Random Generation of Deterministic Tree (Walking) Automata

Pierre-Cyrille Héam[1,3] Cyril Nicaud[2] Sylvain Schmitz[3]

[1] LIFC, Université de Franche-Comté & INRIA, Besan con, France
[2] LIGM, Université Paris Est & CNRS, Marne-la-Vallée, France
[3] LSV, ENS Cachan & CNRS & INRIA, Cachan, France

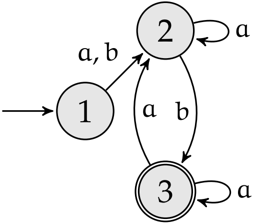
CIAA 2009, Sydney, July 15, 2009

Motivation

- automata are everywhere
  - many automata processing algorithms
- compare implementations
  - benchmarks
  - hard instances
  - random instances
- evaluate average complexities
  - for the uniform distribution
  - up to isomorphism

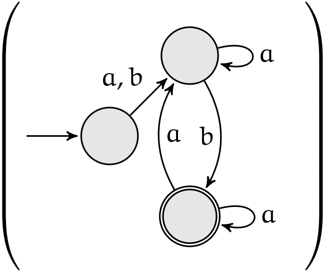
Random Generation

- of finite deterministic accessible automata with n states
- up to isomorphism
- uniformly



Random Generation

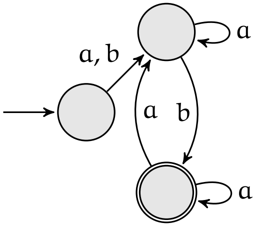
- of finite deterministic accessible automata with n states
- up to isomorphism
- uniformly



Random Generation

- of finite deterministic accessible automata with n states
- up to isomorphism
- uniformly

p (



) $= \frac{1}{\#\text{classes}}$

Overview

- we want to generate tree automata in some family $\mathcal{T}_n$
- we know how to generate deterministic words automata with n states, i.e. elements of $\mathcal{A}_n$

1. define a bijection φ between $\mathcal{T}_n$ and a subclass X of $\mathcal{A}_n$
2. use a rejection algorithm

Rejection Algorithm

- we have a random generator for Y with law p_Y
- $X \subseteq Y$ and p_X is the restriction of p_Y to X
- $\textsc{Generate}(X, p_X)$

  do

  $e \leftarrow \textsc{Generate}(Y, p_Y)$

  until $e \in X$

  return e
- average iterations: $O(1/p_Y(X))$

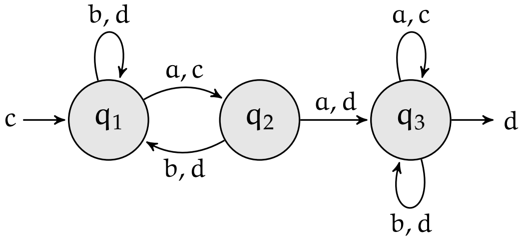
Sequential Letter-to-letter Transducers

$\langle \Sigma_1, \Sigma_2, Q, q_{init}, \delta, \gamma, \rho, a_{init} \rangle$

- Σ_1 input alphabet, Σ_2 output alphabet
- Q finite set of states
- $q_{init} \in Q$ initial state
- $\delta : Q \times \Sigma_1 \rightarrow Q$ partial transition function
- $\gamma : Q \times \Sigma_1 \rightarrow \Sigma_2$ partial output function with $\mathrm{Dom}(\delta) = \mathrm{Dom}(\gamma)$
- $\rho : Q \rightarrow \Sigma_2$ partial final function
- $a_{init} \in \Sigma_2$ initial output

Sequential Letter-to-letter Transducers

Example



$$T(babaaa) = cdcdcdcd$$

Generating SLTs

- generate a deterministic, accessible, and complete automaton $\mathcal{A} = \langle Q, \Sigma_1, \delta, q_{init}, F \rangle$ with n states
- pick a_{init} in Σ_2
- $\forall q \in Q, a \in \Sigma_1$, pick $\gamma(q, a)$ in $\Sigma_2 = \{a_1, \ldots, a_k\}$
- $\forall q \in Q$, pick $\rho(q)$ in $\Sigma_2 \uplus \{\text{undefined}\}$

Generating SLTs

- generate a deterministic, accessible, and possibly incomplete automaton $\mathcal{A} = \langle Q, \Sigma_1, \delta, q_{\mathrm{init}}, F \rangle$ with n states
- pick a_{init} in Σ_2
- $\forall q \in Q, a \in \Sigma_1$, pick $\gamma(q, a)$ in $\Sigma_2 = \{a_1, \ldots, a_k\}$
- reject if $\delta(q, a)$ is undefined and $\gamma(q, a) \neq a_1$
- $\forall q \in Q$, pick $\rho(q)$ in $\Sigma_2 \uplus \{\mathrm{undefined}\}$

Generating SLTs

Define restrictions on initializations r_i, transitions r, and finalizations r_F:

- generate a deterministic, accessible, and possibly incomplete automaton $\mathcal{A} = \langle Q, \Sigma_1, \delta, q_{init}, F \rangle$ with n states
- pick a_{init} in r_i
- $\forall q \in Q, a \in \Sigma_1$, pick $\gamma(q, a)$ in $r(a) = \{a_1, \ldots, a_k\}$
- reject if $\delta(q, a)$ is undefined and $\gamma(q, a) \neq a_1$
- $\forall q \in Q$, pick $\rho(q)$ in $r_F \uplus \{\text{undefined}\}$

Generating SLTs

- uses the technique of Bassino et al. [2009] for the generation of deterministic, accessible, and possibly incomplete automata, in $O(n^{3/2})$ on average
- the proportion of complete automata is greater than a constant
- only incomplete automata can get rejected
- thus the average complexity for SLTs is still $O(n^{3/2})$

New Overview

- we want to generate tree automata in some family $\mathcal{T}_n$
- we know how to generate SLTs with n states, complete $\mathcal{C}_n(\Sigma_1, \Sigma_2, r, r_i, r_F)$ or possibly incomplete $\mathcal{D}_n(\Sigma_1, \Sigma_2, r, r_i, r_F)$

1. define a bijection φ between $\mathcal{T}_n$ and a class X such that

$$\mathcal{C}_n(\Sigma_1, \Sigma_2, r, r_i, r_F) \subseteq X \subseteq \mathcal{D}_n(\Sigma_1, \Sigma_2, r, r_i, r_F)$$

2. use a rejection algorithm

Deterministic Tree Walking Automata

$\langle Q, \Sigma, \Delta, q_{init}, F \rangle$ on binary trees

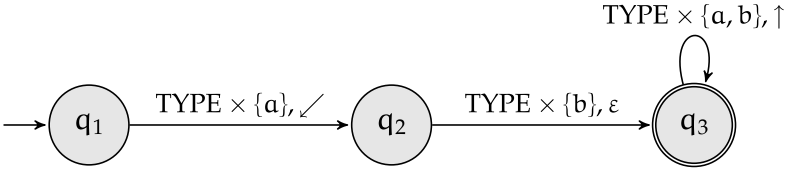
- Q finite set of states
- $\Delta : Q \times \text{TYPE} \times \Sigma \to \text{DIR} \times Q$, where $\text{TYPE} = \{\text{root}, \text{left}, \text{right}\} \times \{\text{internal}, \text{leaf}\}$, $\text{DIR} = \{\varepsilon, \uparrow, \swarrow, \searrow\}$
- $q_{init} \in Q$ initial state
- $F \subseteq Q$ set of final states

Tree Walking Automata

- connections with logics on trees [Engelfriet and Hoogeboom, 1999, ten Cate and Segoufin, 2008]

Example

$$/\texttt{a}/\texttt{child}[1] :: \texttt{b}$$

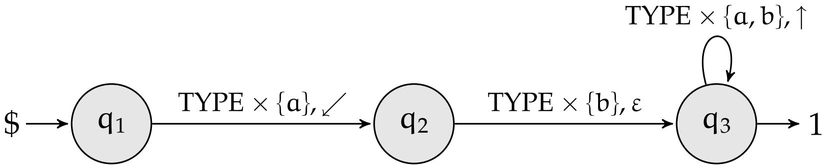


Generating DTWAs

$$\tau(\mathcal{A}) = \langle \Sigma_1, \mathrm{DIR} \uplus \{\$, 1\}, Q, q_{init}, \delta, \gamma, \rho, \$ \rangle$$

- $\Sigma_1 = \mathrm{TYPE} \times \Sigma$
- $\delta(q, (t, a)) = p$ and $\gamma(q, (t, a)) = d$ iff $\Delta(q, t, a) = (d, p)$
- $\mathrm{Dom}(\rho) = F$ with $\rho(q) = 1$ iff $q \in F$

Example



Generating DTWAs

- restriction on initializations: $r_i = \$$
- restriction on transitions: $\forall a \in \Sigma$

$$r(t,a) = \{\varepsilon, \swarrow, \searrow\} \quad \text{for } t \in \{\text{root}\} \times \{\text{internal}, \text{leaf}\}$$
$$r(t,a) = \{\varepsilon, \uparrow\} \quad \text{for } t \in \{\text{root}, \text{left}, \text{right}\} \times \{\text{leaf}\}$$

- restriction on finalizations: $r_F = 1$
- τ is a bijection from DTWAs to restricted SLTs

Generating DTWAs

- restriction on initializations: $r_i = \$$
- restriction on transitions: $\forall a \in \Sigma$

$$r(t,a) = \{\varepsilon, \swarrow, \searrow\} \quad \text{for } t \in \{\text{root}\} \times \{\text{internal}, \text{leaf}\}$$
$$r(t,a) = \{\varepsilon, \uparrow\} \quad \text{for } t \in \{\text{root}, \text{left}, \text{right}\} \times \{\text{leaf}\}$$

- restriction on finalizations: $r_F = 1$
- τ is a bijection from DTWAs to restricted SLTs

Application

Emptiness of a tree-walking automaton:

1. construct an equivalent bottom-up tree automaton with size $O(2^{n^2})$
2. test it for emptiness

An ExpTime-complete problem.

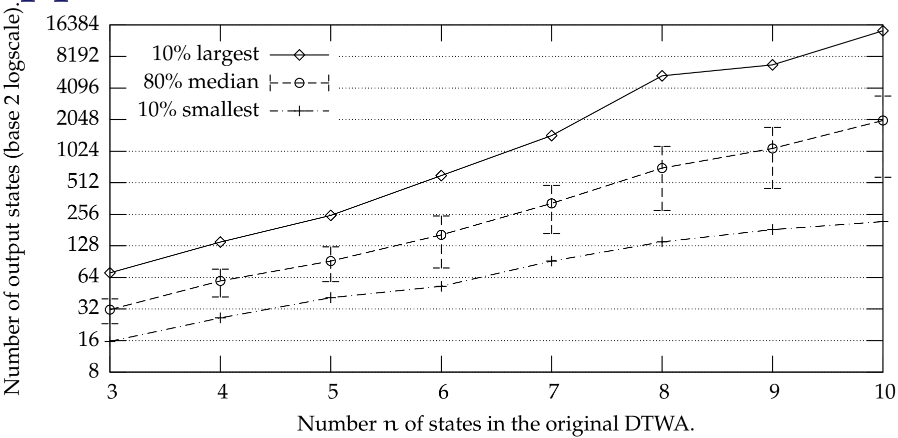
Application

Emptiness of a tree-walking automaton:

1. construct an equivalent bottom-up tree automaton with size $O(2^{n^2})$
2. test it for emptiness

An ExpTime-complete problem.

Application



Average number of states in the 10 smallest, the 10 largest, and the 80 median accessible bottom-up tree automata obtained from transforming 100 2-letter DTWAs with n states

Deterministic Top-Down Tree Automata

$\langle Q, \mathcal{F}, \theta, q_{init} \rangle$

- Q finite set of states, $0 \notin Q$,
- $\mathcal{F}$ ranked alphabet,
- $q_{init} \in Q$ initial state,
- θ partial transition function, $Q \times \mathcal{F}_i \to Q^i$ for all $i \geqslant 1$, and $Q \times \mathcal{F}_0 \to \{0\}$

Deterministic Top-Down Tree Automata

- connections with XML schema languages [Neven, 2002, Murata et al., 2005]

Example

```
<!ELEMENT book (author, title, isbn)>
<!ELEMENT author (firstname, lastname)>
<!ELEMENT firstname (#PCDATA)>
<!ELEMENT lastname (#PCDATA)>
<!ELEMENT title (#PCDATA)>
<!ELEMENT isbn (#PCDATA)>
```

Deterministic Top-Down Tree Automata

Example

```
<!ELEMENT book (author, title, isbn)>
<!ELEMENT author (firstname, lastname)>
<!ELEMENT firstname (#PCDATA)>
<!ELEMENT lastname (#PCDATA)>
<!ELEMENT title (#PCDATA)>
<!ELEMENT isbn (#PCDATA)>
```

$$\delta(q_1, \mathtt{book}) = (q_2, q_3, q_4) \qquad \delta(q_2, \mathtt{author}) = (q_5, q_6)$$
$$\delta(q_3, \mathtt{title}) = (q_7) \qquad \delta(q_4, \mathtt{isbn}) = (q_7)$$
$$\delta(q_5, \mathtt{firstname}) = (q_7) \qquad \delta(q_6, \mathtt{lastname}) = (q_7)$$
$$\delta(q_7, \mathtt{PCDATA}) = 0$$

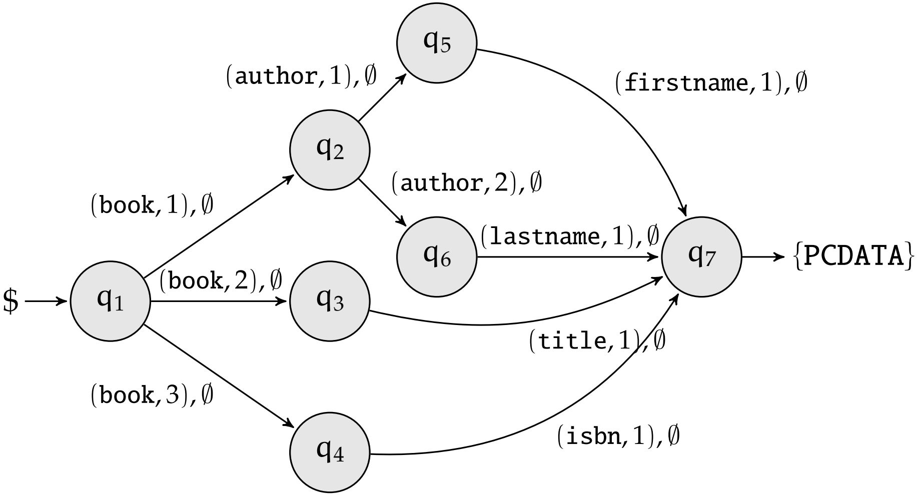
Generating DTDAs

$$\psi(\mathcal{A}) = \langle \overline{\mathcal{F}}, 2^{\mathcal{F}_0} \uplus \{\$\}, Q, q_{init}, \delta, \gamma, \rho, \$\rangle$$

- $\overline{\mathcal{F}} = \{(f, i) \mid f \in \mathcal{F} \setminus \mathcal{F}_0,\ 1 \leqslant i \leqslant \text{arity}(f)\}$
- $\gamma(q,(f,i)) = \emptyset$ and $\delta(q,(f,i)) = p_i$ iff $\theta(q, f) = (p_1, \ldots, p_n)$
- $\rho(q) = \{A \in \mathcal{F}_0 \mid \theta(q, A) = 0\}$ iff this set is not empty, and $\rho(q)$ is undefined otherwise

Generating DTDAs

Example



Generating DTDAs

- restriction on initializations: $r_i = \$$
- restriction on transitions: $\forall(a, j) \in \overline{\mathcal{F}}, r(a, j) = \emptyset$
- restriction on finalizations: $r_F = 2^{\mathcal{F}_0} \backslash \emptyset$
- additional *coherence* condition: if $\delta(q, (a, j))$ is defined for some $a \in \mathcal{F}_i$ and $1 \leqslant j \leqslant i$, then it is defined for all $1 \leqslant j \leqslant i$
- ψ is a bijection from DTDAs to restricted, coherent SLTs

Concluding Remarks

From an applications' viewpoint, one would like

- nondeterministic automata
- unranked alphabets

References

Bassino, F., David, J., and Nicaud, C., 2009. Enumeration and random generation of possibly incomplete deterministic automata. *Pure Mathematics and Applications*.

Engelfriet, J. and Hoogeboom, H.J., 1999. Tree-walking pebble automata. In Karhumäki, J., Maurer, H.A., Paun, G., and Rozenberg, G., editors, *Jewels are Forever*, pages 72–83. Springer. ISBN 3-540-65984-6.

Murata, M., Lee, D., Mani, M., and Kawaguchi, K., 2005. Taxonomy of XML schema languages using formal language theory. *ACM Transactions on Internet Technology*, 5(4):660–704. doi:10.1145/1111627.1111631.

Neven, F., 2002. Automata theory for XML researchers. *SIGMOD Record*, 31 (3):39–46. doi:10.1145/601858.601869.

ten Cate, B. and Segoufin, L., 2008. XPath, transitive closure logic, and nested tree walking automata. In Lenzerini, M. and Lembo, D., editors, *PODS'08*, pages 251–260. ACM. doi:10.1145/1376916.1376952.