Learning Markov Models for Stationary System Behaviors

Yingke Chen Hua Mao Manfred Jaeger
Thomas D. Nielsen Kim G. Larsen Brian Nielsen

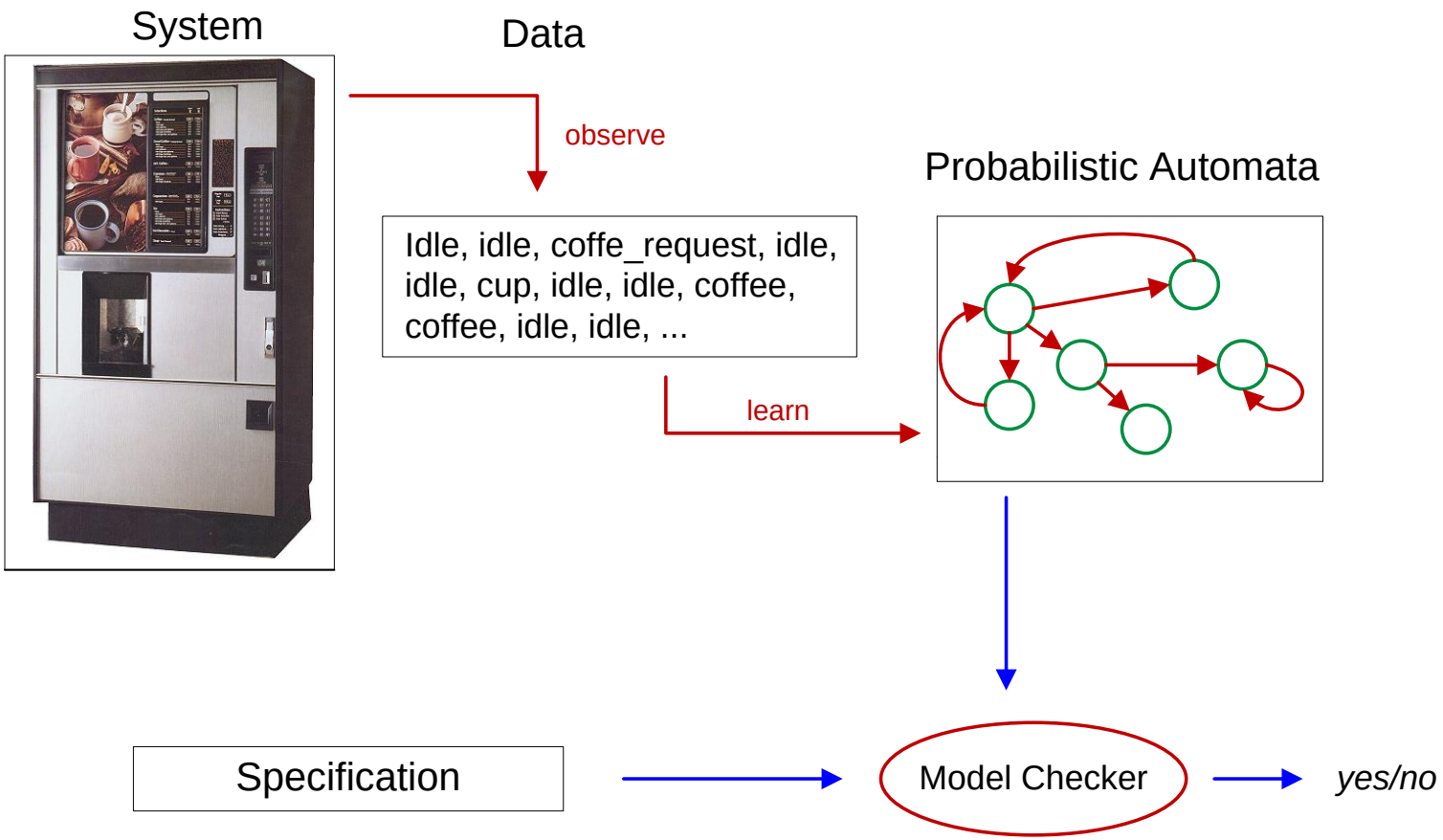
Department of Computer Science,
Aalborg University,
Denmark

NFM 2012
April 4, 2012

Motivation

- Constructing formal models manually can be time consuming
- Formal system models may not exist
  - legacy software
  - 3rd party components
  - black-box embedded system component
- Our proposal: learn models from observed system behaviors

Overview of Our Approach



Related Work

Related Work

- Learning probabilistic finite automata
  - ALERGIA— R. Carrasco and J. Oncina (1994)
  - Probabilistic Suffix Autumata — D. Ron et al. (1996)
- Learning models for model checking
  - Learning CTMCs — K. Sen and et al. (2004)
  - Learning DLMCs — H. Mao and et al. (2011)

Limitation

- Hard to restart the system any number of times.
- Can not reset the system to a well-defined unique initial state.

Proposal

- Learn a model from a single observation sequence

Labeled Markov Chain (LMC)

A LMC is a tuple, $M = \langle Q, \Sigma, \pi, \tau, L \rangle$,

- Q: a finite set of states
- Σ: finite alphabet
- $\pi : Q \rightarrow [0, 1]$ is an *initial probability distribution*
- $\tau : Q \times Q \rightarrow [0, 1]$ is the *transition probability function*
- $L : Q \rightarrow \Sigma$ is a *labeling function*

Probabilistic Suffix Automata - PSA

A PSA is LMC that

- $H : Q \rightarrow \Sigma^{\leq N}$ is a extended *labeling function*, which represents the history of the most recent visited states.
- Each state q_i is associated with a string $s_i = H(q_i)L(q_i)$. If $\tau(q_1, q_2) > 0$, then $H(q_2) \in \mathrm{suffix}^*(s_1)$
- Let S be the set of strings associated with states in the PSA, then $\forall\, s \in S, \mathrm{suffix}^*(s) \cap S = \{s\}$

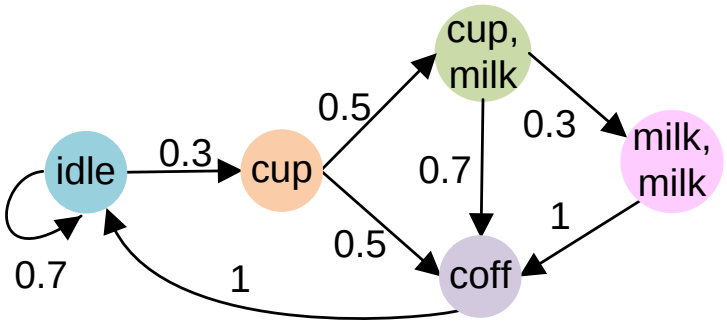


Figure: A PSA over $\Sigma = \{\text{idle, cup, milk, coff}\}$

Prediction Suffix Tree - PST

- A tree over the alphabet $\Sigma = \{\text{idle, cup, milk, coff}\}$
- Each node is labeled by a pair (s, γ_s), and each edge is labeled by a symbol $\sigma \in \Sigma$
- The parent's string is the suffix of its children's

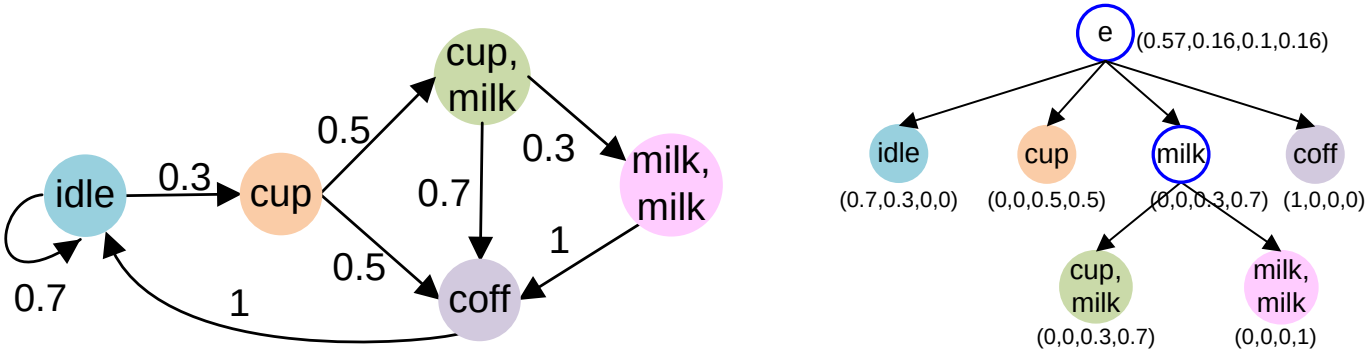


Figure: PSA and PST define the same distribution of strings over Σ

Stationary Probabilistic LTL - SPLTL

Syntax

The syntax of stationary probabilistic LTL is:

$$\phi ::= \mathsf{S}_{\bowtie r}(\varphi) \quad (\bowtie \in \geq, \leq, =;\ r \in [0,1];\ \varphi \in \mathsf{LTL})$$

Semantics

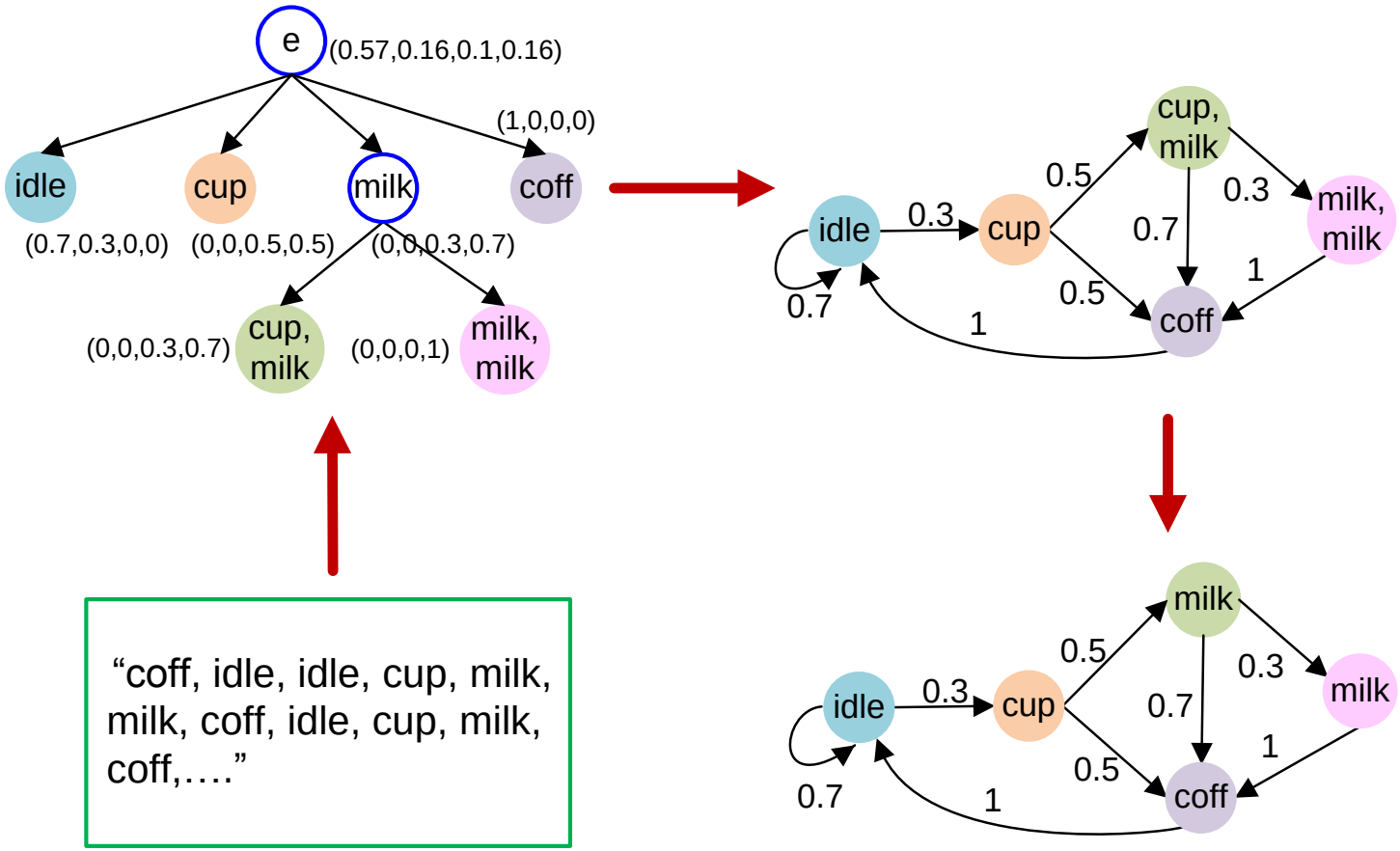
For a model M, the stationary probability of an LTL property φ is

$$M \models \mathsf{S}_{\bowtie r}(\varphi) \text{ iff } P_M^{\pi^s}(\{s \in \Sigma^\omega | s \models \varphi\}) \bowtie r$$

for all stationary distributions π^s.

Outline

Overview

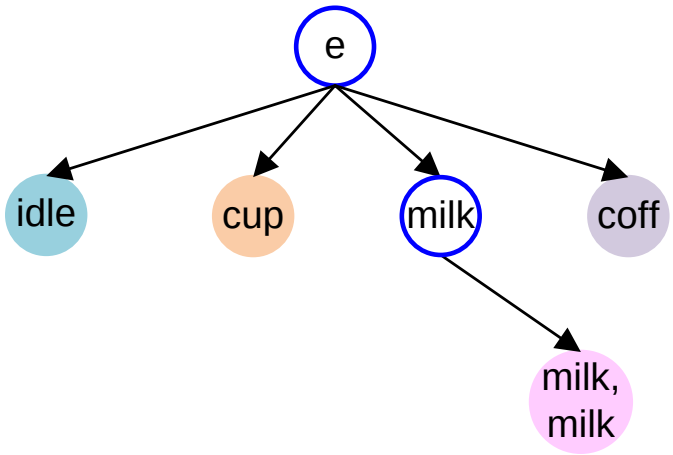


Construct PST

- Start with T, only consisting root node (e), and $S = \{\sigma \mid \sigma \in \Sigma \text{ and } \tilde{P}(\sigma) \geq \epsilon\}$.
- For each $s \in S$, s will be included in T if

$$\tilde{P}(s) \cdot \sum_{\sigma \in \Sigma} \tilde{P}(\sigma|s) \cdot \log \frac{\tilde{P}(\sigma|s)}{\tilde{P}(\sigma|suffix(s))} \geq \epsilon$$

- For each s that $\tilde{P}(s) \geq \epsilon$, for all $\sigma' \in \Sigma$, $\sigma' s$ will be added into S
- Loop until S is empty
- Calculate the next symbol distribution for each node in T

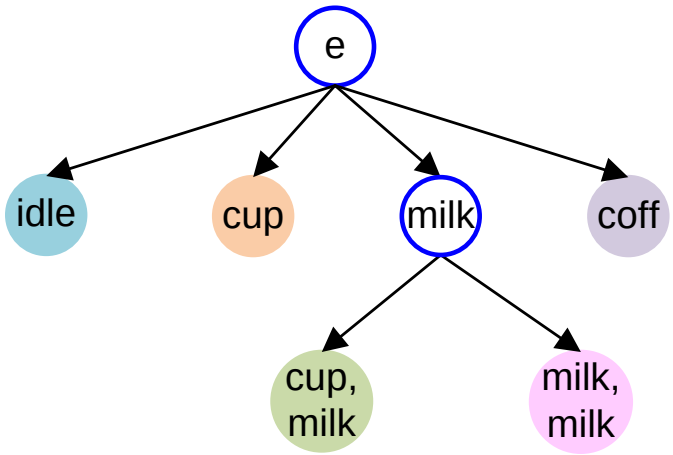


Construct PST

- Start with T, only consisting root node (e), and $S = \{\sigma \mid \sigma \in \Sigma \text{ and } \tilde{P}(\sigma) \geq \epsilon\}$.
- For each $s \in S$, s will be included in T if

$$\tilde{P}(s) \cdot \sum_{\sigma \in \Sigma} \tilde{P}(\sigma|s) \cdot \log \frac{\tilde{P}(\sigma|s)}{\tilde{P}(\sigma|suffix(s))} \geq \epsilon$$

- For each s that $\tilde{P}(s) \geq \epsilon$, for all $\sigma' \in \Sigma$, $\sigma' s$ will be added into S
- Loop until S is empty
- Calculate the next symbol distribution for each node in T

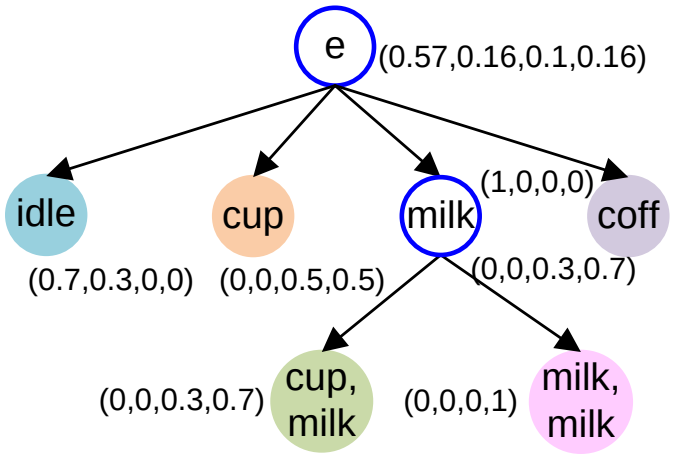


Construct PST

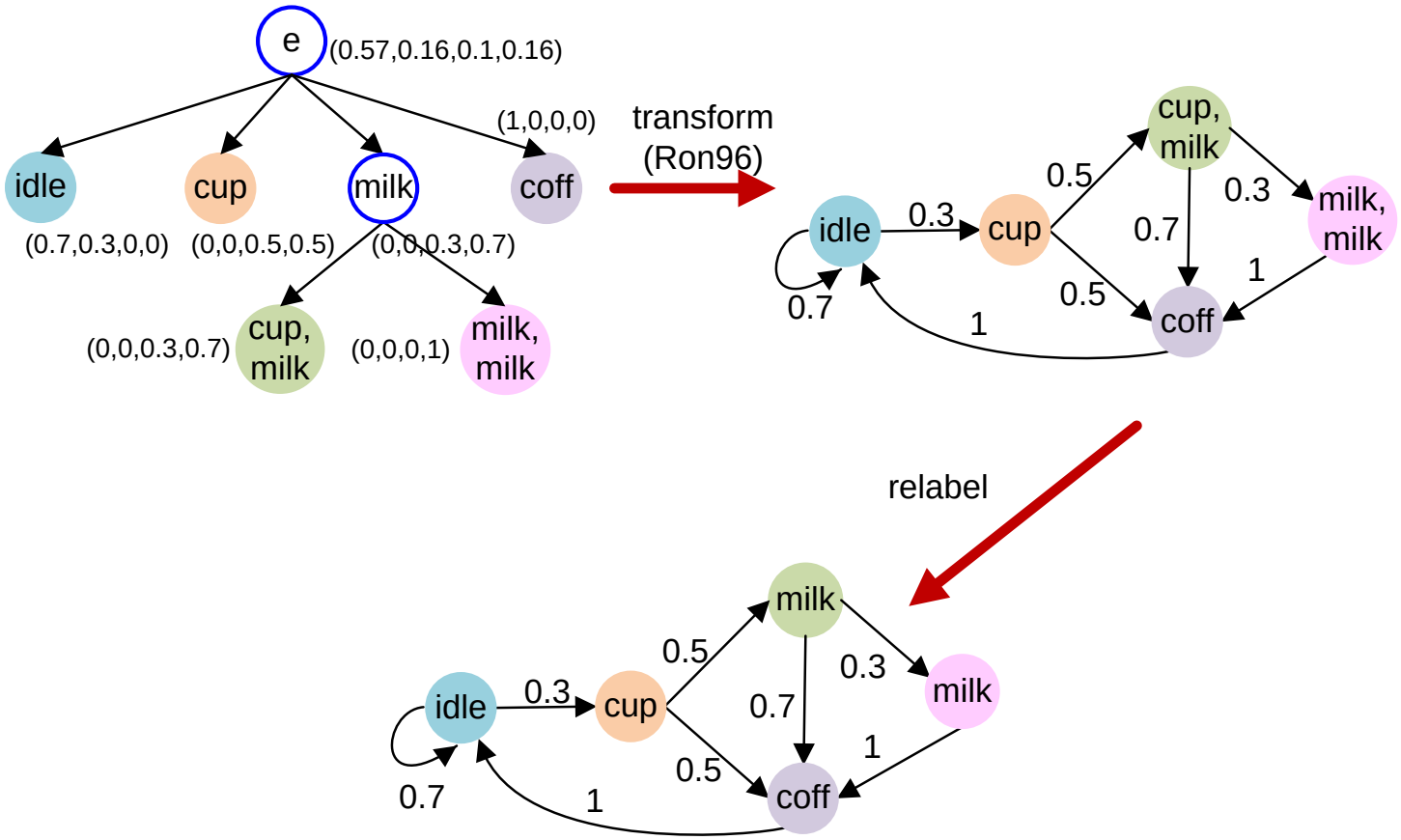
- Start with T, only consisting root node (e), and $S = \{\sigma \mid \sigma \in \Sigma \text{ and } \tilde{P}(\sigma) \geq \epsilon\}$.
- For each $s \in S$, s will be included in T if

$$\tilde{P}(s) \cdot \sum_{\sigma \in \Sigma} \tilde{P}(\sigma|s) \cdot \log \frac{\tilde{P}(\sigma|s)}{\tilde{P}(\sigma|suffix(s))} \geq \epsilon$$

- For each s that $\tilde{P}(s) \geq \epsilon$, for all $\sigma' \in \Sigma$, $\sigma' s$ will be added into S
- Loop until S is empty
- Calculate the next symbol distribution for each node in T



Transform the PST to the LMC



Parameter Tunning

Smaller ϵ induces bigger model

- $\tilde{P}(s) \cdot \sum_{\sigma \in \Sigma} \tilde{P}(\sigma|s) \cdot \log \frac{\tilde{P}(\sigma|s)}{\tilde{P}(\sigma|suffix(s))} \geq \epsilon$
- $\tilde{P}(s) \geq \epsilon$
- Overfitting;

Parameter Tunning

Smaller ϵ induces bigger model

- $\tilde{P}(s) \cdot \sum_{\sigma \in \Sigma} \tilde{P}(\sigma|s) \cdot \log \frac{\tilde{P}(\sigma|s)}{\tilde{P}(\sigma|suffix(s))} \geq \epsilon$
- $\tilde{P}(s) \geq \epsilon$
- Overfitting;

Bayesian Information Criterion - (BIC)

- $BIC(A \mid Seq) := log(L(A \mid Seq)) - 1/2 \, |A| \, log(|Seq|)$

Here, $|A| = |Q_A| \cdot (|\Sigma| - 1)$

Outline

Experiments Setting

- A single sequence is generated by a given LMC model
- The difference between the generating model M_g and the learned model M_l is measured as the mean absolute difference D in stationary probability over a set Φ of randomly generated LTL formula (Computed by PRISM)

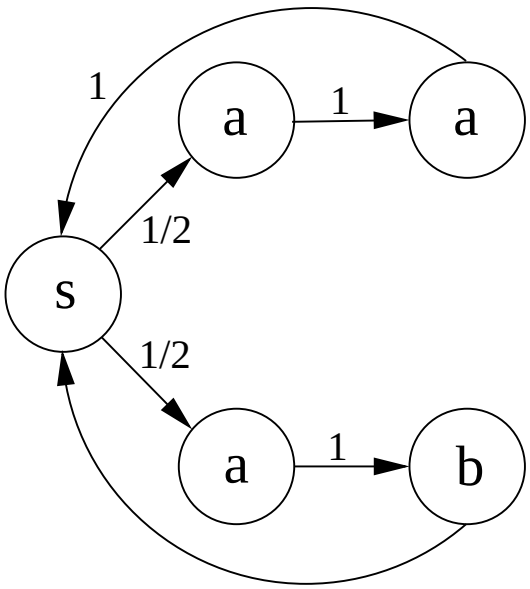
$$D = \frac{1}{|\Phi|} \sum_{\phi \in \Phi} |P^s_{M_g}(\phi) - P^s_{M_l}(\phi)|$$

- PSA-equivalent
- Non PSA-equivalent

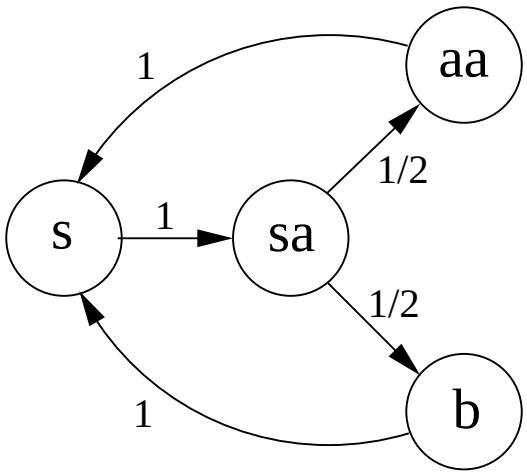
PSA-equivalent

An LMC M is called *PSA-equivalent* if there exists a PSA M', such that for every string s,

$$P_M(s) = P_{M'}(s)$$



(a)



(b)

Phone Model

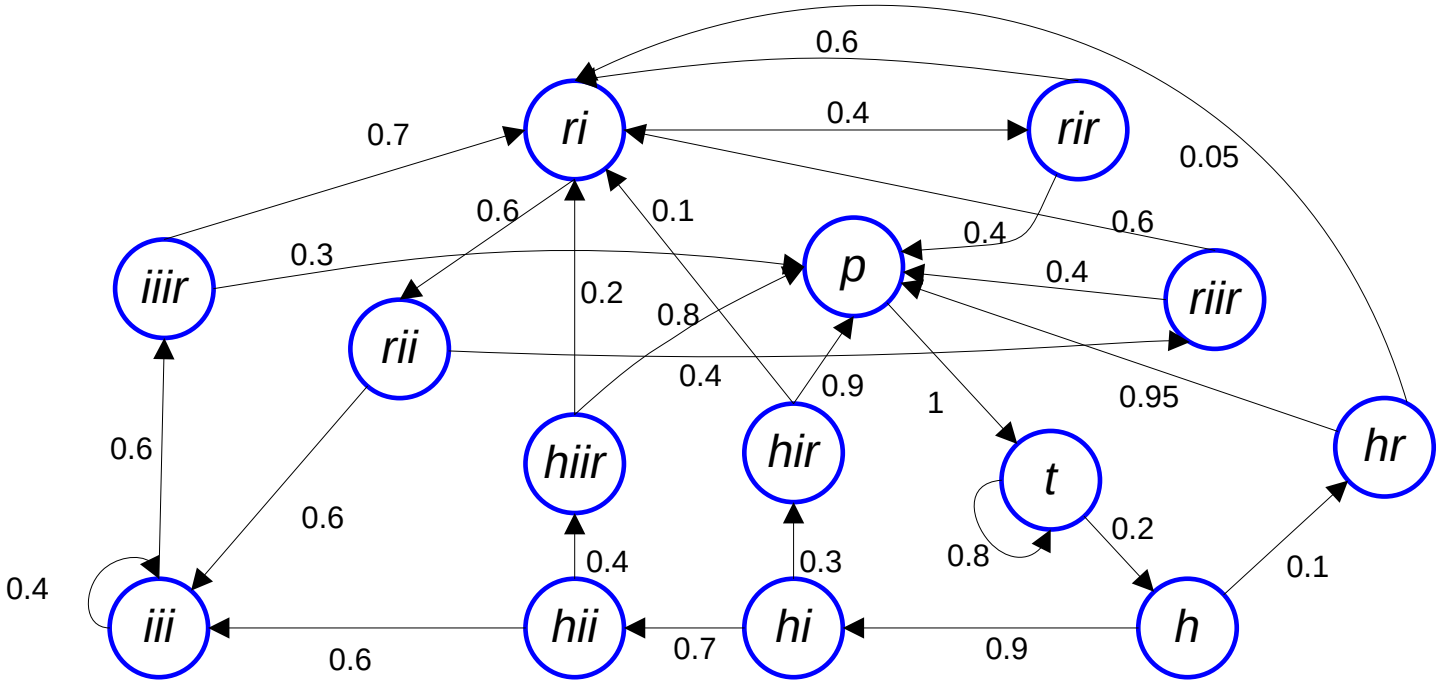


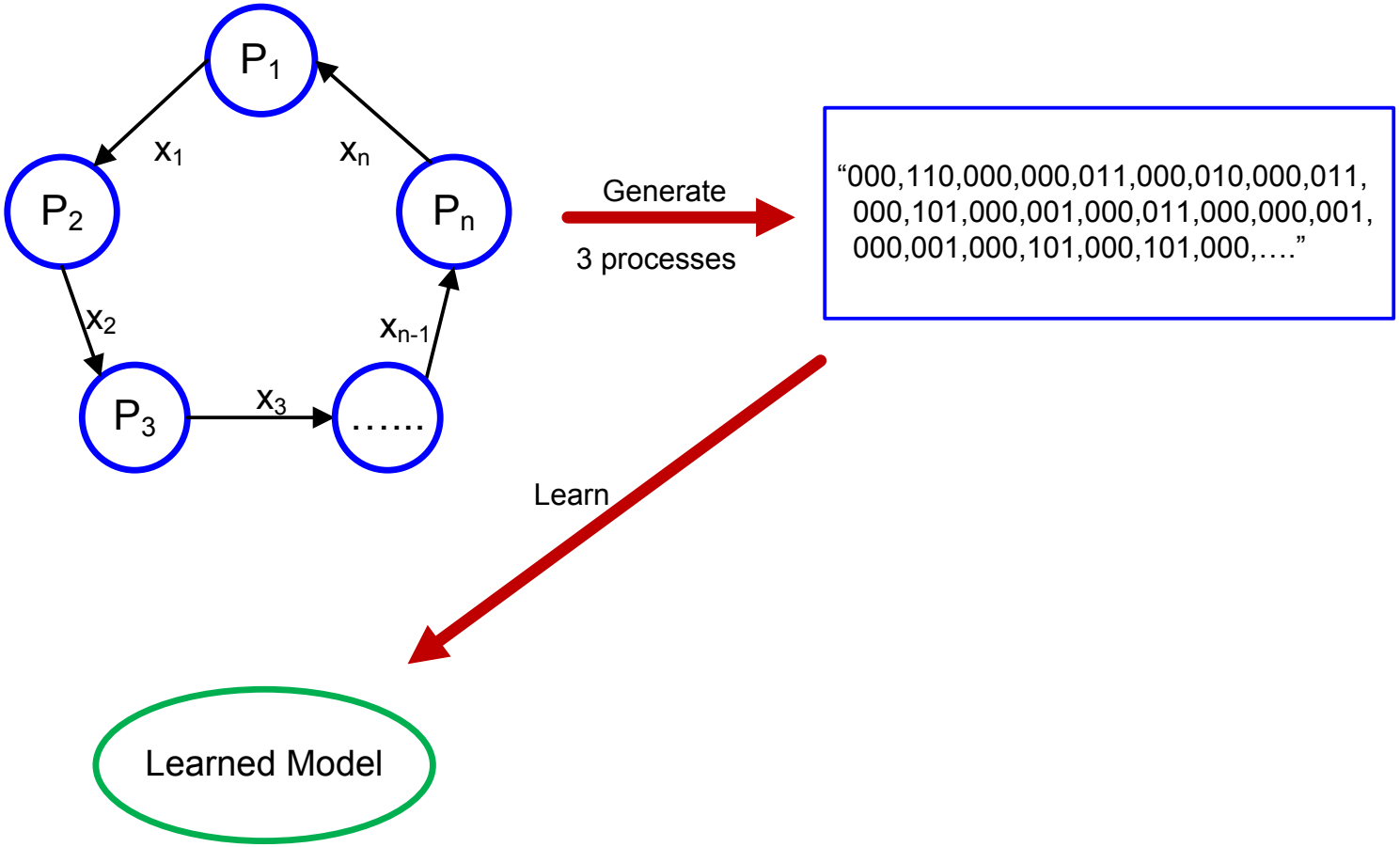
Figure: $\Sigma = \{$(r)ing, (i)dle, (t)alk, (p)ick-up, (h)ang-up$\}$

Phone Model *cont.*

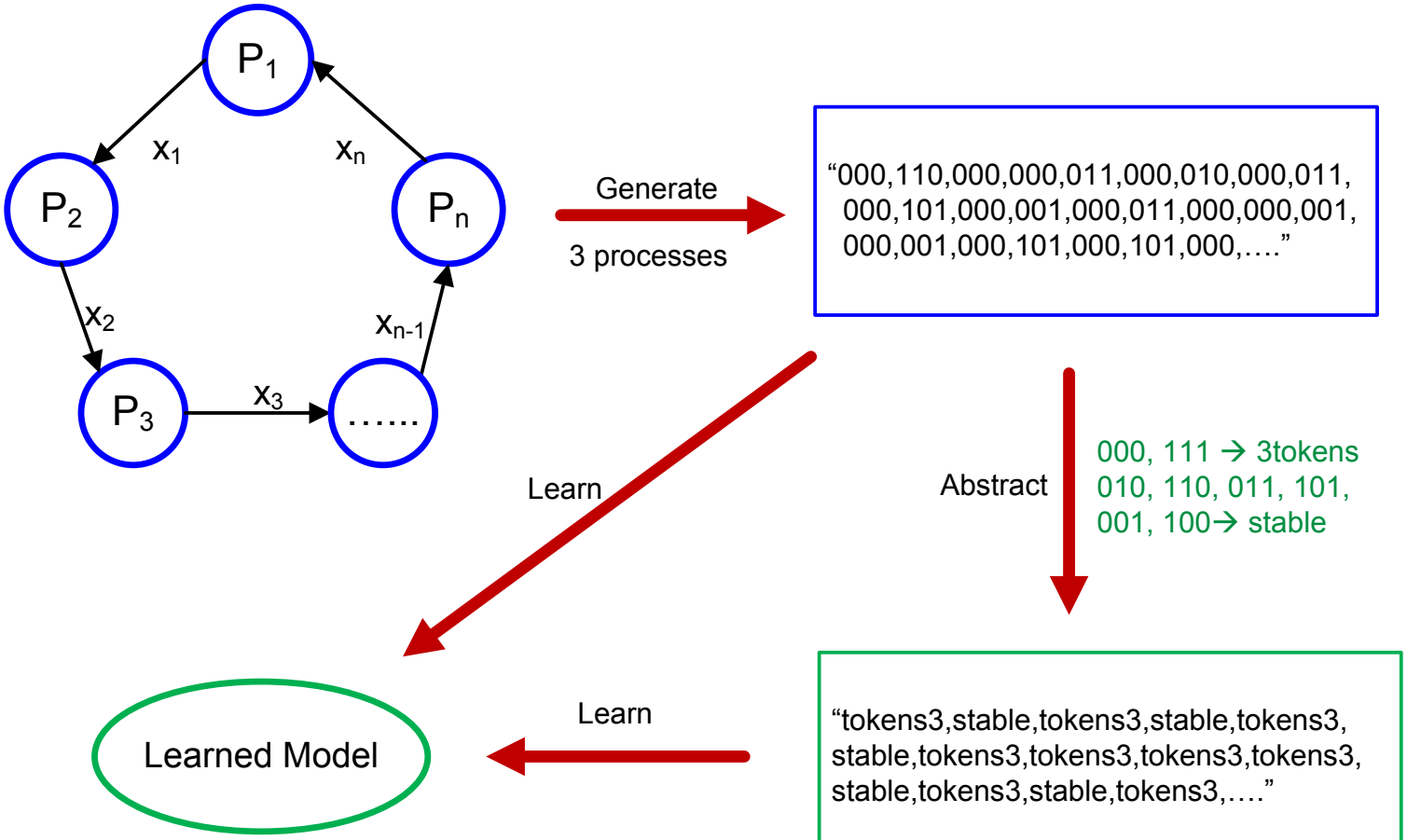
Table: D is based on 507 random LTL formulas. For reference: $D_{dummy} = 0.1569$

$\|S\|$	$\|Q_l\|$	D	t	$rp\|r$	$irp\|ir$	$iirp\|iir$	$\Diamond\Box i$
320	5	0.03200	0.344	0.310	0.309	0.309	0
1280	5	0.04900	0.385	0.446	0.446	0.446	0
5120	10	0.00590	0.379	0.490	0.490	0.490	0
10240	14	0.00160	0.381	0.506	0.477	0.409	0
20480	14	0.00049	0.378	0.515	0.489	0.414	0
M_g	14		0.378	0.512	0.488	0.424	0

Self-stabilizing Protocol



Self-stabilizing Protocol



Self-stabilizing Protocol *cont.*

Table: Self-stabilizing protocol with 7 processes. D is based on 503 random LTL formulas. For reference: $D_d = 0.1669$.

	Full model				Abstract model			
$\|Seq\|$	time(sec)	order	$\|Q_l\|$	D	time(sec)	order	$\|Q_l\|$	D
80	73.0	0	1	0.0192	1.6	1	4	0.0172
160	49.4	0	1	0.0325	2.1	1	4	0.0079
320	162.9	0	1	0.0292	3.3	1	4	0.0369
640	34.3	0	1	0.0234	2.3	1	4	0.0114
1280	37.2	0	1	0.0193	4.1	1	4	0.0093
2560	42.0	0	1	0.0204	5.0	1	4	0.0054
5120	47.9	0	1	0.0182	8.9	1	4	0.0018
10240	59.3	0	1	0.0390	16.3	1	4	0.0013
20480	80.7	0	1	0.0390	31.4	1	4	0.0016
50000	1904.4	1	128	0.00034	152.42	1	4	0.0011
100k	3435.5	1	128	0.00071	308.9	1	4	0.0007

Self-stabilizing Protocol *cont.*

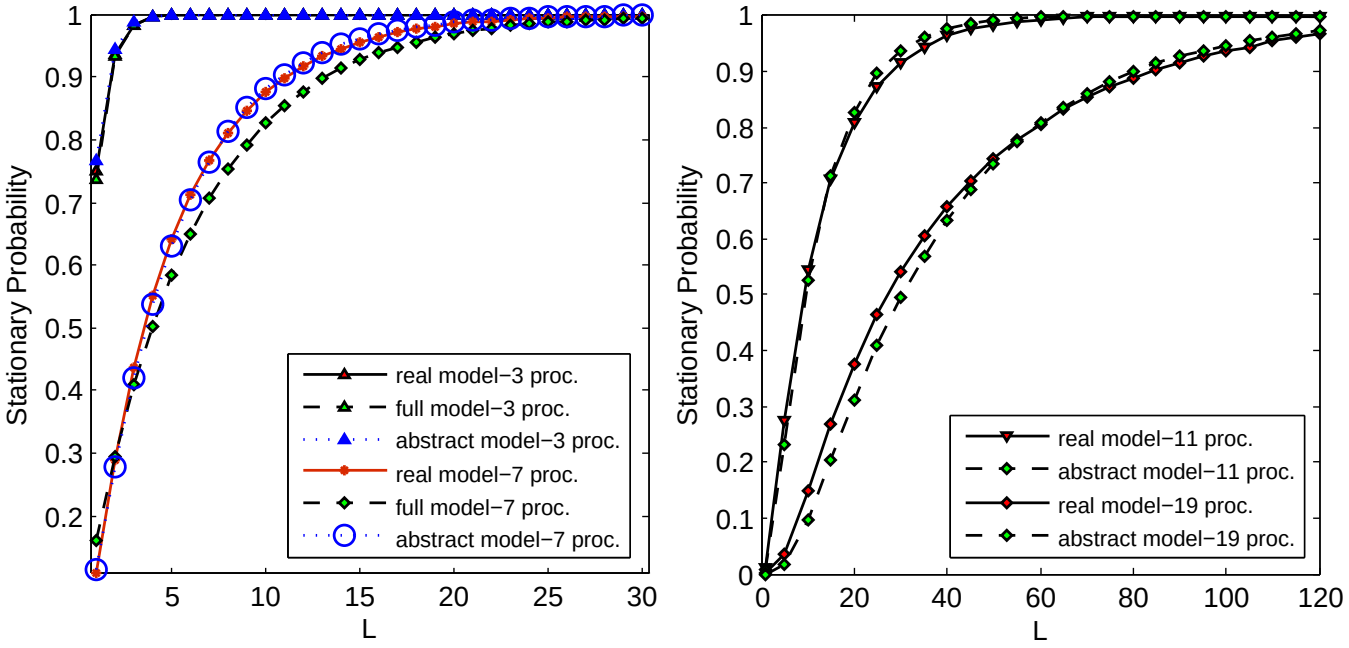


Figure: $P^s_M(true \mathsf{U}_{\leq L}\ stable \mid token = N)$

Self-stabilizing Protocol *cont.*

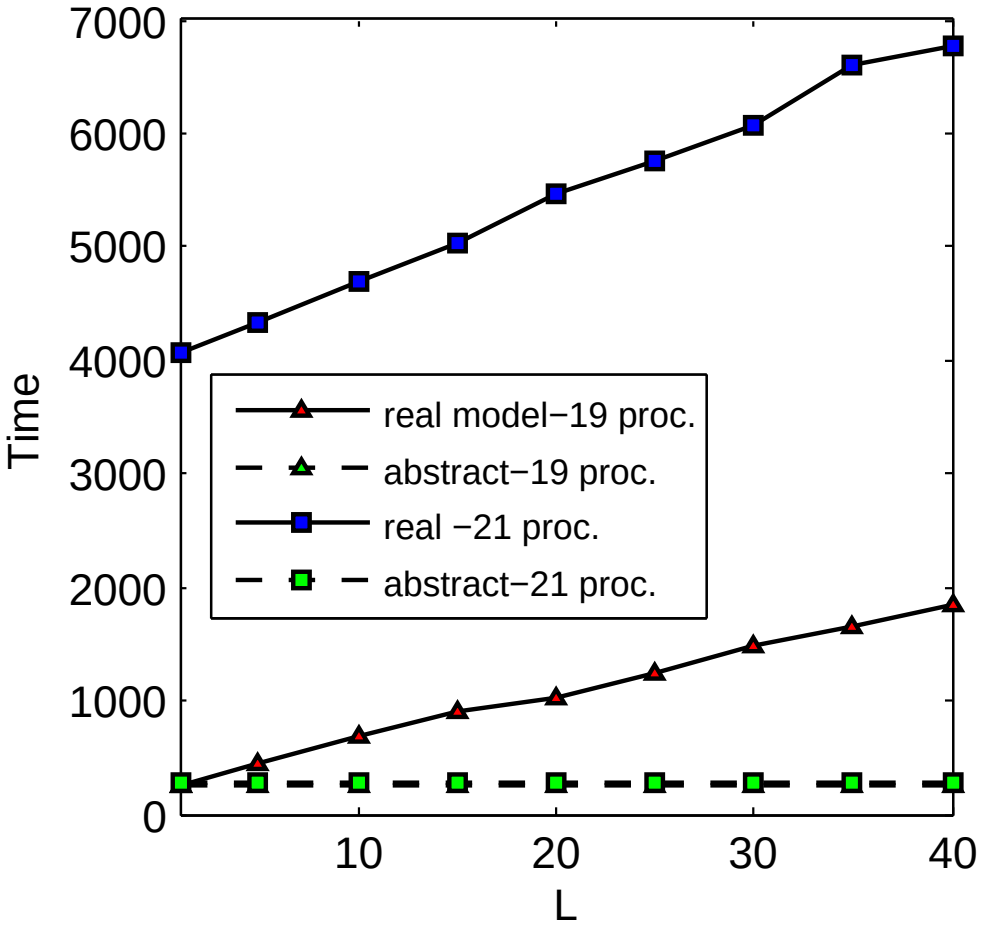


Figure: The time for calculating $P^s_M(true\mathrm{U}_{\leq L}\ stable \mid token = N)$ in the generating model and the abstract model.

Non PSA-equivalent

Dice Model

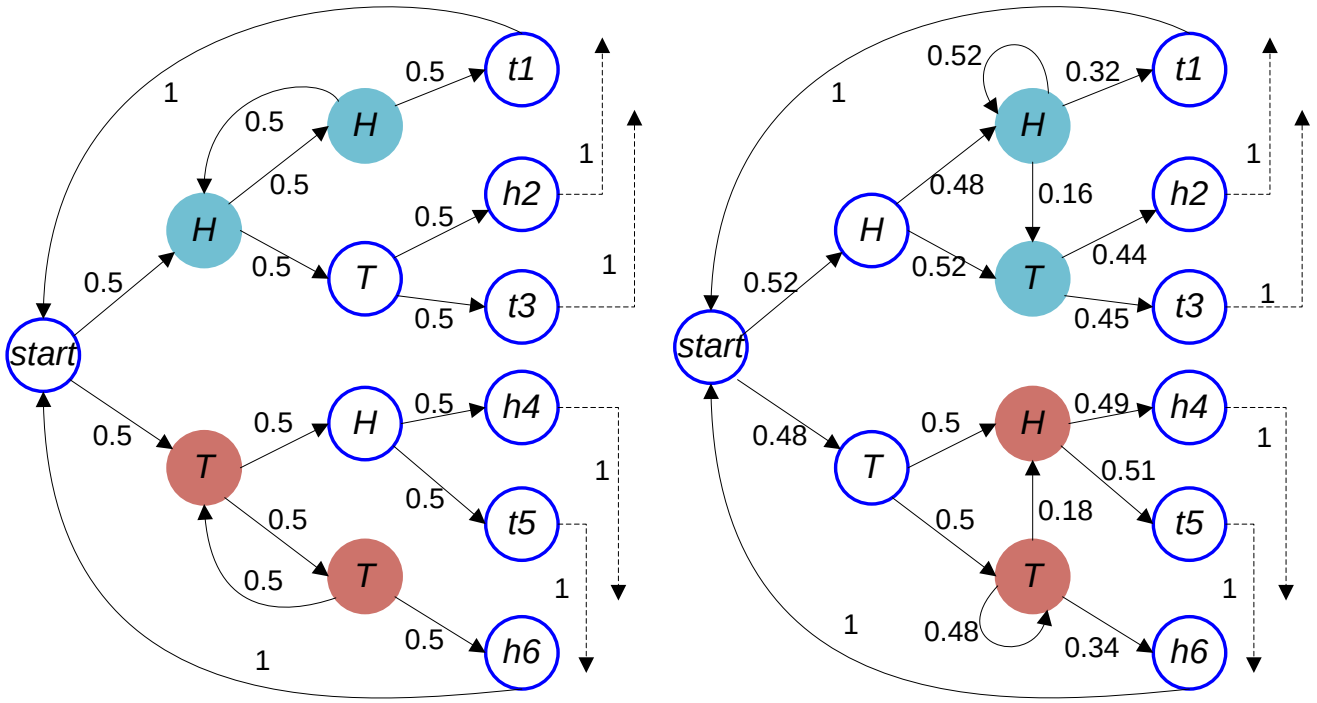


Figure: Left: The generating model. Right: A model learned from a sequence with 1440 symbols.

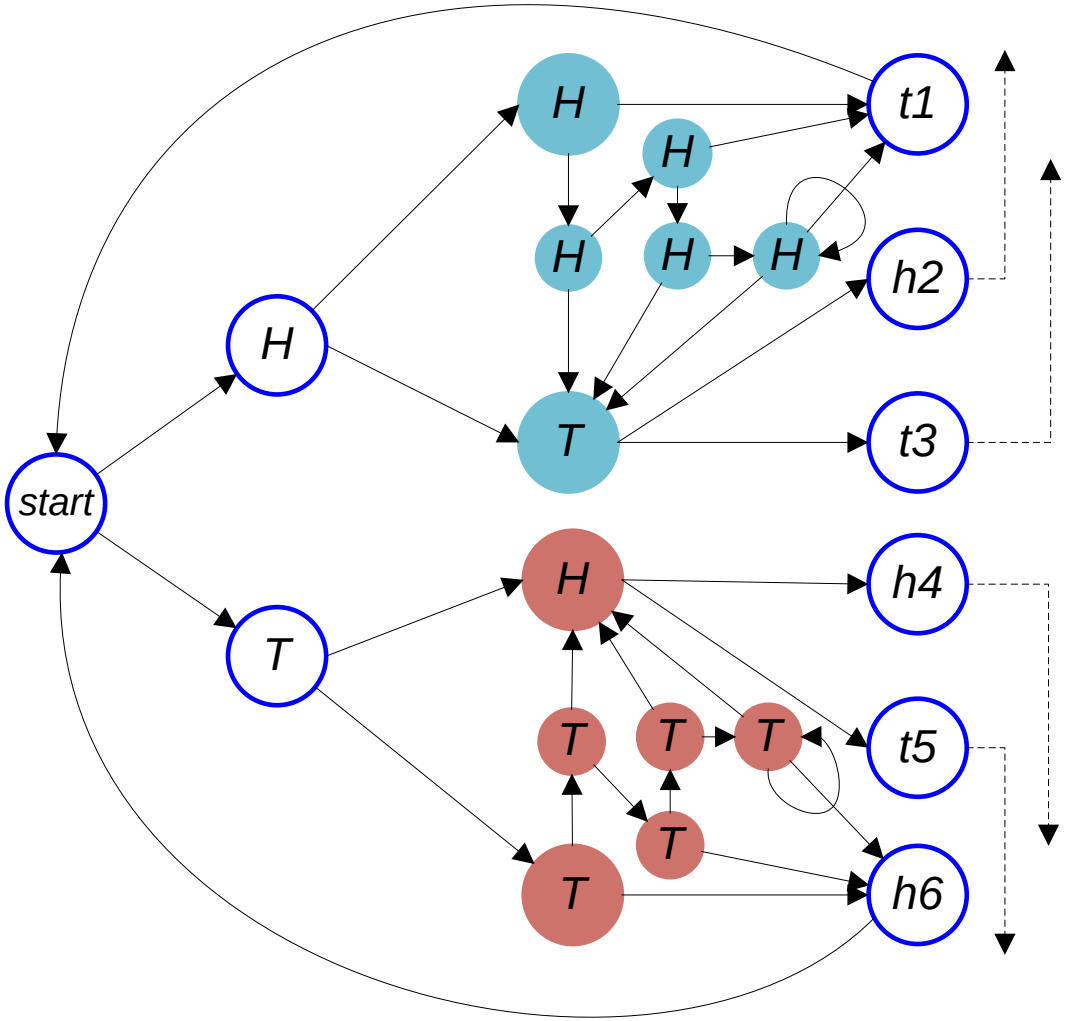
Dice Model *cont.*

Table: D is based on 501 random LTL formulas. For reference: $D_{dummy} = 0.1014$

$\|S\|$	$\|Q_l\|$	D	$P^s(1)$	$P^s(2)$	$P^s(3)$	$P^s(4)$	$P^s(5)$	$P^s(6)$
360	13	0.0124	0.137	0.17	0.182	0.103	0.205	0.203
720	13	0.0043	0.188	0.174	0.174	0.149	0.168	0.147
1440	13	0.0023	0.184	0.166	0.169	0.143	0.153	0.185
2880	17	0.0023	0.173	0.166	0.159	0.142	0.176	0.184
5760	17	0.0016	0.173	0.165	0.153	0.161	0.174	0.174
11520	19	0.00094	0.162	0.17	0.176	0.157	0.168	0.167
20000	21	0.00092	0.164	0.173	0.171	0.166	0.164	0.162
M_g	13		0.167	0.167	0.167	0.167	0.167	0.167

For the non PSA-equivalent system, the learned model still provide good approximation for SPLTL properties.

20000 symbols!



Outline

Conclusion

- *Single* observation sequence
- Learning algorithms
- SPLTL for stationary behavior
- Experimental validation