

Ульянцев Владимир Игоревич

**Генерация конечных автоматов с использованием
программных средств решения задач
выполнимости и удовлетворения ограничений**

Диссертация на соискание ученой степени кандидата технических наук

Специальность 05.13.11 – «Математическое и программное обеспечение
вычислительных машин, комплексов и компьютерных сетей»

Научный руководитель – доктор технических наук, профессор
Шалыто Анатолий Абрамович

24.12.2015



Введение: актуальность (1)

- Компоненты аппаратного и программного обеспечения моделируются **конечными автоматами**
- Представление в виде наглядных диаграмм состояний, процесс верификации методом model checking
- **Методы генерации автоматов:** решение сложных задач, повышение надежности и уменьшение степени влияния человеческого фактора
- Можно выделить задачи автоматизированной генерации **детерминированных** автоматных моделей по заранее заданным **примерам поведения** (в основном «неточные»)
- Применение: генерация и анализ программ, грамматический вывод, биоинформатика, лингвистика, распознавание речи

Введение: актуальность (2)

- Задача генерации детерминированного конечного автомата (ДКА) с наименьшим числом состояний по *обучающим словарям* **NP-полна**
⇒ актуальность разработки и реализации практически применимых методов генерации
- **Высокая производительность** программных средств решения классических NP-полных задач выполнимости и удовлетворения ограничений
- При сведении к ним
 - **без изменения** алгоритма расширяются границы применимости
 - **точность** решения обеспечена использованием поиска с возвратом

Актуальность развития существующих и разработка новых **точных методов генерации** конечных автоматов по примерам поведения с использованием сведения к SAT и CSP

Цель диссертационной работы – разработка точных методов генерации конечных автоматов по примерам поведения с использованием программных средств решения задач выполнимости и удовлетворения ограничений. **Задачи:**

1. Доказать **NP-трудность** задачи генерации управляющих конечных автоматов заданного размера по сценариям работы
2. Разработать и реализовать точные **методы генерации ДКА** по безошибочным и зашумленным обучающим словарям с использованием программных средств решения SAT
3. Разработать и реализовать точные **методы генерации управляющих автоматов** по безошибочным и зашумленным сценариям работы с использованием программных средств решения задачи CSP
4. Внедрить разработанные методы генерации ДКА после их публикации в инструментальное средство **DFASAT** (Делфтский технический университет)

Научная новизна, выносимые на защиту положения

1. Доказательство **NP-трудности** задачи построения управляющих конечных автоматов по сценариям работы
2. Точные методы генерации ДКА по безошибочным и зашумленным обучающим словарям
 - разработанные **предикаты нарушения симметрии**
 - по зашумленным словарям – **точный, в отличие от известных**
3. Точные методы генерации управляющих конечных автоматов по сценариям работы
 - новые **алгоритмы построения дерева сценариев и его графа совместимости**
 - известные методы являются **неточными**
4. Внедрение в средство DFASAT

Глава 1. Задачи выполнимости и удовлетворения ограничений, генерация конечных автоматов

Определения

Обзор работ

Формулировка задач

Задача выполнимости (SAT)

- Задача *выполнимости булевых формул* (задача выполнимости, SAT): нахождение выполняющей подстановки значений переменных для заданной булевой формулы
- Исторически первая NP-полная задача (теорема Кука-Левина)
- Далее по умолчанию используется КНФ (конъюнктивная нормальная форма)

$$(X1 \text{ or } X2 \text{ or } \overline{X3})$$

$$(\overline{X1} \text{ or } \overline{X2} \text{ or } X3)$$

$$(\overline{X1} \text{ or } \overline{X2} \text{ or } \overline{X3})$$

$$(\overline{X1} \text{ or } X2 \text{ or } X3)$$

$$(\boxed{X1} \text{ or } X2 \text{ or } \overline{X3})$$

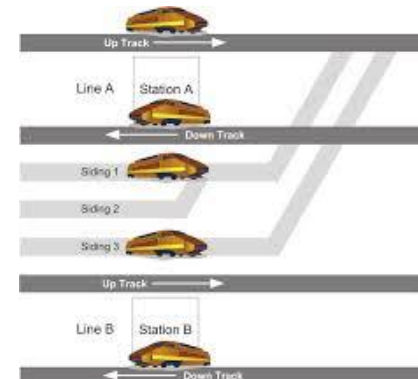
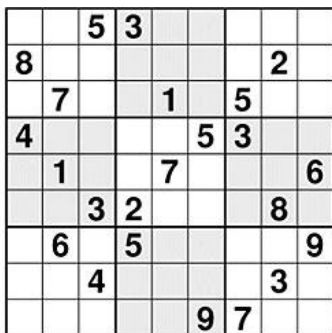
$$(\overline{X1} \text{ or } \boxed{\overline{X2}} \text{ or } \boxed{X3})$$

$$(\overline{X1} \text{ or } \boxed{\overline{X2}} \text{ or } \overline{X3})$$

$$(\overline{X1} \text{ or } X2 \text{ or } \boxed{X3})$$

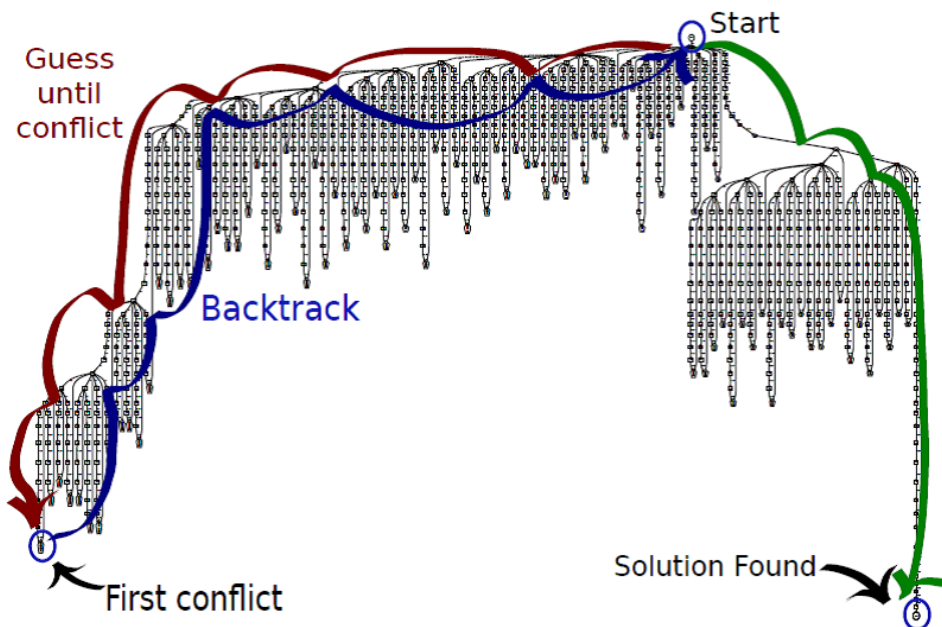
Задача удовлетворения ограничений (CSP)

- Задача удовлетворения ограничений (обобщенная задача выполнимости, constraint satisfaction problem – CSP), расширяет задачу SAT для переменных с произвольными значениями взамен булевых
- Экземпляр задачи CSP представляет собой тройку $\langle X, D, C \rangle$
 - X – набор переменных
 - D – множество их наборов допустимых значений
 - C – множество ограничений на X



Программные средства решения SAT

- Точные основаны на алгоритме DPLL (1962) [32]: поиске с возвратом для решения CNF-SAT
- Ежегодные соревнования
- Выбор программного средства на основе их результатов
 - *SAT Competition*
 - *SAT-Race*
- *Lingeling, CryptoMiniSat*



[32] Davis M., Logemann G., Loveland D.
A Machine Program for Theorem-Proving
// Communications of the ACM. – 1962.
– Vol. 5, no. 7. – P. 394-397



Программные средства решения CSP

- Проводятся ежегодные соревнования **MiniZinc Challenge**
- Будут использованы программы **OR-tools** и **Opturion CPX**
- Язык **MiniZinc** задания ограничений

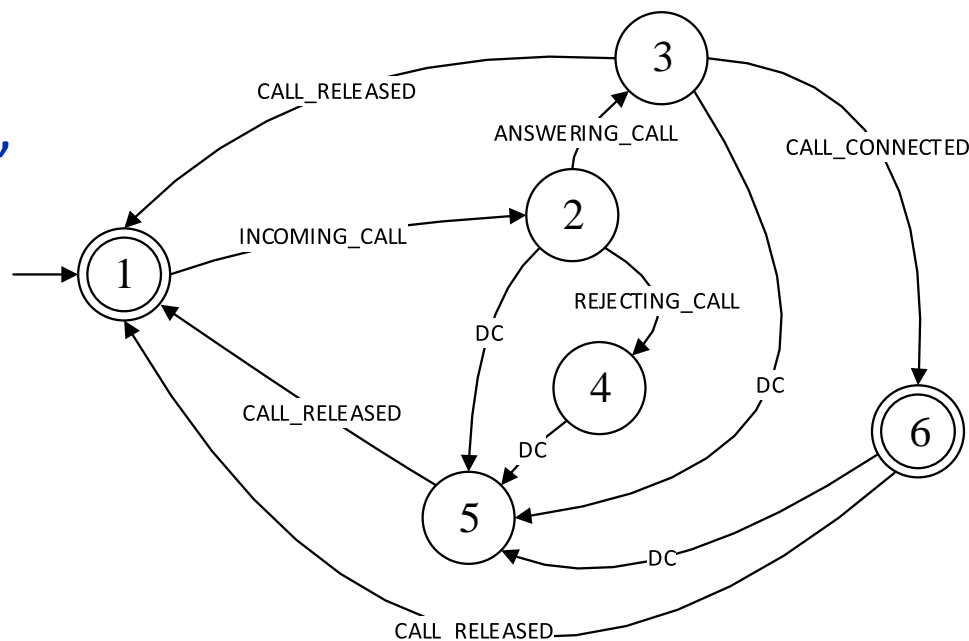
```
int: n;  
int: weight_max;  
array [1..n] of int: values;  
array [1..n] of int: weights;  
array [1..n] of var int: take;  
  
var int: profit = sum(i in 1..n) (take[i]*values[i]);  
solve maximize profit;  
  
constraint  
  forall(i in 1..n) (take[i] >= 0)  
  /\n  knapsack(weights, take, weight_max);  
  
% data  
n = 3;  
weight_max = 10;  
values = [23, 9, 20];
```

1.2 Конечные автоматы

- Не рассматриваются вероятностные и недетерминированные модели, вопросы их генерации
- *Детерминированный конечный автомат (ДКА)* – это пятерка $\langle Q, \Sigma, \delta, q_s, F \rangle$
 - Q – конечное множество состояний
 - Σ – алфавит входных символов
 - $\delta: Q \times \Sigma \rightarrow Q$ – функция переходов
 - $q_s \in Q$ – начальное (стартовое) состояние
 - $F \subset Q$ – множество допускающих состояний
- Строка $a_1 a_2 \dots a_n$ допускается, если, начиная работу в q_s и обработав символы строки, ДКА оказывается в допускающем состоянии $q_f \in F$

Пример ДКА для протокола входящих звонков

- Используется алфавит команд {ANSWERING_CALL, CALL_CONNECTED, CALL_RELEASED, DC, INCOMING_CALL, REJECTING_CALL}
- Допускающие состояния соответствуют режимам «1. Бездействие» и «6. Разговор»
- Моделирует протокол связи с высоким уровнем абстракции [74]



[74] Digital CAS Protocols Installation and Developer's Manual: Responding to Inbound Calls. — URL: <http://www.nmscommunications.com/manuals/6206-14/C-09.htm>.

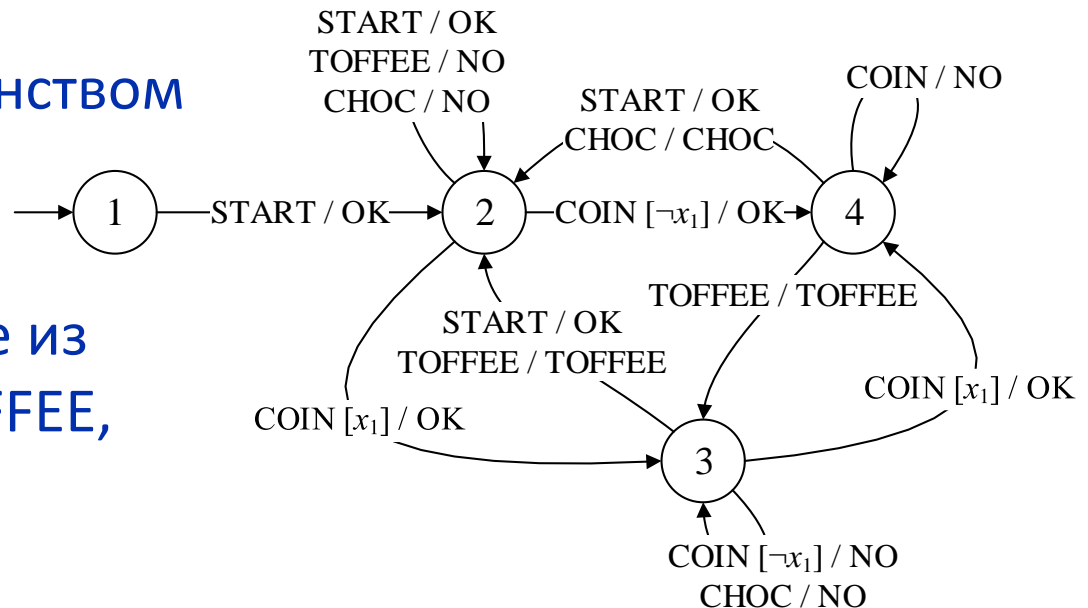


Управляющие конечные автоматы

- Управляющий автомат – каждый переход помечен **событием**, последовательностью **выходных воздействий**, а также **охранным условием**, представляющим собой булеву формулу от входных переменных
- Автомат Мили
- Семерка $\langle Q, \Sigma, Z, X, \delta, \lambda, q_s \rangle$
 - Z – множество выходных воздействий
 - X – множество булевых входных переменных
 - $\delta: Q \times \Sigma \times 2^X \rightarrow Q$
 - $\lambda: Q \times \Sigma \times 2^X \rightarrow Z^*$
- Ключевая модель в автоматном программировании

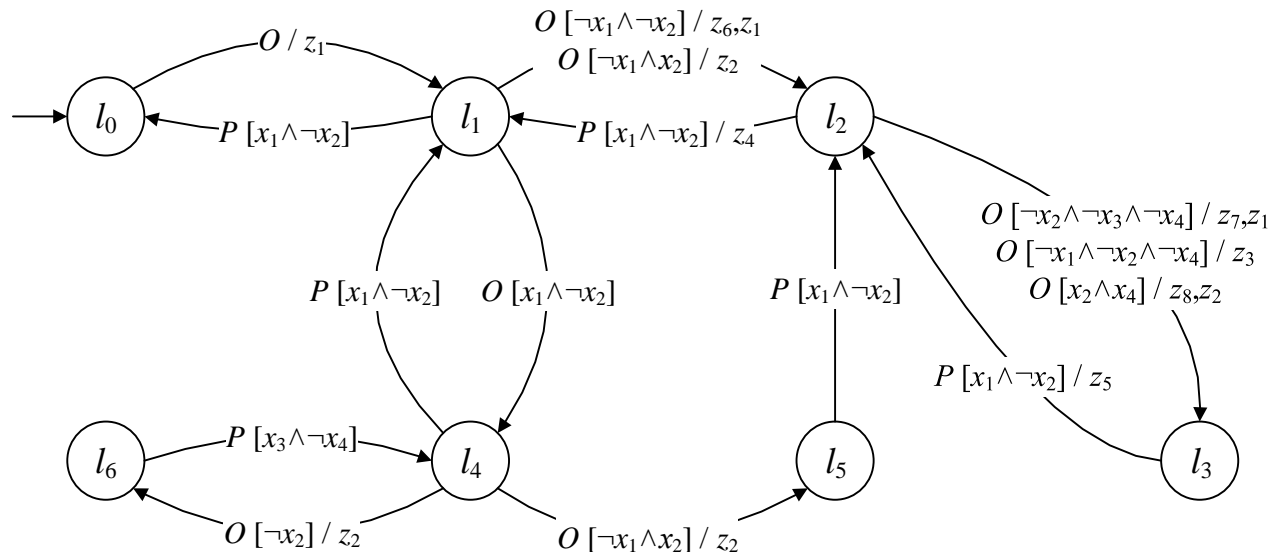
Пример 1: торговый аппарат

- События {CHOC, COIN, START, TOFFEE}
- x_1 для монеты достоинством 1, $\neg x_1$ для монеты достоинством 2
- выходное воздействие из множества {CHOC, TOFFEE, OK, NO}



Пример 2: очередь с приоритетами из стандартной библиотеки *Java*

- события O и P соответствуют вызовам функций «offer(p)» и «poll(p)»;
- $\{x_1 = (p = v_1), x_2 = (p > v_1), x_3 = (p = v_2), x_4 = (p > v_2)\}$
- выходные воздействия соответствуют операциям над внутренними переменными





Сценарии работы

- Сценарий работы – последовательность T_1, \dots, T_n троек $T_i = \langle e_i; f_i; A_i \rangle$
 - e_i – входное событие
 - f_i – булева формула от входных переменных, задающая охранное условие
 - A_i – последовательность выходных воздействий
- Примеры для торгового аппарата
 - $\langle \text{START}; 1; \text{OK} \rangle, \langle \text{COIN}; x_1; \text{OK} \rangle, \langle \text{COIN}; \neg x_1; \text{NO} \rangle, \langle \text{COIN}; x_1; \text{OK} \rangle$
 - $\langle \text{START}; 1; \text{OK} \rangle, \langle \text{START}; 1; \text{OK} \rangle, \langle \text{TOFFEE}; 1; \text{NO} \rangle, \langle \text{CHOC}; 1; \text{NO} \rangle$
 - $\langle \text{START}; 1; \text{OK} \rangle, \langle \text{COIN}; \neg x_1; \text{OK} \rangle, \langle \text{TOFFEE}; 1; \text{TOFFEE} \rangle$
- Аналог слова для ДКА

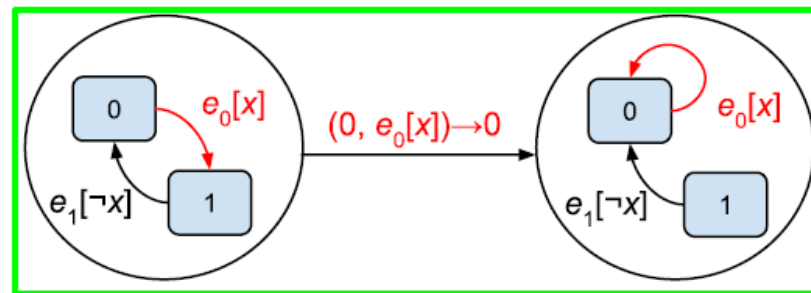
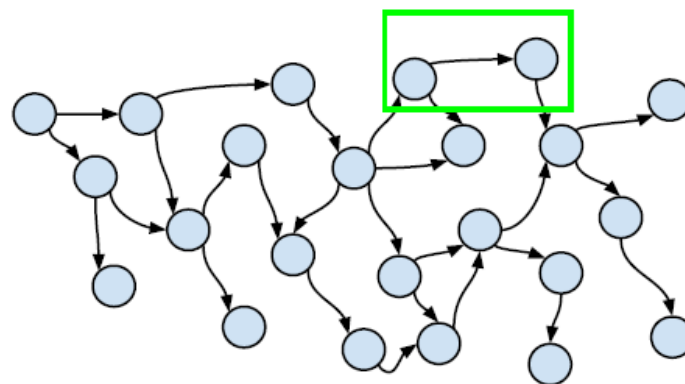


1.3 Методы генерации автоматов

- Генерация ДКА по обучающим словарям
 - Эвристические алгоритмы State merging
 - Метаэвристические методы
 - Использование программных средств решения NP-полных задач (SAT и раскраска графов)
- Генерация управляющих автоматов
 - Генетические, эволюционные, муравьиные алгоритмы
 - На кафедре «Компьютерные технологии» Университета ИТМО

Метаэвристические алгоритмы

- Эволюционные стратегии, генетические алгоритмы, муравьиные алгоритмы
- Популяция особей, мутации, скещивание, функция приспособленности
- Устойчивость к шуму



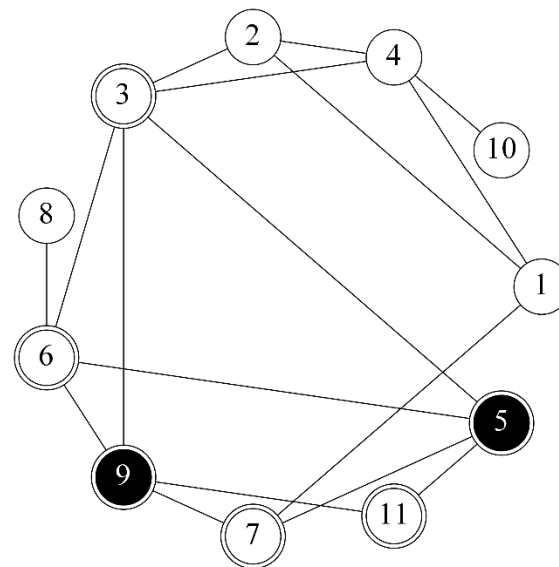
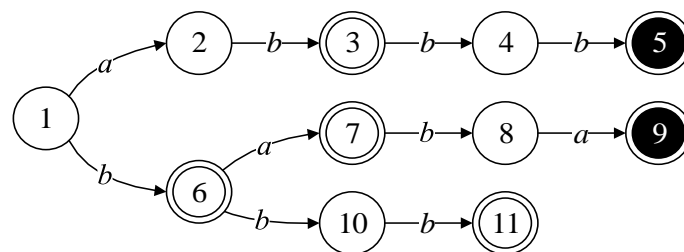
Чивилихин Д.С. Генерация конечных автоматов на основе муравьиных алгоритмов

[26] Chivilikhin D., Ulyantsev V. Learning Finite-State Machines with Ant Colony Optimization // Swarm Intelligence. – Springer, 2012. – P. 268-275. – (Lecture Notes in Computer Science; 7461).

[56] Lucas S., Reynolds J. Learning Finite State Transducers: Evolution versus Heuristic State Merging // IEEE Transactions on Evolutionary Computation. – 2007. – Vol. 11, no. 3. – P. 308-325

Использование средств решения SAT

1. Построение префиксного дерева
2. Построение графа совместимости вершин префиксного дерева
3. Построение булевой КНФ-формулы для числа состояний ДКА C (которое увеличивается в случае невыполнимости формулы)
4. Запуск средства решения SAT для нахождения выполняющей подстановки построенной КНФ-формулы
5. Построение искомого автомата по подстановке



Пропозициональное кодирование

Обязательная часть

$$\begin{aligned} \mathcal{F}_1 &= \bigwedge_{v \in V} \text{ALO} \left(\{x_{v,i}\}_{i=1}^C \right) \wedge \bigwedge_{v \in V_+, 1 \leq i \leq C} (x_{v,i} \Rightarrow z_i) \wedge \bigwedge_{v \in V_-, 1 \leq i \leq C} (x_{v,i} \Rightarrow \neg z_i) \\ &\wedge \bigwedge_{v \in V, 1 \leq i, j \leq C} (x_{p(v),i} \wedge x_{v,j} \Rightarrow y_{l(v),i,j}) \wedge \bigwedge_{a \in \Sigma, 1 \leq i \leq C} \text{AMO} \left(\{y_{a,i,j}\}_{j=1}^C \right) \end{aligned}$$

Дополнительная часть

$$\begin{aligned} \mathcal{F}_2 &= \bigwedge_{v \in V} \text{AMO} \left(\{x_{v,i}\}_{i=1}^C \right) \wedge \bigwedge_{a \in \Sigma, 1 \leq i \leq C} \text{ALO} \left(\{y_{a,i,j}\}_{j=1}^C \right) \\ &\wedge \bigwedge_{v \in V, 1 \leq i, j \leq C} (x_{p(v),i} \wedge y_{l(v),i,j} \Rightarrow x_{v,j}) \wedge \bigwedge_{(v,w) \in \mathcal{G}, 1 \leq i \leq C} (x_{v,i} \Rightarrow \neg x_{w,i}) \end{aligned}$$



Недостатки существующих методов

- ✓ **Быстродействие** существующих точных методов генерации ДКА по безошибочным обучающим словарям при большом числе состояний искомого автомата
- ✓ Качественным недостатком этих методов является **невозможность их работы при наличии ошибок** в метках допуска/недопуска заданных слов
- ✓ **Неточность** существующих методов генерации управляющих конечных автоматов по сценариям работы. Существующие методы основаны на метаэвристиках, применение которых позволяет решать практические задачи, но **не гарантирует** в общем случае того, что автомат будет сгенерирован за конечное время

Решаемые задачи

- 1. Задача генерации ДКА по незашумленным примерам.**
Одна из самых популярных задач грамматического вывода.
- 2. Задача генерации ДКА по примерам с ошибками в пометках слов.** Предлагалось участникам соревнования «Learning DFA from Noisy Samples» конференции GECCO 2004
- 3. Задача генерации управляющих автоматов по безошибочным сценариям работы**
- 4. Задача генерации управляющих автоматов по сценариям работы с ошибками в выходных воздействиях**

Для задачи 1 известно точное решение, для остальных нет

Глава 2. Теоретическая оценка сложности поставленных задач генерации управляющих автоматов

Известные результаты теории сложности

- В работе 1978 года Е. М. Голд доказал [41], что в общем случае задача генерации ДКА с заданным числом состояний по обучающим словарям является NP-полной
- В работе 1993 года [61] теоретическая сложность была усилена – задача нахождения ДКА с числом состояний, приближенным к заданному, также является полной в NP
- Первый результат используется для проводимого доказательства

[41] *Gold E. M.* Complexity of Automaton Identification from Given Data // Information and Control. – 1978. – Vol. 37. – P. 302-320.

[61] *Pitt L., Warmuth M. K.* The Minimum Consistent DFA Problem Cannot be Approximated Within any Polynomial // Journal of the ACM. – 1993. – Vol. 40, no. 1. – P. 95-142.

Язык задачи

- Докажем трудность задач генерации управляющих автоматов в классе NP
- L_{EFSM} – множество пар $\langle S, C \rangle$, для которых существует управляющий автомат $\mathcal{A}_{\text{EFSM}}$ размера C , удовлетворяющий сценариям из S
 - S – набор сценариев работы программы
 - C – натуральное число
- К поставленной задаче можно **свести задачу построения ДКА** с заданным числом состояний по заданным обучающим словарям с языком L_{DFA}
 - множество троек $\langle S^+, S^-, C \rangle$ таких, что существует конечный ДКА размера C , который принимает слова из словаря S^+ , не принимает слова из словаря S^-

Доказательство

- Доказано, что если $w = \langle S^+, S^-, C \rangle$ принадлежит L_{DFA} , то $f(w) = \langle S, C \rangle$ принадлежит языку L_{EFSM}
- Доказано, что если $f(w)$ принадлежит языку L_{EFSM} , то w принадлежит языку L_{DFA}
- Следовательно существует функция $f(w)$ такая, что w принадлежит L_{DFA} **тогда и только тогда**, когда элемент $f(w)$ принадлежит L_{EFSM}

⇒ NP-трудность

Глава 3. Генерация детерминированных конечных автоматов по обучающим словарям

3.1. Метод генерации по безошибочным обучающим словарям

3.2. Метод генерации по зашумленным обучающим словарям

3.3. Реализация и экспериментальные исследования
разработанных методов генерации

Первый разработанный метод

function NoiselessDFAGeneration(S_+, S_-)

S_+, S_- – непересекающиеся наборы слов

$\mathcal{T} \leftarrow \text{APTA}(S_+, S_-)$

$\mathcal{G} \leftarrow \text{consistencyGraph}(\mathcal{T})$

clique $\leftarrow \text{largeClique}(\mathcal{G})$

for $C \leftarrow |\text{clique}|..|\mathcal{T}|$ **do**

$f_{\text{SAT}} \leftarrow \mathcal{F}_{\text{SAT}}(\mathcal{T}, \mathcal{G}, C)$

solution $\leftarrow \text{solveSAT}(f_{\text{SAT}})$

if solution $\neq \{\}$ **then**

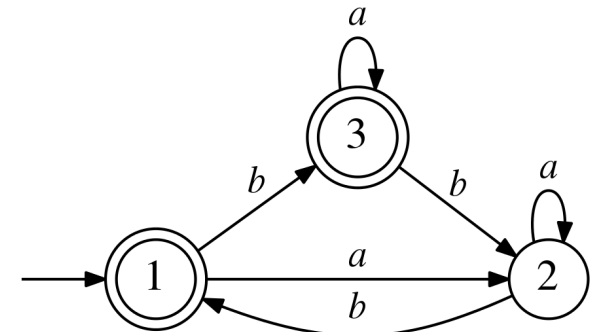
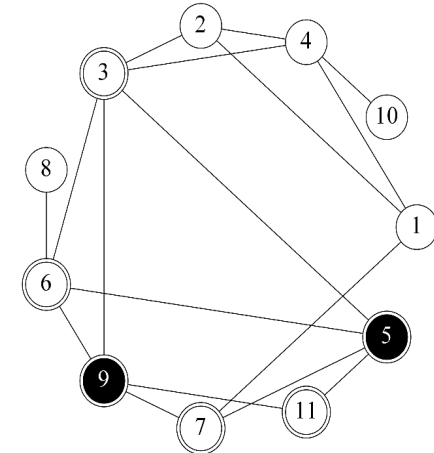
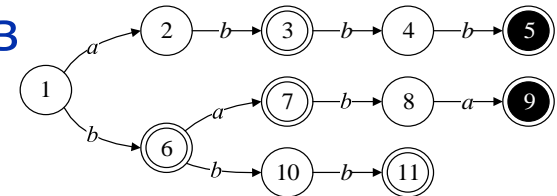
$\mathcal{A} \leftarrow \text{DFA}(C, \mathcal{T}, \text{solution})$

return \mathcal{A}

end if

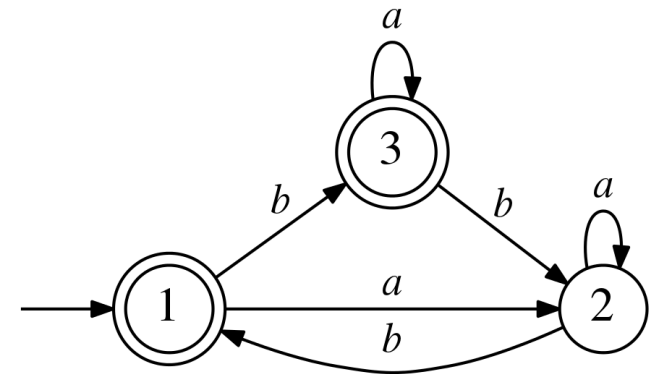
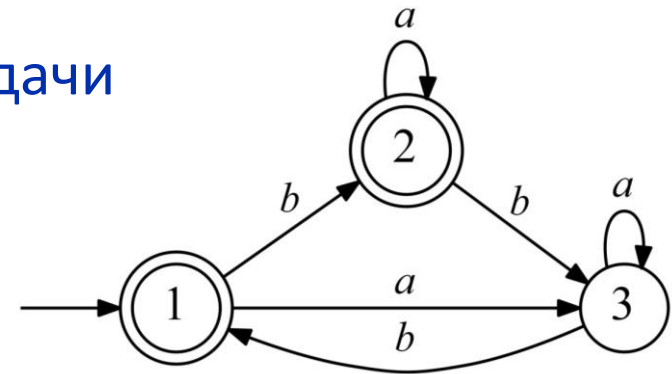
end for

end function



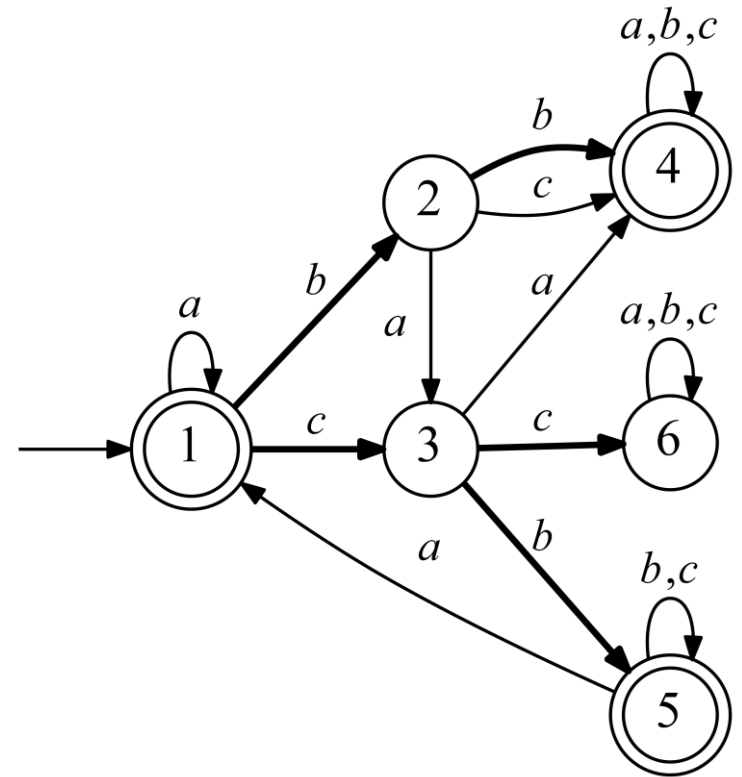
Предикаты нарушения симметрии

- Используются для **сокращения пространства поиска** поставленной задачи
- Если формула выполнима, то после добавления предикатов \mathcal{F}_{SB} **останется выполнимой**
- **Минимум симметричных решений** (а в идеале ровно одно) должно соответствовать \mathcal{F}_{SB}
- Для генерации автоматов с нумерованными состояниями естественной симметрией является **изоморфизм** автоматов



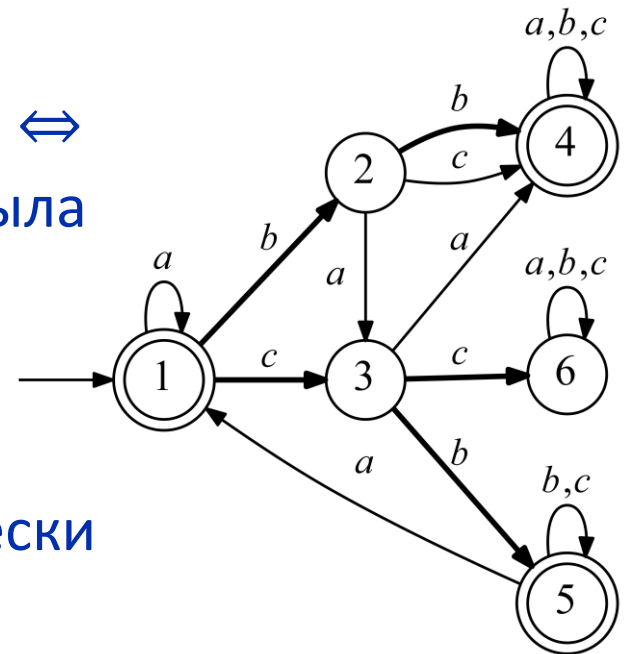
Основная идея новых предикатов

- Фиксирование номеров состояний в порядке некоторого **однозначного обхода** автомата
- Изоморфные решения не должны рассматриваться: среди автоматов для решений \mathcal{F}_{SAT} для одного найдется соответствующая выполняющая подстановка формулы $\mathcal{F}_{\text{SAT}} \wedge \mathcal{F}_{\text{SB}}$
- Обход в ширину



Дополнительные переменные

- **Переменные наличия перехода.** $t_{i,j} = 1 \Leftrightarrow$ в автомате существует хотя бы один переход из состояния i в состояние j
- **Переменные предка в обходе.** $p_{j,i} = 1 \Leftrightarrow$ при обходе ДКА в ширину вершина j была добавлена в очередь при просмотре исходящих ребер из i
- **Переменные наименьшего символа.** $m_{l,i,j} = 1 \Leftrightarrow l$ является лексикографически наименьшим символом, который встретился на данных переходах $i \xrightarrow{l} j$



Дополнительные предикаты

$$\mathcal{F}_{\text{SB}} = \mathcal{F}_t \wedge \mathcal{F}_p \wedge \mathcal{F}_{\text{ALO}(p)} \wedge \mathcal{F}_{\text{BFS}(p)} \wedge \mathcal{F}_m \wedge \mathcal{F}_{\text{BFS}(m)}$$

$$\mathcal{F}_t = \bigwedge_{1 \leq i < j \leq c} (t_{i,j} \Leftrightarrow \bigvee_{l \in \Sigma} y_{l,i,j});$$

$$\mathcal{F}_p = \bigwedge_{1 \leq i < j \leq c} (p_{j,i} \Leftrightarrow t_{i,j} \wedge \bigwedge_{1 \leq k < i} \neg t_{k,j});$$

$$\mathcal{F}_{\text{ALO}(p)} = \bigwedge_{1 < j \leq c} \text{ALO}(\{p_{j,i}\}_{i=1}^{j-1});$$

$$\mathcal{F}_{\text{BFS}(p)} = \bigwedge_{1 \leq k < i < j < c} (p_{j,i} \Rightarrow \neg p_{j+1,k});$$

$$\mathcal{F}_m = \bigwedge_{1 \leq i < j \leq c, l \in \Sigma} (m_{l,i,j} \Leftrightarrow y_{l,i,j} \wedge \bigwedge_{l^* \in \Sigma, l^* < l} \neg y_{l^*,i,j});$$

$$\mathcal{F}_{\text{BFS}(m)} = \bigwedge_{1 \leq i < j < c} \bigwedge_{l, l^* \in \Sigma, l^* < l} (p_{j,i} \wedge p_{j+1,i} \wedge m_{l,i,j} \Rightarrow \neg m_{l^*,i,j+1}).$$

Второй разработанный метод

function NoisyDFAGeneration(S_+, S_-, K)

S_+, S_- – непересекающиеся наборы слов

K – допустимое число ошибочных меток слов

$\mathcal{T} \leftarrow \text{АПТА}(S_+, S_-)$

for $C \leftarrow 1..|\mathcal{T}|$ **do**

$f_{\text{SAT}} \leftarrow \mathcal{F}_{\text{SAT}}(\mathcal{T}, C, K)$

solution $\leftarrow \text{solveSAT}(f_{\text{SAT}})$

if solution $\neq \{\}$ **then**

$\mathcal{A} \leftarrow \text{DFA}(C, \mathcal{T}, \text{solution})$

return \mathcal{A}

end if

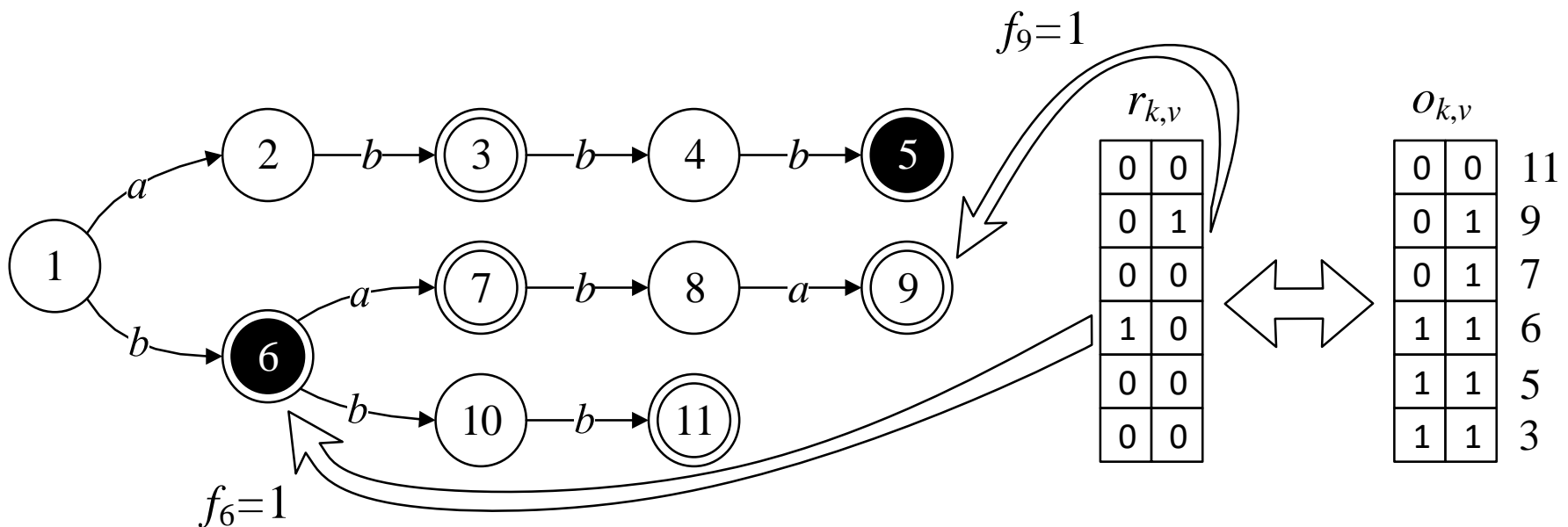
end for

end function

метки в вершинах префиксного дерева
могут быть ошибочными \Rightarrow
граф совместимости \mathcal{G} строить не
целесообразно

Идея обработки ошибочных пометок

Для каждой вершины префиксного дерева T с допускающей или недопускающей меткой будем хранить булеву переменную, истинность которой допускает ошибочность метки





Предикаты обработки ошибочных пометок

$$\mathcal{F}_{\text{NOISY}} = \mathcal{F}_{fr} \wedge \mathcal{F}_o \wedge \mathcal{F}_{\text{AMO}(r)}$$

$$\mathcal{F}_{fr} = \bigwedge_{v \in V_{\pm}} (f_v \Leftrightarrow \bigvee_{1 \leq k \leq K} r_{k,v});$$

$$\mathcal{F}_o = \bigwedge_{1 \leq k \leq K, 1 \leq j < W} (o_{k,v_{j+1}} \Rightarrow o_{k,v_j}) \wedge \bigwedge_{1 \leq k < K, 1 \leq j < W} (o_{k,v_j} \Rightarrow o_{k+1,v_{j+1}});$$

$$\mathcal{F}_{\text{AMO}(r)} = (r_{K,v_W} \Leftrightarrow o_{K,v_W}) \wedge \bigwedge_{1 \leq k \leq K, 1 \leq j < W} (r_{k,v_j} \Leftrightarrow o_{k,v_j} \wedge \neg o_{k,v_{j+1}});$$

Структура разработанного инструментального средства DFAInducer

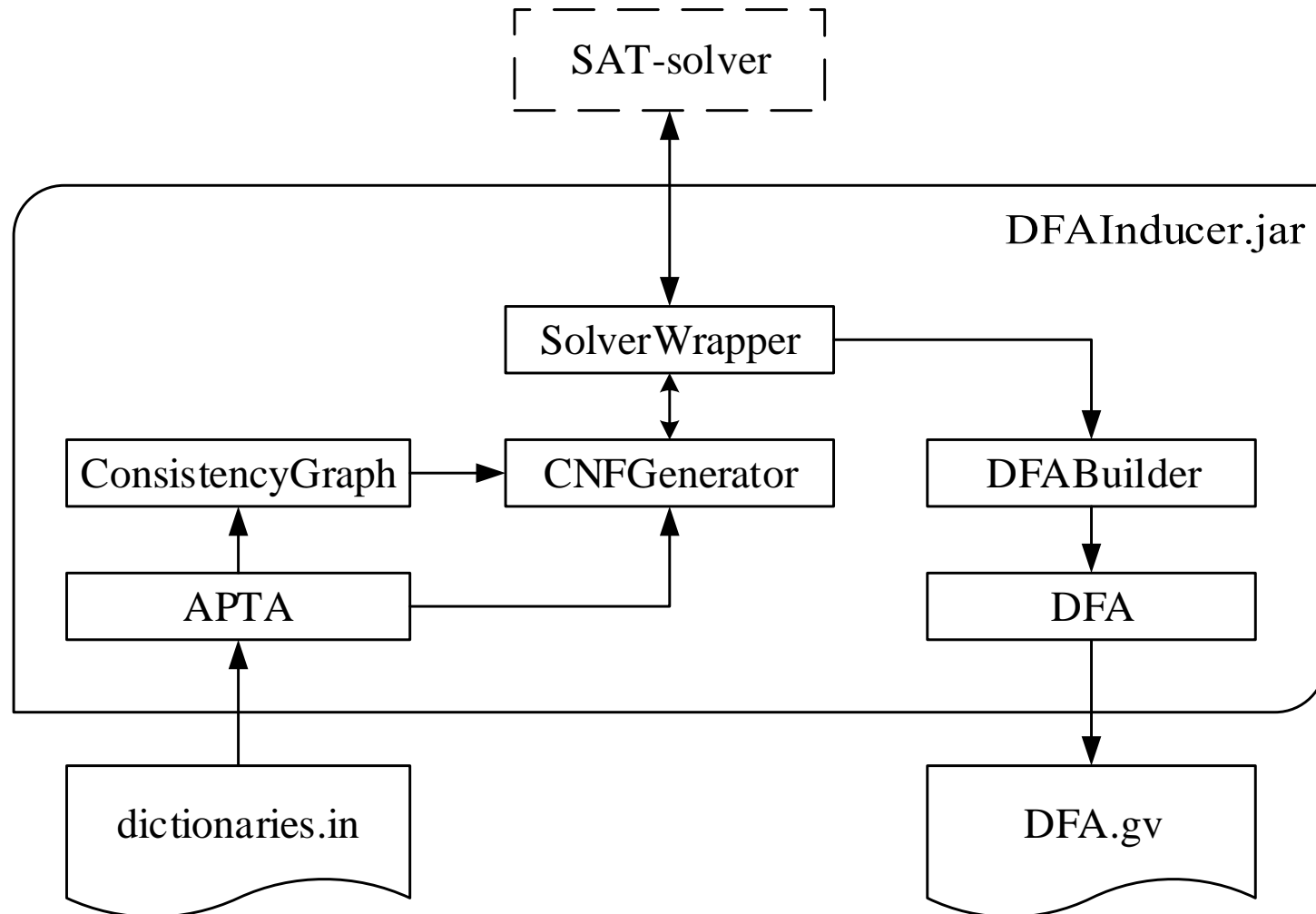
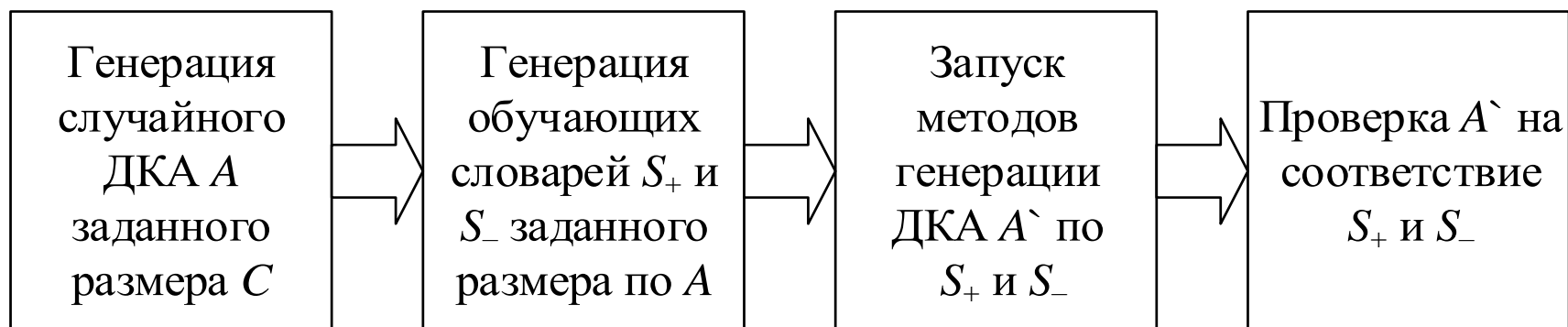


Схема проведения экспериментов

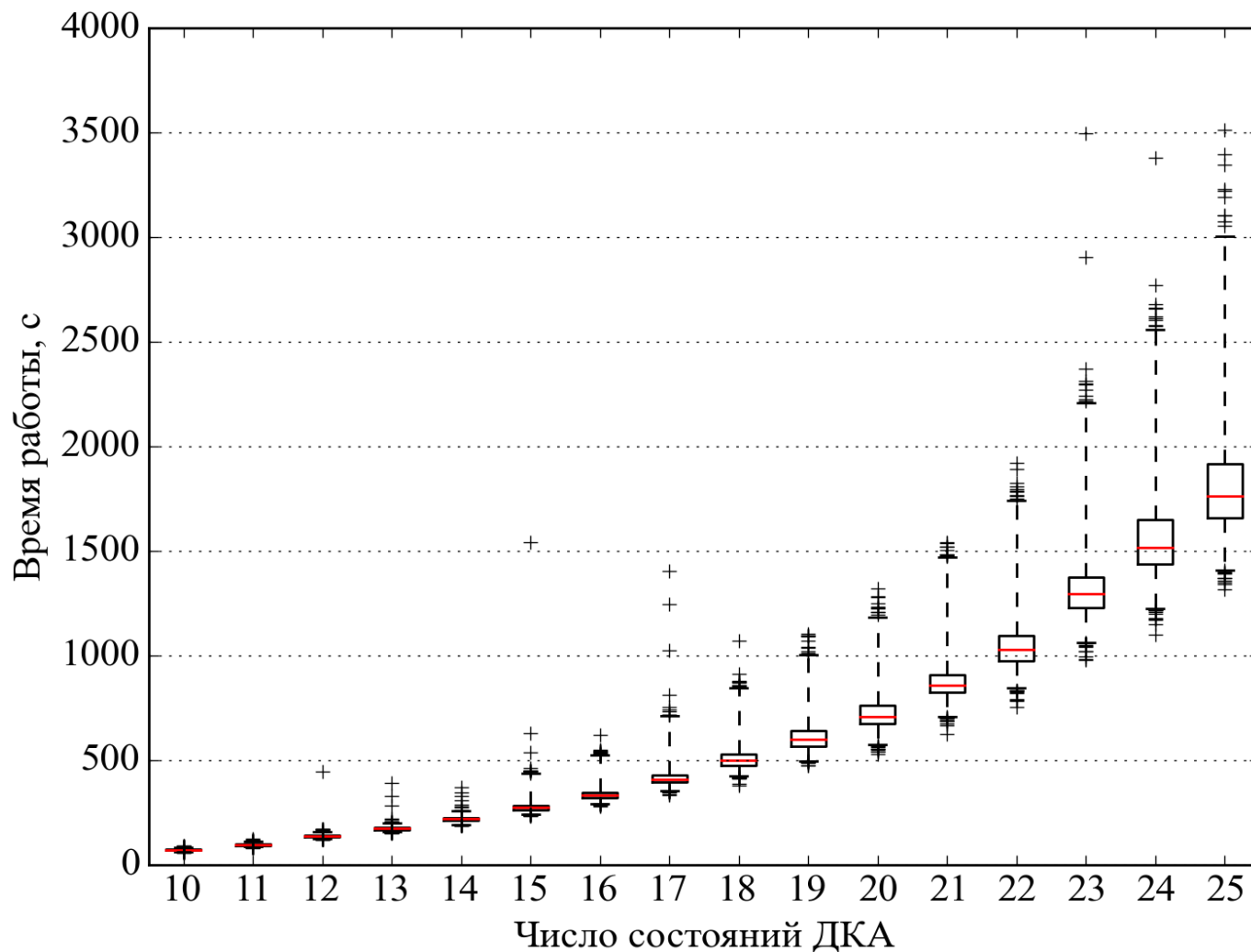


Сравнение с DFASAT

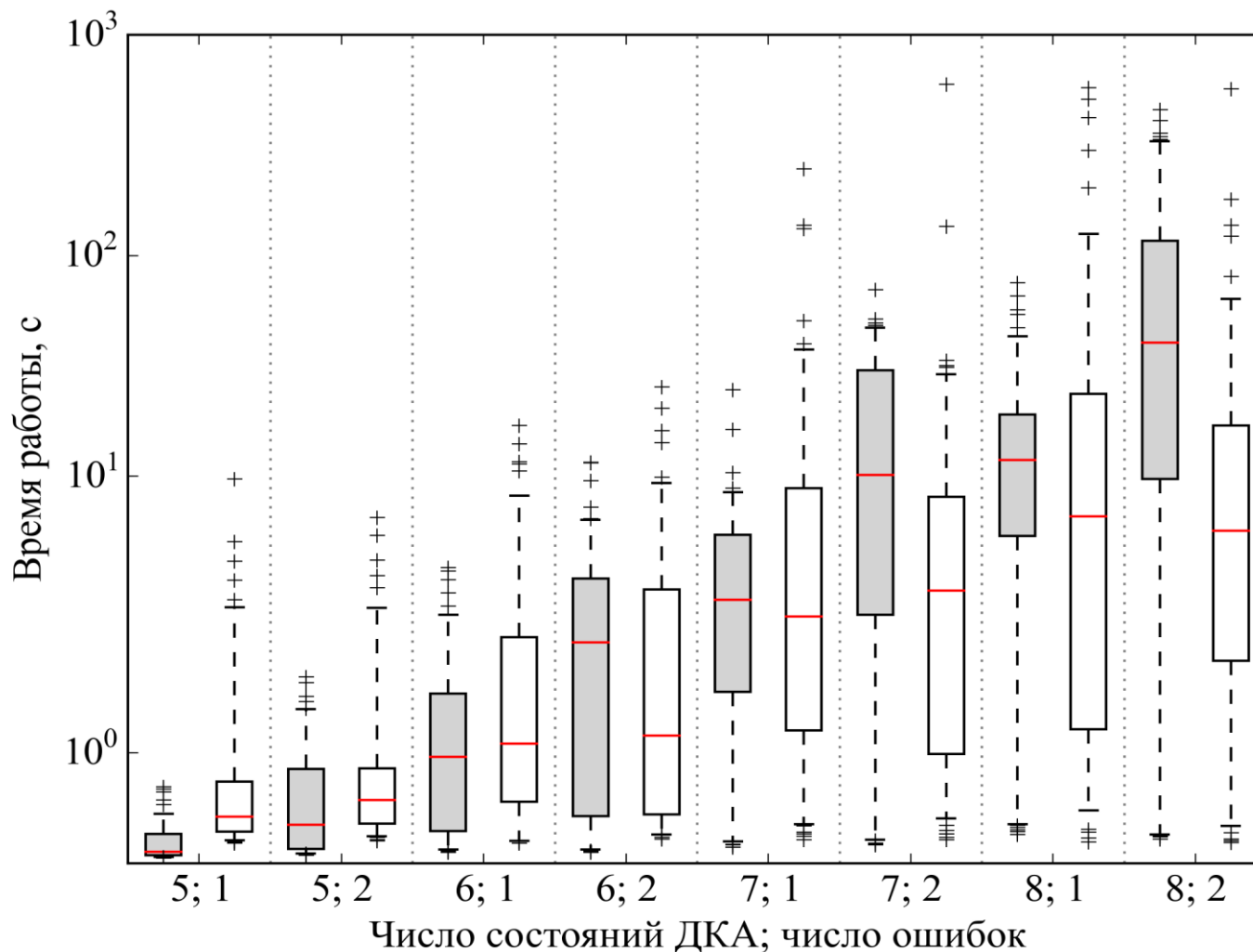
C	$ \mathcal{T} $	$ \text{clique} $	$T_1, \text{с}$	$T_2, \text{с}$
10	1351	5	71,4	100,7
11	1495	5	96,4	166,3
12	1644	5	136,8	751,2
13	1789	6	174,4	3600*
14	1928	6	219,2	3600*
15	2067	6	273,3	3600*
16	2200	6	333,4	3600*
17	2329	6	410,1	3600*
18	2455	6	500,4	3600*
19	2581	7	598,2	3600*
20	2703	7	708,8	3600*



Производительность первого метода



Экспериментальные сравнения второго метода с МуАСО



Глава 4. Генерация управляющих конечных автоматов по сценариям работы

4.1. Метод генерации управляющих конечных автоматов по безошибочным сценариям работы

4.2. Метод генерации по зашумленным сценариям работы

4.3. Реализация и экспериментальные исследования методов

Третий разработанный метод

function NoiselessEFSMGeneration(S)

S – набор непротиворечивых сценариев работы

$\mathcal{T} \leftarrow \text{scenariosTree}(S)$

$\mathcal{G} \leftarrow \text{consistencyGraph}(\mathcal{T})$

for $C \leftarrow 1..|\mathcal{T}|$ **do**

$f_{\text{CSP}} \leftarrow \mathcal{F}_{\text{CSP}}(\mathcal{T}, \mathcal{G}, C)$

solution $\leftarrow \text{solveCSP}(f_{\text{CSP}})$

if solution $\neq \{\}$ **then**

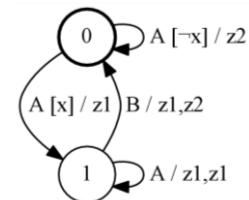
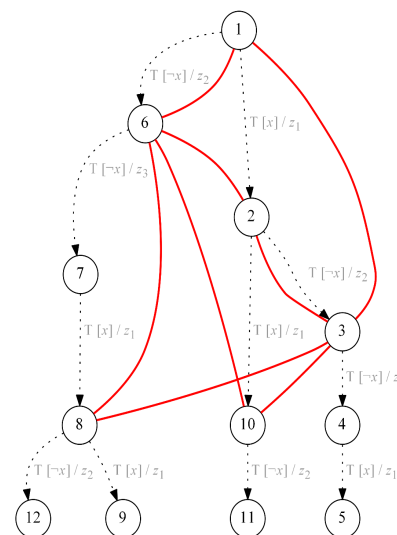
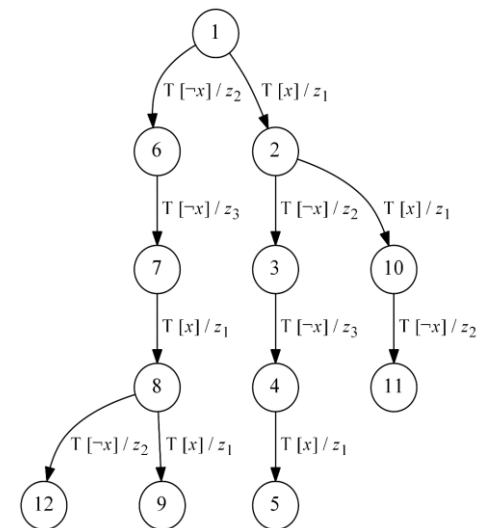
$\mathcal{A} \leftarrow \text{EFSM}(C, \mathcal{T}, \text{solution})$

return \mathcal{A}

end if

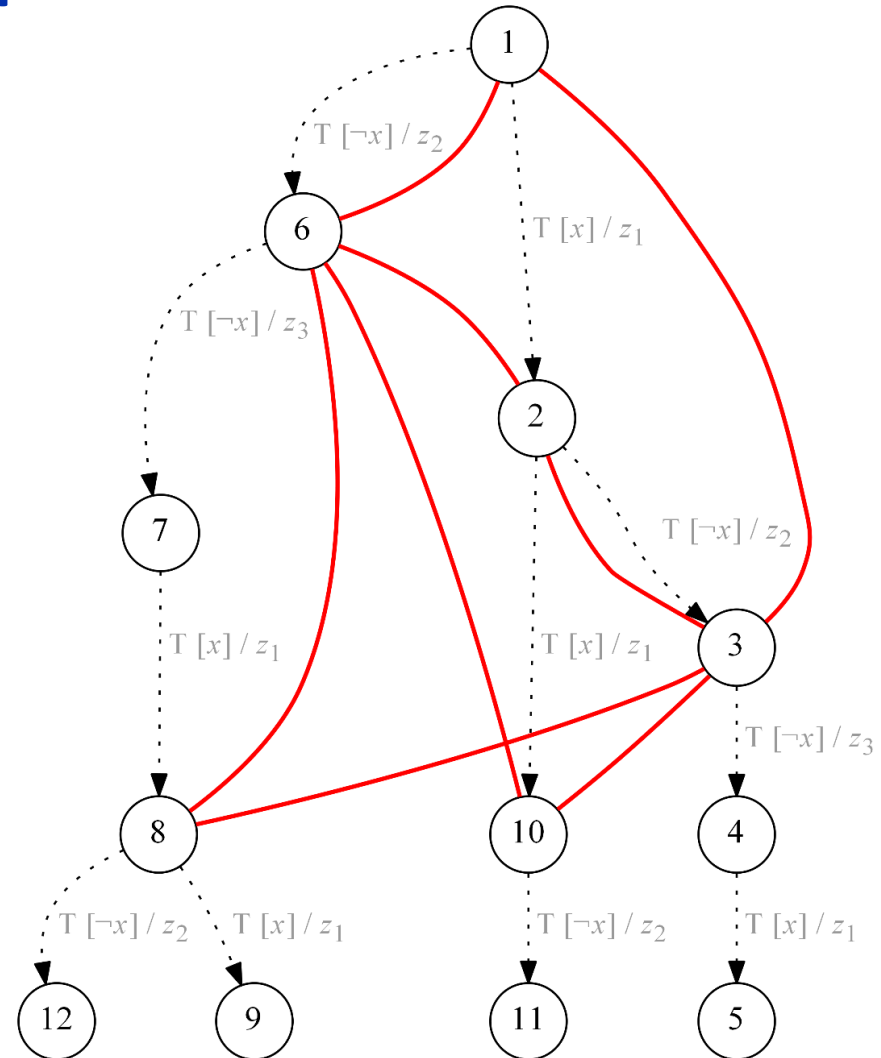
end for

end function



Построение дерева сценариев и графа совместимости

- Аналогично построению бора
- Алгоритм прерывается, если найдено противоречие
- Вершины графа совпадают с вершинами дерева
- Вершины соединены ребром, если существует последовательность, различающая их
- Используется динамическое программирование



Ограничения на целочисленные переменные

- x_v соответствуют цвету каждой вершины дерева сценариев
- Переменные переходов $y_{i,e}$
- Переменные использования переходов $u_{i,e}$
- $x_v \neq x_u$ для всех $(v, u) \in \mathcal{G}$.
- $(x_v = i) \Rightarrow (x_u = y_{i,e})$ для $i \in 1 \dots C$ и каждого ребра $(v, u) \in \mathcal{T}$.
- $(u_{i,e} = 1) \Leftrightarrow \bigvee_{v \in V(e)} (x_v = i)$ для $i \in 1 \dots C, e \in \Sigma_X$.
- $\sum_{e \in \Sigma_X: \text{event}(e) = e_o} (u_{i,e} \cdot c(e)) \in \{0, 2^m\}$ для $i \in 1 \dots C, e_o \in \Sigma$.

+ предикаты нарушения симметрии

Четвертый разработанный метод

function NoisyEFSMGeneration(S, K)

S – набор непротиворечивых сценариев работы

K – допустимое число ошибочных выходных последовательностей

for $C \leftarrow 1.. \Sigma(S)$ **do**

$f_{\text{CSP}} \leftarrow \mathcal{F}_{\text{CSP}}(S, C, K)$

solution $\leftarrow \text{solveCSP}(f_{\text{CSP}})$

if solution $\neq \{\}$ **then**

$\mathcal{A} \leftarrow \text{EFSM}(C, S, \text{solution})$

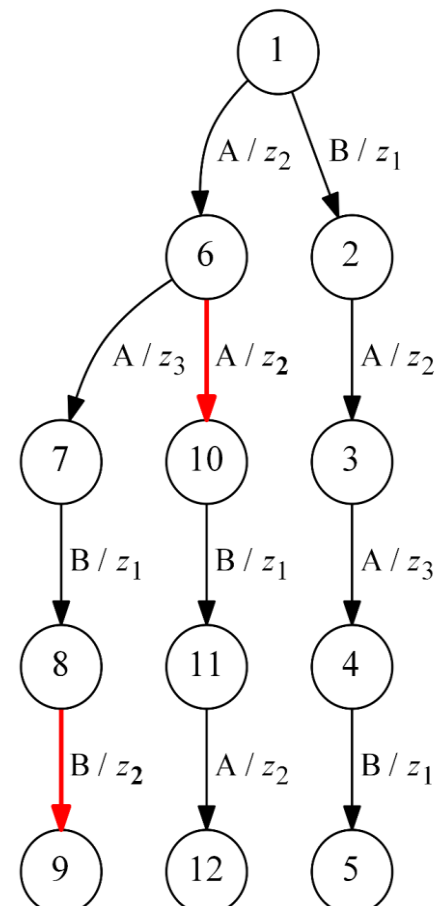
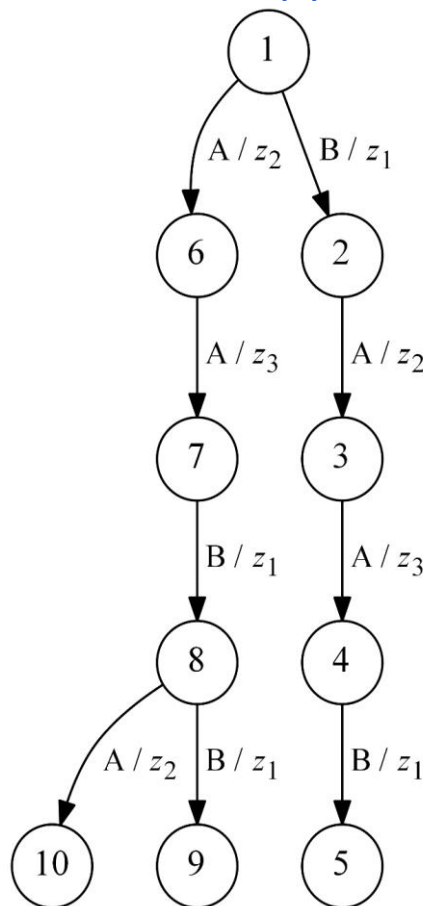
return \mathcal{A}

end if

end for

return $\{\}$

end function



Предлагаемое сведение

- $x_v = 1 \ (v \in V_b)$
- $x_{v+1} = y_{x_v, e(v)} \ (v \in V \setminus V_e)$
- $(f_v = 0) \Rightarrow (z_{x_v, e(v)} = z(v)) \ (v \in V \setminus V_e)$
- $\sum_{v \in V \setminus V_e} f_v \leq K$ – суммарное число ошибок f_v ограничено заданным числом K

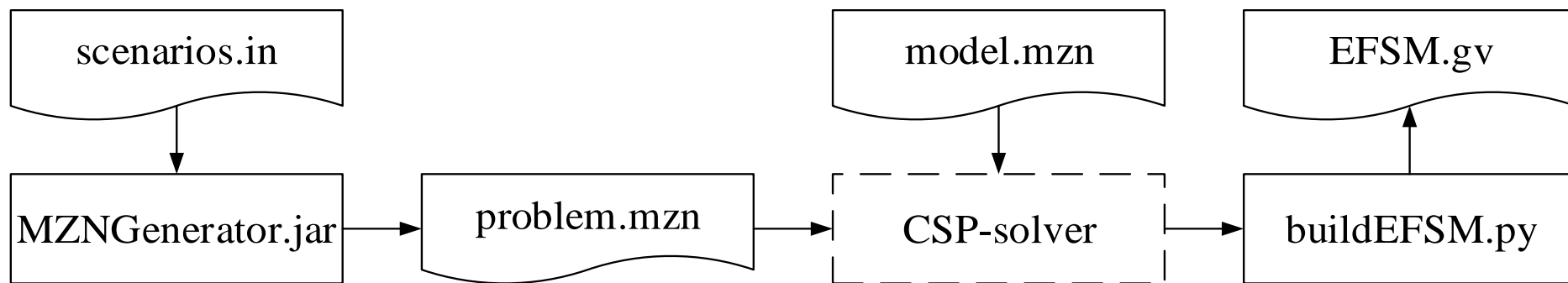
$\mathcal{O}(|V|)$ ограничений четырех типов

+ Предикаты нарушения = $\mathcal{O}(C^2 + |V|)$

Используется $\mathcal{O}(C(C + |\Sigma_X|) + |V|)$ целочисленных переменных



Схема реализованного инструментального средства EFSMTools

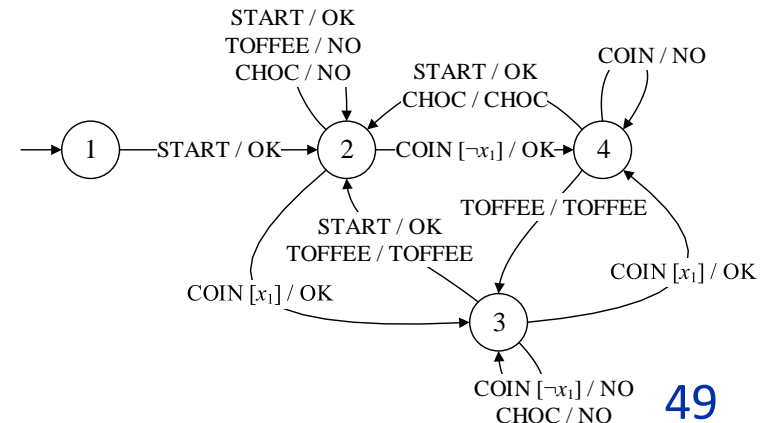


Пример применения разработанного средства к задаче генерации автомата для торгового аппарата

```
digraph EFSM {
    node [shape = circle];

    0 -> 1 [label = "START [1] (OK)"];
    1 -> 1 [label = "START [1] (OK)"];
    1 -> 1 [label = "TOFFEE [1] (NO)"];
    1 -> 1 [label = "CHOC [1] (NO)"];
    1 -> 2 [label = "COIN [x1] (OK)"];
    1 -> 3 [label = "COIN [¬x1] (OK)"];
    2 -> 2 [label = "CHOC [1] (NO)"];
    2 -> 2 [label = "COIN [¬x1] (NO)"];
    2 -> 1 [label = "START [1] (OK)"];
    2 -> 1 [label = "TOFFEE [1] (TOFFEE)"];
    2 -> 3 [label = "COIN [x1] (OK)"];
    3 -> 1 [label = "START [1] (OK)"];
    3 -> 1 [label = "CHOC [1] (CHOC)"];
    3 -> 2 [label = "TOFFEE [1] (TOFFEE)"];
    3 -> 3 [label = "COIN [1] (NO)"];
}
```

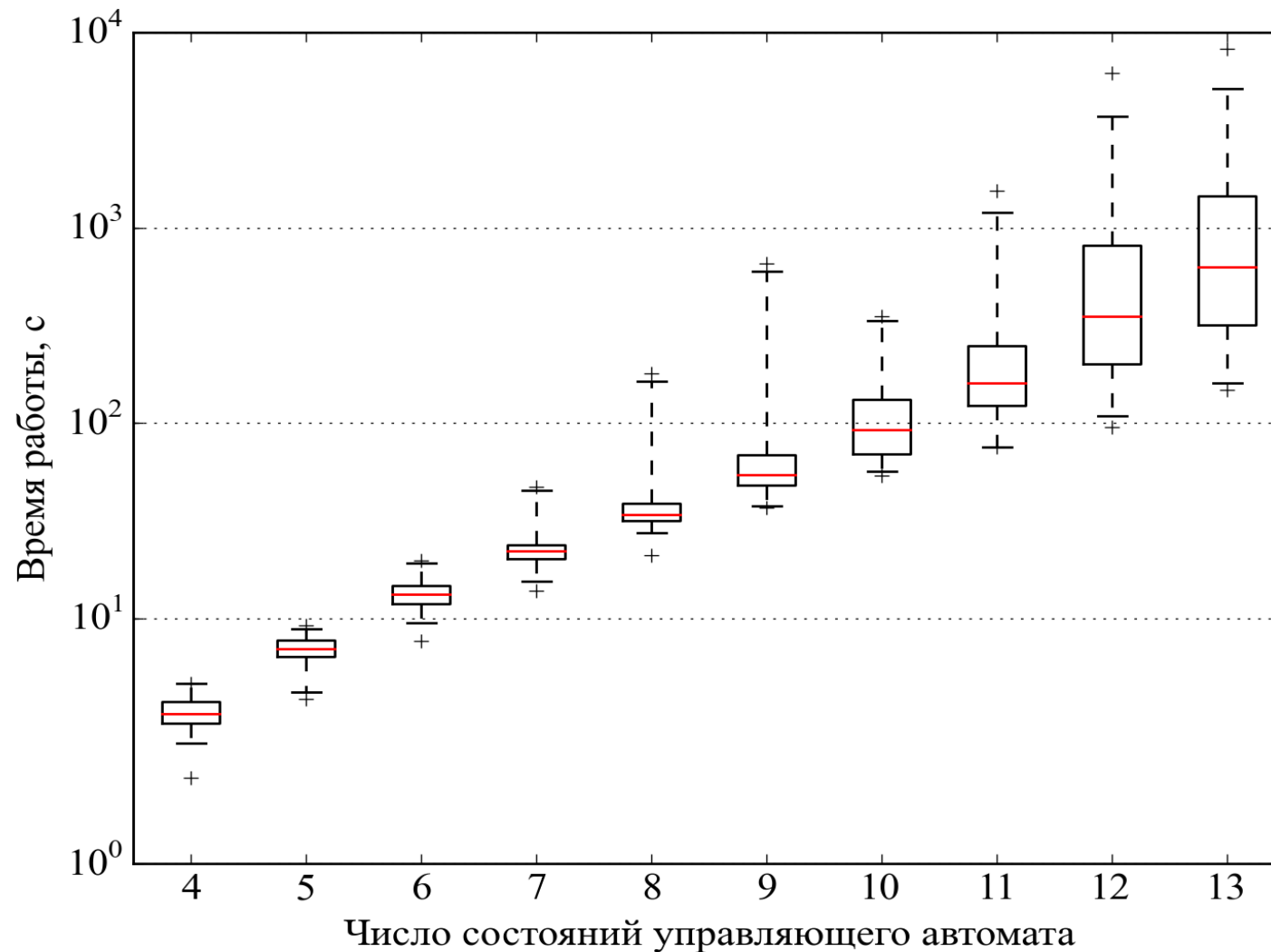
- 25 сценариев
- $|\mathcal{T}| = 340$
- 12884 ребер в \mathcal{G}
- 15240 ограничений на 1870 целочисленных переменных
- 8,7 с (с Opturion CPX)



Сравнение с переборным решением

C	$T_1, \text{с}$	$T_2, \text{с}$
4	3,2	0,3
5	7,0	3,2
6	13,3	38,3
7	22,1	600*
8	34,1	600*
9	54,5	600*
10	92,8	600*
11	160,0	1800*
12	351,3	1800*
13	632,5	1800*

Производительность третьего метода



Сравнение четвертого метода с МуАСО и поиском с возвратом

C	$K, \%$	$T_1, \text{с}$	$T_2, \text{с}$	$T_3, \text{с}$
5	1	9,7	3,8	1,2
5	2	19,6	4,1	3,3
6	1	11,0	10,6	2,1
6	2	21,2	11,5	13,9
7	1	12,9	16,5	7,5
7	2	26,5	15,2	125,9
8	1	15,5	24,4	48,9
8	2	28,2	25,4	300*
9	1	19,6	39,4	300*
9	2	30,0	52,8	300*

Глава 5. Внедрение результатов работы

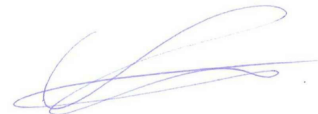
Внедрение в DFASAT

- **DFASAT** – метод и средство, разработанное М. Хеуле (Техасский университет в Остине, США) и С. Вервером (Делфтский технический университет, Нидерланды)
- Контакт установлен после выступления на *LATA-2015*
- С. Вервер реализовал предложенные автором предикаты симметрии в коде DFASAT
- На одном из примеров
 - До внедрения: 885 с
 - После внедрения: 25 с



Statement of the adoption of results of Vladimir Ulyantsev's candidate dissertation

This statement is to certify that some results of Vladimir Ulyantsev's candidate dissertation on inferring finite-state machines with SAT and CSP solvers have been used in a tool for inferring deterministic finite automata DFASAT. Namely, we used the BFS-based symmetry-breaking predicates, which significantly speeds up the algorithm on many problem instances.



03-09-2015

Dr. Sicco Verwer
Assistant Professor in Cyber Security
Delft University of Technology

Внедрение в образовательный процесс

- Произведено на кафедре «Компьютерные технологии» Университета ИТМО
- Курс «Теория автоматов и программирование»
- При руководстве двумя бакалаврскими и выполнении двух магистерских работ:
 - Закирзянов И. Т. Бакалаврская работа, 2015
 - Мельник М. В. Бакалаврская работа, 2015
 - Агапова А. И. Магистерская диссертация, 2014
 - Панченко Е. В. Магистерская диссертация, 2013

Заключение

Апробация

Свидетельства

Публикации

Гранты

Апробация результатов

- Всероссийская научная конференция по проблемам информатики СПИСОК. 2011-2014, Матмех СПбГУ
- Всероссийский конгресс молодых ученых. 2011-2013, Университет ИТМО
- Всероссийское совещание по проблемам управления. 2014, Институт проблем управления РАН, Москва
- 14th IFAC Symposium «Information Control Problems in Manufacturing – INCOM'12». 2012, Бухарест, Румыния
- International Conference on Machine Learning and Applications. 2011 – Гонолулу, США. 2014 – Детройт, США
- 9th International Conference on Language and Automata Theory and Applications. 2015, Ницца, Франция

Свидетельства о регистрации

- № 2012 616462 от 18.07.2012 г. «Программное средство для построения графа совместимости вершин дерева сценариев работы программы»
- № 2012 660438 от 20.11.2012 г. «Программное средство для построения КНФ-формулы по графу совместимости вершин дерева сценариев работы программы»
- № 2013 619840 от 17.10.2013 г. «Программный комплекс для построения и тестирования управляющих конечных автоматов»
- № 2015 619224 от 27.08.2015 г. «Программное средство преобразования полученных методами машинного обучения управляющих автоматов в формат MATLAB/Stateflow»

Публикации в ВАК

- Панченко Е. В., **Ульянцев В. И.** Применение методов решения задачи о выполнимости квантифицированной булевой функции для построения управляющих конечных автоматов по сценариям работы и темпоральным свойствам // Научно-технический вестник информационных технологий, механики и оптики. – 2013. – № 4(86). – С. 151-153. – 0,188 п. л. / 0,1 п. л.
- **Ульянцев В. И.**, Царев Ф. Н. Применение методов решения задачи о выполнимости булевой формулы для построения управляющих конечных автоматов по сценариям работы // Научно-технический вестник информационных технологий, механики и оптики. – 2012. – № 1(77). – С. 96-100. – 0,313 п. л. / 0,16 п. л.

Публикации в Scopus (БАК)

- Chivilikhin D., **Ulyantsev V.**, Shalyto A. Combining Exact And Metaheuristic Techniques For Learning Extended Finite-State Machines From Test Scenarios and Temporal Properties // Proceedings of the 13th International Conference on Machine Learning and Applications (ICMLA'14). – 2014. – P. 350-355. – 0,375 п.л. / 0,15 п. л.
- **Ulyantsev V.**, Tsarev F. Extended Finite-State Machine Induction using SAT-Solver / Proceedings of the Tenth International Conference on Machine Learning and Applications. – IEEE, 2011. – Vol. 2. – P. 346-349. – 0,25 п. л. / 0,125 п. л.
- **Ulyantsev V.**, Zakirzyanov I., Shalyto A. BFS-Based Symmetry Breaking Predicates for DFA Identification / Language and Automata Theory and Applications. – Springer, 2015. – P. 611-622. – (Lecture Notes in Computer Science; 8977). – 0,375 п. л. / 0,2 п. л.
- **Ulyantsev V.**, Tsarev F. Extended Finite-State Machine Induction using SAT-Solver / Proceedings of the 14th IFAC Symposium «Information Control Problems in Manufacturing – INCOM'12». – IFAC, 2012. – P. 512-517. – 0,375 п. л. / 0,25 п. л.

Другие публикации

- Ведерников Н. В., Демьянюк В. Ю., Кротков П. А., **Ульянцев В. И.**, Шалыто А. А. Применение методов машинного обучения для автоматизированного построения управляющих автоматов в высокоуровневых средствах проектирования систем // Труды XII Всероссийского совещания по проблемам управления ВСПУ-2014. – М.: Институт проблем управления им. В.А. Трапезникова РАН, 2014. – С. 3159-3166. – 0,5 п. л. / 0,2 п. л.
- И 7 других на конференциях в вузах Санкт-Петербурга

Гранты – руководитель

- ✓ «Разработка методов автоматизированного построения надежного программного обеспечения на основе автоматного подхода по обучающим примерам и темпоральным свойствам» (грант **РФФИ «Мой первый грант»**, 2014, 2015);
- ✓ «Построение управляющих автоматов с помощью методов решения задачи удовлетворения ограничений» (программа **«У.М.Н.И.К.»**, 2012)

Гранты – исполнитель

- ✓ «Разработка метода машинного обучения на основе алгоритмов решения задачи о выполнимости булевой формулы для построения управляющих конечных автоматов» (ФЦП «**Научные и научно-педагогические кадры инновационной России**» на 2009–2013 годы, 2011-2013);
- ✓ «Разработка методов построения управляющих конечных автоматов по обучающим примерам на основе решения задачи удовлетворения ограничений» (ФЦП «**Научные и научно-педагогические кадры инновационной России**» на 2009–2013 годы, 2012, 2013);
- ✓ «Технология разработки программного обеспечения систем управления ответственными объектами на основе методов машинного обучения и конечных автоматов» (научно-исследовательская работа в рамках **проектной части государственного задания** в сфере научной деятельности, Минобрнауки, 2014-2016)

Результаты

- ✓ Теоретическое доказательство NP-трудности задачи генерации управляющих конечных автоматов по сценариям работы.
- ✓ Точные методы генерации детерминированных конечных автоматов по безошибочным и зашумленным обучающим словарям
 - Сведение к задаче выполнимости булевой формулы
 - Безошибочные данные – ускорение по времени
 - По зашумленным данным – точных методов не известно
- ✓ Точные методы генерации управляющих конечных автоматов по безошибочным и зашумленным сценариям работы
 - Сведение к задаче удовлетворения ограничений
 - Точные методы не известны
- ✓ Внедрение разработанных методов, осуществленное при разработке программного средства *DFASAT* (Делфтский технический университет) и на кафедре «Компьютерные технологии» Университета ИТМО.

Спасибо за внимание!

Ульянцев Владимир Игоревич

Генерация конечных автоматов с использованием программных средств решения задач выполнимости и удовлетворения ограничений

Диссертация на соискание ученой степени кандидата технических наук

Специальность 05.13.11 – «Математическое и программное обеспечение
вычислительных машин, комплексов и компьютерных сетей»