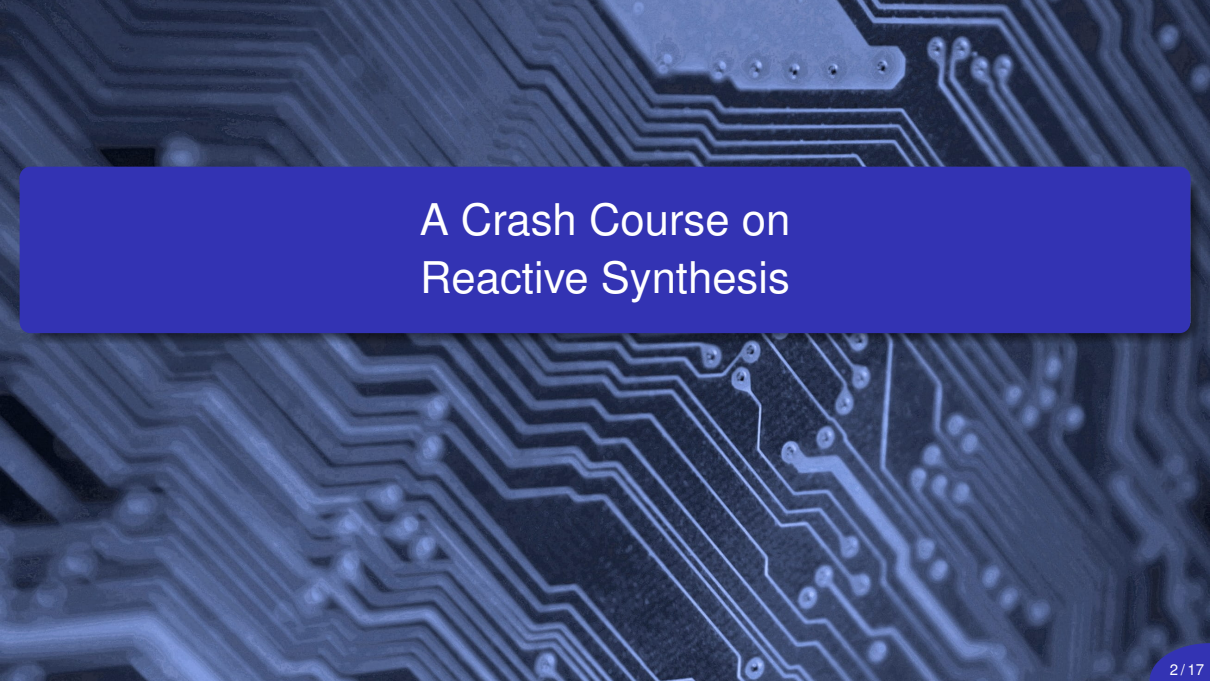


Practical Applications of the Alternating Cycle Decomposition

Antonio Casares Alexandre Duret-Lutz
Klara J. Meyer Florian Renkin Salomon Sickert

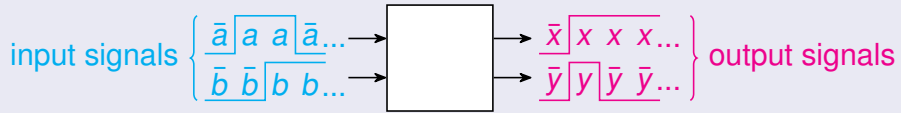
Joint DIMEA and FORMELA seminar — May 23

The background of the slide is a close-up, high-contrast photograph of a printed circuit board (PCB). The image shows a complex network of fine, light-colored conductive traces on a dark substrate. Several circular solder pads and through-hole components are visible, adding to the technical and digital aesthetic of the background.

A Crash Course on Reactive Synthesis

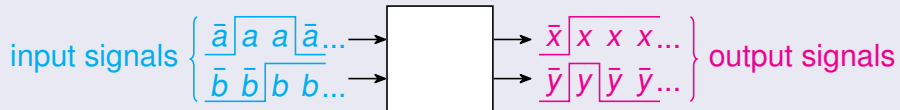
Reactive Synthesis in a Nutshell

A *reactive controller* produces output as a reaction to its input



Reactive Synthesis in a Nutshell

A *reactive controller* produces output as a reaction to its input



The reactive synthesis problems

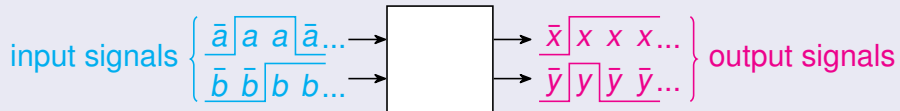
Given a specification relating **input signals** and **output signals** over time:

realizability: decide if a controller exist,

synthesis: construct it.

Reactive Synthesis in a Nutshell

A *reactive controller* produces output as a reaction to its input



The reactive synthesis problems

Given a specification relating **input signals** and **output signals** over time:

realizability: decide if a controller exist,

synthesis: construct it.

Our setup (from SYNTCOMP)

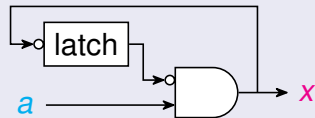
- ▶ the specification is an LTL formula, over ω -words such as “ $\bar{a}\bar{b}\bar{x}\bar{y}; a\bar{b}xy; abx\bar{y}; \dots$ ”
- ▶ the controller should be an *And-Inverter Graph*

Reactive Synthesis Example

1. LTL Spec.

$$a \leftrightarrow \mathbf{F}(x)$$

6. Output AIG



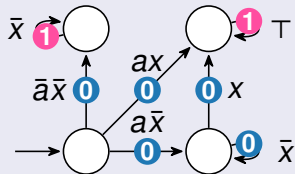
Reactive Synthesis Example

1. LTL Spec.

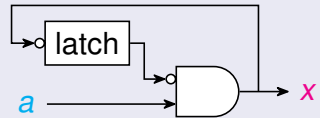
$$a \leftrightarrow \mathbf{F}(x)$$

2. DPA

$$\text{Fin}(\textcircled{0}) \vee \text{Inf}(\textcircled{1})$$



6. Output ALG

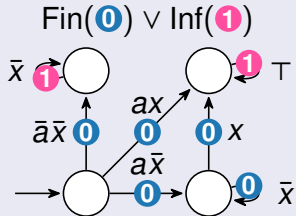


Reactive Synthesis Example

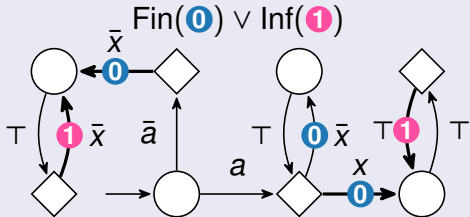
1. LTL Spec.

$$a \leftrightarrow \mathbf{F}(x)$$

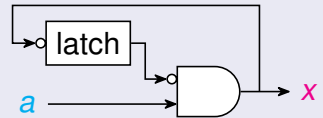
2. DPA



3. Parity Game



6. Output AIG

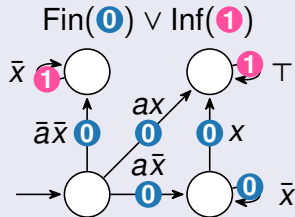


Reactive Synthesis Example

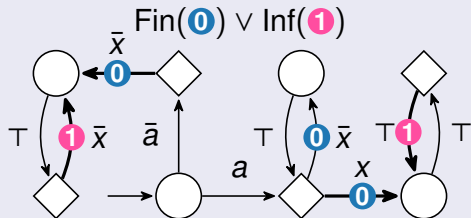
1. LTL Spec.

$$a \leftrightarrow \mathbf{F}(x)$$

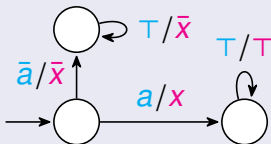
2. DPA



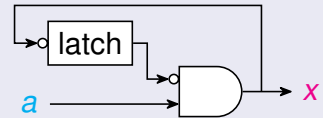
3. Parity Game



4. Winning Strat. as Mealy M.



6. Output ALG

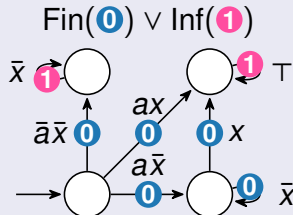


Reactive Synthesis Example

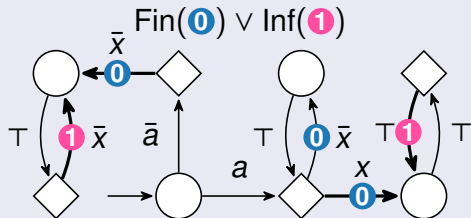
1. LTL Spec.

$$a \leftrightarrow \mathbf{F}(x)$$

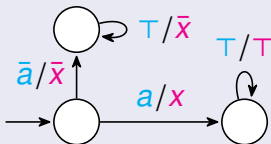
2. DPA



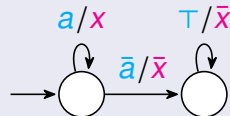
3. Parity Game



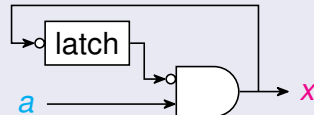
4. Winning Strat. as Mealy M.



5. Simpl. Mealy M.



6. Output ALG



Reactive Synthesis Example

1. LTL Spec.

$$a \leftrightarrow \mathbf{F}(x)$$

2. DPA

$\text{Fin}(\text{0}) \vee \text{Inf}(\text{1})$

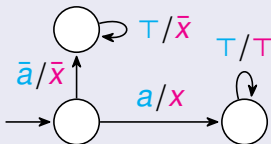
Determinism is required, so we cannot use Büchi acceptance

3. Parity Game

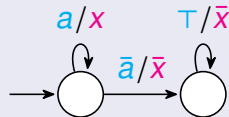
Parity games:

- ▶ have memory-less strategies
- ▶ can be solved in near polynomial time

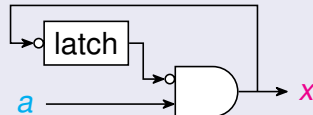
4. Winning Strat. as Mealy M.



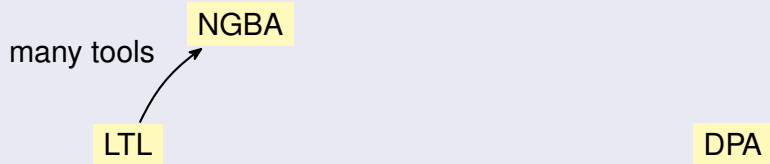
5. Simpl. Mealy M.



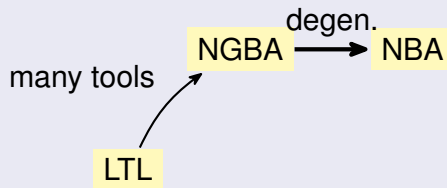
6. Output ALG





How to turn an LTL formula into a DPA?



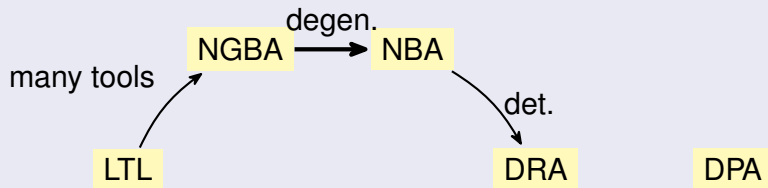
How to turn an LTL formula into a DPA?



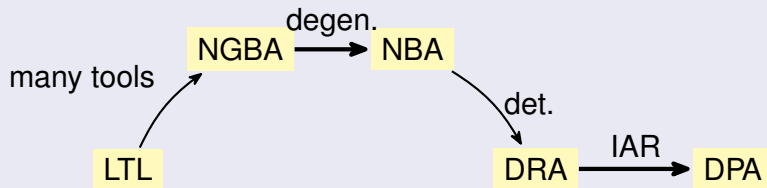
 Gastin and Oddoux. Fast LTL to Büchi automata translation. *CAV'01*. [doi](#)

 Babiak et al. Compositional approach to suspension and other improvements to LTL translation. *SPIN'13*. [doi](#)

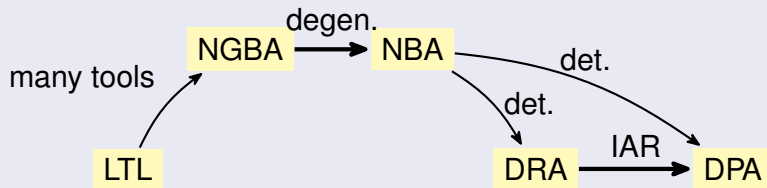
How to turn an LTL formula into a DPA?



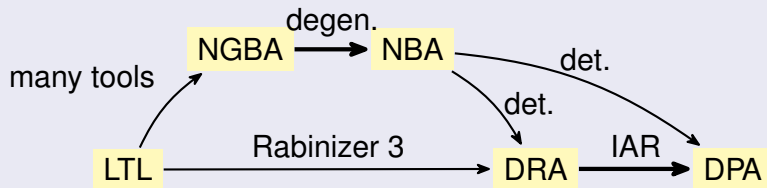
How to turn an LTL formula into a DPA?



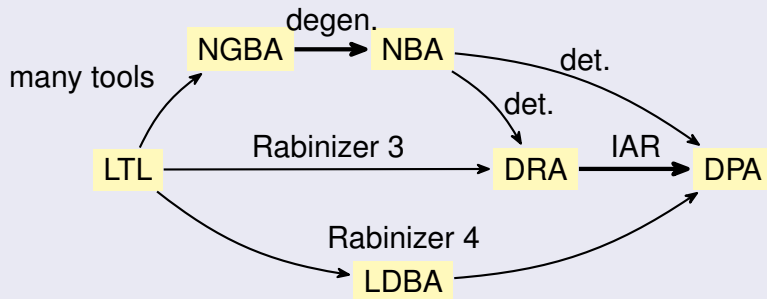
How to turn an LTL formula into a DPA?



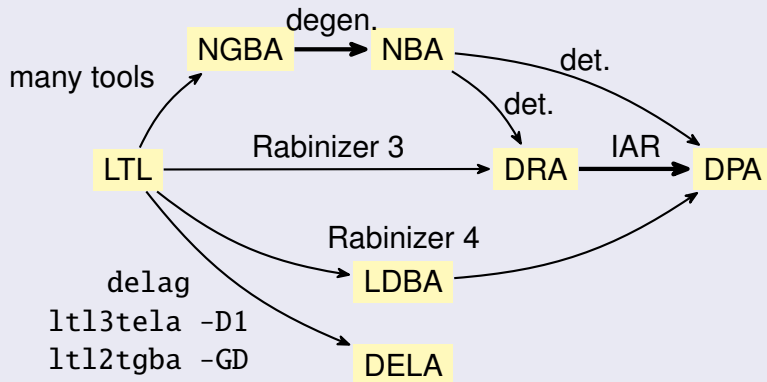
How to turn an LTL formula into a DPA?





How to turn an LTL formula into a DPA?



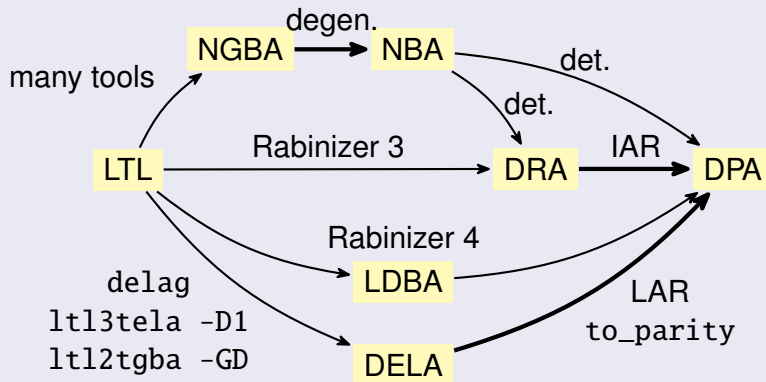
How to turn an LTL formula into a DPA?





 Major et al. ltl3tela: LTL to small deterministic or nondeterministic Emerson-Lei automata. *ATVA'19*. [doi](#)

 Müller and Sickert. LTL to deterministic Emerson-Lei automata. *GandALF'17*. [doi](#)

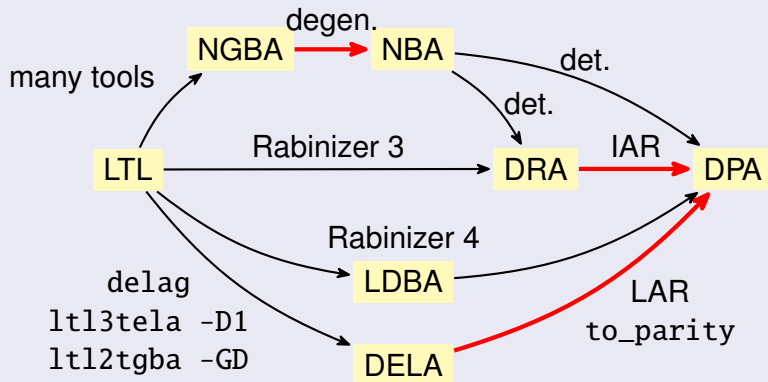
How to turn an LTL formula into a DPA?



 Löding. Optimal bounds for transformations of ω -automata. *FSTTCS'99*. [▶ doi](#)

 Renkin, Duret-Lutz, and Pommellet. Practical “paritizing” of Emerson-Lei automata. *ATVA'20*. [▶ doi](#)

How to turn an LTL formula into a DPA?



These **paritization** procedures are all improved by ACD!

Practical Applications of the Alternating Cycle Decomposition

Antonio Casares Alexandre Duret-Lutz
Klara J. Meyer Florian Renkin Salomon Sickert

TACAS'22



Alternating Cycle Decomposition

[ICALP'21]

Optimal Transformations of Games and Automata Using Muller Conditions

Antonio Casares 
LaBRI, Université de Bordeaux, France

Thomas Colcombet 
CNRS, IRIF, Université de Paris, France

Nathanaël Fijalkow 
CNRS, LaBRI, Université de Bordeaux, France
The Alan Turing Institute of Data Science, London, UK

Abstract

We consider the following question: given an automaton or a game with a Muller condition, how can we efficiently construct an equivalent one with a parity condition? There are several examples of such transformations in the literature, including in the determinisation of Büchi automata.

We define a new transformation called the alternating cycle decomposition, inspired and extending Zielonka's construction. Our transformation operates on transition systems, encompassing both automata and games, and preserves semantic properties through the existence of a locally bijective morphism. We show a strong optimality result: the obtained parity transition system is minimal both in number of states and number of priorities with respect to locally bijective morphisms.

We give two applications: the first is related to the determinisation of Büchi automata, and the second is to give crisp characterisations on the possibility of relabelling automata with different acceptance conditions.

2012 ACM Subject Classification Theory of computation → Automata over infinite objects

and phrases Automata over infinite words, Omega regular languages, Determinisation of

- ▶ Defines the ACD structure of Muller automata.
- ▶ How to use ACD to paritize an automaton. (With an optimality result.)
- ▶ How to use ACD to check automaton types.
- ▶ Purely theoretical (no implementation).

Alternating Cycle Decomposition

[ICALP'21]

Optimal Transformations of Games and Automata Using Muller Conditions

Antonio Casares  
LaBRI, Université de Bordeaux, France

Thomas Colcombet  
CNRS, IRIF, Université de Paris, France

Nathanaël Fijalkow  
CNRS, LaBRI, Université de Bordeaux, France
The Alan Turing Institute of Data Science, London, UK

Abstract

We consider the following question: given an automaton or a game with a Muller condition, how can we efficiently construct an equivalent one with a parity condition? There are several examples of such transformations in the literature, including in the determinisation of Büchi automata.

We define a new transformation called the alternating cycle decomposition, inspired and extending Zielonka's construction. Our transformation operates on transition systems, encompassing both automata and games, and preserves semantic properties through the existence of a locally bijective morphism. We show a strong optimality result: the obtained parity transition system is minimal both in number of states and number of priorities with respect to locally bijective morphisms.

We give two applications: the first is related to the determinisation of Büchi automata, and the second is to give crisp characterisations on the possibility of relabelling automata with different acceptance conditions.

2012 ACM Subject Classification Theory of computation → Automata over infinite objects

and phrases Automata over infinite words, Omega regular languages, Determinisation of

- ▶ Defines the ACD structure of Muller automata.
- ▶ How to use ACD to paritize an automaton. (With an optimality result.)
- ▶ How to use ACD to check automaton types.
- ▶ Purely theoretical (no implementation).

But is this ACD construction practical?

This Paper

- ▶ Adaptation of the definition of ACD and `acd_transform()` to TELA (Transition-based Emerson-Lei Automata, as supported by the HOA format.)
- ▶ Implementation in two tools:



Owl 21.0

`owl.model.in.tum.de`



Spot 2.10

`spot.lrde.epita.fr`

Motivation is LTL synthesis with: $\text{LTL} \rightarrow \text{DTELA} \rightarrow \text{DPA} \rightarrow \text{game} \rightarrow \text{controller}$

paritization

This Paper

- ▶ Adaptation of the definition of ACD and `acd_transform()` to TELA (Transition-based Emerson-Lei Automata, as supported by the HOA format.)
- ▶ Implementation in two tools:



Owl 21.0

`owl.model.in.tum.de`



Spot 2.10

`spot.lrde.epita.fr`

Motivation is LTL synthesis with: $\text{LTL} \rightarrow \text{DTELA} \rightarrow \text{DPA} \rightarrow \text{game} \rightarrow \text{controller}$

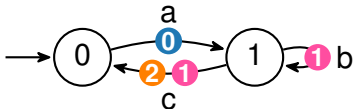
- ▶ Comparison to other existing paritization procedures
- ▶ State-based version of `acd_transform()`
- ▶ Comparison to degeneralization procedures (transition-based generalized Büchi \rightarrow state-based Büchi)

Emerson-Lei Automata (Using the HOA Syntax)

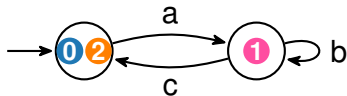
Acceptance condition = any positive Boolean combination of $\text{Inf}(n)$ or $\text{Fin}(n)$ terms.

Büchi	$\text{Inf}(\textcircled{0})$
generalized Büchi	$\text{Inf}(\textcircled{0}) \wedge \text{Inf}(\textcircled{1}) \wedge \text{Inf}(\textcircled{2}) \wedge \dots$
co-Büchi	$\text{Fin}(\textcircled{0})$
generalized co-Büchi	$\text{Fin}(\textcircled{0}) \vee \text{Fin}(\textcircled{1}) \vee \text{Fin}(\textcircled{2}) \vee \dots$
Rabin	$(\text{Fin}(\textcircled{0}) \wedge \text{Inf}(\textcircled{1})) \vee (\text{Fin}(\textcircled{2}) \wedge \text{Inf}(\textcircled{3})) \vee \dots$
Streett	$(\text{Inf}(\textcircled{0}) \vee \text{Fin}(\textcircled{1})) \wedge (\text{Inf}(\textcircled{2}) \vee \text{Fin}(\textcircled{3})) \wedge \dots$
parity min even	$\text{Inf}(\textcircled{0}) \vee (\text{Fin}(\textcircled{1}) \wedge (\text{Inf}(\textcircled{2}) \vee (\text{Fin}(\textcircled{3}) \wedge \dots)))$
parity min odd	$\text{Fin}(\textcircled{0}) \wedge (\text{Inf}(\textcircled{1}) \vee (\text{Fin}(\textcircled{2}) \wedge (\text{Inf}(\textcircled{3}) \vee \dots)))$

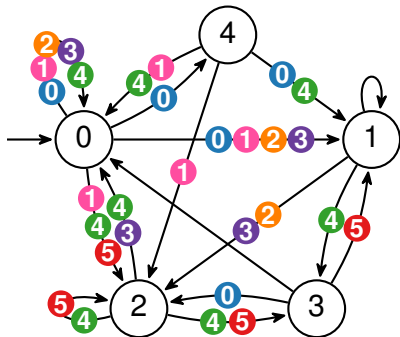
transition-based acceptance (TELA):



state-based acceptance:



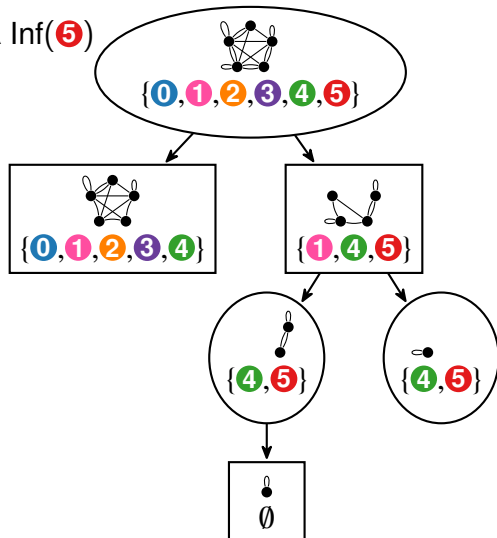
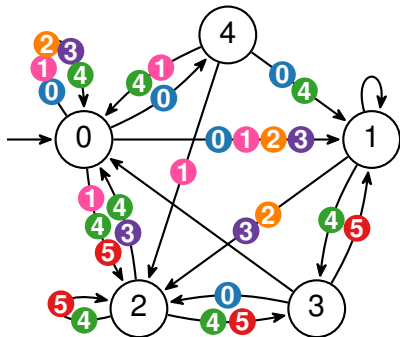
$$(\text{Inf}(\textcircled{0}) \vee \text{Fin}(\textcircled{1}) \vee \text{Inf}(\textcircled{2}) \vee \text{Inf}(\textcircled{3})) \wedge \text{Inf}(\textcircled{4}) \wedge \text{Inf}(\textcircled{5})$$



Automaton labels/alphabet hidden for simplicity.

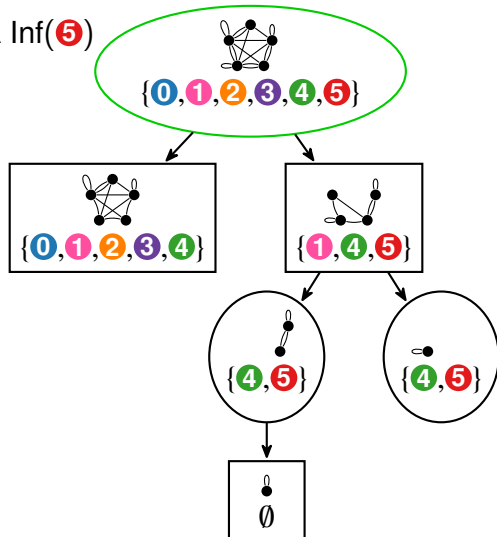
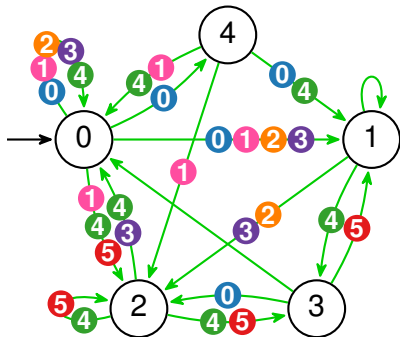
ACD for TELA

$$(\text{Inf}(\textcircled{0}) \vee \text{Fin}(\textcircled{1}) \vee \text{Inf}(\textcircled{2}) \vee \text{Inf}(\textcircled{3})) \wedge \text{Inf}(\textcircled{4}) \wedge \text{Inf}(\textcircled{5})$$



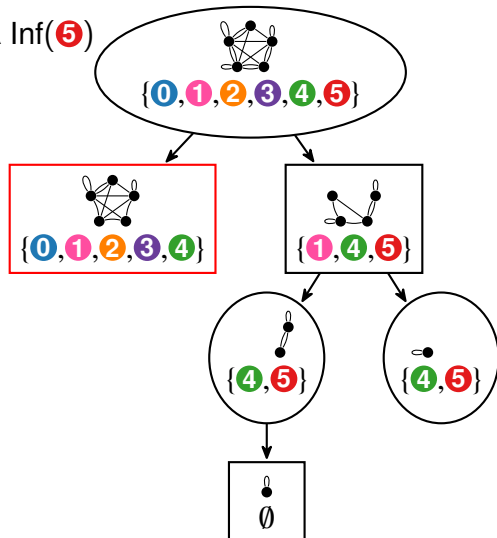
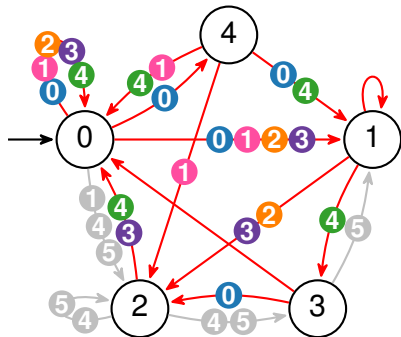
ACD for TELA

$$(\text{Inf}(\textcircled{0}) \vee \text{Fin}(\textcircled{1}) \vee \text{Inf}(\textcircled{2}) \vee \text{Inf}(\textcircled{3})) \wedge \text{Inf}(\textcircled{4}) \wedge \text{Inf}(\textcircled{5})$$



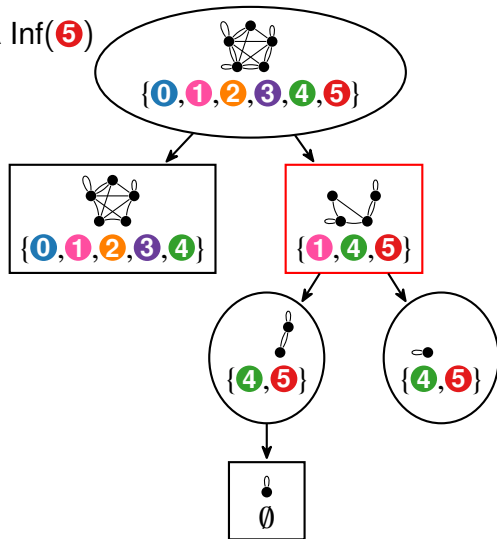
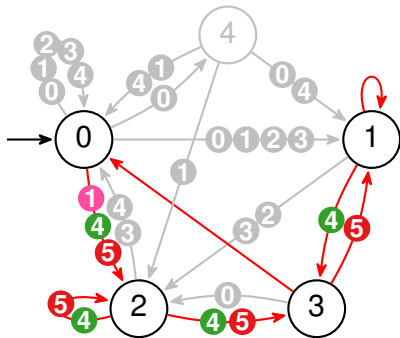
ACD for TELA

$$(\text{Inf}(\textcolor{blue}{0}) \vee \text{Fin}(\textcolor{pink}{1}) \vee \text{Inf}(\textcolor{orange}{2}) \vee \text{Inf}(\textcolor{purple}{3})) \wedge \text{Inf}(\textcolor{green}{4}) \wedge \text{Inf}(\textcolor{red}{5})$$



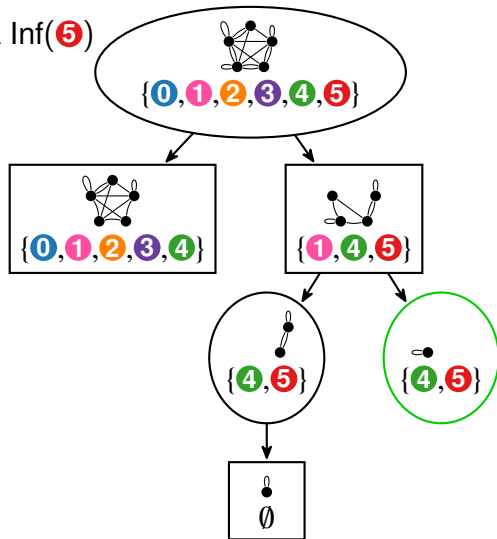
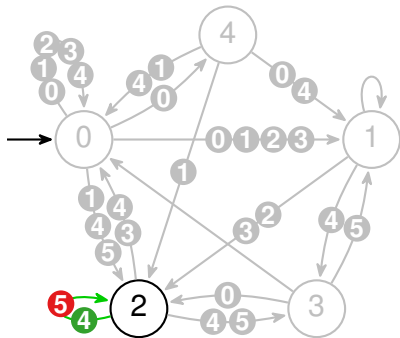
ACD for TELA

$$(\text{Inf}(\textcircled{0}) \vee \text{Fin}(\textcircled{1}) \vee \text{Inf}(\textcircled{2}) \vee \text{Inf}(\textcircled{3})) \wedge \text{Inf}(\textcircled{4}) \wedge \text{Inf}(\textcircled{5})$$



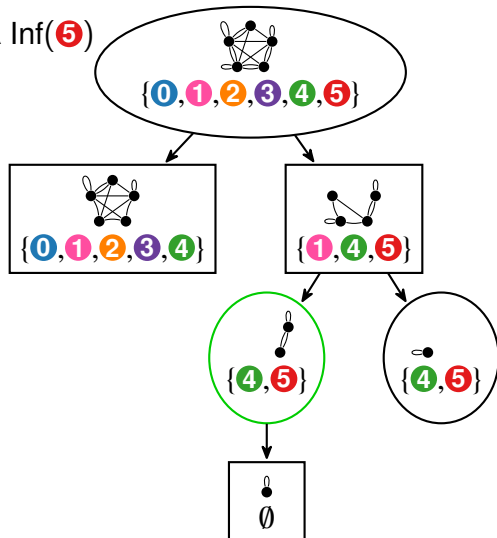
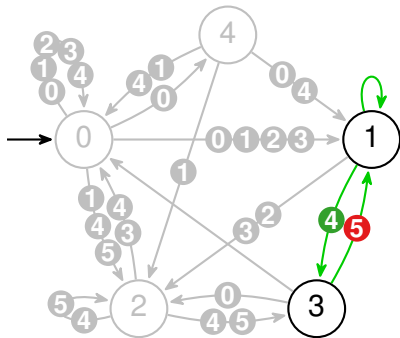
ACD for TELA

$$(\text{Inf}(\textcircled{0}) \vee \text{Fin}(\textcircled{1}) \vee \text{Inf}(\textcircled{2}) \vee \text{Inf}(\textcircled{3})) \wedge \text{Inf}(\textcircled{4}) \wedge \text{Inf}(\textcircled{5})$$



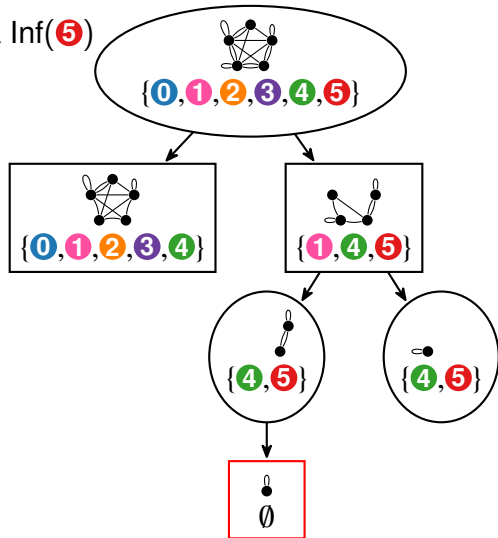
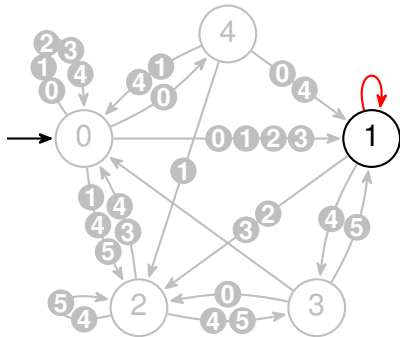
ACD for TELA

$$(\text{Inf}(\textcircled{0}) \vee \text{Fin}(\textcircled{1}) \vee \text{Inf}(\textcircled{2}) \vee \text{Inf}(\textcircled{3})) \wedge \text{Inf}(\textcircled{4}) \wedge \text{Inf}(\textcircled{5})$$



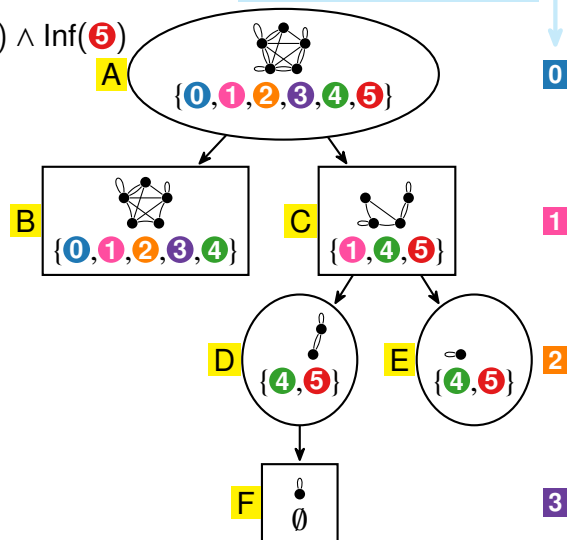
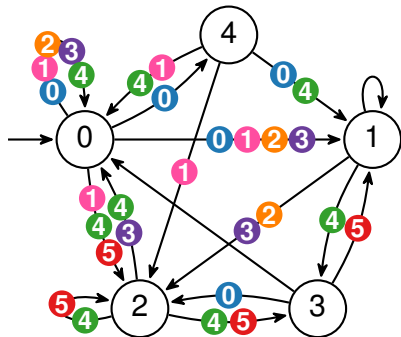
ACD for TELA

$$(\text{Inf}(\textcircled{0}) \vee \text{Fin}(\textcircled{1}) \vee \text{Inf}(\textcircled{2}) \vee \text{Inf}(\textcircled{3})) \wedge \text{Inf}(\textcircled{4}) \wedge \text{Inf}(\textcircled{5})$$



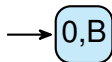
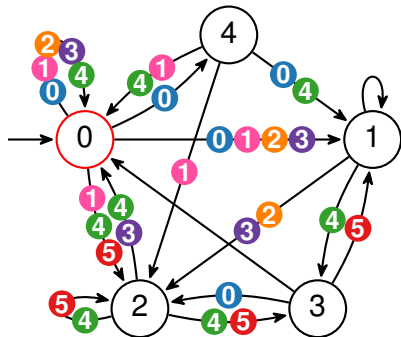
Paritization of a TELA

$$(\text{Inf}(\textcircled{0}) \vee \text{Fin}(\textcircled{1}) \vee \text{Inf}(\textcircled{2}) \vee \text{Inf}(\textcircled{3})) \wedge \text{Inf}(\textcircled{4}) \wedge \text{Inf}(\textcircled{5})$$

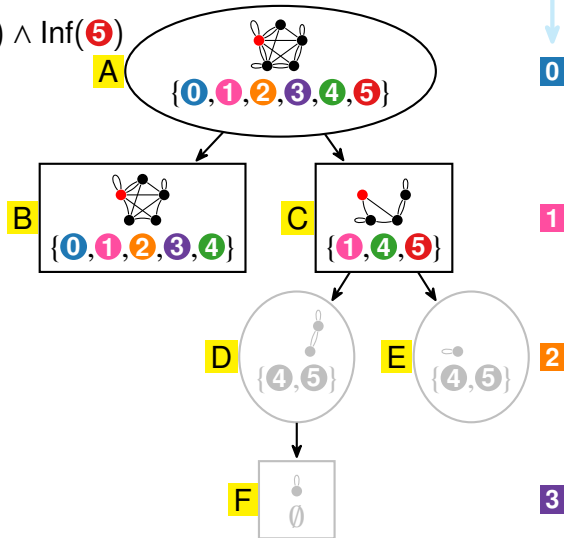


Paritization of a TELA

$$(\text{Inf}(\textcircled{0}) \vee \text{Fin}(\textcircled{1}) \vee \text{Inf}(\textcircled{2}) \vee \text{Inf}(\textcircled{3})) \wedge \text{Inf}(\textcircled{4}) \wedge \text{Inf}(\textcircled{5})$$

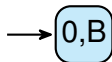
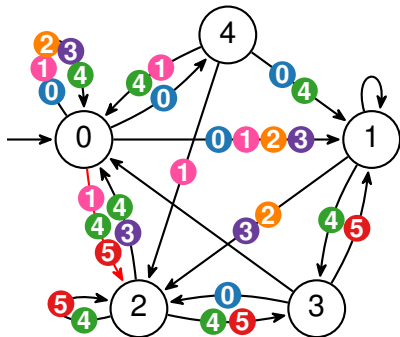


priorities for
min even acceptance

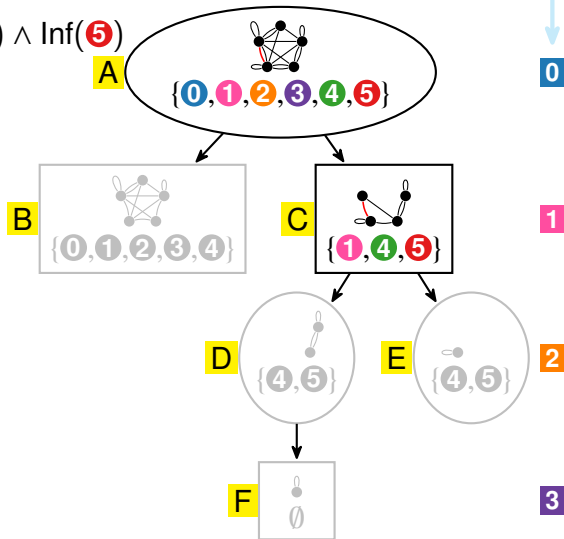


Paritization of a TELA

$$(\text{Inf}(\textcircled{0}) \vee \text{Fin}(\textcircled{1}) \vee \text{Inf}(\textcircled{2}) \vee \text{Inf}(\textcircled{3})) \wedge \text{Inf}(\textcircled{4}) \wedge \text{Inf}(\textcircled{5})$$

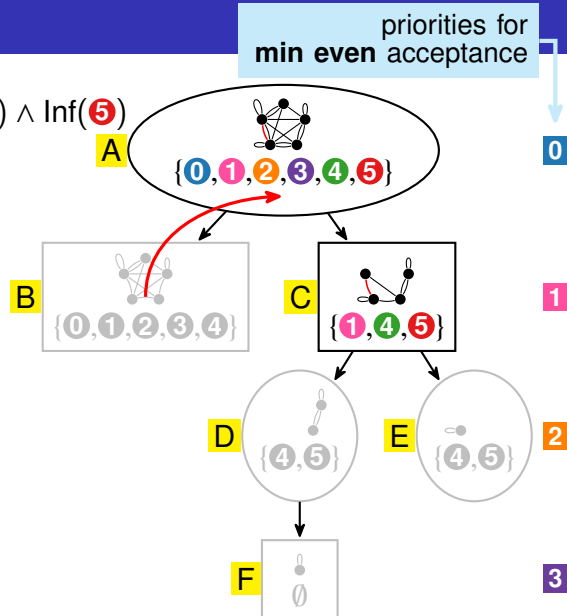
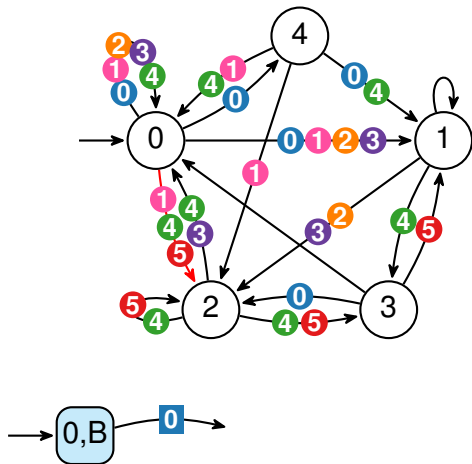


priorities for
min even acceptance



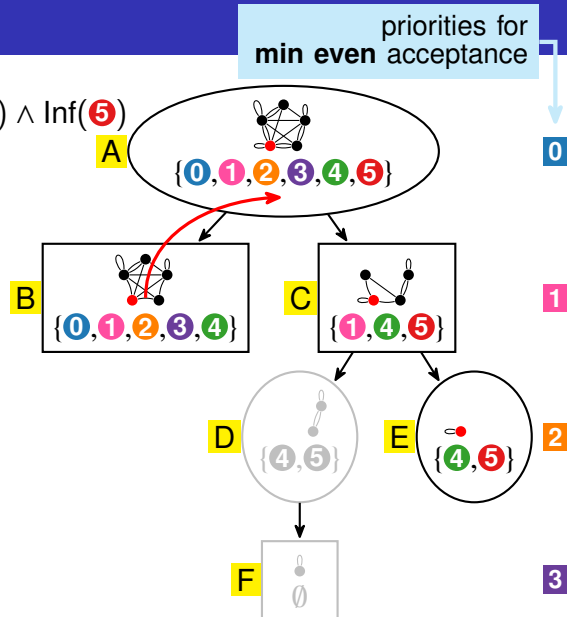
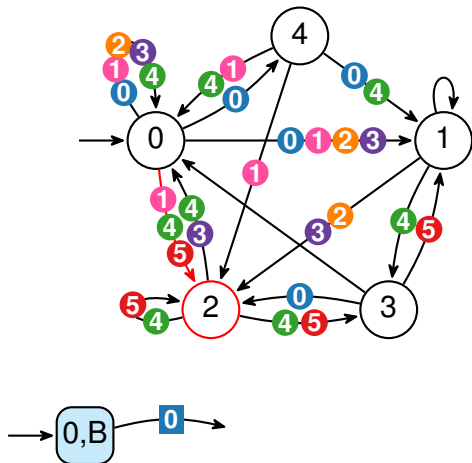
Paritization of a TELA

$$(\text{Inf}(\textcircled{0}) \vee \text{Fin}(\textcircled{1}) \vee \text{Inf}(\textcircled{2}) \vee \text{Inf}(\textcircled{3})) \wedge \text{Inf}(\textcircled{4}) \wedge \text{Inf}(\textcircled{5})$$



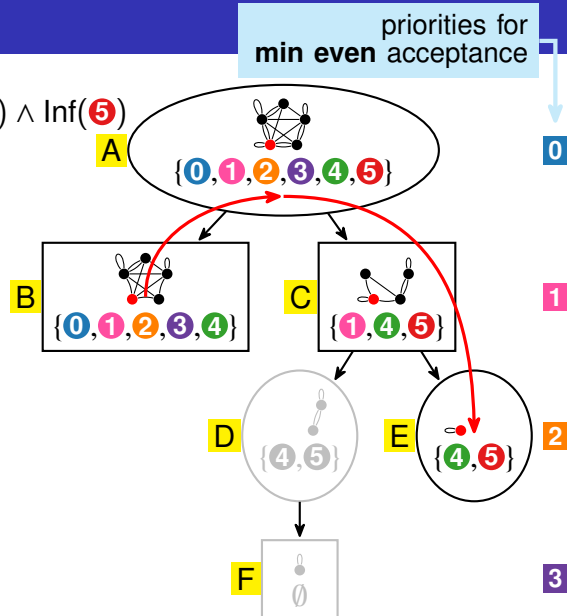
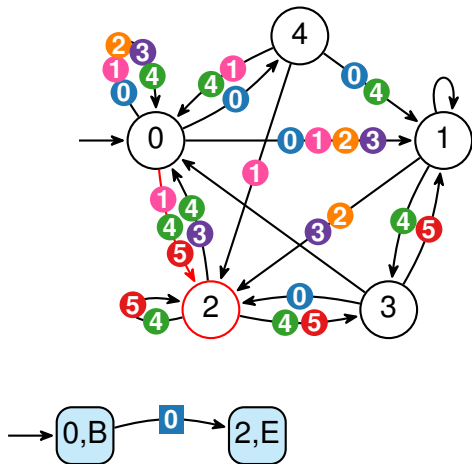
Paritization of a TELA

$$(\text{Inf}(\textcircled{0}) \vee \text{Fin}(\textcircled{1}) \vee \text{Inf}(\textcircled{2}) \vee \text{Inf}(\textcircled{3})) \wedge \text{Inf}(\textcircled{4}) \wedge \text{Inf}(\textcircled{5})$$



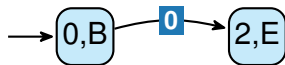
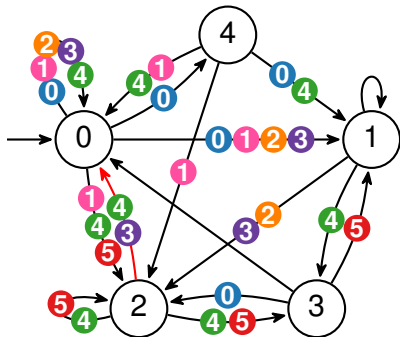
Paritization of a TELA

$$(\text{Inf}(\textcircled{0}) \vee \text{Fin}(\textcircled{1}) \vee \text{Inf}(\textcircled{2}) \vee \text{Inf}(\textcircled{3})) \wedge \text{Inf}(\textcircled{4}) \wedge \text{Inf}(\textcircled{5})$$

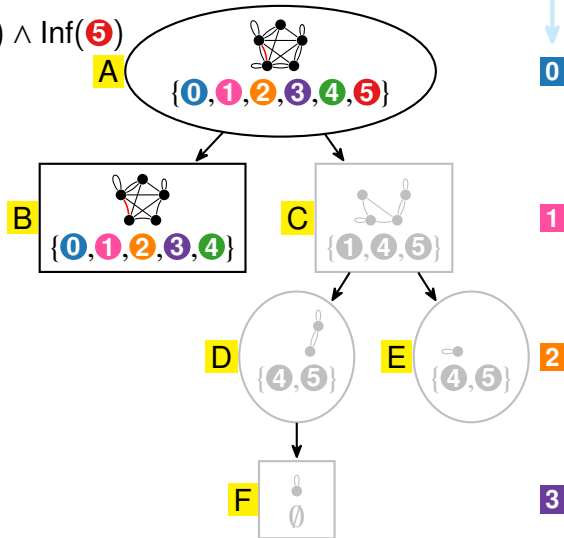


Paritization of a TELA

$$(\text{Inf}(\textcircled{0}) \vee \text{Fin}(\textcircled{1}) \vee \text{Inf}(\textcircled{2}) \vee \text{Inf}(\textcircled{3})) \wedge \text{Inf}(\textcircled{4}) \wedge \text{Inf}(\textcircled{5})$$

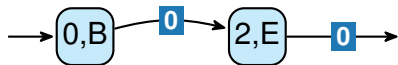
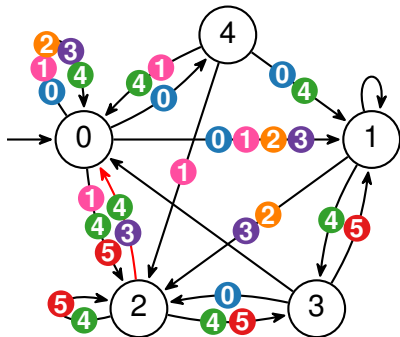


priorities for
min even acceptance

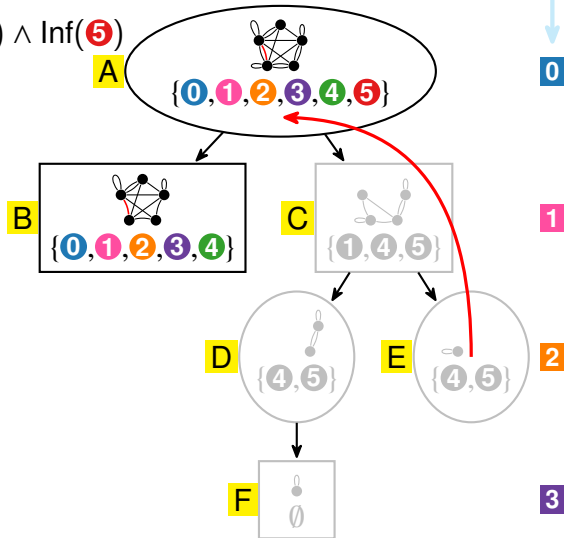


Paritization of a TELA

$$(\text{Inf}(\textcircled{0}) \vee \text{Fin}(\textcircled{1}) \vee \text{Inf}(\textcircled{2}) \vee \text{Inf}(\textcircled{3})) \wedge \text{Inf}(\textcircled{4}) \wedge \text{Inf}(\textcircled{5})$$

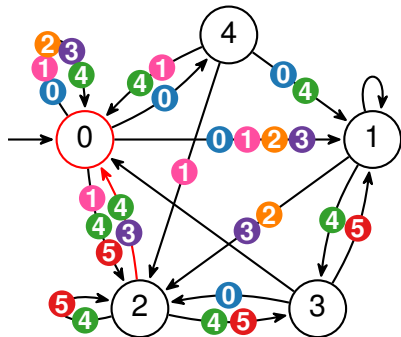


priorities for
min even acceptance

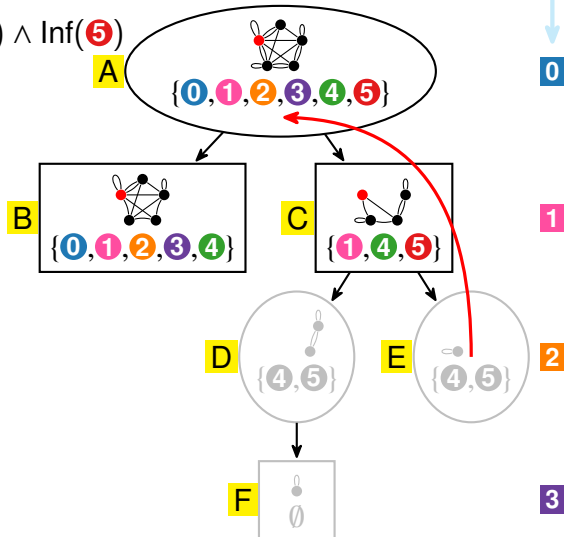


Paritization of a TELA

$$(\text{Inf}(\textcircled{0}) \vee \text{Fin}(\textcircled{1}) \vee \text{Inf}(\textcircled{2}) \vee \text{Inf}(\textcircled{3})) \wedge \text{Inf}(\textcircled{4}) \wedge \text{Inf}(\textcircled{5})$$

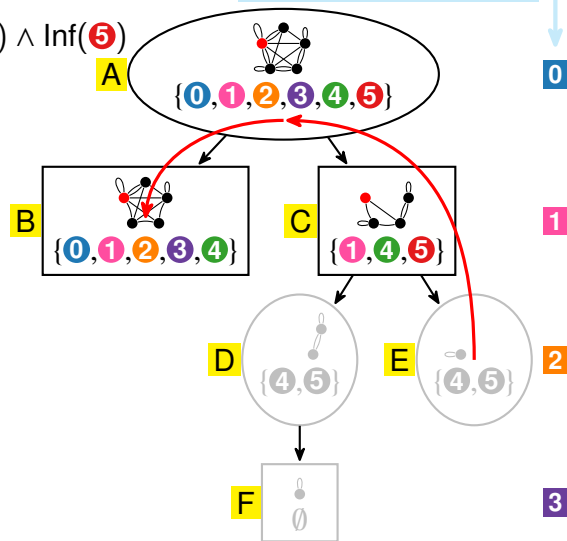
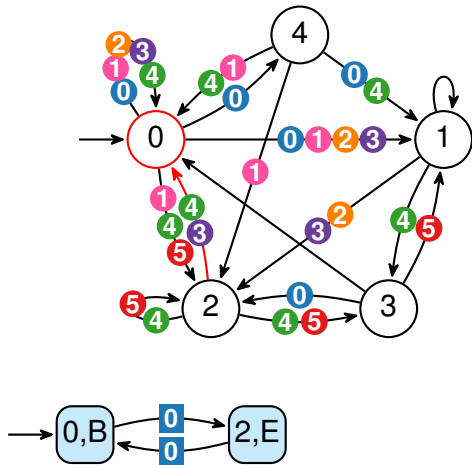


priorities for
min even acceptance



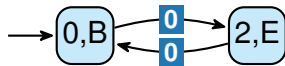
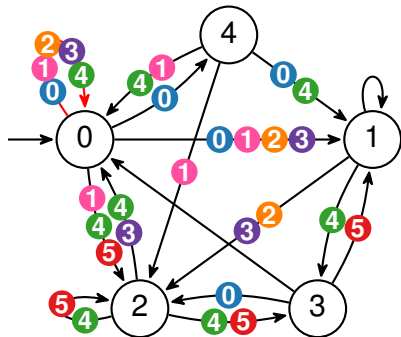
Paritization of a TELA

$$(\text{Inf}(\textcircled{0}) \vee \text{Fin}(\textcircled{1}) \vee \text{Inf}(\textcircled{2}) \vee \text{Inf}(\textcircled{3})) \wedge \text{Inf}(\textcircled{4}) \wedge \text{Inf}(\textcircled{5})$$

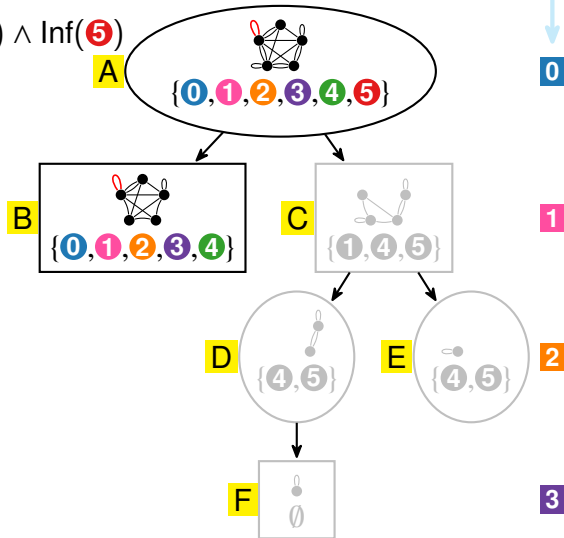


Paritization of a TELA

$$(\text{Inf}(\textcircled{0}) \vee \text{Fin}(\textcircled{1}) \vee \text{Inf}(\textcircled{2}) \vee \text{Inf}(\textcircled{3})) \wedge \text{Inf}(\textcircled{4}) \wedge \text{Inf}(\textcircled{5})$$



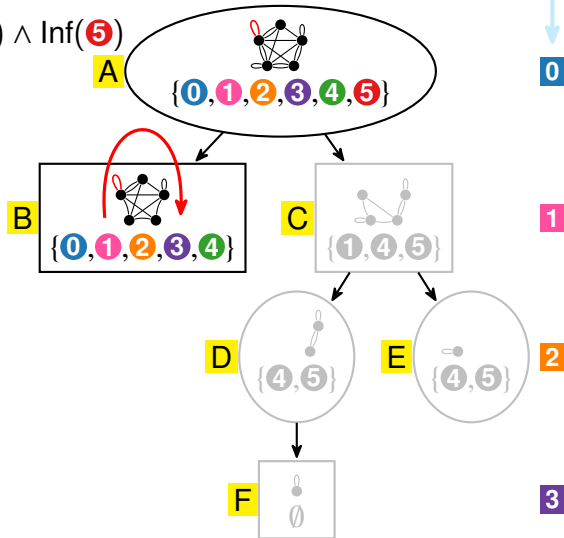
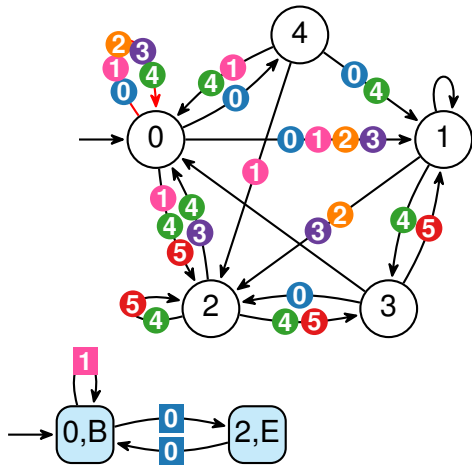
priorities for
min even acceptance



Paritization of a TELA

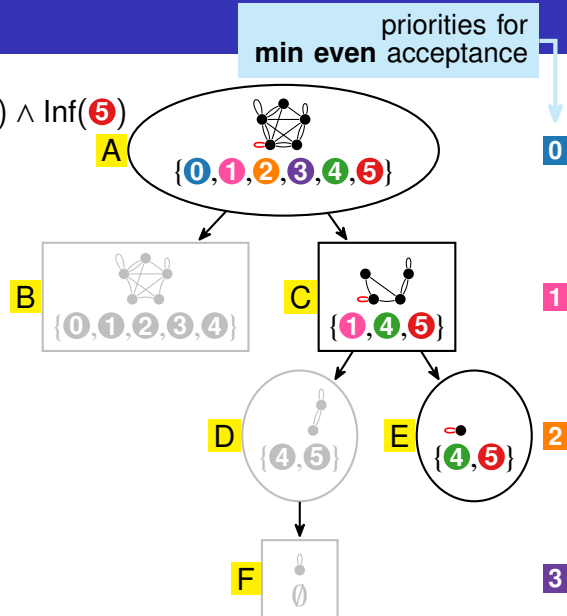
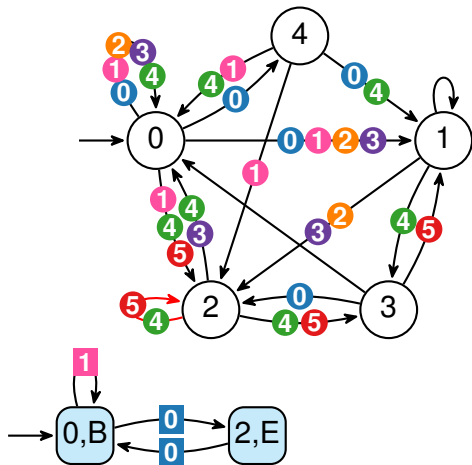
priorities for
min even acceptance

$$(\text{Inf}(\textcircled{0}) \vee \text{Fin}(\textcircled{1}) \vee \text{Inf}(\textcircled{2}) \vee \text{Inf}(\textcircled{3})) \wedge \text{Inf}(\textcircled{4}) \wedge \text{Inf}(\textcircled{5})$$



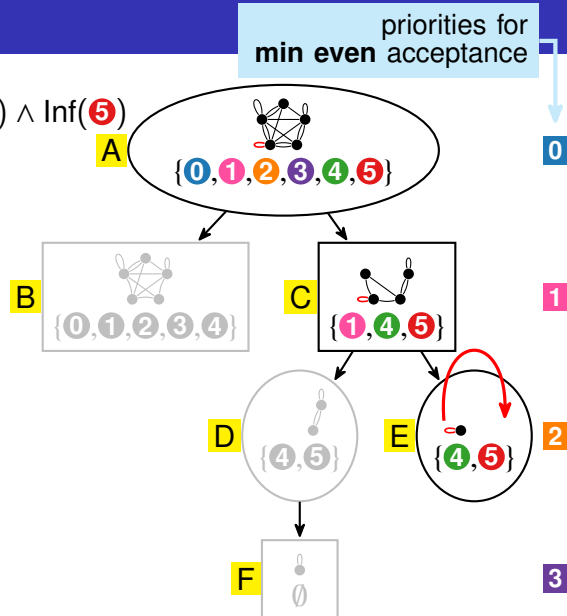
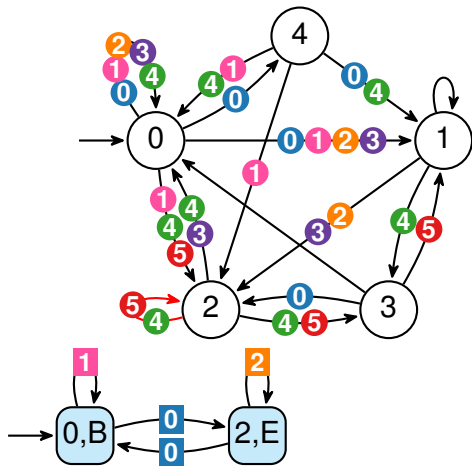
Paritization of a TELA

$$(\text{Inf}(\textcircled{0}) \vee \text{Fin}(\textcircled{1}) \vee \text{Inf}(\textcircled{2}) \vee \text{Inf}(\textcircled{3})) \wedge \text{Inf}(\textcircled{4}) \wedge \text{Inf}(\textcircled{5})$$



Paritization of a TELA

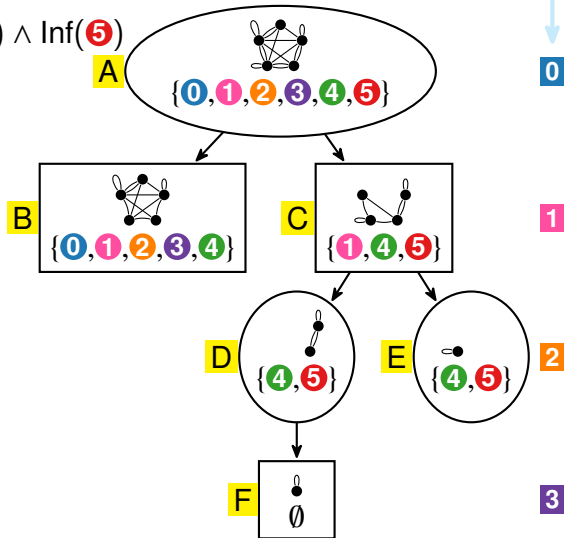
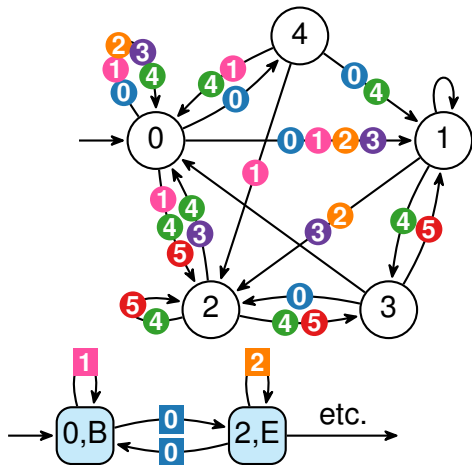
$$(\text{Inf}(\textcircled{0}) \vee \text{Fin}(\textcircled{1}) \vee \text{Inf}(\textcircled{2}) \vee \text{Inf}(\textcircled{3})) \wedge \text{Inf}(\textcircled{4}) \wedge \text{Inf}(\textcircled{5})$$



Paritization of a TELA

priorities for
min even acceptance

$$(\text{Inf}(\textcircled{0}) \vee \text{Fin}(\textcircled{1}) \vee \text{Inf}(\textcircled{2}) \vee \text{Inf}(\textcircled{3})) \wedge \text{Inf}(\textcircled{4}) \wedge \text{Inf}(\textcircled{5})$$



Other Paritization Procedures

Many paritization procedures create states (s, m) where $\begin{cases} s: \text{original state} \\ m: \text{memory} \end{cases}$

Latest Appearance Record (LAR) works on any TELA,
 m is an order of all colors

Other Paritization Procedures

Many paritization procedures create states (s, m) where $\begin{cases} s: \text{original state} \\ m: \text{memory} \end{cases}$

Latest Appearance Record (LAR) works on any TELA,
 m is an order of all colors

Index Appearance Record (IAR) take Rabin or Streett as input,
 m is an order of the acceptance pairs

Other Paritization Procedures

Many paritization procedures create states (s, m) where $\begin{cases} s: \text{original state} \\ m: \text{memory} \end{cases}$

Latest Appearance Record (LAR) works on any TELA,

m is an order of all colors

Index Appearance Record (IAR) take Rabin or Streett as input,

m is an order of the acceptance pairs

Degeneralization takes generalized Büchi as input,

m is a color number

Other Paritization Procedures

Many paritization procedures create states (s, m) where $\begin{cases} s: \text{original state} \\ m: \text{memory} \end{cases}$

Latest Appearance Record (LAR) works on any TELA,
 m is an order of all colors

Index Appearance Record (IAR) take Rabin or Streett as input,
 m is an order of the acceptance pairs

Degeneralization takes generalized Büchi as input,
 m is a color number

Spot's `to_parity()` works on any TELA,
combines all the above + optimizations

Other Paritization Procedures

Many paritization procedures create states (s, m) where $\begin{cases} s: \text{original state} \\ m: \text{memory} \end{cases}$

Latest Appearance Record (LAR) works on any TELA,
 m is an order of all colors

Index Appearance Record (IAR) take Rabin or Streett as input,
 m is an order of the acceptance pairs

Degeneralization takes generalized Büchi as input,
 m is a color number

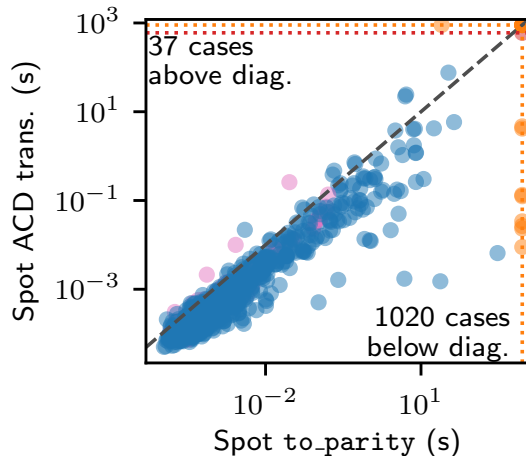
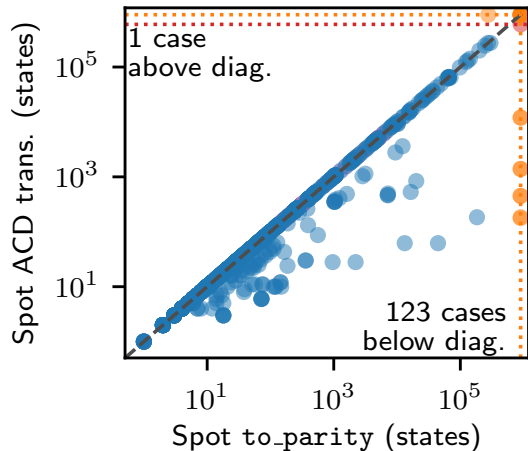
Spot's to_parity() works on any TELA,
combines all the above + optimizations

ACD-transform works on any TELA,
 m denotes a node of the ACD

ACD-transform is optimal among algorithms that build states of shape (s, m) .

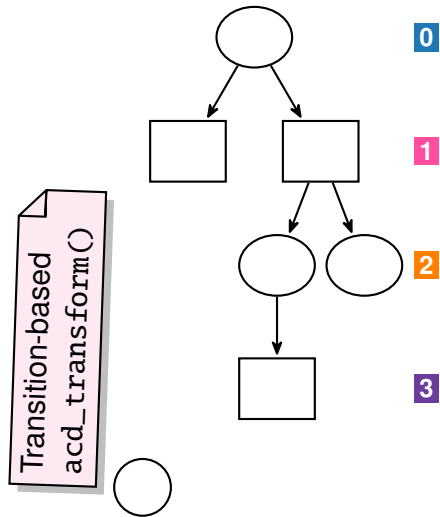


Comparison between `to_parity()` and `acd_transform()`

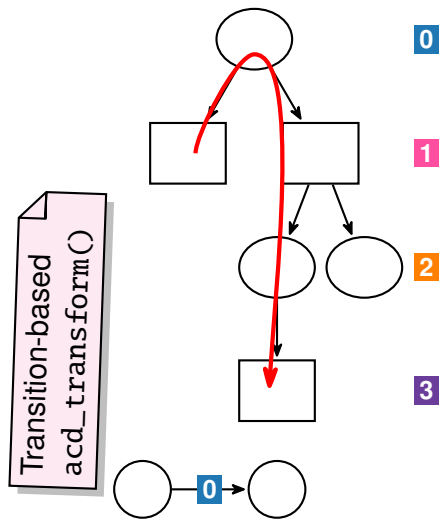


Benchmark of 1065 TELA generated from LTL formulas from SyntComp. These TELA have between 2 and 55 colors (median 5) and up to 245761 states (median 20).

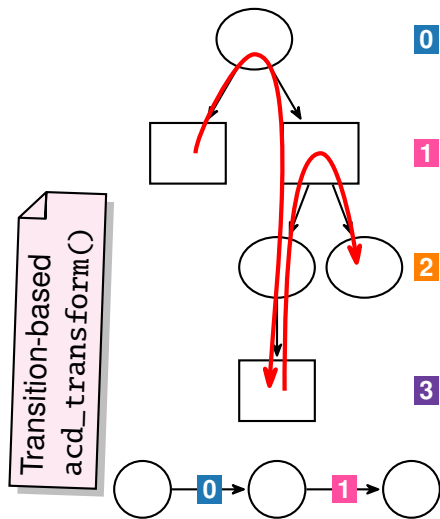
Producing State-Based Parity Automata (Intuition)



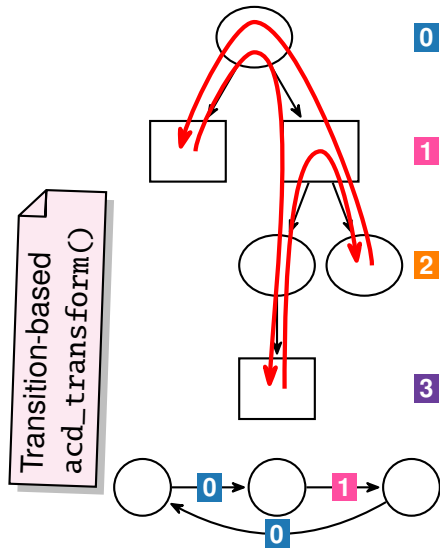
Producing State-Based Parity Automata (Intuition)



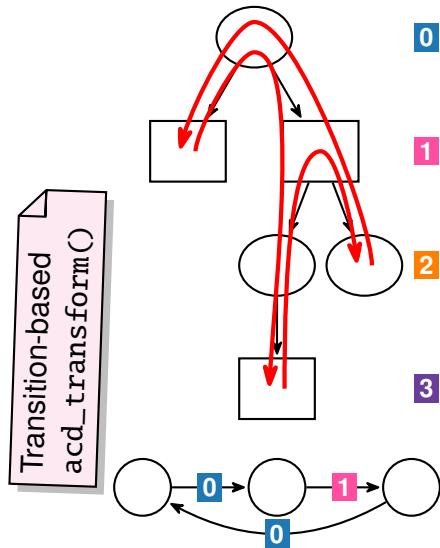
Producing State-Based Parity Automata (Intuition)



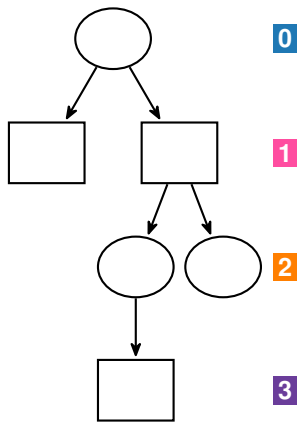
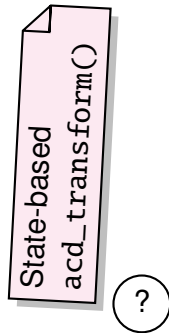
Producing State-Based Parity Automata (Intuition)



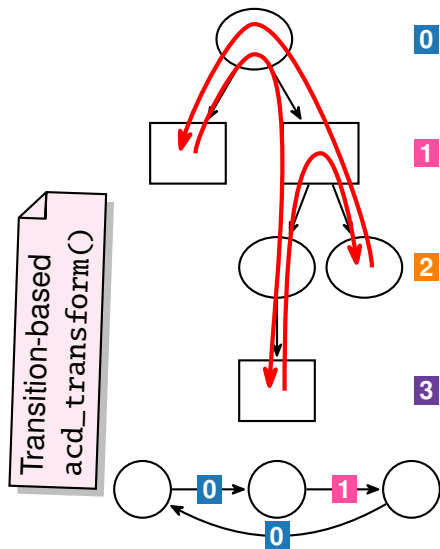
Producing State-Based Parity Automata (Intuition)



Every cycle has at least one leftward jump in the ACD.

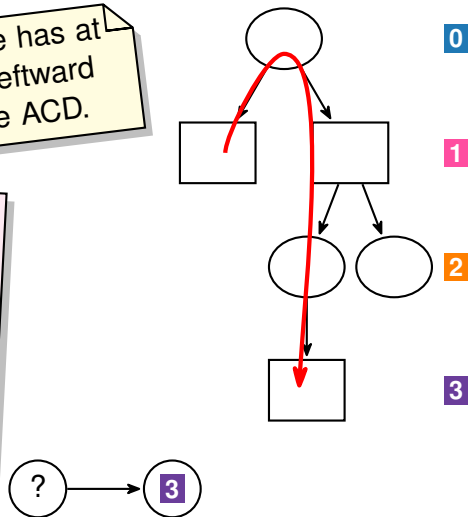


Producing State-Based Parity Automata (Intuition)

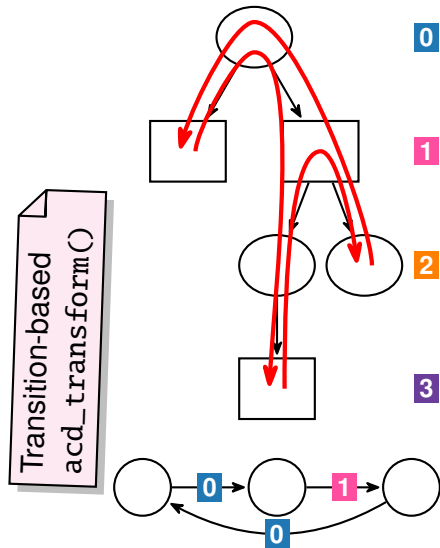


Every cycle has at least one leftward jump in the ACD.

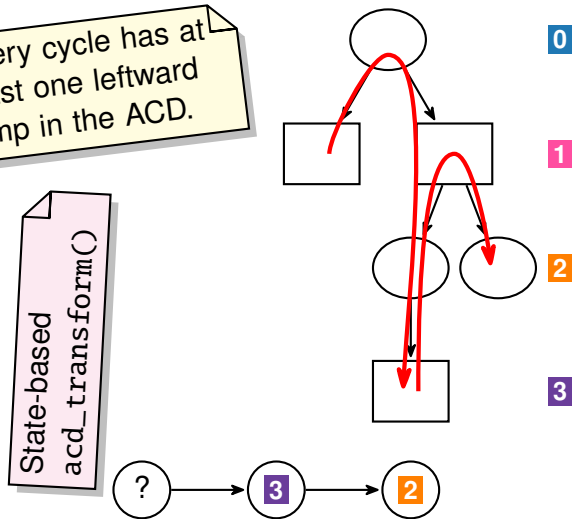
State-based
acd_transform()



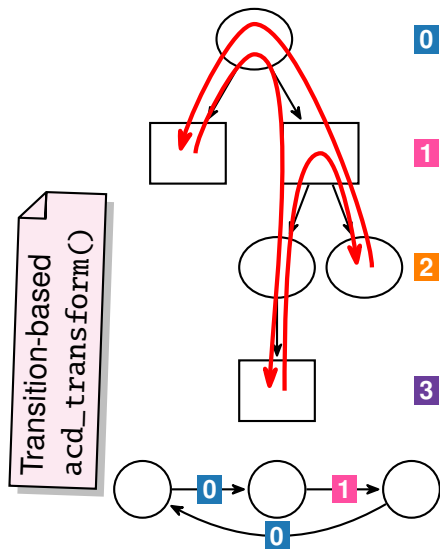
Producing State-Based Parity Automata (Intuition)



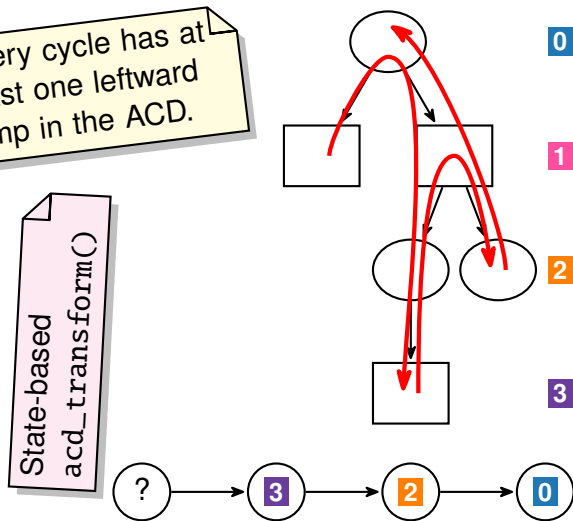
Every cycle has at least one leftward jump in the ACD.



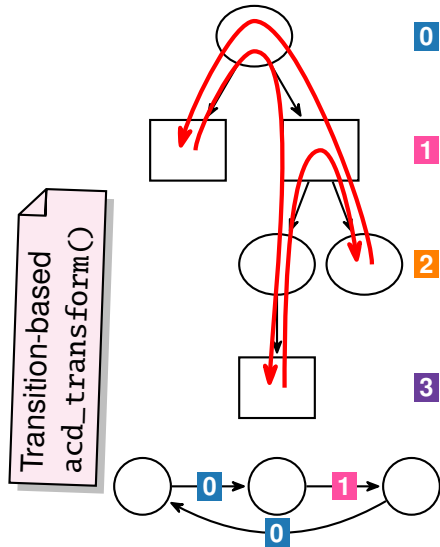
Producing State-Based Parity Automata (Intuition)



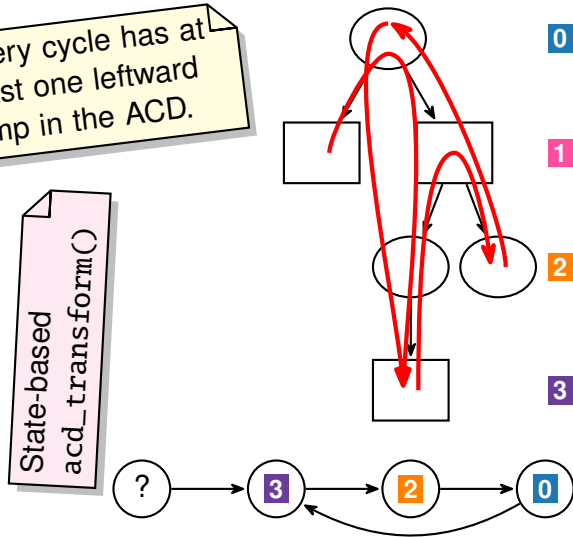
Every cycle has at least one leftward jump in the ACD.



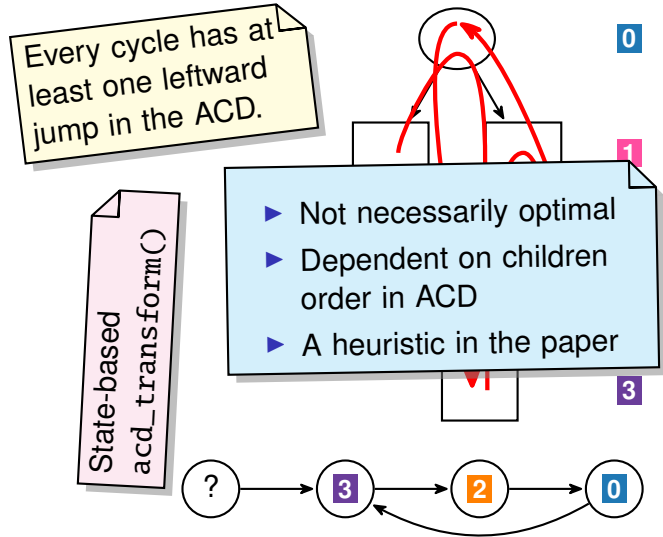
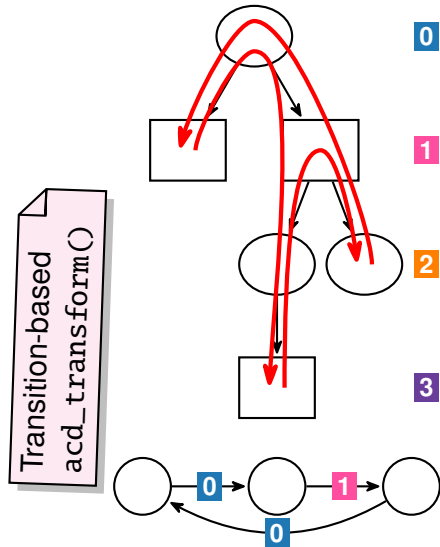
Producing State-Based Parity Automata (Intuition)



Every cycle has at least one leftward jump in the ACD.

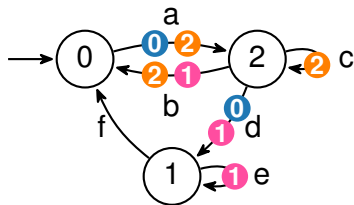


Producing State-Based Parity Automata (Intuition)



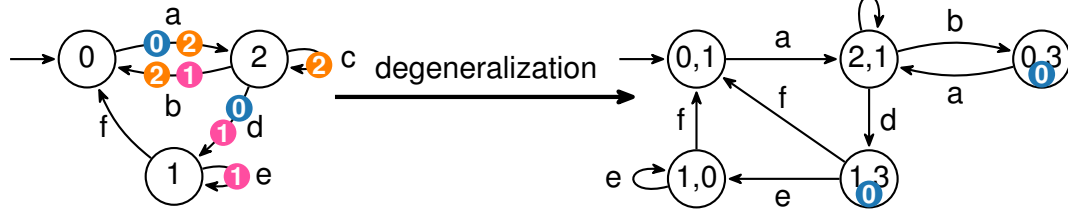
Application to Degeneralization (TGBA \rightarrow SBA)

$$\text{Inf}(\textcircled{0}) \wedge \text{Inf}(\textcircled{1}) \wedge \text{Inf}(\textcircled{2})$$



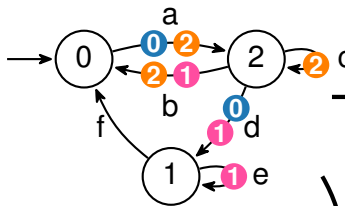
Application to Degeneralization (TGBA \rightarrow SBA)

$$\text{Inf}(\textcircled{0}) \wedge \text{Inf}(\textcircled{1}) \wedge \text{Inf}(\textcircled{2})$$

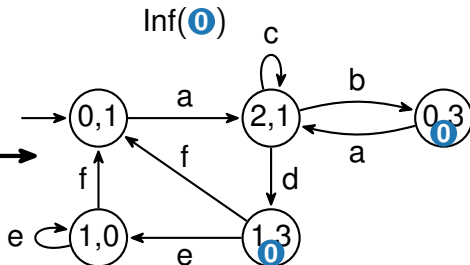


Application to Degeneralization (TGBA \rightarrow SBA)

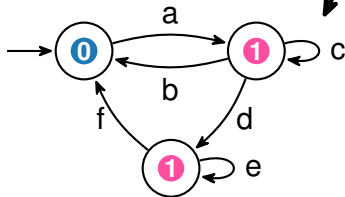
$$\text{Inf}(\textcircled{0}) \wedge \text{Inf}(\textcircled{1}) \wedge \text{Inf}(\textcircled{2})$$



degeneralization



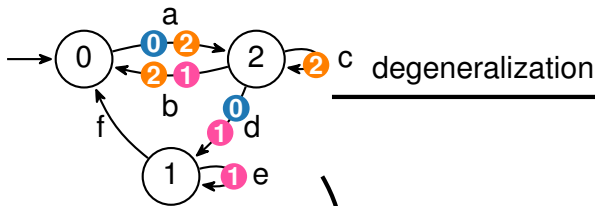
$$\text{Inf}(\textcircled{0}) \vee \text{Fin}(\textcircled{1})$$



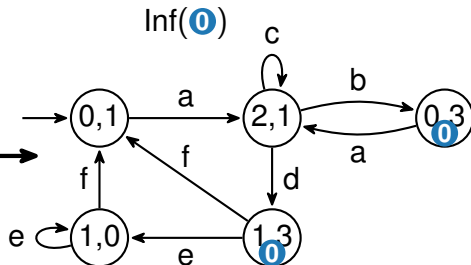
state-based
ACD transform

Application to Degeneralization (TGBA \rightarrow SBA)

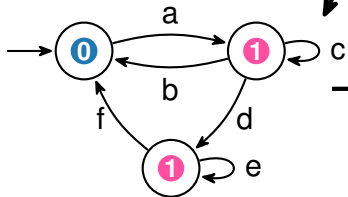
$\text{Inf}(\textcircled{0}) \wedge \text{Inf}(\textcircled{1}) \wedge \text{Inf}(\textcircled{2})$



degeneralization

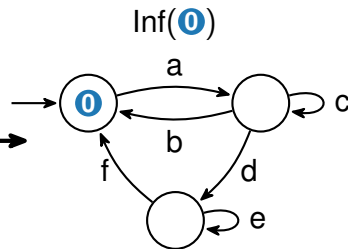


$\text{Inf}(\textcircled{0}) \vee \text{Fin}(\textcircled{1})$



state-based
ACD transform

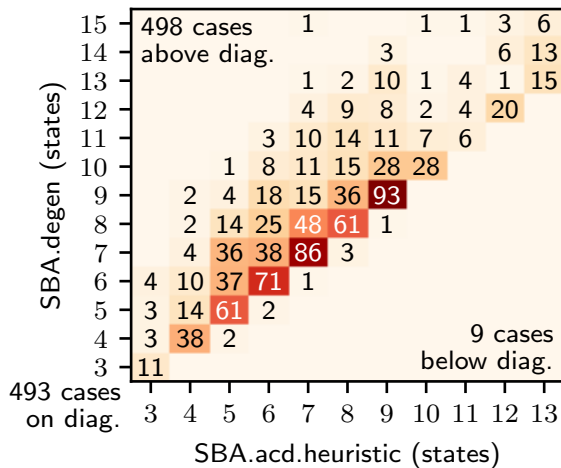
simplifies to



Application to Degeneralization: Benchmark

Application of the state-based version of `acd_transform()` on 1000 random transition-based generalized Büchi automata, with 3–4 states and 2–3 colors.

Comparison with Spot 2.10's best degeneralization routine.

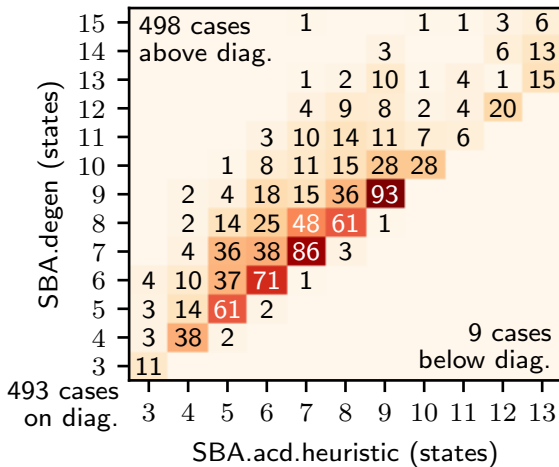


Application to Degeneralization: Benchmark

Application of the state-based version of `acd_transform()` on 1000 random transition-based generalized Büchi automata, with 3–4 states and 2–3 colors.

Comparison with Spot 2.10's best degeneralization routine.

Compared to min. sizes (found via SAT), ACD gives **+0.17** states on avg., and Spot's degen. gives **+1.21** states on avg.



Conclusion

ACD is versatile, and can replace several existing algorithms:

- ▶ paritization of automata with arbitrary acceptance (always better than LAR, IAR, and their combinations)
- ▶ degeneralization (better than traditional algorithms)
- ▶ minimization of the number of priorities in a parity automaton
- ▶ several typeness checks (looking at the shape of the ACD)



Two implementations can be a basis for further experiments:



Owl 21.0

`owl.model.in.tum.de`



Spot 2.10

[▶ demo](#)

`spot.lrde.epita.fr`