Skobtsov 7 digital evolutionary synthesis.doc
Со страниц 1 по 21(включительно) или с начала Лекции 7 по 7.4.4

Лекция 7

Эволюционное цифровое VLSI проектирование

Описание: В этой лекции рассматриваются основные подходы эволюционного синтеза цифровых микросхем на различных уровнях проектирования. Эволюционный алгоритм синтеза описывается на функциональном уровне, где цифровая схема представлена булевой функцией. Показаны схемы кодирования, определена фитнесс-функция и прокомментированы опытные данные эволюционного синтеза для некоторых цифровых схем. Эволюционный синтез представлен на входном уровне, где определены основные компоненты генетического алгоритма: схема кодирования, фитнесс-функция, генетические операторы. Для эволюционного синтеза используется FPGA технология, которая описывает аппарат декартового генетического программирования. Показан основной подход используемый в эволюционном оборудовании.

Ключевые слова: синтез, цифровая схема, генетический алгоритм, генетические операторы, функция репродукции, оператор кроссинговера, оператор мутации.

Использование генетических алгоритмов в области VLSI проектирования и оптимизации расположения было исследовано более интенсивно, по крайней мере до середины 90-х, по сравнению с другими областями применения ГА. Основной причиной этого является то, что, в данном случае, использование ГА является простым и вызвано исключительно для оптимизации. Без этого класса приложений не возникли бы инновационные разработки.

ГА используются в качестве инструмента оптимизации, а не в качестве инструмента поиска, как образца гена пространства новых конструкций. Тем не менее, это является важной областью применения с точки зрения снижения затрат (площади) в производстве интегральных схем, в основном, в области цифровых чипов.

Эсбенсен и **Дрекслер** предлагают следующие признаки, которыми может быть охарактеризована система автоматизированного проектирования (CAD) электронных [1,2]:

Глобальные, мультимодальные и многоцелевые

- NP-трудные задачи
- Взаимозависимые проблемы

• Высокие задачи с ограничениями

Для того, чтобы преодолеть эти проблемы, исследователи этой области следовали стратегии искусственного деления этой сложной и многоцелевой задачи на подзадачи, и, что более важно, используя эвристику вместе с ГА. В этой конкретной области применения, исследователи сосредоточились на том, что конкурировать с современным проектированием в САПР, и, для достижения этой цели, они предпочитают широко использовать проблемно-ориентированные знания, а не чистый эволюционный алгоритм. С точки зрения исследования ГА, этот класс приложений, не так трудно, использовать, как в случае аналоговых оптимизационных схем.

Эволюционные алгоритмы широко используются в САПР цифровых устройств, в основном на следующей стадии:

1) логическое проектирование (синтез);

2)физическое проектирование (разложение, размещение, ярусное планирование, маршрутизация каналов, размещение, ...);

3) тестирование

В этой области применения, ГА был в состоянии поставить конкурентоспособные решения с точки зрения качества, но, до сих пор, они не являются конкурентоспособными с точки зрения выполнения. Эта область исследований включает в себя приложения, охватывающие большинство вопросов, связанных с осуществлением цифровых интегральных схем, начиная с логического синтеза, проходя через отображение технологии и физическое проектирование(размещение и маршрутизация), и заканчивая на стадии тестирования. Мы опишем в этом разделе эти три уровня оптимизации.

Различные типы кодирования решения, генетические операторы и фитнес-функции разработаны и применяются в зависимости от рассматриваемой задачи.

7.1 Эволюционное проектирование

В проектировании (синтезе) фазы цифровых схем эволюционные методы применяются на различных уровнях: 1) уровне транзисторов, 2) на вентильном уровне 3) функционального уровня [2]. Эволюционный алгоритм используется, чтобы узнать схему конфигурации, соответствующей данной спецификации в синтезе. В этом случае конфигурация схемы представляет либо двоичную строку, либо более сложные структуры, которые переведены на схеме (рис.7.1).

Рисунок 7.1 Эволюционный алгоритм

Программа моделирования используется для вычисления фитнесс-функции значения. Некоторые учебные последовательности ввода часто используемые в цифровой цепи сравнивают свои ответы с выходными желаемыми ответами. Как правило, целью является минимизация разницы между результирующими ответами и прогнозируемым; задержки распространения сигнала от входа до выхода схемы указаны в схеме рабочей скорости; площадь кристалла и т.д.

В эволюционном синтезе небольших схем используется уровень транзистора, когда элементы - резисторы, конденсаторы и транзисторы. На этом уровне проектируются новые типы вентилей триггеров. Надо отметить, что ГА позволяют просто формализовать задачу параметрического синтеза. В простейшем случае хромосома состоит из значений параметров. Но генетическое программирование(ГП) позволяет намного больше - формализовать проблемы структурного синтеза – поиск структуры схемы.

Вполне возможно, в результате графа хромосомы представление, которое описывает структуру схемы.

Уровень вентилей двоичного кодирования и классических генетических операторов скрещивания и мутации используются намного чаще. На этапе логического синтеза основной целью является получение минимального представления булевой функции по отношению к используемой технологии.

Схемы состоят из нескольких основных функциональных компонентов, например, сумматоров и мультиплексоров, в случае эволюционного проектирования на функциональном уровне. Для всех, измерение хромосомы остается равным размерности уровня вентилей, но сложность синтезированных устройств существенно выше. Этот подход был применен в дизайне многих реальных схем, для примера, различные цифровые фильтры, фильтры изображения, мультипликаторы и т.д.

Проблемным местом эволюционного синтеза является стоимостная оценка ценных фитнесс-функций. Схемы с N входами индивидуальны под

прямым подходом. А для того, чтобы оценить значение фитнесс-функции, значения по булевой функции на всех 2^n входных наборах должны рассчитываться для каждой цепи отдельно.

7.2 Эволюционный синтез комбинационной логики цифровых схем на функциональном уровне

Идея использования эволюционных алгоритмов для синтеза схем впервые предложенная в [3], была изначально протестирована в области цифровых схем. Есть много причин для этого: цифровые системы используются более широко, чем аналоговые, их моделирование, как правило, более надежны, чем моделирование аналоговых схем, и, самое главное, появление сложных реконфигурируемых цифровых схем, которые могут быть использованы в качестве целевого устройства для реализации эволюционных схем.

Давайте сосредоточиться на последней причине - появлении высокой производительности реконфигурируемых цифровых чипов. Эта тема получит дальнейшее развитие в следующей главе, поэтому мы просто дадим здесь поверхностное описание. Последнее поколение реконфигурируемых цифровых чипов, помимо отображения очень короткого времени реконфигурации, также очень гибкое, в том смысле, что могут быть реализованы многие различные механизмы цифровых вентилей. Эти различные механизмы включают, например, сочетание элемента AND-OR и AND-XOR логики. Реконфигурируемые чипы позволяют осуществлять последовательные а также комбинационные схемы. Эти чипы являются очень привлекательными для реализации цифровых схем, синтезирования эволюционных систем, так как последние обычно состоят из нетрадиционных механизмов логических вентилей.

Основной проблемой применения эволюционных систем для синтеза цифровых схем является то, что уже есть очень эффективные автоматические CAD-инструменты для цифрового проектирования, такие как ESPRESSO и SIS15. Таким образом, даже при том, что синтез цифровых схем интересный испытательный стенд для сравнения эффективности различных эволюционных алгоритмов, он часто уступает производительности по сравнению с обычными инструментами. Это связано с тем, что когда использовалась логическая парадигма, она разрабатывалась людьми, так, такие парадигмы как сумма произведений, обычно являются эвристиками, которые могут найти оптимальные схемы. Эти эвристики чаще всего включены в ранее упомянутом программном обеспечении САПР. Тем не менее, есть три точки которые заставляют нас исследовать эволюционное проектирование цифровых схем:

- Нельзя сказать, что разработанная людьми логическая парадигма, такая как сумма произведений, всегда обеспечивают оптимальное решение.
- Учитывая вышеизложенное, весьма вероятно, что инструмент, который способен изучить всю область топологий схемы, в том числе нетрадиционные топологии, будет превосходить эвристики, которые создал человек.
- Всякий раз, когда нужно преодоление трудностей технологического характера, таких, как какие-то ограничения в чипе ресурсов, традиционные методы могут оказаться в состоянии найти оптимальное решение проблемы.

Мы начинаем эту главу, анализируя три возможных представления цифровых схем: функциональный уровень; уровень вентилей и представления транзисторного уровня. Эти представления различаются по степени сложности схемы основных строительных блоков. Мы исходим из описания схемы оценки методов применяемых и, впоследствии, описанных в четырех тематических исследованиях: комбинационной цепи эволюции; последовательных эволюционных схем; эволюции схем на основе транзисторной логики.

7.2.1 Представление на функциональном уровне

Учитывая, что любая булевая функция может быть представлена как сумма произведений выражения, мы задумали представление генотипов в виде набора генов, которые кодируют любое возможное произведение булевых переменных, полный или неполный. Для цифровой схемы с N входами, каждый ген будет иметь n локусов (в биологии означает местоположение определённого гена на генетической карте хромосомы), которые могут принимать три значения: 0, если соответствующая входная переменная дополняется; 1, если входная переменная находится в первоначальном виде (не дополняется); и 2, если входной переменной нет. Это можно увидеть на рис. 7.2, который содержит представление гипотетической булевой функции трех переменных (a, b, и c), f = $\overline{a} \cdot b \cdot c + a \cdot \overline{c} + a$, состоящей из трех произведений, в итоге хромосомы состоят из трех генов, каждый из трех локусов. Как видно, число 2s в булевом условии определяет его скупость(экономию).

Таким образом, мы будем называть порядок булевого выражения, как число 2s в соответственном гене, так что более высокий порядок генов наиболее экономный.

Использование представления переменной длины в этой случае является естественным выбором, поскольку ни один не знает, в принципе, из скольких членов состоит минимальное решение, т.е. степень упрощения булевой функции [1].

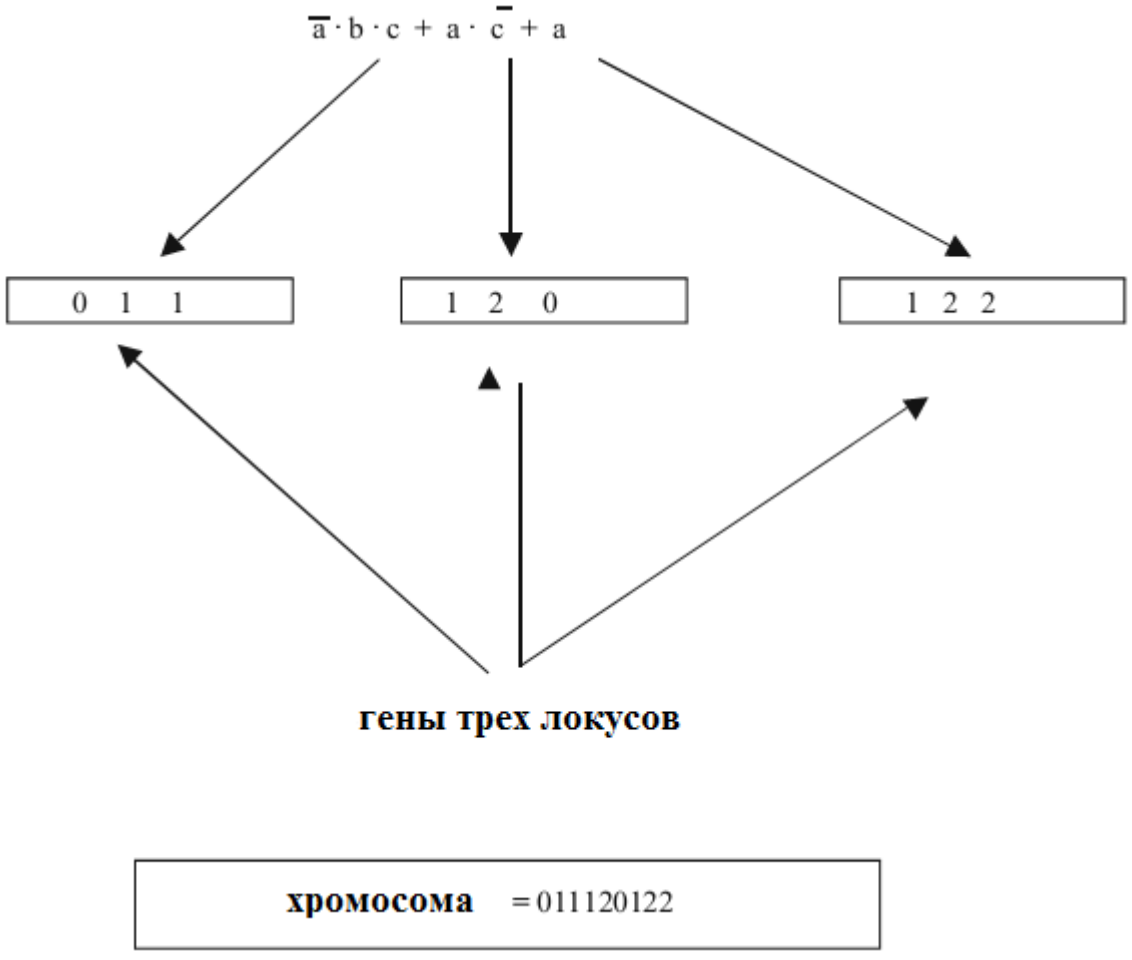


Рисунок 7.2 Представление функционального уровня, основанное на сумме произведений парадигмы: гипотетический пример Булевого выражения и его представления хромосомы.

Так как мощность алфавита этого представления равняется 3, есть в общей сложности 3^n гена в области поиска, где n - общее количество входов. ГА должен искать набор генов, которые соответствуют конкретной спецификации схемы, данный в форме таблицы истинности, которая решает проблему оптимальным способом. С учетом конкретной таблицы истинности, обычно есть много различных логических выражений, которые удовлетворяют спецификации, но только одно, которое решает проблему оптимальным образом. Это представление достаточно для применения продвинутых парадигм. Это - типичный пример нетривиальной многомодальной задачи, из-за следующих причин:

- Практический интерес решения комбинационных функций обычно состоит из многих генов
- Эти гены, как правило, отобираются у разных хромосом эволюционного процесса, то есть, задача собирания вместе желаемых генов только в одной хромосоме сильно увеличивает сложность проблемы.

• Гены, которые решают проблему могут быть отдалены друг от друга (с использованием критерия генетического расстояния), и будут тогда расположены на разных пиках в фитнесс-пространстве.

• Набор генов, который необходим для решения конкретной задачи обычно представляет очень разные индивидуальны фитнесс-значения, будучи тогда расположены на вершинах различных высот в фитнес-пространстве.

Рисунок 7.3 разъясняет вопросы, упомянутые выше. Он иллюстрирует фитнесс пространство 4ех-входной булевой функции, оптимальное решение которой состоит из двух произведений термов. Первое произведение терма, состоит только из входной переменной "a", значение "a" выше фитнесс-значения второго произведения терма, который состоит из произведения "b" "c", "d". В следующем разделе будет описано, что это вызвано тем, что первый член произведения, будучи более экономным, охватывает больше случаев таблицы истинности, чем второй член. Соответствующие гены этих двух произведений, (1 2 2 2) и (2 1 1 1), соответственно, не близки друг к другу, поскольку их символы различны во всех четырех локусах. Следовательно, они расположены на вершинах различной высоты, таким образом, проиллюстрированные проблемы, описаны выше. В данном случае, ген (1 22 2), будет легче найти, чем ген (2 1 1 1), из-за его более высокому фитнесс-значению.

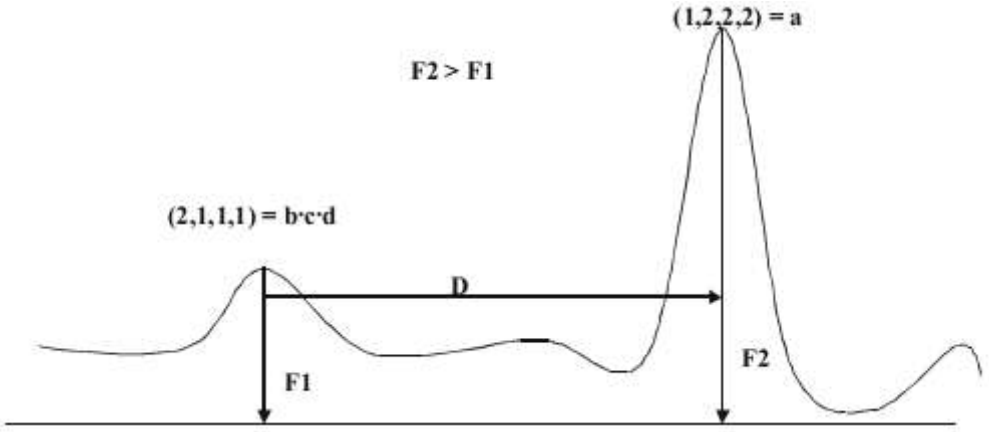


Рисунок 7.3 Фитнесс пространство для конкретной комбинационной функции. D представляет собой расстояние между двумя решениями. F1 и F2 высоты вершин (фитнесс-значения) двух генов (2,1,1,1) и (1,2,2,2) соответственно.

7.2.2 Фитнесс оценка на функциональном уровне

По сравнению с аналоговыми схемами, моделирование и, следовательно, оценка цифровых схем требует меньше вычислений, будучи быстрее по скорости. Кроме того, они могут быть легко достигнуты

рукописными симуляторами. Различные процедуры оценки могут быть использованы, в зависимости от уровня представления использованного ГА. В этом разделе описываются процедуры оценки относительно трех уровней представления описаных ранее.

Функция оценки, в этом уровне представления, считает производительность схемы через количество попаданий в выходную схему, по сравнению с указанной таблицей истинности. Например, в случае схемы с n входами, цель состоит в том, чтобы развивать схему, которая может достичь 2n попаданий, соответственно размеру таблицы истинности. Таким образом, в общем виде.

Однако, эта функция оценки может быть обманчива. Всякий раз, когда развиваются цифровые функции, целью является достижение полностью соответствующей схемы, где фитнесс значением является максимальное количество попаданий. Например, схемы достижения производительности в 99% попаданий не имеют никакого интереса к проектированию. Для ГА очень легко попасть в ловушку в этот вид локального оптимума. Для того чтобы избежать этих ошибок, необходимо включить другие условия в функции оценки фитнесс-значения. В случае сумма произведений представления целевой функции изменится на:

Фитнесс-значение = Попадания – k*Штрафы (7.2)

Штрафы относятся к вышеупомянутым ловушкам, которые могут заставить ГА застревать в локальном оптимуме, и k - постоянная величина, которая будет объяснена позже в этом разделе. В функции оценки есть два источника штрафа:

- Для каждой пары повторных генов в хромосоме термин штрафа увеличен одной единицей, то есть, функция оценки пригодности заставляет ГА достигать хромосом с не повторными генами.
- Гены, которые дают исходящим “1”, где таблица истинности налагает “0” также, способствуют терму штрафа. Причина состоит в том, что, даже при том, что этот вид гена может представить высокое число попаданий, он никогда не должен принимать участие в окончательном решении.

Мы достигли этих условий посредством экспериментального наблюдения, и когда функция оценки пригодности считала только число попаданий, самые пригодные хромосомы составлялись из многих повторных генов, и некоторые из них не были жизнеспособными, как выражено вторым условием выше. Этот вид гена выиграл большое количество попаданий, но не мог принять участие в окончательном решении.

Следующий рисунок 7.4 иллюстрирует этот недостаток на примере

a b c	ab	$\bar{a}bc$	a	ab + $\bar{a}bc$(цель)
0 0 0	0	0	0	0
0 0 1	0	0	0	0
0 1 0	0	0	0	0
0 1 1	0	0	0	0
1 0 0	0	0	1	0
1 0 1	0	1	1	1
1 1 0	1	0	1	1
1 1 1	1	0	1	1
Hits	7	6	7	

m1 = ab
m2 = $\bar{a}bc$
m3 = a

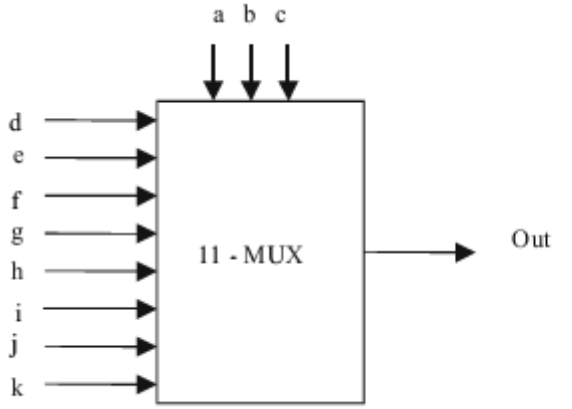
Рисунок 7.4. Пример проблемы локального оптимума, который появляется, когда функция фитнесс оценки берет во внимание только количество попаданий.

На рисунке показана гипотетическая булевая функция от 3 входов и 2 случаев в таблице истинности. Целевая функция оптимально удовлетворяет термам m1 и m2, чьи попадания равны 7 и 6 соответственно. Тем не менее, терм m3 также насчитывает большое количество попаданий, 7 из 8, а значит, более вероятно по сравнению с m2, что попадания будут найдены. Поскольку мы используем сумму произведений парадигмы, m3 не может быть частью решения, так как это вставит 1 выход в случае 100 (строка 5), где таблица истинности указывает 0. Этот терм будет указываться новой функцией оценки пригодности, описанной в уравнении (7.2).

Терм «k» в уравнение (7.2) является постоянной, установленной пользователем, который будет определять силу штрафа терма в функции оценки пригодности. После многих экспериментов, авторы обнаружили наилучшие результаты при значениях k, около 2^n, где п-число входных переменных. Это можно объяснить тем, что, в данном случае, оба терма фитнесс-функции, количество попаданий и штрафов, будут иметь приблизительно в одинаковый диапазон.

7.2.3 Примеры для изучения комбинационных схем на функциональном уровне

Здесь мы описываем две серии экспериментов, эволюцию мультиплексоров и четность функций. Эволюция мультиплексоров и четности схем широко применяется для оценки эффективности эволюционных алгоритмов [7]. Эти приложения не ставят себе цель обнаружить новое или более эффективное проектирование, но оценивание потенциала эволюционных алгоритмов вновь открывают проектирование сделанное людьми. Здесь мы опишем эволюцию 11-входного мультиплексора посредством использования суммы произведений представления. Это устройство включает в себя три линии выбора и восемь линий передачи данных, как показано на рисунке 7.5.



a	b	c	out
0	0	0	d
0	0	1	e
0	1	0	f
0	1	1	g
1	0	0	h
1	0	1	i
1	1	0	j
1	1	1	k

Рис. 7.5. Блок-схема и таблица истинности из 11-входного мультиплексора.

11-входной мультиплексор определяется 1024 минтермами, которые упрощены до всего восемь термов. Приняв сумму произведений представления, каждый ген кодирует произведение терма в виде (a, b, c, d, e, f, g, h, i, j, k), где переменные определяются на рисунке выше . Как описано выше, каждый локус гена может принимать три различных значения, 0, 1, и 2, ссылаясь на состояние, в котором соответствующая переменная появляется в произведении терма. Эта проблема представляет множество различных решений, однако, есть одно оптимальное решение, которое включает в себя следующие восемь генов:

(0,0,0,1,2,2,2,2,2,2,2)
(1,0,0,2,1,2,2,2,2,2,2)
(0,1,0,2,2,1,2,2,2,2,2)
(1,1,0,2,2,2,1,2,2,2,2)
(0,0,1,2,2,2,2,1,2,2,2)
(1,0,1,2,2,2,2,2,1,2,2)
(0,1,1,2,2,2,2,2,2,1,2)
(1,1,1,2,2,2,2,2,2,2,1)

Есть 311 = 177 147 различных генов в соответствии с нашим представлением. Сложность подбора 8 минимальных термов от общего

числа генов может быть оценена как комбинация из 8 более 177 147, C177, 1478, что составляет примерно 2,41 x 1037. Поэтому, мы имеем мультимодальную задачу, в огромном пространстве поиска, которая не позволяет использовать стандартные эволюционные алгоритмы без улучшений. Также можно отметить, что восемь термов показаных выше можно представить в том же порядке (с учетом количества 2s) и, как следствие, этого то же индивидуальное фитнесс-значение.

Для решения этой проблемы были использованы параллельные ГА[1]. Каждый ГА обрабатывает 30 особей по 150 поколений. Гены, найденные каждым параллельным ГА могут составлять общее решение, так как на их долю приходятся непересекающиеся области пространства геномов. Используя эти критерии, 6 из 10 выполнений параллельных ГА нашли решение, четыре из них нашли оптимальное решение, а два других достигли суб-оптимальных решений. Суб-оптимальные решения не включают в себя 8 генов, описанных выше, которые решают проблемы с использованием более чем 8 термов. Рис. 7.6 показывает среднее фитнесс-значение пяти ГА из десяти выполнений; наилучшее фитнесс-значение 211 = 2,048, не достигнуто ни одним из процессов, так как решение теперь распространяется.

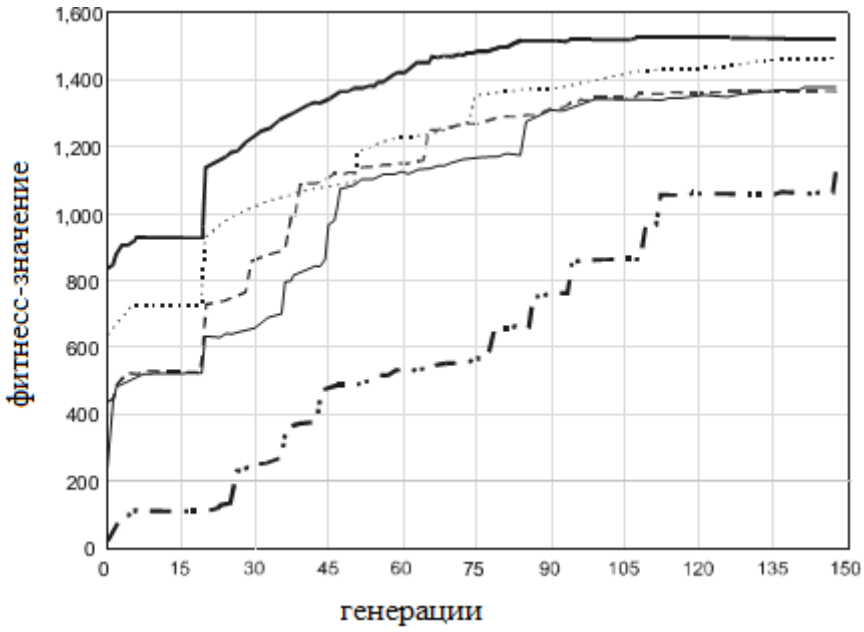


Рис. 7.6 Фитнесс-значение пяти ГА по поколениям(генерациям) для 11-MUX эволюции.

Рассмотрим теперь второй пример комбинационной функции четности детекторов. В частности, мы ориентируемся на 8 входов нечетной функции четности, на которые таблица истинности выводит "1", когда есть нечетное число 1s на входах. В соответствии с нашим представлением, пространство поиска состоит из 38 или 6561 генов. Контрастные с мультиплексором, то есть, в этом случае только один выход, который состоит из 128 генов в виде:

(1,0,0,0,0,0,0,0)
(0,1,0,0,0,0,0,0)
(1,1,1,0,0,0,0,0)
.
.
.

и так далее. Все 128 генов не упрощаемые минтермы, так как все входные переменные присутствуют (не 2 символы). Затем мы можем оценить сложность C6561128, приблизительно равную 10 271. В случае решения этой проблемы, пространство генов было разделено, в этом случае на десять регионов, с одним ГА обычно выборка одного региона. Рис. 7.7 показывает лучшее индивидуальное фитнесс-значение эволюции для каждого ГА; те, в которых выборка региона с большим количеством минтермов достигает более высоких фитнесс-значений. В конце концов, все 128 минтермов были найдены. Тем не менее, это пример, для которого наше представление не является эффективным: около 106 особей были отобраны ГА, требующие слишком много времени вычислений и ресурсов. Этот пример показывает один недостаток суммы произведений представления: четные функции могут быть более легко выражены с помощью оператора исключающего ИЛИ, который не используется в этом представлении.

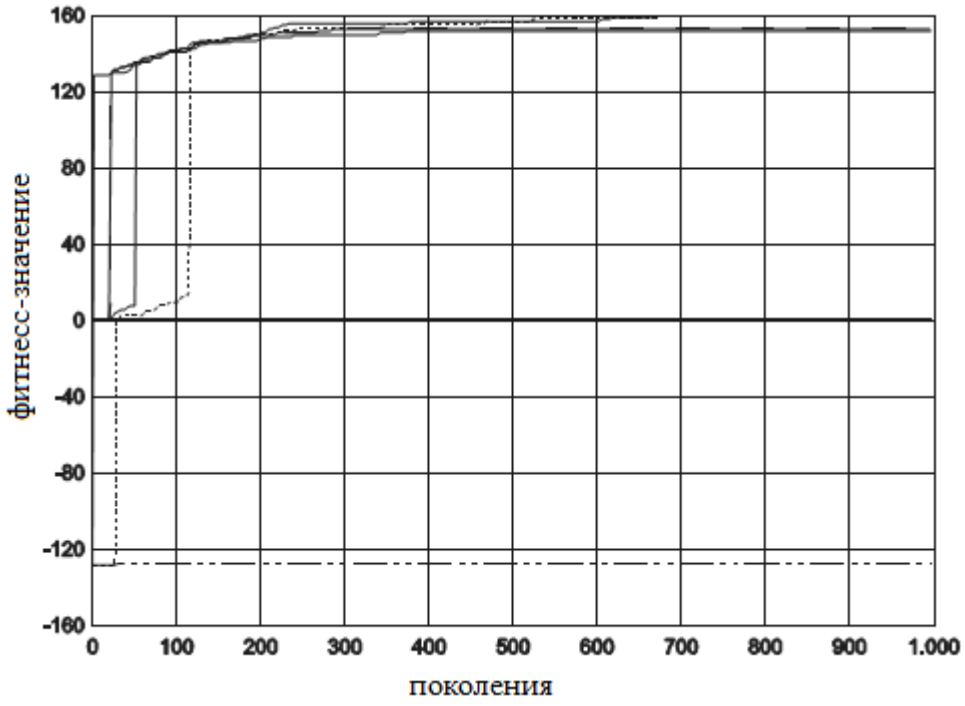


Рис. 7.7. 8-кратный эксперимент: фитнесс-значение 1000 поколений для 10 ГА. Каждая строка соответствует одному ГА; обратите внимание, что пять из них показывают нулевое фитнесс-значение на протяжении всей

эволюции, потому что нет никакого решения в "срезе" пространства поиска, отобранных им.