УДК 004.421.2

Автоматизированное формирование тестов при характеризации цифровых ячеек с использованием веб-доступа

А.А. Лялинский

Институт проблем проектирования в микроэлектронике РАН, zelyal@inbox.ru

***Аннотация* — Рассматривается задача автоматизированного формирования входных тестовых последовательностей, используемых на этапе характеризации цифровых комбинационных ячеек. Определены ограничения, накладываемые на состав входных сигналов. Проанализированы алгоритмы, используемые для решения задачи на ее различных этапах, и разработана система подготовки тестов. Приведены результаты расчета ряда тестовых ячеек, показывающие высокую эффективность используемых алгоритмов. Интерактивный доступ к системе представлен на сайте http://www.ippm.ru/lftest3/.**

***Ключевые слова* — цифровые схемы, формирование тестов, веб-доступ.**

I. Введение

Задача формирования набора входных воздействий возникает на одном из этапов процесса характеризации типовых цифровых ячеек ИС. Ее успешным решением следует считать получение такого тестового набора, который обеспечивает проверку всех возможных переходов выходных сигналов, вызванных изменениями на его входах. Отметим, что данная постановка задачи применима только к комбинационному типу логики, когда для генерации тестов достаточно знать логическую функцию блока. Для схем памяти было бы необходимо дополнительно учитывать переходы, вызванные изменением их внутреннего состояния. Критерий оптимальности решения данной задачи очевиден – минимальная длительность во времени входных воздействий, или, что то же самое – отсутствие «холостых» переключений входных сигналов, не приводящих к переключению выходов схемы.

«Ручное» формирование набора тестов приемлемо только для схем, работа которых описывается достаточно простыми логическими выражениями. Для более сложных схем, а также в случае их большой номенклатуры, желательна автоматизация этого процесса.

Данная задача перекликается с ATPG-задачами (Automatic Test Pattern Generation [1]), в которых решается проблема генерации входных тестовых воздействий для выявления дефектов цифрового устройства, но существенно отличается от нее тем, что предлагаемая система генерации входных тестов нацелена на расчет динамических процессов. Не ставится задача покрытия всех тестовых комбинаций – необходимы только те, которые вызывают реальные изменения выходного (или внутреннего – для схем памяти) состояния схемы. Частичным обоснованием этого упрощения является то, что при измерениях временных характеристик не только вычисляется величина собственно временного параметра (задержки, фронта), но также контролируется и направление выходного сигнала (проверяется, что переход 0 → 1 на входе инвертора приведет к переходу 1 → 0 на его выходе). Практика показывает, что этого достаточно для предотвращения ошибок по статической величине сигнала, и, в то же время, позволяет не вводить лишние такты в тестовую последовательность.

Автоматизированное формирование набора входных воздействий существует во многих системах характеризации библиотечных ячеек (см., например, Charisma [2]). Особенностью данной работы является то, что

(1) предложенные алгоритмы показали высокую эффективность получаемого решения и

(2) на их основе разработан сайт, позволяющий в режиме непосредственного отклика исследовать тестовые последовательности для произвольного логического выражения логической функции.

II. Ограничения

Анализ проблемы показывает, что введя некоторые ограничения на поставленную выше задачу можно существенно повысить эффективность ее решения.

1. Будем рассматривать только такие процессы, когда изменение выходного сигнала вызвано изменением только *одного входного* сигнала. Например, для элемента OR2 будем рассматривать переходы входных сигналов 00 → 01 и 00 → 10, и не будем принимать в расчет случай с одновременным изменением обоих входов 00 → 11. В качестве обоснования этого ограничения отметим, что при одновременном изменении сигналов на нескольких входах начинается экспоненциальный рост количества состояний, которые необходимо отследить. Кроме того, при реальной работе ИС существует некий разброс задержек в путях, предшествующих данному моделируемому элементу. Этот разброс приводит к тому, что сигналы

на входах проверяемого блока в пределах такта приходят не одновременно, и срабатывание блока вызывается одним (первым из пришедших) сигналом.

2. Ограничимся случаем, когда схема имеет *один выход*. Это существенно упростит решение поставленной задачи, не изменяя ее сути. Одним из вариантов решения задачи подготовки тестов для схемы с n выходами (n > 1) может стать ее разбиение на n схем, имеющих одинаковый набор входов и единственную логическую функцию выхода. Далее здесь возможна оптимизация полученных тестовых воздействий, что выходит за рамки данной статьи.

3. Для общности отметим также как ограничение то, что предлагаемые алгоритмы пригодны для тестирования *комбинационных* схем, а также блока управляющей логики последовательностных схем, который всегда можно рассматривать как чисто комбинационную схему с одним или более входами и одним выходом, являющимся, в свою очередь, одним из управляющих сигналов оставшейся части триггерной схемы.

Предложенные ограничения отражают реальную физическую картину процессов, сопровождающих переключения схемы, и подготавливают почву для разработки тестовых последовательностей для схем, не подпадающих под перечисленные выше ограничения.

III. Математический аппарат

В качестве математической основы для решения поставленной задачи будем использовать теорию графов. Под **состоянием** S будем понимать упорядоченное множество булевых значений входных и выходных сигналов комбинационного (т.е. не имеющего памяти) блока. Для определенности вначале расположим входы, за ними - выход. Таким образом, для логического элемента NAND2 полное множество состояний состоит из: 001, 101, 011 и 110. Введем граф G, **вершинами** которого являются состояния, а **ребрами** - переходы из одного состояния (т.е. комбинации входных и выходных сигналов) в другое.

Вследствие наложенных нами выше ограничений на то, что:

(1) мы рассматриваем только процессы, связанные с *изменением* выходного состояния, и

(2) допустимо изменение только *одного* входного сигнала,

из полного множества ребер выпадают те ребра, которые отображают переходы, нарушающие эти ограничения. Это, в свою очередь, приводит к выпадению из графа части состояний (вершин графа, у которых не осталось входящих-выходящих ребер). На рис. 1 показано полное множество состояний и усеченный вариант, используемый в дальнейшей работе.

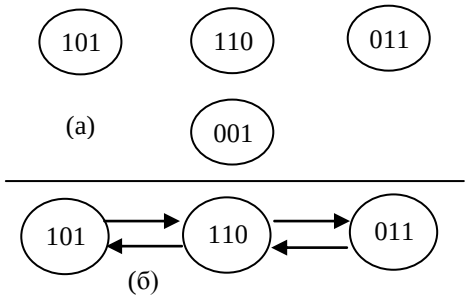

Рис. 1. (а) – полное множество состояний для схемы NAND2; (б) – усеченное множество и граф, используемый для расчета тестовых последовательностей.

Особенности полученного графа:

1. Граф может состоять из нескольких несвязанных между собой подграфов. Причина в том, что существуют (и достаточно часто) логические выражения с таким набором состояний, для которых нельзя найти корректный переход (такой, в котором изменение выхода вызвано изменением только одного входного входа). Например, такими чертами обладает функция a&b|c, имеющая два подграфа.
2. Граф является ориентированным. Все непосредственно связанные между собой вершины имеют исходящее и входящее ребро. Отсюда следует, что степень любой вершины графа – четная. Это свойство вытекает из комбинационного характера схем – если можно перейти из одного состояния в другое, то ничто не мешает повторить это и в обратном направлении.

IV. Основные этапы решения задачи

Задача нахождения нужной тестовой последовательности сводится к задаче нахождения *эйлерова пути* графа [3] – пути, проходящего через все ребра графа и притом только по одному разу. Первая часть этого определения обеспечит нам проверку всех возможных переходов входных сигналов, вторая гарантирует их минимальное количество в создаваемой тестовой последовательности. Согласно теореме Эйлера *граф является эйлеровым (т.е. содержащим эйлеров цикл) тогда и только тогда, когда он связен и все его локальные степени четны.* Связность графа в целом, как показывает ряд примеров, гарантировать нельзя. Поэтому вначале граф схемы разбивается на несвязанные между собой подграфы, для каждого из которых решается отдельная задача нахождения эйлерова пути. Четность его вершин вытекает из комбинационного характера схемы.

Общая последовательность действий такова:

1. Граф схемы разбивается на подграфы. Каждый подграф является связным графом (т.е. между любой парой вершин этого подграфа существует как минимум один путь). Между собой подграфы не связаны.

2. Для каждого подграфа:
    a. выбирается стартовая вершина;
    b. строится эйлеров путь, начиная со стартовой вершины.
3. Для получения единой тестовой последовательности подграфы соединяются между собой дополнительно введенными ребрами.

Рассмотрим эти этапы более подробно.

A. *Разбиение графа схемы на подграфы*

На основе алгоритма Флойда–Уоршелла [4, 5] строится матрица достижимости на основе предварительно построенной матрицы инцидентности графа. Анализ структуры полученной матрицы достижимости позволяет легко выделить блоки связанных между собой вершин, составляющих отдельные подграфы. Сложность алгоритма оценивается как $O(n^3)$, где n – число вершин графа, в нашем случае – количество состояний усеченного графа. Быстроту вычислений на данном этапе обеспечивает то, что основными операциями при работе данного алгоритма являются две простейших логических операции дизъюнкции и конъюнкции.

B. *Выбор стартовой вершины*

Так как для работы алгоритма построения эйлерова пути не имеет значения, какая вершина выбрана в качестве стартовой, то можно обосновать этот выбор иными, удобными для нас условиями.

Учитывая, что на последнем этапе работы алгоритма полученные в подграфах эйлеровы пути надо будет соединить в одну цепочку (обеспечивая тем самым единственную тестовую последовательность), в качестве стартовой будем выбирать вершину, выходной бит которой совпадает с выходным битом последней вершины пути предыдущего подграфа. При этом, как и при переходах внутри подграфа, количество переключений на входах должно отличаться на единицу. Таким образом, такое связывающее переключение не приводит к срабатыванию на выходе схемы, а только подготавливает условия для дальнейших переключений.

Выбор стартовой вершины в самом первом подграфе произволен, например, можно выбирать вершину с наименьшим количеством исходящих связей.

Отметим, что если по каким-либо причинам эйлеров путь от данной вершины построить не удается, то в качестве стартовой выбирается другая вершина подграфа, удовлетворяющая указанным выше критериям. Если в данном подграфе нет ни одной вершины, подходящей в качестве стартовой, то выполняются попытки найти такую вершину в других подграфах, остающихся необработанными на данный момент.

C. *Построение эйлерова пути в подграфе*

Для построения эйлерова пути в подграфе были протестированы два алгоритма.

Первый вариант был построен на основе волнового алгоритма. Показывая удовлетворительные результаты для большинства схем, он потерпел неудачу на схемах с большим количеством входов (в частности, на AND9). Причина неудачи – огромное количество порождаемых промежуточных путей, которые необходимо хранить и обрабатывать.

Более удачным оказался вариант на основе поиска в глубину (для каждой непройденной вершины находятся все непройденные смежные вершины и для каждой из них поиск повторяется). При этом все промежуточные неудачные варианты сразу отбрасываются. Поиск прекращается по достижению первого удачного варианта.

D. *Связывание эйлеровых путей подграфов в единое целое*

Основные критерии, на основании которых строятся дополнительные ребра для связывания подграфов, описаны выше в пункте «Выбор стартовой вершины». Дополнительно отметим, что в случае невозможности связывания подграфов на основании перечисленных там критериев, для связи выбираются вершины, имеющие одно и то же выходное состояние, и любое количество несовпадений во входах. В этом случае нам приходится выбирать между необходимостью либо останова алгоритма по причине невозможности нахождения решения, либо отказом от ограничения по количеству переключений (строго одно) на входах при переходе от одного состояния к другому. Можно предоставить возможность выбора типа выхода из этой непростой ситуации пользователю системы.

V. Веб-доступ к системе

Описанные выше алгоритмы реализованы в программе на языке PHP, размещенной на серверной части сайта http://www.ippm.ru/lftest3/. Пользователь имеет возможность удобным ему способом задать произвольное логическое выражение и получить в текстовом и табличном виде оптимальный набор соответствующих ему тестовых последовательностей.

В настоящий момент программное обеспечение сайта реализовано на основе стандартных синхронных методов взаимодействия связи клиент-сервер с вытекающим отсюда 30-секундным ограничением на время отклика. Это, в свою очередь, не позволяет анализировать предельные возможности системы с большим количеством входов (независимых переменных). В дальнейшем предполагается реализовать связь клиент-сервер на основе AJAX-технологий [6], обеспечивающих асинхронный доступ, что позволит выявить предельные характеристики системы.

VI. Результаты работы алгоритма

Работа системы проверена на ряде тестов различной сложности (см. табл. 1).

Результаты тестов представлены в табл. 2. Для трех тестов из этого набора также дано графическое представление результатов.

Таблица 1

Логические выражения для тестируемых схем

Схема	**Выражение**	**Схема**	**Выражение**
nor2	~(a|b)	*exnor2*	((a&b)|(~a&~b))
and3	a&b&c	*mux2*	(d0&~sl0)|(d1&sl0)
and6	a&b&c&d&e&~f	*mux4*	(d0&~sl0&~sl1)|(d1&sl0&~sl1)|(d2&~sl0&sl1)|(d3&sl0&sl1)
and9	~((~(a&b&c))|(~(d&e&f))|(~(g&h&i)))	*mx4*	(sl0&d0)|(sl1&d1)|(sl2&d2)|(sl3&d3)
ao211	(a&b)|c|d	*half_sum_s*	a&~b&~ci|~a&b&~ci|~a&~b&ci|a&b&ci
ao21i1	(a&b)|~c|~d	*half_sum_co*	a&b&~ci|~a&b&ci|a&~b&ci|a&b&ci
ao221	(a&b)|(c&d)|e		

При задании логических функций используются следующие обозначения:

a, b, c, …	независимые переменные (входы схемы)
x	функция (выход схемы).

Обозначения логических операций взяты из языка Verilog:

~	логическая операция «отрицание»,
&	логическая операция «И»,
|	логическая операция «ИЛИ»,
^	логическая операция «исключающее ИЛИ»,
^~ ~^	логическая операция «отрицание исключающего ИЛИ».

На рисунках в таблице состояний выше диаграмм кривых R и F обозначают, соответственно, Rise и Fall – нарастание и спад сигнала. Clk – условные такты, при этом, простоты ради, состояниям, содержащим R и F, выделены целые такты. Красным цветом показаны входные кривые, зеленым – выходные (на всех рисунках выходная кривая – самая нижняя).

В качестве критерия оптимальности используется количество дополнительных переходов (не приводящих к изменению выхода схемы) введенных в тестовую последовательность.

На рис. 2 показаны результаты для AND9. Для данной схемы мы наблюдаем 100% эффективность работы алгоритма – все переключения входов приводят к переключению выхода.

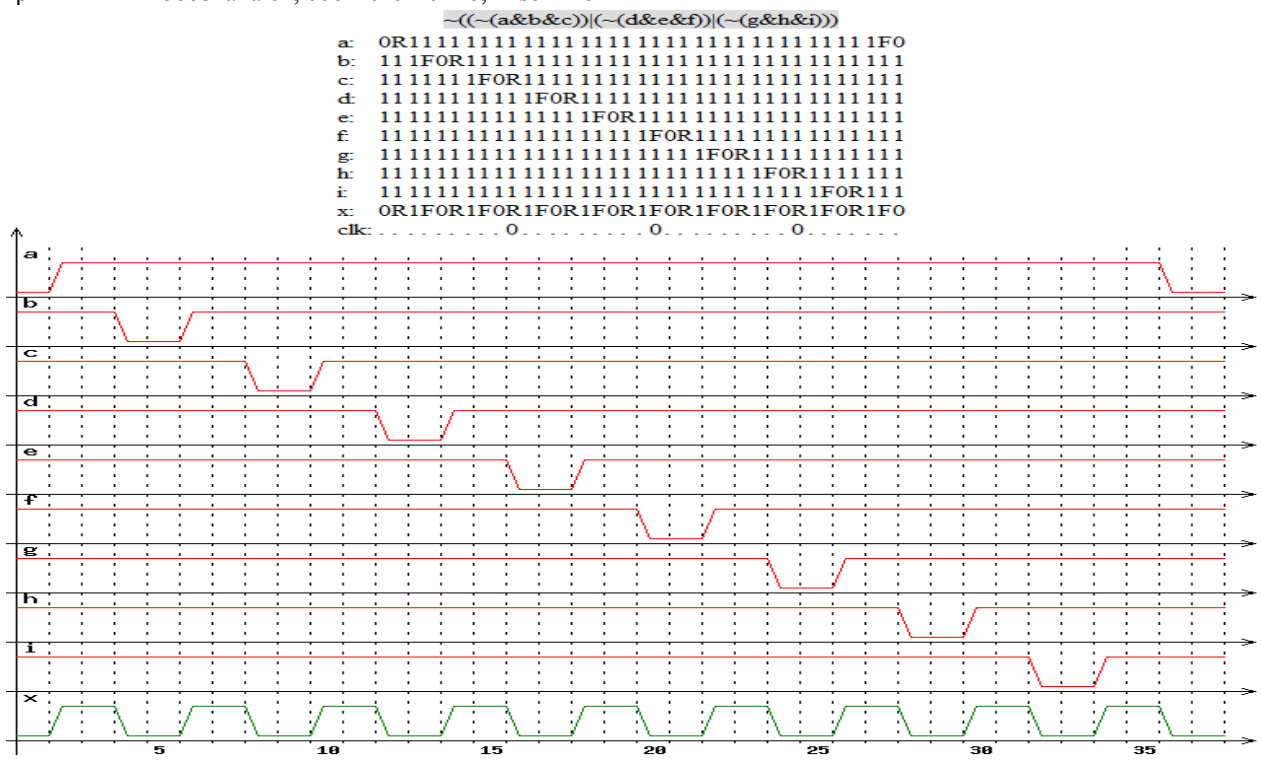

Рис. 2. Результаты построения тестовой последовательности для AND9.

На рис. 3 показаны результаты для функции «a ^~ b & c». Здесь уже потребовалось введение одного «холостого» переключения (первое переключение входа «c»). Все остальные переключения входов приводили к переключению выхода.

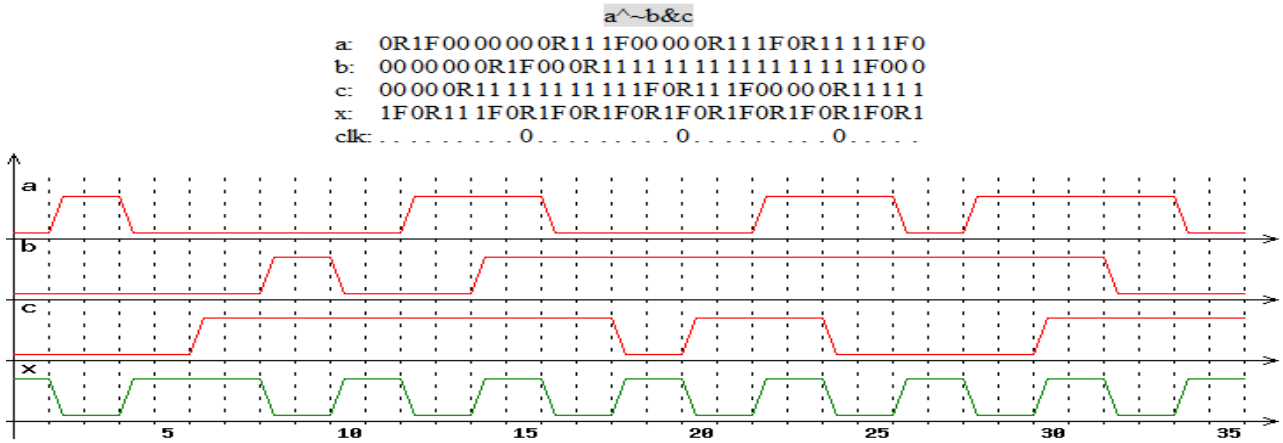

Рис. 3. Результаты построения тестовой последовательности для функции «a ^~ b & c».

На рис. 4 показаны результаты для функции ao21i1. Потребовалось введение двух «холостых» переключений (первое переключение входа «b», последнее переключение входа «c»). Эффективность – 92%.

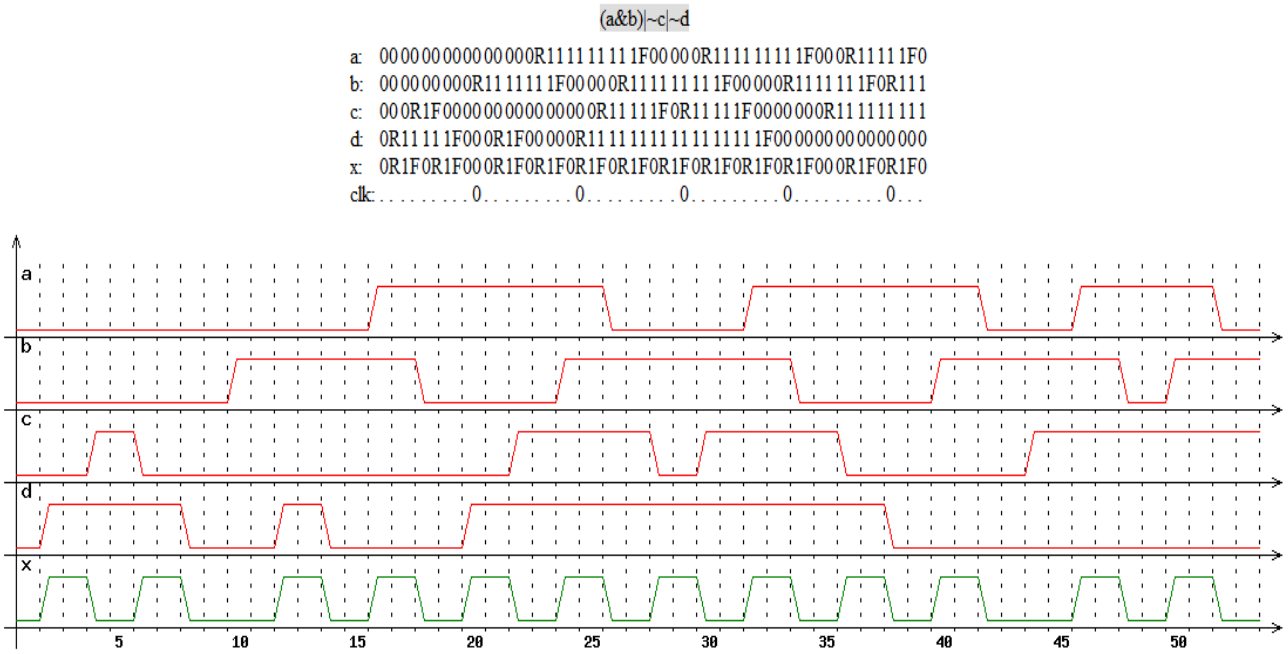

Рис. 4. Результаты построения тестовой последовательности для функции ao21i1.

Для всех случаев, когда мы имеем дело с достаточно простой формой логического выражения, алгоритм показал 100% эффективность – было создано ровно столько переключений входных сигналов, столько требуется для вычисления фронта и спада от каждого входа к выходу. Для более сложных случаев появляются дополнительные («холостые») переходы, снижающие эффективность алгоритма, которая, тем не менее, не падает ниже 90%.

VII. Выводы

Разработано математическое и программное обеспечение системы автоматизированного формирования входных тестовых воздействий, используемых при характеризации цифровых комбинационных ячеек. Показана высокая эффективность ее работы. Веб-доступ к системе можно получить на сайте http://www.ippm.ru/lftest3/.

Таблица 2

Численные характеристики работы алгоритма

Схема	**Кол-во вхо-дов**	**Кол-во входных комбинаций**		**Кол-во переходов**		**Эффективность алгоритма** $(n_{total} - n_{add}) / n_{total}$ **(%%)**
		полное	**актив-тив-ных**	**общее** n_{total}	**из них дополнительных** n_{add}	
nor2	2	4	3	4	0	100 %
and3	3	8	4	6	0	100 %
and6	6	64	7	12	0	100 %
and9	9	512	10	18	0	100 %
ao211	4	16	10	17	1	94 %
ao21i1	4	16	14	26	2	92 %
ao221	5	32	24	45	3	93 %
exnor2	2	4	3	4	0	100 %
mux2	3	8	8	13	1	92 %
mux4	6	64	50	95	3	97 %
mx4	8	256	188	446	14	97 %
half_sum(sum)	3	8	7	11	1	91 %
half_sum(carry out)	3	8	4	6	0	100 %

ЛИТЕРАТУРА

[1] Louis Scheffer, Grant Martin, Luciano Lavagno. Electronic Design Automation For Integrated Circuits Handbook. CRC Press, Taylor & Francis Group. ISBN 0-8493-3096-3. 2006.

[2] Программная система Charisma для автоматизированной характеризации библиотек стандартных ячеек и ячеек ввода/вывода. URL: http://www.radixtools.ru/products-charisma (дата обращения: 23.04.2012).

[3] Харари Ф. Теория графов. М.: Мир. 1998.

[4] Robert W. Floyd. Algorithm 97: Shortest Path // Communications of the ACM **5** (6). 1962. p. 345.

[5] Stephen Warshall. A theorem on Boolean matrices // Journal of the ACM **9** (1). 1962. P. 11–12.

[6] Дейв Крейн, Эрик Паскарелло, Даррен Джеймс. AJAX в действии: технология - Asynchronous JavaScript and XML = Ajax in Action // М.: Вильямс. 2006. С. 640. ISBN 1-932394-61-3.