1. Введение в генетические алгоритмы

1.1 Понятие оптимизации

Пусть у нас есть некоторый определенный тип или класс объектов (т.е. множество объектов, удовлетворяющих некоторому набору условий). И пусть нам необходимо найти в этом классе некоторый объект, удовлетворяющий другому некоторому условию. Такой процесс можно обобщенно назвать поиском или решением. Класс, среди объектов которого производится поиск, назовем областью поиска или пространством поиска. Искомый объект можно назвать целью или целевым объектом поиска, а условие, которому он должен удовлетворять – целевым условием. Для определения условия обычно задается некоторая функция на пространстве поиска. Достижение функцией определенного значения и является целевым условием. Такая функция называется целевой функцией. Таким образом, поиск заключается в просмотре по определенным правилам пространства поиска всех объектов, пока не будет обнаружен целевой. Функции выбора нового кандидата для проверки называются операторами поиска. Оператор поиска определяет своего рода прыжок или шаг по пространству поиска.

Примером задачи оптимизации может служить поиск минимума функции $y = x^2$. Областью поиска является все пространство вещественных чисел, целевым условием – минимум функции.

1.2 Естественная эволюция

Как известно, у Природы есть свой метод создания лучших организмов. Дарвин назвал его Эволюцией вследствие Естественного отбора. Эволюция подразумевает под собой последовательное развитие организмов – непрерывную последовательность родителей и их детей, когда дети многое наследуют от своих родителей, но кое в чем от них отличаются. Естественный отбор – это непрерывное сражение за жизнь между всеми. "Выживает сильнейший" – вот жизненное кредо Природы, если награждать титулом "сильный" самого подходящего, самого приспособленного для жизни.

Если подходить к описанию эволюции более формально, то вначале необходимо отметить, что объектом развития (т.е. эволюции) являются не сами организмы, а виды в целом. Вид – это совокупность организмов, сходных по строению и другим признакам. Пользуясь терминологией объектно-ориентированного программирования, вид – это класс, а принадлежащие виду индивиды – объекты этого класса. Совокупность индивидов одного вида назовем популяцией. Чтобы эволюция вообще была возможна, организмы должны отвечать 4 важнейшим свойствам:

Каждый индивид в популяции способен к размножению.

Отличия индивидов друг от друга влияют на вероятность их выживания.

Каждый потомок наследует черты своего родителя (подобное происходит от подобного).

Ресурсы для поддержания жизнедеятельности и размножения ограничены, что порождает конкуренцию и борьбу за них.

Все процессы в живых организмах работают за счет сложных молекул – белков. Каждый белок представляет собой маленький биологический автомат. Молекула белка состоит из последовательности аминокислот. Совокупность информации и строение всех белков в организме определяет его изначальную структуру

(развитие организма происходит также и под действием внешней среды). Вся эта информация называется генетической информацией, или генотипом. Процесс построения, развития организма по информации из генотипа называется онтогенезом. А строение, качества и свойства организма – фенотип. Т.к. внешняя среда воздействует на организм в целом, то можно сказать, что вероятность выживания организма определяется фенотипом.

Генетическая информация в клетке хранится в специальных молекулах – нуклеиновых кислотах. Нуклеиновая кислота представляет собой полимер, т.е. молекулу, представляющую собой последовательность из соединенных между собой небольших молекул – мономеров. Мономерами нуклеиновых кислот являются нуклеотиды. Для кодирования информации используется 4 вида нуклеотидов, обозначаемых по названиям входящих в них азотистых оснований – A,T,G,C. Таким образом, алфавит кодировки состоит из 4 букв.

При сексуальном размножении потомку передается информация о строении родителей путем передачи ДНК. При этом, для построения ДНК потомка, родительские ДНК меняются своими участками. Это процесс называется скрещиванием (кросовер – crossover). При этом новый ген представляет собой комбинацию информации из родительских ДНК (рекомбинация наследственной информации). При размножении может произойти мутация ДНК, т.е. случайное изменение небольшой ее части.

Из этого небольшого обзора хорошо видно, что естественный процесс оптимизации является некоторым компромиссом между вариацией потомства (получением новых индивидов), обеспечением достаточного процента жизнеспособности и стремлением получить хорошее

потомство, т.е. не хуже, или в большинстве случаев лишь чуть хуже предков.

1.3 Генетические алгоритмы

Для компьютерных алгоритмов решения (поиска), использующих вычислительные модели механизмов естественной эволюции в качестве ключевых структурных элементов, используется обобщенное название – эволюционные алгоритмы. Существует множество разновидностей подобного рода алгоритмов, отличающихся использованием или неиспользованием конкретных механизмов, а также различиями трактовки этих механизмов и представлением индивидов.

В генетическом алгоритме (ГА) каждый индивид кодируется сходным с ДНК методом – в виде строки из символов одного типа. Длина строки (ДНК) постоянна. Популяция из индивидов подвергается процессу эволюции с интенсивным использованием скрещивания и мутаций.

Назовем представление каждого индивида геномом. Для каждого вида и каждого представления для данного целевого условия задается целевая функция. Значение целевой функции назовем целевым значением. Вектор, состоящий из целевых значений всех индивидов в популяции, назовем вектором целевых значений. Тогда если вычислен вектор целевых значений, то можно определить приспособленность (fitness) индивида в популяции, для чего задается специальная функция приспособленности от данного целевого значения и от вектора целевых значений. Аналогично вектору целевых значений введем вектор при-

способленности. Мы отделяем приспособленность от целевого значения специально, т.к. приспособленность индивида зависит и от остальных индивидов, и важна для выживаемости индивида, а целевое значение важно в первую очередь для нас. Часто целевое значение называют приспособленностью, а значение приспособленности, в смысле вероятность участия в размножении, неявно вычисляется во время отбора. Процесс эволюции останавливается, когда популяция отвечает определенному критерию – критерию завершения.

Принципиальная схема работы ГА состоит из следующих основных фаз:

Создание начальной популяции. Задание генома каждому из индивидов. Расчет вектора целевых значений.

Шаг эволюции – построение нового поколения.

Проверка критерия завершения, если не выполнено – переход на 2.

Шаг эволюции можно разделить на следующие этапы:

- Вычисление вектора приспособленности.
- Отбор кандидатов на скрещивание (Отбор – Selection) .
- Скрещивание, т.е. порождение каждой парой отобранных кандидатов новых индивидов, путем геномов.
- Мутация геномов.
- Вычисление вектора целевых значений и построение новой популяции (нового поколения).

1.4 Целевая функция и кодирование

Среди компонентов, образующих генетический алгоритм, в большинстве случаев только два непосредственно определяются конкретной задачей – это кодировка задачи (отображение пространства поиска на пространство битовых строк) и целевая функция. Рассмотрим задачу параметрической оптимизации, заключающуюся в определении набора переменных, миниимизирующих некоторую величину (цель). В

более традиционных терминах, задача состоит в поиске минимума некоторой функции $F(X_1, X_2, \ldots, X_M)$.

На первом этапе обычно делается предположение, что переменные, представляющие параметры, могут быть представлены битовыми строками. Это означает, что переменные предварительно дискретизируются некоторым образом и что область дискретных значений соответствует некоторой степени 2. Например, с 10 битами на параметр мы получаем область из $2^{10} = 1024$ дискретных значений. Если параметры непрерывны, то проблема дискретизации не заслуживает особого внимания. Разумеется, предполагается, что дискретизация обеспечивает достаточное разрешение, чтобы сделать возможным регулирование получения результата с желаемым уровнем точности. Предполагается также, что дискретизация в некотором смысле представляет основную функцию.

Битовую строку длины N можно рассматривать как целое двоичное число I, которому соответствует некоторое вещественное значение r из заданного диапазона $[Min, Max]$. Это соответствие устанавливается формулой

$$r = Min + (Max - Min)\frac{I}{2^N - 1}.$$

За исключением проблемы кодирования, целевая функция обычно дается как часть постановки задачи. С другой стороны, разработка целевой функции иногда может непосредственно входить в разработку алгоритма оптимизации или поиска решения. В других случаях значение целевой функции может зависеть от реализации и давать только приближенный или частный результат.

1.5 1.5 Общая структура генетического алгоритма

В природе особи в популяции конкурируют друг с другом за различные ресурсы, такие, например, как пища или вода. Кроме того, члены популяции одного вида часто конкурируют за привлечение брачного партнера. Те особи, которые наиболее приспособлены к окружающим условиям, будут иметь относительно больше шансов на воспроизводство потомков. Слабо приспособленные особи либо совсем не произведут потомства, либо их потомство будет очень немногочисленным. Это означает, что гены от высоко адаптированных или приспособленных особей будут распространяться в увеличивающемся количестве потомков на каждом последующем поколении. Таким образом, вид развивается,

все лучше приспосабливаясь к среде обитания.

Генетические алгоритмы используют прямую аналогию с таким механизмом. Они работают с совокупностью "особей" – популяцией, каждая из которых представляет возможное решение данной проблемы. Каждая особь оценивается мерой ее "приспособленности" согласно тому, насколько "хорошо" соответствующее ей решение задачи. Например, мерой приспособленности могло бы быть отношение силы/веса для данного проекта моста. (В природе это эквивалентно оценке того, насколько эффективен организм при конкуренции за ресурсы.) Наиболее приспособленные особи получают возможность "воспроизводить" потомство с помощью "перекрестного скрещивания" с другими особями популяции. Это приводит к появлению новых особей, которые сочетают в себе некоторые характеристики, наследуемые ими от родителей. Наименее приспособленные особи с меньшей вероятностью смогут воспроизвести потомков, так что те свойства, которыми они обладали, будут постепенно исчезать из популяции в процессе эволюции.

Таким образом, воспроизводится вся новая популяция допустимых решений, выбирая лучших представителей предыдущего поколения, скрещивая их и получая множество новых особей. Это новое поколение содержит более высокое соотношение характеристик, которыми обладают хорошие члены предыдущего поколения. Таким образом, из поколения в поколение хорошие характеристики распространяются по всей популяции. Скрещивание наиболее приспособленных особей приводит к тому, что исследуются наиболее перспективные участки пространства поиска. В конечном итоге популяция будет сходиться к оптимальному решению задачи.

Имеется много способов реализации идеи биологической эволюции в рамках генетических алгоритмов. Каноническим считается генетический алгоритм, представленный в виде следующего псевдокода.

```
begin {генетический алгоритм}
    t := 0;
    done := false;
    init_population( P (0) );
        {генерация начальной популяции P (0) (случайным образом
        или случайным в комбинации с некоторым набором началь-
        ных решений) и оценка пригодности каждой особи}
```

while not *done* **do begin**

P' := select_parents**(** *P* (*t*));

{выбор родителей для скрещивания при помощи оператора селекции согласно пригодности и помещение их в промежуточную популяцию *P*'}

recombine *P*'(*t*);

{промежуточная популяция попарно подвергается оператору рекомбинации или скрещивания}

mutate *P*' (*t*);

{каждая особь в промежуточной популяции подвергается оператору мутации}

evaluate *P*' (*t*);

{расчет целевой функции для всех новых особей}

P(*t*+1) := *P*'(*t*);

{промежуточная популяция копируется в новое поколение *P*(*t*+1)}

Done := || *P*(*t*+1) - *P*(*t*+1)||<ε;

{проверка выполнения критерия сходимости}

t := *t* + 1;

end {while}

end

Хотя модель эволюционного развития, применяемая в генетических алгоритмах, сильно упрощена по сравнению со своим природным аналогом, тем не менее генетические алгоритмы являются достаточно мощным средством и могут с успехом применяться для широкого класса прикладных задач, включая те, которые трудно, а иногда и вовсе невозможно решить другими методами. Однако, генетические алгоритмы не гарантируют обнаружения глобального решения за конечное время. Генетические алгоритмы не гарантируют и того, что глобальное решение будет найдено (впрочем, для произвольной целевой функции за конечное время этого невозможно сделать ни одним алгоритмом), но они хороши для поиска "достаточно хорошего" решения задачи "достаточно быстро". Главным же преимуществом генетических алгоритмов является то, что они могут применяться даже на сложных задачах, там, где не существует никаких специальных методов. Даже там, где хорошо рабо-

тают существующие методики, можно достигнуть улучшения сочетанием их с генетическими алгоритмами.

2. Описание простого генетического алгоритма

Генетические алгоритмы носят итерационный характер и имеют дело с обработкой популяций индивидуумов $P(t)=\{x_1^t, x_2^t,..., x_n^t\}$ для итерации t (поколение t). Каждый индивидуум представляет собой потенциальное решение задачи (испытание) и представлен в некоторой, возможно достаточно сложной, структуре данных S. В качестве S будем рассматривать бинарные строки

Каждое решение x_i^t оценивается и определяется мера его "пригодности". Затем формируется новая популяция (итерация или поколение t + 1). На первом шаге этого формирования – этапе селекции – происходит отбор индивидуумов, обладающих лучшей пригодностью. На следующем шаге некоторые из отобранных таким образом индивидуумов подвергаются преобразованиям с помощью "генетических операторов": мутации и скрещивания. Оператор мутации создает нового индивидуума путем относительно малого изменения в одном индивидууме, а оператор скрещивания осуществляет более сильные трансформации и создает нового индивидуума путем комбинирования частей из нескольких (двух или больше) индивидуумов. После ряда итерационных шагов алгоритм сходится к лучшему из возможных решений. Остановимся теперь более подробно на трех "генетических операторах" – селекции, скрещивании и мутации.

2.1 Селекция.

Целью селекции является осуществление выборки индивидуумов в текущей популяции (т.е. из некоторого набора) пропорционально их пригодности. Обычно используют четыре различных механизма селекции – “колесо рулетки”, остаточная стохастическая выборка, стохастическая равномерная выборка и турнирная селекция. Первые три алгоритма являются вариантами пропорциональной селекции, а последний – непропорциональной.

Вам предлагается использовать так называемую турнирную селекцию, не требующую предварительного ранжирования функции пригодности. При этом последовательно берутся два соседних элемента текущей популяции (первый и второй, третий и четвертый и т.д.) и лучший из них (т.е. элемент, обладающий меньшим значением целевой функции или функции пригодности) помещается в промежуточную по-

пуляцию P'. После первого прохода (пока сформирована только половина промежуточной популяции) исходная популяция случайным образом перемешивается и описанный процесс повторяется еще один раз. Здесь лучшие или худшие индивидуумы рассматриваются в смысле их упорядочивания согласно соответствующим значениям целевой функции.

2.2 Скрещивание.

Наиболее простым является *одноточечное скрещивание* – каждая выбранная таким образом пара строк скрещивается следующим образом: случайным образом выбирается положение точки сечения (целое число k в промежутке от 1 и l-1, где l – длина строки). Затем, путем обмена всеми элементами между позициями k+1 и l включительно, рождаются две новых строки. Например, пусть первая особь – $A=(x_1,x_2,x_3,x_4,x_5)$ а вторая соответственно $B=(y_1,y_2,y_3,y_4,y_5)$ и пусть случайно выбранная точка сечения будет после третьего гена (бита). Тогда в результате скрещивания получим две особи-потомки – $A'=(x_1,x_2,x_3,y_4,y_5)$ и $B'=(y_1,y_2,y_3,x_4,x_5)$. После этого потомки замещают родительские особи в промежуточной популяции P'. Схематично этот вариант показан на рисунке 1.1.

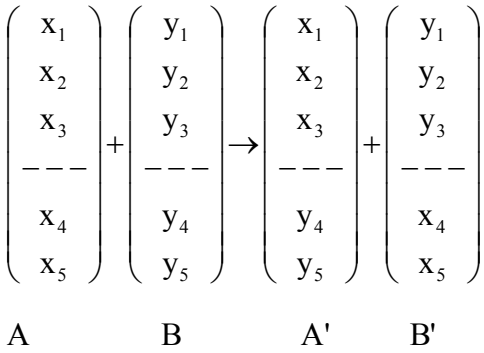

Рисунок – 1.1

Одноточечное скрещивание легко обобщается на ***n-точечное*** с n точками сечения. Предельным случаем является ***равномерное*** скрещивание, при котором каждый ген первого из родителей случайным обра-

зом передается любому из потомков, при этом другой потомок, соответственно, получает ген от другого родителя.

После рекомбинации, применяется также оператор ***мутации***. Каждый бит каждой особи в популяции мутирует (точнее, пытается мутировать) с некоторой низкой вероятностью p_m. Обычно мутация применяется с вероятностью меньше чем 1 %. Обычно мутация интерпретируется как “зеркальное отражение“ бита (инверсия его значения, т.е. изменение его с 1 на 0 или с 0 на 1).

Кроме описанной ***инверсионной*** мутации можно применить оператор ***двухточечной*** мутации. В этом случае если мутация происходит, то случайным образом выбираются два гена, которые обмениваются своими значениями.

После завершения процессов выбора, рекомбинации и мутации, следующая популяция может быть оценена. Процесс оценки, выбора, рекомбинации и мутации формирует одно поколение в выполнении генетического алгоритма.

Полезно рассмотреть выполнение генетического алгоритма как двухстадийный процесс (рис. 1.2). Начинается он с *текущей популяции,* к которой применяется оператор выбора, чтобы создать промежуточную популяцию. При этом, например, особь s_2 копируется в промежуточную популяцию дважды, а s_3 – ни разу. После этого к промежуточной популяции применяются операторы рекомбинации и мутации для того, чтобы создать следующую популяцию. Процесс продвижения от текущей популяции до следующей популяции составляет одно поколение в выполнении генетического алгоритма.

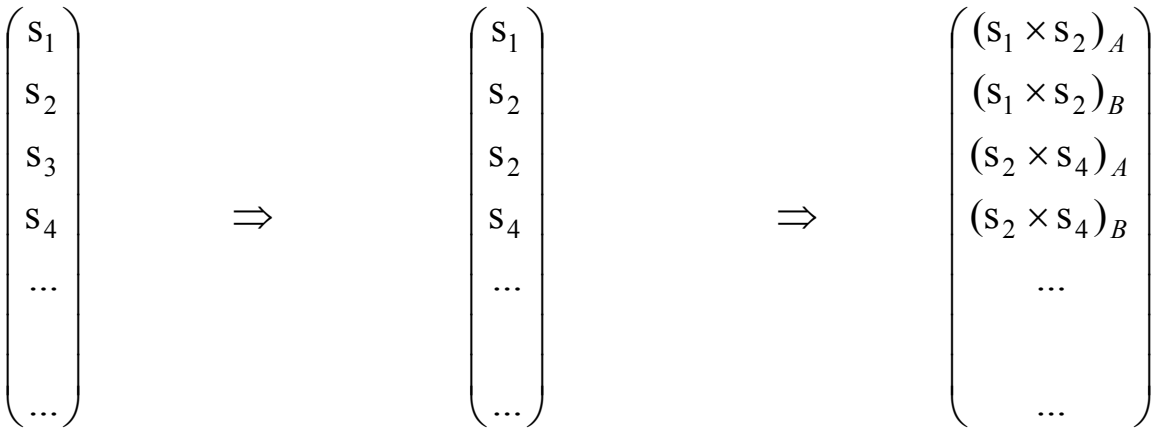

Рисунок – 1.2

В приложении приведен пример программы, реализующей простой генетический алгоритм (турнирная селекция, одноточечное скрещивание, одноточечная мутация) для нахождения минимума функции одного переменного $y = x^2, -5.12 \le x \le 5.12$.

3. Задание.

Целевая функция

$$z(x, y) = \text{int}(x) + \text{int}(y), \quad -5.12 \le x \le 5.12, \quad -5.12 \le y \le 5.12$$

Рассмотреть одноточечное скрещивание и двухточечную мутацию. Каждая переменная кодируется 30 битами. Провести расчеты для 30 и 100 поколений. Сравнить получающиеся решения при размерах популяции 10, 20, 30 особей.

Решение.

3.1 Анализ задания

В соответствии с заданием составим программу на языке Pascal для решения указанной целевой функции. Программа использует генетический алгоритм для поиска решения.

В генетическом алгоритме каждый индивид кодируется сходным с ДНК методом – в виде строки из символов одного типа. Длина строки постоянна. Популяция из индивидов подвергается процессу эволюции с интенсивным использованием скрещивания и мутаций.

Принципиальная схема работы генетического алгоритма состоит из следующих основных фаз:

1. Создание начальной популяции. Задание генома каждому из индивидов. Расчет вектора целевых значений.
2. Шаг эволюции – построение нового поколения.
3. Проверка критерия завершения, если не выполнено – переход на 2.

Шаг эволюции можно разделить на следующие этапы:

1. Вычисление вектора приспособленности.
2. Отбор кандидатов на скрещивание.
3. Скрещивание.
4. Мутация.
5. Вычисление вектора целевых значений и построение новой популяции.

В соответствии с заданием рассмотрим одноточечное скрещивание и двухточечную мутацию.

3.2 Одноточечное скрещивание

Одноточечное скрещивание – каждая выбранная пара строк скрещивается следующим образом: случайно выбирается положение точки сечения (целое число k в промежутке от 1 и l-1, где l – длина строки). Затем, путем обмена всеми элементами между позициями k+1 и l включительно, рождаются две новых строки.

Например, пусть первая особь – A = (x_1, x_2, x_3, x_4, x_5), а вторая соответственно B = (y_1, y_2, y_3, y_4, y_5) и пусть случайно выбранная точка сечения будет после третьего гена (бита). Тогда в результате скрещивания получим две особи-потомки – A' = (x_1, x_2, x_3, y_4, y_5) и B' = (y_1, y_2, y_3, x_4, x_5). После этого потомки замещают родительские особи в промежуточной популяции P'. Схематично этот вариант показан на рисунке 1.

$$\begin{pmatrix} x_1 \\ x_2 \\ x_3 \\ --- \\ x_4 \\ x_5 \end{pmatrix} + \begin{pmatrix} y_1 \\ y_2 \\ y_3 \\ --- \\ y_4 \\ y_5 \end{pmatrix} \rightarrow \begin{pmatrix} x_1 \\ x_2 \\ x_3 \\ --- \\ y_4 \\ y_5 \end{pmatrix} + \begin{pmatrix} y_1 \\ y_2 \\ y_3 \\ --- \\ x_4 \\ x_5 \end{pmatrix}$$

A B A' B'

Рисунок 1

Блок-схема алгоритма одноточечного скрещивания показана на рисунке 2.

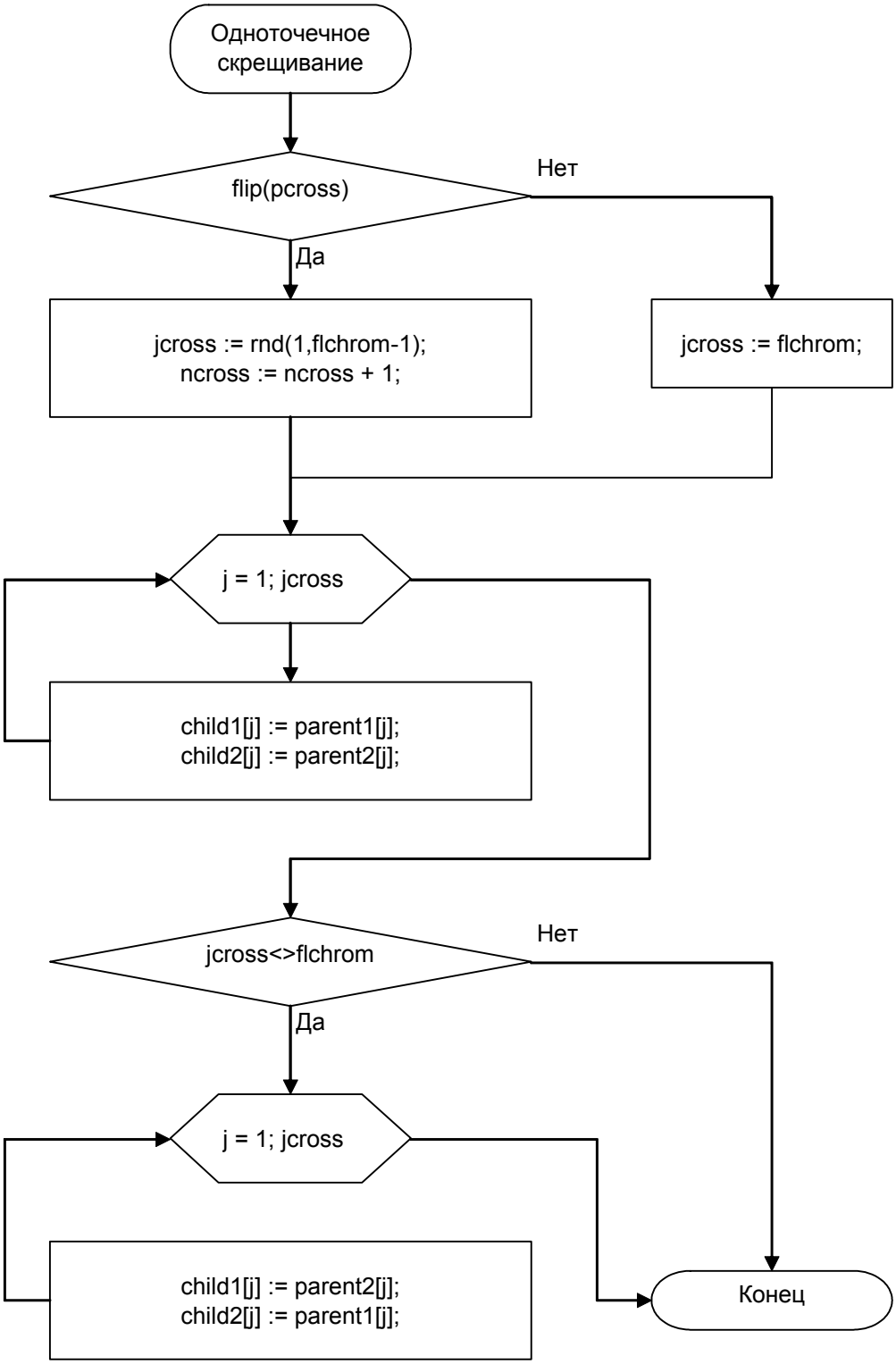
Одноточечное скрещивание
flip(pcross)
Нет
Да
jcross := rnd(1,flchrom-1);
ncross := ncross + 1;
jcross := flchrom;
j = 1; jcross
child1[j] := parent1[j];
child2[j] := parent2[j];
jcross<>flchrom
Нет
Да
j = 1; jcross
child1[j] := parent2[j];
child2[j] := parent1[j];
Конец

Рисунок 2

3.3 Двухточечная мутация

Каждый бит каждой особи в популяции мутирует (точнее, пытается мутировать) с некоторой низкой вероятностью p_m. Обычно мутация применяется с вероятностью меньше чем 1%. Обычно мутация интерпретируется как “зеркальное отражение“ бита (инверсия его значения, т.е. изменение его с 1 на 0 или с 0 на 1).

Блок-схема алгоритма двухточечной мутации показана на рисунке 3.

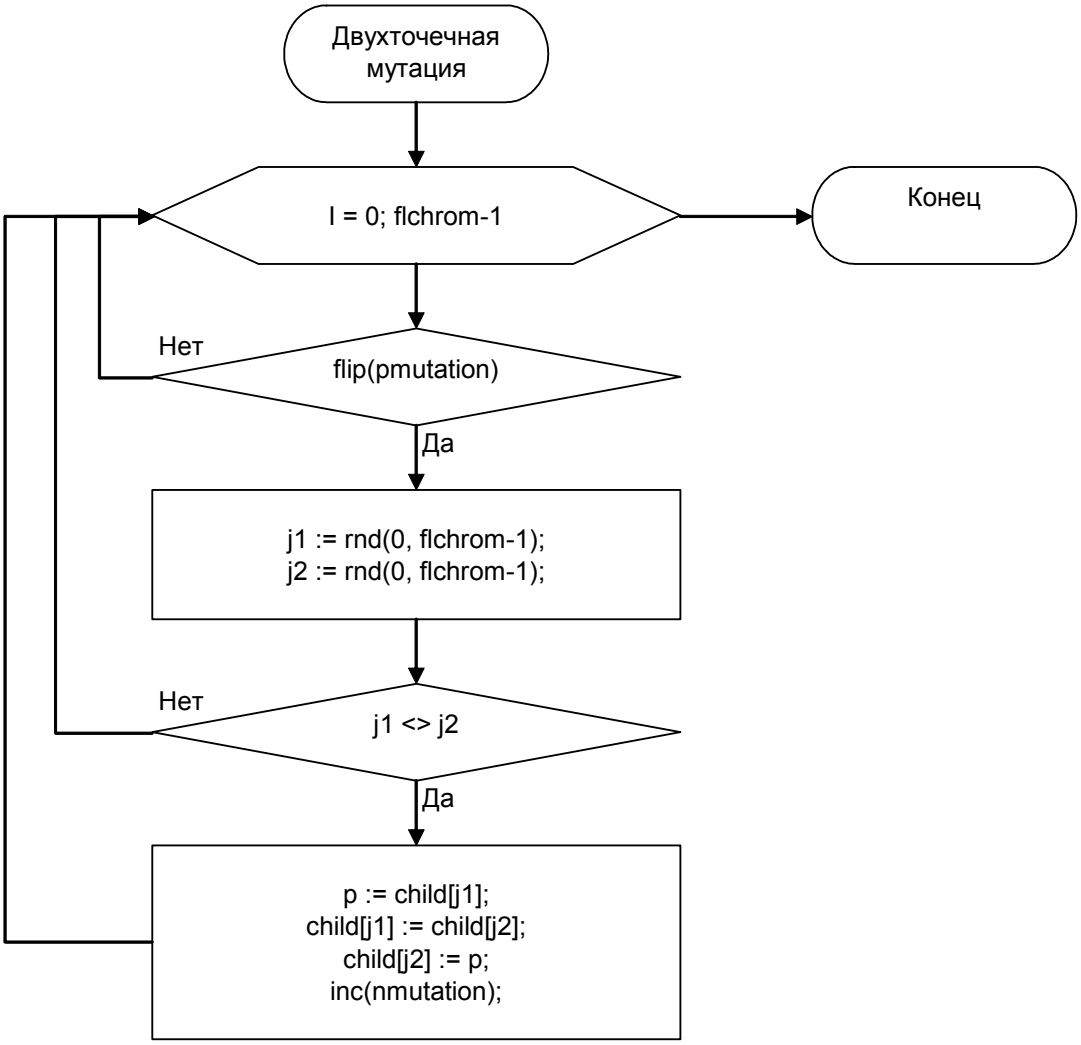

Рисунок 3

3.4 Анализ работы программы

Выполним анализ результатов работы программы. Согласно заданию, необходимо провести расчеты для размера популяции в 10, 20 и 30 особей при числе поколений 30 и 100. Поскольку решение задачи носит вероятностный характер, организуем программно поиск решений 500 раз для каждого набора исходных данных.

Поскольку ответ задачи получается целочисленным, то можно анализировать частоту появления правильного ответа. Сводные таблицы результатов показаны ниже.

Поколений	30
Особей	10

Ответ	Количество
-11	0
-10	187
-9	214
-8	78
-7	15
-6	6
-5	0
-4	0
-3	0
-2	0

Поколений	30
Особей	20

Ответ	Количество
-11	0
-10	384
-9	109
-8	7
-7	0
-6	0
-5	0
-4	0
-3	0
-2	0

Поколений	30
Особей	30

Ответ	Количество
-11	0
-10	447
-9	53
-8	0
-7	0
-6	0
-5	0
-4	0
-3	0
-2	0

Поколений	100
Особей	10

Ответ	Количество
-11	0
-10	441
-9	56
-8	1
-7	0
-6	0
-5	2
-4	0
-3	0
-2	0

Поколений	100
Особей	20

Ответ	Количество
-11	0
-10	498
-9	2
-8	0
-7	0
-6	0
-5	0
-4	0
-3	0
-2	0

Поколений	100
Особей	30

Ответ	Количество
-11	0
-10	500
-9	0
-8	0
-7	0
-6	0
-5	0
-4	0
-3	0
-2	0

Согласно полученным результатам, вероятность правильного ответа возрастает с увеличением числа поколений и размера популяции. Так, для 30 поколений количество правильных ответов уже для 30 особей можно считать приемлемым – 447 против 53 неверных. При меньшем размере популяции результаты получаются неудовлетворительными.

Для 100 поколений хорошие результаты показывает уже популяция размером от 20 особей. Для 30 особей результат получается 100%-ным.