Alternation Elimination by Complementation

Christian Dax, Felix Klaedtke

ETH Zurich

Recent results and ongoing work

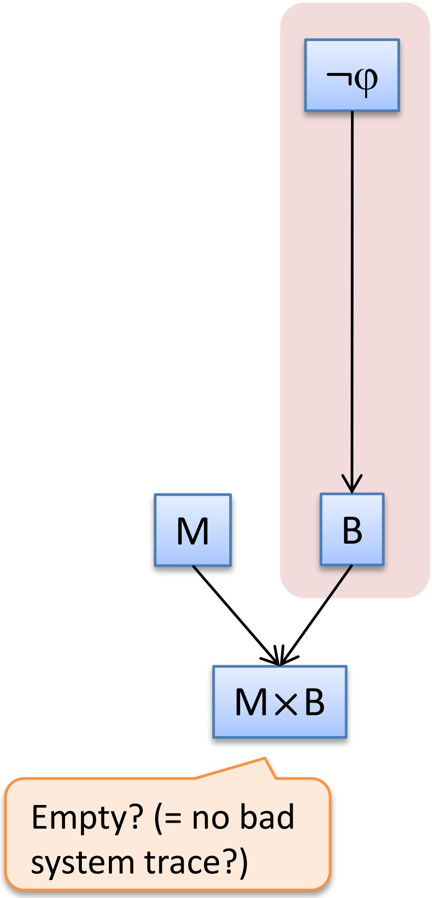
LMU Munich, July 22nd, 2008

Motivation I: Finite-State Model Checking

- Question: system M fulfills specification φ?
  - M : nondeterministic automaton (all system traces)
  - φ : temporal formula

- Automata-based approach:
  - Reduction to emptiness check of nondet. automaton

  1. Negated specification → nondet. automaton B (bad traces)
  2. Product of M and B (system traces that are bad)
  3. Emptiness check of M×B (is there a bad system trace?)

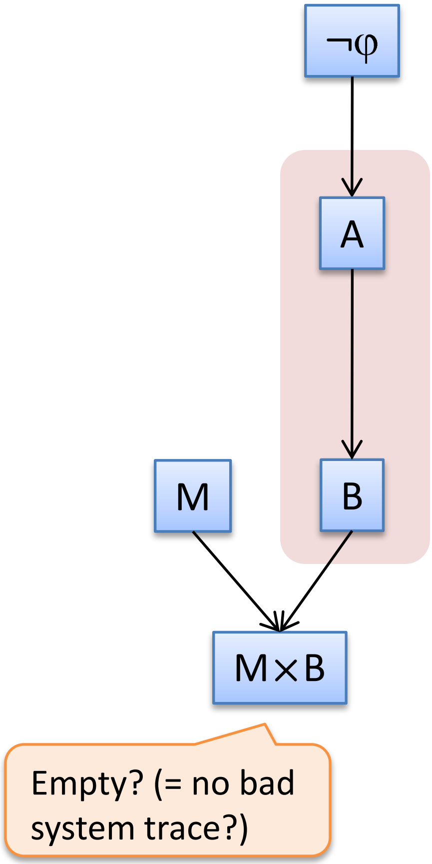


- This talk: focus on step 1.

Motivation II: Alternation Elimination

- What is crucial?
  1. Specification (with past operators) $\rightarrow$ (2-way) alternating automaton $\Rightarrow$ direct/easy
  2. 2-way alternating $\rightarrow$ 1-way nondeterministic automaton $\Rightarrow$ complex/difficult
  3. Emptiness check for 1-way nondeterministic automaton $\Rightarrow$ efficient/easy

- This talk: focus on step 2 + a bit on step 1.



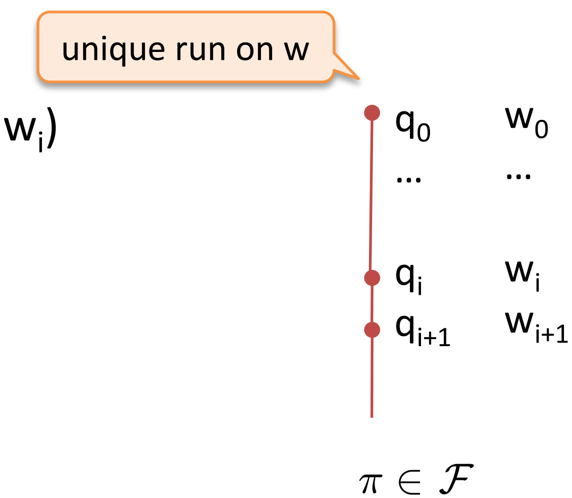
Outline

1. Background: automata types
2. From alternating to nondeterministic automata
3. Complementing loop-free 2-way
   nondeterministic Büchi automata (NBA)
4. Outlook: from PSL logic with past operators to NBAs

Background: Automata Types

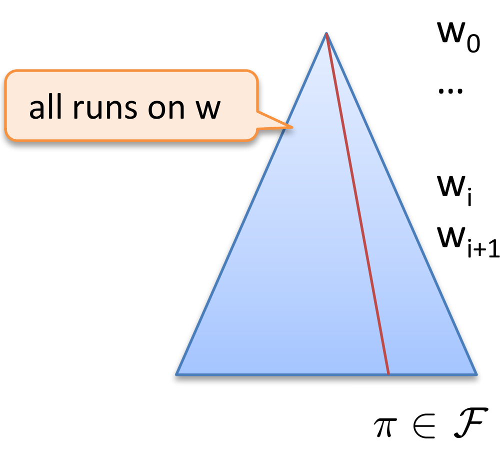
Deterministic Automata (DA)

- A DA is a tuple $(Q, \Sigma, \delta, q_0, \mathcal{F})$
  - δ: $Q \times \Sigma \to Q$ transition function
  - $\mathcal{F} \subseteq Q^\omega$ set of sequences over Q that are accepting
  - Remark: Büchi and co-Büchi conditions are given as a subset $F \subseteq Q$
    $\mathcal{F}_F = \{\pi \in Q^\omega \mid \pi$ visits F-states ∞-often$\}$ (Büchi condition)
    $\mathcal{F}_F = \{\pi \in Q^\omega \mid \pi$ does not visit F-states ∞-often$\}$ (co-Büchi condition)
- For a word $w = w_0 w_1 \ldots$
  - A run $q_0 q_1 \ldots$ is a sequence of states with $q_{i+1} = \delta(q_i, w_i)$
  - w is accepted :⇔ the run $\pi = q_0\, q_1 \ldots$ on w is in $\mathcal{F}$
- Syntax: 'automaton as relation over words'
  - $\mathcal{A}(w)$:⇔ word w is accepted by automaton $\mathcal{A}$



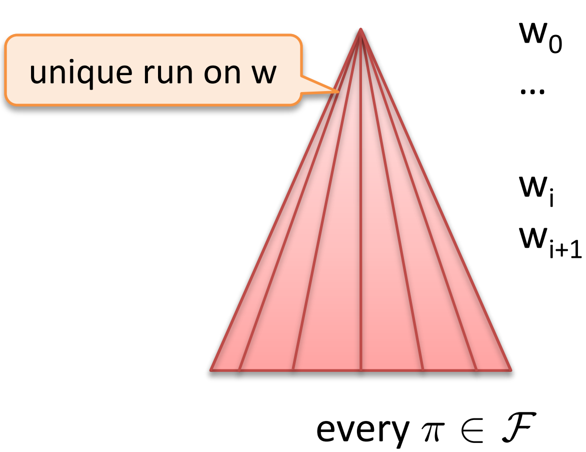
Nondeterministic/Universal Automata (NA/UA)

- An NA/UA is a tuple $(Q, \Sigma, \delta, q_0, \mathcal{F})$
  - δ: $Q \times \Sigma \rightarrow 2^Q$ transition function



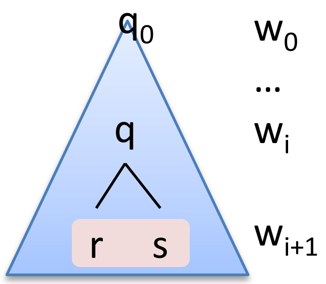
- For a word $w = w_0 w_1 \ldots$
  - A nondeterministic run $q_0 q_1 \ldots$ is a sequence of states with $q_{i+1} \in \delta(q_i, w_i)$
  - w is accepted :$\Leftrightarrow$ there is a run on w that is in $\mathcal{F}$

- For a word $w = w_0 w_1 \ldots$
  - A universal run is a Q-labeled tree
    - the root is labeled by q_0, and
    - a q-labeled node in level i has children labeled by $\delta(q, w_i)$
    - w is accepted :$\Leftrightarrow$ every path in the run is in $\mathcal{F}$



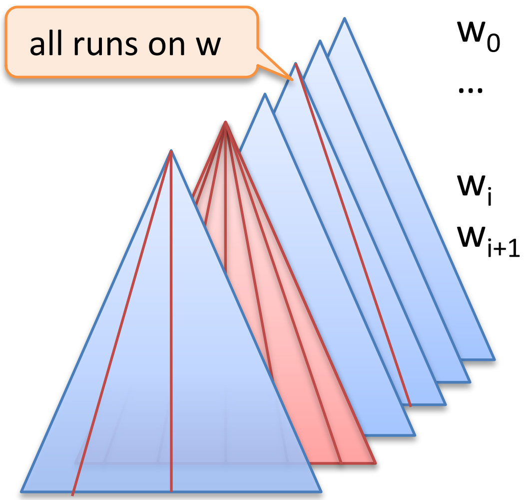
Alternating Automata (AA)

- An AA is a tuple $(Q, \Sigma, \delta, q_0, \mathcal{F})$
  - $\delta: Q \times \Sigma \rightarrow \mathcal{B}^{+}(Q)$ transition function
  - Here, we assume that $\delta(q, a)$ is in DNF, for all (q, a)



$\delta(q, w_i) = (r \wedge s) \vee (s \wedge t)$

- For a word $w = w_0 w_1 \ldots$
  - A run is a Q-labeled tree, where
    - the root is labeled by q_0, and
    - a q-labeled node in level i has children that are labeled by one of the monomials of $\delta(q, w_i)$
  - a run is accepting :⇔ every path is in $\mathcal{F}$
  - w accepted :⇔ there is an accepting run



every $\pi \in \mathcal{F}$

From Alternating to Nondeterministic Automata

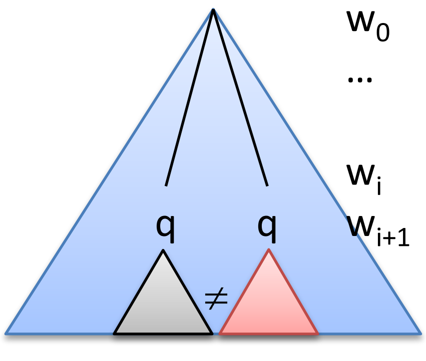
Related Work

- We use building blocks that appeared in
  - Vardi (POPL ’88, ICALP ‘98),
  - Miyano-Hayashi (TCS ’92),
  - Lange-Stirling (LICS ’01),
  - Kupferman-Piterman-Vardi (CONCUR ’01),
  - Gastin-Oddoux (CAV ’01, MFCS ‘03),
  - Dax-Hofmann-Lange (FSTTCS ’06).

- We unify and generalize building blocks:
  - Theses papers solve particular translation problems.
  - We identify the main ingredients of the idea and investigate for which class of translations this idea can be used.
  - Unify and simplify constructions and proofs.

Word Representation of Memoryless Runs

- Memoryless automata
  - A run is memoryless :⇔ equally labeled nodes in the same level have equally labeled subtrees
  - An AA is memoryless :⇔ every accepted word has an memoryless accepting run
  - Remark: Rabin automata are memoryless.



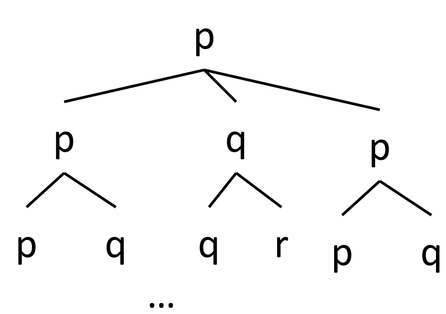
not memoryless

- Encode memoryless run as word $f_0 f_1 f_2 \ldots \in (Q \to 2^Q)^\omega$
- $f_i(q)$: 'labels of children of q-labeled node in level i'

$f_0(p) = \{p, q\}$

$f_1(p) = \{p, q\}$, $f_1(q) = \{q, r\}$

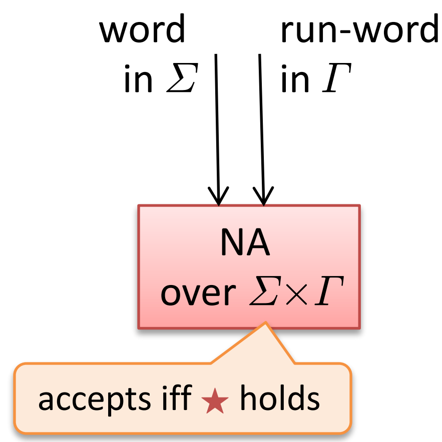
$f_2(p) = \ldots$, $f_2(q) = \ldots$, $f_2(r) = \ldots$



Alternation Elimination

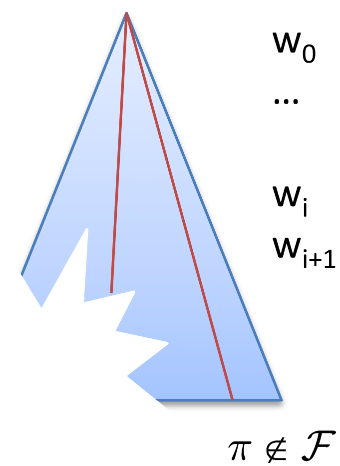
- Let $\mathcal{A} = (Q, \Sigma, \delta, q_0, \mathcal{F})$ be an AA and $\Gamma := (Q \to 2^Q)^\omega$
- A word w is accepted
  - $\Leftrightarrow$ there is a run on w such that all paths are in $\mathcal{F}$
  - $\Leftrightarrow \exists\, r: \forall\, \pi: r \in runs(w) \wedge (\pi \in r \to \pi \in \mathcal{F})$
  - $\Leftrightarrow \exists\, r: \neg\, \exists\, \pi: r \notin runs(w) \vee (\pi \in r \wedge \pi \notin \mathcal{F})$ ★
  - $\Leftrightarrow \exists\, r: \neg\, \mathcal{B}(w, r)$

'refuter's strategy'



- It is easy to build an NA $\mathcal{B}$ over $\Sigma\times\Gamma$ for ★
  - $\mathcal{B} := (Q, \Sigma\times\Gamma, \eta, q_0, Q^\omega \setminus \mathcal{F})$
  - $\eta(q, (a,f)) := \begin{cases} f(q) & f(q) \text{ is monomial in } \delta(q, a) \\ \{acc\text{-}sink\} & \text{otherwise} \end{cases}$

- Finally: complement the NA $\mathcal{B}$ and project it on Σ.



Some Instances

- Extension: alternation elimination for 2-way automata
  1. From given 2-way AA over Σ, construct 2-way NA
  2. Complement 2-way NA + eliminate bidirectionality
  3. Project resulting 1-way NA on Σ

- Translations to 1-way NBAs

	1-Weak Büchi LTL (+ Past)	**Büchi** PSL (+ Past)	**Parity** μLTL (+ Past)	**Rabin**
1-way	$O(n2^{n})$	$O(2^{2n})$	$O(2^{nk \log n})$	$\mathbf{O(2^{nk \log nk})}$
2-way	$\mathbf{O(n2^{3n})}$	$O(2^{n*n})$	$O(2^{nk*nk})$	
2-way + loop-free	$O(n2^{2n})$	$\mathbf{O(2^{4n})}$	-- in progress --	-- in progress --

Some Instances

- Extension: alternation elimination for 2-way automata
  1. From given 2-way AA over Σ, construct 2-way NA
  2. Complement 2-way NA + eliminate bidirectionality
  3. Project resulting 1-way NA on Σ

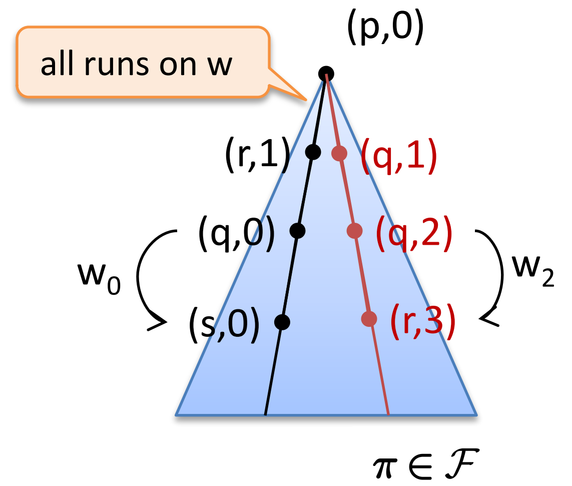
- Some instances

	1-Weak Büchi LTL (+ Past)	**Büchi** PSL (+ Past)	**Parity** μLTL (+ Past)	**Rabin**
1-way	$O(n2^{n})$	$O(2^{2n})$	$O(2^{nk \log n})$	$O(2^{nk \log nk})$
2-way	$O(n2^{3n})$	$O(2^{n^*n})$	$O(2^{nk^*nk})$	
2-way + loop-free	$O(n2^{2n})$	$\mathbf{O(2^{4n})}$	-- in progress --	-- in progress --

Complementing Loop-Free 2-way Nondeterministic Büchi Automata (NBA)

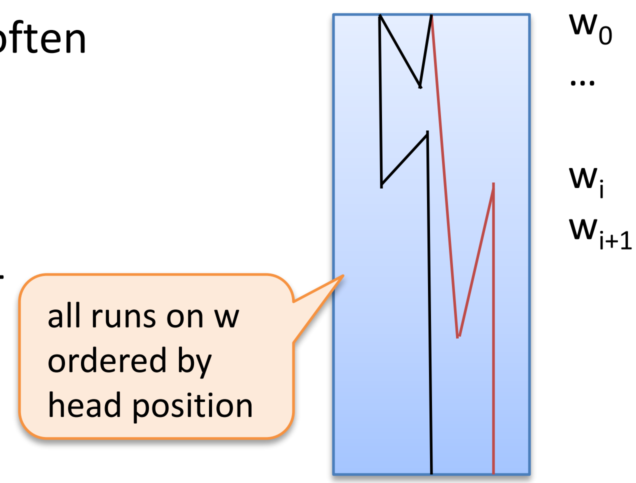
2-Way Nondeterministic Büchi Automata (2NBA)

- A 2NBA is a tuple $(Q, \Sigma, \delta, q_0, F)$
  - $\delta: Q \times \Sigma \to 2^{Q \times \{-1, 0, 1\}}$ transition function
  - Additional info where to move the read-only head



- For a word $w = w_0 w_1 \ldots$
  - A configuration (q, j) is a pair in $Q \times$ 'head positions'
  - A run $(q_0, j_0)\ (q_1, j_1) \ldots$ is a sequence of configurations with $(q_{i+1}, j_{i+1} - j_i) \in \delta(q_i, w\_j_i)$
  - w accepted $\Leftrightarrow$ ex. run on w that visits F-states ∞-often

- For AAs, we have $Q \times$ 'head positions'-labeled run-trees

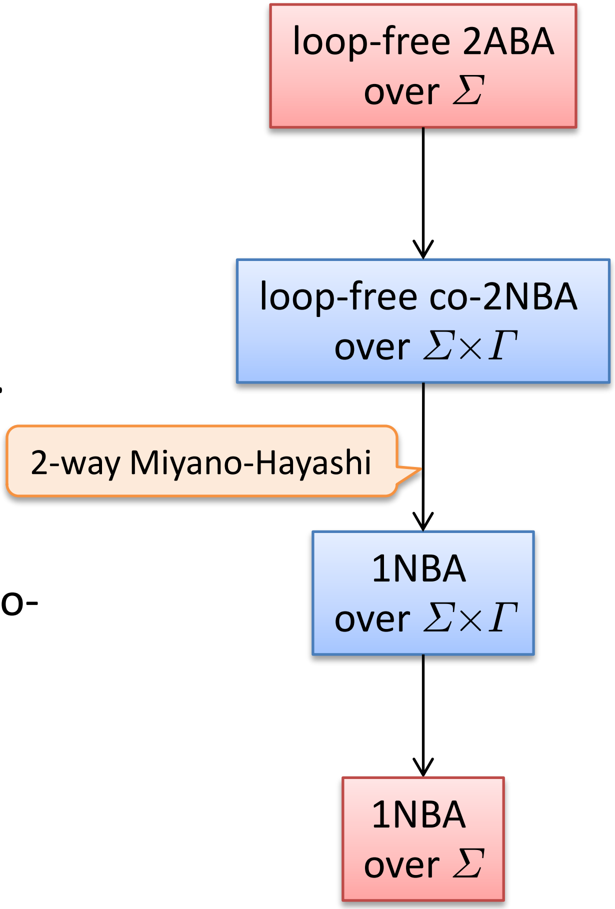


From Loop-Free 2-Way ABA to 1-Way NBA

- Loop-freeness
  - A run of an AA is loop-free :⇔ for every path, no configuration occurs twice on the path
  - An AA is loop-free :⇔ every run is loop-free

- Lemma: if AA is loop-free then the NA is loop-free.

- Loop-free 2-way ABA $\rightarrow$ 1-way NBA
  1. For 2-way ABA over Σ, construct loop-free 2-way co-NBA over $\Sigma \times \Gamma$
  2. Complement result with 2-way Miyano-Hayashi
  3. Project resulting 1-way NBA on Σ

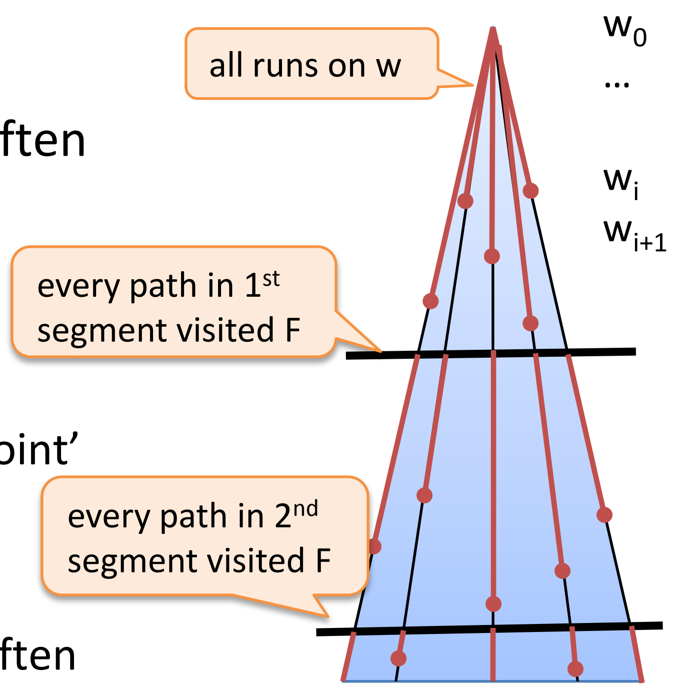


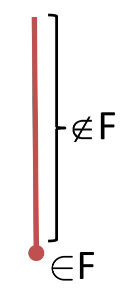
1-Way Miyano-Hayashi Complementation

- A co-NBA $\mathcal{A}$ = (Q, Σ, δ, q_0, F) accepts a word w
  :⇔ ex. run on w that does not visit F-states ∞-often

- NBA for the complement
  - $\mathcal{B}$:= ($2^Q \times 2^Q$, Σ, η, ({q_0},$\emptyset$), $2^Q \times \{\emptyset\}$)
  - η((R, $\emptyset$), a) := (δ(R,a), δ(R,a) \ F) ‘breakpoint’
  - η((R, S), a) := (δ(R,a), δ(S,a) \ F)

  - w accepted ⇔ every run on w visits F-states ∞-often
  - Subset-construction with R-component:
    compute all runs in parallel (black lines)
  - States of S-component have to visit F (red lines)
  - $2^Q \times \{\emptyset\}$ is visited ∞-often ⇔ every run visits F ∞-often



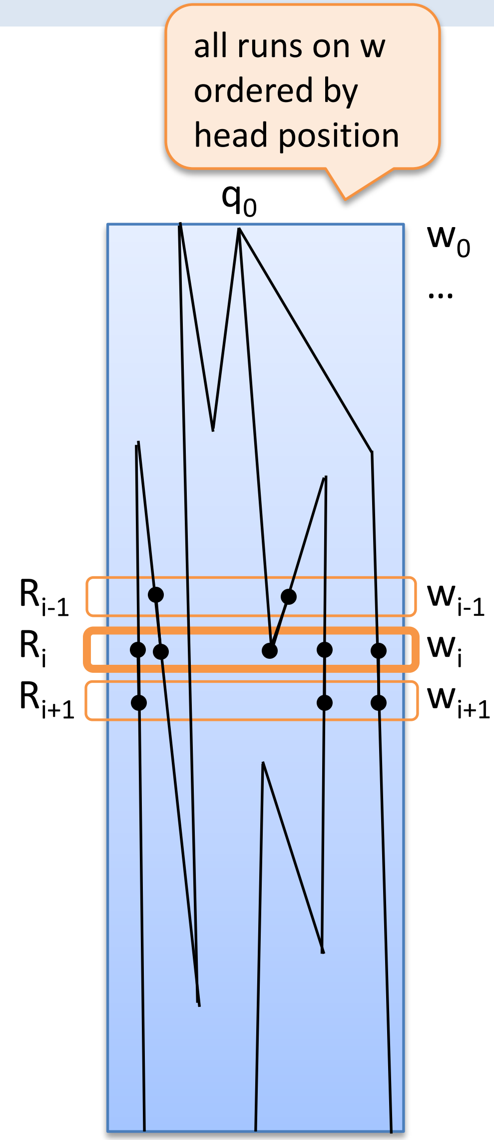


2-Way Miyano-Hayashi Complementation

- A loop-free co-2NBA $\mathcal{A}$ = (Q, Σ, δ, q_0, F) accepts w
  :⇔ ex. run on w that does not visit F-states ∞-often

- 1-way NBA for the complement
  - w rejected ⇔ every run on w visits F ∞-often

  1. Guess sequence $R_0R_1\ldots \in (2^Q)^\omega$ that represents all runs on w ordered by head positions.
  2. Check locally that guess is correct:
     if $p \in R_i$ and $(q, d) \in \delta(p, w_i)$ then $q \in R_{i+d}$

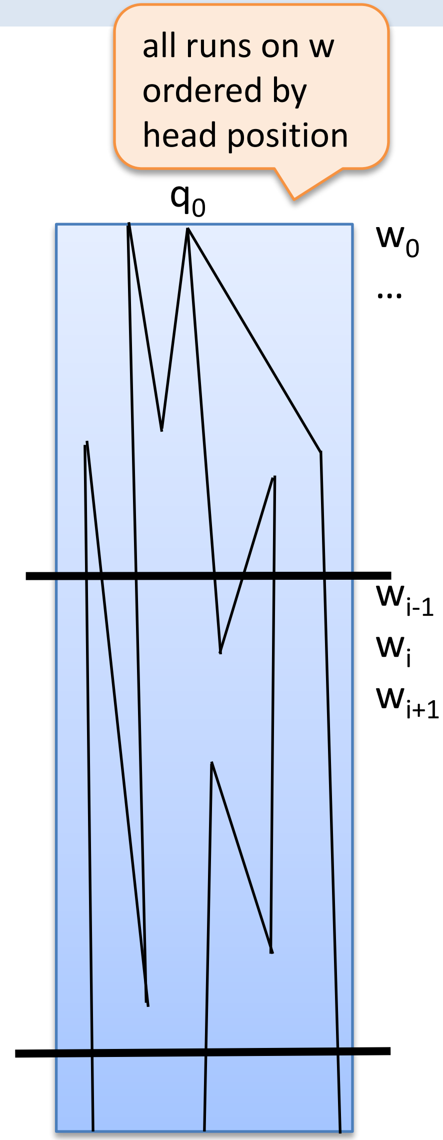


2-Way Miyano-Hayashi Complementation

- A loop-free co-2NBA $\mathcal{A}$ = (Q, Σ, δ, q_0, F) accepts w
  :⇔ ex. run on w that does not visit F-states ∞-often

- 1-way NBA for the complement
  - w rejected ⇔ every run on w visits F ∞-often

1. Guess sequence $R_0R_1\ldots \in (2^Q)^\omega$ that represents all runs on w ordered by head positions.
2. Check locally that guess is correct:
   if $p \in R_i$ and $(q, d) \in \delta(p, w_i)$ then $q \in R_{i+d}$
3. Guess breakpoints:
   - partitioning of the R-sequence in segments
   - each run starting at the previous breakpoint visits F before reaching the breakpoint

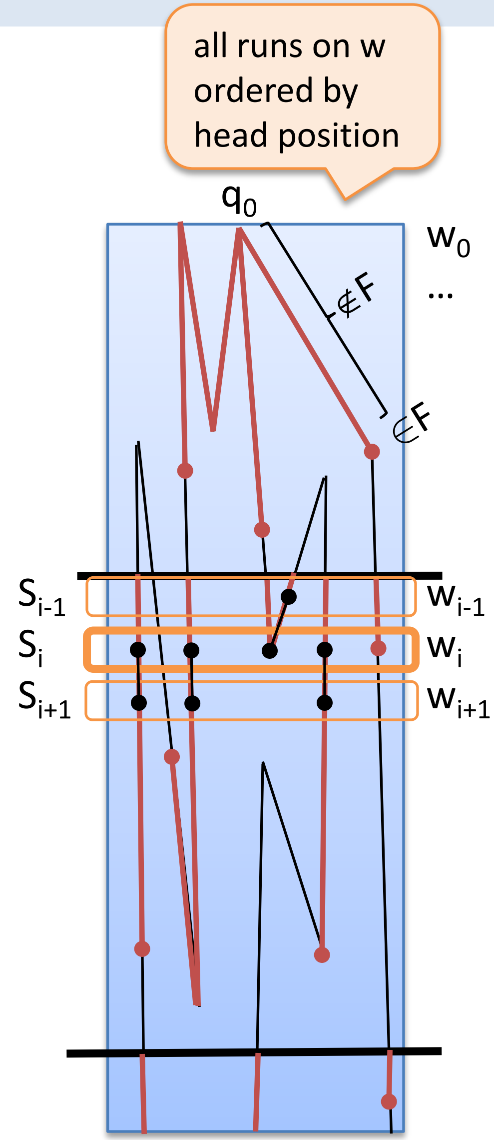


2-Way Miyano-Hayashi Complementation

- A loop-free co-2NBA $\mathcal{A}$ = (Q, Σ, δ, q_0, F) accepts w :⇔ ex. run on w that does not visit F-states ∞-often

- 1-way NBA for the complement
  - w rejected ⇔ every run on w visits F ∞-often

4. Guess sequence $S_0S_1\ldots \in (2^{Q \setminus F})^\omega$ that represents all runs from q_0 or a breakpoint to an F-state.
5. Check locally that guess is correct:
   if $p \in S_i$, $(q, d) \in \delta(p, w_i)$ and $q \notin F$ then either $q \in S_{i+d}$ or $S_{i+d} = \emptyset$ (breakpoint).



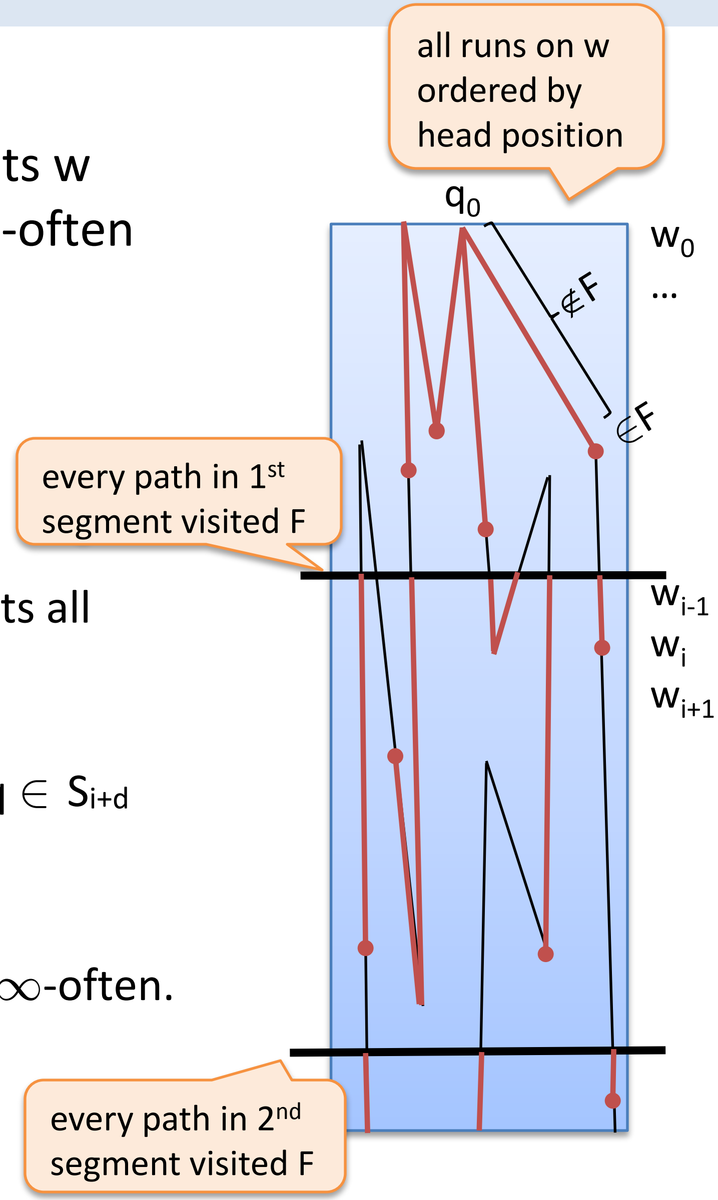
2-Way Miyano-Hayashi Complementation

- A loop-free co-2NBA $\mathcal{A} = (Q, \Sigma, \delta, q_0, F)$ accepts w
  :⇔ ex. run on w that does not visit F-states ∞-often

- 1-way NBA for the complement
  - w rejected ⇔ every run on w visits F ∞-often

4. Guess sequence $S_0S_1\ldots \in (2^{Q \setminus F})^\omega$ that represents all runs from q_0 or a breakpoint to an F-state.
5. Check locally that guess is correct:
   if $p \in S_i$, $(q, d) \in \delta(p, w_i)$ and $q \notin F$ then either $q \in S_{i+d}$ or $S_{i+d} = \emptyset$ (breakpoint).

6. Check that pattern '$S_i = \emptyset$, $S_{i+1} = R_{i+1} \setminus F$' occurs ∞-often.



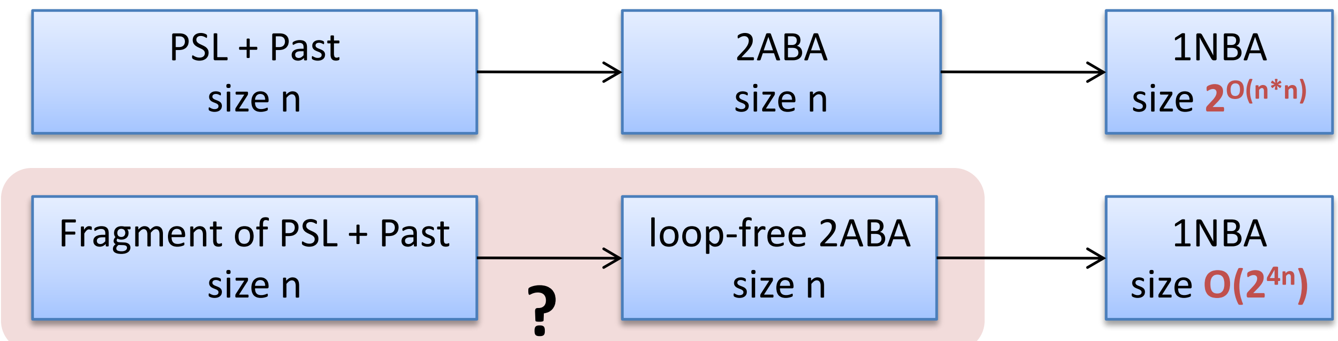
Outlook: From PSL with Past to NBAs

Outlook: PSL with Past Operators

- linear-time fragment of PSL = LTL + (semi-)regular expressions

- [Gastin, Oddoux] LTL + Past → loop-free 2ABA
- For which fragment of PSL + Past is that possible?

- The benefit would be



Fragment of PSL with Past Operators

- Fragments that can be translated to loop-free ABAs
  1. Pure future PSL
  2. LTL + Past
  3. Boolean combinations of 1. and 2.
  4. …?

- We are quite sure that even the whole linear-time fragment can be translated to loop-free ABAs.
  - Substitute regular expressions by propositions in PSL + Past formula
  - Translate LTL + Past formula to loop-free AA
  - Substitute the states for the propositions by AA for regular expressions.

Conclusion

- Construction scheme for translating AAs to NAs
  - Requires complementation construction for NA with co-acceptance condition
  - Requires AA to accept by memoryless runs
  - 3 new translations
  - Other translations can be seen as instances:
    simplify + unify constructions and proofs
- Novel complementation for loop-free co-2NBAs
  - 1-way Miyano-Hayashi can be seen as special case
  - Constructions of Gastin-Oddoux can be seen as special cases

- Ongoing and future work
  - Scheme for automata that do not accept by memoryless runs
  - Translations for fragments of PSL and μLTL with past operators:
    need of complementation for loop-free 2NParityA
  - Practical experiences for 2-way translations