Complementation of Emerson-Lei Automata

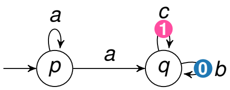
Vojtěch Havlena[1] Ondřej Lengál[1] Barbora Šmahlíková[1]

[1] Faculty of Information Technology, Brno University of Technology, Czech Republic

FoSSaCS'25

Transition-based Emerson-Lei Automata

- automata over infinite words

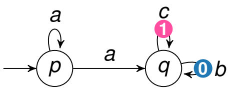


- Fin(0) $\wedge$ Inf(1)
- $aa(bc)^\omega \notin \mathcal{L}(\mathcal{A})$
- $aabbb(c)^\omega \in \mathcal{L}(\mathcal{A})$

- Emerson-Lei acceptance condition
  - $\Gamma = \{$0, 1 $\ldots,$ k-1$\}$
  - $\mathbb{EL}(\Gamma)$ are formulae according to the grammar $\alpha ::= tt \mid ff \mid \text{Inf}(c) \mid \text{Fin}(c) \mid (\alpha \wedge \alpha) \mid (\alpha \vee \alpha)$

Transition-based Emerson-Lei Automata

- automata over infinite words



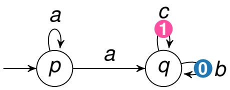
- $\mathrm{Fin}(0) \wedge \mathrm{Inf}(1)$
- $aa(bc)^\omega \notin \mathcal{L}(\mathcal{A})$
- $aabbb(c)^\omega \in \mathcal{L}(\mathcal{A})$

- Emerson-Lei acceptance condition
  - $\Gamma = \{0, 1, \ldots, k\text{-}1\}$
  - $\mathbb{EL}(\Gamma)$ are formulae according to the grammar $\alpha ::= tt \mid ff \mid \mathrm{Inf}(c) \mid \mathrm{Fin}(c) \mid (\alpha \wedge \alpha) \mid (\alpha \vee \alpha)$

- $\mathcal{A} = (Q, \delta, I, \mathrm{p}, \mathrm{Acc})$ over colors Γ
  - Q finite set of states
  - $\delta \subseteq Q \times \Sigma \times Q$ transition relation
  - I initial states
  - $\mathrm{p} \colon \delta \to 2^\Gamma$ colouring of transitions and
  - $\mathrm{Acc} \in \mathbb{EL}(\Gamma)$ acceptance condition
- run ρ over $w \in \Sigma^\omega$ is accepting if $\mathit{infs}_\rho \models \mathrm{Acc}$
- define the class of ω-regular languages

Transition-based Emerson-Lei Automata

- automata over infinite words



- Fin(0) $\wedge$ Inf(1)
- $aa(bc)^\omega \notin \mathcal{L}(\mathcal{A})$
- $aabbb(c)^\omega \in \mathcal{L}(\mathcal{A})$

- Emerson-Lei acceptance condition
  - $\Gamma = \{$0, 1 $\ldots,$ k-1$\}$
  - $\mathbb{EL}(\Gamma)$ are formulae according to the grammar $\alpha ::= tt \mid ff \mid \mathrm{Inf}(c) \mid \mathrm{Fin}(c) \mid (\alpha \wedge \alpha) \mid (\alpha \vee \alpha)$

- $\mathcal{A} = (Q, \delta, I, \mathsf{p}, \mathrm{Acc})$ over colors Γ
  - Q finite set of states
  - $\delta \subseteq Q \times \Sigma \times Q$ transition relation
  - I initial states
  - $\mathsf{p}\colon \delta \to 2^\Gamma$ colouring of transitions and
  - $\mathrm{Acc} \in \mathbb{EL}(\Gamma)$ acceptance condition
- run ρ over $w \in \Sigma^\omega$ is accepting if $\mathit{infs}_\rho \models \mathrm{Acc}$
- define the class of ω-regular languages

- Büchi Acc = Inf(0)
- Co-Büchi Acc = Fin(0)
- GBA Acc = $\bigwedge_j$ Inf(j)

TELA Complementation

Complementation:

- Given $\mathcal{A}$, get a TELA $\mathcal{A}^{\complement}$ such that $\mathcal{L}(\mathcal{A}^{\complement}) = \overline{\mathcal{L}(\mathcal{A})}$.

TELA Complementation

Complementation:

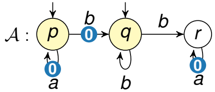
- Given $\mathcal{A}$, get a TELA $\mathcal{A}^{\complement}$ such that $\mathcal{L}(\mathcal{A}^{\complement}) = \overline{\mathcal{L}(\mathcal{A})}$.

Motivation:

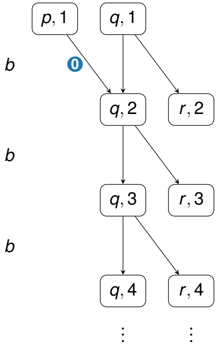
- Model checking of linear-time properties

$$\underbrace{S}_{\text{system}} \models \underbrace{\varphi}_{\text{property}} \rightsquigarrow \quad \mathcal{L}(\mathcal{A}_S) \subseteq \mathcal{L}(\mathcal{A}_\varphi) \quad \rightsquigarrow \quad \mathcal{L}(\mathcal{A}_S) \cap \mathcal{L}(\mathcal{A}_\varphi^{\complement}) = \emptyset$$

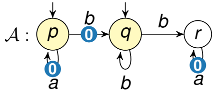
- Termination analysis of programs: Ultimate Automizer
  - removing traces with proved termination
  - difference automaton
- Decision procedures: implements negation
  - S1S: MSO over $(\omega, 0, +1)$
  - QPTL: quantified propositional temporal logic
  - HyperLTL, FO over Sturmian words
- Basic operation for inclusion/equivalence checking



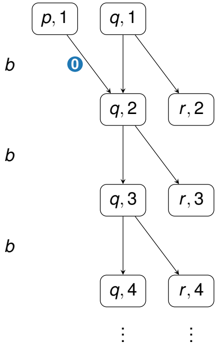
- run DAG $\mathcal{G}_w$: all runs of $\mathcal{A}$ on w



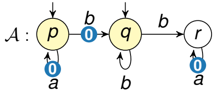
Rank-based Complementation (Inf(0)) [Schewe'09, FKV'06,KV'01]



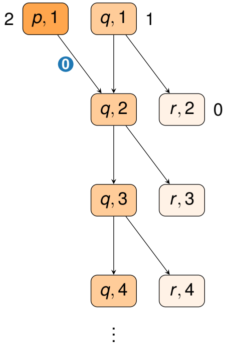
- run DAG $\mathcal{G}_w$: all runs of $\mathcal{A}$ on w



- Labelling algorithm: repeat until $\mathcal{G} \neq \emptyset$ ($i := 0$)
  1. assign rank i to finite vertices and remove them
  2. assign rank $i + 1$ to engangered vertices and remove them
  3. $i := i + 2$
- $w \notin \mathcal{L}(\mathcal{A})$ iff $\max(r) \leq 2n$

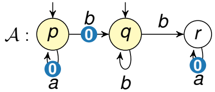


- run DAG $\mathcal{G}_w$: all runs of $\mathcal{A}$ on w

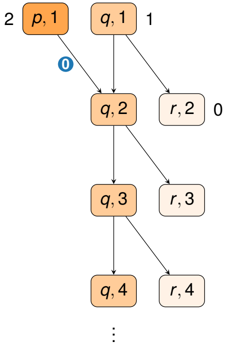


- Labelling algorithm: repeat until $\mathcal{G} \neq \emptyset$ $(i := 0)$
  1. assign rank i to finite vertices and remove them
  2. assign rank $i + 1$ to engangered vertices and remove them
  3. $i := i + 2$
- $w \notin \mathcal{L}(\mathcal{A})$ iff $\max(r) \leq 2n$

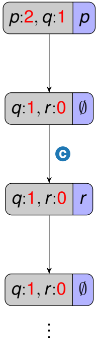
Rank-based Complementation (Inf(⓿)) [Schewe'09, FKV'06,KV'01]



- run DAG $\mathcal{G}_w$: all runs of $\mathcal{A}$ on w



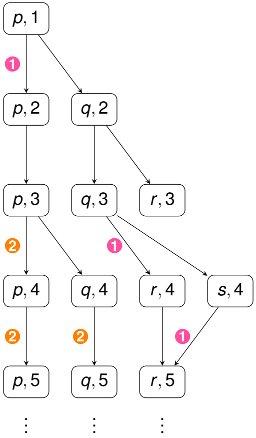
- Labelling algorithm: repeat until $\mathcal{G} \neq \emptyset$ $(i := 0)$
  1. assign rank i to finite vertices and remove them
  2. assign rank $i+1$ to engangered vertices and remove them
  3. $i := i + 2$
- $w \notin \mathcal{L}(\mathcal{A})$ iff $\max(r) \leq 2n$
- complementation algorithm
  - guess rankings and check Inf(⓿)
  - macrostates (S, O, f)
  - nonincreasing ranks wrt δ
  - even rank when traversing ⓿-transition
  - empty breakpoint $\rightsquigarrow$ acc mark



Contribution

Inf-TELA

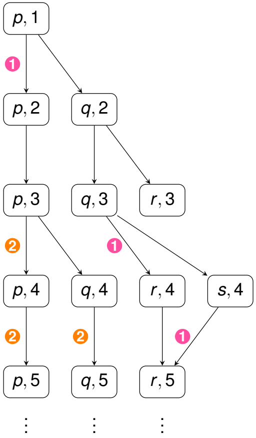
Inf(❶) ∧ Inf(❷) ⇝ ❶ ∨ ❷



- negating φ and transforming to NNF (Fin(ⓒ) ⇝ ⓒ) $\overline{\varphi}$
  - $\varphi =$ Inf(⓿) ∧ (Inf(❶) ∨ Inf(❷)); $\overline{\varphi} =$ ⓿ ∨ (❶ ∧ ❷)
  - minimal models $\mathcal{M} = \{\{$⓿$\}, \{$❶, ❷$\}\}$

Inf-TELA

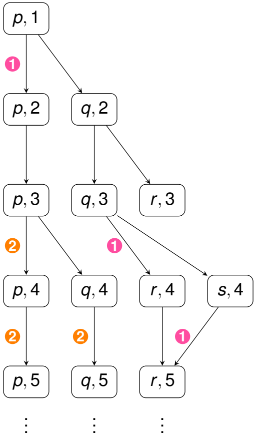
Inf(❶) ∧ Inf(❷) ⇝ ❶ ∨ ❷



- negating φ and transforming to NNF (Fin(ⓒ) ⇝ ⓒ) $\overline{\varphi}$
  - φ = Inf(⓿) ∧ (Inf(❶) ∨ Inf(❷)); $\overline{\varphi}$ = ⓿ ∨ (❶ ∧ ❷)
  - minimal models $\mathcal{M}$ = {{⓿}, {❶, ❷}}
- labelling algorithm: repeat until $\mathcal{G} \neq \emptyset$ ($i := 0$)
  1. set $r(u) := i$; $m(u) := \ell$ for finite vertices
     - remove assigned vertices from $\mathcal{G}$
  2. if there is endangered map $\mu : U \to \mathcal{M}$ then
     - $r(u) := i + 1$; $m(u) := \mu(v)$ for $v \in U$ and $u \in reach_{\mathcal{G}}(v)$
     - remove assigned vertices from $\mathcal{G}$
  3. if there is no endangered map, return $\perp$
  4. $i := i + 2$

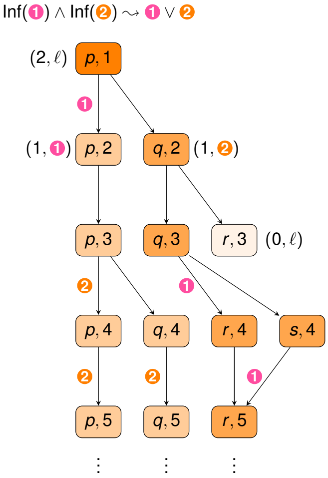
Inf-TELA

Inf(①) ∧ Inf(②) ⇝ ① ∨ ②



- negating φ and transforming to NNF (Fin(ⓒ) ⇝ ⓒ) $\overline{\varphi}$
  - φ = Inf(⓪) ∧ (Inf(①) ∨ Inf(②)); $\overline{\varphi}$ = ⓪ ∨ (① ∧ ②)
  - minimal models $\mathcal{M}$ = {{⓪}, {①, ②}}
- labelling algorithm: repeat until $\mathcal{G} \neq \emptyset$ ($i := 0$)
  1. set $r(u) := i$; $m(u) := \ell$ for finite vertices
     - remove assigned vertices from $\mathcal{G}$
  2. if there is endangered map $\mu : U \to \mathcal{M}$ then
     - $r(u) := i + 1$; $m(u) := \mu(v)$ for $v \in U$ and $u \in reach_{\mathcal{G}}(v)$
     - remove assigned vertices from $\mathcal{G}$
  3. if there is no endangered map, return $\perp$
  4. $i := i + 2$
- always terminates with $i \leq 2n$
- $w \notin \mathcal{L}(\mathcal{A})$ iff the algorithm terminates with (r, m)
- r is tight (or $\max(r) = 0$)

Inf-TELA



- negating φ and transforming to NNF (Fin(ⓒ) $\rightsquigarrow$ ⓒ) $\overline{\varphi}$
  - $\varphi = \mathrm{Inf}(⓪) \land (\mathrm{Inf}(①) \lor \mathrm{Inf}(②))$; $\overline{\varphi} = ⓪ \lor (① \land ②)$
  - minimal models $\mathcal{M} = \{\{⓪\}, \{①, ②\}\}$
- labelling algorithm: repeat until $\mathcal{G} \neq \emptyset$ ($i := 0$)
  1. set $r(u) := i$; $m(u) := \ell$ for finite vertices
     - remove assigned vertices from $\mathcal{G}$
  2. if there is endangered map $\mu : U \to \mathcal{M}$ then
     - $r(u) := i + 1$; $m(u) := \mu(v)$ for $v \in U$ and $u \in reach_{\mathcal{G}}(v)$
     - remove assigned vertices from $\mathcal{G}$
  3. if there is no endangered map, return $\bot$
  4. $i := i + 2$
- always terminates with $i \leq 2n$
- $w \notin \mathcal{L}(\mathcal{A})$ iff the algorithm terminates with (r, m)
- r is tight (or $\max(r) = 0$)

Inf-TELA Algorithm

- guess rankings and a minimal model for each state

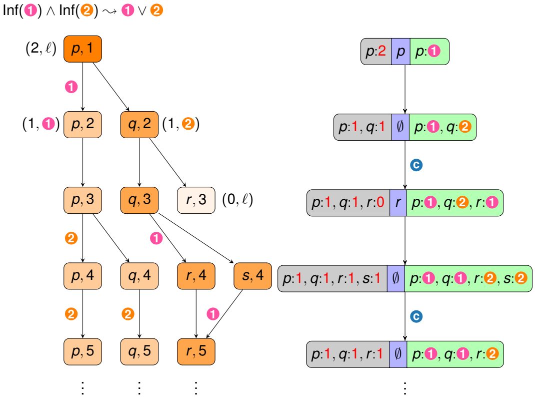
Inf-TELA Algorithm

- guess rankings and a minimal model for each state
- macrostates (S, O, f, μ)
- accepting mark when leaving macrostate with $O = \emptyset$

Inf-TELA Algorithm

- guess rankings and a minimal model for each state
- macrostates (S, O, f, μ)
- accepting mark when leaving macrostate with $O = \emptyset$
- ensure transition consistency
  - nonincreasing ranks wrt δ
  - $\lfloor$decrease$\rfloor$ rank for transitions incompatible with μ
  - $\lfloor$decrease$\rfloor$ rank when model changes
  - assign ℓ to states with even rank

Inf-TELA Algorithm



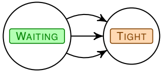
- guess rankings and a minimal model for each state
- macrostates (S, O, f, μ)
- accepting mark when leaving macrostate with $O = \emptyset$
- ensure transition consistency
  - nonincreasing ranks wrt δ
  - $\lfloor$decrease$\rfloor$ rank for transitions incompatible with μ
  - $\lfloor$decrease$\rfloor$ rank when model changes
  - assign ℓ to states with even rank

Inf-TELA Algorithm Optimisations

- use tight rankings Idea of Schewe'09,FKV'06
  - allow only consistent models: (S, μ)-tightness
- waiting and tight part
  - guess the point with tight rankings

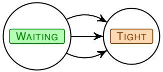
Inf-TELA Algorithm Optimisations

- use tight rankings Idea of Schewe'09,FKV'06
  - allow only consistent models: (S, μ)-tightness
- waiting and tight part
  - guess the point with tight rankings
- macrostates $2^Q \cup \{(S, O, f, i, \mu)\}$
  - f is (S, μ)-tight
  - $O \subseteq S \cap f^{-1}(i)$
  - μ is consistent



Inf-TELA Algorithm Optimisations

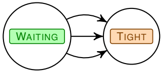
- use tight rankings — Idea of Schewe'09,FKV'06
  - allow only consistent models: (S, μ)-tightness
- waiting and tight part
  - guess the point with tight rankings
- macrostates $2^Q \cup \{(S, O, f, i, \mu)\}$
  - f is (S, μ)-tight
  - $O \subseteq S \cap f^{-1}(i)$
  - μ is consistent

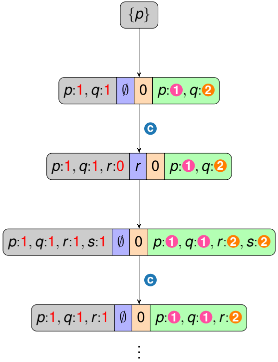


- state complexity
  - Inf-TELA: $\mathcal{O}(k^n \cdot \text{tight}(n+1)) = \mathcal{O}(n(0.76nk)^n)$ for $k = |\mathcal{M}|$
  - GBAs: $\mathcal{O}(k^n \cdot \text{tight}(n+1))$ for k colors

Inf-TELA Algorithm Optimisations

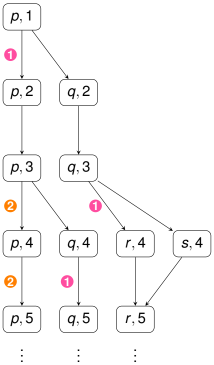
- use tight rankings — Idea of Schewe'09,FKV'06
  - allow only consistent models: (S, μ)-tightness
- waiting and tight part
  - guess the point with tight rankings
- macrostates $2^Q \cup \{(S, O, f, i, \mu)\}$
  - f is (S, μ)-tight
  - $O \subseteq S \cap f^{-1}(i)$
  - μ is consistent



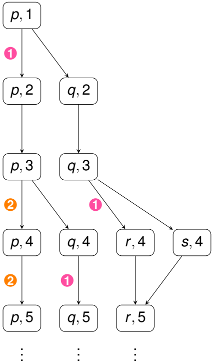


- state complexity
  - Inf-TELA: $\mathcal{O}(k^n \cdot \text{tight}(n+1)) = \mathcal{O}(n(0.76nk)^n)$ for $k = |\mathcal{M}|$
  - GBAs: $\mathcal{O}(k^n \cdot \text{tight}(n+1))$ for k colors

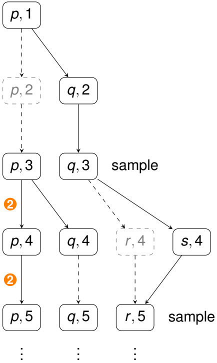
Fin(❶) ∧ Inf(❷)



Fin(❶) ∧ Inf(❷)

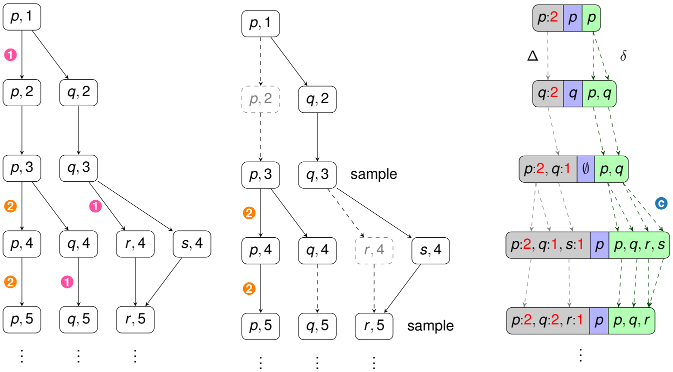


p, 1
❶
p, 2
q, 2
p, 3
q, 3
❷
❶
p, 4
q, 4
r, 4
s, 4
❷
❶
p, 5
q, 5
r, 5



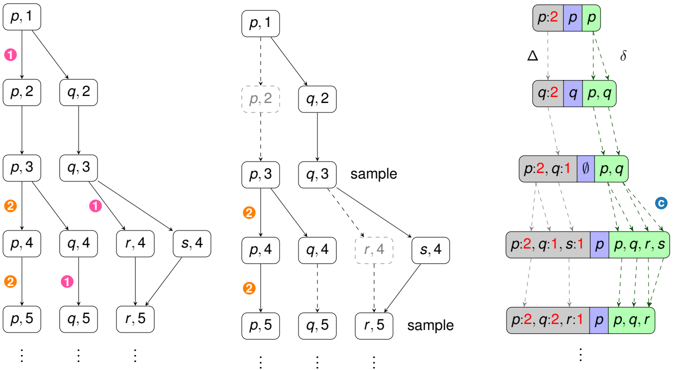
p, 1
p, 2
q, 2
p, 3
q, 3
sample
❷
p, 4
q, 4
r, 4
s, 4
❷
p, 5
q, 5
r, 5
sample

Fin(❶) $\wedge$ Inf(❷)



- **Relaxed run DAG $\mathcal{G}_w^{\Delta}$**
  - restriction to Δ with sampling
  - no ❶-edges
  - accepting incomplete runs
  - inf-often sampling (arbitrary)
- **Complementation algorithm**
  - check Inf(❷) using rank-based
  - change the ranking domain after sampling

Fin(❶) ∧ Inf(❷)



- **Relaxed run DAG $\mathcal{G}_w^{\Delta}$**
  - restriction to Δ with sampling
  - no ❶-edges
  - accepting incomplete runs
  - inf-often sampling (arbitrary)
- **Complementation algorithm**
  - check Inf(❷) using rank-based
  - change the ranking domain after sampling

- Sampling is steered by the Algorithm
  - when the accepting mark is emitted (empty breakpoint)

Fin(❶) $\wedge \varphi$

- modular construction based on a subprocedure $\mathbb{S}^{\varphi}_{\Delta}$ for φ
- acceptance checking of relaxed run DAGs

Fin(❶) $\wedge\ \varphi$

- modular construction based on a subprocedure $\mathbb{S}^{\varphi}_{\Delta}$ for φ
- acceptance checking of relaxed run DAGs
- master algorithm simulates creation of relaxed run DAGs
  - master's macrostates $2^Q \times 2^Q \times \mathrm{M}$

Fin(❶) $\land \varphi$

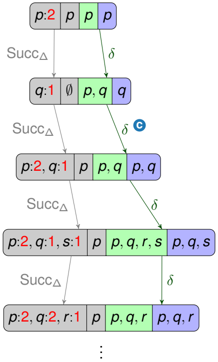
- modular construction based on a subprocedure $\mathbb{S}_{\Delta}^{\varphi}$ for φ
- acceptance checking of relaxed run DAGs
- master algorithm simulates creation of relaxed run DAGs
  - master's macrostates $2^Q \times 2^Q \times \mathsf{M}$
- subprocedure $\mathbb{S}_{\Delta}^{\varphi} = (\mathsf{M}, \mathsf{M}_0, \mathsf{Succ}_{\Delta}, \mathsf{EmptyBreak})$
  - M: set of macrostates
  - M_0: set of initial macrostates
  - $\mathsf{Succ}_{\Delta} : 2^Q \times \Sigma \times \mathsf{M} \to 2^{\mathsf{M}}$: transition function
  - EmptyBreak: empty breakpoint predicate; EmptyBreak $\rightsquigarrow$ acc mark

Fin(❶) $\wedge\ \varphi$

- modular construction based on a subprocedure $\mathbb{S}^{\varphi}_{\Delta}$ for φ
- acceptance checking of relaxed run DAGs
- master algorithm simulates creation of relaxed run DAGs
  - master's macrostates $2^Q \times 2^Q \times \mathsf{M}$
- subprocedure $\mathbb{S}^{\varphi}_{\Delta} = (\mathsf{M}, \mathsf{M}_0, \mathsf{Succ}_{\Delta}, \mathsf{EmptyBreak})$
  - M: set of macrostates
  - M_0: set of initial macrostates
  - $\mathsf{Succ}_{\Delta} : 2^Q \times \Sigma \times \mathsf{M} \to 2^{\mathsf{M}}$: transition function
  - EmptyBreak: empty breakpoint predicate; EmptyBreak $\leadsto$ acc mark
- correctness: $\mathcal{G}$ is not acc wrt φ iff EmptyBreak holds ∞-often on $\mathcal{G}$

Fin(1) $\wedge \varphi$

- modular construction based on a subprocedure $\mathbb{S}^{\varphi}_{\Delta}$ for φ
- acceptance checking of relaxed run DAGs
- master algorithm simulates creation of relaxed run DAGs
  - master's macrostates $2^Q \times 2^Q \times \mathsf{M}$
- subprocedure $\mathbb{S}^{\varphi}_{\Delta} = (\mathsf{M}, \mathsf{M}_0, \mathsf{Succ}_{\Delta}, \mathsf{EmptyBreak})$
  - M: set of macrostates
  - M_0: set of initial macrostates
  - $\mathsf{Succ}_{\Delta} : 2^Q \times \Sigma \times \mathsf{M} \to 2^{\mathsf{M}}$: transition function
  - EmptyBreak: empty breakpoint predicate; EmptyBreak $\rightsquigarrow$ acc mark



- correctness: $\mathcal{G}$ is not acc wrt φ iff EmptyBreak holds ∞-often on $\mathcal{G}$
- complexity: split $\mathsf{Succ}_{\Delta} = \mathsf{SuccAct}_{\Delta} \cup \mathsf{SuccTrack}_{\Delta}$ + macrostates
  - active, tracking transition functions
  - simpler structure for tracking; richer for active
  - EmptyBreak for active only

Modular Construction Instantiation Fin(❶) $\land \varphi$

- Co-Büchi ($\varphi = tt$)
  - $M^{tt} = 2^Q$
  - automaton size: $\mathcal{O}(3^n)$

Modular Construction Instantiation Fin(❶) $\wedge \varphi$

- Co-Büchi ($\varphi = tt$)
  - $\mathsf{M}^{tt} = 2^Q$
  - automaton size: $\mathcal{O}(3^n)$

- Rabin automata ($\varphi = \mathrm{Inf}(❷)$); single Rabin pair
  - $\mathsf{M}^{\mathrm{inf}} = \overbrace{2^Q \cup (\mathcal{T} \times 2^Q \times \{0, 2, \ldots, 2n-2\})}^{\mathsf{M}^{\mathrm{inf}}_{\mathrm{Act}}} \cup \overbrace{(\mathcal{T} \times \{0, 2, \ldots, 2n-2\})}^{\mathsf{M}^{\mathrm{inf}}_{\mathrm{Track}}}$
  - single Rabin pair: $\mathcal{O}(\mathrm{tight}(n+1))$
  - Rabin automaton: $\mathcal{O}(\mathrm{tight}(n+1)^k) = \mathcal{O}(n^k (0.76n)^{nk})$

Modular Construction Instantiation Fin(❶) $\land \varphi$

- Co-Büchi ($\varphi = tt$)
  - $\mathrm{M}^{tt} = 2^Q$
  - automaton size: $\mathcal{O}(3^n)$
- Rabin automata ($\varphi = \mathrm{Inf}(❷)$); single Rabin pair
  - $\mathrm{M}^{\mathrm{inf}} = \overbrace{2^Q \cup (\mathcal{T} \times 2^Q \times \{0, 2, \ldots, 2n-2\})}^{\mathrm{M}^{\mathrm{inf}}_{\mathrm{Act}}} \cup \overbrace{(\mathcal{T} \times \{0, 2, \ldots, 2n-2\})}^{\mathrm{M}^{\mathrm{inf}}_{\mathrm{Track}}}$
  - single Rabin pair: $\mathcal{O}(\mathrm{tight}(n+1))$
  - Rabin automaton: $\mathcal{O}(\mathrm{tight}(n+1)^k) = \mathcal{O}(n^k(0.76n)^{nk})$
- Generalized Rabin automata ($\varphi = \bigwedge_j \mathrm{Inf}(ⓙ)$)
  - $\mathrm{M}^{\land\mathrm{inf}} = 2^Q \cup (\mathcal{T} \times 2^Q \times \{0, 2, \ldots, 2n-2\} \times \mathrm{LM}) \cup (\mathcal{T} \times \{0, 2, \ldots, 2n-2\} \times \mathrm{LM})$
  - Gen. Rabin automaton: $\mathcal{O}(\ell^{nk}\mathrm{tight}(n+1)^k)$; ℓ number of Infs; k pairs

Modular Construction Instantiation Fin(1) $\wedge \varphi$

- Co-Büchi ($\varphi = tt$)
  - $\mathrm{M}^{tt} = 2^Q$
  - automaton size: $\mathcal{O}(3^n)$
- Rabin automata ($\varphi = \mathrm{Inf}(2)$); single Rabin pair
  - $\mathrm{M}^{\mathrm{inf}} = \overbrace{2^Q \cup (\mathcal{T} \times 2^Q \times \{0, 2, \ldots, 2n-2\})}^{\mathrm{M}^{\mathrm{inf}}_{\mathrm{Act}}} \cup \overbrace{(\mathcal{T} \times \{0, 2, \ldots, 2n-2\})}^{\mathrm{M}^{\mathrm{inf}}_{\mathrm{Track}}}$
  - single Rabin pair: $\mathcal{O}(\mathrm{tight}(n+1))$
  - Rabin automaton: $\mathcal{O}(\mathrm{tight}(n+1)^k) = \mathcal{O}(n^k(0.76n)^{nk})$
- Generalized Rabin automata ($\varphi = \bigwedge_j \mathrm{Inf}(j)$)
  - $\mathrm{M}^{\wedge\mathrm{inf}} = 2^Q \cup (\mathcal{T} \times 2^Q \times \{0, 2, \ldots, 2n-2\} \times \mathrm{LM}) \cup (\mathcal{T} \times \{0, 2, \ldots, 2n-2\} \times \mathrm{LM})$
  - Gen. Rabin automaton: $\mathcal{O}(\ell^{nk}\mathrm{tight}(n+1)^k)$; ℓ number of Infs; k pairs
- TELA: $\mathcal{O}(k^{n2^k}\mathrm{tight}(n+1)^{2^k}) = \mathcal{O}(k^{n2^k}(0.76nk)^{n2^k})$; k colours

Conclusion

- rank-based complementation of TELA
  - Inf-TELA
  - modular construction for Fin $\wedge\, \varphi$
  - better upper-bounds than the SOTA

Conclusion

- rank-based complementation of TELA
  - Inf-TELA
  - modular construction for Fin $\wedge\ \varphi$
  - better upper-bounds than the SOTA
- what's more in the paper?
  - details of constructions
  - parity automata
  - proofs

Conclusion

- rank-based complementation of TELA
  - Inf-TELA
  - modular construction for Fin $\wedge \varphi$
  - better upper-bounds than the SOTA
- what's more in the paper?
  - details of constructions
  - parity automata
  - proofs
- next steps?
  - implementation
  - practical optimisations

Conclusion

- rank-based complementation of TELA
  - Inf-TELA
  - modular construction for Fin $\wedge \varphi$
  - better upper-bounds than the SOTA
- what's more in the paper?
  - details of constructions
  - parity automata
  - proofs
- next steps?
  - implementation
  - practical optimisations

THANK YOU!