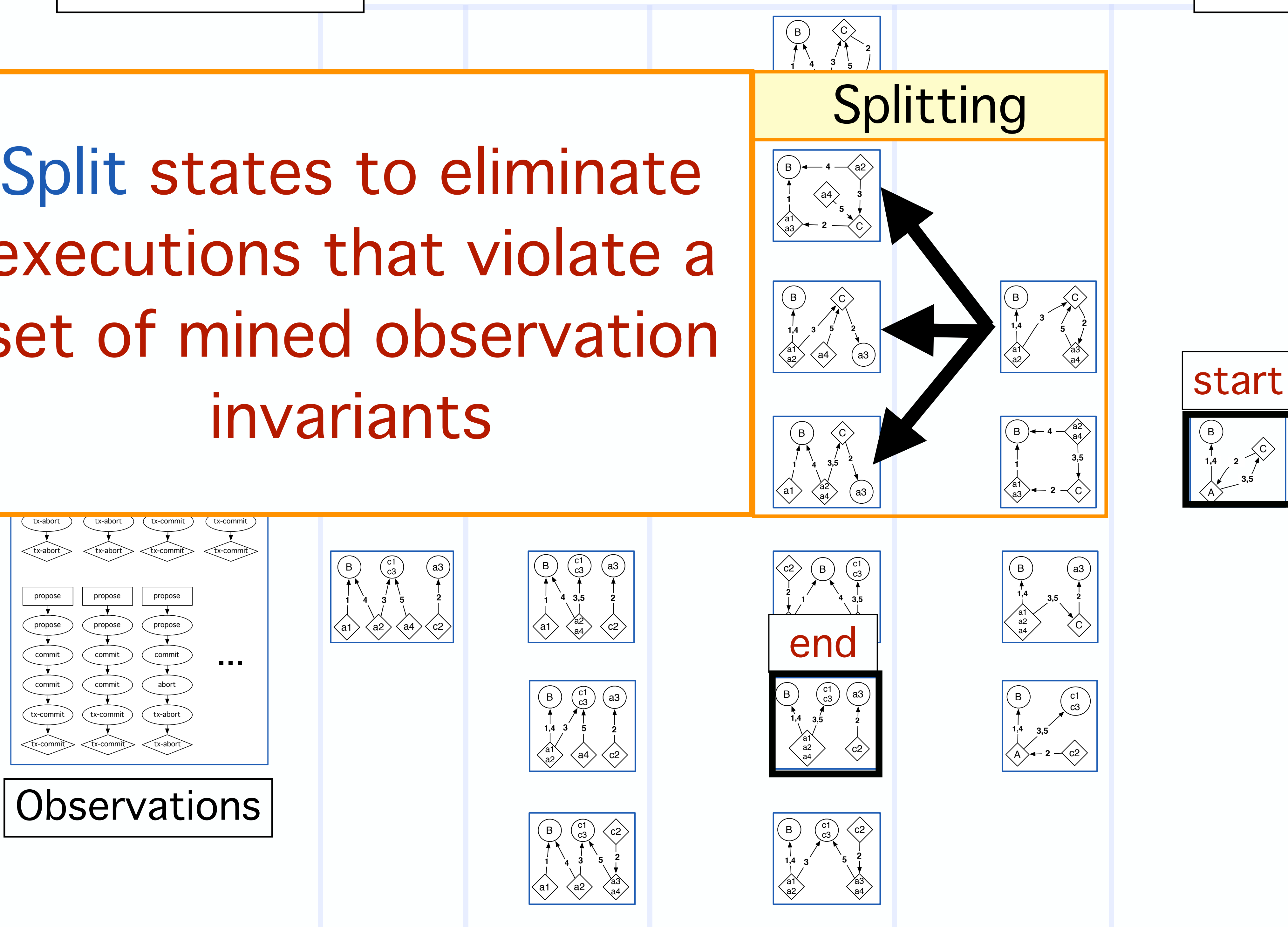
1. Choose any merge
2. Terminate when no more states can be merged

kTails(k) and Synoptic overview

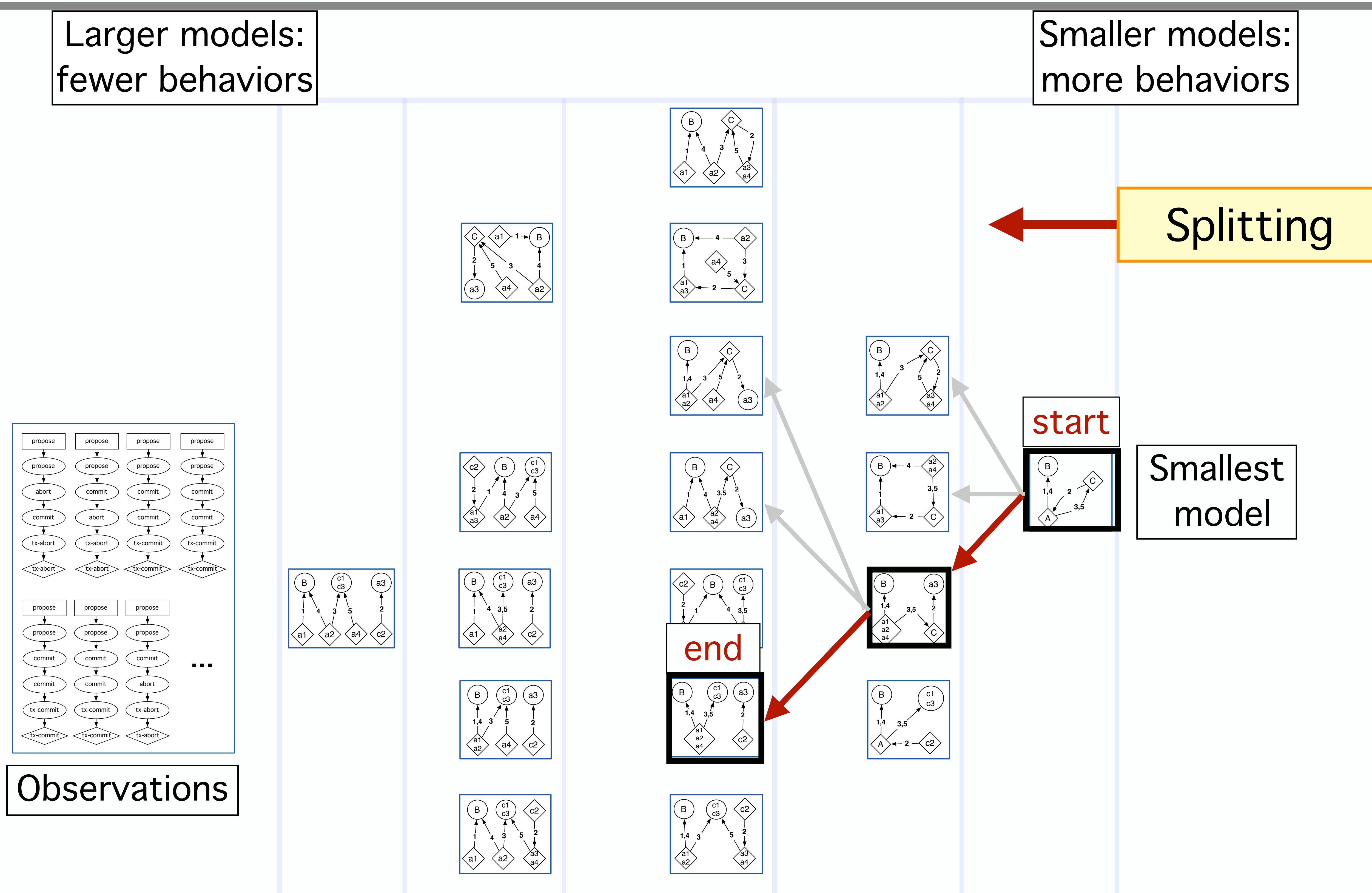
Larger models: fewer behaviors

Smaller models: more behaviors

Split states to eliminate executions that violate a set of mined observation invariants



kTails(k) and Synoptic overview

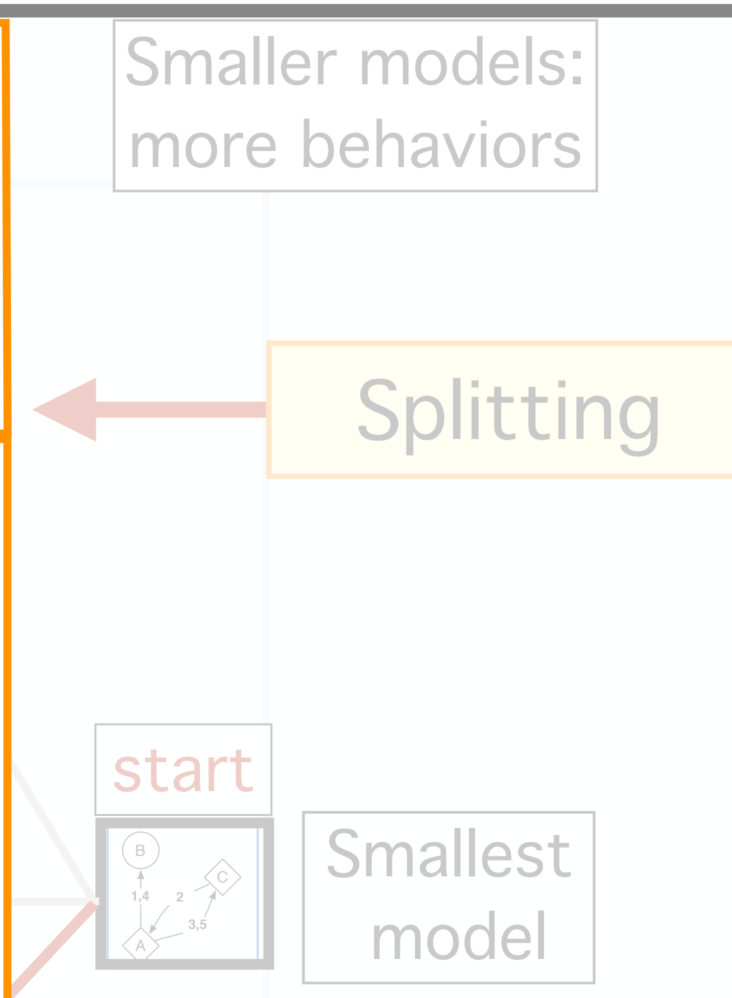


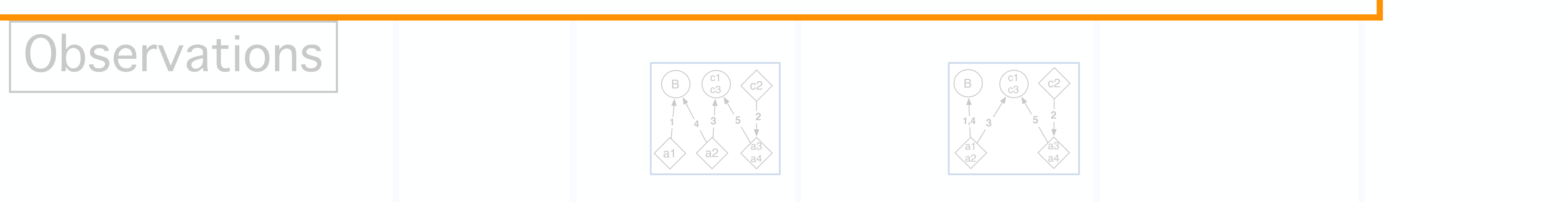
kTails(k) and Synoptic overview

Synoptic is non-deterministic:

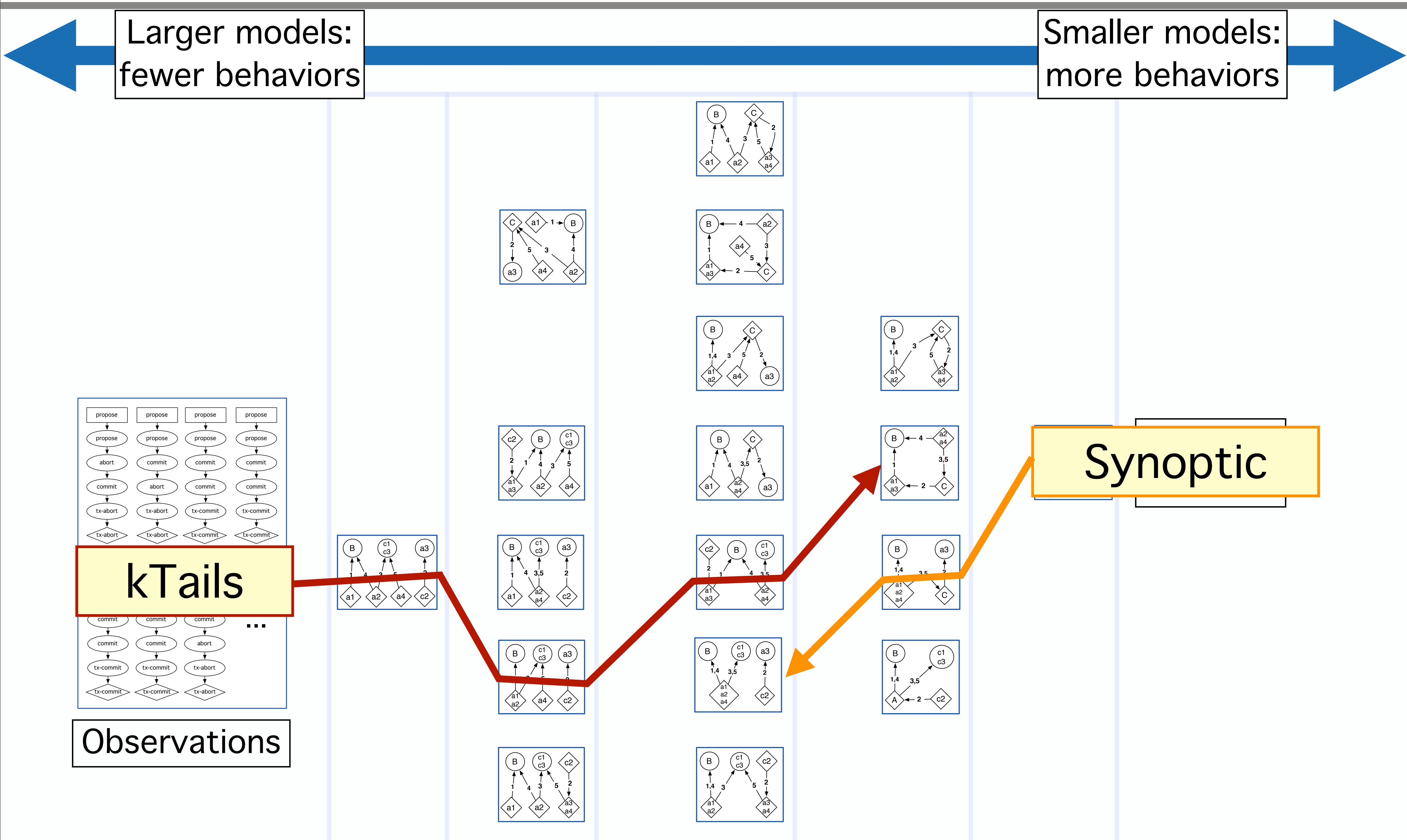
Final model depends on splitting choices

1. Choose split to eliminate counter-examples to unsatisfied mined invariants:
   - x AlwaysFollowedBy y
   - x NeverFollowedBy y
   - x AlwaysPrecedes y
2. Terminate when model satisfies all mined invariants





Motivating questions



Motivating questions

Larger models: fewer behaviors

Smaller models: more behaviors

How can we easily:

... get kTails to ignore certain k-length sequences?

... add the x AlwaysFollowedBy y invariant to kTails?

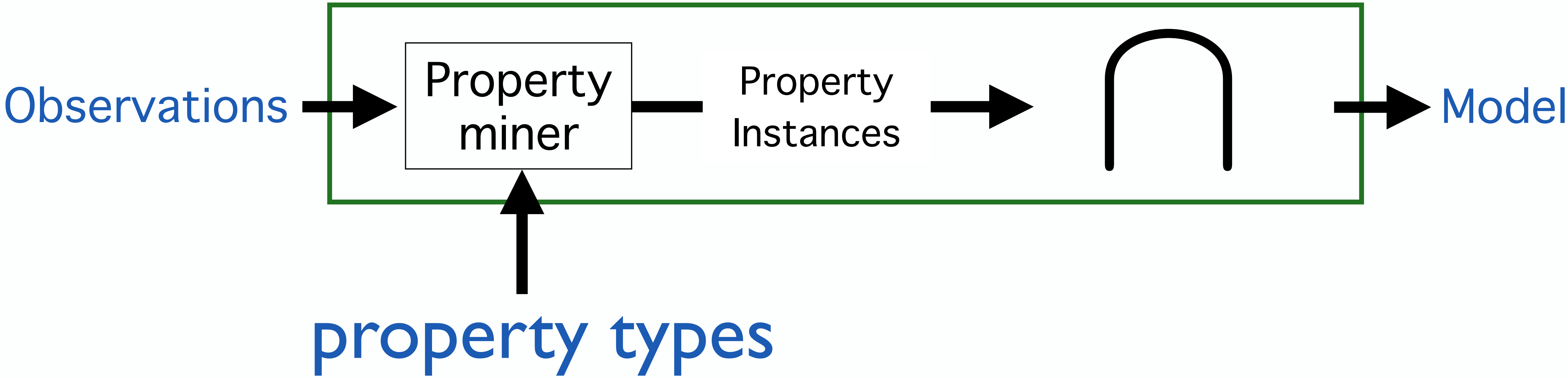
... make Synoptic deterministic?

... add a new kind of invariant to Synoptic?

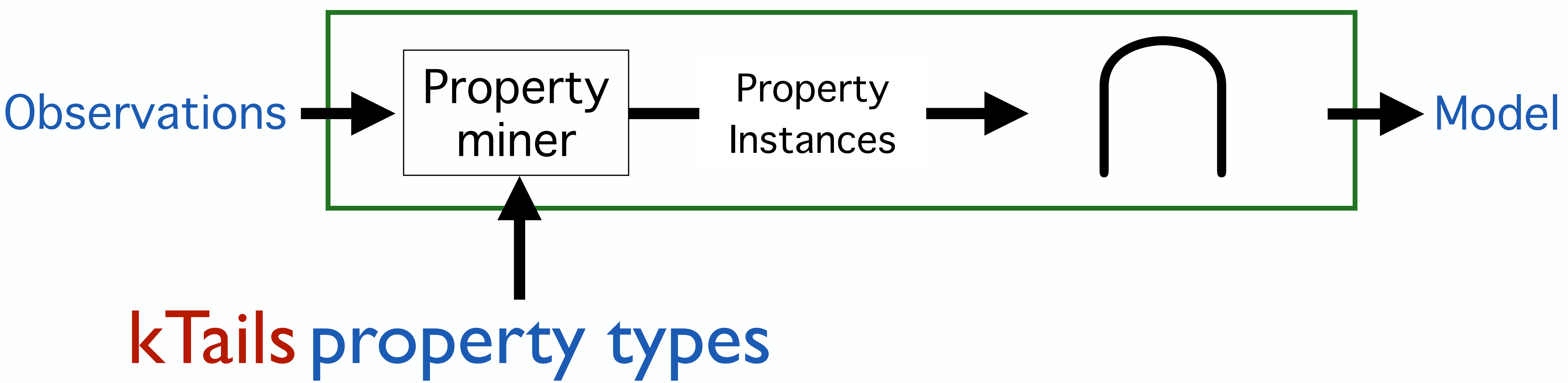
... learn which properties kTails/Synoptic preserve?

We can answer all of these questions by representing these algorithms with InvariMint

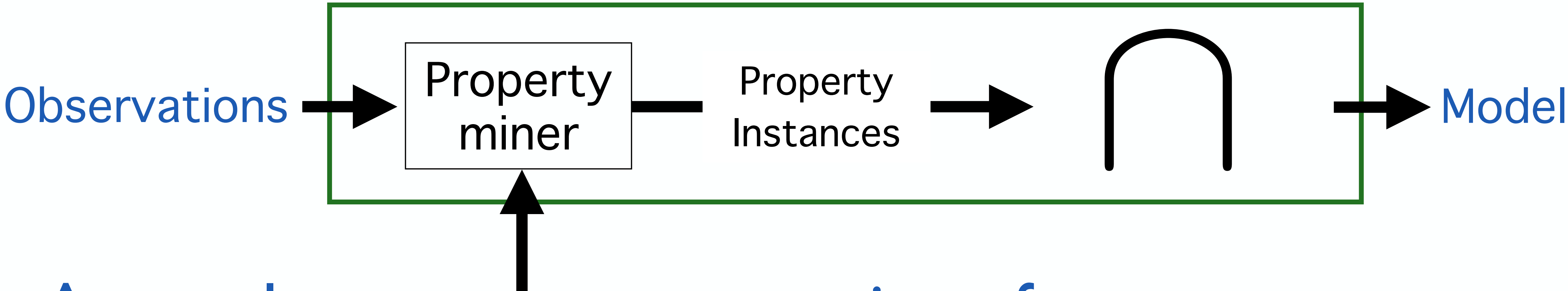
InvariMint: modular and declarative



Expressing kTails(k=1)



Expressing kTails(k=1)



A template to express merging of observation sequences that are identical in the first 1 event(s)

=

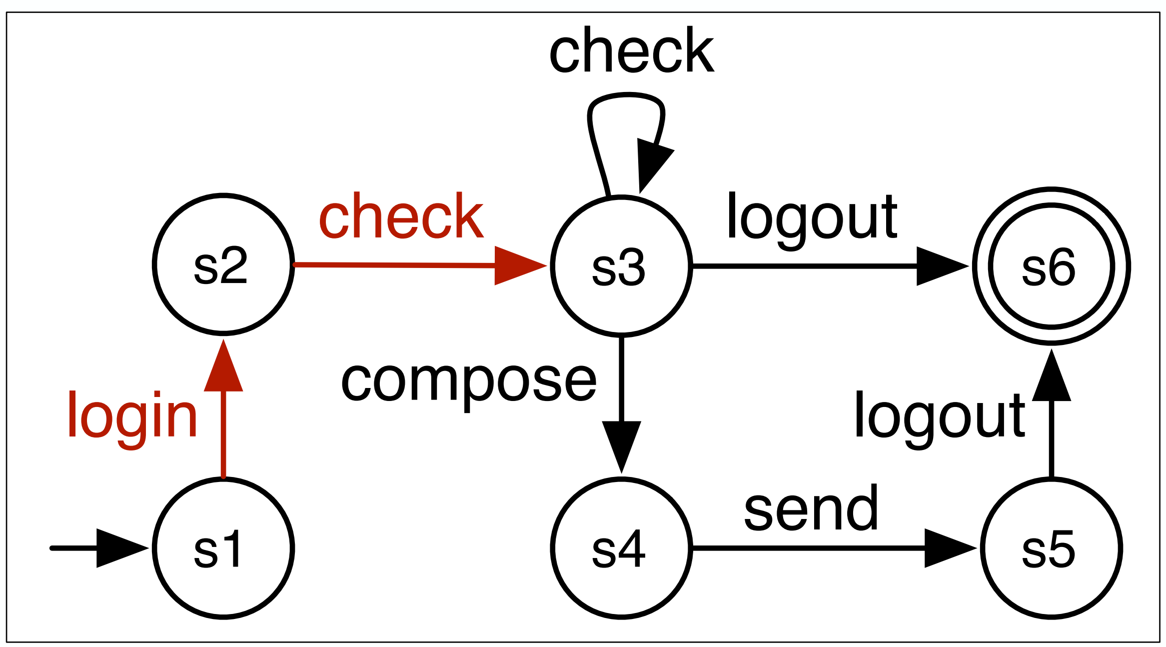
A property type:

"x can be immediately followed by one of Y"

Expressing kTails(k=1)

"x can be immediately followed by one of Y"

trace 1:	trace 2:
login	login
check	check
check	compose
logout	send
	logout



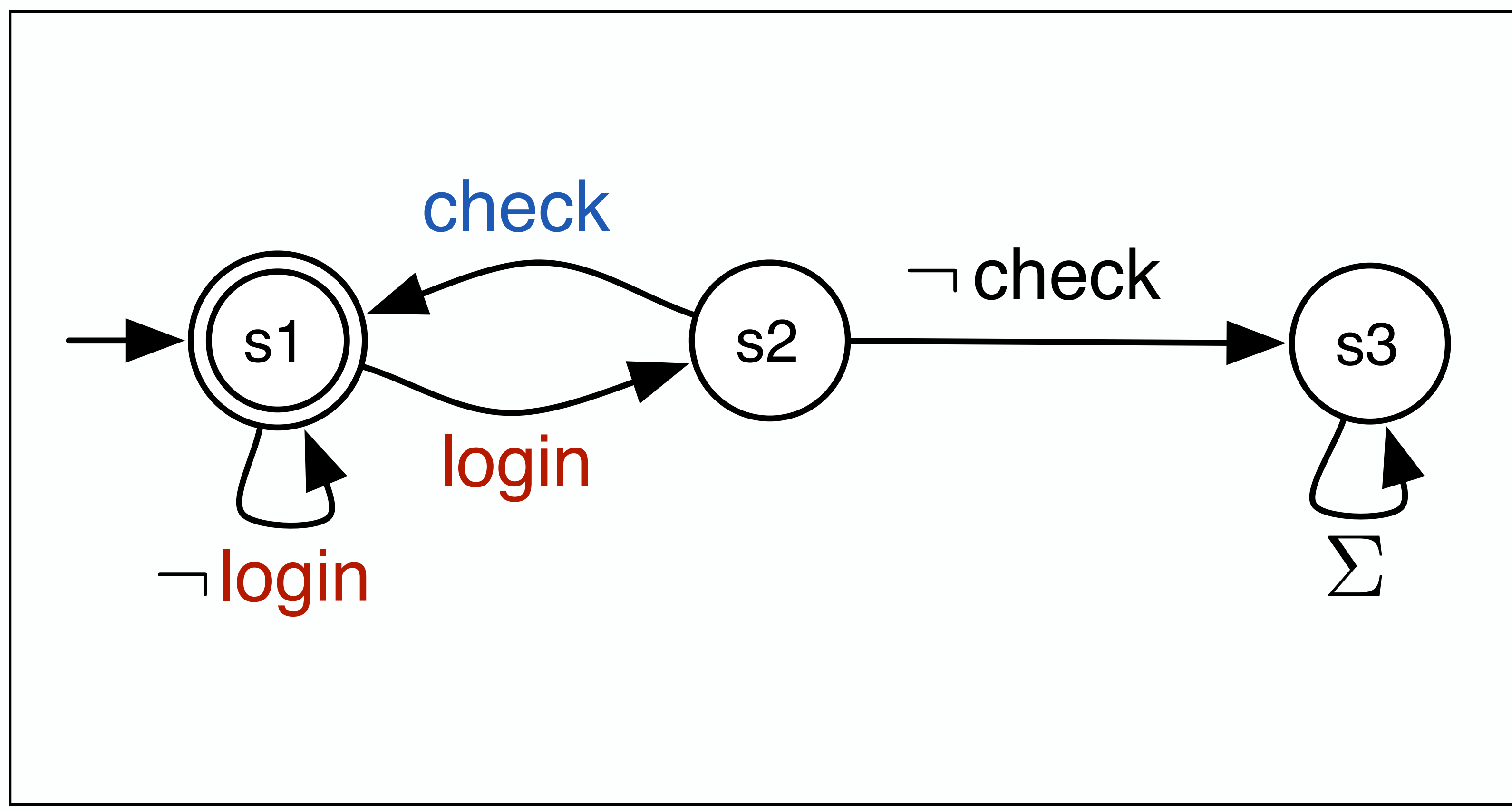
Observations

kTails output

Expressing kTails(k=1)

"x can be immediately followed by one of Y"

trace 1:	trace 2:
login check check logout	login check compose send logout



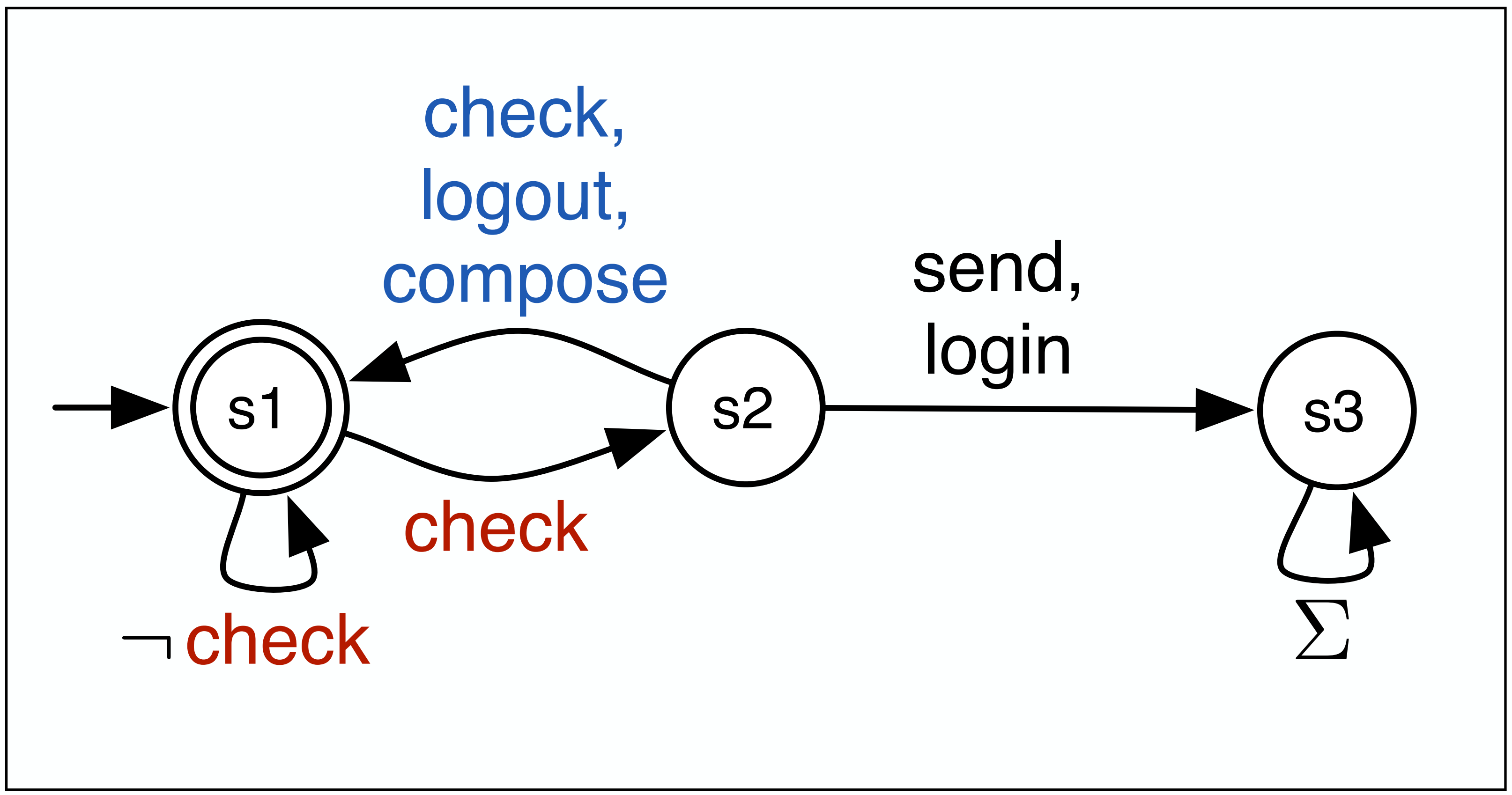
Observations

x=login property instance

Expressing kTails(k=1)

"x can be immediately followed by one of Y"

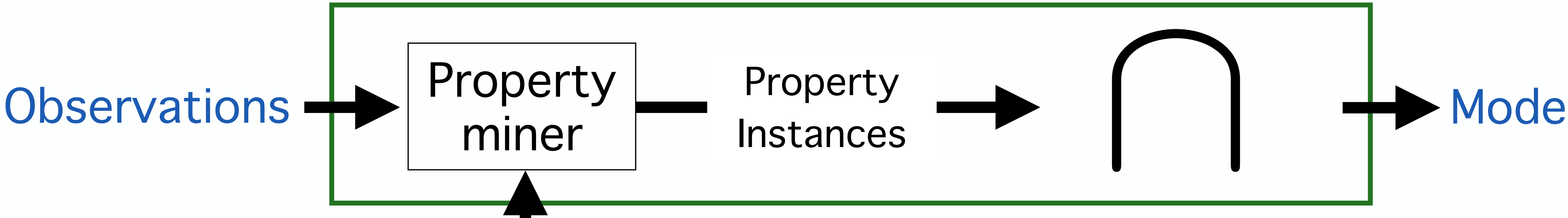
trace 1:	trace 2:
login check check logout	login check compose send logout



Observations

x=check property instance

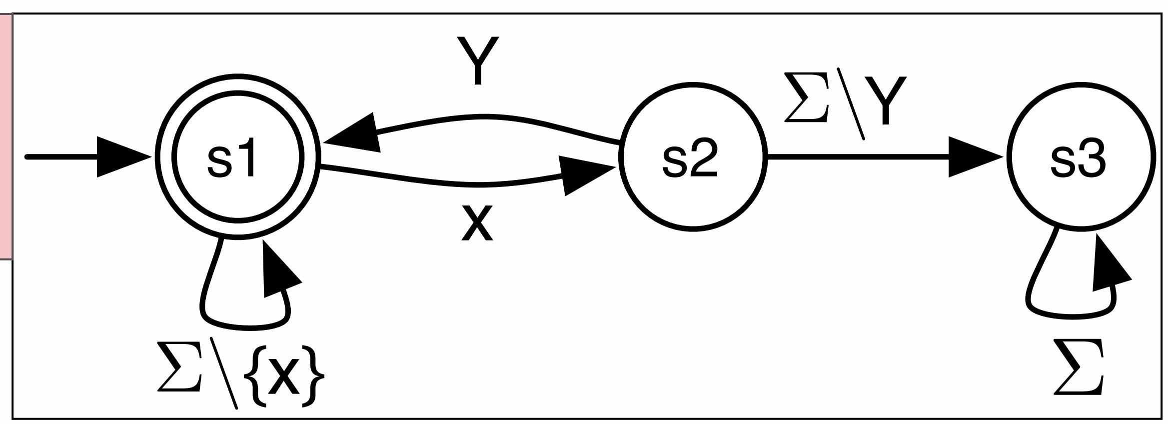
Expressing kTails(k=1)



“x can be immediately followed by one of Y”

=

FSM template:

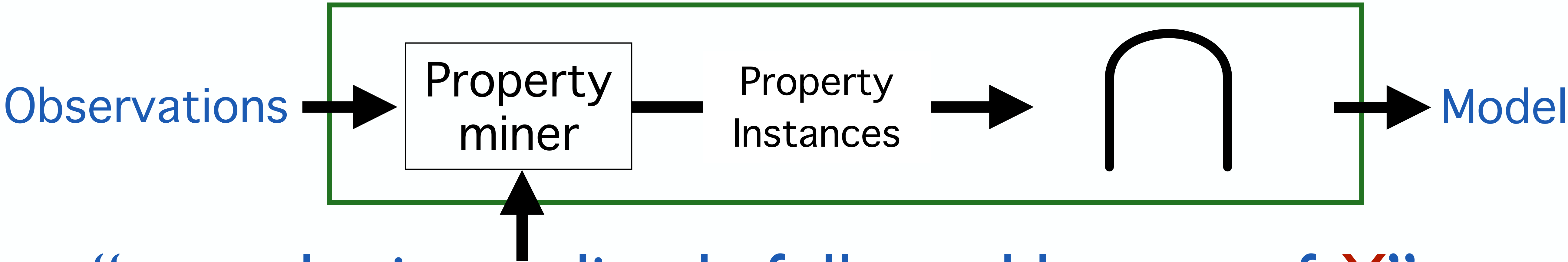


+

When to instantiate:

$$Eval(\text{Log } L, \langle x{=}a, Y{=}B\rangle) = \begin{cases} \text{true}: \forall t \in L, \exists b \in B,\ \Diamond(a \rightarrow \bigcirc b) \text{ in } t \wedge \\ \qquad\quad \forall b \in B, \exists t \in L,\ \Diamond(a \rightarrow \bigcirc b) \text{ in } t \\ \text{false}: \text{otherwise} \end{cases}$$

Expressing kTails(k=1)



"x can be immediately followed by one of Y"

=

FSM template:

Complete template

+

When to instantiate:

$$Eval(\text{Log } L, \langle x{=}a, Y{=}B\rangle) = \begin{cases} \text{true}: \forall t \in L, \exists b \in B,\ \Diamond(a \rightarrow \bigcirc b) \text{ in } t \wedge \\ \qquad \forall b \in B, \exists t \in L,\ \Diamond(a \rightarrow \bigcirc b) \text{ in } t \\ \text{false}: \text{otherwise} \end{cases}$$

Expressing kTails(k=1)

- This formulation is exact (identical models)
- Generalizes to arbitrary k value

Observ… …odel

"x can be immediately followed by one of Y"

=

FSM template:

Complete template

+

When to instantiate:

$$Eval(\text{Log } L, \langle x{=}a, Y{=}B\rangle) = \begin{cases} \text{true} : \forall t \in L, \exists b \in B,\ \Diamond(a \rightarrow \bigcirc b) \text{ in } t \ \wedge \\ \qquad\quad \forall b \in B, \exists t \in L,\ \Diamond(a \rightarrow \bigcirc b) \text{ in } t \\ \text{false} : \text{otherwise} \end{cases}$$

```java
package synoptic.algorithms;

import java.util.ArrayList;
import java.util.Collections;
import java.util.LinkedHashMap;
import java.util.LinkedHashSet;
import java.util.List;
import java.util.Map;
import java.util.Set;
import java.util.logging.Logger;

import synoptic.algorithms.graphops.PartitionMultiMerge;
import synoptic.model.ChainsTraceGraph;
import synoptic.model.Partition;
import synoptic.model.PartitionGraph;
import synoptic.model.event.EventType;
import synoptic.model.interfaces.INode;
import synoptic.model.interfaces.ITransition;
import synoptic.util.InternalSynopticException;
import synoptic.util.NotImplementedException;

/**
 * Implements the KTails algorithm as defined in Biermann & Feldman '72.
 */
public class KTails {
    public static Logger logger;
    static {
        logger = Logger.getLogger("KTails");
    }

    /**
     * Constructs and returns a PartitionGraph generated by applying kTails with
     * the given k value to the given trace graph
     */
    public static PartitionGraph performKTails(ChainsTraceGraph g, int k) {
        PartitionGraph pGraph = new PartitionGraph(g, false, null);
        attemptMerge(pGraph, k);
        return pGraph;
    }

    /**
     * Finds all possible merges in pGraph. Requires making a new call to
     * attemptMerge after every merge in case previously un-merge-able pairs
```

$\cap$ → Model

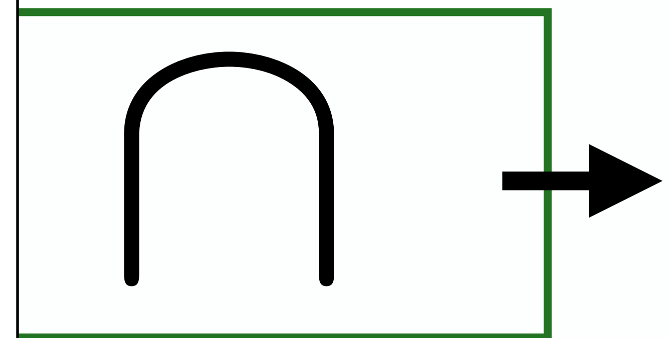
by one of Y”

Complete template

$\in L, \exists b \in B,\ \Diamond(a \rightarrow \bigcirc b)$ in $t \wedge$
$\in B, \exists t \in L,\ \Diamond(a \rightarrow \bigcirc b)$ in t
herwise

kTails Procedural description

```java
        for (ITransition<NodeType> t1 : n1Trans) {
            NodeType c1 = t1.getTarget();
            // Skip c1 if it was visited by this method earlier.
            if (visitedN1Children.contains(c1)) {
                continue;
            }

            boolean kEqual = false;

            // Make sure to get transitions of the same relation.
            for (ITransition<NodeType> t2 : n2
                    .getTransitionsWithExactRelations(t1.getRelation())) {
                NodeType c2 = t2.getTarget();

                // Skip c2 if it was visited by this method earlier.
                if (visitedN2Children.contains(c2)) {
                    continue;
                }

                // Skip c2 if its already been mapped to a c1 previously in the
                // outer loop.
                if (childKEquivMatches.contains(c2)) {
                    continue;
                }

                if (kEqualsWithoutSubsumption(c1, c2, k - 1, allVisitedMatches)) {
                    kEqual = true;
                    childKEquivMatches.add(c2);
                    break;
                }
            }
            // Could not find any kEqual c2 to match with c1.
            if (!kEqual) {
                // Remove the record of visiting n1 and n2.
                allVisitedMatches.remove(n1);
                return false;
            }
        }
        return true;
    }
}
```



Model

by one of Y”

Complete template

$\in L, \exists b \in B, \Diamond(a \to \bigcirc b)$ in $t \wedge$

$\in B, \exists t \in L, \Diamond(a \to \bigcirc b)$ in t

herwise

kTails Procedural description

Expressing kTails(k)

- This formulation is exact (identical models)
- Generalizes to arbitrary k value

Observ odel

“x can be immediately followed by one of Y”

How can we:

... get kTails to ignore certain k-length sequences?

... add the x AlwaysFollowedBy y invariant to kTails?

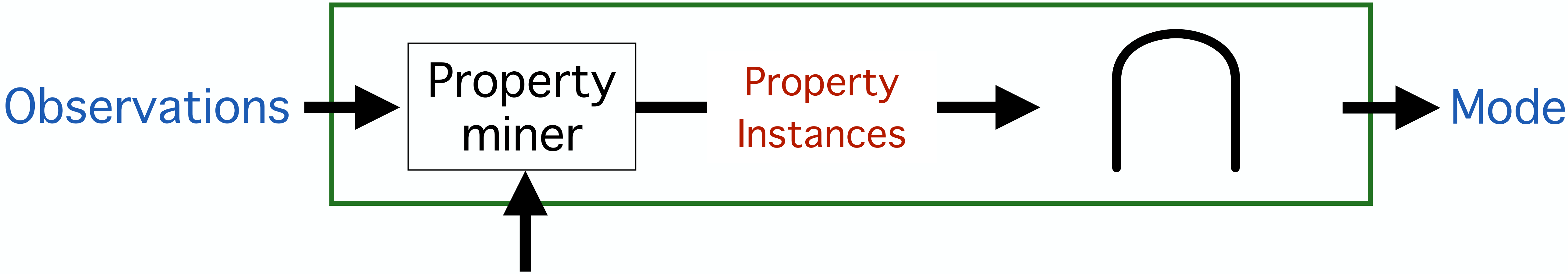
... make Synoptic deterministic?

... add a new kind of invariant to Synoptic?

... learn which properties kTails/Synoptic preserve?

Ev

Expressing kTails(k)



"x can be immediately followed by one of Y"

How can we:

... get kTails to ignore certain k-length sequences?

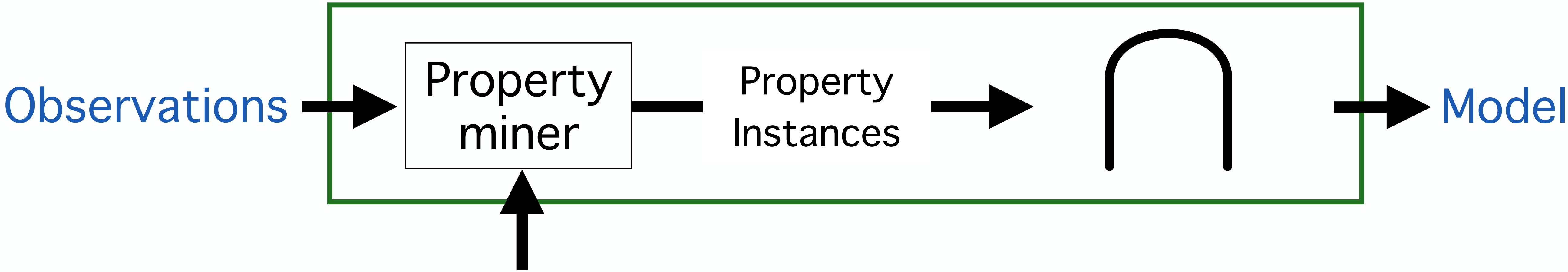
... add the x AlwaysFollowedBy y invariant to kTails?

... make Synoptic deterministic?

... add a new kind of invariant to Synoptic?

... learn which properties kTails/Synoptic preserve?

Expressing kTails(k)



"x can be immediately followed by one of Y"

How can we:

... get kTails to ignore certain k-length sequences? ✓

... add the x AlwaysFollowedBy y invariant to kTails?

... make Synoptic deterministic?

... add a new kind of invariant to Synoptic?

... learn which properties kTails/Synoptic preserve?

Expressing kTails(k)

"x can be immediately followed by one of Y"

How can we:

... get kTails to ignore certain k-length sequences? ✓

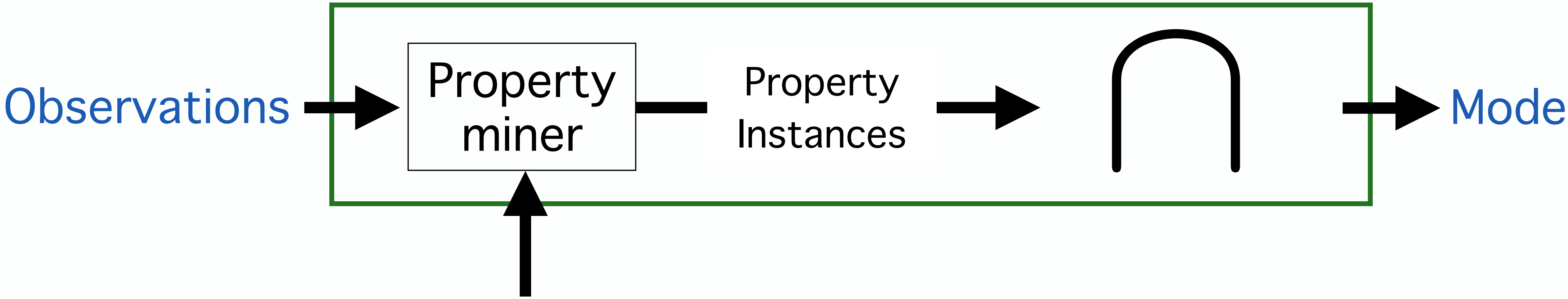
... add the x AlwaysFollowedBy y invariant to kTails?

... make Synoptic deterministic?

... add a new kind of invariant to Synoptic?

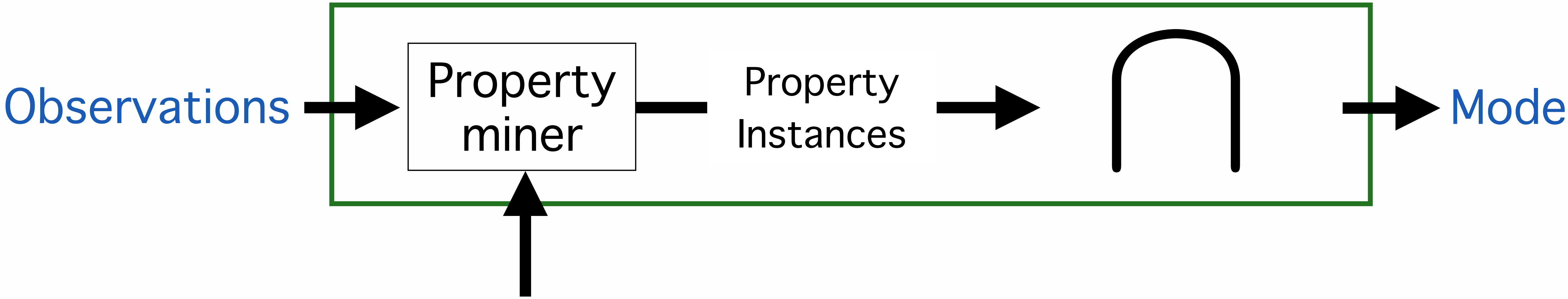
... learn which properties kTails/Synoptic preserve?

Expressing Synoptic with InvariMint



Synoptic property types

Expressing Synoptic with InvariMint



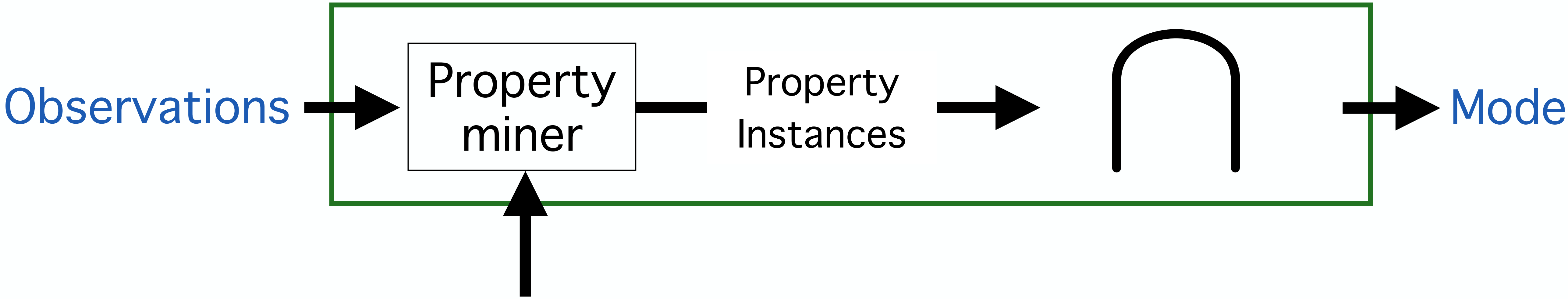
x AlwaysFollowedBy y

x AlwaysPrecedes y

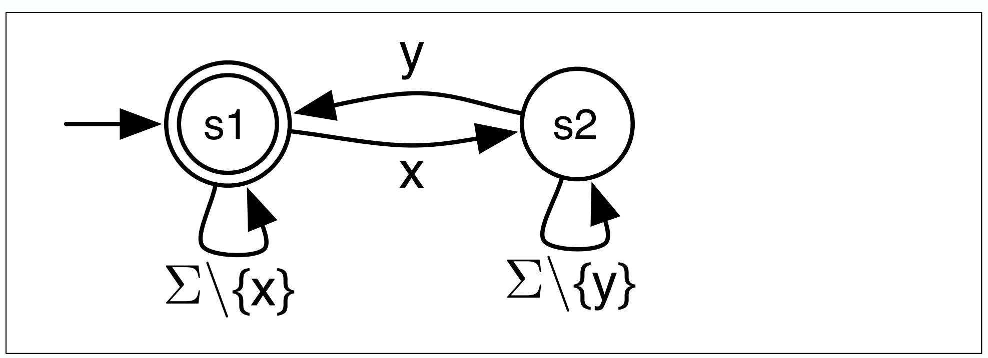
x NeverFollowedBy y

Synoptic invariants

Expressing Synoptic with InvariMint



x AlwaysFollowedBy y

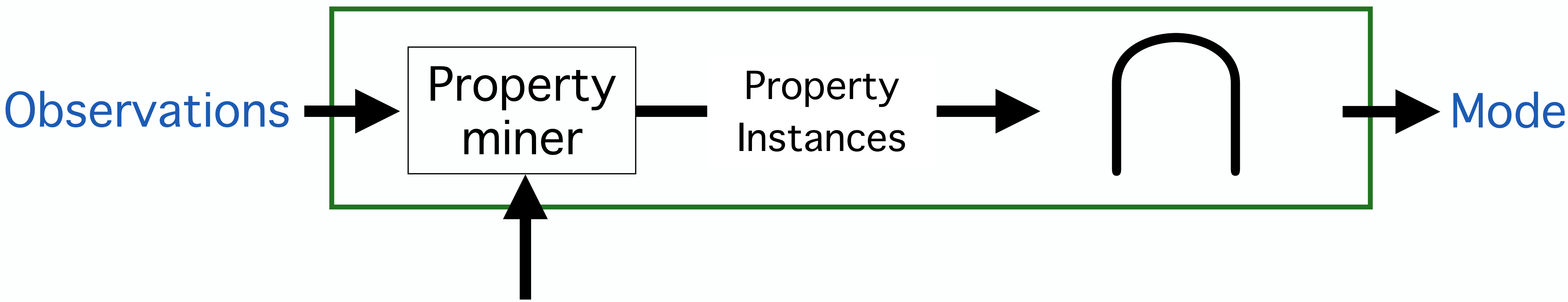


x AlwaysPrecedes y

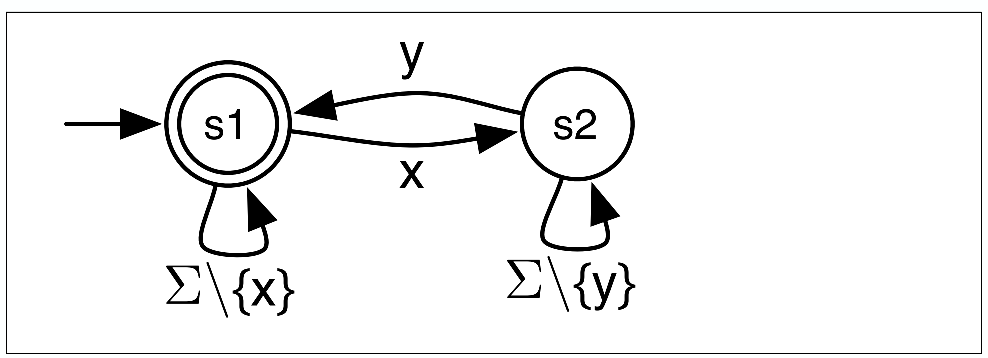
x NeverFollowedBy y

Synoptic invariants

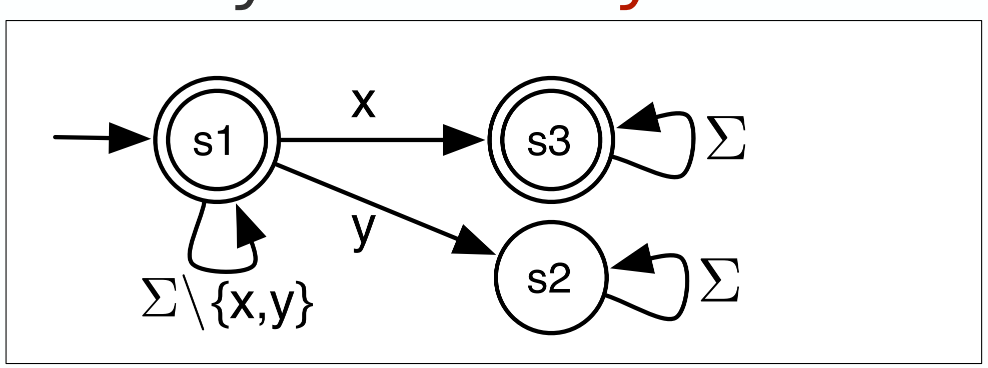
Expressing Synoptic with InvariMint



x AlwaysFollowedBy y



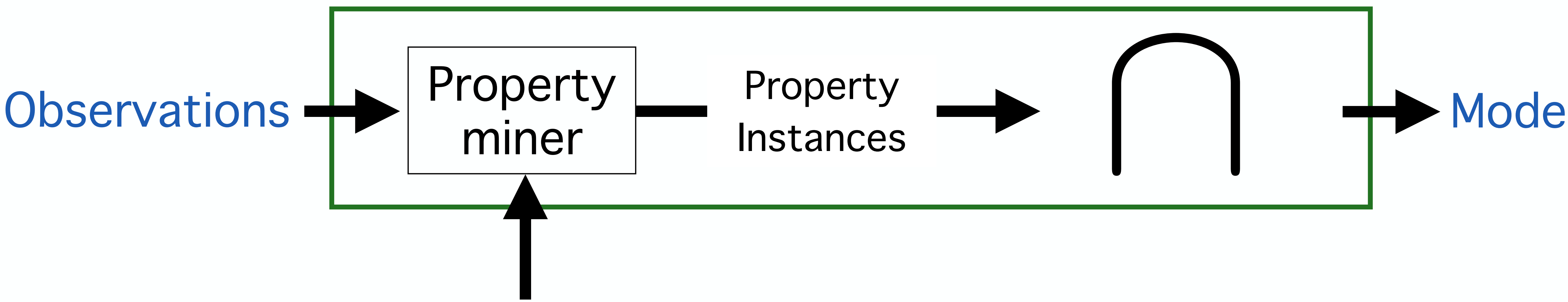
x AlwaysPrecedes y

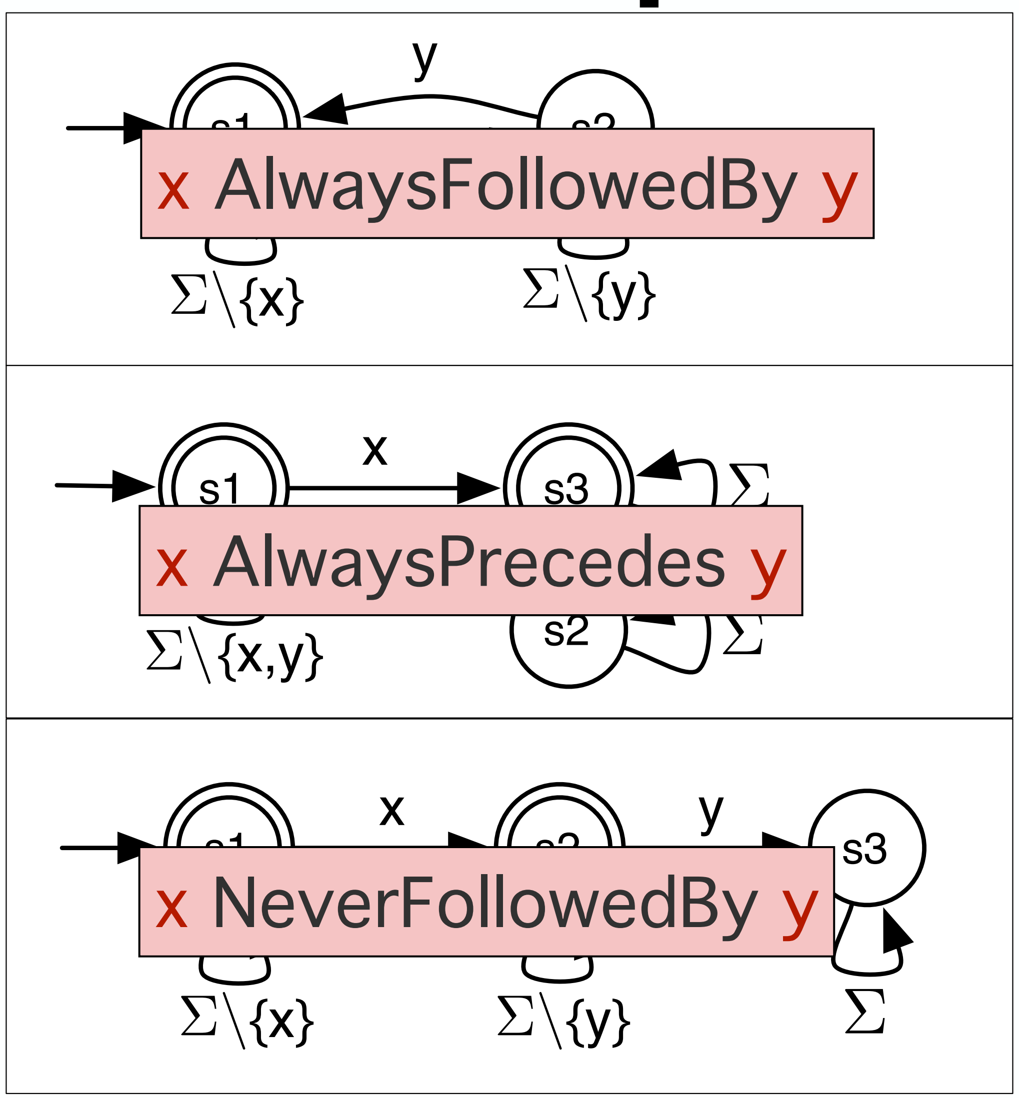


x NeverFollowedBy y

Synoptic invariants

Expressing Synoptic with InvariMint





Synoptic invariants

Expressing Synoptic with InvariMint

This formulation is approximate:

InvariMint model superset of Synoptic models

Obser

Synoptic invariants

Expressing Synoptic with InvariMint

How can we:

... get kTails to ignore certain k-length sequences? ✓

... add the x AlwaysFollowedBy y invariant to kTails?

... make Synoptic deterministic?

... add a new kind of invariant to Synoptic?

... learn which properties kTails/Synoptic preserve?

Expressing Synoptic with InvariMint

How can we:

... get kTails to ignore certain k-length sequences? ✓

... add the x AlwaysFollowedBy y invariant to kTails? ✓

... make Synoptic deterministic?

... add a new kind of invariant to Synoptic?

... learn which properties kTails/Synoptic preserve?

Expressing Synoptic with InvariMint

How can we:

... get kTails to ignore certain k-length sequences? ✓

... add the x AlwaysFollowedBy y invariant to kTails? ✓

... make Synoptic deterministic?

... add a new kind of invariant to Synoptic?

... learn which properties kTails/Synoptic preserve?

Expressing Synoptic with InvariMint

How can we:

... get kTails to ignore certain k-length sequences? ✓

... add the x AlwaysFollowedBy y invariant to kTails? ✓

... make Synoptic deterministic? ✓

... add a new kind of invariant to Synoptic?

... learn which properties kTails/Synoptic preserve?

Expressing Synoptic with InvariMint

How can we:

... get kTails to ignore certain k-length sequences? ✓

... add the x AlwaysFollowedBy y invariant to kTails? ✓

... make Synoptic deterministic? ✓

... add a new kind of invariant to Synoptic?

... learn which properties kTails/Synoptic preserve?

Expressing Synoptic with InvariMint

How can we:

... get kTails to ignore certain k-length sequences? ✓

... add the x AlwaysFollowedBy y invariant to kTails? ✓

... make Synoptic deterministic? ✓

... add a new kind of invariant to Synoptic? ✓

... learn which properties kTails/Synoptic preserve?

Expressing Synoptic with InvariMint

How can we:

... get kTails to ignore certain k-length sequences? ✓

... add the x AlwaysFollowedBy y invariant to kTails? ✓

... make Synoptic deterministic? ✓

... add a new kind of invariant to Synoptic? ✓

... learn which properties kTails/Synoptic preserve?

Expressing Synoptic with InvariMint

How can we:

... get kTails to ignore certain k-length sequences? ✓

... add the x AlwaysFollowedBy y invariant to kTails? ✓

... make Synoptic deterministic? ✓

... add a new kind of invariant to Synoptic? ✓

... learn which properties kTails/Synoptic preserve? ✓

Expressing Synoptic with InvariMint

How can we:

... get kT ... ences?

... add th ... o kTails?

... make S

... add a r

... learn which properties kTails/Synoptic preserve?

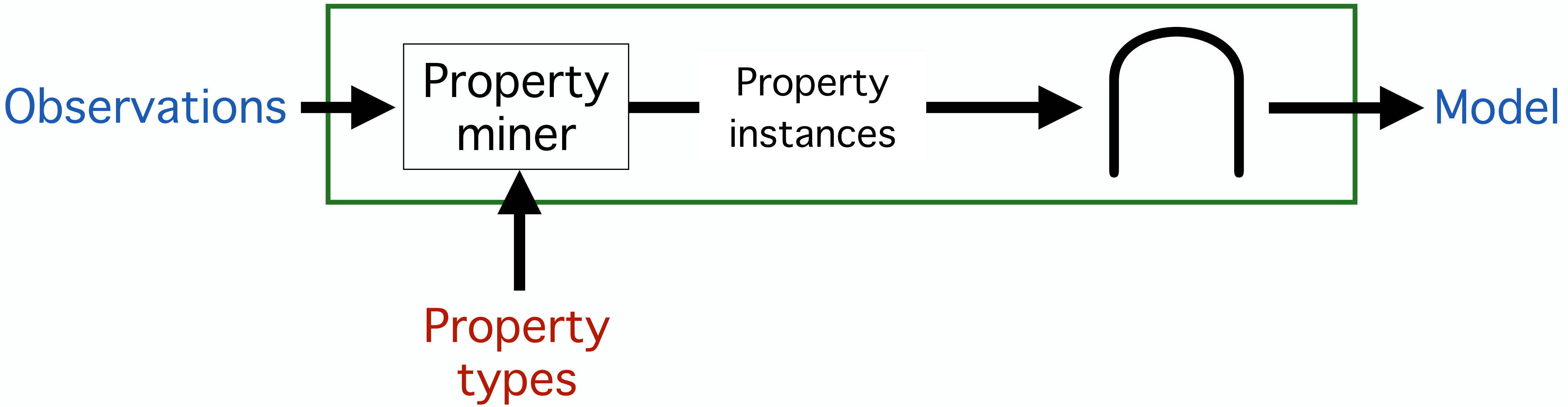
InvariMint advantages:

- ✓ Transparency
- ✓ Extensibility

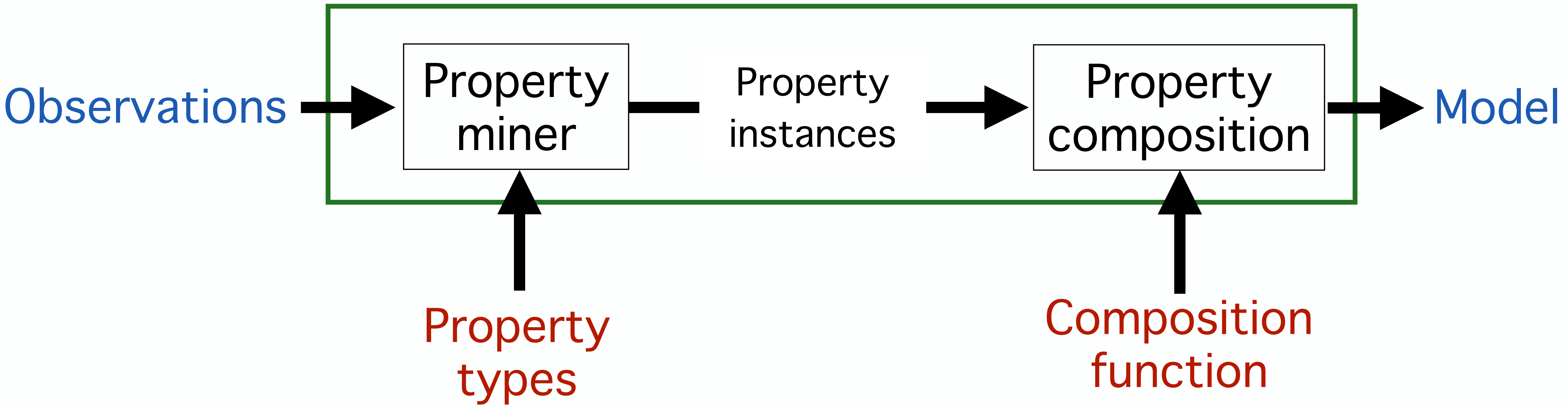
Outline

- FSM-inference algorithms not transparent/extensible
  - InvariMint -- modular and declarative FSM-inference
- Expressing algorithm using InvariMint
  - kTails -- a state-merging algorithm
  - Synoptic -- a state-splitting algorithm
- General specification formalism
- InvariMint evaluation

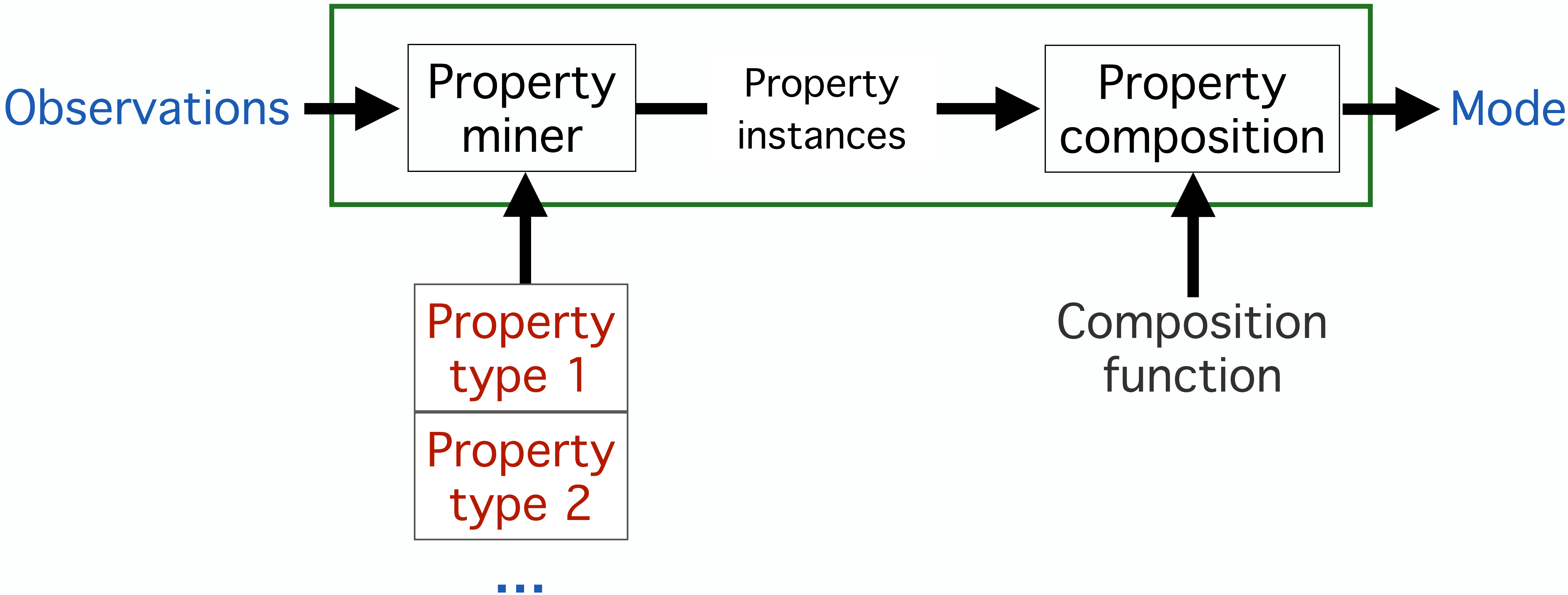
InvariMint specifications



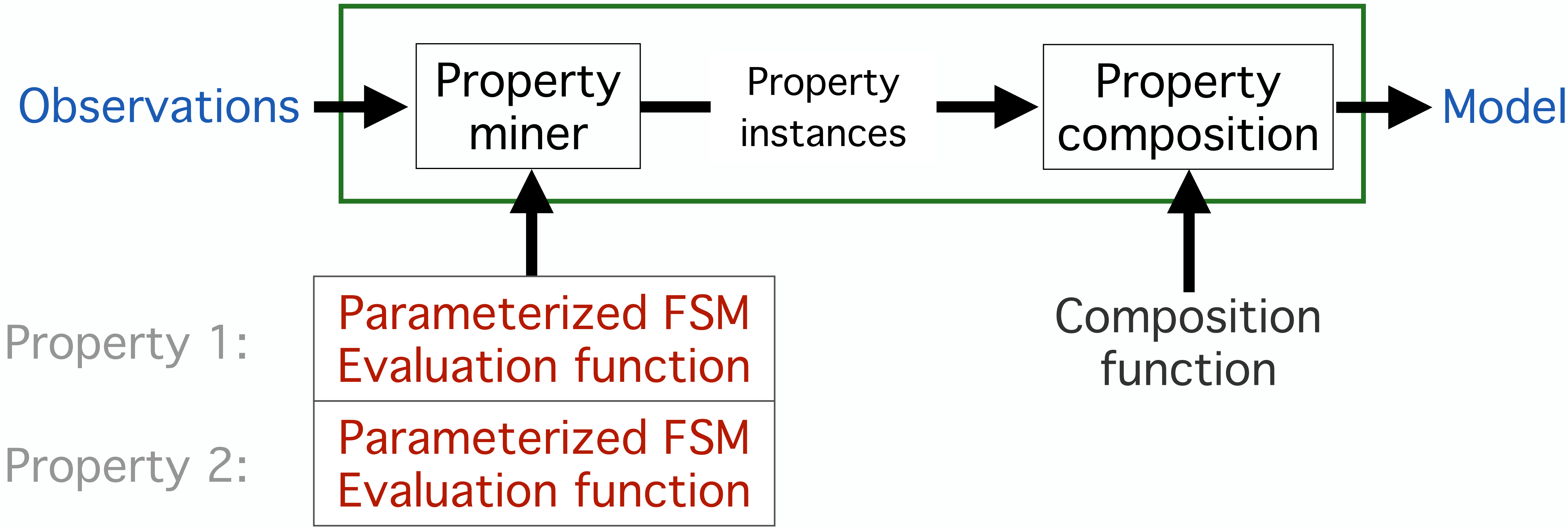
InvariMint specifications



InvariMint specifications

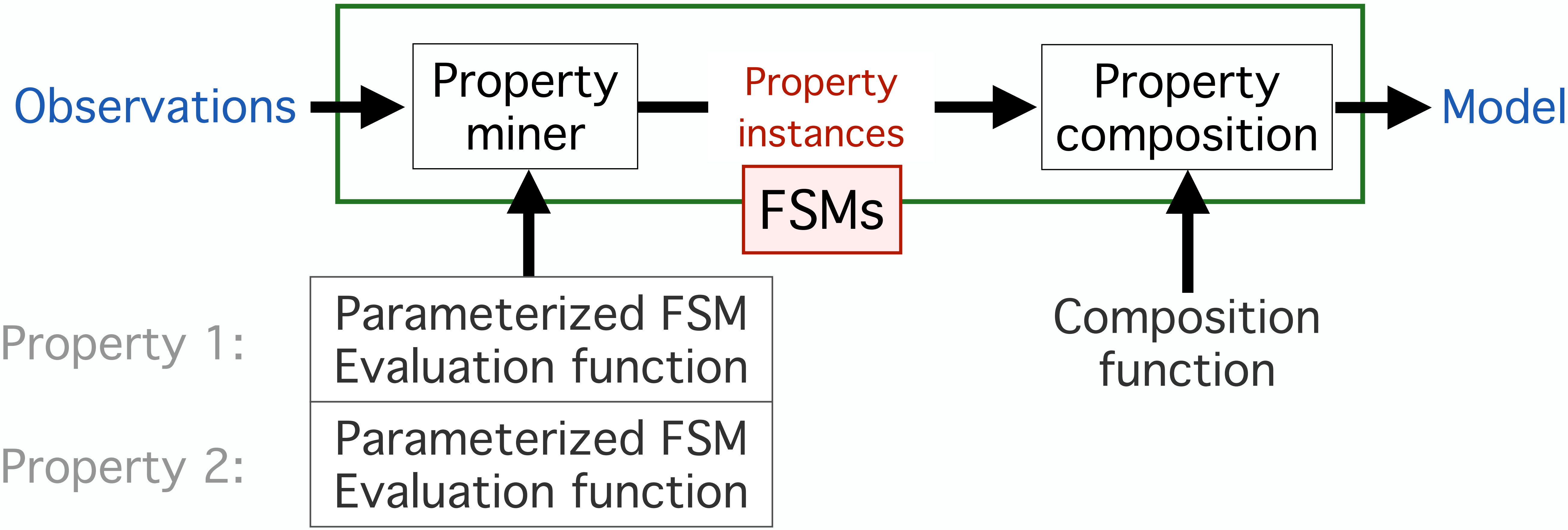


InvariMint specifications



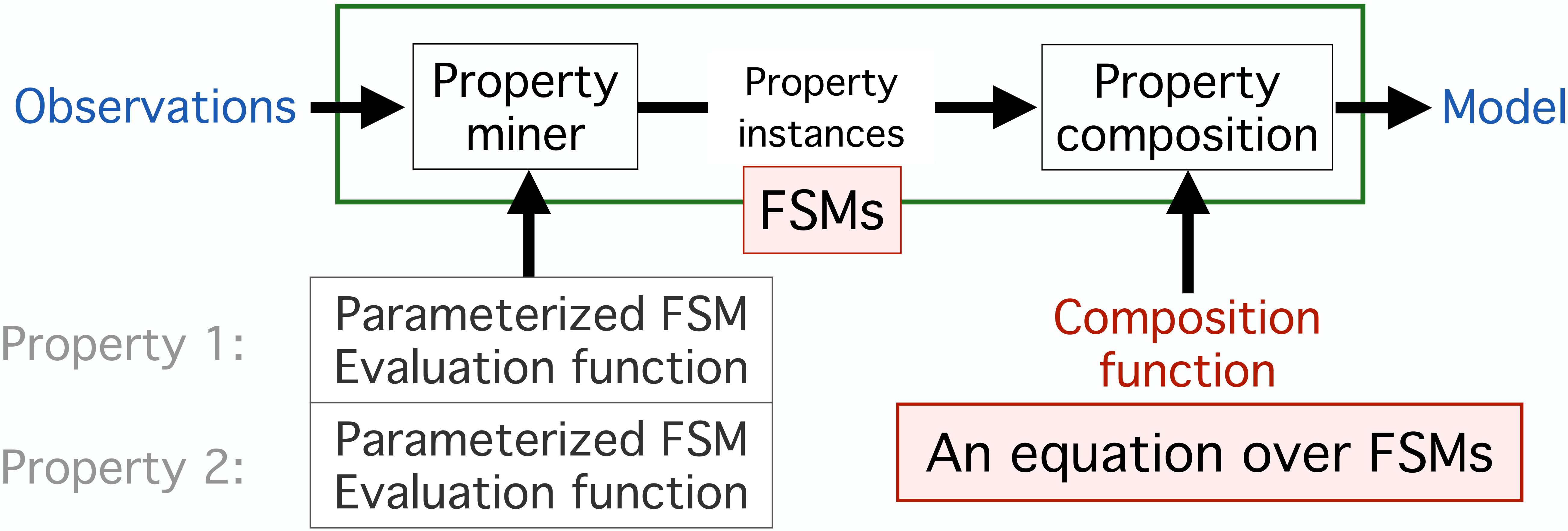
- Parameterized FSM (PFSM) -- property template
  - An FSM with variable-labeled transitions
- Evaluation function
  - Controls when a PFSM is instantiated as an FSM

InvariMint specifications



- Parameterized FSM (PFSM) -- property template
  - An FSM with variable-labeled transitions
- Evaluation function
  - Controls when a PFSM is instantiated as an FSM

InvariMint specifications



- Parameterized FSM (PFSM) -- property template
  - An FSM with variable-labeled transitions
- Evaluation function
  - Controls when a PFSM is instantiated as an FSM

Outline

- FSM-inference algorithms not transparent/extensible
  - InvariMint -- modular and declarative FSM-inference
- Expressing algorithm using InvariMint
  - kTails -- a state-merging algorithm
  - Synoptic -- a state-splitting algorithm
- General specification formalism
- InvariMint evaluation

InvariMint expressiveness

- FSM-inference algorithms from prior work
  - Expressed kTails and Synoptic
  - Useful for understanding both algorithms
  - Noticed overlap in their InvariMint specifications!
- What can and can't we express with InvariMint?
  - Limited to FSM algorithms that consider temporality
  - Constrains inference to composition of properties
  - Benefits from a formal background

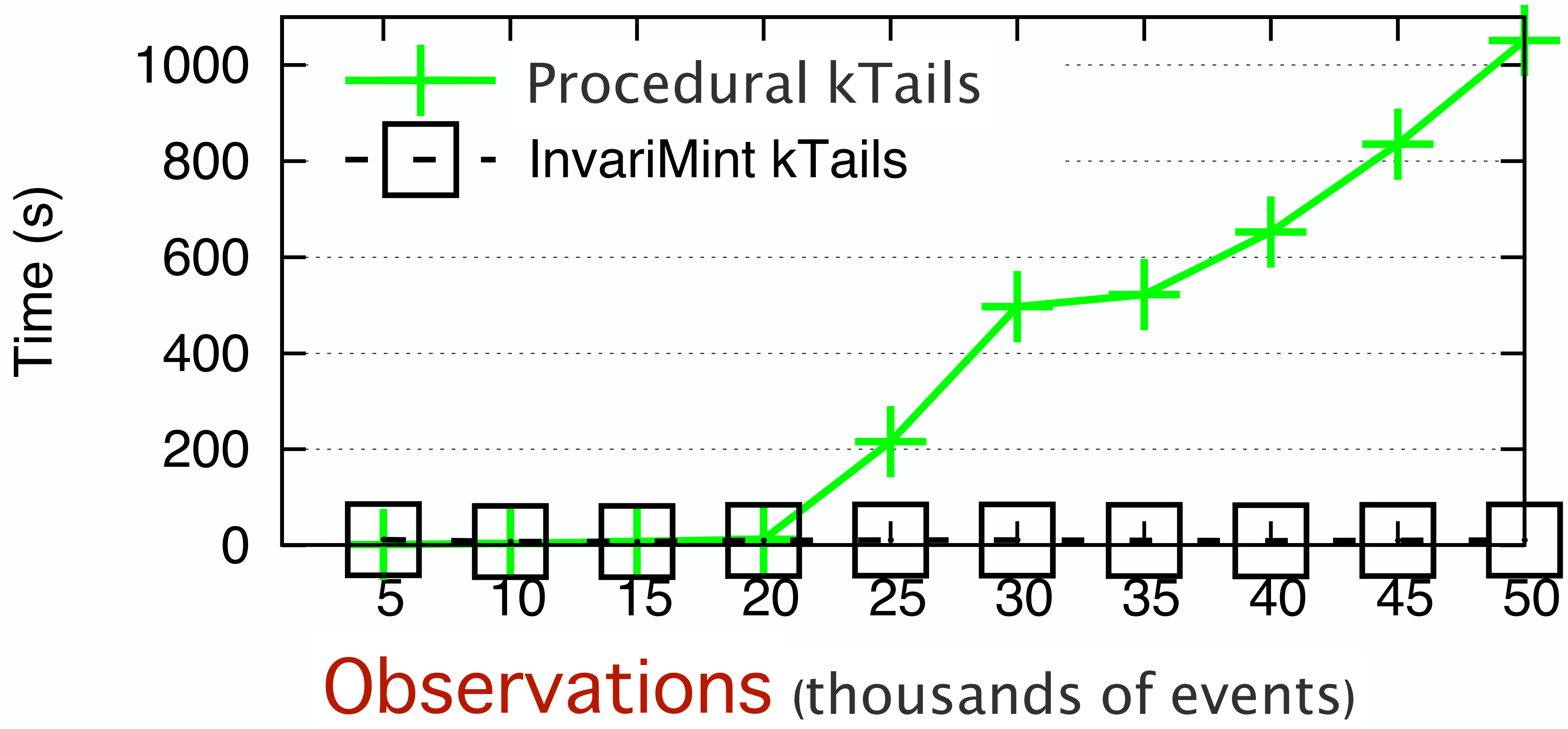
InvariMint performance

- FSM inference useful for large systems
    - Compact representation of mass executions
    - A node at Google generates a million messages/day

- Does transparency and extensibility impact performance?
    - InvariMint versions of kTails and Synoptic over 100x faster than procedural variants

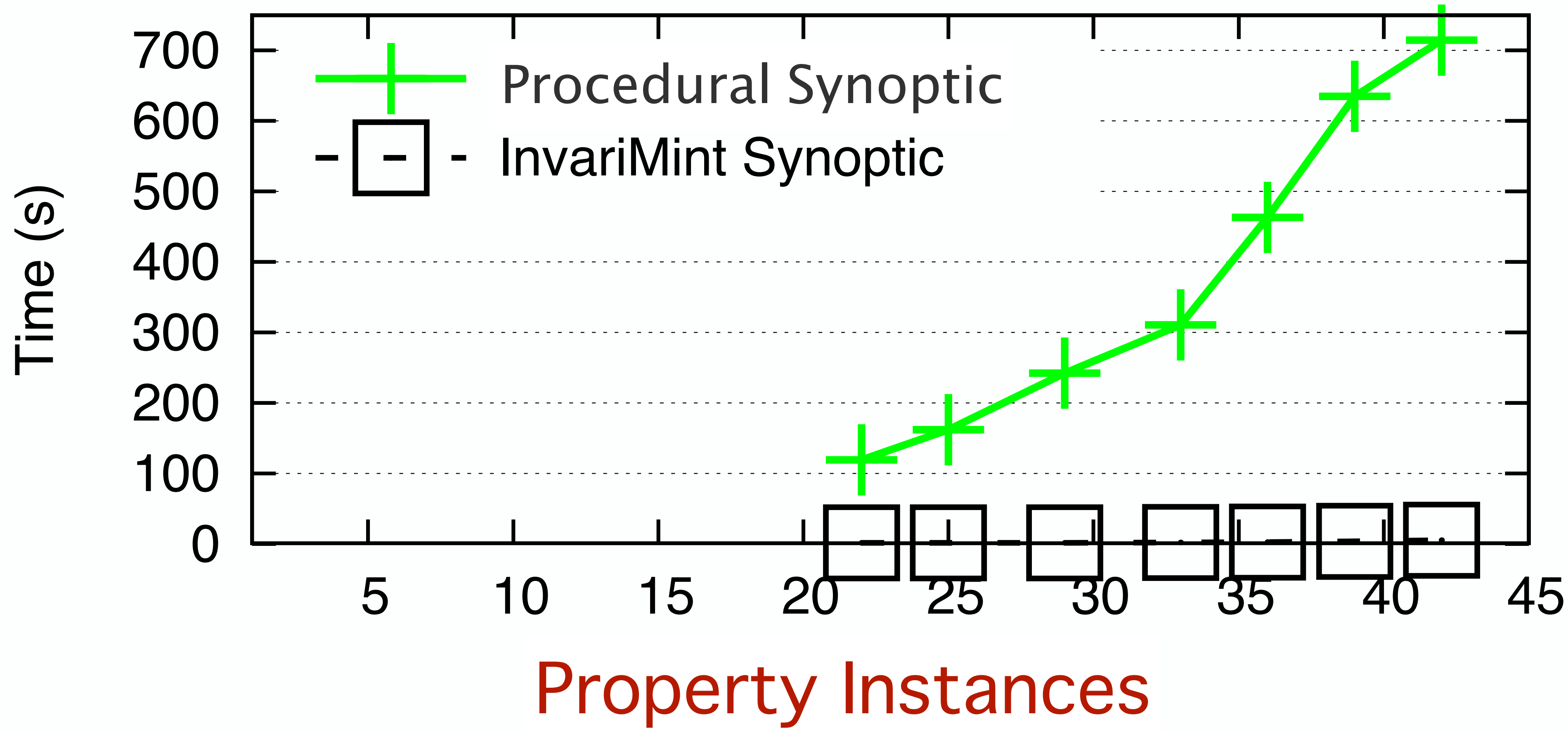
Xu et al. SLAML 2010

Performance: kTails



- InvariMint kTails much faster than procedural kTails
  - Efficient FSM composition (e.g., intersection)
  - The full log is never maintained in memory

Performance: Synoptic

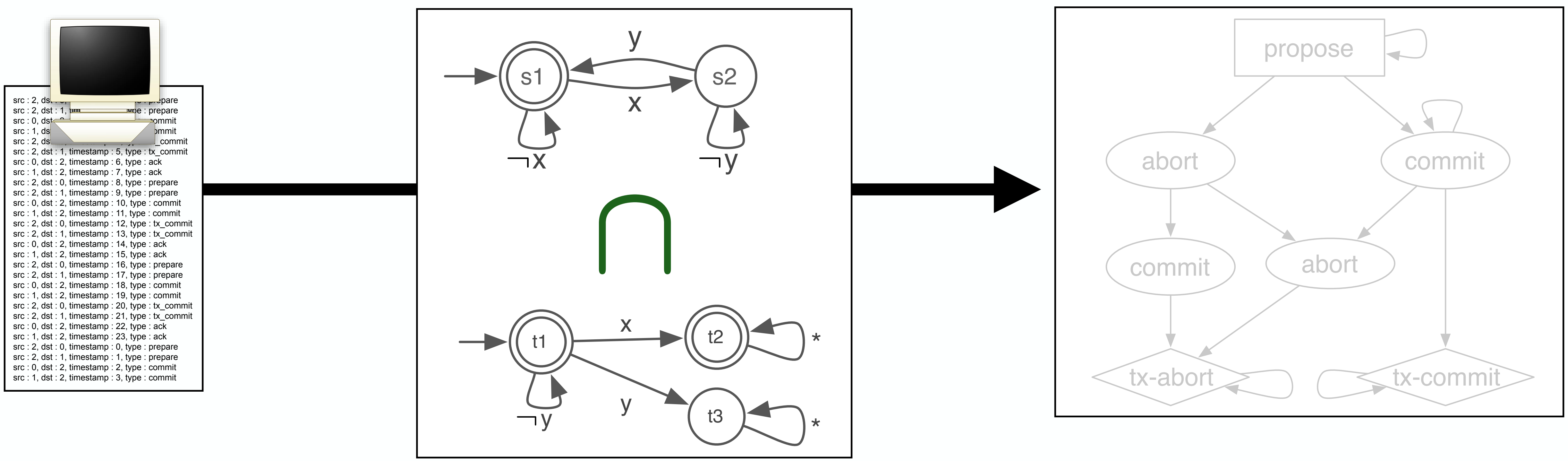


- Procedural Synoptic much slower than InvariMint Synoptic
  - Synoptic's modeling check invariants
  - Synoptic's refinement deals with concrete event instances

Contributions

FSM-inference algorithms are not transparent/extensible

InvariMint



Open source synoptic.googlecode.com

Contributions

FSM-inference algorithms are not transparent/extensible

InvariMint

Declarative specification of FSM-inference algorithms

- Provides insight into how an algorithms works
- A common language for inference algorithms
- Simplifies extension of existing algorithms
- Faster than procedural equivalents

Open source synoptic.googlecode.com