МИНИСТЕРСТВО ОБРАЗОВАНИЯ И НАУКИ РОССИЙСКОЙ ФЕДЕРАЦИИ
НАЦИОНАЛЬНЫЙ ИССЛЕДОВАТЕЛЬСКИЙ
ТОМСКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ
ИНСТИТУТ ОПТИКИ АТМОСФЕРЫ СО РАН им. В.Е. ЗУЕВА

НОВЫЕ ИНФОРМАЦИОННЫЕ ТЕХНОЛОГИИ В ИССЛЕДОВАНИИ СЛОЖНЫХ СТРУКТУР

МАТЕРИАЛЫ
ДВЕНАДЦАТОЙ КОНФЕРЕНЦИИ С МЕЖДУНАРОДНЫМ УЧАСТИЕМ
4–8 июня 2018 г.

Мероприятие проведено при финансовой поддержке
Российского фонда фундаментальных исследований (проект № 18-07-20033)

Томск
Издательский Дом Томского государственного университета
2018

`PreConcatPrefix`, при этом производится разбивка программы на составные блоки вида *A.X* и *X.A*. Здесь *A* является фиксированной частью программы, которая не содержит пользовательских данных (например, это часть SQL запроса к базе данных), *X* – данные от пользователя, которые будут конкатенироваться с фиксированной частью запроса. В качестве примера можно привести очень распространённую форму логин-пароль. В ней в качестве фиксированной части программы можно использовать следующий SQL запрос в базу данных – "`SELECT username FROM accounts WHERE password =`", в этот запрос вставляются пользовательские данные (собственно, пароль) и в результате конкатенации образуется конструкция вида *A.X*.

В данной работе вместо операций `PreConcatSuffix` и `PreConcatPrefix` рассматриваются операции правого/левого частного, введенные на регулярных языках и описанные, например, в работе [1]. Важным преимуществом использования данных операций является возможность рассматривать все уязвимые фрагменты PHP-программы одновременно, а не каждый уязвимый фрагмент отдельно. Для этого все фрагменты одной программы, к которым будет применена операция правого частного, можно объединить в один полуавтомат. Фрагменты программы, к которым необходимо применять левое частное, также можно объединить, но в отдельный полуавтомат. Такое объединение возможно за счёт того, что каждая атака в полуавтомате описывается отдельной «ветвью» переходов. При этом после отсечения фиксированной части информация о том, из какого поля данных пришла атака, остаётся потенциально доступной. Это связано с тем, что фиксированная часть «отрезается» от регулярного языка полуавтомата с помощью изменения множества начальных (финальных) состояний. Множества же состояний и переходов полуавтомата остаются неизменными. Сохранение связи между полем данных и связанной с ним возможной атакой позволяет организовать более точную очистку вводимых пользователем данных.

На данный момент программно реализованы операции правого и левого частного, реализации проверены на ряде примеров. При программной реализации использовалось разработанное ранее и использованное в работе [4] представление полуавтомата. При реализации правого частного использовался алгоритм, предложенный Питером Линцем [3]. Алгоритм получения левого частного был разработан самостоятельно на основе формулы, приведенной в [5]. В дальнейшем планируется проведение экспериментов с реальными примерами.

Литература

1. *Fang Yu, Muath Alkhalaf, Tevfik Bultan.* Generating Vulnerability Signatures for String Manipulating Programs Using Automata-based Forward and Backward Symbolic Analyses // ASE '09 Proceedings of the 2009 IEEE/ACM International Conference on Automated Software Engineering. P. 605–609.
2. *Hung-En Wang, Tzung-Lin Tsai, Chun-Han Lin, Fang Yu, Jie-Hong R. Jiang.* String Analysis via Automata Manipulation with Logic Circuit Representation. // Computer Aided Verification: 28th International Conference, CAV 2016, Toronto, ON, Canada, July 17–23, 2016, Proceedings. Part I. P. 241–260.
3. *Linz P.* An introduction to formal languages and automata / Peter Linz. 5th ed. p. cm. Includes bibliographical references and index.
4. *Сотников А.П.* Программная реализация преобразования глобального полуавтомата во временной автомат // Труды Четырнадцатой Всероссийской конференции студенческих научно-исследовательских инкубаторов. Томск 17–18 мая 2017 г. / под ред. В.В. Демина. Томск : Изд-во НТЛ, 2017. С. 86–89.
5. *Matz O., Miller A., Potthoff A., Thomas W., Valkema E.* Report on the Program AMoRE / Bericht Nr. 9507 October 1995.

К ПОСТРОЕНИЮ ПОДАВТОМАТОВ БЕЗ СЛИЯНИЙ ДЛЯ НЕДЕТЕРМИНИРОВАННЫХ КОНЕЧНЫХ АВТОМАТОВ*

А.С. Твардовский[1], Н.В. Евтушенко[1, 2]

Национальный исследовательский Томский государственный университет, Томск, Россия
Институт системного программирования РАН, Москва, Россия
tvardal@mail.ru, nyevtush@gmail.com

Тестирование на основе конечно автоматных моделей [1] является эффективным инструментом проверки корректности программного и аппаратного обеспечения, позволяя строить проверяющие тесты с гарантированной полнотой покрытия неисправностей (ошибок) [2–3]. При тестировании реальных систем часто приходится учитывать, что спецификации могут быть недетерминированными. Соответственно, в общем случае, при описании реальных систем используются недетерминированные конечные автоматы, для которых проверяющие тесты строятся относительно редукции [3] и часто являются слишком длинными для практического применения. В работе [4] для сокращения длины тестов предлагается использовать подавтомат исходного автомата, обладающий адаптивной различающей последовательностью. В общем случае эти последовательности также

* Работа выполнена при поддержке гранта РНФ № 16-49-03012.

могут быть достаточно длинными, однако известно [5], что для определённого класса автоматов, а именно автоматов без слияний, эти последовательности имеют полиномиальную длину относительно числа состояний автомата. В настоящей работе мы предлагаем алгоритм проверки существования подавтомата без слияний для исходного автомата-спецификации.

Под *конечным автоматом* [2] понимается четвёрка $\mathsf{S} = (S, s_0, I, O, h_S)$, где I – множество *входных символов*, O – множество *выходных символов*, S – конечное непустое множество *состояний,* s_0 – начальное состояние, $h_S \subseteq (S \times I \times O \times S)$ – *отношение переходов*. Соответственно, кортеж (s, i, o, s') описывает *переход* из состояния s в состояние s' под действием входного символа i с выдачей выходного символа o. Если для некоторой пары $(s, i) \in S \times I$ существует более одного перехода в множестве h_S, то конечный автомат называется *недетерминированным*. В настоящей работе мы рассматриваем полностью определённые возможно недетерминированные автоматы, т.е. автоматы, для которых в любом состоянии существует выходная реакция на каждый входной символ. Конечный автомат называется *автоматом без слияний* [5], если для любых двух состояний s_1 и s_2 и любого входного символа i справедливо: если $(s_1, i, o, s_1') , (s_2, i, o, s_2') \in h_S$, то состояния s_1' и s_2' не совпадают.

Для проверки существования в полностью определенном недетерминированном автомате подавтомата без слияний воспользуемся понятием максимальной клики в неориентированном графе. Под *кликой* в неориентированном графе понимается подмножество вершин графа, каждая пара из которых соединена ребром. Для каждого входного символа i исходного автомата построим граф совместимости G_i, вершины которого соответствуют переходам под действием этого входного символа. Переходы (s_1, i, o_1, s_1'), $(s_2, i, o_2, s_2') \in h_S$ назовем *совместимыми*, если $o_1 \neq o_2$ или $s_1' \neq s_2'$.

Утверждение 1. В автомате существует полностью определенный подавтомат без слияний с входным символом i, если и только если существует клика в графе G_i, содержащая переход (s, i, o, s') для каждого состояния s исходного автомата.

Проблемы, возникающие при построении максимальной клики в графе, достаточно хорошо исследованы [6], и существует большое количество программных реализаций для нахождения максимальной клики в заданном графе. Если для некоторого i в графе G_i отсутствует клика, содержащая переход (s, i, o, s') для каждого состояния s исходного автомата, то при построении подавтомата без слияний данный входной символ не рассматривается. Из найденных подавтоматов без слияний выделяются те, в которых каждая пара состояний адаптивно различима, и, соответственно, адаптивная различающая последовательность для всего подавтомата строится по алгоритму из [5]. Далее осуществляется проверка существования адаптивных передаточных последовательностей для каждого состояния подавтомата, и в случае существования таковых, проверяющий тест строится методом, предложенным в [4].

Литература

1. *Гилл А.* Введение в теорию конечных автоматов. М. : Наука, 1966. 272 с.
2. *Dorofeeva R., El-Fakih K., Maag S., Cavalli A.R., Yevtushenko N.* FSM-based conformance testing methods: a survey annotated with experimental evaluation. Information and Software Technology. 2010. 52. Elsevier P. 1286–1297.
3. *Petrenko A., Yevtushenko N.* Conformance tests as checking experiments for partial nondeterministic FSM // Grieskamp W., Weise C. (eds.) FATES 2005. LNCS. Vol. 3997. P. 118–133. Springer, Heidelberg (2006). doi:10.1007/11759744_9.
4. *Aleksandr Tvardovskii.* Refining the Specification FSM When Deriving Test Suites w.r.t. the Reduction Relation // Lecture Notes in Computer Science (LNCS). 2017. Vol. 10533. P. 333–339.
5. *Yevtushenko N., Kushik N.* Nondeterministic merging-free finite state machines. Proceedings of IEEE East-West Design & Test Symposium (EWDTS). 2015. P. 338–341.
6. *Гэри М., Джонсон Д.* Вычислительные машины и труднорешаемые задачи. М. : Мир, 1982. 416 с.

ПОИСК ПОДСХЕМ В КМОП СХЕМЕ ИЗ ТРАНЗИСТОРОВ

Д.И. Черемисинов, Л.Д. Черемисинова

Объединений институт проблем информатики НАН Беларуси, Минск, Беларусь
cher@newman.bas-net.by, cld@newman.bas-net.by

Сегодня графы широко используются для моделирования данных в различных предметных областях, требующих выяснения и определения правил и схем отношений объектов. Graph Mining – одно из направлений интеллектуального анализа данных, в котором объемные комплексные данные представлены в виде графов, а анализ ведется для того, чтобы получить новые знания. В настоящем докладе рассматривается задача поиска часто встречающихся подграфов в большом графе, являющимся моделью КМОП схемы из транзисторов, и приводится обзор подходов к поиску решений этой задачи.

В последние годы наблюдается повышенный интерес к разработке алгоритмов интеллектуального анализа данных, работающих на графах. Такие графы возникают естественно в ряде различных областей таких как: обнаружение сетевое атак, семантический web, поведенческое моделирование, перепроектирование СБИС, ана-