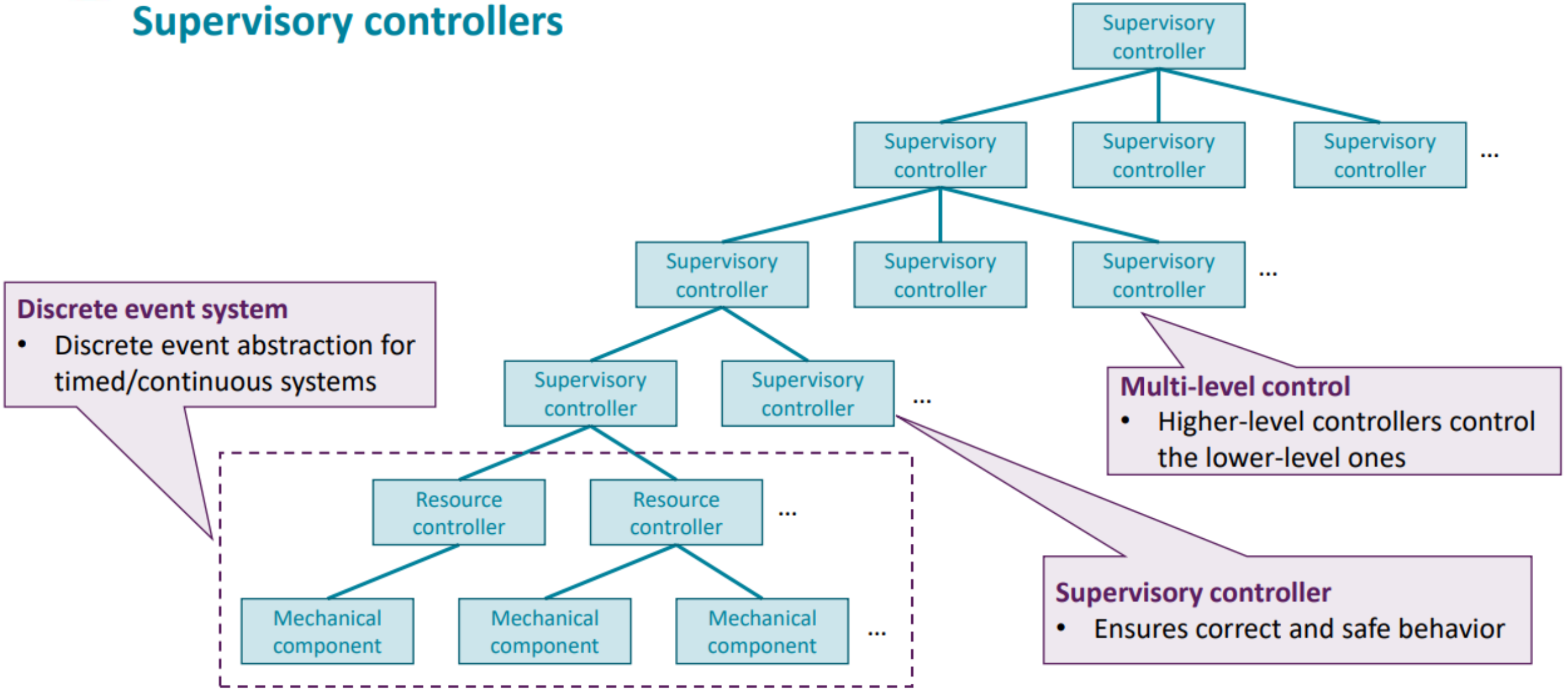
Overview and Performance Evaluation of Supervisory Controller Synthesis with Eclipse ESCET v4.0

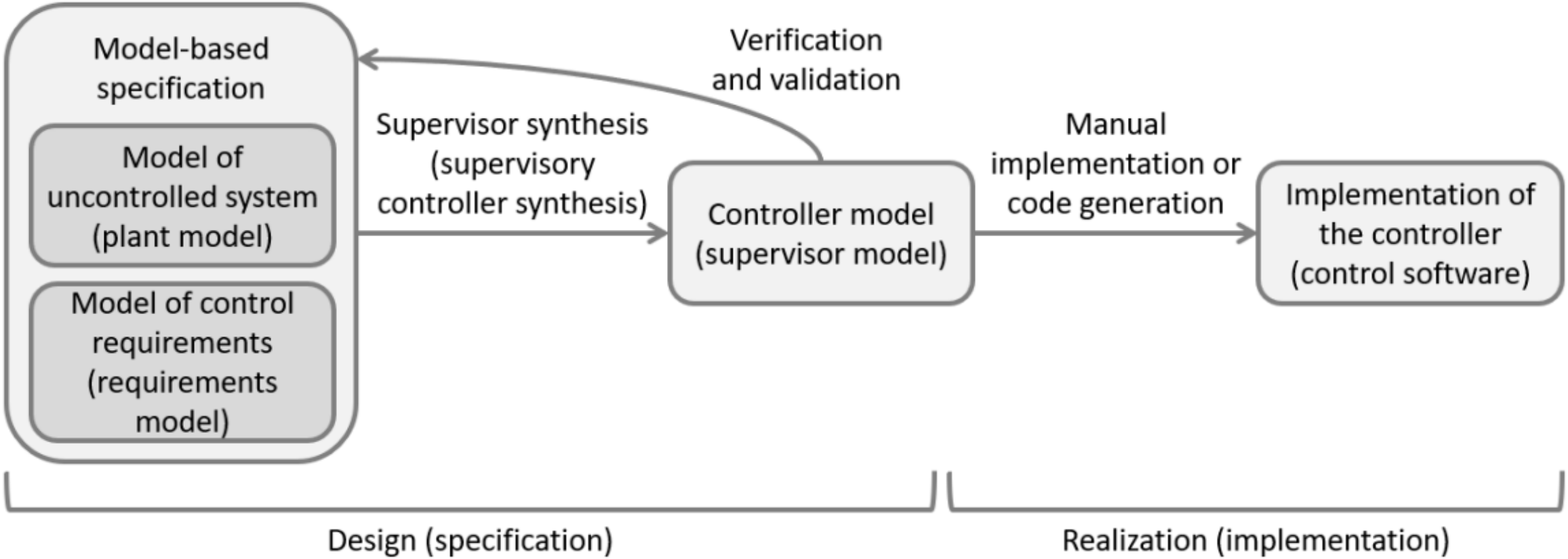
Dennis Hendriks, Michel Reniers, Wan Fokkink, and Wytse Oortwijna

(Presentation: Ivo Melse)

Supervisory controllers



The Synthesis-Based Engineering process



(Simplified process.)

Why is synthesis useful?

- Reduce implementing a system to providing
  - (formal) **requirements** and
  - **Plants** (unsupervised system behaviour)
- More secure
- Easier

ESCET

- Toolkit for SBE
- In Eclipse
- Modelling language CIF

Objective

- ESCET model synthesis already existed but was slow
- Needed improvement

Proposal

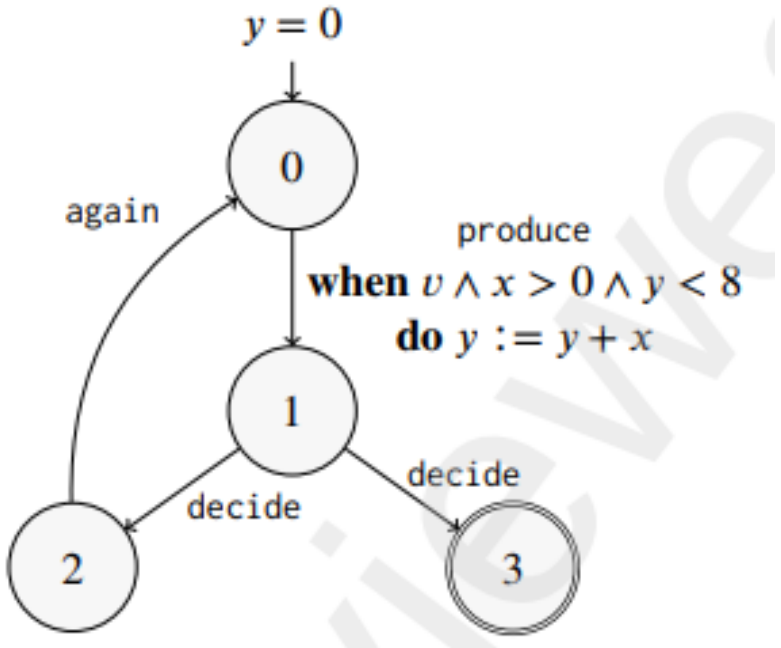
- Optimizations to algorithm in ESCET v0.8 to v4.0
- Overview for improving algorithm

Synthesis algorithm: input

- **Requirements**: desired properties
- **Plants**: unsupervised system
- EFA
- Specified in CIF language
- E.g. 5 plant automata and 3 reqs

EFA

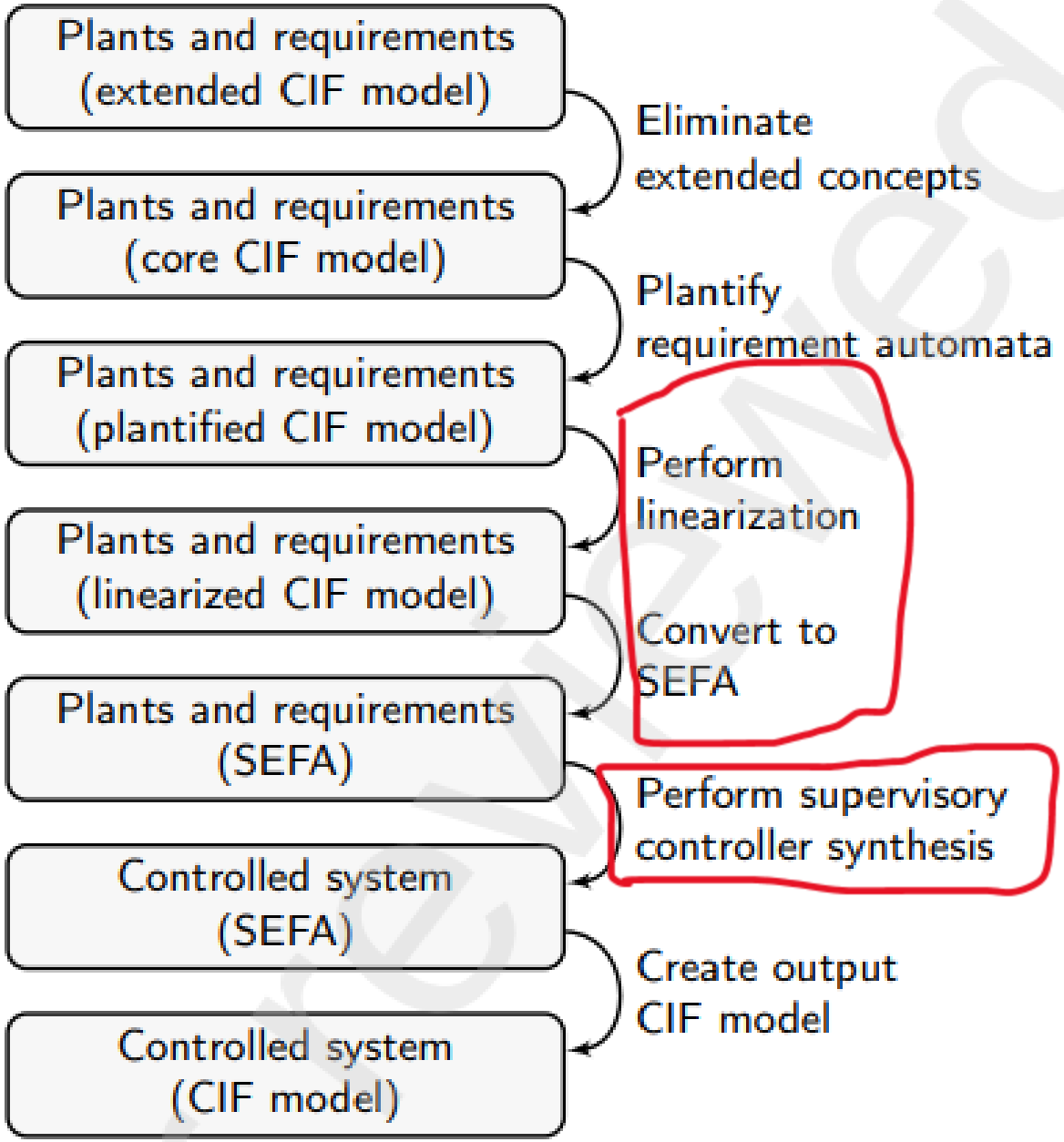
- Locations
- Variables
- States = locations + var. values
- EFAs sync on **events**
  - **Controllable** or **uncontrollable**
- Guard
- Update



Synthesis algorithm: input

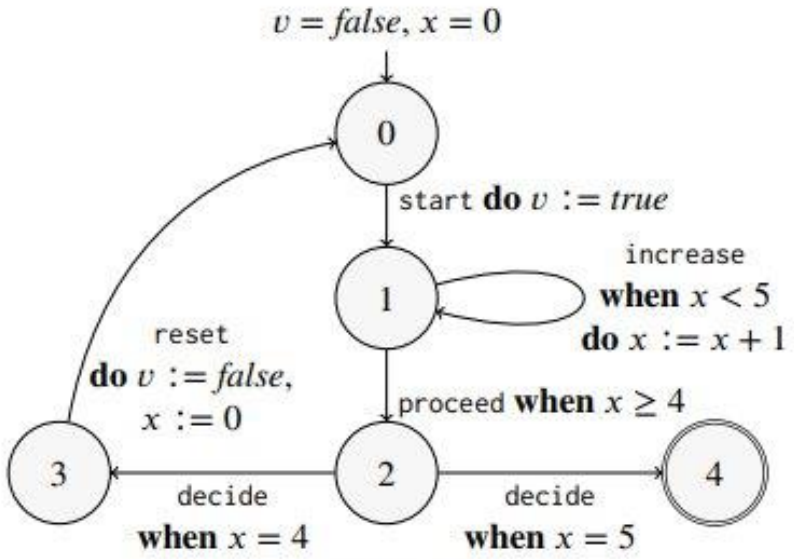
- **Marked** states q_m must be reachable (non-blocking)
  - Prevent "safe but useless"
- Set of **forbidden** states q_f
  - Guarantee safety

Synthesis algorithm

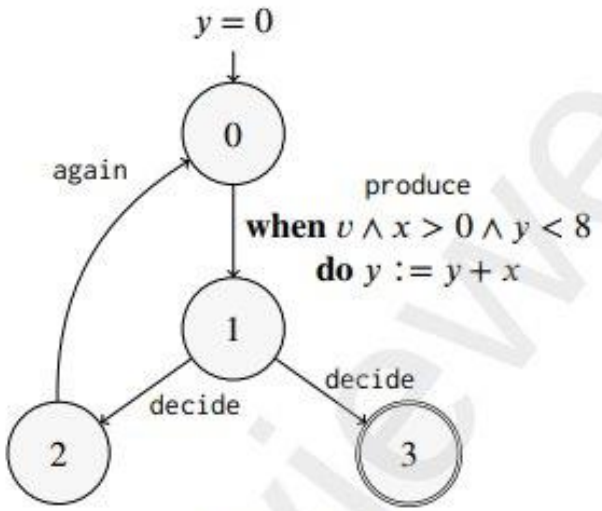


Linearization

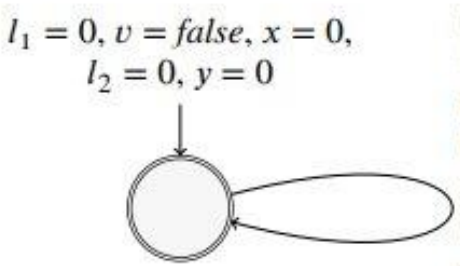
- Introduce vars for locations



(a) First EFA



(b) Second EFA



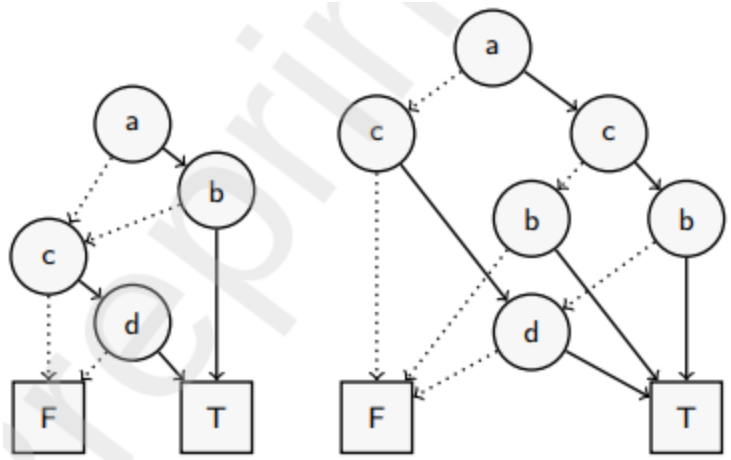
start **when** $l_1 = 0$ **do** $l_1 := 1, v := true$
increase **when** $l_1 = 1 \wedge x < 5$ **do** $x := x + 1$
proceed **when** $l_1 = 1 \wedge x \geq 4$ **do** $l_1 := 2$
produce **when** $l_2 = 0 \wedge v \wedge x > 0 \wedge y < 8$ **do** $l_2 := 1, y := y + x$
decide **when** $l_1 = 2 \wedge l_2 = 1 \wedge x = 4$ **do** $l_1 := 3, l_2 := 2$
decide **when** $l_1 = 2 \wedge l_2 = 1 \wedge x = 4$ **do** $l_1 := 3, l_2 := 3$
decide **when** $l_1 = 2 \wedge l_2 = 1 \wedge x = 5$ **do** $l_1 := 4, l_2 := 2$
decide **when** $l_1 = 2 \wedge l_2 = 1 \wedge x = 5$ **do** $l_1 := 4, l_2 := 3$
reset **when** $l_1 = 3$ **do** $l_1 := 0, v := false, x := 0$
again **when** $l_2 = 2$ **do** $l_2 := 0$

(c) Linearized EFA

Figure 4: Example of two EFAs and their linearized form.

Symbolic representation

- Predicates instead of sets of states
- ROBDDs



$$V = \{l_1, v, x, l_2, y\}$$
$$D = l_1 \to \mathbb{N}_{0..4}, v \to \mathbb{B}, x \to \mathbb{N}_{0..5}, l_2 \to \mathbb{N}_{0..3}, y \to \mathbb{N}_{0..12}$$
$$\Sigma = \{\texttt{start}, \texttt{increase}, \texttt{proceed}, \texttt{decide}, \texttt{reset}, \texttt{produce}, \texttt{again}\}$$
$$E = \{e_1, e_2, e_3, e_4, e_5, e_6, e_7, e_8, e_9, e_{10}\}$$
$$e_1 = (l_1 = 0,\ \mathit{false},\ \texttt{start},\ l_1^+ = 1 \wedge v^+)$$
$$e_2 = (l_1 = 1 \wedge x < 5,\ x + 1 > 7,\ \texttt{increase},\ x^+ = x + 1)$$
$$e_3 = (l_1 = 1 \wedge x \geq 4,\ \mathit{false},\ \texttt{proceed},\ l_1^+ = 2)$$
$$e_4 = (l_2 = 0 \wedge v \wedge x > 0 \wedge y < 8,\ y + x > 15,\ \texttt{produce},\ l_2^+ = 1 \wedge y^+ = y + x)$$
$$e_5 = (l_1 = 2 \wedge l_2 = 1 \wedge x = 4,\ \mathit{false},\ \texttt{decide},\ l_1^+ = 3 \wedge l_2^+ = 2)$$
$$e_6 = (l_1 = 2 \wedge l_2 = 1 \wedge x = 4,\ \mathit{false},\ \texttt{decide},\ l_1^+ = 3 \wedge l_2^+ = 3)$$

$$e_7 = (l_1 = 2 \wedge l_2 = 1 \wedge x = 5,\ \mathit{false},\ \texttt{decide},\ l_1^+ = 4 \wedge l_2^+ = 2)$$
$$e_8 = (l_1 = 2 \wedge l_2 = 1 \wedge x = 5,\ \mathit{false},\ \texttt{decide},\ l_1^+ = 4 \wedge l_2^+ = 3)$$
$$e_9 = (l_1 = 3,\ \mathit{false},\ \texttt{reset},\ l_1^+ = 0 \wedge \neg v^+ \wedge x^+ = 0)$$
$$e_{10} = (l_2 = 2,\ \mathit{false},\ \texttt{again},\ l_2^+ = 0)$$
$$p_0 = l_1 = 0 \wedge \neg v \wedge x = 0 \wedge l_2 = 0 \wedge y = 0$$
$$p_m = l_1 = 4 \wedge l_2 = 3$$

Synthesis

Keep set of 'good' states **C**

Start: **C** = allowed states = $\neg p_f$

repeat

Remove blocking states

Remove states that "can escape from" **C** uncontrolled

(Optional) remove unreachable states

End once **C** no longer changes (fixed point)

Finally, extract the **supervisor** from **C**

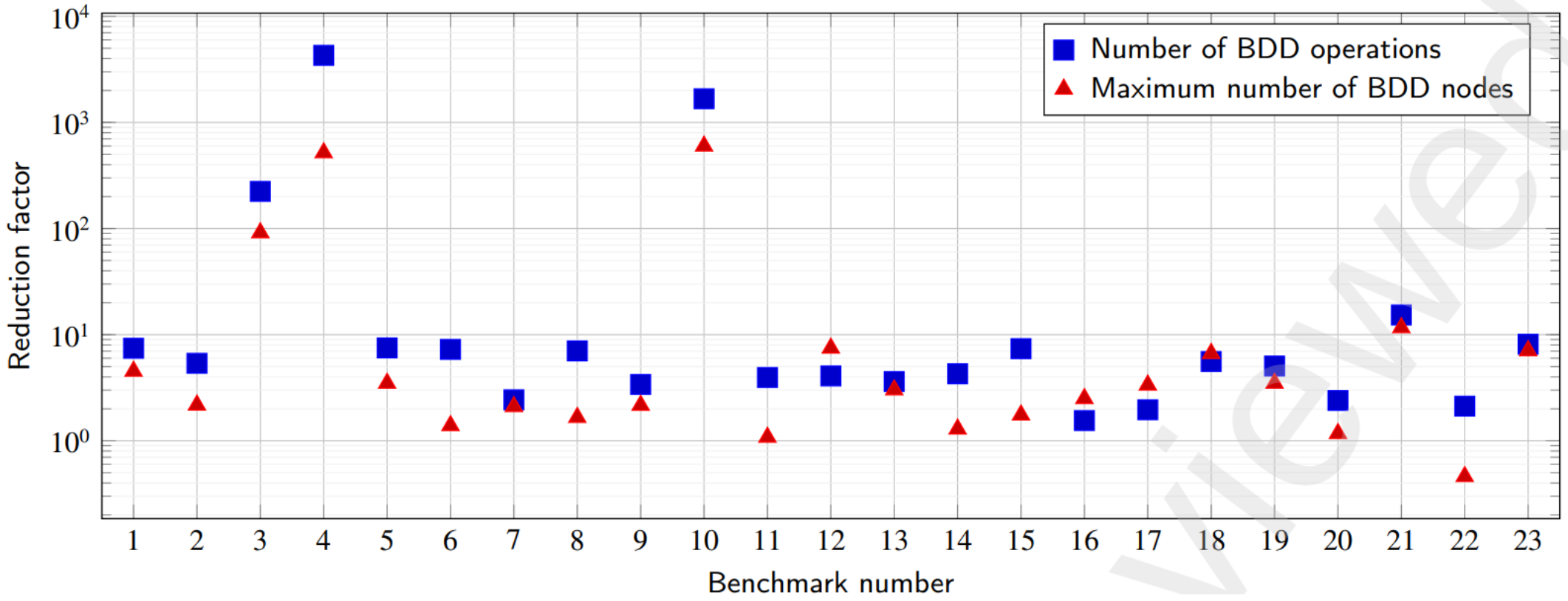
Improvements

- Variable ordering for BDDs
- Early fixed point detection
- (and 3 more)
- Used in v4.0

Evidence

- Set of 23 benchmarks
- measure performance of v0.8 vs. v4.0
- *Industry* benchmarks: lithography, waterlock
- *Academic* (i.e. silly) dining philosophers, cat and mouse

Evidence



Conclusion

- Improvements are effective

Influential previous work

- Early Automata and Control Theory (1960s–1970s)
  - Church, Kalman, Büchi, Rosen
- Ramadge & Wonham (1980s)
  - Framework for supervisory controller synthesis
- Ouedraogo *et al.* (2012)
  - Created the supervised synthesis algorithm

Impact

- ESCET is used in practice
- Paper: unkown – still in review

Writing

- Accessible
- Could have used more examples

Future work

- Scalability (exponential)
- Non-monolithic synthesis – multiple controllers at once
  - Multi-level synthesis (hierarchy)

Questions?