

Магистерская диссертация на тему

Построение управляющих конечных автоматов по сценариям работы на основе решения задачи удовлетворения ограничений

Ульянцев В.И., гр. 6538

Научный руководитель: Шалыто А.А.

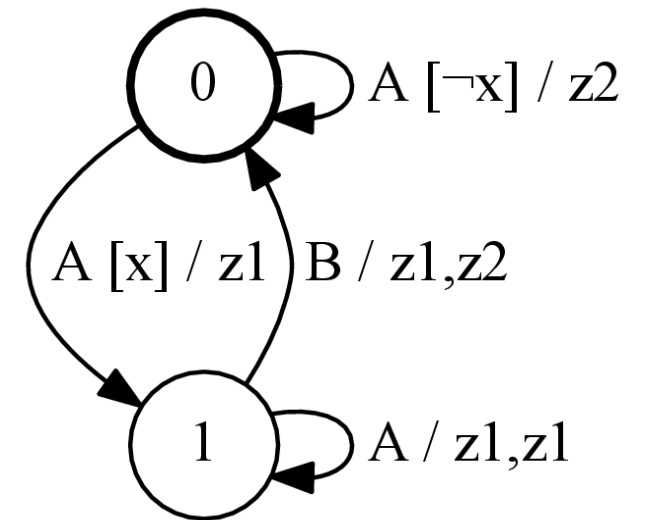
22.06.2013



Санкт-Петербург, 2013

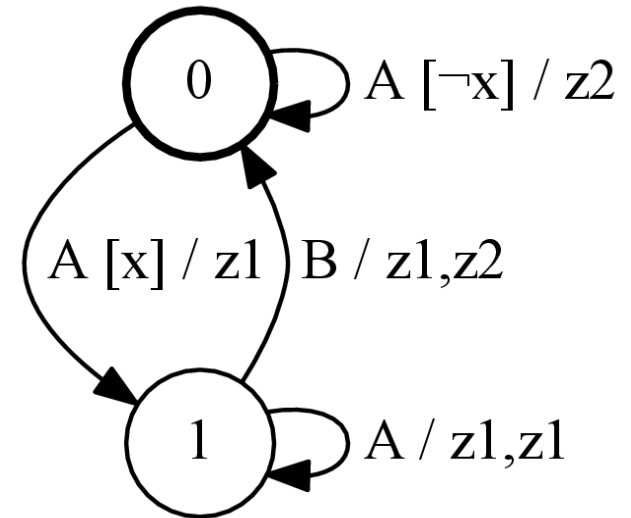
Управляющие автоматы и сценарии

- Управляющий автомат:
 - e – входное событие
 - f – охранное условие – Булева формула, зависящая от входных переменных
 - A – последовательность выходных воздействий
- Сценарий работы – последовательность троек $\langle e, f, A \rangle$



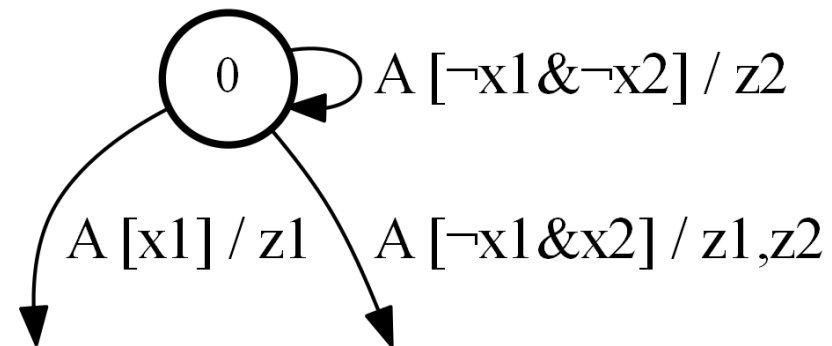
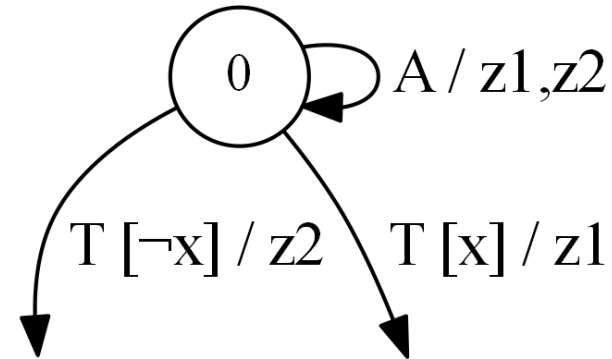
Примеры сценариев

- Приведенный управляющий автомат:
 - Удовлетворяет $\langle A, \neg x, (z2) \rangle \langle A, x, (z1) \rangle$
 - Не удовлетворяет $\langle A, x, (z2) \rangle$



Требования, предъявляемые к автоматной программе

- Непротиворечивость
- Полнота
 - «слабая»
 - «сильная»



Постановка задачи

- Входные данные:
 - Набор сценариев работы программы (S_c)
 - Число состояний управляющего автомата (C)
- Необходимо найти управляющий автомат
 - Состоящий из C состояний
 - Удовлетворяющий всем сценариям работы
 - Удовлетворяющий требованию полноты
 - Каждый переход подтвержден хотя бы одним сценарием работы

Известные решения

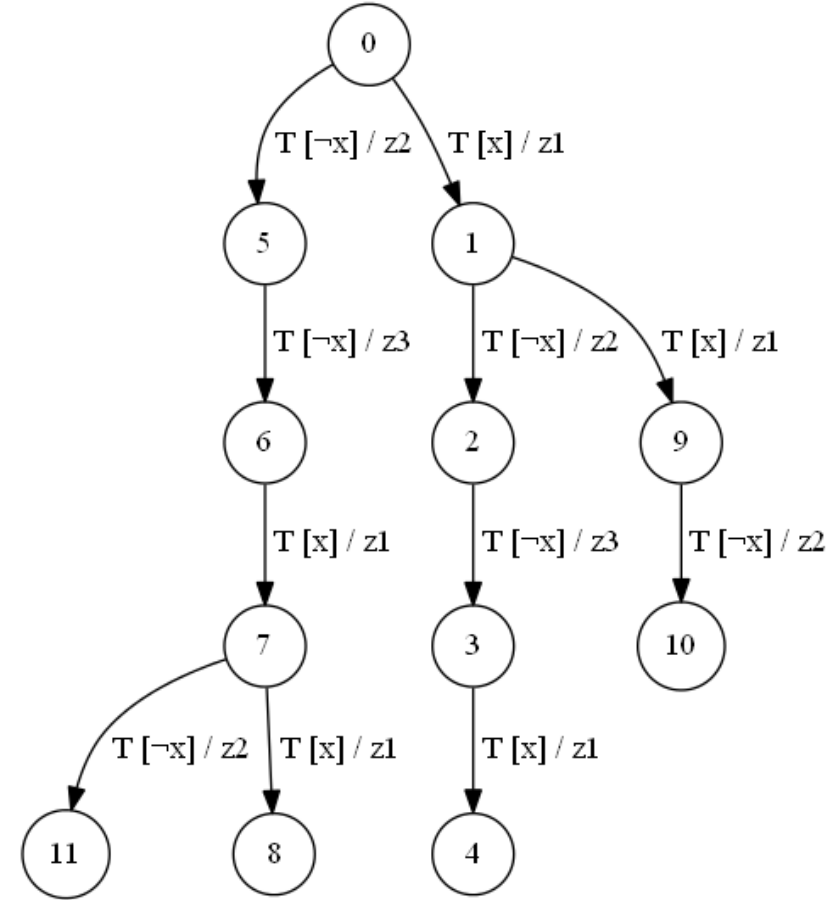
- В работах исследователей, строивших автоматы при помощи эволюционных алгоритмов, не рассматривается вопрос полноты
- Ulyantsev V., Tsarev F. Extended Finite-State Machine Induction using **SAT-Solver** / Proceedings of the Tenth International Conference on Machine Learning and Applications, ICMLA 2011, Honolulu, HI, USA, 18-21 December 2011. IEEE Computer Society, 2011. Vol. 2. P. 346-349
 - Не гарантируется полнота

Этапы работы предлагаемого алгоритма

- Построение дерева сценариев
- Построение графа совместимости
- **Построение набора ограничений на целочисленные переменные**
- Запуск сторонней программы, находящей решение
- Построение искомого управляющего автомата

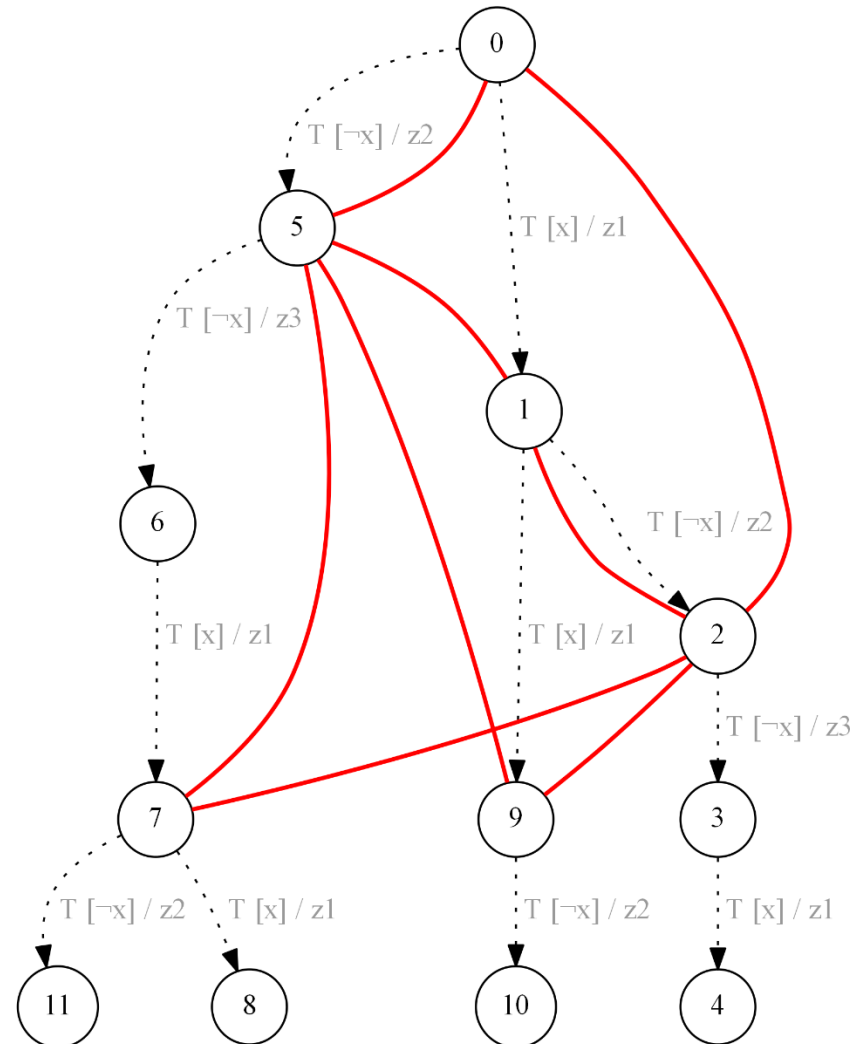
1. Построение дерева сценариев

- Аналогично построению бора
- Алгоритм прерывается, если найдено противоречие



2. Построение графа совместимости

- Вершины совпадают с вершинами дерева
- Вершины соединены ребром, если существует последовательность, различающая их
- Используется динамическое программирование



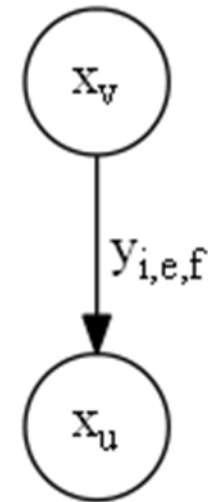
3.1. Используемые в ограничениях переменные

Целочисленные переменные:

- x_v – цвет, в который покрашена вершина v ($1 \leq x_v \leq C$)
- $y_{i,e,f}$ – номер состояния, в которое ведет переход из состояния i , помеченный событием e и охранным условием f ($1 \leq y_{i,e,f} \leq C$)

Булевы переменные:

- $z_{i,e,f}$ – существует ли вершина v в дереве сценариев, цвет которой равен i ($x_v = i$), и из нее ведет ребро, помеченное событием e и условием перехода f



3.2. Построение ограничений для требования непротиворечивости

Типы ограничений:

- $x_v \neq x_u$ – цвета несовместимых вершин должны быть различны
- $(x_v = i) \Rightarrow (y_{i,e,f} = x_u)$ – цвета вершин дерева не должны противоречить переходам автомата

3.3. Построение ограничений для требования полноты

$$z_{i,e,f} = 1 \Leftrightarrow (x_{v_1} = i \vee \dots \vee x_{v_n} = i)$$

задание значений переменных $z_{i,e,f}$

$$\left(\sum_{f \in F_e} (z_{i,e,f} \cdot c(f)) = 0 \right) \vee \left(\sum_{f \in F_e} (z_{i,e,f} \cdot c(f)) = 2^m \right)$$

«Слабая» полнота



- $c(f)$ – число выполняющих подстановок для булевой формулы f
- для любого значения входных переменных найдется переход, или ни для одного из значений переменных перехода не существует

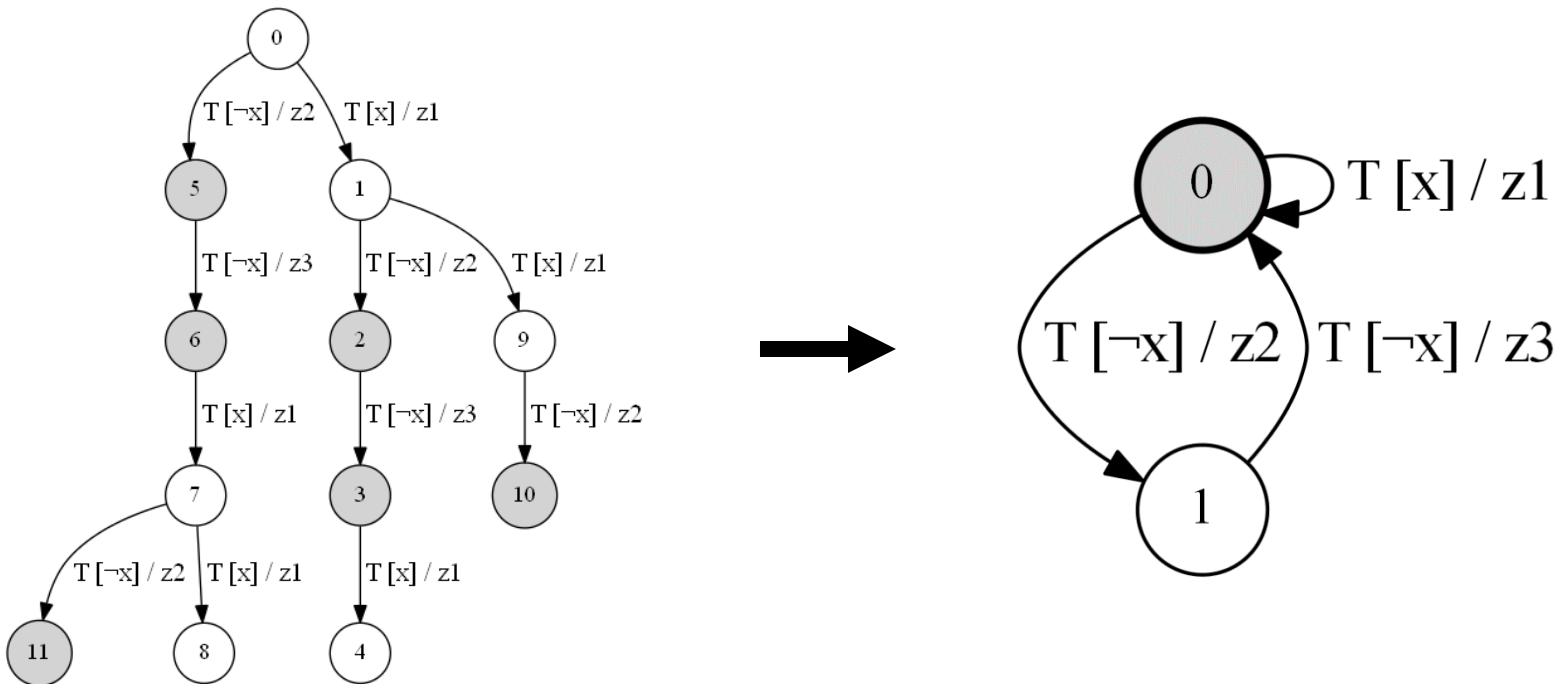
4. Сторонняя программа для нахождения значений переменных

Используются следующие методы Choco:

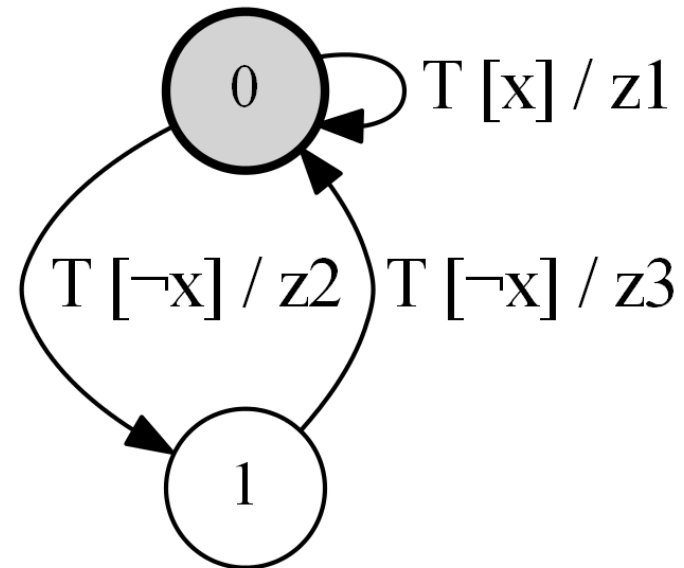
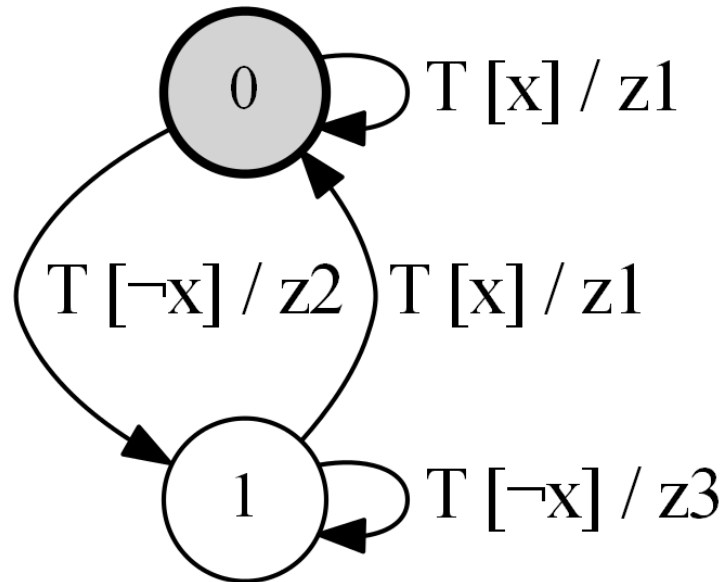
- `model.addConstraint()`
- `makeIntVarArray()`
- `eq()`, `neq()`, `or()`, `ifOnlyIf()`, `implies()`, `equation()`, `constant()`
- `solver.solve()`, `solver.read()`

5. Построение искомого автомата

- Раскраска дерева
 - По значениям переменных x , задающих цвета вершин дерева сценариев
- Слияние вершин одного цвета



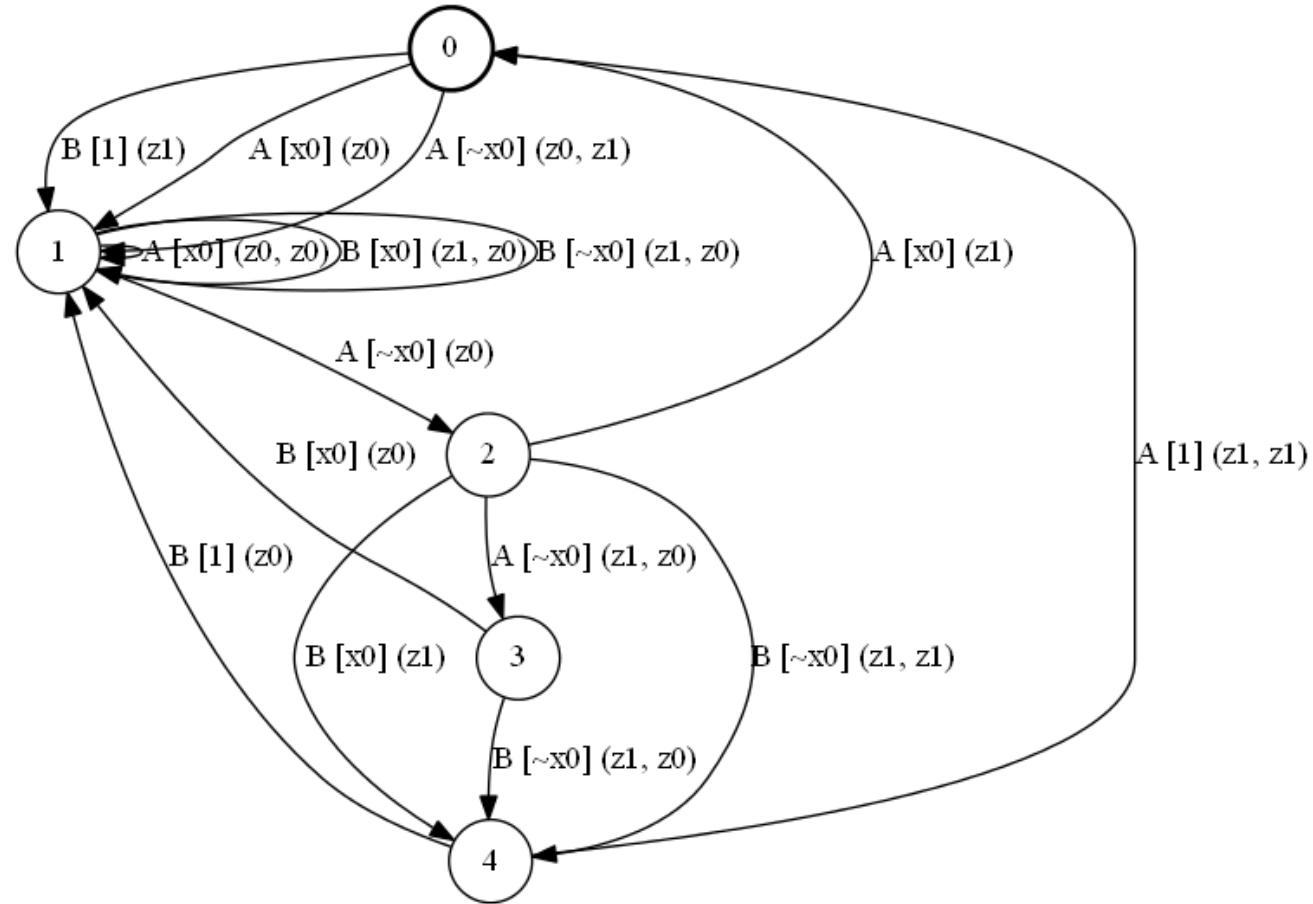
Пример построения автоматов с требованием полноты и без



Экспериментальные исследования

1. Генерация полного случайного управляющего автомата
2. Построение сценариев работы, покрывающих все переходы автомата (случайные пути в автомате)
3. Запуск алгоритма на данных сценариях (с требованием полноты и без)
4. Сравнение исходного и полученного автоматов

Пример сгенерированного автомата



Результаты для 4 и 6 состояний

Число состояний	Суммарная длина сценариев	% полных автоматов A_N	Доля изоморфных A_N автоматов	Доля изоморфных A_C автоматов
4	800	0.9	0.1	0.2
4	900	0.9	0.5	0.4
4	1000	1.0	0.4	0.5
4	1100	1.0	0.5	0.6
4	1200	1.0	0.1	0.2
6	800	1.0	0.1	0.1
6	900	0.7	0.2	0.2
6	1000	0.9	0.0	0.2
6	1100	0.9	0.2	0.3
6	1200	0.9	0.1	0.1

Результаты для 8 и 10 состояний

Число состояний	Суммарная длина сценариев	% полных автоматов A_N	Доля изоморфных A_N автоматов	Доля изоморфных A_C автоматов
8	800	0.8	0.5	0.5
8	900	0.7	0.5	0.6
8	1000	0.8	0.6	0.6
8	1100	0.8	0.7	0.7
8	1200	0.5	0.8	0.8
10	800	0.4	0.3	0.3
10	900	0.6	0.3	0.3
10	1000	0.4	0.2	0.2
10	1100	0.4	0.4	0.4
10	1200	0.7	0.3	0.4

Результаты экспериментов

- Всегда были найдены полные автоматы
 - Метод, основанный на SAT, не всегда строит полные автоматы
 - В среднем производительность – несколько секунд
- Доля изоморфных автоматов выше
- Некоторые задачи считались десятки часов работы ПК
 - Дальнейшее исследование – доработка метода

Результаты

- Разработан и реализован метод построения управляющих автоматов, основанный на сведении задачи к задаче удовлетворения ограничениям
- Экспериментальные исследования продемонстрировали множество задач, для которых метод находит качественно лучшее решение
- «Научные и научно-педагогические кадры инновационной России» на 2009–2013 годы. «Разработка метода машинного обучения на основе алгоритмов решения задачи о выполнимости булевой формулы для построения управляющих конечных автоматов»
- Программа «УМНИК»

Спасибо за внимание!

Владимир Ульянов

