# Parameter Synthesis for Timed Automata with Clock-Aware LTL Properties

**Nikola Beneš**

*joint work with*
Peter Bezděk, Ivana Černá, Vojtěch Havel, Jiří Barnat







Masaryk University
Brno, Czech Republic

April 7, 2019

# Based on

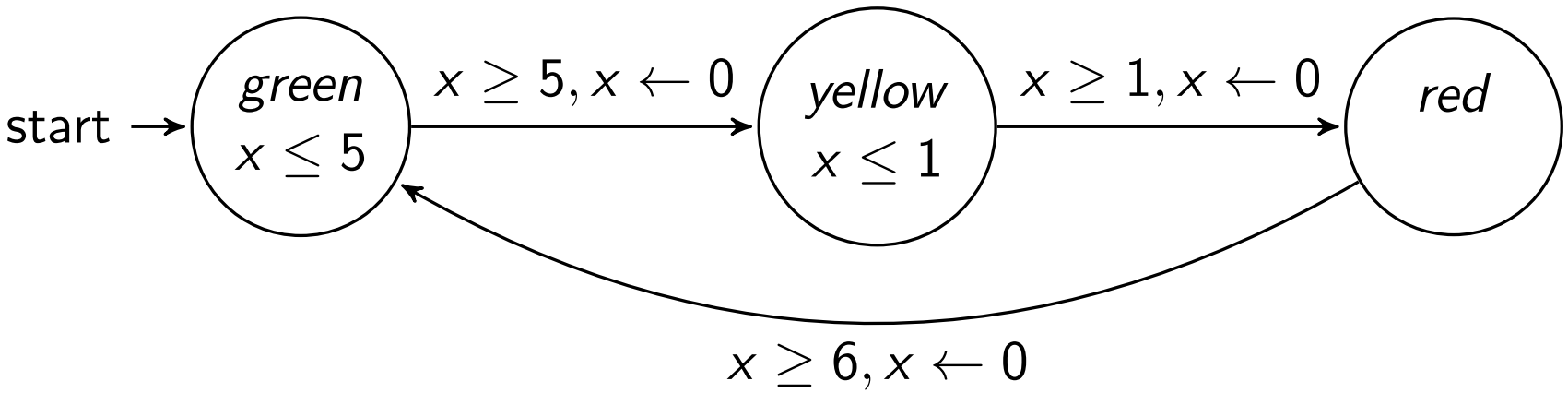


P. Bezděk, N. Beneš, J. Barnat, and I. Černá. LTL Parameter Synthesis of Parametric Timed Automata. In *Software Engineering and Formal Methods, SEFM 2016*, volume 9763 of *LNCS*, pages 172–187. Springer, 2016.

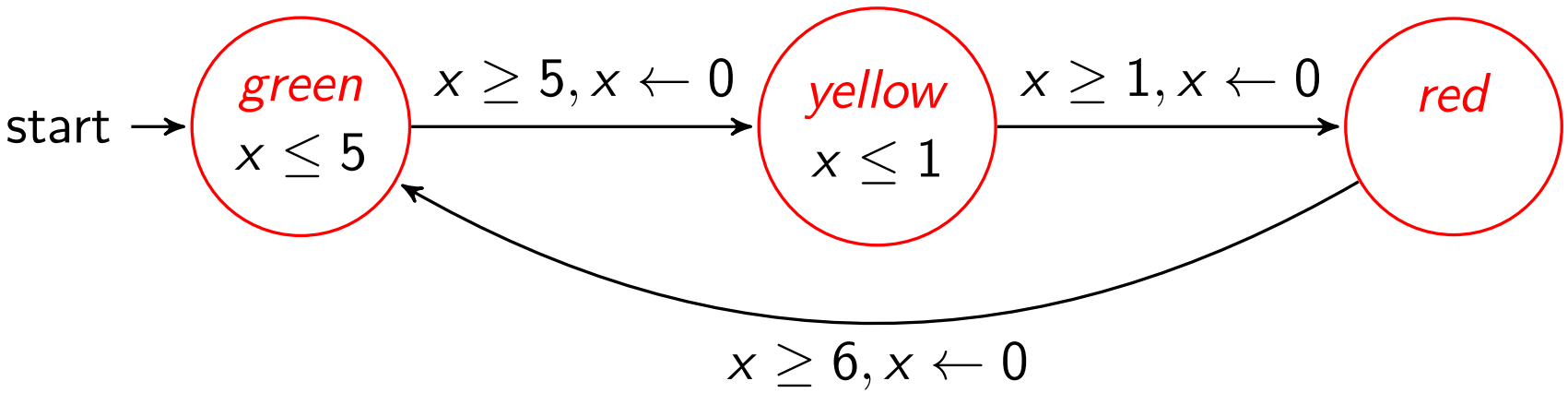

P. Bezděk, N. Beneš, V. Havel, J. Barnat, and I. Černá. On Clock-Aware LTL Properties of Timed Automata. In *Theoretical Aspects of Computing – ICTAC 2014*, volume 8687 of *LNCS*, pages 43–60. Springer, 2014.


P. Bezděk, N. Beneš, J. Barnat, and I. Černá. On Clock-Aware LTL Parameter Synthesis of Timed Automata. *Journal of Logical and Algebraic Methods in Programming (JLAMP)* 99, pages 114–142. Elsevier, 2018.

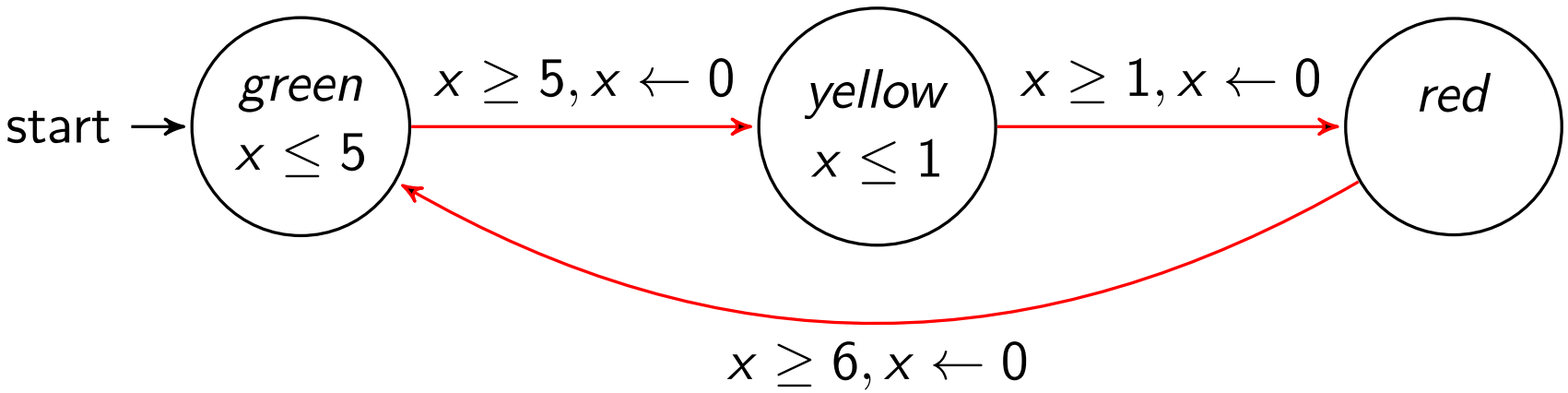
# Parametric Timed Automata



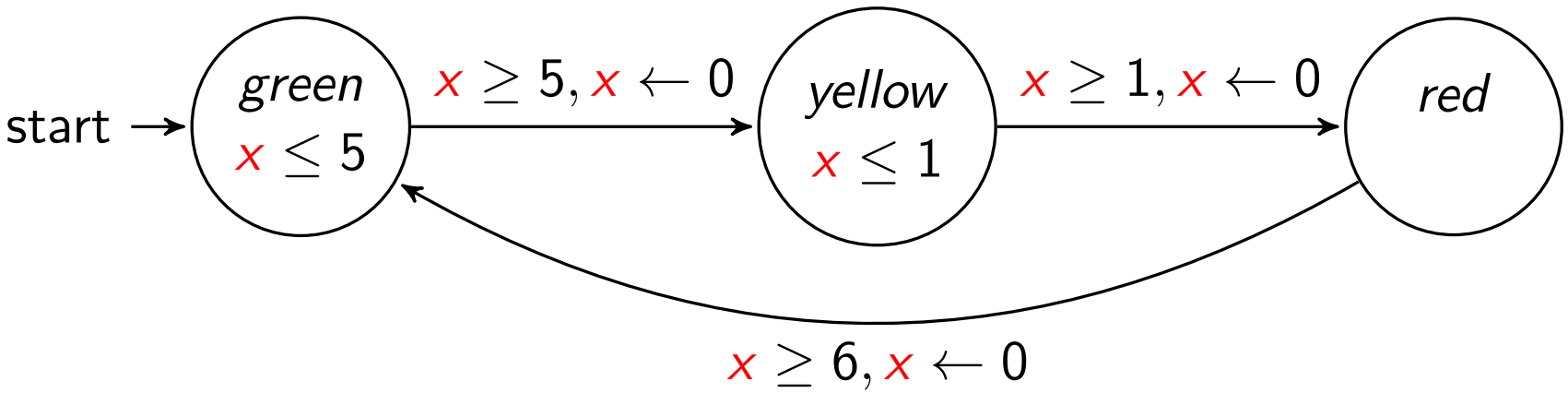
# Parametric Timed Automata



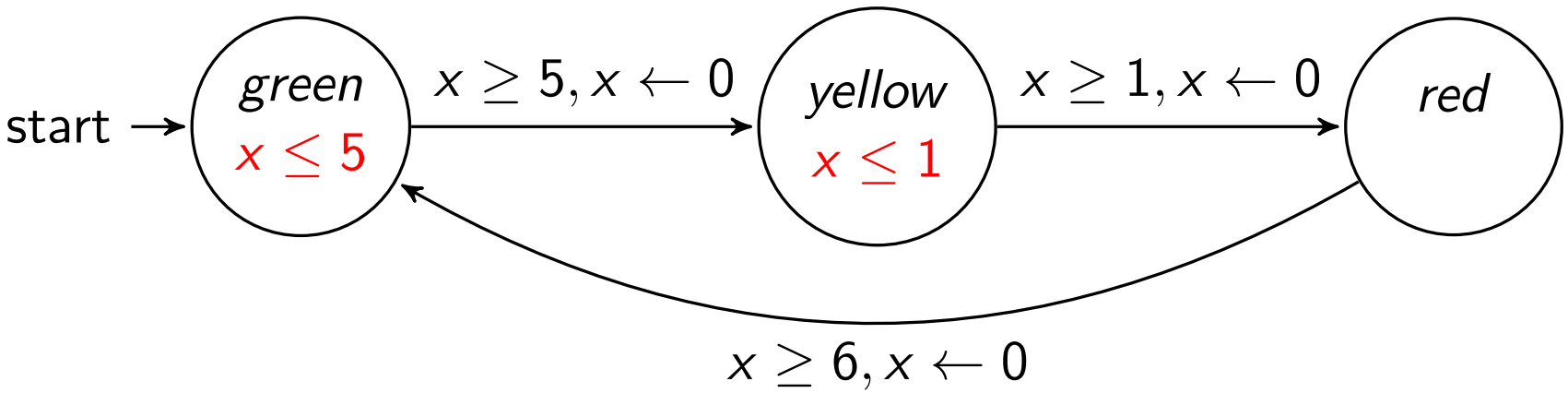
# Parametric Timed Automata



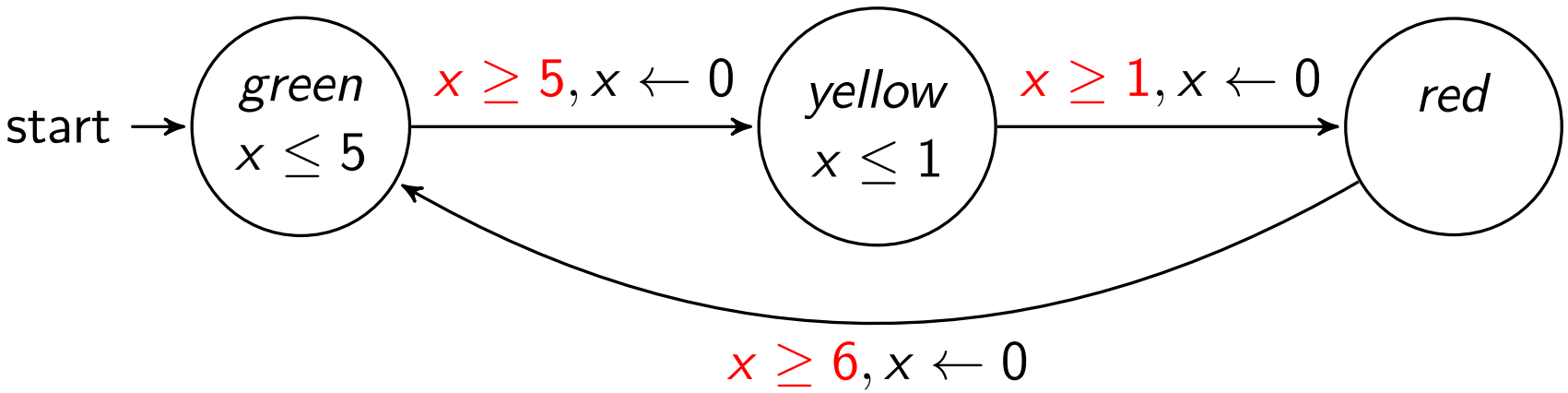
# Parametric Timed Automata



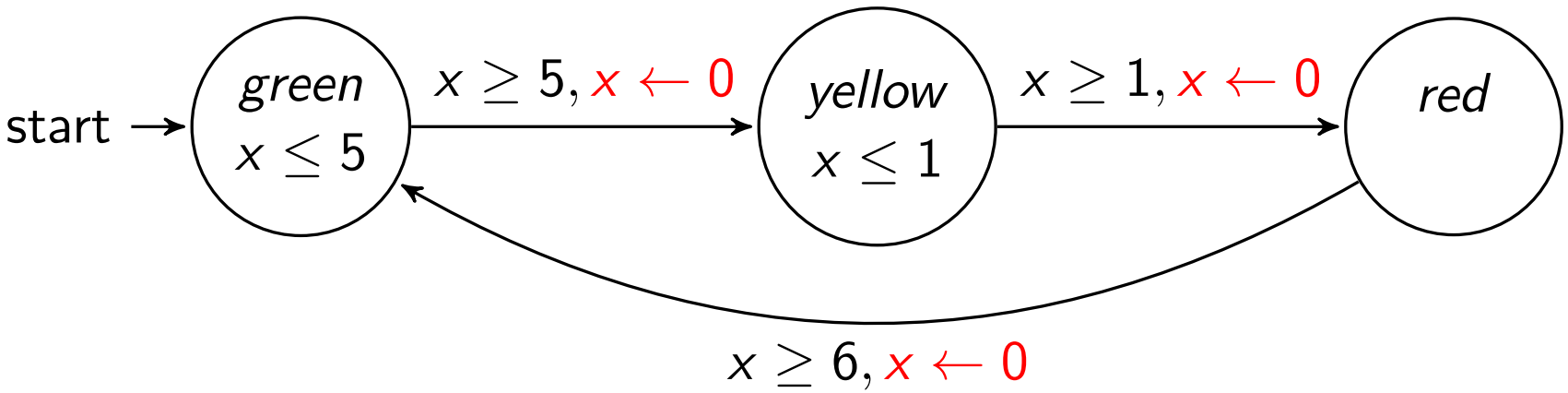
# Parametric Timed Automata



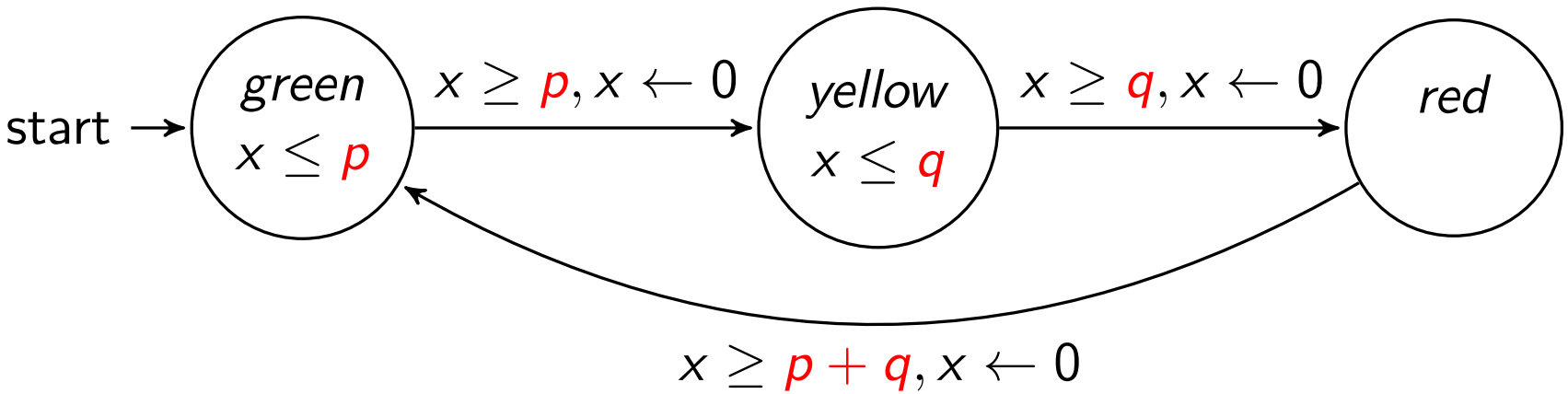
# Parametric Timed Automata



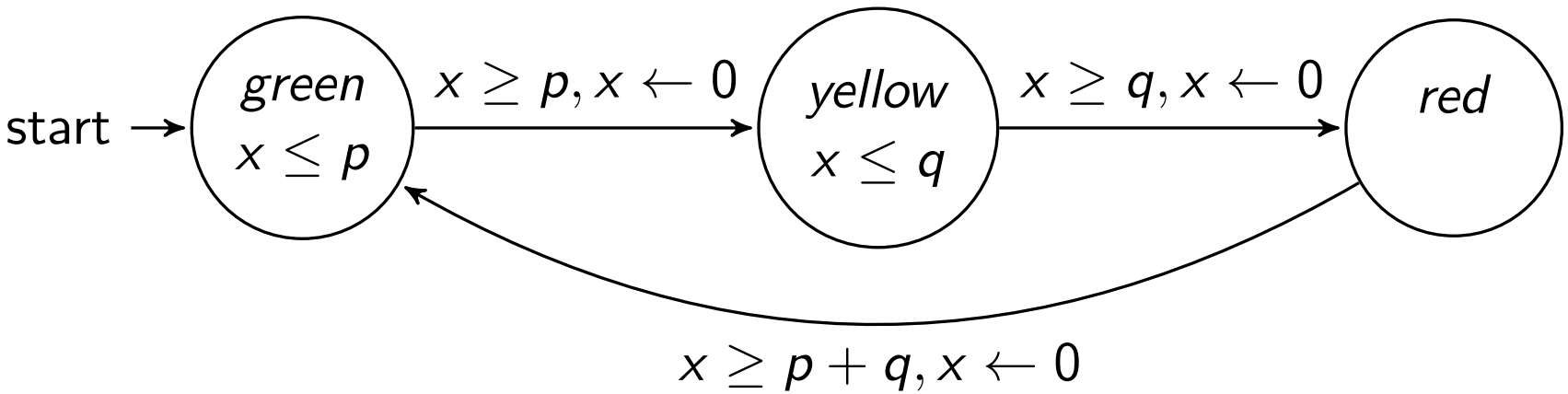
# Parametric Timed Automata



# Parametric Timed Automata



# Parametric Timed Automata




**Parameter Synthesis Problem**

- given a PTA $A$ and a specification $\varphi$
- compute the set of all parameter valuations $v$ such that $A_v$ satisfies $\varphi$

# Parametric Reachability Problem

| | discrete time<br>integer parameters | continuous time<br>integer parameters | continuous time<br>real parameters |
|---|---|---|---|
| L/U-automata | decidable | decidable | decidable |
| 1c-PTA | decidable | decidable | decidable |
| 1pc-PTA | decidable | **decidable** | undecidable |
| 2c-PTA | open | open | open |
| 1p-PTA (3c) | **undecidable** | **undecidable** | undecidable |
| PTA (3c) | undecidable | undecidable | undecidable |



N. Beneš, P. Bezděk, K. G. Larsen, and J. Srba. Language Emptiness of Continuous-Time Parametric Timed Automata. In *Automata, Languages, and Programming, ICALP 2015*, volume 9135 of *LNCS*, pages 69–81. Springer, 2015.

# Our Focus

**Bounded Integer Parameter Synthesis Problem**

- given a PTA $A$ and a specification $\varphi$
- given integer bounds for each parameter
- compute the set of all integer parameter valuations $v$ within the given bounds such that $A_v$ satisfies $\varphi$

# Our Focus

**Bounded Integer Parameter Synthesis Problem**

- given a PTA $A$ and a specification $\varphi$
- given integer bounds for each parameter
- compute the set of all integer parameter valuations $v$ within the given bounds such that $A_v$ satisfies $\varphi$

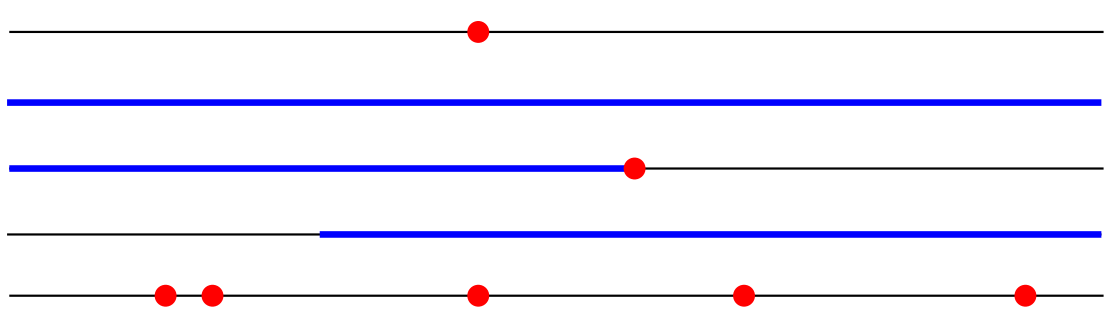
**Solutions**

- explicit (on parameters)
  - enumeration of all (finitely many) admissible parameter valuations
- symbolic (on parameters)

# Linear Temporal Logic

## LTL

- evaluated over runs
- atomic propositions (labels of locations)
- Boolean operators
- temporal operators
  - **F**uture
  - **G**lobally
  - **U**ntil
  - **F G**
  - **G F**



## Automata-Based Model Checking

- Büchi automaton for the (negation of) the formula
- combine with the model of a system (timed automaton)
- check emptiness of the product (timed Büchi automaton)

# Clock-Aware Linear Temporal Logic

**CA-LTL**

- evaluated over runs
- atomic propositions (labels of locations)
  + simple comparisons over clocks
- Boolean operators
- temporal operators

Examples:

- $\mathbf{FG}\, x < 10$
- $x < 5\ \mathbf{U}\ \mathit{ready}$

**Model Checking** – ???

# Clock-Aware Linear Temporal Logic

**CA-LTL**

- evaluated over runs
- atomic propositions (labels of locations)
  + simple comparisons over clocks
- Boolean operators
- temporal operators

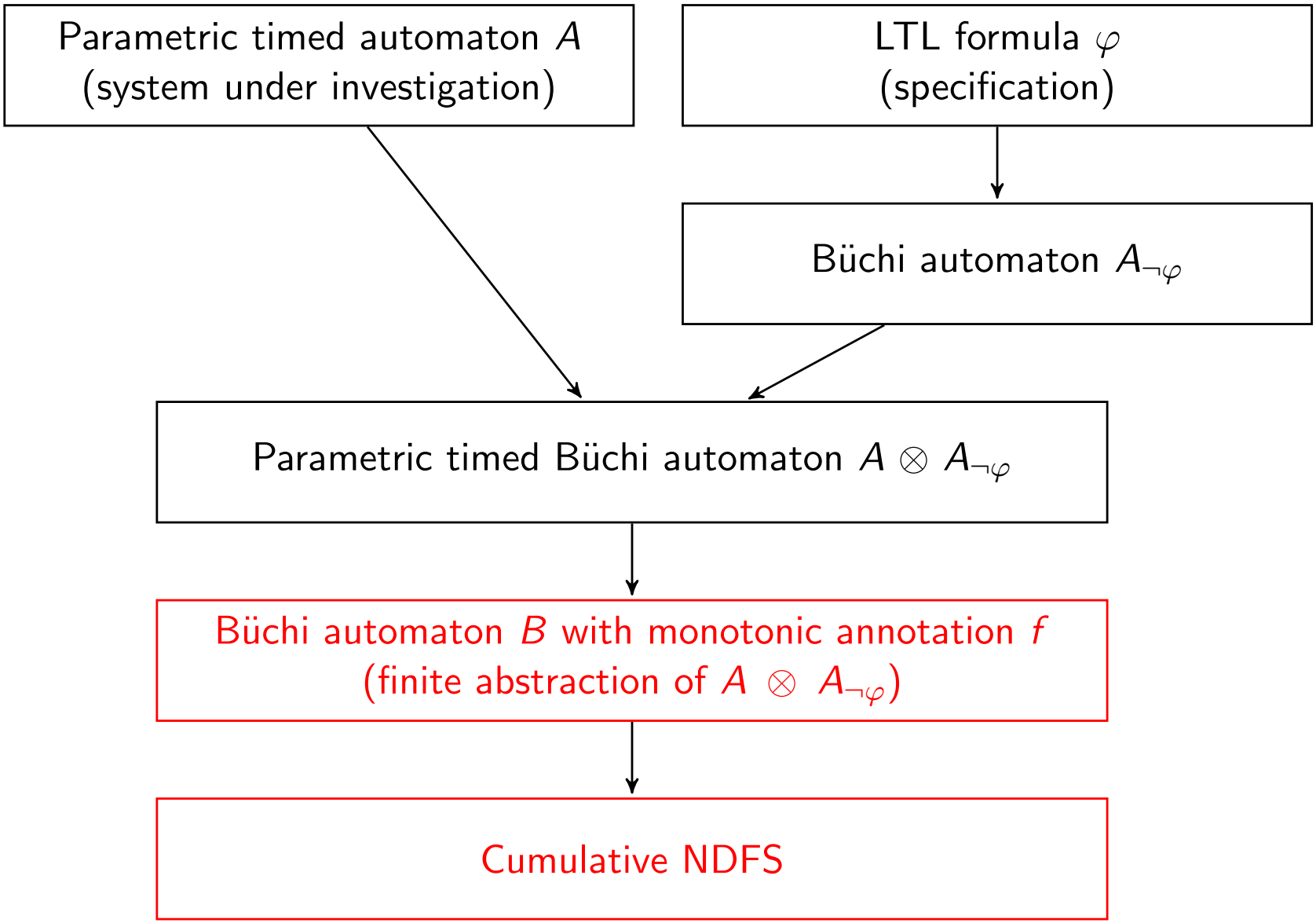
Examples:

- $\mathbf{FG}\, x < 10$
- $x < 5\ \mathbf{U}\ \mathit{ready}$

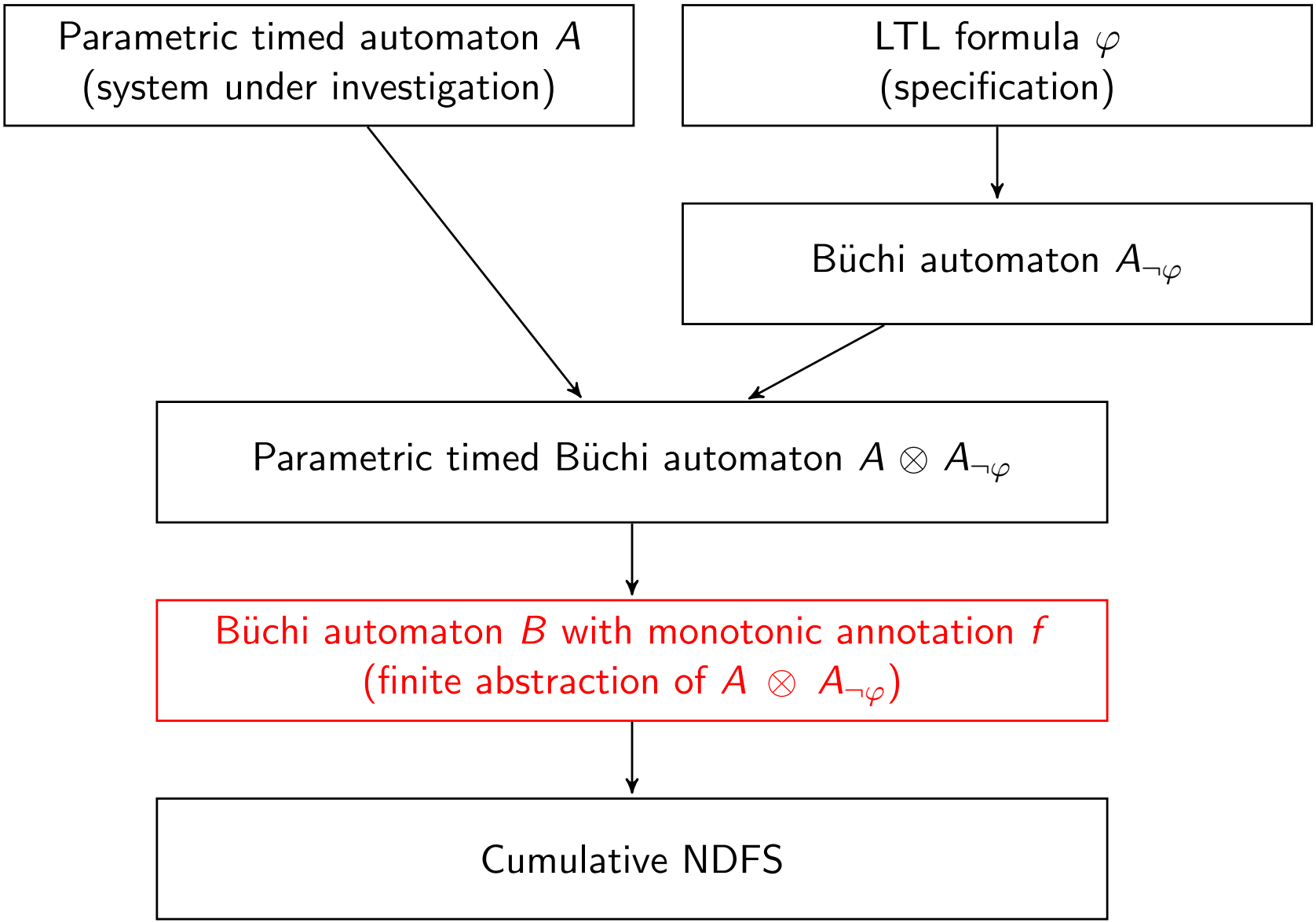
**Model Checking** – ???

- $x < 42\ \mathbf{U}\ x \geq 42$ cannot be represented as a TBA
  - TA transitions are *instantaneous*

# LTL Parameter Synthesis



# LTL Parameter Synthesis



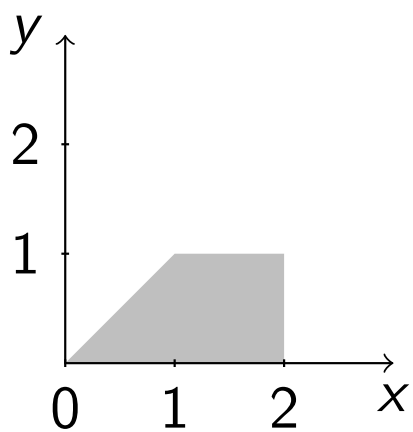
# Symbolic State Space of Timed Automata

**Zone**

- convex set of clock valuations given by conjunction of guards
- represents all possible clock valuations in one particular state

**Data structure**

- difference bound matrix (DBM)
- efficient operations, canonical form




| | $0$ | $x$ | $y$ |
|---|---|---|---|
| $0$ | $-$ | $(\leq, 0)$ | $(\leq, 0)$ |
| $x$ | $(\leq, 2)$ | $-$ | $(\leq, 2)$ |
| $y$ | $(\leq, 1)$ | $(\leq, 0)$ | $-$ |

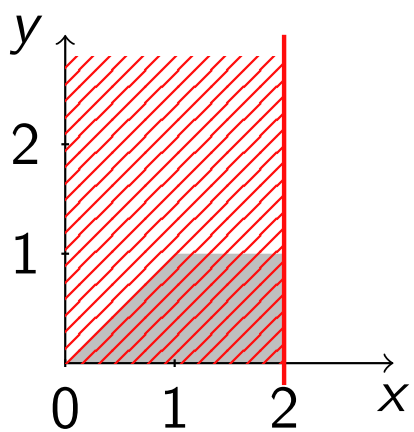
# Symbolic State Space of Timed Automata

**Zone**

- convex set of clock valuations given by conjunction of guards
- represents all possible clock valuations in one particular state

**Data structure**

- difference bound matrix (DBM)
- efficient operations, canonical form




| | 0 | $x$ | $y$ |
|---|---|---|---|
| 0 | – | $(\leq, 0)$ | $(\leq, 0)$ |
| $x$ | $(\leq, 2)$ | – | $(\leq, 2)$ |
| $y$ | $(\leq, 1)$ | $(\leq, 0)$ | – |

$x \leq 2$

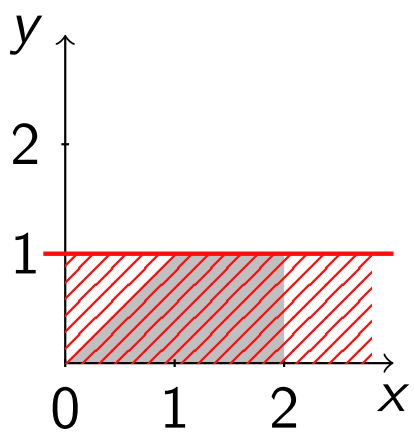
# Symbolic State Space of Timed Automata

**Zone**

- convex set of clock valuations given by conjunction of guards
- represents all possible clock valuations in one particular state

**Data structure**

- difference bound matrix (DBM)
- efficient operations, canonical form




| | 0 | $x$ | $y$ |
|---|---|---|---|
| 0 | – | $(\leq, 0)$ | $(\leq, 0)$ |
| $x$ | $(\leq, 2)$ | – | $(\leq, 2)$ |
| $y$ | $(\leq, 1)$ | $(\leq, 0)$ | – |

$y \leq 1$

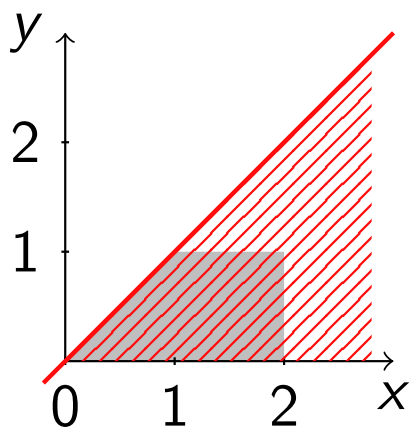
# Symbolic State Space of Timed Automata

**Zone**

- convex set of clock valuations given by conjunction of guards
- represents all possible clock valuations in one particular state

**Data structure**

- difference bound matrix (DBM)
- efficient operations, canonical form




| | 0 | $x$ | $y$ |
|---|---|---|---|
| 0 | – | $(\leq, 0)$ | $(\leq, 0)$ |
| $x$ | $(\leq, 2)$ | – | $(\leq, 2)$ |
| $y$ | $(\leq, 1)$ | $(\leq, 0)$ | – |

$y - x \leq 0$

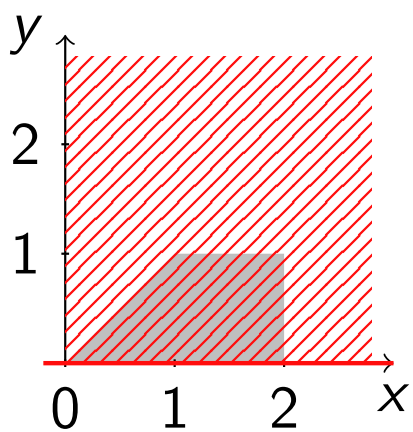
# Symbolic State Space of Timed Automata

**Zone**

- convex set of clock valuations given by conjunction of guards
- represents all possible clock valuations in one particular state

**Data structure**

- difference bound matrix (DBM)
- efficient operations, canonical form




| | 0 | $x$ | $y$ |
|---|---|---|---|
| 0 | – | $(\leq, 0)$ | $(\leq, 0)$ |
| $x$ | $(\leq, 2)$ | – | $(\leq, 2)$ |
| $y$ | $(\leq, 1)$ | $(\leq, 0)$ | – |

$y \geq 0$

# Symbolic State Space of PTA

**Parametric zone**

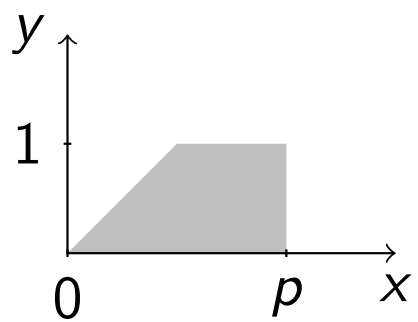
- given by conjunction of parametric guards, and
- constraints on parameter values (context)

**Data structure**

- Constrained parametric difference bound matrix (CPDBM) [1]

**CPDBM example**

- $Context = \{3 < p, p \leq 10\}$
-




| | 0 | $x$ | $y$ |
|---|---|---|---|
| 0 | - | $(\leq, 0)$ | $(\leq, 0)$ |
| $x$ | $(\leq, p)$ | - | $(\leq, p)$ |
| $y$ | $(\leq, 1)$ | $(\leq, 0)$ | - |

[1] Hune, T., Romijn, J., Stoelinga, M., Vaandrager, F.: Linear parametric model checking of timed automata. JLAP 52 (2002)

# Symbolic State Space of PTA

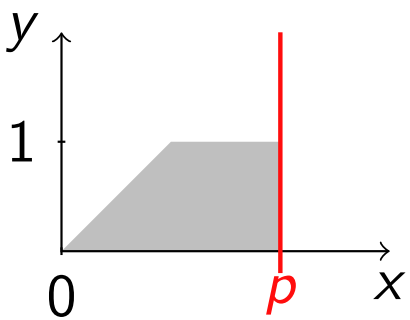
**Parametric zone**

- given by conjunction of parametric guards, and
- constraints on parameter values (context)

**Data structure**

- Constrained parametric difference bound matrix (CPDBM) [1]

**CPDBM example**

- $Context = \{3 < p, p \leq 10\}$
- 

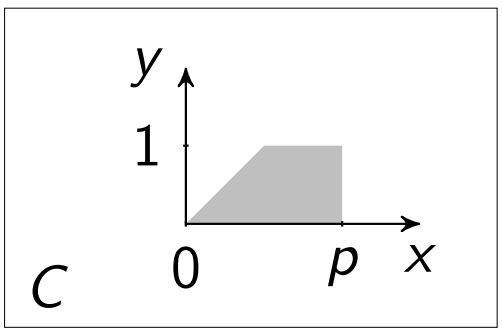

| | 0 | $x$ | $y$ |
|---|---|---|---|
| 0 | - | $(\leq, 0)$ | $(\leq, 0)$ |
| $x$ | $(\leq, p)$ | - | $(\leq, p)$ |
| $y$ | $(\leq, 1)$ | $(\leq, 0)$ | - |

[1] Hune, T., Romijn, J., Stoelinga, M., Vaandrager, F.: Linear parametric model checking of timed automata. JLAP 52 (2002)

# Parametric Zone Splitting

- result of CPDBM operations can be ambiguous
- the application of a guard leads to a split of the parametric context

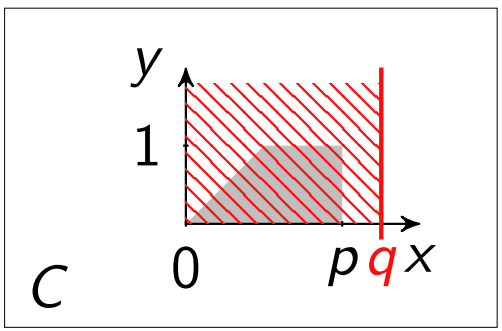
Example: $x \leq q$



# Parametric Zone Splitting

- result of CPDBM operations can be ambiguous
- the application of a guard leads to a split of the parametric context

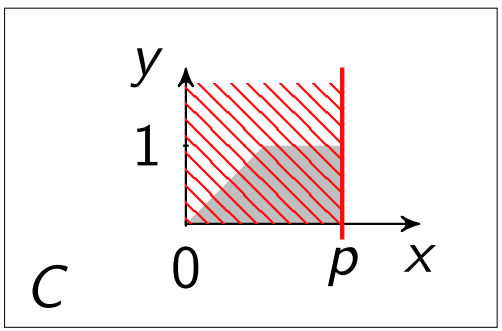
Example: $x \leq q$



# Parametric Zone Splitting

- result of CPDBM operations can be ambiguous
- the application of a guard leads to a split of the parametric context

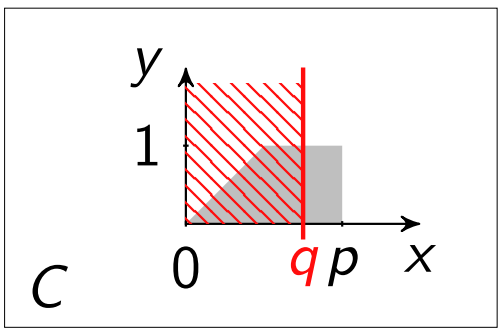
Example: $x \leq q$



# Parametric Zone Splitting

- result of CPDBM operations can be ambiguous
- the application of a guard leads to a split of the parametric context

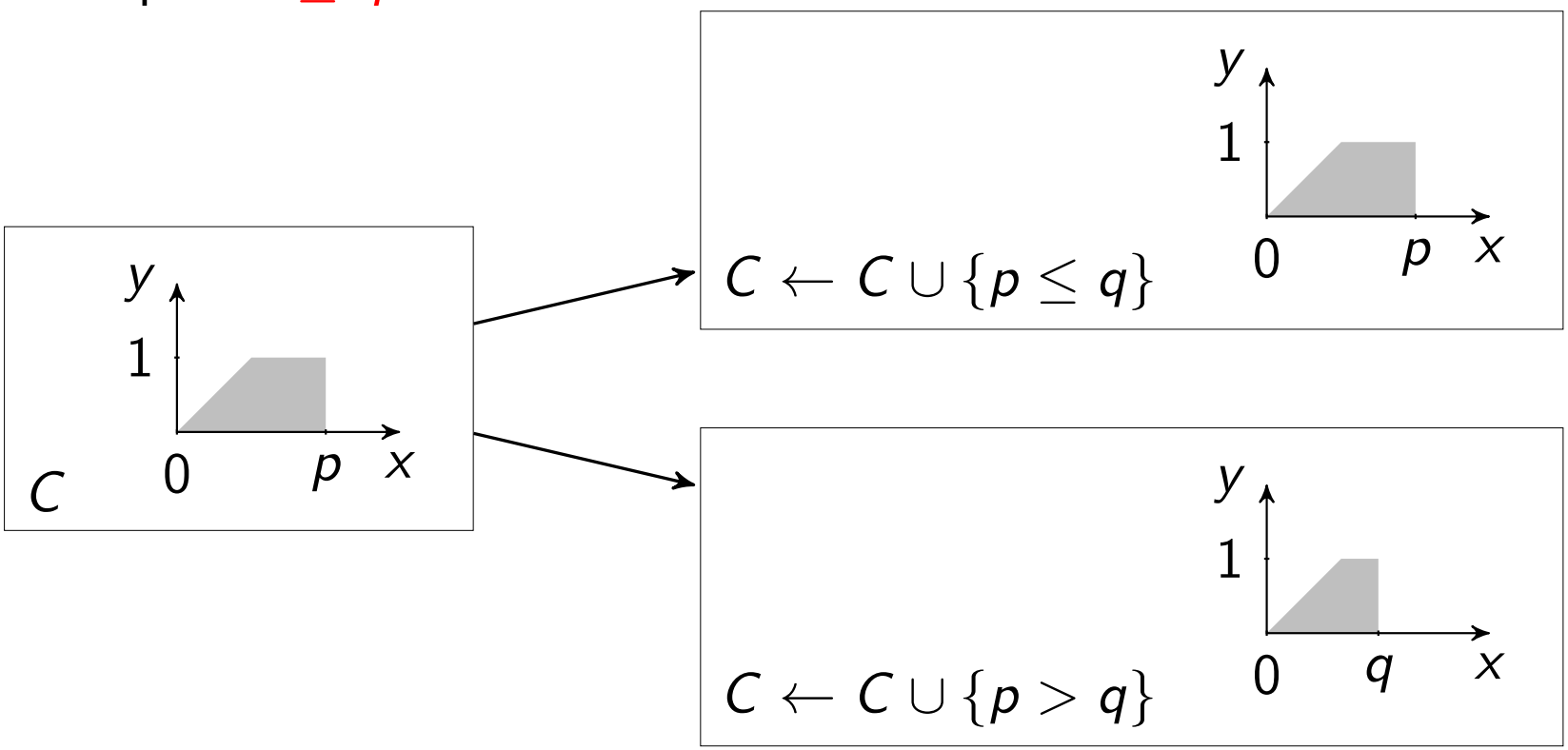
Example: $x \leq q$



# Parametric Zone Splitting

- result of CPDBM operations can be ambiguous
- the application of a guard leads to a split of the parametric context

Example: $x \leq q$

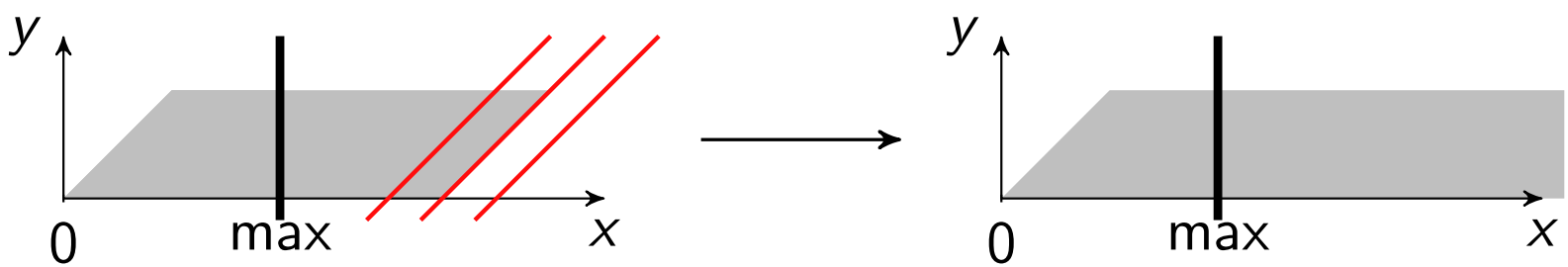


# Zone Extrapolation

- The number of (non-parametric) zones can be unbounded

**k-extrapolation**

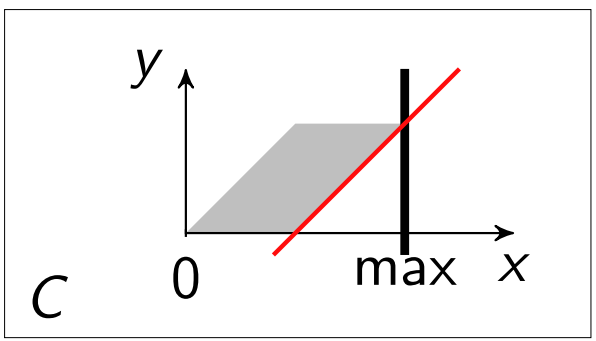
- zones that differ only in bounds exceeding the *maximal bound* on clock valuations cannot be distinguished
- replace the bounds with $\infty$



# Parametric Zone Extrapolation

**pk-extrapolation**

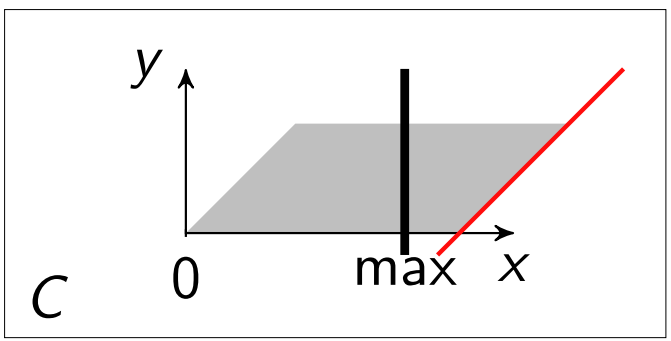
- based on k-extrapolation
- parametric zone bounds may exceed the maximal bound for only a subset of the allowed parameter valuations
- leads to a split of the parametric context



# Parametric Zone Extrapolation

**pk-extrapolation**

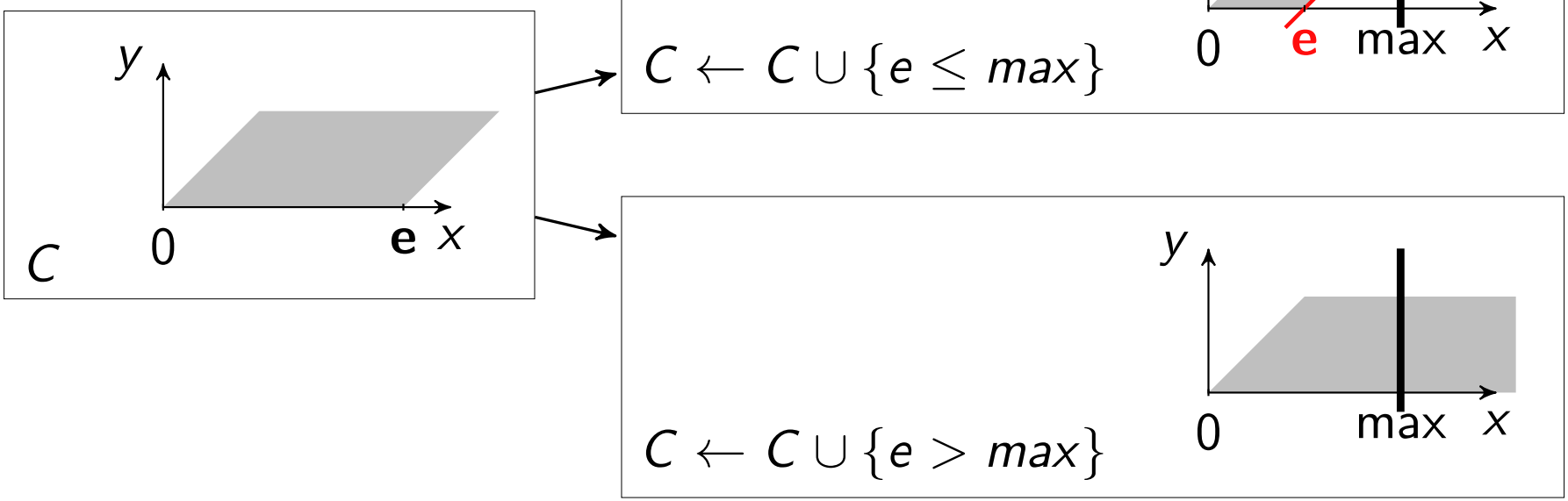
- based on k-extrapolation
- parametric zone bounds may exceed the maximal bound for only a subset of the allowed parameter valuations
- leads to a split of the parametric context



# Parametric Zone Extrapolation

**pk-extrapolation**

- based on k-extrapolation
- parametric zone bounds may exceed the maximal bound for only a subset of the allowed parameter valuations
- leads to a split of the parametric context



# State Space Exploration

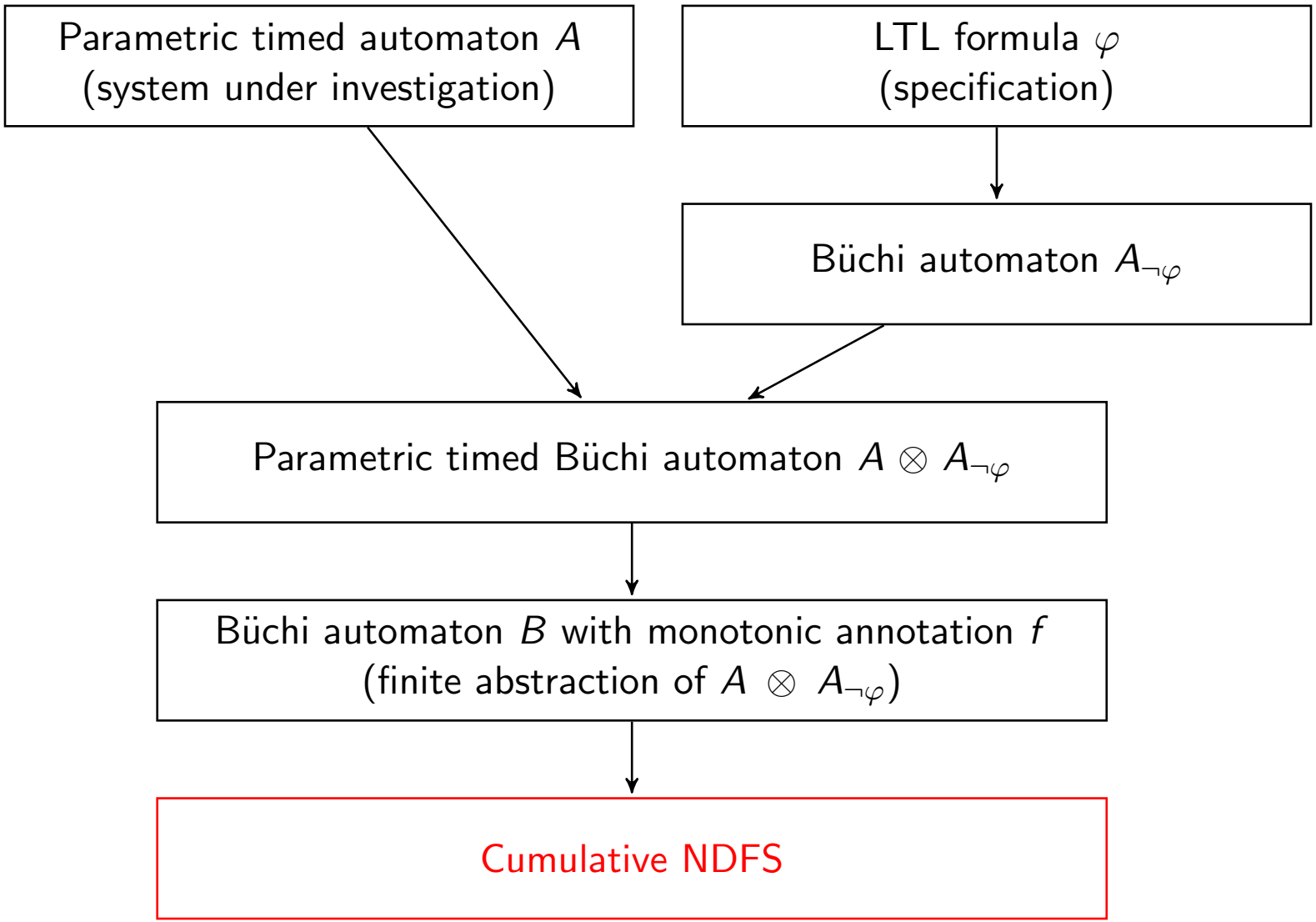
- state space storage needs unique representation of states
- one state represented with syntactically different CPDBMs
- $\Rightarrow$ semantic equivalence checks

**Heuristics**

- representative: CPDBM of the state's first occurence
- integer hull[2] of the state; hashtable
- caching

[2] A. Jovanović, D. Lime, and O. H. Roux. Integer Parameter Synthesis for Real-Time Systems. IEEE Trans. Software Eng., 41(5):445–461,2015.

# LTL Parameter Synthesis



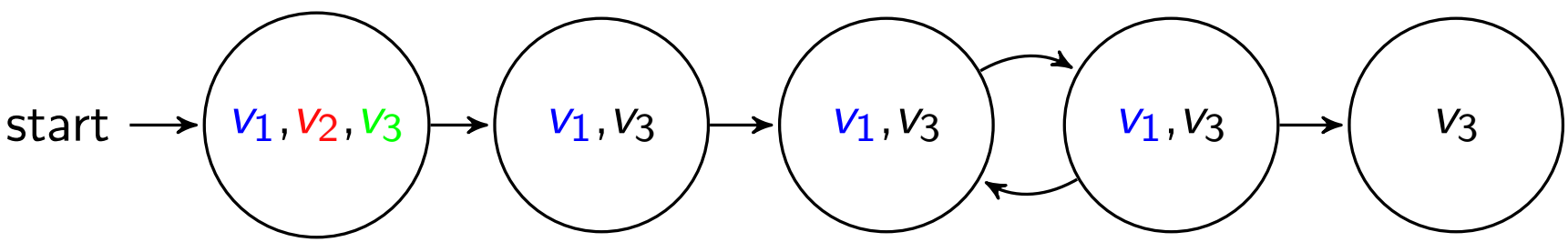
# Cumulative Nested Depth First Search

**Input**

- Büchi automaton
- each state is associated a set of parameter valuations

**Monotonicity property**

- the set of associated parameter valuations does not grow along a run
- $\Rightarrow$ does not change on a cycle



# Cumulative Nested Depth First Search

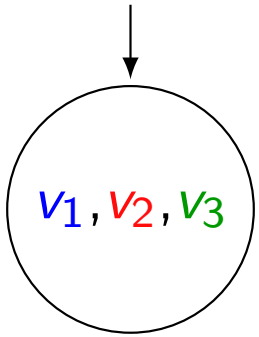
**Goal**

- find the set of all parameter valuations associated with an accepting cycle

**Algorithm**

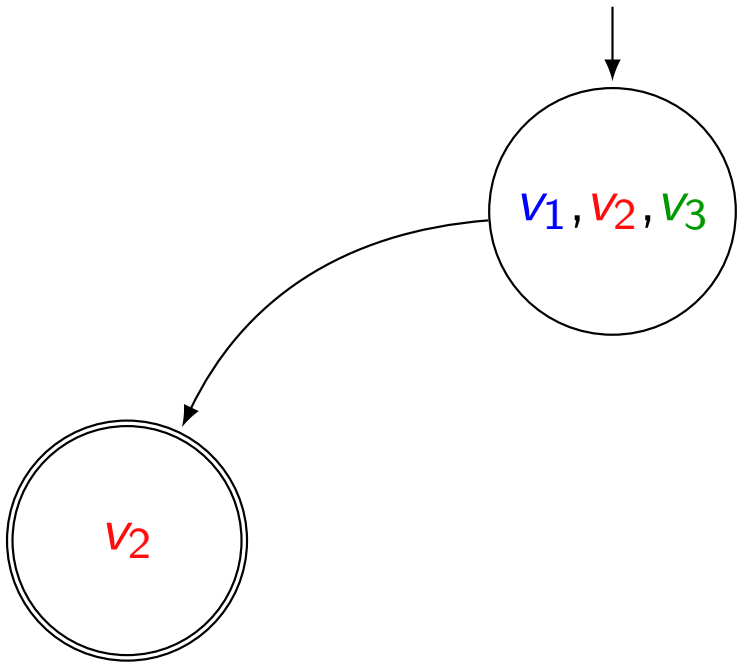
- based on Nested Depth First Search
- detects multiple accepting cycles on-the-fly
- the parameter valuations from the accepting cycles are accumulated during the computation
- backtracks when all parameter valuations associated with current state are already in the accumulated set

# Cumulative Nested Depth First Search



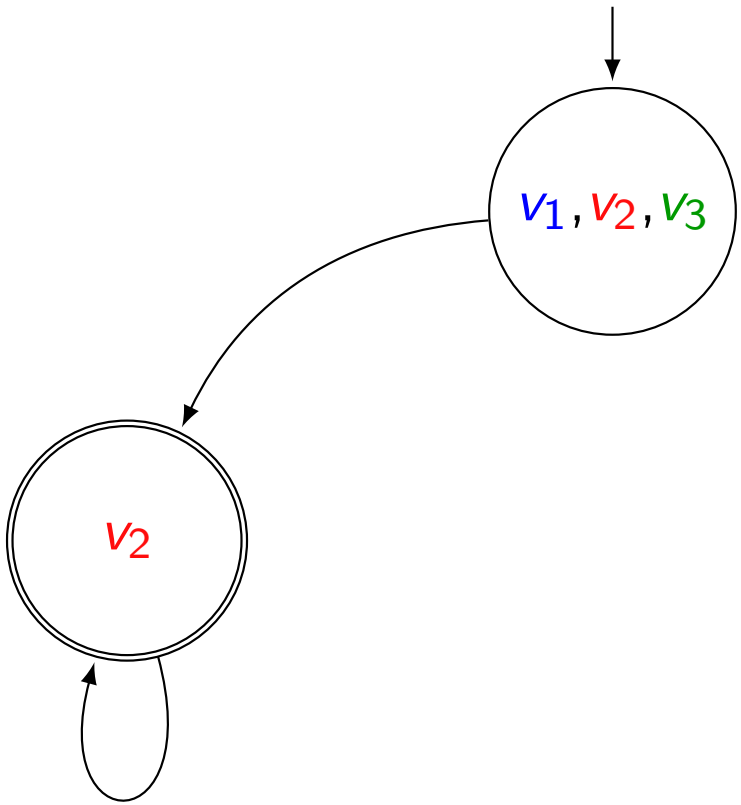

Found:

# Cumulative Nested Depth First Search



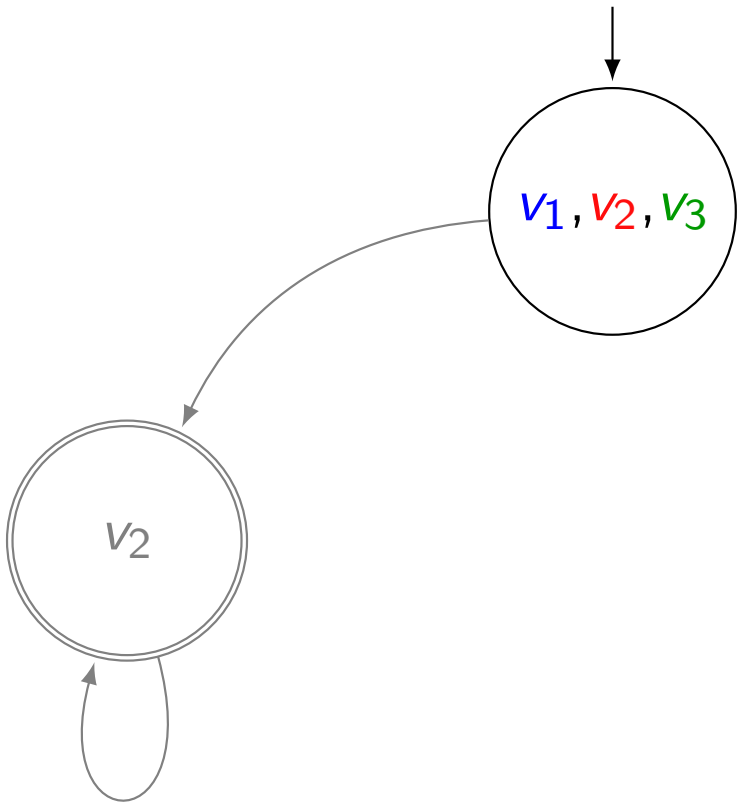

Found:

# Cumulative Nested Depth First Search



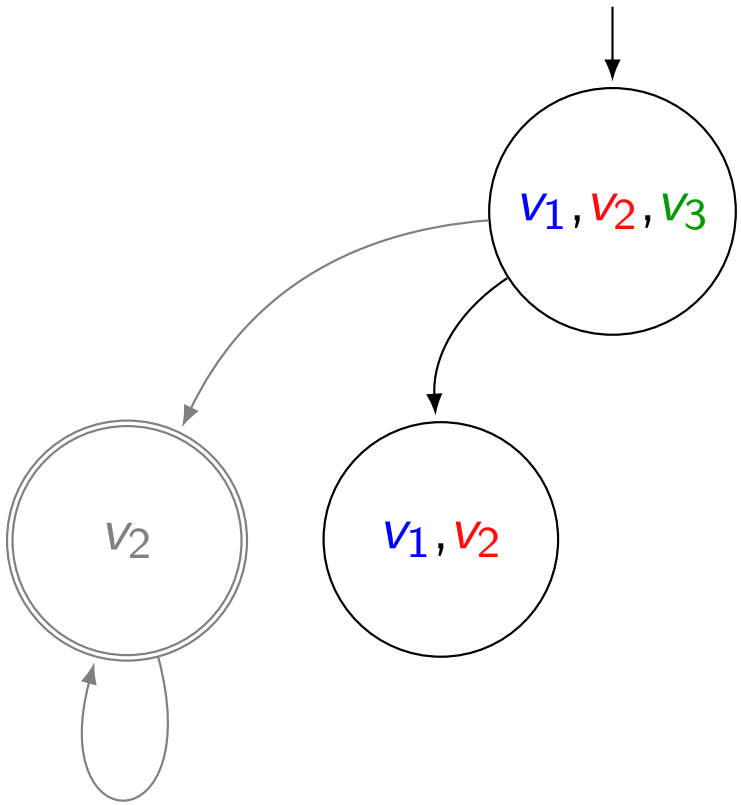

Found: $v_2$

# Cumulative Nested Depth First Search



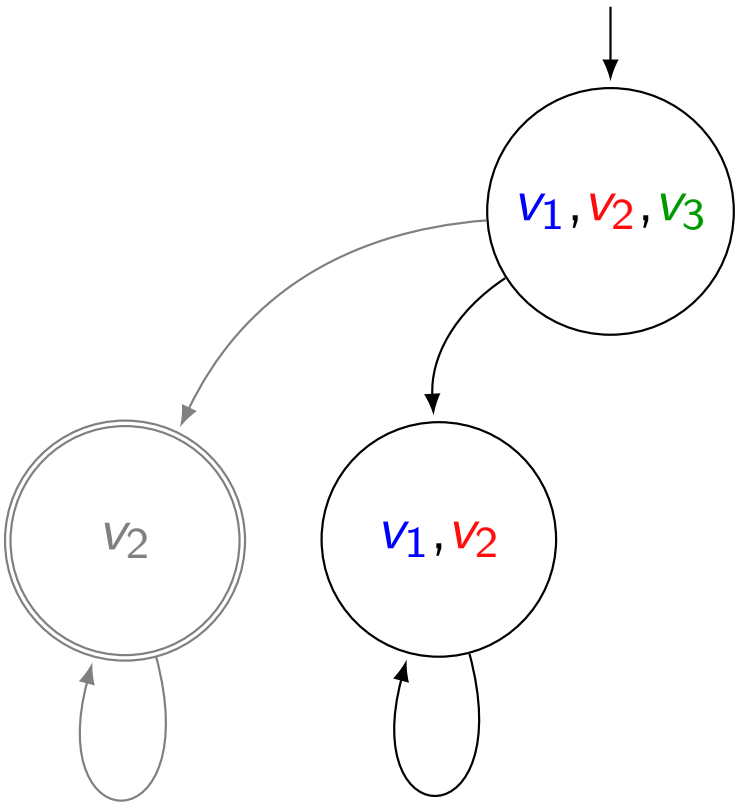

Found: $v_2$

# Cumulative Nested Depth First Search



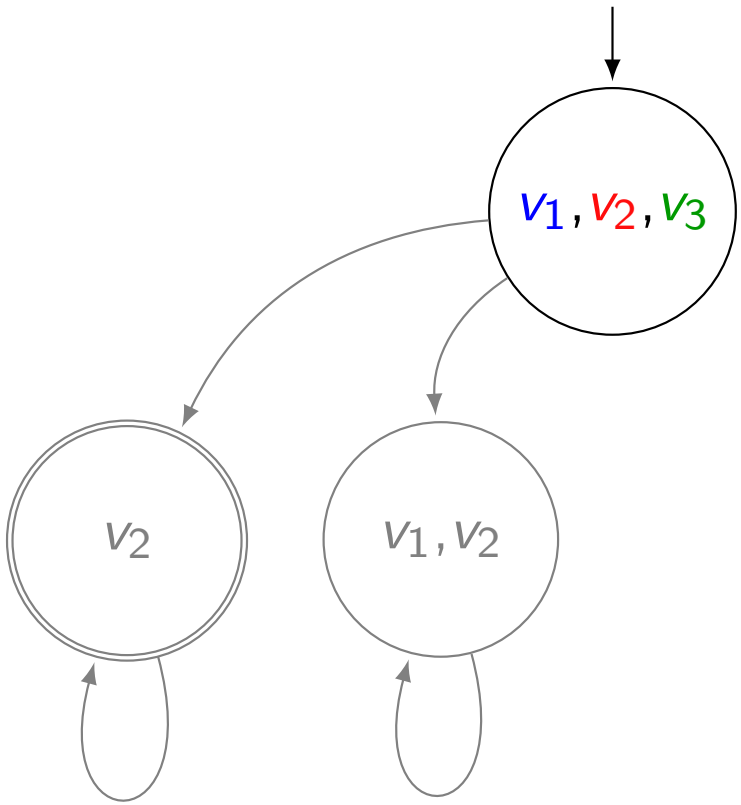

Found: $v_2$

# Cumulative Nested Depth First Search



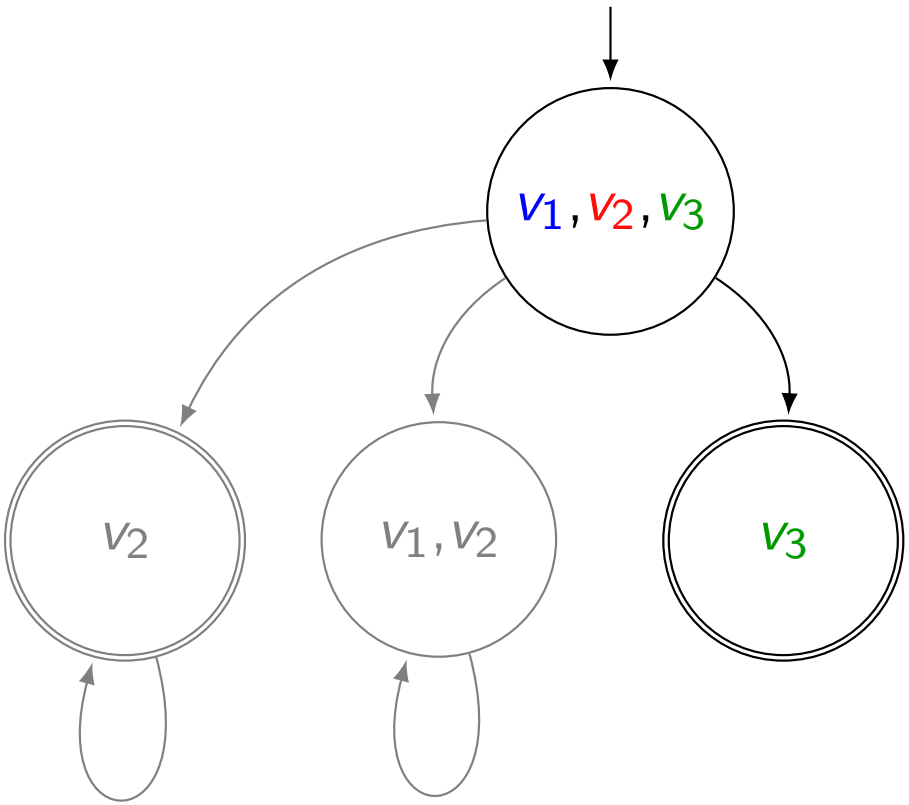

Found: $v_2$

# Cumulative Nested Depth First Search



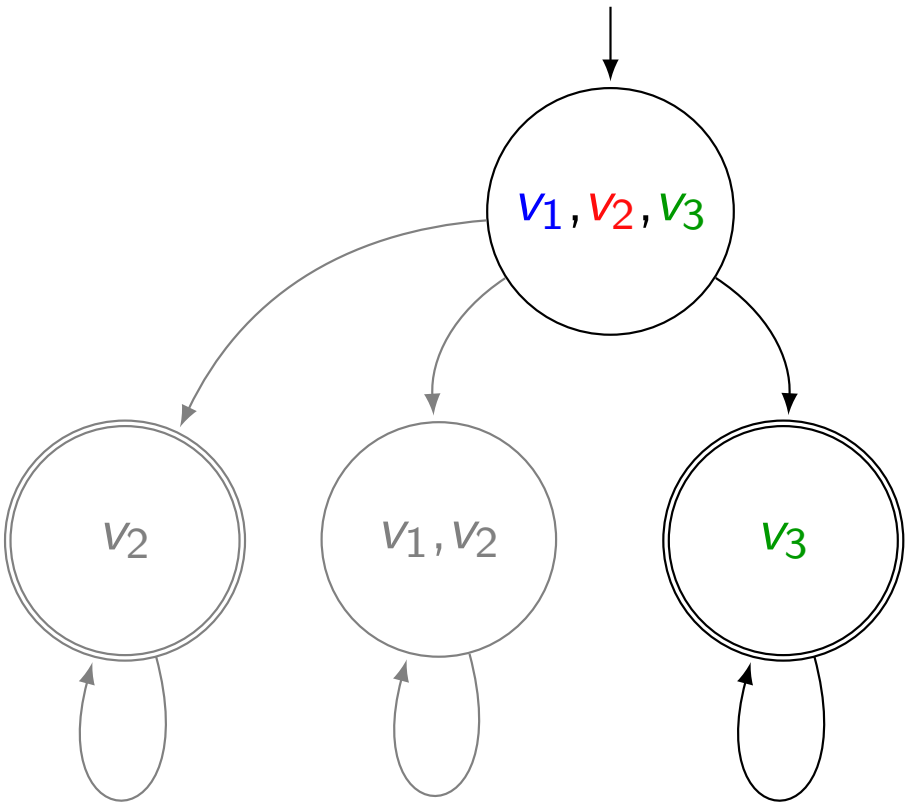

Found: $v_2$

# Cumulative Nested Depth First Search



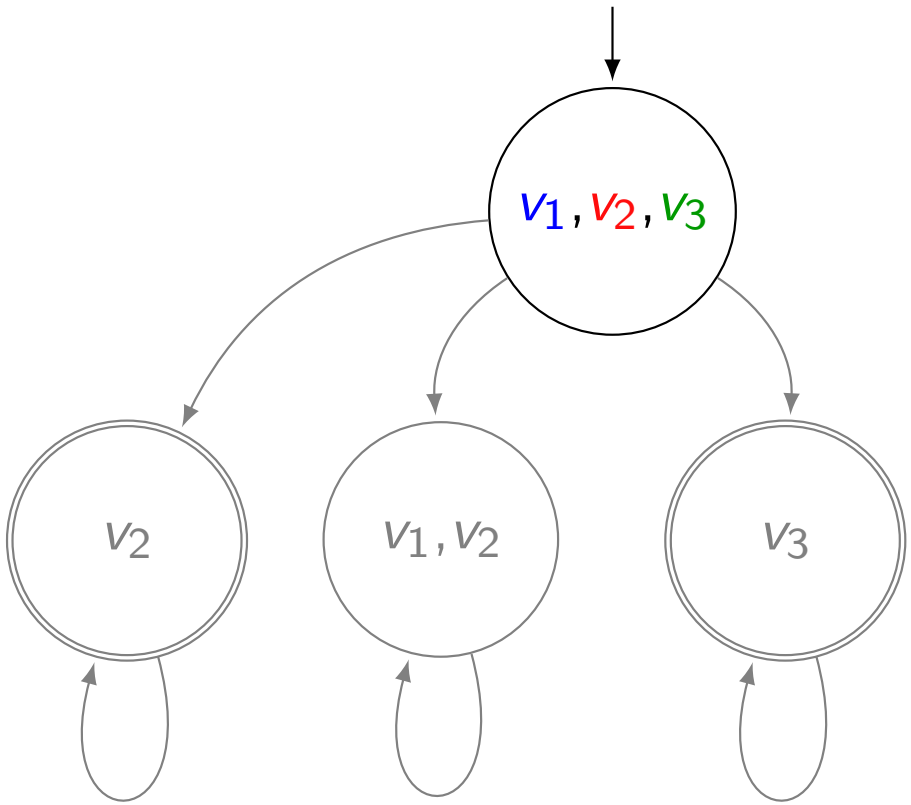

Found: $v_2$

# Cumulative Nested Depth First Search



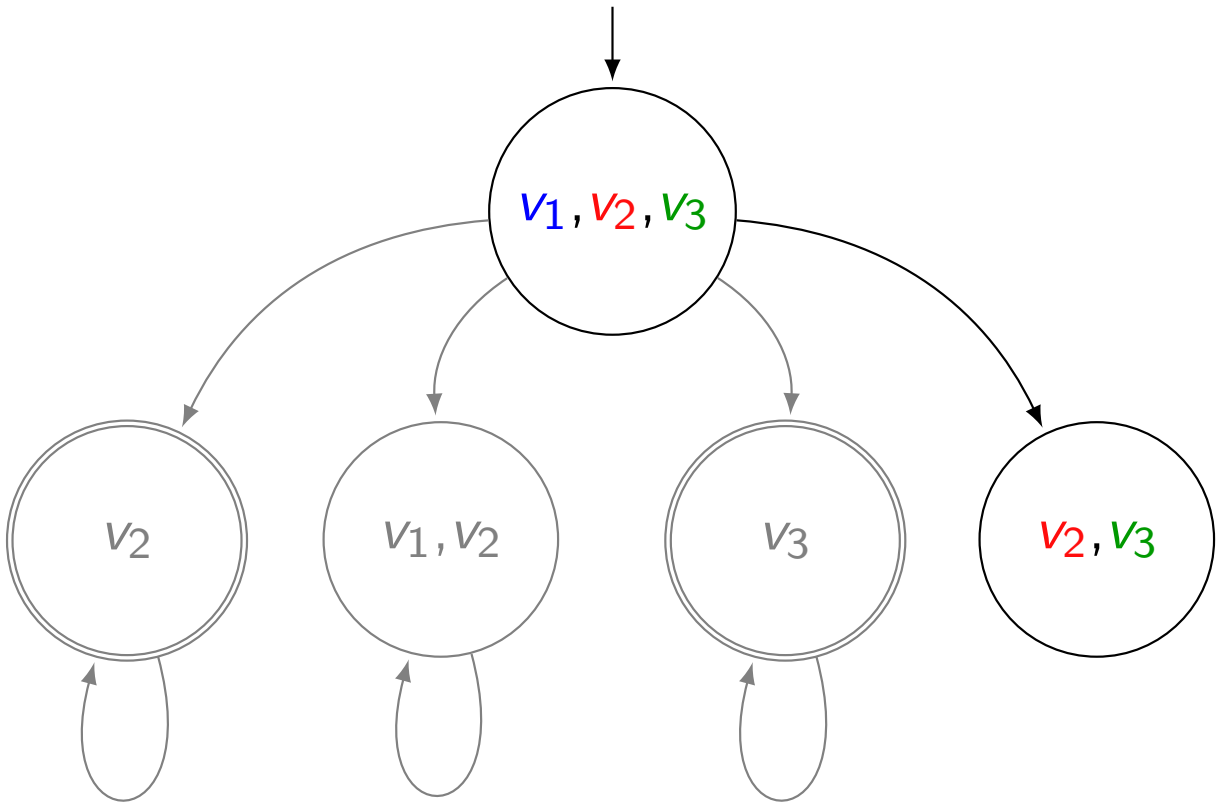

Found: $v_2$, $v_3$

# Cumulative Nested Depth First Search



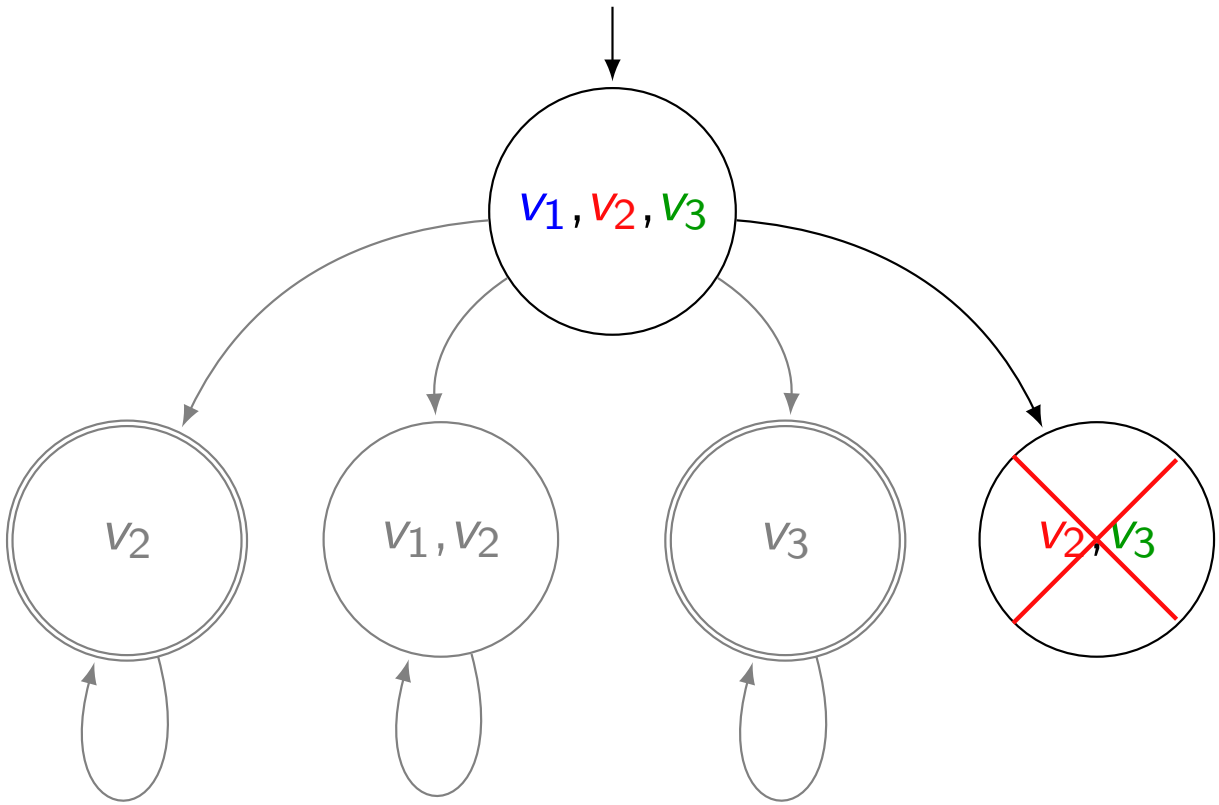

Found: $v_2$, $v_3$

# Cumulative Nested Depth First Search



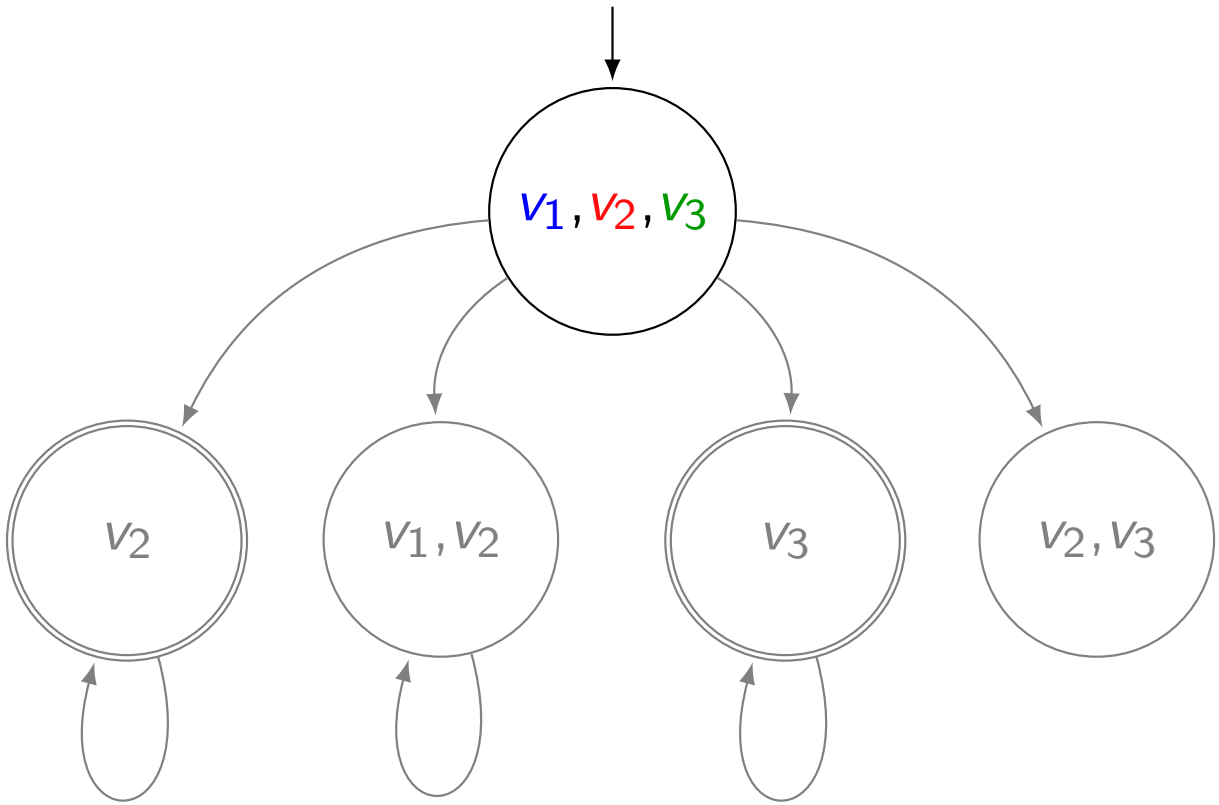

Found: $v_2$, $v_3$

# Cumulative Nested Depth First Search



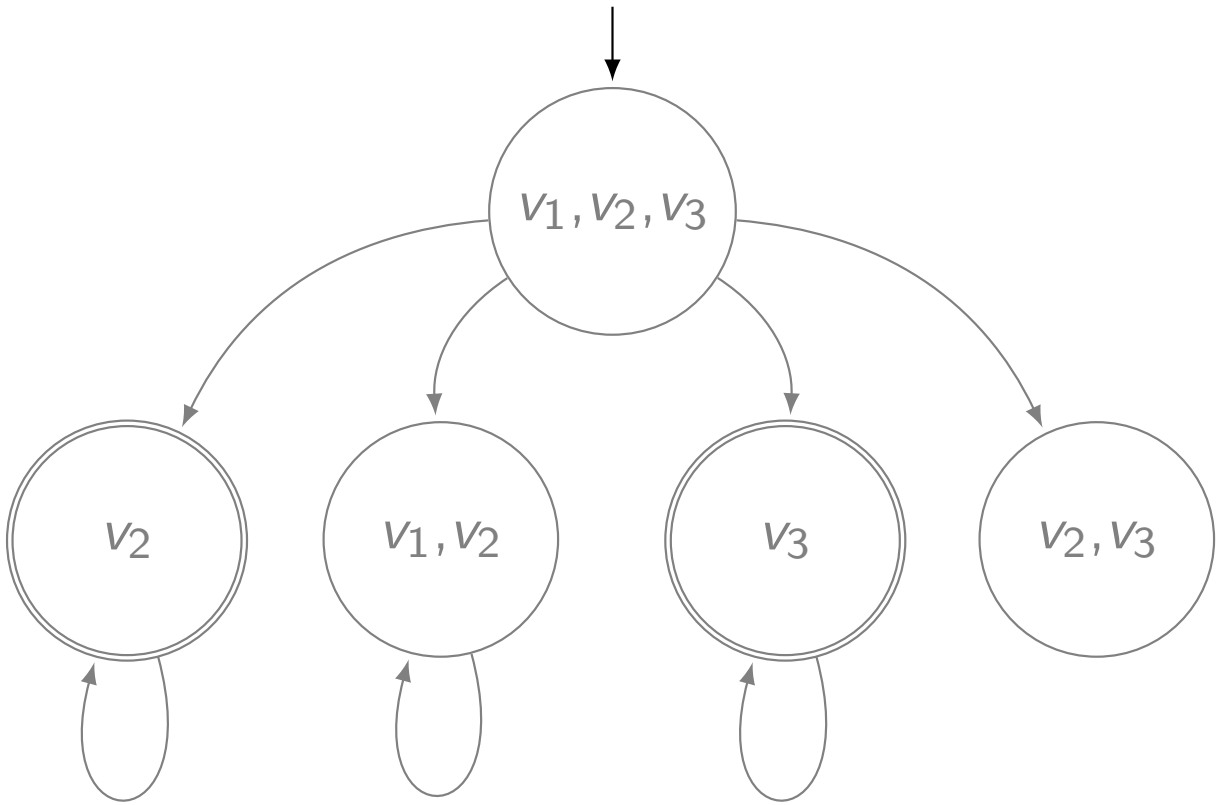

Found: $v_2$, $v_3$

# Cumulative Nested Depth First Search



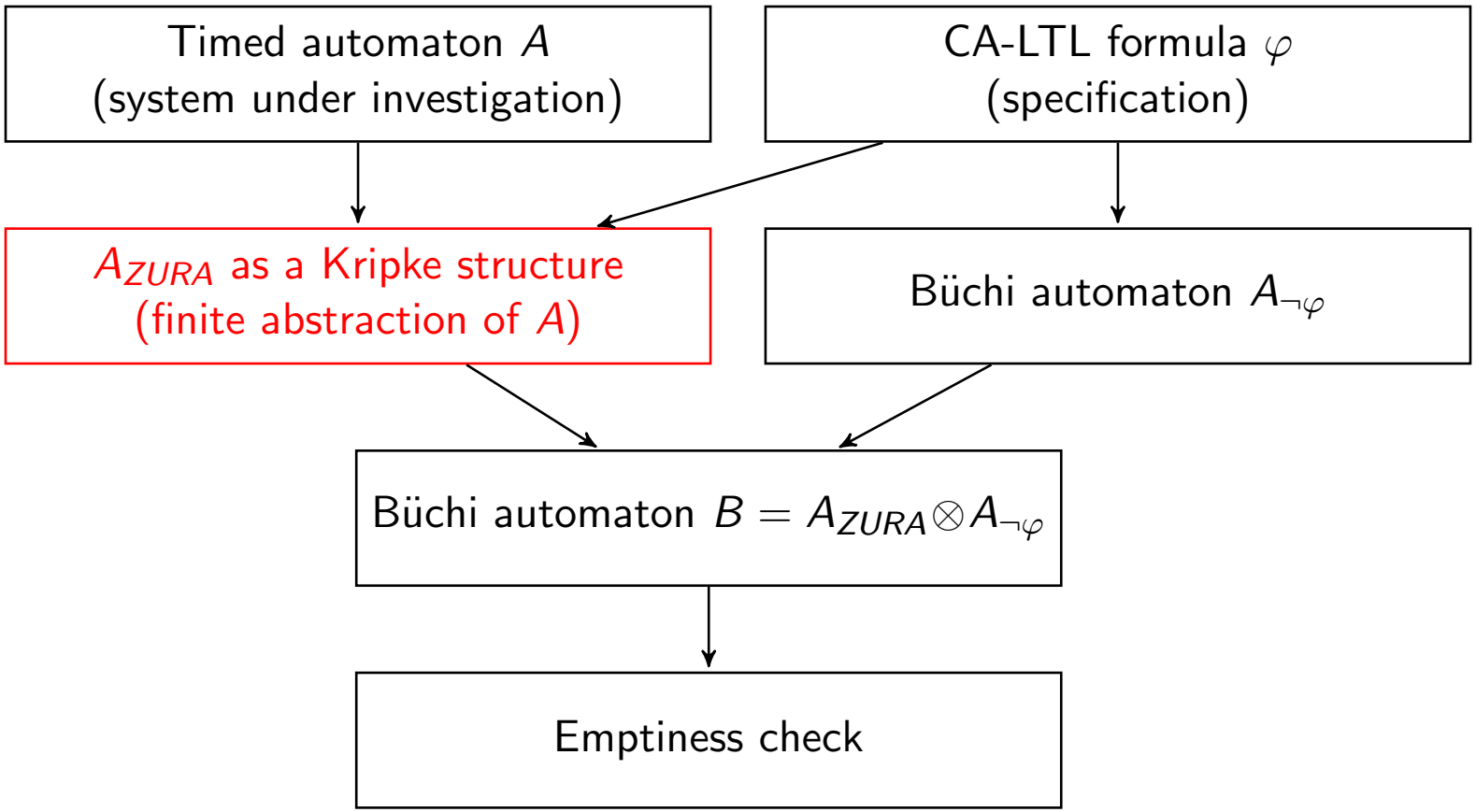

Found: $v_2$, $v_3$

# Cumulative Nested Depth First Search



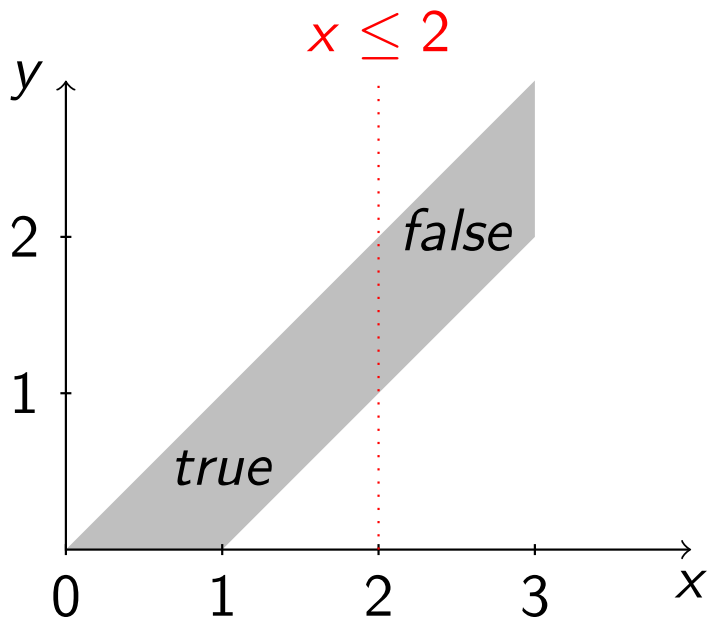

Found: $v_2$, $v_3$

# Clock-Aware LTL Model Checking



# Zone-Based Abstraction for CA-LTL

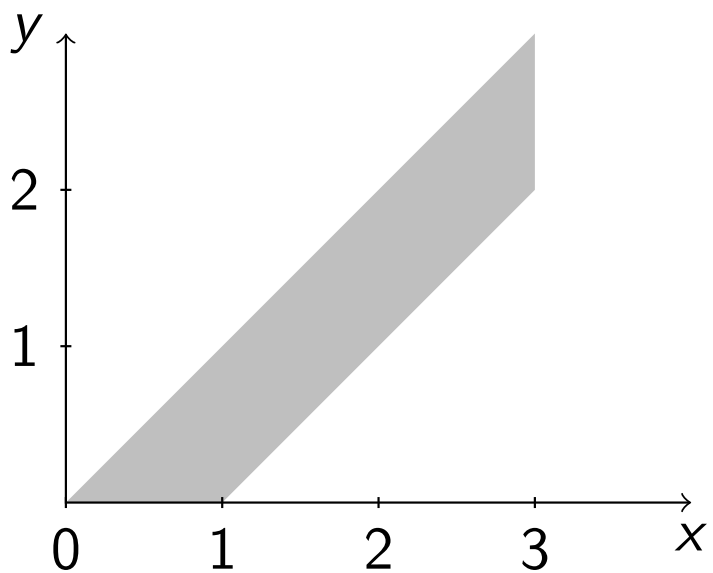
- Formula guard satisfaction may not be consistent



# Naive Partitionining

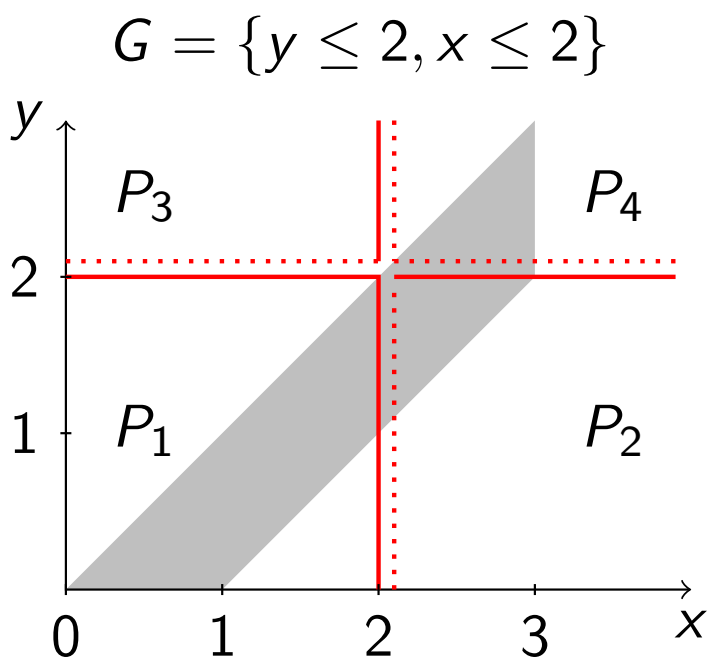
- consider the set of all constraints appearing in the formula

$$G = \{y \leq 2, x \leq 2\}$$



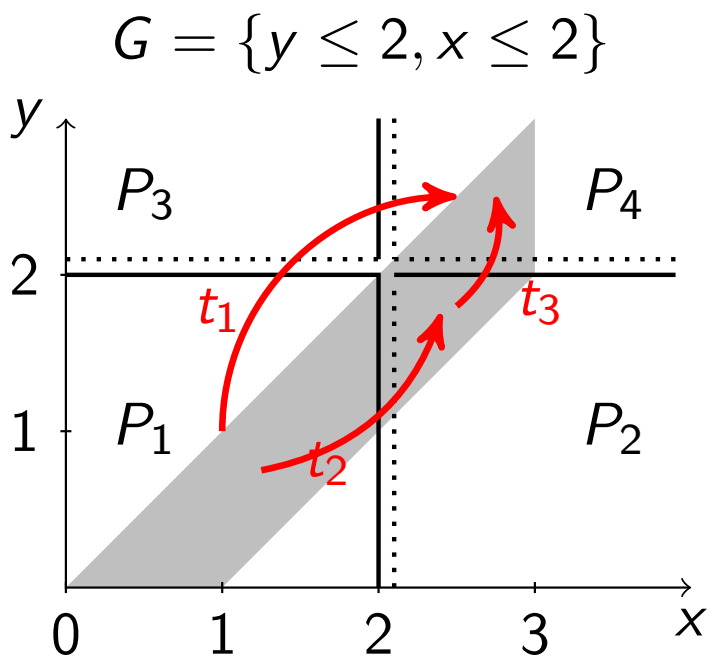
# Naive Partitionining

- consider the set of all constraints appearing in the formula
- partition the zones w.r.t. the set



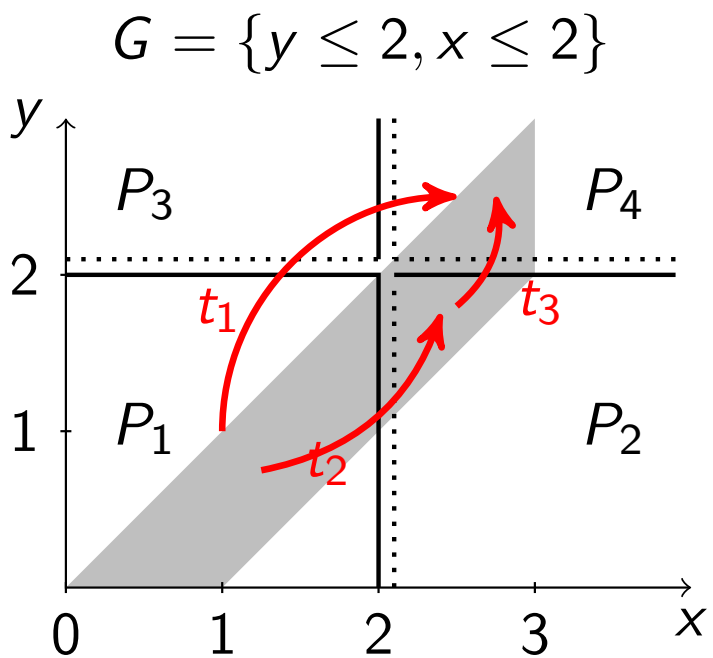
# Naive Partitionining

- consider the set of all constraints appearing in the formula
- partition the zones w.r.t. the set
- add delay transitions



# Naive Partitionining

- consider the set of all constraints appearing in the formula
- partition the zones w.r.t. the set
- add delay transitions



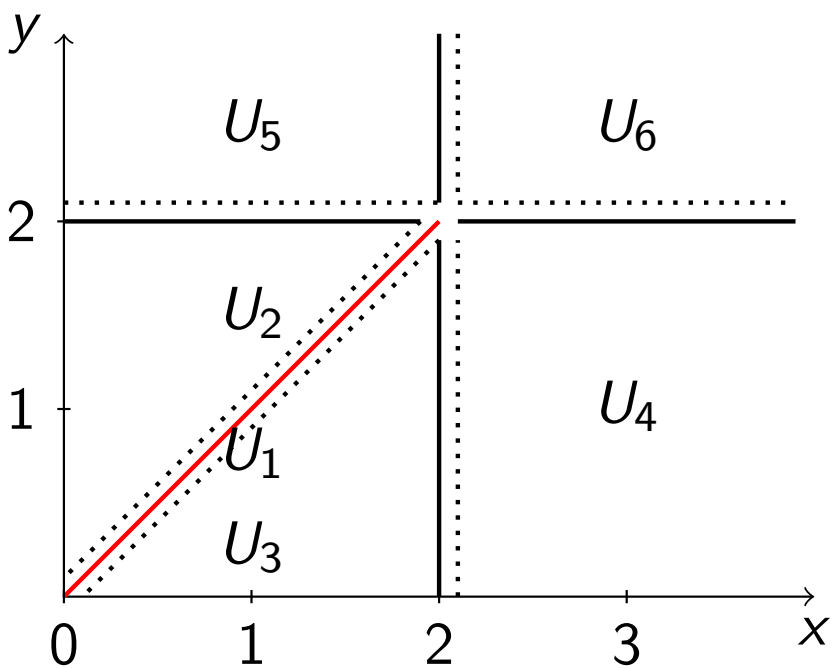

Incorrect: consider the formula: $(y \leq 2 \wedge x \leq 2)\,\mathbf{U}\,(y > 2 \wedge x > 2)$

# Correct Partitioning

**Ultraregions**

- partition with respect to G
- add diagonals to the partitioning

$$G = \{y \leq 2, x \leq 2\}$$

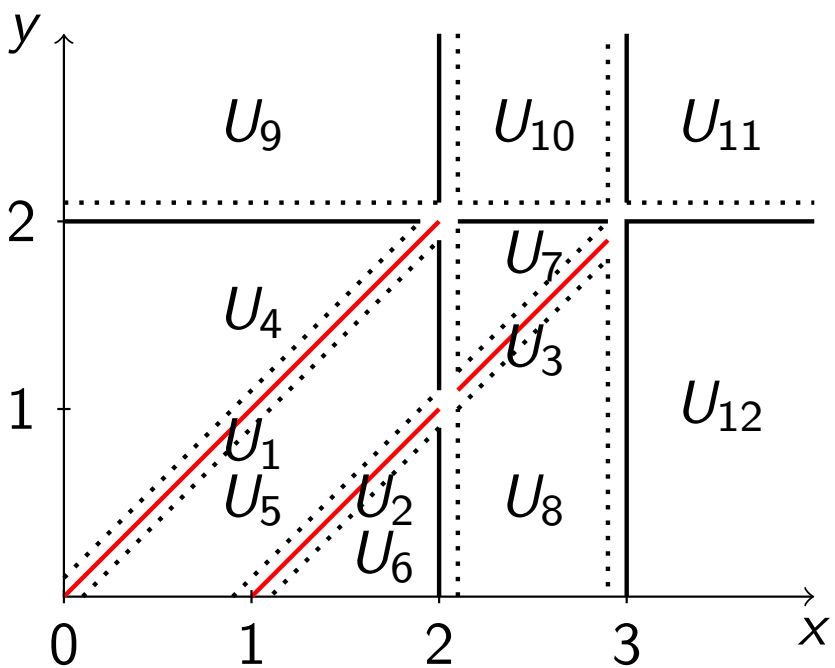


# Correct Partitioning

**Ultraregions**

- partition with respect to G
- add diagonals to the partitioning

$$G = \{y \leq 2, x \leq 2, x < 3\}$$



# Zone-Ultraregion Abstraction

$A_{ZURA}$

- combine zone-based abstraction with ultraregions
- symbolic states $(I, Z, U)$
- action + delay transitions
- branching reset operation
- preserves all runs (w.r.t. CA-LTL)
- atomic propositions + satisfaction of the formula clock guards

# Zone-Ultraregion Abstraction

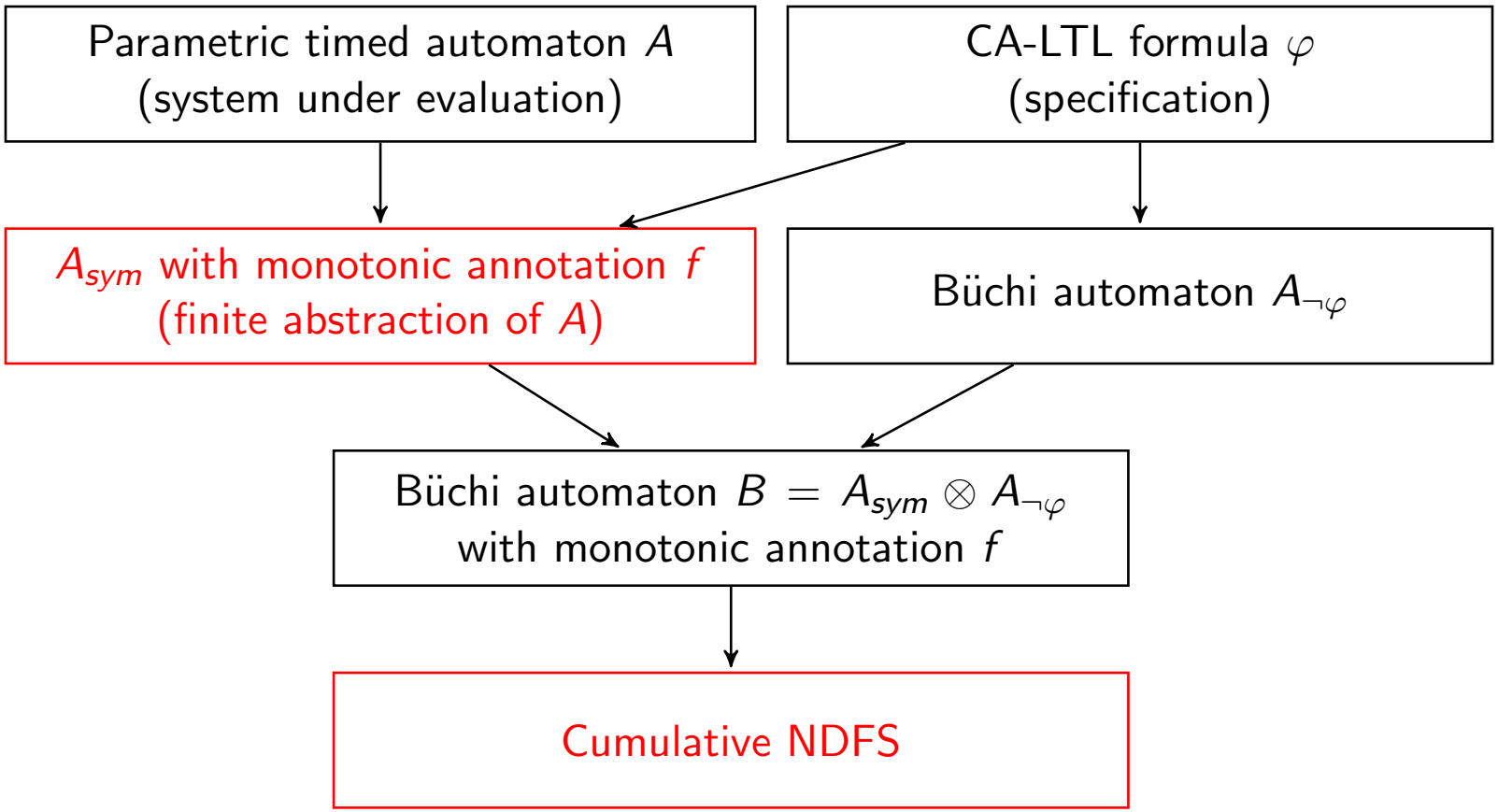
$A_{ZURA}$

- combine zone-based abstraction with ultraregions
- symbolic states $(I, Z, U)$
- action + delay transitions
- branching reset operation
- preserves all runs (w.r.t. CA-LTL)
- atomic propositions + satisfaction of the formula clock guards

**Parametric version**

- use parametric zones + pk-extrapolation
- finite-state symbolic Kripke structure $A_{sym}$ with annotations (sets of parameter valuations)
- every $v$-run of $A_{sym}$ is equivalent to a run in $A_v$ and vice versa

# CA-LTL Parameter Synthesis



# Experimental Evaluation

## Implementation

- prototype tool:
  `https://paradise.fi.muni.cz/parameterSynthesis/`
- symbolic manipulation using Parma Polyhedra Library

## Experimental Model

- parametric timed network of three sensors + a controller
- controller gathers data from sensors and provides a final value
- seven parameters
- properties to check:
  - $\varphi_1 = \mathbf{G}((l_1 \vee l_6) \Rightarrow (y \leq 500\ \mathbf{U}\ l_8))$
  - $\varphi_2 = \mathbf{G}((l_1 \vee l_6) \Rightarrow (y \leq 150\ \mathbf{U}\ l_8))$
  - $\varphi_3 = \mathbf{G}((l_1 \vee l_6) \Rightarrow \mathbf{F}\ l_8)$

# Experimental Evaluation

Table: Impact of model parameter count

| | 2 params | 3 params | 4 params | 5 params | 6 params | 7 params |
|---|---|---|---|---|---|---|
| $\varphi_1$ explicit | 3.5 s | 351 s | TO (17%) | TO (0%) | TO (0%) | TO (0%) |
| $\varphi_1$ CNDFS | 0.4 s | 2.2 s | 3.3 s | 5.7 s | 8.6 s | 36 s |
| $\varphi_2$ explicit | 2.5 s | 302 s | TO (20%) | TO (0%) | TO (0%) | TO (0%) |
| $\varphi_2$ CNDFS | 2 s | 25 s | 151 s | 1188 s | 4924 s | TO |
| $\varphi_2^*$ CNDFS | 2.5 s | 29 s | 193 s | 866 s | 3120 s | TO |
| $\varphi_3$ explicit | 1.7 s | 213 s | TO (22%) | TO (0%) | TO (0%) | TO (0%) |
| $\varphi_3$ CNDFS | 0.5 s | 3.9 s | 52 s | 124 s | 189 s | 1383 s |
| $\varphi_3^*$ CNDFS | 0.3 s | 1.5 s | 2 s | 3.8 s | 5.6 s | 24 s |

$^*$ run with larger maximum constant (500) for pk-extrapolation

timeout 2 hours

# Experimental Evaluation

Table: Impact of parameter range size

| | $[1, 10]$ | $[1, 50]$ | $[51, 100]$ | $[1, 100]$ |
|---|---|---|---|---|
| $\varphi_1$ explicit | 427 s | TO (0%) | TO (0%) | TO (0%) |
| $\varphi_1$ CNDFS | 8.4 s | 8.4 s | 8.5 s | 8.6 s |
| $\varphi_2$ explicit | 426 s | TO (0%) | TO (0%) | TO (0%) |
| $\varphi_2$ CNDFS | 8.4 s | 33 s | 1231 s | 4924 s |
| $\varphi_2^*$ CNDFS | 8.4 s | 35 s | 864 s | 3120 s |
| $\varphi_3$ explicit | 357 s | TO (0%) | TO (0%) | TO (0%) |
| $\varphi_3$ CNDFS | 189 s | 190 s | 6.6 s | 189 s |
| $\varphi_3^*$ CNDFS | 6.2 s | 6.2 s | 6.2 s | 6.3 s |

$^*$ run with larger maximum constant (500) for pk-extrapolation

timeout 2 hours

# Conclusion

## Summary

- CA-LTL extends LTL with simple clock constraints
- symbolic method for CA-LTL parameter synthesis
  - new parametric abstraction (pk-extrapolation)
  - Cumulative NDFS
  - ultraregion technique for CA-LTL properties
- experimental evaluation
  - better than parameter scan
  - performance hard to predict
  - larger max for pk-extrapolation may help

## Future Work

- try different abstractions
- parallel version of CNDFS
- extension of CA-LTL (action-based, difference constraints)
- parameters in CA-LTL properties