



Rigorous Approximated Determinization of Weighted Automata

Benjamin Aminof (Hebrew University)

Orna Kupferman (Hebrew University)

Robby Lampert (Weizmann Institute)

Israel

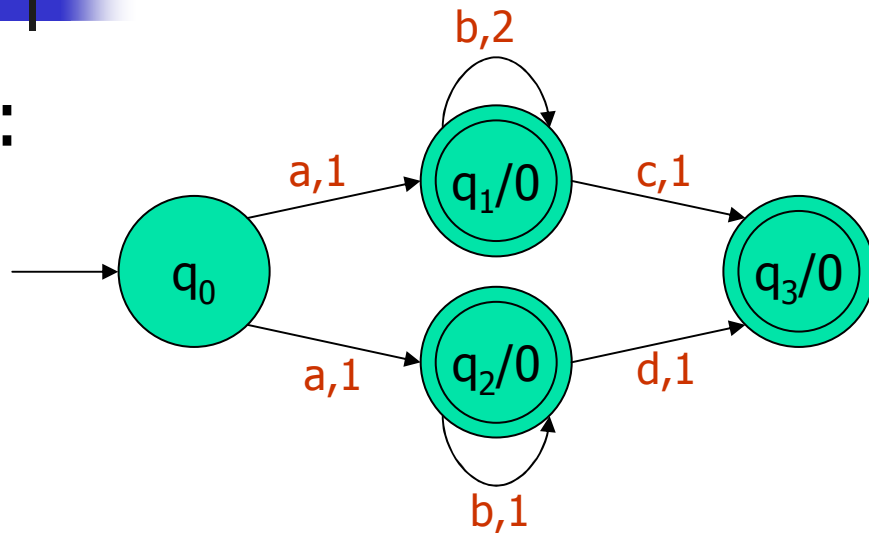


Outline

- n Weighted automata
- n Determinizability of weighted automata
- n Mohri's determinization algorithm
- n Approximated-determinization algorithm
- n Correctness and termination
- n Summary
- n Future work

Weighted Automata (WFA)

\mathcal{A} :



weight functions

c : transitions $\rightarrow \mathbb{R}^{\geq 0}$

f : accepting states $\rightarrow \mathbb{R}^{\geq 0}$

$w=abc$

$$\text{cost}(w) = (1+2+1)+0=4$$

$w=abbd$

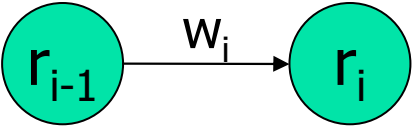
$$\text{cost}(w) = (1+1+1+1)+0=4$$


$w=abb$

$$\text{cost}(w) = \min\{5,3\}=3$$



Weighted Automata – language

n A **run** of \mathcal{A} on a word $w = w_1 \dots w_n$ is a sequence $r = r_0 r_1 r_2 \dots r_n$ over Q such that $r_0 \in Q_0$ and for all $1 \leq i \leq n$, we have  .

n A run r is **accepting** $\Leftrightarrow r_n$ is accepting. 
(standard finite-word accepting condition)

n $L(\mathcal{A}) = \{w : \mathcal{A} \text{ has an accepting run on } w\}$



Weighted Automata – costs

n A cost of a run $r = r_0 r_1 r_2 \dots r_n$ is

$$\text{cost}(r) = \sum_{i=1}^n c(\text{state } r_{i-1} \xrightarrow{w_i} \text{state } r_i) + f(\text{state } r_n)$$

n defined only for accepting runs

n A cost of a word $w = w_1 \dots w_n$ is

$$\text{cost}(w) = \min_{\text{accepting runs } r \text{ of } \mathcal{A} \text{ on } w} \text{cost}(r)$$

n If $w \notin L(\mathcal{A})$ then $\text{cost}(w) = \infty$.



Weighted Automata – more

- n A WFA \mathcal{A} is **trim** if each of its states is reachable from some initial state, and has a reachable accepting state.
- n A WFA \mathcal{A} is **unambiguous (single-run)** if it has at most one accepting run on every word.

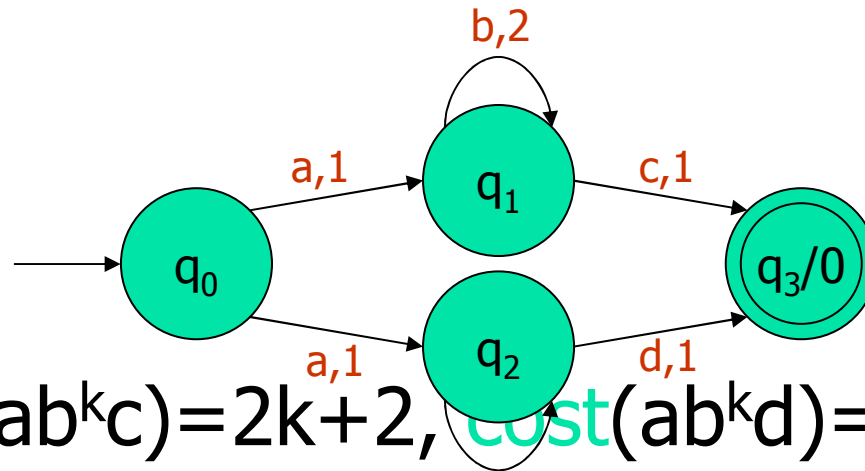


Applications of WFA

- n formal verification of quantitative properties
- n automatic speech recognition
- n image compression
- n pattern matching (widely used in computational biology)
- n ...

\mathcal{A}_1 is non-determinizable

\mathcal{A}_1 :



$\text{cost}(ab^k c) = 2k + 2, \text{cost}(ab^k d) = k + 2$

After reading the word ab^k , the difference between the costs of reading c and d is k .

For $i \neq j$, a deterministic WFA must be in different states after reading ab^i and ab^j .

A deterministic WFA must have ∞ states.



Determinizability

- n Weighted automata are not necessarily determinizable.
- n To decide whether a given weighted automaton is determinizable is an **open question**.
- n A sufficient condition for determinizability + algorithm [Mohri '97].

A sufficient condition [Mohri '97]

n The twins property:

For every two states $q, q' \in Q$,
and two words $u, v \in \Sigma^*$,
if $q, q' \in \delta(Q_0, u)$, $q \in \delta(q, v)$, and
then $\text{cost}(q, v, q) = \text{cost}(q', v, q)$ and $q' \in \delta(q', v)$,

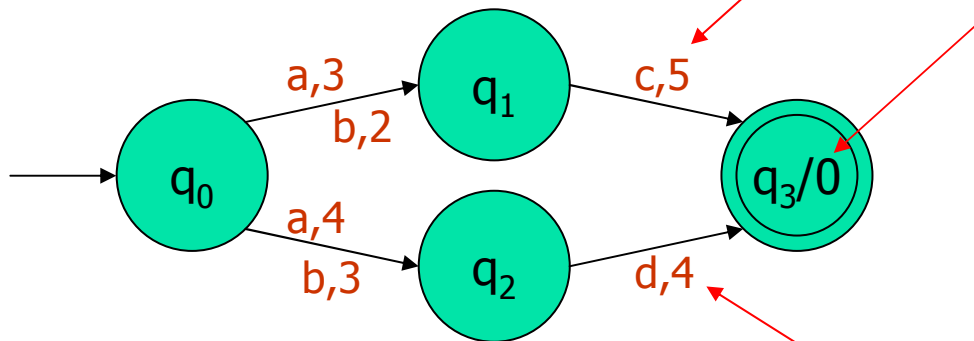
n In case the automaton is trim (no empty states) and unambiguous (single-run),
the twins property is a characterization.



Determinization algorithm

[Mohri '97] - example

\mathcal{A}_2 :



word / cost

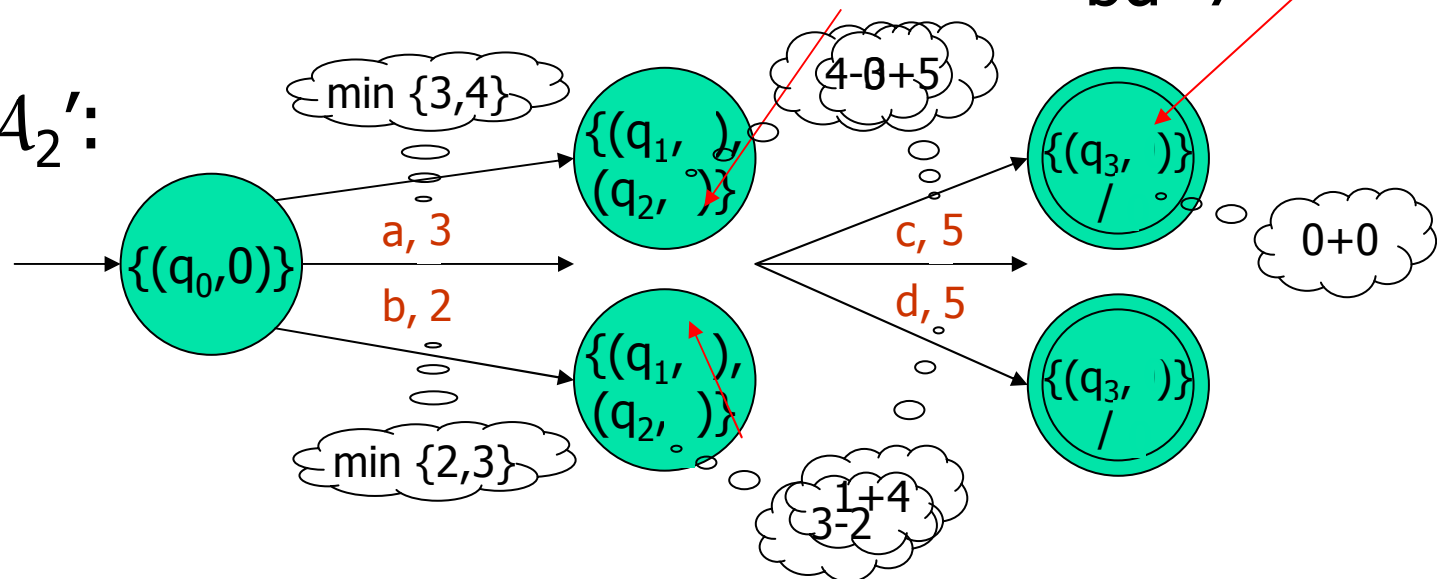
ac 8

bc 7

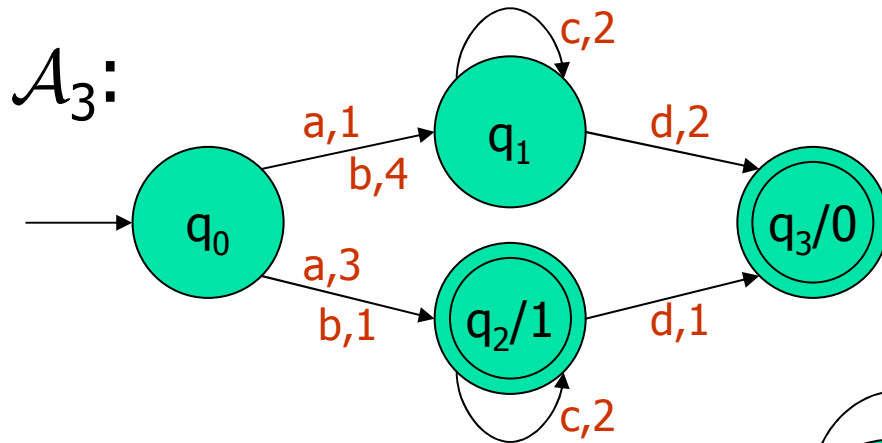
ad 8

bd 7

\mathcal{A}_2' :



Determinization algorithm - another example



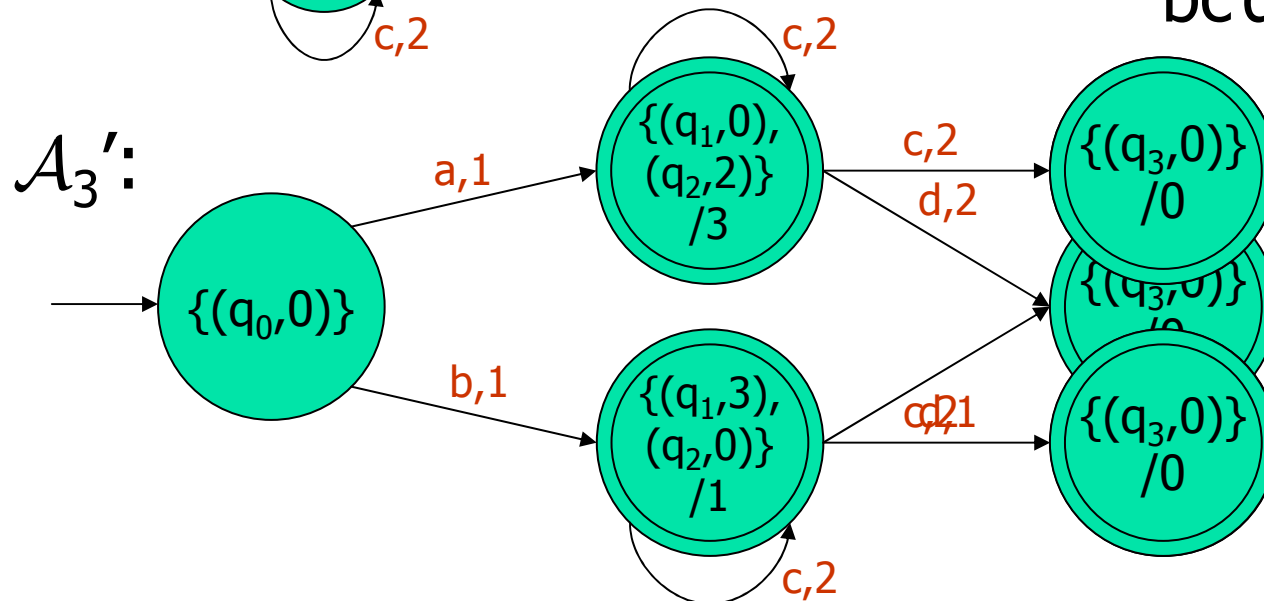
word / cost

ac^i $3+2i+1$

bc^i $2+2i$

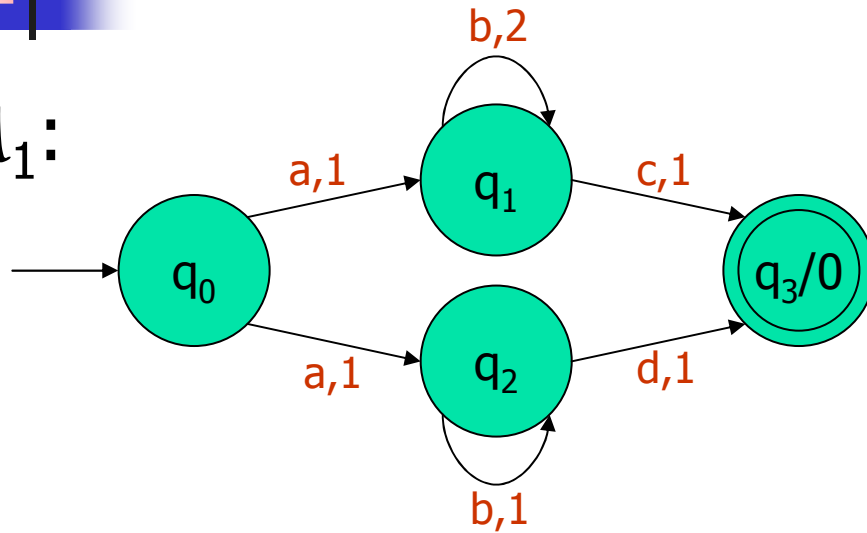
ac^id $3+2i$

bc^id $2+2i$



Determinization algorithm - non-determinizable example

\mathcal{A}_1 :

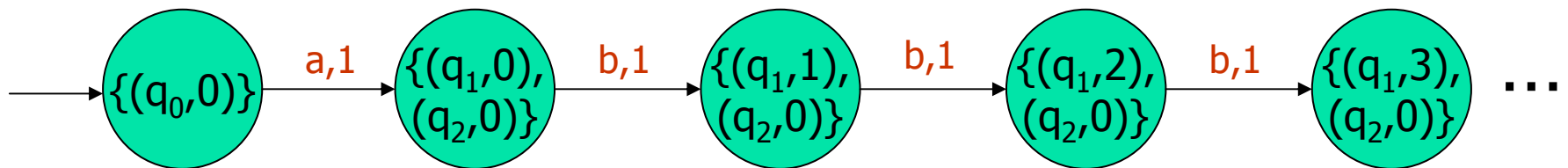


word / cost

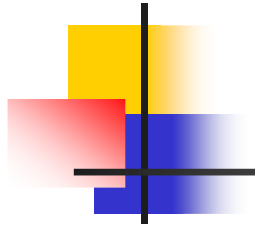
abⁱc 2+2i

abⁱd 2+i

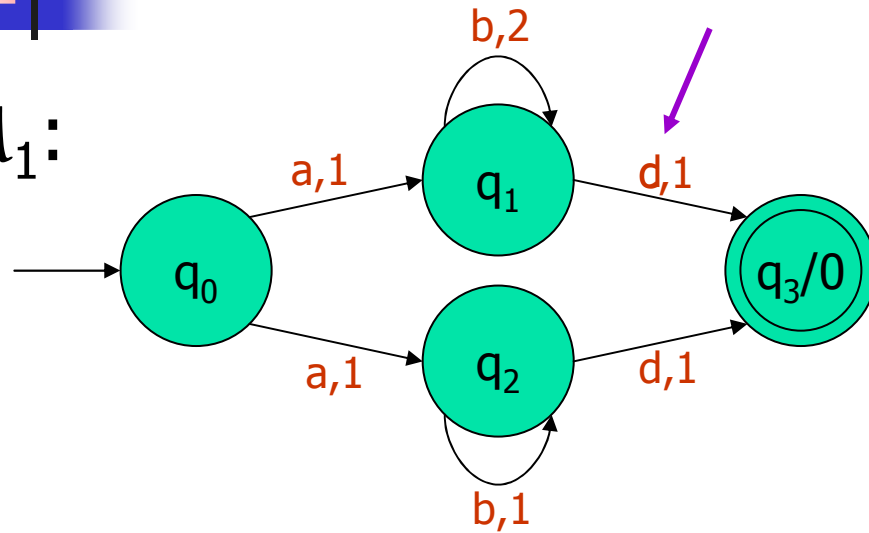
\mathcal{A}_1' :



Determinization algorithm - a bad determinizable example



\mathcal{A}_1 :

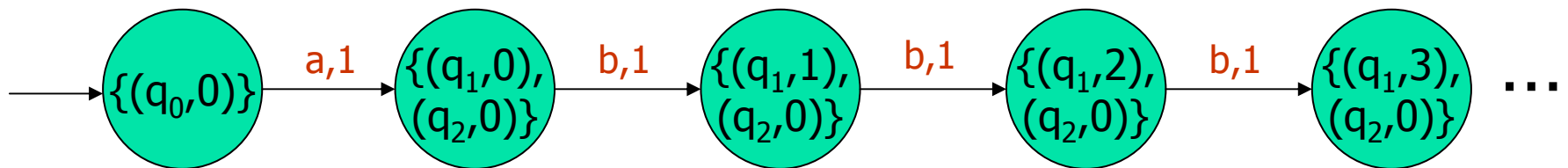


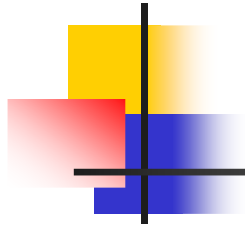
word / cost

~~abⁱc 2+2i~~

abⁱd 2+i

\mathcal{A}_1' :





Mohri's algorithm - remarks

- n Mohri's algorithm terminates iff the original automaton has the twins property.
- n For trim and unambiguous WFAs, there is a polynomial algorithm for testing the twins property.
- n There are determinizable WFAs that do not satisfy the twins property.



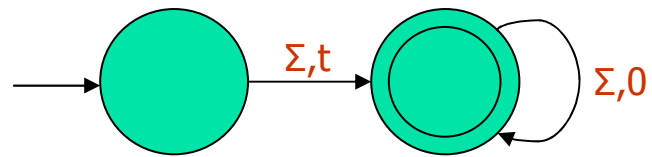
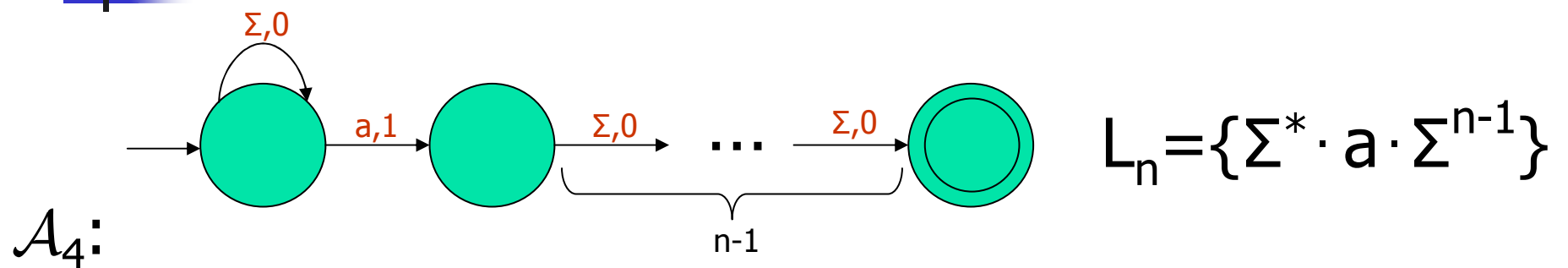
Approximated determinization

Given a WFA \mathcal{A} and an approximation factor $t \geq 1$, construct a deterministic WFA \mathcal{A}' , such that for every word w we have

$$\text{cost}(\mathcal{A}, w) \leq \text{cost}(\mathcal{A}', w) \leq t \cdot \text{cost}(\mathcal{A}, w).$$

- n When exact determinization is impossible.
- n When the result of exact determinization is too large.

Succinctness



$$L(\mathcal{A}_4) = \Sigma^+$$

A deterministic equivalent
requires 2^n states

A t -approximate
deterministic?

2 states

$$\text{cost}(w) = \begin{cases} \infty & w = \varepsilon \\ t & w \in L_n \\ 1 & w \in \Sigma^+ \setminus L_n \end{cases}$$



Approx. determinization algorithm

[Buchsbaum-Giancarlo-Westbrook '01]

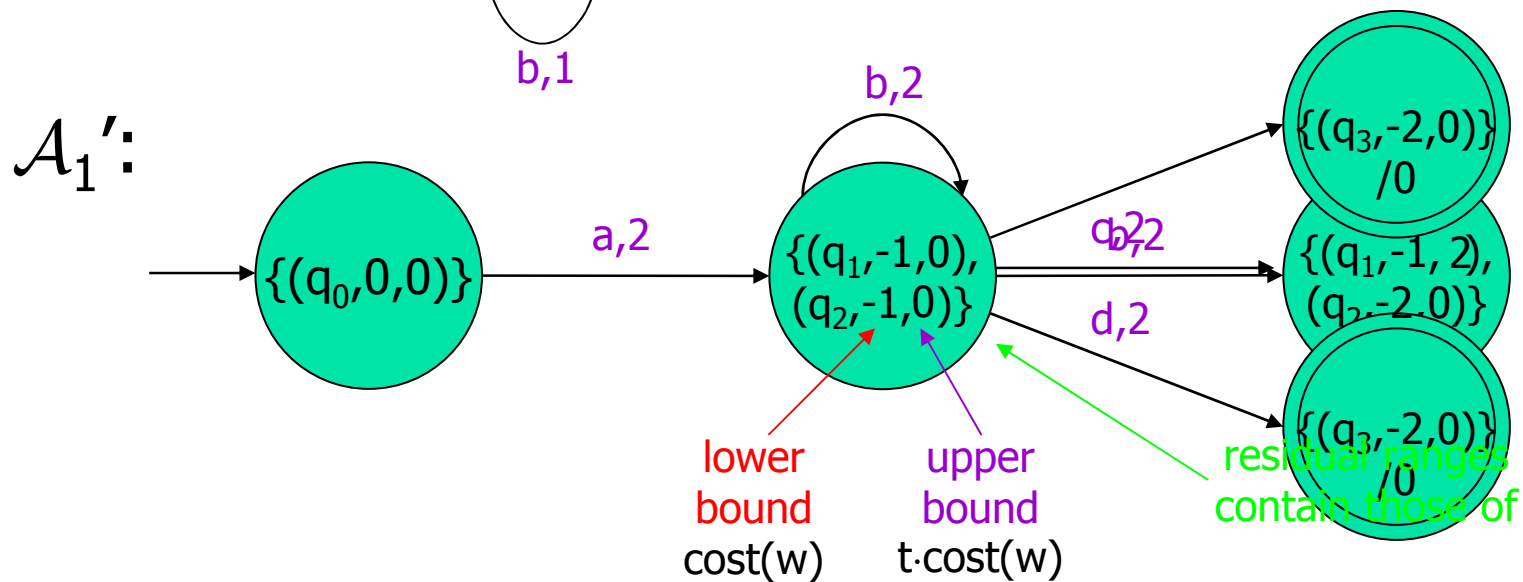
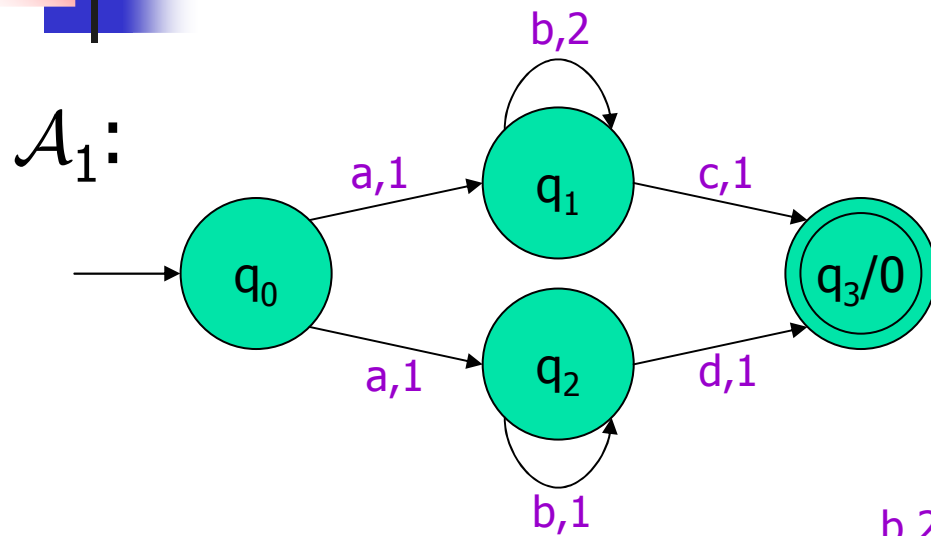
- n Based on Mohri's algorithm.
- n Relaxes the condition for unification of states – rather than requiring residuals of corresponding states to be identical, requires them to be close (within $1+\epsilon$ of the smaller one).
- n No guarantees about the new costs.
- n No sufficient condition for termination.



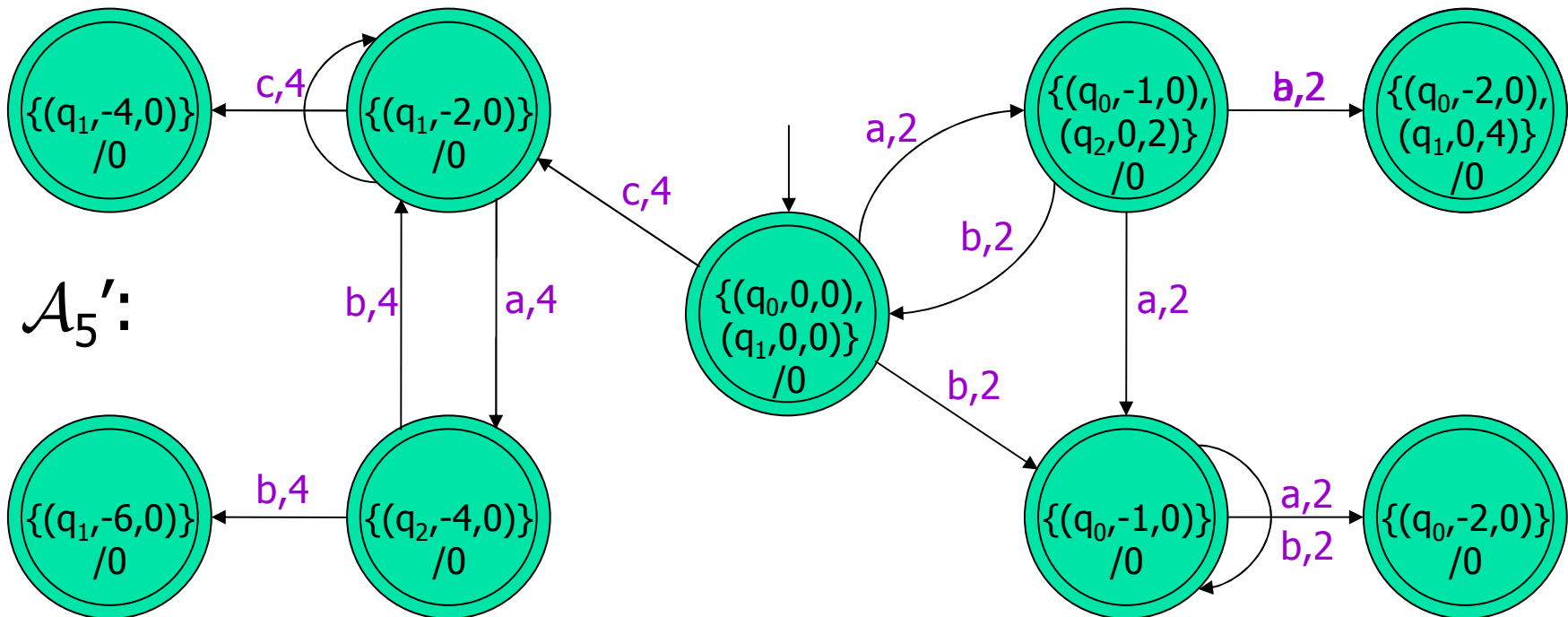
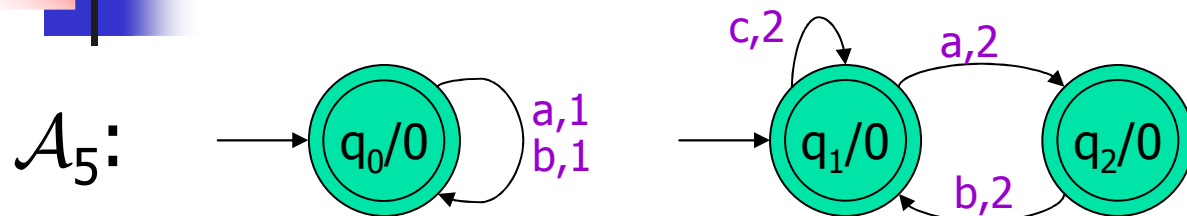
Our algorithm: t-determinization

- n Determinization up to a factor t
 - n The new cost of any accepted word w is between $\text{cost}(w)$ and $t \cdot \text{cost}(w)$.
- n differs from Mohri's algorithm
 - n Weights are multiplied by t .
 - n For each state in a subset we maintain a range of residues rather than one.
 - n The criterion for unification of states is relaxed (they may be non-identical).

2-determinization of \mathcal{A}_1



2-determinization of \mathcal{A}_2





Correctness of the algorithm

- n Thm: If the algorithm terminates on a given WFA \mathcal{A} , with the result \mathcal{A}' , then for every word w we have

$$\text{cost}(\mathcal{A}, w) \leq \text{cost}(\mathcal{A}', w) \leq t \cdot \text{cost}(\mathcal{A}, w).$$



Termination of the algorithm

- n Thm: If a WFA has the t-twins property, then the algorithm terminates on it.
 - n The weights and the factor t are rational.
- n Thm: For trim unambiguous WFAs, a WFA is t -determinizable iff it has the t-twins property.
- n Thm: Deciding the t-twins property for trim unambiguous WFAs can be done in polynomial time.



Summary

- n Why approximate determinization?

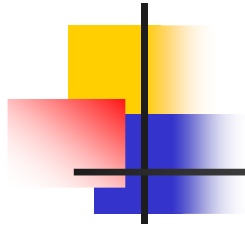
- n Non-determinizable WFA
 - n Equivalent deterministic is large

- n t-determinization algorithm

- n Weights multiplied by t
 - n Use ranges rather than single residues
 - n Collapse to a state whose ranges are contained in mine

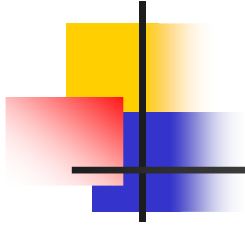
- n A sufficient condition

- n The t-twins property
 - n For unambiguous WFAs – characterizes determinizability
 - n Decidable in polynomial time



Future work

- n Generalize the termination proof to the case where the weights and the factor t are real numbers ($\mathbb{R}^{\geq 0}$).
- n An algorithm to decide whether a WFA is determinizable. Alternatively – prove that it is undecidable.



Thank you!