Universität Konstanz

Minimization of Weighted Automata

Fabian Klopfer

Modelling of Complex Self-Organizing Systems Group, 12.06.2020

Introduction

In the last presentation . . .

- two models for stochastic dynamical systems were considered: Weighted Automata (WA) and Differential Equations (DE)

- an example for modelling a CRN's dynamics in both models was given:

  DE Solving Chemical Master Equation
  WA Monte Carlo CTMC

Goals specified

1. Implement minimization algorithm for weighted automata [1]. ✓
2. Implement model reduction algorithm for ODEs [2].
3. Develop reproducible benchmarks
4. Write report

What has been done so far

- Software Requirement Specification & Software Design Document
- Random Basis Minimal WA Construction Algorithm by Kiefer/Schützenberger [1]
- Execution of example by Matlab script and hand
- Implementation of minimization & equivalence algorithm, interfaces, TUI, CLI, tests

The Weighted Automaton Minimization Algorithm [1] I

Weighted Automaton $A = (n, \Sigma, \alpha, \mu, \eta)$, where

- n the number of states
- Σ the input alphabet
- α the initial vector with a non-zero value for all starting states
- μ the set of transition matrices, one per input character
- η the final vector with non-zero values for all ending states

The Weighted Automaton Minimization Algorithm [1] II

Reduction to Schwarz-Zippel Lemma (Polynomial Identity Testing) provide bounds on correctness $\frac{n}{K}$

Author claims complexity is in $\mathcal{NC}$, this is not correct (c.f. later)

In the following slides use forward reduction, the backwards reduction is similar

The Weighted Automaton Minimization Algorithm [1] III

- Find a basis F of the prefix space using random vectors r_i
  - Add the vectors of all prefix words up to length n together and multiply this vector by n different factors yielding $\{v_1, \ldots, v_n\}$
  - Factors are derived by random vectors and structure of prefixes
  - Base is then the maximally linear independent subset of $\{\alpha, v_1, \ldots, v_n\}$

- Use basis to do Schützenberger Construction [3]: $\overrightarrow{A} = (\overrightarrow{n}, \Sigma, \overrightarrow{\alpha}, \overrightarrow{\mu}, \overrightarrow{\eta})$ With
  - $\overrightarrow{\mu} = \overrightarrow{F} \mu \overrightarrow{F}^{-1}$ or $\overrightarrow{F} \mu = \overrightarrow{\mu} \overrightarrow{F}$
  - $\overrightarrow{\alpha} = e_1$
  - $\overrightarrow{\eta} = \overrightarrow{F} \eta$
  - $\overrightarrow{n} = \text{rank}(\overrightarrow{\mu})$

The Weighted Automaton Minimization Algorithm: Pseudo Code I

Algorithm 1: minimize(WeightedAutomaton WA)

Input: A weighted automata WA
Parameters: K setting the maximal random number
Output: A minimal version of WA

```
begin
    List<Matrix> randVs;
    WeightedAutomaton minWA;

    randVs ← generate_random_vectors(WA, K);
    minWA ← forward_reduction(WA, randVs);

    randVs ← generate_random_vectors(minWA, K);
    minWA ← backward_reduction(minWA, randVs);

    return minWA;
```

Array randVs has size A.n and matrices $r_i \in \{1, \ldots, K \cdot n\}^{\Sigma \times n}$

The Weighted Automaton Minimization Algorithm: Pseudo Code II

Algorithm 2: forward_reduction(WA, randVs)

Input: A weighted automata WA, random vectors randVs
Output: WA transformed by a random minimal forward space base

begin
- List<Vector> rhoVectors $\leftarrow$ calculate_rho_vectors(WA, randVs);
- Matrix $\overrightarrow{F} \leftarrow$ vstack(A.α, rhoVectors[0], $\ldots$, rhoVectors[n - 1]);
- int $\overrightarrow{n} \leftarrow$ rank($\overrightarrow{F}$);
- $\overrightarrow{F} \leftarrow \overrightarrow{F}[:\overrightarrow{n} - 1, :]$;
- RowVector $\overrightarrow{\alpha} \leftarrow$ standard_basis($\overrightarrow{n}$)[0, :];
- Vector $\overrightarrow{\eta} \leftarrow \overrightarrow{F} \cdot \eta$;
- **foreach** $\mu_i \in WA.\mu$ **do**
  - $\overrightarrow{\mu_i} \leftarrow$ Solver($\overrightarrow{F}^T$).solve(($\overrightarrow{F} \cdot \mu_i)^T)^T$;
- **return** $\left(\overrightarrow{n}, \text{WA}.\Sigma, \overrightarrow{\alpha}, \overrightarrow{\mu}, \overrightarrow{\eta}\right)$;

The Weighted Automaton Minimization Algorithm: Pseudo Code III

Algorithm 3: calculate_rho_forward_vectors(WA, randVs)

Input: A weighted automata WA, random vectors randVs
Output: Candidate vectors as base of the Forward space

```
begin
    List<(Word, Vector)> words ← generate_words_forward(WA, WA.n);
    for i = 0 to randVs.length - 1 do
        for j = 0 to words.length - 1 do
            v_i += words[j].vector * get_word_factor(words[j].word, randVs[i]);
    return {v_0, ..., v_{randVs.length-1}};
```

Algorithm 4: get_word_factor(WA, randVs)

Input: Word w, randVs random vector
Output: WA transformed by a random minimal forward space base

```
begin
    result ← 1;
    for i = 0 to do
        result *= randVs(word[i], i);
```

The Weighted Automaton Minimization Algorithm: Pseudo Code IV

Algorithm 5: generate_words_forward(WA, randVs)

Input: A weighted automata WA, length of words k
Output: List of Tuples of word and corresponding vector

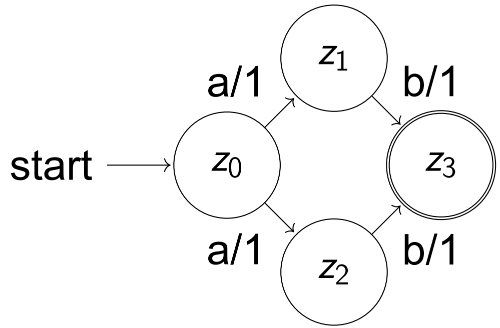
```
begin
    List<Word, Vector> result;
    if k == 1 then
        result = ;
        foreach μ_i ∈ WA.μ do
            vect ← WA.α · μ_i;
            if !vect.isZero() then
                result.add(i, vect);
    else
        result = generate_words_forward(WA, k − 1);
        for (word, wVector) ∈ result do
            if word.length == k - 1 then
                foreach μ_i ∈ WA.μ do
                    vect ← wVector·μ_i;
                    if !vect.isZero() then
                        newWord ← word;
                        newWord.append(i);
                        result.add(newWord, vect);
    return result;
```

The Weighted Automaton Minimization Algorithm: Example I

Weighted automaton $\mathcal{A} = (n, \Sigma, \mu, \alpha, \eta)$ with

- $n = 3$,
- $\Sigma = \{a, b\}$,
- $\alpha = (1, 0, 0, 0)$,
- $\eta = (0, 0, 0, 1)$,

- $\mu(a) = \begin{pmatrix} 0 & 1 & 1 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \end{pmatrix}$,
- $\mu(b) = \begin{pmatrix} 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 \end{pmatrix}$.



The Weighted Automaton Minimization Algorithm: Example II

$$r^{(1)} = \begin{pmatrix} 9 & 5 & 5 & 7 \\ 6 & 11 & 2 & 1 \end{pmatrix}; \quad r^{(2)} = \begin{pmatrix} 2 & 3 & 1 & 2 \\ 12 & 3 & 9 & 4 \end{pmatrix} r^{(3)} = \begin{pmatrix} 2 & 7 & 9 & 10 \\ 1 & 11 & 2 & 6 \end{pmatrix} r^{(4)} = \begin{pmatrix} 4 & 5 & 2 & 10 \\ 5 & 9 & 5 & 5 \end{pmatrix}$$

$$\alpha\mu(a)r_i + \alpha\mu(a)\mu(b)r_i = (0,1,1,0)r_i + (0,0,0,2)r_i$$

$$v_1 = 9 \cdot (0,1,1,0) + 9 \cdot 11 \cdot (0,0,0,2) = (0,9,9,198)$$

$$v_2 = 2 \cdot (0,1,1,0) + 2 \cdot 3 \cdot (0,0,0,2) = (0,2,2,12)$$

$$v_3 = 2 \cdot (0,1,1,0) + 2 \cdot 11 \cdot (0,0,0,2) = (0,2,2,44)$$

$$v_4 = 4 \cdot (0,1,1,0) + 4 \cdot 9 \cdot (0,0,0,2) = (0,4,4,72)$$

$$\overrightarrow{F} = \begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & 9 & 9 & 198 \\ 0 & 2 & 2 & 12 \end{pmatrix}$$

The Weighted Automaton Minimization Algorithm: Example III

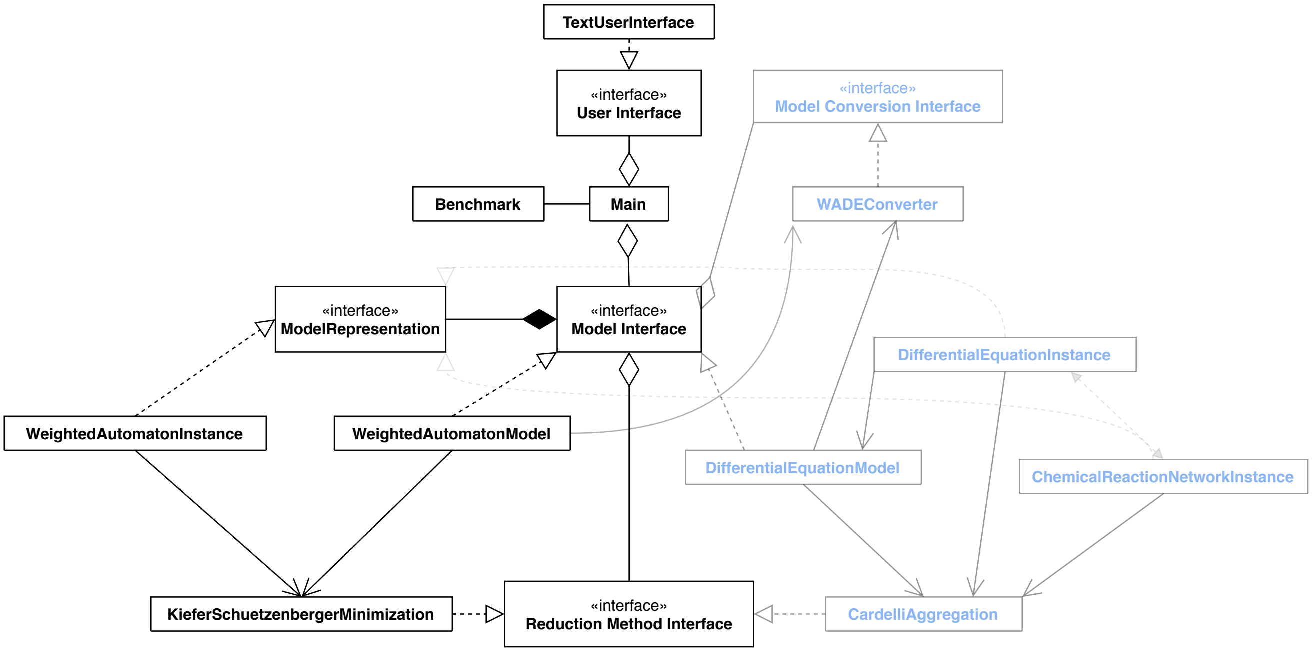
$$\overrightarrow{F}\mu(\sigma) = \overrightarrow{\mu}(\sigma)\overrightarrow{F} \equiv \overrightarrow{\mu}(\sigma) = \overrightarrow{F}\mu(\sigma)\overrightarrow{F}_R^{-1}$$

$$\overrightarrow{\mu}(a) = \begin{pmatrix} 0 & \frac{-1}{24} & \frac{11}{16} \\ 0 & 0 & 0 \\ 0 & 0 & 0 \end{pmatrix} \qquad \overrightarrow{\mu}(b) = \begin{pmatrix} 0 & 0 & 0 \\ 0 & \frac{1}{8} & \frac{-9}{16} \\ 0 & \frac{1}{36} & \frac{-1}{8} \end{pmatrix}$$

$$\overrightarrow{\eta} = \overrightarrow{F}\eta = \begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & 9 & 9 & 198 \\ 0 & 2 & 2 & 12 \end{pmatrix} \cdot (0, 0, 0, 1)^T = (0, 198, 12)^T$$

$$\Rightarrow \overrightarrow{\mathcal{A}} = (3, \{a, b\}, \overrightarrow{\mu}, (1, 0, 0), (0, 198, 12)^T)$$

Implementation Details I



Up Next

e

Bibliography

- S. Kiefer, A. S. Murawski, J. Ouaknine, B. Wachter, and J. Worrell, "On the complexity of equivalence and minimisation for q-weighted automata", *Logical Methods in Computer Science*, vol. 9, 2013.
- L. Cardelli, M. Tribastone, M. Tschaikowski, and A. Vandin, "Maximal aggregation of polynomial dynamical systems", *Proceedings of the National Academy of Sciences of the United States of America*, vol. 114 38, pp. 10 029–10 034, 2017.
- M. P. Schützenberger, "On the definition of a family of automata", *Information and control*, vol. 4, no. 2-3, pp. 245–270, 1961.