ВЫЧИСЛИТЕЛЬНАЯ ТЕХНИКА И ПРИКЛАДНАЯ МАТЕМАТИКА

УДК 681.317.75:519.2

В.П. Корячко, Д.А. Перепелкин

АЛГОРИТМ АДАПТИВНОЙ УСКОРЕННОЙ МАРШРУТИЗАЦИИ НА БАЗЕ ПРОТОКОЛА IGRP ПРИ ДИНАМИЧЕСКОМ ДОБАВЛЕНИИ ЭЛЕМЕНТОВ КОРПОРАТИВНОЙ СЕТИ

Предложен алгоритм адаптивной ускоренной маршрутизации на базе протокола IGRP при динамическом добавлении элементов корпоративной сети, повышающий качество ее функционирования.

Ключевые слова: *адаптивная ускоренная маршрутизация, протокол IGRP, динамические изменения, алгоритмы маршрутизации, корпоративные сети.*

Введение. Цель работы – разработка нового эффективного алгоритма поиска оптимальных маршрутов на базе протокола IGRP при динамическом добавлении элементов корпоративной сети, повышающего качество ее функционирования.

Важнейшим условием повышения конкурентоспособности российских предприятий является внедрение современных информационных технологий (ИТ). Для повышения качества продукции и услуг на предприятиях активно внедряются корпоративные информационные системы. Важнейшим звеном в ИТ - инфраструктуре предприятия являются корпоративные сети, предназначенные для обеспечения взаимодействия различных приложений информационных систем.

Необходимость обеспечения качественного обслуживания современного трафика, передаваемого через IP-сети, обусловливает высокие требования к эффективности передачи пакетов данных от отправителя к получателю. Задача маршрутизации в корпоративных сетях решается при условии, что кратчайший маршрут, обеспечивающий передачу пакета за минимальное время, зависит от топологии сети, пропускной способности и нагрузки на линии связи. Топология сети изменяется в результате развития телекоммуникационной системы при подключении новых узлов и линий связи.

Применение новых перспективных подходов для решения задачи маршрутизации позволяет повысить качество функционирования корпоративной сети за счет уменьшения трудоемкости построения оптимальных маршрутов.

Теоретическая часть. Для повышения качества функционирования корпоративных сетей наиболее важной задачей является выбор эффективного алгоритма маршрутизации, который будет обеспечивать поиск оптимальных маршрутов с учетом различных свойств той или иной корпоративной сети. В настоящее время широкое распространение получили дистанционно-векторные алгоритмы (Distance Vector Algorithm, DVA), которые нашли свое применение в протоколе IGRP *(Interior Gateway Routing Protocol – протокол внутреннего шлюза).*

Протокол IGRP разработан компанией Cisco для больших сетей со сложной топологией и сегментами, которые обладают различной полосой пропускания и задержкой.

Пакеты IGRP передаются с использованием дейтаграмм IP с полем протокола IGP. Пакеты начинаются с заголовка IGRP, за которым сразу же следует заголовок IP. Структура заголовка IGRP показана на рисунке 1.

0 31

Версия
Код операции
Редактирование
ASystem
NInterior
NSystem
NExterior
Контрольная сумма

Рисунок 1 – Структура заголовка IGRP

Версия
Номер версии протокола
(текущее значение – 1)

Код операции
Код операции, связанной с сообщением:
1) Update (обновление);
2) Request (запрос).

Редактирование
Порядковый номер, значение которого уменьшается при каждом внесении изменения в таблицу маршрутизации. Номер редактирования позволяет шлюзам избежать обработки обновлений таблиц маршрутизации, которые уже были учтены.

ASystem
Номер автономной системы. Шлюз может входить в несколько автономных систем, в каждой из которых используется свой протокол IGRP. Для каждой автономной системы используются свои таблицы маршрутизации. Это поле позволяет шлюзу выбрать набор используемых таблиц маршрутизации.

NInterior, NSystem, NExterior
Эти поля показывают номера записей в каждой из трех секций сообщений об обновлении таблиц. Первый элемент (NInterior) является внутренним, следующий (NSystem) – системным и последний (NExterior) – внешним.

Контрольная сумма
Контрольная сумма IP, рассчитанная по тому же алгоритму, который используется для дейтаграмм UDP. При вычислении контрольной суммы принимаются во внимание заголовок IGRP и маршрутная информация, которая следует после заголовка. При расчете поле контрольной суммы предполагается нулевым (не учитывается). Контрольная сумма не включает заголовок IP и не использует виртуальных заголовков, как в UDP и TCP.

Запрос IGRP требует от получателя передать таблицу маршрутизации. Для запросов используется только поле версии, кода операции и ASystem, остальные поля имеют нулевые значения.

Сообщения об обновлении таблиц содержат заголовок, сразу за которым располагается таблица маршрутизации. Количество записей в таблице ограничено размером дейтаграмм (1500 байтов с учетом заголовка IP). При используемой в настоящее время структуре записей таблица может содержать до 104 элементов. Если таблица маршрутизации содержит большее число записей, нужно использовать несколько сообщений.

Протокол IGRP использует несколько типов метрики выбранного пути, по одной на каждый вид QoS (Quality of services). Метрика характеризуется 32-разрядным числом. В однородных средах этот вид метрики вырождается в число шагов до цели. Маршрут с минимальным значением метрики является предпочтительным. Актуализация маршрутной информации для этого протокола производится каждые 90 секунд. Если какой-либо маршрут не подтверждает своей работоспособности в течение 270 с, он считается недоступным. После семи циклов (630 с) актуализации такой маршрут удаляется из маршрутных таблиц. IGRP производит расчет метрики для каждого вида сервиса (ToS – Type of services) отдельно.

Метрика, используемая в IGRP, учитывает:
1) время задержки;
2) пропускную способность самого слабого сегмента пути;
3) загруженность канала;
4) надежность канала.

Время задержки предполагается равным времени, необходимому для достижения места назначения при нулевой загрузке сети. Расчет метрики производится для каждого сегмента пути. Время от времени каждый маршрутизатор широковещательно рассылает свою маршрутную информацию всем соседним маршрутизаторам. Получатель сравнивает эти данные с уже имеющейся информацией и вносит, если требуется, необходимые коррекции. На основании вновь полученной информации могут быть приняты решения об изменении маршрутов. Одним из преимуществ IGRP является простота реконфигурации.

Оптимальный маршрут в протоколе IGRP определяется с использованием комбинированной метрики, вычисляемой по следующей формуле:

Метрика = $[(k1 / b_e) + (k2 \cdot d_c)] \cdot r$ **,** где:

1) k1 и k2 – константы;
2) b_e – пропускная способность канала · (1 - загрузка канала);
3) d_c – топологическая задержка;
4) r – относительная надежность (% пакетов, успешно передаваемых по данному сегменту пути).

По умолчанию определение метрики в протоколе IGRP выполняется по критериям пропускной способности и задержки, значения которых приведены в таблице 1.

Пропускная способность в протоколе IGRP измеряется в величинах, обратных бит/с, умноженных на 10^{10}. Например, если пропускная способность равна *N* Кбит/с, то ее измерением в IGRP будет 10000000 / *N*). Надежность измеряется в долях от 255 (255 соответствует 100 %).

Загрузка измеряется также в долях от 255, а задержка в десятках миллисекунд.

В протоколе IGRP маршрут, имеющий наименьшую комбинированную метрику, считается лучшим. При использовании такой схемы появляется возможность, используя весовые коэффициенты, адаптировать выбор маршрутов к задачам конечного пользователя.

Основные достоинства протокола IGRP:

1) стабильность маршрутов даже в очень больших и сложных сетях;

2) быстрый отклик на изменения топологии сети;

3) минимальная избыточность. Протокол IGRP не требует дополнительной пропускной способности каналов для своей работы;

4) разделение потока данных между несколькими параллельными маршрутами примерно равного достоинства;

5) учет частоты ошибок и уровня загрузки каналов;

6) возможность реализовать различные виды сервиса для одного и того же набора информации.

Таблица 1

Наименование канала передачи данных	Пропускная способность	Задержка
Спутник 500 Мбит/с	20	200 000
Сеть Ethernet 10 Мбит/с	1 000	100
Канал 1 1.544 Мбит/с	6 476	2 000
Канал 2 64 Кбит/с	156 250	2 000
Канал 3 56 Кбит/с	178 571	2 000
Канал 4 10 Кбит/с	1 000000	2 000
Канал 5 1 Кбит/с	10 000000	2 000

В протоколе IGRP для хранения маршрутных данных используются специализированные базы данных. IGRP формирует эту базу данных на основе информации, которую он получит от соседних маршрутизаторов. В простейшем случае находится один путь для каждой из сетей. Сегменты пути характеризуются используемым сетевым интерфейсом, метрикой и маршрутизатором, куда следует сначала послать пакет. Предусматривается возможность разделять информационный поток между несколькими доступными эквивалентными маршрутами. Пользователь может сам разделить поток данных, если два или более пути оказались почти равными по метрике, при этом большая часть трафика будет послана по пути с лучшей метрикой.

Выбор оптимального маршрута в протоколе IGRP определяется по алгоритму Беллмана – Форда. Трудоемкость данного алгоритма составляет порядка $O(N^3)$, где N – число маршрутизаторов в сети.

В работе [1] предложен алгоритм парных переходов, позволяющий за счет сбора дополнительной информации учесть возможные изменения конфигурации корпоративной сети и не производить полный пересчет маршрутных таблиц. Это позволило снизить трудоемкость расчета таблиц маршрутизации до величины порядка $O(k{\cdot}N)$, где k – число фактически выполненных парных переходов.

При динамическом подключении узлов и линий связи корпоративной сети использование данного алгоритма оказывается неэффективным, так как трудоемкость расчета дополнительной информации для осуществления парного перехода оказывается выше существующих алгоритмов.

Разработка новых, более эффективных алгоритмов адаптивной маршрутизации позволяет уменьшить трудоемкость построения таблиц маршрутизации в корпоративных сетях, использующих в своей работе протокол IGRP.

Разработка алгоритма. Для повышения качества функционирования корпоративных сетей на базе протокола IGRP предложен алгоритм адаптивной ускоренной маршрутизации, который позволяет уменьшить трудоемкость построения таблиц маршрутизации до величины $O(N)$ при динамическом добавлении элементов корпоративной сети.

Представим корпоративную сеть в виде неориентированного взвешенного связного графа $G = (V,E,W)$, где V – множество вершин, $|V| = N$, E – множество ребер, $|E| = M$, W – множество весов ребер.

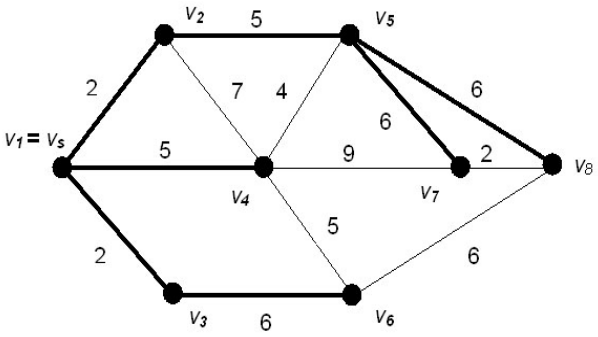

Рисунок 2 – Граф *G* корпоративной сети

Пусть на графе G в некоторый момент времени уже решена задача поиска кратчайших путей до всех вершин множества $V_{\bar{s}} = V\backslash\{v_s\}$ из

начальной вершины v_s, т. е. построено дерево кратчайших путей с корнем в вершине v_s. Обозначим это дерево как T_g. На рисунке 2 жирными линиями обозначено построенное дерево кратчайших путей.

Рассмотрим множество ребер E графа G. По признаку вхождения ребер в дерево T_g можно разделить исходное множество E на два подмножества: $E_T \in T_g$ и $E_R \notin T_g$, $E_T \cup E_R = E$.

Множество ребер дерева E_T – множество ребер дерева T_g для графа G. Для заданного графа G, согласно свойству дерева, мощность множества E_T будет равняться мощности множества V минус единица $|E_T|=|V| - 1$.

Множество ребер замены для дерева E_R – множество ребер графа G, не вошедших в дерево T_g. При соответствующих условиях некоторое ребро $e_{i,j} \in E_R$, инцидентное вершинам v_i и v_j, может перейти во множество ребер дерева E_T, заменив собой некоторое ребро $e_{k,p} \in E_T$. При этом инцидентность ребра $e_{k,p}$ вершине v_i или v_j является обязательным условием. В свою очередь, ребро $e_{i,j}$ перейдет во множество E_R.

Будем называть такие переходы **парными переходами** и обозначать $e_{i,j} – e_{k,p}$.

Во множестве E_R можно выделить 2 подмножества.

Множество ребер замены E_S для дерева – это такое подмножество множества E_R, элементы-ребра которого участвуют, по крайней мере, в одном отношении парного перехода.

Множество непарных ребер E_P – это такое подмножество множества E_R, элементы-ребра которого не участвуют ни в одном отношении из множества R.

В общем случае множество E_P может быть пустым $|E_P| = 0$. Множество E_S будет пустым только при условии, что исходный связный граф G является деревом, и задача поиска кратчайших путей в этом случае лишена смысла.

Для каждого ребра $e_{i,j} \in E$ на шкале значений весов определены точка вхождения в дерево $w_{i,j}^{t}$ и точка вхождения во множество замены $w_{i,j}^{s}$, причем $w_{i,j}^{t} \le w_{i,j}^{s}$.

Так, для графа G, представленного на рисунке 2, множество ребер дерева составляет $E_T = \{e_{1,2}, e_{1,3}, e_{1,4}, e_{2,5}, e_{3,6}, e_{5,7}, e_{5,8}\}$; множество ребер замены $E_S = \{e_{2,4}, e_{4,5}, e_{4,6}, e_{4,7}, e_{6,8}\}$; множество непарных ребер будет $E_P = \{e_{7,8}\}$. Если рассмотреть ребро $e_{2,5}$, то для него точка вхождения в дерево будет составлять 7, а точка вхождения во множество замены – 18. При этом данное ребро находится в отношении парного перехода с ребром $e_{4,5}$, а после попадания $e_{2,5}$ во множество непарных ребер эта парная перестановка примет вид: $e_{4,5} - e_{5,7}$.

Обозначим множество путей до вершины v_i из исходной вершины v_s через Π_i, где элемент множества $\pi_{i,k} \in \Pi_i$ будет множеством не повторяющихся ребер $e_{i,j} \in E$, образующих вместе путь, соединяющий v_s и v_i. Каждому $\pi_{i,k} \in \Pi_i$ поставим в соответствие некоторое число, равное сумме весов, входящих в него ребер, т.е. длину пути $d_{i,k} \in D_i$, где D_i представляет собой множество оценок кратчайших путей до вершины v_i из исходной вершины v_s. Кратчайший путь до вершины v_i будем обозначать π_i, а его оценку длины – d_i.

Для разработки алгоритма адаптивной ускоренной маршрутизации на базе протокола IGRP при динамическом добавлении элементов корпоративной сети сформулируем следующие теоремы.

Теорема 1. При добавлении некоторой вершины V_i для всех вершин, не инцидентных вершине с номером i (т.е. не имеющих ребра $e_{i,k}$), кратчайшие пути и их оценки окажутся без изменения.

Доказательство. Новая вершина V_i не входит в кратчайший путь к некоторой вершине V_j. Следовательно, и в пути к этой вершине $\pi_{v,j}$ отсутствует ребро $e_{i,k}$. Поэтому новый путь к вершине V_j, содержащий данное ребро, будет иметь оценку $d_{j,i} > = d_j$ и дерево кратчайших путей не изменится для всех вершин, не инцидентных вершине V_i. Теорема доказана.

Следствие 1. При добавлении некоторой вершины V_k, имеющей ребро с вершиной V_i и V_j, если $d_i < d_j$, то дерево кратчайших путей до вершины V_i не изменится.

Следствие 2. При добавлении некоторой вершины V_k, имеющей ребро с вершиной V_i и V_j, если $d_i < d_j$, то необходимо определить новые кратчайшие пути для множества вершин, инцидентных вершине V_k, кроме вершины V_i.

Данному случаю соответствует граф, представленный на рисунке 3, при добавлении вершины V_9.

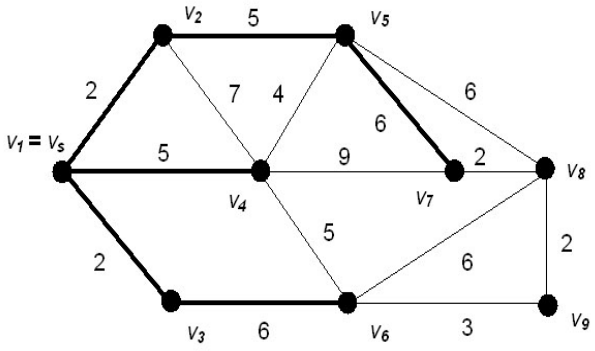

Рисунок 3 - Добавление вершины V_9

На рисунке 3 жирными линиями обозначено

дерево кратчайших путей, которое не требует изменения.

То есть при добавлении вершины V_9 дерево кратчайших путей для вершин V_2, V_3, V_4, V_5, V_6, V_7 согласно теореме 1 не изменится. Для добавленной вершины V_9, а также вершины V_8 необходимо построить новые кратчайшие пути с учетом ребра $e_{6,9}$ с весом $w_{6,9} = 3$ и ребра $e_{9,8}$ с весом $w_{9,8} = 2$. Кратчайший путь до вершины V_8 составит $\pi_8 = \{e_{1,2}, e_{2,5}, e_{5,8}\}$, а его оценка $d_8 = 13$. Кратчайший путь до вершины V_9 составит $\pi_9 = \{e_{1,3}, e_{3,6}, e_{6,9}\}$, а его оценка $d_9 = 11$.

Таким образом, для графа G представленного на рисунке 3, множество ребер дерева составляет $E_T = \{e_{1,2}, e_{1,3}, e_{1,4}, e_{2,5}, e_{3,6}, e_{5,7}, e_{5,8}, e_{6,9}\}$; множество ребер замены $E_S = \{e_{2,4}, e_{4,5}, e_{4,6}, e_{4,7}, e_{6,8}, e_{9,8}\}$; множество непарных ребер будет $E_P = \{e_{7,8}\}$.

Теорема 2. При добавлении нового ребра $e_{i,j}$, инцидентного вершинам i и j, причем i лежит выше, чем j по дереву иерархии, с весом $w_{i,j}$, без изменения окажутся кратчайшие пути и их оценки для вершин множества $V^{(Vi)}$.

Доказательство. Пусть ребро $e_{i,j}$ входит в путь $\pi_{i,k} <> \pi_i$ и $\pi_{j,p} <> \pi_j$. Если данное ребро не уменьшает оценок обеих инцидентных ему вершин v_i и v_j, то есть $d_{i,k} >= d_i$ и $d_{j,p} >= d_j$, дерево кратчайших путей не изменится, так как ребро $e_{i,j}$ оказывает влияние, прежде всего, на инцидентные ему вершины множества V. Так как существовавшие до изменения пути до вершин i и j имели меньшую длину, то ребро $e_{i,j}$ не включается и дерево кратчайших путей не изменится. Если уменьшилась оценка какой-либо инцидентной вершины, например v_j, то эта оценка $d_{j,p}$ будет оценкой кратчайшего пути до вершины v_j и ребро $e_{i,j}$ войдет в новое дерево кратчайших путей, так как не существует другого кратчайшего пути π_j до вершины v_j, кроме пути $\pi_{j,p}$, содержащего ребро $e_{i,j}$. Этот кратчайший путь $\pi_{j,p}$ не будет существовать, если не будут существовать кратчайшие пути до всех промежуточных вершин $v_k \in V^{(Vi)}$ этого пути. Невозможно будет сказать, останутся ли неизмененными кратчайшие пути до остальных вершин графа. Теорема доказана.

Следствие. При добавлении нового ребра $e_{i,j}$, инцидентного вершинам i и j, с весом $w_{i,j}$ при не изменении оценок d_i и d_j дерево не изменится. Если изменяется оценка какой–либо вершины v_i или v_j, то необходимо определить новые кратчайшие пути для множества вершин, не принадлежащих множеству $V^{(Vi)}$.

Данному случаю соответствует граф, представленный на рисунке 4, при добавлении ребра $e_{6,7}$.

То есть при добавлении ребра $e_{6,7}$ с весом $w_{6,7} = 2$ дерево кратчайших путей для вершин V_2, V_3, V_4, V_6 не изменится согласно теореме 2. Для вершин V_5, V_7, V_8 необходимо определить новые кратчайшие пути с учетом добавленного ребра $e_{6,7}$. Кратчайший путь до вершины V_5 составит $\pi_5 = \{e_{1,2}, e_{2,5}\}$, а его оценка $d_5 = 7$. Кратчайший путь до вершины V_7 составит $\pi_7 = \{e_{1,3}, e_{3,6}, e_{6,7}\}$, а его оценка $d_7 = 10$. Кратчайший путь до вершины V_8 составит $\pi_8 = \{e_{1,2}, e_{2,5}, e_{5,8}\}$, а его оценка $d_8 = 13$.

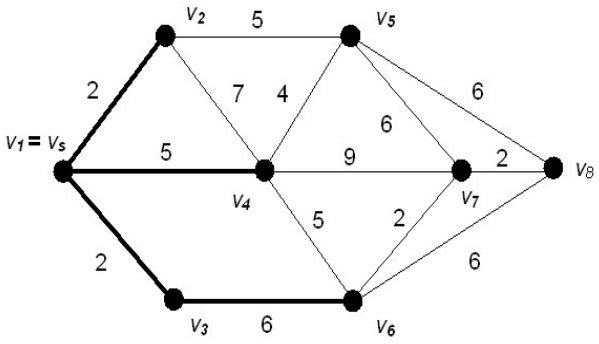

Рисунок 4 - Добавление ребра $e_{6,7}$

На рисунке 4 жирными линиями обозначено дерево кратчайших путей, которое не требует изменения.

Таким образом, для графа G представленного на рисунке 4, множество ребер дерева составляет $E_T = \{e_{1,2}, e_{1,3}, e_{1,4}, e_{2,5}, e_{3,6}, e_{6,7}, e_{5,8}\}$; множество ребер замены $E_S=\{e_{2,4}, e_{4,5}, e_{4,6}, e_{4,7}, e_{6,8}\}$; множество непарных ребер будет $E_P=\{e_{7,8}\}$.

Использование доказанных выше теорем позволяет разработать алгоритм адаптивной ускоренной маршрутизации на базе протокола IGRP, уменьшающий размерность задачи поиска кратчайших путей при динамическом добавлении элементов корпоративной сети.

Рассмотрим работу алгоритма адаптивной ускоренной маршрутизации на базе протокола IGRP в корпоративных сетях. Укрупненная схема алгоритма имеет следующий вид.

Шаг 1. Первоначальная инициализация соседей. Используя пакет HELLO протокола IGRP, определить веса линий связи $w_{i,j}$ до ближайших соседей.

Шаг 2. Используя сообщение query (запрос), получить маршрутную информацию от соседей для построения таблиц маршрутизации.

Шаг 3. Таблицы маршрутизации построены:

а) если да, то перейти к шагу 9;

б) иначе – к шагу 4.

Шаг 4. Построить дерево оптимальных маршрутов корпоративной сети.

Шаг 5. Для вершины, являющейся листом дерева, производится поиск всех парных переходов

без ограничений. Эти списки для удобства дальнейшей работы привязываются к вершине, инцидентной рассматриваемому ребру и расположенной ниже по иерархии.

Шаг 6. Если вершина не является листом дерева, то вычисляются парные переходы для этой вершины и выбираются лучшие значения потенциалов парных переходов для потомков вершины и собственных парных переходов. Подобная процедура выполняется для формирования списков парных переходов в случае динамического добавления элементов корпоративной сети.

Шаг 7. Для каждой вершины формируется полный список парных переходов. Число элементов в каждом из этих списков не превышает количества вершин графа. Такое решение позволяет отказаться от предварительной сортировки потенциалов или приращений для парных переходов без значительного усложнения алгоритма обработки изменения.

Шаг 8. Для каждого ребра графа корпоративной сети определить точку вхождения в дерево оптимальных маршрутов и точку вхождения во множество замены.

Шаг 9. Определить, есть ли пакеты на передачу:

а) если да, то перейти к шагу 10;

б) иначе – к шагу 17.

Шаг 10. Используя поля «Время жизни» и «Контрольная сумма заголовка» протокола IP, определить, требуется ли уничтожить (отбросить) данный пакет:

а) если да, то перейти к шагу 16;

б) иначе - к шагу 11.

Шаг 11. Используя таблицы маршрутизации, определить, требуется ли разделять информационный поток между несколькими доступными эквивалентными маршрутами:

а) если да, то перейти к шагу 12;

б) иначе – к шагу 16.

Шаг 12. Разделить информационный поток между несколькими доступными эквивалентными маршрутами.

Шаг 13. а) передать пакеты по доступным эквивалентным маршрутам;

б) установить флаг передачи.

Шаг 14. Для исключения осцилляции маршрутов сделать временную задержку.

Шаг 15. Передать пакет со служебной информацией соседним маршрутизаторам.

Шаг 16. Проверка флага передачи:

а) если флаг установлен, то перейти к шагу 27;

б) иначе – к шагу 9.

Шаг 17. Послать сообщение query (запрос) соседним маршрутизаторам.

Шаг 18. Ожидать ответа от соседей.

Шаг 19. Сообщение reply (отклик) от соседних маршрутизаторов получен:

а) если да, то перейти к шагу 20;

б) иначе – к шагу 18.

Шаг 20. Анализируя полученную протоколом IGRP информацию, определить, произошло ли добавление элементов корпоративной сети:

а) если да, то перейти к шагу 21;

б) иначе - к шагу 9.

Шаг 21. а) *если добавилась новая вершина V_k, имеющая ребра с вершинами V_i и V_j, причем $d_i < d_j$ (i-вершина расположена ниже по иерархии в дереве кратчайших путей), то:*

1) дерево кратчайших путей до вершины V_i не изменится;

2) для всех вершин, не инцидентных вершине V_i, дерево кратчайших путей не изменится;

3) потенциал вершины V_j (путь d_j до вершины V_j) с учетом новых связей $d_{i,k}$ и $d_{k,j}$ не изменился. Первоначальное дерево кратчайших путей не изменится. Из возможных путей $d_i + d_{i,k}$ и $d_j + d_{k,j}$ выбрать путь минимальной длины и включить его в дерево кратчайших путей. Перейти к пункту 5;

4) потенциал вершины V_j (путь d_j до вершины V_j) с учетом новых связей $d_{i,k}$ и $d_{k,j}$ уменьшился (то есть $d_i + d_{i,k} + d_{k,j} < d_j$). Включить в дерево кратчайших путей ребра $d_{i,k}$ и $d_{k,j}$. Для всех вершин V_m, инцидентных вершине V_j, определить их новые потенциалы. Если потенциал до вершины V_m уменьшился, то ребро $d_{j,m}$ включить в дерево кратчайших путей, исключив первоначальное ребро до вершины V_m;

5) для всех ребер, включенных в дерево кратчайших путей, определить возможные ребра для парных переходов и рассчитать для них точки вхождения во множество ребер замены и в дерево кратчайших путей;

б) *если добавилось новое ребро $e_{i,j}$ с весом $w_{i,j}$, причем вершина V_i располагается ниже по иерархии в дереве оптимальных маршрутов, чем вершина V_j, то:*

1) оценка длины маршрута для вершины V_j не изменилась. Дерево оптимальных маршрутов не изменится. Определить, находится ли ребро $e_{i,j}$ в отношении парного перехода к какому-либо ребру для вершины V_j. Если да, то включить его во множество вершин, состоящих в отношении парного перехода к вершине V_j. Если нет, то оставить все без изменения;

2) оценка длины маршрута для вершины V_j уменьшилась. Включить ребро $e_{i,j}$ в дерево оптимальных маршрутов. Для всех вершин, кроме V_i, инцидентных вершине V_j, рассчитать новые оценки маршрутов. Для тех

вершин, у которых оценки уменьшились, соответствующее инцидентное ребро включить в дерево оптимальных маршрутов, удалив из дерева ребро, входящее в оптимальный маршрут до этой вершины.

Шаг 22. Используя список парных переходов, определить, требуется ли сделать парный переход:

а) если да, то перейти к шагу 23;

б) иначе – к шагу 24.

Шаг 23. Для каждой вершины, у которой в список возможных маршрутов входит ребро с изменившейся метрикой, определить путь минимальной длины и поместить его в дерево кратчайших путей.

Шаг 24. Для каждого ребра графа корпоративной сети пересчитать точки вхождения в дерево оптимальных маршрутов и точки вхождения во множество замены.

Шаг 25. Построить новое дерево оптимальных маршрутов с учетом изменений.

Шаг 26. Сформировать таблицы маршрутизации.

Шаг 27. Проверка окончания работы маршрутизатора:

а) если да, то перейти к шагу 28;

б) иначе – сбросить флаг передачи и перейти к шагу 9.

Шаг 28. Завершение работы маршрутизатора.

Результаты моделирования. Для подтверждения правильности предложенного алгоритма адаптивной ускоренной маршрутизации на базе протокола IGRP при динамическом добавлении элементов корпоративной сети разработано программное обеспечение моделирования процессов маршрутизации.

При разработке основное внимание уделялось корректности предлагаемого алгоритма и размерности решаемой задачи.

Для каждого испытания на множестве обработанных изменений выбирались минимальное, максимальное и среднее значения размерности задачи, выраженные через количество вершин, для которых необходим поиск кратчайшего пути. Для каждого эксперимента были найдены значения оценок математического ожидания и среднего квадратичного отклонения числа изменений. Для алгоритма адаптивной ускоренной маршрутизации на базе протокола IGRP определялось число фактически выполненных парных переходов при динамическом добавлении элементов корпоративной сети.

В таблице 2 приведены обобщенные статистические характеристики для средней размерности задачи. В представленной таблице через СКО обозначено значение оценки среднего квадратичного отклонения.

Таблица 2

Число вершин графа	Min значение	Max значение	Значение оценки МО	Значение оценки СКО
10	0	0,4	0,140	0,1213
100	0	0,05	0,0135	0,0129
200	0	0,055	0,0058	0,0073

На рисунках 5 – 7 представлены результаты моделирования алгоритма адаптивной ускоренной маршрутизации на базе протокола IGRP.

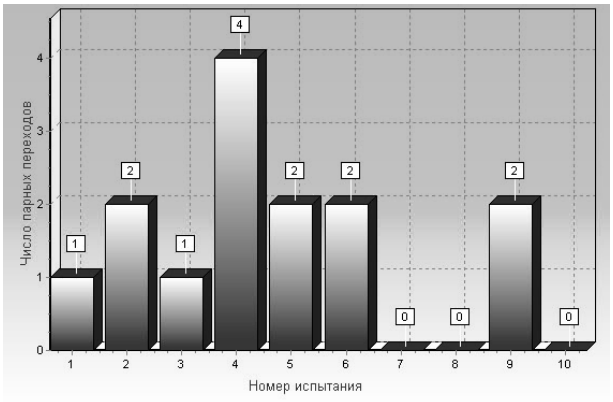

Рисунок 5 - Число изменений дерева в графе из 10 вершин

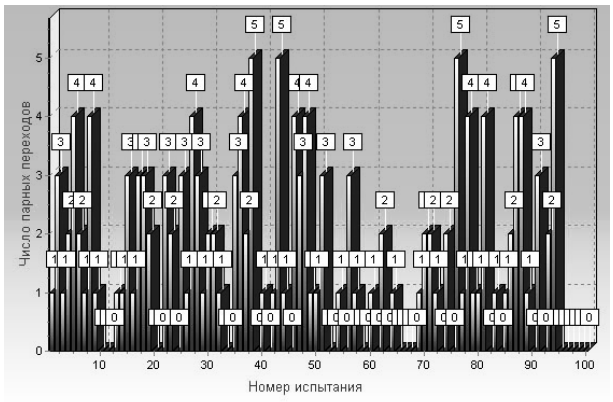

Рисунок 6 - Число изменений дерева в графе из 100 вершин

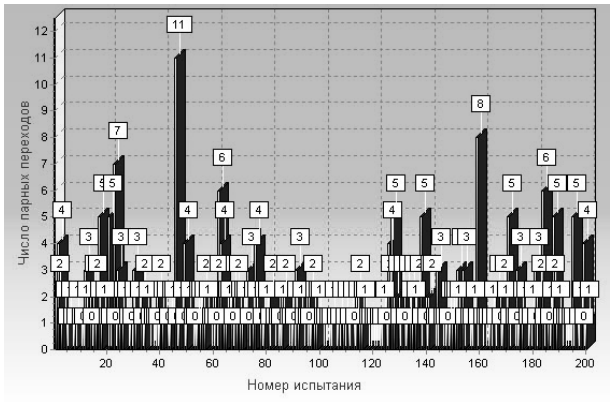

Рисунок 7 - Число изменений дерева в графе из 200 вершин

Были проведены исследования графов, состоящих из 10, 100 и 200 вершин. Исследование

разработанного алгоритма адаптивной ускоренной маршрутизации на базе протокола IGRP показало, что максимальное значение числа изменений существенно ниже размерности для каждого из рассмотренных графов, а значение оценки математического ожидания не превышает единицы. Более того, обнаружена тенденция уменьшения значения оценки математического ожидания числа изменений дерева оптимальных маршрутов с увеличением количества вершин графа.

При динамическом добавлении элементов корпоративной сети разработанный алгоритм адаптивной ускоренной маршрутизации на базе протокола IGRP в отличие от алгоритма Беллмана - Форда позволяет производить перестроение таблиц маршрутизации не полностью, а только той ее части, в которой произошли изменения. При этом трудоемкость изменения дерева кратчайших путей является линейной функцией от числа вершин и определяется выражением $O(N)$. То есть при добавлении элементов корпоративной сети необходимо один раз просмотреть все вершины графа сети и на основе предварительно собранной информации о парных переходах выполнить изменения для вычисления новых оптимальных маршрутов. Если рассмотреть полученные результаты при моделировании предложенного алгоритма адаптивной ускоренной маршрутизации и сравнить их с алгоритмом Беллмана - Форда, то видно:

1) для графа из 10 вершин максимальное число парных переходов составляет четыре изменения (= 4), при этом для алгоритма Беллмана - Форда при каждом добавлении элементов трудоемкость построения таблицы маршрутизации составляет $O(N^3) = 10^3 = 1000$ элементарных операций;

2) для графа из 100 вершин максимальное число парных переходов составляет пять изменений (= 5), при этом для алгоритма Беллмана - Форда при каждом добавлении элементов трудоемкость построения таблицы маршрутизации составляет $O(N^3) = 100^3 = 1000000$ элементарных операций;

3) для графа из 200 вершин максимальное число парных переходов составляет одиннадцать изменений (= 11), при этом для алгоритма Беллмана - Форда при каждом добавлении элементов трудоемкость построения таблицы маршрутизации составляет $O(N^3) = 200^3 = 8000000$ элементарных операций.

На основе этого можно сделать вывод, что предложенный алгоритм адаптивной ускоренной маршрутизации на базе протокола IGRP является эффективным при поиске оптимальных маршрутов в условиях динамических изменений в структуре корпоративной сети и нагрузках на линиях связи за счет использования дополнительной информации о возможных парных переходах.

Заключение. Разработанный алгоритм адаптивной ускоренной маршрутизации на базе протокола IGRP позволяет повысить качество функционирования корпоративных сетей за счет уменьшения трудоемкости построения таблиц маршрутизации до величины порядка $O(N)$ при динамическом добавлении элементов корпоративной сети.

Библиографический список

1. *Уваров Д.В., Перепелкин А.И.* Построение дерева кратчайших путей на основе данных о парных переходах // Системы управления и информационные технологии. 2004. № 4 (16). Москва–Воронеж. С. 93-96.

УДК 681.518

Л.А. Демидова, С.Б. Титов

ИССЛЕДОВАНИЕ ВЛИЯНИЯ ОСНОВНЫХ ПАРАМЕТРОВ АЛГОРИТМА ФУНКЦИОНИРОВАНИЯ ИСКУССТВЕННОЙ ИММУННОЙ СЕТИ НА КАЧЕСТВО КЛАСТЕРИЗАЦИИ ОБЪЕКТОВ

Проведены экспериментальные исследования влияния основных параметров алгоритма функционирования искусственной иммунной сети на качество решения задач кластеризации. Даны рекомендации по выбору значения коэффициента супрессии на основании предполагаемого взаимного расположения кластеров.

Ключевые слова: *кластеризация, искусственная иммунная сеть, коэффициент супрессии, антитело, антиген, алгоритм aiNet.*

Введение. Способность искусственных иммунных систем (ИИС) к самоорганизации определяет возможность их применения в различных задачах обработки информации, например, в задачах кластеризации. Под «самоорганизацией» понимается процесс, в результате которого на основе внутренних свойств функций и структур системы самостоятельно, без внешних управляющих воздействий, создается, воспроизводится или совершенствуется ее организация. В алгоритмах кластеризации с применением ИИС отсутствует явно выраженная целевая функция, а используются лишь критерии распознавания кластеризуемых объектов множеством генерируемых антител (детекторов).

В настоящее время наибольшую известность получили две модели анализа данных, основанные на иммунной метафоре: иммунная сеть AINE (Artificial Immune NEtwork) и иммунная сеть aiNet (artificial immune Net) [1].

Обе иммунные сети используют исходное множество объектов в качестве обучающей выборки, в которой каждый объект воспринимается как антиген, с целью создания множества антител, представляющих эти антигены.

Алгоритм aiNet является одним из самых известных в литературе алгоритмов кластеризации, основанных на использовании механизма ИИС, а его эффективность подтверждена при решении широкого спектра сложных прикладных задач. В иммунной сети aiNet управление динамикой и метадинамикой сети осуществляется посредством клонального отбора. При этом первоначально созданная случайным образом популяция антител Ab видоизменяется с помощью операторов клонального отбора, гипермутации и апоптоза (удаления нестимулируемых антител). Недостатком алгоритма aiNet является большее количество определяемых пользователем параметров управления, чем в алгоритме AINE, а также необходимость использования стандартных инструментов кластерного анализа, таких как алгоритмы построения минимального остовного дерева. Достоинством же алгоритма aiNet является возможность сжатия данных, которое в некоторых случаях достигает 90 % [2].

Цель работы. Целью настоящей работы является исследование влияния основных параметров алгоритма функционирования искусственной иммунной сети aiNet на качество решения задачи кластеризации и разработка рекомендаций по выбору значения коэффициента супрессии на основании предполагаемого взаимного расположения кластеров произвольной формы.

Теоретические исследования. Применительно к задачам распознавания и классификации в качестве популяции антигенов Ag иммунной сети выступает множество объектов $X = \{x_1, x_2, ..., x_n\}$, представляющих набор данных (векторов), которые нужно распознать. При этом каждый вектор x_i $(i = \overline{1,n})$ состоит из q элементов, соответствующих некоторым характеристикам оценивания.

Для количественной оценки взаимодействия антигенов Ag и антител Ab между собой, а также антител Ab между собой в теории ИИС используется понятие «аффинитет».

Под аффинитетом понимается мера взаимодействия (сила связи) комплементарных (взаимно соответствующих) участков антигена Ag и антитела Ab или двух антител Ab, которая формально может быть представлена в виде одной из метрик (например, евклидова расстояния), указывающей на степень подобия или различия между соответствующими позициями векторов, характеризующих антигены и антитела [1].

Аффинитет определяется как число, принадлежащее отрезку $[0,1]$, и вычисляется как евклидово расстояние между двумя векторами, характеризующими два объекта, например антиген Ag и антитело Ab. Низкое значение аффинитета указывает на сильную близость двух объектов.

В ИИС различают два вида аффинности: аффинность связи «антиген – антитело» $Ag - Ab$, характеризующую степень расхождения антигена Ag и антитела Ab, и аффинность связи «антитело – антитело» $Ab - Ab$, характеризующую степень подобия двух антител Ab.

Алгоритм функционирования ИИС можно представить следующим образом:

$$immNET = \left(X, C, M, N_C, D, S, \eta, \xi, \sigma_d, \sigma_s\right), \quad (1)$$

где X – множество объектов x_i $(i = \overline{1,n})$ размерности q, определяющих популяцию антигенов; C – матрица, содержащая все антитела сети ($C \in R^{n \times q}$), определяющие популяцию антител; M – матрица, состоящая из N клеток памяти ($M \subseteq C$); N_C – общее количество клонов, создаваемых стимулируемыми антителами в каждом поколении (при активации сети); D – матрица значений аффинностей d_{ij} связей

типа «антиген – антитело» $Ag-Ab$; S – матрица значений аффинностей S_{ij} связей типа «антитело – антитело» $Ab-Ab$; η – количество лучших антител, выбираемых из матрицы C для клонирования и мутации; ξ – процент улучшенных антител, выбираемых из популяции клонов для последующей обработки (процент переизбрания); σ_d – пороговый коэффициент подавления антитела в зависимости от значений аффинностей d_{ij} его связей типа «антиген – антитело» $Ag-Ab$; σ_s – пороговый коэффициент супрессии («сжатия») сети [1].

В целом алгоритм функционирования ИИС aiNet представляет собой итерационную процедуру, которая может быть получена при реализации следующих трех основных этапов (рисунок 1) [3].

1. Антигены предъявляются антителам, которые подвергаются гипермутации для лучшего соответствия антигенам (при этом происходит реализация взаимодействия антитела и антигена).

2. Антитела, которые сильнее стимулируются, то есть лучше соответствуют антигенам, отбираются для дальнейшего клонирования и роста иммунной сети.

3. Взаимодействие между антителами определяется количественно, и, если одно антитело распознает другое, то одно из них удаляется из популяции антител (при этом происходит реализация взаимодействия одного антитела с другим).

ИИС реализует эволюционные стратегии, основанные на наследственной изменчивости и отборе в пределах популяции антител и используемые для управления динамикой и пластичностью сети [4].

Кластеры в сети будут служить в качестве внутренних образов, ответственных за отображение существующих кластеров в множестве объектов в кластеры сети. Как уже отмечалось ранее, количество антител в сети обычно существенно меньше, чем количество объектов кластеризации (антигенов), на основе которых осуществляется «сжатие» данных, и размер иммунной сети определяется автоматически с учетом коэффициента супрессии («сжатия») σ_s.

Алгоритм функционирования иммунной сети aiNet стремится сформировать такое множество антител, которое распознает и представляет структурную организацию исходных данных – множества объектов кластеризации. При этом, чем более специфичны антигены (объекты кластеризации), тем менее «бережлива» сеть по отношению к количеству антител: реализуется низкий уровень «сжатия»; чем более общие антигены (объекты кластеризации), тем более «бережлива» сеть по отношению к количеству антител: реализуется высокий уровень «сжатия» [1].

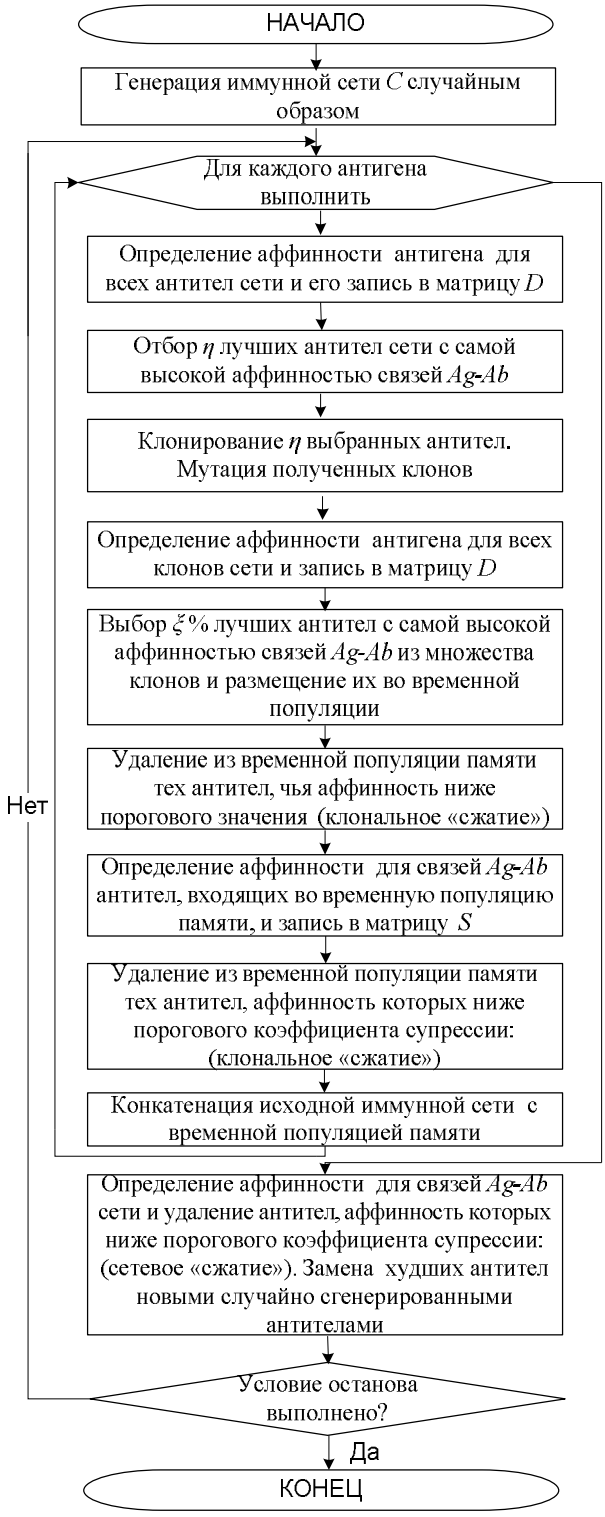

Рисунок 1 – Алгоритм функционирования иммунной сети

Пороговый коэффициент супрессии («сжатия») σ_s контролирует уровень специфики анти-

тел, кластерную точность и пластичность сети. Увеличение значения данного коэффициента приводит к формированию более универсальных антител, в то время как уменьшение коэффициента приводит к появлению узкоспециализированных антител. Этот параметр крайне важен, поскольку слишком высокое значение порогового коэффициента супрессии может привести к ошибочному объединению соседних кластеров. В связи с этим при выборе коэффициента супрессии σ_s обычно сначала устанавливается его некоторое начальное значение, которое в дальнейшем регулируется для повышения производительности сети (например, σ_s выбираются первоначально так, чтобы выполнялось условие: $\sigma_s \le 10^{-3}$).

Аффинность антитела с по отношению к антигенам может быть улучшена посредством прямой мутации с использованием следующей формулы:

$$C = C - \alpha \cdot (C - X), \qquad (2)$$

где C – матрица антител иммунной сети, X – матрица антигенов и α – скорость обучения (скорость мутации).

Величина скорости мутации α устанавливается согласно аффинности связи $Ag - Ab$: чем выше аффинность, тем меньше величина скорости мутации α. Уравнение вида (2) реализует целенаправленный поиск лучших антител, обеспечивая локальную оптимизацию антител иммунной сети посредством «жадного» поиска и улучшая способности антител к распознанию антигенов в ходе смены поколений антител.

Качество распознавания иммунной сетью множества антигенов характеризуется значением среднего аффинитета связей типа «антиген – антитело» $Ag - Ab$ для всех антигенов, распознанных как «свои» антителами соответствующих кластеров: чем меньше данное значение, тем лучше антитела иммунной сети распознают антигены:

$$averageD = \left(\sum_{i=1}^{n_1} \min_{j=1,k} d_{ji}\right) / n_1, \qquad (3)$$

где n_1 – количество антигенов Ag_i; k – количеств антител Ab_j; d_{ji} – аффинность связи типа «антиген – антитело» $Ag - Ab$, определяемая через евклидово расстояние между j-м антителом Ab_j и i-м антигеном Ag_i ($j = \overline{1,k}$, $i = \overline{1,n_1}$).

Условием окончания работы алгоритма функционирования ИИС является достижение предопределенных количества смен поколений или значения среднего аффинитета вида (3).

После завершения работы алгоритма функционирования ИИС образованная популяция антител используется для построения минимального остовного дерева (minimum spanning tree – MST), разбиение которого на кластеры посредством удаления максимальных связей (связей с наибольшими весами) завершает процесс кластеризации [1].

Минимальное остовное дерево является мощным механизмом поиска, реализующим локально адаптивную стратегию соединения антител иммунной сети в кластеры, и может быть использовано для описания количества, структуры и топологии кластеров иммунной сети.

Экспериментальные исследования. Экспериментальные исследования влияния различных параметров алгоритма функционирования ИИС на качество решения задач кластеризации выполнялись с использованием одного из наборов данных для тестирования алгоритмов кластеризации Марбургского университета (*Philipps-University Marburg*), *http://www.uni-marburg.de / fb12/daten-bionik/data*: *Lsun* (объем выборки – 400 объектов, две переменные и три кластера).

Как видно из рисунка 2, на котором представлено исходное множество объектов кластеризации, определяемое набором данных *Lsun*, три кластера расположены довольно близко друг к другу, что существенно затрудняет решение задачи кластеризации с применением традиционных подходов к кластеризации данных.

В таблице 1 приведены параметры алгоритма функционирования ИИС, при которых выполнялись экспериментальные исследования.

Таблица 1 – Параметры работы алгоритма функционирования искусственной иммунной сети

Параметр	Значение
Размер множества антигенов X	400
Пороговый коэффициент супрессии σ_s	0,01 0,03 0,05 0,07 0,09
Пороговый коэффициент подавления антител (порог аффинности) σ_d	1,0
Количество лучших антител η	5
Процент переизбрания ξ	0,20
Скорость гипермутации α	4,00
Количество поколений	100

На рисунках 3 – 5 приведены результаты работы алгоритма функционирования ИИС с параметрами, представленными в таблице 1, при

значении порогового коэффициента супрессии σ_s=0,05. Данное значение порогового коэффициента супрессии σ_s выбрано для иллюстрации работы алгоритма на основании результатов моделирования (таблица 2): бóльшее значение σ_s приводит к существенному снижению процента правильно классифицированных объектов, меньшее значение σ_s приводит к недостаточному сжатию сети (формированию избыточного количества антител без повышения качества распознавания).

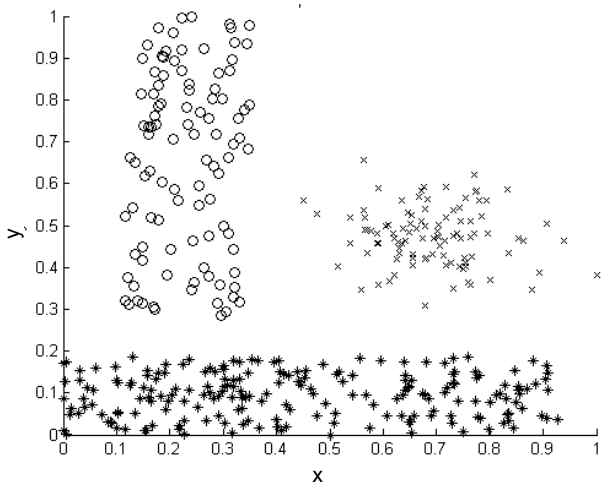

Рисунок 2 – Исходное множество объектов кластеризации

На рисунке 3 показано минимальное остовное дерево, полученное для исходного множества объектов кластеризации (рисунок 2) в ходе реализации алгоритма функционирования ИИС.

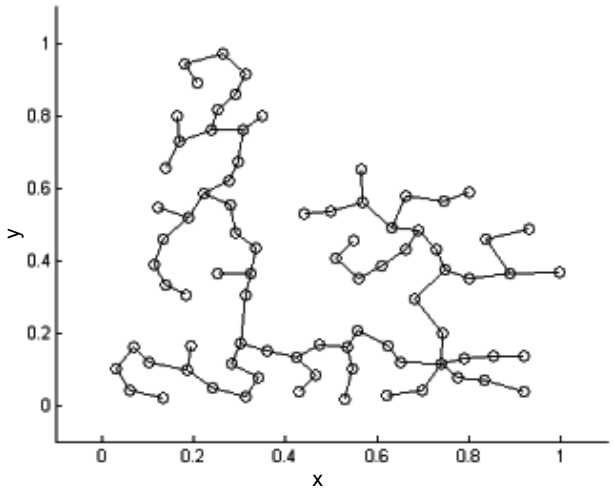

Рисунок 3 – Минимальное остовное дерево

Дендрограмма, представленная на рисунке 4, позволяет осуществить выявление конечной топологии иммунной сети (в частности, определить количество кластеров).

На рисунке 5 приведена сама финальная структура иммунной сети, реализующей разбиение исходного множества объектов кластеризации (рисунок 2) на 3 кластера посредством распознавания антигенов (объектов кластеризации) в качестве «своих» антителами сети.

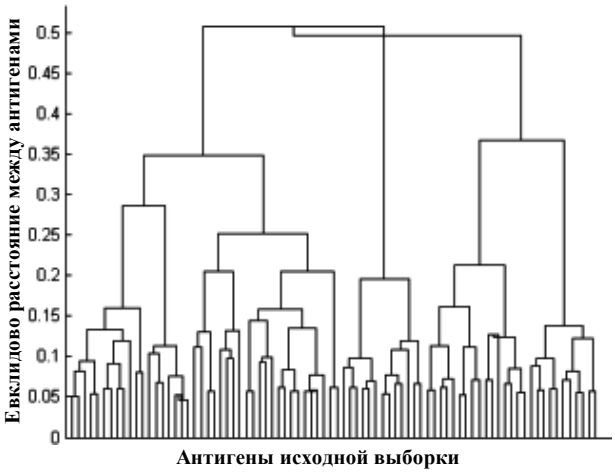

Рисунок 4 – Дендрограмма иммунной сети

Пороговый коэффициент подавления антител σ_d отвечает за устранение из популяции (и соответственно иммунной сети) антител с низкой аффинностью связей типа «антиген – антитело».

Экспериментальные исследования показали, что пороговый коэффициент подавления антител σ_d определяет количество непродуктивных антител (антител, плохо распознающих антигены) и исключаемых из популяции.

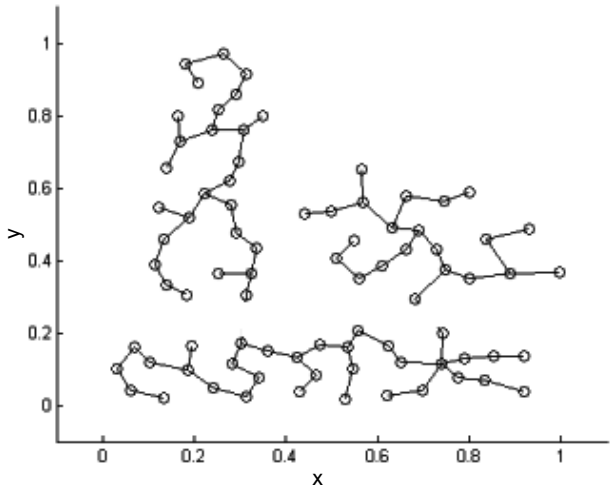

Рисунок 5 – Финальная структура иммунной сети

Полученные результаты свидетельствуют о том, что во всех тестовых прогонах влияние порогового коэффициента подавления антител σ_d привело к сокращению антител иммунной сети только в первом поколении, в то время как в следующих поколениях удаления антител вследствие низкой аффинности связей $Ag - Ab$ не происходило.

Данный факт может объясняться тем, что сформированная случайным образом начальная популяция антител в результате процедуры клонального отбора в первом же поколении в значительной степени очищается от непродуктивных антител, а оставшиеся в популяции антитела достаточно хорошо приспособлены к распознаванию имеющейся выборки антигенов.

Таким образом, можно сделать вывод о том, что инициируемый процесс клонального сжатия, определяемый пороговым коэффициентом подавления антител σ_d, достаточно быстро (за одно поколение) приводит начальную популяцию антител иммунной сети к репрезентативному представлению антигенов.

В то же время, как было отмечено ранее, наиболее важным параметром, в значительной степени определяющим результирующее количество антител иммунной сети, является пороговый коэффициент супрессии σ_s.

В таблице 2 приведены результаты кластеризации исходного набора данных (рисунок 2) с применением описанного алгоритма функционирования ИИС при различных значениях коэффициента супрессии σ_s.

Таблица 2 – Результаты кластеризации с применением алгоритма функционирования искусственной иммунной сети

Значение порогового коэффициента супрессии	Размер сети (количество антител)	Процент правильно классифицированных объектов
0,01	330	100%
0,03	161	100%
0,05	82	100%
0,07	51	75%
0,09	32	43%

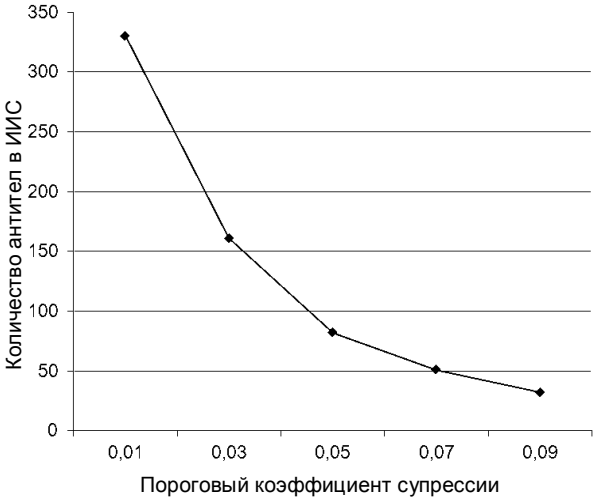

Рисунок 6 – Зависимость размера искусственной иммунной сети от коэффициента супрессии

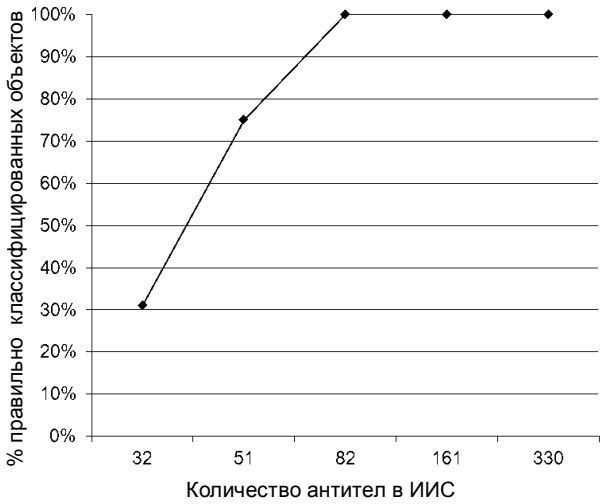

Рисунок 7 – Зависимость качества кластеризации от размера искусственной иммунной сети

Зависимости размера сети от значения порогового коэффициента супрессии σ_s, а также качества кластеризации от размера сети наглядно представлены графически на рисунках 6 и 7.

Полученные экспериментальные данные позволяют дать следующую рекомендацию по выбору значения порогового коэффициента супрессии σ_s: чем ближе (плотнее) друг к другу расположены кластеры, тем меньшее значение порогового коэффициента супрессии σ_s целесообразно использовать для обеспечения высокого качества кластеризации с применением ИИС.

Зависимость качества классификации от размера ИИС является нелинейной, и, начиная с некоторого момента, увеличение размера ИИС не дает прироста качества классификации. Результаты экспериментов показали, что выбор значения порогового коэффициента супрессии σ_s крайне важен для определения итогового размера сети, количества и формы кластеров, полученных с использованием минимального остовного дерева. Таким образом, управление степенью сжатия путем корректного задания значения порогового коэффициента супрессии σ_s для кластеризуемой совокупности объектов в конечном итоге позволяет ускорить процесс распознавания новых объектов и отнесения их к одному из существующих кластеров.

Если рассматривать задачу управления степенью сжатия ИИС упрощенно, в смысле достижения требуемого значения отношения количества сформированных антител к общему количеству антигенов, то поиск соответствующего значения коэффициента супрессии σ_s может быть реализован с помощью наиболее простого варианта итеративного поиска: $\sigma_s = \sigma_s \cdot V_{\sigma_s}$ для уменьшения значения коэффициента супрессии

и $\sigma_s = \sigma_s / V_{\sigma_s}$ для увеличения значения коэффициента супрессии, где V_{σ_s} – скорость изменения коэффициента супрессии; $0 < V_{\sigma_s} < 1$ (например, $V_{\sigma_s} = 0{,}95$), которая (при необходимости) может корректироваться (уменьшаться или увеличиваться).

Начальное значение коэффициента супрессии полагается равным некоторому числу: $\sigma_s = \sigma_{s_0}$, а итерационные вычисления заканчиваются при достижении (с некоторой точностью ε) заданного размера иммунной сети (количества антител) N_{max} [5].

Заключение. В статье проведено исследование влияния основных параметров функционирования ИИС на качество кластеризации объектов. Даны рекомендации по выбору значения порогового коэффициента супрессии в зависимости от предполагаемого взаимного расположения кластеров относительно друг друга. Показано, что зависимость качества классификации от размера ИИС является нелинейной, и, начиная с некоторого момента, увеличение размера сети не дает прироста качества классификации.

Задание коэффициента супрессии с учетом взаимного расположения кластеров произвольной формы относительно друг друга позволяет в итоге уменьшить вычислительную сложность процесса распознавания нового объекта сформированной ИИС, характеризующегося линейной сложностью $O(n)$, где n – количество антител иммунной сети. В частности, в ходе экспериментальных исследований влияния коэффициента супрессии на качество решения задачи кластеризации (таблица 2) с использованием набора данных для тестирования алгоритмов кластеризации (рисунок 2) было показано, что использование значения коэффициента супрессии σ_s=0,05 вместо σ_s=0,01 позволяет уменьшить вычислительную сложность распознавания нового объекта иммунной сетью более чем в 4 раза без ухудшения качества кластеризации.

Использование алгоритма функционирования ИИС позволяет получать адекватные результаты кластеризации в случае присутствия в множестве объектов кластеров произвольной формы в отличие от известных алгоритмов кластеризации, в частности, четкого алгоритма k-средних и его модификаций.

Полученные результаты могут быть использованы при дальнейшем исследовании потенциала применения ИИС при решении задач кластеризации объектов различной природы, в частности, при решении задач, связанных с кластеризацией объектов жилой недвижимости [5], а также задач медицинской диагностики и сегментации изображений.

Библиографический список

1. *de Castro, L.N., Von Zuben, F.J.* aiNet: An artificial Immune Network for Data Analysis // Data Mining: A Heuristic Approach / Eds. H.A. Abbass, R.A. Saker, C.S. Newton, Idea Group Publ., USA, Chapter XII. – 2001. – P. 231–259.

2. *Wierzchoc S.T.* Artificial Immune Systems. Theory and Appl. Akademicka Oficyna Wydawnicza EXIT. – (Polish) Warszawa, 2001. – 282 p.

3. *Литвиненко В.И.* Кластерный анализ данных на основе модифицированной иммунной сети // Управляющие системы и машины. Международный научный журнал. – № 1(219). – Киев: Академпериодика, 2009. – С. 54-62.

4. *Емельянов В.В., Курейчик В.В., Курейчик В.М.* Теория и практика эволюционного моделирования. – М.: ФИЗМАТЛИТ, 2003. – 432 с.

5. *Демидова Л.А., Титов С.Б.* Комплексное применение инструментария искусственных иммунных систем, методов нечеткой кластеризации и теории мультимножеств при решении задачи классификации объектов городской жилой недвижимости // Информационные технологии моделирования и управления. Научно-технический журнал. – Воронеж: Научная книга, 2010. – № 6(65). – С. 718-725.

УДК 004.383.3

В.Н. Ручкин, В.А. Романчук

АЛГОРИТМЫ АНАЛИЗА ВЫЧИСЛИТЕЛЬНЫХ СТРУКТУР НА БАЗЕ НЕЙРОПРОЦЕССОРОВ

Разрабатываются алгоритмы описания интеллектуальных вычислительных структур на базе отечественных нейропроцессоров семейства NM 640X в результате анализа автоматных моделей связей элементов

вычислительных структур с целью определения вида структуры нейропроцессорной системы, и предлагаются программные средства комплекса «НейроКС» реализации нового класса интеллектуальных вычислительных систем.

Ключевые слова: *нейропроцессор, нейропроцессорная система, интеллектуальная вычислительная структура, алгоритмы, автоматные модели связей, программный комплекс «НейроКС».*

Введение. В настоящее время для микропроцессоров наступает так называемый ”технологический предел”, означающий что они достигли максимального уровня повышения быстродействия. Все разработки в данное время направлены на повышение числа процессоров на кристалле. Одним из выходов из данной ситуации является новая элементная база, например использование нейрокомпьютеров. В области нейрокомпьютеров в настоящее время ведутся разработки с использованием новых технологий, перспективными можно назвать технологии создания оптических, молекулярных и нанонейрокомпьютеров [1, 2].

Вычислительные системы на базе нейропроцессоров – нейропроцессорные системы (НПС) отличаются высокой эффективностью при их использовании вследствие таких причин:

- алгоритмы нейроинформатики высокопараллельны за счет использования нейросетевого базиса;
- НПС можно обучить устойчивости к помехам и в дальнейшем перейти к самообучению.

Но для дальнейшего развития области нейропроцессорных технологий имеется ряд проблем, основными из которых являются [3]:

1) невысокая частота нейрочипов (30-150 МГц) не позволяет получить конкурирующую с обычными микропроцессорами производительность нейропроцессорных устройств;

2) недостаточное программное обеспечение для нейропроцессоров по сравнению с общераспространенными микропроцессорами;

3) коммерческая недоступность (секретность) информационных материалов в данной области;

4) слишком большая цена перехода от существующих процессоров к нейропроцессорам (изменение не только аппаратных, но и программных средств).

Одним из способов решения первой проблемы является организаци***я интеллектуальных вычислительных структур*** [4]. В настоящее время в области нейропроцессорных технологий ведутся исследования в части многопроцессорности, уже разработаны модули, включающие несколько процессоров с различными связями. Однако имеются проблемы, мешающие созданию эффективных мультимикропроцессорных структур на базе нейропроцессоров, одной из которых является проблема описания связей элементов НПС и описание различных структур НПС на основании описания связей ее элементов.

Цель работы. Целью работы является разработка алгоритмов описания интеллектуальных вычислительных структур на базе отечественных нейропроцессоров.

Постановка задачи. В соответствии с целью работы были определены две задачи: разработка алгоритмов определения связей элементов интеллектуальных вычислительных структур на базе нейропроцессоров и разработка алгоритмов определения вида структуры НПС на основе описания связей ее элементов.

Для дальнейших исследований в качестве примера было выбрано семейство нейропроцессоров NM640X по следующим причинам.

- Процессоры семейства NM640X обладают функциональными возможностями, наиболее полно отражающими принципы функционирования всего класса нейропроцессоров.
- Информация о процессе функционирования размещена в открытом доступе и содержит необходимый для дальнейшего исследования теоретический материал.

Результаты исследования могут быть адаптированы и для других нейропроцессоров, соответствующих общим принципам функционирования.

Теоретические исследования. ***Разработка алгоритмов определения связей элементов вычислительных структур на базе нейропроцессоров***

Нейропроцессорная система состоит из множества элементов – процессорных модулей (ПМ). Под процессорным модулем будем понимать нейромикропроцессор и ряд вспомогательных интегральных схем.

Результаты определения связей элементов НПС будем представлять в виде матрицы связи ПМ $M = [M_{ij}]$. Ее размерность $q \times q$, где q - число ПМ. Элементами матрицы M являются: ‘0’ - нет связи между ПМ; ‘1’ - есть связь между ПМ (в подпрограмме с большим порядковым

номером необходимы данные подпрограммы с меньшим порядковым номером); ‘X’ – запрещенные ячейки (между подпрограммами не может быть связей).

Для того чтобы определить элементы матрицы M, введем вспомогательную матрицу $M' = [M'_{ij}]$ размерности $q \times E$, в которой число столбцов E - это количество элементов процессора. Для нейропроцессоров семейства NM640X $E = 44$ (дополнительно заносятся области памяти, занятые переменными). Элементами матрицы M' являются: ‘0’ - элемент не использовался в данной подпрограмме; ‘1’ – элемент был присвоен в данной подпрограмме; ‘2’ - элемент был использован в данной подпрограмме; ‘3’ - элемент был использован, а затем присвоен в данной подпрограмме.

Случай, когда элемент сначала присваивается в подпрограмме, а затем используется, не рассматривается, т.к. важен только случай присваивания элемента в текущей подпрограмме, т.е. случай с обозначением ‘1’.

Данная матрица описывает состояние каждого элемента процессора в каждой подпрограмме и позволяет определить элемент матрицы M_{ij} следующим образом:

$$((M'_{ik} = 1) \wedge (M'_{jk} = 2) \vee (M'_{ik} = 3)) \vee$$
$$((M'_{nk} = 1) \vee (M'_{nk} = 3)) \rightarrow M_{ij} = 1; n = \overline{i+1, j};$$
$$((M'_{ik} = 3) \wedge (M'_{jk} = 2) \vee (M'_{ik} = 3)) \wedge x.$$
$$((M'_{nk} = 1) \vee (M'_{nk} = 3)) \rightarrow M_{ij} = 1; n = \overline{i+1, j}.$$

Тогда целью алгоритма является заполнение элементов матрицы связей ПМ для дальнейшего определения вида структуры. На входе имеем кортеж подпрограмм, например $PR = < RO_1, RO_2, RO_5, RO_1, ... >$. На выходе - матрица связи ПМ M и вспомогательная матрица связи M' с описанием связей между элементами НПС.

Для того чтобы определить состояние элемента процессора R в подпрограмме P, был использован математический аппарат конечных автоматов. Пусть $K = (S, Z, W, \delta, \lambda, S_1)$ - конечный автомат описания (рисунок 1), где:

S - множество состояний автомата K;

Z - множество входных сигналов автомата K;

W - множество выходных сигналов автомата K;

δ - функция переходов автомата K;

λ - функция выходов автомата K;

S_1 - начальное состояние автомата K.

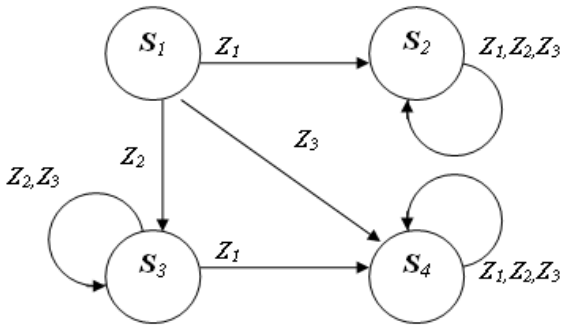

Рисунок 1 – Конечный автомат K определения состояния элемента ПМ R

Множество состояний S автомата K: S_1 - состояние M'_{RP}=0; S_2 - состояние M'_{RP}=1; S_3 - состояние M'_{RP}=2; S_4 - состояние M'_{RP}=3.

Входной алфавит Z автомата K: Z_1 - элементу R было присвоено значение; Z_2 - значение элемента R было использовано; Z_3 - значение элемента R было сначала использовано, потом присвоено.

Пусть q - число ПМ, E - число элементов процессора. Рассмотрим *алгоритм заполнения матрицы связей ПМ*. Целью алгоритма является получение значений $M_{ij}; i = \overline{1, q}; j = \overline{1, q}$.

Шаг 1. Если $p \geq q$, то переход на шаг 4.

Шаг 2. Если $i \geq E$, то $p = p + 1$, переход на шаг 1.

Шаг 3. Выборка микрокоманды MK_j. Заполнение элементов матрицы M' (алгоритм рассматривается далее), $i = i + 1$, переход на шаг 2.

Шаг 4. Если $p \geq q$, то переход на шаг 7.

Шаг 5. Если $i \geq q$, то $p = p + 1$ и переход на шаг 4.

Шаг 6. Заполнение элементов матрицы M (алгоритм рассматривается далее), $i = i + 1$ и переход на шаг 5.

Шаг 7. Конец алгоритма.

Рассмотрим *алгоритм, необходимый для того, чтобы определить значения элементов матрицы* M'. Целью алгоритма является получение значений $M'_{in}; i = \overline{1, q}; n = \overline{1, E}$.

Для реализации алгоритмов используется понятие лексемы Lex – последовательности допустимых символов языка нейроассемблера, имеющей смысл для транслятора. Алгоритмы распознавания лексем Lex приведены далее.

Далее показан *алгоритм распознавания лексем для языка нейроассемблера*, используемого для семейства нейропроцессоров NM640X. На

входе алгоритма некоторая микрокоманда MK_i из кортежа $PR =< MK_1, MK_2, ..., MK_m, ..., MK_M >$. На выходе элемент матрицы M' для микрокоманды MK_i.

Шаг 1. Разделение микрокоманды MK_i на левую и правую часть.

Шаг 2. Если микрокоманда скалярная, то переход на шаг 12.

Шаг 3. Если в отладочном коде MK_i нет лексемы wtw, то на шаг 5.

Шаг 4. Заполнение элементов матрицы M', связанных с операцией wtw.

Шаг 5. Если в отладочном коде MK_i нет лексемы ftw, то переход на шаг 7.

Шаг 6. Заполнение элементов матрицы M', связанных с операцией ftw.

Шаг 7. Если в отладочном коде MK_i нет лексемы $vregs$, то на шаг 9.

Шаг 8. Заполнение элементов матрицы M', связанных с операцией $vregs$.

Шаг 9. Если в отладочном коде MK_i нет лексем выборки памяти, то шаг 11.

Шаг 10. Заполнение элементов матрицы M', связанных с операцией wtw.

Шаг 11. Заполнение элементов матрицы M', связанных с микрокомандой MK_i. Переход на шаг 23.

Шаг 12. Если в отладочном коде MK_i нет лексем перехода, то на шаг 14.

Шаг 13. Заполнение элементов M', связанных с обработкой перехода.

Шаг 14. Если в отладочном коде MK_i нет лексемы $pswr$, то на шаг 16.

Шаг 15. Заполнение элементов M', связанных с обработкой блока $pswr$.

Шаг 16. Если в отладочном коде MK_i нет лексем выборки памяти, то на шаг 18.

Шаг 17. Заполнение элементов M', связанных с обработкой памяти.

Шаг 18. Если в отладочном коде MK_i нет лексем изменения флагов N,Z,V,C, то переход на шаг 20.

Шаг 19. Заполнение элементов M', связанных с обработкой флагов.

Шаг 20. Если в отладочном коде MK_i нет лексем изменения регистров $ar0..ar7, gr0..gr7$, то переход на шаг 22.

Шаг 21. Заполнение элементов M', связанных с обработкой регистров.

Шаг 22. Заполнение элементов матрицы M', связанных с микрокомандой MK_i.

Шаг 23. Конец алгоритма.

Обработка микрокоманды реализуется с помощью конечного автомата.

Пусть: $Z = \{1,2,3\}$ - входящий сигнал конечного автомата,

$C = \{1,2,3\}$ - текущее состояние элемента процессора.

Одним из шагов разрабатываемого алгоритма является шаг заполнения матрицы M. Для его реализации необходимо использовать значения элементов матрицы связей ПМ M' и определить отображение:

$$\varphi : M_{ij} \to M'_{nj}; i = \overline{1,q}; j = \overline{1,q}; n = \overline{1,E}.$$

Пусть: k - элемент процессора (для семейства процессоров NM640X k =44);

$M'_{ij} = \{1,2,3\}$ - текущее состояние элемента матрицы M';

$M_{ij} = \{0,1\}$ - текущее состояние элемента матрицы M.

Тогда можно описать *алгоритм определения значения элементов матрицы M по значениям элементов матрицы M'*, который является реализацией конечного автомата K, показанного на рисунке 1. На входе алгоритма - множество элементов матрицы M'. На выходе множество элементов матрицы M.

Шаг 1. Начальная инициализация элемента $M_{ij} = 0$.

Шаг 2. Если $k \geq E$, то переход на шаг 12.

Шаг 3. Если $j \leq i$, то $M_{ij} = 'X'$, переход на шаг 12.

Шаг 4. Если $(M'_{ik} = 1) \wedge (M'_{jk} = 2) \vee (M'_{ik} = 3)$, то переход на шаг 8.

Шаг 5. Если $(n > i+1) \wedge (n < j)$, то переход на шаг 7.

Шаг 6. Если $(M'_{nk} = 1) \vee (M'_{nk} = 3); n = \overline{i+1, j}$, то $flag = true$, $n = n+1$, переход на шаг 5.

Шаг 7. Если $flag = false$, то $M_{ij} = 1$.

Шаг 8. Если $(M'_{ik} = 3) \wedge (M'_{jk} = 2) \vee (M'_{ik} = 3)$, то $k = k+1$, переход на шаг 3.

Шаг 9. Если $(n > i+1) \wedge (n < j)$, то переход

на шаг 11.

Шаг 10. Если $(M'_{nk} = 1) \vee (M'_{nk} = 3); n = \overline{i+1, j}$, то $flag = true$, $n = n + 1$, переход на шаг 9.

Шаг 11. Если $flag = false$, то $M_{ij} = 1$.

Шаг 12. Конец алгоритма.

Таким образом, имеем матрицу M размерности $q \times q$, где q - число ПМ в НПС. Она содержит элементы, обозначающие зависимости ПМ по всем элементам процессора семейства NM640X. На основе полученной матрицы связей между ПМ можно определить вид нейропроцессорной структуры.

Разработка алгоритмов определения вида структуры НПС на основе описания связей ее элементов

На данном этапе необходимо определить вид многопроцессорной структуры исходя из матрицы связей ПМ. Для этого проверяется соответствие полученной матрицы связей ПМ всем признакам матриц для каждого типа архитектур. Если данная матрица не обладает всеми признаками ни одного из этих типов архитектур, то НПС является структурой произвольного вида. На вход алгоритма поступает матрица связей M, полученная ранее. На выходе имеем флаги вида структуры ($FKonv, FVect, VKV, FVK, FN$) - значения, равные *true,* если НПС имеет структуру конвейерного, векторного, векторно-конвейерного, конвейерно-векторного и произвольного типа соответственно.

Шаг 1. Определение соответствия матрицы всем признакам матрицы для конвейерной архитектуры.

Шаг 2. Если матрица обладает всеми признаками конвейерной архитектуры, то $FKonv = true$, переход на шаг 10.

Шаг 3. Определение соответствия матрицы всем признакам матрицы для векторной архитектуры.

Шаг 4. Если матрица обладает всеми признаками векторной архитектуры, то $FVect = true$, переход на шаг 10.

Шаг 5. Определение соответствия матрицы всем признакам матрицы для конвейерно-векторной архитектуры.

Шаг 6. Если матрица обладает всеми признаками конвейерно-векторной архитектуры, то $FKV = true$, переход на шаг 10.

Шаг 7. Определение соответствия матрицы всем признакам матрицы для векторно-конвейерной архитектуры.

Шаг 8. Если матрица обладает всеми признаками векторно-конвейерной архитектуры, то $FVK = true$, переход на шаг 10.

Шаг 9. Установка флага $FN = true$.

Шаг 10. Конец алгоритма.

Рассмотрим матрицы и алгоритмы для определения каждого вида структур.

1. Конвейерная структура

Отличием матрицы для данной структуры является наличие значений ‘1’ по диагонали над главной диагональю матрицы. В остальных ячейках должны быть значения ‘0’. Пример матрицы конвейерной структуры для пяти ПМ представлен на рисунке 2.

$$M = \begin{bmatrix} X & 1 & 0 & 0 & 0 \\ X & X & 1 & 0 & 0 \\ X & X & X & 1 & 0 \\ X & X & X & X & 1 \\ X & X & X & X & X \end{bmatrix}$$

Рисунок 2 – Матрица связи конвейерной структуры

Рассмотрим *алгоритм определения конвейерной структуры*. На вход алгоритма поступает матрица связей M. На выходе имеем флаг $FKonv = \{true, false\}$ конвейерной структуры.

Шаг 1. Начальная инициализация $FKonv = true$.

Шаг 2. Если $i \geq q$, то переход на шаг 8.

Шаг 3. Если $j \geq q$, то $i = i + 1$, переход на шаг 2.

Шаг 4. Если $i \leq j$, то $j = j + 1$, переход на шаг 3.

Шаг 5. Если $i = j - 1$, то переход на шаг 7.

Шаг 6. Если $M_{ij} \neq 0$, то $FKonv = false$, $j = j + 1$, переход на шаг 3.

Шаг 7. Если $M_{ij} \neq 1$, то $FKonv = false$, $j = j + 1$, переход на шаг 3.

Шаг 8. Конец алгоритма.

2. Векторная структура

Отличием матрицы для данной структуры является то, что все значения ячеек равны ‘0’. Пример матрицы для пяти ПМ представлен на рисунке 3.

$$M = \begin{bmatrix} X & 0 & 0 & 0 & 0 \\ X & X & 0 & 0 & 0 \\ X & X & X & 0 & 0 \\ X & X & X & X & 0 \\ X & X & X & X & X \end{bmatrix}$$

Рисунок 3 – Матрица связи векторной структуры

Рассмотрим *алгоритм определения векторной структуры*. На вход алгоритма поступает матрица связей M. На выходе имеем флаг $FVect = \{true, false\}$ векторной структуры.

Шаг 1. Начальная инициализация $FVect = true$.

Шаг 2. Если $i \geq q$, то переход на шаг 6.

Шаг 3. Если $j \geq q$, то $i = i + 1$, переход на шаг 2.

Шаг 4. Если $i \leq j$, то $j = j + 1$, переход на шаг 3.

Шаг 5. Если $M_{ij} = 1$, то $FVect = false$, $j = j + 1$, переход на шаг 3. В противном случае $j = j + 1$, переход на шаг 3.

Шаг 6. Конец алгоритма.

3. Векторно-конвейерная структура

Отличием матрицы структуры является то, что все значения ячеек равны ‘0’, кроме некоторых (не всех) значений ‘1’ над главной диагональю матрицы. Пример матрицы для пяти ПМ представлен на рисунке 4.

$$M = \begin{bmatrix} X & 1 & 0 & 0 & 0 \\ X & X & 0 & 0 & 0 \\ X & X & X & 1 & 0 \\ X & X & X & X & 0 \\ X & X & X & X & X \end{bmatrix}$$

Рисунок 4 – Матрица связи векторно-конвейерной структуры

Рассмотрим *алгоритм определения векторно-конвейерной структуры*. На вход алгоритма поступает матрица связей M. На выходе имеем флаг $FVK = \{true, false\}$ векторно-конвейерной структуры.

Шаг 1. Начальная инициализация $FVK = true$.

Шаг 2. Если $i \geq q$, то переход на шаг 8.

Шаг 3. Если $j \geq q$, то $i = i + 1$, переход на шаг 2.

Шаг 4. Если $i \leq j$, то $j = j + 1$, переход на шаг 3.

Шаг 5. Если $i \neq j - 1$, то переход на шаг 7.

Шаг 6. Если $M_{ij} = 1$, то $FVK = false$, $j = j + 1$, переход на шаг 3.

Шаг 7. $j = j + 1$, переход на шаг 3.

Шаг 8. Конец алгоритма.

4. Конвейерно-векторная структура

Отличием матрицы для данной структуры является то, что все значения ‘1’ сгруппированы в прямоугольные контуры, которые упорядочены в матрицы по типу лестницы и нет рядов и строк матрицы без хотя бы одного значения ‘1’. Пример матрицы векторной структуры для пяти ПМ представлен на рисунке 5.

$$M = \begin{bmatrix} X & 1 & 1 & 1 & 0 \\ X & X & 0 & 0 & 1 \\ X & X & X & 0 & 1 \\ X & X & X & X & 1 \\ X & X & X & X & X \end{bmatrix}$$

Рисунок 5 – Матрица связи конвейерно-векторной структуры

Рассмотрим *алгоритм определения конвейерно-векторной структуры*. На вход алгоритма поступает матрица связей M. На выходе имеем флаг $FKV = \{true, false\}$ конвейерно-векторной структуры.

Шаг 1. Начальная инициализация $FKV = true$.

Шаг 2. Если $i \geq q$, то переход на шаг 8.

Шаг 3. Если $j \geq q$, то $i = i + 1$, переход на шаг 2.

Шаг 4. Проверка условия: элемент входит в строго прямоугольный контур, состоящий из значений, равных ‘1’.

Шаг 5. Проверка условия: все элементы справа и сверху прямоугольного контура равны ‘0’.

Шаг 6. Проверка условия: все элементы слева и снизу граничного элемента прямоугольного контура имеют значение ‘X’.

Шаг 7. Проверка условия: все прямоугольные контуры упорядочены в виде “лестницы”, “ступени” которой не прерываются по вертикали и горизонтали, $j = j + 1$, переход на шаг 3.

Шаг 8. Конец алгоритма.

Процедура “Проверка условия: элемент входит в прямоугольную область” реализуется методом обхода границ области (рисунок 6):

$$M = \begin{bmatrix} X & 0 & 1 & 1 & 0 \\ X & X & 1 & 1 & 0 \\ X & X & X & 0 & 0 \\ X & X & X & X & 0 \\ X & X & X & X & X \end{bmatrix}$$

Рисунок 6 – Метод обхода границ для матрицы связи

1. Маркер двигается вправо до ‘0’, пройденный путь обозначим как x_1.

2. Маркер двигается вниз до ‘0’, пройден-

ный путь обозначим как y_1.

3. Маркер двигается влево до ‘0’, пройденный путь обозначим как x_2.

4. Маркер двигается вниз до ‘0’, пройденный путь обозначим как y_2.

Если $x_1 = x_2$ и $y_1 = y_2$, то контур прямоугольный.

Практические исследования. Программные средства описания многопроцессорной вычислительной структуры на базе нейропроцессоров были реализованы в программном комплексе «НейроКС» [4,5].

Алгоритм определения связей элементов вычислительной структуры на базе нейропроцессоров с целью определения вида НПС реализован следующим образом: после вставки директив в текстовом редакторе становится возможным режим «Генерация матрицы связи подпрограмм» для визуального представления матрицы M (рисунок 7).

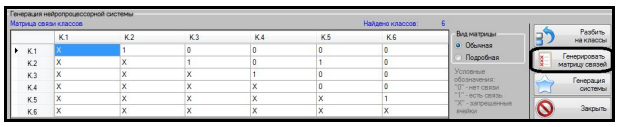

Рисунок 7 – Визуальное представление матрицы M

После генерации матрицы M возможна генерация матрицы M' (рисунок 8).

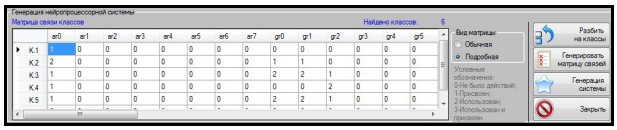

Рисунок 8 – Визуальное представление матрицы M'

Алгоритм определения вида структуры НПС на основе описания связей ее элементов с целью использования оценок эффективности для этого вида структуры реализован следующим образом: после вызова соответствующей процедуры на основании матрицы связей M' происходит запуск подсистемы «Конструктор систем» для визуального представления вычислительной структуры (рисунок 9).

Было произведено экспериментальное исследование разработанного программного комплекса на примере реализации алгоритма шифрования по методу ГОСТ 28147-89. Реализованы НПС конвейерной структуры с различным числом ПМ, что позволило в дальнейшем сравнить время выигрыша относительно однопроцессорного варианта (в данной статье не рассматривается) и получить следующие результаты: 2 ПМ – выигрыш на 50.31 %, 3 ПМ - на 98.07 %, 4 ПМ - на 152.59 %, 5 ПМ - на 195.53 %, 6 ПМ - на 244.63 %, 7 ПМ - на 291.48 %, 8 ПМ - на 393.62 %.

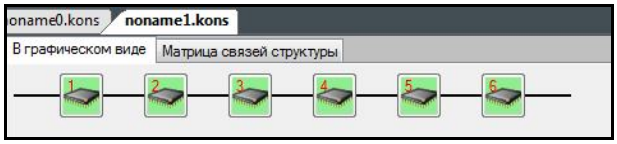

Рисунок 9 – Визуальное представление интеллектуальной вычислительной системы

Результаты исследования внедрены в НТЦ «Модуль», который является производителем нейропроцессоров семейства NM 640X.

Заключение. Таким образом, для проектирования и анализа вычислительных структур получены следующие основные результаты:

- алгоритм определения связей элементов вычислительной структуры на базе нейропроцессоров с целью определения вида НПС;
- алгоритм определения вида структуры НПС на основе описания связей ее элементов;
- программные средства анализа интеллектуальных вычислительных структур на базе нейропроцессоров семейства NM 640X.

Библиографический список

1. *Галушкин А.И.* Нейрокомпьютеры. Кн.3. – М.: ИПРЖР, 2000. - 528 с.

2. *Злобин В.К., Ручкин В.Н.* Нейросети и нейрокомпьютеры: учеб. пособие / В.К., Злобин. В.Н. Ручкин. - СПб.: БХВ-Петербург, 2011. - 256 с.

3. *Головкин Б.А.* Вычислительные системы с большим числом процессоров. М.: Радио и связь, 1995. - 320 с.

4. *Калинкина Т.И., Костров Б.В., Ручкин В.Н.* Телекоммуникационные и вычислительные сети. Архитектура, стандарты и технологии. – СПб.: БХВ-Петербург, 2010. – 288 с.

5. *Романчук В.А., Ручкин В.Н.* Разработка программных средств анализа нейропроцессорных систем // Вестник РГРТУ. 2010. № 2. Вып.32. С.61-67.

УДК: 004.855.5, 004.896

С.И. Бабаев, М.В. Акинин

НЕЙРОСЕТЕВОЙ ПОДХОД К РЕШЕНИЮ ЗАДАЧИ ОБМЕНА КЛЮЧЕВОЙ ИНФОРМАЦИЕЙ С ИСПОЛЬЗОВАНИЕМ ДРЕВОВИДНЫХ МАШИН ЧЕТНОСТИ БЕЗ ОБРАТНЫХ СВЯЗЕЙ

В статье рассмотрен способ обмена ключевой информацией, основанный на использовании искусственной нейронной сети класса «древовидная машина четности без обратных связей».

Ключевые слова: *искусственные нейронные сети, протокол, протоколы обмена ключевой информацией, древовидная машина четности без обратных связей.*

Введение. Цель работы – исследование способа обмена ключевой информацией с использованием искусственных нейронных сетей (ИНС). Обмен ключевой информацией является важным этапом в процессе организации защищенного канала передачи данных между несколькими абонентами. К протоколам, реализующим обмен ключевой информацией, предъявляются следующие требования:

– стойкость протокола к различного рода атакам, целью которых является компрометация (в частном случае – полное раскрытие) ключа;

– простота реализации;

– минимизация объема вычислений.

В основе протоколов обмена ключевой информацией, удовлетворяющих перечисленным требованиям, может использоваться теория ИНС [1].

Теоретическая часть. В данной работе для реализации протокола обмена ключевой информацией используется древовидная машина четности (Tree Parity Machine; TPM) без обратных связей. Данная TPM представляет собой двухслойную полносвязную ИНС без обратных связей, состоящую из входного слоя, содержащего M нейронов, и выходного слоя из одного нейрона.

На вход каждого нейрона входного слоя подается соответствующая строка входной бинарной матрицы X (ноль кодируется как -1):

$$X = \begin{pmatrix} x_{11} & x_{12} & \dots & x_{1N} \\ x_{21} & x_{22} & \dots & x_{2N} \\ \dots & \dots & \dots & \dots \\ x_{M1} & x_{M2} & \dots & x_{MN} \end{pmatrix}. \qquad (1)$$

В качестве функции активации нейронов входного слоя используется пороговая функция:

$$y_m = f(S_m) = \begin{cases} 1, & S_m > 0 \\ -1, & S_m \le 0 \end{cases}, \qquad (2)$$

где $S_m = \sum_{n=1}^{N} w_{mn} x_{mn}; m = \overline{1, M}$; m – индекс нейрона; n – индекс входа нейрона; w_{mn} – вес, соответствующий mn–му элементу входной матрицы X.

Веса w_{mn} выбираются таким образом, чтобы удовлетворять условию:

$$w_{mn} \in [-L, L]; w_{mn} \in Z; L \in N. \qquad (3)$$

Функция активации нейрона выходного слоя имеет вид:

$$f_{out}(y) = \prod_{m=1}^{M} y_m. \qquad (4)$$

Протокол обмена ключевой информацией. Для генерации ключа симметричного алгоритма шифрования с использованием описанной TPM абоненты A и B должны взаимодействовать по следующему алгоритму [2, 3].

Шаг 1 – абонент A выбирает числа N, M такие, чтобы выполнялись требования к разрядности генерируемого ключа $G = MN$.

Шаг 2 – абонент A выбирает не очень большое число L ($L \le 16$).

Шаг 3 – абонент A выбирает натуральное число D.

Шаг 4 – абонент A отправляет числа N, M, L абоненту B.

Шаг 5 – абоненты A и B создают свои TPM, устанавливая веса нейронов входных слоев в случайные значения из диапазона $[-L, L]$,

используя для этого независимые друг от друга генераторы псевдослучайных чисел.

Шаг 6 – абонент А генерирует случайную матрицу X и отправляет ее абоненту В.

Шаг 7 – абоненты А и В прогоняют свои ТРМ над матрицей X.

Шаг 8 – в случае, если выходы ТРМ абонентов А и В совпадают, абоненты А и В корректируют веса своих ТРМ по одному из способов, рассмотренных ниже.

Шаг 9 – если на D предыдущих итерациях выходы ТРМ абонентов А и В совпадали, то синхронизация ТРМ завершается, иначе выполняется переход к Шагу 6.

Шаг 10 – абоненты А и В генерируют ключ симметричного алгоритма шифрования по знакам весов нейронов входных слоев своих ТРМ – бит k $(k = mn)$ ключа устанавливается в 1, если $w_{mn} > 0$, и в 0 в противном случае. Предполагается, что в результате синхронизации данные веса будут совпадать.

Важным параметром приведенного алгоритма является число D. Действительно, выбор слишком малого D ведет к тому, что абоненты А и В получат по завершению алгоритма слабосинхронизированные ТРМ, а следовательно, различные ключи симметричного алгоритма шифрования. Выбор слишком большого D ведет к тому, что шансы злоумышленника, прослушивающего канал и пытающегося синхронизировать свою ТРМ с ТРМ абонентов А и В, на удачную синхронизацию своей ТРМ с ТРМ абонентов А и В существенно повышаются.

Для коррекции весов ТРМ абонентами А и В на шаге 8 могут быть использованы следующие способы:

$$-w_{mn} = w_{mn} + x_{mn} f_{out}(y);$$

$$-w_{mn} = w_{mn} - x_{mn} f_{out}(y);$$

$$-w_{mn} = w_{mn} + x_{mn}.$$

Анализ успешной реализации атак на протокол обмена ключевой информацией

1. *Атака методом грубой силы.* Злоумышленник С проверяет все возможные комбинации значений весов w_{mn} нейронов входного слоя ТРМ в попытке найти начальную комбинацию весов ТРМ абонента А или абонента В. Всего существует $(2L+1)^{MN}$ возможных комбинаций значений весов w_{mn} нейронов входного слоя ТРМ. Вероятность успешной реализации данной атаки $P = \dfrac{1}{(2L+1)^{MN}}$. $P \to 0$ при возрастании значений M и N.

2. *Атака путем синхронизации злоумышленником собственной ТРМ.* Злоумышленник С перехватывает случайную матрицу X и выходы $f_A(X)$ и $f_B(X)$ ТРМ абонентов А и В на каждой итерации процесса синхронизации и корректирует веса своей ТРМ в случае, если абоненты А и В выполняют коррекцию весов своих ТРМ и выход ТРМ абонента С $f_C(X)$ удовлетворяет условию $f_C = f_A$, иначе пропускает коррекцию весов.

Можно оценить вероятность успешной атаки. В простейшем случае $M = 2$. Вероятность того, что выходы m–х нейронов ТРМ абонентов А и В совпадут, равна p. Пусть абоненты А и В выполнили единичный прогон своих ТРМ – в этом случае возможны три ситуации:

1) выходы первых нейронов входного слоя совпали, выходы вторых нейронов входного слоя совпали, следовательно, совпали выходы ТРМ и абоненты А и В корректируют значения весов своих ТРМ. В этом случае происходит полезная коррекция весов [4]. Вероятность такового исхода составляет p^2;

2) выходы первых (вторых) нейронов входного слоя совпали, выходы вторых (первых) нейронов входного слоя не совпали, следовательно, выходы ТРМ не совпали и абоненты А и В не выполняют коррекцию значений весов своих ТРМ. Вероятность такового исхода составляет $2(1-p)p$;

3) выходы первых нейронов входного слоя не совпали, выходы вторых нейронов входного слоя не совпали, следовательно, совпали выходы ТРМ (четность сохранилась) и абоненты А и В корректируют значения весов своих ТРМ. В этом случае происходит бесполезная коррекция весов [4]. Вероятность такового исхода составляет $(1-p)^2$.

В паре А и В вероятность полезной коррекции весов составляет p^2. Абонент С вынужден корректировать веса своей ТРМ только в случае, если $f_A = f_B$, что происходит с вероятностью $p^2 + (1-p)^2$. При этом вероятность полезной коррекции весов в паре абонентов А и С составляет $p^2(p^2 + (1-p)^2)$. Причем $p^2 \geq p^2(p^2 + (1-p)^2); p \in [0,1]$. Равенство достигается, если $p = 1$, что соответствует случаю, когда ТРМ абонентов А и В и злоумышленника С априори синхронизированы.

Таким образом, вероятность синхронизации ТРМ злоумышленника С с ТРМ абонента А (или В) в p^2 раз меньше, чем вероятность синхронизации ТРМ абонентов А и В.

В случае $M > 2$ картина остается той же, но вероятность синхронизации ТРМ в паре А (или В) и С становится меньше, чем в случае $M = 2$.

3. *Атака с использованием генетического алгоритма*. Злоумышленник С создает ТРМ (текущий размер популяции $F_C = 1$) и определяет максимально допустимый размер популяции F. В процессе синхронизации ТРМ абонентов А и В возможны следующие ситуации:

1) выходы ТРМ абонентов А и В не совпадают – абоненты А и В и злоумышленник С не выполняют на данной итерации коррекцию весов своих ТРМ;

2) выходы ТРМ абонентов А и В совпадают, и $F_C < F$. Злоумышленник С заменяет все ТРМ своей популяции, чьи выходы не совпали с выходами ТРМ абонентов А и В, и добавляет несколько новых ТРМ таких, что выходы новых ТРМ совпадают с выходами ТРМ абонентов А и В. Создание новых ТРМ осуществляется с помощью операций мутации или скрещивания ТРМ из предыдущего состояния популяции. Для всех ТРМ выполняется коррекция весов нейронов входного слоя;

3. Выходы ТРМ абонентов А и В совпадаю, и $F_C \geq F$. Злоумышленник С удаляет все ТРМ, чьи выходы не совпали с выходами ТРМ абонентов А и В. Для оставшихся ТРМ выполняется коррекция весов нейронов входного слоя.

Генетический алгоритм может быть успешно применен для атаки на протоколы обмена ключевой информацией, использующие ТРМ с малыми значениями N, M, L. По мере увеличения значений перечисленных параметров вычислительная сложность применения генетического алгоритма для осуществления успешной атаки на протокол обмена ключевой информацией растет в экспоненциальном масштабе [5].

4. *Геометрическая атака.* Геометрическая атака основана на геометрической интерпретации ТРМ. Каждый нейрон входного слоя ТРМ может быть представлен точкой W_m в пространстве Z^N. Входная матрица X ТРМ состоит из $G = MN$ элементов. Матрица X описывает M гиперплоскостей в пространстве Z^N:

$$L_m(W_m) = \sum_{n=1}^{N} x_{mn} w_{mn}\,; m = \overline{1, M}\,. \qquad (5)$$

Аргументом функции $L_m(W_m)$ является вектор весов m–го нейрона входного слоя ТРМ.

Злоумышленник С может создать свою собственную ТРМ и выполнять ее синхронизацию параллельно с синхронизацией ТРМ пары А и В. На каждой итерации процесса синхронизации возможно возникновение одной из следующих ситуаций:

1) выходы ТРМ абонентов А и В не совпадают. Абоненты А и В и злоумышленник С не выполняют на данной итерации коррекцию весов своих ТРМ;

2) выходы ТРМ абонентов А и В совпадают, выход ТРМ злоумышленника С совпадает с выходом ТРМ абонентов А и В. Все три участника выполняют коррекцию весов своих ТРМ;

3) выходы ТРМ абонентов А и В совпадают, выход ТРМ злоумышленника С не совпадает с выходом ТРМ абонентов А и В. В данном случае абоненты А и В корректируют веса своих ТРМ. Злоумышленник С должен также выполнить некоторые корректирующие действия для того, чтобы повысить шансы на успешное завершение атаки. Неравенство $f_C \neq f_A$ достигается за счет того, что в матрице X нечетное число нейронов входного слоя ТРМ абонента А и злоумышленника С дало различные друг от друга результаты. Для синхронизации своей ТРМ злоумышленник должен:

1) определить индекс m нейрона входного слоя, давшего результат, отличный от результата m–го нейрона входного слоя ТРМ абонента А;

2) инвертировать веса m–го нейрона входного слоя, чтобы стало выполняться равенство: $f_C = f_A$;

3) выполнить коррекцию весов своей ТРМ.

Индекс m злоумышленник С выбирает по принципу минимизации влияния ошибочного выбора.

Пусть выходы m–х нейронов входного слоя ТРМ абонента А и злоумышленника С не совпадают, тогда выполняется неравенство:

$$\left|L_m(W_m^C) + L_m(W_m^A)\right| > \left|L_m(W_m^C)\right|. \qquad (6)$$

Следовательно, инверсия весов m–го нейрона ТРМ злоумышленника С приведет к тому, что вектор весов данного нейрона приблизится к вектору весов m–го нейрона ТРМ абонента А (то есть $\left|L_m(W_m^C) + L_m(W_m^A)\right|$ уменьшиться) и условие (6) перестанет выполняться.

Если выходы m–х нейронов входного слоя ТРМ абонента А и злоумышленника С совпадают, то условие (6) заведомо не выполняется

и инверсия весов m–го нейрона входного слоя TPM злоумышленника C только отдалит вектор весов данного нейрона от вектора весов соответствующего нейрона входного слоя TPM абонента A.

Инверсия весов m–го нейрона входного слоя TPM злоумышленника C всегда перемещает нейрон на противоположную сторону относительно гиперплоскости L_m на расстояние, равное $r = 2\left|L_m(W_m^C)\right|$. Минимизация влияния ошибочного выбора состоит в минимизации значения r, таким образом, индекс m выбирается по формуле:

$$m = \arg\min_j r(m) = \arg\min_m \left|L_m(W_m^C)\right|.$$

Вероятность успешного осуществления атаки крайне мала – скорость синхронизации TPM абонентов A и B тем выше, чем ближе веса нейронов входного слоя TPM абонентов A и B друг к другу. Злоумышленник C на каждом шаге процесса синхронизации вынужден делать некоторые действия, выходящие за рамки процесса синхронизации, дабы поддержать темп синхронизации своей TPM с TPM абонентов – эффективность таковых действий уменьшается на каждой итерации процесса синхронизации. Таким образом, геометрическая атака может быть успешно применена для компрометации протоколов обмена ключевой информацией, использующих простейшие версии TPM – TPM с малыми значениями N, M, L.

Экспериментальное исследование. Для проведения экспериментального исследования вероятности успешной реализации атак на протокол обмена ключевой информацией разработан программный комплекс, моделирующий выполнение протокола и осуществляющий рассмотренные атаки (кроме атаки методом грубой силы). Целью исследования является получение зависимости вероятности $p(M,N)$ успешной реализации атаки на протокол от длины ключа (чисел M и N, определяющих размер ключа $G = MN$ в битах). Для каждой из исследуемых атак по результатам ряда экспериментов была построена зависимость вероятности $p(M,N)$.

На рисунке 1 приведена зависимость $p(M,N)$ для атаки методом синхронизации злоумышленником собственной TPM (количество опытов – 10003). На рисунке 2 - для генетической атаки (количество опытов – 765). На рисунке 3 – для геометрической атаки (количество опытов – 12397).

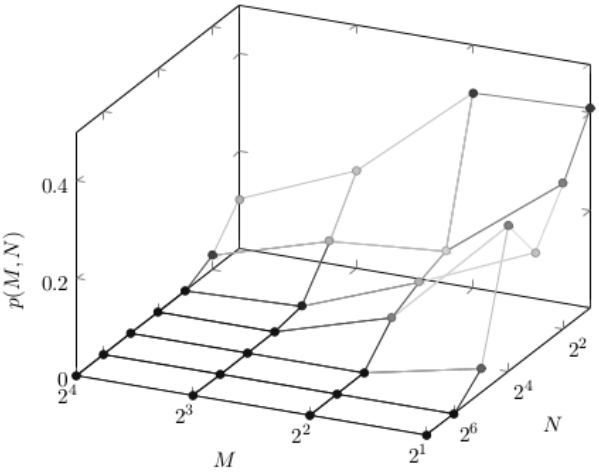

Рисунок 1 – Атака методом синхронизации злоумышленником собственной TPM

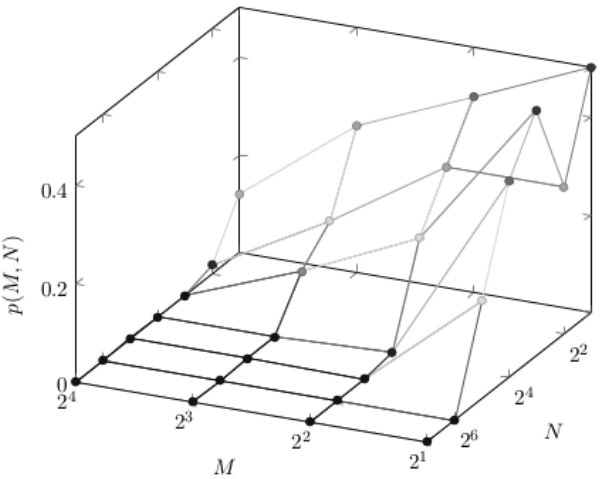

Рисунок 2 – Генетическая атака

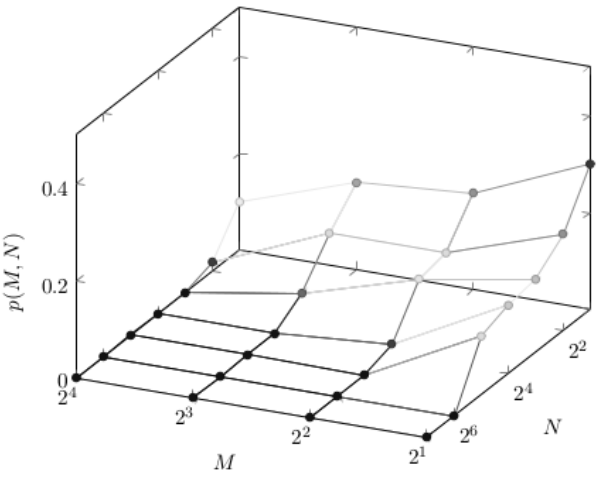

Рисунок 3 – Геометрическая атака

Рассмотрев возможные атаки на протокол обмена ключевой информацией, использующий древовидные машины четности без обратных связей, можно сделать следующие выводы:

– для малых значений N, M и L ($N, M, L \le 2^6$) возможна вычислительно несложная компрометация протокола рассмот-

ренными атаками с вероятностью $p(M,N)=0.2..0.4$;

– для больших значений N, M и L ($N,M,L \geq 2^7$) вероятность успешного осуществления любой из рассмотренных атак существенно снижается из–за экспоненциального роста их вычислительных сложностей и $p(M,N) \leq 0.01$.

Заключение. В статье рассмотрен протокол обмена ключевой информацией, основанный на использовании ИНС класса «древовидная машина четности без обратных связей». По результатам теоретического и практического исследований наиболее опасной атакой на протокол следует признать геометрическую атаку. Каждая из рассмотренных атак на протокол практически не осуществима для ключей, размер которых превышает $2^7 = 128$ бит, что в сравнении с современными способами обмена ключевой информацией дает выигрыш от 8 до 10 раз в размере ключа и, следовательно, в пропорциональных масштабах снижает вычислительные затраты на дальнейшее его использование.

Библиографический список

1. *Ruttor A.* Neural Synchronization and Cryptography — Institute for Theoretical Physics and Astrophysics, Universitat Wurzbur, Wurzburg, Germany, 2006.

2. *Volkmerand M., Wallner S* Tree Parity Machine Rekeying Architectures – Hamburg University of Technology Department of Computer Engineering, Hamburg, Germany, 2004.

3. *Volkmerand M., Wallner S* Tree Parity Machine Rekeying Architectures for Embedded Security – Hamburg University of Technology Department of Computer Engineering, Hamburg, Germany, 2005.

4. *Klimov A., Mityaguine A., Shamir A.* Analysis of Neural Cryptography — Computer Science department, The Weizmann Institute, Rehovot, Israel, 2002.

5. *Рутковская Д., Пилиньский М., Рутковский Л.* Нейронные сети, генетические алгоритмы и нечеткие системы / пер. с польск. И.Д. Рудинского. — М.: Горячая линия – Телеком, 2006.