Symbolic Unfoldings for Networks of Timed Automata

Franck Cassez[1] Thomas Chatain[2] Claude Jard[2]

[1]CNRS/IRCCyN
Nantes, France

[2]IRISA
Rennes, France

Automated Technology for Verification and Analysis (ATVA'06)
October 23–26th, 2006

Beijing, China

Outline of the talk

- Unfoldings for Network of Automata
- Symbolic Unfoldings for Network of Timed Automata
- Conclusion

Outline of the talk

- Unfoldings for Network of Automata
- Symbolic Unfoldings for Network of Timed Automata
- Conclusion

Outline of the talk

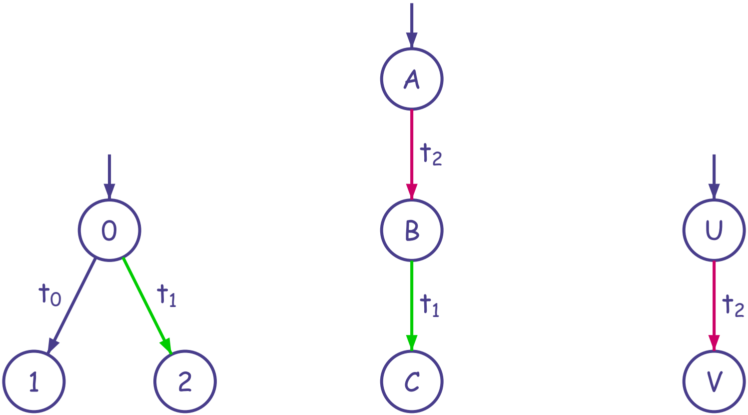
- Unfoldings for Network of Automata
- Symbolic Unfoldings for Network of Timed Automata
- Conclusion

Outline

- Unfoldings for Network of Automata
- Symbolic Unfoldings for Network of Timed Automata
- Conclusion

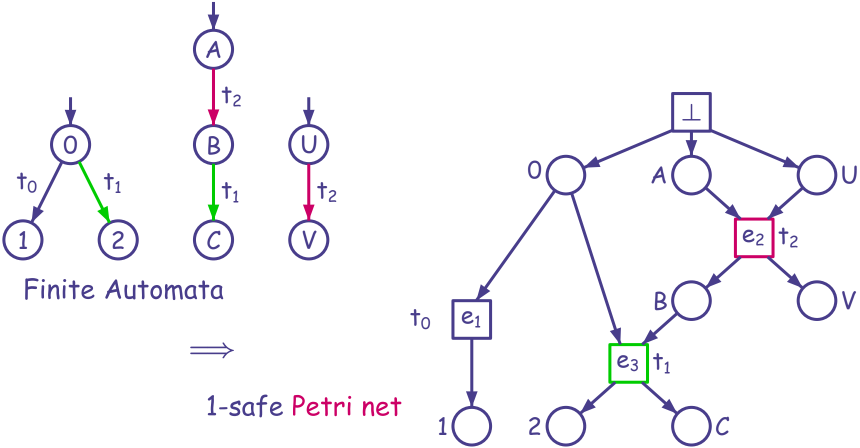
Unfoldings à la McMillan

For Petri Nets [McMillan, FMSD'95]
For Network of Automata [Esparza & Römer, CONCUR'99]



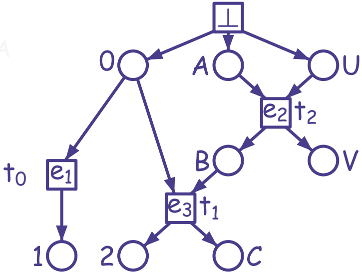
Unfoldings à la McMillan

For Petri Nets [McMillan, FMSD'95]
For Network of Automata [Esparza & Römer, CONCUR'99]



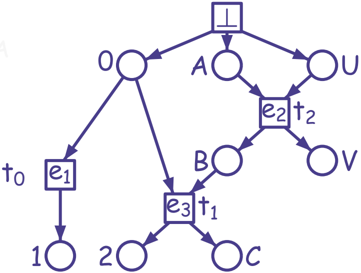
Features of Unfoldings

- Unfolding = 1-safe Petri net
- Finite "good" unfoldings exist
  finite complete prefix
- Preserves concurrency
  size(unfolding) < synchronous product of TA
- Can be constructed efficiently
- Can be used for checking properties:
  - coverability or reachability properties
  - deadlock detection
  - temporal logics properties
- Can be used for diagnosis:
  - Induces a partial order on events
  - Event structure = explanations for set of events



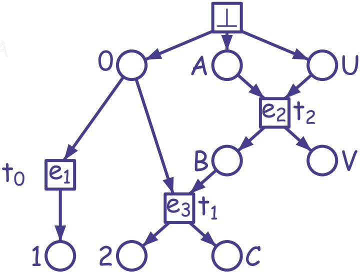
Features of Unfoldings

- Unfolding = 1-safe Petri net
- Finite "good" unfoldings exist
  finite complete prefix
- Preserves concurrency
  size(unfolding) < synchronous product of TA
- Can be constructed efficiently
- Can be used for checking properties:
  - coverability or reachability properties
  - deadlock detection
  - temporal logics properties
- Can be used for diagnosis:
  - Induces a partial order on events
  - Event structure = explanations for set of events



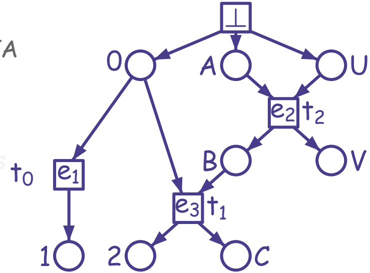
Features of Unfoldings

- Unfolding = 1-safe Petri net
- Finite "good" unfoldings exist
  finite complete prefix
- Preserves concurrency
  size(unfolding) < synchronous product of TA
- Can be constructed efficiently
- Can be used for checking properties:
  - coverability or reachability properties
  - deadlock detection
  - temporal logics properties
- Can be used for diagnosis:
  - Induces a partial order on events
  - Event structure = explanations for set of events



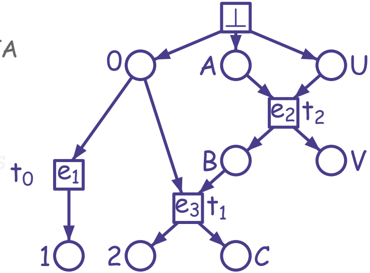
Features of Unfoldings

- Unfolding = 1-safe Petri net
- Finite "good" unfoldings exist
  finite complete prefix
- Preserves concurrency
  size(unfolding) < synchronous product of TA
- Can be constructed efficiently
- Can be used for checking properties:
  - coverability or reachability properties
  - deadlock detection
  - temporal logics properties
- Can be used for diagnosis:
  - Induces a partial order on events
  - Event structure = explanations for set of events



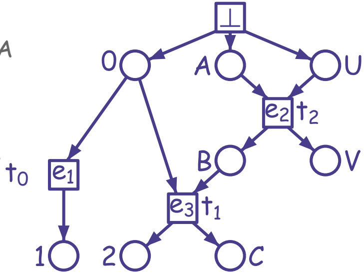
Features of Unfoldings

- Unfolding = 1-safe Petri net
- Finite "good" unfoldings exist
  finite complete prefix
- Preserves concurrency
  size(unfolding) < synchronous product of TA
- Can be constructed efficiently
- Can be used for checking properties:
  - coverability or reachability properties
  - deadlock detection
  - temporal logics properties
- Can be used for diagnosis:
  - Induces a partial order on events
  - Event structure = explanations for set of events



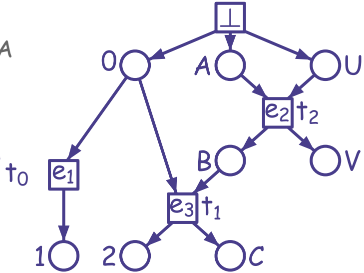
Features of Unfoldings

- Unfolding = 1-safe Petri net
- Finite "good" unfoldings exist
  finite complete prefix
- Preserves concurrency
  size(unfolding) < synchronous product of TA
- Can be constructed efficiently
- Can be used for checking properties:
  - coverability or reachability properties
  - deadlock detection
  - temporal logics properties
- Can be used for diagnosis:
  - Induces a partial order on events
  - Event structure = explanations for set of events



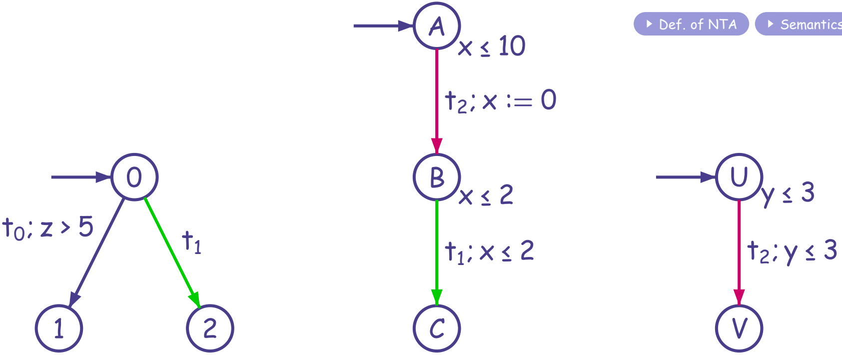
Features of Unfoldings

- Unfolding = 1-safe Petri net
- Finite "good" unfoldings exist
  finite complete prefix
- Preserves concurrency
  size(unfolding) < synchronous product of TA
- Can be constructed efficiently
- Can be used for checking properties:
  - coverability or reachability properties
  - deadlock detection
  - temporal logics properties
- Can be used for diagnosis:
  - Induces a partial order on events
  - Event structure = explanations for set of events



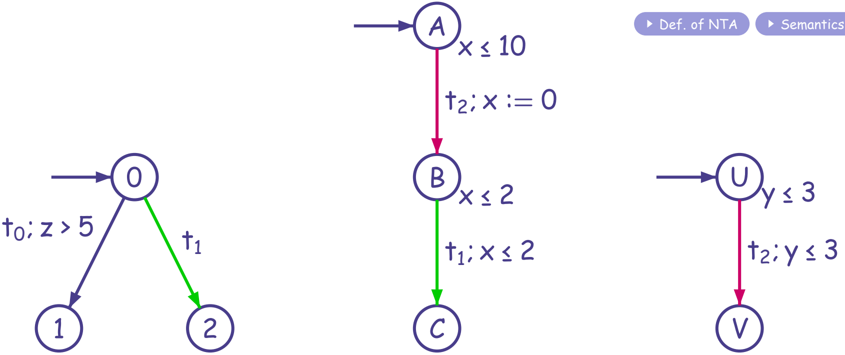
Network of Timed Automata

▸ Def. of NTA ▸ Semantics of NTA



Clocks are NOT shared

Network of Timed Automata



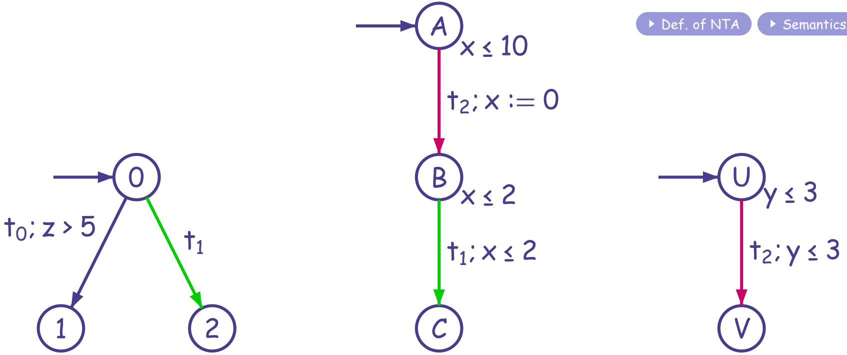
Def. of NTA
Semantics of NTA

State of a NTA: $((1, A, U), x = 1, y = 1, z = 1)$
Symbolic state: $((1, A, U), x = y = z \wedge y \leq 3)$

Clocks are NOT shared

Network of Timed Automata

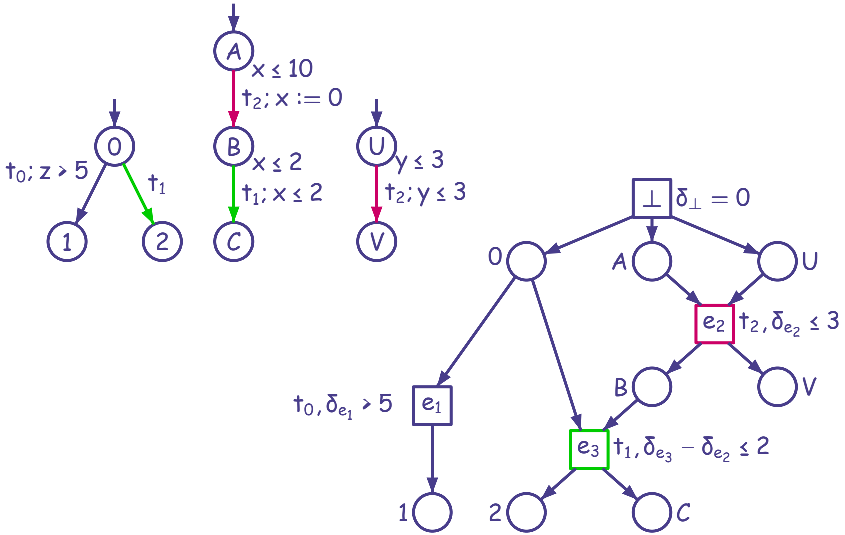
Def. of NTA | Semantics of NTA



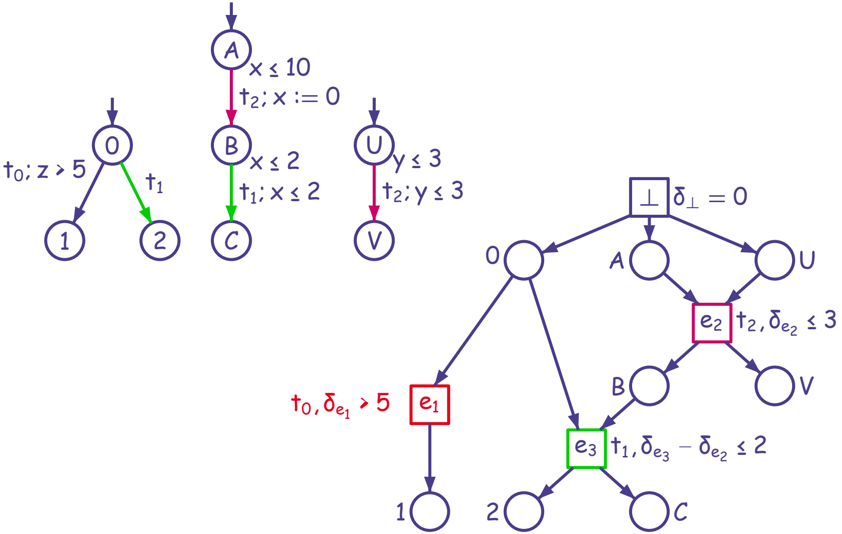
State of a NTA: $((1, A, U), x = 1, y = 1, z = 1)$
Symbolic state: $((1, A, U), x = y = z \wedge y \le 3)$

Clocks are NOT shared

Unfoldings for Network of Timed Automata ?



Unfoldings for Network of Timed Automata ?



Related Work

- Unfoldings for Network of Timed Automata (NTA)
  - [Bengtsson et al., CONCUR'99, Minea, CONCUR'99]:
    semantics of NTA based on local time elapsing, assumption of time-stop freeness
  - [Lugiez et al., TACAS'04]:
    independence between transitions
    symbolic states have more clocks than the NTA
  - [Ben Salah, CONCUR'06]: interleavings preserve union of zones
    Applied to efficient model-checking of Timed Automata
- Unfoldings for Time Petri Nets (TPNs)
  - [Aura-Lilius, TCS'00]: Process Semantics for TPNs
    check realizability of a timed configuration
    Apply only to restricted types of TPNs (e.g. Free Choice)
  - [Fleischhack-Stehno, ICATPN'02]: Discrete Time + Unfolding
  - [Chatain-Jard, ICATPN'06]: Symbolic unfoldings for TPNs
    In TPNs transitions are urgent not in TA

Related Work

- Unfoldings for Network of Timed Automata (NTA)
  - **[Bengtsson et al., CONCUR'99, Minea, CONCUR'99]**:
    semantics of NTA based on local time elapsing, assumption of time-stop freeness
  - **[Lugiez et al., TACAS'04]**:
    independence between transitions
    symbolic states have more clocks than the NTA
  - **[Ben Salah, CONCUR'06]**: interleavings preserve union of zones
    Applied to efficient model-checking of Timed Automata
- Unfoldings for Time Petri Nets (TPNs)
  - **[Aura-Lilius, TCS'00]**: Process Semantics for TPNs
    check realizability of a timed configuration
    Apply only to restricted types of TPNs (e.g. Free Choice)
  - **[Fleischhack-Stehno, ICATPN'02]**: Discrete Time + Unfolding
  - **[Chatain-Jard, ICATPN'06]**: Symbolic unfoldings for TPNs
    In TPNs transitions are urgent not in TA

Related Work

- Unfoldings for Network of Timed Automata (NTA)
  - **[Bengtsson et al., CONCUR'99, Minea, CONCUR'99]**:
    semantics of NTA based on local time elapsing, assumption of time-stop freeness
  - **[Lugiez et al., TACAS'04]**:
    independence between transitions
    symbolic states have more clocks than the NTA
  - [Ben Salah, CONCUR'06]: interleavings preserve union of zones
    Applied to efficient model-checking of Timed Automata
- Unfoldings for Time Petri Nets (TPNs)
  - [Aura-Lilius, TCS'00]: Process Semantics for TPNs
    check realizability of a timed configuration
    Apply only to restricted types of TPNs (e.g. Free Choice)
  - [Fleischhack-Stehno, ICATPN'02]: Discrete Time + Unfolding
  - [Chatain-Jard, ICATPN'06]: Symbolic unfoldings for TPNs
    In TPNs transitions are urgent not in TA

Related Work

- Unfoldings for Network of Timed Automata (NTA)
  - **[Bengtsson et al., CONCUR'99, Minea, CONCUR'99]**: semantics of NTA based on local time elapsing, assumption of time-stop freeness
  - **[Lugiez et al., TACAS'04]**: independence between transitions
    symbolic states have more clocks than the NTA
  - **[Ben Salah, CONCUR'06]**: interleavings preserve union of zones
    Applied to efficient model-checking of Timed Automata
- Unfoldings for Time Petri Nets (TPNs)
  - **[Aura-Lilius, TCS'00]**: Process Semantics for TPNs
    check realizability of a timed configuration
    Apply only to restricted types of TPNs (e.g. Free Choice)
  - **[Fleischhack-Stehno, ICATPN'02]**: Discrete Time + Unfolding
  - **[Chatain-Jard, ICATPN'06]**: Symbolic unfoldings for TPNs
    In TPNs transitions are urgent not in TA

Related Work

- Unfoldings for Network of Timed Automata (NTA)
  - **[Bengtsson et al., CONCUR'99, Minea, CONCUR'99]**:
    semantics of NTA based on local time elapsing, assumption of time-stop freeness
  - **[Lugiez et al., TACAS'04]**:
    independence between transitions
    symbolic states have more clocks than the NTA
  - **[Ben Salah, CONCUR'06]**: interleavings preserve union of zones
    Applied to efficient model-checking of Timed Automata
- Unfoldings for Time Petri Nets (TPNs)
  - **[Aura-Lilius, TCS'00]**: Process Semantics for TPNs
    check realizability of a timed configuration
    Apply only to restricted types of TPNs (e.g. Free Choice)
  - **[Fleischhack-Stehno, ICATPN'02]**: Discrete Time + Unfolding
  - **[Chatain-Jard, ICATPN'06]**: Symbolic unfoldings for TPNs
    In TPNs transitions are urgent not in TA

Related Work

- Unfoldings for Network of Timed Automata (NTA)
  - **[Bengtsson et al., CONCUR'99, Minea, CONCUR'99]:**
    semantics of NTA based on local time elapsing, assumption of time-stop freeness
  - **[Lugiez et al., TACAS'04]:**
    independence between transitions
    symbolic states have more clocks than the NTA
  - **[Ben Salah, CONCUR'06]:** interleavings preserve union of zones
    Applied to efficient model-checking of Timed Automata
- Unfoldings for Time Petri Nets (TPNs)
  - **[Aura-Lilius, TCS'00]:** Process Semantics for TPNs
    check realizability of a timed configuration
    Apply only to restricted types of TPNs (e.g. Free Choice)
  - **[Fleischhack-Stehno, ICATPN'02]:** Discrete Time + Unfolding
  - **[Chatain-Jard, ICATPN'06]:** Symbolic unfoldings for TPNs
    In TPNs transitions are urgent not in TA

Related Work

- Unfoldings for Network of Timed Automata (NTA)
  - **[Bengtsson et al., CONCUR'99, Minea, CONCUR'99]**:
    semantics of NTA based on local time elapsing, assumption of time-stop freeness
  - **[Lugiez et al., TACAS'04]**:
    independence between transitions
    symbolic states have more clocks than the NTA
  - **[Ben Salah, CONCUR'06]**: interleavings preserve union of zones
    Applied to efficient model-checking of Timed Automata
- Unfoldings for Time Petri Nets (TPNs)
  - **[Aura-Lilius, TCS'00]**: Process Semantics for TPNs
    check realizability of a timed configuration
    Apply only to restricted types of TPNs (e.g. Free Choice)
  - **[Fleischhack-Stehno, ICATPN'02]**: Discrete Time + Unfolding
  - **[Chatain-Jard, ICATPN'06]**: Symbolic unfoldings for TPNs
    In TPNs transitions are urgent not in TA

Related Work

- Unfoldings for Network of Timed Automata (NTA)
  - **[Bengtsson et al., CONCUR'99, Minea, CONCUR'99]**:
    semantics of NTA based on local time elapsing, assumption of time-stop freeness
  - **[Lugiez et al., TACAS'04]**:
    independence between transitions
    symbolic states have more clocks than the NTA
  - **[Ben Salah, CONCUR'06]**: interleavings preserve union of zones
    Applied to efficient model-checking of Timed Automata
- Unfoldings for Time Petri Nets (TPNs)
  - **[Aura-Lilius, TCS'00]**: Process Semantics for TPNs
    check realizability of a timed configuration
    Apply only to restricted types of TPNs (e.g. Free Choice)
  - **[Fleischhack-Stehno, ICATPN'02]**: Discrete Time + Unfolding
  - **[Chatain-Jard, ICATPN'06]**: Symbolic unfoldings for TPNs
    In TPNs transitions are urgent not in TA

Objectives & Results of this Paper

- Our Main Goal: give a concurrent semantics for NTA
  - Model for a concurrent semantics for timed systems
  - Define what is the concurrent semantics of a NTA
  - Finite representation
- Requirements for the concurrent semantics:
  - Preserves the concurrency of the system
  - Can be constructed efficiently
  - Allows to check basic properties (e.g. reachability)
- Results:
  - Model: 1-safe Petri nets with read arcs and timing information
    Symbolic Unfolding
  - An algorithm to compute a symbolic unfolding of a NTA
  - Finite complete prefixes (of the unfolding) exist
    no canonical representative
  - Concurrency preserved
  - Reachability is easily decidable

Objectives & Results of this Paper

- Our Main Goal: give a concurrent semantics for NTA
  - Model for a concurrent semantics for timed systems
  - Define what is the concurrent semantics of a NTA
  - Finite representation
- Requirements for the concurrent semantics:
  - Preserves the concurrency of the system
  - Can be constructed efficiently
  - Allows to check basic properties (e.g. reachability)
- Results:
  - Model: 1-safe Petri nets with read arcs and timing information
    Symbolic Unfolding
  - An algorithm to compute a symbolic unfolding of a NTA
  - Finite complete prefixes (of the unfolding) exist
    no canonical representative
  - Concurrency preserved
  - Reachability is easily decidable

Objectives & Results of this Paper

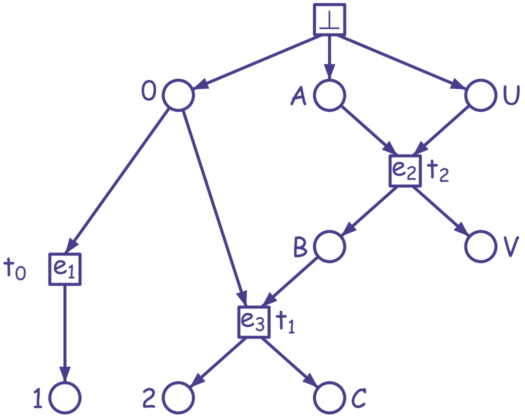
- Our Main Goal: give a concurrent semantics for NTA
  - Model for a concurrent semantics for timed systems
  - Define what is the concurrent semantics of a NTA
  - Finite representation
- Requirements for the concurrent semantics:
  - Preserves the concurrency of the system
  - Can be constructed efficiently
  - Allows to check basic properties (e.g. reachability)
- Results:
  - Model: 1-safe Petri nets with read arcs and timing information
    Symbolic Unfolding
  - An algorithm to compute a symbolic unfolding of a NTA
  - Finite complete prefixes (of the unfolding) exist
    no canonical representative
  - Concurrency preserved
  - Reachability is easily decidable

Outline

- Unfoldings for Network of Automata
- **Symbolic Unfoldings for Network of Timed Automata**
- Conclusion

Basics: Configurations, Co-sets, Cuts

- Configuration: feasible set of events; past-closed
- Co-set: feasible set of places
- Cut: maximal co-set



Symbolic Unfolding - Step 1

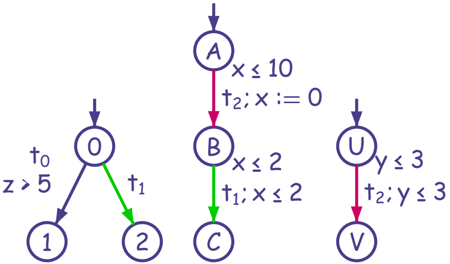
Symbolic Unfolding - Step 1

1. Build the unfolding of the underlying untimed network

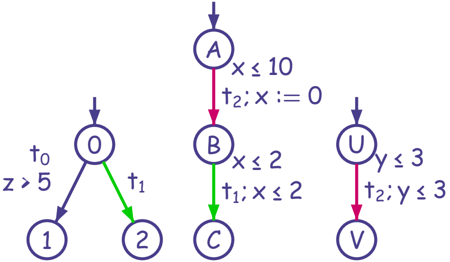
Symbolic Unfolding - Step 1

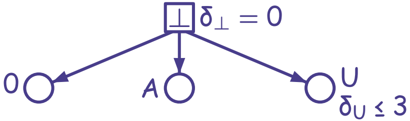
1. Build the unfolding of the underlying untimed network
2. Annotate it with timing constraints on events

Symbolic Unfolding - Step 1

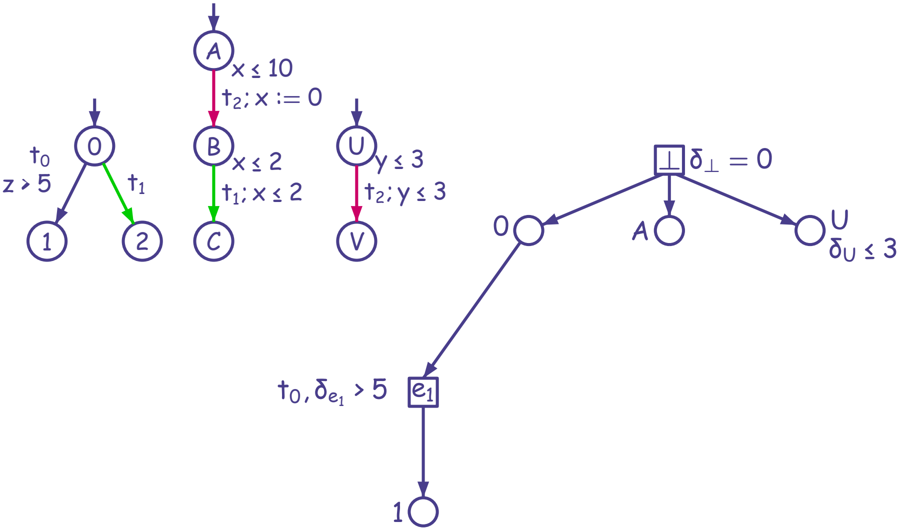


Symbolic Unfolding - Step 1

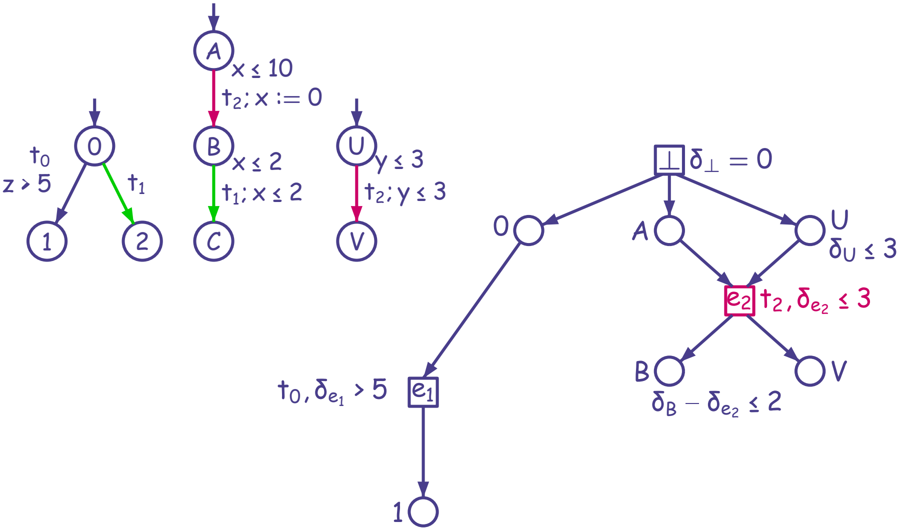




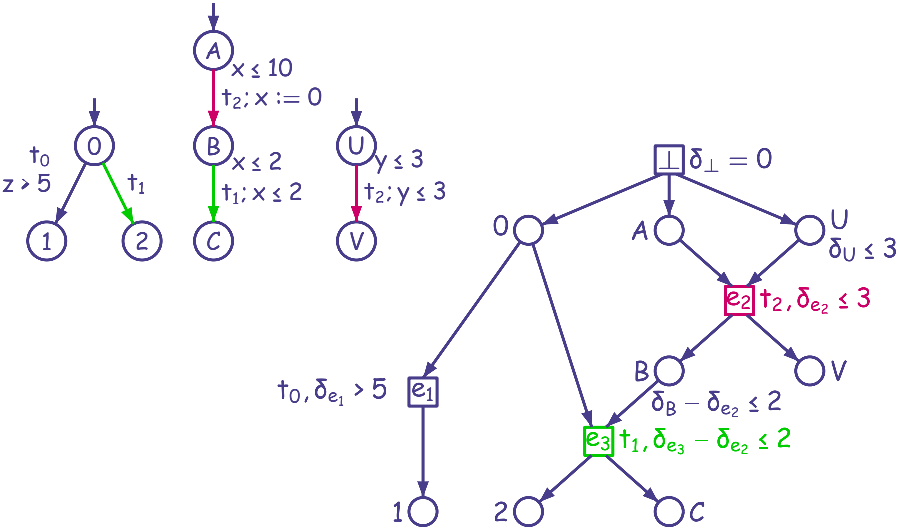
Symbolic Unfolding - Step 1



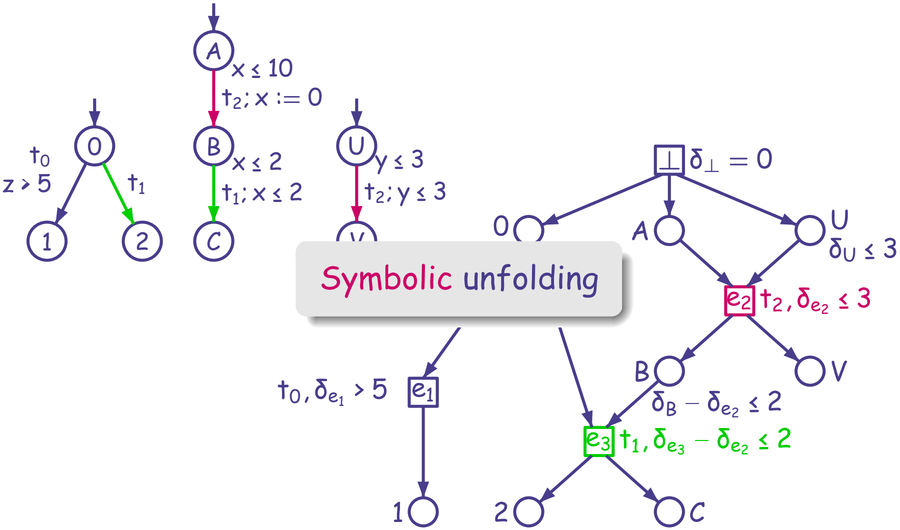
Symbolic Unfolding - Step 1



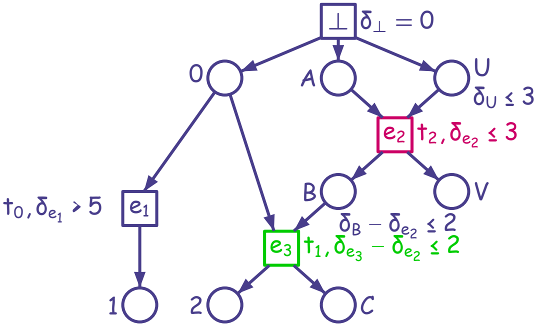
Symbolic Unfolding - Step 1



Symbolic Unfolding - Step 1



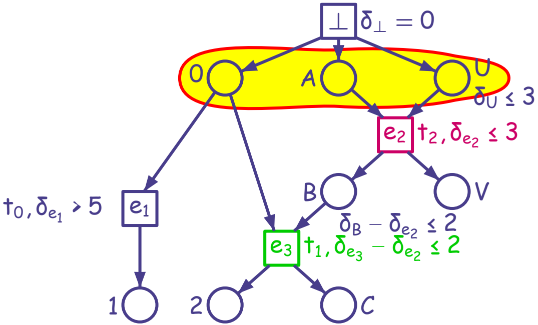
Properties of the Symbolic Unfolding



- Symbolic Cuts: $(C, \Phi(C))$
  $\Phi(C)$ is a constraint on the global time δ at which tokens can be in C

Def. of Symbolic Cuts

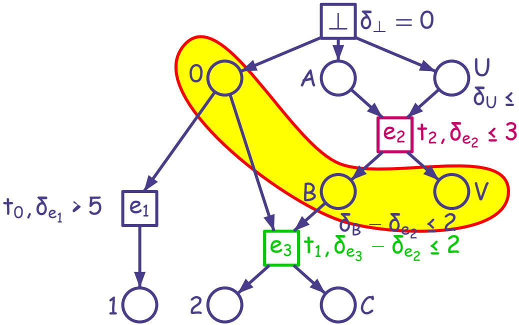
Properties of the Symbolic Unfolding



- Symbolic Cuts: $(C, \Phi(C))$ ▸ Def. of Symbolic Cuts
  $\Phi(C)$ is a constraint on the global time δ at which tokens can be in C

$$(O, A, U), \delta \leq 3 \qquad (\delta = \delta_O = \delta_A = \delta_U \wedge \delta_U \leq 3)$$

Properties of the Symbolic Unfolding

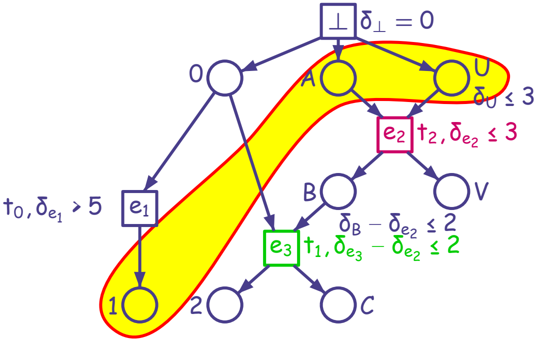


- **Symbolic Cuts**: $(C, \Phi(C))$
  $\Phi(C)$ is a constraint on the global time δ at which tokens can be in C

Def. of Symbolic Cuts

$$(O, B, V), \delta \geq \delta_{e_2} \wedge \delta - \delta_{e_2} \leq 2 \wedge \delta_{e_2} \leq 3$$

Properties of the Symbolic Unfolding

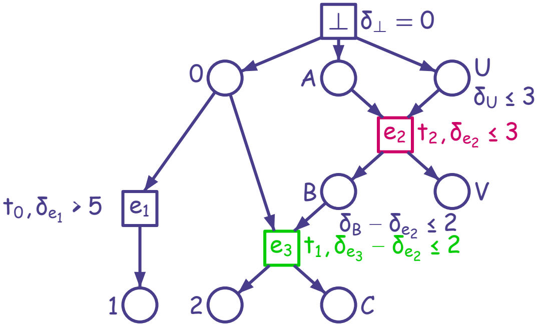


- Symbolic Cuts: $(C, \Phi(C))$
  $\Phi(C)$ is a constraint on the global time δ at which tokens can be in C

Def. of Symbolic Cuts

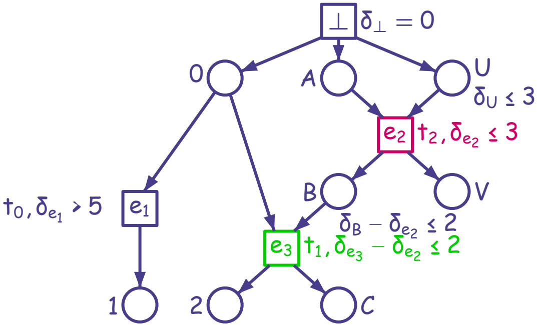
$$(1, A, U), \delta \geq \delta_{e_1} \wedge \delta_{e_1} > 5 \wedge \delta \leq 3$$

Properties of the Symbolic Unfolding



- one-to-one mapping f:
  symbolic cut $(C, \Phi(C)) \iff \cup_{p \in paths}(\vec{C}, Z_p)$ symbolic state of the NTA

Properties of the Symbolic Unfolding



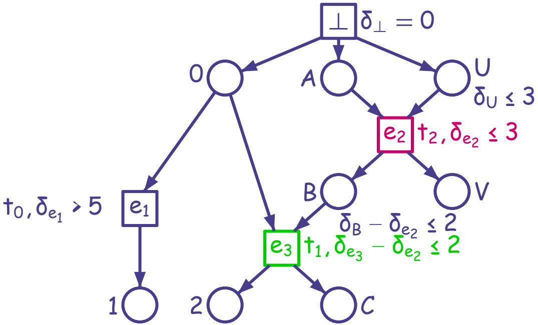
- one-to-one mapping f:
  symbolic cut $(C, \Phi(C)) \Longleftrightarrow \cup_{p \in paths}(\vec{C}, Z_p)$ symbolic state of the NTA

Theorem

For each cut C, $\Phi(C)$ is a zone. f preserves zones. $\cup_p(\vec{C}, Z_p)$ is a zone.

Gives an alternative proof of the result in **[Ben Salah, CONCUR'06]**

Properties of the Symbolic Unfolding



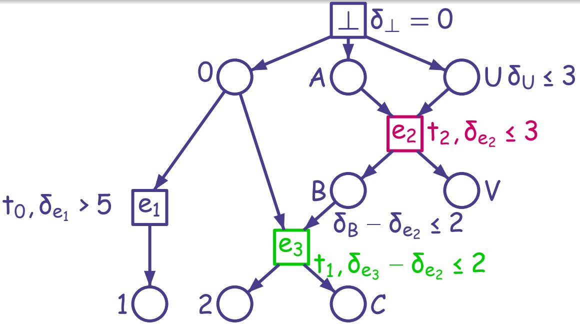
Theorem

For each cut C, $\Phi(C)$ is a zone. f preserves zones. $\cup_p(\vec{C}, Z_p)$ is a zone.

Theorem (Finite Complete Prefix)

Finite Complete prefixes exist for Network of Timed Automata.

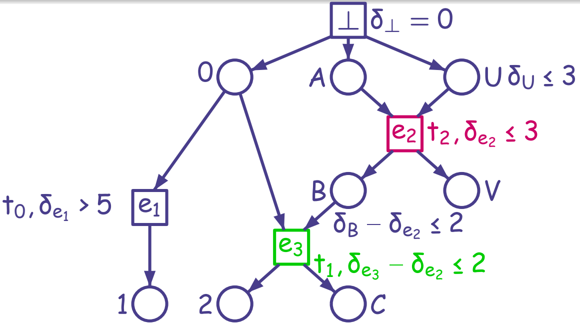
Properties of the Symbolic Unfolding (Cont'd)



- Allows to check (non) emptyness of a symbolic cut: $[\![\Phi(C)]\!] \neq \emptyset$
  And thus check that a set of events can be extended to a configuration sub-configuration
  $\Longrightarrow$ try all the possible cuts ... not efficient
- Timed Automata version of the work of [Aura-Lilius, TCS'00]

How to check directly that a set of events is a sub-configuration ?

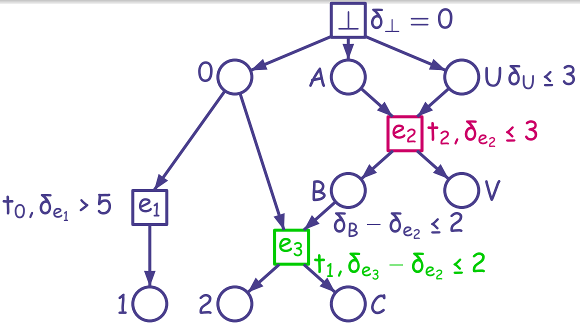
Properties of the Symbolic Unfolding (Cont'd)



- Allows to check (non) emptyness of a symbolic cut: $[\![\Phi(C)]\!] \neq \emptyset$
  And thus check that a set of events can be extended to a configuration
  sub-configuration
  $\Longrightarrow$ try all the possible cuts ...not efficient
- Timed Automata version of the work of [Aura-Lilius, TCS'00]

How to check directly that a set of events is a sub-configuration ?

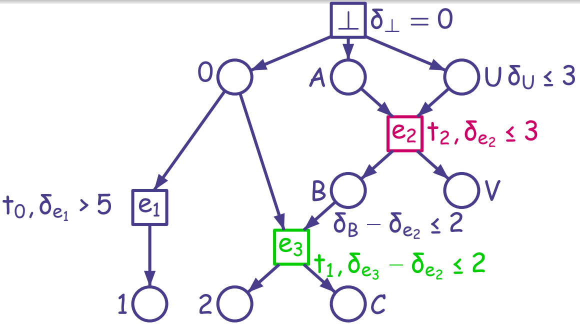
Properties of the Symbolic Unfolding (Cont'd)



- Allows to check (non) emptyness of a symbolic cut: $[\![\Phi(C)]\!] \neq \emptyset$
  And thus check that a set of events can be extended to a configuration sub-configuration
  $\Longrightarrow$ try all the possible cuts ... not efficient
- Timed Automata version of the work of **[Aura-Lilius, TCS'00]**

How to check directly that a set of events is a sub-configuration ?

Properties of the Symbolic Unfolding (Cont'd)

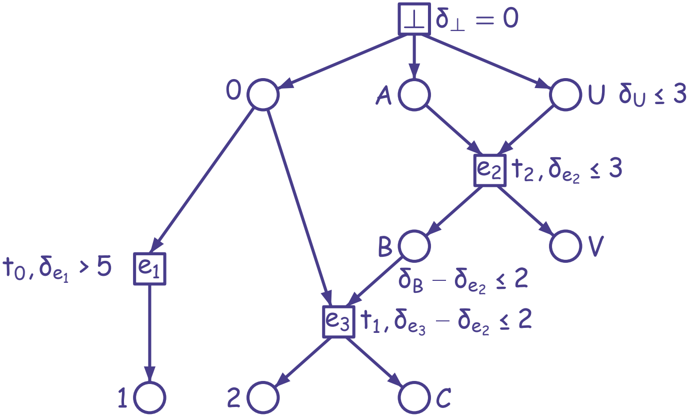


- Allows to check (non) emptyness of a symbolic cut: $[\![\Phi(C)]\!] \neq \emptyset$
  And thus check that a set of events can be extended to a configuration sub-configuration
  $\Longrightarrow$ try all the possible cuts ... not efficient
- Timed Automata version of the work of **[Aura-Lilius, TCS'00]**

How to check directly that a set of events is a sub-configuration ?

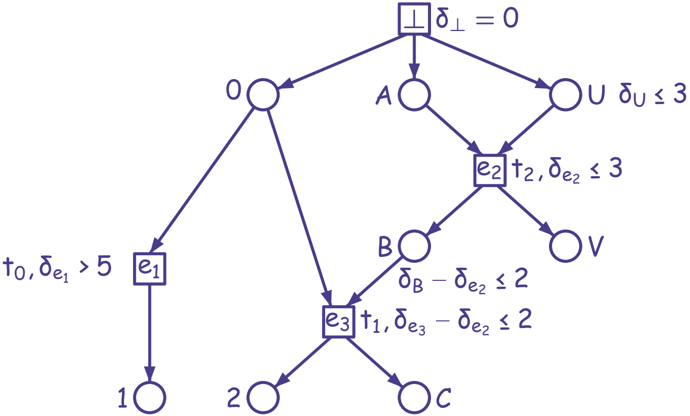
Symbolic Unfoldings – Step 2

- constraints on firing e_1 depends on the cuts that enable e_1



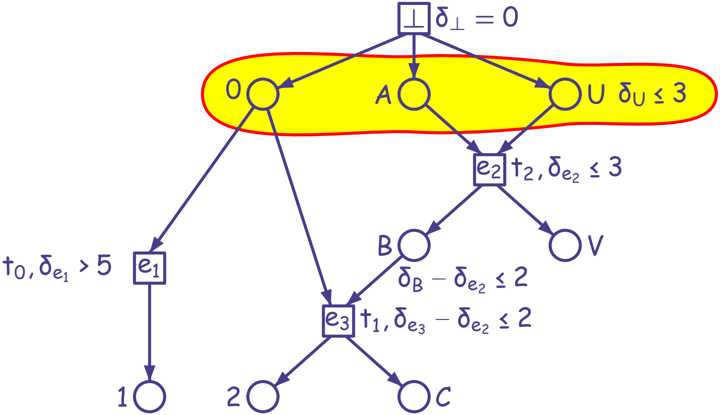
Symbolic Unfoldings – Step 2

- constraints on firing e_1 depends on the cuts that enable e_1



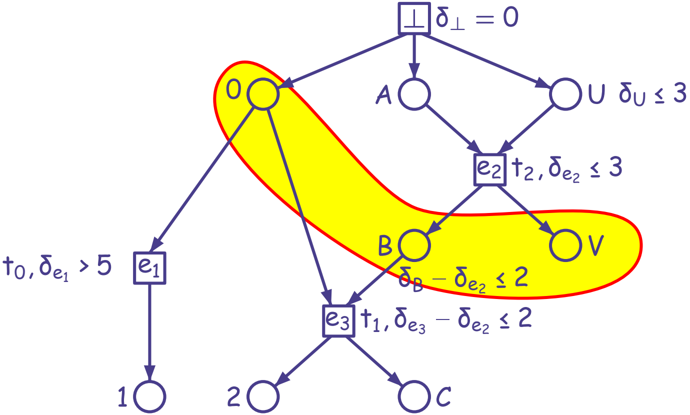
Symbolic Unfoldings – Step 2

- constraints on firing e_1 depends on the cuts that enable e_1
  for $(0, A, U)$: $\delta_{e_1} \leq 3$



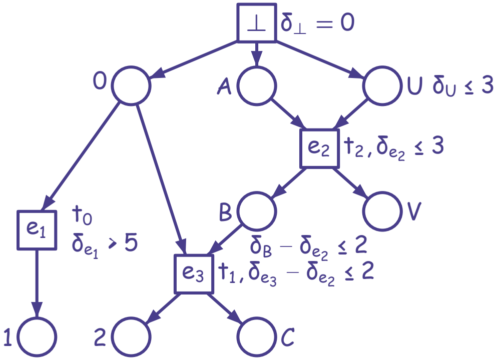
Symbolic Unfoldings – Step 2

- constraints on firing e_1 depends on the cuts that enable e_1
  for $(0, B, V)$: $\delta_{e_1} - \delta_{e_2} \leq 2$. $(\delta_{e_1} = \delta_0 = \delta_B = \delta_V \wedge \delta_B - \delta_{e_2} \leq 2)$



From Symbolic Unfolding to Extended Unfolding

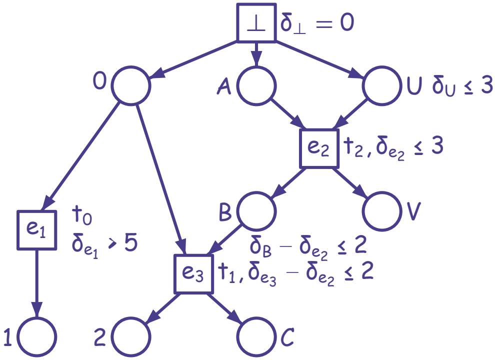
- for each event e of the symbolic unfolding:
  1. $C(e)$ = set of enabling cuts of e
  2. compute the set of constraints generated by $C(e)$ on the (global) firing time δ of e



$$C(e_1) = \{(0, A, U), (0, B, V)\}$$
$$C(e_2) = \{(0, A, U), (1, A, U)\}$$

From Symbolic Unfolding to Extended Unfolding

- for each event e of the symbolic unfolding:
  1. $C(e)$ = set of enabling cuts of e
  2. compute the set of constraints generated by $C(e)$ on the (global) firing time δ of e



$C(e_1) = \{(0, A, U), (0, B, V)\}$

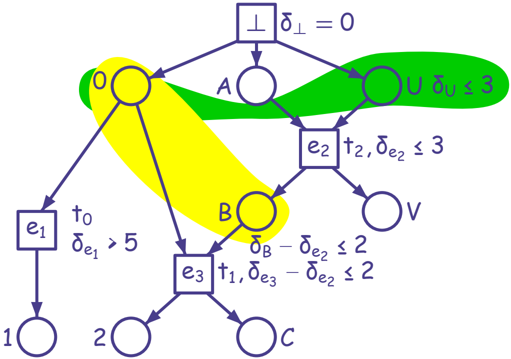
- $(0, B, V)$:
  $\delta - \delta_{e_2} \leq 2 \wedge 0 \leq \delta_{e_2} \leq 3 \wedge \delta \geq \delta_{e_2}$
- $(0, A, U)$: $\delta \leq 3$

$C(e_2) = \{(0, A, U), (1, A, U)\}$

- $(1, A, U)$: $\delta \leq 3$
- $(0, A, U)$: $\delta \leq 3$

From Symbolic Unfolding to Extended Unfolding

- for each event e of the symbolic unfolding:
  1. $C(e)$ = set of enabling cuts of e
  2. compute the set of constraints generated by $C(e)$ on the (global) firing time δ of e



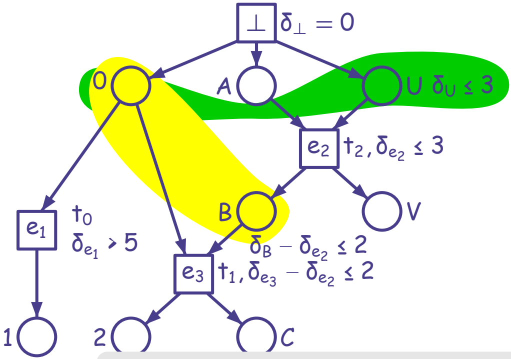
$C(e_1) = \{(0, A, U), (0, B, V)\}$

- $(0, B, V)$:
  $\delta - \delta_{e_2} \le 2 \wedge 0 \le \delta_{e_2} \le 3 \wedge \delta \ge \delta_{e_2}$
- $(0, A, U)$: $\delta \le 3$

For $C(e_1)$, $(0, B)$ and $(0, U)$ generates the good constraints

From Symbolic Unfolding to Extended Unfolding

- for each event e of the symbolic unfolding:
  1. $C(e)$ = set of enabling cuts of e
  2. compute the set of constraints generated by $C(e)$ on the (global) firing time δ of e



$C(e_1) = \{(O, A, U), (O, B, V)\}$

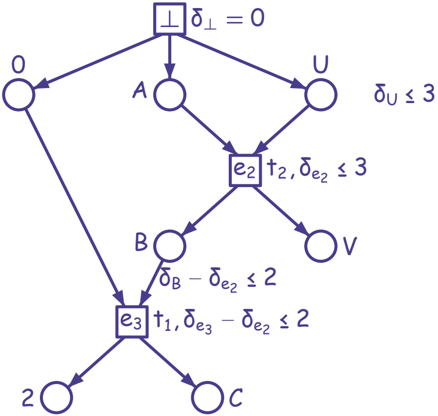
- (O, B, V):
  $\delta - \delta_{e_2} \leq 2 \wedge 0 \leq \delta_{e_2} \leq 3 \wedge \delta \geq \delta_{e_2}$
- (O, A, U): $\delta \leq 3$

For $C(e_1)$, (O, B) and (O, U) generates the good constraints

(O, B) and (O, U) are Safe Representatives for event e_1

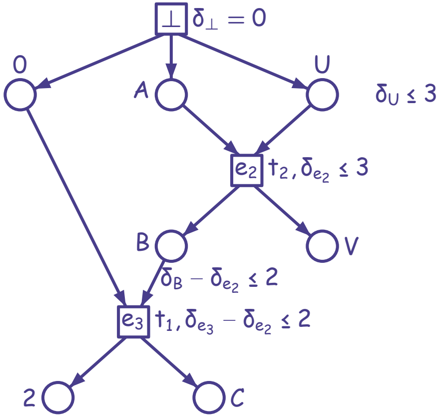
Extended Symbolic Unfolding

- to add an event e to a prefix:
  1. find a co-set C containing $^\bullet e$ and extend it to a safe representative S
     for e_1: $(0, U)$ and $(0, B)$
  2. use normal arcs from C to e and read arcs from $S \setminus C$ to e



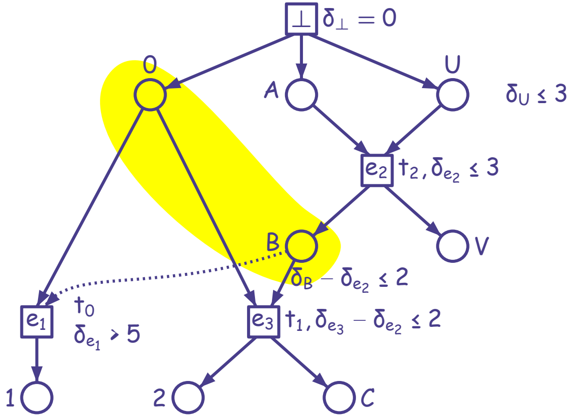
Extended Symbolic Unfolding

- to add an event e to a prefix:
  1. find a co-set C containing $^\bullet e$ and extend it to a safe representative S
     for e_1: (O, U) and (O, B)
  2. use normal arcs from C to e and read arcs from $S \setminus C$ to e



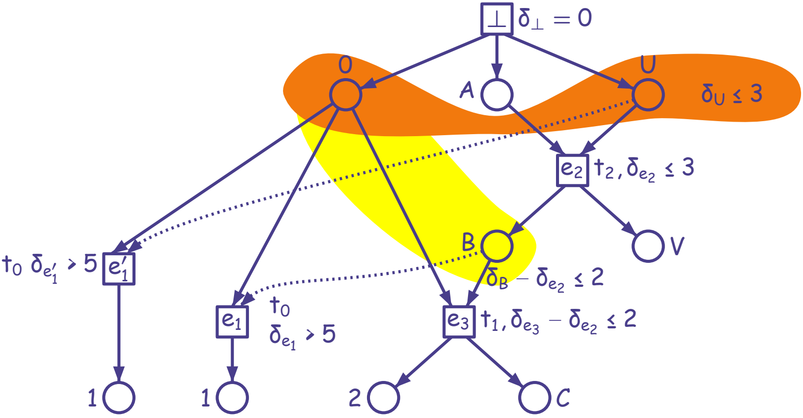
Extended Symbolic Unfolding

- to add an event e to a prefix:
  1. find a co-set C containing $^\bullet e$ and extend it to a safe representative S
     for e_1: $(0, U)$ and $(0, B)$
  2. use normal arcs from C to e and read arcs from $S \setminus C$ to e



Extended Symbolic Unfolding

- to add an event e to a prefix:
  1. find a co-set C containing $^\bullet e$ and extend it to a safe representative S
     for e_1: (O, U) and (O, B)
  2. use normal arcs from C to e and read arcs from $S \setminus C$ to e



Properties of the Extended Unfolding

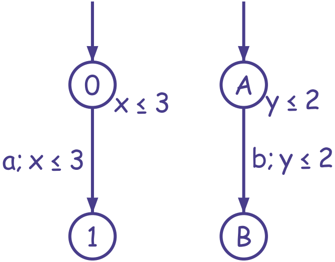
- Complete and Finite Extended Prefixes exists (not unique) even for NTA with "loops"
- Preserves concurrency
- Assumption: no automaton can prevent time from elapsing

Properties of the Extended Unfolding

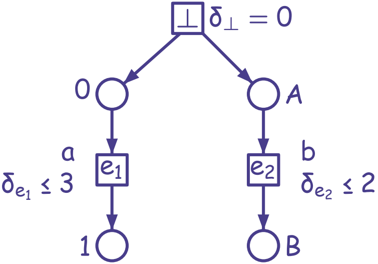
- Complete and Finite Extended Prefixes exists (not unique) even for NTA with "loops"
- Preserves concurrency
- Assumption: no automaton can prevent time from elapsing

Properties of the Extended Unfolding

- Complete and Finite Extended Prefixes exists (not unique) even for NTA with "loops"
- Preserves concurrency
- Assumption: no automaton can prevent time from elapsing



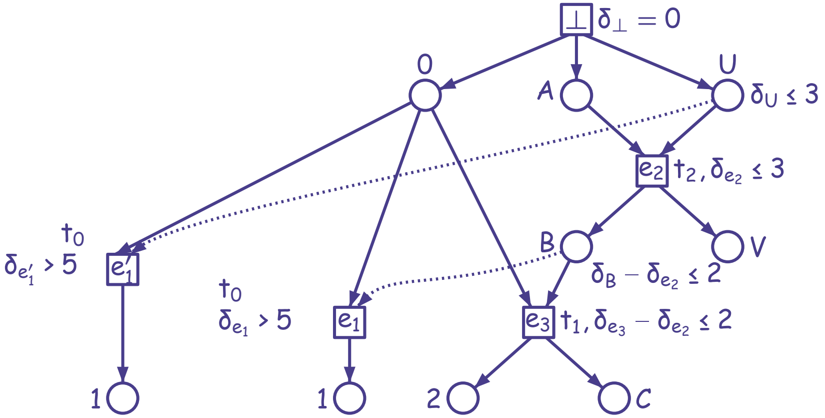
(a) Two Independent Automata



(b) The Unfolding

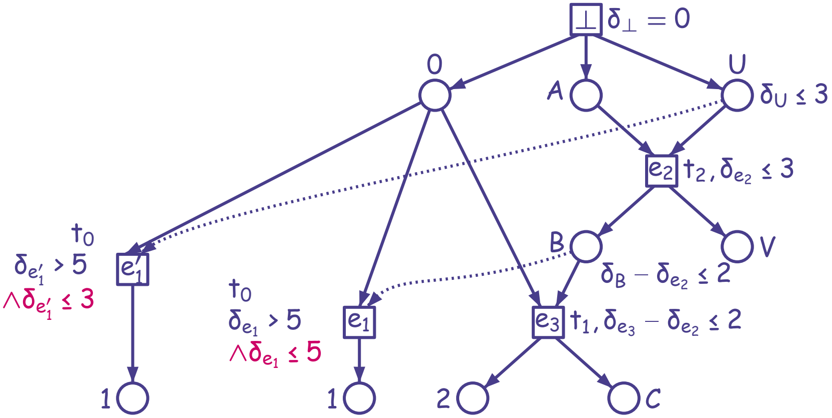
Properties of the Extended Unfolding (Cont'd)

- We can check **directly** that a set of **timed** events can be extended to a **(timed) configuration**



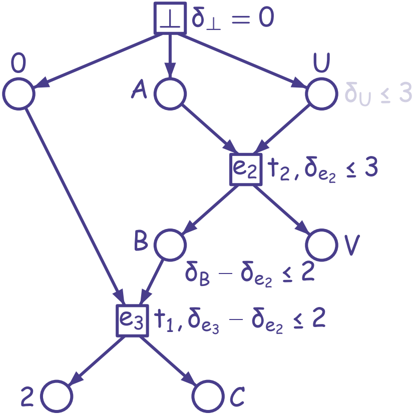
Properties of the Extended Unfolding (Cont'd)

- We can check directly that a set of timed events can be extended to a (timed) configuration



Properties of the Extended Unfolding (Cont'd)

- We can check directly that a set of timed events can be extended to a (timed) configuration



Outline

- Unfoldings for Network of Automata
- Symbolic Unfoldings for Network of Timed Automata
- **Conclusion**

Conclusion & Future Work

- Results:
  - Extended unfoldings for network of TA
    Petri nets with read arcs and timing constraints
    not a unique or canonical unfolding
  - Existence of a finite complete prefixes for Network of Timed Automata
  - Unfolding preserves concurrency
  - Can be used to decide reachability
  - Induces a partial order of timed events
- Future Work:
  - Evaluate the size of the unfolding
  - Build directly the extended unfolding
    In one step
  - Build the unfolding efficiently
  - Compare our approach with [Bouyer-Haddad-Reynier, ATVA'06]

Conclusion & Future Work

- Results:
  - Extended unfoldings for network of TA
    Petri nets with read arcs and timing constraints
    not a unique or canonical unfolding
  - Existence of a finite complete prefixes for Network of Timed Automata
  - Unfolding preserves concurrency
  - Can be used to decide reachability
  - Induces a partial order of timed events
- Future Work:
  - Evaluate the size of the unfolding
  - Build directly the extended unfolding
    In one step
  - Build the unfolding efficiently
  - Compare our approach with [Bouyer-Haddad-Reynier, ATVA'06]

References

[Alur & Dill, TCS'94] Rajeev Alur and David Dill.
A theory of timed automata.
Theoretical Computer Science (TCS), 126(2):183–235, 1994.

[Ben Salah, CONCUR'06] Ramzi Ben Salah, Marius Bozga, and Oded Maler.
On interleaving in timed automata.
In Proceedings of the 17th International Conference on Concurrency Theory (CONCUR'06), volume 4137 of *Lecture Notes in Computer Science* pages 465–476, Springer, august 2006.

[Bouyer-Haddad-Reynier, ATVA'06] Patricia Bouyer, Serge Haddad and Pierre-Alain Reynier.
Timed Unfoldings for Networks of Timed Automata.
In Proceedings of the 4th International Symposium on Automated Technology for Verification and Analysis, 23-26 October 2006, Beijing, China, *Lecture Notes in Computer Science*, Springer, october 2006.

[Chatain-Jard, ICATPN'06] Thomas Chatain and Claude Jard.
Complete finite prefixes of symbolic unfoldings of safe time Petri nets.
In *ICATPN*, volume 4024 of *LNCS*, pages 125–145, june 2006.

[Esparza & Römer, CONCUR'99] Javier Esparza and Stefan Römer.
An unfolding algorithm for synchronous products of transition systems.
In *CONCUR*, volume 1664 of *LNCS*, pages 2–20. Springer, 1999.

[Fleischhack-Stehno, ICATPN'02] Hans Fleischhack and Christian Stehno.
Computing a finite prefix of a time Petri net.
In *ICATPN*, pages 163–181, 2002.

References (cont.)

[Bengtsson et al., CONCUR'99] J. Bengtsson, B. Jonsson, J. Lilius, W. Yi.
Partial order reductions for timed systems.
In *CONCUR 99*, volume 1466 of *LNCS*, pages 485–500, 1999.

[Lugiez et al., TACAS'04] Denis Lugiez, Peter Niebert, and Sarah Zennou.
A partial order semantics approach to the clock explosion problem of timed automata.
In Proc. 10th International Conference on Tools and Algorithms for the Construction and Analysis of Systems (TACAS'2004), volume 2988 of *Lecture Notes in Computer Science*, pages 296–311. Springer, 2004.

[McMillan, FMSD'95] Kenneth L. McMillan.
A technique of state space search based on unfolding.
Formal Methods in System Design, 6(1):45–65, 1995.

[Minea, CONCUR'99] M. Minea.
Partial order reduction for model checking of timed automata.
In *CONCUR 99*, volume 1664 of *LNCS*, pages 431–446, 1999.

[Aura-Lilius, TCS'00] T. Aura and J. Lilius.
A causal semantics for time petri nets.
Theoretical Computer Science, 1-2(243):409–447, 2000.

Timed Automata [Alur & Dill, TCS'94]

A Timed Automaton $\mathcal{A}$ is a tuple $(L, \ell_0, \Sigma, X, Inv, \longrightarrow)$ where:

- L is a finite set of locations
- ℓ_0 is the initial location
- X is a finite set of clocks
- Σ is a finite set of actions
- $\longrightarrow$ is a set of transitions of the form $\ell \xrightarrow{g,a,R} \ell'$ with:
  - $\ell, \ell' \in L$,
  - $a \in \Sigma$
  - a guard g which is a clock constraint over X
  - a reset set R which is the set of clocks to be reset to 0

Clock constraints are boolean combinations of $x \sim k$ with $x \in C$ and $k \in \mathbb{Z}$ and $\sim \in \{\leq, <\}$.

Back

Semantics of Timed Automata

Let $\mathcal{A} = (L, \ell_0, \Sigma, X, \mathrm{Inv}, \longrightarrow)$ be a Timed Automaton.

A **state** (ℓ, v) of $\mathcal{A}$ is in $L \times \mathbb{R}_{\geq 0}^X$

The semantics of $\mathcal{A}$ is a **Timed Transition System** $S_\mathcal{A} = (Q, q_0, \Sigma \cup \mathbb{R}_{\geq 0}, \longrightarrow)$ with:

- $Q = L \times \mathbb{R}_{\geq 0}^X$
- $q_0 = (\ell_0, \overline{0})$
- $\longrightarrow$ consists in:

discrete transition: $(\ell, v) \xrightarrow{a} (\ell', v') \iff \begin{cases} \exists\, \ell \xrightarrow{g,a,r} \ell' \in \mathcal{A} \\ v \models g \\ v' = v[r \leftarrow 0] \\ v' \models \mathrm{Inv}(\ell') \end{cases}$

delay transition: $(\ell, v) \xrightarrow{d} (\ell, v + d) \iff d \in \mathbb{R}_{\geq 0} \wedge v + d \models \mathrm{Inv}(\ell)$

Back

Network of Timed Automata

Back

Let $\mathcal{A}_i = (L_i, \ell_0^i, \Sigma_i, X_i, Inv_i, \longrightarrow_i)$ be Timed Automata.

$L_i \cap L_j = \emptyset$ and $X_i \cap X_j = \emptyset$ $\quad$ $L = \cup_i L_i$ and $X = \cup_i X_i$

A state $(\vec{\ell}, \vec{v})$ of $\mathcal{A}$ is in $L \times \mathbb{R}_{\geq 0}^X$.

Assume each TA has a loop transition $(\ell, true, \varepsilon, \emptyset, \ell)$.

I is the synchonization function.

The network $\mathcal{A} = (\mathcal{A}_1 \times \cdots \times \mathcal{A}_n)_I$ is defined by:

- $Q = L \times \mathbb{R}_{\geq 0}^X$
- $q_0 = (\ell_0, \overline{0})$ with $\ell_0 = (\ell_0^1, \cdots, \ell_0^n)$
- $\longrightarrow$ consists in:

discrete transition: $$(\vec{\ell}, v) \xrightarrow{a} (\vec{\ell}', v') \iff \begin{cases} \exists a = (a_1, \cdots, a_n) \in I \\ \exists\ \ell_i \xrightarrow{g_i, a_i, r_i}_i \ell_i' \in \mathcal{A}_i \\ v \models \wedge_i g_i \\ v' = v[\cup_i v_i \leftarrow 0] \\ v' \models \wedge_i Inv_i(\ell_i') \end{cases}$$

delay transition: $$(\vec{\ell}, v) \xrightarrow{d} (\vec{\ell}, v + d) \iff d \in \mathbb{R}_{\geq 0} \wedge v + d \models \wedge_i Inv_i(\ell_i)$$

Symbolic (or Timed) Cuts

Back

$(C, \Phi(C))$ is a symbolic cut if:

1. C is a untimed cut
2. $\Phi(C) = \Phi_1(C) \wedge \Phi_2(C) \wedge \Phi_3(C) \wedge \Phi_4(C)$ where $\Phi_i(C), 1 \leq i \leq 4$ are defined by:

$$\Phi_1(C) = \bigwedge_{x \in \lceil C \rceil} \gamma(x) \quad (1) \qquad \Phi_3(C) = \bigwedge_{p \in C} (\delta_{\bullet p} \leq \delta_p) \quad (3)$$

$$\Phi_2(C) = \bigwedge_{e \in \lceil C \rceil \cap E} (\wedge_{p \in {}^\bullet e} \delta_p = \delta_e) \quad (2) \qquad \Phi_4(C) = (\bigwedge_{p,p' \in C} \delta_p = \delta_{p'}) \quad (4)$$

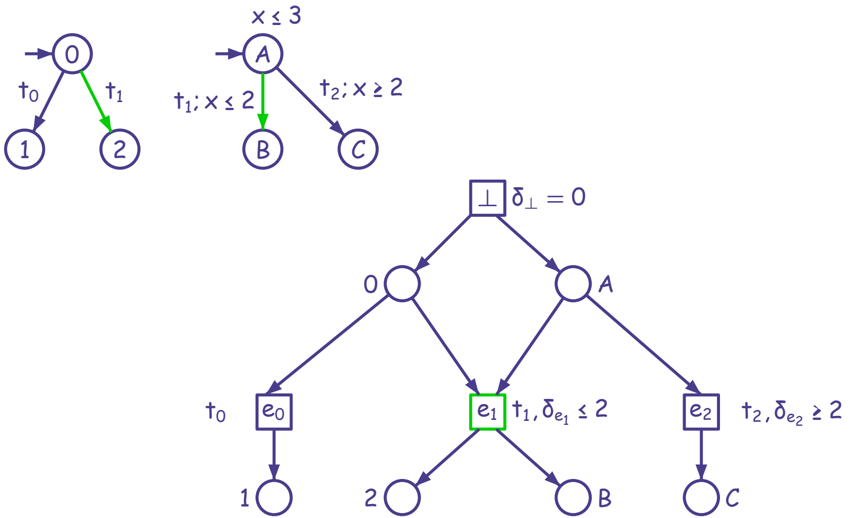
with $\gamma(x)$ the constraint associated with node x.

Let G be the simulation graph of the network of TA and $\mathcal{N}$ be a symbolic unfolding of the NTA

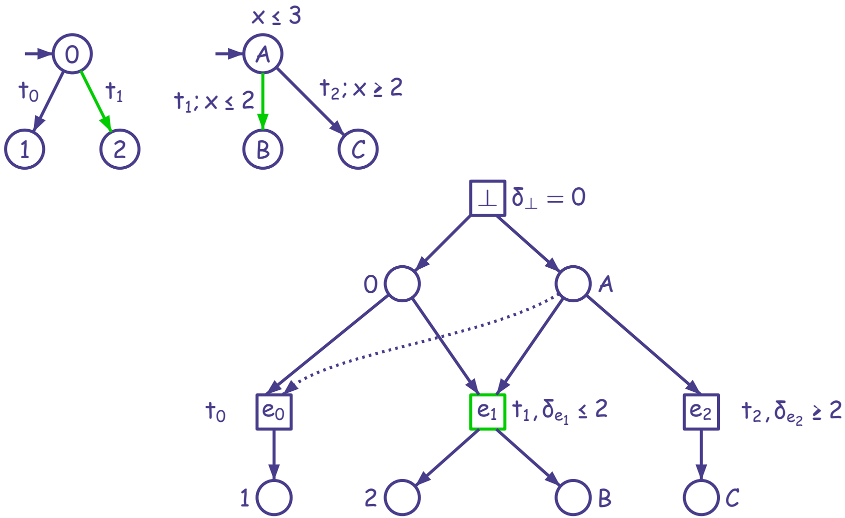
Theorem

$(C, \Phi(C))$ is a symbolic cut of $\mathcal{N}$ and $[\![\Phi(C)]\!] \neq \emptyset$ iff C is reachable in G.

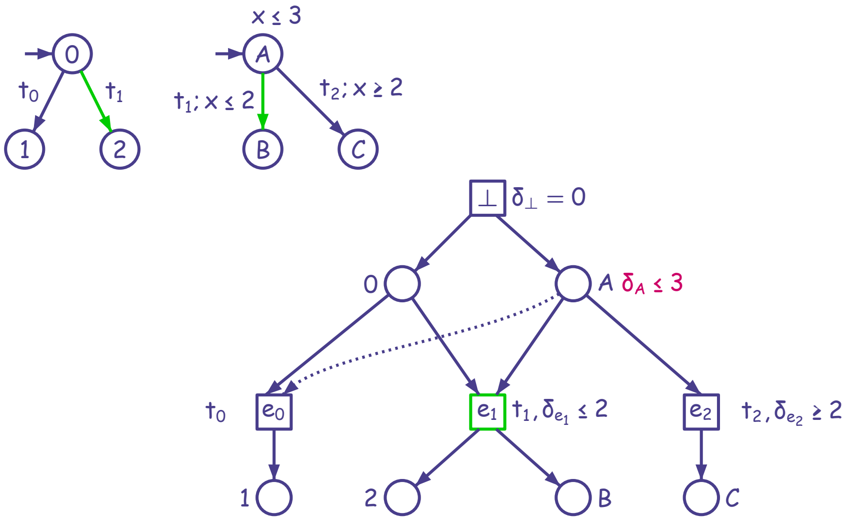
Time Constraints on Places



Time Constraints on Places



Time Constraints on Places



Time Constraints on Places