THE ALMOST EQUIVALENCE BY ASYMPTOTIC PROBABILITIES FOR REGULAR LANGUAGES AND ITS COMPUTATIONAL COMPLEXITIES

Yoshiki Nakamura
Tokyo Institute of Technology
GandALF2016 September 14, 2016

OUR CONTRIBUTIONS

- We new introduced *p-equivalence* ($\simeq_p$)
  - based on asymptotic probabilities, and
  - p-equivalence is one of weak (almost) equivalences.
- p-equivalence and equivalence are the same in terms of the computational complexities.

	Unary alphabet			General case		
	REG	DFA	NFA	REG	DFA	NFA
equivalence	coNP	in L	coNP	PSPACE	NL	PSPACE
p-equivalence	**coNP**	**in L**	**coNP**	**PSPACE**	**NL**	**PSPACE**

MOTIVATION : FINITE MODEL THEORY (OVER **FINITE GRAPHS**)

Let Φ be a first-order sentence. Then,

- The *probabilistic function* μ_n is

$$\mu_n(\Phi) = \frac{\text{the number of finite graphs with } n \text{ nodes that satisfy } \Phi}{\text{the number of finite graphs with } n \text{ nodes}}$$

- The *asymptotic probability* μ is

$$\mu(\Phi) = \lim_{n \to \infty} \mu_n(\Phi)$$

MOTIVATION : FINITE MODEL THEORY

- Φ is *finite almost valid* iff Φ holds over **almost all** finite graphs.

  Formally, "almost all" is defined by that $\mu(\Phi) = 1$.

  Finite almost validity is **decidable** [Fagin 76].

- Φ is *finite valid* iff Φ holds over **all** finite graphs.

  Finite validity is **undecidable** [Trakhtenbrot 50].

Thus, there exists <u>a gap</u> between almost validity and validity.

MOTIVATION

- Finite model theory
  - Finite validity problems are undecidable.
  - Finite **almost** validity problems are decidable.

- Formal language theory
  - Equivalence problems are well-known. (e.g, PSPACE for REGs,…)
  - **Almost** *Equivalence (based on finite model theory) problems* are **???**.

p-equivalence

What are computational complexities of p-equivalence?

NOTATIONS

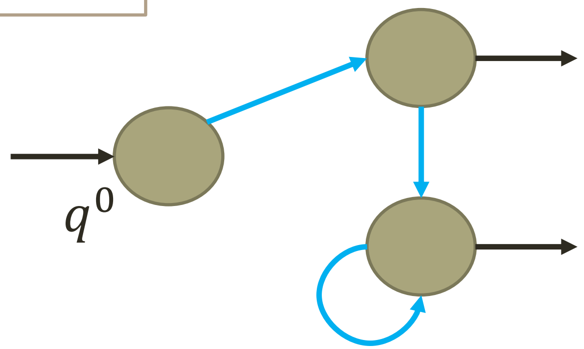
A : a finite alphabet.

A^n : the set of all strings of length n over A.

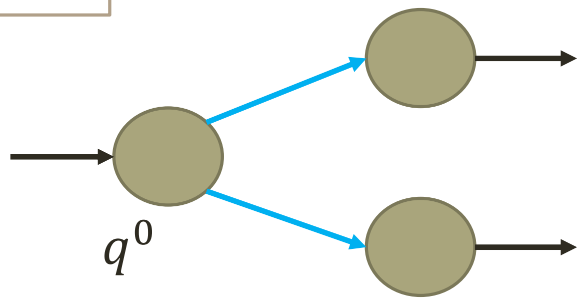
A^* : the set of all strings over A.

REG $\quad \alpha \coloneqq 0 \mid 1 \mid a\ (\in A) \mid \alpha_1 \cdot \alpha_2 \mid \alpha_1 \cup \alpha_2 \mid \alpha^*$

DFA $\quad \mathcal{A} = (Q, A, \delta, q^0, F)$



NFA $\quad \mathcal{A} = (Q, A, \delta, q^0, F)$



P-EQUIVALENCE : ASYMPTOTIC PROBABILITY

- The *probabilistic function* μ_n (over A) is

$$\mu_n(L) = \frac{\text{the number of strings of length } n \text{ in } L}{\text{the number of strings of length } n}$$

$$= \frac{|L \cap A^n|}{|A^n|}.$$

$\mu_n(L)$ is exactly the probability that a random string of length n is in L.

- The *asymptotic probability* $\mu(L)$ (over A) is that

$$\mu(L) = \lim_{n\to\infty} \mu_n(L).$$

P-EQUIVALENCE

- $L_1 \, \Delta \, L_2$ is *the symmetric difference* of L_1 and L_2 :

$$L_1 \, \Delta \, L_2 = (L_1 \setminus L_2) \cup (L_2 \setminus L_1).$$

Then,

- L_1 and L_2 are *<u>p-equivalent</u>* iff $\boldsymbol{\mu(L_1 \, \Delta \, L_2) = 0}$.
  - Intuitively, $L_1 \simeq_p L_2$ means that the difference of L_1 and L_2 converges to "zero" as n approaches infinity.

EXAMPLE : μ

1. Obviously,

   $$\mu(A^*) = \mathbf{1}. \quad \mu(\emptyset) = \mathbf{0}.$$

2. **μ may not exist.**

   $$\mu_n\left(L\big((AA)^*\big)\right) = \begin{cases} \mathbf{1} \ (n \text{ is even}) \\ \mathbf{0} \ (n \text{ is odd}) \end{cases}$$ and $\mu\left(L\big((AA)^*\big)\right)$ <u>does not exist</u>.

3. **Note that μ is depended on A.**

   I. When $A = \{a_1, a_2\}$,

   $$\mu_n\big(L(a_1^*)\big) = \frac{1}{2^n} \text{ and } \mu\big(L(a_1^*)\big) = \mathbf{0}.$$

   II. When $A = \{a_1\}$,

   $$\mu_n\big(L(a_1^*)\big) = 1 \text{ and } \mu\big(L(a_1^*)\big) = \mathbf{1}.$$

EXAMPLE : P-EQUIVALENCE

1. When $A = \{a_1, a_2\}$, $\mu_n\big(\mathrm{L}(A^* \,\Delta\, a_1 A^*)\big) = \dfrac{|a_2 A^{n-1}|}{|A^n|} = \dfrac{1}{2}$.
   $\therefore A^* \simeq_p a_1 A^*$ does **not** hold.

2. When $A = \{a_1, a_2, a_3\}$, $\mu_n\big(\mathrm{L}((a_1 \cup a_2)^* \,\Delta\, 0)\big) = \dfrac{2^n}{3^n} \xrightarrow{n\to\infty} 0$.
   $\therefore (a_1 \cup a_2)^* \simeq_p 0$ **holds.**

3. When $A = \{a_1, a_2\}$, $\mu_n\big(\mathrm{L}((a_1 \cup a_2)^* \,\Delta\, 0)\big) = 1$.
   $\therefore (a_1 \cup a_2)^* \simeq_p 0$ does **not** hold.

A ROBUSTNESS OF P-EQUIVALENCE

$1.\ \mu_n(L) = \frac{|L \cap A^n|}{|A^n|}$ $2.\ \mu_n^*(L) = \frac{\sum_{k=0}^{n-1} |L \cap A^k|}{\sum_{k=0}^{n-1} |A^k|}$ $3.\ \delta_n(L) = \frac{\sum_{k=0}^{n-1} \mu_k(L)}{n}$

(1 was already defined. 2 and 3 are new definitions.)

Theorem The three conditions are <u>equivalent</u> for any **regular languages.**

1. $\lim_{n\to\infty} \mu_n(L_1 \,\Delta\, L_2) = 0.$

2. $\lim_{n\to\infty} \mu_n^*(L_1 \,\Delta\, L_2) = 0.$

3. $\lim_{n\to\infty} \delta_n(L_1 \,\Delta\, L_2) = 0.$

P-EQUIVALENCE : DFA CHARACTERIZATION

$\mathrm{Reach}(q, q')$: q' is reachable from q on $\mathcal{A}$.

Theorem For any DFA $\mathcal{A} = (Q, A, \delta, q^0, F)$, the followings are **equivalent**.

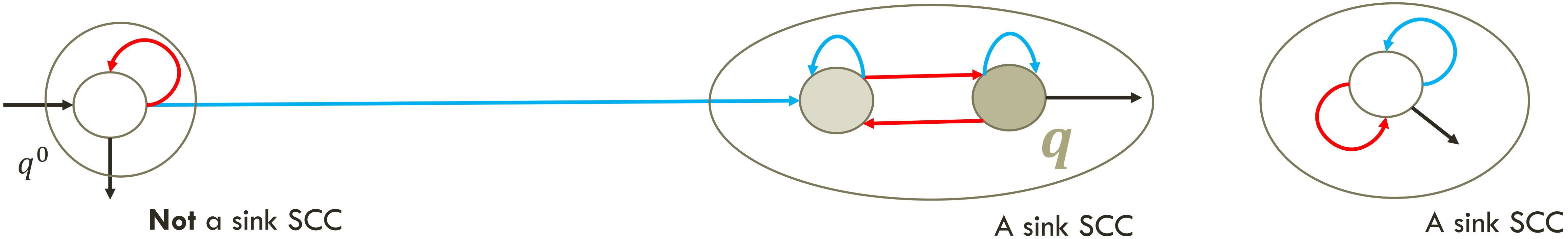
1. $\mu\big(L(\mathcal{A})\big) \neq 0$

2. $\exists q \in F. \Big(\mathrm{Reach}\big(q^0, q\big) \wedge \forall q' \in Q. \big(\mathrm{Reach}(q, q') \to \mathrm{Reach}(q', q)\big)\Big)$

In other words, there exists an acceptance state q s.t.

I. q is reachable from q^0, and

II. q belongs to <u>a sink SCC</u>.



PROOF SKETCH

1. $\mu(L(\mathcal{A})) \neq 0$
2. $\exists q \in F. \left(\text{Reach}(q^0, q) \wedge \forall q' \in Q. \left(\text{Reach}(q, q') \rightarrow \text{Reach}(q', q)\right)\right)$

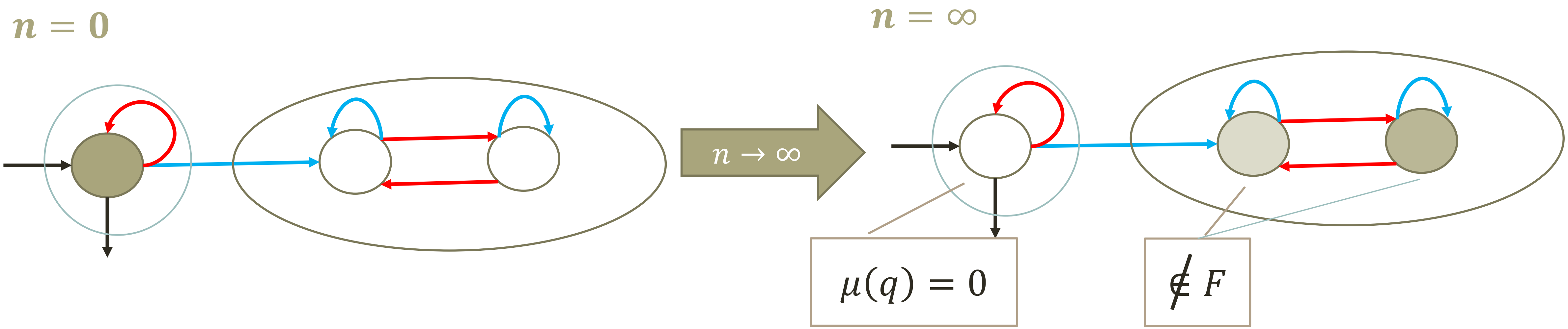
$\neg 2. \Rightarrow \neg 1.$

$\mu_n(q)$: the probability a string s of length n satisfies $\delta(q^0, s) = q$.

I. For any q belonging to a sink SCC, $q \notin F$. ($\because \neg 2.$)

II. For any q **not** belonging to a sink SCC, $\mu(q) = 0$.

So, $\sum_{q \in F} \mu(q) = 0$. $\therefore \mu(L(\mathcal{A})) = 0$.



PROOF SKETCH

1. $\mu(L(\mathcal{A})) \neq 0$
2. $\exists q \in F.\left(\mathrm{Reach}(q^0, q) \wedge \forall q' \in Q.\left(\mathrm{Reach}(q, q') \to \mathrm{Reach}(q', q)\right)\right)$

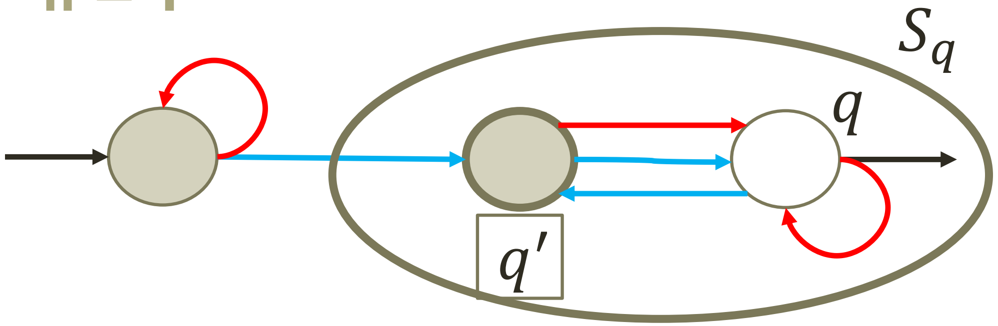
$2. \Rightarrow 1.$

Let S_q be the SCC containing q of 2. .

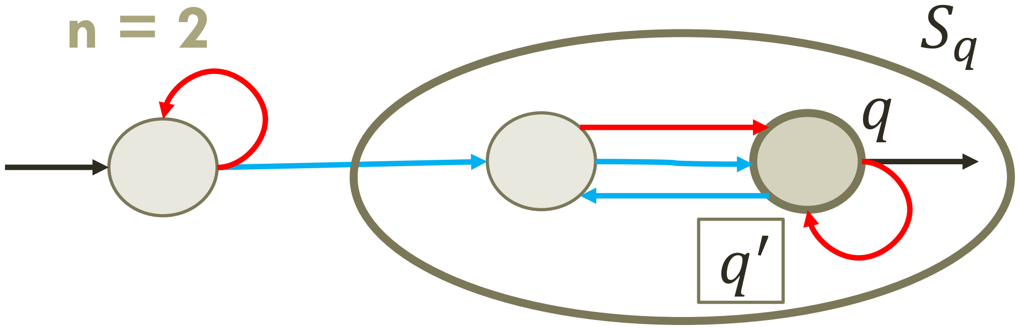
Then, there exists a state $\boldsymbol{q}' \in S_q$ s.t.

$$\mu_n(q') \geq \frac{\mu_n(S_q)}{|S_q|} \ (\because \text{ pigeon hole principle})$$

n = 1



n = 2



PROOF SKETCH

1. $\mu(L(\mathcal{A})) \neq 0$
2. $\exists q \in F. \left(\mathrm{Reach}(q^0, q) \land \forall q' \in Q. \left(\mathrm{Reach}(q, q') \to \mathrm{Reach}(q', q)\right)\right)$

$2. \Rightarrow 1.$

Let $\boldsymbol{k}$ be the distance from q' to q.

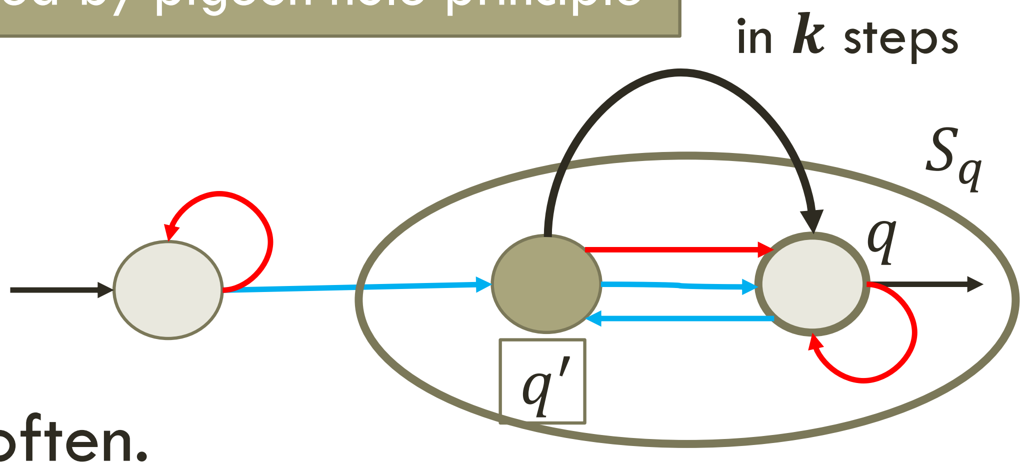
$$\mu_{n+k}(q) \geq \mu_n(q') \times \frac{1}{|A|^k}$$

$$\geq \frac{\mu_n(S_q)}{|S_q|} \times \frac{1}{|A|^k}$$

$\because q'$ is selected by pigeon hole principle

$$\geq \frac{1}{|A|^{|Q|}} \times \frac{1}{|Q|} \times \frac{1}{|A|^{|Q|}} \ (> 0)$$

$\because S_q$ is reachable from q^0



$\therefore \exists M > 0. \mu_n(q) \geq M$ holds <u>infinitely often</u>.

DESCRIPTIVE COMPLEXITY

- The following results in descriptive complexity [Immerman 12] give the computational complexities of p-equivalence.

1. FO(TC) = NL
   - FO(TC) : First Order Logic with **Transitive Closure** function TC
   - TC(R) is the transitive closure of R.
2. FO(DTC) = L
   - DTC : We can only use TC for ***deterministic** relations.*

$(x, y) \in R \land (x, y') \in R \rightarrow y = y'$

3. SO(TC) = PSPACE
   - SO(TC) : Second Order Logic with Transitive Closure function TC

REDUCTION : COMPUTATIONAL COMPLEXITY

1. A DFA is expressible by a first order structure.

DFA → First order structure

2. The characterization is expressible in FO(TC).

- $\mathrm{Reach}(q, q')$ is expressible by using TC.

(restated) $\exists q \in F.\left(\mathrm{Reach}\left(q^0, q\right) \wedge \forall q' \in Q.\left(\mathrm{Reach}(q, q') \rightarrow \mathrm{Reach}(q', q)\right)\right)$

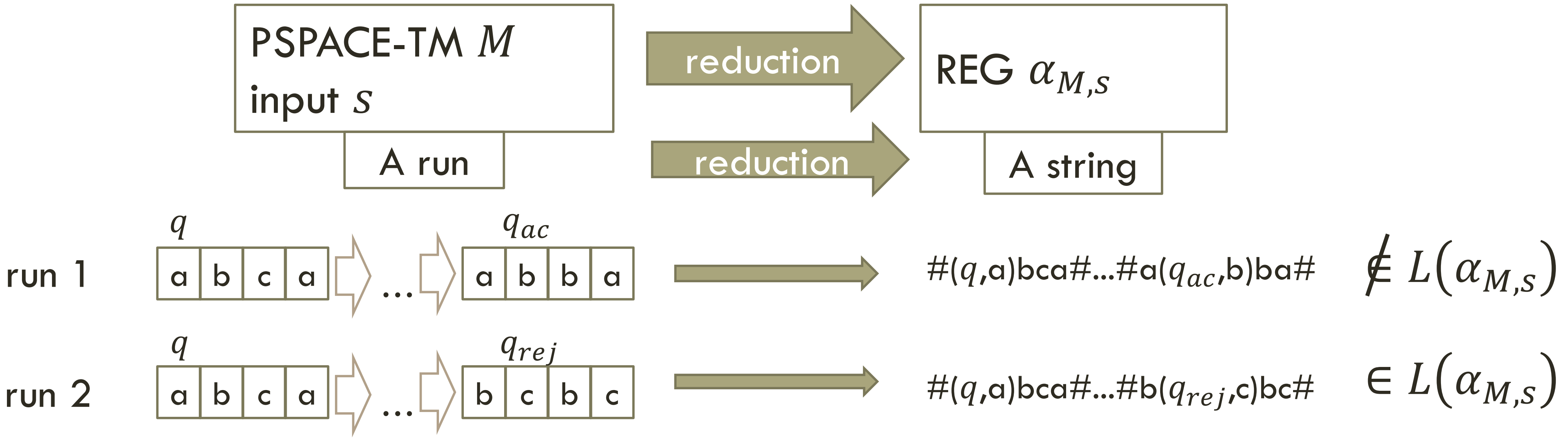
The characterization → First order sentence with TC

∴ the p-equivalence problem for DFAs is in NL.

COMPUTATIONAL HARDNESS

- The hardness is shown by <u>modifying</u> the proofs of the equivalence problems.

cf.) the equivalence problem for REGs is PSPACE-hard [Hunt 76].



M accepts s $\Longleftrightarrow$ $L(\alpha_{M,s}) \neq A^*$

COMPUTATIONAL HARDNESS : A MODIFICATION

[Hunt 76] $\alpha_{M,s} = \alpha_1 \cup \alpha_2 \cup \alpha_3$

- $w \in L(\alpha_1)$ iff the *input string* expressed by w is invalid.
- $w \in L(\alpha_2)$ iff w does not contain *acceptance state*.
- $w \in L(\alpha_3)$ iff the *transitions* expressed by w are invalid.

#(q,a)bca#…#a(q_{ac},b)ba#a#b# $\in L(\alpha_{M,s})$

[N 16] $\alpha'_{M,s} = \alpha_1 \cup \alpha_2 \cup \boldsymbol{\alpha'_3}$

- $w \in L(\alpha'_3)$ iff the transitions expressed by w are invalid, where <u>the invalidity is not checked **after** an acceptance state occurs</u>.

#(q,a)bca#…#a(q_{ac},b)ba#a#b# $\notin L(\alpha'_{M,s})$

COMPUTATIONAL HARDNESS : A MODIFICATION

- [N 16] $\alpha_{M,s} = \alpha_1 \cup \alpha_2 \cup \boldsymbol{\alpha'_3}$
  - $w \in L(\alpha'_3)$ iff the transitions expressed by w are invalid, where <u>the invalidity is not checked **after** an acceptance state occurs</u>.
  - if $w \notin L(\alpha_{M,s})$, then $ww' \notin L(\alpha_{M,s})$ … ☆.

Lemma

$$L(\alpha_{M,s}) \neq A^* \Leftrightarrow L(\alpha_{M,s}) \not\simeq_p A^*$$

(Proof)

$\Leftarrow$ is followed by that $= \subseteq \simeq_p$.

$\Rightarrow$ is <u>newly</u> followed by ☆.

COMPUTATIONAL HARDNESS : OUTLINE

[Hunt 76]

PSPACE-TM M accepts s	reduction $\Longleftrightarrow$	*REG Equivalence* $L(\alpha_{M,s}) \neq A^*$

[N 16]

PSPACE-TM M accepts s	reduction $\Longleftrightarrow$	*REG Equivalence* $L(\alpha'_{M,s}) \neq A^*$
		$\Updownarrow$ **Lemma**
		REG p-Equivalence $L(\alpha'_{M,s}) \neq A^*$

CONCLUSION AND FUTURE WORK

(Conclusion) For **regular languages**, we have shown

- that p-equivalence has some robustness,
- the DFA characterization of p-equivalence, and
- p-equivalence and equivalence are the same in terms of the computational complexities.

(Future Work)

- Finding *some applications* of p-equivalence.
  - e.g., Hyper-minimization based on p-equivalence.
- p-equivalence for non-regular languages, e.g., CFL, DCFL, ….

Thank you for your attention!