Комбинаторика слов и построение тестовых последовательностей[1]

В. В. Кулямин

1. Введение

При тестировании систем, поведение которых определяется не только последним обращением к ним, а и предшествующей историей работы, т.е. зависит от внутреннего состояния системы, необходимо строить тесты в виде последовательностей обращений, чтобы покрыть возникающие разнообразные ситуации.

Если о системе известно немного, например, только список обращений, которые можно делать, построить ее тест в полном смысле этого слова нельзя, поскольку частью теста всегда является проверка правильности работы системы в ответ на тестовые обращения.

Однако можно попробовать построить входные данные теста — определить, какие операции, с какими параметрами и когда вызывать, предположив, что проверка правильности работы системы производится отдельно. Ее может выполнять специальный модуль автоматической проверки, или же это может делать человек, понимающий, когда очередная реакция системы правильна, а когда — нет.

При этом возникает две задачи: построение значений параметров вызовов и построение последовательности вызовов. Непосредственное решение этих задач путем построения всех возможных комбинаций значений параметров и всех возможных последовательностей обращений сразу приводит к комбинаторному взрыву, поэтому требуются методы построения небольшого числа тестов, которые, тем не менее, были бы достаточно качественны, то есть покрывали бы максимально возможное число различных ситуаций, связанных с поведением тестируемой системы.

Первая задача при отсутствии дополнительной информации о тестируемой системе обычно решается на основе разбиений возможных значений параметров на конечное число групп и использования различных комбинаций значений из разных групп. Для построения этих комбинаций можно

[1] Данная работа поддержана грантом Российского Фонда содействия отечественной науке.

использовать *покрывающие массивы* (covering arrays) [1—3], которые обеспечивают минимально возможные множества комбинаций, перебирающие все возможные сочетания пар, троек или другого числа значений отдельных параметров. Другие методы основываются на эвристических алгоритмах, вычисляющих приближения к минимальным покрывающим массивам; это оправдано, поскольку построение такого массива с нужными параметрами является NP-полной задачей. Обзоры имеющихся результатов по построению покрывающих массивов см. в [1, 2].

Эта статья целиком посвящена возможным подходам к решению второй задачи — задаче построения тестовых последовательностей, поскольку она в имеющейся литературе практически не затрагивается. В статье не излагается ее полное решение с каких бы то ни было позиций, а, скорее, рассматриваются несколько подходов к такому решению, основанных на похожих идеях, и освещаются известные автору результаты, полученные в рамках этих подходов.

2. Формулировка задачи

Будем представлять возможные тестовые последовательности словами в алфавите из возможных вызовов. Число возможных вызовов можно считать конечным — для этого из множества всех возможных вызовов нужно выбрать конечное множество представителей, например, выделив «наиболее интересные» комбинации значений параметров (см. выше). Такому подходу практически нет альтернатив, поскольку тестирование, будучи принципиально конечной процедурой, не может обеспечить качественную проверку работы системы, если имеется бесконечное множество существенно различных способов обратиться к ней.

Далее будем рассматривать тестовые последовательности как слова в конечных алфавитах. Поскольку мы не обладаем дополнительной информацией о тестируемой системе, все вызовы (представители) для нас ничем не отличаются друг от друга. Можно обозначить их символами от *0* до *(m-1)* и рассматривать *m-последовательности* или *m-слова* — слова произвольной длины, состоящие только из таких символов.

Предположим теперь, что, помимо известного нам числа m различных воздействий на систему, от пользователя мы можем получить только ограничение на суммарную длину теста. Стоящая перед нами задача может быть в итоге сформулирована так: как построить одно m-слово длины, не превосходящей N, имеющее *«как можно более разнообразное»* поведение? Возможные интерпретации «как можно более разнообразного» поведения рассматриваются ниже.

Сразу отметим, что можно пытаться построить несколько m-слов, сумма длин которых не превосходит N, совместно обеспечивающих нужное разнообразие. Такая постановка задачи и возможные подходы к ее решению оставлены за рамками данной статьи.

3. Максимизация числа различных подслов

Наверное, наиболее очевидный способ интерпретации «как можно более разнообразного поведения» слова — это наличие у него *как можно большего числа различных подслов*.

В слове длины N имеется всего

$$N + (N-1) + (N-2) + \ldots + 2 + 1 = N(N+1)/2 \qquad (*)$$

подслов (слагаемые соответствуют подсловам длины *1, 2* и т.д.). Можно попытаться максимизировать количество разных подслов, сделав все подслова какой-то длины k различными. При этом все подслова большей длины тоже автоматически будут различны, т.е. мы получим как минимум $(N-k+1)(N-k+2)/2$ разных подслов длины k. Значение k при этом можно выбрать так, чтобы максимизировать полученное выражение при фиксированном N, т.е. как можно меньшее.

При этом имеется всего m^k разных m-слов длины k, а если они все реализуются как подслова одного слова, длина этого слова должна быть не меньше m^k+k-1. Предположим, что существуют такие *«наиболее плотные»* слова, что все их подслова длины k различны и в то же время включают в себя все возможные m-слова длины k, и они нам известны. Тогда для заданного N можно найти минимальное k, такое что $N \le m^k+k-1$, т.е. $m^{(k-1)}+(k-1)-1 < N \le m^k+k-1$, и в качестве искомого слова взять начало длины N соответствующего «наиболее плотного» слова. Это гарантирует различие всех подслов длины k в нем.

Для того, чтобы еще больше увеличить количество разных подслов в нашем слове, можно попытаться найти «наиболее плотное» слово для $(k-1)$, которое продолжается в «наиболее плотное» слово для k. Взяв начало этого последнего слова, мы получим наилучший результат — в полученном слове длины N все подслова минимально возможной длины k различны, и в то же время в нем в качестве подслов содержатся все возможные слова длины $(k-1)$ и, соответственно, меньшие — все слагаемые в формуле (*) имеют максимальные возможные значения.

Осталось выяснить два момента.

- Существуют ли «наиболее плотные» слова для всех m и k, и если это так, можно ли их продолжать до «наиболее плотных» слов для m и $(k+1)$? Есть ли достаточно эффективные алгоритмы для построения таких слов?
- Какой тестовой гипотезе, т.е. какому допущению относительно свойств тестируемой системы, соответствует выбранное понимание «наиболее разнообразного» поведения слова? Для какого рода систем этот подход дает действительно оптимальное покрытие различных возможных ситуаций?

Ответу на первый вопрос посвящен весь следующий раздел. В рамках данного раздела проще ответить на второй вопрос.

Такой подход к построению тестов базируется на предположении о том, что *поведение тестируемой системы целиком определяется последними k обращениями*. При этом, выбирая тестовую последовательность, содержащую как можно больше различных подпоследовательностей длины k, мы покрываем наибольшее количество различных «поведений» системы. В качестве реального примера такой системы можно привести кодовый замок, реагирующий на последние набранные k (обычно 4 или 5) цифр (автор сам пользовался такими замками, см. также статью [4], посвященную стратегии эффективного вскрытия замка в отеле Baltimore Hilton).

Дополнительное условие, обеспечивающее содержание в тестовой последовательности всех возможных слов длины $(k-1)$, по отношению к этой гипотезе является некоторой необязательной «оптимизацией».

3.1. Слова де Бройна

Итак, что можно сказать про «наиболее плотные» слова для m и k, т.е. m-слова, содержащие в качестве своих подслов длины k все возможные m-слова длины k, причем ровно по одному разу каждое?

Такие слова или последовательности известны под именем *слов* или *последовательностей де Бройна* (de Bruijn) *шага k*. Они были известны для случая $m = 2$ еще в 1894 г. [5], а впоследствии были независимо переоткрыты де Бройном [6] и еще несколькими авторами [7, 8, 9]. В статье [6] де Бройн поставил и решил задачу о том, сколько имеется циклов длины m^k (цикл — это класс эквивалентности слов по циклическим перестановкам их символов, например, *0110, 0011* и *1001* относятся к одному циклу), содержащих все возможные m-слова длины k. Полученный им ответ $(m!)^{m^{k-1}} / m^k$ (см. упражнение 2.3.4.2-23 в [10]) показывает, что такие циклы существуют при всех $m, k \ge 1$. Слово де Бройна получается из цикла разрезанием его в некотором месте и копированием $(k-1)$ символа из начала в конец получившегося слова в обратном порядке, чтобы сохранить подслова длины k.

Обзор [11] дает наиболее полный экскурс в теорию слов де Бройна и историю их использования для решения различных задач. Там они названы циклами нелинейных сдвиговых регистров полной длины (full length nonlinear shift register cycles). Среди задач, связанных с комбинаторикой слов, в которых возникают слова де Бройна, можно отметить построение псевдослучайных последовательностей [12], построение кодов [13, 14], кодирование образов [15, 16], построение машин на основе сдвиговых регистров [11, 17, 18], организацию CDMA-сетей. В связи с тестированием программного обеспечения они упоминаются в [19].

Слова де Бройна связаны с графами специального вида, называемыми также графами де Бройна. *Граф де Бройна с параметрами* $m \ge 1$ и $k \ge 1$ $B(m, k)$ — это ориентированный граф с m^{k-1} вершинами $V(m, k) = [0..(m-1)]^{k-1}$, являющимися всеми возможными m-словами длины $(k-1)$, и ребрами $E(m, k) = [0..(m-1)]^k$,

являющимися всеми возможными m-словами длины k. При этом ребро $x_1x_2\ldots x_{k-1}x_k$ начинается в вершине $x_1x_2\ldots x_{k-1}$ и заканчивается в вершине $x_2\ldots x_{k-1}x_k$.

Примеры графов де Бройна показаны на рис. 1.

Достаточно легко убедиться в выполнении следующего утверждения.

Утверждение 1.

1) Граф *B(m, 1)* имеет ровно одну вершину — пустое слово — и m ребер-петель.
2) Граф *B(m, 2)* изоморфен полному ориентированному графу с петлями на m вершинах.
3) Количества входящих и выходящих ребер для любой вершины *B(m, k)* равны m.

$$\forall v \in V(m, k)\ in\text{-}deg(v) = out\text{-}deg(v) = m$$

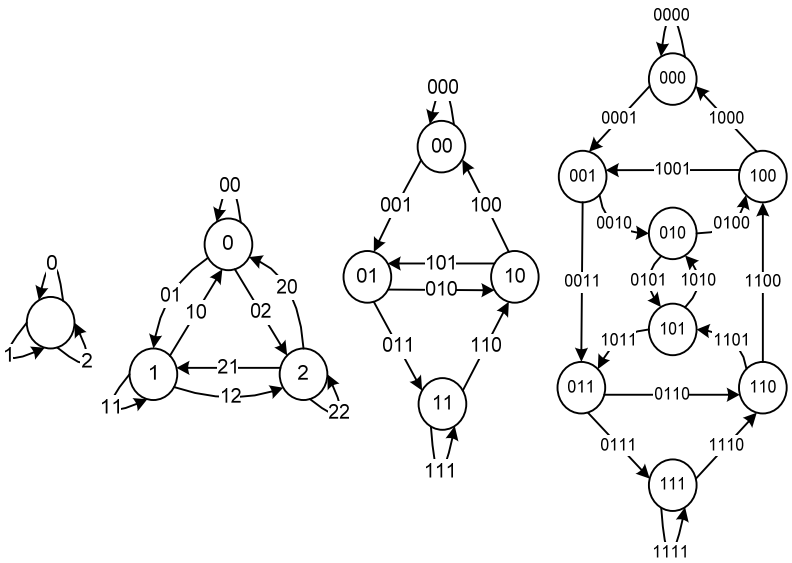

Рис. 1. Графы B(3, 1), B(3, 2), B(2, 3), B(2, 4).

Последний пункт непосредственно влечет следующее.

Утверждение 2.

1) Для всех $m, k \geq 1$ граф *B(m, k)* эйлеров, т.е. в нем существует цикл, включающий все ребра.
2) Любой эйлеров путь в *B(m, k)* однозначно соответствует слову де Бройна, а значит, такие слова существуют для всех $m, k \geq 1$. Для построения слова, соответствующего пути в *B(m, k)*, выпишем слово,

соответствующее первому ребру пути, затем для каждого следующего ребра пути будем приписывать в конец полученного слова последний символ этого ребра.

Определим для ориентированного графа G *дуальный граф L(G)* как граф, имеющий в качестве вершин множество ребер G, а в качестве ребер — множество пар смежных ребер G, т.е. таких, что конец первой является началом второй. При этом ребро *L(G)* начинается в вершине, соответствующей первому элементу пары, а кончается в вершине, соответствующей второму.

Утверждение 3.

1) Для всех $m, k \geq 1$ дуальный граф к *B(m, k)* изоморфен *B(m, k+1)*. Это легко проверить, заметив, что паре смежных ребер *B(m, k)* соответствует слово длины *(k+1)*, построенное по правилу из второго пункта предыдущего утверждения. Кроме того, все m-слова длины *(k+1)* могут быть получены таким способом.
2) Эйлеров путь или цикл на графе G соответствует гамильтонову (проходящему через каждую вершину ровно один раз) пути или циклу на графе *L(G)*.
3) Для всех $m, k \geq 1$ граф *B(m, k)* имеет гамильтонов цикл.

Сформулированные утверждения позволяют определить достаточно эффективный алгоритм построения слов де Бройна — для этого достаточно построить граф *B(m, k)*, что требует объема памяти и времени $O(m^k)$, и найти в нем эйлеров цикл, что можно сделать за время, пропорциональное количеству ребер графа, т.е. опять за $O(m^k)$, и используя такой же объем памяти. Длина слова де Бройна равна m^k+k-1, т.е. тоже $O(m^k)$, следовательно, такой алгоритм оптимален по порядку.

Можно улучшить «внутренние» показатели эффективности такого алгоритма, т.е. уменьшить объем памяти, занимаемый внутренними структурами данных и время их обработки, не учитывая внешнюю память и время, используемые для вывода результата.

Различные алгоритмы для порождения слов де Бройна довольно часто упоминаются в литературе, в том числе и алгоритмы с лучшими показателями «внутренней» эффективности. Часть из них основана на неожиданной связи между словами де Бройна и словами Линдона (Lyndon words). *Слово Линдона* длины k в алфавите мощности m — это лексикографически минимальный представитель m-цикла, т.е. класса эквивалентности m-слов по циклическим перестановкам их символов. Оказывается, что верно следующее утверждение.

Утверждение 4. [11]

Конкатенация лексикографически упорядоченной последовательности всех слов Линдона длин, делящих k, в некотором алфавите дает лексикографически минимальный цикл де Бройна шага k в том же алфавите (для получения из него слова де Бройна достаточно добавить первые *k-1* символов из начала в конец).

Эффективный (требующий ограниченного константой «внутреннего» времени на построение одного слова Линдона) алгоритм построения слов Линдона и слов де Бройна на их основе представлен в [20]. Другие алгоритмы можно найти в [21–28]. В работе [29] эмпирически сравнивается эффективность по времени алгоритмов из [20], [25] и [27], и последний алгоритм демонстрирует наиболее высокое быстродействие.

3.2. Продолжение слов де Бройна

Нас, однако, интересует еще вопрос возможности продолжения слова де Бройна шага k до слова де Бройна шага $(k+1)$. Ответ на этот вопрос дается следующим утверждением.

Утверждение 5.

1) При $m = 1$ для всякого $k \geq 1$ единственное слово де Бройна шага k представляет собой слово 0^k, соответственно, оно может быть продолжено до слова де Бройна шага $(k+1)$ — 0^{k+1}.
2) При $m \geq 3$ для всякого $k \geq 1$ любое слово де Бройна шага k может быть продолжено до слова де Бройна шага $(k+1)$.

   Это значит, что для $m = 1$ или $m \geq 3$ существуют бесконечные слова, каждое начало которых имеет число разных подслов, максимальное среди слов той же длины.
3) При $m = 2$ ни для какого $k \geq 2$ ни одно слово де Бройна шага k не может быть продолжено до слова де Бройна шага $(k+1)$, но любое такое слово может быть продолжено до слова де Бройна шага $(k+2)$.

   Для $k = 1$ слова де Бройна — *01* и *10*; каждое из них продолжается до слова де Бройна шага 2 — *01100* и *10011*.

Поскольку пункт 1 достаточно очевиден, докажем основные утверждения из пунктов 2 и 3. Отдельные утверждения этих пунктов доказаны в [30-32]. В [32], кроме того, несколько иначе доказано утверждение, что именно продолжения слов де Бройна имеют максимально возможное количество разных подслов.

Слово де Бройна шага k соответствует эйлерову циклу в графе $B(m, k)$ и гамиль-тонову в графе $B(m, k+1)$. Если выбросить все ребра этого гамильтонова цикла, при $m > 2$ граф $B(m, k+1)$ останется связным (см. ниже) и эйлеровым, посколь-ку входящие и исходящие полустепени всех вершин уменьшатся на 1 и оста-нутся равными. Поэтому можно дополнить выброшенный гамильтонов цикл до эйлерова цикла в $B(m, k+1)$, который соответствует искомому продолжению.

Связность $B(m, k+1)$ не нарушится от выбрасывания ребер гамильтонова цикла, поскольку каждую его вершину v можно соединить с 0^{k+1} $(m-1)$-м непересекающимся путем, и наоборот, 0^{k+1} можно соединить с v таким же количеством непересекающихся путей. Для доказательства этого достаточно рассмотреть следующую конструкцию. Пусть в v $i \geq 0$ первых символов равны 0, и x — первый символ v, отличный от 0, т.е. $(i+1)$-й. Тогда $(m-2)$ искомых

путей соответствуют словам, полученным конкатенацией 0^{k+1}, символа y, не равного 0 или x, и v. Еще один путь получается, если взять конкатенацию 0^{k+1} и конца v, начинающегося с индекса $(i+1)$. В полученных словах все подслова длины k, кроме начального и конечного, различны. Аналогично показывается существование $(m-1)$-го обратного пути из v в 0^{k+1}.

Для $m = 2$ и $k > 1$ аналогичное выбрасывание ребер гамильтонова цикла оставит несвязанными с остальными вершины 0^k и 1^k, в каждой из которых имеется по петле. Значит, хотя бы одна из этих петель не может войти ни в какое продолжение исходного гамильтонова цикла, соответственно, никакое его продолжение не будет соответствовать слову де Бройна шага $(k+1)$. Доказательство того, что 2-слово де Бройна шага k можно продолжить до 2-слова де Бройна шага $(k+2)$, можно найти в [30].

Утверждение 6 (гипотеза).

При $m = 2$ для всякого $k \geq 2$ существует слово де Бройна шага k, которое можно продолжить до слова длины $m^{k+1}+(k+1)-2$, содержащего все возможные 2-слова длины $(k+1)$, кроме одного.

Это значит, что при $m = 2$ некоторые (не все!) слова де Бройна (назовем их *продолжающимися*) можно продолжить почти до нужной длины (на 1 меньшей длины следующего слова де Бройна), а значит, мы можем модифицировать предложенный выше способ построения слова длины N с максимально возможным числом разных подслов для $m = 2$ следующим образом: находим минимальное k, такое что $2^{k+1}+(k+1)-1 > N$, строим продолжа-ющееся слово де Бройна шага k и продолжаем его до длины N. Поскольку $N \leq 2^{k+1}+(k+1)-2$, мы сможем это сделать, и полученное слово будет иметь максимально возможное число разных подслов — для длин, не превосхо-дящих k, это следует из того, что его начало является словом де Бройна шага k, а для б**о**льших длин из того, что все такие подслова в нем различны.

Доказательство этого утверждения автору неизвестно, хотя оно кажется истинным. Примеры продолжающихся 2-слов де Бройна шагов *2, 3, 4, 5, 6* приведены в Таблице 1. В этих примерах продолжающееся слово де Бройна отделено точкой от продолжения, которое дополняет его до слова, содержащего все слова длины $(k+1)$, кроме 1^{k+1}.

k	**Продолжающееся *2*-слово вместе с продолжением**
2	00110.1000
3	0001011100.11010000
4	0000100110101111000.1110110010100000
5	0000010101100100011011111010011100 00.1111001100010010111011010100 0000
6	0000001010001001011100100001100011101111110100110101011011001111000 00. 1111100110010101001100111000101100001000110111010111011010000000

Таб. 1. Продолжающиеся 2-слова де Бройна.

Примеры можно искать как гамильтоновы пути в $B(2, k+1)$, начинающиеся в 0^k и заканчивающиеся в 10^{k-1}. Предположительно, такой путь всегда можно выбрать так, чтобы он пересекался с каждым циклом графа $B(2, k+1)$ (т.е. имел хотя бы одно общее с циклом ребро), за исключением петель 0^{k+1} и 1^{k+1}; тогда он дополняется до «почти эйлерова» пути, который не покрывает только ребро 1^{k+1}.

4. Универсальные покрывающие последовательности

Теперь рассмотрим другой способ интерпретации «как можно более разнообразного» поведения m-слова. Можно считать, что тестируемая система является некоторым небольшим конечным автоматом, и строить искомое слово как покрывающее *все возможные* конечные автоматы с числом состояний, меньшим некоторого k (нужно рассматривать только сильно связные автоматы, в которых все входные стимулы допустимы во всех состояниях, иначе их нельзя покрыть с помощью одного слова; кроме того, ограничимся пока детерминированными автоматами). Нужное k можно выбрать как максимальное, покрывающее все автоматы слово длины, не большей N.

Таким образом, мы ищем m-слова, покрывающие все детерминированные сильно связные конечные автоматы с не более чем k состояниями и m входными символами, определенными во всех состояниях (автоматы, в которых во всех состояниях допустимы одни и те же m символов называют *m-регулярными*). Однако «покрывать» можно разные элементы автомата. Достаточно естественно считать такими элементами все состояния, все переходы, пары смежных переходов и пр. и рассматривать для этих случаев разные слова.

Универсальной покрывающей m-последовательностью или *универсальным покрывающим m-словом шага $k \geq 1$ и глубины $l \geq 0$* (universal covering word) назовем m-слово, которое, будучи подано на вход любому детерминированному сильно связному m-регулярному автомату с k состояниями, определит в нем маршрут, содержащий все возможные маршруты длины l данного автомата. При $l = 0$ считаем маршрутами длины 0 все его состояния. Обозначим множество универсальных покрывающих m-слов шага k и глубины l через $UC(m, k, l)$.

Можно усомниться в том, что наша гипотеза о тестируемой системе как автомате с не более чем k состояниями, хорошо согласуется с реальностью — ведь число состояний в большинстве реальных систем таково, что требующиеся последовательности будут иметь колоссальную длину. Однако такая гипотеза приобретает более глубокий смысл, если считать, что состояния системы разбиваются на не более чем k групп таких, что переход по любому стимулу осуществляется из одной группы в другую или в ту же (т.е. отсутствуют такие стимулы, что из некоторой группы состояний переходы по этому стимулу ведут в состояния нескольких разных групп). При этом иногда можно считать, что различия между состояниями в рамках одной группы

гораздо меньше, чем между состояниями различных групп, и поэтому, прежде всего, важно протестировать поведение системы относительно разных групп ее состояний.

В имеющейся литературе универсальные покрывающие слова практически не упоминаются, в отличие от слов де Бройна. Некоторое количество работ посвящено аналогу универсальных покрывающих слов глубины 0 (т.е. покрывающих все состояния) для *неориентированных* графов под названием *универсальных обходящих последовательностей* (universal traversal sequences, введены Куком, S. A. Cook, в конце 70-х годов прошлого века). В центре внимания этих работ находится не собственно построение таких последовательностей, а одна из проблем теории сложности алгоритмов — как соотносятся классы сложности P-log-SPACE детерминированных алгоритмов, требующих полиномиально-логарифмической памяти, и NP-log-SPACE недетерминированных алгоритмов с такими же требованиями к памяти. Дело в том, что задача построения пути между двумя произвольными вершинами графа является примером NP-log-SPACE задачи, а универсальная обходящая последовательность дает для нее детерминированный алгоритм решения. Тем самым, верхние и нижние границы длины универсальных обходящих последовательностей задают правила преобразования сложности NP-log-SPACE алгоритма для решения некоторой задачи в сложность P-log-SPACE алгоритма для нее же. В первой известной автору статье [33], в которой появилось понятие универсальной обходящей последовательности, было показано, что для любых $m, k \geq 1$ существует универсальная обходящая последовательность для m-регулярных неориентированных графов с k вершинами длины $O(m^2k^3 \log k)$, а при $m = 2$ даже $O(k^3)$.

Для ориентированного случая, который нас интересует, все обстоит несколько сложнее — длина универсальных покрывающих слов как минимум экспоненциальна в зависимости от k. Для доказательства этого достаточно заметить, что такие слова, как минимум, не короче слов де Бройна (см. Утверждение 7).

Доказать, что универсальные покрывающие слова существуют для всех $m, k \geq 1$ и $l \geq 0$, достаточно просто. Для этого заметим, что m-регулярных автоматов с k состояниями конечное множество — число способов направить m переходов из одного состояния равно k^m, число возможных способов их компоновки в автомат k^{km}, а если учесть возможность произвольной перенумерации состояний, отличных от начального, остается $k^{km}/(k-1)!$ неизоморфных автоматов. «Почти все» из них сильно связны (т.е. доля автоматов, не являющихся сильно связными, уменьшается с ростом k и m). Если строить универсальное покрывающее слово достаточно прямолинейно — покрывать один за другим пути длины l в одном автомате, затем в другом и т.д. (при этом на проход в первую вершину очередного непокрытого пути тратится не более $(k-1)$ шагов), то максимум через $(km^l)(l + k - 1)(k^{km}/(k-1)!)$ шагов все такие пути во всех автоматах будут покрыты — $(l + k - 1)$ шаг делается для

того, чтобы покрыть один путь, в каждом состоянии начинается m^l путей, в одном автомате k состояний.

Обозначим через *US(m, k)* множество *m*-слов, содержащих все возможные m-слова длины *k* в качестве подслов. Слова де Бройна являются наиболее короткими словами в *US(m, k)*.

Утверждение 7.

1) Для всех $m \geq 1$ пустое слово лежит в *UC(m, 1, 0)*.
   Для всех $m \geq 1$ слово *012...(m-1)* лежит в *UC(m, 2, 0)*.
   Для всех $m, l \geq 1$ $UC(m, 1, l) = US(m, l-1)$, т.е. в качестве универсального покрывающего слова шага *1* и глубины *l* можно взять слово де Бройна шага *(l-1)*.
2) Для всех $m \geq 1, k \geq 2$ $UC(m, k, 0) \subseteq US(m, k-1)$.
   Т.е. слово может быть универсальным покрывающим шага *k* и глубины *0*, только если оно содержит в качестве подслов все возможные слова длины *(k-1)*. Таким образом, длина такого слова не меньше $m^{k-1}+k-2$.
3) Для всех $m, k, l \geq 1$ $UC(m, k+l, 0) \subseteq UC(m, k, l)$.
   Зная универсальные покрывающие слова глубины *0*, мы будем знать универсальные покрывающие слова для всех глубин, хотя, быть может, и не самые короткие.
4) Для всех $m, k \geq 1$ $UC(m, k+1, 0) = UC(m, k, 1)$.
   Т.е. универсальные покрывающие слова глубины *0* в точности совпадают с универсальными покрывающими словами глубины *1* для шага с номером, меньшим на единицу.
   Следствие: $UC(m, k, 1) \subseteq US(m, k)$.

Для доказательства п. 2 рассмотрим семейство графов с *k* состояниями, изображенное на Рисунке 2.

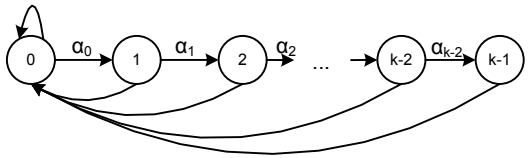

Рис. 2. Семейство "плохих" графов.

В каждом графе этого семейства некоторая выделенная последовательность *(k-1)* символов приводит из начального состояния в *(k-1)*-е, а все остальные символы во всех состояниях ведут в начальное состояние. Всякая последовательность символов длины *(k-1)* встречается в качестве выделенной в одном из графов семейства. Если в слове из *UC(m, k, 0)* нет какой-то последовательности длины *(k-1)* в качестве подслова, то состояние *(k-1)* соответствующего графа не будет покрыто.

Для доказательства п. 3 предположим, что слово из *UC(m, k+l, 0)*, будучи применено к некоторому автомату с *k* состояниями, не покрывает некоторый путь в нем длины *l*. Добавим в этот автомат новые *l* состояний так, чтобы этот путь начинался в том же состоянии, что и раньше, а дальше шел по новым состояниям. Переходы по всем символам из *[0..(m-1)]*, не ведущим вдоль выделенного пути, из новых состояний направим в то состояние, которое было вторым на этом пути в исходном автомате. При этом получится m-регулярный автомат с *(k+l)* состояниями, по-прежнему сильно связный. Поскольку в исходном автомате наше слово не могло покрыть выделенный путь, а только по этому пути можно попасть в *l*-е состояние из добавленных, то в результирующем автомате наше слово не может покрывать это состояние, что противоречит его принадлежности *UC(m, k+l, 0)*.

Для доказательства утверждения п. 4 (осталось доказать включение $UC(m, k, 1) \subseteq UC(m, k+1, 0)$) предположим, что слово из *UC(m, k, 1)* не покрывает некоторое состояние в некотором автомате с *(k+1)*-м состоянием. Поскольку автомат сильно связен, в это состояние ведет некоторое множество ребер и из него выводит хотя бы одно ребро. Перенаправим все ребра, ведущие в это состояние, в состояние, в которое входит это самое выводящее ребро. При этом сильная связность не нарушится, а в автомате останется *k* состояний. Значит, наше слово покрывает все ребра из числа перенаправленных. Рассмотрим то ребро из этого множества, которое покрывается первым. Поскольку оно первое из перенаправленных, путь, покрываемый до него словом в исходном и результирующем автоматах, останется неизменным. Значит, в исходном автомате он должен далее пройти по этому ребру и попасть в непокрытое состояние.

Полученная при доказательстве существования универсальных покрывающих слов верхняя оценка их длины слишком велика — на практике наиболее короткие покрывающие слова оказываются не намного длиннее соответствующих слов де Бройна.

К сожалению, кроме приведенного выше утверждения, автору не много известно о свойствах универсальных покрывающих слов и об алгоритмах их построения.

То, что каждое универсальное покрывающее слово глубины *0* содержит все последовательности определенной длины в качестве подслов, позволяет предположить, что можно строить такие слова на основе слов де Бройна. Сами по себе слова де Бройна не являются универсальными покрывающими в большинстве случаев (при $k \geq 3$ в п. 2 Замечания 7). Например, минимальная длина элемента $UC(2, 3, 0) = UC(2, 2, 1)$ равна 6 (*001011* и *110100*), а соответствующие слова де Бройна имеют длину *5* (и универсальные покрывающие слова в данном случае — даже не продолжения слов де Бройна *00110, 01100, 10011, 11001*).

Неизвестно, выполнен ли аналог утверждения п. 4 Утверждения 7 для глубин, больших *2* при $k \geq 2$ (при $k = 1$ он точно не выполнен, поскольку *UC(m, 1, l)* совпадает с *US(m, l)*, а, как только что было сказано, *UC(m, 2, 1)* уже

отличается от *US(m, 2)*). Если это так, то можно было бы иметь дело либо только с универсальными покрывающими словами глубины *0* для разных шагов, либо только с универсальными покрывающими словами шага *2* для разных глубин (т.е. искать все универсальные покрывающие слова только на автоматах с двумя состояниями). Последнее свойство выглядит очень сильным, и поэтому есть сомнения в том, что оно выполняется.

Все (с точностью до перестановок символов) известные автору универсальные покрывающие слова минимальных длин сведены в Таблице 2 (могут существовать универсальные покрывающие слова меньшей длины с заданными параметрами — те слова, про которые точно известно, что они имеют минимальную возможную длину, помечены звездочкой, в конце остальных слов стоит точка, кроме того, знаками вопроса помечены слова предположительно минимальной возможной длины).

m, k, l	**Слова из *UC(m, k, l)***
2, 2, 1	001011*
2, 2, 2	01011000100111* 01110110001001*
2, 2, 3	0101011010010000111000111100? 0101011110100100001110001100? 0101100010100100001111011100? 0101101010000100111100011100? 0101101010010011110001110000?
2, 2, 4	00100110001111101101010000010111001010011101101100100001000111011. 00100110001111101101010000010111001010111101100100010011100001101. 00100110001111101101010000010111001010111101100100010011101100001. 00100110001111101101010000010111001010111101100111000010001001101.
2, 2, 5	00000010001100111110101110010110100110111100010101000001110110000100101001000011011010111110110010011100010110110011000100001.
3, 2, 1	0010112022121* 0010112122020* 0010120221211* 0010201220211* 0010202201211* 0010211012022* 0010211020122* 0010212201120* 0010212202011* 0010220201211* 0010221201121* 0010221201211* 0012010220211* 0012022010211* 0012022101121* 0012022121011* 0012102212011* 0012110102022* 0012201121020* 0012202010211* 0012202101120* 0012202102011* 0100201211022* 0100211012022* 0101121220200* 0102122011200* 0120100211022* 0121101002022* 0122011210200* 0122021011200*
3, 2, 2	010021122022120011101210002010212220110201202210. 010021122022120011101210002011021222020102101120.
4, 2, 1	0010213123301120301322? 0010213123303013201122? 0012322133011203102130?
4, 2, 2	0010223212031133033121302011002322003013233210122112313103000211122202032001 0012303023331233202103303121211322323101023301023201330. 0010223212031133033121302011002322003013233210122112313103000211122202032001 0012303023331233202103303121211322323101023301023223133. 0010223212031133033121302011002322003013233210122112313103000211122202032001 0012303023331233202103303121211322323101023301023223133.
5, 2, 1	012032441343102230421401123300434l0.

6, 2, 1	0012342455133521404310532202541130344150435120145234O. 0012344253155413021045203351432240115435024205031235 4.
7, 2, 1	012234546165330410256051436242063115526403213500662440152346513024631543l2.
8, 2, 1	012234456757361607153042031746526432705114725063376241002135577660145402367372423601565313403O7. 012234456757361607153042031746526432705114725063376241002135577660145402367372423601565313407O3.
9, 2, 1	012345678876543210204137586081425368071624735051826374617285703064483152722558408776600113387685123806481731357412436704526183238364101781402.

Таб. 2. Минимальные известные универсальные покрывающие слова.

5. Заключение

Данная статья представляет два возможных подхода к построению тестовых последовательностей при отсутствии какой-либо информации, кроме числа возможных воздействий на тестируемую систему и длины последовательности, которую хотелось бы получить. Оба подхода приводят к построению слов в конечном алфавите, обладающих специфическими комбинаторными свойствами.

Первый подход основывается на предположении, что поведение тестируемой системы определяется фиксированным числом последних оказанных воздействий. Последовательности, построенные на его основе, соответствуют словам де Бройна — самым коротким словам, содержащим в качестве подслов все последовательности определенной длины. Поскольку слова де Бройна имеют помимо тестирования массу других приложений, имеется достаточно много посвященных им работ и различных эффективных алгоритмов их построения.

Второй подход предлагает использовать универсальные покрывающие слова, обеспечивающие покрытие путей некоторой длины во всех регулярных сильно связных детерминированных конечных автоматах с фиксированным числом состояний. Несмотря на наглядность предлагаемой идеи, она, по-видимому, до сих пор не рассматривалась, и найти работы, в которых использовалось бы понятие, эквивалентное определенным выше универсальным покрывающим словам, автору не удалось. Такие слова устроены сложнее, чем слова де Бройна, и пока не найдено ни хорошего описания их свойств, ни удобного аппарата для работы с ними, подобного графам де Бройна, ни достаточно эффективных алгоритмов их построения. Все это — задачи для продолжения исследований.

Другое возможное направление развития — использование знаний пользователя о системе, которые в реальности чаще всего не нулевые, для построения более эффективных и более коротких тестовых последовательностей. Для начала можно использовать знания о том, что

некоторые операции не изменяют состояния системы, некоторые другие, такие, как конструкторы объектов, не зависят от него, впоследствии можно добавить чаще всего известные предусловия операций и т.д.

Литература

1. A. Hartman. *Software and Hardware Testing Using Combinatorial Covering Suites.* Haifa Workshop on Interdisciplinary Applications and Graph Theory, Combinatorics and Algorithms, June 2002. Available at http://www.agedis.de/documents/d435_1/CombinatorialProblemsinSWTesting-finalDraft180703.pdf
2. A. Hartman, L. Raskin. *Problems and Algorithms for Covering Arrays.* Discrete Mathematics 284:149-156, 2004. Available at http://www.agedis.de/documents/d434_1/AlgorithmsForCoveringArraysPublication191203.pdf
3. C. J. Colbourn. *Combinatorial aspects of covering arrays*. In Proc. of Combinatorics 2004, Capomulini, Italy, September 2004. To appear in Le Matematiche (Catania). Available at http://www.dmi.unict.it/combinatorics04/documenti%20pdf/colbourn.pdf
4. H. Fredricksen and J. Maiorana. *The Baltimore Hilton Problem.* Technology Review, v. 83, no. 7, June 1980.
5. F.-S. Marie. *Solution to problem number 58.* L'Intermédiaire des Mathématiciens, 1 (1894), 107–110.
6. N. G. de Bruijn. *A Combinatorial Problem.* Koninklijke Nederlandse Akademie van Wetenschappen 49, 758-764, 1946.
7. M. H. Martin. *A problem in arrangements.* Bulletin of American Mathematical Society, 40:859–864, 1934.
8. I. J. Good. *Normally recurring decimals.* J. London Math. Soc. 21:167–169, 1946.
9. D. Rees. *Note on a paper by I. J. Good.* The Journal of the London Mathematical Society, 21:169–172, 1946.
10. Д. Кнут. Исскуство программирования. Том 1. Основные алгоритмы. Вильямс, 2002.
11. H. Fredricksen. A survey of full length nonlinear shift register cycle algorithm. SIAM Review, 24(2):195–221, 1982.
12. F. J. MacWilliams and N. J. A. Sloane. *Pseudo-random sequences and arrays.* Proc. IEEE 64:1715–1729, 1976.
13. C. D. Savage. *A survey of combinatorial Gray codes.* SIAM Review, 39(4):605–629, 1997.
14. Fan Chung and J. N. Cooper. *De Bruijn Cycles for Covering Codes.* 2003. Available at http://arxiv.org/abs/math/0310385
15. L. Zhang, B. Curless, S. M. Seitz. *Rapid shape acquisition using color structured light and multi-pass dynamic programming.* In Int. Symposium on 3D Data Processing Visualization and Transmission, pages 24–36, Padova, Italy, June 2002.
16. J. Pagès and J. Salvi. *A new optimised De Bruijn coding strategy for structured light patterns.* 17th International Conference on Pattern Recognition, ICPR 2004, Cambridge, UK, 23–26, August 2004.
17. S. W. Golomb. *Shift Register Sequences.* Aegean Park Press, Laguna Hills, CA, USA, rev. ed., 1981.
18. A. Lempel. On a homomorphism of the de Bruijn graph and its applications to the design of feedback shift registers. IEEE Transactions on Computers, C-19:1204–1209, 1970.

19. H. Robinson. *Graph Theory Techniques in Model-Based Testing.* 1999 International Conference on Testing Computer Software, 1999. Available at http://www.geocities.com/harry_robinson_testing/graph_theory.htm
20. H. Fredricksen, J. Maiorana. *Necklaces of beads in k colors and k-ary de Bruijn sequences.* Discrete Math., 23:207–210, 1978.
21. T. Etzion, A. Lempel. *Algorithms for the generation of full-length shift-register sequences.* IEEE Transactions on Information Theory, 30:480–484, 1984.
22. T. Etzion. An algorithm for constructing m-ary de Bruijn sequences. Journal of Algorithms, 7:331–340, 1986.
23. R. A. Games. *A generalized recursive construction for de Bruijn sequences.* IEEE Transactions on Information Theory, 29:843–850, 1983.
24. C. J. A. Jansen, W. G. Franx, D. E. Boekee. *An efficient algorithm for the generation of DeBruijn cycles.* IEEE Transactions on Information Theory, 37:1475–1478, 1991.
25. A. Ralston. A new memoryless algorithm for de Bruijn sequences. Journal of Algorithms, 2:50–62, 1981.
26. E. Roth. *Permutations arranged around a circle.* The American Mathematical Monthly, 78:990–992, 1971.
27. S. Xie. *Notes on de Bruijn Sequences.* Discrete Mathematics, 16:157–177, 1987.
28. F. S. Annexstein. *Generating de Bruijn Sequences: an Efficietn Implementation.* IEEE Transactions on Computers, 46(2):198–200, 1997.
29. M. Vassallo, A. Ralston. Algorithms for de Bruijn sequences — a case study in the empirical analysis of algorithms. The Computer Journal, 35:88–90, 1992.
30. M. J. O'Brien. *De Bruijn graphs and the Ehrenfeucht-Mycielski sequence*. Master's thesis, Mathematical Sciences Department, Carnegie Mellon University, 2001.
31. A. Iványi. *On the d-complexity of words*. Ann. Univ. Sci. Budapest. Sect. Comput. 8, 69-90, 1987.
32. A. Flaxman, A. W. Harrow, G. B. Sorkin. Strings with maximally many distinct subsequences and substrings. Electronic Journal of Combinatorics 11(1), 2004. Available at http://www.combinatorics.org/Volume_11/PDF/v11i1r8.pdf
33. R. Aleliunas, R. M. Karp, R. J. Lipton, L. Lovász, C. W. Rackoff. *Random walks, universal traversal sequences, and the complexity of maze problems*. In Proc. of 20-th Annual Symposium on Foundations of Computer Science, San Juan, Puerto Rico, October 1979, pp. 218–223.