РАЗДЕЛ 3

ФОРМИРОВАНИЕ ЗНАНИЙ ДЛЯ ЭКСПЕРТНОЙ СИСТЕМЫ ОПРЕДЕЛЕНИЯ СТЕПЕНИ РИСКА СВСГР

3.1. Разработка метода классификации на основе корреляционного анализа

На сегодняшний день не все медицинские учреждения оборудованы вычислительной техникой. Поэтому не менее актуальной задачей является разработка метода определения степени риска СВСГР без применения технических ресурсов. С этой целью разработана классификационная таблица определения степени риска СВСГР [12,58,59].

Предобработка входных данных

Входное обучающее множество должно быть представлено в виде числовых переменных. Для этого исходные нечисловые данные были закодированы. При кодировании использовались методы, описанные в разделе 1.3. Учитывая, что метод корреляционного анализа, как и другие статистические методы, не предъявляет высоких требований к данным, другая предобработка не использовалась. Способы предобработки для корреляционного анализа представлены в приложении Ж.

Реализация метода

Согласно с постановкой задачи раздела 2.3, функция классификации описывается формулами (2.2) и (2.3). Из чего следует:

- необходимо задать количество и интерпретации вариантов классификации, $K=\{K_1,K_2,\ldots,K_l\}$ - возможные результаты классификации (степень риска СВСГР), где l - количество вариантов классификации;
- определить вектор весовых коэффициентов $W=\{w_1,w_2,\ldots,w_n\}$ для факторов риска $X=\{x1,x2,\ldots,x_n\}$, где n - количество

факторов риска;

– определить некоторые пороговые значения $N=\{N_1,N_2,\ldots,N_l\}$, где l -количество вариантов классификации;

Классификационная таблица (предложенные пороговые значения и число градаций степени риска, в зависимости от срока беременности) представлена в таблице 3.1. Получение весовых коэффициентов было показано в разделе 2.6, результаты представлены в таблице 3.1 в приложении З. Классификация предложенным методом [12,58,59] сводится к подсчету балов для всех присутствующих факторов риска, формула (2.3) и определению по таблице 3.1 степени риска СВСГР.

Таблица 3.1.

Определение степени риска СВСГР.

Срок беременности	Степень риска			
	Очень низкий риск	Низкий риск	Высокий риск	Очень высокий риск
До 24 недель	До 14	14—27	27—55	Более 55
После 24 недель	До 15	15—31	31—61	Более 61
После родов	До 20	20—40	40—80	Более 80

Если при определении степени риска СВСГР установлено, что риск высокий или очень высокий, осуществляется поиск факторов, которые наиболее повлияли на результат, т.е. присутствующие факторы с максимальным весом. Выбираются те из них, которыми можно управлять, и разрабатываются мероприятия, направленные на снижение степени риска СВСГР. К тому же, если определение степени риска проводилась еще на этапе беременности, рассматриваются факторы, которые не были включены в анализ, но имеют большой вес, и планируются профилактические меры, направленные на предупреждение их влияния. Перечень факторов риска, которыми можно управлять, приведен в приложении Л.

3.2. Формирование знаний на основе нейронных сетей

Как правило, экспертная система предназначена реализовывать логические рассуждения, которые имитируют анализ ситуации человеком, но иногда достаточно, чтобы экспертная система представляла собой просто компьютерную систему (по методу черного ящика). В соответствии с постановкой задачи описанной в разделе 2.3, применительно к нашей проблеме, задачу можно сформулировать так: имеются факторы риска, которые подаются на входы экспертной системы, на выходе получаем прогноз, который может быть следующим: высокая или низкая степень риска СВСГР. Для реализации поставленной задачи в такой форме предложено использовать многослойную нейронную сеть.

Предобработка входных данных

Нейронные сети [24,26,32-35,41,42,56] работают только с числовой информацией. Необходимо представить нечисловую входную информацию в виде чисел, т.е. закодировать. Нейронные сети, являются нелинейными методами обработки данных, поэтому важно сохранить смысл и внутренние взаимосвязи в данных. При кодировании использовались следующие описанные в разделе 1.3 методы: кодирование бинарных признаков; кодирование неупорядоченных качественных признаков - двоичное кодирование типа «n -> n»; кодирование упорядоченных качественных признаков - разными значениями одного входного сигнала. Для числовых данных была выполнена нормировка [11,13,41,43]. Выполнена предварительная обработка данных различным образом, результаты представлены в приложениях В, Г, Д, Е. Наиболее подходящие варианты кодирования представлены в приложениях В и Г. В разделе 2.7 показано, что с использованием варианта кодирования, предложенного в приложении В, результаты обучения НС лучше чем, если использовать вариант кодирования в приложении Г.

Реализация метода

В разделе 2.7 представлены исследования по выбору архитектуры и обучению нейронной сети. Архитектура сети определяется количеством слоев, количеством нейронов на каждом слое и активационной функцией для каждого слоя (рассматривались наиболее распространенные для решения подобных задач варианты: линейная, сигмоидальная, гиперболический тангенс). В разделе 2.7 выбрана архитектура сети, представленная на рисунке 2.10, с учетом дальнейшей работы ГА. Т.к. на каждом шаге выполнения ГА происходит обучение сети, то время сходимости алгоритма обучения НС существенно влияет на общее время работы ГА. С целью минимизировать временные затраты на обучение сети, выбрана нейронная сеть минимальной вычислительной сложности. Понятие вычислительной сложности рассмотрено разделе 2.7. Для задачи прогнозирования скорость обучения не является определяющим параметром по выбору архитектуры сети, в первую очередь будем ориентироваться на точность прогнозирования. Так же не столь принципиален и размер обучающей выборки. Для экспертной системы на основе нейронной сети не обязательно производить предварительную обработку с целью минимизации входного набора данных, т.е. выполнять отбор информативных факторов.

Возможны следующие архитектуры НС рисунки 3.1-3.2. Архитектуры данных сетей рассматривались в разделе 2.7, и приведены в таблице 2.1.

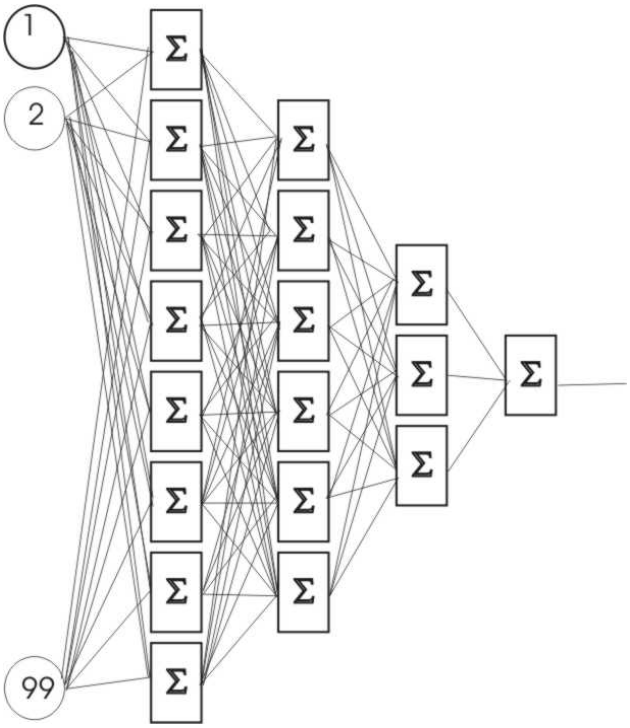

Рисунок 3.1.Архитектура нейронной сети. Выбранные активационные функции – гиперболический тангенс, сигмоид, гиперболический тангенс, линейная.

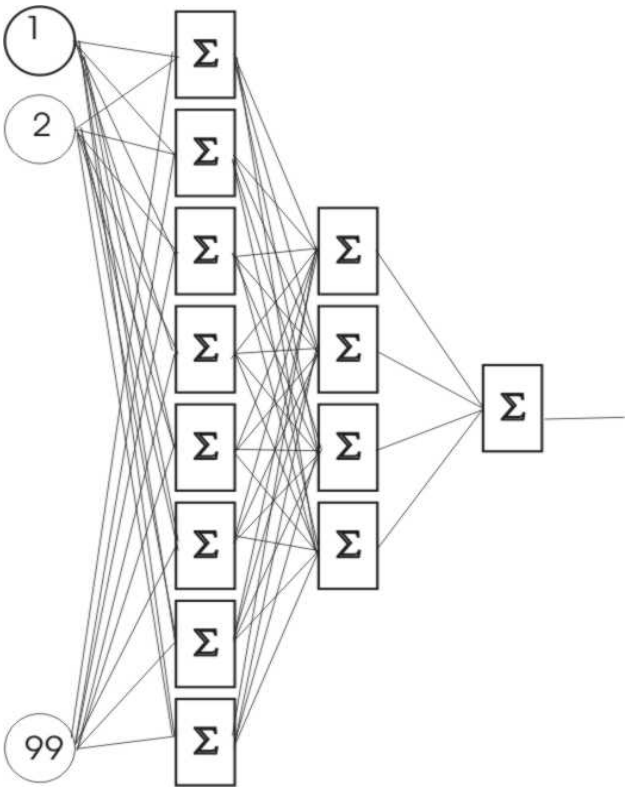

Рисунок 3.2.Архитектура нейронной сети. Выбранные активационные функции – гиперболический тангенс, сигмоид, линейная.

3.3. Формирование знаний на основе генетического программирования

Для формирования знаний можно использовать генетическое программирование (ГП) [66,73,77,81-83].

3.3.1. Общий алгоритм генетического программирования

Для решения задачи с помощью ГП необходимо выполнить следующие предварительные этапы:

1. Определить терминальное множество;
2. Определить функциональное множество;
3. Определить фитнесс-функцию;
4. Определить значения параметров, такие как мощность популяции, максимальный размер особи, вероятности кроссинговера и мутации, способ отбора родителей, критерий окончания эволюции (например, максимальное число поколений) и т.п.

После этого можно разрабатывать непосредственно сам эволюционный алгоритм, реализующий ГП для конкретной задачи. Возможны различные подходы. Например, решение задачи на основе ГП можно представить следующей последовательностью действий.

1. Установка параметров эволюции;
2. Инициализация начальной популяции;
3. T:=0;
4. Оценка особей, входящих в популяцию;
5. T:=T+1;
6. Отбор родителей;
7. Создание потомков выбранных пар родителей – выполнение оператор кроссинговера;
8. Мутация новых особей;
9. Расширение популяции новыми порожденными особями;
10. Сокращение расширенной популяции до исходного размера;

11. Если критерий останова алгоритма выполнен, то выбор лучшей особи в конечной популяции – результат работы алгоритма. Иначе переход на шаг 4.

Приведенный алгоритм относится к классу стационарных синхронных эволюционных алгоритмов, где мощность популяции поддерживается постоянной и переход к следующему поколению «синхронизирован». Здесь обрабатываются все потомки предыдущего поколения, а затем следует переход на следующую итерацию (поколение). Обобщенный алгоритм работы ГП представлен на рисунке 3.3.

3.3.2. Применение генетического программирования при построении дерева для прогнозирования СВСГР.

Аппарат ГП позволяет построить эволюционным алгоритмом на основе обучающей выборки программу, которая выполняет прогнозирование заболевания. Полученную программу можно использовать как для предсказания (прогнозирования СВСГР), так и для понимания. В соответствии с постановкой задачи в разделе 2.3 необходимо получить булеву функцию вида:

$$F(x_1,x_2,\ldots,x_n)=K \qquad (3.1)$$

Значение K=1 на выходе означает высокую степень риска СВСГР, а K=0 соответственно низкую.

Учитывая, что выражение (3.1) представляет собой булеву функцию, переменные x_1-x_n это булевы переменные. Это означает, что входные данные должны быть предварительно обработаны, основное назначение предобработки преобразовать входное обучающее множество в булевы переменные. Далее в соответствии с приведенным выше алгоритмом определяются терминальное и функциональное множества, фитнесс-функция, параметры эволюционного алгоритма и т.п. Решением задачи будет нахождение функции (3.1), которую можно представить деревом.

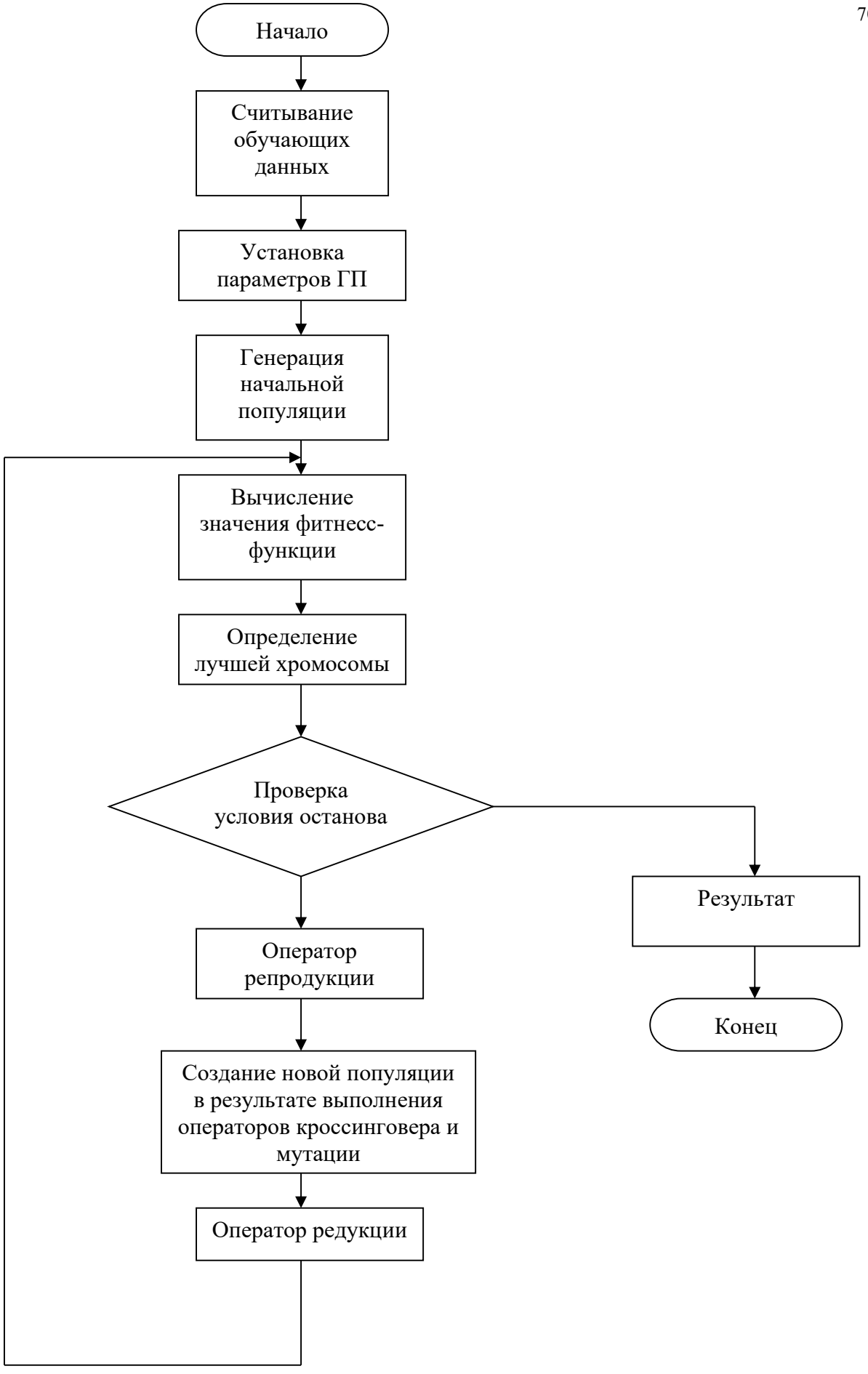

Рисунок 3.3. Обобщенный алгоритм работы ГП.

Предобработка входных данных

Входное обучающее множество должно быть представлено в виде булевых переменных. Для этого исходные данные были преобразованы как показано в приложении Д.

Терминальное множество

Терминальное множество в данном случае составляют выделенные ранее информативные факторы риска, которые после предобработки представляют собой булевы переменные.

Функциональное множество

Функциональное множество состоит из логических операций: AND, OR, NOT. Так как первые две операции могут иметь два и более входов и один выход, а последняя операция всегда имеет один вход и один выход, то для более удобной программной реализации заменим операцию NOT на AND-NOT и OR-NOT. Такая замена выполнена с целью унифицировать количество входов для всех операций (их будет всегда 2). Таким образом, функциональное множество состоит из 4 логических операций AND, OR, AND-NOT и OR-NOT. Так же подобная замена позволяет избавится от так называемых интронов (бесполезных участков кода), например двойное отрицание (NOT(NOT(X))).

Фитнесс-функция

В качестве фитнесс-функции рассматривается доля пациентов с правильно поставленным диагнозом:

$$E = \frac{1}{M} * (M - \sum_{i=1}^{M} (| F_i - Y_i |)) \qquad (3.2)$$

Переменная диагноза Y и выходное значение F принимают булевы значения 0 или 1. Единица соответствует положительному диагнозу (высокой

степени риска СВСГР) и ноль отрицательному (низкой степени риска СВСГР), *M* - количество обучающих примеров.

Реализация метода

В контексте нашей задачи ГП используется для получения дерева, которое позволяет распознавать высокую степень риска СВСГР [45,89]. Примером может, быть дерево как на рисунке 3.4.

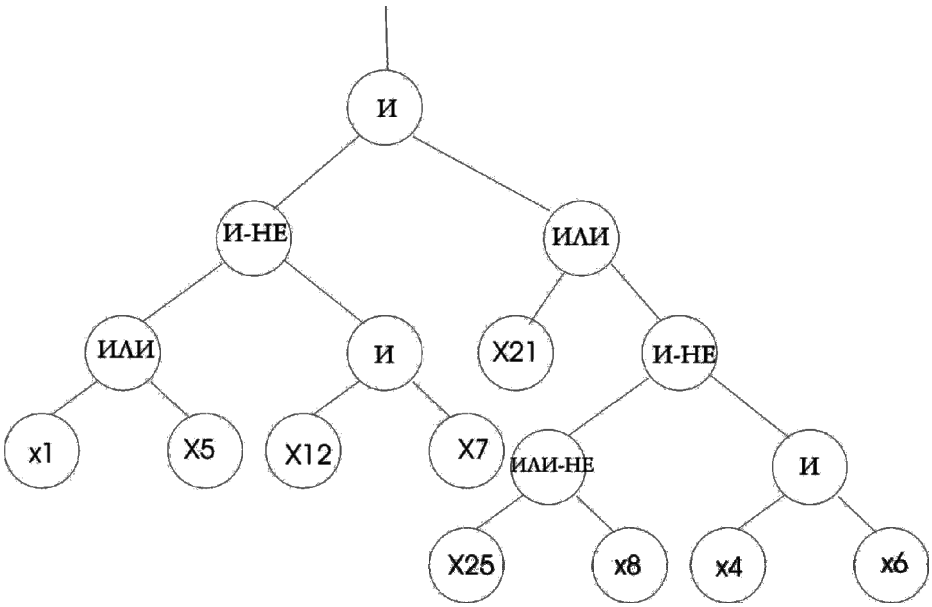

Рисунок 3.4. Пример дерева прогнозирования.

Дереву на рисунке 3.4 соответствует булева функция (3.3).

$$F(x_1, x_2, ..., x_n) = \overline{(x_1 + x_5) \bullet (x_{12} \cdot x_7)} \bullet (x_{21} + \overline{(x_{25} + x_8) \bullet (x_4 \cdot x_6)}) \qquad (3.3)$$

Для получения дерева прогнозирования необходимо выполнить следующие этапы:

1. Установка параметров ГП. Возможные варианты приведены в таблице 3.2.
2. Генерация начальной популяции. Популяция представляет собой набор хромосом. Каждая хромосома соответствует определенному дереву, представляющее собой решение, например как на рисунке 3.4. Дерево (хромосома), на начальном этапе генерируется случайным образом и состоит из функциональных и терминальных

узлов (множество которых описано выше).

3. По значению фитнесс-функции (формула (3.2)) оцениваются особи входящие в популяцию.
4. Применение генетических операций.
5. Проверка критерия останова. При его выполнении переход на шаг 6, иначе шаг 3.
6. Лучшее дерево (максимальное значение фитнесс-функции), полученное на любом этапе работы запоминается и является результатом работы ГП, а само дерево считается решением поставленной задачи.

Обобщенный алгоритм получения дерева с помощью ГП представлен на рисунке 3.3.

Параметры эволюционного алгоритма

В процессе поиска решений возможно использование параметров генетического алгоритма, приведенных в таблице 3.2.

Таблица 3.2.

Параметры эволюционного алгоритма

Параметр	Допустимые значения
Мощность популяции	Задается
Максимальная глубина особи в начальной популяции	Задается
Максимальная глубина особи в эволюционирующей популяции	Задается
Метод генерации начальной популяции	– растущая p_g =100% – полная p_c =100% – смешенная $p_g + p_c$ =100%
Вероятность функционального узла	50%
Вероятность терминального узла	50%
Вероятность кроссинговера	Задается
Вероятность мутации	Задается
Отбор родителей	– турнир – рулетка

Генерация начальной популяции

На данном этапе происходит генерация начальной популяции, в соответствии с заданными параметрами. Популяция состоит из набора деревьев, сгенерированных случайным образом.

Генерация каждого дерева происходит рекурсивно, начиная с генерации функционального узла и его аргументов. Для каждого дочернего узла (аргумента) случайным образом определяется будет данный узел функциональным или терминальным, далее в соответствии с типом узла выбирается его значение из соответствующего терминального или функционального множества. Процесс выполняется по левой ветви до тех пор, пока не будет выбран дочерним терминальный узел. Затем генерируются правые ветви.

Предусмотрены следующие методы создания деревьев:

1. Полный. При генерации случайного дерева каждая ветвь имеет одинаковую (максимальную) глубину.
2. Растущий. При генерации случайного дерева каждая ветвь может иметь различную глубину, но не более чем заданна максимальная.
3. Комбинированный. Половина деревьев всей популяции генерируется полным методом, вторая половина – растущим.

Применение генетических операций

1. Отбор родителей. Важнейшим фактором, влияющим на эффективность генетического алгоритма, является оператор отбора. Слепое следование принципу «выживает сильнейший» может привести к сужению пространства области поиска и попаданию найденного решения в область локального экстремума целевой функции. С другой стороны, слишком слабый оператор отбора может привести к замедлению роста качества популяции, а значит, и к замедлению поиска. Кроме того, популяция при этом может не только не улучшатся, но и ухудшатся.

Наиболее распространенные варианты отбора родителей:

- случайный равновероятностный;
- рангово-пропорциональный;
- отбор пропорционально значению целевой функции;
- элитный отбор;
- отбор с вытеснением.

Предложено использовать отбор пропорционально значению целевой функции реализованный методом рулетки или турниром.

2. Кроссинговер. Для древообразной формы представления используются следующие три основных операторов кроссинговера:

- узловой кроссинговер;
- кроссинговер поддеревьев;
- смешанный.

В узловом операторе кроссинговера выбираются два дерева и узлы в этих деревьях. Так как узлы в деревьях могут быть разного типа сначала необходимо убедиться, что выбранные узлы у родителей являются взаимозаменяемыми. Если узел во втором родителе не соответствует типу узла первого родителя, то случайным образом выбирается другой узел во втором родителе, который опять подлежит проверке на предмет совместимости. Далее производится обмен узлов. Пример узлового ОК на рисунке 3.5.

В кроссинговере поддеревьев родители обмениваются не узлами, а определяемыми ими поддеревьями. Оператор выполняется следующим образом:

- Выбираются родители. Далее необходимо убедиться, что выбранные узлы взаимозаменяемы, т.е. принадлежат одному типу. Иначе, как и в предыдущем случае выбирается другой узел с последующей проверкой.

– Затем производится обмен поддеревьев, определяемых этими узлами. Вычисляется глубина дерева потомка. Если ожидаемый размер не превышает заданный порог то потомок запоминается.

Этот тип оператора кроссинговера является основным. Обмен поддеревьями возможен и в одном родителе. Пример ОК поддеревьев на рисунке 3.6.

При смешанном операторе кроссинговера для некоторых узлов выполняется узловой оператор кроссинговера, а для других - кроссинговер поддеревьев.

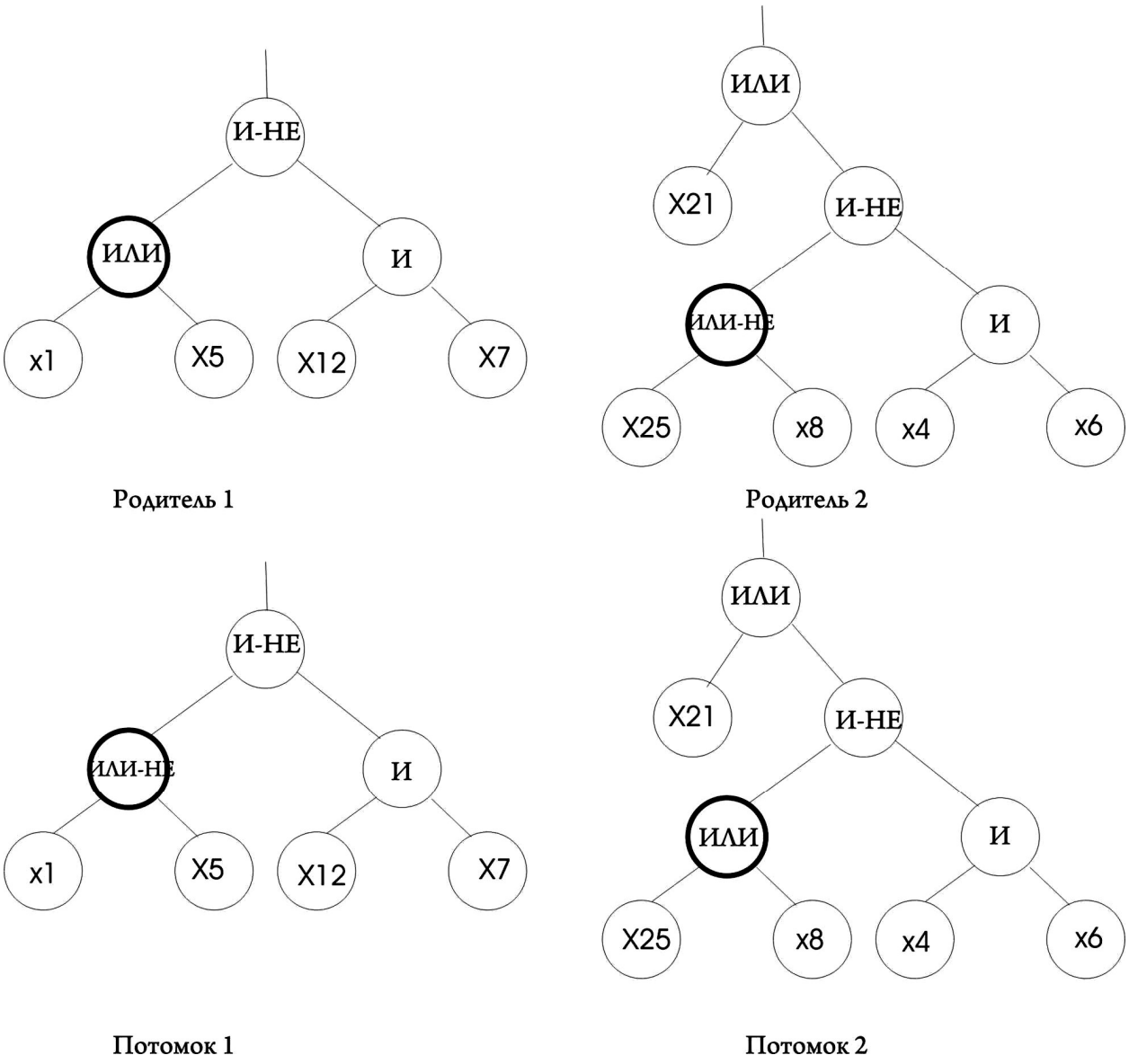

Рисунок 3.5. Пример выполнения узлового ОК.

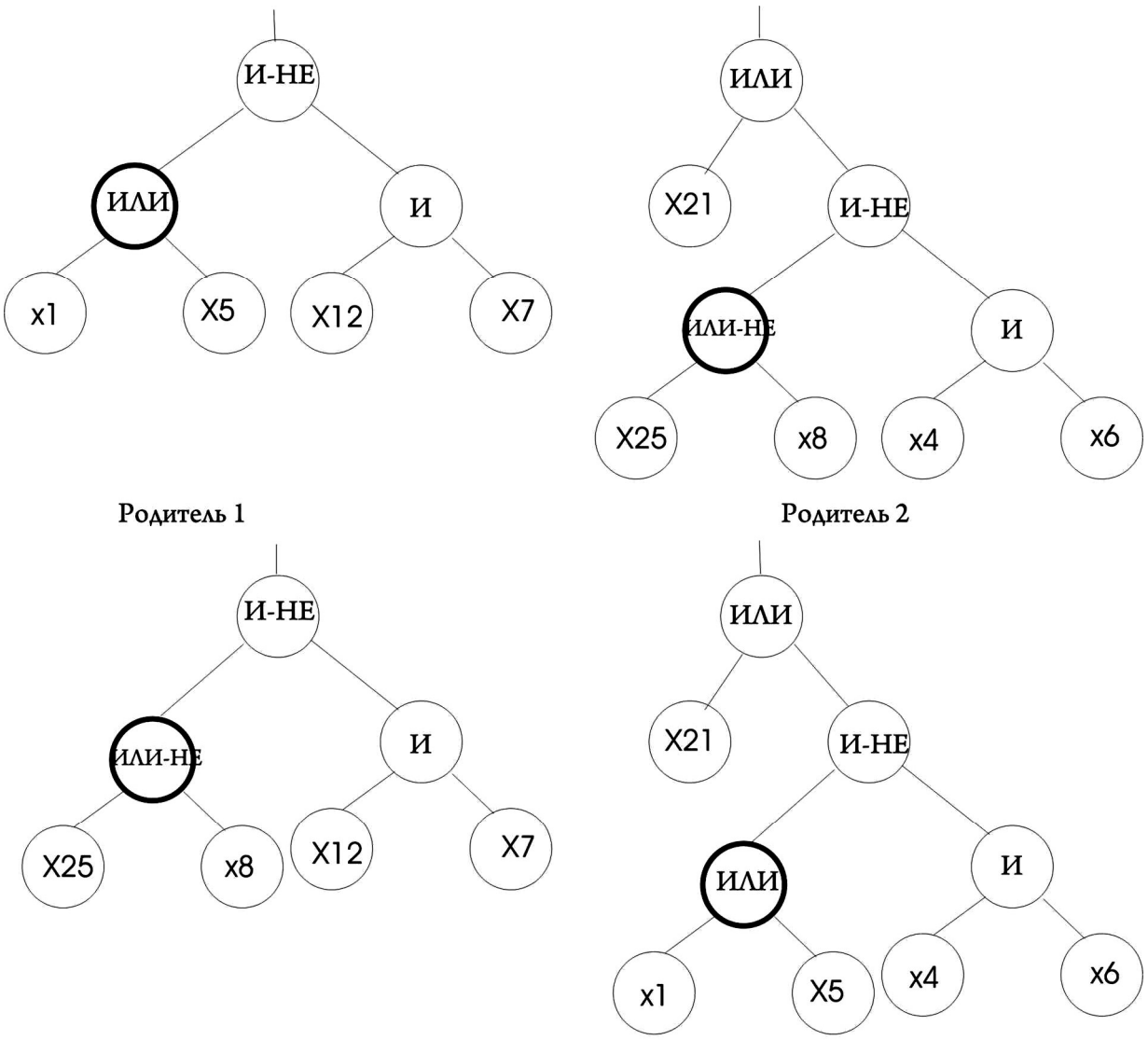

Рисунок 3.6. Пример выполнения ОК поддеревьев.

3. Мутация. Для деревьев используются следующие операторы мутации:

– узловая;
– усекающая;
– растущая.

Узловая мутация выполняется следующим образом:

– выбирается случайным образом узел, подлежащий мутации;
– случайно выбирается из множества типов узлов отличный от рассматриваемого типа узел;
– поменять исходный тип узла на выбранный.

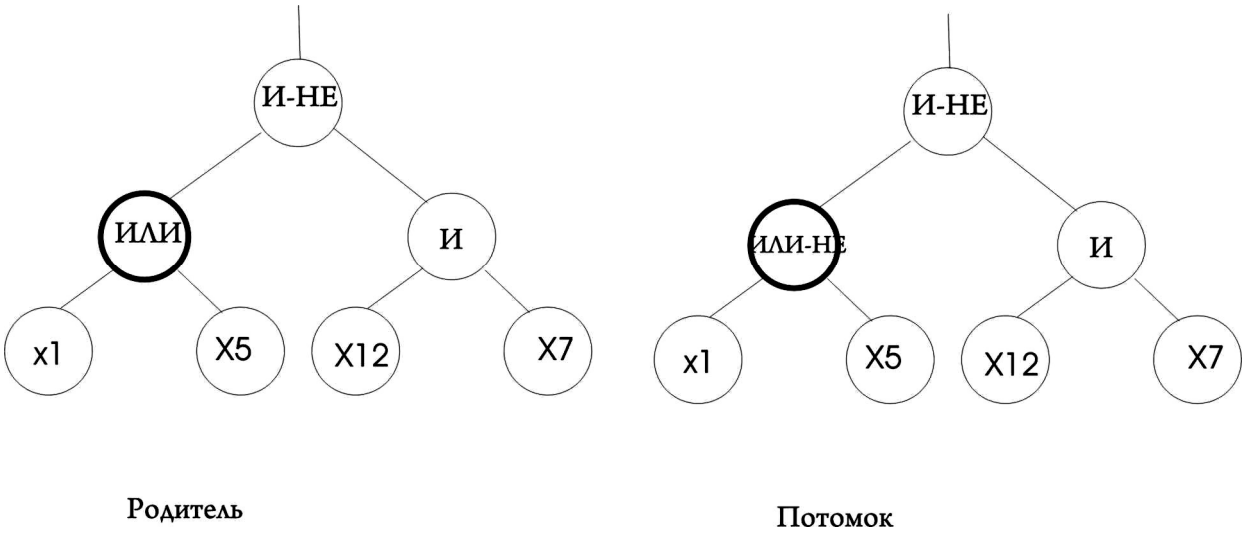

Рисунок 3.7. Пример выполнения узлового ОМ.

Усекающая мутация производится так:

– определяется или выбирается узел;
– случайным образом выбирается терминальный символ из заданного множества;
– обрезается ветвь узла мутации;
– вместо обрезанной ветви помещается выбранный терминальный символ.

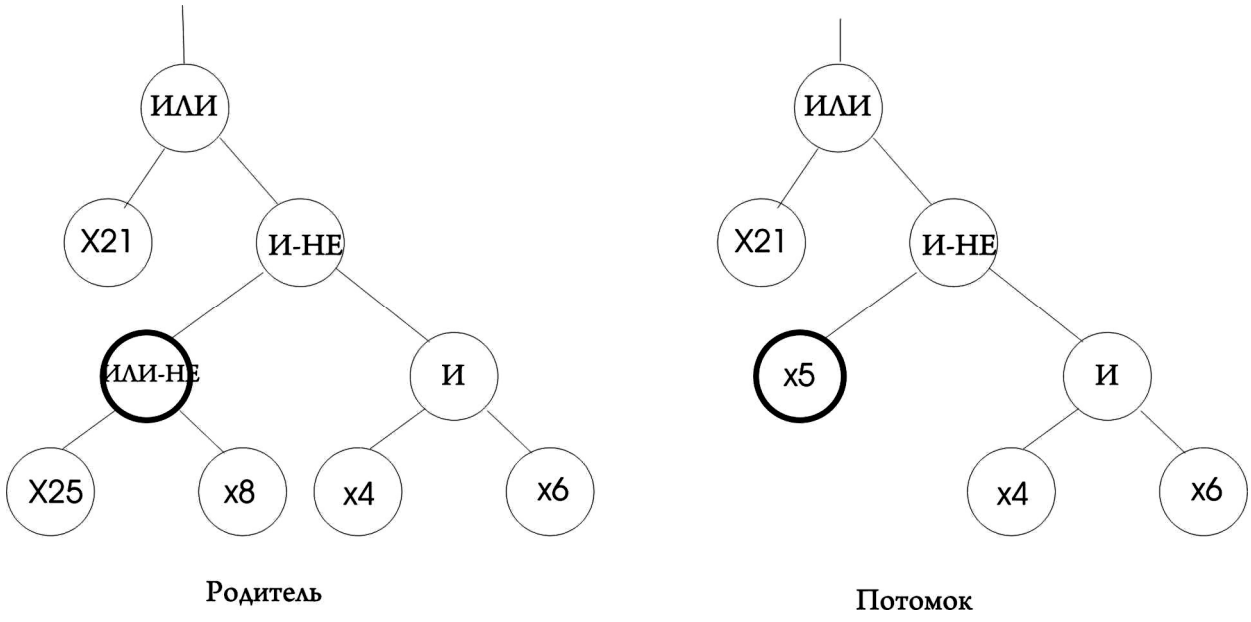

Рисунок 3.8. Пример выполнения усекающего ОМ.

Растущая мутация выполняется следующим образом:

- определяется узел мутации;
- если узел нетерминальный то необходимо отсечь ветви исходящие из него, иначе выбрать другой узел.
- вычислить размер (сложность) остатка дерева;
- вместо отсеченного дерева вырастить случайным образом новое дерево так, чтобы размер нового построенного дерева не превышал заданный порог.

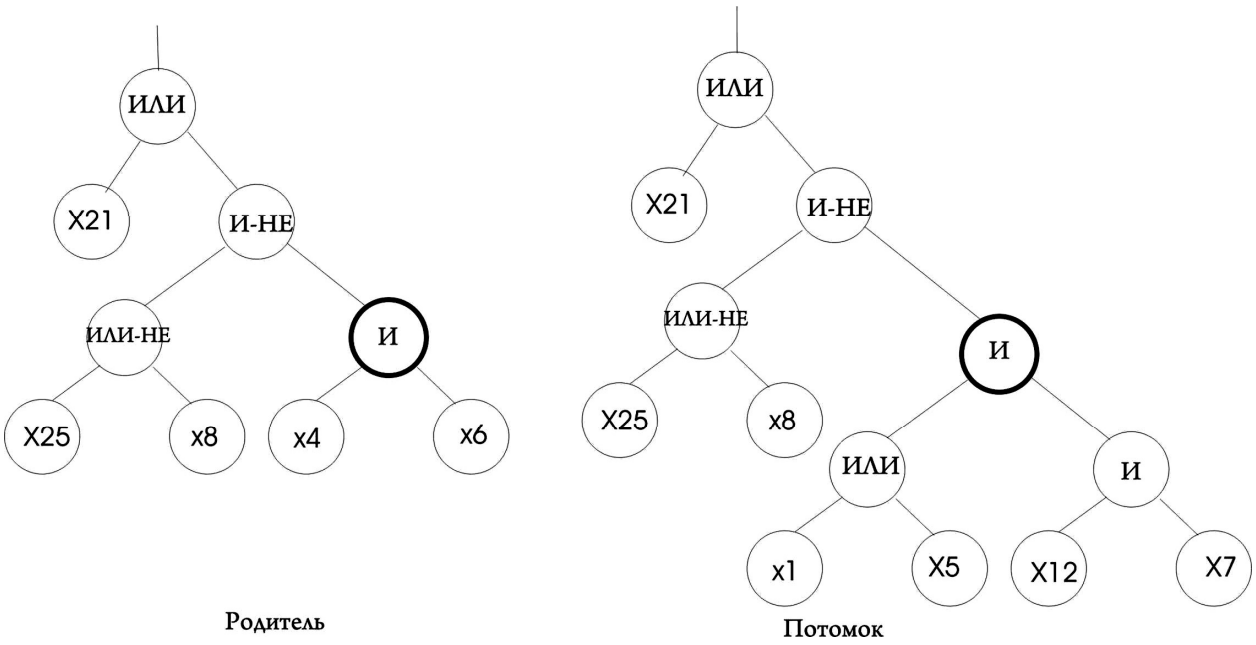

Рисунок 3.8. Пример выполнения растущего ОМ.

Это очень мощный оператор, который обладает большими возможностями.

4. Редукция. Оператор редукция выполняется с целью сохранения размера популяции. Виды оператора практически совпадают с видами оператора отбора родителей. Но процедуры редукция и отбора родителей разнесены по действию во времени и имеют разный смысл. Возможно выполнение следующих вариантов редукции:

- элитная стратегия;
- чистая замена;
- равномерная случайная замена (с указанием количества заменяемых особей в %).

Критерий останова

В качестве критерия останова можно использовать указание определенного числа итераций или указание определенного числа повторения лучшего результата.

3.3.3. Применение генетического программирования для получения продукционных правил.

Основная идея данного метода заключается в инновационном методе кодирования особей для генетического программирования [14,46,49]. Как и ранее особь представляет собой дерево, только теперь это дерево соответствует синтаксическому выражению, представляющее булеву функцию в дизъюнктивной нормальной форме, что полностью соответствует (2.5) и (2.6) в постановке задачи поставленной в разделе 2.3.

На рисунке 3.9. Представлен пример дерева соответствующего функции представленной в дизъюнктивной нормальной форме.

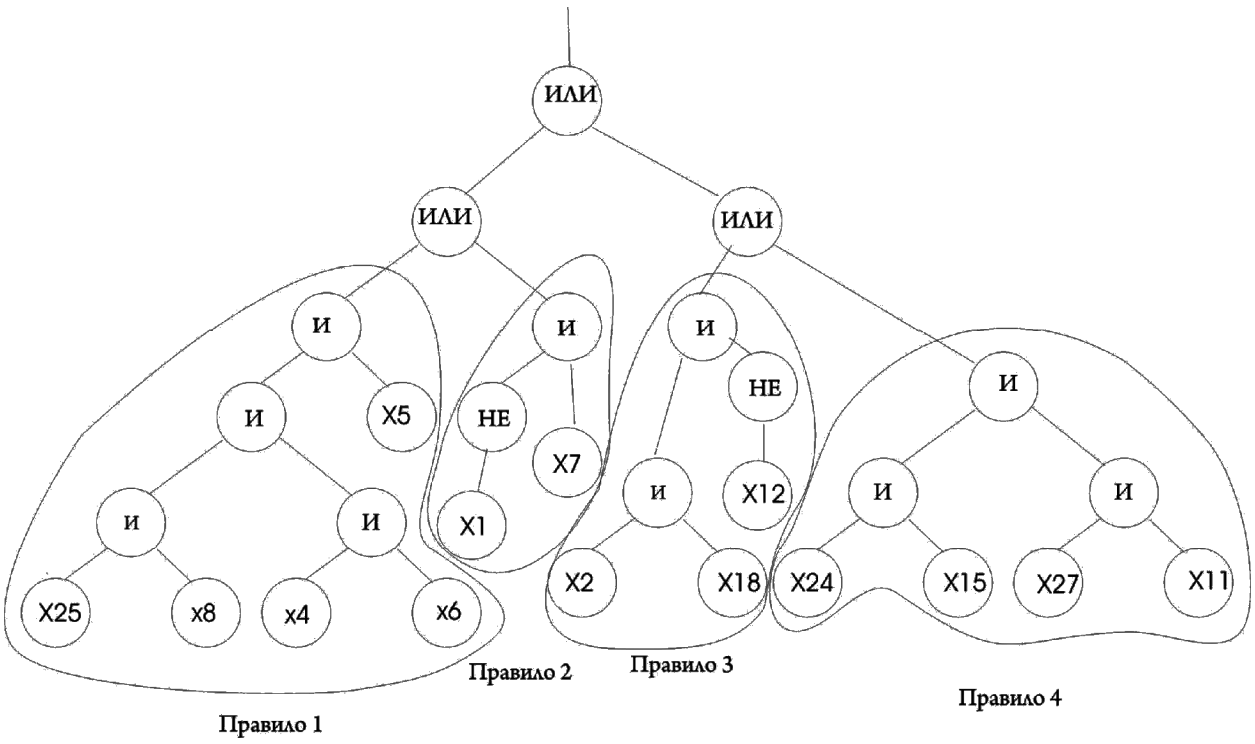

Рисунок 3.9. Пример дерева, которое представляет булеву функцию в дизъюнктивной нормальной форме.

Дерево представлено 4-мя правилами. Такое представление особи

значительно упрощает интерпретацию результата. В данном примере расшифровка будет следующей:

ЕСЛИ выполняется правило 1 ИЛИ выполняется правило 2 ИЛИ выполняется правило 3 ИЛИ выполняется правило 4, ТО результат 1, ИНАЧЕ результат 2.

Где правила 1-4 соответствуют булевым выражениям (3.4)-(3.7), а выполнение какого-либо правила означает, что значение соответствующего ему выражения равно 1.

$$f_1(x_1, x_2, ..., x_n) = x_{25}x_8 \cdot x_4x_6 \cdot x_5 \quad (3.4)$$

$$f_2(x_1, x_2, ..., x_n) = \bar{x}_1x_7 \quad (3.5)$$

$$f_3(x_1, x_2, ..., x_n) = x_2x_{18}\bar{x}_{12} \quad (3.6)$$

$$f_4(x_1, x_2, ..., x_n) = x_{24}x_{15} \cdot x_{27}x_{11} \quad (3.7)$$

Соответственно все дерево можно представить выражением (3.8).

$$F(x_1, x_2, ..., x_n) = x_{25}x_8 \cdot x_4x_6 \cdot x_5 + \bar{x}_1x_7 + x_2x_{18}\bar{x}_{12} + x_{24}x_{15} \cdot x_{27}x_{11} \quad (3.8)$$

Выполнение равенства (3.9) означает выполнение конструкции ЕСЛИ, т.е. результат 1, не выполнение – ИНАЧЕ т.е. результат 2.

$$F(x_1, x_2, ..., x_n) = 1 \quad (3.9)$$

В общем виде дерево можно описать выражением (3.10).

$$F(x_1, x_2, ..., x_n) = \sum_{j=1}^{m} f_j(x_1, x_2, ..., x_n), \quad (3.10)$$

где m – количество продукционных правил, n – размер терминального множества. Применительно к нашей задаче выполнение равенства (3.9) будет означать высокую степень риска СВСГР, не выполнение – низкую.

Предобработка входных данных

Как и в предыдущем варианте реализации ГП данные должны быть предварительно обработаны, с целью преобразования входного обучающего множества в булевы переменные, как показано в приложении Д.

Терминальное множество

Терминальное множество, как и в предыдущем варианте состоит из оптимального набора информативных параметров, полученного в разделе 2.7, который после предобработки представлен булевыми переменными.

Функциональное множество

Функциональное множество состоит из логических операций: AND, OR, NOT.

Фитнесс-функция

В качестве фитнесс-функции будем как и ранее будем использовать функцию (3.2), долю пациентов с правильно поставленным диагнозом.

Генерация начальной популяции

На данном этапе происходит генерация начальной популяции, определенного набора деревьев, сгенерированных случайным образом. Генерация каждого дерева происходит рекурсивно, начиная с генерации первым функционального узла ИЛИ и его аргументов. В качестве аргументов на первом шаге может быть только узел ИЛИ. Далее для каждого дочернего узла случайным образом определяется тип и значения его аргументов по следующим принципам:

- после узла ИЛИ может быть только функциональный узел (значениями которого могут быть – ИЛИ или И);
- после узла И может быть функциональный узел (значениями которого могут быть – И или НЕ) или терминальные узлы;
- после узла НЕ может быть только терминальный узел.

Процесс выполняется по левой ветви до тех пор, пока не будет выбран дочерним терминальный узел. Затем генерируются правые ветви.

Вероятность функционального и терминального узлов меняется по

следующему принципу: чем ниже вершина, тем больше вероятность терминального узла и меньше функционального (3.11)-(3.13). Для функционального узла на каждом последующем шаге увеличивается вероятность узла И и уменьшается вероятность узла ИЛИ (3.14)-(3.15).

$$P_{and} = 75 - \left(\frac{75 \bullet (G_i - 4)}{G - 1 - 4} \right), \quad (3.11)$$

$$P_{not} = (100 - P_{and}) \bullet \left(\frac{50}{100} \right), \quad (3.12)$$

$$P_{term} = (100 - P_{and}) \bullet \left(\frac{50}{100} \right), \quad (3.13)$$

$$P_{or} = 50 - \left(\frac{50 \bullet (G_i - 3)}{G - 1 - 3} \right), \quad (3.14)$$

$$P_{and} = 100 - P_{or}, \quad (3.15)$$

где P_{and} – вероятность функционального узла И; P_{not} - вероятность функционального узла НЕ; P_{or} - вероятность функционального узла ИЛИ; G_i – текущая глубина вершины; G - максимальная глубина дерева. Формулы (3.11)-(3.13) определяют вероятности узла следующего после узла И, а (3.14)-(3.15) после узла ИЛИ.

При формировании дерева в одной ветви ИЛИ (т.е. для одного правила) не используется один и тот же терминальный символ более одного раза.

Предусмотрены методы создания деревьев как и предыдущем варианте использования ГП: полный, растущий и комбинированный.

Применение генетических операций

Отбор родителей. Предложено использовать отбор пропорционально значению целевой функции реализованный методом рулетки или турниром. При этом если два или более потомка имеют одинаковое значение фитнесс-функции, то выбирается дерево минимальной сложности.

Под сложностью дерева будем понимать:

$$Comp = 2 \bullet N_{rule} + N_{term} \quad (3.16)$$

где N_{rule} – количество правил; N_{term} – количество терминальных узлов

1. Кроссинговер. Учитывая строго определенное представление дерева необходимо модифицировать операторы кроссинговера:

- в узловом операторе кроссинговера обмен возможен только для терминальных узлов, рисунок 3.10;
- в кроссинговере поддеревьев родители могут обмениваться только поддеревьями ветви И, рисунок 3.11;

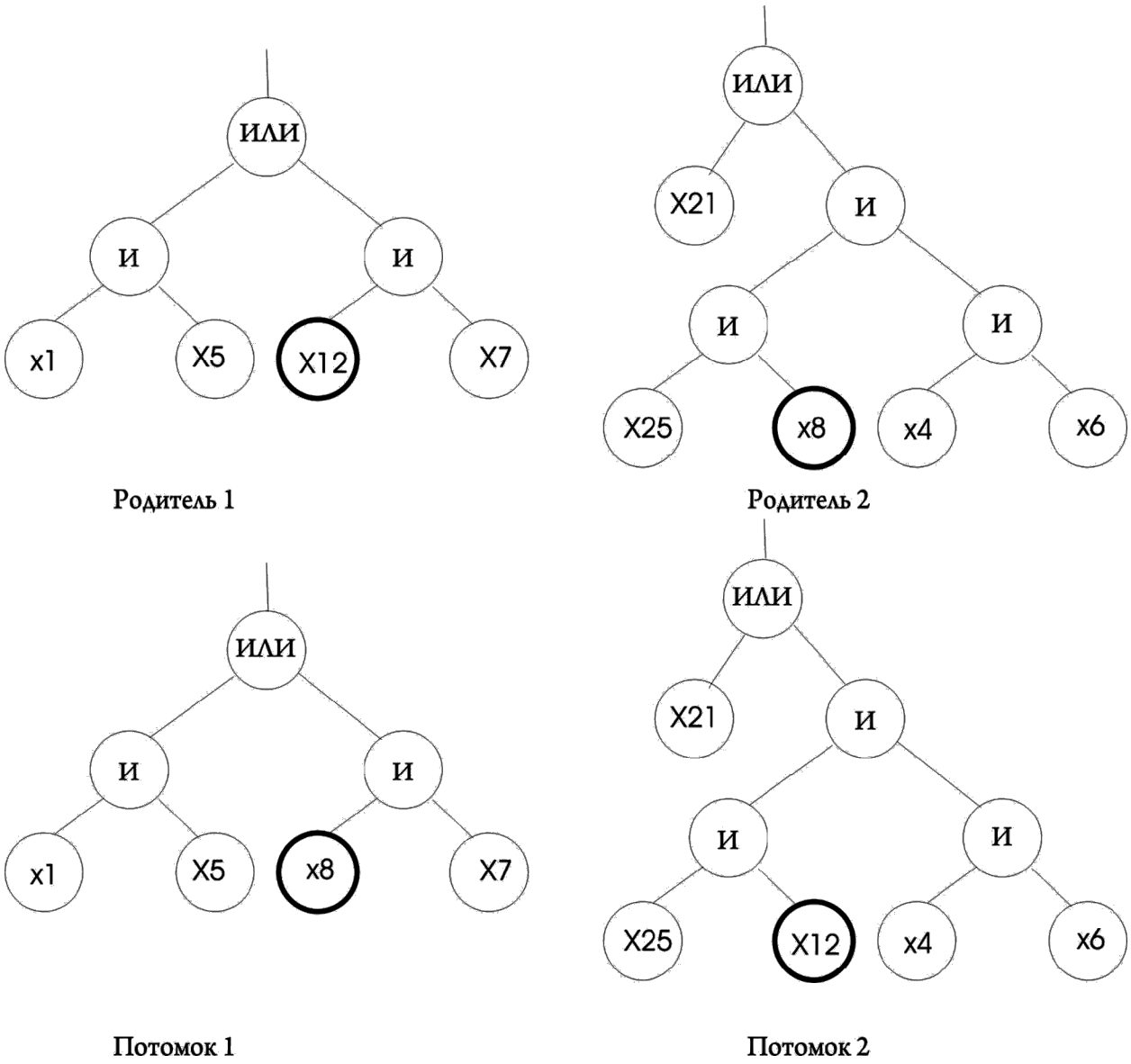

Рисунок 3.10. Пример выполнения узлового ОК.

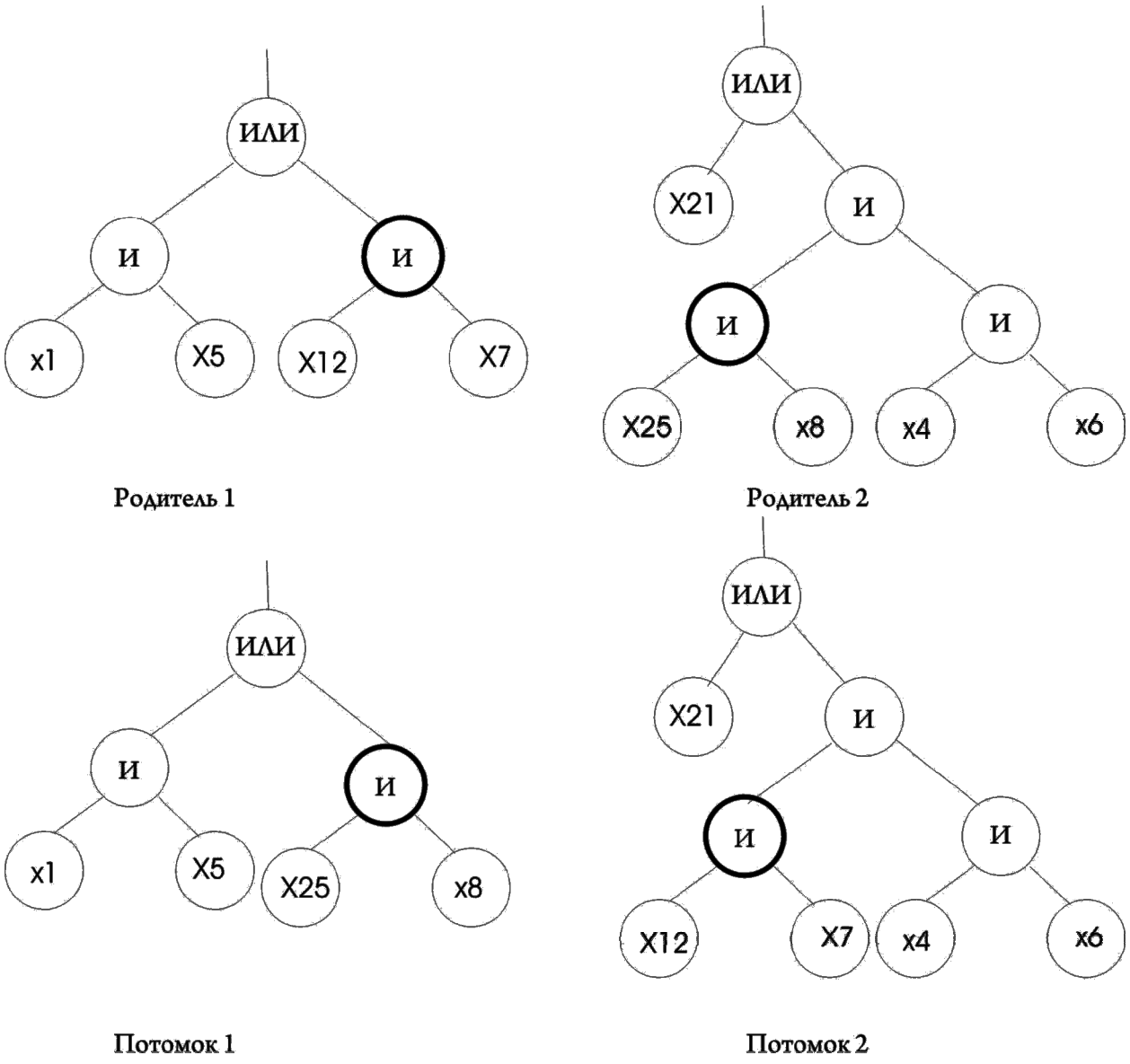

Рисунок 3.11. Пример выполнения ОК поддеревьев.

При смешанном операторе кроссинговера для некоторых узлов выполняется узловой оператор кроссинговера, а для других - кроссинговер поддеревьев.

Так же предлагается выполнять оператор кроссинговера для худшего правила в дереве. Каждое правило (ветвь узла ИЛИ) можно рассматривать как отдельное дерево, способное решать поставленную задачу, поэтому вычисление фитнесс-функции для каждого правила в отдельности (3.17) логически обосновано. Правило считается худшим, у которого значение фитнесс-функции (3.17) минимальное.

$$E_j = \frac{1}{M} * (M - \sum_{i=1}^{M} (| F_{ij} - Y_{ij} |)), j \in [1, N_{rule}] \quad (3.17)$$

где N_{rule} – количество правил; M - количество обучающих примеров.

Вычисление фитнесс функции не только для каждого правила в отдельности, но и каждого узла И также имеет смысл. При выполнении оператора кроссинговера поддеревьев предлагается осуществлять поиск точки разрыва следующим образом: вычисляется фитнесс-функция для каждого узла И начиная с первого снизу. Если значение фитнесс-функции для узла И находящегося выше, хуже чем на предыдущем шаге то обмену подлежит один из узлов аргументов данного узла И.

2. Мутация. Как и в случае с оператором кроссинговера оператор мутации должен быть модифицирован:

- узловая мутация выполняется для терминального узла или первой снизу вершины ИЛИ, рисунок 3.12;
- усекающая мутация выполняется только для узлов И или НЕ, рисунок 3.13;
- при растущей мутация ветви наращиваются согласно правилам инициализации деревьев;

3. Редукция деревьев. Выполняется усечение деревьев, которые распознают менее определенного процента диагнозов. Если дерево имеет более чем задано минимальное количество правил, то обрезаются худшие правила. В противном случае обрезаются ветви правил.

Редукция. Предлагается использовать выполнения следующих вариантов редукции:

- элитная стратегия;
- чистая замена;
- равномерная случайная замена (с указанием количества заменяемых особей в %).

Обобщенный алгоритм получения продукционных правил с помощью ГП представлен на рисунке 3.3. При выполнении алгоритма в блоке

«результат» происходит интерпретация результата: переход от древовидного представления к продукционным правилам.

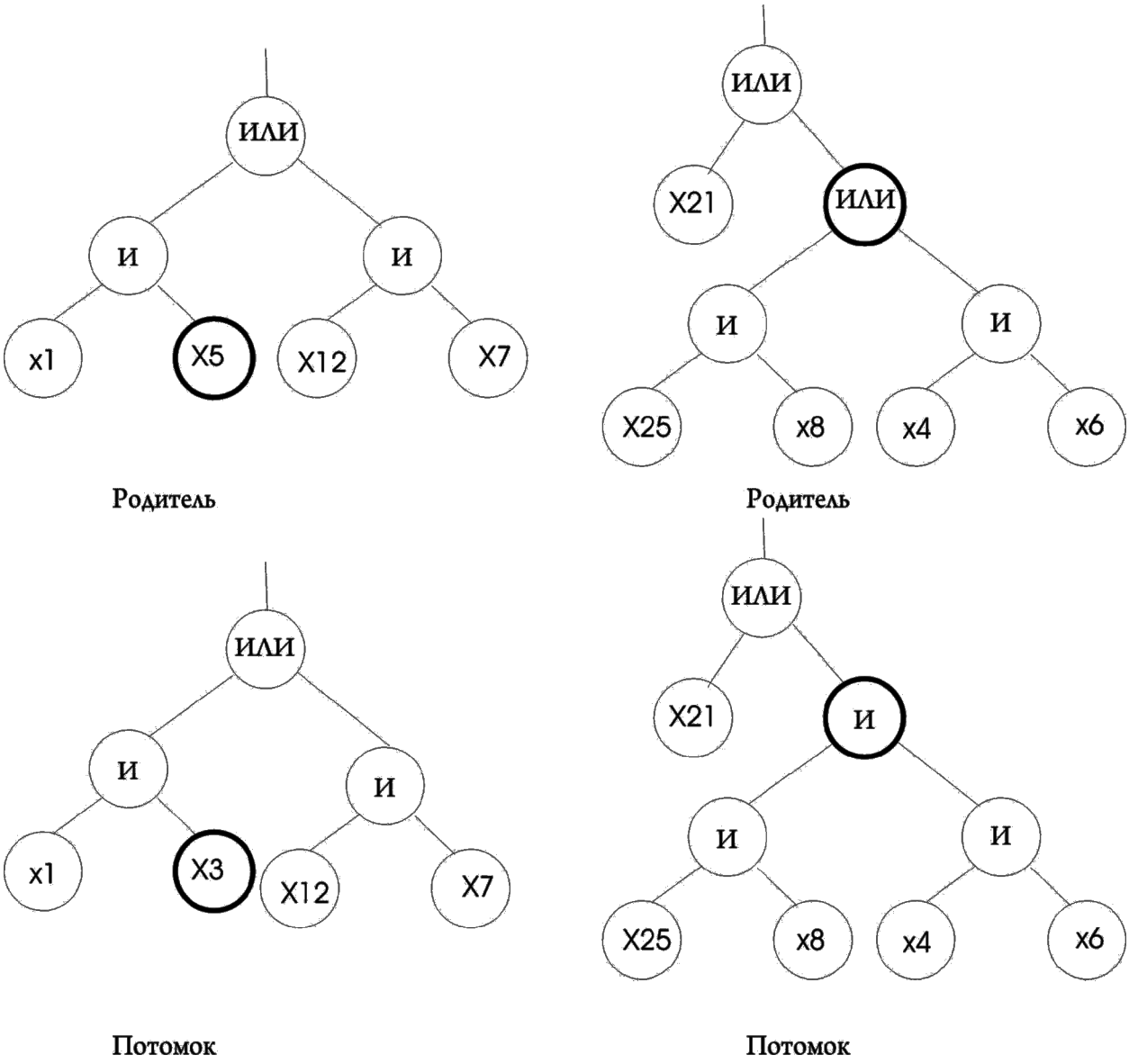

Рисунок 3.12. Пример выполнения узлового ОМ.

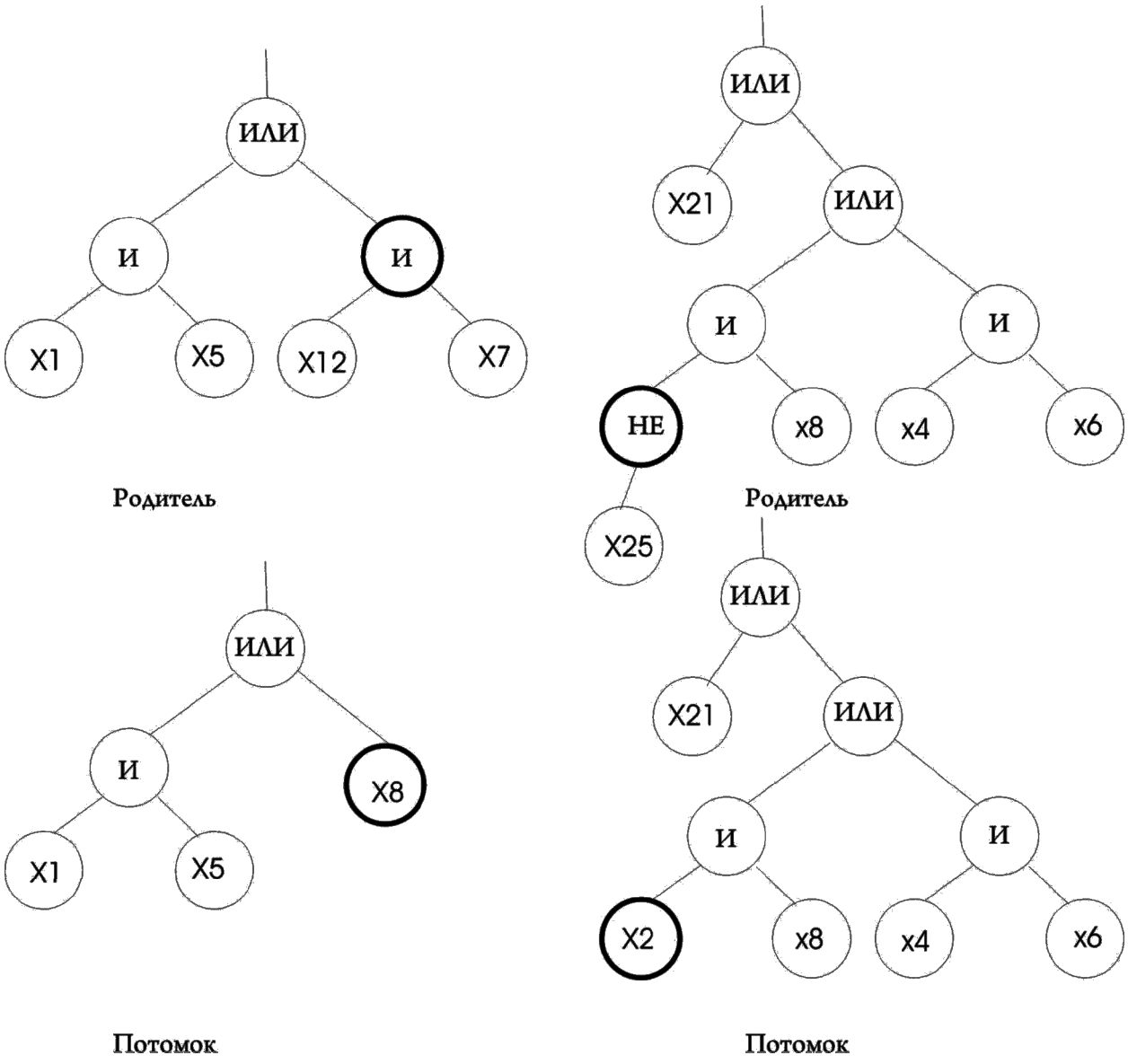

Рисунок 3.13. Пример выполнения усекающего ОМ.

3.3.4. Применение генетического программирования для получения продукционных правил в условиях неопределенности.

При работе с медицинскими данными, достаточно часто возникает ситуация, когда некоторые параметры неизвестны. Это затрудняет, как и обучение системы, так и ее тестирование, а также использование. При формировании обучающих данных используются данные, предоставленные медицинскими работниками. Как правило, эти данные собираются по карточкам пациентов, которые находились на лечении несколько лет назад. Поэтому при отсутствии некоторой информации практически не возможно ее

восстановить. Автоматизированные методы формирования знаний на базе машинного обучения (machine learning) [72,86] работают если известны все выделенные факторы риска для каждого пациента. Если какой-нибудь параметр неизвестен только у одного пациента необходимо, либо удалить пациента из обучающей выборки, либо удалить данный параметр из списка факторов риска. Так как в большинстве случаев у разных пациентов отсутствуют данные о разных факторах риска, формирование обучающей выборки выполняется с существенной потерей данных.

После разработки системы список влияющих параметров уже строго определен и для корректной работы системы все информативные составляющие должны быть заполнены. При тестировании отсутствие информации сказывается на достоверности результата или невозможности диагностирования в целом.

С целью минимизировать потерю данных при обучении и получить возможность диагностировать при неизвестных значениях некоторых факторов риска предлагается использовать троичную логику в булевых вычислениях [50] .

В таблицах 3.3-3.5 приведены таблицы истинности для следующих логических функций: И, ИЛИ и НЕ.

Таблица 3.3

N_1	N_2	И
0	0	0
0	1	0
1	0	0
1	1	1
*	0	0
*	1	*
*	*	*

Таблица 3.4

N_1	N_2	ИЛИ
0	0	0
0	1	1
1	0	1
1	1	1
*	0	*
*	1	1
*	*	*

Таблица 3.5

N_1	НЕ
0	1
1	0
*	*

Применение системы, которая оперирует с неизвестными состояниями, позволит выполнять диагностику даже при отсутствии некоторых

параметров, что не приведет к невозможности функционирования разработанной системы. На этапе обучения, такой подход позволит сформировать оптимально полный набор входных параметров, и не упустить важные, информативные параметры.

Обобщенный алгоритм получения продукционных правил в условиях неопределенности с помощью ГП такой же как и предыдущем разделе и представлен на рисунке 3.3.

3.4. Выводы

1. Реализован метод извлечения знаний на основе корреляционного анализа. Разработана прогностическая таблица для прогнозирования СВСГР, что позволило определять степень риска СВСГР и планировать профилактические меры пациентов, воздействуя на факторы риска которыми можно управлять.
2. Реализован метод извлечения знаний с помощью НС. Разработана архитектура НС, что позволило определять высокую степень риска СВСГР.
3. Получил дальнейшее развитие метод прогнозирования на основе генетического программирования (ГП), что позволило получить дерево способное определять высокую степень риска СВСГР.
4. Получил дальнейшее развитие метод прогнозирования на основе генетического программирования (ГП), что позволило получить продукционные правила для прогнозирования высокой степени риска СВСГР.
5. Получил дальнейшее развитие метод прогнозирования на основе генетического программирования (ГП), что позволило получить продукционные правила для прогнозирования высокой степени риска СВСГР в условиях неопределенности.