В. М. КУРЕЙЧИК

ГЕНЕТИЧЕСКИЕ АЛГОРИТМЫ И ИХ ПРИМЕНЕНИЕ

В. М. КУРЕЙЧИК

Генетические алгоритмы и их применение

Таганрог 2002

УДК 681.3.001.63

Курейчик В. М. Генетические алгоритмы и их применение. Таганрог: Изд-во ТРТУ, издание второе, дополненное, 2002, 242с

Рассмотрены основные принципы нового, бурно развивающегося направления «Генетические алгоритмы». Изложены основные задачи и алгоритмы, связанные с разработкой и реализацией генетических алгоритмов. Описаны фундаментальные основы генетических алгоритмов и эволюционного моделирования. Проанализированы архитектуры генетического поиска и модели генетических операторов. Приведены конкретные примеры решения основных задач проектирования на основе генетических алгоритмов. Работа может быть полезна для студентов вузов всех специальностей, обучающихся по направлению «Информатика и ВТ». Она предназначена для специалистов, занятых разработкой интеллектуальных САПР, разработкой новых информационных технологий в науке, технике, бизнесе, экономике, юриспруденции.

Научное издание.

Рецензенты:

Кафедра прикладной математики и вычислительной техники Ростовской-на-Дону государственной академии сельскохозяйственного машиностроения, зав. кафедрой д.т.н., профессор, заслуженный деятель науки РФ Чернышев Ю.О. (г. Ростов-на-Дону),

В.Н.Вагин д.т.н., проф., лауреат премии Президента РФ (МЭИ, г.Москва).

© Курейчик В.М., 2002

Памяти моих дорогих родителей посвящается

Все сущее во вселенной подчинено одним и тем же вечным, неизмененным законам эволюции.

А. Клизовский

Природа – творец всех творцов.

И. Гете

ПРЕДИСЛОВИЕ

Основные достижения в науке и технике в настоящее время, как показывает развитие человечества, возможны только на стыке вроде бы не связанных между собой наук. Такой дисциплиной стало направление «Эволюционное моделирование» и основная ее часть «Генетические алгоритмы». Используя опыты развития природы и человека, генетические алгоритмы применяются для повышения качества создаваемых устройств, эффективности решения задач оптимизации и принятия решений. Это одно из важных направлений науки сегодня. В США и Европе наблюдается бум развития и применения генетических алгоритмов. Прошло 10 международных конференций, опубликовано огромное количество литературы по генетическим алгоритмам, а интерес к этой тематике постоянно растет.

В монографии описаны математические модели генетического поиска, принципы создания и применения генетических алгоритмов. Основная цель монографии - изучение нового перспективного направления в области информатики и оптимизации. Книгу целесообразно изучать после ознакомления с элементами дискретной

математики и методов оптимизации. Автор придерживается терминологии Холланда и Гольдберга, приведенной в их книгах. К сожалению, монографии и учебников по данной теме на русском языке пока нет. Автор предпринял попытку восполнить этот пробел. Материалы монографии основаны на анализе работ Холланда, Гольдберга, Девиса, Гудмана и др. В России сейчас успешно развиваются четыре крупных научных школы по генетическим алгоритмам профессоров Букатовой И.Л., Батищева Д.И., Норенкова И.П. и автора книги. Хочу отметить огромное число исследований по этой тематике по всей России. Интерес к этой области исследований в нашей стране с каждым годом возрастает.

В течение последних пяти лет на постоянно действующих международных конференциях по САПР и по новым информационным технологиям в Гурзуфе и Геленджике генетические алгоритмы стали основным предметом исследований. В 1996г. в Москве прошла Первая международная конференция по эволюционным методам вычислений. Журнал «Известия ТРТУ» раз в год один выпуск посвящает эволюционным вычислениям и генетическим алгоритмам. Но для дальнейшего успешного развития этого направления в России необходимы учебники и монографии различных школ. Хочется надеяться, что этот пробел в ближайшее время будет восполнен.

Для лучшего усвоения материала книги она содержит упражнения, контрольные вопросы и краткие выводы.

Автор благодарен всем сотрудникам кафедры САПР Таганрогского государственного радиотехнического университета за обсуждение материалов книги. Особенно хочу поблагодарить доцента каф. САПР Литвиненко В.А., аспиранта каф. САПР Бондалетова АВ.,

докторантов и аспирантов кафедры САПР за помощь при написании некоторых подразделов. Автор благодарен профессорам Мичиганского государственного университета (США) Т. Бикарту, Э. Гудману, Б. Панчу за предоставленную литературу, возможность проведения исследований и доброжелательное обсуждение материалов, способствующих появлению этой монографии.

Она может быть использована как междисциплинарное пособие, позволяющее сблизить позиции генетиков и специалистов по информатике.

Монография, конечно, не свободна от недостатков. Все замечания, предложения и пожелания будут с благодарностью приняты.

Автор.

Природа – это лучшая и объективнейшая учительница при решении самых трудных вопросов науки.

В.В. Докучаев

ВВЕДЕНИЕ

Задачи синтеза и анализа объектов любой природы относятся к классу комбинаторных и, в большинстве случаев, имеют не одно, а множество решений различной значимости. Для решения таких задач разработано множество методов, способов и алгоритмов. Ядром всех комбинаторных алгоритмов являются операции полного, или сокращенного перебора подмножеств решений, но суть комбинаторного перебора в различных алгоритмах проявляется по-разному. Для поиска лучшего решения можно осуществлять направленный, случайный и направленно-случайный перебор всевозможных значений параметров задачи. Генетические алгоритмы сейчас – это мощный механизм поиска оптимальных решений в задачах науки, техники, производства.

В конце 1960-х годов американский исследователь Джон Холланд и его аспиранты в качестве механизма комбинаторного перебора вариантов решения оптимизационных задач предложили использовать методы и модели механизма развития (эволюционирования) органического мира на Земле. Предложенный механизм получил название «генетические алгоритмы» (ГА). Первым ввел в обиход термин ГА американский ученый Д. Багли в 1967 г.

Монография предназначена также для студентов, обучающихся по всем специальностям направления «Информатика и вычислительная техника», и может быть полезна студентам других специальностей, желающих углубить свои знания в области специальных методов оптимизации. Кроме того, она может быть полезна аспирантам соответствующих специальностей и использована при переподготовке специалистов по информатике.

Монография включает, кроме оригинальных научных исследований автора, методически проработанные исследования Холланда, Гольдберга (США), а также известных российских и зарубежных ученых. Использовалась зарубежная и отечественная научная литература [1-72], список которой приведен в конце книги. В работе содержатся материалы курса лекций «Методы генетического поиска», который автор читает в Таганрогском государственном радиотехническом университете. Частично материалы книги были апробированы при чтении лекций автором в Мичиганском государственном университете (США) в 1992, 1993, 1994 годах.

Монография включает материал, который в отличие от документации на программные системы содержит процедуру генетического поиска, а также процедуры использования средств программных систем для решения конкретных задач проектирования на основе генетических алгоритмов.

Центральной частью работы являются разделы 2-5, где приведены математические основы ГА, описаны все основные процедуры генетического поиска, в которых шаг за шагом комментируется решение практических задач проектирования с

момента постановки задачи до описания алгоритма и выполнения процедур работы.

В монографии рассмотрены описания комбинаторно–логических алгоритмов на графах, а также схемы реальных генетических алгоритмов проектирования. Она включает: исторический экскурс в генетику (раздел 1); основные понятия ГА, математические основы ГА (раздел 2); элементы программной реализации ГА (раздел 3); общие сведения о генетических операторах (раздел 4); подходы и методы решения практических задач на основе ГА (раздел 5).

Материал можно считать усвоенным, если после работы с ним обучаемый сможет выполнить перечисленные ниже действия:

- построение структуры генетического алгоритма, ориентированного на знания о конкретной задаче;
- выбор наилучшего метода селекции; построение эффективной целевой функции; определение числа «выживающих» схем;
- применение классических генетических операторов и разработка модифицированных генетических операторов для реализации поиска;
- создание инструментальной оболочки для решения задач на основе генетического поиска;
- решение задач проектирования и производства с применением методов адаптации, эволюционного моделирования и генетических алгоритмов. .

При решении задач проектирования и создании инструментальной оболочки обучаемый должен уметь выполнять процедуры: селекции; отбора; реализации операторов кроссинговера,

мутации, инверсии, сегрегации, транслокации; генерации генетических алгоритмов.

Если изучение монографии поможет расширению кругозора в области перспективных информационных технологий и становлению специалиста 21-го века, заставит большее внимание уделять изучению поведения окружающей нас природы, то автор будет считать свою задачу выполненной.

Из истории мы черпаем опыт, на основе опыта образуется самая живая часть нашего практического ума.

И. Гердер

1. ИСТОРИЯ РАЗВИТИЯ ЭВОЛЮЦИИ И ГЕНЕТИКИ

1.1. Генетика и основы хромосомной теории эволюции организмов

1.1.1. Краткие исторические сведения

В 1866 г. Иоганн Грегор Мендель, преподаватель училища в Брюнке (Брно), послал статью профессору Мюнхенского университета Нэгели, в которой было сделано величайшее открытие. Мендель верил, что открыл общий закон природы, что им выведены рациональные формулы расщепления признаков в гибридном потомстве. Открытие Менделя осталось незамеченным, а затем через 34 года трое ученых в трех разных странах одновременно повторили его.

Существует три знаменитых закона Менделя [1 – 7]:

Закон однородности и реципрокности (взаимность, эквивалентность). Первое гибридное поколение оказывается полностью однородным.

Закон расщепления. Он относится ко второму поколению и касается расщепления в отношении 1:2:1.

Закон независимой комбинации. Относится к потомкам родителей, отличающихся более чем одной парой признаков, и говорит о том, что признаки наследуются независимо друг от друга.

Мендель пришел к выводу, что наследственность прерывиста (дискретна), что наследуется не большая совокупность свойств, а отдельные признаки.

Мендель предположил, что в половых клетках есть какие-то материальные структуры (позже их назвали генами), ответственные за формирование признаков.

Одна из основных заслуг Менделя – его гипотеза о материальных задатках, которые в двойном комплекте находятся в клетках и которые зародыш получает от обоих родителей.

Исключительную роль в явлениях наследственности играют **хромосомы** – нитевидные структуры, находящиеся в клеточном ядре. Каждый вид характеризуется вполне определенным числом хромосом, во всех случаях оно четное, и их можно распределить попарно. В клетках человека – 46 хромосом, в клетках кукурузы – 10. При делении половых клеток и в процессе оплодотворения каждая хромосома ищет себе подобную. Они сближаются и располагаются параллельно друг другу и почти сливаются. Затем они расходятся. Но во время контакта почти все хромосомы обменялись кусками. Сперматозоид проникает внутрь клетки. Его оболочка, шейка, хвост теряются, а ядро остается. Получается двуядерная клетка. Затем ядра сливаются, и образуется нормальное ядро с нормальным двойным набором хромосом. Получившаяся клетка (**зигота**), многократно делясь, образует зародыш, из которого развивается организм [1-3].

В клетках существует сложный и важный механизм перераспределения хромосом. Каждая клетка организма имеет одинаковое число хромосом. У потомков то же самое число хромосом, причем ровно половина от отца, а половина от матери. Хромосомы передают наследственные признаки. Вильсон в 1900 г. определил, что гены находятся в хромосомах [1].

Две гомологичные хромосомы (одна от отца, а другая от матери) сближаются при созревании половых клеток и обмениваются частями. Это явление назвали crossover (**кроссинговер**, (К)). К происходит между разными генами с разной частотой. Он происходит случайным образом. Модель хромосомы в настоящее время – это нить, на которую, словно бусины, нанизаны гены.

В 1809 г. вышла книга Жан Батиста Ламарка «Философия зоологии». Ламарк говорил об **эволюции** – о возникновении новых форм путем **постепенного** изменения старых и движения от простого к сложному. Ламарк считал, что организмы изменяются под прямым воздействием окружающей среды, что причиной эволюции являются «упражнения» органов и внутреннее стремление к прогрессу.

Дарвин – первый ученый, который усмотрел в живой природе существование общего принципа – **естественного отбора**. В эволюционном учении нужно различать две стороны: учение о материале для эволюции и учение о ее факторах, ее движущей силе. Движущая сила эволюции – естественный отбор.

Теперь ясно, что в основе наследования лежат неделимые и несмешиваемые факторы - **гены**. Именно через отбор и происходит направленное влияние условий жизни на наследственную изменчивость. Сама по себе наследственная изменчивость случайна.

Под воздействием окружающей среды отбираются признаки, которые лучше других соответствуют условиям жизни [1 – 7].

А. Клизовский, анализируя философские учения, писал, что «законы эволюции отбирают все ценное и пригодное для эволюции и отметают в сторону, как мусор как непригодное, все отсталое Они не знают ни пощады, ни сострадания и проводят оценку каждого лишь по степени пригодности или непригодности его для дальнейшего развития» [5].

Де Фриз [3] наследственные изменения отдельных генов назвал **мутациями**. Мутации и служат элементарным материалом для эволюционного процесса. Они закономерно возникают в природных условиях.

Крестным отцом понятия ген и генетики является датский генетик Иогансен [3].

В 1908 г. американец Т. Харди обосновал закон равновесия при свободном скрещивании. Закон определяет условия, при которых соотношение гомозиготных и гетерозиготных индивидуумов остается постоянным.

Пирсон установил закон стабилизирующего скрещивания. В условиях свободного скрещивания равновесие в сообществе устанавливается после первого же скрещивания независимо от исходного состояния гомозиготных и гетерозиготных особей [1].

Приведем факторы, которые меняют генетический состав природной **популяции**:

1. – Мутационный процесс.

Согласно закону Пирсона равновесие устанавливается после первого же скрещивания. Но в каждом поколении возникают новые

мутации, происходит скрещивание, вызывающее новое равновесие и т.д. Т.е. благодаря мутационному процессу состояние равновесия непрерывно меняется.

2. – Изоляция.

Чем меньше размер сообщества, тем более вероятно, что скрытые изменения проявятся.

3. – «Волны жизни».

Изоляция ведет к уменьшению величины популяции путем ее разделения на несколько частей. Но объем популяции может меняться и без ее разделения в зависимости от окружающей среды, т.е. на основе обратной связи.

4. – Отбор.

Факторы 1, 2, 3 – ненаправленные, они изменяют состав сообщества случайным образом. Фактор 4 в отличие от них действует во вполне определенном направлении, а именно в том, которое наилучшим образом соответствует условиям жизни.

Особое положение генов состоит в их уникальности. В хромосомном наборе каждый ген представлен только один раз. Правда в большинстве клеток тела содержится диплоидный (двойной) набор хромосом. Удвоение (увеличение в k раз) числа хромосом – называют полиплоидией. Один из путей возникновения новых видов в природе – образование полиплоидных форм.

В 30-е годы было доказано, что генная мутация – это небольшое химическое изменение. Следовательно, ген имеет химическую природу, являясь молекулой или частью большой молекулы [6].

В 1953 г. с работ Уоткена и Крика началась новая наука – молекулярная генетика. Биохимические особенности живых организмов наследуются по законам, которые открыл Мендель. Все биохимические реакции в клетках управляются сложными белковыми веществами – ферментами. Существует следующая гипотеза: один ген – один фермент. Т.е. функция гена состоит в образовании какого-либо одного фермента. В генах «записаны» планы строения белков – планы всех наследственных признаков [3].

В 1668 г. Реди высказал предположение: «Omne vivum ex vivo» – «Все живое от живого!». Продолжая далее, можно утверждать: каждая клетка от клетки! Каждое ядро от ядра! Каждая молекула от молекулы! [3].

Нуклеиновая кислота играет важную роль в наследственности. Ядерную нуклеиновую кислоту называют **дезоксирибонуклеиновой** (ДНК). Цитоплазматическую нуклеиновую кислоту называют **рибонуклеиновой кислотой** (РНК). Само строение молекулы ДНК говорит о том, что она может обладать удивительным свойством самовоспроизведения. Строение белков закодировано в ДНК, находящейся в хромосомах, и записано в виде последовательности составляющих ее нуклеотидов. Эта информация передается с помощью РНК в цитоплазму: молекулы РНК выходят из ядра и присоединяются к мелким частицам (рибосомам), на которых и происходит сборка белка из аминокислот. Аминокислоты сначала активизируются, получая заряд энергии, необходимой для соединения друг с другом, а затем прикрепляются к молекулам транспортной РНК, которая ставит их на то место, где каждая из них должна быть [7].

Генетический код оказался общим для всего растительного и животного царства на нашей планете. Он практически расшифрован. Всего возможны 64 различных триплета, 61 из них определяет наличие в белке определенных аминокислот. Большинство аминокислот кодируются несколькими аминокислотами. 3 триплета до конца не ясны. Но считают, что они кодируют начало и конец белковой цепочки [6].

Наследственностью человека управляют законы, справедливые для всего органического мира. Жизнь Человеческого сообщества сейчас развивается по другим законам, которые не совсем похожи на те, что управляют сообществом животных и растений.

1.1.2. Носители наследственной информации

Основными носителями наследственности являются хромосомы, находящиеся в ядре клетки, а точнее один из химических компонентов хромосомы – ДНК. Каждая хромосома уникальна морфологически и генетически и не может быть заменена другой либо восстановлена при утере. При потере хромосомы клетка, как правило, погибает. Каждый биологический вид имеет определенное, постоянное количество хромосом. Многие организмы получают один набор хромосом от материнского, а другой, такой же, от отцовского организма. Поэтому у этих организмов в клетке каждая хромосома имеет гомологичного партнера, т.е. хромосомы в парах морфологически и генетически сходны. В процессах наследования признаков определяющую роль играет поведение хромосом при делении клеток. Существует два основных типа деления клеток: **митоз и мейоз**.

Митоз – непрямое деление клеток тела, это механизм точного распределения хромосом между двумя образующимися дочерними клетками. Подготовка клеток к делению происходит в период, называемый интерфазой. В период **интерфазы** в результате самовоспроизведения молекулы ДНК (репликации) образуются две такие же молекулы. Параллельно происходит редупликация хромосом, вследствие чего к моменту деления клетки каждая хромосома будет состоять из двух копий исходной материнской хромосомы – сестринских хроматид.

Мейоз – механизм редукции (уменьшения числа хромосом вдвое). Мейоз состоит из двух делений: редукционного и эквационного.

Редукционное деление. Гомологичные хромосомы (так называют хромосомы, имеющие общее происхождение и поэтому морфологически и генетически сходные) испытывают сильное притяжение друг к другу (конъюгация). Конъюгируют хромосомы строго гомологичными участками. В процессе конъюгации несестринские хроматиды обмениваются друг с другом гомологичными участками. В результате последующего отталкивания образуются переплетения хромосом в виде Х-образных фигур – **хиазм**. Обмен участками и, следовательно, генами, называется кроссинговером. Смысл кроссинговера заключается во внутрихромосомной комбинаторике участков.

Эквационное деление необходимо для редукции числа хроматид, т.к. каждая хромосома состоит из двух сестринских хроматид [7].

Параллелизм в поведении хромосом был замечен еще в 1902 г. Т.Бовери в Германии и Сеттоном в США, однако первые доказательства хранения и передачи наследственной информации нуклеиновыми кислотами были получены лишь в 1944г. Эвери. С этого времени принято датировать начало молекулярной биологии [3].

Термины «ген», «генотип», «фенотип», «аллель» в науку впервые ввел датский генетик В. Иоганнсен. **Ген** – реально существующая, независимая, комбинирующаяся и расщепляющаяся при скрещиваниях единица наследственности, самостоятельно наследующийся наследственный фактор. Совокупность генов составляет **генотип**. **Фенотип** – совокупность всех внешних и внутренних признаков.

Классическая генетика к началу 1940-х годов пришла к пониманию дискретности таких качеств, как наследственность и изменчивость. Это стало возможным в первую очередь благодаря формированию теории гена в работах школы Т. Моргана. Основные положения этой теории можно сформулировать следующим образом [2, 3].

- Все признаки организмов находятся под контролем генов.
- Гены – элементарные единицы наследственной информации, они находятся в хромосомах.
- Гены могут изменяться – мутировать.
- Мутации отдельных генов приводят к изменению отдельных элементарных признаков, или фенов.

Последующее развитие генетики поставило много вопросов о сущности генов в природе их действия и привело к формированию

биохимии и молекулярной биологии, конкретизируя основные положения классической генетики.

Было выяснено, что нуклеиновые кислоты высокомолекулярные биополимеры, которые состоят всего из четырех нуклеотидов, различным образом чередующихся в длинных цепях нуклеиновых кислот. Роль ДНК заключается в хранении и передаче генетической информации следующим поколениям. Различные типы РНК реализуют наследственную информацию в ходе индивидуального развития организмов. При этом все процессы, в которых участвуют нуклеиновые кислоты, – матричного типа. Хранение генетической информации заключается в соблюдении определенного порядка пар нуклеотидов в пределах каждого участка молекулы ДНК (гена).

В информационной РНК закодирован порядок расположения аминокислот в молекулах белка в виде определенного порядка нуклеотидов. Кодирование каждой аминокислоты осуществляется определенным сочетанием трех нуклеотидов (триплетом). Кодирование включения аминокислот в белковую молекулу есть генетический код. Оказалось, что одну аминокислоту могут кодировать от одного до шести триплетов, которые называют также кодонами.

В результате многочисленных исследований на молекулярном уровне были выяснены основные свойства генетического кода [7]:

- каждая аминокислота кодируется группой из трех нуклеотидов (триплетом или кодоном);
- непрерывность кода, т.е. один и тот же нуклеотид не может входить в состав различных триплетов (кодонов);

- вырожденность кода, т.е. одна аминокислота может кодироваться не только одним триплетом, многие аминокислоты – несколькими триплетами (до 6);
- считывание нуклеотидов (триплетов) начинается с одной фиксированной точки и идет в одном направлении подряд без выбора и знаков препинания до триплета, являющегося стоп-сигналом.

В настоящее время ген определяют как структурную единицу наследственной информации, далее неделимую в функциональном отношении. Его рассматривают как участок молекулы ДНК, кодирующий синтез одной макромолекулы или выполняющий какую-либо другую элементарную функцию. Комплекс генов, содержащихся в наборе хромосом одного организма, образует **геном** [1].

1.1.3. Рекомбинация

Рекомбинация сцепленных генетических факторов свойственна всем группам организмов, исследованным к настоящему времени.

Генетическая рекомбинация подразумевает несколько типов перераспределения наследственных факторов:

- Рекомбинация хромосомных и нехромосомных генов.
- Рекомбинация целых негомологичных хромосом.
- Рекомбинация участков хромосом, представленных непрерывными молекулами ДНК.

Каждое решение некоторой рассматриваемой задачи мы представляем как некоторую информацию, способную к обновлению посредством введения элементов иного решения, поэтому нас будет интересовать именно этот последний тип рекомбинации.

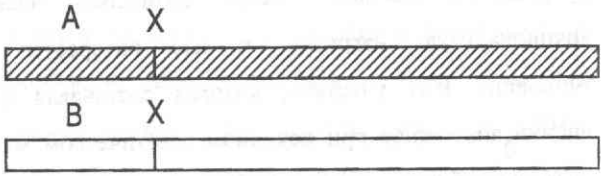
Хромосомы, представляющие собой отображения решений, должны быть гомологичны, так как являются взаимозаменяемыми альтернативами. Интерес представляет не замена одного сгенерированного решения на другое, что осуществимо простой оценкой исходных решений в соответствии с принятым критерием, а получение новых решений посредством обмена между ними информацией. Каждый участок хромосомы несет определенную функциональную нагрузку, и нас интересует создание такой комбинации этих участков, которая составляла бы наилучшее из решений, возможное при исходном генетическом материале. Поэтому целью рекомбинации будем считать накопление в конечном решении всех лучших функциональных признаков, какие имелись во всем наборе исходных решений.

Рекомбинация участков хромосом, представленных непрерывными молекулами ДНК, в свою очередь разделяется на несколько подтипов:

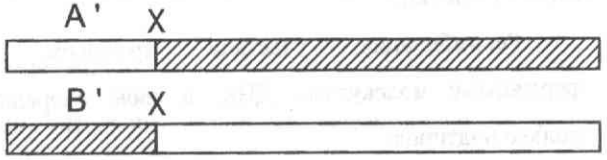
- регулярная (общая) рекомбинация, представляющая собой **кроссинговер**, т.е. обмен гомологичными участками в различных точках гомологичных хромосом, приводящий к появлению нового сочетания сцепленных генов;
- **сайт** – специфическая рекомбинация, т.е. обмен генофоров (генных носителей), часто различных по объему и составу генетической информации, на коротких специализированных участках;
- «иллегальная» (нереальная) рекомбинация, включающая негомологичные обмены, т.е. транслокации, инверсии, а также случаи неравного кроссинговера. Негомологичный

обмен в целом не представляет интереса. Однако, такие преобразования, как транслокации, инверсии могут быть полезны при генерации нового решения.

Современные представления о механизме кроссинговера восходят к представлениям школы Т. Моргана, согласно которым рекомбинация заключается в разрыве гомологичных хроматид с



а) - до кроссинговера



б) - после кроссинговера
X - точка разрыва (кроссинговера)

последующим соединением их в новом сочетании. Обычно схему кроссинговера представляют, как показано на рис.1.1, который

Рис 1.1 Схема кроссинговера

демонстрирует образование двух новых хромосом после обмена генетическим материалом.

На Рис. 1.1, а, б – родительские хромосомы до кроссинговера, а $A^{/}$, $B^{/}$ – хромосомы – потомки после кроссинговера.

Т. Морган предположил, что кроссинговер может происходить не только в одной, но и в двух и даже большем числе точек. На рис.1.2 представлена экспериментально установленная схема двойного кроссинговера между хромосомами f и w.

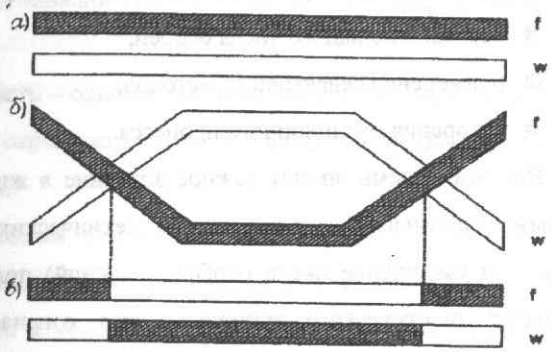


Рис 1.2. Схема двойного кроссинговера: а – до кроссинговера; б – во время кроссинговера; в – после кроссинговера.

1.1.4. Мутация

Расщепление при независимом наследовании и при кроссинговере определяет изменчивость организмов вследствие комбинаторики существующих генов (**аллелей**). Аллелями называют определенное химическое состояние гена. В результате генных мутаций появляются химические изменения гена, которые через РНК могут реализовывать изменение функционирования молекулы. Мутации – это возникновение качественно новых генов (аллелей), хромосом и наборов хромосом. Сочетание обоих типов изменчивости вызывает общую изменчивость генотипа.

Генетический материал обладает такими универсальными свойствами, как дискретность, непрерывность, линейность и относительная стабильность, выявляемыми в ходе генетического анализа [1-3, 6,7]. Повышение разрешающей способности генетического анализа возможно с помощью:

- изучения большого числа особей;
- применения селективных методов;
- ускорения мутационного процесса.

Все эти приемы имеют важное значение в живой природе и могут найти применение при решении технических задач. Легко заметить, что увеличение числа особей (решений) приводит к росту разнообразия генетического материала, что означает увеличение исходного набора контролируемых генами функций, а это, в свою очередь, позволяет провести более широкий отбор функций и тем самым улучшить конечное решение.

Применение селективных методов означает методический отбор генетического материала в соответствии с принятым критерием, имеющий целью повысить скорость получения конечного результата и его качество. Ускорение мутационного процесса ведет к получению все более разнообразного генетического материала.

Введем несколько определений [6 – 7].

Дискретность – существование гена, хромосомы, генома – проявляется в виде множества аллелей, соответствующих гену, множества генов, соответствующих группе сцепления (хромосоме), множества групп сцепления, соответствующих геному. Это значит, что мы можем выделять в исходном генетическом материале отдельные фрагменты, контролирующие те или иные функции.

Непрерывность – физическое единство групп сцепления – проявляется в существовании множества генов, обнаруживающих сцепление между собой. Это свойство находит выражение также в различных типах эффекта положения гена. Это свойство означает, что определенные комбинации генов совместно контролируют некоторую функцию.

Линейность – одномерность записи генетической информации – проявляется в определенной последовательности генов в пределах группы сцепления.

Относительная стабильность генетического материала – его способность к изменениям – проявляется в мутациях, что означает способность генов к воспроизведению и изменению с последующим воспроизведением измененных вариантов.

М у т а ц и я – генетическое изменение, приводящее к качественно новому проявлению основных свойств генетического материала: дискретности, непрерывности или линейности.

Таким образом, мутации представляют собой изменения генов, хромосом, геномов. Генные мутации на молекулярном уровне – результат различных повреждений в молекулах ДНК. В результате ошибок редупликации в ДНК появляются замены пар нуклеотидов. В пределах одного гена в одном поколении обычно встречаются единичные повреждения, редко – большие. Результатом этого могут быть гибель клетки, изменения характера индивидуального развития, изменение признаков и т.д [6]. В зависимости от характера изменений, возникающих в генетическом материале, мутации делят на:

- точковые;
- инсерции;

- хромосомные аберрации, или перестройки;
- полиплоидизацию.

Точковые мутации – это изменения пары нуклеотидов ДНК (или нуклеотида РНК). Этот класс мутаций подразделяется на:

- транзиции – замены пар нуклеотидов не приводящие к изменению ориентации;
- трансверсии – замены пар нуклеотидов, изменяющие ориентацию;
- вставку лишней пары нуклеотидов;
- выпадение пары нуклеотидов.

Точковые мутации могут проявляться как изменения функции отдельных генов и нескольких сцепленных генов одного оперона. Последствия возникновения точковых мутаций принято рассматривать в зависимости от изменения смысла генетической информации.

Инсерции – это внедрение молекул ДНК в ген. Подобное внедрение неальтернативной информации можно признать «нелегальным».

Хромосомные мутации, или хромосомные перестройки, принято подразделять на:

- делеции – выпадения участков генетического материала протяженностью от нескольких нуклеотидов до участков, включающих ряд генов;
- дефишенси, или концевые нехватки (также потеря информации). Нехватки обычно вызывают понижение жизнеспособности особи;
- инверсии – поворот участка гена или хромосомы на 180 градусов. В общем клетки при инверсиях сохраняют

нормальную жизнеспособность, кроме тех случаев, когда проявляется эффект положения гена, т.е. некоторые гены при изменении положения в хромосоме могут оказывать неблагоприятное действие на жизнеспособность;

- дупликации – повторение участка генетического материала (в зависимости от задачи операцию можно считать приемлемой либо нет). Дупликация является результатом потерь другими хромосомами участков при разрывах. Располагаться они могут как рядом друг с другом, так и в других местах данной хромосомы или даже в других хромосомах;
- транслокации – перенос части одной хромосомы на другую. Если обнаруживается недостаток одних и избыток других участков хромосом, то клетки оказываются нежизнеспособными;
- транспозиции – также перемещения генетического материала между хромосомами или в пределах одной хромосомы, однако этот термин обычно применяют к перемещениям небольших участков генетического материала, сопоставимых по протяженности с геном.
- Полиплоидизация – изменение числа хромосом, кратное гаплоидному набору (рост числа равноценных по качеству решений). С точки зрения формирования новой полезной наследственной информации, мутации во многом оказываются подобными кроссинговеру

В естественных системах роль мутаций заключается в том, что именно они генерируют новые функции, затем происходит дупликация,

закрепляющая обе функции, а после того начинается отдельная эволюция исходной и новой функции. Эта эволюция и показывает, что новая или возникшая в результате мутации функция обладает более высокими адаптационными качествами либо прежняя функция выполняет эту роль лучше.

В конце 1940-х – начале 1950-х годов А.Новик и Л.Сциллард обнаружили, что частота мутаций пропорциональна продолжительности клеточной генерации, а не числу делений; т.е. мутации могут происходить не только при удвоении генов. Еще в конце 1920-х годов предложено рассматривать мутабельность как адаптивный признак вида. Очевидно только, что мутабельность, как и все прочие признаки организма, является объектом естественного отбора [3].

Э.Г.Лей, рассматривая эту проблему теоретически, пришел к выводу, что мутаторные штаммы обладают лучшей приспособленностью, чем исходные дикого типа, но только при наличии постоянных условий. Этот факт показывает, что именно мутация при постоянстве целевой функции оказывается решающим фактором адаптации. Установлено, что в неизменных условиях биологические виды дают более высокую адаптивность. Существует гипотеза, что низшие организмы стремятся увеличить мутабельность, чтобы уравновесить положительные и отрицательные последствия мутаций, а высшие стремятся уменьшить [3, 4].

Интересную идею приводит Клизовский. «Законы природы не есть приказания, отданные властью, но наличность условий, при которых неизменно совершается одно и то же явление. Если изменяется условие, то изменяется и явление. Природа не знает нарушения своих законов. Но природа побеждается послушанием ее закона» [5].

В общем же эволюция стремится к усреднению (так как происходит все более однородное смешение разного по качеству с нашей точки зрения генетического материала). Поэтому в качестве одного из методов для получения наилучших результатов используется селекция.

1.1.5. Селекция

Селекция – наука о выведении новых сортов растений и пород животных и об улучшении уже существующих. Селекция представляет собой форму искусственного отбора, в отличие от естественного, где эволюция направляется факторами естественной среды. Селекция, как наука, создана Ч.Дарвином, который выделял три формы отбора [1]:

- **естественный отбор**, вызывающий изменения, связанные с приспособлением к новым условиям;
- **бессознательный отбор**, при котором «на племя» сохраняют лучшие экземпляры;
- **методический отбор**, при котором проводится целенаправленное изменение породы в сторону установленного идеала.

В учении об отборе Дарвин доказал, что главной движущей силой является отбор наилучших форм, требующий для успеха таких условий, как:

- правильный выбор исходного материала;
- правильная постановка цели;
- проведение селекции в достаточно широких масштабах и возможно более жесткая браковка материала;

■ отбор по одному основному признаку.

В современной селекции наиболее широко применяют **массовый и индивидуальный отбор**. Для создания генетически устойчивых пород, сортов, штаммов, обладающих желаемыми признаками, необходимо знать генетическую структуру исходных популяций. Массовый отбор, производимый по фенотипу, является менее эффективным, чем индивидуальный. При индивидуальном отборе популяцию делят на отдельные линии, причем для размножения выбирают носителей желаемых свойств. Иногда отбор ведут по боковым родственникам. Такой метод называют **сиб-селекцией** [1].

Инбридинг – близкородственное скрещивание – позволяет наиболее быстро выделить линию, несущую желаемые гены. В случае, когда эти гены не являются доминантными и их проявление может быть подавлено нежелательными доминантными, инбридинг является единственным средством их закрепления [7].

Рассмотрим схему селекции. Вначале создают несколько линий, несущих желаемые признаки. Затем из этих линий получают межлинейные гибриды первого поколения. Экземпляры, дающие наилучшие комбинации признаков, отбирают для последующего размножения. При этом межлинейные гибриды превосходят по качеству исходные экземпляры на 40-80%. При скрещивании различных пород, сортов, видов гибриды первого поколения часто проявляют признаки **гетерозиса**, или **гибридной мощности**, и превосходят лучшую из родительских форм. В последующих поколениях, даже уже во втором, признаки гетерозиса резко ослабляются и потом затухают. Закрепление гетерозиса в поколениях – одна из важнейших генетических задач селекции [1-3].

Подобную схему селекции можно использовать при решении задач оптимизации. В соответствии с принятым критерием даем оценку каждому из решений и отбираем для проведения над ними генетических операций решения, получившие наиболее высокую оценку. В этом генетическом материале может изначально отсутствовать необходимый для желаемого решения ген. В этой связи необходимо ввести в популяцию решение, предположительно несущее этот ген, либо использовать мутацию для создания необходимого признака.

1.1.6. Генная инженерия

Универсальность свойств генетического материала, установленная методами генетического анализа и молекулярной биологии, проявляется в том, что способы обмена генетической информацией, обнаруженные у одних объектов, применимы к другим объектам. Трансформация и трансдукция, открытые у бактерий, в 70-х годах были успешно осуществлены в культуре клеток высших организмов. Универсальность молекулярных носителей наследственной информации послужила основой для разработки методологии генной инженерии, базирующейся на создании рекомбинантов между молекулами ДНК из разных источников с последующим введением таких рекомбинантных молекул в живые клетки [4, 5].

Методы генной инженерии применяются для создания продуктов ряда биологически активных соединений, в том числе гормонов – инсулина, интерферона и других. Основа генной инженерии – это система для выделения отдельных фрагментов ДНК и устойчивого культивирования их в составе так называемых векторных

автономно редуплицирующихся молекул в организме хозяина. На выходе получается «музей» клонов бактерий, в каждом из которых размножается в составе векторной молекулы индивидуальный фрагмент, введенный в ДНК. Молекулы, объединяющие ДНК разных организмов, называют рекомбинантной ДНК, а генную инженерию определяют как совокупность методов для получения и операций над рекомбинантной ДНК. Таким образом комбинируются необходимые признаки, полученные из разных источников.

Согласно последним сообщениям из научных лабораторий мира, американские биологи близки к тому, чтобы сделать клетку «вечной». На основе исследований в Калифорнии и медицинском центре Техасского университета в августе 1997 г. найден способ клонирования гена «теломераза». Как известно, структура «теломераз», обволакивает все 46 хромосом человеческой клетки. При делении клеток структура постоянно уменьшается и, наконец, клетка перестает делиться и умирает. Старение клеток замедляется, если заставить их возобновить выделение «теломераза» и появляется возможность управлять механизмом старения.

Участки решений, удовлетворяющие нашим требованиям, могут быть выделены и распространены на всю популяцию решений. Это может иметь важное значение для таких задач, как упаковка, разбиение, сжатие, где сохранение удачно упакованного участка при последующих операциях даст возможность найти желаемое решение значительно быстрее.

1.2. Особенности механизма эволюционной адаптации

1.2.1. Роль популяции в эволюции

Основой эволюционных процессов в органическом мире служит популяция. **Популяция** – это многочисленная совокупность особей определенного вида, в течение длительного времени (большого числа поколений) населяющих определенный участок географического пространства, внутри которого осуществляется та или иная степень случайного свободного скрещивания – **панмиксии**. Вид подразделяется на несколько популяций, каждая из которых эволюционирует самостоятельно. В популяции нет абсолютно тождественных особей. Каждая особь является носителем уникального генотипа, который управляет формированием фенотипа. Элементарным носителем генотипа в организме каждой особи является клетка. Но поскольку жизнь клетки намного меньше жизни особи, то фактическим носителем генотипа является длинная череда поколений клеток. Рождение новых поколений клеток обеспечивается биосинтезом клеток. В процессе биосинтеза генотип особи не меняется [1-3].

Существование каждой особи ограничено некоторым временным интервалом, по истечении которого особь погибает. При этом генотип особи исключается из генофонда популяции. Но при жизни особь может передать **наследственную информацию** (НИ).

В настоящее время известно 3 механизма передачи НИ при рождении потомства [6,7]:

- бесполое размножение – основано на эквационном (равном) делении ядер клеток путем митоза, при котором происходит простое копирование НИ без какого-либо изменения;

- половое размножение – основано на редукционном и последующем эквационном делениях ядер клеток 2 родителей (отца и матери) путем мейозиса, в результате чего потомок получает копии НИ обоих родителей;
- промежуточная (между бесполым и половым) форма размножения - апомиксис у растений и партеногенез у животных.

Рассмотрим подробнее процесс биосинтеза наследственной информации и механизм передачи НИ при половом размножении, т.к. они представляют наибольшую ценность для генетических алгоритмов.

1.2.2. Биосинтез наследственной информации

Химический состав клетки стойко сохраняется во многих поколениях клеток и зависит от устойчивого сохранения основных особенностей биосинтеза клетки. Устойчивость биосинтеза, в свою очередь, определяется взаимодействием двух групп наиболее сложных органических веществ – белков и нуклеиновых кислот. Молекула белка состоит из сотен мономеров (остатков аминокислот) и представляет собой гетерополимер. Нуклеиновые кислоты также относятся к гетерополимерам и слагаются из остатков различных нуклеотидов.

В процессах наследственности и развития организмов ведущую роль играют две нуклеиновые кислоты- ДНК и РНК. ДНК располагается преимущественно в хромосомах. Биологическая роль ДНК состоит в передаче наследственной информации от материнских клеток к дочерним, а при половом размножении – от одного организма к другому.

Передача наследственной информации (НИ) от материнских клеток к дочерним происходит следующим образом. Молекула ДНК состоит из 2 абсолютно тождественных неразветвленных гетерополимеров. Нити гетерополимеров связаны между собой водородными мостиками, которые представляют собой слабые электрические связи. При определенных условиях 2 нити ДНК расходятся, после чего возле каждой половины начинают достраиваться недостающие нити гетерополимеров до образования двух дочерних молекул ДНК. Этот процесс называется авторепродукцией ДНК и в биосинтезе клетки играет значительную роль, являясь биосинтезом наследственной информации [7].

Свойства молекул ДНК зависят от количественного и качественного состава, а также от взаимного расположения входящих в их состав нуклеотидов. Взаимное расположение нуклеотидов играет важную роль для определения основных свойств нуклеиновых кислот.

Молекулы ДНК, соединяясь в длинные неразветвленные цепи, образуют хромонемы. Хромосомы представляют собой **хромонемы**, соединенные с молекулами РНК и молекулами белков. Молекулы ДНК располагаются главным образом в ядре, а в цитоплазму клетки наследственная информация передается благодаря молекулам информационной РНК. РНК несколько отличается от ДНК только по составу, но не по взаимному расположению нуклеотидов. Благодаря этому свойства дочерней РНК соответствуют свойствам материнской ДНК. РНК, в отличие от ДНК, состоят не из двух нитей гетерополимеров, а из одной (за исключением РНК ряда вирусов) и не способны к делению (авторепродукции).

Кроме молекул информационной РНК в клетке существуют молекулы растворимой РНК и транспортной РНК. Взаимодействие трех форм РНК обусловливает их способность контролировать синтез строго специфических белков-ферментов. Между тем, молекулы белков ферментов контролируют синтез аминокислот и азотистых оснований, т.е. строительных блоков для образования белков и нуклеиновых кислот. Таким образом, белки-ферменты и нуклеиновые кислоты в процессе биосинтеза активно взаимодействуют между собой и играют в нем значительную роль.

1.2.3. Механизм передачи наследственной информации

При половом размножении жизнь потомству дают два разнополых организма, НИ которых различна. Каждая клетка таких организмов несет двойной (диплоидный) набор хромосом, по одному от отца и матери. В образовании потомства принимают участие только 2 половые клетки – **гаметы**, поставляемые отцом и матерью.

Каждая гамета несет одиночный набор хромосом в отличие от обычных клеток организма. При слиянии двух родительских гамет с одиночным набором хромосом образуется первоклетка потомка – **з и г о т а**, имеющая двойной набор хромосом, из которой впоследствии развивается потомок. Диплоидные организмы, заключающие два одинаковых аллеломорфа определенного гена, называются **гомозиготными** по этому гену, а заключающие разные аллеломорфы определенного гена, – **гетерозиготными**.

Неизбежно встает вопрос: как образуются половые клетки - гаметы, которые содержат вдвое меньший, по сравнению с обычными клетками организма, набор хромосом?

«Уменьшение числа хромосом вдвое совершается путем особого типа деления ядра – редукционного деления, или мейозиса, кратко – мейоза. Мейоз всегда заключает два последовательных деления ядра – 1 и 2 деления мейоза», - говорится в [1]. Рассмотрим последовательность фаз этих двух делений [1 – 4].

- оболочка ядра растворяется; возникает «ахроматиновое веретено» – веретеновидная, состоящая из тонких волокон фигура, расположенная вдоль оси деления клетки. Некоторые нити ахроматинового веретена прикрепляются к хромосомам в строго определенном порядке.
- нити веретена «растаскивают» к полюсам клетки дочерние биваленты, каждая из которых обязательно содержит одну чистую хромосому (неперекомпонованную) от одного родителя и, в подавляющем большинстве случаев (т.е., когда происходит обмен участками хромосом во время диплонемы), перекомпонованную хромосому.
- достигнув полюсов ахроматинового веретена, хромосомы сбиваются в плотный клубок, внутри которого выделяется ядерный сок и ядрышко, а снаружи образуется ядерная оболочка.
- биваленты полностью раскручиваются.

Второе деление мейоза представляет собой обычное эквационое деление ядра, в результате которого из 2 клеток (каждая из которых содержит бивалент, состоящий из 2 хромосом) образуются 4 клетки – гаметы (которые имеют одиночный набор хромосом). Эквационное деление ядра (образование от одного материнского ядра двух дочерних) состоит из ряда последовательных фаз: интеркинез,

профаза, метафаза, анафаза и телофаза [7]. Основные операции при этом следующие:

- фаза между двумя делениями клетки; хромонемы (длинные неразветвленные цепочки молекул ДНК) находятся в свободном состоянии и активно совершают свойственный им обмен веществ;
- непосредственно перед профазой происходит авторепродукция ДНК, т.е. из одной материнской хромонемы, несущей НИ, расщепляются 2 дочерние хромонемы (происходит удвоение числа хромонем);
- дочерние хромонемы, полученные от одной материнской, обвивают друг друга и начинают путем скручивания утолщаться и укорачиваться;
- средняя и парные хромонемы путем более сильного скручивания образуют компактные парные хромосомы;
- в конце профазы оболочка ядра растворяется, и хромосомы выходят в химически активную среду цитоплазмы;
- парные хромосомы передвигаются в экваториальную область клетки.
- образуется «ахроматиновое веретено»;
- нити веретена растягивают пару хромосом в противоположные стороны;
- на месте полюсов образуются ядра, т.е. всего 2 ядра на каждую клетку;
- после восстановления благоприятной для хромонем внешней среды хромосомы сильно разрыхляются,

соединяются между собой перемычками и постепенно образуют тонкую сеть хромонем.

Таким образом, в результате мейозиса из одной клетки с двойным набором хромосом образуются 4 половых клетки – гаметы с одиночным набором хромосом. Хромосомы, входящие в состав гамет, образованы путем обмена участками (генами) между гомологичными хромосомами родителей. Одна из этих четырех гамет затем участвует в образовании первоклетки потомка – зиготы. Наследственная информация зиготы образуется путем смешения НИ родителей.

Каждая особь в течение жизни подвергается воздействиям внешней среды. В некоторых случаях эти воздействия могут привести к перестройкам молекул ДНК, переносящих НИ. Изменение первоначальной последовательности генов в молекулах ДНК приводит к изменению свойств этой молекулы, а следовательно и наследственной информации.

Частота возникновения отдельных мутаций всегда относительно очень низка – 10^{-4} – 10^{-6} на поколение. Но в связи с большим числом генов общая частота возникающих мутаций у живых организмов высокая. Она колеблется в пределах от единиц (у бактерий, одноклеточных водорослей) до нескольких десятков процентов (20-25% у дрозофилы) гамет на одно поколение. Мутационный процесс осуществляет неопределенную изменчивость. Эта ненаправленность мутационного процесса исключает возможность его направляющего влияния на ход эволюционных изменений.

Внешние воздействия вызывают мутации отдельных индивидов. Закрепление генотипа мутанта в популяции происходит при скрещивании этого мутанта с другой особью и образовании потомства. В малочисленной популяции в результате свободного скрещивания генотип мутанта быстро размножается и вызывает резкие изменения в структуре генофонда популяции. И наоборот, в достаточно многочисленной популяции генотип мутанта вскоре «растворяется» среди других генотипов. Таким образом, мутационный процесс приводит к изменению (реорганизации) наследственной информации, а колебание численности управляет «давлением» возникающих мутаций на генофонд популяции. Эти два фактора являются своего рода «поставщиками» материала для эволюции.

Если в пределах популяции свободное скрещивание особей нарушается на длительное время, то в процесс эволюции включается фактор – изоляция. **Изоляция**, нарушая панмиксию (всегда ведущую к сглаживанию различий путем скрещивания), закрепляет возникшие случайно (в результате работы мутационного процесса и колебания численности популяции) различия в наборах генотипов в разных частях популяции. В результате действия изоляции из одной исходной популяции образуются две и более генотипически отличающихся друг от друга популяций.

Несомненно, важнейшим эволюционным фактором является естественный отбор. **Естественный отбор** – процесс, направленный к повышению (или понижению) вероятности оставления потомства одной формой организмов по сравнению с другими. Отбор прежде всего действует в пределах каждой популяции, отбирая (или отметая) те или иные входящие в ее состав генотипы. При относительной стабильности

внешних условий преобладающие генотипы все время будут сохранять свое доминирующее положение. Напротив, все уклонения от этой группы будут уничтожаться. Такая форма отбора названа центростремительным или стабилизирующим отбором. Однако при изменении условий существования может произойти отбор, ведущий к замене одних количественно преобладающих генотипов другими. Эта форма отбора названа движущей, или ведущей. Естественный отбор – единственный направляющий эволюцию элементарный фактор. Его действие всегда направляется складывающимися условиями существования [1].

Рассмотренные выше факторы по степени влияния на эволюцию можно упорядочить по убыванию влияния:

- естественный отбор;
- изоляция;
- колебание численности популяции;
- мутационные процессы.

Пусковой механизм эволюции функционирует в результате совместного действия эволюционных факторов в пределах популяции. В результате действия эволюционных сил в каждой популяции тысячекратно возникали элементарные эволюционные изменения. Со временем некоторые из них суммируются и ведут к возникновению новых приспособлений, что и лежит в основе видообразования.

1.3. Направленная молекулярная эволюция

Направленная молекулярная эволюция (НМЭ) подобна искусственному отбору. Если надо создать молекулу, обладающую каким-либо химическим свойством, следует выбирать из большой

популяции молекул те, которые в наилучшей степени выражают это свойство, и произвести из них дочерние, в разной степени похожие на родителей. Этот процесс отбора и дупликации повторяется до тех пор, пока не будет достигнут нужный результат. Успех НМЭ определяется большим числом участников.

Согласно теории Дарвина, эволюция осуществляется во взаимодействии трех повторяющихся вновь процессов: отбора, амплификации, мутации [1-7].

Амплификация- процесс производства потомков или более точно, копирование унаследованных членов, действует в природе совместно с отбором.

Впервые в лаборатории процесс эволюции был воспроизведен в 1960 г. Спигельманом из Иллинойского университета. Критерий отбора подобен библейскому: «плодитесь и размножайтесь».

НМЭ похожа на бег с препятствиями. Макромолекулы как бы соревнуются между собой. Последовательное повторение циклов селекции и амплификации создает популяцию молекул с желаемыми свойствами.

В НМЭ используется три стратегии создания стартовой популяции:

Стратегия «одеяла» - в данном случае исходная популяция включает все возможности популяции в некоторых пределах. Когда эти пределы трудно определимы, применяется другая стратегия.

Стратегия «дробовика» - подразумевает случайный набор возможностей.

Стратегия «фокусировки» - выполняется на основе какой-либо одной молекулы создается популяция ее мутантов.

Эволюция лишена предвидения; в любой момент времени члены в популяции отображают свойства, дающие преимущества предшествующим поколениям. Стратегия повторяющейся рандомизации может заставить молекулы эволюционизировать в направлении улучшения функциональных характеристик.

Обычно рассматривают микроэволюцию и макроэволюцию. В процессе микроэволюции эволюция протекает внутри индивидуумов, имеющих определенную степень внутренней и внешней связности. Эти процессы протекают в популяциях и на молекулярном уровне. Приблизительный временной масштаб этих процессов $t \approx 4N$, где N – размер популяции. В процессе макроэволюции эволюционные преобразования происходят в родах, семействах, отрядах. Причем завершенные процессы микроэволюции рассматриваются как элементарные события макроэволюции [3, 4, 6, 7].

1.4. Выводы

На основе приведенного исторического экскурса и краткого обзора генетики, отметим, что полезными для решения различных технических задач могут быть также генетические «операции», такие как:

- кроссинговер;
- мутация;
- инверсия;
- транслокация;
- транспозиция;
- селекция (инбридинг и гибридизация);
- генная инженерия.

Роль кроссинговера – основная, она состоит в создании из имеющегося генетического материала желаемой комбинации функции в одном решении. Роль мутации состоит в том, чтобы вывести качественно не улучшающееся посредством кроссинговера решение на новый уровень (в том числе генерацией изначально отсутствующих функций). Роль селекции – в ускорении получения качественного решения.

В последнее время проявляется тенденция использования естественных аналогов при создании моделей, технологий, методик, алгоритмов для решения тех или иных задач, стоящих перед человечеством. В большинстве случаев использование естественных аналогов дает положительные результаты. Например, открытие массового воздушного транспорта связано с появлением самолета, крыло которого является аналогом крыла птицы. Как правило, это объясняется тем, что аналог, взятый из природы, совершенствовался в течение многих лет эволюции и имеет на данный момент самую совершенную в своем роде структуру.

Кратко рассмотренный выше механизм эволюционирования предлагается использовать в качестве механизма случайно-направленного комбинаторного перебора при решении задач оптимизации, принятия решений, искусственного интеллекта, проектирования и производства.

1.5. Контрольные вопросы

1. Законы Менделя. Гипотезы Менделя.
2. Опишите процесс деления клеток.
3. Ч.Дарвин и его учение об эволюции.

4. Основные идеи наследования.
5. Поясните суть мутации.
6. Законы Харди и Пирса.
7. Перечислите основные факторы, меняющие генетический состав популяции.
8. Молекулярная генетика.
9. Алгоритм синтеза белка в живой клетке.
10. Генетические операторы кроссинговера.
11. Отбор и его влияние на эволюцию.
12. Основные положения теории Моргана.
13. Механизм передачи наследственной информации.
14. Биосинтез наследственной информации.
15. Идеи направленной молекулярной эволюции.

Научное мировоззрение, проникнутое естествознанием и математикой, есть величайшая сила не только настоящего, но и будущего.

В.И. Вернадский

2. ОСНОВНЫЕ ПОНЯТИЯ ГЕНЕТИЧЕСКИХ АЛГОРИТМОВ*

2.1. Определение ГА

Исторически считается, что основой использования механизмов живой природы является направление «эволюционные вычисления», состоящее из трех самостоятельных разделов: генетические алгоритмы, эволюционные стратегии (ЭС) (в некоторых работах говорят об эволюционном моделировании (ЭМ)) и эволюционное программирование (ЭП) [8-61].

В своих работах Букатова [17,18,23] сформулировала новое направление «эвоинформатика», которое предшествовало появлению генетических алгоритмов. Эвоинформатика определена как совокупность алгоритмических, программных и аппаратных средств, основанных на имитации механизмов естественной эволюции для синтеза структур обработки информации [17]. Букатова предложила архитектуру эволюционного поиска, состоящего из следующих компонент: моделирование цели, имитация основных законов

*Материалы этой главы подготовлены на основе классической монографии Гольдберга [9].

дарвинизма, соревнование, отбор. Основу алгоритмов эволюционного структурного поиска здесь составляют самообучающиеся и адаптивные алгоритмы. Самоорганизующиеся вычисления основаны на распараллеливании эволюционных вычислений аналогично естественной эволюции, происходящей на основе механизмов синергетики. Такая эволюционно-вычислительная технология согласно [17,18] включает: генерацию эволюционных систем, эволюционный синтез заданных моделей, поэтапное управление и интерпретацию моделей из базы знаний.

Основой для возникновения генетических алгоритмов считается модель биологической эволюции и методы случайного поиска [20, 32]. Один из известных специалистов в мире в области случайного поиска и стохастической оптимизации Растригин пишет [32], что случайный поиск (СП) возник как реализация простейшей модели эволюции, когда случайные мутации моделировались случайными шагами оптимального решения, а отбор «устранением» неудачных вариантов. Например, для прикладных оптимизационных задач

$$K(x) \rightarrow extr,$$

где K – функционал, x – искомое решение, extr – экстремум (принимает в зависимости от условий задачи минимальное или максимальное значение). Тогда, например, для максимизации

$$K(x) \rightarrow max \Rightarrow x^*,$$

где x* – наилучшее решение. Это выражение реализуется с учетом ограничений и граничных условий. Эволюционный поиск [32] – это последовательное преобразование одного конечного множества промежуточных решений в другое. Само преобразование можно назвать алгоритмом поиска, или алгоритмом эволюции. Растригин выделяет три особенности алгоритма эволюции:

- каждое новая популяция состоит только из «жизнеспособных» хромосом;
- каждая новая популяция «лучше» (в смысле целевой функции) предыдущей;
- в процессе эволюции последующая популяция зависит только от предыдущей [32].

Согласно [32] природа, проводя эволюцию, как бы решает оптимизационную задачу на основе случайного поиска. Выделяются три основные бионические эвристики случайного поиска:

- клеточный СП,
- моделирование целесообразного поведения особей,
- моделирование передачи наследуемой биологической информации.

Генетические алгоритмы (ГА) есть поисковые алгоритмы, основанные на механизмах натуральной селекции и генетики. Они реализуют «выживание сильнейших» среди рассмотренных структур, формируя и изменяя поисковый алгоритм на основе моделирования эволюции поиска. В каждой генерации новое множество искусственных последовательностей создается с использованием старых и добавлением новых частей с «хорошими свойствами». ГА – это не

просто случайный поиск. Они эффективно используют информацию, накопленную в процессе эволюции [8 -18]. Цель ГА двоякая [8, 9]:

- абстрактно и формально объяснить адаптацию процессов в естественных системах;
- спроектировать искусственные программные системы, которые содержат механизмы естественных систем.

Центральная тема поиска в ГА – поиск баланса между эффективностью и качеством для «выживания», т.е. получения оптимальных решений в различных условиях. Можно условно выделить три типа поисковых методов:

I. вычислительные – подразделяются на два основных класса:
   - A. непрямые методы – они находят локальные экстремумы, решая обычно множество линейных или нелинейных уравнений. Данные методы находят экстремум функции, анализируя ограниченное пространство точек во всех направлениях;
   - B. прямые методы – осуществляют поиск оптимума путем градиентных алгоритмов. Уже для двух локальных оптимумов это большая проблема - найти лучшее решение, а для многоэкстремальных схем применение этих методов затруднительно. Сейчас применяются методы линейного, нелинейного и динамического программирования;

II. невычислительные;

III. случайные:
   - A. чисто случайный поиск;
   - B. моделирование отжига;

С. генетические алгоритмы.

ГА согласно [8, 9, 10] отличаются от других оптимизационных и поисковых процедур следующим:

- работают в основном не с параметрами, а с закодированным множеством параметров;
- осуществляют поиск из популяции точек, а не из единственной точки;
- используют целевую функцию, а не ее различные приращения для оценки информации;
- используют не детерминированные, а вероятностные правила.

ГА берет множество натуральных параметров оптимизационной проблемы и кодирует их как последовательность конечной длины в некотором конечном алфавите.

ГА работает до тех пор, пока не будет выполнено заданное количество генераций или на некоторой генерации будет получено решение заданного качества или возникнет преждевременная сходимость, когда найден локальный оптимум и ГА не может найти из него выход.

В отличие от других методов оптимизации ГА, как правило, анализируют различные области пространства решений одновременно и более приспособлены к нахождению новых областей с лучшими значениями целевой функции за счет объединения квазиоптимальных решений из разных популяций.

Дадим сравнение некоторых терминов естественной генетики и ГА [8,9]:

- хромосома – решение, стринг, последовательность, индивидуальность, «родитель», «потомок», «ребенок» и т.п.;
- популяция – набор решений (хромосом);
- локус – местоположение гена в хромосоме;
- поколение – генерация ГА (процесс реализации одной итерации алгоритма);
- ген – элемент;
- аллель – величина (значение) элемента;
- фенотип – структура;
- эпистасис – множество параметров, альтернативные решения.

Эволюция популяции, согласно [19], – это чередование поколений, в которых хромосомы изменяют свои признаки, чтобы каждая новая популяция наилучшим способом приспосабливалась к внешней среде.

В естественных системах общая генетическая упаковка называется генотипом. В натуральных системах организм формируется посредством связи генетической упаковки с окружающей средой и называется фенотипом.

В естественной терминологии хромосомы состоят из генов, которые могут иметь числовые значения, называемые «аллели». Отметим, что в литературе встречаются и те, и другие термины. Мы также будем использовать различные определения одних и тех же терминов, чтобы ориентироваться в существующих публикациях по ГА. Другие понятия и определения будем вводить по мере необходимости.

2.2. Простой ГА

Эволюционный процесс представляется как способность «лучших» хромосом оказывать большее влияние на состав новой популяции на основе длительного выживания и более многочисленного потомства [32].

Основные этапы процесса эволюции, на основе которого создаются нужные схемы генетического поиска, согласно Холланду следующие [8].

1°. Конструируется начальная популяция. Введена точка отчета поколений t = 0. Вычисляется приспособленность хромосом популяции, а затем средняя приспособленность популяции.

2°. Установить t= t+1. Выбор двух родителей (хромосом) для кроссинговера. Он выполняется случайным образом пропорционально приспособляемости родителей.

3°. Формирование генотипа потомка. С заданной вероятностью произвести над генотипами выбранных хромосом кроссинговер. Выбрать с вероятностью 0,5 один из потомков A(t) и сохранить его как члена новой популяции. Последовательно применить к A(t) оператор инверсии, а затем мутации с заданными вероятностями. Полученный генотип потомка сохранить как A′(t).

4°. Отбор хромосомы случайным образом для исключения ее из популяции. Обновление текущей популяции заменой отобранной хромосомы на A′(t).

5°. Определение приспособленности (целевой функции) A′(t) и пересчет средней приспособленности популяции.

6°. Если $t = t_{заданному}$, то переход к 7°, если нет, то переход к 2°.

7°. Конец работы.

Мы привели так называемый репродуктивный план Холланда [8] в несколько упрощенном виде. Заметим, что в технических задачах оптимизации часто вместо понятия «приспособленность» используют целевую функцию (ЦФ).

Простой генетический алгоритм был впервые описан Гольдбергом на основе работ Холланда [8, 9]. Механизм простого ГА (ПГА) несложен. Он копирует последовательности и переставляет их части. Предварительно ПГА случайно генерирует популяцию последовательностей – стрингов (хромосом). Затем ПГА применяет множество простых операций к начальной популяции и генерирует новые популяции.

ПГА состоит из 3 операторов:

- репродукция;
- кроссинговер;
- мутация.

Репродукция – процесс, в котором хромосомы копируются согласно их ЦФ. Биологи называют эту функцию – fitness. Копирование хромосом с «лучшим» значением ЦФ имеет большую вероятность их попадания в следующую генерацию. Согласно учению Дарвина, оператор репродукции (ОР), конечно, является искусственной версией натуральной селекции «выживания сильнейших». ОР представляется в алгоритмической форме различными способами. Самый простой –

создать колесо рулетки, в котором каждая хромосома имеет поле, пропорциональное его ЦФ. Например, мы хотим максимизировать функцию $f(x)=x^2$ на целочисленном интервале [0, 31]. Традиционными методами мы будем менять x, пока не получим max f(x) [9].

Для объяснения и реализации ПГА построим следующую табл. [2.1]:

Таблица 2.1

Номер хромосом	Хромосома (двоичный код)	Десятич-ный код	ЦФ(fitness)	Значение ЦФ, в процентах
1	0 1 1 0 1	13	169	14.4
2	1 1 0 0 0	24	576	49.2
3	0 1 0 0 0	8	64	5.5
4	1 0 0 1 1	19	361	30.9

Хромосомы столбца 2 табл.2.1 сгенерированы случайным образом. Соответственно в табл. 2.1 суммарное значение ЦФ =1170.

Значение ЦФ для каждой хромосомы определим как возведенное в квадрат значение двоичного числа, которое кодирует хромосому в столбце 2 табл. 2.1. Для селекции хромосом используется СП на основе колеса рулетки.

На рис. 2.1 поля колеса рулетки соответствуют значению ЦФ в процентах. В одной генерации колесо рулетки вращается, и после останова ее указатель определяет хромосому, выбранную для следующего оператора. Очевидно, не всегда хромосома с большой ЦФ в результате ОР будет выбрана для следующего оператора. Будем считать для упрощения, что 14,4=14; 49,2 = 49; 5,5 = 6; 30,9 = 31.

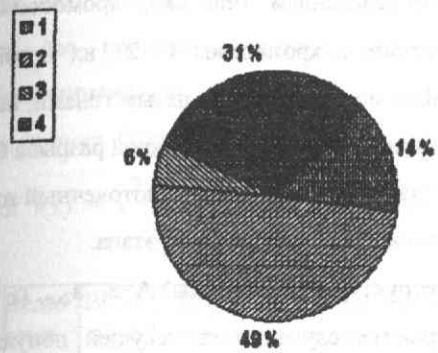


Рис.2.1. Колесо рулетки

ОР выбирает хромосомы для оператора кроссинговера. После выполнения ОР оператор кроссинговера (ОК) может выполниться в 3 шага. На первом шаге члены нового репродуцированного множества хромосом сначала выбираются. Далее каждая пара хромосом (стрингов) пересекается по следующему правилу: целая позиция k вдоль стринга выбирается случайно между L и длиной хромосомы, меньшей единицы, т.е. в интервале (1,L-1). Длина L хромосомы – это число значащих цифр в ее двоичном коде. Длина всех хромосом во втором столбце табл.2.1 L=5. Две новых хромосомы создаются, меняя все характеристики между позициями (k+1) и L соответственно. Например, рассмотрим хромосомы 1 и 2 из начальной популяции (табл. 2.1). Пусть k=4. Тогда получим

1 : 0 1 1 0 | 1 До применения оператора кроссинговера
2 : 1 1 0 0 | 0.

1′ : 0 1 1 0 | 0 После применения оператора кроссинговера
2′ : 1 1 0 0 | 1.

Следуя определениям генетики, хромосомы 1 и 2 часто называют родителями, а хромосомы 1′, 2′- их потомками. Число k, выбранное случайно между первым и пятым генами, называется точкой ОК или разделяющим знаком ОК, или точкой разрыва ОК.

Итак, согласно Холланду [8], одноточечный или простой ОК в технических системах выполняется в три этапа:

1°. Две структуры (хромосомы) $A=a_1, a_2,.., a_L$ и $B=a'_1, a'_2,.., a'_L$ выбираются случайно из текущей популяции после ОР. Заметим, что этап 1 в некоторых случаях может выполняться и без ОР, т.е. непосредственно из начальной популяции.

2°. Число k выбирается из {1,2,...,L-1} также случайно. Здесь L- длина хромосомы, k- точка ОК.

3°. Две новые хромосомы формируется из A и B, заменяя множество элементов. В результате получим:

$$A' = a_1, a_2, ..., a_K, a'_{K+1}, ..., a'_L,$$

$$B' = a'_1, a'_2, ..., a'_K, a_{K+1}, ..., a_L.$$

После применения ОК имеем две старые хромосомы и всегда получаем две новые хромосомы. Схематически простой ОК показывает преобразование двух хромосом и частичный обмен информации, используя точку пересечения, выбранную случайно. Например:

Перед ОК →	После ОК
хромосома 1 : 1 1 1| 1 1	1 1 1 0 0 новая хромосома 1′
хромосома 2 : 0 0 0| 0 0	0 0 0 1 1 новая хромосома 2′.

Механизм ОР и ОК удивительно прост. Он включает случайную генерацию чисел, копирование хромосом и частичный

обмен частей хромосом. Гольдберг отмечает: «Сначала это кажется удивительным. Как два простых оператора могут образовывать мощный поисковый механизм?» [9]

Рассмотрим пример Гольдберга работы ПГА для вычисления максимума функции $f(x) = x^2$ [9]. Данные для удобства сведены в табл.2.2. [9].

Номер хромосомы	Начальная популяция (генерируется случайно)	Величина x	Величина f(x)	Нормализованная величина $f_i(x)/sum[f(x)]$	Ожидаемое кол-во копий хромосомы в следующем поколении	Реальное кол-во копий хромосомы в следующем поколении
1	0 1 1 0 1	13	169	0.14	0.58	1
2	1 1 0 0 0	24	576	0.49	1.97	2
3	0 1 0 0 0	8	64	0.06	0.22	0
4	1 0 0 1 1	19	361	0.31	1.23	1
Суммарная ЦФ (sumf(x))			1170	1.00	4.00	4
Среднее значение ЦФ $\overline{f}(x)$			293	0.25	1.00	1
Max значение ЦФ			576	0.49	1.97	2

Таблица 2.2

Генерация ГА начинается с репродукции. Мы выбираем хромосомы для следующей генерации, вращая колесо рулетки 4 раза, что соответствуют мощности начальной популяции. Величину отношения $f_i(x) / sumf(x)$ называют вероятностью выбора копий (хромосом) при ОР и обозначают

$$P_i(OP) = \frac{f_i(x)}{sum\, f(x)}, \qquad (2.1)$$

где $f_i(x)$- значение ЦФ i-й хромосомы в популяции, sum f(x) суммарное значение ЦФ всех хромосом в популяции. Величину (2.1) также называют нормализованной величиной. Ожидаемое число копий i-й хромосомы после ОР определяют так:

$$N = P_i(OP) \cdot n, \qquad (2.2)$$

где n- число анализируемых хромосом.

Число копий хромосомы, переходящее в следующее поколение, также иногда определяют на основе выражения

$$A_i = \frac{f_i(x)}{\bar{f}(x)}. \qquad (2.3)$$

Согласно выражению (2.2), первая хромосома из табл. 2.2 получит 0,14 · 4=0,56 копий, вторая- 0,49 · 4=1,96 копий, третья- 0,06 · 4 = 0,24 и четвертая 0,31 · 4=1,24. Значения, полученные согласно (2.3), приведены в 6 столбце табл. 2.2. Смоделировать этот процесс можно, например, используя подброс монеты. В результате хромосомы 1 и 4 получают 1 копию, хромосома 2 – 2 копии, и хромосома 3 не получает копий. Сравнивая этот результат с ожидаемым числом копий, получаем то, что ожидали: «лучшие» хромосомы дают большее число копий, «средние» остаются и плохие «умирают» после ОР. Для рассмотренного примера Гольдберга построим табл.2.3.

ТАБЛИЦА 2.3

Номер хромосомы	Популяция после репродукции	Пары (случайно выбраны)	Точка ОК	Новая популяция хромосом	Значения x	Значение f(x)
1	0 1 1 0 | 1	1-2	4	0 1 1 0 0	12	144
2	1 1 0 0 | 0	1-2	4	1 1 0 0 1	25	625
3	1 1 | 0 0 0	3-4	2	1 1 0 1 1	27	729
4	1 0 | 0 1 1	3-4	2	1 0 0 0 0	16	256
Sum f(x)	1754					
$\bar{f}(x)$	439					
Maxf(x)	729					

Применяя к популяции после ОР (столбец 2 табл.2.3) оператор кроссинговера, получим новую популяцию хромосом (5 столбец таблицы 2.3). В принципе можно применять ОК заданное число раз. Например, если в столбце 5 табл.2.3 выбрать хромосомы 1 и 3 и между ними провести ОК, то можно получить

1: 01 | 100 (ЦФ-12),
3: 11 | 011 (ЦФ-27),
1′: 01 011 (ЦФ-11),
3′: 11 100 (ЦФ-28).

После проведения одной генерации ПГА улучшились все показатели: среднее и max значение ЦФ.

Далее, согласно схеме классического ПГА, выполняется оператор мутации. Из раздела 1 следует, что существует некоторое количество видов мутаций, и этот оператор играет существенную роль в естественной генетике и эволюции.

Оператор мутации (ОМ) играет вторичную роль в ПГА. Обычно выбирают одну мутацию на 1000 бит. Считают, что мутация – вторичный механизм в ГА.

Согласно Холланду, ОМ состоит из двух этапов [8]:

1° В хромосоме $A=a_1,a_2,a_3,...,a_{L-2},a_{L-1},a_L$ определяются случайным образом две позиции, например позиция 2 и позиция L-1.

2° Гены, соответствующие выбранным позициям, меняются местами, и формируется новая хромосома. Например, после получаем

ОМ-$A'= a_1,a_{L-1},a_3,...,a_{L-2},a_2,a_L$.

Заметим, что в ПГА ОК - бинарная операция, а ОМ - унарная операция. Кроме того, ОК и ОМ соответствуют перестановкам элементов внутри заданного множества. Очевидно, что при небольшой длине хромосомы (L- порядка 10-20) можно выполнить полный перебор за приемлемое время и найти наилучшие решения. При увеличении L до 50-200 и выше полный перебор произвести затруднительно и необходимы другие механизмы поиска. Здесь как раз и приходит на помощь направленно- случайный поиск, который можно реализовать на основе ПГА. Применим, например ОМ, к хромосоме 3 с ЦФ=27 (столбец 5 табл.2.3). Выберем позиции 3 и 4 и после ОМ получим:

3: 1 1 0 1 1 ⇒3′:11101

Как видно, хромосома 3′ имеет ЦФ=29. Снова применим к хромосоме 3′ ОМ. Тогда при случайном выборе позиций можно выбрать гены 4 и 5 и после ОМ получим:

3′ 1 11 0 1⇒3″:11110.

Хромосома 3″ имеет ЦФ=30, т.е. мы нашли квазиоптимальное решение задачи нахождения максимального значения функции $f(x)=x^2$ на интервале [0, 31].

Выделяют два основных механизма воспроизводства хромосом:

- потомки являются точными копиями родителей (неполовое воспроизводство без мутации);
- потомки имеют большие отличия от родителей.

В эволюционном программировании и ГА в основном используют комбинации этих механизмов. Причем согласно [32], в ЭП и ЭС в основном используют механизмы родительского воспроизводства с мутацией, а в ГА – механизм родительского воспроизводства с рекомбинацией и мутацией.

Примечания.

1. Начальная популяция может выбираться любым образом, например, подбросом монеты (орел = 1, решка = 0).
2. Репродукция выполняется через моделирование движения колеса рулетки.
3. ОК в задачах вычислительного характера обычно выполняется через двоичное декодирование двух положений монеты. В комбинаторно-логических задачах применяют другие типы ОК и другие технологии его выполнения.
4. Вероятность ОК допускается равной P(ОК) = 1.0 и меньше,
5. Вероятность ОМ допускается равной P(ОМ) = 0.001 и больше.

Как видно из табл.2.3, лучший стринг в первой генерации (11000) получает 2 копии, потому что у него высокая ЦФ.

Приведем верхний уровень генетического алгоритма, описанный Девисом [10].

1° Инициализация популяции хромосом.

2° Оценка каждой хромосомы в популяции.

3° Создание новых хромосом посредством спаривания текущих хромосом; применение мутации и рекомбинации.

4° Устранение хромосом из популяции, чтобы освободить место для новых хромосом.

5° Оценка новых хромосом и вставка их в популяцию.

6° Если время исчерпано, то останов и возврат к наилучшей хромосоме, если нет, то переход к 3°.

Сравнивая описание ГА Гольдберга, Холланда и Девиса, видно, что реализована одна основная идея моделирования эволюции с некоторыми модификациями. Заметим однако, что эти изменения могут существенно влиять на окончательное качество решения.

Генетический оператор инверсии (ОИ) в ГА Холланда [8] состоит из следующих шагов:

1. Структура (хромосома) $B=b_1,b_2,\ldots,b_L$ селектируется (выбирается) случайным образом из текущей популяции.
2. Два числа y_1' и y_2' выбираются случайным образом из множества $Y=\{0, 1, 2, \ldots, L+1\}$, причем считается, что $y_1=\min\{y_1', y_2'\}$ и $y_2=\max\{y_1', y_2'\}$.
3. Новая структура (хромосома) формируется из B, инвертируя сегмент, который лежит справа от позиции y_1 и слева от позиции y_2 в хромосоме B. Тогда, после применения ОИ получаем B':

$$B'=\{b_1, \ldots, b_{y1}, b_{y2-1}, b_{y2-2}, \ldots, b_{y1+1}, b_{y2}, \ldots, b\}.$$

Например, пусть до применения ОИ B={1, 2, 3, 4, 5, 6}, $y_1'=6$, $y_2'=2$, тогда $y_1=2$, $y_2=6$ и после применения ОИ к хромосоме B получим $B'=\{1, 2, 5, 4, 3, 6\}$.

Существует понятие «элитизм». Он определен как операция, когда хромосома с наилучшей ЦФ сразу копируется в новую популяцию [9,10,32].

Понятие «схема» (схемата), согласно Холланду, есть шаблон, описывающий подмножество стрингов, имеющих подобные значения в некоторых позициях стринга. Для этого вводится новый алфавит {0,1,*}, где *- означает: не имеет значения 1 или 0 [8].

Например, схема (*0000) соответствует двум стрингам {10000 и 00000}. Другой пример: схема (*111*) описывает подмножество с 4 членами {01110, 11110, 01111, 11111}. Схема (0*1**), будет иметь 8 стрингов длины 5, т.е. в общем случае в стринге L=5 мы можем иметь 3^5= 243 шаблона, но для стринга L=5 можно иметь только 2^5 = 32 различных альтернативных стринга. Это следует из того, что схема (**) имеет в общем 3^2= 9 стрингов, т.е. (**, *1, *0, 1*, 0*, 00, 01, 10, 11). Из этих 9 стрингов мы берем 4: 00, 01, 10, 11 [9].

В ПГА основная идея – это объединить две хромосомы с ЦФ выше средней, а именно найти подстринги типа 11*** и ***111. (Здесь *** могут соответствовать любым значениям). Тогда, применяя ОК к данным подстрингам, можно получить хромосому 11111 с наилучшим значением ЦФ.

Для вычисления числа схем или их границы в популяции требуются точные значения о каждом стринге в популяции. Поэтому

сначала вычисляют число схем, содержащихся в индивидуальном стринге, а затем находят верхнюю границу общего числа схем в популяции.

Рассмотрим, например, хромосому длины L=6, 111111. Она имеет 2 шаблона, потому что каждая позиция может быть 1 или *. В общем случае считают, что хромосома содержит 2 шаблона. Тогда популяция размера n будет содержать от 2 до n^2 шаблонов (схемат). Схематы определенной короткой длины называют «строительными блоками».

Размер строительных блоков очень важен для быстроты и качества нахождения результата. Рассмотрим примеры строительных блоков для разных схем. Например, в шаблоне ***1 строительным блоком будет элемент 1. В шаблоне 10*** строительным блоком будет составной элемент 10. Заметим, что вид строительного блока должен выбираться из знаний о решаемой задаче, чтобы далее из строительных блоков как из «кирпичиков» собирать «здание», то есть решение с лучшей целевой функцией. Причем, в ГА должно выражаться условие, чтобы строительные блоки разрывались только в крайних случаях, указанных пользователем.

При построении и оценивании большого числа строительных блоков ГА, когда осуществляется случайная генерация и тестирование популяций и хромосом, говорят о так называемых беспорядочных ГА (БГА) [32].

ГА в зависимости от размера популяции разделяются на:

- стационарного состояния,
- поколенческие.

В стационарных ГА размер популяции является входным постоянным параметром, задаваемым пользователем. В поколенческих ГА разрешается увеличивать или уменьшать размер популяции в следующей генерации. Следует отметить, что речь в основном идет о появлении N+K потомков (K>>1), прежде чем начинает реализовываться оператор отбора, устраняющий K хромосом с меньшей ЦФ [32].

Вопросы удаления «лишних» хромосом как в стационарных, так и поколенческих ГА основаны на нескольких эвристиках:

ЭУ1-случайное равновероятное удаление хромосом;

ЭУ2-удаление K хромосом, имеющих худшее значение ЦФ;

ЭУ3-удаление хромосом на основе обратно пропорционального значения ЦФ.

ЭУ4-удаление хромосом на основе заданной турнирной стратегии.

Можно предложить еще ряд эвристик удаления, но в [32] отмечено, что удаление худших хромосом может привести к преждевременной утрате разнообразия и, следовательно, к попаданию ЦФ в локальный оптимум. Оставление в популяции большого числа хромосом с «плохой» ЦФ приведет к слепому поиску.

2.3. Фундаментальная теорема ГА

Генетический поиск требует наличия популяции хромосом. Будем считать, что популяция хромосом A_j, j=1,2,...,n содержится в популяции A(t) во время (или генерацию) t. Пусть схема H строится из алфавита V={0,1,*}. Так, например, схема H длины 7 следующая:

H=*11*0** и стринг A=0111000 есть пример реализации этой схемы H.

Согласно сказанному, схема может быть определена на двоичной хромосоме длины L. Очевидно, что для алфавита мощности k существует $(k+1)^L$ схем. Более того, существует $n \cdot 2^L$ схем, содержащихся в популяции размера n, потому что каждый стринг представляется двумя схемами.

Все созданные схемы не являются равными. Некоторые более специфичны, чем другие. Например, схема 011*1** имеет больше определенных состояний, чем схема 0******.

Для количественной оценки схем введены 2 характеристики [8, 9]:

- порядок схемы – O(H);
- определенная длина схемы – L(H).

Порядок схемы – число закрепленных позиций (в двоичном алфавите – число единиц и нулей), представленных в шаблоне. Например, для схемы H=0****** – O(H)= 1.

Определенная длина–есть расстояние между первой и последней числовой стринговой позицией. Например, в схеме H=0011*1** вычислим, L(H)=6-1=5,а в схеме 0****** вычислим O(H)=1-1=0.

Предположим, что заданы шаг (временной) t, m примеров схем H, содержащихся в популяции A(t). Все это записывают как m=m(H,t) - возможное различное число разных схем H при заданном t. В процессе репродукции хромосомы копируются согласно их ЦФ или более точно: хромосома A_i получает выбор с вероятностью, определяемой выражением (2.1)

После сбора непересекающихся популяций размера n с перемещением из популяции A(t) мы ожидаем получить m(H,t+1)

представителей схемы H в популяции за время t+1. Это вычисляется уравнением

$$m(H,t+1) = m(H,t) \cdot n \cdot \frac{f(H)}{sum[f(j)]}, \qquad (2.4)$$

где f(H) - средняя ЦФ хромосом, представленных схемой H за время t. Если обозначить среднюю ЦФ всей популяции как $\bar{f}$ =sum[f(j)]/n, то

$$m(H,t+1) = m(H,t) \cdot \frac{f(H)}{\bar{f}}, \qquad (2.5)$$

Другими словами, частные схемы «растут» как отношение средней ЦФ схемы к средней ЦФ популяции. Схема с ЦФ выше средней в популяции будет получать больше возможности для копирования и наоборот. Правило репродукции Холланда: «схема с ЦФ выше средней живет, копируется, а с ниже средней ЦФ умирает» [8].

Предположим, что схема H остается с выше средней ЦФ с величиной $c \cdot \bar{f}$, где c-константа. Тогда выражение (2.5) можно модифицировать так:

$$m(H,t+1) = m(H,t) \cdot \frac{\bar{f} + c \cdot \bar{f}}{\bar{f}} = (1+c) \cdot m(H,t). \qquad (2.6)$$

Начиная с t=0 и предполагая, что c - величина постоянная согласно [9], получим

$$m(H,t) = m(H,0) \cdot (1+c)^t. \qquad (2.7)$$

Некоторые исследователи считают, что репродукция может привести к экспоненциальному уменьшению или увеличению схем, особенно если выполнять генерации параллельно.

Лемма 2.1 (Холланд) [8]. Если на некотором шаге генерации ГА P_1 есть вероятность того, что хромосома А определяет потомка на этом шаге и P_2 есть вероятность, что А уничтожается в течение этого шага, тогда ожидаемое число потомков хромосомы А есть P_1/P_2.

Известно также, что вероятность «выживания» хромосомы A_i на шаге t после ОР определяется величиной

$$P_S(t) = (1 - P_2)^{t-1} \cdot P_2. \qquad (2.8)$$

Величина t изменяется от 1, 2,.. g, где g – число генераций генетического алгоритма.

Отметим, что если мы только копируем старые хромосомы без обмена, поисковое пространство не увеличивается и процесс затухает. Потому и используется ОК. Он создает новые хромосомы и увеличивает или уменьшает число схем в популяции.

Рассмотрим, например, частный стринг длины L=7 и две схемы, представляющие этот стринг [9]:

A = 0 1 1 | 1 0 0 0,
H1= * 1 * | * * * 0,
H2= * * * | 1 0 * *,
где | – символ, обозначающий точку кроссинговера.

Схема H1 после ОК скорей всего будет уничтожена потому, что 1 в позиции 2 и 0 в позиции 7 расположатся в различных новых стрингах после ОК. Ясно, что схема H2 будет выживать, так как после ОК число 1 в позиции 4 и 0 в позиции 5 будут в том же самом новом стринге. Хотя мы взяли точку ОК случайно, ясно, что схема H1 менее приспособлена к выживанию, чем схема H2. Если точка ОК выбирается случайно среди L-1=7-1=6 возможных позиций, то ясно, что схема H1 уничтожается с вероятностью P(d)=L(H1)/(L-1)=5/6, она же (схема H1)

выживает с вероятностью P(s)=1-P(d)=1/6. Аналогичным образом, схема H2 имеет L(H2)=1 и вероятность ее уничтожения P(d)=1/6, а вероятность выживания P(s)=5/6 [9]. Очевидно, что нижняя граница вероятности выживания схемы после применения ОК может быть вычислена для любой схемы. Так как схема выживает, когда точка ОК попадает вне «определенной длины», вероятность выживания для простого ОК определяется по формуле

$$P(s) = 1 - \frac{O(H)}{L-1}. \qquad (2.9)$$

Если ОК выполняется посредством случайного выбора, например, с вероятностью P(OK), то вероятность выживания схемы определится выражением

$$P(s) \geq 1 - \frac{P(OK) \cdot L(H)}{L-1}. \qquad (2.10)$$

Очевидно, что выражение (2.10) уменьшается с P(OK) => 1.0. Теперь можно асимптотически учесть эффект репродукции и ОК. Снова определим число частных схем H, ожидаемых в новой генерации. Допуская независимость репродукции (ОР) и ОК, получим [9]:

$$m(H,t+1) \geq m(H,t) \cdot \frac{f(H)}{\bar{f}} \cdot \left[1 - P(OK) \cdot \frac{L(H)}{L-1}\right]. \qquad (2.11)$$

То есть число схем H в новой генерации зависит от двух факторов:

- ЦФ схемы выше или ниже ЦФ популяции;
- схема имеет «длинную» или «короткую» L(H) (определенную длину).

Из выражения (2.11) следует, что схемы с выше средней ЦФ короткой L(H) имеют возможность экспоненциального роста в нов популяции [8, 9].

Рассмотрим влияние мутации на возможности выживани Напомним, что ОК есть случайная альтернативная перестанов элементов в стринге с вероятностью P(OM). Для того, чтобы схема выжила, все специфические позиции должны выжить. Следователь единственная хромосома выживает с вероятностью (1-P(OM)) и частн схема выживает, когда каждая из L(H) закрепленных позиций схем выживает. Умножая вероятность выживания (1-P(OM)) на L(H) ра получим, что вероятность выживания OM $(1-P(OM))^{P(OM)}$. Для малы величин P(OM) (P(OM) << 1) схема вероятности выживания може быть аппроксимирована к выражению [9]

$$P_S(OM) = 1 - L(H) \cdot P(OM). \qquad (2.12)$$

Тогда согласно [8] частная схема H получает ожидаемое числ копий в следующей генерации после OP, OK OM, которое определяет по формуле:

$$m(H,t+1) > m(H,t) \cdot \frac{f(H)}{\bar{f}} \cdot \left[1 - P(OK) \cdot \frac{L(H)}{1-L} - L(H) \cdot P(OM)\right]. \qquad (2.13)$$

Это выражение называется «схема теорем» и фундаментальная теорема ГА [8, 9].

Вероятность выживания схемы H после применения ОИ можн записать [8]:

$$P(OИ) \approx 2 \cdot P_I \cdot \left[\frac{L}{L-1} - \frac{L}{(L-1)^2}\right], \qquad (2.14)$$

где L – длина схемы H, а P_I – вероятность выбора хромосомы, соответствующей схеме H из популяции на заданном шаге генерации.

Вернемся к примеру Гольдберга вычисления max функции $f(x)=x^2$. В добавление к имеющейся информации пусть имеются три частные схемы H1=1****, H2=*10** и H3=1***0, описанные в табл.2.4.

Таблица 2.4

Схема	Перед репродукцией	Представители стрингов	Средняя fitness схемы f(H)
H1	1 * * * *	2, 4	469
H2	* 1 0 * *	2, 3	320
H3	1 * * * 0	2	576

Таблица 2.5

Схема	После репродукции			После всех операторов		
	Ожидаемые вычисления	Действительные вычисления	Представители стрингов	Ожидаемые вычисления	Действительные вычисления	Представители стрингов
H1	3.20	3	2, 3, 4	3.20	3	2, 3, 4
H2	2.18	2	2, 3	1.64	2	2, 3
H3	1.97	2	2, 3	0.0	1	4

Рассмотрим сначала схему H1. В течение фазы репродукции стринги копируются с вероятностью, определенной согласно величине их ЦФ. Из первой строки табл.2.4 следует, что стринги 2, 4 являются представителями схемы H1 (см. табл. 2.2). После OP мы видим, что три копии схемы получены и стринг 3 также вошел в популяцию.

Проверим, соответствует ли это число схеме теорем. Из (2.13) мы ожидаем m $f(H)/\bar{f}$ копий. Вычисляя среднюю ЦФ f(H1), получим (576+361)/2=468,5. Разделив это число на среднюю ЦФ популяции

($\bar{f}$ =293) и умножая на число H1 схем за время t (m(H,t)=2), получим ожидаемое число H1 схем за время (t+1), т.е.

m(H,t+1)=2·468,5/293 = 3.20.

Сравнивая это число с реальным числом копий (схем) – 3, видим, что получили реальное число копий. Продолжая первый шаг дальше определили, что ОК не может дать дальнейший эффект, так как определенная длина L(H1)=1 предотвращает разрыв единственного бита. Далее, для ОМ P(OM)=0,001 мы должны ожидать m-P(OM)=3·0,001=0,003, но нет битового обмена с тремя копиями в трех стрингах. Как результат для схемы H1 получаем ожидаемое экспоненциальное увеличение числа схем, что соответствует (2.13) [9].

Рассмотрим теперь схемы H2 и H3 с двумя закрепленными позициями. Схема H2 имеет также два стринговых представителя в начальной популяции (2,3) и имеет две копии в следующей репродукции. Вычисляем m(H2)=2·320/293=2,18. Здесь 320 – средняя ЦФ схемы, а 293 – средняя ЦФ популяции. Для H3 получим только одно стринговое представление и m(H3)=1·576/293=1,97. Здесь 576 – средняя ЦФ схемы.

Заметим, что для короткой схемы H2 две копии - это хороший результат и ОК может случиться только один раз за четыре возможности (L-1=5-1)=4. В схеме H2=*|1|0|*|* - единственная реальная возможность, которая может дать новую копию. Как результат, схема H2 выживает с высокой вероятностью. Действительное, число H2 схем есть m(H2,t+1)=2,18·0,75=1,64=2. Для схемы H3, т.к. L(H3)=4, ОК обычно разрушает эту схему [9].

Но ответа на вопрос, почему необходимо давать выживание схемам с лучшей ЦФ, не существует или он расплывчатый, или каждый раз зависит от конкретной задачи. Часто возникает вопрос, а сколько новых схем в новой генерации необходимо, сколько реально, сколько полезно?

Холланд считает что качество схем (шаблонов), которое может быть получено из популяции размера «n» с длинной хромосом, равной L, лежит между 2^L и $n \cdot 2^L$. Гольдберг и Холланд считают, что количество «эффективных» шаблонов составляет ориентировочно величину n^3 [8, 9].

Рассмотрим популяцию n двоичных стрингов длины L. Будем считать только схемы, которые выживают с вероятностью больше, чем константа P(s). Как результат упрощения, рассмотрим простой ОК и ОМ с малой вероятностью выполнения. Это приведет к рассмотрению только тех схем, где δ<1-P(s). Будем считать только те схемы, где длина L(s)<δ(L-1)+1.

Ясно, что существует $2^{L(s)-1}$ таких схем. Для вычисления общего числа таких схем берут заданный шаблон и перемещают его вдоль стринга на каждом шаге. Выполним этот шаг общее число L-L(s)+1 раз и можем вычислить общее число схем длины L(s) или меньше следующим образом:

$$2^{L(s)-1} \cdot (L-L(s)+1). \qquad (2.15)$$

Формула (2.15) показывает число схем для одного стринга. Очевидно, для целой популяции число схем ориентировочно определяется как

$$n \cdot 2^{L(s)-1} \cdot (L-L(s)+1). \qquad (2.16)$$

Размер популяции обычно берут равным $n=2^{L(s)/2}$. Считают, что нижняя граница числа схем, «выживающих» после ГА, равна

$$n(s) >= n\,(L-L(s)+1)\,2^{L(s)-2} \qquad (2.17)$$

Если учесть, что $n=2^{L(s)-2}$, то получим

$$n(s) = \frac{\lfloor (L - L(s) + 1) \cdot n^3 \rfloor}{4}. \qquad (2.18)$$

Следовательно, число схем «выживания» пропорционально кубу размера популяции и имеет порядок $n(s)=cn^3$ [8-10].

В большинстве ГА и ЭА используется механизм или процедура случайного спаривания. В биологии это имеет место, когда пол индивида не имеет места. В последнее время обсуждаются механизмы неслучайного спаривания. Основные из них описаны в [32].

Это М1 – вытеснение (crowding factor). Этот механизм определяет способ и порядок замены старых хромосом из генерации t новыми после генерации t+1. Хромосомы стремятся удалить из популяции «похожих на себя» и оставлять отличающиеся.

М2 – разделение (sharing). Этот механизм вводит зависимость ЦФ хромосомы от распределения хромосом в популяции и поисковом пространстве. Это позволяет не появляться в популяциях копиям старых хромосом или близких к ним.

М3 – введение идентификаторов (tagging). Специальные хромосомам присваиваются метки. Операторы ГА применяются только к помеченным хромосомам.

2.4. Выводы

Существует 4 основных отличия ГА от оптимизационных методов:

- прямое преобразование кодов;
- поиск из популяции, а не из единственной точки;
- поиск через элементы (слепой поиск);
- поиск, использующий стохастические операторы, а не детерминированные правила.

В разделе приведена основная теорема ГА, предложенная Холландом. Она определяет асимптотическое число схем, «выживающих» при реализации ПГА на каждой итерации. Очевидно, что это число, конечно приблизительное и меняется в зависимости от вероятности применения ГА. Особенно сильное влияние на число «выживающих» и «умирающих» схем при реализации ПГА оказывает значение целевой функции отдельной хромосомы и всей популяции. Размер строительных блоков оказывает важное влияние на эффективность реализации генетических алгоритмов.

2.5. Контрольные вопросы

1. Поясните смысл понятия «генетические алгоритмы».
2. Выделите основные отличительные особенности ГА.
3. Охарактеризуйте простой ГА.
4. Опишите оператор репродукции в ПГА.
5. Опишите оператор кроссинговера в ПГА.
6. Приведите пример использования ПГА для вычисления min функции $f(x)=x^3$ на интервале [1,2,3,4,5].

7. Покажите пример использования ПГА для вычисления max функции $f(x)=x^4$ на интервале [0,1,2,3,4].
8. Покажите на примере вычисление min функции $f(x)=x^3$ на интервале [1,10].
9. Приведите пример работы ОР.
10. Приведите пример работы ОК.
11. Приведите пример работы ОМ.
12. Приведите пример работы ОИ.
13. Поясните смысл понятия «схема» в ПГА.
14. Фундаментальная теорема (ФТ) генетических алгоритмов.
15. Приведите часть ФТ ГА для ОР.
16. Приведите часть ФТ ГА для ОК.
17. Приведите часть ФТ ГА для ОМ.
18. Постройте пример вычисления на основе ФТ ГА.
19. Покажите на примере вычисление по формуле m(H,t+1) для всего ПГА.
20. Покажите на примере вычисление m(H,t+1) для ОР и ОК.
21. Покажите на примере вычисление m(H,t+1) для ОР и ОМ.
22. Покажите на примере вычисление P(d).
23. Покажите на примере вычисление P(s).
24. Постройте структурную схему решения задач линейного программирования на основе ПГА.
25. Постройте структурную схему ПГА для решения графовых задач.

Истина – это то, что выдерживает проверку опытом.

А. Эйнштейн

3. ВЫЧИСЛИТЕЛЬНОЕ ПРЕДСТАВЛЕНИЕ ГЕНЕТИЧЕСКИХ АЛГОРИТМОВ

3.1. Структура данных

Первичная структура данных для ПГА представляется популяцией стрингов. Можно выбрать много способов для конструирования популяции. Обычно конструируют (задают) популяцию как массив хромосом (индивидуальностей), где каждая индивидуальность содержит фенотип, генотип и ЦФ. Схематично пример популяции из трех хромосом показан в табл.3.1.

Таблица 3.1

Члены	Хромосома	X	$f(x)=x^3$
1	0111	7	343
2	0001	1	1
3	1001	9	729

Приведем структуру данных ПГА, определяющую наибольшее значение $f(x)=x^3$ на заданном интервале, декларированных на языке Си.

```
#define maxpop 20 //Максимальная длина популяции
#define maxstring 10 //Максимальная длина хромосомы
struct population //Структура популяции, описывающая
//популяцию на языке Си
{
float x,fitness;
```

int parent1,parent2,xsite;

BOOL chromosome[maxstring]; /* Массив, содержащий значение хромосомы */

}population[maxpop]; //Массив структур индивидуальностей

Поясним эту запись. В начале программы команды препроцессора устанавливают :

maxpop – наибольший размер популяции, равный 20;

maxstring – наибольшая длина стринг, равная 10 .

Это верхние границы размеров популяции и длины стринга. Далее декларируем саму популяцию с ее компонентами в массиве структур population. Как видно, популяция есть массив структур индивидуальностей, отмеченных от 1 до наибольшего размера. Структура индивидуальностей есть объединенная запись, включающая:

chromosome – одномерный массив элементов типа BOOL, содержащий значения генов хромосомы;

fitness - переменная типа float;

x - переменная типа float.

Приведем описание данных на языке Си:

struct population oldpop[maxpop], newpop[maxpop];

/*Определяются массивы старой и новой популяций*/

int popsize=0,lchrom=0,gen=0,maxgen=0,ncross=0,nmutation=0;

/*Определяются внешние переменные целого типа*/

float pcross=0,pmutation=0,sumfitness=0; /* Определяются внешние вещественные данные */

float avg=0,max=0,min=0; //Вещественные данные статистики

Приведем теперь примерную структуру данных ПГА, декларированных на языке Паскаль [9].*

Constmaxpop = 200

Maxstring = 40

Type allele = boolean; {аллеле=битовые позиции}

chromosom^2 = array [1...max длина] of allele; {хромосома в виде двоичного кода}

individual = record {наилучшее значение}

chrom: chromosome; {Генотип = закодированные хромосомы}

x: real; {Фенотип = неназначенные целые}

fitness: real; {Величина целевой функции = истина}

parent1, parent2, xsite: integer; {Первичные родители и точка ОК целые}

end;

population = array [1.. maxpop] of individual.

Далее отметим, что тип chrom есть сам массив типа allele (в диапазоне между 1 и длиной строки).

В ПГА известные генетические операторы (ГО) применяются к целой популяции в каждой генерации, что схематически показано на рис. 3.1.

---

* ПГА на языке Паскаль приведен в [9].

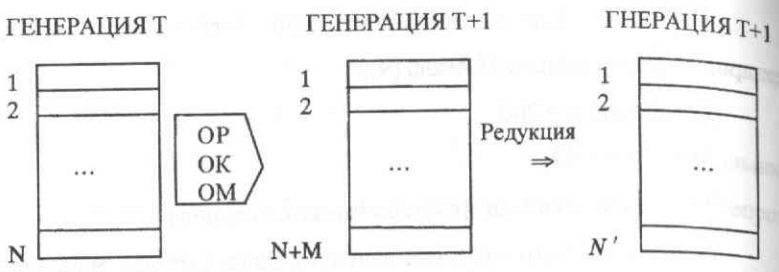


Рис.3.1. Применение ГО.

После применения ГО получаются 2 популяции, рожденные снова, и перемещенные родители. Обычно декларируют 2 популяции: старую (oldpop) и новую (newpop). Новые стринги создаются из членов oldpop, используя ГО, и производится размещение новых индивидуальностей в newpop. В ПГА каждый раз размер популяции остается постоянным. Это реализуется за счет редукции, т.е. случайной или направленной отбраковки «лишних» хромосом. На рис. 3.1 $N' = N$, а М-это число добавленных хромосом после одной генерации. Вообще говоря, в естественном мире популяции, конечно, изменяют свой размер. Следовательно, меняя размер популяции от генерации к генерации, можно улучшить результат решения.

3.2. Реализация генетических операторов на Си

Приведем теперь пример реализации генетических операторов и ПГА на языке Си:

Данная программа, реализующая ПГА, вычисляет функцию $f(x) = x^3$. В начале программы подключаются библиотеки стандартных функций:

```
#include<stdio.h>
```



```
#include<conio.h>
#include<stdlib.h>
#include<time.h>
#include<dos.h>
```

Необходимо отметить, что каждый из трех ГО, представленных ниже, зависит от случайного выбора. В ПГА предусматривается существование двух случайных выборов:

rand – выбирает целые, псевдослучайные числа на интервале [0,(k-1)], где k-число, стоящее после знака %;

rnd – выбирает булевые истинные значения (0 или 1).

```
int rnd(void) /*Функция rnd генерирует случайное число между 0 и 1 */
{
int randc=0; /*определяется переменная, возвращающая целое число, выбранное функцией rand */
randomize(); /*здесь и далее данная функция используется для инициализации датчика случайных чисел*/
randc=rand() % 2; /*генерируется случайное число 0 или 1 и присваивается переменной randc */
return randc; /*возвращается значение randc в функцию, из которой была вызвана функция rnd */
}
```

В связи с тем, что значение переменной x генерируется в двоичной форме (функция initialize), а вычисление целевой функции $f(x)=x^3$ необходимо производить в десятичной форме, используется функция decode для перевода числа из двоичной в десятичную форму.

В эту функцию передается одномерный массив chrom (хромосома), и возвращается десятичное число.

float decode(int chrom[maxstring]) /*данная функция осуществляет перевод значения хромосомы из двоичной в десятичную форму*/

{

int j=lchrom-1; /*определяется первый элемент цикла */

float accum=0,powerof2=1; /*определяются переменные, вычисляющие десятичное число, соответствующее положению единицы в хромосоме */

for(j=(lchrom-1);j>-1;j--)

{

if(chrom[j] = = 1) //проверка на наличие единицы в хромосоме

accum=accum + powerof2; /*если единица присутствует, то переменная accum увеличивается на величину, равную powerof2*/

powerof2=powerof2*2; /*данная переменная наращивается в каждой итерации 2 раза */

}

return accum; /*после выполнения цикла возвращается значение хромосомы в десятичной форме */

}

float objfunc(float x) //функция вычисляет ЦФ

{

return (x*x*x); /*вычисляется и возвращается значение ЦФ хромосомы*/

}

void initialize(void) /*данная функция производит инициализацию начальных данных о популяции*/

int i=0,j=0,kx=2,ky=7; /*определяются переменные для организации циклов и вывода информации на экран*/

cprintf("введите размер популяции (не более N)"); /*здесь и далее везде функция cprintf служит для вывода на экран информации, заключённой в кавычках */

cscanf("%d", &popsize); /*вводится размер популяции и присваивается popsize, с одновременным выводом на экран*/

cprintf("\r\nвведите размер хромосомы (не более L):");

cscanf("%d", &lchrom); /*вводится размер хромосомы и присваивается lhrom*/

cprintf("\n\r введите максимальное число генераций:");

cscanf("%d", &maxgen); /*Вводится число генераций новых поколений и присваивается maxgen*/

cprintf("\n\r Введите вероятность кроссинговера(%):");

cscanf("%f",&pcross); /*Вводится вероятность кроссинговера и присваивается переменной pcross*/

pcross=pcross/100; /*полученную вероятность делим на 100, т.к. она должна быть меньше 1 в %*/

cprintf("\n\rвведите вероятность мутации(%):");

cscanf("%f",&pmutation); /*вводится вероятность кроссинговера и присваивается pmutation*/

pmutation=pmutation/100; /*полученную вероятность делим на 100, т.к. она должна быть меньше 1*/

cprintf("\n\rсгенерированная хромосома:");

for(i=0;i<popsize;i++) /*в данном цикле случайно генерируются хромосомы данной популяции*/

{

for(j=0;j<lchrom;j++) /*в этом цикле формируется одна хромосома*/

{

population[i].chromosome[j]=rnd(); /*генерируется случайно элемент хромосомы(ген)*/

gotoxy(kx,ky); /*данная функция переводит курсор экрана в новую позицию*/

cprintf("%d",population[i].chromosome[j]); /* выводится на экран элемент хромосомы, полученный на данной генерации цикла*/

kx=kx+2; // Вычисляется новая координата курсора по X

}

kx=2; //Вычисляются координаты курсора по X

ky++; // и по Y

}

for(i=0;i<popsize;i++) /*В данном цикле вычисляются первоначальные статистические данные*/

{

population[i].x=decode(population[i].chromosome); //переводится значение хромосомы в десятичную форму

population[i].fitness=objfunc(population[i].x); /*вычисляется значение ЦФ данной хромосомы*/

oldpop[i]=population[i]; /*в oldpop заносятся первоначальные сведения о популяции*/

sumfitness=sumfitness+population[i].fitness; /*формируется первоначальная суммарная ЦФ*/

}

}

В ПГА ОР преобразован в функцию селекции selec как линейный поиск на основе колеса рулетки с весовыми пропорциями каждой части.

Функция selec на языке Си имеет следующий вид:

int selec(struct population pop[maxpop]) /*передается информация о всей популяции*/

{

float randc=0,partsum=0; /*определяются переменные точки останова(randc) и частичной суммы целевой функции(partsum)*/

int j=0; /*определяется счетчик итераций цикла(do-while) или точки останова*/

randomize();

randc=rand()%101; /*генерируется случайное число от 0 до 100 и заносится в randc*/

randc=randc/100; /*randc делится на 100 для получения числа меньше 1*/

randc=randc*sumfitness; /*вычисляется значение, по которому будет рассчитана точка останова*/

do //В данном цикле вычисляется точка останова рулетки

{

partsum=partsum+pop[j].fitness; /*наращивается значение переменной partsum прибавлением к ней значений ЦФ каждой хромосомы*/

j++; //наращивается счетчик

}

while(partsum < randc && j < popsize); /*цикл продолжается пока не будет найдена точка останова или пока не будет достигнута последняя хромосома*/

return (j-1); /*после завершения цикла возвращается значение счетчика, т.е. точка останова*/

}

Функция selec выбирает из популяции величину индекса, соответствующего выбранной индивидуальности. Чтобы сделать это, частичная величина суммы ЦФ превращается в реально изменяемую величину partsum. Реально изменяемое randc содержит место, где колесо останавливается после случайного поворота согласно вычислению randc=randc * sumfitness. Здесь сумма ЦФ - популяции (вычисленная в функции statistics) умножается на нормализованное псевдослучайное число. Поиск продолжается (цикл do-while) через колесо рулетки, пока частичная сумма ЦФ больше или равна точке остановки randc. Итак, функция selec дает путь выбора потомка из родителя для следующей генерации.

Функция однобитовой мутации генерирует случайное число (от 0 до 1) и определяет, где есть обмен (1 к 1), а где нет (1 к 0).

Функция однобитовой мутации на языке Си:

int mutation(int alleleval) //передается значение одного гена

{

float mutate=0; /*декларируется переменная mutate, определяющая, есть обмен или нет*/

randomize();

mutate=rand()%101; /*генерируется случайное число от 0 до 100 и присваивается переменной mutate*/

mutate=mutate/100; /*полученное значение переменной mutate делится на 100, чтобы впоследствии можно было сравнить с вероятностью мутации*/

if(pmutation >= mutate) //выполняется сравнение

{

nmutation++; //Наращивается счетчик числа мутаций

if(alleleval = = 0) /*Выполняется обмен, если 0, то 1, и наоборот*/

alleleval = 1;

else alleleval = 0;

}

return alleleval; /*возвращается полученное значение гена в результате выполнения функции mutate*/

}

Рассмотрим теперь функцию ОК. Функция одноточечного ОК на языке Си имеет следующий вид:

void crossover(int parent1[maxstring],int parent2[maxstring],int child1[maxstring],int child2[maxstring],int jcross) /*данная функция реализует одноточечный кроссинговер*/

{

int j=0; /*определяется переменная, позволяющая организовать цикл for*/

float randc=0; /*определяется переменная randc, устанавливающая есть кроссинговер или нет*/

```
randomize();
randc=rand()%101; /*генерируется случайное число от 0 до 100 и присваивается переменной randc*/
randc=randc/100; /*полученное значение переменной randc делится на 100, чтобы впоследствии можно было сравнить с вероятностью кроссинговера*/
if(pcross >= randc) //выполняется сравнение
{
m0: randomize();
jcross=rand()%(lchrom-1); /*случайно генерируется точка кроссинговера*/
if(jcross = = 0) goto m0; /*если точка равна 0, то возврат на предыдущий шаг и новая генерация точки кроссинговера*/
ncross++; //счетчик числа кроссинговеров
}
else
jcross=lchrom; /*если кроссинговера нет, то за точку кроссинговера принимается местоположение после последнего элемента хромосомы*/
for(j=0;j<jcross;j++) /*в этом цикле формируются новые стринги*/
{
child1[j]=mutation(parent1[j]); /*/выполнение функции мутации, с занесением результата в новый стринг*/
child2[j]=mutation(parent2[j]);
}
```



```
if(jcross != lchrom) /*если кроссинговер есть, то в цикле первому потомку заносятся гены второго родителя, и наоборот, конечно каждый раз выполняется мутация*/
{
for(j=jcross;j<lchrom;j++)
{
child1[j]=mutation(parent2[j]);
child2[j]=mutation(parent1[j]);
}
}
}
```

В финале производится раскодировка пар хромосом, вычисление ЦФ и переход к другой популяции через индекс j=1. Функция decode раскодирует двоичные стринги как единственные неотмеченные целые. Функция objfunc вычисляет ЦФ из раскодированных параметров x.

```
void generation(void) /*Данная функция осуществляет новую генерацию через функции селекции(selec), кроссинговера (crossover) и мутации (mutation)*/
{
int j=0,mate1=0,mate2=0,jcross=0;
/*определяются переменные, необходимые для организации генераций*/
do //Начало цикла перебора каждой индивидуальности
{
```

```
mate1=selec(oldpop); //выбираются произвольно хромосомы
mate2=selec(oldpop); /*для формирования из них новых хромосом*/
crossover(oldpop[mate1].chromosome,oldpop[mate2].chromosome,
        newpop[j].chromosome,newpop[j+1].chromosome,jcross);
//выполняются кроссинговер и мутация вызовом функции crossover
/*Далее формируются две хромосомы новой популяции путем выполнения ранее рассмотренных функций*/
newpop[j].x=decode(newpop[j].chromosome);
newpop[j].fitness=objfunc(newpop[j].x);
newpop[j].parent1=mate1;
newpop[j].parent2=mate2;
newpop[j].xsite=jcross;

newpop[j+1].x=decode(newpop[j+1].chromosome);
newpop[j+1].fitness=objfunc(newpop[j+1].x);
newpop[j+1].parent1=mate1;
newpop[j+1].parent2=mate2;
newpop[j+1].xsite=jcross;

j=j+2; //Наращивается счетчик номеров хромосом
}
while(j <= popsize); //условие окончания цикла
}

void statistics(void) /*данная функция производит все вычисления, необходимые для статистики о популяции*/
```



```
{
int j=0; //декларируется переменная для организации циклов
sumfitness=newpop[0].fitness; /*переменная суммарной ЦФ принимает значение первой хромосомы*/
min=newpop[0].fitness; /*такие же значения принимают переменные min и max, определяющие*/
max=newpop[0].fitness; /*минимум и максимум ЦФ в данной генерации*/
for(j=1;j<popsize;j++) /*Цикл формирования суммы, минимума и максимума ЦФ*/
{
sumfitness=sumfitness+newpop[j].fitness;
if(newpop[j].fitness > max)
max=newpop[j].fitness;
if(newpop[j].fitness < min)
min=newpop[j].fitness;
}
avg=sumfitness/popsize; /*вычисляется среднее значение ЦФ в данной генерации*/
}

void report(void) /*функция report отображает полную информацию о популяции на экране после каждой генерации.*/
{
int i=0,j=0,kx=2,ky=3;
clrscr(); //очистка экрана
```

```
cprintf("\n\r номер генерации: %d",gen+1); /*выводится номер популяции*/
for(i=0;i<popsize;i++) /*в этом цикле выводятся на экран хромосомы*/
{
for(j=0;j<lchrom;j++)
{
gotoxy(kx,ky);
cprintf(" %d",newpop[i].chromosome[j]);
kx=kx+2;
}
kx=2;
ky++;
}
gotoxy(25,1); /*данная функция, как и везде, переводит курсор в новое положение*/
cprintf("максимальная частичная ЦФ: %f",max); /*данная функция выводит те переменные, которые указаны после запятой*/
gotoxy(25,2);
cprintf("средняя частичная ЦФ: %f",avg);
gotoxy(25,3);
cprintf("минимальная частичная ЦФ: %f",min);
gotoxy(25,4);
cprintf("суммарная ЦФ: %f",sumfitness);
gotoxy(25,5);
cprintf("количество кроссинговеров: %d",ncross);
gotoxy(25,6);
```



```
cprintf("количество мутаций: %d",nmutation);
}
```

Приведем код основной программы ПГА на Си:

```
int main(void) //главная функция программы
{
int i=0; //определяется счетчик числа генераций
clrscr(); //очистка экрана
initialize(); /*генерируются первоначальные данные о популяции*/
do /*основной цикл программы, в котором выполняются функции, рассмотренные ранее*/
{
generation();
statistics();
report();
getch(); /*функция задержки результатов программы на экране до нажатия любой клавиши*/
for(i=0;i<maxpop;i++) /*цикл формирования популяции для следующей генерации*/
oldpop[i]=newpop[i];
gen++; //счетчик числа генераций
}
while(gen < maxgen); /*условие окончания основного цикла программы*/
return (0);
```

}

//Конец главной функции

Если функция имеет несколько локальных оптимумов и мы попали в один из них, то увеличение числа генераций может не привести к увеличению ЦФ, то есть наступила предварительная сходимость. Следует отметить, что, изменяя параметры, меняя схему генетического поиска, в некоторых случаях, можно выходить из локальных оптимумов. Эта проблема продолжает оставаться одной из важнейших во всех методах оптимизации. Но в генетических алгоритмах существует прогресс в ее решении.

Регулировка числа копий особенно важна в малых популяциях. При этом может быть, что среднее значение ЦФ близко или совпадает с наибольшим. Это может привести к предварительной сходимости ПГА. В этом случае полезно использование шкалирующей процедуры - линейная шкала. Эта процедура требует линейного отношения между f1 и f в следующем виде: f1=af-b, коэффициенты a и b могут быть выбраны различными способами. Причем f1avg должна быть равна favg.

Выбирая различные шкалы, можно управлять числом потомков в популяции. Причем,

f'max = Cmult·favg,

где Cmult – есть число ожидаемых копий, полученных для лучшего члена в популяции. Для типичных малых популяций (n = 50-100) Cmult= 1,2,.. 10 дает успешные результаты [9, 10].

Для выбора кодирования используют 2 базовых принципа:

- (эффективных) полных строительных блоков;
- минимального алфавита.

Генетические алгоритмы – это мощная стратегия выхода из локальных оптимумов. Она заключается в параллельной обработке множества альтернативных решений. При этом поиск концентрируется на наиболее перспективных из них. Причем периодически в каждой итерации можно проводить стохастические изменения в менее перспективных решениях.

Временная сложность таких алгоритмов лежит в пределах $O(k \cdot n^2)$ – $O(\alpha \cdot n^3)$, где k, α – коэффициенты; n – число входов алгоритмов. Это говорит о перспективности использования генетических алгоритмов при решении любых задач искусственного интеллекта, оптимизации, принятия решений.

3.3. Выводы

Программная реализация простого генетического алгоритма на Си помогает понять сущность генетического поиска. Программная реализация основных генетических операторов может быть использована при решении практических задач. На основе приведенных программных модулей можно построить практически любую оболочку генетического поиска для реализации в ней различных задач с элементами адаптации.

Опыт использования программной реализации ГА показывает, что их преимущество проявляется в таких комбинаторно-логических задачах искусственного интеллекта, где время решения растет экспоненциально. Заметим, что эффективность ГА достигается после настройки всех основных параметров популяции и параметров генетического поиска.

3.4. Контрольные вопросы

1. Приведите структуру данных ПГА на Си.
2. Приведите описание данных ПГА на Си.
3. Опишите функцию Select на Си.
4. Опишите функции ОК и ОМ на Си.
5. Опишите кодирование и декодирование хромосом на Си.
6. Приведите код основной программы ПГА на Си.
7. Опишите процедуру report.
8. В чем состоит проблема предварительной сходимости в ПГА?
9. В чем заключается изменения шкалы?
10. Опишите функцию ОИ на Си.
11. Постройте код основной программы параллельного ГА на Си.
12. Постройте код основного ГА с операторами ОК, ОМ, ОИ в различных последовательностях.
13. Запишите код программы ПГА на языке Си для реализации графовых задач.
14. Запишите код программы ПГА на языке Си для реализации оптимизационных задач.

Все течет и нет ничего неподвижного в мире.

Овидий

4. ИНСТРУМЕНТАЛЬНЫЕ СРЕДСТВА ГЕНЕТИЧЕСКИХ АЛГОРИТМОВ

4.1. Технология генетического поиска

Для повышения качества ГА используют различные технологии генетического поиска. Они связаны в основном с распараллеливанием ГА, структурированием популяции, миграцией хромосом и популяций и т.п. [27].

Например, обычно используется два подхода структурирования популяции:

- После конструирования популяции она случайным или заданным образом разбивается на несколько подпопуляций. ПГА или любая его модификация применяется внутри каждой подпопуляции. При этом возможно случайное или направленное перемещение хромосом между любыми или заданными популяциями.
- После инициализации популяции для каждой хромосомы определяется ее пространственное местоположение в популяции. Далее возможно разбиение популяции или нет. Выбор пар для использования ГО зависит от «близости» решений в пространстве (значение ЦФ).

Возможно большое количество комбинаций и модификаций этих подходов. Причем эффективность их устанавливается в основном эмпирически.

Гудман, Панч, Декстер, Ванг [32] предложили двухуровневую систему ГА (DAGA2). В DAGA2 реализована идея многоуровневой эволюции, когда ГА используются на двух уровнях. На первом уровне DAGA2 выполняет параллельный прогон ГА (обычных ПГА), пока они не начнут сходиться. Затем в наилучших решениях (хромосомах) каждой популяции определяются ближайшие друг к другу объекты (гены), которые образуют кластеры. На основе кластеров создается новая хромосома как объект для второго уровня DAGA2. Отметим, что DAGA2 может действовать как параллельный ГА со скрещиванием в локальной области или на основе «островов». DAGA2 действует как самоадаптируемый ГА в смысле «выживания» самого приспособленного решения. DAGA2 аналогична архитектуре ГА «впрыскивания» [32]. В этой архитектуре хромосомы из подпопуляций в грубой (неточной) модели задачи мигрируют в другие подпопуляции в негрубой (точной) модели задачи. DAGA2 может быть модифицирована на любое число уровней в зависимости от памяти и временных ограничений на получение результата.

Вейсберг предложил следующую схему многоуровневого ГА [9] (рис.4.1).

Рис.4.1. Схема многоуровневого ГА

Данная схема предлагает последовательное использование неадаптированного ГА, адаптированного ГА и различных преобразований внутри одной популяции.

Известны также параллельные ГА со скрещиванием в локальной области [32] и параллельные ГА на основе «островов» [32]. В первом случае хромосомы одной популяции моделируются как точки в некотором пространстве. Вводится отношение близости, и ГО применяется к хромосомам, расположенным близко друг от друга. Во втором случае популяции эволюционируют в течение нескольких заданных генераций независимо на определенных «островах», а затем

производится обмен генетическим материалом, и снова независимая эволюция. Этот процесс может происходить в цикле разнообразными способами.

Парми [32] ввел понятие кластерно–ориентированного ГА (КОГА), основанного на стратегии быстрого разбиения поисковых пространств на области высоких значений функции полезности. В КОГА установлен режим мутации переменных ГА, введен адаптивный фильтр, отсекающий решения с низким значением функции полезности. Причем нижняя граница функции полезности адаптивна в смысле зависимости от решений на каждой генерации. В качестве модификации КОГА предлагается начальное структурирование популяции на основе ЦФ, использование двухточечного ОК и универсальный стохастический выбор.

Шаффер и Эшельман [32] предложили проблемно-ориентированные ГА (ПОГА), вырабатывающие решения (фенотипы) комбинаторных задач, представленных в виде задач параметрической оптимизации, из строк параметров (генотипов). В ПОГА создается проблемно–ориентированный генератор решений (хромосом). Кроме этого создается эвристический генератор планов (ЭПГ), состоящий из набора шаблонов решений. Далее на основе «жадной» эвристики [34] и ЦФ строится возможный план решения задачи. Изменяя параметры, ГА заставляет «жадные» эвристики вырабатывать большое число разнообразных хромосом. Взаимодействие ГА и ЭПГ позволяет устанавливать время генерации, производить адаптацию наборов параметров, фокусировать наилучшие решения.

Босворт [9] предложил 5 различных операторов мутации:

- FR мутация использовала идею золотого сечения;

- случайная мутация;
- квадратичная гауссова аппроксимационная мутация;
- кубичная гауссова аппроксимационная мутация;
- нулевая мутация.

Все эти мутации связаны с эвристическими правилами выбора точки разрыва хромосомы для мутации.

Фогель и др. [9] предложили механизм эволюционного программирования [24]. В этой работе последовательные символы задач были представлены посредством поиска через пространства автоматов с конечными состояниями. Рассмотрим пример [9]. Пусть задана диаграмма автомата с тремя состояниями: {0,1}-входы, {α,β,γ}-выходы {A,B,C}-состояние (рис.4.2).

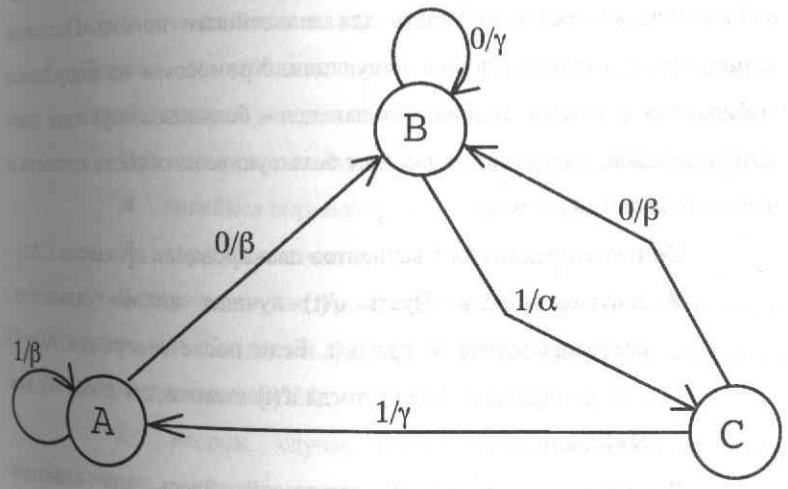


Рис.4.2. Автомат с тремя состояниями

Например, если автомат находился в состоянии A и получил входной символ 1, тогда β - будет выход и автомат останется в состоянии A. С другой стороны, если автомат находится в состоянии A

и получает на вход 0, то автомат переходит в состояние B. Этот тип автомата был использован для повторения циклов выходных символов, используя идею ЭП, которая состоит из 2 операторов-селекции и мутации. Оператор селекции выбирает 2 автомата родителей и потомка (мутированный автомат). В ОМ здесь был только механизм структурной модификации. ОМ модифицировал диаграмму конечных состояний автомата. Потомок получался отличным от родителя в выходном символе, состоянии перехода, числе состояний или входа. ЭП со случайным выбором диаграмм автоматов с конечными состояниями и селекцией лучших - это мощный метод для проблем с малым пространством поиска.

Де Йонг [9,27]предложил в пересекающихся популяциях число n·G-индивидуальностей выбирать для дальнейшего поиска. Потомки размещаются в существующие популяции. Хромосомы из популяции выбираются случайно. Вывод: нет панацеи - большая популяция дает потери аллелей, малая популяция дает большую возможность попасть в плохой локальный оптимум.

Он также предложил 5 вариантов планирования процесса ГА:

- элитная модель. Пусть a'(t) лучшая индивидуальность, сгенерированная за время t. Если после генерации A(t+1) a'(t) не попала в A(t+1), тогда a'(t) включается в A(t+1) как N+1 член;
- модель с ожидаемой величиной. Здесь для селекции выбирается индивидуальность и вычисляется ожидаемое число потомков для каждого стринга.
- элитная модель с ожидаемой величиной. Это объединение 1 и 2 модулей.

- crowding factor (CF). Когда индивидуальность появилась, она была выбрана к «смерти». Умирающая популяция была выбрана из подмножества CF членов, выбранных из полной популяции случайно. Обычно CF=2;
- модель генерации кроссинговера. Здесь выбираются число точек ОК.

Одной из наиболее простых технологий генетического поиска является так называемая «устойчивая репродукция» (stady-state reproduction) [10]. Она имеет следующий механизм:

1. Создать n потомков (хромосом) из начальной популяции через репродукцию.
2. Убрать n членов из начальной популяции, чтобы освободить место для получаемых потомков.
3. Оценить и вставить полученные хромосомы (потомки) на освободившиеся места в популяции.

При оценке хромосом используется два основных подхода:

- линейная нормализация,
- определение «окна».

В первом случае производится упорядочивание хромосом по убыванию ЦФ. Создается фитнесс с постоянной величиной и шкала линейного уменьшения ЦФ.

Во втором случае определяется наименьший фитнесс. Хромосомы, у которых значение фитнесса меньше определенного, не участвуют в репродукции.

Такой метод показал эффективные результаты на детерминированных проблемах [10].

Далее рассмотрим модификации селекции. Бриндл [9] предложил шесть схем селекции:

- детерминированная схема селекции. Здесь вероятность селекции вычисляется как обычно: $P_{SELECTi}=f_i/sum[f]$. Тогда ожидаемое число индивидуальностей для каждого стринга a_i вычисляется как $K_i=P_{SELECTi}\cdot N$. Каждый стринг есть свободный элемент согласно целой части величины K_i, и популяция сортируется согласно фракционным частям величин K_i. Оставшиеся стринги пополняют популяцию из вершины сортировочного листа;
- остаточная стохастическая схема с перемещением;
- остаточная стохастическая схема без перемещения. Обе стартуют идентично первой схеме. В предыдущей схеме фракционные части ожидаемых значений чисел используются для вычисления веса с помощью рулетки и используются для заполнения элементов в популяции. В этой схеме фракционные части используются как вероятности. Один за одним после броска монеты заполняются фракционные части как успешные вероятности;
- стохастические схемы с перемещением (обыкновенная рулетка);
- стохастические схемы без перемещения - это третий вариант планирования;
- стохастическая турнирная схема. Здесь вероятность селекции вычисляется обычно и лучшие пары выбираются

на основе рулетки. Затем в каждой паре стринг с высшей ЦФ объявляется победителем и вставляется в новую популяцию.

Шапиро, Реттрей, Прюгель–Бенетта [32] разработали теорию, которая на основе принципов статистической механики анализирует динамику ГА при решении комбинаторных задач. Описание, рассматривающее, какие решения (хромосомы) содержатся в популяции, названо микроописанием. Макроскопический уровень описывает ГА в терминах статистических свойств популяции. Статистическая механика используется в двух эвристиках: в первой - для надежного расчета ожидаемых результатов применения ГО и для расчета отклонений от ожидаемых значений; во второй - для анализа информации, полученной из макропараметров методом максимума энтропии. В [32] введено понятие корреляции хромосом, т.е. насколько «похожи» между собой хромосомы. Очевидно, что для любой популяции на основе математического ожидания и распределения ЦФ можно вычислить корреляцию. Каждому значению корреляции ставится в соответствие энтропия, равная минус логарифму от количества популяций, на которых корреляция принимает данное значение. Тогда максимум энтропии, согласно [32], соответствует наиболее часто встречающемуся значению корреляции среди всех популяций фиксированного размера. Эти утверждения основаны на том, что имеется высокая вероятность ОК, ОМ и других операторов при фиксированных известных значениях макроопераций. Следовательно, основная проблема здесь – это выбор для каждой конкретной задачи своих макропараметров. Использование предсказаний на основе

максимума энтропии может дать прогнозы ожидаемых значений при усредненном числе прогонов ГА.

Корреляция (сходство) между строками A_i и A_j популяции A, взвешенной в соответствии с относительной важностью элементов, определяется

$$q^{ij} = \frac{1}{L}\sum_t A_i^t A_j^t, \qquad (4.1)$$

где L – длина хромосом в популяции

$A=\{A_1, A_2, \ldots A_i, A_j, \ldots\}$, $|A| = N$, t – элемент в хромосоме, i, j – метки хромосомы в популяции.

Средняя корреляция согласно [32] определяется

$$q = \frac{1}{N(N-1)}\sum_i \sum_{j \neq i} q^{i,j}. \qquad (4.2)$$

В [19] предлагается понятиям ЦФ и функции полезности (пригодности) (ФП) придать разный смысл. ЦФ – это средство измерения качества отдельно взятой хромосомы. ФП – описывает способ выбора хромосом из популяции. Тогда вероятность выбора хромосомы A_i определяется ее ФП F^i в соответствии с выражением

$$P(A_i) = \frac{F^i}{\sum_{j=1, j \neq i}^{N} F^j}. \qquad (4.3)$$

Конкретная форма отбора хромосом определяется функциональной зависимостью между ЦФ и ФП.

В работе [28] предложена новая архитектура ГА, основанная на миграции и искусственной селекции (GAMAS). GAMAS основан на концепции ПГА и кроме того включает шкалирование ЦФ, эволюцию

нескольких параллельных популяций, или видов, и три новых оператора – миграции (ОМГ), искусственной селекции (ОИС) и новых циклов развития (ОНЦР). Основная цель GAMAS заключается в том, чтобы снять или снизить проблему предварительной сходимости (попадания в локальные ямы без выхода из них) ГА.

На рис. 4.3 показана концепция макроэволюции на примере архитектуры GAMAS [28].

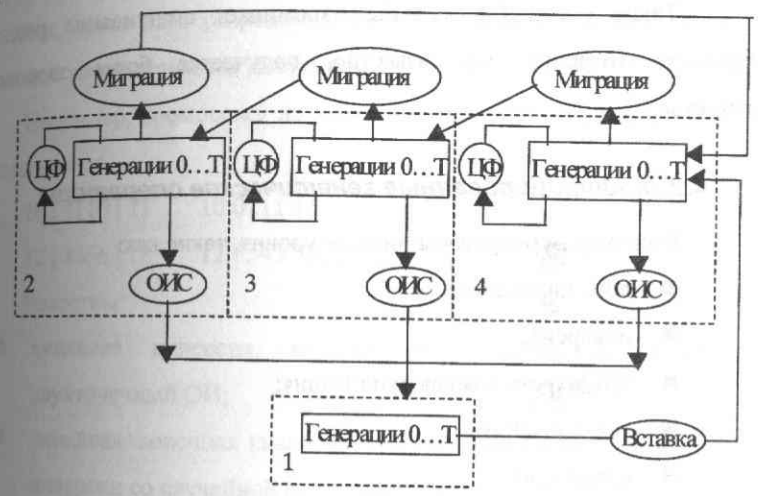


Рис. 4.3. Архитектура GAMAS

На начальном этапе GAMAS (рис. 4.3) создает блоки 2,3,4. Блок 1 создается позднее, как результат работы ОИС. В блоке 2 оператор мутации с вероятностью P_m=0,05 применяется к части популяции. Блок 4 использует случайно созданный ОМ с P_m=0,005, а блок 3 использует ОМ с 0,005< P_m<0,05. Оператор искусственной селекции использует элитную стратегию.

Элитная стратегия закрепляет "наличие" хромосомы посредством обязательного копирования хромосом с лучшей ЦФ в

каждой генерации [10]. Эта стратегия может увеличить скорость обработки популяции, но не гарантирует выход из локальных оптимумов и не решает проблемы преждевременной сходимости ГА.

Блок 1 является как бы хранителем хромосом, которые не подвергаются ОМ и ОК. Он выбирает наилучшие хромосомы, которые анализируются в блоке- вставке и лучшие из них направляются для новой обработки в блок 4.

Такая схема ГА как бы напоминает смешивание рас, в результате которого, как известно, получается более здоровое потомство.

4.2. Модифицированные генетические операторы

Рассмотрим операторы низкого уровня, такие как:

- доминирование;
- инверсия;
- интрахромосомная дупликация;
- транслокация;
- сегрегация.

Кроме того, в ГА используются и операторы высокого уровня, ориентированные на преобразование популяций, такие как:

- миграция;
- ограничение;
- разделяющая функция.

Существование в природе диплоидных и полиплоидных организмов ставит вопрос: почему не использовать это в искусственном генетическом поиске?

В диплоидной форме генотип происходит на одной или более парах хромосом (называемых гомологичные хромосомы), где каждая содержит информацию о той же самой функции.

Например, давайте считать, что диплоидная хромосомная структура-это различные буквы, представляющие различные аллели (различные функциональные значения гена).

Оператор инверсии (ОИ). Две точки выбираются вдоль длины хромосомы, хромосома разрезается по этим точкам, и концы точек разрезанной точки меняются местами.

Например, хромосомы до инверсии приведены справа, а после инверсии - слева.

10 | 1110 | 11 10 0111 11,

12 | 3456 | 78 12 6543 78.

Известны:

- линейная инверсия, которая реализована как простой двухточечный ОИ;
- линейная+конечная инверсия, реализованная в виде линейной инверсии со случайной вероятностью;
- продолжающаяся инверсия. Здесь инверсия со специальной вероятностью инверсии Р(ОИ) применяется к каждой новой созданной хромосоме;
- массовая инверсия. Не выполняется, пока вся новая популяция не создана, затем половина популяции подвергается ОИ, с использованием двух точек инверсии.

Рассмотрим, например, обычный одноточечный ОК между произвольной негомологичной парой хромосом:

A = 1 2 3 4 | 5 6 7 8

B = 1 2 6 5 | 4 3 7 8

A′ = 1 2 3 4 4 3 7 8,

B′ = 1 2 6 5 5 6 7 8.

Анализируя хромосомы-потомки, мы видим, что в A′ повторяются гены 3, 4, а гены 6, 5 отсутствуют. В хромосоме потомка B′ ген 5, 6 повторяется, а гены 3, 4-отсутствуют.

Решения технологических задач в ГА, когда происходит потеря генетического материала, называют «нелегальными». Очевидно, необходима разработка таких генетических операторов, которые обеспечивают отсутствие нелегальных решений и высокую вероятность выживания решений с «хорошими» свойствами.

Для технических задач наиболее применимы три ОК [9,29]:

- частично соответствующий ОК (PMX);
- упорядоченный ОК (OX);
- циклический ОК (CX).

В PMX 2 точки ОК выбираются случайно вдоль стринга. Эти две точки определяют соответствующую секцию. PMX осуществляет позиционный обмен. Например:

До ОК	После ОК
A = 9 8 4 | 5 6 7 | 1 3 2 0	A′=984 | 23 0 | 1657
B = 8 7 1 | 2 3 0 | 9 5 4 6	B′=8 01 | 567 | 9243.

Сначала устанавливают соответствие: B-A, при этом гены 5-2, 6-3, 0-7 меняются местами. Все остальные места заполняются согласно частичному соответствию.

В OX начало аналогично PMX, то есть выбирается секция соответствий. Затем в хромосоме B вместо элементов из секции A ставятся знаки H:

A= 9 8 4 | 5 6 7 | 1 3 2 0;

B= 8 7 1 | 2 3 0 | 9 5 4 6;

B= 8 H 1 | 2 3 0 | 9 H 4 H.

Затем хромосома B переупорядочивается, вместо знаков H ставят элементы из A:

B = 2 3 0 H H H 9 4 8 1;

B' = 2 3 0 5 6 7 9 4 8 1;

A' = 5 6 7 2 3 0 1 9 4 8.

Видно, что OX и PMX подобны.

Рассмотрим теперь CX. Он производит рекомбинацию под ограничением, что каждый номер приходит из одного родителя или другого, например:

C = 9 8 2 1 7 4 5 0 6 3

D = 1 2 3 4 5 6 7 8 9 0

После применения CX хромосомы C, D примут вид:

C′ = 9 2 3 1 5 4 7 8 6 0

D′ = 1 8 2 4 7 6 5 0 9 3.

Вместо вектора точки ОК выбирается цифра 9 из первого родителя. Следующим выбирается ген 1, так как 1-9 в D. Далее 1-4, 4-6 и 6-9 и мы выбрали цикл 1. Цикл 2 заполняется из стринга 2 (2,3,0,8,2,), III цикл-(5,7,5). Для D' IV (1,4,6,9,1), V-(8,2,3,0,8), VI (5,7,5).

Сегрегация – это случайный селекционный процесс, создающий новую хромосому, которая выбирает по одному или более членов из каждого стринга популяции для формирования новой хромосомы.

Например, покажем реализацию оператора сегрегации (ОС):

A_1 = (1 2) 3 4 5;

A_2 = 2 3 1 (4 5) => ОС => $A_н$ = 1 2 4 5 3;

A_3 = 1 4 5 2 (3).

Оператор транслокации (ОТ) является модификацией ОК. Он инвертирует часть хромосомы, расположенной справа от точки разрыва хромосомы.

Например:

A_1 = 1 3 | 4 5 2 => A_1' = 1 3 2 5 4;

A_2 = 3 2 | 1 5 4 => A_2' = 3 2 4 5 1.

Здесь A'_1, A'_2-хромосомы, полученные после применения ОТ.

Дупликация - это удвоение некоторых генов и расположение их на хромосоме.

Делеция - это удаление повторяющихся генов внутри хромосомы.

4.3. Методы, основанные на знаниях

Считается, что обмен строительных блоков под ОР и ОК подобен человеческой процедуре инновационного мышления. Параллельный обмен между стрингами и популяциями формирует новые стринги и новые идеи. Пользователи всегда приносят дозу знаний о решаемой проблеме, а затем используют различные типы

моделирования. Другими словами, операторы человеческой инновации мышления направляются (по крайней мере, иногда) посредством знаний. Здесь они вступают в противоречие с ГА слепого поиска. Поэтому сейчас развивается направление, базирующееся на объединении проблемно-специфической информации с ГА. Рассмотрим кратко: гибридные методы; операторы, ориентированные на знаниях [9].

Схема одного из гибридных ГА имеет вид представлений на (рис.4.4).

Рис.4.4. Схема гибридного ГА

Здесь ГА сортирует пики, а затем применяется локальный поиск типа "восхождение на холм". Другой метод называется "G-bit улучшение" и состоит из трех следующих шагов:

- выбрать одну или более лучших стрингов из текущей популяции;
- переставляя, анализировать бит за битом, выполнять успешный однобитовый обмен в стрингах, остающихся лучшими после последних двух обменов;
- в конце анализа вставить лучшую структуру или k лучших структур в популяцию и продолжать нормальный генетический поиск.

Этот метод также может включать двухбитовый или трехбитовый обмен. Однако в комбинаторных задачах резко возрастает пространство поиска [9].

Параллельный гибридный ГА имеет вид, представленный на рис.4.5[9].

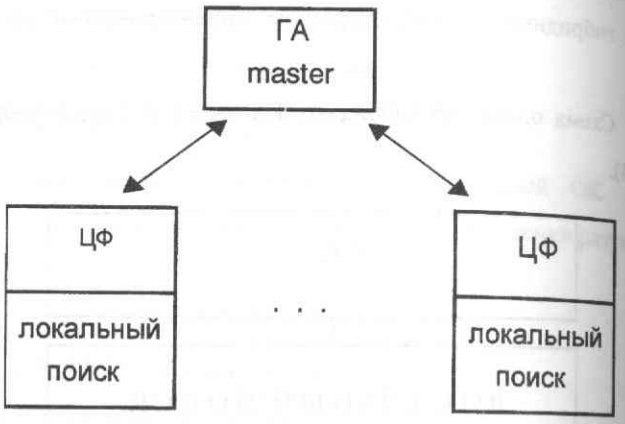


Рис.4.5. Параллельный гибридный ГА

Здесь мы имеем возможность использовать одновременно набор процессоров, чтобы моделировать функции различных стрингов в популяции. Процессоры могут использоваться для вычисления величины ЦФ стрингов. Они могут также быть использованы с целью выполнения итераций локального поиска для попыток улучшения текущих стрингов.

Рассмотрим операторы, ориентированные на знания. Первым был разработан ориентированный на знания ОМ. Он использовал градиентный метод и метод золотого сечения как ОМ.

Грефенстете и другие ученые развили ориентированный на знания ОК для решения задачи коммивояжера (TSP) [9, 31,34]. Он

использует "жадную" эвристику (greedy). Рассмотрим задачу TSP. Имеем тур (1 3 5 4 2): 1-3-5-4-2-1. Очевидно, что простой ОК не гарантирует потомка в виде нового тура. Например, выполним ОК на двух турах:

До ОК:	После ОК:
T_1 : 5 4 3 | 1 2,	T_1': 5 4 3 4 5,
T_2 : 1 2 3 | 4 5.	T_2': 1 2 3 1 2.

Пересечение двух смежностных туров может так же дать нелегальные решения. Например:

T_1 : 5 4 1 | 3 2,	T_1': 5 4 1 5 1,
T_2 : 2 3 4 | 5 1.	T_2': 2 3 4 3 2.

В [31] предложен ОК, использующий представление смежности, и эвристический ОК (жадный кроссинговер), который конструирует потомков, выбирая лучшие из двух родительских ребер. Этот оператор конструирует потомков из двух родительских туров следующим образом: выбирается случайный город, как стартующая точка в туре потомка. Далее сравниваются два ребра, выходящие из стартующего города в родителях, и выбирается короткое ребро. Продолжая расширять частичный тур, выбирают каждый раз короткое ребро из двух возможных родителей. Если короткое родительское ребро дает частный цикл, то выбирается случайное ребро. Процесс продолжается, пока тур не сгенерирован.

При решении проблем используются специальные знания, позволяющие построить аппроксимационную модель нашей проблемы. При использовании ГА это может уменьшить объем и время

вычислений и упростить моделирование функций, сократить число ошибок моделирования.

4.4. ГА и параллельные процессоры

Рассмотрим применение ГА на параллельных вычислительных архитектурах. Существуют четыре основных архитектурных прототипа [9,12,14,21]:

- синхронные управляющие. Здесь единственный управляющий процессор (УП) (хозяин) управляет работой k процессоров (рабов). УП управляет селекцией, спариванием и выполнением ГО. Процессоры просто выполняют моделирование и вычисление функций. Архитектура очень проста, но имеет 2 недостатка: время работы зависит от вычисления ЦФ; если УП вышел из строя, то весь процесс остановится;
- асинхронные управляющие. Эта архитектура требует синхронизации операций посредством селектирования членов на процессорах-«рабах». Они также сильно зависят от процессора-«хозяина»;
- распределенные, асинхронные конкуренты. В асинхронных ГА К идентичных процессоров выполняют одновременно ГО и моделирование функций. Распределенная память требует, чтобы процессоры избегали одновременного обращения к идентичным местам памяти.
- сети. В схеме сетей k независимых простых ГА работают с независимыми областями памяти, независимыми ГО и независимо моделируют функции. k процессоров работают

нормально, за исключением того, что лучшие стринги, найденные в генерации, передаются другим подпопуляциям через коммуникации сети. Способность таких схем высока, потому что происходит автономный независимый процесс.

Гольдберг [9] предложил объектно-основанную проектирующую процедуру для параллельных ГА. Она состоит из двух моделей:

- модель общества;
- модель работы заводов.

В модели общества ГА представляет собой множество соединенных сообществ, как показано на рис.4.6.

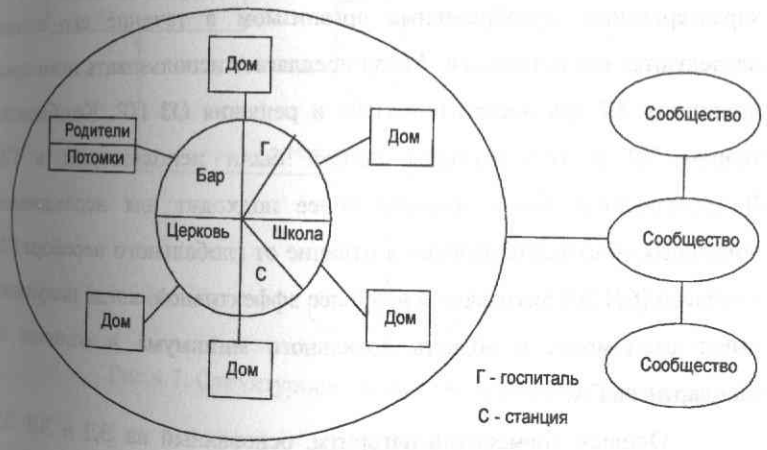


Рис.4.6. Модель общества Гольдберга.

Сообщества состоят из домов, соединенных связями в города. Родители рожают потомков в домах и вычисляют ЦФ там же. Дети идут в бар, церковь, где они встречаются с возможными индивидуумами для соответствия. После встреч они идут на биржу и ищут дом. Дома

выдаются в соответствии с ЦФ. Если город переполнен, то биржа посылает детей в другое сообщество, или можно идти на станцию, или случайно, либо направленно поехать в другое сообщество.

Модель работы заводов или цехов завода состоит из серии вершин, соединенных сетью связей. Такая модель позволяет получать подпопуляции или популяции более или менее изолированно друг от друга. Отметим, что возможны различные способы связей: звезда, полный граф, четыре соседних вершины, 6 соседних и т.д. Можно также использовать нейронные сети и нечеткие процессоры.

Отметим, что раньше теории эволюции Дарвина (ЭД) о естественном отборе Ламарк предложил многогранную теорию эволюции (ЭЛ) [62-67]. Теория основана на понятии, что характеристики, приобретенные организмом в течение его жизни, наследуются его потомками. Автор предлагает использовать некоторые принципы ЭЛ для построения ИИС и решения ОЗ ПР. Комбинации теории ЭЛ и генетического поиска были использованы в ГА. Локализованный поиск Ламарка более подходит для исследования локальных областей популяции в отличие от глобального перебора ГА. Согласно [62] ЭЛ оказывается наиболее эффективной, когда популяция имеет сходимость в область локального минимума в отличие от стандартного ГА.

Опишем совместный алгоритм, основанный на ЭД и ЭЛ. ЭД реализован в виде ГА, в который встроен алгоритм ЭЛ. Он содержит элемент Ламарка (Шкала, G), который используется для контроля эволюции. Шкала – это действительное число от 0 до 1, определяющее процентное соотношение в популяции, к которому будет применена

ЭЛ, а G – число итераций ГА. На рис.4.7 показана схема алгоритма

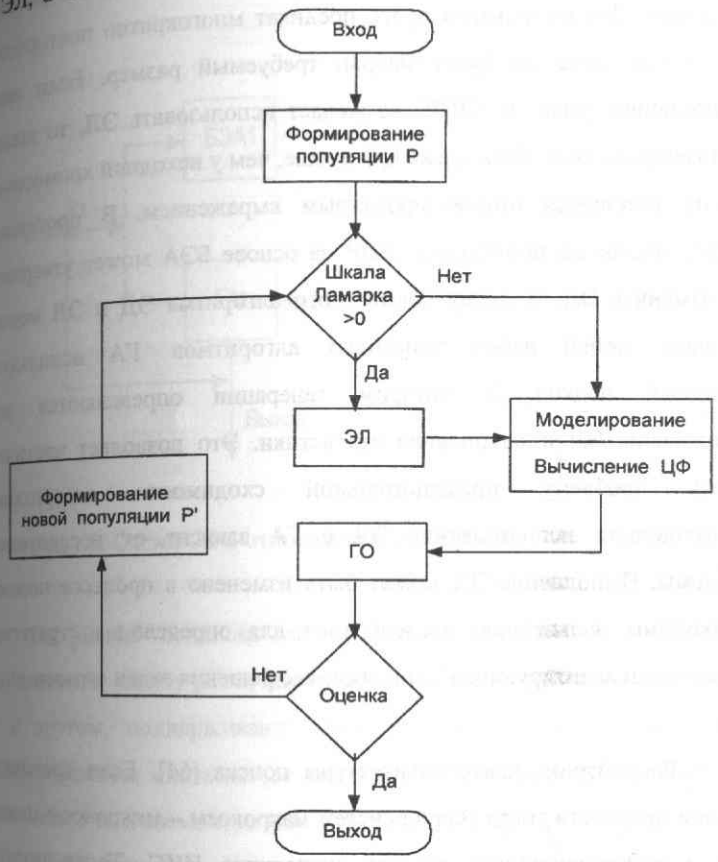


Рис.4.7. Структурная схема совместного алгоритма ЭЛ-ЭД

совместной ЭД и ЭЛ. Первый элемент предусматривает случай, когда пользователь применяет ЭЛ ко всей популяции (Шкала –1). Этот элемент собирает полное множество номеров хромосом в популяции и для них вызывает ЭЛ. Второй элемент используется, если процентное соотношение в популяции больше 0. Он вызывает предикат для выбора

множества номеров хромосом, лежащих в пределах от 1 до размера популяции. Для выполнения этого предикат многократно повторяется до тех пор, пока не будет набран требуемый размер. Если лицо принимающее решение (ЛПР) не желает использовать ЭЛ, то шкалу выбираем равную 0. Если новая ЦФ лучше, чем у исходной хромосомы, то она заменяется оптимизированным выражением. В противном случае, замены не происходит. ЛПР на основе БЭА может утвердить или отменить ЭЛ. В схему совместного алгоритма ЭД и ЭЛ можно включить любой набор поисковых алгоритмов. ГА использует гибридный подход, в котором генерации определяются для исследования ЭЛ и накопления статистики. Это позволяет частично решать проблему предварительной сходимости алгоритмов. Эффективность использования ЭЛ с ГА зависит от исследуемой проблемы. Выполнение ЭЛ может быть изменено в процессе поиска. Необходимы дальнейшие исследования для определения стратегии, динамически адаптирующей ЭЛ в процессе решения задач оптимизации (ЗО).

Рассмотрим новые архитектуры поиска [64]. Если следовать учениям древности, тогда теория систем макрокосм – микрокосм может быть в информационном смысле аналогична ИИС. Такая модель представляет две взаимодействующие противоположности. Все последующие иерархические уровни увеличивают число противоречий. Строительный блок (СБ), на основе которого может быть построена многоуровневая система для решения ОЗ ПР, показан на рис.4.8. Здесь на создание новой популяции Р' оказывает влияние не только блок эволюционной адаптации (БЭА), но и внешняя среда. Из таких СБ, как из «кирпичиков» может быть построена система любой сложности.

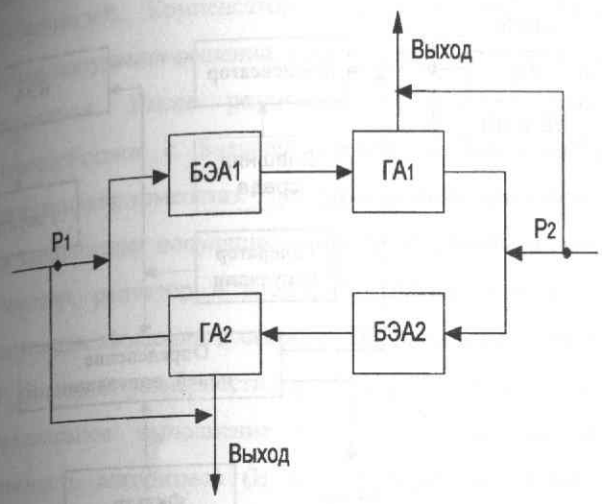


Рис.4.8. Схема СБ с относительным балансом

На рис.4.9 показана схема СБ с относительным балансом. Здесь элементы имеют различные свойства, но, будучи взаимосвязаны друг с другом, поддерживают относительный баланс. Нормальное функционирование системы (ее свойства и возможности) связано с равновесием в нем элементов Инь – Ян (БЭА1 – БЭА2).

Приведенные схемы СБ для создания многоуровневых иерархических процессов в ГА во многих случаях помогают выходить из локальных оптимумов [64].

Рис. 4.9. Комбинированная схема ПР на основе БЭА

Стандартная структура системы поддержки принятия решений (СППР) состоит из следующих основных блоков : генерации возможных альтернатив решений (сценариев); оценки решений (построения ЦФ)); согласование решений; анализ динамики развития ситуации; выбор решения (группы решений), сценария; оценка соответствия ПР заданным целям. Сравнивая эту структуру со структурой генетического или эволюционного поиска, можно заметить много общего. В этой связи предложенные концепции, принципы, архитектуры и ГА могут быть эффективно использованы в СППР. Предлагается комбинированная схема ПР с обратной связью, имеющая вид (рис.4.9). На рис.4.9 ЭД и ЭЛ - это совместная реализация эволюции Дарвина и Ламарка. Редуктор объединяет свойства экспертной системы, ЛПР, блоков установления аналогий и

повторителей. Компенсатор регулирует динамически изменяемый размер популяции решений, а именно расширяя, сужая или оставляя ее постоянной. После реализации ЭД и ЭЛ компенсатор при взаимодействии с внешней средой реализует баланс, а фильтр поддерживает гомеостаз. При этом лучшие хромосомы отправляются для смешивания популяций и выхода из локальных оптимумов. Блоки сумматор, редуктор и фильтр позволяют повысить эффективность реализации эволюции и скорость ПР [64]. Следует отметить, что в ЗО ПР большой размерности процесс решения резко усложняется, но параллельное выполнение ГО на порядок снижает временную сложность алгоритмов (ВСА). Отметим, что схемы с подобными обратными связями присутствуют в ЕС и эффективно функционируют.

Наиболее популярны в настоящее время следующие программные пакеты генетических алгоритмов:

- GENI TOR [9],
- GENESIS [9],
- HYPERGEN [32],
- GARAG [32],
- OOGA [10].
- GA Workbench [32] и др.

1,2 пакеты используются в основном для развития последовательных ГА [9]. Пакет 4 разработан под руководством Гудмана в Мичиганском государственном университете (США).

Пакет 5 представляет собой объектно–ориентированный ГА, (ООГА), приведенный Девисом в [10]. Основные элементы ООГА описаны выше.

Программа ГА Workbench является диалоговым инструментом для демонстрации проведения экспериментов с ГА на IBM совместимых персональных компьютерах. Здесь генерируется начальная популяция, вычисляются и масштабируются значения ЦФ для всех хромосом. Далее выбираются хромосомы для размножения и к выбранным парам применяются ГО. Объединение новой популяции со старой производится путем замещения некоторых хромосом из старой популяции. В GA Work bench реализованы ГО селекции с ожидаемым значением ЦФ и колеса рулетки. Применяется ротация популяции, т.е. замена части хромосом из популяции в течение каждой генерации. Обычно это заданное число. Кроме известных одноточечного и двухточечного ОК, применен однородный ОК [9]. В этом ОК анализируются две хромосомы родителей и производится выбор каждой порции либо от одного родителя, либо от другого с заданной вероятностью. Кроме этого, используются модификации известных ГО, заключающиеся в перемешивании популяции и выборе «жертвы» для замены после генерации ГА.

HYPERGEN – это распределительный ГА для Гиперкуба (ГК) [32]. ГК – это мультипроцессорная система с современной памятью. Считается, что n- мерный ГК состоит из 2-х процессоров, соединенных как n- мерный двоичный куб. Каждый процессор непосредственно связан с n другими смежными процессорами. Коммуникации в ГК выполняются путем передачи сообщений. Предварительно начальная популяция распределяется поровну между процессорами. Каждый процессор выполняет последовательный ГА на своей части популяции. HYPERGEN использует схему воспроизводства, схожую с описанной выше в ПГА. Здесь только несколько членов популяции удаляются

после каждой итерации. Это значит, что цикл воспроизводства (одна генерация) состоит из выхода "лучших" хромосом из популяции, выполняя ОК и ОМ для производства одного потомка. Родители остаются в фонде популяции. Потомки помещаются в фонд популяции в соответствии со значением ЦФ, и хромосомы с худшими значениями ЦФ удаляются. Размер популяции остается постоянным. После заданного числа генераций, хромосомы с лучшей ЦФ в каждом процессоре обмениваются между другими процессорами, вводя новый генетический материал в каждую подпопуляцию. Количество генетического материала (хромосом) для обмена называется степенью (числом) миграции. Этот процесс продолжается до тех пор, пока вся популяция не стабилизируется. HYPERGEN-это модульная конструкция пакета программ для генерации начальной популяции, построение ЦФ, селекции, основанной на функции отбора, репродукции мутации, миграции и обобщенной статистики. Последовательный ГА на каждом процессоре и периодическая миграция хромосом между процессорами выполняются для пользователя автоматически.

Рассмотренные и различные другие модели могут помочь получить более эффективную схему реализации ГА на параллельных процессорах

Эмпирические исследования показывают эффективность применения ГА в задачах, где время решения растет экспоненциально при увеличении размерности задачи [32].

4.5. Выводы

Описанные базовые технологии генетического поиска показывают, что перспективными следует считать комплексные и смешанные генетические алгоритмы. Такие алгоритмы должны

содержать элементы слепого поиска, градиентные методы, точные процедуры, различные эвристики. В настоящее время находят широкое применение гибридные генетические алгоритмы, использующие знания о решаемых задачах. Это позволяет строить точные целевые функции и разрабатывать генетические операторы, ориентированные на структуру решаемых задач, что дает возможность повышать качество решений, избегать предварительной сходимости алгоритмов за приемлемое время.

4.6. Контрольные вопросы

1. Схема многоуровневого ГА.
2. Схема эволюционных операций.
3. Механизмы эволюционного программирования.
4. Варианты планирования процесса ГА.
5. Методы, основанные на знаниях (гибридные ГА, «жадная» эвристика).
6. Гибридные ГА.
7. Модифицированные схемы селекции.
8. Операторы доминирования, инверсия, дупликации, делеции.
9. Частично-соответствующий, упорядоченный, циклические операторы кроссинговера.
10. Операторы сегрегации, транслокации.
11. Прототипы параллельных вычислительных архитектур.
12. Схема сетей.
13. Модель общества.
14. Модель работы заводов.
15. Приведите различные модели гибридных ГА.
16. Приведите модели ГА, использующие различные способы связей.

17. Постройте структуру ГА на основе нейронных сетей.
18. Опишите структуру пакетов ГА.
19. Приведите способы получения реальных решений на основе ГО.

Теория, не проверяемая опытом, при всей красоте концепции теряет вес, не признается: практика, не опирающаяся на взвешенную теорию, оказывается в проигрыше и убытке.

Д. И. Менделеев

5. ИСПОЛЬЗОВАНИЕ ГА ПРИ РЕШЕНИИ ОПТИМИЗАЦИОННЫХ ИНЖЕНЕРНЫХ ЗАДАЧ САПР И ИСКУССТВЕННОГО ИНТЕЛЛЕКТА

В данном разделе будут рассмотрены различные аспекты применения ГА для решения задач искусственного интеллекта, САПР, оптимизационных и других проблем [14-61].

5.1. Архитектура интеллектуальных САПР

В основе интеллектуальных САПР (ИСАПР) лежит понятие искусственного интеллекта (ИИ). Проблемы ИИ тесно связаны с организацией знаний об окружающем мире в виде математических структур, например, множеств, графов, алгоритмов, которые отражают реальные связи и отношения между любыми объектами в природе (в частности в предметной области).

ИСАПР является основной частью компьютерного, интегрированного, интеллектуального проектирования и производства. Под ИСАПР понимают организационно-техническую систему, состоящую из интеллектуального комплекса средств автоматизации проектирования взаимосвязанного и взаимодействующего с

пользователями и сетями ЭВМ, и выполняющую автоматическое проектирование заданных систем.

Интеллектуальный комплекс средств автоматизации должен состоять из совокупности средств и компонентов математического, информационного, программного, технического, лингвистического, методического и организационного обеспечения. Взаимодействие пользователей с ИСАПР производится средствами технического обеспечения. Для повышения уровня "интеллекта" и расширения ее возможностей САПР должна содержать набор сценариев диалога, меню диалога, генераторы случайного и направленного поиска, акселераторы перебора и других подзадач. Пользователь во взаимодействии с ДП на основе блочно-иерархического подхода и принципов декомпозиции может производить разбиение необходимой задачи проектирования систем больших размерностей на совокупности функциональных

рис.5.1. Подсистемы САПР

подзадач меньшей размерности (рис. 5.1;5.2).

Основная цель планировщика-поиск информации. Совместно с экспертной системой (ЭС) он осуществляет анализ преобразованных данных, полученных из ДП. В зависимости от значений внутренних, внешних и управляющих параметров, конструктивных ограничений, заданных режимов работы и т.п. выбирается несколько альтернативных путей решения. Каждому пути соответствует управляющий програм-

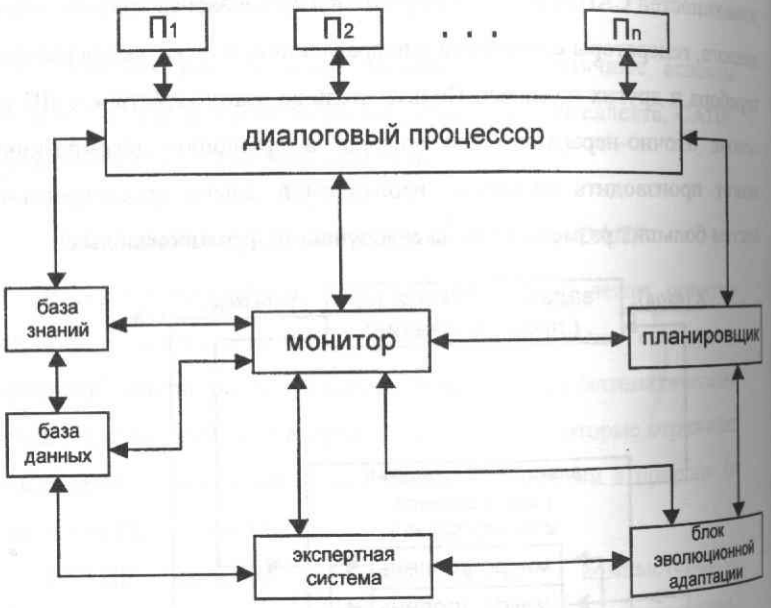


Рис.5.2 Концептуальная схема ИСАПР.

мный блок из монитора. Под управлением этого блока осуществляется выбор, редактирование, настройка и стыковка отдельных программных модулей, определяются типы и форматы всех данных, формируется рабочий комплекс программ. Совместно с базой знаний (БЗ) и блоком

эволюционной адаптации планировщик осуществляет реализацию процедур поиска для нахождения множества решений и для выбора некоторого подмножества эффективных решений среди имеющихся.

Одним из основных интеллектуальных компонентов ИСАПР является БЗ. Понятие "знания" является расплывчатым неформальным понятием. Сейчас под знаниями понимают расплывчатые и четкие множества факторов и сведений, описывающих информацию относительно определенной предметной области позволяющих решать определенный класс задач. Основная цель БЗ получать, хранить, перерабатывать и создавать знания на основе данных и знаний, введенных до начала проектирования и представляющих собой дополнительные знания для пользователя. Обычно в интеллектуальных системах БЗ состоит из трех основных компонентов: правил, вопросов-подсказок, переводов [40]. В ИСАПР БЗ будет состоять из трех основных блоков: база общих знаний, база системных знаний, база прикладных знаний. В первом блоке будут храниться общие знания, необходимые для решения всех задач проектирования. Во втором блоке будут храниться знания о всех внутренних связях самой системы. В третьем блоке будут храниться все прикладные знания, например, описание предметных областей, правила и ограничения на проектирование, комплексы алгоритмов и т.п. В отличие от стандартных баз данных (БД), которые также присутствуют в ИСАПР и взаимодействуют с БЗ, последние позволяют обрабатывать знания и в результате этого получать новые знания. БЗ удобно представлять в виде сети фреймов или графов. Причем каждый фрейм связывает воедино знания о данной ситуации проектирования и предсказывает, какие

объекты будут обрабатываться, какие события могут произойти. В математических моделях ИСАПР фреймы являются совокупностью процедурных и декларативных знаний о моделях проектирования (например, СБИС) и состоят из множества слотов. Слоты содержат знания о конкретной ситуации проектирования СБИС. Кроме того, слоты могут осуществлять вызов прикладных программ из программного обеспечения и т.д. Слоты одного фрейма взаимодействуют со слотами других фреймов, образуя в случае бинарных связей граф, а в случае n-арных связей-гиперграф[40]. Важной задачей является поиск и преобразование информации в таких графовых и гиперграфовых моделях. Можно предложить совместное использование экспертных систем (ЭС) и методов эволюционной адаптации для эффективного решения таких задач. ЭС ИСАПР позволяют разработать структурированную схему, отражающую весь ход проектирования реального объекта. Эта схема дает возможность составить конечное множество вопросов, которые помогут пользователю эффективно произвести весь процесс проектирования.

ЭС ИСАПР - это программно-алгоритмические комплексы, выполняющие разнообразные функции проектирования. Они могут: консультировать пользователя; анализировать промежуточные и конечные результаты проектирования; обучать пользователя и ЭВМ; обучаться в процессе решения конкретной задачи проектирования; давать советы пользователю и ЭВМ; классифицировать проблему проектирования` производить поиск на заданных математических моделях; принимать конкретное решение на любом этапе проектирования; делать достоверные выводы из неполных и расплывчатых данных и знаний; взаимодействовать с другими ЭС;

передавать и приобретать новые знания. Другими словами, ЭС ИСАПР помогают проектировщику накапливать и обрабатывать различные виды глубинных и поверхностных знаний или их представлений в виде соответствующих моделей. Глубинные знания обычно помогают отображать разрабатываемую структуру. К ним относятся причинные модели, категории, абстракции, аналогии. Поверхностные представления знаний - это простые эмпирические ассоциации, которые формируются на основании условных правил типа "Если "X", то "У"". ЭС ИСАПР в основном моделируют и интерпретируют действия проектировщика по организации своих знаний об объекте проектирования и делают из них выводы. В основном ЭС ИСАПР может состоять из трех основных блоков: синтаксический анализатор (интерпретатор), выполняющий грамматический разбор предложений пользователя во время работы; примитивы в обработке знаний; структура языка, обеспечивающая возможность выполнения пошаговой компиляции. В определенном смысле ЭС ИСАПР помогают перенести опыт пользователя в ЭВМ для принятия эффективных решений.

Общий цикл проектирования СБИС на основе ИСАПР включает следующие этапы [37, 41, 43]:

- системная спецификация;
- функциональное проектирование;
- логическое проектирование;
- проектирование цепей;
- физическое проектирование;
- изготовление;
- упаковка и тестирование.

Цель первого-этапа представление системы на верхнем уровне. Он представляет реализацию компромисса между требованиями рынка, технологий изготовления и экономическими возможностями. Выходными данными здесь является размер, скорость, мощность, и функциональные возможности системы на СБИС.

Основная цель второго этапа - определить функциональные требования для всех частей системы. Выходом являются временные диаграммы и структурные связи между всеми блоками системы.

На третьем этапе синтезируются и минимизируются булевы функции, проводится описание системы на языке типа VHDL, выполняется моделирование и верификация для проверки правильности функционирования системы.

Цель четвертого этапа - преобразовать булевы выражения в схематические представления, учитывающие временные и электрические характеристики системы. Выходом обычно являются детальные диаграммы цепей.

На пятом этапе схемное представление каждого блока системы конвертируется в его геометрическое представление. Выходом является детальная топология, зависящая от правил и ограничений проектирования, которые основаны на технологических ограничениях, электрических свойствах и применяемых материалах. Этот этап считается одним из важнейших. В течение этого этапа выполняются различные верификации и проверки правильности полученной топологии.

На шестом этапе топология СБИС превращается в фотолитографические маски. Здесь проводятся различного рода тестирования.

На седьмом этапе производится упаковка блоков и тестирование по выполнению всех правил проектирования и функциональных свойств.

Основные этапы физического (конструкторского) проектирования, согласно [37, 56], следующие:

- разбиение;
- планирование кристаллов;
- размещение;
- трассировка;
- сжатие;
- экстракция и верификация.

На первом этапе кристалл, содержащий ориентировочно миллион транзисторов, разбивают на блоки с минимизацией количества связей между ними. Обычно этот процесс выполняется иерархически: верхний уровень содержит 5-25 блоков. Каждый блок затем разбивается рекурсивно на более мелкие блоки.

На втором этапе производится планирование прямоугольных блоков (со связями или без) на плоскости с минимизацией занимаемой площади и выполнением конструктивно-технологических ограничений и требований.

На третьем этапе блоки точно размещаются на кристалле. Основная цель размещения - это найти минимум площади для каждого блока и минимум суммарной длины соединений между ними с выполнением последующих ограничений. Кроме того, цель размещения - это будущая трассируемость и выполнение временных требований по задержке сигнала. Обычно размещение проводят в два этапа :

- конструктивно-начальное (последовательное) размещение,
- итерационное (коллективное) улучшающее размещение.

О качестве трассировки можно судить только после трассировки соединений. Поэтому современные алгоритмы реализуют эвристики совместного размещения и трассировки.

Цель четвертого этапа - проложить соединения на плоскости между блоками согласно заданному списку цепей схем. Плоскость разбивается на каналы и коммутаторы (switchboxes). Цель трассировки - провести все соединения с минимумом длинны изгибов, переходов между слоями используя только каналы и switchboxes (SB).

Трассировку делят на два этапа :

- глобальная,
- детальная.

Глобальная трассировка определяет различные районы в трассировочном пространстве, где будут проходить все соединения. Остальная трассировка выполняется для каждого канала и SB и определяет координаты прохождения каждой трассы для всех цепей схемы.

На пятом этапе выполняется сжатие топологии кристалла во всех направлениях, чтобы уменьшить занимаемую площадь.

На шестом этапе решаются задачи проверки соблюдения всех правил проектирования на функциональном и физическом уровнях.

Развитие компьютерных сетей привело к быстрому изменению самого способа ведения конструкторских разработок, превратив

системы автоматизированного проектирования (САПР или CAD) в своеобразное окно, через которое осуществляется доступ к обширным взаимосвязанным программным, информационным и людским ресурсам. ПО CAD-«броузер», с помощью которого тот или иной проект связывается с этими ресурсами. С точки зрения конструкторской разработки применение сетей сулит интеграцию усилий в рамках предприятия на протяжении всего жизненного цикла изделия.

Сеть Internet вызвала к жизни множество новых инструментов, предназначенных для сотрудничества и совместного использования данных в глобальном масштабе. Передовые компании с их помощью смогут предоставить необходимую информацию и инструменты в нужное время и именно тем, кому это требуется, независимо от того, где они находятся.

Имеются две технологии, поддерживающие доступ к данным сетевых CAD-систем. Одна из них – это CAD-броузеры, другая базируется на типах данных, приспособленных для передачи в Internet.

Броузер для просмотра CAD-информации является ключевым элементом сетевых систем автоматизированного проектирования. Многие инженерные задачи требуют наличия средств быстрого доступа к текущим проектным данным как у потребителей и конкурентов, так и у издателей соответствующей информации. Интерфейсы CAD-броузеров должны удовлетворять потребности самого широкого круга пользователей, а не только профессиональных проектировщиков. Концепция таких броузеров была бы более предпочтительна, если бы все CAD-данные находились в некоем стандартном формате,

обеспечивающем простые интерфейсы с другими службами и системами [44].

CAD-броузер выполняет следующие функции:

- просматривает печать, а также передачу запросов к CAD-моделям, построенным в самых разных патентованных и нейтральных форматах;
- проводит минимальный набор функций трансляции файлов, например для сохранения CAD в виде файлов VRML или Computer Graphics Metafile (CGM).
- имеет низкую цену для завоевания рынка;
- поддерживает стандарты и протоколы Internet;
- обеспечивает фильтрацию данных. Модели CAD могут иметь огромные объемы информации, однако большинству пользователей потребуются только выборочные представления таких данных;
- предусматривает переносимость между разнородными платформами для выполнения различных проектных процедур. Это могут быть рабочие Unix-станции, различные среды Windows, сетевые компьютеры и даже ПК с процессором 486 под управлением DOS.

Некоторые современные интегрируемые модули броузера Netcape Navigator и универсальные средства просмотра существенно приблизились к идеалу CAD-броузера. Интегрируемый модуль Whip корпорации Autodesk дает возможность броузерам просматривать данные CAD и манипулировать ими.

Возможность согласования CAD-данных с Internet зависит от того, можно ли вставить в них гипертекстовые ссылки и связи, а также

создавать ассоциации между геометрическими или топологическими элементами и внешними ресурсами. Интеграция даже простейшего механизма гиперссылок с геометрией и топологией CAD позволит пользователям анализировать более информативные структуры, содержащие, например, сведения о намерениях конструктора или коллектива разработчиков либо соображения, в соответствии с которыми принималось то или иное проектное решение. Одна только интеграция в настольную CAD-систему функций перемещения по текстовым гиперссылкам могла бы стать шагом вперед по сравнению с сегодняшними узконаправленными, ориентированными лишь на геометрию решениями.

Последние достижения в создании компонентного ПО твердотельного моделирования для систем CAD представляют процессы твердотельного моделирования и информационные шины для построения систем, способных манипулировать трехмерными объектами.

Среды разработки, интегрированные в CAD-системы, обычно обеспечивают более широкий доступ к функциям последних, нежели автономные инструменты. В некоторых случаях эти среды представляют собой расширенные макроязыки, позволяющие создавать программы, которые управляют интерфейсом пользователя CAD-системы. Так, ПО SDRC поддерживает два типа сред программирования: один - это форма макроязыка для выполнения команд пользовательского интерфейса в рамках модели клиент/сервер (вы пишите клиентское ПО, а I-DEAS CAD выступает в роли сервера); второй - полноценная среда разработки с ограниченным доступом к API-интерфейсу I-DEAS. Некоторые наиболее мощные среды

включают средства для извлечения данных из CAD-систем и не столь развитые - для программирования сложных геометрических операций. Это делается для защиты целостности данных: контролируя операции, которые пользователю разрешается делать с помощью среды разработки, можно снизить вероятность того, что какая-нибудь программа разрушит внутренние данные.

Технологии Internet, особенно Java и COBRA, играют важную роль в интеграции старых технологий в современные интра-сети. Язык Java может быть использован для построения внешних интерфейсов к запатентованным системам на базе мейнфреймов, а протокол COBRA способен «обернуть» объектную оболочку, унаследованный код и исполняемые модули [44].

Развертывание сервисов автоматизированного производства в Internet имеет много потенциальных достоинств, не последним из которых является обеспечение конструкторам (независимо от их местоположения) доступа к службам макетирования, специализированным процессам и быстрое достижение результата.

Большинство проектировщиков технических устройств тратят уйму времени на поиск той или иной информации. Поиск может производиться в личных записных книжках, документах компании или печатных каталогах поставщиков готовых узлов и деталей. С учетом этого не удивительно, что именно он-лайновые каталоги и базы данных - это те направления, на которые технологии Internet оказывают заметное влияние в повседневной деятельности инженеров. Оперативная информация приобретает формы производственных и программных сервисов, справочников о продукции и каталогов с поисковыми средствами.

Концепция интеллектуальных агентов была предложена для того, чтобы помочь людям искать необходимую им информацию. Интеграция инструментария агентов с огромным наборов гетерогенно-структурированных он-лайновых инженерных ресурсов - исследовательская задача и для направления искусственного интеллекта, и для инженерных дисциплин. Разработка обобщенной архитектуры агентов для поддержки основных технических приложений поможет небольшим фирмам создавать специализированных агентов, которые могут встраиваться в гетерогенные CAD-системы [44].

Информация о технологическом процессе сопровождает проектные данные на всех этапах жизненного цикла продукта. По мере того, как конструкторы, поставщики, контрагенты и производители становятся все более связанными друг с другом, наличие моделей технологических процессов, пригодных для совместного использования, станет важным для реализации их взаимодействия.

Разработка агентов для проектирования ставит множество нетривиальных проблем перед специалистами в области искусственного интеллекта, представления инженерных знаний, автоматизированного проектирования и сетевых вычислений.

Необходимым условием эффективной работы ИСАПР является ее автоматическая или автоматизированная адаптивная настройка на объект проектирования и все виды САПР. Рассмотрим блок эволюционной адаптации (БЭА), который основан на стандартных моделях наследственности и эволюции из области популяционной генетики, являющихся воплощением тех механизмов адаптации, которые имеют место в различных видах живой природы [1-36].

Эволюция в живых и искусственных системах может осуществляться в результате взаимодействия трех основных факторов, а именно: изменчивости, наследственности и естественного отбора. Формы отбора тесно связаны с процессом селекции. Для эффективного выполнения селекции необходимо выполнение четырех основных правил. Во-первых, это правильный выбор индивидуальностей, во-вторых, это четкая постановка задачи селекции, в-третьих, это проведение селекции на большом числе популяций, в-четвертых, это проведение отбора только по одному свойству. Заметим, что на самом деле таких факторов значительно больше. Все более полное их изучение позволяет строить более адекватные модели живой природы и искусственной системы. Изменчивость обычно является основой для образования новых видов и признаков в строении и функциях так называемых органических структур (ОС).Наследственность закрепляет эти признаки. Естественный отбор позволяет отобрать лучшие ОС и устранить ОС, неприспособленные к условиям внешней окружающей среды. Методы эволюционной адаптации используются для повышения качества и эффективности конструкторских САПР.

Отметим, что решение основных задач конструкторского проектирования СБИС осуществляется последовательными, итерационными, точными или комбинированными алгоритмами [37, 43, 56]. В основном эти алгоритмы выполняют совершенствование начального варианта решения, полученного одним из конструктивных алгоритмов. Такой подход, кроме известных преимуществ, имеет недостаткам:

- все алгоритмы, кроме точных, часто останавливаются на локальных решениях, которые могут быть далеки от

оптимальных, и поэтому затруднительно осуществлять выход из локальных "ям";
- точные алгоритмы могут работать только с очень ограниченным количеством элементов, что для структуры СБИС является недостаточным;
- в качестве исходного используется лишь один вариант решения.

При применении БЭА в ИСАПР алгоритмы проектирования будут использовать в качестве начального не одно, а несколько альтернативных решений. Причем в зависимости от сложности перерабатываемой информации исходные решения могут получаться на основе применения стохастических, детерминированных или комбинированных алгоритмов. Далее полученные решения обрабатываются адаптированными к решаемой задачи конструкторского проектирования генетическими алгоритмами поиска. Тогда, применительно к проблеме конструкторского проектирования СБИС, базисную структуру генетических алгоритмов можно представить в виде рис.5.3. Согласно данной схеме на первом этапе случайным, направленным или комбинированным методом получают некоторое подмножество решений рассматриваемой задачи САПР. Эти решения образуют текущую генерацию или популяцию исследуемых решений. Далее вводится или вычисляется ЦФ. Вычисление ЦФ является очень сложной задачей, причем от точности ЦФ зависит качество будущих решений.

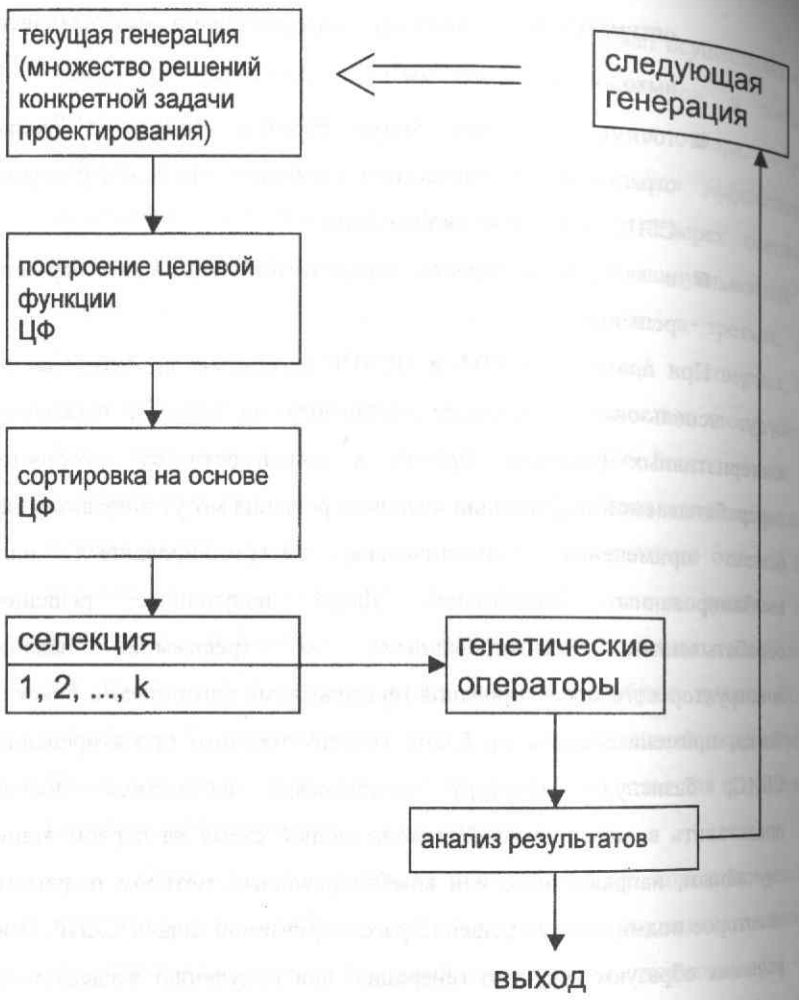


Рис 5.3. Структура БЭА применительно к задачам конструкторского проектирования

Для каждой задачи конструкторского проектирования будет строиться своя ЦФ. При построении ЦФ используют знания проектировщика о конкретной задаче и элементы ЭС. На основе ЦФ

производятся ранжирование и сортировка популяции решений (Р) .Затем в результате методов селекции подбираются пары для применения различных генетических операторов. После применения ОК, ОМ,ОИ, ОТ и т.д. получается новое подмножество решений. Оно объединяется с первоначальным подмножеством решений. Получается новое множество $A_{га} = A \cup A'$. Используя ЦФ, производится анализ $A_{га}$.Все элементы в $A_{га}$ (решения задачи), значения ЦФ которых хуже заданного порога, являются с нашей точки зрения неперспективными решениями и удаляются из $A_{га}$.Получается новое множество $A'_{га}$.Причем $|A'_{га}| = |A|$. Если данное условие не выполняется, например $|A'_{га}| < |A|$, то в $A'_{га}$ включается элемент с лучшими характеристиками из отброшенных. Множество $A'_{га}$ объявляется новой текущей популяцией решений, и далее процесс может повторяться итерационно до получения подмножества или одного оптимального решения.

Заметим, что такая стратегия позволяет быстрее находить локально-оптимальные результаты. Это связано с параллельной обработкой множества альтернативных решений. Причем в такой схеме возможно концентрировать поиск на наиболее перспективных решениях [19, 32]. Отметим, что периодически в каждой итерации можно проводить различные изменения в перспективных, не перспективных и, вообще, в любых решениях. Временная сложность таких алгоритмов в основном совпадает со сложностью быстрых итерационных алгоритмов и лежит в пределах $O(\alpha_i n)$-$O(\beta_j n^3)$, где α_i, β_j-коэффициенты, n-число входов алгоритмов. Такая сложность обещает перспективность использования ГА в ИСАПР. Отметим, что в описанной структуре простых генетических алгоритмов существует

один трудно устранимый недостаток, заключающийся в предварительной сходимости алгоритмов в локальном оптимуме, далеком от глобального [9, 10]. Для устранения этого недостатка используется большое количество модифицированных генетических операторов и схем селекции. Часть из них описана выше, другая будет описана в конкретных алгоритмах проектирования. В общем случае использование таких подходов не всегда дает выходы из локальных оптимумов.

В последнее время появились новые «нестандартные» архитектуры генетического поиска, позволяющие в большинстве случаев решить проблему предварительной сходимости алгоритмов. Это методы миграции и искусственной селекции [28], метагенетической параметрической оптимизации [29], стохастически-итерационные генетические и поисковые [22, 23], «прерывистого» равновесия [16, 49], объединения генетического поиска и моделирования отжига [11, 26, 46].

Процесс метагенетической оптимизации заключается в следующем (рис.5.4). Основным является первый блок, в котором осуществляется реализация генетического алгоритма, генерация новых решений, определение целевой функции и использование предыдущих решений для генерации лучших результатов. Второй блок позволяет использовать «историю» предыдущих решений для генерации лучшего множества параметров. В третьем блоке генерируется новое множество оптимизационных параметров. Используя метагенетический оптимизационный процесс в САПР, можно случайным, направленным или случайно-направленным способом генерировать начальные популяции, моделировать каждую индивидуальность посредством

выполнения ГА на основе реализации генетических операторов. Можно случайно выбирать родителей из популяции с вероятностью селекции каждого элемента пропорционально его значению. Причем вероятность выполнения каждого оператора может определяться пропорционально его целевой функции. Окончательное множество параметров селектируется после моделирования из конечной популяции. Отметим, что для каждой задачи проектирования СБИС лучше всего строить свой конкретный метагенетический алгоритм.

Рис. 5.4. Метагенетический оптимизационный процесс в САПР

Стохастически-итерационный метод заключается в следующем. На основе генетического поиска определяются стартовые точки для направленного поиска. Причем направленный поиск

осуществляется совместно с генетическими операторами. После нахождения стартовых точек можно параллельно использовать такие методы оптимизации, как "золотое сечение", градиентный спуск, поиск в глубину и ширину, ветви и границы и др. [43].

Метод прерывистого равновесия [16, 49] основан на палеонтологической теории прерывистого равновесия, которая описывает быструю эволюцию за счет вулканических и других изменений земной коры. Для применения данного метода в технических задачах в [16] предлагается после каждой генерации случайным образом "перемешивать" индивидуальности в популяции, а затем формировать новые текущие генерации. Здесь можно предложить, как аналог из живой природы, бессознательный отбор родительских пар и направленный отбор "лучших" родительских пар. Далее случайным образом смешать результаты обоих отборов и не оставлять размер популяции постоянным, а управлять им в зависимости от наличия "лучших" хромосом. Такая модификация метода прерывистого равновесия может позволить сократить неперспективные популяции и расширить популяции, в которых находятся "лучшие" индивидуальности. Согласно [49], метод прерывистого равновесия - это мощный стрессовый метод изменения окружающей среды, который используется для эффективного выхода из локальных "ям".

Объединение ГА и метода моделирования отжига позволяет получать качественные результаты за счет усложнения процедуры оптимизации [26]. Например, на основе простого ГА можно получить некоторое подмножество родителей с лучшими характеристиками и для

одного из них (наилучшего) или некоторого подмножества применить оптимизационную процедуру моделирования отжига. Такое объединение можно делать различными способами. К сожалению, процедуры моделирования отжига требуют больших вычислительных затрат. Поэтому, такие подходы применяют при проектировании элементов топологии внутри ячеек, когда их число ≤ 50.

5.2. Разработка генетических алгоритмов компоновки

Согласно приведенной структуре решения задач конструкторского проектирования элементов СБИС задача компоновки является первой и одной из важнейших. Отметим, что задача компоновки относится к классу NP-полных проблем, т.е. не существует эффективных алгоритмов ее решения с полиномиальной временной сложностью. В [43] дана классификация существующих методов решения задач компоновки. При проектировании устройств на СБИС наибольший интерес представляют последовательные (конструктивно-начальные) и итерационные эвристические методы. Временная сложность таких методов лежит в интервале $O(\alpha n)$-$O(\beta n^3)$, где α, β-коэффициенты, n-число проектируемых элементов.

Представим коммутационную схему СБИС в виде гиперграфовой модели $H = (X, E, W)$, где X представляет множество вершин гиперграфа, E - множество гиперребер, а W- общий суммарный вес вершин. Каждой вершине $x_i \in X$ соответствует элемент СБИС и вес $w_i \in W$. Вес вершины соответствует интегральной оценке, в которую входят площадь, занимаемая элементом СБИС, а также различные конструкторско-технологические ограничения на проектирование.

Причем, значения $w_i \in W$ не превышают некоторой пороговой величины. Пусть $B = \{B_1, B_2, \ldots, B_s\}$ будет множеством разбиений гиперграфа H на части $B_1, B_2, \ldots, B_s$; такие, что $B_1 \cap B_2 \cap \ldots \cap B_s = 0$, и $B_1 \cup B_2 \cup \ldots \cup B = B$. Пусть каждое разбиение B_i состоит из элементов $B_i = \{b_1, b_2, \ldots, b_n\}$, $n = |X|$. Тогда задача компоновки коммутационной схемы или разбиение гиперграфа H на части заключается в получении разбиения $B_i \in B$, удовлетворяющего трем условиям и ограничениям:

$(\forall B_i \in B)\ (B \neq \varnothing)$,

$(\forall B_i, B_j \in B)\ ([B_i \neq B_j \rightarrow X_i \cap X_j = \varnothing] \wedge [(E_i \cap E_j = E_{ij}) \vee (E_i \cap E_j = \varnothing)])$, (5.1)

$$\bigcup_{i=1}^{s} B_i = B, \bigcup_{i=1}^{l} E_i = E, \bigcup_{i=1}^{n} X_i = X, \left|E_{ij}\right| = K_{ij}.$$

Весовая функция для разбиения гиперграфа H запишется так:

$$K = \frac{1}{2}\sum_{i=1}^{n}\sum_{j=1}^{n} K_{ij}, (i \neq j), \qquad (5.2)$$

где K_{ij} - число связей между частями H_i и H_j при разбиении гиперграфа H на две части; l - количество частей в разбиении; K - суммарное количество связей при разбиении гиперграфа на части.

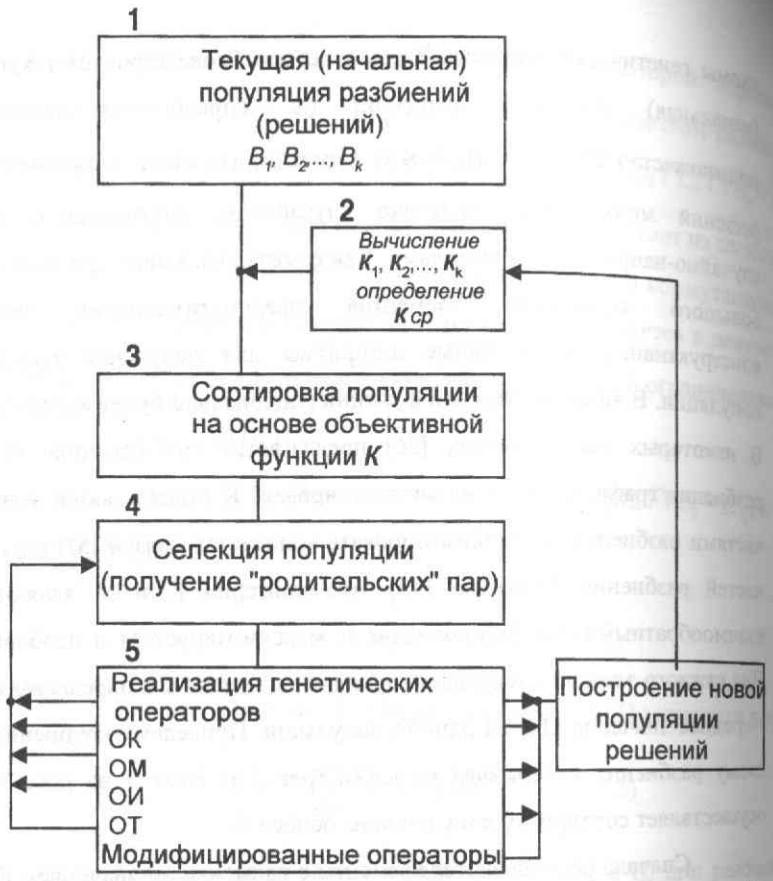
Стандартная задача разбиения заключается в минимизации K (K=>min). Минимизация K при разбиении коммутационных схем на части позволяет косвенно учитывать практически все критерии и конструкторско-технологические ограничения СБИС [37, 43, 56].

Построим схему разбиения гиперграфа на части на основе генетического подхода. При этом будем учитывать приведенные выше

схемы генетического поиска. Здесь в первом блоке строится текущая (начальная) популяция решений, т.е. определяется заданное подмножество $B_1, B_2, \ldots, B_k$ ($k \leq s$). Отметим, что такое подмножество решений может быть получено случайным, направленным или случайно-направленным методами. Как отмечалось выше, при наличии большого количества элементов предпочтительными будут конструктивные и случайные алгоритмы для получения текущей популяции. В качестве целевой функции рационально будет выбрать K. В некоторых исследованиях [56] предлагается при решении задач разбиения графа на части не минимизировать K (число связей между частями разбиения), а максимизировать количество связей (M) внутри частей разбиения. Известно [43], что критерии K и M являются взаимообратными, т.е. минимизация K максимизирует M и наоборот. Для каждого элемента текущей популяции вычислим K и определим $K_{ср}$ - среднее значение ЦФ на данной популяции. Приведем укрупненную схему разбиения гиперграфа на части (рис 5.5) Блок 3 на рис. 5.5. осуществляет сортировку популяции на основе K.

Сначала расставляются элементы с наименьшим значением K и т.д. по возрастанию. Блок 4 выполняет селекцию популяции для получения родительских пар. Селекция выполняется одним из методов, описанных ранее. Простейшим случаем селекции может являться последовательный метод выбора наилучших (в смысле функции K) индивидуальностей после сортировки. Блок 5 осуществляет реализацию генетических операторов. В шестом блоке осуществляется построение новой популяции решений. Далее для каждой индивидуальности из

Рис.5.5. Укрупненная схема разбиения с применением ГА



новой популяции вычисляется К и выживают те элементы из старой и новой популяции, у которых $K \geq K_{ср}$, так чтобы количество элементов в новой популяции не превышало числа элементов в старой популяции. Можно предложить большое число аналогичных схем генетического поиска. Их эффективность должна проверяться экспериментальным путем. Рассмотрим последовательный генетический алгоритм разбиения. Пусть задан гиперграф схемы H = (X,E). Опишем основную идею последовательного алгоритма разбиения гиперграфа H с

минимизацией К на основе методов генетического поиска [14, 16, 19]. Первоначально упорядочим все вершины гиперграфа по возрастанию локальных степеней вершин. Это соответствует списку (первому) вершин, а также может соответствовать тривиальному разбиению, когда количество групп разбиения равно упорядоченным вершинам гиперграфа. Далее начинаем составлять пары вершин и для каждой пары вычислять оценку связности

$$\delta_{ij} = e_{ij}-(e_{it} + e_{jt}). \qquad (5.3)$$

В (5.3.) e_{ij} - число связей между вершинами x_i и x_j, образовавшими пару; $e_{i,t}$ и $e_{j,t}$ - это число связей, соединяющих выбранную пару со всеми остальными вершинами модели, причем t = 1, 2,..., n-2. Во втором списке будут оставлены такие пары вершин, у которых $\delta_{i,j} \geq 0$. Отметим, что если пар $\delta_{i,j} \geq 0$ не находится, то возможно два случая. В первом - будут образовываться пары с наименее отрицательным $\delta_{i,j}$. Во втором случае пары из двух вершин не образуются и сразу происходит переход к образованию групп из трех вершин. Далее процесс повторяется аналогично, пока не будет выполнено разбиение гиперграфа на заданное или произвольное число частей. Это основная идея стандартной процедуры разбиения. При небольшом количестве вершин она дает удовлетворительные результаты, т.к. практически здесь осуществляется полный перебор вариантов решений. С увеличением числа вершин напрямую использование такого подхода становится нецелесообразно. Для сокращения числа просматриваемых пар применим простые генетические операторы. Отметим, что вместо выражения (5.3) можно использовать выражение

$$\tau_i + \tau_j \geq \tau_t, \qquad (5.4)$$

где $\tau_i + \tau_j$-внутренние ребра разбиения, τ_t (t = 1, 2,..., n-2)-внешние ребра разбиения.

Рассмотрим пример. Для простоты схему зададим в виде графа (рис.5.6). Определим локальные степени вершин и упорядочим их в виде списка по возрастанию (табл.5.1). Построим теперь пары вершин для определения разбиения: графа по две в каждой. На этом уровне выделим два блока разбиения: это вершины (1,3), (2,6). Если стоит задача разбить граф на части по две вершины в каждой с минимизацией К, то в качестве третьего блока разбиения можно взять блок с наименее отрицательным значением выражения (5.4). Такое разбиение показано на рис.5.7. В табл. 5.2 приведены все пары, которые можно получить, объединяя все вершины заданной модели. Очевидно, что для большого числа вершин данная процедура нецелесообразна. Применим схему генетического поиска, описанную выше.

Построим случайным образом некоторую популяцию решений:

$P = \{P_1, P_2, P_3, P_4, P_5, P_6\}$. $P_1 = (1,2)$ Теперь проверим выполнение условия (5.4) для всех элементов популяции P. Для первого элемента-1≥8 (нет). Продолжая далее, получим: $P_2 = (2,3)$ 0≥11, $P_3 = (3,4)$ 1≥8, $P_4 = (4,5)$ 3≥4, $P_5 = (5,6)$ 1≥8, $P_6 = (3,6)$ 0≥10.

В качестве целевой функции возьмем выражение (5.4). Определим для каждой хромосомы в популяции значение ЦФ. Произведем сортировку анализируемой популяции. В результате получим: P_4, P_5, P_1, P_3, P_6, P_2. Произведем селекцию и образуем родительские пары.

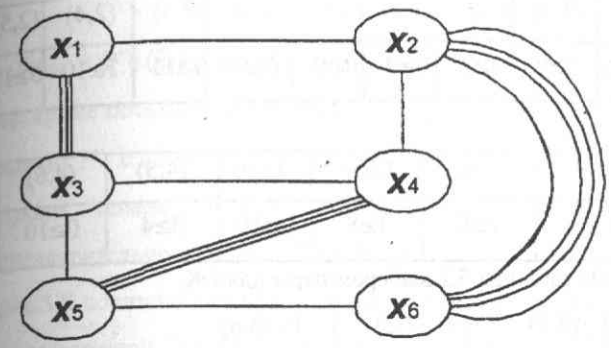


Рис. 5.6. Граф G

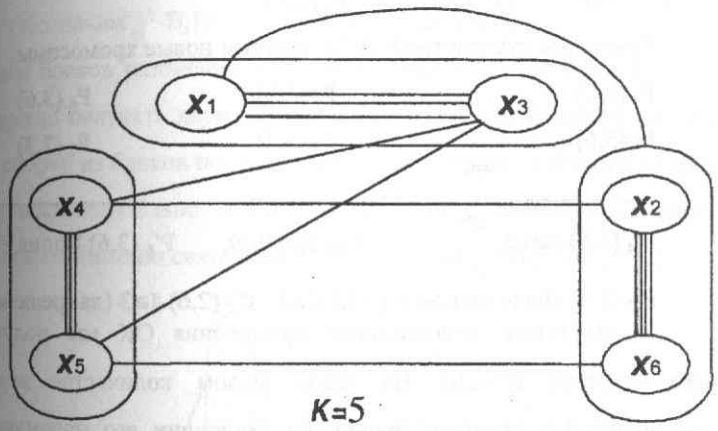


Рис. 5.7. Разбиение G на три части.

Таблица 5.1

x_i	x_2	x_3	x_4	x_5	x_6	x_1
ρ_{xi}	6	5	5	5	5	4

Таблица 5.2

(xi xj)	(1,2)	(1,3)	(1,4)	(1,5)	(1,6)	(2,3)	(2,4)	(2,5)	(2,6)
$\tau_i+\tau_j \geq \tau_t$	1≥8	3≥3	0≥9	0≥9	0≥90	0≥11	1≥10	0≥11	4≥3

$(x_i\ x_j)$	(3,4)	(3,5)	(3,6)	(4,5)	(4,6)	(5,6)
$\tau_i+\tau_j \geq \tau_t$	1≥8	1≥8	0≥10	3≥4	0≥10	1≥8

Из таблицы 5.2 выберем пары для ОК.

$$\begin{cases} P_4\ (4,5) \\ P_5\ (5,6) \end{cases} \quad \begin{cases} P_1\ (1,2) \\ P_3\ (3,4) \end{cases} \quad \begin{cases} P_6\ (3,6) \\ P_2\ (2,3). \end{cases}$$

Применим стандартный ОК и получим новые хромосомы.

P_4 (4,5)	P_1 (1,2)	P_6 (3,6)
P_5 (5,6)	P_3 (3,4)	P_2 (2,3)
P'_4 (4,6) 0≥10,	P'_1 (1,4) 0≥9,	P'_6 (3,6) Копия P6,

P'_5 (5,5) Нелегальное P'_3 (3,2)0≥1 P'_2 (2,6) 4≥3 (да) решение

В результате однократного применения ОК мы получили лучшее решение P'_2(2,6). На таком малом количестве вершин преимущество ГА показать трудно, но тенденции его преимуществ видны даже на простом примере. Следует сказать, что использование стандартного ОК может приводить к несуществующим решениям (разбиение P'_5). В этом случае можно предложить несколько модификаций ОК [15, 16, 32]. В частности, при скрещивании пар потомки могут получаться путем объединения первых элементов родителей (ОК) и вторых элементов родителей (ОК). Тогда в нашем примере получим:

$$\begin{cases} P''_4\ (4,5), \\ \\ \end{cases} \quad \begin{cases} P''_1\ (1,3), \\ \\ \end{cases} \quad \begin{cases} P''_6\ (3,2), \\ \\ \end{cases}$$

P''_5 (5,6), P''_3 (2,4), P''_2 (6,3),

Здесь, кроме потомков, повторяющих родителей (P''_4, P''_5, P''_2), находится элемент P''_1 с одним из лучших значений целевой функции.

Построим теперь группы разбиения по три вершины. Обозначим предварительно <1,3>-А; <2,6>-В; <4,5>-С.

На рис.5.8 показано разбиение графа G (рис.5.6) на две части, по три вершины в каждой. В табл. 5.3. приведены сведения о связности групп по 3 вершины в каждой.

Обозначим <В,1>-Д; <С,3>-Е. Построим табл. 5.4 для выделения блоков разбиения по четыре вершины. Заметим, что такие блоки можно получать двумя основными способами. В первом из них делают сборки из блоков по две вершины, во втором - к блокам из трех вершин последовательно добавляют по одной вершине. В итоге получили максимально связанные блоки из четырех вершин

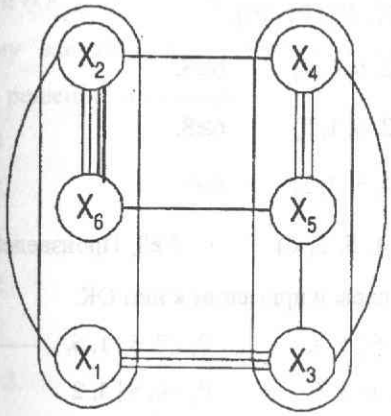


Рис. 5.8. Разбиение G на две части

Таблица 5.3

$((x_i\ x_j)x_m)$	(A,2)	(A,4)	(A,5)	(A,6)	(B,1)	(B,3)	(B,4)
$\tau_{ij}+\tau_m \geq \tau_t$	4≥7	4≥6	4≥6	3≥8	5≥5	4≥8	5≥6

$((x_i\ x_j)x_m)$	(C,1)	(C,2)	(C,3)	(C,6)
$\tau_{ij}+\tau_m \geq \tau_t$	3≥8	4≥8	5≥5	4≥6

Таблица 5.4

$((x_i\ x_j\ x_m)x_p)$	(D, 3)	(D, 4)	(D, 5)	(E, 1)	(E, 2)	(E, 6)
$\tau_{ijm}+\tau_p \geq \tau_t$	8 ≥ 4	6 ≥ 8	6 ≥ 8	8 ≥ 3	6 ≥ 9	6 ≥ 8

Блок 1-<2, 6, 1, 3>,

Блок 2-<4, 5, 3, 1>.

Покажем применение ГА для получения эффективных результатов.

Пусть популяция решений состоит из заведомо "плохих" решений.

$P = \{P_1, P_2, P_3, P_4\}$,

$P_1 = (2, 6, 1, 4)$ 6≥8,

$P_2 = (2, 6, 1, 5)$ 6≥8,

$P_3 = (4, 5, 3, 2)$ 6≥9,

$P_3 = (4, 5, 3, 6)$ 6≥8, Произведем сортировку, выберем родительские пары и применим к ним ОК.

$P_1 : 2, 6 \mid 1, 4,$ $P_2 : 2, 6 \mid 1, 5,$

$P_4 : 4, 5 \mid 3, 6$ $P_3 : 4, 5 \mid 3, 2$

$P'_1 : 2, 6 \mid 3, 6$ $P'_2 : 2, 6 \mid 3, 2$

$P'_4 : 4, 5 \mid 1, 4$ $P'_3 : 4, 5 \mid 1, 5.$

Точка ОК была выбрана случайно. При этом все потомки получились «нелегальными». Поэтому для задач компоновки обычно разрабатывается свой модифицированный ОК, часто основанный на предварительных знаниях о задаче [14, 15, 16, 19, 36]. Покажем на примере работу предложенного ОК разбиения (ОКР).

Пусть P_1: 1, 2| 6, 4, 5, 3

P_2: 5 |6, 2, 1 |4, 3.

В обоих родителях случайно находятся точки скрещивания. Причем количество точек скрещивания в обоих родителях может быть различным. Фрагмент из второго родителя между точками ОК копируется в точку ОК первого родителя. При этом получим

P'_1: 1, 2, 6, 2, 1, 6, 4, 5, 3.

После удаления повторяющихся элементов из P'_1 получим

P'_1: 6, 2, 1, 4, 5, 3.

Подходящим оператором для задачи компоновки является оператор сегрегации (ОС), описанный выше. Так, выбирая случайным образом по одному элементу из родителей в популяции, можно получить легальное решение. Например:

P_1: 2, 6, <u>1</u>, 4

P_4: <u>4</u>, 5, 3, 2

P_2: 2, 6, 1, <u>5</u>

P_3: 4, 5, <u>3</u>, 2

P'_1: 1, 4, 5 , 3.

Целевая функция для P'_1 является наилучшей с точки зрения рассматриваемого критерия. Продолжая аналогично выделение

максимально связных блоков, получим шесть максимально связных блоков (рис.5.9).

В работе [42] разбиение СБИС структур реализовано как разбиение множества вершин X графа G на подмножества $\Pi_1, \Pi_2, \ldots \Pi_n$, таких что: все элементы $\Pi i \in \Pi = \{\Pi_1, \Pi_2, \ldots \Pi_n\}$ удовлетворяют заданным ограничениям, целевая функция $F:\Pi \rightarrow R$ оптимальна для Π_i.

ГА начинает работу с начальной популяцией хромосом (разбиений). В каждой генерации потомки хромосом получаются на основе ОК. Процесс мутации происходит путем обмена генов в выбранных хромосомах в популяции в каждой генерации. В конце работы алгоритма лучшие разбиения выбираются как решение данной проблемы. Строительными блоками ГА являются наборы генов. Шаблоны или схемы – это частично полученные решения, например, $\Pi_1=x_1, x_2, x_i, x_j, *, *, *$-это означает, что в первое разбиение попали обязательно элементы x_1, x_2, x_i, x_j, а остальные три элемента могут быть любыми. Для эффективности генетического поиска начальная популяция должна представлять решения из различных частей поискового пространства. В [42] ОК и ОМ ориентированы на лучшую ЦФ. Селекция основана на «выживании сильнейших» во всех этапах генерации и сокращении числа генерации.

Основные характеристики решений задачи разбиения следующие:

- Каждое решение – это хромосома, состоящая из генов
- В маленькой популяции обеспечивается большое разнообразие

- Обеспечивается простота применения операторов кроссинговера и мутации
- Обеспечено простое вычисление ЦФ.

Каждый ген – это закодированная величина номера вершины графа, попавшая в рассматриваемое разбиение. Начальная популяция решений конструируется случайным образом. Размер каждого решения (хромосомы, подмножества разбиения) включается случайно для обеспечения разнообразия в популяции. Селекция популяции, выполняемая с ВСА $\approx 0\ (n^2)$, где $n = |X|$, оставляет только «легальные» (реальные) разбиения. Технология ОК следующая. Он получает два кандидата – решения Π_α и Π_β. В каждом кандидате случайным образом определяется точка ОК. Затем первая часть генов из Π_α до точки ОК перемещается в Π_γ-потомок – решение. Вторая часть генов (которая не вошла в Π_γ) из Π_β от точки ОК перемещается в Π_γ. Третья часть генов из Π_α и Π_β последовательно перемещается в Π_γ, чтобы получить реальное решение. Например:

$\Pi_\alpha : x_1, x_3, x_5, | x_6, x_2, x_7, x_4, x_8$

$\Pi_\beta : x_8, x_3, x_5, x_7, x_1, | x_2, x_4, x_6$

$\Pi\gamma : x_1, x_3, x_5, x_2, x_4, x_6, x_7, x_8$. Здесь $\Pi\gamma$-родительская хромосома после работы ОК. Например, граф $G,=(X, U)$, $|X|=8$ разбивается на 2 подмножества $G_1,=(X_1, U_1)$, $|X_1|=4$, $G_2,=(X_2, U_2)$, $|X_2|=4$. В решении родителя $\Pi\alpha:(X_1=\{x_1, x_3, x_5, x_6\}\ X_2=\{x_2, x_7, x_4, x_8\})$, в $\Pi\beta:(X_1=\{x_8, x_3, x_5,$

$x_7\}$ $X_2=\{x_1, x_2, x_4, x_6\}$). Результат ОК есть потомок $П\gamma:(X_1=\{x_1, x_3, x_5, x_2\}$ $X_2=\{x_4, x_6, x_7, x_8\})$.

Оператор мутации просматривает выбранные хромосомы и в них производится перестановка случайно выбранных генов согласно шкале мутации (вероятности P(OM)). Используется следующая модификация технологии генетического поиска. «Лучшая» хромосома может участвовать в репродукции не более одного раза в одной генерации. Кроме того, 2 потомка, полученные из двух родителей, обязательно перемещаются в следующую популяцию. Такой подход позволял увеличивать пространство поиска и не давал быстрой сходимости ГА [42].

Опишем идею итерационного алгоритма компоновки с применением генетических операторов. Рассмотрим разбиение схемы СБИС, представленной в виде гиперграфа H = (X, E), на части с минимизацией K. Для каждого ребра можно определить разрезающую функцию как число точек, попадающих в различные блоки. Если хотя бы одна вершина ребра гиперграфа лежит не в той части, что остальные вершины этого ребра, то обязательно будет существовать по крайней мере одно разрезающее ребро. Просуммировав все разрезающие, ребра получим общее число K (связей) между блоками, которое необходимо минимизировать. Если заданы размеры элементов, т.е. вершин гиперграфа, то часто необходимо максимизировать сумму площадей элементов блока.

Предлагаемый алгоритм будет состоять из двух уровней. На первом уровне строится популяция решений, которые могут быть получены случайным, последовательным или итерационным методами.

Далее согласно разработанным схемам генетического поиска определяется ЦФ. В качестве ЦФ выбирается K с ограничением на сумму площадей элементов в блоке. Затем полученная

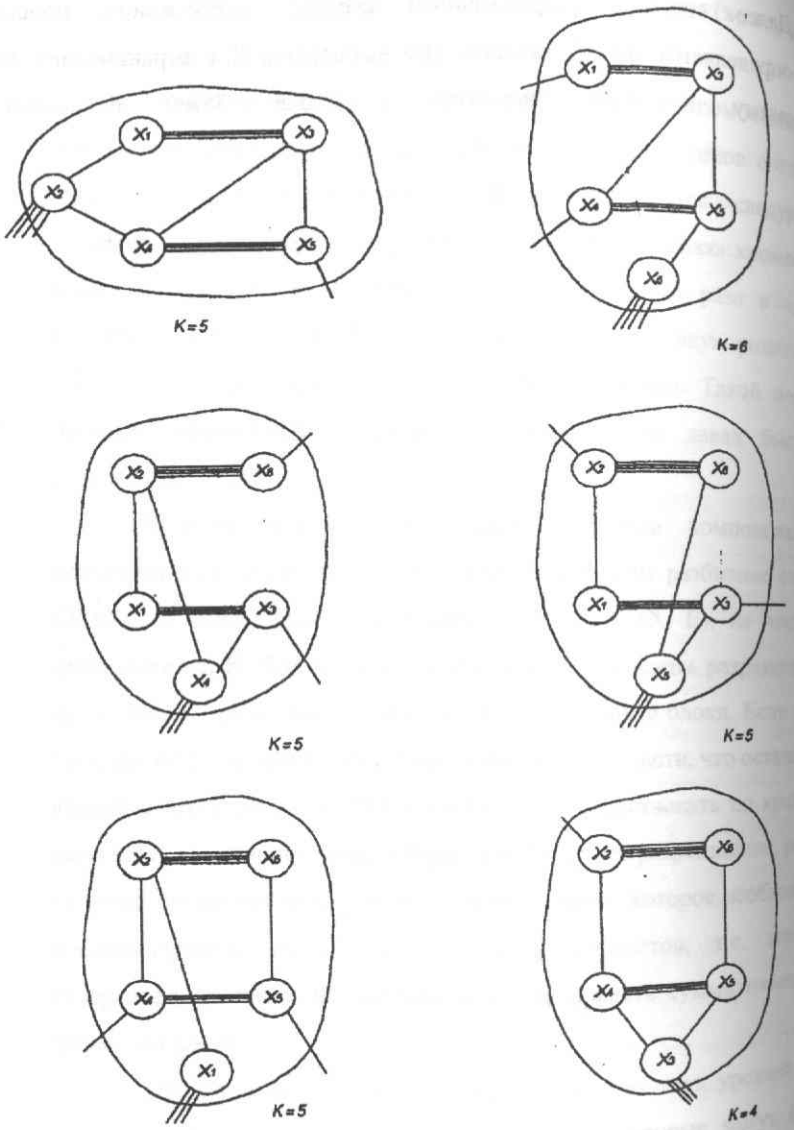


Рис.5.9 Максимальные связные графы из 5 вершин графа (рис.5.6)

популяция сортируется и упорядочивается согласно ЦФ. После этого, используя описанные схемы селекции, выбираются родительские пары для выполнения генетических операторов. Перед выполнением ГО в каждой хромосоме (текущий вариант разбиения) определяется стоимость каждого строительного блока. Работа ГО начинается с модифицированного ОК. При этом точка или точки ОК определяются не случайно, а направленно, выделяя лучшие строительные блоки в каждой хромосоме.

Например, на рис. 5.10. показана схема МОК. Здесь P_1 , P_2-хромосомы из популяции P. Строительные блоки, т.е. части разбиения, - {a, b, c}, {d, e}, {f, g}; {d, e, c}, {a, f}, {b, g}. Определяя ЦФ, найдено, что K_2 и K_5 наименьшие. Тогда точки ОК в P_1 направленным образом выделяют блок {d, e}, а в P_2-блок {a, f}. Затем производится копирование лучших строительных блоков из P_1 и P_2 в P'_1. Свободные места в P'_1 заполняются оставшимися элементами из P_1 и P_2. Возможны случаи, когда выбранные строительные блоки могут иметь частичное или полное совпадение элементов. При полном совпадении элементов производится копирование в потомки только одного из строительных блоков. При частичном совпадении строительный блок из первого родителя копируется полностью, а из второго родителя копируются несовпадающие элементы. Оператор мутации может выполняться случайно и направленно. На рис.5.11. показано возможное применение ОМ. Причем элементы могут мутировать только между строительными блоками. При направленной мутации в каждом строительном блоке выбирается наихудший элемент. Важную роль в генетических операторах играет вероятность ОК (P(OK)) и вероятность мутации

P(OM). Эксперименты, проведенные в [16], отмечают, что P(OK), лежащая в пределах 0,6-0,97, и $P_{ом}$-0,01-0,08, позволяют получать большое число локальных оптимумов за 100-500 генераций.

Для выхода из локальных «ям» будем применять две процедуры. В основе первой лежит модифицированный ОС. На рис. 5.12. показан пример такого оператора. Заметим, что строительные блоки из хромосом копируются полностью, если они не пересекаются с уже выделенными. В противном случае копируются только непересекающиеся элементы из строительных блоков.

В основе второй процедуры лежат идеи миграции хромосом, описанные выше. В зависимости от наличия вычислительных ресурсов можно первоначально строить не одну популяцию, а некоторый набор популяций. Далее необходимо выбирать «лучшего» представителя из каждой популяции и на этом множестве осуществлять описанный генетический поиск. Важным вопросом является управление ходом этого процесса. Выше приведены структурные схемы управления процессом генетического поиска. На этом макро-уровень алгоритма можно считать законченным. На втором микроуровне в качестве строительных блоков будут выбираться отдельные гены (элементы). Причем лучшие представители этих генов будут копироваться в потомки при выполнении генетических операторов.

Определим трудоемкость данного алгоритма. Пусть N_p-размер популяции, $N_п$-количество полученных потомков, N_G-количество генераций ГА. Тогда общую трудоемкость алгоритма можно ориентировочно определить

$T \approx [(N_p\, t_p + N_п\, t_п + (N_p + N_п)\, t_c\,]\, N_G$, где t_p-трудоемкость построения одной индивидуальности в популяции; $t_п$-трудоемкость генерации одного потомка; t_c-совокупная трудоемкость селекции и отбора.

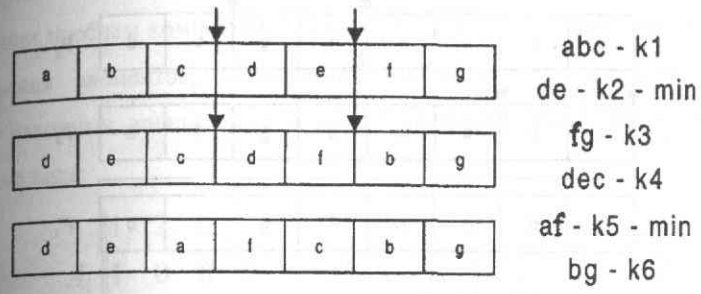


Рис.5.10 Схема модифицированного оператора кроссинговера

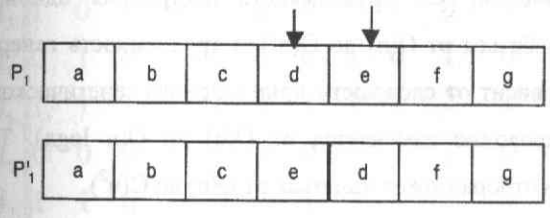


Рис.5.11. Пример оператора мутации

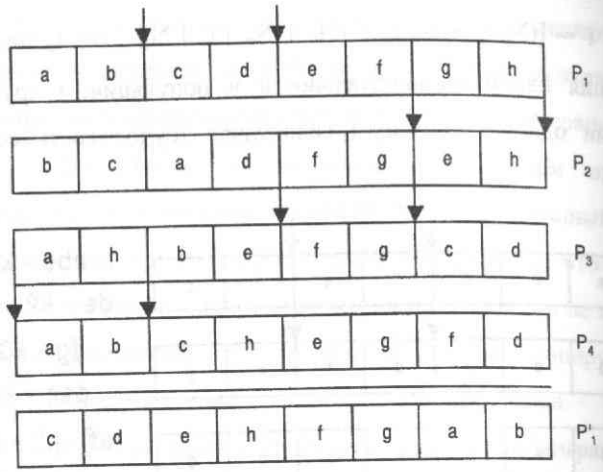


Рис. 5.12. Пример оператора сегрегации

Заметим, что трудоемкость построения одной хромосомы может колебаться от O(n) до O(n!), а трудоемкость генерации одного потомка зависит от сложности применяемого генетического оператора и ориентировочно изменяется от O(n) до O(n logn). Трудоемкость селекции и отбора может меняться от O(n) до $O(n^2)$.

$$r(i) = \frac{N_{лок}}{N_{общ}}. \qquad (5.5)$$

где $N_{лок}$-количество удовлетворительных решений (локальных оптимумов), n-размер индивидуальности (решения, хромосомы), n-общее количество решений. Причем $N_{общ}$ может быть равно n!. Данную величину можно повысить, увеличив число генераций ГА с 1 до N_G. Тогда

$$r(i) = \frac{N_{лок}}{N_{общ}} \cdot N_G. \qquad (5.6)$$

В заключение этого подраздела отметим, что сложность описанного алгоритма ориентировочно составляет $O(\lambda n^2)$ для одной генерации. Здесь λ-коэффициент, определяемый количеством ГО.

Приведем генетический алгоритм группирования элементов. При решении проблем компоновки элементов СБИС возникают задачи группирования элементов, обладающих одинаковыми свойствами. Моделью исходных данных задачи может, например, быть матрица следующего вида:

$$R = \begin{array}{c} \\ A_1 \\ A_2 \\ A_3 \\ A_4 \\ A_5 \\ A_6 \\ A_7 \\ A_8 \end{array} \begin{array}{c} \begin{array}{cccccccccccc} 1 & 2 & 3 & 4 & 5 & 6 & 7 & 8 & 9 & 10 & 11 & 12 \end{array} \\ \begin{Vmatrix} 1 & 0 & 0 & 0 & 1 & 0 & 1 & 1 & 1 & 0 & 1 & 1 \\ 0 & 0 & 1 & 1 & 0 & 0 & 1 & 1 & 0 & 0 & 1 & 1 \\ 1 & 0 & 1 & 0 & 1 & 0 & 1 & 0 & 1 & 0 & 1 & 1 \\ 0 & 1 & 0 & 1 & 0 & 1 & 0 & 1 & 0 & 1 & 0 & 1 \\ 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 \\ 0 & 0 & 0 & 1 & 1 & 1 & 0 & 1 & 1 & 1 & 0 & 0 \\ 1 & 1 & 0 & 1 & 0 & 1 & 1 & 0 & 1 & 0 & 1 & 1 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \end{Vmatrix} \end{array}.$$

Здесь столбцы матрицы R соответствуют заданным элементам схемы, а строки - свойствам или выполненным функциям этих элементов. Например, единица на пересечении A_4 строки и 4 столбца означает, что элемент 4 обладает свойствами A_4.

Необходимо сгруппировать элементы, обладающие наибольшим числом одинаковых свойств. Данная задача относится к классу задач компоновки и может быть решена одним из эвристических алгоритмов. Кроме того, здесь может быть применена одна из схем генетического поиска.

Опишем кратко основную идею предлагаемого эвристического алгоритма. Предварительно выполняется анализ матричной модели схемы. В результате анализа предлагается устранить из рассмотрения строки матрицы, содержащие все единицы, все нули, а также строки, содержащие около 1% нулей или единиц. Такие свойства не будут влиять на создание требуемых групп, так как все или почти все элементы будут входить или не входить в формируемые группы. Тогда в рассматриваемом примере исключим строки A_5 и A_8. Далее произведем сортировку матрицы R. При этом на первое место поместим столбец с наибольшим числом единиц и т.д. по убыванию. В этом случае матрица R запишется

$$R = \begin{array}{c} \\ A_1 \\ A_2 \\ A_3 \\ A_4 \\ A_6 \\ A_7 \end{array} \left\| \begin{array}{ccc|ccc|ccc|cc|c} 12 & 4 & 7 & 8 & 9 & 11 & 1 & 5 & 6 & 2 & 3 & 10 \\ 1 & 0 & 1 & 1 & 1 & 1 & 1 & 1 & 0 & 0 & 0 & 0 \\ 1 & 1 & 1 & 1 & 0 & 1 & 0 & 0 & 0 & 0 & 1 & 0 \\ 1 & 0 & 1 & 0 & 1 & 1 & 1 & 1 & 0 & 0 & 1 & 0 \\ 1 & 1 & 0 & 1 & 0 & 0 & 0 & 0 & 1 & 1 & 0 & 1 \\ 0 & 1 & 0 & 1 & 1 & 0 & 0 & 1 & 1 & 0 & 0 & 1 \\ 1 & 1 & 1 & 0 & 1 & 1 & 1 & 0 & 1 & 1 & 0 & 0 \end{array} \right\|$$

Затем производится разбиение матрицы R на две части: R_1 «перспективных» хромосом и R_2 «неперспективных» хромосом. Разбиение, включающее элементы {12, 4, 7, 8, 9, 11}, относится к «перспективным», а разбиение {1, 5, 6, 2, 3, 10} – к «неперспективным». В свою очередь, для распараллеливания процесса части R_1 и R_2 можно снова разбить на две части R'_1, R''_1 и R'_2, R''_2. Данный процесс можно продолжать до получения подмножеств разбиения, которые легко поддаются обработке генетическими операторами. Предполагается

процедура поэлементного сложения столбцов матрицы K по следующим правилам: 1 + 1 = 1, 1 + 0 = 0, 0 + 0 = 0.

Данная процедура аналогична описанной в [16]. Например, выполняя указанные сложения для «перспективных» столбцов матрицы R внутри частей R'_1 и R''_1, получим

12 : 1 1 1 1 0 1	12 : 1 1 1 1 0 1	4 : 0 1 0 1 1 1
4 : 0 1 0 1 1 1	7 : 1 1 1 0 0 1	7 : 1 1 1 0 0 1
_____,	_____,	_____.
(12, 4) : 0 1 0 1 0 1	(12, 7) : 1 1 1 0 0 1	(4, 7) : 0 1 0 0 0 1

В группе (12, 4) – 3 общих свойства, в группе (12, 7) – 4 общих свойства и в группе (4, 7) – 2 общих свойства. Продолжая далее, получим

8 : 1 1 0 1 1 0	8 : 1 1 0 1 1 0	9 : 1 0 1 0 1 1
9 : 1 0 1 0 1 1	11 : 1 1 1 0 0 1	11 : 1 1 1 0 0 1
_____,	_____,	_____.
(8, 9) : 1 0 0 0 1 0	(8, 11) : 1 1 0 0 0 0	(9, 11) : 1 0 1 0 0 1

В группе (8, 9) – 2 общих свойства, в группе (8, 11) – 2 общих свойства и в группе (9, 11) – 3 общих свойства. Условимся фиксировать группы, имеющие три и более общих свойств. Выполняя аналогичные преобразования внутри «перспективных» частей R'_2 и R''_2, получим следующие группы (1, 5) – 2 общих свойства; (1, 6)-1; (5, 6)-1; (2, 3)-0; (2, 10)-1; (3, 10)-0. Производя сортировку полученных решений, сформируем следующую популяцию:

P = {(12,7), (12,4), (9,11), (4,7), (8,9), (8,11), (1,5), (2,10)}. Для выполнения ОК сформируем следующие родительские пары

P_1 : 12 | 7	P_2 : 12 | 4	P_5 : 8 | 9	P_6 : 8 | 11
P_3 : 9 | 11	P_4 : 4 | 7	P_7 : 1 | 5	P_8 : 2 | 10

_____, _____, _____, _____.

$P'_1 : 12 | 11$ $P'_2 : 12 | 7$ $P'_5 : 8 | 5$ $P'_6 : 8\ 10$

$P'_3 : 9 | 7$ $P'_4 : 4 | 4$ $P'_7 : 1 | 9$ $P'_8 : 2 | 11$

В группах после применения стандартного одноточечного ОК имеем: (12, 11) – 4 общих свойства; (9, 7) – 3; P'_2 совпадает с P_1, поэтому не рассматривается, P'_4 – нелегальная группа; (8, 5)-2; (1, 9)-3; (8, 10)-2; (2, 11)-1.

Сформируем теперь новую популяцию:

P_1 = {(12,7), (12,11), (9,11), (12,4), (9,7), (1,9), (8,5), (8,10)}.

Здесь размер P_1 оставлен равным P. Как отмечалось выше, величину размера популяции можно варьировать в зависимости от качества получаемых решений. В принципе можно выполнить некоторое множество генераций данной операции. Произведем теперь склеивание (объединение) групп по критерию максимума общих свойств. Тогда получим следующие группы: (12, 7, 11) – 4 общих свойства; (12, 7, 9)-3; (12, 11, 9)-3; (7, 11, 9)-3. Продолжая аналогично, получим группу (12, 7, 11, 9, 1)-3 общих свойства. ВСА данного алгоритма для одной генерации меняется от O(αn) до O(αn logn). Данный алгоритм несложен и может выполняться параллельно. Ориентировочная трудоемкость алгоритма

$T \approx (N_p t_p + N_P t_{OK} + N_p t_{скл}) N_G$, где t_{OK} - трудоемкость ОК, а $t_{скл}$ - трудоемкость операции склеивания.

В работе [19] предлагается генетический алгоритм дихотомического разбиения графа G=(X, U) на два подграфа, основанный на работе [43]: $G_1=(X_1, U_1)$ и $G_2=(X_2, U_2)$, $X_1 \cup X_2 = X$, $U_1 \cup U_2 \cup U_3 = U$, где U_3 -подмножество ребер, соединяющих два подграфа,

причем для любого ребра $U_i \in U_3$, $U_i=(x_i, x_j)$, $x_i \in X_1$ и $x_j \in X_2$ или $x_i \in X_2$ и $x_j \in X_1$.

Целевая функция или критерий развития - это число ребер, соединяющих подграфы ($K=|U_3|$). Тогда оптимально дихотомическое разбиение является решением (X_1^*, X_2^*) следующей экстремальной задачи однокритериального вектора

$$Q^* = Q(X_1^*, X_2^*) = \text{MINK}, (X_1, X_2 \subset X),$$

причем $K = |U \setminus (U_1 \cup U_2)|$,

где U_1, U_2 -подмножества ребер, соединяющих только вершины $X_1(X_2)$ между собой.

Вместо числа К соединяющих ребер можно рассматривать в общем случае сумму весов соединяющих ребра.

Очевидно, что экстремальная задача минимизации К эквивалентна задаче максимизации числа внутренних ребер подграфов.

Дихотомическое разбиение согласно [19] можно представить в виде бинарных строк. Например, для графа G = (X, U), |X| = 8, при разбиении его на два подграфа, $|X_1| = 3$, $|X_2| = 5$, можно записать:

1 2 3 4 5 6 7 8 – номер бита

x_1 x_2 x_3 x_4 x_5 x_6 x_7 x_8 – номер вершины графа.

1 0 0 0 1 0 0 1

$G(X_1, X_2)$.

Здесь $G(X_1, X_2)$–закодированное разбиение. 1 означает, что вершины x_1, x_5, x_8 -в подграфе X_1, а 0 означает, что вершины x_2, x_3, x_4, x_6, x_7 -в подграфе X_2.

Согласно [19] аналог хромосомы a_k^t (особи) в экстремальной задаче дихотомического разбиения принимает произвольно допустимое решение, которому присвоено имя a_k^t (здесь – к индекс хромосомы, t – номер генерации). Тогда эволюционный процесс интерпретируется как перераспределение вершин графа G между подграфами G_1, G_2 с целью отыскания минимума критерия оптимальности. Хромосома содержит n сцепленных между собой генов, расположенных в линейной последовательности слева направо. Тогда в бинарной строке номер бита – это номер гена, номер вершины графа – локус т.е. местоположение отдельного гена в хромосоме. И, наконец, например первая 1 в строке $G(X_1, X_2)$-это аллель 1 гена, находящегося в локусе x_1. Напоминаю, что аллель-это альтернативная форма одного и того же гена. Здесь аллель может быть равен 1 или 0. Хромосома имеет два изменяемых признака – качественный и количественный. Качественный определяется видом бинарной строки, количественный – числом. Хромосому, содержащую в своих локусах конкретные значения аллелей, в [19] называют генотипом с генетическим кодом $R(a_k^t)$. Он содержит наследственную информацию, передаваемую о хромосоме a_k^t от предков к потомкам. Дальнейшая оптимизация выполняется на основе ПГА или другого ГА.

В заключение этого раздела отметим, что генетическая процедура поиска может быть применена как единственная оптимизационная процедура. В этом случае сложность алгоритма компоновки будет лежать в пределах $O(n)$-$O(n^3)$. Причем крайний случай с $O(n^3)$ будет иметь место при популяции больше пятидесяти и

от ста и выше числа генераций алгоритма. Кроме того, генетические операторы могут быть вставлены как вспомогательные блоки для повышения качества алгоритмов.

5.3. Алгоритмы размещения элементов с использованием методов генетического поиска

Алгоритмы размещения согласно [37, 43] можно разделить на два основных класса. Это конструктивные (последовательные) алгоритмы и методы итеративного улучшения. Для их решения применяют пять основных подходов. К ним относятся моделирование отжига; силовое размещение; размещение посредством разбиения; числовые оптимизационные подходы; размещение посредством ГА.

Основная цель алгоритмов размещения – минимизировать общую площадь (кристалла) чипа, где размещаются элементы, и минимизировать общую суммарную длину всех цепей. Часто также ставится задача (или ограничение) минимизировать общее число внутрисхемных пересечений соединительных проводников.

На рис.5.13 показана модель плоскости для задачи размещения. На исходную плоскость чипа накладывается декартова система координат с осями s и t. В каждой ячейке плоскости может быть размещен элемент гиперграфа схемы. Расстояние между элементами схемы вычисляется на основе одной из известных формул [37]:

$$d_{ij} = |s_i - s_j| + |t_i - t_j|, \text{ или}$$

$$d_{ij} = \sqrt{|s_i - s_j|^2 + |t_i - t_j|^2}, \qquad (5.6)$$

Здесь (s_i, t_i); (s_j, t_j)-координаты x_i, x_j элементов, d_{ij}-расстояние между элементами x_i, x_j на заданной плоскости. Причем первое выражение (5.6) позволяет определить Манхеттоново расстояние, т.е. расстояние между двумя точками, определенное по вертикальным и горизонтальным направлениям. Шаг (расстояние) между двумя рядом лежащими элементами по горизонтали и вертикали считается равным единице. Выражение (5.6) позволяет вычислить прямолинейное расстояние между двумя точками. В случае многозвенной цепи чья длина подсчитывается как полупериметр прямоугольника, охватывающего ее концевые точки (длина цепи е на рис. 5.13. равна 5). Тогда общая длина всех соединений (ребер модели) определяется по известной формуле

$$L(G) = \frac{1}{2}\sum_{i=1}^{n}\sum_{j=1}^{n} d_{ij}c_{ij}. \qquad (5.7)$$

Здесь L(G)-общая суммарная длина ребер графа, c_{ij} - количество ребер, соединяющих элементы x_i и x_j. Число пересечений ребер графовой модели определяют по формуле

$$\Pi(G) = \frac{1}{2}\sum_{u_{ij}\in I}^{n} \Pi(u_{ij}). \qquad (5.8)$$

Здесь П(G)-число пересечений ребер графовой модели схемы.

Задача, решаемая в работе, заключается в разработке алгоритмов минимизирующих L(G) и П(G) (L(G) →min, П(G) → min). Отметим, что при размещении элементов СБИС минимизация суммарной длины проводников обычно приводит к минимизации числа внутрисхемных пересечений. Такая одновременная минимизация длины и пересечений происходит до некоторого предела, начиная с которого

уменьшение длины приводит к увеличению числа пересечений. Поэтому введем комплексный критерий A(G).

$$A(G) = \alpha\, L(G) + \beta\, \Pi(G). \qquad (5.9)$$

Здесь α, β-коэффициенты, учитывающие степень важности того или иного критерия ($\alpha + \beta = 1$). Значения α и β могут определяться на основе опыта конструктора или экспертных оценок. На рис.5.14 показан пример кристалла для размещения стандартных ячеек СБИС в линейных рядах.

Рис.5.13. Модель плоскости для размещения.

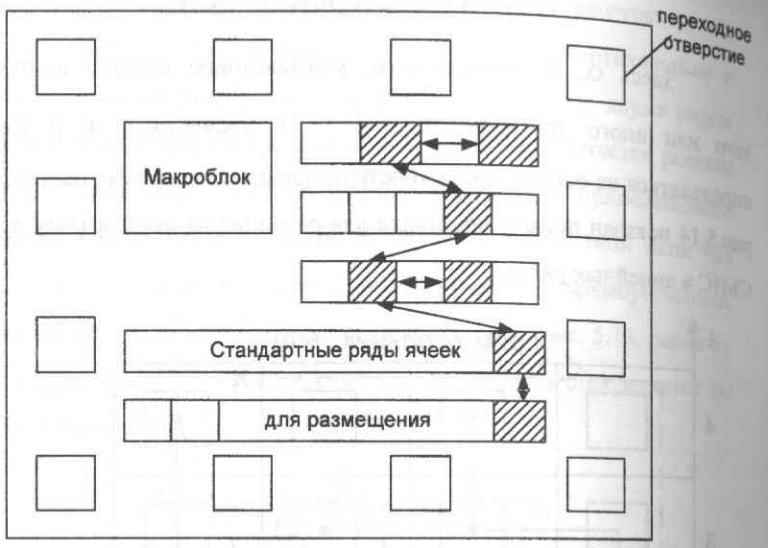


Рис.5.14. Пример кристалла (чипа) для размещения ячеек СБИС в рядах

Итак, в задаче размещения входными данными будут множество модулей (вершин графовой или гиперграфовой модели) $X=\{x_1, x_2, ..., x_n\}$, множество сигнальных цепей (ребер гиперграфа) $E=\{e_1, e_2, ..., e_m\}$, множество позиций $Q = \{q_1, q_2, ..., q_f\}$. Причем $|X| \leq |Q|$, если производится размещение элементов на заданные позиции. Иногда производится размещение сигнальных цепей, и тогда количество позиций должно быть $\geq$ количества концевых точек размещаемых цепей. Сформулируем задачу размещения как назначение каждого элемента в единственную позицию (ячейку) таким образом, чтобы оптимизировать ЦФ. В качестве ЦФ выбирается выражение (5.9). Выходными данными будут позиции назначения каждого модуля.

В качестве ограничений выбираются следующие условия. Для каждого $x_i \in X$ назначается только одна позиция $q_i \in Q$. Каждой позиции $q_i \in Q$ соответствует по крайней мере один элемент $x_i \in X$. Известно [37],что проблема размещения относится к классу NP-трудных проблем. Поэтому постоянно производится поиск эвристических процедур для повышения качества и уменьшения сложности алгоритма. Алгоритм считается «вероятностно хорошим алгоритмом», если его ВСА-O(n), O(n log n), или $O(n^2)$ [37, 41].

В данной работе описана новая методология размещения СБИС на основе ГА. В настоящее время выделяют два подхода применения ГА для размещения стандартных ячеек СБИС на плоскости кристалла (чипа). Первый подход использует методологию Клинга [16] для размещения элементов, одинаковых по высоте и различных по ширине на основе моделирования эволюции. Второй подход основан на идеях Кохона, Париса, Шахокара и Мазумдера [29, 49].

Первый подход заключается в следующем. Сначала формируется популяция решений, обычно случайным образом. Далее в каждом элементе популяции выполняется оператор мутации. Затем вычисляется ЦФ, производится сортировка решений и назначение элементов в заданные позиции. В случае необходимости производится возврат к блоку мутации и вся процедура продолжается аналогично до преждевременной сходимости или пока не выполнится заданное число итераций.

Предлагается модификация данного подхода. Она заключается в использовании дополнительных генетических операторов, имеющих аналоги в биологии. Структурная схема предлагаемого алгоритма

показана на рис.5.15. Начальная популяция будет конструироваться тремя способами G_1 , G_2 , G_3. В случае G_1 популяция решений (заданных размещений) формируется случайным образом. В G_2 популяция решений получается путем неоднократного применения последовательного алгоритма. Формирование популяции в G_3 производится совместно случайным и направленным образом. После формирования популяции к каждому ее элементу применяется оператор мутации. Причем ОМ может выполняться независимо для каждого элемента популяции. Вводится три варианта ОМ:

- случайная мутация (P_m = 0,01-0,05);
- направленная мутация (P_m = 0,01-0,05);
- случайно-направленная мутация (P_m = 0,01-0,05).

Затем выполняется оператор инверсии (ОИ). Предлагается две версии ОИ: случайная ($ОИ_1$) и направленная ($ОИ_2$). Потом выполняются три версии оператора сегрегации затем переход к выполнению оператора транслокации (ОТ). Также вводится две версии ОТ: случайная ($ОТ_1$) и направленная ($ОТ_2$). Последними выполняются операторы кроссинговера (ОК). Здесь $ОК_1$ одноточечный ОК, $ОК_2$ - двухточечный ОК, $ОК_3$ - упорядоченный ОК, $ОК_4$ - циклический ОК, $ОК_5$ - модифицированный ОК. Предварительно вычисляется верхняя оценка абсолютно минимального размещения для заданной конфигурации плоскости. После каждого ГО результаты размещения сравниваются с оценкой. При получении искомых результатов алгоритм заканчивает свою работу. После ОК вычисляется ЦФ и на ее основе сортируются варианты решений. После этого формируется новая популяция, и процесс продолжается аналогично. Возможен

выход из каждого оператора в отдельности при получении локального оптимума.

Рассмотрим пример размещения элементов СБИС в одной линейке. Пусть задан граф схемы, показанный на рис.5.16. [16, 37]. Данный граф необходимо разместить в линейке с шестью позициями с минимизацией суммарной длины связей. В [37, 43] описаны практически все методы, кроме генетического, для решения задач размещения. При линейном размещении в случае небольшого количества элементов перспективными являются метод ветвей и границ и симплекс метод [43]. С ростом числа элементов СБИС эффективное применение нашли генетические алгоритмы [29].

Пусть для графа (рис.5.16) случайным образом получена следующая популяция решений (размещение элементов в линейке). $P=\{P_1,..., P_6\}$, причем

P_1 : 1 2 3 4 5 6, P_4 : 3 5 1 6 4 2,

P_2 : 1 3 5 2 4 6, P_5 : 4 6 2 5 1 3,

P_3 : 2 4 6 1 3 5, P_6 : 5 1 4 3 6 2.

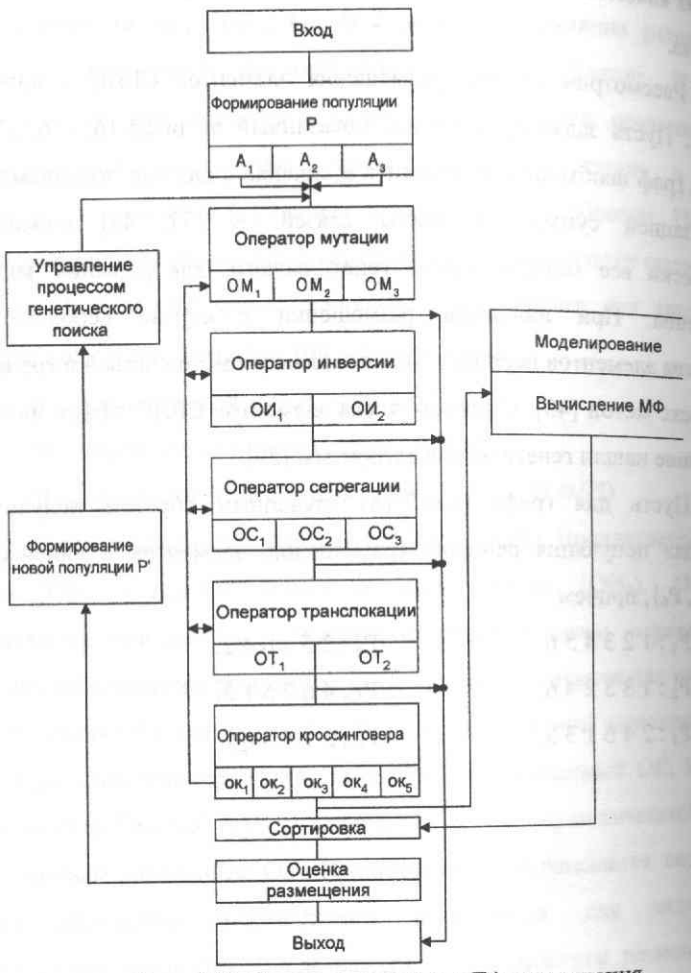


Рис. 5.15. Структурная схема ГА размещения

Согласно [43] определим оценку минимальной суммарной длины для данного графа. Она будет равна $L(G\Delta)_{min} = 11$. Теперь вычислим ЦФ для каждой индивидуальности в популяции. Тогда суммарная длина связей для решения P_1 определится

$L(G)_{p1} = 1\cdot3 + 2\cdot2 + 3\cdot1 + 4\cdot1 + 5\cdot1 = 19;$

$L(G)_{p2} = 21;\ L(G)_{p3} = 17;\ L(G)_{p4} = 16;\ L(G)_{p5} = 18;\ L(G)_{p6} = 20.$

Произведем сортировку индивидуальностей согласно их ЦФ и получим

$P_4; P_3; P_5; P_1; P_6; P_2.$

Начиная с P_4, применим OM_1 . Пусть точка мутации случайным образом определится между 5 и 6 элементами. Тогда получим

$P_4 : 3\ 5\ 1\ 6\ 4\ |\ 2,.$ $\quad L(G)_{p4} = 16,$

$P'_4 : 3\ 5\ 1\ 6\ 2\ 4,$ $\quad L(G)_{p'4}\ 16.$

Далее предлагается два основных решения. В первом – выполняется мутация всех P_i из P и строится новая популяция P'. Во втором – популяция изменяется после каждой операции мутации. Например, вставим P'_4 в P' исключив P_2 , тогда $P' = \{P_4 , P'_4, P_3 , P_5 , P_1 , P_6\}$.

Покажем теперь применение оператора инверсии $OИ_1$. Возьмем P_4, в котором случайным образом точка инверсии попадает между 1 и 2 элементами. Тогда

$P_4 : 3\ |\ 5\ 1\ 6\ 4\ 2,$ $\quad L(G)\ p_4 = 16,$

$P''_4 : 3\ |\ 2\ 6\ 4\ 15,$ $\quad L(G)\ p_4'' = 12.$

Величина L(G) приближается к абсолютному минимуму. В принципе на этом процесс размещения может быть закончен. Рис.(5.17.) Отметим, что в алгоритме возможны обратные связи после каждого ГО. Так, например, для лучшей индивидуальности P''_4 выполним направленный оператор мутации (OM_2).

P_4'' : 3 2 4 6 | 1 5, $L(G)\ p_4'' = 12$,

P_4'' : 3 2 4 1 | 6 5, $L(G)\ p_4'' = 11 = L(G)_{min}$, .

На рис.5.18 показано минимальное размещение графа G (рис.5.16) в линейке. Для того чтобы получить конечное размещение во всех линейках, применяются описанные выше генетические алгоритмы компоновки. Для этого нужно разбить схему на заданное число линеек с минимизацией количества связей между линейками, а затем произвести оптимизацию размещения внутри каждой линейки, используя методы генетического поиска. Определим ориентировочную трудоемкость разработанного линейного алгоритма.

$T \approx [N_p t_p + N_p(t_{ом} + t_{ои} + t_{ос} + t_{от} + t_{ок})]\ N_G$,здесь $t_{ом}$ - трудоемкость оператора мутации, $t_{ои}$ - трудоемкость оператора инверсии, $t_{ос}$ - трудоемкость оператора сегрегации, $t_{от}$ - трудоемкость оператора транслокации.

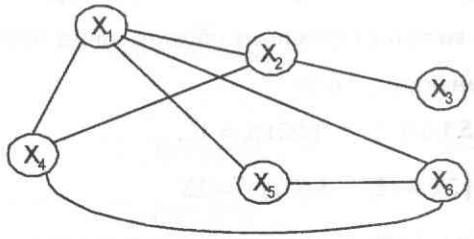


Рис. 5.16. Граф схемы G = (X,U)

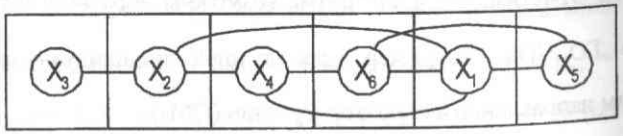


Рис.5.17. Размещение P"4 с L(G)P"4=12

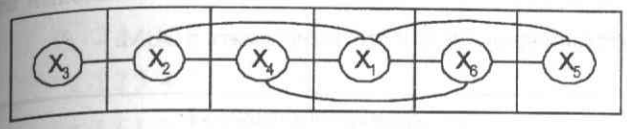


Рис.5.18. Наилучшее размещение графа G (рис.5.16) в линейке (L(G)=11)

ВСА линейного размещения меняется от $O(\alpha n)$ до $O(\beta n^2)$, где α, и β-коэффициенты, зависящие от используемых процедур поиска и их специфики. Вероятность получения оптимального результата можно определить так:

$$P_{i(опт)} = \rho_i * (N_p + N_n) * N_G,$$

где P_i - вероятность получения оптимального результата при генерации одной хромосомы в популяции.

Рассмотрим теперь размещение элементов СБИС в решетке на основе ГА. В качестве базовых будут использованы методы, описанные в [37, 41, 43].

Опишем применение разработанной выше архитектуры метагенетической оптимизации для размещения стандартных ячеек СБИС в решетке. Структурная схема алгоритма показана на рис.5.19. Размер популяции обычно выбирается исходя из опыта конструктора и имеющихся вычислительных ресурсов. Теоретические исследования в [9-25] рекомендуют следующий размер популяции: $N \approx n/2$, где n-число элементов в исследуемой модели. Определения числа генераций, наряду с размером популяций, являются нерешенными проблемами в ГА. Отметим, что глобальный минимум может быть получен как на первой генерации, так и не найден на последней. Процесс генерации

следует прекращать при попадании алгоритма в локальный минимум, при отсутствии времени или ресурсов памяти в ЭВМ.

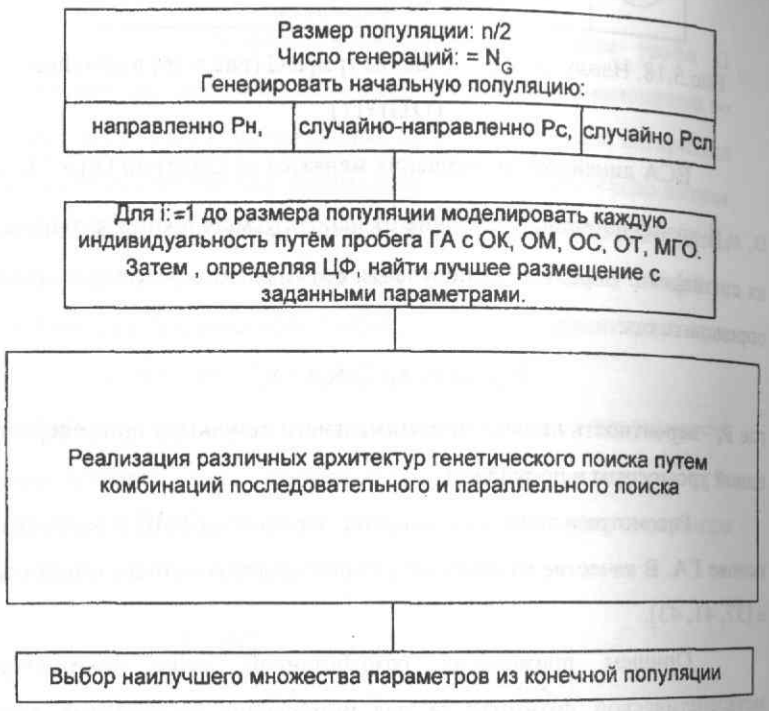


Рис.5.19. Укрупненная структурная схема метагенетического алгоритма размещения

Применение стандартных ГА может не давать реальных размещений, поэтому необходимо использовать различные модифицированные ГО, приведенные выше. Например, на рис.5.20 показаны два родительских сегмента размещения ячеек.

P_1 : 1 2 3 4 5 6 | 7 8 9 10 (ячейки)

s -1 2 3 4 5 5 | 4 3 2 1

t -1 1 1 1 1 2 | 2 2 2 2 (координаты)

P_2 : 2 8 1 10 9 7 | 3 5 6 4 (ячейки)

s -1 2 3 4 5 5 | 4 3 2 1

t -1 1 1 1 1 2 | 2 2 2 2 (координаты)

Применим к P_1 и P_2 одноточечный ОК. Тогда

P'_1 : 1 2 3 4 5 6 | 3 5 6 4.

В P'_1 имеются повторяющиеся элементы (3, 5, 6, 4), поэтому использование стандартного ООК нецелесообразно. Простая модификация любого стандартного ОК позволит получать легальные размещения. Используя модифицированный ОК, выбираем в P'_1 левый от точки ОК сегмент хромосомы и копируем его в P'_1. В P'_2 хромосоме выбираем правый от точки ОК сегмент и производим его анализ на предмет отсутствия повторяющихся генов. Все повторяющиеся гены исключаются из P'_1, а вместо них добавляются отсутствующие элементы из левого сегмента P_2. Тогда в рассматриваемом примере получим

P'_1 : 1 2 3 4 5 6 8 10 9 7

s -1 2 3 4 5 5 4 3 2 1.

На рис.5.21 показано возможное оптимальное размещение ячеек хромосомы P'_1(1, 2, 3, 4, 5, 6, 8, 10, 9, 7) после применения модифицированного ОК.

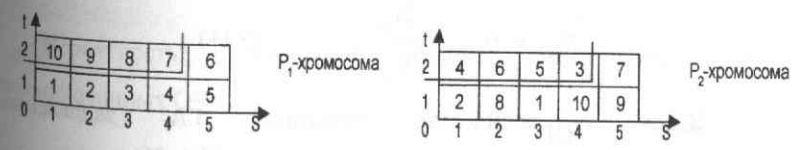


Рис.5.20. Пример двух сегментов ячеек

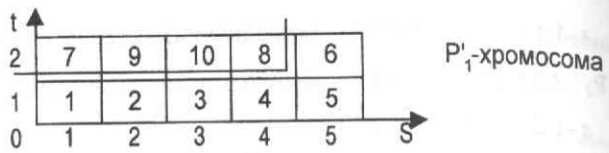


Рис.5.21. Размещение после выполнения модифицированного ОК

Важным вопросом в метагенетическом алгоритме размещения является понятие шкал ГО [9, 10, 29]. Обозначим размер популяции P- N_p, число генераций - N_G, число размещаемых вариантов для копирования N_{off}. Тогда шкалы ОК и ОМ можно определить согласно [29] следующим образом:

$$R_{ok} = \frac{N_{off}}{N_p} \qquad (5.10)$$

Оператор ОК управляется посредством шкалы ОК (R_{ok}). R_{ok} определяется как отношение числа offspring (потомков), полученных в каждой генерации, к размеру популяции. Отметим, что R_{ok} помогает находить отношение числа потомков (применение ОК) в районах с высокой ЦФ к числу потомков в других районах популяций. Высокая R_{ok} позволяет анализировать большее пространство и шанс нахождения оптимума решения увеличивается. В то же время очень высокая R_{ok} требует огромных ресурсов вычислительного времени. Количество генераций в задачах САПР можно находить по формуле

$$N_G = N_{GП} \cdot \frac{N_p}{N_{off}}. \qquad (5.11)$$

Здесь N_{GP}количество генераций ГА, задаваемых пользователем.

Шкала мутации $R_{ом}$ в [29] определяется как процент общего числа генов в популяции, которые в ней мутируют. При размещении n стандартных ячеек в решетке

$$R_{ом} = \frac{n \cdot N_p \cdot R_{ok}}{2} \qquad (5.12)$$

В (5.12.) n-число генов в популяции P. R_{OK} символизирует парный обмен генов при мутации. Шкалы остальных ГО определяются аналогично. В [29] приведена вероятность выживания лучших строительных блоков от родителей к потомкам. Так для упорядоченного ОК вероятность выживания схемы из первого родителя (для каждого целого взятого родителя) к потомку определена следующим образом

$$\rho_1 = \frac{K - D + 1}{L}. \qquad (5.13)$$

Здесь L- длина хромосомы, K - часть длины хромосомы до разрезающей точки, D-часть длины хромосомы от раздела до последнего определенного гена. Для элементов с наилучшими значениями ЦФ необходимо $\rho_1 \to \max$. Вероятность выживания второго родителя

$$\rho_2 = \left(1 - \frac{K}{L}\right)^D. \qquad (5.14)$$

Вероятность выживания схемы для введенных модифицированных ОК

$$\rho_3 \approx \frac{K - D + 1}{L + Q} + \left(1 - \frac{K + Q}{L}\right)^D. \qquad (5.15)$$

В (5.15) Q - число повторяющихся генов в нелегальной хромосоме (потомке).

В заключение подраздела отметим, что сложность описанных алгоритмов размещения является полиномиальной величиной. Для одной генерации ВСА лежит в пределах $O(\alpha_1 n^2)$-$O(\alpha_2 n^3)$, где α_1, α_2- коэффициенты, определяемые количеством ГО и их спецификой.

5.4. Разработка алгоритма планирования кристалла на основе генетического поиска

Планирование кристалла – одна из важнейших задач, определяющая качество конструкторского проектирования СБИС. Проблема планирования кристалла состоит в назначении данного множества модулей на плоскости с минимизацией весовой суммы площади при размещении заданных прямоугольников и минимизации суммарной длины цепей, которые должны соединять модули. Тогда проблема планирования может быть определена как оптимизационная проблема, и, следовательно, для ее решения можно использовать ГА. В [16, 49] данная проблема формулируется следующим образом.

Дано множество M = {1, 2, ..., m} модулей. Все модули представляются прямоугольниками. Каждый i модуль характеризуется кортежем длины 3 $<S_i, l_i, u_i>$, где S_i-площадь модуля, l_i, u_i-верхние и нижние границы отношения высоты h_i к ширине w_i. Здесь $w_i h_i = S_i$ и $l_i \le h_i/w_i \le u_i$. Проблема планирования согласно [49] состоит в разбиении плоскости горизонтальными и вертикальными линиями в m непересекающихся райнов. Каждый район r_i имеет ширину x_i и высоту y_i, такие, что $S_i \le x_i, y_i (x_i \ge w_i, y_i > h_i)$. Каждый план может оцениваться

посредством ЦФ, учитывающей общую площадь компонента и суммарную длину соединений. Общая площадь определяется как $S = \sum_{i=1}^{m} x_i y_i$, длина определяется по формулам (5.6). Тогда общая ЦФ для задачи планирования может быть записана

$$ЦФ = \sum_{i=1}^{m} x_i y_i + \frac{1}{2}\sum_{i,j=1}^{m} c_{ij} d_{ij}. \qquad (5.16.)$$

Здесь c_{ij} - количество связей, соединяющих i и j модули, d_{ij}- расстояние между модулями i, j. Для получения математической модели плана кристалла используют так называемую кусочную структуру или "польские выражения" [16]. На рис.5.22, а показан план кристалла, на рис.5.22,б - его кусочная структура и на рис.5.22,в - «Польское выражение». Модели кусочной структуры или «польского выражения» находят широкое применение при решении задач планирования. Эти модели используются при построении популяций в ГА.

Структура ГА для решения данной задачи описывается следующим образом. Сначала создается одна или несколько популяций решений, что соответствует $P_a = \{P^a_1, P^a_2, ..., P^a_k\}$, $k \approx n/2$, где n- количество элементов (модулей), $P_b = \{P^b_1, P^b_2, ..., P^b_k\}$, и т.д. Популяции, как уже отмечалось выше, создаются случайным, направленным или направленно-случайным образом. Следует заметить, что в популяции будут оставаться только реальные решения.

При наличии сети ЭВМ каждую генерацию можно параллельно реализовывать на своем процессоре. Рассмотрим случай наличия нескольких популяций. Произведем сортировку хромосом в каждой из

популяций. Затем применим описанные схемы селекции для выбора родительских пар. Причем, в каждой паре родители должны быть из разных популяций. В [49] описаны четыре модифицированных ОК. Для повышения качества решения кроме рассмотренных ГО используются все описанные выше ГО. Например, пусть заданы хромосомы (планы кристалла). Применяя к ним модифицированные операторы ОК, ОТ, ОМ можно, например получить следующий результат (хромосомы P'_1, P'_2, P_1'', P_2'').

P_1 : 1 5 8 7 * + + 2 4 3 * * + 6 +

P_2 : 2 6 8 * * 4 + 1 3 * + 7 5 * +

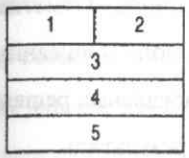
P'_1 : 1 5 8 7 * + + 3 4 2 * * + 6 +

P'_2 : 8 6 2 * * 4 + 1 3 * + 7 5 * +

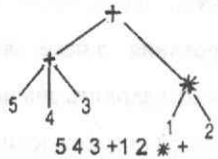
P''_1 : 7 8 5 1 * * + 3 4 2 * + * 6 +

P''_2 : 8 6 2 * + 4 + 3 1 * * 5 7 + +

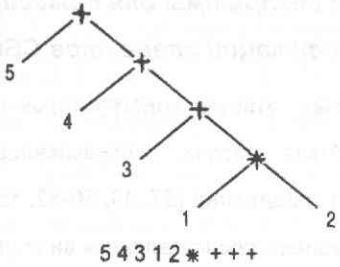
Каждой хромосоме соответствует определенный план кристалла. Основная задача ГО - найти строительные блоки с лучшими характеристиками для копирования их в потомки. С этой точки зрения для задач планирования эффективными оказались операторы сегрегации, мутации и модифицированной инверсии.



а). план кристалла



б). кусочная структура



в). польское выражение

Рис. 5.22 План кристалла и его модели

Задачи планирования кристалла тесно связаны с оптимизационными задачами упаковки и сжатия [43, 55].Трудоемкость алгоритма ориентировочно определена так

$T \approx T_M + [(N_p t_p + (N_p + N_п)(t_{ок} + t_{ос} + t_{ом} + t_{ои}) + N_p t_c] N_G$, где T_M-трудоемкость реализации модели, например в виде польской записи; t_c - трудоемкость селекции; $t_{ок}$, $t_{ос}$, $t_{ом}$, $t_{ои}$, - трудоемкости соответствующих генетических операторов.

Увеличить вероятность получения оптимального решения можно за счет управления процессом генетического поиска путем изменения (определенного увеличения до попадания в локальный оптимум) размера популяций, количества генераций, вероятности применения генетических операторов, искусственной миграции

хромосом, направленной селекции и отбора. Отметим, что задача планирования также является NP-полной. Описанные эвристики помогают находить значительное число локальных решений, с большой вероятностью нахождения оптимального результата.

5.5. Генетические алгоритмы для трассировки, сжатия и верификации элементов СБИС

Заключительным этапом конструкторского синтеза является комплексная трудоемкая задача, определяющая качество всего процесса- трассировка соединений [37, 43, 50-52, 55, 59].

Задачу трассировки представляют в виде укрупненных четырех этапов: определение перечня соединений – размещение по слоям, определение порядка соединений – трассировка проводников [43].

Рассмотрим разработку алгоритмов первого этапа. Основными алгоритмами здесь являются алгоритмы построения кратчайших связывающих деревьев Прима и Штейнера. Дадим краткое описание указанных задач. Например, на рис. 5.23 показаны пять контактов цепи, которые должны быть соединены. Обычно ставится задача соединения заданных точек кратчайшим образом (сумма расстояний между точками должна быть минимальной или стремиться к ней). На рис.5.24 показано одно из возможных кратчайших связывающих деревьев (КСД), построенных на основе алгоритма Прима [43]. Условная суммарная длина L(T) КСД (рис.5.24) равна L(T) = 23.

Для минимизации суммарной длины соединений в КСД предложено при соединении множества вершин (точек, контактов цепей) использовать дополнительные точки (вершины). Возникающую при этом задачу называют задачей Штейнера [37, 43]. Она

формулируется следующим образом. Для заданных n точек плоскости построить соединяющее их КСД с n' ≥ n точками (вершинами), общая суммарная длина соединений которых минимальна. Построенное таким образом дерево называется деревом Штейнера (ДШ), дополнительные вершины – точками Штейнера (ТШ). На рис.5.25 показано одно из возможных ДШ для пяти соединяемых контактов (рис.5.23). Суммарная длина соединений ДШ (рис.5.25) L(G) = 19 и введено две ТШ (q_1 , q_2). Задача Штейнера, как известно [37, 43], является NP-полной. Для n = 3 известно точное решение задачи. Для n = 4 существует ряд эффективных алгоритмов [43]. При n ≥ 5 требуется огромное число вычислений, поэтому продолжается разработка различных эвристических алгоритмов решения данной задачи.

В [43] сформулирован ряд лемм и теорем о построении ДШ. Так, при соединении трех контактов единственная точка Штейнера находится внутри прямоугольника, связывающего эти точки. Причем координаты единственной ТШ определятся как s_{cp} и t_{cp} значения т, n_i = 1, 2, 3 и $\sum_{i=1}^{3} d(q,n_i) = \frac{1}{2}\Pi(R(n_1,n_2,n_3))$, где П[R]-длина периметра прямоугольника, построенного на трех точках. Так, для ДШ (рис.5.26,) q_1 и q_2 будут иметь координаты, определяемые следующим образом

$$S_1 = 0,\quad S_2 = 4,\quad S_3 = 2,\quad S_{cpq_q} = 2;$$
$$t_1 = 6,\quad t_2 = 3,\quad t_3 = 0,\quad t_{cpq_q} = 3;$$
$$S_2 = 4,\quad S_4 = 8,\quad S_5 = 10,\quad S_{cpq_2} = 8;$$
$$t_2 = 3,\quad t_4 = 5,\quad t_5 = 2,\quad t_{cpq_2} = 3;$$

Длина ДШ (рис.5.25.) определяется

$$L(T) = \frac{1}{2}\Pi_1(R(1,2,3)) + \frac{1}{2}\Pi_2(R(2,4,5)) = 19.$$

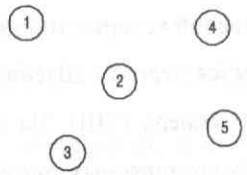


Рис.5.23. Пять контактов цепи

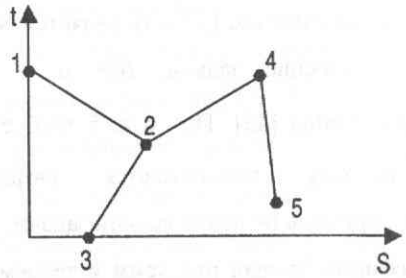


Рис.5.24. Кратчайшее связывающее дерево

L(T)=7+5+6+5=23

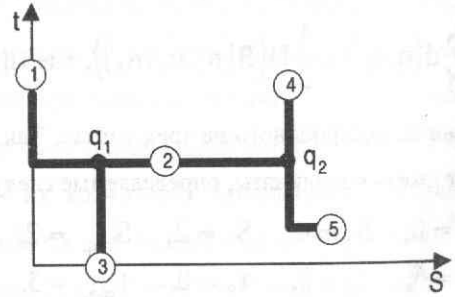


Рис.5.25 Дерево Штейнера, 2ТШ, L(T)=19.

Предложим простой эвристический алгоритм, ВСА которого $O(\alpha n)$. Идея построения алгоритма заключается в следующем.

Выбираются первые три точки из n точек, на которых должно быть построено ДШ. Согласно [43] строится оптимальный фрагмент ДШ для трех точек. Затем выбираются следующие по порядку три точки, одна из которых принадлежит построенному фрагменту. На этих точках строится второй оптимальный фрагмент ДШ. Далее процесс повторяется аналогично, пока не будет построено ДШ. Отметим, что каждый фрагмент ДШ оптимальный, но полное ДШ в общем случае может оказаться не оптимальным. При наличии достаточных вычислительных ресурсов или распараллеливания алгоритма можно строить оптимальные фрагменты по три вершины и, выполнив перебор на них, можно получить оптимальное или квазиоптимальное ДШ.

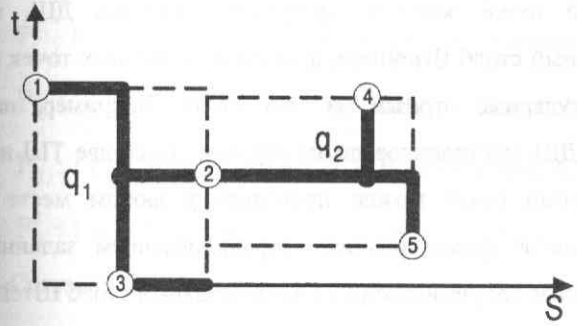


Рис.5.26. Дерево Штейнера, 2ТШ, L(T)=19.

Рис.5.27 ДШ, 2 ТШ, L(T)=22.

Рис.5.28 Дерево Штейнера 2 ТШ, L(T)=19

Предлагается упрощенная эвристическая процедура построения ДШ, основанная на так называемых «столбах» Штейнера. Из любой точки, которую требуется соединить ДШ, проводится вертикальный столб Штейнера, а затем из остальных точек проводятся перпендикулярные отрезки на этот столб. Например, на рис.5.27. показано ДШ, построенное таким образом. Здесь две ТШ и L(T) = 22. Вертикальный столб можно проводить в любом месте плоскости (желательно в прямоугольнике, ограничивающем заданные точки). Аналогичным образом строится горизонтальный столб Штейнера и ДШ с использованием описанной выше идеи. На рис.5.28. показано такое ДШ с двумя ТШ и L(T) = 19.

В связи с возможностью быстрого получения набора различных ДШ, по нашему мнению, представляется возможным использовать схемы генетического поиска, описанные выше, для построения прямоугольных ДШ.

Сначала, используя предложенные эвристические алгоритмы, строится популяция $P_I = \{P_1^I, P_2^I, ..., P_k^I\}$, каждый элемент популяции

представляет собой закодированное ДШ. Далее производится сортировка популяции и селекция родительских пар. Затем возможно применение любых из описанных выше ГО на основе мета, -микро или макроэволюции.

Рассмотрим пример. Пусть необходимо построить ДШ на пяти точках (рис.5.23). Используя горизонтальные столбы Штейнера, построим четыре ДШ (рис.5.29-5.34). На рис.5.29 столб проходит через точку 1; на рис.5.30 - через 4; на рис.5.31 - через 5; на рис.5.32 - через 3. Получили популяцию $P_I = \{P_1^I, P_2^I, P_3^I, P_4^I\}$. Кодирование каждой хромосомы в популяции будем проводить следующим образом. Отметим, что П означает движение прямо, Н - направо, В - влево.

P_1^I : 1 П T_1 Н 3 П T_2 Н 2 П T_3 Н 4 П Н 5 (1-столб)-1

P_2^I : 1 Н П T_1 Н 3 П T_2 Н 2 П 4 П Н 5 (2-столб)-4

P_3^I : 1 Н П T_1 Н 3 П T_2 В 2 П T_3 В 4 П 5 (3-столб)-5

P_4^I : 1 Н П 3 П T_1 В 2 П T_2 В 4 П В 5 (4-столб)-3

Согласно вычисленной суммарной длине выбранных деревьев Штейнера, соответствующих хромосомам P_1^I, P_2^I, P_3^I, P_4^I, произведем сортировку ДШ. Тогда, в результате получим - P_3^I, P_2^I, P_1^I P_4^I. Построим новую популяцию $P_{II} = \{P_1^{II}, P_2^{II}, P_3^{II}, P_4^{II}\}$, используя вертикальные столбы Штейнера. На рис.5.27, 5.33-5.35 показаны четыре ДШ. Произведя их сортировку, получим: P_2^{II} P_3^{II} P_4^I P_1^{II}. Выберем в качестве родительских пар элитные элементы из первой и второй популяций:

P_3^I : | 1 Н П T_1 Н З П | T_2 В 2 П T_3 | В 4 | П 5;

P_2^{II} : | 1 П Н T_1 П 4 Н | 2 Н T_2 П 5 | Н Л 3;

$P'_{3,2}$: | 1 Н П T_1 Н З П 2 Н T_2 П 5 В 4.

На примере выбранных родительских пар P_3^I и P_2^{II} покажем работу модифицированного оператора сегрегации. Он может выполняться на одной, двух и т.д. числе хромосом. Когда ОС выполняется на одной хромосоме, тогда точки сегрегации позволяют определить строительные блоки (части хромосомы) для формирования потомка. В этом случае как бы происходит перераспределение генетического материала внутри одной хромосомы. При реализации ОС на двух и более хромосомах точки ОС, выбранные направленно или случайно направленно, позволяют найти строительные блоки и скопировать их в новую хромосому или в некоторое множество хромосом (потомков). На рис.5.36 показано ДШ с двумя ТШ и L(T) = 20. После применения модифицированного оператора сегрегации здесь первый строительный блок позволил построить фрагмент, соединяющий точки 1, 3. Второй строительный блок – точки 2, 5 и третий строительный блок позволил подсоединить точку 4. Отметим, что мы специально не включили в популяцию P_I лучшее решение с горизонтальным столбом,

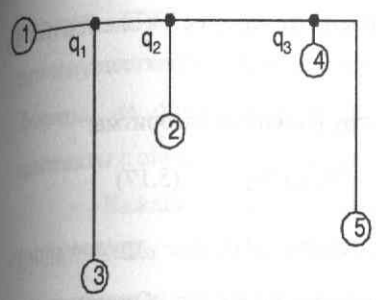


Рис.5.29. Дерево Штейнера L(T)=24 3ТШ.

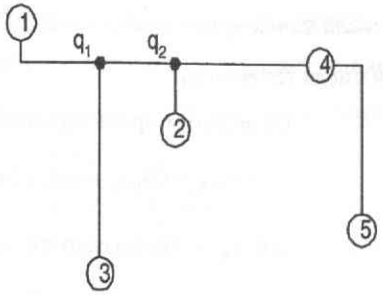


Рис. 5.30. Дерево Штейнера, L(T)=22, 2ТШ.

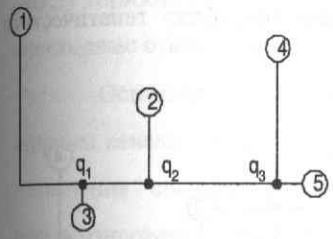


Рис.5.31. Дерево Штейнера L(T)=20, 3ТШ.

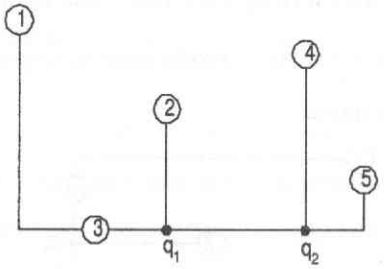


Рис.5.32. Дерево Штейнера L(T)=26, 2ТШ.

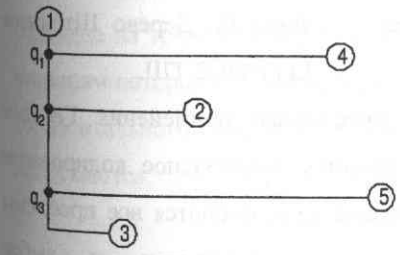


Рис.5.33. Дерево Штейнера L(T)=30, 3ТШ.

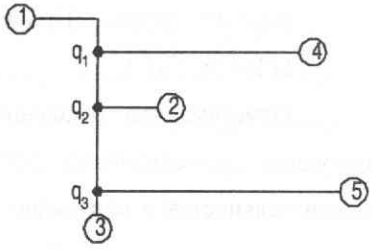


Рис.5.34. Дерево Штейнера L(T)=24, 2ТШ.

проходящим через точку 2. При включении этого элемента в популяцию было бы получено другое ДШ с L(T) = 19. С увеличением

числа точек время построения ДШ зависит от количества элементов P_I и числа генераций.

Определим ориентировочную трудоемкость алгоритма:

$$T \approx t_k + [N_p\, t_п + (N_p + N_п)\, t_{oc} + N_p\, t_c]\, N_G, \qquad (5.17)$$

где t_k - трудоемкость кодирования ДШ, $t_п$ - трудоемкость получения одного потомка, N_p-количество потомков. Отметим, что трудоемкость получения одного потомка меняется в пределах $O(n)$-$O(n^2)$ и ВСА для разработанных эвристических алгоритмов ДШ $\approx O(\alpha\, n^2)$, где α - коэффициент, определяющий специфику генетического поиска.

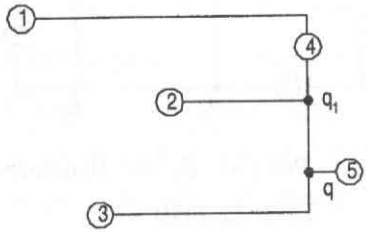


Рис.5.35. Дерево Штейнера L(T)=26, 2 ТШ.

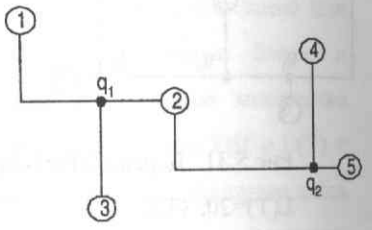


Рис.5.36. Дерево Штейнера L(T)=20, 2 ТШ.

Отметим, что основной сложностью применения ГА для построения прямолинейных ДШ является оптимальное кодирование индивидуальностей в популяции. Кроме того, остаются все проблемы селекции, предварительной сходимости алгоритма и выбор эффективных ГО.

Рассмотрим теперь этап прокладки соединений на основе канальных алгоритмов трассировки. Канальные алгоритмы базируются на представлении о каналах и магистралях. Магистралью называют

отрезок прямой, по которой может проходить соединение в преимущественном направлении. Канал – это область прямоугольной формы, на одной или нескольких сторонах которой расположены контакты с системой однонаправленных магистралей.

Каждая цепь, т.е. соединение эквипотенциальных контактов, представлена как одиночный горизонтальный сегмент с несколькими вертикальными сегментами, которые соединяют горизонтальный сегмент с контактами цепи [37, 59]. Горизонтальные сегменты располагаются в одном слое, вертикальные – в другом. Соединения между горизонтальными и вертикальными сегментами делаются через переходные отверстия.

Основная задача канальной трассировки – выбор наименьшей ширины канала, достаточной для размещения в нем всех соединений и назначения соединений на магистрали. Кроме того, необходимо минимизировать суммарную длину соединений (цепей), число переходных отверстий и т.п.

Задача канальной трассировки в классической постановке основана на трассировке двухстороннего канала, по верхней и нижней сторонам которого проходят линейки контактов. Изломы, т.е. переходы горизонтального участка с одной магистрали на другую, не допускаются.

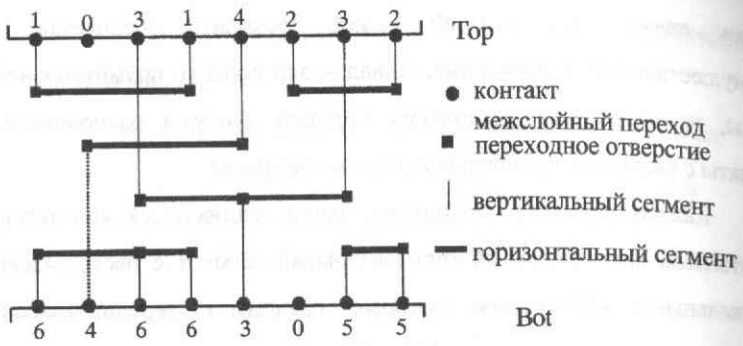


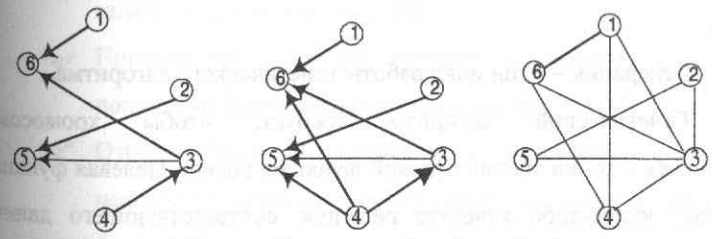
Рис. 5.37.

Канал описывается двумя последовательностями Top и Bottom, в которых размещены верхняя и нижняя линейки канала, соответственно [37]. Размер обоих последовательностей равен C,-числу колонок в канале. Множество цепей определяется как Net={N_1, ..., N_n}, где n-число цепей. Рассмотрим следующий пример: Top ={1, 0, 3, 1, 4, 2, 3, 2} и Bottom ={6, 4, 6, 6, 3, 0, 5,5}, где C=8, n=6, Net={1, 2, 3, 4, 5, 6}, элемент 0 в Top или в Bottom обозначает свободный контакт. На Рис. 5.37 дан эскиз канала с разведенными цепями.

При канальной трассировке не допускаются наложения вертикальных и горизонтальных сегментов цепей. Для решения этой задачи вводятся графы вертикальных и горизонтальных ограничений. Вертикальные ограничения описываются ориентированным графом вертикальных ограничений (ГВО) G_V=(E_{Net}, E_V), где Net-множество вершин, соответствующих множеству цепей Net, E_V-множество направленных ребер. Ребро (n, m) $\in E_V$, существует тогда и только тогда, когда цепь n должна быть расположена выше цепи m для предотвращения наложений вертикальных сегментов цепей. Например,

на рис. 5.38 в ГВО существует путь из вершины 1 в вершину 6, это означает что цепь 1 должна быть размещена выше цепи 6, для того чтобы не было наложений вертикальных сегментов на 1 и 4 контактах (см. рис. 5.37). Для того чтобы задача решалась в рамках классической постановки, граф G_V должен быть ациклическим, в противном случае задача может быть решена только с введением изломов, а это противоречит условиям классической постановки задачи канальной трассировки.

Будем использовать расширенный граф вертикальных ограничений (РГВО) G_{RV} = (E_{Net}, E_{RV}), где Net-множество вершин, соответствующих множеству цепей Net, E_{RV}-множество направленных ребер. Ребро (n, m) $\in E_{RV}$ существует тогда и только тогда, когда в ГВО существует путь из вершины n в вершину m. Например, на рис. 5.38, в ГВО есть путь из вершины 4 в вершину 5 через вершину 3. Следовательно, в РГВО существует ребро (4,5).



Граф вертикальных ограничений | Расширенный граф вертикальных ограничений | Граф горизонтальных ограничений

Рис. 5.38.

Горизонтальные ограничения представлены неориентированным графом горизонтальных ограничений (ГГО) G_H = (E_{Net},E_H), где E_{Net}-множество цепей, E_H- множество ребер.

Ребро (n, m) $\in E_H$ существует тогда и только тогда, когда магистрали для n и m должны быть разными для исключения наложения горизонтальных сегментов n и m. Например, на рис. 5.37 в ГГО существует ребро (1,4), это означает, что цепь 1 не может размещаться на одной магистрали с цепью 4 (см. рис. 5.37).

Приведем соответствие терминов ГА и классической задачи канальной трассировки.

Фенотип – топология расположения цепей на кристалле.

Генотип – генетическая схема кодирования топологии.

Хромосома-кодированное представление одного варианта топологии. Хромосома состоит из генов.

Ген-элемент хромосомы, задающий некоторый фрагмент топологии.

Популяция-набор хромосом (закодированных решений задачи трассировки).

Фитнесс – целевая функция, определяющая качество решения задачи.

Генерация – один цикл работы генетического алгоритма.

Генетический алгоритм требует, чтобы хромосомы оценивались с точки зрения целевой функции задачи. Целевая функция оценивает какие-либо качества решения, соответствующего данной хромосоме (длину цепей, число магистралей и т.д.). Как было указано выше, генетический алгоритм обрабатывает популяцию решений, закодированных в хромосомы. В процессе обработки популяции к ней последовательно применяются различные генетические операторы, такие как кроссинговер, мутация с заданными вероятностями (P(OK) и

P(OM) соответственно) и другие операторы. Затем проводится редукция увеличившейся популяции для отбора лучших решений, которые составят следующее поколение, после чего цикл (генерация) повторяется. Отметим, что может быть построено большое число архитектур реализации генетического поиска. В нашем подходе выбрана стандартная схема генетического поиска.

Для генетического алгоритма трассировки принята следующая схема:

1° Определение размера популяции M, числа генераций T, вероятности кроссинговера P(OK) и вероятности мутации P(OM).

2° Задание случайным образом начальной популяции П(0) размером M.

3° t=0 (t = 1,2, ..., T).

4° Выбор случайным образом M пар хромосом из популяции П(t) и применение операции кроссинговера к каждой паре с заданной вероятностью P(OK).

5° Применение операции мутации к каждой хромосоме популяции П(t) с заданной вероятностью P(OM).

6° Отбор. Отбор M хромосом с наилучшим значением целевой функции из получившейся популяции П(t) в новую популяцию П(t+1).

7° t=t+1.

8° Если t<T, то переход к шагу 4.

9° Вывод хромосомы с наилучшим значением целевой функции.

Здесь используется представление хромосомы, основанной на топологическом описании канала, т.е. хромосома описывает взаимное расположение цепей. Для этого каждой паре цепей (m,n), где m ≠ n, ставится в соответствие ген, который может принимать три состояния: 0-если цепь m должна располагаться выше n; 1-если цепь m должна располагаться ниже n; и * - если их взаимное расположение не имеет значения или определяется из взаимного расположения остальных цепей (это происходит, если цепи не имеют горизонтальных ограничений). Для примера, изображенного на рис. 5.37, хромосома будет иметь следующий вид:

Цепь m	1	1	1	1	1	2	2	2	2	3	3	3	4	4	5
Цепь n	2	3	4	5	6	3	4	5	6	4	5	6	5	6	6
Ген	*	0	0	*	0	0	*	0	*	1	0	0	*	0	*

где в строке таблицы, обозначенной как "ген", записано значение гена в хромосоме, задающей расположение, показанное на рис. 5.37. Очевидно, что длина хромосомы при таком кодировании будет равна

$$L = \sum_{i=0}^{N-1} i \qquad (5.18)$$

где N-число цепей.

Как видно, длина хромосомы получается достаточно большой (для примера на рис. 5.37 она равна 15), что не приемлемо. Поэтому для уменьшения длины хромосомы используется анализ РГВО и ГГО. Уменьшение длины получается за счет того, что взаимное расположения некоторых пар цепей уже зафиксировано в РГВО, и

изменение этого расположения приводит к образованию нарушений в канале, т.е. к наложению вертикальных сегментов цепей в канале, что ведет к образованию «мертвой» хромосомы (т.е. такой, из которой не может быть получено решение, удовлетворяющее заданным требованиям). Второй прием, позволяющий уменьшить длину хромосомы, основывается на том, что если цепи не имеют горизонтальных ограничений, то их взаимное положение либо не важно, либо определяется из соотношений с остальными цепями. Такие соотношения отмечены знаком "*". Следовательно, длина хромосомы будет равна $L=N_{GO}-N_{RVO}$, где N_{GO}-число горизонтальных ограничений, N_{RVO}-число вертикальных ограничений в РГВО. В примере для пар цепей (1,6), (2,5), (3,4), (3,5), (3,6), (4,6) взаимное расположение жестко задано РГВО, а взаимное расположение цепей в парах (1,2), (1,5), (2,4), (2,6), (4,5), (5,6) не имеет значения, поскольку цепи в этих парах не конфликтуют по горизонтали. Поэтому в примере хромосома будет выглядеть следующим образом:

Цепь m	1	1	2
Цепь n	3	4	3
Ген	0	0	0

длина хромосомы будет равна 3.

Для получения из хромосомы эскиза канала с разведенными цепями используется граф топологии (ГТ). ГТ – это ориентированный граф

G_T = (Net, E_T), где Net-множество цепей, E_T–множество направленных ребер, описывающих взаимное расположение цепей в канале. Ребро (m, n) $\in E_T$, существует тогда и только тогда, когда цепь m должна быть

расположена в канале выше цепи n, т.е. на магистрали с меньшим номером. ГТ строится из хромосомы по следующему алгоритму:

1° Шаг 1. Преобразуем РГВО G_{RV} в ГТ G_T.

2° Шаг 2. i=1.

3° Шаг 3. Если ребро (m_i , n_i) не принадлежит G_T , то это ребро добавляется к G_T.

4° Шаг 4. Проверяем для всех вершин k существование пути из вершины m_i к вершине k, при условии, что k = 1, ... , N, k $\neq m_i$ и $k \neq n_i$ и не существует ребра (m_i, k) в G_T. существование пути из вершины m_i к вершине k. Если такой путь существует, то добавляем ребро (m_i, k) к G_T.

5° Шаг 5. Если $i \leq L$, то увеличиваем i на 1 и переходим к шагу 3.

6° Шаг 6. Считаем построение ГТ завершенным.

Канал восстанавливается из хромосомы следующим образом:

1° Строим по хромосоме ГТ G_T.

2° i=0.

3° Находим вершины, у которых есть только исходящие дуги. Размещаем их на магистрали с номером i или на магистрали с меньшим номером, если это возможно, и удаляем эти вершины из графа.

4° i = i + 1.

5° Если граф топологии G_T не пустой, возвращаемся к шагу 3.

6° Не нарушая взаимного расположения, смещаем цепи по магистралям так, чтобы минимизировать длину вертикальных сегментов цепей.

7° Возвращаем полученный образец размещения цепей в канале шириной i.

Например, пусть задана хромосома A = 000. Рис. 5.37. показывает полученный образец трассировки, который является оптимальным решением для примера. Если A = 010 (отличие от предыдущего решения заключается в том, что цепь 4 располагается выше цепи 1), то эскиз канала с разведенными цепями для такой хромосомы показан на рис. 5.39.

Когда генетический алгоритм применяется для решения оптимизационных задач, важнейшим требованием является разработка целевой функции (ЦФ) для оценки полученных решений. Генетический алгоритм обычно используют для максимизации (минимизации), но желательно, чтобы целевая функция была положительной. В нашем случае задача заключается в минимизации целевой функции. Принимая во внимание характеристики задачи канальной трассировки, целевая функция определяется следующим образом:

$$F(A)=(UsedTrack(A)+2)*C+TotalVertSeg(A), \qquad (5.19)$$

где C-число контактов, программная функция UsedTrack(A) возвращает число магистралей, занимаемых каналом, полученным при восстановлении из хромосомы A, а программная функция TotalVertSeg(A) возвращает длину вертикальных сегментов цепей в полученном решении. Длина вертикальных сегментов цепи определяется как расстояние между контактами и переходными отверстиями, которые соединяют вертикальные и горизонтальные

сегменты. Например, для канала на рис. 5.37 число занятых магистралей равно 4, длина вертикальных сегментов равна 22. Таким образом, целевая функция хромосомы будет равна F(A)=(4+2)*8+22=70, а для канала, показанного на рис. 5.39, F(A)=72.

Данная методика определения целевой функции направлена, в первую очередь, на минимизацию ширины канала и во вторую – на минимизацию суммарной длины соединений.

Согласно структуре генетического алгоритма все хромосомы (случайно полученные решения) оцениваются описанной выше целевой функцией. Далее согласно процедуре селекции производится построение списка упорядоченных хромосом. На основе анализа этого списка производится подбор пар хромосом с использованием элитной селекции (хромосома с лучшим значением ЦФ и случайно выбранная хромосома). После подбора пар применяется ОК с вероятностью P(OK) к каждой паре хромосом. Алгоритм применения ОК реализован следующим образом:

1° i=0.

2° Выбрать хромосому с лучшим значением ЦФ как первого родителя.

3° Выбрать произвольно второго родителя.

4° Сгенерировать случайное число rnd в интервале [0,1].

5° Если rnd < P(OK), то применить ОК к выбранным хромосомам и добавить получившихся потомков к популяции.

6° i = i + 1.

7° Если i < M, то перейти к шагу 2.

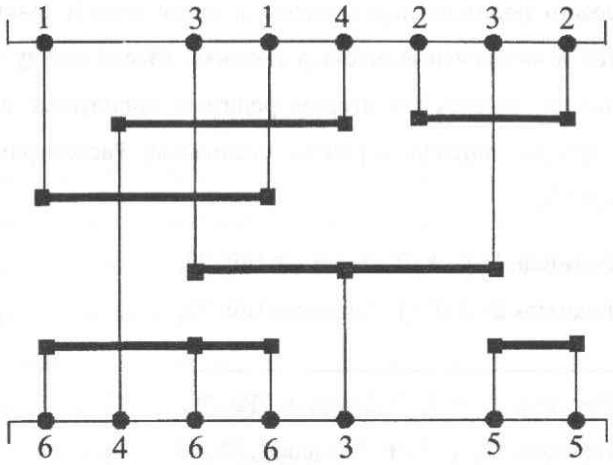


Рис. 5.39.

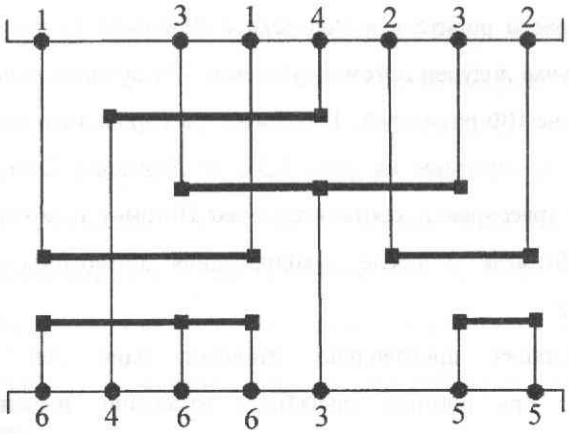


Рис. 5.40.

Под ОК подразумевается обмен генетического материала между двумя различными хромосомами (потенциальными решениями). После окончания ОК полученные потомки подвергаются мутации. В нашем алгоритме после анализа выбран стандартный двухточечный ОК. Первая и вторая точка разреза в этом ОК выбираются случайно.

Часть первого родителя перед первой и после второй точки разреза копируется в аналогичные места в потомке. Место между первой и второй точкой разреза во втором родителе копируется в первого потомка. Второй потомок строится аналогично. Рассмотрим пример реализации ОК.

Родитель 1: 0 | 1 | 0 Значение ЦФ: 72,

Родитель 2: 1 | 0 | 1 Значение ЦФ: 77,

Потомок 1: 0 | 0 | 0 Значение ЦФ: 70,

Потомок 2: 1 | 1 | 1 Значение ЦФ: 77.

Потомок 1 (Потомок 2) получен с учетом Родителя 1 (Родителя 2), как первого родителя и Родителя 2 (Родителя 1) как второго. В данном случае получен потомок (Потомок 1) с лучшим значением ЦФ, чем значение ЦФ родителей. Топология трассировки соответствующая Родителю 1, показана на рис. 5.39, и Родителю 2-на рис. 5.40. Топология трассировки, соответствующая Потомку 1, показана на рис. 5.37, а Потомок 2 после декодирования хромосомы аналогичен Родителю 2.

Мутация произвольно изменяет один ген выбранной хромосомы при помощи случайного изменения значения гена с вероятностью P(OM), равной норме мутации. Алгоритм применения OM выглядит следующим образом:

1° i=0.

2° Сгенерировать случайное число rnd в интервале [0, 1].

3° Если rnd < P(OM), то применить OM к i-й хромосоме популяции M.

4° i = i + 1.

5° Если i < M, то перейти к шагу 2.

Смысл оператора мутации – это введение добавочных изменений в популяцию. В нашей структуре ГА выбрана "точечная" мутация. Она заключается в изменении одного произвольного гена в хромосоме с вероятностью P(OM). Это происходит следующим образом. Один из генов хромосомы выбирается случайно и затем меняет свое значение на противоположное. Следующий пример показывает реализацию этого механизма:

Хромосома (до применения OM): 0 (1) 0 Значение ЦФ: 72,

Хромосома (после применения OM): 0 (0) 0 Значение ЦФ: 70,

выбранная точка мутации показана в скобках. После применения OM в данном случае, получается хромосома с лучшим значением ЦФ. На рис. 5.39 показана топология канала, соответствующего хромосоме до применения OM, а на рис. 5.37-после применения OM.

Теоретическая временная сложность алгоритма составляет $T(MP(OK)2)P(OM)O(N^2)$, где T – число поколений, M – размер популяции, P(OK) – вероятности кроссинговера, P(OM) – вероятность мутации, $O(N^2)$ – временная сложность декодирования хромосомы, N – число цепей.

После выполнения трассировки соединений решаются вопросы сжатия топологии СБИС и верификации. Сжатие топологии есть заключительная процедура фазы детальной трассировки, когда она транслируется в топологию таким образом, чтобы конвертировать

элементы электрических схем в соответствующие элементы топологии и минимизировать площадь кристалла [37, 43].

Проектировщики сталкиваются с тремя противоречивыми требованиями:

- спроектировать минимизированную по площади топологию;
- максимизировать функциональные возможности СБИС;
- минимизировать время и стоимость проектирования.

Проблема сжатия топологии связана с методологиями топологического проектирования (размещение компонентов и проводников) и геометрического проектирования (физическое размещение элементов топологии).

Задачи сжатия обычно решаются на основе символической топологии (simbolic layout (SL)). Символическая топология дает возможность проектировщику анализировать такие компоненты СБИС, как транзисторы, проводники, ячейки и т.п. SL обычно транслируется в топологию в виде прямоугольников различного размера. В САПР СБИС SL представляется в виде стиковых диаграмм (СД), представляющих проводники и элементы как различного рода компоненты. Следовательно, методы сжатия на основе SL и СД меняют геометрию топологии с минимизацией площади кристалла.

В работе [55] дан детальный обзор и классификация методов сжатия. Проблемы сжатия, как и остальные проблемы проектирования топологии, относятся к классу комбинаторно-логических задач и являются NP-полными.

Алгоритмы сжатия обычно классифицируются по двум направлениям. Первое направление показывает, как движутся элементы

в процессе сжатия. К этому классу относятся алгоритмы одноразмерного сжатия (1-Д), двухразмерного (2-Д), трехразмерного (3-Д) и специального сжатия типа (1,5-Д). Сжатие типа 1-Д обычно выполняется только в s или t направлениях, как показано на рис.5.41. Проблема 1-Д сжатия NP-полная. В 2-Д сжатие происходит одновременно по осям s и t.

Второе направление показывает, каким образом производится размещение компонентов. Здесь выделяют три класса алгоритмов. Это алгоритмы сжатия на основе графов ограничений, виртуально решетчатые алгоритмы и алгоритмы иерархического сжатия. Основными алгоритмами здесь являются: алгоритмы сжатия на основе графа прямоугольников [55], алгоритм Айкерса или алгоритм "тени" [55].

Иерархические алгоритмы заключаются в многоуровневом сжатии. Элементы топологии, подлежащие сжатию, обычно описываются в виде прямоугольников: $ЭТ_i = (w_i, h_i, s_i, t_i, m_i)$, здесь w_i-ширина прямоугольника; h_i-высота; s_i, t_i-декартовы координаты; m_i-номер элемента ($1 \le w_i \le m$).

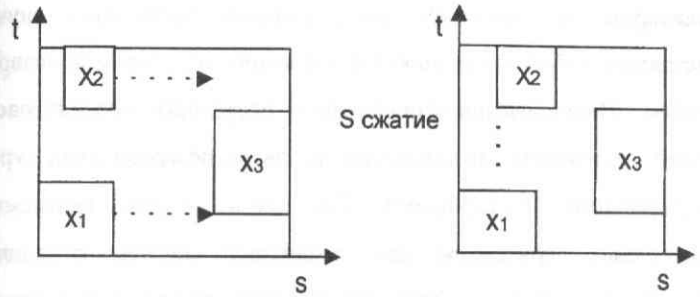
Расстояния между элементами на плоскости измеряются по формулам приведенным выше для определения d_{ij}. Задача состоит в размещении прямоугольников таким образом, чтобы они не пересекались и площадь, где они расположены, была минимизирована.

В такой постановке задачи сжатия перекликаются с задачами планирования кристалла, а также с задачами упаковки и разбиения. В работе [55] приведен обзор всех методов упаковки и разрезания, в котором показана общность задач упаковки и сжатия. В [39, 55]

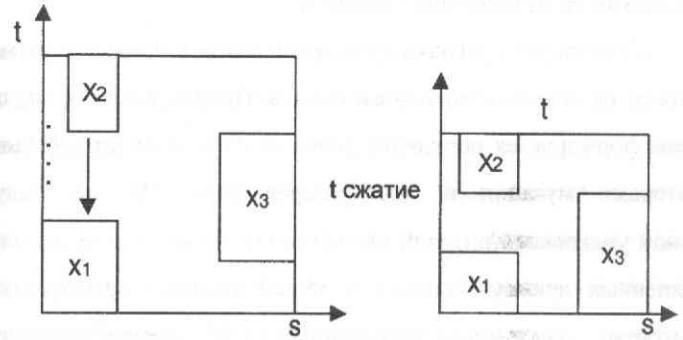
приведены существующие эвристические методы решения задач упаковки. Основными из них являются:

- метод первого встречного (FF);
- метод первого встречного после сортировки (FFD);
- метод выбора наилучшего (BF);
- метод выбора наилучшего после сортировки (BFB);
- комбинированные метод (RFF, RFFD, RBF, RBFD).

Дадим постановку задач методов FF, AFFD. Считается, что задан список элементов $R = (x_1, x_2, ..., x_n)$. Алгоритм FF назначает последовательно элементы x_j ($j = 1, 2, ..., n$) в заданную плоскость (кристалл, печатную плату, блок и т.п.). Алгоритм FFD предварительно сортирует элементы в уменьшающемся или увеличивающемся порядке (с точки зрения размеров или заданного критерия), а затем выполняет FF. Известно, что ВСА FF $\approx O(\alpha n)$, а ВСА FFD $\approx O(n \log n)$.



а) сжатие топологии по оси S



б) сжатие топологии по оси t

Рис.5.41. Пример I-Д сжатия топологии

Существующие методы упаковки и сжатия на основе стохастических методов, моделирования отжига и генетического поиска используют в качестве исходных описанные эвристики.

В работе [55] показана возможность сжатия символической топологии с применением генетического алгоритма. Сжатие представляется как задача 2-Д упаковки. Множество прямоугольников размещается на плоскости, удовлетворяя заданным ограничениям и минимизируя стоимостную функцию площади кристалла. Прямоугольники размещаются на плоскости согласно списку

элементарных ограничений. Набор списков, полученных случайно, представляет собой популяцию ГА. Селекция популяции производится случайно. Выполняются стандартные операторы кроссинговера и мутации. Алгоритм выполняется в двух иерархических уровнях (поверхностный и глубокий). ГА здесь – это программа с селекционным критерием для сравнения списков ограничений. Достоинством алгоритма является простота реализации. Недостатки связаны с предварительной сходимостью алгоритма и невысокими результатами по минимизации площади.

Существует упаковка прямоугольников различного размера на плоскости на основе адаптивного поиска. Предварительно, случайным образом формируется популяция решений. Она затем модифицируется операторами мутации и модифицированным ОК для получения реальной упаковки. Критерий оптимизации – это отношение площади размещенных прямоугольников к общей площади прямоугольников. Предложено два модифицированных ОМ («карабкающаяся» и «щелчковая»). ГА заключается в преобразовании списка размещений. Основной недостаток алгоритма заключается в том, что каждое исследование производится с новой упаковкой и отсутствуют итерационные процедуры улучшения. ВСА $\approx O(n^3)$.

В [55] описано 2-Д сжатие топологии на основе моделирования отжига. Алгоритм сначала уменьшает размер исследуемой поверхности, а затем на основе моделирования отжига выбирает множество ограничений. Основной недостаток состоит в том, что начальное решение выбирается случайно, а оно в основном влияет на

конечный результат. Сложность алгоритма лежит в пределах $O(n^2)$-$O(n^4)$.

В работе [55] описано решение задачи упаковки 2-Д на основе ГА. Целевая функция определяется как максимум средней заполняемости заданной плоскости. Достоинством алгоритма является введение нового ОК, ориентированного на решение задач упаковки. Данный оператор кроссинговера напоминает ОС, выполненный направленным образом.

В [39] рассмотрен стохастический метод для решения 1-Д упаковки. Он представляет собой объединение генетического поиска с процедурами моделирования отжига. Основная идея этого метода-повышение скорости моделирования отжига за счет использования простых, стандартных операторов ГА. Для получения популяции решений используются две стратегии – «жадная» и случайная. Основной недостаток – требование больших затрат на вычислительные процедуры отжига, а также быстрая сходимость алгоритма с попаданием в локальный оптимум.

Предлагается иерархическая процедура решения задач сжатия топологии с использованием параллельного генетического поиска. Опишем кратко схему генетического поиска. Как и во всех описанных ранее алгоритмах, первоначально производится конструирование, но не одной, а некоторого множества (четырех) популяций. В качестве элемента популяции выбирается порядок (реальный) расположения фрагментов топологии и соединений на плоскости. Причем каждое соединение также представляется прямоугольником, как и размещаемый элемент. Другими словами, каждая популяция будет представлять собой набор списков размещенных элементов и их

соединений. Первая популяция будет конструироваться случайным образом, вторая на основе «жадной» стратегии [31], третья – на основе последовательных эвристик и четвертая популяция – смешанная (направленно-случайная). В четвертой популяции половина элементов выбирается случайно, а вторая – используя знания о решаемой задачи. Согласно схеме, описанной выше, производим сортировку элементов популяций P_I-P_{IV} согласно выбранной целевой функции. ЦФ выбрана, как стоимостная функция, максимизирующая среднюю площадь, занятую компонентами топологии и их соединениями. Производится элитная селекция родителей из каждой популяции. Далее применяются три ОК. Операторы ОК применяются таким образом, чтобы каждый раз получались легальные решения. После выполнения ОК производится сортировка и устранение такого числа наихудших элементов популяций, чтобы размер популяций оставался постоянным. При получении удовлетворительных результатов процесс завершается. В противном случае возможно управление процессом поиска за счет перехода на другой оператор или изменения вероятности применения оператора. Данные процедуры выполняются итерационно и число генераций зависит от наличия вычислительных ресурсов. Трудоемкость алгоритма ориентировочно определена как

$$T = N_G\,[N_p\,t_p + (t_{ok} + t_{ом} + t_{от} + t_{oc} + t_o)\,N_П].$$

ВСА лежит в пределах $O(n \log n)$-$O(n^3)$.

Заключительным этапом конструкторского синтеза СБИС является верификация. Одной из важнейших задач верификации является установление взаимно-однозначного соответствия между структурной и электрической схемой СБИС и ее топологической

реализации [37, 43]. Данные задачи решаются путем установления изоморфизма между графовой моделью электрической схемы и графовой моделью ее топологии. Данная проблема, как и все остальные рассмотренные выше проблемы конструкторского синтеза, является NP-полной. Поэтому разрабатываются различные эвристики для получения приемлимых практических результатов. В [43] описаны эффективные методы решения задачи установления изоморфизма графов. Наибольшую трудоемкость представляет установление изоморфизма однородных графов, имеющих автоморфные подграфы. Для решения таких задач используются методы разбиения исследуемых графов на различные уровни. При этом ВСА алгоритма снижается с $O(n!)$ до $O(k!)$, где n-число элементов в графе, а k-число элементов в наибольшем автоморфном подграфе. Основная идея таких алгоритмов заключается в следующем [43]. В обоих графах выбираются две предполагаемо изоморфные вершины. Относительно их производится разбиение всех оставшихся вершин на два класса. В первый класс включаются вершины, смежные предполагаемым изоморфным, а во второй нет. Например, на рис.5.42 показаны два графа G и G': $G = (X, U)$, $G' = (X', U')$, $|X| = |X'| = 6$, $|U| = |U'| = 9$, локальные степени всех вершин графа равны трем. Необходимо установить изоморфизм графов G и G'. Другими словами, необходимо определить, существует ли отношение эквивалентности

$X \Leftrightarrow X'$, $U \Leftrightarrow U'$ такое, что,

если ребро $(x_i, x_j) \Leftrightarrow$ ребру (x'_i, x'_j),

то $x_i \Leftrightarrow x'_i$, $x_j \Leftrightarrow x'_j$, x_i, x_j, X, $x'_i, x'_j, \in X'$.

Выберем одну вершину в графе G и одну в графе G'. И относительно них выполним указанное выше разбиение:

$\{1\}$ $\{2\}, 4, 6\}_{1+}$ $\{3, 5\}_{1-}$ (G),

$\{1'\}$ $\{4',5',6'\}_{1'+}$ $\{2',3'\}_{1'-}$ (G').

В данном разбиении вершины x_2, x_4 , x_6 смежны вершине x_1 , а вершины, x_3, x_5 не смежны вершине x_1 графа G.

Аналогичное разбиение выполнено для графа G'. Здесь считается, что вершины x_1 и x'_1 предполагаемо изоморфны (П-изоморфны). Далее выбираются подмножества меньшей мощности и внутри их проверяется смежность вершин. В нашем примере вершины x_3, x_5 и x'_2, x'_3 не смежны. Поэтому процесс разбиения продолжается

$\{1\}$ $\{3\}_{1-}$ $\{\{2, 4, 6\}_{3+}\}_{1+}$ $\{\{5\}_{3-}\}_{1-}$,

$\{1'\}$ $\{2'\}_{1'-}$ $\{\{4', 5', 6'\}_{2'+}\}_{1'+}$ $\{\{3'\}_{2'-}\}_{1'-}$.

Продолжая процесс аналогично, чему получим, что граф G изоморфен графу G'. Подстановка вершин запишется так:

$$t = \begin{Bmatrix} 1 & 2 & 4 & 6 & 3 & 5 \\ 1' & 4' & 5' & 6' & 2' & 3' \end{Bmatrix}.$$

В рассматриваемом примере подмножества вершин $\{X_2, X_4, X_6\}$ и $\{X'_4, X'_5, X'_6\}$ являются автморфными, т.е. каждая вершина в одном подграфе может быть изоморфна любой вершине второго подграфа. В примере ВСА O(6!) была сведена к ВСА O(3!). В алгоритмах такого типа необходим перебор между элементами автоморфных подграфов. Причем, как следует из рассмотрения примера, такой перебор здесь выполняется последовательно для всех вершин, кроме последней в подграфе. В той связи применение

генетического поиска позволяет повысить скорость установления изоморфизма или неизоморфизма графов.

Предлагается, после получения разбиения, в подмножествах наибольшей мощности выполнять генетические операторы мутации и инверсии. Это позволяет параллельно анализировать все подмножество автоморфных вершин и повышает время нахождения результата.

Например, на рис.5.43 показан нетривиальный (трудный) граф для установления изоморфизма. Необходимо установить, существует ли соответствие между подмножествами автоморфных вершин:

2 3 4 5,

b c d e

В описанном выше алгоритме необходимо выполнить перебор 4!. Применяя, например, оператор мутации, получим

2 4 3 5,

b c d e.

Графы G и G' (рис.5.42) изоморфны, а подстановка изоморфизма запишется так:

$$t = \begin{Bmatrix} 1 & 2 & 3 & 4 & 5 & 6 & 7 \\ a & b & d & c & e & f & g \end{Bmatrix}.$$

Следует отметить, что в графах большой размерности с нетривиальными автоморфизмами (K > 100) процесс установления изоморфизма резко усложняется, но параллельное выполнение генетических операторов на порядок снижает ВСА.

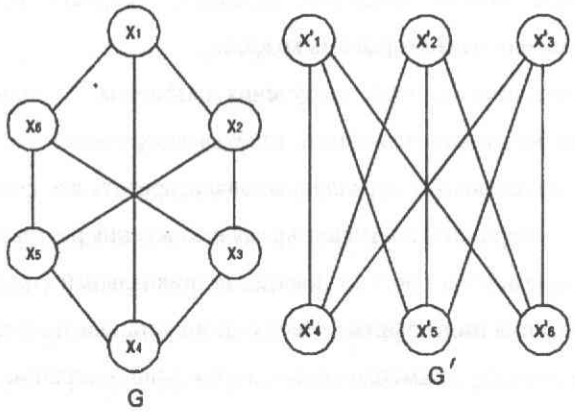


Рис.5.42. Пример графов G и G'

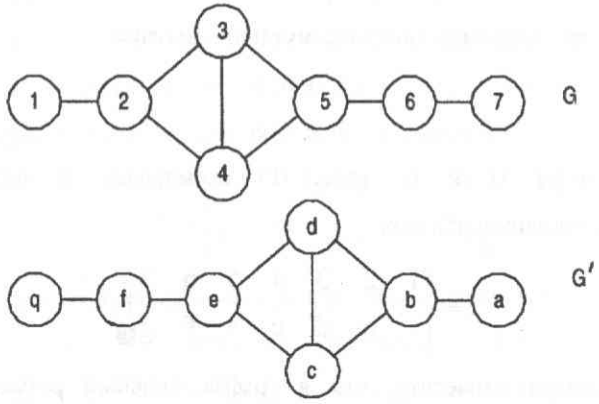


Рис5.43. Пример нетривиальных графов

ГА могут использоваться при решении различных задач диагностики на различных этапах автоматизированного проектирования СБИС. Так, в [30] предложено создание имитизационных тестов для последующего анализа цепей СБИС на

основе ГА. Методика основана на непрерывной мутации данной входной последовательности и анализе полученных векторов для выбора конечного теста. Для получения высокого качества теста для СБИС необходимо хранить сведения о некоторых полезных частях теста для последующего использования. Реализация ГА в [30] начинается с начальной популяции возможных решений. Она играет центральную роль в поиске теста, так как управляет наследственностью и характеристиками следующих генераций. Мутация является основным шагом, который вводит изменение, улучшающие решение. После формирования начальной популяции вычисляется целевая функция (ЦФ). Она определяет дефект на основе проверки логики работы цепи в различных ее узлах. Для этого исследуемая цепь разбивается на части (подсхемы). ГА балансирует размеры подсхем, улучшая ЦФ. Строится модель цепи как взаимосвязь смешанных транзисторов, вентелей и функциональных блоков. Новые векторы генерируются из старых. ГА выбирает один или два объекта из популяции, на которых выполняется мутация. Видоизменения объекта, в зависимости от порога ЦФ, сохраняются или отвергаются.

После мутации выполняется скрещивание (кроссинговер). Он выбирает два теста (хромосомы) A_1, A_2 и строит новый тест $A_к$, объединяющий наилучшие «хорошие» части A_1, A_2 с точки зрения ЦФ. Если они перекрываются, то вместо общих частей из A_1, A_2 копируются неповторяющиеся гены. Тестовые последовательности генерируются случайно, селекция выполняется случайно-направленным образом, блок анализа тестовых последовательностей управляет процессом генетического поиска. Генерацию векторных тестов можно выполнять как для комбинационных, так и для последовательных цепей СБИС. К

сожалению, описанная методика не позволяет идентифицировать истинность и не может генерировать тест для определенных дефектов. Перспективными исследованиями в этой области являются объединение генетического поиска с традиционными детерминированными методами генерации тестов.

5.6. Контрольные вопросы

1. Приведите архитектуру ИСАПР.
2. Опишите блок эволюционной адаптации.
3. Опишите генетический алгоритм дихотомического разбиения графа.
4. Приведите генетический алгоритм последовательной компоновки СБИС.
5. Постройте структурную схему генетического алгоритма итерационной компоновки элементов.
6. Какие особенности имеют генетические алгоритмы компоновки элементов СБИС?
7. Опишите генетический алгоритм линейного размещения элементов.
8. Постройте структурную схему генетического алгоритма для размещения элементов СБИС на основе разбиения графа.
9. Приведите основные эвристики алгоритма совместного разбиения и размещения элементов СБИС путем генетического поиска.
10. Какие схемы генетического поиска целесообразно применять при решении задач САПР?
11. Опишите схему генетического поиска для решения задач трассировки.

12. Какие методы кодирования хромосом применяются при решении задач САПР?
13. Приведите генетические алгоритмы построения кратчайших связывающих деревьев Прима и Штейнера.
14. Опишите методику планирования кристалла СБИС на основе методов генетического поиска.
15. Постройте схему генетического поиска для решения задач определения изоморфизма графов.
16. В чем особенности и преимущества применения генетических алгоритмов для решения задач САПР?

ЗАКЛЮЧЕНИЕ

Генетические алгоритмы это новая фундаментальная область исследований, основанная Холландом, впервые примененная для решения задач оптимизации и распознавания образов. ГА моделируют в определенной степени механизмы развития органического мира на Земле. В настоящее время ГА это мощный адаптивный поисковый метод, основанный на эволюционных факторах конструирования популяции, естественного отбора, селекции, мутации, эволюционной теории Ч. Дарвина.

В технике наибольшее применение находят простые и объектно-ориентированные ГА, выполняющие 4 основных функции: селекция, кроссинговер, мутация и рекомбинация. Основной проблемой применения таких ГА при решении задач оптимизации является предварительная сходимость, то есть попадание в локальные оптимумы, выход из которых затруднен. В монографии представлены модифицированные схемы генетического поиска, позволяющие в большинстве случаев решать проблемы предварительной сходимости алгоритмов.

Предложенные схемы эффективно используются для решения комбинаторно-логических задач на графах. ГА позволяют одновременно анализировать некоторое подмножество решений, формируя квазиоптимальные решения. Временная сложность алгоритмов зависит от параметров генетического поиска и числа генераций.

Основные проблемы при реализации ГА заключаются в следующем. Важно найти оптимальный размер популяции, достаточное

число генераций. Наибольшие затруднения вызывают вопросы управления процессом генетического поиска. При построении некоторого подмножества популяций появляется возможность эффективного использования параллельных вычислений с использованием транспьютеров, гиперкубовых ЭВМ, многопроцессорных вычислительных систем, нейронных компьютеров и др.

Отметим, что ГА это стратегия выхода из локальных оптимумов. Она заключается в параллельной обработке множества альтернативных решений с концентрацией поиска на наиболее перспективных из них. Причем периодически в каждой итерации можно проводить стохастические изменения в менее перспективных решениях.

В последнее время предпринимаются попытки использовать философские учения древности и их толкование эволюционных законов развития для разработки эффективных эвристических алгоритмов решения проблем выбора и других задач искусственного интеллекта. Так высказывается гипотеза о том, что эволюционные принципы поддерживаются некоторым гомеостатическим механизмом. Исследуются вопросы иерархии в живой и неживой природе на основе фрактальных моделей, т.е. самоподобных иерархических организованных структур. Реализация таких моделей может помочь в решении проблем управления процессом генетического поиска.

Решение рассмотренных в монографии прикладных задач с использованием разработанных генетических алгоритмов показало актуальность использования методов моделирования эволюции и подтвердило важность расширения исследований в этой области.

Составлять много книг –
конца не будет ...

Экклезиаст

ЛИТЕРАТУРА

1. Айала Ф. Введение в популярную и эволюционную генетику. М:Мир, 1984.
2. Алиханян А. И. и др. Общая генетика. М., 1985.
3. Инге-Вечтомов С.Г. Введение в молекулярную генетику. М., 1982.
4. Петров Д.Ф. Генетика с основами селекции. М.: Высшая школа, 1971.410 с.
5. Клизовский А. Основы миропонимания новой эпохи. Минск: Лотоць, 1997, 816с.
6. Дубинин Н.П. Новое в современной генетике. М.: Наука, 1986, 322с.
7. Приходченко Н.Н., Шкурат Т.П. Основы генетики человека. Ростов-на-Дону: Феникс, 1997, 360 с.
8. Holland J.H. Adaptation in Natural and Artificial Systems. An Introductory Analysis with Application to Biology, Control, and Artificial Intelligence. University of Michigan , 1975, 210p.
9. Goldberg David E. Genetic Algorithms in Search, Optimization and Machine Learning. Addison-Wesley Publishing Company, Inc. 1989, 412p.
10. Handbook of Genetic Algorithms, Edited by Lawrence Davis. Van Nostrand Reinhold, New York, 1991, 412p.

11. Ackley D.H., A Connection Machine for Genetic Hillclimbing. Kluwer Academic Publishers, Boston, USA, 1987, 240 p.
12. Foundation of Genetic Algorithms. Rawens Gregory (Editor). Morgan Kaufman Publishers, San Mateo, California, USA, 1991, 473p.
13. Guckles B.P., Party F.E. Genetic Algorithms. IEEE Computer Society Press, Los Alamos, LA, USA, 1992, 89 p.
14. Курейчик В. М. Методы генетического поиска: Учебное пособие. Часть 1. Таганрог. Издательство ТРТУ, 1998, 118с.
15. Курейчик В.М., Лебедев Б.К., Лях А.В., Проблемы эволюционной адаптации в САПР. Новинтех, № 3, 1991.
16. Курейчик В.В. Генетические алгоритмы в проектировании СБИС: Учебное пособие. Таганрог, Издательство ТРТУ, 1997, 106 с.
17. Букатова И.Л. Эволюционное моделирование и его приложения. М: Наука, ., 1979.
18. Букатова И.Л. и др. Эвоинформатика. Теория и практика эволюционного моделирования. М: Наука, 1991.
19. Батищев Д.И. Генетические алгоритмы решения экстремальных задач: Учебное пособие, Воронеж, ВГТУ, 1995, 65 с.
20. Растригин Л.А. Статистические методы поиска. М: Наука, 1968.
21. Mitchell M. An Introduction to Genetic Algorithms. MIT Press, Cambrige, Mass, 1996.
22. Mattfeld D.C. Evolutionary Search and Job Shop: Investigations of Genetic Algorithms for Production Sheduling. Springer-Verlag, Berlin-Heidelberg, 1996, 152 p.
23. Michalewicz Z. Genetic Algorithms + Data Structures = Evolution Programs. Springer-Verlay, New York, 1992.

24. Koza J.R., Genetic Programming: On the Programming of Computers by Means of Natural Selection. MIT Press, Cambrige, Mass, 1992.

25. Курейчик В.М., Зинченко Л.А. Эволюционное моделирование на основе символьных ИТ. Интеллектуальное управление: Новые ИТ в задачах управления. М.: Наука, Физматлит, 1999, с.65-68.

26. Genetic Algorithms and Simulated Annealing. Editor L.Davis, Pitman, London, 1987, 216 p.

27. Back T., Fogel D.B., and Michalewuz Z. Handbook of Evolutionary Computation. Oxford University Press, New York, and Institute of Public Publishing, Bristol, 1997.

28. Potts C.I., Giddens T.D., Yadav S.B. The Development and Evaluation of an Improved Genetic Algorithm Based on Migration and Artificial selection. IEEE Trans. On Systems, Man and Gybernetics, vol. 24, № 1, 1994, p. 73-86.

29. Shahookar R. Mazumder P. Genetic Approoch to Standart Cell Placement Using Meta-Genetic Parameter Optimization. IEEE Trans. On CAD, vol 9, 1990, p.500-511.

30. Saab D. G., Saab Y. G., and Abraham J. A. Automatic Test Vector Cultivation for Seguential VLSI Trans. On CAD of IC&S, №10, 1996, p. 1278-1285.

31. Grefenstette J, Gopal G, Rosmaita B., D. van Gucht. Genetic algorithms for the traveling salesman problem. Proc. Intern. Conf. of Genetic Algorithms and their applications. John Grefenstette, ed. New Yersey, 1987, p. 160-165.

32. Эволюционные вычисления и генетические алгоритмы. Составители Гудман Э.Д., Коваленко А.П. Обозрение

прикладной и промышленной математики, Том 3, Вып. 5, Москва: ТВП, 1966, 760с.

33. Back T. Evolutionary Algorithmus in Theîry and Practice. Oxford University Press, New York, 1996.

34. Grefensette J.J. (ed) Genetic Algorithms for Machine Learning. Kluwer Academic Press, USA, 1994.

35. Herrera F., and Verdegay J.L (eds) Genetic Algorithms and Soft Computing. Physica-Verlag, 1996.

36. Oliver I., Smith D., and Holland J.R.. A study of permutation crossover operators on the traveling salesman problem. Proc. of the Second International Conf. on Genetic Algorithms, p: 224-230.

37. Shervani N. Algoritms for VLSI Physical Design Automation. Kluwer Academic Publ., 1995, 536p.

38. Falkenauer E. Setting New Limits in Bin Packing with a Grouping GA Using Reduction. Tecnical report R0108 (Belgium), March 1994.

39. Kao, Cheng-Yan, "A Stochastic Approach for The One-Dimensional Bin-Packing Problems", Proc., Int. Conf. on Systems, Man, and Cybernetics, Chicago, 1992, p. 1545-1551.

40. Искусственный интеллект: В 3 кн. 1 Системы общения и экспертные системы. Справочник под ред. Попова Э. В. М., Радио и связь, 1990, 464с.

41. Leipins G., Hilliard M., Palmer M., Morrow M. Greedy Genetics. Proc. of the Second International Conf. on Genetic Algorithms. New Jersey, 1987., p. 90-99.

42. Chandraserharam R., Subhramanian S., Chaundhury S. Genetic algorithm for node partitioning problem and application in VLSI design. IEE Proceedings-E, Vol. 140, №.5, 1993, p. 255-260.

43. Курейчик В. М. Математическое обеспечение КТП с применением САПР. М: Радио и связь., 1990, 351 с.
44. Автоматизация проектирования. М., РАН, 1-6, 1977, 1978
45. Курейчик В. М. Генетические алгоритмы. Обзор и состояние. Новости искусственного интеллекта, №3, М: РАИИ, 1998, с.14-63.
46. Fogel L.J., Owens A.J., Walsh M.J., Artificial Intelligence through Simulated Evolution, J.Wiley & Sons, 1966.
47. Nissen V. Solving the Quadratic Assignment Problem with Clues from Nature. IEEE Transactions on Neural Networks, vol. 5, No. 1, 1994.
48. Dao J., Matsumoto N., Hamai T., Ogawa E., Mori S. A Compaction method for full chip VLSI layouts. Proc. 30th Design Automation Couf., ACM/IEEE, 1993, p. 407-412.
49. Cohoon J.P. and Paris W.D. Genetic placement. IEEE Trans. Comput.-Aided Des. Integrated Circuits & Syst., vol.6, no.6, 1987, p. 956-964.
50. Liu X., Sakamoto A., Shimamoto T. Restructive Channel Routing with Evolution Programs. Trans. IEICE, vol.E76-A, no.10, p.1738-1745, 1993.
51. Yoshimura T. and Kuh E.S. Efficient algorithms for channel routing. IEEE Trans. Comput.-Aided Des. Integrated Circuits & Syst., vol.1, no.1, 1982, p. 25-35.
52. Burstein M. Channel routing. Layout Design and Verification. Elsevier Science, 1986, p. 133-167.
53. Попов Э. В. И др. Статические и динамические экспертные системы М:, Финансы и статистика, 1996. 320с.

54. Genetic Algorithms and Evolution Strategy in Engineering and Computer Science. Editid by D. Quagliarella and et all. John Wiley & Song, NY, USA, 1998, 391p.
55. Kureichik V., Goodman E., Tetelbaum A., Genetic Algorithm Approach to Compaction, Bin Packing, and Nesting Problems, Technical Report ¹ 940702, Michigan State University, 1994, 68 c.
56. Корячко В. П. Курейчик В. М. Норенков И. П. Теоретические основы САПР. М., Энергоатомиздат, 1987, 400с.
57. Kureichik V., Miagkikh V., Topchy A. Combined Genetic and Local Search Algorithms for Quadratic Assigment Problem. Proc. First International conference on Evolutionary computation and its Applications, Moscow, Russia. 1996, p.335-341.
58. Kureichik V., Miagkikh V., et al. Some new features in genetic solution of the TSP (traveling salesman problem). Proc. Of the second Internatianal conference adaptive computing in engineering design and control-96. UK, Plymouth, March, 1996.
59. Kureichik V., Davidenko V., Miagkikh V., Genetic algorithms for restrictive channel routing. Proc. of the 7th International conference on Genetic Algorithms, USA, MSU, East Lansing, 1997, p. 636-642.
60. Вороновский Г. К., Махотило К. В., Петрашев С. Н., Серегеев С. Л. Генетические алгоритмы, искусственные нейронные сети и проблемы виртуальной реальности. Харьков: Основа, 1997, 112с.
61. Курейчик В. М. Генетические алгоритмы в технике. Методы кибернетики и информационные технологии, РАЕН, Саратов, СГУ: 1997, С. 45-54.

62. Курейчик В.В. Эволюционные, синергетические и гомеостатические методы принятия решений. Монография. Таганрог: изд-во ТРТУ, 2001, 221 с.

63. Курейчик В.М. Генетические алгоритмы. Монография. Таганрог: изд-во ТРТУ, 1998, 242 с.

64. Курейчик В.В., Курейчик В.М. Об управлении на основе генетического поиска. Автоматика и телемеханика, №10, 2001, с.174-187.

65. Practical Handbook of GENETIC ALGORITHMS: applications, volume 1. Edited by Lance Chambers. CRC Press, 1995, 555 p.

66. Practical Handbook of GENETIC ALGORITHMS: new frontiers, volume 2. Edited by Lance Chambers. CRC Press, 1995, 435 p.

67. Practical Handbook of GENETIC ALGORITHMS: complex coding systems, volume 3. Edited by Lance Chambers. CRC Press, 1999, 572 p.

68. Курейчик В.М. Генетические алгоритмы. Состояние. Проблемы. Перспективы. Известия АН. Теория и системы управления, №1, 1999, с.144-160.

69. Генетический алгоритм разбиения графа. Известия АН. Теория и системы управления, №4, 1999, с.79-87.

70. Курейчик В.В., Курейчик В.М. Генетический алгоритм размещения графа. Известия АН. Теория и системы управления, №5, 2000, с.67-75.

71. Muhlenbein H., Mahnig T., Kureichik V.M., Zinchenko L. Evolutionary modeling with hierarchy and dynamic coding in computer-aided design. Proceedings EUROGEN 2001, Athens, Greece, 2001, p.36-37.

72. Muhlenbein H., Mahnig T., Kureichik V.M., Zinchenko L Adaptive algorithms of evolutionary modeling with knowledge for multi-agent CAG systems. Proceedings 2001 International Symposium on Nonlinear Theory and its Applications, Japan, 2001, p.299-306.

73. Курейчик В.М., Курейчик В.В. Методы управления в ИС принятия решений на основе эволюционного моделирования. М.: Физматлит. Труды международного конгресса «Искусственный интеллект в XXI веке», с.488-499.

СОДЕРЖАНИЕ

Предисловие 3
Введение 6
1. История развития эволюции и генетики 10
1.1. Генетика и основы хромосомной теории эволюции организмов 10
1.1.1. Краткие исторические сведения 10
1.1.2. Носители наследственной информации 16
1.1.3. Рекомбинация 20
1.1.4. Мутация 23
1.1.5. Селекция 29
1.1.6. Генная инженерия 31
1.2. Особенности механизма эволюционной адаптации 33
1.2.1. Роль популяции в эволюции 33
1.2.2. Биосинтез наследственной информации 34
1.2.3. Механизм передачи наследственной информации 36
1.3. Направленная молекулярная эволюция 41
1.4. Выводы 43
1.5. Контрольные вопросы 44
2. Основные понятия генетических алгоритмов 46
2.1. Определение ГА 46
2.2. Простой ГА 52
2.3. Фундаментальная теорема ГА 65
2.4. Выводы 75
2.5. Контрольные вопросы 75
3. Вычислительное представление генетических алгоритмов 77
3.1. Структура данных 77
3.2. Реализация генетических операторов на Си 80
3.3. Выводы 95
3.4. Контрольные вопросы 96
4. Инструментальные средства генетических алгоритмов 97
4.1. Технология генетического поиска 97
4.2. Модифицированные генетические операторы 108
4.3. Методы, основанные на знаниях 112
4.4. ГА и параллельные процессоры 116
4.5. Выводы 125
4.6. Контрольные вопросы 126

5. Использование ГА при решении оптимизационных инженерных задач САПР и искусственного интеллекта 128
5.1. Архитектура интеллектуальных САПР 128
5.2. Разработка генетических алгоритмов компоновки . 149
5.3. Алгоритмы размещения элементов с использованием методов генетического поиска 175
5.4. Разработка алгоритма планирования кристалла на основе генетического поиска 190
5.5. Генетические алгоритмы для трассировки, сжатия и верификации элементов СБИС 194
5.6. Контрольные вопросы 228
Заключение 230
Литература 232

АББРЕВИАТУРЫ

БД - база данных;
БЗ - база знаний;
ГА - генетический алгоритм;
ГК - гиперкуб;
ДНК - дезоксирибонуклеиновая кислота;
ИИ - искусственный интеллект;
ИСАПР - интеллектуальная САПР;
КОГА - кластерно-ориентированный ГА;
МОК - модифицированный ОК;
МОС - модифицированный ОС;
НИ - наследственная информация;
НМЭ - направленная молекулярная эволюция ;
ОК - оператор кроссинговера;
ОКР - ОК разбиения;
ОИ - оператор инверсии;
ОИС - оператор искусственной селекции;
ОМГ - оператор миграции;
ОМ - оператор мутации;
ОР - оператор репродукции;
ОС - оператор сегрегации;
ОТ - оператор транслокации;
ПОГА - проблемно-ориентированный ГА;
ПГА - простой ГА;
РНК - рибонуклеиновая кислота;
САПР - система автоматизированного проектирования;
СП - случайный поиск;
ФП - функция полезности;
ЦФ - целевая функция;
ЭМ - эволюционное моделирование;
ЭП - эволюционное программирование;
ЭПГ - эвристический генератор планов;
ЭС - эволюционные стратегии.

Научное издание

Виктор Михайлович Курейчик

Генетические алгоритмы и их применение

Редактор Белова Л.Ф.
Корректор Пономарева Н.В.
Верстка Барсукова О.В.

ЛР 020565 от 25.06.1997г. Подписано в печать 25.12.2001

Формат 60 × 84 $^{1}/_{16}$ Бумага офсетная

Офсетная печать Усл. п. л. – 15.1 Уч.- изд. л. – 14.8

Заказ № 49 Тираж 250 экз.

"С"

Издательство Таганрогского государственного радиотехнического университета

ГСП 17 А, Таганрог, 28, Некрасовский, 44

Типография Таганрогского государственного радиотехнического университета

ГСП 17 А, Таганрог, 28, Энгельса, 1

Курейчик Виктор Михайлович

Проректор по информатике Таганрогского государственного радиотехнического университета, д.т.н., профессор, академик международной академии информатизации, Российской академии естественных наук, заслуженный деятель науки РФ.

Крупный ученый в области разработки и создания математических моделей проектирования больших и сверхбольших интегральных схем, программно-аппаратных средств интеллектуальных систем автоматизированного проектирования, основатель и руководитель признанной в России и за рубежом научной школы по теории графов, эволюционному моделированию и генетическим алгоритмам, разработке на их основе интеллектуальных САПР.

В 1971 г. защитил кандидатскую диссертацию, а в 1978 году - докторскую. С 1982 года - профессор. Автор 400 печатных работ, в т.ч. 11 монографий и учебников, 30 изобретений. Им подготовлено 40 кандидатов, 6 докторов наук.

В.М. Курейчик награжден серебряной и бронзовой медалями ВДНХ. За заслуги в области высшего образования СССР награжден нагрудным знаком "За отличные успехи в работе".

Избирался академиком Нью-Йоркской академии наук, член американского математического общества, старший член корпорации инженеров по электротехнике и электронике (США) Трехкратный Соросовский профессор по математике.

Награжден серебряной медалью Российской академии естественных наук за открытие в области генетических алгоритмов.