technische universität
dortmund

Probabilistic Arithmetic Automata and their Application to Pattern Matching Statistics

Tobias Marschall and Sven Rahmann

Bioinformatics for High-Throughput Technologies
Chair of Algorithm Engineering
TU Dortmund, Germany

June 18th, 2008

Motivation

Given

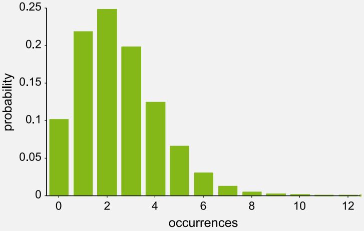
- an alphabet Σ
- a pattern, for example a finite set of strings over Σ
- a text model (for now: an i.i.d. model)

Sought

- **distribution** of random variable X_n (=number of matches in random string of length n)
- **p-value** for a given k, i.e. $\mathbb{P}(X_n \geq k)$

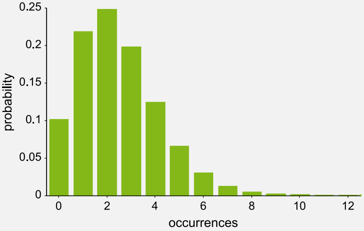
Example

- Pattern: ACACAC
- Textlength: 10,000
- Uniform distribution over $\Sigma = \{A,C,G,T\}$



Example

- Pattern: ACACAC
- Textlength: 10,000
- Uniform distribution over $\Sigma = \{A, C, G, T\}$



Related Work

- Régnier, 2000
- Reinert, Schbath, and Waterman, 2000
- Nicodème, Salvy, and Flajolet, 2002

Overview

1. Definition of **probabilistic arithmetic automata** (PAA) and generic algorithms on PAAs
2. Using PAAs for pattern matching statistics
3. Applicability in Computational Biology

Definition: Probabilistic Arithmetic Automaton

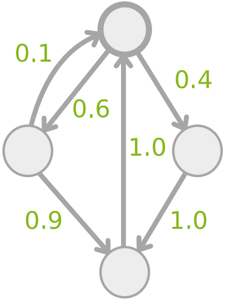
A **PAA** is a tuple $\big(Q, T, q_0, E, (\pi_q)_{q\in Q}, N, n_0, (\theta_q)_{q\in Q}\big)$:

Definition: Probabilistic Arithmetic Automaton

A **PAA** is a tuple $\big(Q, T, q_0, E, (\pi_q)_{q\in Q}, N, n_0, (\theta_q)_{q\in Q}\big)$:

- Q: finite set of **states**
- $T : Q \times Q \to [0,1]$: stochastic **transition function**, i.e. $T(q,q')$ is the probability of going from q to q'
- $q_0 \in Q$: **start state**

Definition: Probabilistic Arithmetic Automaton

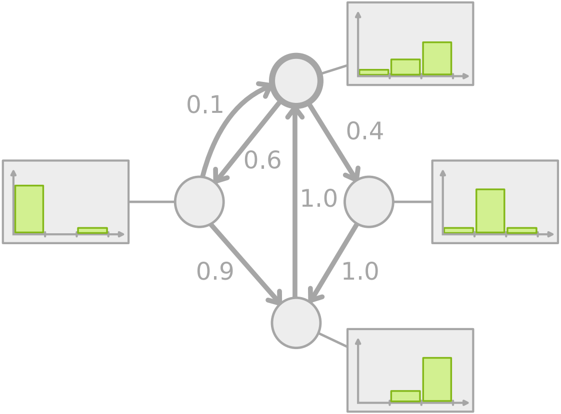


Definition: Probabilistic Arithmetic Automaton

A **PAA** is a tuple $\big(Q, T, q_0, E, (\pi_q)_{q\in Q}, N, n_0, (\theta_q)_{q\in Q}\big)$:

- Q: finite set of **states**
- $T : Q \times Q \rightarrow [0,1]$: stochastic **transition function**, i.e. $T(q,q')$ is the probability of going from q to q'
- $q_0 \in Q$: **start state**
- E: finite set called **emission set**
- $\pi_q : E \rightarrow [0,1]$: a **emission distribution** associated with state q

Definition: Probabilistic Arithmetic Automaton

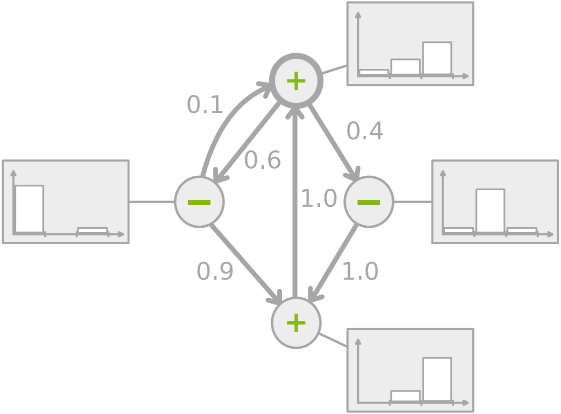


Definition: Probabilistic Arithmetic Automaton

A **PAA** is a tuple $\big(Q, T, q_0, E, (\pi_q)_{q \in Q}, N, n_0, (\theta_q)_{q \in Q}\big)$:

- Q: finite set of **states**
- $T : Q \times Q \to [0,1]$: stochastic **transition function**, i.e. $T(q, q')$ is the probability of going from q to q'
- $q_0 \in Q$: **start state**
- E: finite set called **emission set**
- $\pi_q : E \to [0,1]$: a **emission distribution** associated with state q
- N: finite set called **value set**
- $n_0 \in N$: **start value**
- $\theta_q : N \times E \to N$: an **operation** associated with state q

Definition: Probabilistic Arithmetic Automaton



Computing the Joint State-Value Distribution

Basic recurrence

$$p_{k+1}(q,v) = \sum_{q' \in Q} \sum_{(v',e) \in \theta_q^{-1}(v)} p_k(q',v') \cdot T(q',q) \cdot \pi_q(e)$$

$p_k(q,v)$: probability of being in state q and having computed a value of v after k steps

θ_q: operation associated with state q

T: transition function

π_q: emission distribution associated with state q

Q: set of all states

Runtime of Basic Algorithm

Basic recurrence

$$p_{k+1}(q,v) = \sum_{q' \in Q} \sum_{(v',e) \in \theta_q^{-1}(v)} p_k(q',v') \cdot T(q',q) \cdot \pi_q(e)$$

Time

$\mathcal{O}(m \cdot |Q|^2 \cdot |N|^2 \cdot |E|)$

Space

$\mathcal{O}(|Q| \cdot |N|)$

m: number of steps
Q: set of states
N: value set
E: emission set

Runtime of Basic Algorithm

Basic recurrence

$$p_{k+1}(q,v) = \sum_{q' \in Q} \sum_{(v',e) \in \theta_q^{-1}(v)} p_k(q',v') \cdot T(q',q) \cdot \pi_q(e)$$

Time

$\mathcal{O}(m \cdot |Q|^2 \cdot |N| \cdot |E|)$

Space

$\mathcal{O}(|Q| \cdot |N|)$

m: number of steps
Q: set of states
N: value set
E: emission set

Doubling Algorithm

Consider

$U^{(k)}(q_1, q_2, v_1, v_2)$: probability of being in state q_2 with value v_2 after k steps, given to have started in state q_1 with value v_1

Recurrence

$$U^{(1)}(q_1, q_2, v_1, v_2) = T(q_1, q_2) \cdot \sum_{\substack{e \in E: \\ \theta_{q_2}(v_1, e) = v_2}} \pi_{q_2}(e)$$

$$U^{(k_1+k_2)}(q_1, q_2, v_1, v_2) = \sum_{\substack{q' \in Q \\ v' \in N}} U^{(k_1)}(q_1, q', v_1, v') U^{(k_2)}(q', q_2, v', v_2)$$

Runtime of Doubling Algorithm

Recurrence

$$U^{(k_1+k_2)}(q_1, q_2, v_1, v_2) = \sum_{\substack{q' \in Q \\ v' \in N}} U^{(k_1)}(q_1, q', v_1, v') U^{(k_2)}(q', q_2, v', v_2)$$

Time

$\mathcal{O}(\log m \cdot |Q|^3 \cdot |N|^3)$

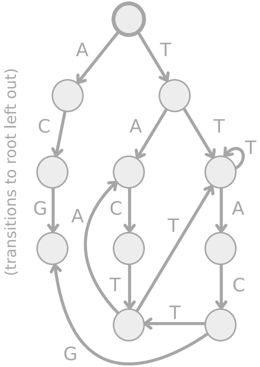
Space

$\mathcal{O}(|Q|^2 \cdot |N|^2)$

m: number of steps
Q: set of states
N: value set

Pattern Matching Statistics

{AC, ACG, TACT, TTAC}

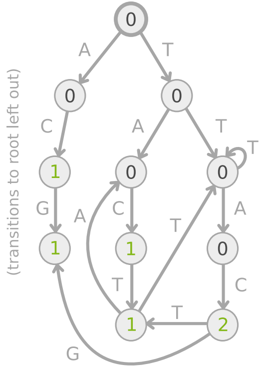


DFA construction

Step 1: Build Aho-Corasick automaton
Step 2: Transform into DFA

Pattern Matching Statistics

{AC, ACG, TACT, TTAC}



DFA construction

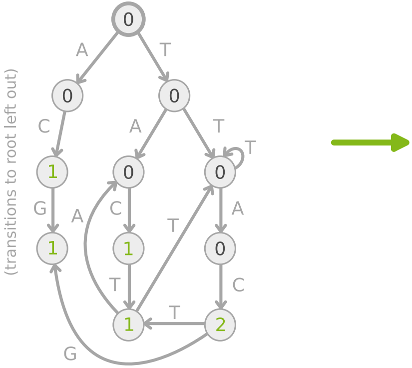
Step 1: Build Aho-Corasick automaton

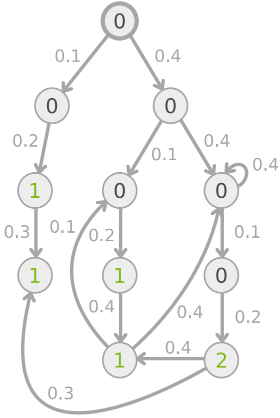
Step 2: Transform into DFA

Step 3: Annotate each state with number of matches to be counted when entering this state

Pattern Matching Statistics

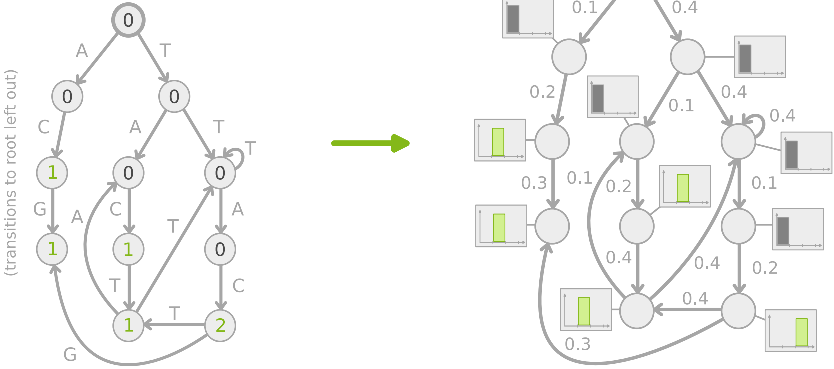
{AC, ACG, TACT, TTAC}





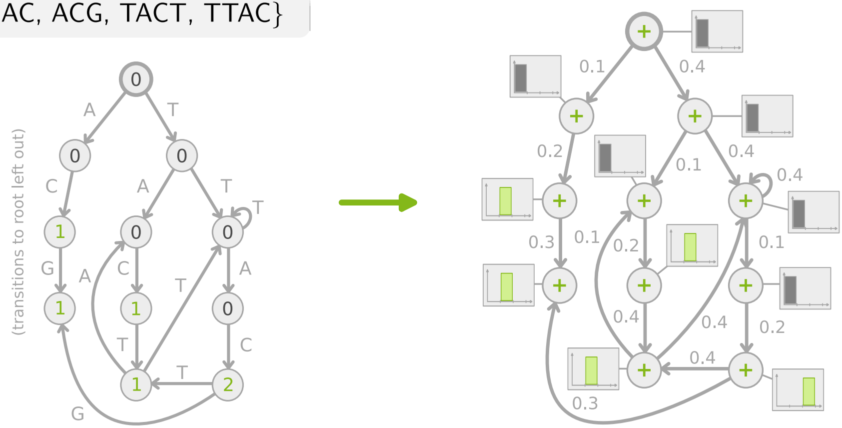
Pattern Matching Statistics

{AC, ACG, TACT, TTAC}



Pattern Matching Statistics

{AC, ACG, TACT, TTAC}



(transitions to root left out)

Runtimes for Pattern Matching Statistics

Algorithms

	Generic	Pattern Matching Statistics
Basic	$\mathcal{O}(m \cdot \lvert Q\rvert^2 \cdot \lvert N\rvert \cdot \lvert E\rvert)$	$\mathcal{O}(m \cdot \lvert\Sigma\rvert \cdot \lvert Q\rvert \cdot \lvert N\rvert)$
Doubling	$\mathcal{O}(\log m \cdot \lvert Q\rvert^3 \cdot \lvert N\rvert^3)$	$\mathcal{O}(\log m \cdot \lvert Q\rvert^3 \cdot \lvert N\rvert^2)$

m: number of steps
Q: set of states
N: value set
E: emission set
Σ: alphabet

Application: Amino Acid Motifs

PROSITE

Database with 1303 biologically meaningful patterns, examples:

`[LIV]-[STAG]-V-[DEQV]-[FLI]-D-[ST]`

`C-x(4,5)-C-C-S-x(2)-G-x-C-G-x(3,4)-[FYW]-C`

Experiment

For each pattern from PROSITE:

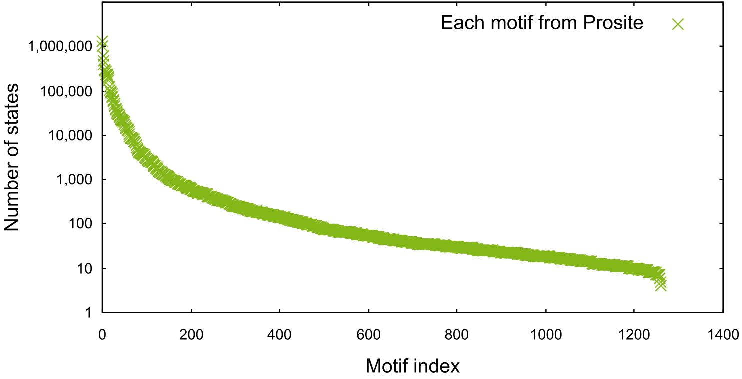
Pattern → NFA → DFA → PAA

Result

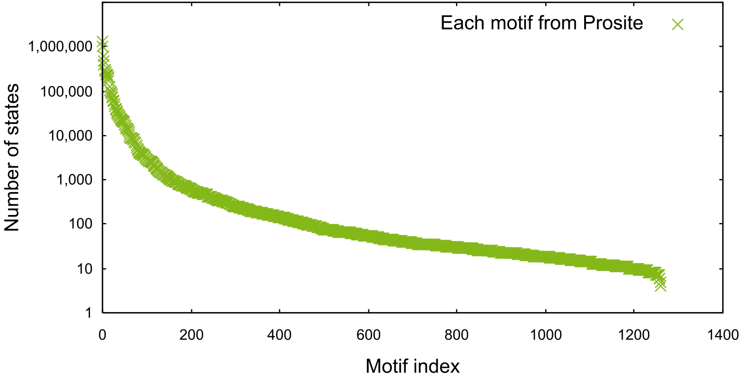
Despite exponential increase in the number of states in theory, automata fit into main memory for 1261 of 1303 patterns (96.8%).

Average runtime: 2 seconds

PROSITE: Automata (PAA) Sizes



PROSITE: Automata (PAA) Sizes



Runtime: textlength: 1000, matches: 50, states: 500 ⇒ 1s

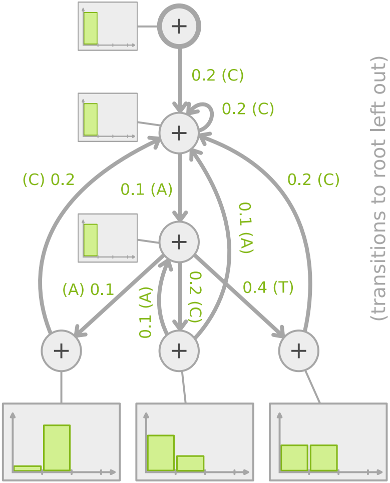
Probabilistic String Sets

String set

string	probability
CAA	0.9
CAT	0.5
CAC	0.3

Text model

character	probability
A	0.1
C	0.2
G	0.3
T	0.4



Applications of Stochastic Emissions

Transcription factor binding site statistics

JASPAR: Database containing position weight matrices

Step 1: Enumerate the *n* best-scoring strings
Step 2: Based on a biophysical model (Roider et al., 2007), calculate the probability that TF binds each string
Step 3: Use resulting probabilistic string set to build PAA

Statistics of fragment masses in cleavage reactions

- States emit masses of amino acids (Kaltenbach et al., 2006)
- Emission distribution may take isotopic distribution into account

Other things possible with PAAs

- Markovian text models
- Inhomogeneous text models
- Different counting schemes

Advantages of PAAs

- Built on DFAs, allows reuse of algorithms
- Easy to implement
- Permit exact statistics for practical problems
- Flexible

Other things possible with PAAs

- Markovian text models
- Inhomogeneous text models
- Different counting schemes

Advantages of PAAs

- Built on DFAs, allows reuse of algorithms
- Easy to implement
- Permit exact statistics for practical problems
- Flexible

Thank you for your attention!