Принцип построения многопутевого протокола динамической маршрутизации видеопотоков

В.Ю. Казанов

Аннотация. Рассматривается задача построения протокола эффективной маршрутизации видеопотоков форматов WMV и MPEG4 с учетом их классификационных и статистических особенностей в общем трафике, а также возможности использования многопутевой маршрутизации и адаптации таблицы маршрутизации с учетом динамики сети по выбранному критерию. В качестве алгоритмической базы построения таблицы предложен эвристический алгоритм нахождения максимальных потоков в сети с применением элементов теории линейных пространств.

Ключевые слова: маршрутизация, протокол, поток, минимальный разрез.

Введение

Любой протокол маршрутизации предназначен для автоматического построения таблицы маршрутизации, на основе которой происходит продвижение пакетов данных. При этом одним из основных требований, предъявляемых к любому протоколу, является возможность динамической адаптации таблицы маршрутизации к текущему набору рабочих маршрутов в сети с целью, повышения ее производительности и надежности. Однако понятие «рабочий маршрут» для каждого протокола может отличаться в зависимости от набора включенных в него характеристик, в том числе и с учетом накладываемых на них ограничений. Он также определяет механизм отслеживания изменения характеристик и может выступать как в качестве критериев оптимизации, так и адаптации. Однако основными оптимизационными критериями качества работы самого протокола маршрутизации, как правило, являются среднее время $(t_{\tilde{n}\partial})$ и целостность доставки пакета конечному узлу. Поэтому при разработке любого протокола маршрутизации данных методическую базу следует выбирать исходя из постановки задачи, предъявляемых требований и критериев оптимизации.

Рассмотрим основные подходы к решению задачи оптимизации потоков в сети путем их перераспределения как одному из ключевых факторов сокращения времени доставки пакетов конечному узлу. К ним можно отнести: методы по обеспечению требуемого качества обслуживания (Quality of Service, QoS), методы маршрутизации и технологию инжиниринга трафика (TE). Работа службы QoS заключается в следующем: при передаче пакета в сеть QoS дополняет его своей служебной информацией, в которой указывается приоритет данного пакета по отношению к другим, требования к скорости его доставки и др. [1]. При этом каждый промежуточный узел должен зарезервировать часть своей вычислительной мощности с целью обработки привилегированных пакетов для обеспечения предъявляемых требований по их обслуживанию. Это является существенным недостатком в работе данной службы в условиях дефицита пропускной способности канала связи наряду с тем, что данная служба не отве-

чает за маршрут передачи данных. Безусловно, особое место занимают методы маршрутизации. Однако анализ существующих методов маршрутизации выявляет низкую эффективность даже такого мощного средства, как выбор путей прохождения трафика через сеть, традиционно используемого во многих типах сетей, в значительной степени из-за прямолинейности при выборе критерия оптимизации маршрута. Например, дистанционно-векторные алгоритмы (DVA) в качестве критерия оптимизации маршрута (метрики) используют число транзитных переходов до сети назначения, что далеко не всегда является обоснованным. Алгоритмы состояния связей (LSA) как метрику обычно используют пропускную способность каналов связи [1]. В условиях постоянно растущей загрузки каналов передачи данных, а также ужесточения требований по качеству и скорости обслуживания пакетов, оптимизация маршрута на базе пропускной способности сети является более предпочтительной, так как в этих условиях кратчайший маршрут по расстоянию далеко не всегда будет еще и самым коротким по времени. Наиболее распространенный протокол OSPF, работающий по принципу LSA, имеет один существенный недостаток – это его совокупная вычислительная сложность, которая быстро растет с увеличением количества сетей, маршрутизаторов и связей между ними [1]. Одним из путей решения этой проблемы является разбиение одной большой непрерывной области маршрутизации всей сети на подобласти. Такой подход рассматривался К.В. Кринкиным при разработке алгоритма маршрутизации в динамических компьютерных сетях с использованием неполных данных [2]. Но при этом возникают проблемы выбора размера области эффективной маршрутизации и критериев её адаптации к динамике сети. Одним из мощных методов оптимизации потоков в сети является технология TE, где используются методы и механизмы достижения сбалансированной загрузки всех компонентов сети за счет рационального выбора пути прохождения трафика через сеть с учетом его специфики [3]. Ниже рассматривается принцип построения протокола маршрутизации пакетов потокового видео, имеющего конечную ширину полосы скорости передачи данных с применением графовой модели топологического представления сети и критерия оптимизации маршрута (метрики) на базе остаточных пропускных способностей каналов передачи данных с возможностью выбора нескольких маршрутов транспортировки пакетов.

Под потоковым видео будем понимать непрерывный случайный (вероятностный) стационарный в условиях прямопотоковой технологии процесс ($X(t)$) передачи видеопакетов, полученных сжатием по технологии WMV или MPEG4 и имеющих свои классификационные особенности. В свою очередь прямопотоковая технология - это технология сжатия и буферизации данных, которая позволяет транслировать видео в реальном времени через Интернет [4].

Принципиальной особенностью потокового видео является его непрерывность, и оно проигрывается по мере того, как передается на конечный узел. Иначе, среднее значение интенсивности потока равно исходному значению скорости передачи данных, с которой данный поток был сжат. Условие стационарности исключает наличие в нем пульсаций, а средняя скорость передачи данных говорит о его средней интенсивности на всем временном периоде передачи. Данные результаты были получены в ходе анализа видеопотоков WMV и MPEG4 их стационарность, а также в результате проведенного статистического моделирования [5]. Таким образом, принимая предположение о стационарности потока и зная среднее значение его интенсивности, можно прогнозировать ход протекания процесса передачи видеопотока на весь временной период.

Постановка задачи

Основная задача формулируется следующим образом: реализовать основные механизмы функционирования многопутевого протокола динамической маршрутизации видеопотоков с учетом особенностей маршрутизируемой информации при выборе критериев адаптации и оптимизации маршрута, а также при реализации механизма (алгоритма) построения таблицы маршрутизации. Так как центральным звеном во всей технологической цепочке является таблица маршрутизации, алгоритм ее построе-

ния предложено подчинять комплексному критерию F, введенному на множестве возможных алгоритмов P и включающему в себя такие показатели, как сложность и гибкость подхода. В этом случае применительно к разработке алгоритма маршрутизации постановка задачи формулируется следующим образом:

$$F(p_i) \to \max, \quad p_i \in P.$$

Основные процедуры построения протокола маршрутизации видеопотоков

Процесс функционирования протокола маршрутизации данных, как правило, осуществляется по определенному замкнутому циклу, причем перечень выполняемых при этом процедур и их последовательность в принципе детерминированы в независимости от вида протокола. Различия заключаются только в подходах к реализации той или иной процедуры. Перечень этих процедур обычно включает:

- процедуру построения топологической базы данных сети;
- процедуру сбора информации о состоянии сети для вычисления выбранной метрики;
- процедуру (алгоритм) построения таблицы маршрутизации;
- процедуру подтверждения и обновления информации о состоянии каналов передачи данных;
- процедуру адаптации таблицы маршрутизации по определенному критерию.

Однако прежде чем перейти к рассмотрению каждой из них в рамках разработки динамического протокола маршрутизации видеопотоков, следует отметить, что выполнение каждой процедуры сопровождается обменом служебной информацией между участниками процесса маршрутизации в сети. Обмен происходит по правилам, описанным в каждой из процедур, с использованием сообщений единого формата со следующим возможным набором реквизитов:

1. $P_{type} \in \{1,2,3,4,5\}$ – тип сообщения: 1 – сообщение TREE, 2 – сообщение HELLO, 3 – сообщение ANSWER, 4 – сообщение CHANGE_FLOW, 5 – сообщение DROP;
2. L_{msg} – длина сообщения;
3. GUID – уникальный идентификатор маршрутизатора;
4. t_{msg} – временная метка сообщения;
5. $I(v_i)$ – множество активных подсетей с соответствующими масками;
6. R – число транзитных переходов;
7. $c(v_i, v_j)$ – остаточная пропускная способность канала $(v_i, v_j) \in E$, $\left[Кбайт/с\right]$.

Процедура $B(v)$ построения топологической базы данных сети $M(v)$ предполагает использование механизма построения графа связей $G(V,E)$ по принципу маршрутизатор (u)-маршрутизатор (v), где $E = \{(u,v) \mid u,v \in V\}$ – множество связей. При этом следует отметить ряд ограничений: любой канал передачи данных (u,v), $u,v \in V$ принято считать полным дуплексным; значение остаточной пропускной способности канала $c(u,v) = a(u,v) - f(u,v)$, где $a(u,v)$ – ширина полосы пропускной способности канала $(u,v) \in E$, $f(u,v)$ – текущая интенсивность потока в канале $(u,v) \in E$, ограниченная сверху $c_{\max}$ и снизу $c_{\min}$.

Процедура $B(v_i)$, $i = \overline{1,|V|}$ выполняется маршрутизатором v_i непосредственно после выполнения инициализации протоколом после включения. По всем активным каналам посылается сообщение TREE, содержащее значения всех реквизитов, которые, за исключением $c(v_i, v_j)$ и R, остаются неизменными на протяжении всего жизненного цикла данного сообщения. При этом начальное значение $c(v_i, v_j) = -1$ соответствует начальному значению $R = 0$. Каждый соседний с v_i маршрутизатор $v_j \in V$, получивший данное сообщение, выполняет его обработку с целью обновления своего графа связей, представленного в виде матрицы инцидентности $M(v_j)$, а также установления значения остаточной пропускной способности $c(v_i, v_j)$ в полученном сообщении TREE. Стоит отметить, что матрица инцидентности $M(v_j)$ к моменту получения данного сообщения уже содержит по крайней мере хотя бы одну запись с данными по каналам

связи, инцидентным узлу v_j. В конечном счете, размерность матрицы $M(v_j)$ будет равна $n \times m$, где $n = |V|$, $m = |E|$. Обработка сообщения TREE в случае $R = 0$ начинается с вызова *процедуры сбора информации о состоянии сети для вычисления выбранной метрики.* Результатом вызова данной процедуры является возвращение установленного значения остаточной пропускной способности канала $c(v_i, v_j)$. После этого или в случае, если $R > 0$, обработка сообщения TREE начинается с выборки поля GUID и поиска строки в матрице $M(v_j)$ с соответствующей записью о маршрутизаторе. Если таковая отсутствует, соответствующая запись о маршрутизаторе добавляется в матрицу $M(v_j)$. После этого просматривается множество $I(v_i)$ на предмет отсутствия записи о какой-либо подсети из $I(v_i)$ в столбцах $M(v_j)$. В случае, если обнаруживается такая подсеть, запись о ней вместе с маской добавляется в $M(v_j)$, при этом проставляются отметки об инцидентности соответствующих подсетей маршрутизаторам $v \in M(v_j)$. Следующим шагом работы данной процедуры является наращивание значения реквизита R на единицу, и в случае, если матрица $M(v_j)$ была расширена вследствие обработки полученного сообщения TREE, данный маршрутизатор v_j ретранслирует обработанное сообщение TREE на все свои интерфейсы за исключением того, на который данный пакет был принят. При этом значения всех его реквизитов остаются в неизменном виде, кроме R и $c(v_i, v_j)$, значения которых могли быть модифицированы. В противном случае, если в процессе обработки данного сообщения TREE было установлено, что запись о данном маршрутизаторе и его подсетях уже существовала в базе $M(v_j)$, то оно удаляется и дальнейшей рассылке не подлежит. Однако в любом случае считывается значение поля $c(v_i, v_j)$ и запоминается в топологической базе $M(v_j)$ как значение метрики соответствующего канала связи.

Таким образом, в результате рассылки объявлений в виде пакетов TREE через зону все маршрутизаторы $v \in V$ строят идентичную базу данных состояния каналов связей. После истечения заданного времени ожидания пакетов объявлений TREE считается, что процедура построения топологии сети завершена, и узел v_i переходит в режим маршрутизации.

Процедура сбора информации о состоянии сети для вычисления выбранной метрики выполняется узлом $v_j \in V$ по вызову из *процедуры* $B(v_i)$ *построения топологической базы данных сети* непосредственно во время выполнения операции построения топологической базы $M(v_j)$. Она необходима для сбора информации о загруженности канала связи $(v_i, v_j) \in E$. Для этих целей узел v_j использует значения параметров полученного сообщения TREE, и в особенности, значения его временных параметров. Значение параметра $c(v_i, v_j)$ может быть рассчитано на основании полученного пакета TREE по следующим выражениям:

$$c(v_i, v_j) = \frac{S \cdot t_{обсл}}{t_{расч}}, \left[Кбайт/с \right], \; t_{расч} = \frac{\mathbf{L}}{\mathbf{S}},$$

где S – пропускная способность канала связи $(v_i, v_j) \in E$ $\left[Кбайт/с \right]$, L – величина пакета $[Кбайт]$, $t_{расч}$ – расчетное время обслуживания пакета размерности L с пропускной способностью S канала связи $(v_i, v_j) \in E$ $[c]$, $t_{обсл}$ – экспериментальное время обработки пакета длины L $[c]$. Таким образом, на этапе пересылки сообщения TREE от маршрутизатора v_i к соседнему маршрутизатору v_j может быть установлено значение параметра $c(v_i, v_j)$, $(v_i, v_j) \in E$.

Процедура (алгоритм) построения таблицы маршрутизации является ключевой в работе любого протокола маршрутизации данных и основная трудоемкость процесса приходится на эту процедуру. От эффективности ее реализации в значительной степени зависит и эффективность протокола в целом, так как все остальные составляющие протокола призваны

выполнять функции по сбору и предоставлению информации, необходимой для построения таблицы маршрутизации согласно алгоритмическому обеспечению. Применительно к данному протоколу необходимо отметить, что процесс построения таблицы маршрутизации $R^{table}(v_j)$ выполняется в два этапа. Таблица включает в себя следующие поля:

- номер сети N_l,
- маска сети,
- множество адресов следующих маршрутизаторов $A(N_l)$,
- множество интерфейсов $X(N_l)$,
- множество значений метрики $S(N_l)$,

На первом этапе происходит заполнение таблицы маршрутизации узла $v_j \in V$ первичной информацией об имеющихся в сети узлах маршрутизации, и этот процесс выполняется параллельно с *процедурой построения топологической базы данных сети*. После включения и инициализации протокола и получения информации из матрицы $M(v_j)$ таблица маршрутизации $R^{table}(v_j)$ будет содержать записи только о непосредственно подсоединенных сетях. Дальнейшие записи о сетях в таблице маршрутизации будут появляться в результате параллельной с *процедурой построения топологической базы данных сети* обработки получаемых сообщений TREE. При этом правила, по которым будет постепенно наращиваться информация о сетях в таблице маршрутизации, будут следующими.

1. Получив сообщение TREE, маршрутизатор v_j запоминает интерфейс, через который от маршрутизатора $v_i \in V$ была получена информация о сетях, содержащихся во множестве $I(v_i)$. Адрес маршрутизатора $v_i \in V$ также запоминается, так как он станет адресом следующего маршрутизатора при обновлении или добавлении записей о сетях из множества $I(v_i)$ в таблице $R^{table}(v_j)$.

2. Маршрутизатор v_j производит сравнение новой информации о сетях с той, которая хранится в его таблице маршрутизации. Результат сравнения записей множества $I(v_i)$ и $N_l \in R^{table}(v_j)$ может быть одним из следующих:

a) $I(v_i) \setminus N_l(R^{table}(v_j)) = \varnothing$, в этом случае все записи о сетях из сообщения TREE уже содержатся в таблице $R^{table}(v_j)$,

b) $I(v_i) \setminus N_l(R^{table}(v_j)) \neq \varnothing$, в сообщении TREE существуют записи о сетях, отсутствующих в таблице $R^{table}(v_j)$.

3. В случае (b) новые записи об отсутствующих сетях и их масках добавляются в таблицу $R^{table}(v_j)$ и обновляется информация во множествах $A, X \in N_l(R^{table}(v_j))$, определяющих альтернативные маршруты при транспортировке пакетов в сеть N_l.

4. В случае (a) происходит только обновление маршрутной информации для сетей $N_l \in I(v_i)$ во множествах $A, X \in N_l(R^{table}(v_j))$.

Таким образом, в результате обработки сообщений TREE, параллельно с процедурой построения топологической базы данных сети формируется таблица многопутевой маршрутизации пакетов видеопотоков, содержащая первичную информацию об имеющихся в сети узлах маршрутизации, завершая тем самым первый этап работы алгоритма.

На втором этапе выполняется эвристический алгоритм, позволяющий сформировать множества $S(N_l)$ для всех $N_l \in R^{table}(v_j)$ на основании полученных $M(v_j)$ и $R^{table}(v_j)$. Таким образом, основная задача данного алгоритма заключается в том, чтобы расставить значения метрик для всех потенциальных маршрутов транспортировки пакетов видеопотоков от истока v_j ко всем $v_i \in V \setminus \{v_j\}$. С учетом того, что метрика для каждого маршрута определяется на основе остаточных пропускных способностей каналов передачи данных, связывающих маршрутизаторы между собой, необходимо сформировать величины максимальных потоков для всех $N_l \in R^{table}(v_j)$. Алгоритм решает эту задачу с применением методов теории линейных пространств. Сформулированная и доказанная

Фордом и Фалкерсоном в 1961 году теорема о максимальном потоке и минимальном разрезе позволяет переформулировать задачу о нахождении максимального потока как задачу поиска минимального разреза для всех пар узлов маршрутизации $(v_j, v_i),\ v_i \in V \setminus \{v_j\}$. Применение теории линейных пространств позволяет рассматривать разрез как вектор $t = (t^1, t^2, ..., t^m)$ с координатами t^a над числовым полем $(Z_2, \oplus, \cdot)$ с операциями сложения и умножения по модулю, где каждая координата t^a взаимно однозначно соответствует ребру a, $t^a = 1$, если ребро a входит в разрез, в противном случае $t^a = 0$.

При нахождении минимальных разрезов для всех пар $(v_j, v_i),\ v_i \in V \setminus \{v_j\}$ на множестве E алгоритм решает ряд подзадач, таких как: построение максимального остова $G^T(V^T, E^T)$, где $V^T \subseteq V, E^T \subset E$, формирование базиса пространства разрезов T^B, формирование линейных комбинаций базисных разрезов $T^{\cup} = \left\{ (x \cdot t_1 \oplus x \cdot t_2 \oplus ... \oplus x \cdot t_l) \middle|\ t_i \in T^B,\ x \in Z_2,\ l = \left|E^T\right| \right\}$ и, наконец, на основании $T^{\cup}$ и T^B для каждого узла $v_i \in V \setminus \{v_j\}$ определяется минимальный разрез, а соответственно, и величина максимального потока.

Построение максимального остова $G^T(V^T, E^T)$ выполняется по схеме Прима с применением фибоначчиевых куч при организации очередей Q и Q^T, а также при написании процедуры выборки из них максимальных элементов *Extract-Max* по заданному ключу. Выполнение этой процедуры необходимо для того, чтобы впоследствии сформировать базис пространства разрезов T^B на графе $G(V, E)$ размерностью $|V| - 1$ или l, так как $\left|E^T\right| = |V| - 1$. При этом формирование векторов $t \in T^B$ начинается уже на этапе построения максимального остова $G^T(V^T, E^T)$ при выборке для каждого узла $u \in V^T$ списка смежных вершин $v \in Adj[u],\ (u, v) \in E$. Факт построения T^B на остове $G^T(V^T, E^T)$ определяется тем, что удаление любого ребра, входящего в каркас $G^T(V^T, E^T)$, разбивает его на две связные компоненты. Принадлежность вершин этим компонентам определяет разрез. При этом любое ребро каркаса $G^T(V^T, E^T)$ в совокупности с ребрами хорд образует один линейно независимый разрез $t_u \in T^B$ [6,7].

Введем для всех $v_i \in V^T$ поле $\pi(v_i),\ \pi(v_j) = null$, указывающее на предшествующую v_i вершину. Тогда для формирования системы T^B необходимо организовать выборку вершин v_i из очереди $Q^T = V^T \setminus \{v_j\}$ с приоритетами, окончательно формируя при этом для вершины v_i вектор

$$t_{\pi(v_i)} = (1 \cdot t^{(v_i, \pi(v_i))},\ x \cdot t^{(1)},\ x \cdot t^{(2)},\ ...,\ x \cdot t^{(m)}), \quad (1)$$

где $m = \left|E \setminus E^T\right|$, $t^{(v_i, \pi(v_i))} \in E^T$, $t_{\pi(v_i)} \in T^B$, $t^{(1)},\ t^{(2)}, ...,\ t^{(m)} \in E \setminus E^T$, $\omega(t^{(v_i, \pi(v_i))} \geq \omega(t^{(a)}))$, $a = \overline{1, m}$, $x \in Z_2$. Приоритет вершины v_i определяется в порядке убывания значения $d[v_i]$, которое равно расстоянию v_i от корня v_j, $d[v_j] = 0$. Сформированный таким образом базис пространства векторов T^B дает возможность получить множество линейно зависимых разрезов $T^{\cup}$ на графе $G(V, E)$, среди элементов которого выполняется поиск минимальных разрезов для вершин $v_i \in V \setminus \{v_j\}$. Ниже приведен эвристический алгоритм поиска минимальных разрезов $\langle v_j | v_i \rangle,\ v_i \in V \setminus \{v_j\}$ на графе $G(V, E)$ в псевдокоде с его основной процедурой *Max-Flows-Calculation.*

```
procedure Max-Flows-Calculation(v_j, G(V,E))
    array[V] Q^T
    call Build-Max-Spanning-Tree (v_j, G(V,E), Q^T)
```

call Build-of-Basis-Cuts ($G^T(V^T, E^T)$)

call Linear-Combinations-of-Basis-Cuts (T^B)

Из полученных множеств T^B и $T^{\cup}$ назначаются минимальные разрезы для вершин $v \in V \setminus \{v_j\}$

end procedure

//Процедура формирования максимального остова

procedure Build-Max-Spanning-Tree (v_j, $G(V,E)$, Q^T)

Q=V[G]

Q^T=V[G]

for each $u \in Q$

do key[u]=0

d[u]=0

key[v_j]=∞

$\pi[v_j] = null$

while $Q \neq 0$ **do**

u=Extract-Max(Q) (в порядке убывания key[u])

for each $v \in Adj[u]$ **do**

if $v \in Q$ **and** $\omega(u,v) < key[v]$ **then**

$\pi[v] = u$

$key[v] = \omega(u,v)$

$d[v] = d[u] + 1$

$t_u[t^{(u,v)}] = 1$

$t_v[t^{(v,u)}] = 1$

end if

end for each

end while

return $G^T(V^T, E^T)$

end procedure

//Процедура формирования базиса T^B на графе $G(V,E)$

procedure Build-of-Basis-Cuts($G^T(V^T, E^T)$)

while $Q^T \neq 0$ **do**

u=Extract-Max(Q^T) (в порядке убывания d[u])

$v = \pi(u)$

if v=null

do continue

$t_v[t^{(v,u)}] = 0$

for each $t^{(i)} \in t_u$, $t^{(i)} \in E \setminus E^T$ **do**

$$t_v[t^{(i)}] = 1 \Leftrightarrow t_u[t^{(i)}] = 1 \vee t_v[t^{(i)}] = 0$$

$$t_v[t^{(i)}] = 0 \Leftrightarrow t_u[t^{(i)}] = 1 \vee t_v[t^{(i)}] = 1$$

end for each

end while

return T^B

end procedure

//Процедура формирования множества линейных комбинаций $T^{\cup}$ в базисе T^B

procedure Linear-Combinations-of-Basis-Cuts (T^B)

```
T^∪ = ∅
for each t_i ∈ T^B do
        for each t_j ∈ T^B, t_j ≠ t_i
        do if t_j ⋂ t_i ≠ ∅
                do T^∪ ⋃ (t_i ⊕ t_j)
    end for each
return T^∪
end procedure
```

Описанный алгоритм имеет ряд особенностей, которые требуют особого рассмотрения. Прежде всего это эвристический подход и необходимость построения максимального остова. Применение эвристики в алгоритме позволяет улучшить его асимптотическую оценку за счет оптимизации процесса вычисления линейных комбинаций векторов $t \in T^B$, максимальное количество которых оценивается выражением $T^{\cup} = 2^{|V|-1} - 1$ [6,7]. Таким образом, очевиден экспоненциальный рост линейных комбинаций в зависимости от размерности множества V, что и показано на Рис. 1.

Сформулируем подход, базирующийся на принципах эвристики и заключающийся в том, что поиск минимальных разрезов производится на множестве T^B и подмножестве элементов множества $T^{\cup}$, образуемых линейными комбинациями двух векторов $t_i \oplus t_j$, $t_i, t_j \in T^B$, $t_i \bigcap t_j \neq \varnothing$, $t_i \neq t_j$. При этом формирование множества T^B по выражению (1) на основании максимального остова $G^T(V^T, E^T)$ позволяет равномерно распределить вес множества E^T между всеми разрезами $t \in T^B$ путем включения в каждый из них ровно по одному ребру $e \in E^T$. Данный механизм уже на первом этапе позволяет во многом добиться минимизации величин базисных разрезов $t \in T^B$. Благодаря этой гипотезе имеется возможность ввести ограничения на поиск минимальных разрезов на множестве $T^{\cup}$, так как статистические результаты показывают, что в этом случае с увеличением числа линейных комбинаций базисных разрезов $t \in T^B$ наблюдается значительное замедление скорости прироста вероятности (p) получения минимальных разрезов и значительно замедляется прирост качества формируемых разрезов линейными комбинациями результатов. Статистический анализ проводился на различных топологиях сетей, которые создавались

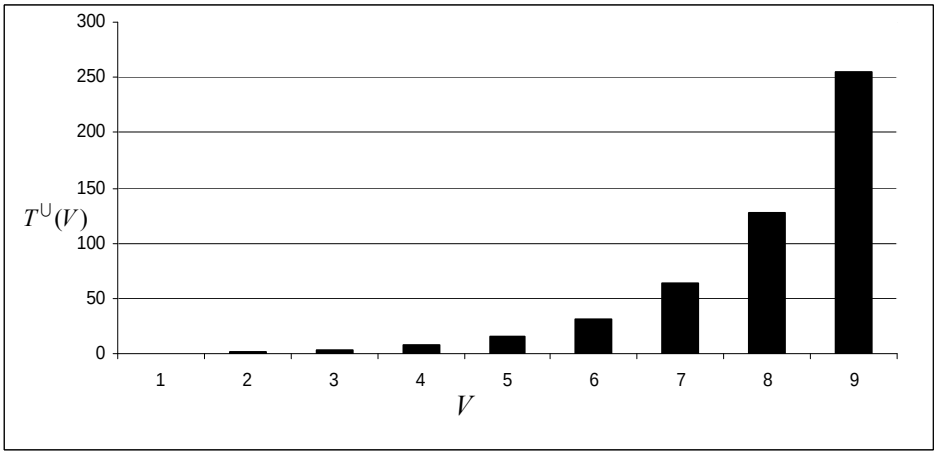

Рис. 1. Рост числа линейных комбинаций в зависимости от размерности множества V

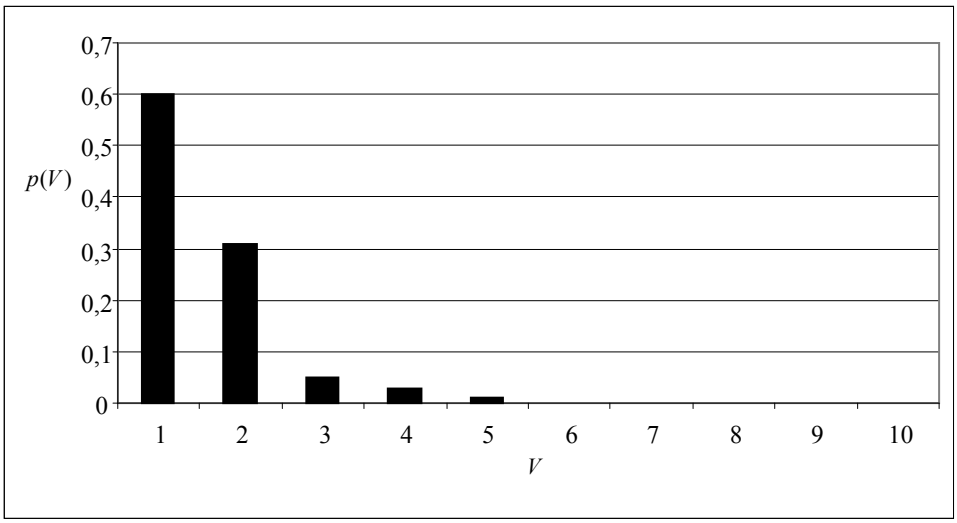

Рис. 2. Частотное распределение вероятности получения минимальных разрезов в зависимости от размерности множества *V*

визуальным конструктором топологий на основе заданных параметров (количество узлов и степень связности) при различной динамике структуры. Веса каналов передачи данных назначались генератором случайных чисел, подчиненным нормальному закону распределения. На Рис. 2 представлены полученные результаты в виде гистограммы, которая отражает частотное распределение вероятности получения минимальных разрезов в зависимости от числа линейных комбинаций базисных разрезов $t \in T^B$, определяемое размерностью V.

Учитывая тот факт, что суммарная пропускная способность вычислительной сети находится в постоянной динамике, можно утверждать, что тот незначительный процент возможных узлов, для которых не будут найдены в рамках описанного алгоритма минимальные разрезы, не окажет значительного влияния на процесс маршрутизации видеопотоков в целом.

Рассмотрим в качестве примера неориентированный взвешенный граф $G(V,E)$, который представлен на Рис. 3.

Определив в качестве истока вершину $s=\{1\}$, построим для данного графа процедурой *Build-Max-Spanning-Tree* максимальный остов $G^T(V^T,E^T)$, $E^T=\{a,b,d,e,g\}$, $V^T=\{1,2,3,4,5,6\}$ и систему базисных векторов T^B на основе максимального остова $G^T(V^T,E^T)$ с помощью процедуры *Build-of-Basis-Cuts*. В результате получим T^B={(*g,h*), (*d, f, h*), (*e, f, h*), (*a, c, f, h*),

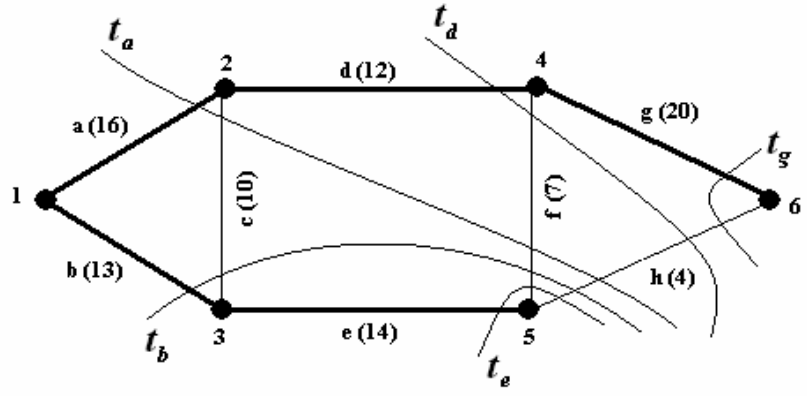

Рис. 3. Неориентированный взвешенный граф *G*(*V,E*), его максимальный остов $G'(V',E')$ (выделен жирными линиями) и система базисных разрезов T^B

$(b, c, f, h)\}$, $\omega(t_g)$=24, $\omega(t_d)$=23, $\omega(t_e)$=25, $\omega(t_b)$=34, $\omega(t_a)$=34, $\omega(t_b)$=34. Пусть $T_{\min}(v)$ – минимальный разрез $\langle s|v\rangle$, $v \in V$, тогда по результатам работы процедуры *Linear-Combinations-of-Basis-Cuts* получим $T_{\min}(\{4,6\}) = \omega(t_d) = 23$, $T_{\min}(\{5\}) = \omega(t_e) = 25$, $T_{\min}(\{2,3\}) = t_a \oplus t_b = 29$. Таким образом можно видеть, что для вершин {4,5,6} минимальные разрезы принадлежат множеству T^B, а для вершин {2,3} минимальный разрез определяется в результате линейной комбинации базисных разрезов t_a и t_b.

В силу того, что процедура *Build-Max-Spanning-Tree* является реализацией алгоритма Прима, а множество $T^{\cup}$ с учетом применения эвристики - результатом алгебраических операций во множестве T^B над числовым полем $(Z_2, \oplus, \cdot)$, корректность работы алгоритма во многом определяется процедурой построения системы T^B *Build-of-Basis-Cuts*. В момент вызова данной процедуры во всех векторах $t_i \in T^B$, однозначно соответствующих вершинам $v_i \in V^T$, ребра t^a, инцидентные вершине v_i, определены как $t^a[t_i] = 1$. При этом векторы $t_i \in T^B$ базисными не являются. Однако в процессе просмотра очереди Q^T в порядке убывания значений $d[v_i]$ и движения от висячих вершин $(d[v_i] = 1)$ вдоль ветвей $((v_i, \pi(v_i)) \in E^T)$ к корню $\pi(v_i) = null$ избыточные ребра $t^a \in E^T$ в разрезах $t_{\pi(v_i)}$ заменяются на хорды $(t^a \in E \setminus E^T)$ из разрезов t_{v_i} алгебраической операцией над числовым полем $(Z_2, \oplus, \cdot)$. В момент, когда очередь $Q^T = \varnothing$, согласно теореме о размерности пространства разрезов связного графа [6] все разрезы $t_i \in T^B$ можно считать базисными.

Рассматривая асимптотическую оценку алгоритма, необходимо отметить, что она носит аддитивный характер и зависит от сложности реализации каждой из процедур в отдельности. Так, оценка процедуры построения максимального остова с учетом отмеченных ранее механизмов составляет $O(E + 2V \log V)$ [8]. Оценку работы процедуры построения базиса T^B определяет в основном цикл, организованный для просмотра подмножества $E \setminus E^T$, число выполнений которого, в свою очередь, определяется длиной очереди Q^T. Таким образом, оценка для данной процедуры составляет $O(V^T(E \setminus E^T))$. Скорость работы процедуры построения системы линейных комбинаций $T^{\cup}$ определяется количеством возможных пар линейных комбинаций базисных разрезов $t \in T^B$, которое оценивается выражением $O(0{,}5V^2 - 1{,}5V + 1)$. С учетом выполнения операции сложения векторов по модулю 2 сложность всей процедуры в целом составляет $O\left((0{,}5V^2 - 1{,}5V + 1)E\right)$. Таким образом, асимптотическая оценка алгоритма в целом представляется выражением $O(E + 2V \log V + V^T(E \setminus E^T) + (0{,}5V^2 - 1{,}5V + 1)E)$.

Второй этап работы алгоритма маршрутизации завершается тем, что полученные значения максимальных потоков для всех узлов маршрутизации $v_i \in V \setminus \{v_j\}$ позволяют заполнить множества $S(N_l)$, $N_l \in R^{table}(v_j)$, и таким образом полностью сформировать таблицу маршрутизации $R^{table}(v_j)$.

Процедура подтверждения и обновления информации о состоянии каналов передачи данных выполняется после первоначального построения таблицы маршрутизации, и основная ее задача заключается в отслеживании изменений состояния сети. При этом следует отметить, что данная процедура реагирует только на те изменения в сети, которые привели к физическому падению канала связи. Для контроля физического состояния связей и соседних маршрутизаторов используются короткие специальные сообщения HELLO, содержащие только значения следующих реквизитов $\langle L_{msg}, t_{msg}, GUID\rangle$. Каждый маршрутизатор периодически выполняет рассылку сообщений HELLO всем своим соседним маршрутизаторам. Реакцией на получение данного сообщения является ответ маршрутизатору, от которо-

го данное сообщение было получено, в виде сообщения ANSWER, содержащего поля $\langle GUID, t_{msg} \rangle$. Если состояние связи не изменилось, то маршрутизатор, получив сообщение ANSWER как ответ на сообщение HELLO, не корректирует свою таблицу маршрутизации и не посылает соседям объявления о связях. Если же состояние связи изменилось, то маршрутизатор v_i ближайшим соседям посылает объявление DROP с реквизитами $\langle GUID, t_{msg}, I(v_i) \rangle$, содержащее во множестве $I(v_i)$ только один элемент, касающийся физического состояния данного канала связи. Подобная схема позволяет экономить пропускную способность сети, так как каждый маршрутизатор выполняет контроль только инцидентных ему каналов связи. Получив новое объявление об изменении состояния связи, протокол маршрутизации на узле v_j вынужден произвести перестройку своей топологической базы данных сети и скорректировать свою таблицу маршрутизации. Одновременно маршрутизатор v_j ретранслирует объявление каждому из своих ближайших соседей, кроме того, от которого он получил это объявление.

При появлении новой связи или нового соседа маршрутизатор узнает об этом из новых сообщений TREE. В этих сообщениях указывается достаточно детальная информация о том маршрутизаторе, который послал это сообщение, а также о его ближайших соседях, чтобы данный маршрутизатор можно было однозначно идентифицировать.

Процедура адаптации таблицы маршрутизации к изменившимся условиям в отличие от процедуры подтверждения и обновления информации о состоянии каналов передачи данных реагирует только на такие события в сети, которые привели к частичному или полному изменению остаточной пропускной способности одного или нескольких каналов связи, но при этом сами каналы связи остаются работоспособными. Таким образом, в рамках данной процедуры выполняется мониторинг динамики потоков в сети. Он производится каждым маршрутизатором в отдельности. Для этого маршрутизатор измеряет интенсивности входящих потоков по каждому своему сетевому интерфейсу $\lambda^{(v_i, v_j)}(v_i)$. Тот факт, что потоковое видео как подкласс трафика по результатам проведенных и описанных в работе [5] исследований имеет свои характеристические особенности, такие как определенная степень разброса интенсивности от среднего значения скорости трансляции потока ($\overline{\lambda}$), стационарность, а также чувствительность к задержкам и потерям пакетов, формирует определенные требования к процедуре адаптации таблицы маршрутизации и критерию адаптации. Прежде всего они связаны с необходимостью беспрепятственного продвижения пакетов потокового видео в рамках выбранных технологий сжатия. Поэтому реализацию данной процедуры маршрутизатором v_i следует разбить на две составляющие:

- определение критерия адаптации (ψ) и его ограничений;
- технология оповещения маршрутизаторов сети об изменении остаточных пропускных способностей каналов связи $c(v_i, v_j),\ (v_i, v_j) \in E$.

Ключевую роль играет первая составляющая, так как от эффективности ее реализации будет непосредственно зависеть объем передаваемого по сети служебного трафика, а также степень адекватности таблиц маршрутизации текущей загрузке каналов передачи данных в рамках особенностей выбранного трафика. По результатам статистических исследований видеопотоков, полученных для выбранных технологий сжатия [5], в качестве базового критерия адаптации ψ было принято пороговое изменение остаточной пропускной способности в канале связи $c(v_i, v_j),\ (v_i, v_j) \in E$. На основе выбранного критерия адаптации, а также исходя из необходимости оптимизации объемов служебного трафика в рамках данного протокола маршрутизации реализован механизм адаптации таблиц маршрутизации. Он реализуется по принципу принятия решения, основанного на критерии ψ. Критерий ψ отражает соотношение между максимально возможным значением интенсивности видеопотока WMV

$(\overline{\lambda}_{\max} + \sigma_{\max})\left[Кбайт/с\right]$ и текущим значением $c(v_i, v_j)$, $(v_i, v_j) \in E$, здесь $\overline{\lambda}_{\max}$ – максимальное значение скорости сжатия видеопотока, $\sigma_{\max}$ – возможное максимальное отклонение интенсивности видеопотока WMV от $\overline{\lambda}_{\max}$. Тот факт, что величина $\sigma_{\max}$ для формата WMV имеет большее значение, чем для формата MPEG4, тогда как значения $\overline{\lambda}_{\max}$ одинаковы, обуславливает выбор $\sigma_{\max}$ для потока WMV в качестве эталона. В процессе мониторинга интенсивностей входящих потоков критерий ψ рассчитывается каждый раз маршрутизатором v_i для каналов $(v_i, v_j) \in E$, $j \neq i$, $j = \overline{1, |V|}$ по следующей формуле:

$$\psi = \frac{c(v_i, v_j)}{(\overline{\lambda}_{\max} + \sigma_{\max})}.$$

Решение о необходимости перестроить свою таблицу маршрутизации, а также об информировании других маршрутизаторов в сети об изменениях $c(v_i, v_j)$ принимается, если значение ψ меняет свой порядок, но при этом, сохраняется $\psi \geq 1$. В этом случае канал (v_i, v_j) имеет запас по пропускной способности на величину $\psi \cdot (\overline{\lambda}_{\max} + \sigma)\,[Кбайт/с]$ и видеопоток может быть пропущен по данному каналу в полном объеме без задержек. В том случае, если ψ принимает значение $\psi < 1$, частота обновления таблиц маршрутизации возрастает и определяется вероятностными показателями видеопотоков WMV. Для этого введем функцию $\xi(r)$:

$$\xi(r) = F(r) - F(b),$$

вычисляемую на основании распределения интенсивностей видеопотоков WMV, которые согласно проведенным автором исследованиям [3] подчиняются закону Вейбулла:

$$F(x) = 1 - e^{-\left(\frac{x}{\lambda}\right)^k},$$

где: λ — параметр масштаба, который определяет степень растянутости кривой, $k, k > 0$ - параметр закона Вейбулла, определяемый на основе анализа потока WMV; $b = const$ — параметр сдвига, минимально возможное значение интенсивности $[Кбайт/с]$; r - интенсивность потока: $b \leq r \leq \overline{\lambda}_{\max} + \sigma_{\max}$, $[Кбайт/с]$.

Пусть $\xi(r)$ – вероятность появления видеопотока интенсивности меньше или равной r, тогда на вход протокола подается значение $p_{êð}$, $0 \leq p_{êð} < 1$. Исходя из этого, процесс обновления таблиц маршрутизации при $\psi < 1$ происходит только в том случае, если $\xi(c(v_i, v_j)) \leq p_{êð}$. Таким образом, если $\psi \to 0$, процесс обновления таблиц маршрутизации происходит при изменении ψ на любое значение $\Delta\psi$.

После принятия решения об изменении остаточных пропускных способностей в каналах $(v_i, v_j) \in E$ маршрутизатор v_i обновляет свою таблицу маршрутизации с учетом вновь полученных значений $c(v_i, v_j)$. Затем маршрутизатор v_i через рассылку сообщений FLOW_CHANGE, содержащих реквизиты $\left\langle P_{type}, GUID, L_{msg}, t_{msg}, c(v_i, v_j)\right\rangle$, информирует об этом свои соседние маршрутизаторы. Те, в свою очередь, получив данное сообщение, выполняют обновление своих таблиц, после чего отправляют это сообщение, оставляя его в неизменном виде, на все свои интерфейсы, кроме того, на который данное сообщение было получено. При этом временная метка данного сообщения хранится еще некоторое время в памяти маршрутизатора, но при повторном получении сообщения с той же меткой оно просто удаляется и обработке не подлежит.

Литература

1. Олифер В.Г., Олифер Н.А. Компьютерные сети. Принципы, технологии, протоколы: Учебник для вузов. - СПб.: Питер, 2004. – 864с.
2. Кринкин К.В. Создание алгоритмов маршрутизации в динамических компьютерных сетях с использованием неполных данных. Дис. канд. техн. наук: 05.13.11. - М.: РГБ, 2005. – 140 с.
3. Казанов В.Ю. Современные подходы к решению задачи маршрутизации в компьютерных сетях // «Программное и информационное обеспечение систем различного назначения на базе персональных ЭВМ»

Межвузовский сборник научных трудов. - М.: МГУПИ, 2007, вып. 10, с. 84 – 88.
4. Энг Т. Цифровое видео. Справочник. - М.: АСТ, 2006.
5. Ульянов М.В., Казанов В.Ю. Изучение видеопотоков WMV и MPEG4 в аспекте их стационарности // «Фундаментальные и прикладные проблемы приборостроения, информатики и экономики» Научные труды XI международной научно-практической конференции. - М.: МГУПИ, 2008, с. 187 – 192.
6. Плаксин В.А., Основы теории графов: Курс лекций. - Новочеркасск: ЮРГТУ (НПИ), 2002. – 95 с.
7. Головина Л.И. Линейная алгебра и некоторые ее приложения. - М.: Наука, 1985.
8. Кормен Т., Лейзерсон Ч. Алгоритмы: построение и анализ. - М.: МЦНМО: БИНОМ, 2004. – 960 с.

Василий Юрьевич Казанов. **А**спирант. Окончил Московский государственный университет приборостроения и информатики в 2006 году. Автор 3 печатных работ. Область научных интересов: анализ сложности алгоритмов. E-mail: kaz_an@rambler.ru.