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 \( (\sim_p) \)
  - 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.

<table>
<thead>
<tr>
<th></th>
<th>Unary alphabet</th>
<th>General case</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>REG</td>
<td>DFA</td>
</tr>
<tr>
<td>equivalence</td>
<td>coNP</td>
<td>in L</td>
</tr>
<tr>
<td>( p)-equivalence</td>
<td>coNP</td>
<td>in L</td>
</tr>
</tbody>
</table>
MOTIVATION: FINITE MODEL THEORY (OVER FINITE GRAPHS)

Let $\Phi$ be a first-order sentence. Then,

- The probabilistic function $\mu_n$ 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 $\mu$ is
  \[
  \mu(\Phi) = \lim_{n \to \infty} \mu_n(\Phi)
  \]
• 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 a **gap** 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 ???.

What are computational complexities of p-equivalence?
A : a finite alphabet.

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

\( A^* \) : the set of all strings over \( A \).

\[ \alpha := 0 \mid 1 \mid \alpha (\in A) \mid \alpha_1 \cdot \alpha_2 \mid \alpha_1 \cup \alpha_2 \mid \alpha^* \]

REG

DFA \( \mathcal{A} = (Q, A, \delta, q^0, F) \)
P-EQUIVALENCE: ASYMPTOTIC PROBABILITY

• The probabilistic function $\mu_n$ (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 $p$-equivalent iff $\mu(L_1 \Delta L_2) = 0$.
  - 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 : $\mu$

1. Obviously,
   \[ \mu(A^*) = 1. \quad \mu(\emptyset) = 0. \]

2. $\mu$ may not exist.
   \[ \mu_n\left(L((AA)^*)\right) = \begin{cases} 1 & (n \text{ is even}) \\ 0 & (n \text{ is odd}) \end{cases} \quad \text{and } \mu\left(L((AA)^*)\right) \text{ does not exist.} \]

3. Note that $\mu$ is depended on $A$.
   I. When $A = \{a_1, a_2\}$,
      \[ \mu_n(L(a_1^*)) = \frac{1}{2^n} \quad \text{and } \mu(L(a_1^*)) = 0. \]
   II. When $A = \{a_1\}$,
      \[ \mu_n(L(a_1^*)) = 1 \quad \text{and } \mu(L(a_1^*)) = 1. \]
EXAMPLE : P-EQUIVALENCE

1. When \( A = \{a_1, a_2\} \), \( \mu_n(L(A^* \Delta a_1 A^*)) = \frac{|a_2 A^{n-1}|}{|A^n|} = \frac{1}{2} \).
   \[ \therefore A^* \simeq_p a_1 A^* \text{ does not hold.} \]

2. When \( A = \{a_1, a_2, a_3\} \), \( \mu_n(L((a_1 \cup a_2)^* \Delta 0)) = \frac{2^n}{3^n} \xrightarrow{n \to \infty} 0 \).
   \[ \therefore (a_1 \cup a_2)^{\ast} \simeq_p 0 \text{ holds.} \]

3. When \( A = \{a_1, a_2\} \), \( \mu_n(L((a_1 \cup a_2)^* \Delta 0)) = 1 \).
   \[ \therefore (a_1 \cup a_2)^{\ast} \simeq_p 0 \text{ does not hold.} \]
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 equivalent 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**

Reach\((q, q')\) : \(q'\) is reachable from \(q\) on \(A\).

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

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

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

I. \(q\) is reachable from \(q^0\), and

II. \(q\) belongs to a sink SCC.

---

[Diagram showing states and transitions]
PROOF SKETCH

\[ \neg 2. \Rightarrow \neg 1. \]

\( \mu_n(q) \): 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, \( \sum_{q \in F} \mu(q) = 0. \because \mu(L(\mathcal{A})) = 0. \)

\( n = 0 \)

\( n = \infty \)
PROOF SKETCH

2. $\Rightarrow$ 1.

Let $S_q$ be the SCC containing $q$ of 2.

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

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

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

2. $\Rightarrow$ 1.

Let $k$ be the distance from $q'$ to $q$.

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

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

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

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

$\therefore S_q$ is reachable from $q^0$

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

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

$(x, y) \in R \land (x, y') \in R \rightarrow y = y'$
1. A DFA is expressible by a first order structure.

\[ \text{DFA} \quad \rightarrow \quad \text{First order structure} \]

2. The characterization is expressible in $\text{FO(TC)}$.
   \[ \text{Reach}(q, q') \] is expressible by using TC.

\[
\text{(restated) } \exists q \in F. \left( \text{Reach}(q^0, q) \land \forall q' \in Q. \left( \text{Reach}(q, q') \rightarrow \text{Reach}(q', q) \right) \right)
\]

\[ \text{The characterization} \quad \rightarrow \quad \text{First order sentence with TC} \]

\[ \therefore \text{the } p\text{-equivalence problem for DFAs is in } \text{NL}. \]
The hardness is shown by modifying the proofs of the equivalence problems. cf.) the equivalence problem for REGs is PSPACE-hard [Hunt 76].

PSPACE-TM $M$

input $s$

A run

$\alpha_{M,s}$

A string

run 1

$q$

$\underline{a}$ $\underline{b}$ $\underline{c}$ $\underline{a}$ $\ldots$ $\underline{a}$ $\underline{b}$ $\underline{b}$ $\underline{a}$

$\Rightarrow$ $\Rightarrow$

$q_{ac}$

$\underline{a}$ $\underline{b}$ $\underline{b}$ $\underline{a}$

$\Rightarrow$

$\Rightarrow$

$\Rightarrow$

$\Rightarrow$

run 2

$q$

$\underline{a}$ $\underline{b}$ $\underline{c}$ $\underline{a}$ $\ldots$ $\underline{b}$ $\underline{c}$ $\underline{b}$ $\underline{c}$

$\Rightarrow$ $\Rightarrow$

$q_{rej}$

$\underline{b}$ $\underline{c}$ $\underline{b}$ $\underline{c}$

$\Rightarrow$

$\Rightarrow$

$\Rightarrow$

$\Rightarrow$

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

$\#(q,a)\text{bca}#\ldots#(q_{ac,b})\text{ba}#a#b# \in L(\alpha_{M,S})$

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

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

$\#(q,a)\text{bca}#\ldots#(q_{ac,b})\text{ba}#a#b# \not\in L(\alpha'_{M,S})$
COMPUTATIONAL HARDNESS : A MODIFICATION

\[ \alpha_{M,S} = \alpha_1 \cup \alpha_2 \cup \alpha'_{3} \]

- \( w \in L(\alpha'_{3}) \) iff the transitions expressed by \( w \) are invalid, where the invalidity is not checked after an acceptance state occurs.
- if \( w \not\in L(\alpha_{M,S}) \), then \( ww' \not\in L(\alpha_{M,S}) \) ...

**Lemma**

\[
L(\alpha_{M,S}) \neq A^* \iff L(\alpha_{M,S}) \not\equiv_p A^*
\]

(Proof)

\( \iff \) is followed by that \( = \subseteq \approx_p \).

\( \implies \) is newly followed by \( \star \).
COMPUTATIONAL HARDNESS : OUTLINE

[Hunt 76]

\[ \text{PSPACE-TM} \]
\[ M \text{ accepts } s \]

\[ \text{REG Equivalence} \]
\[ L(\alpha_{M,s}) \neq A^* \]

[N 16]

\[ \text{PSPACE-TM} \]
\[ M \text{ accepts } s \]

\[ \text{REG Equivalence} \]
\[ L(\alpha'_{M,s}) \neq A^* \]

\[ \text{REG p-Equivalence} \]
\[ L(\alpha'_{M,s}) \neq A^* \]

\[ \text{Lemma} \]
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!