Оглавление

Введение

Актуальность

Повсеместное внедрение компьютерных сетей, успехи в развитии оптоволоконных и беспроводных средств связи споровождаются непрерывной сменой сетевых технологий, направленной на повышение быстродействия и надёжности сетей. Однако создание опытного образца сети для оценки её эффективности не всегда является оправданным с точки зрения времени и трудоёмкости, поэтому разработка математических моделей является актуальной задачей.

Для непрерывного количественного и качественного роста компьютерных сетей необходимо развитие фундаментальной теории в этой области и создание инженерных методов анализа, направленных на сокращение сроков и повышение качества проектирования компьютерных сетей.

В качестве такой теории выступает теория систем и сетей массового обслуживания. Математические методы этой теории обеспечивают возможность решения многочисленных задач расчёта характеристик качества функционирования различных компонентов компьютерных сетей.

Цель

В данной работе рассматривается анализ критериев времени и надёжности доставки информации в информационно-вычислительных сетях (ИВС) большой размерности различных топологий с множественным методом доступа без коллизий, построенных на основе технологий семейства Ethernet.

Задачи

В задачи исследования входит:

1. Изучение методики разработки моделей сетей.
2. Разработка аналитических математических моделей ИВС.
3. Разработка программы для вычисления стационарных и интегральных вероятностных характеристик заданной ИВС.
4. Проведение модельного эксперимента.

Методы

Модельный эксперимент и математические модели фрагментов сетей основываются на математическом аппарате и методах теории систем и сетей массового обслуживания.

Значимость

Разработанная программа автоматизирует рутинную работу по вычислению стационарных и интегральных вероятностных характеристик. Она будет полезна при:

- предварительной оценке характеристик проектируемой ИВС
- оценке характеристик уже существующих ИВС
- изучении влияния изменений топологии и/или оборудования на характеристики ИВС

•

Глава 1. Теоретичские основы разработки математической модели ИВС

1. 1. Однородные экспоненциальные сети

Предметом изучения сетей массового обслуживания (СеМО) являются методы количественного анализа очередей при взаимодействии множества центров обслуживания и потоков сообщений.

СеМО представляет собой совокупность конечного числа M обслуживающих центров, в которой циркулируют сообщения, переходящие в соответствии с маршрутной матрицей (см. 1.1.2) из одного центра сети в другой. Центром обслуживания является система массвого обслуживания, состоящую из A $(1 \leqslant A \leqslant \infty)$ одинаковых приборов и буфера объёмом C $(0 \leqslant C \leqslant \infty)$. Если в момент поступления сообщения все обслуживающие приборы центра заняты, то сообщение занимает очередь в буфере и ожидает обслуживания [2, стр. 90].

В дальнейшем будем полагать, что объём буфера в центре обслуживания $C = \infty$, время обслуживания заявок распределено по экспоненциальному закону, а распределение входящего потока имеет распределение Пуассона.

СеМО с такими распределениями длительности обслуживания и входящего потока являются однородными экспоненциальными сетями или сетями Джексона [2, стр. 94]. Такая модель даёт верхнюю границу оценки (худший вариант) и стационарные вероятности состояний сети имеют муль-

типликативную форму.

В данной работе используются открытые сети Джексона, обрабатывающие F входящих потоков. В открытую сеть сообщения поступают из внешнего источника, могут покидать сеть после завершения обслуживания и интенсивность входного потока не зависит от состояния сети.

1. 1. 1. Пуассоновский поток

Предположение о том, что входящий поток является Пуассоновским, значительно облегчает математические выкладки при достаточной точности.

Пуассоновский поток имеет следующие свойства [12, стр. 12]:

1. Стационарность — вероятность появления k событий на любом промежутке времени зависит только от числа k и от длительности t промежутка.

2. Ординарность — вероятность наступления за элементарный промежуток времени более одного события мала по сравнению с вероятностью наступления за этот промежуток не более одного события и ей можно пренебречь.

3. Независимость — вероятность появления k на любом промежутке времени не зависит от того, появлялись или не появлялись события в моменты времени, предшествующие началу рассматриваемого промежутка.

1. 1. 2. Маршрутная матрица

Маршрутная матрица задаёт структуру соединений узлов сети (топологию) и вероятности переходов сообщения из одного центра сети, после завершения обслуживания в нём, в другой. Для открытой сети в качестве внешнего источника вводится новый центр с индексом 0. Таким образом маршрутная матрица имеет вид $P = \| P_{ij} \|$, где [7, стр. 17]:

$i, j = \overline{0, n}$, n - число узлов в сети,

P_{0j} - вероятность поступления сообщения в M_j узел сети из внешнего источника,
P_{i0} - вероятность покидания сообщением сети после окончания обработки в M_i узле,
P_{ij} - вероятность перехода сообщения в узел M_j после обработки в узле M_i.
$P_{00} = 0$.
В маршрутной матрице должно выполняться равенство $\sum_{j=0}^{n} P_{ij} = 1, i = \overline{1,n}$.
Т.е. событие, состоящие в том, что сообщение после обработки в узле сети перейдёт в другой узел или покинет сеть — достоверное.

Для сети, обрабатывающей F входящих потоков, необходимо задать F маршрутных матриц P^m, $m = \overline{1,F}$ для каждого входного потока.

1. 2. Методика разработки математической модели ИВС

Одним из самых распространённых методов для разработки аналитической математической модели ИВС является приближённая декомпозиционная модель сети массового обслуживания, основанная на составлении уравнений баланса средних для класса мультипликативных сетей. Эта модель допускает простую декомпозицию всей сети на отдельные элементы и обратную операцию - композицию. Такой подход позволяет проводить анализ каждого фрагмента сети независимо, а затем объединять эти фрагменты, получая обобщённые характеристики.

Первый этап методики состоит в декомпозиции сети на отдельные фрагменты. В зависимости от уровня детализации выделяют 4 уровня декомпозиции:

1. Состоит из набора функциональных элементов (терминалов, моноканалов, канальных станций), каждый из которых может быть представлен в виде отдельной СМО. Самый подробный уровень декомпозиции.

2. Учитывает особенности взаимодействия отдельных элементов 1-го уров-

ня в пределах всей ИВС.

3. Учитывает взаимодействие нескольких ИВС элементарных топологий (физическая шина, физическое кольцо), связанных между собой в единую ИВС простой топологии (дерево, звезда, и т.д.).

4. Учитывает взаимодействие любых ИВС 2-го и 3-го уровня, связанных в единую ИВС произвольной топологии.

Декомпозиция позволяет преодолеть трудности анализа ИВС большой размерности за счёт разделения ИВС на иерархический набор более простых моделей [6, стр. 11]

Второй этап методики независимо от уровня декомпозиции заключается в разработке отдельных математических моделей всех составляющих на всех уровнях декомпозиции и состоит из следующих подэтапов:

1. Составление уравнений баланса интенсивностей потоков.

2. Вычислеие коэффициентов передачи из уравнений баланса.

3. Вычисление стационарных вероятностно-временных характеристик (ВВХ) для каждого отдельного элемента СеМО.

4. Вычисиление интегральных ВВХ при взаимодействии двух любых абонентов сети.

Исходными параметрами модели являются интенсивности обслуживающих узлов сети μ_i^m, интенсивности поступления сообщений из внешнего источника λ_i^m и маршрутная матрица P^m для каждого входного потока $m = \overline{1, F}$.

1. 2. 1. Уравнения баланса

Уравнения баланса позволяют найти общие интенсивности потоков $\lambda_i'^m$ сообщений в стационарном режиме открытой СеМО (стационарным режимом называется состояние сети, при $t \to \infty$ и $\rho \leqslant 1$ (см. 1.2.2)).

$$\lambda_i^{'m} = e_i^m \lambda_0^m$$

e_i^m - коэффициенты передачи, получаемые при решении уравнений баланса, $\lambda_0^m = \sum_{i=1}^{n} \lambda_i^m$ - суммарная интенсивность всех внешних потоков типа m.

Общая интенсивность птоков складывается из интенсивностей поступления сообщений в M_i узел из внешнего источника $P_{0i}^m \lambda_0^m$, $P_{0i}^m = \frac{\lambda_j^m}{\lambda_0^m}$ и интенсивностей поступления сообщений от других узлов $e_j^m P_{ji}^m \lambda_0^m$ [7, стр. 17].

$$e_i^m \lambda_0^m = P_{0i}^m \lambda_0^m + \sum_{j=1}^{n} e_j^m P_{ji}^m \lambda_0^m, i = \overline{1,n}, m = \overline{1,F}$$

Видно, что можно сократить обе части уравнения на λ_0^m.

$$e_i^m = P_{0i}^m + \sum_{j=1}^{n} e_j^m P_{ji}^m, i = \overline{1,n}, m = \overline{1,F}$$

$$\Updownarrow$$

$$\begin{cases} e_1^m = P_{01}^m + e_1^m P_{11}^m + \cdots + e_n^m P_{n1}^m \\ \vdots \\ e_n^m = P_{0n}^m + e_1^m P_{1n}^m + \cdots + e_n^m P_{nn}^m \end{cases}$$

После решения этой системы уравнений получаем e_i^m для каждого узла, что позволяет рассчитать $\lambda_i^{'m}$.

1. 2. 2. Коэффициент загрузки

Коэффициент загрузки для узла M_i вычисляется по формуле

$$\rho_i = \sum_{m=0}^{F} \rho_i^m, \ \rho_i^m = \frac{\lambda_i^{'m}}{\mu_i^m}, \ i = \overline{1,n}$$

ρ_i^m- коэффициент использования узла, характеризующий соотношение интенсивности входящего потока к интенивности обработки [4, стр. 34].

Для существования стационарного распределения числа сообщений в системе необходимо выполнение условия

$$0 \leqslant \rho_i, \rho_i^m \leqslant 1,\ i = \overline{1,n}$$

что согласуется с интуитивными соображениями: для того, чтобы в системе не накапливалась бесконечная очередь, необходимо, чтобы в среднем сообщения в системе обслуживались быстрее, чем они туда поступают [2, стр. 35].

1. 2. 3. Стационарные вероятностно-временные характеристики

Для каждого узла M_i сети определяются четыре вероятностно-временные характеристики [7, стр. 19]:

1. Средняя длительность ожидания обслуживания.

$$W_i = \frac{\frac{1}{2}\sum\limits_{m=0}^{F}\frac{\rho_i^m(1+{V_i^m}^2)}{\mu_i^m}}{1-\rho_i} = \left| \begin{array}{l} V_i^m = 1 \\ \text{для распределения Пуассона} \end{array} \right| = \frac{\sum\limits_{m=0}^{F}\frac{\rho_i^m}{\mu_i^m}}{1-\rho_i}$$

   V_i^m - коэффициент вариации времени обработки.

2. Средняя длительность пребывания сообщения в узле для потока m.

$$U_i^m = W_i + \frac{1}{\mu_i^m}$$

   $\frac{1}{\mu_i^m}$ - время обработки сообщения.

3. Средняя длина очереди сообщений в узле для потока m.

$$L_i^m = {\lambda_i'}^m W_i$$

4. Среднее число сообщений в узле для потока m, хакатеризующее количество сообщений, находящихся в очереди на обработку, и количество

сообщений, обрабатывающихся в узле.

$$N_i^m = \lambda_i^{'m} U_i^m$$

Эти формулы справедливы для многих моделей СМО и называются формулами Литтла [2, стр. 37].

1. 2. 4. Интегральные вероятностно-временные характеристики

Для определения интегральных ВВХ используются стационарные ВВХ, полученные для каждого узла сети, и анализ маршрутов движения сообщений между двумя абнентами A_i и A_j, $i \neq j$.

Любой маршрут между двумя любыми абонентами принадлежит к одному из трёх типов:

1. Последовательная обработка сообщений на конечном числе элементов сети.

Рис. 1.1. Последовательная обработка

Маршрут состоит из последовательно соединённых узлов сети (рис. 1.1). Интегральные характеристики маршрута вычисляются как сумма соответсвующих характеристик узлов, входящих в маршрут.
Среднее время ожидания обслуживания в маршруте.

$$W = \sum_{i=1}^{n} W_i$$

Среднее время пребывания требования в маршруте для потока m.

$$U^m = \sum_{i=1}^{n} U_i^m$$

Средняя длина очереди требований в маршруте для потока m.

$$L^m = \sum_{i=1}^{n} L_i^m$$

Среднее число требований в маршруте для потока m.

$$N^m = \sum_{i=1}^{n} N_i^m$$

2. Параллельные варианты обработки с определёнными значениями вероятности выбора рассматриваемого варианта.

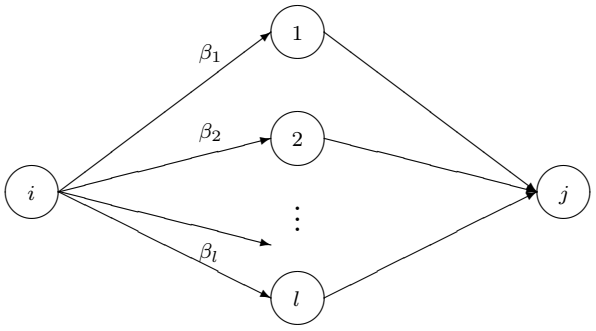

Рис. 1.2. Параллельные варианты

Маршрут зависит от одного из выбранного параллельного варианта (рис. 1.2). Заданы параллельные варианты обработки с вероятностями выбора (см. 1.2.5) β_k, $k = \overline{1,l}$, l - количество альтернатив и должно выполняться нормирующее условие $\sum\limits_{k=1}^{l} \beta_k = 1$.

Интегральные ВВХ определяются следующим образом.

Среднее время ожидания обслуживания в маршруте.

$$W = W_i + \sum_{k=1}^{l} \beta_k W_k + W_j$$

Среднее время пребывания требования в маршруте для потока m.

$$U^m = U_i^m + \sum_{k=1}^{l} \beta_k U_k^m + U_j^m$$

Средняя длина очереди требований в маршруте для потока m.

$$L^m = L_i^m + \sum_{k=1}^{l} \beta_k L_k^m + L_j^m$$

Среднее число требований в маршруте для потока m.

$$N^m = N_i^m + \sum_{k=1}^{l} \beta_k N_k^m + N_j^m$$

3. Комбинация первых двух типов.

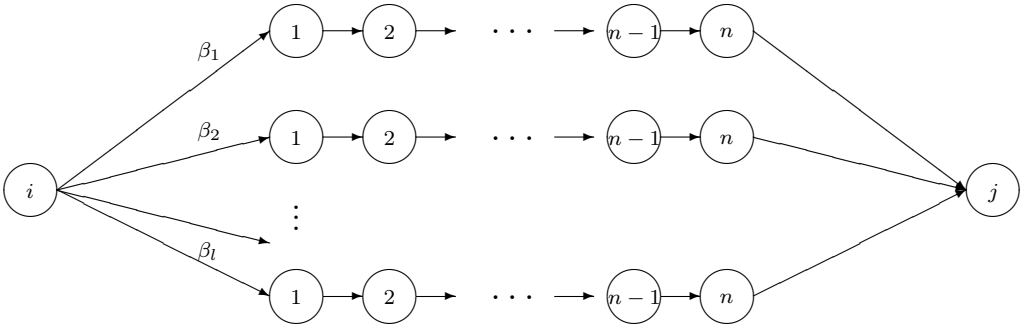

Рис. 1.3. Комбинированный тип обработки

Маршрут зависит от одного из выбранных параллельных маршрутов (рис. 1.3).

Комбинированный тип обработки включает в себя рассмотрение сложных альтернатив, представляющих собой конечную комбинацию двух предыдущих топологических типов.

Интегральные ВВХ в этом случае определяются как в предыдущем случае, но вместо альтернативных вариантов рассматриваются альтернативные маршруты.

Среднее время ожидания обслуживания в маршруте.

$$W = W_i + \sum_{k=1}^{l} \left(\beta_k \sum_{x=1}^{n} W_x \right) + W_j$$

Среднее время пребывания требования в маршруте для потока m.

$$U^m = U_i^m + \sum_{k=1}^{l} \left(\beta_k \sum_{x=1}^{n} U_x^m \right) + U_j^m$$

Средняя длина очереди требований в маршруте для потока m.

$$L^m = L_i^m + \sum_{k=1}^{l} \left(\beta_k \sum_{x=1}^{n} L_x^m \right) + L_j^m$$

Среднее число требований в маршруте для потока m.

$$N^m = N_i^m + \sum_{k=1}^{l} \left(\beta_k \sum_{x=1}^{n} N_x^m \right) + N_j^m$$

1. 2. 5. Вероятность выбора маршрута

Для вычисления интегральных ВВХ необходимо определить вероятности выбора альтернативных маршрутов (см. 1.2.4).

Для этого нужно учитывать вероятности перехода между узлами, заданные маршрутной матрицей P^m (см. 1.1.2), и нормирующее условие $\sum_{k=1}^{l} \beta_k = 1$, указывающее, один из маршрутов будет обязательно выбран.

Вероятность выбора маршрута определяется отношением произведения вероятностей перехода требований из узла $M_{R_j^i}$ в узел $M_{R_{j+1}^i}$ i - го маршрута к сумме произведений вероятностей переходов требований всех альтернативных маршрутов (нормировочной величине).

$$\beta_i = \frac{\prod_j P_{R_j^i, R_{j+1}^i}}{S}$$

$S = \sum_i \left(\prod_j P_{R_j^i, R_{j+1}^i} \right)$ - нормировочная величина.

R^i - совокупность узлов, составляющих альтернативный маршрут i.

$P_{R_j^i, R_{j+1}^i}$ - вероятность перехода между узлами, задаваемая маршрутной матрицей.

1. 2. 6. Плотность распределения количества сообщений в маршруте

Плотность распределения количества сообщений для произвольного маршрута определяется следующим способом:

$$g_i(t) = \sum_{i=1}^{n} H_i(\mu_i - \lambda_i^{'}) e^{-(\mu_i - \lambda_i^{'})t}$$

$$H_i = \prod_{\substack{j=1 \\ j \neq i}}^{n} \frac{\mu_j - \lambda_j^{'}}{\mu_j - \lambda_j^{'} - \mu_i - \lambda_i^{'}}$$

где n - количество узлов в маршруте.

1. 3. Определение интенсивностей обслуживающих приборов, работающих на основе технологий семейства Ethernet

Для определения интенсивности обслуживания μ, которая характеризует максимальное количество кадров в секунду, обрабатываемых обслуживающим прибором, рассмотрим кадр формата Ethernet Version 2, т.к. он наиболее распространён.

Стандарт Ethernet определяет длину служебных полей кадра и преамбулу по 18 и 8 байт соответственно. От сюда следует что длина кадра в битах равна $8 * (X + 26)$, X - длина поля данных. Поле данных может иметь длину от 46 до 1500 байт. Таким образом минимальная длина кадра

равна $46 + 26 = 72$ байта, а максимальная 1526 байт.

Время передачи битов кадра определяется битовыми интервалами $bt = \frac{1}{S}$, S - битовая скорость. Для передачи кадра длиной X потребуется время равное $8 * (X + 26) * bt$. Прибавив межкадровый интервал $IFG = 96 * bt$ получим период следования кадров с длиной поля данных X.

$$T = 8 * (X + 26) * bt + IFG$$

Итенсивность обслуживания μ, которая характеризует максимальное количество кадров в секунду, обрабатываемых обслуживающим прибором, есть величина обратно пропорциональная периоду T.

$$\mu = \frac{1}{T} = \frac{1}{8 * (X + 26) * bt + IFG}$$

Проведя не сложные расчёты можно получить интенсивности обслуживания для пакетов различной длины с учётом выбранной технологии передачи данных (табл. 1.1).

Технология Ethernet	Битовая скорость	Длина кадра (байт)	Интенсивность μ (кадр/мс)
Fast Ethernet	100 Мбит/с	72	148.800
		1526	8.127
Gigabit Ethernet	1 Гбит/с	72	1488.095
		1526	81.274
10G Ethernet	10 Гбит/с	72	14880.952
		1526	812.744
40G Ethernet	40 Гбит/с	72	59523.800
		1526	3250.975
100G Ethernet	100 Гбит/с	72	148809.524
		1526	8127.438

Таблица 1.1. Интенсивности обслуживания для пакетов различной длины с учётом выбранной технологии передачи данных

1. 4. Структурная надёжность сетей

Надёжность какого-либо объектана – свойство, заключающееся в способности выполнять поставленные задачи в определённых условиях эксплуатации. Состояние объекта, при котором он способен выполнять заданные функции, сохраняя значения основных параметров в пределах, установленных нормативно-технической документацией, называют работоспособностью, а состояние, в котором объект удовлетворяет указанным требованиям, – его исправностью. Событие, заключающееся в нарушении работоспособности объекта, называют отказом.

Для информационых сетей, являющихся сложными системами, состоящими из элементов разнородных по своим свойствам, показателям надёжности, назначению, дате изготовления, сроку ввода в эксплуатацию и т.п., выделяют два основных аспекта надёжности:

1. Аппаратурный. Под ним понимают проблему надёжности отдельных элементов, входящих в узлы и линии сети.

2. Структурный. Связан с возможностью существования в сети путей доставки информации.

В данной работе рассматривается структурная надёжность, определяемая количеством резервных маршрутов между двумя произвольными вершинами сети. Более строгие модели надёжности сетей будут рассмотрены в магистерской диссертации.

•

Глава 2. Программная реализация

2. 1. Постановка задачи

Программа должна удовлетворять следующим требованиям:

1. Вычислять стационарные и интегральные ВВХ для заданной пользователем модели ИВС.
2. Проверять корректность введённой пользователем модели ИВС.
3. При невозможности вычисления каких-либо характеристик сообщать об этом пользователю.

Программа будет выполнена в виде консольного приложения, т.к. этого достаточно для поставленной задачи и позволяет сконцентрироваться на реализации и достижении поставленных требований. Также это позволит программе без проблем запускаться на различных операционны системах.

2. 2. Программная платформа

Выбрана программная платформа Mono — кроссплатформенная реализация Microsoft .Net Framework с открытым исходным кодом. Mono поддерживает Windows, Linux, BSD, Mac OS X и другие операционные системы и множетсво процессорных платформ, что позволяет писать переносимые приложения на C#. Среда разработки — MonoDevelop.

2. 3. Описание модели ИВС

Модель ИВС одонозначно задаётся интенсивностями обслуживания μ_i^m, интенсивностями поступления сообщений λ_i^m и маршрутной матрицей P^m для каждого входного потока $m = \overline{1, F}$ (см. 1.2). Таким образом надо решить как пользователь будет передавать программе описание модели ИВС.

Для этой цели используется XML файл, в котором задаются параметры модели. XML имеет простой синтаксис, удобный как для составления и чтения файлов человеком, так и для машинной обработки и генерации.

```
1   <NetworkConfiguration Name="Linear topology">
2       <RoutingMatrix>
3           <Row>0; -;  -;  -;  -</Row>
4           <Row>0.25;  0;  0.75;   0;  0</Row>
5           <Row>0.25;  0.375;  0;  0.375;  0</Row>
6           <Row>0.25;  0;  0.375;  0;  0.375</Row>
7           <Row>0.25;  0;  0;  0.75;   0</Row>
8       </RoutingMatrix>
9
10      <Nodes Count="4">
11          <Lambda>17; 14; 13; 16</Lambda>
12
13          <Mu>
14              <Ethernet Type="Gigabit" FrameLength="Max"/>
15          </Mu>
16      </Nodes>
17  </NetworkConfiguration>
```

Рис. 2.1. Описание ИВС линейной топологии из 4-х узлов и одним потоком

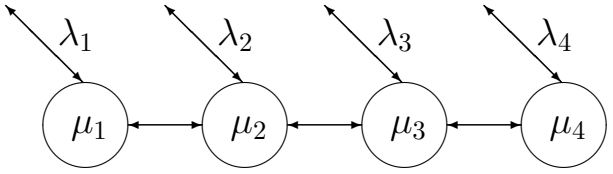

Рис. 2.2. Сеть из 4-х узлов с линейной топологией

На рис. 2.1 показан XML файл, задайющий модель ИВС, состоящей из 4-х узлов с линейной топологией и единственным потоком (рис. 2.2).

Интенсивность обслуживания μ задаётся для пакетов максимальной длины технологии Gigabit Ethernet (см. 1.3), интенсивность поступления сообщений λ задаётся в явном виде для каждого узла сети (пакеты/мс).

$$\mu = (81.274, 81.274, 81.274, 81.274),\ \lambda = (17, 14, 13, 16)$$

Со 2-й по 8-ю строки задаётся маршрутная матрица.

$$P = \begin{pmatrix} 0 & - & - & - & - \\ 0.25 & 0 & 0.75 & 0 & 0 \\ 0.25 & 0.375 & 0 & 0.375 & 0 \\ 0.25 & 0 & 0.375 & 0 & 0.375 \\ 0.25 & 0 & 0 & 0.75 & 0 \end{pmatrix}$$

Прочерки в строках задаваемой матрицы указывают программе, что надо вычислить соответствующие вероятности. В данном случае будут вычисляться вероятности поступления сообщений из внешнего источника.

За более подробной информацией о структуре файла обращайтесь к Приложению А.

Парсинг (разбор) XML файла осуществляется с помощью стандартных средств языка C# из пространств имён System.Xml и System.Xml.Linq.

2. 4. Аргументы командной строки

При запуске программе передаются несколько обязательных аргументов и один опциональный. Обязательные аргументы `netconfig`, `startnode` и `targetnode` задают путь к конфигурационному файлу, начальный и конечный узлы (для вычисления интегральных ВВХ) соответственно.

Необязательный аргумент `log` обозначает необходмость создать лог, в который будет выведена вся информация, если вычисления закончились успешно.

2. 5. Алгоритм работы программы

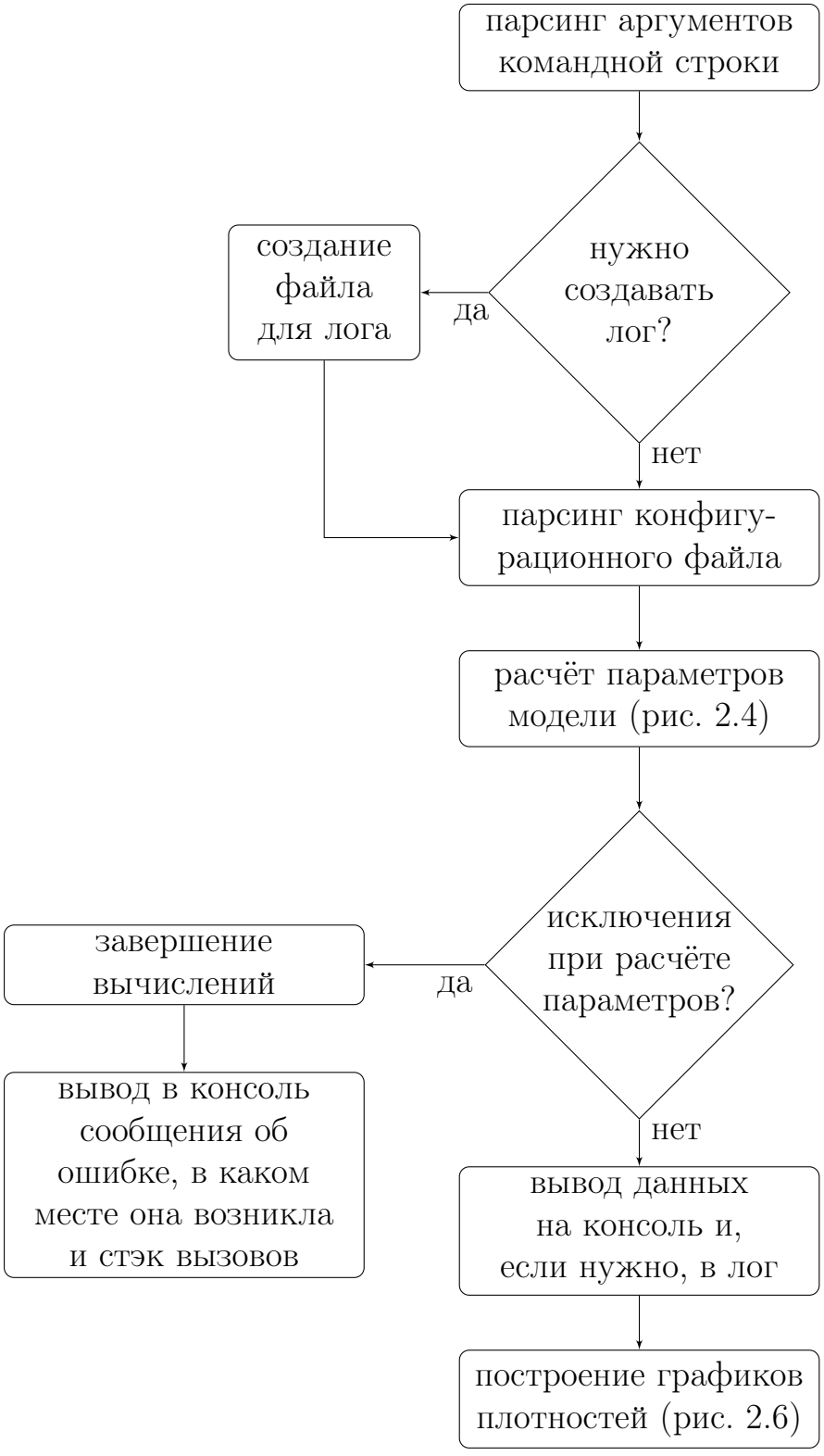

Рис. 2.3. Общая блок-схема программной реализации

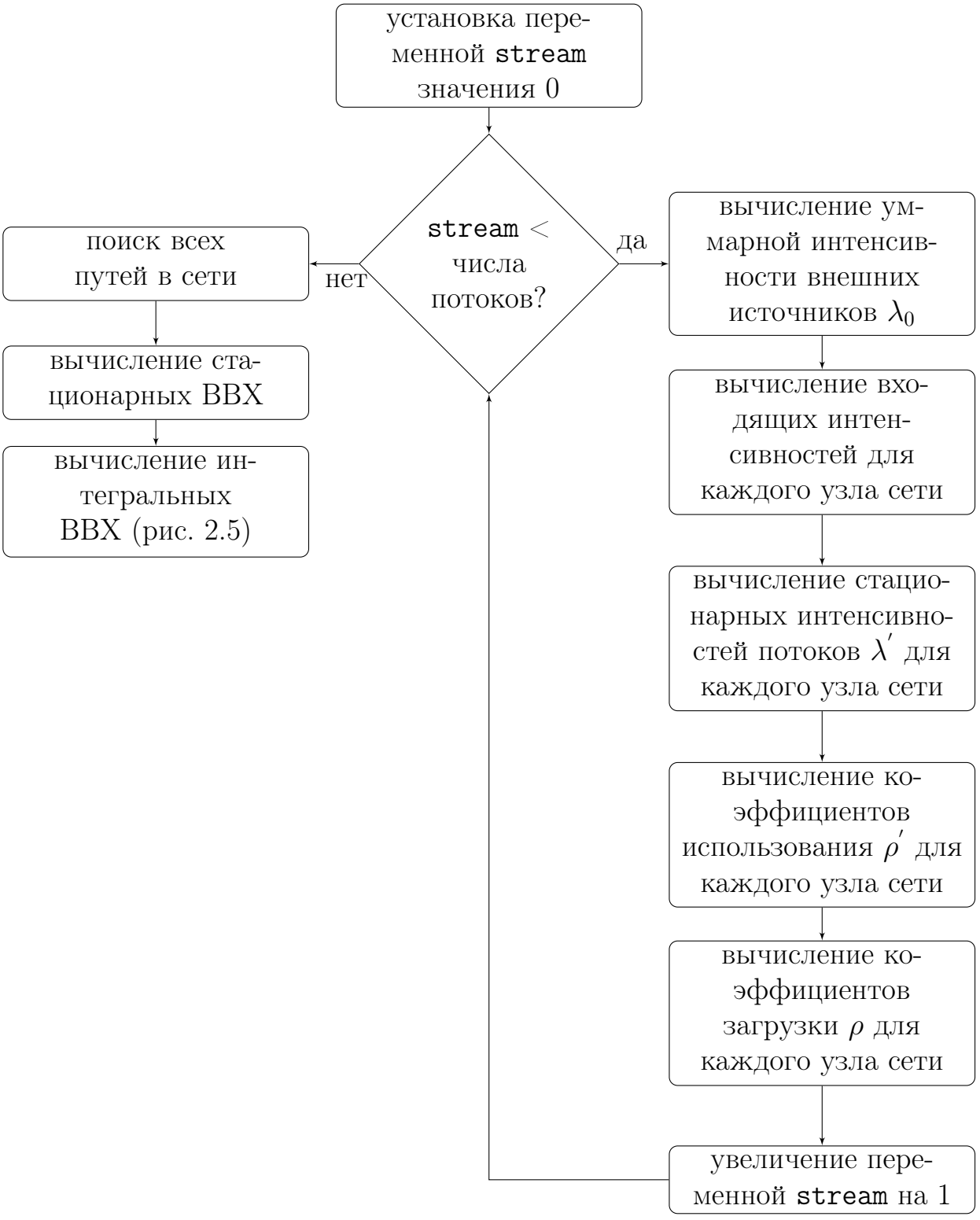

Рис. 2.4. Рассчёт параметров модели

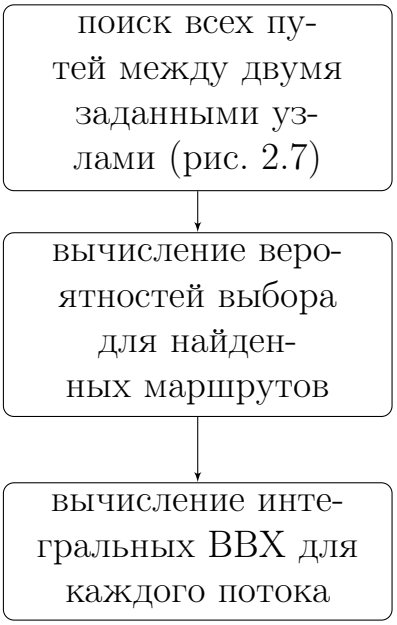

Рис. 2.5. Рассчёт интегральных ВВХ

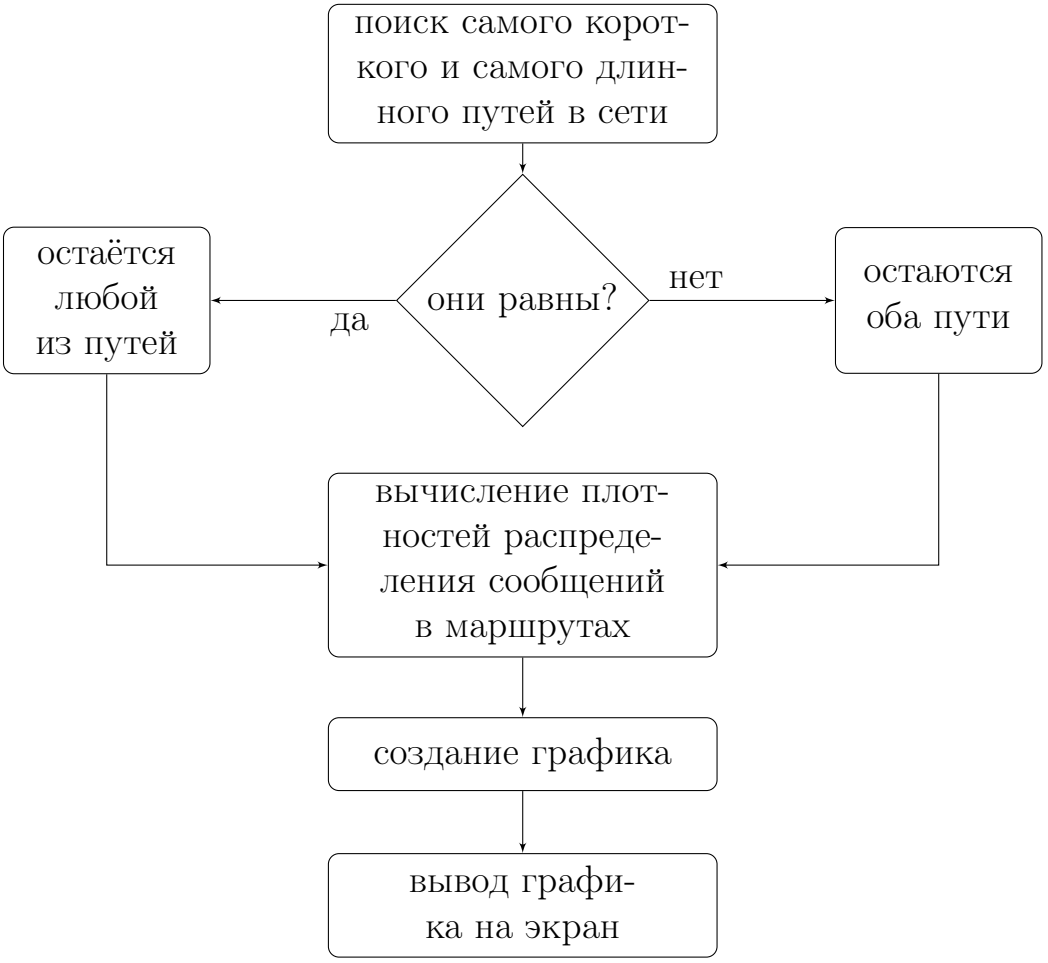

Рис. 2.6. Построение графиков плотностей

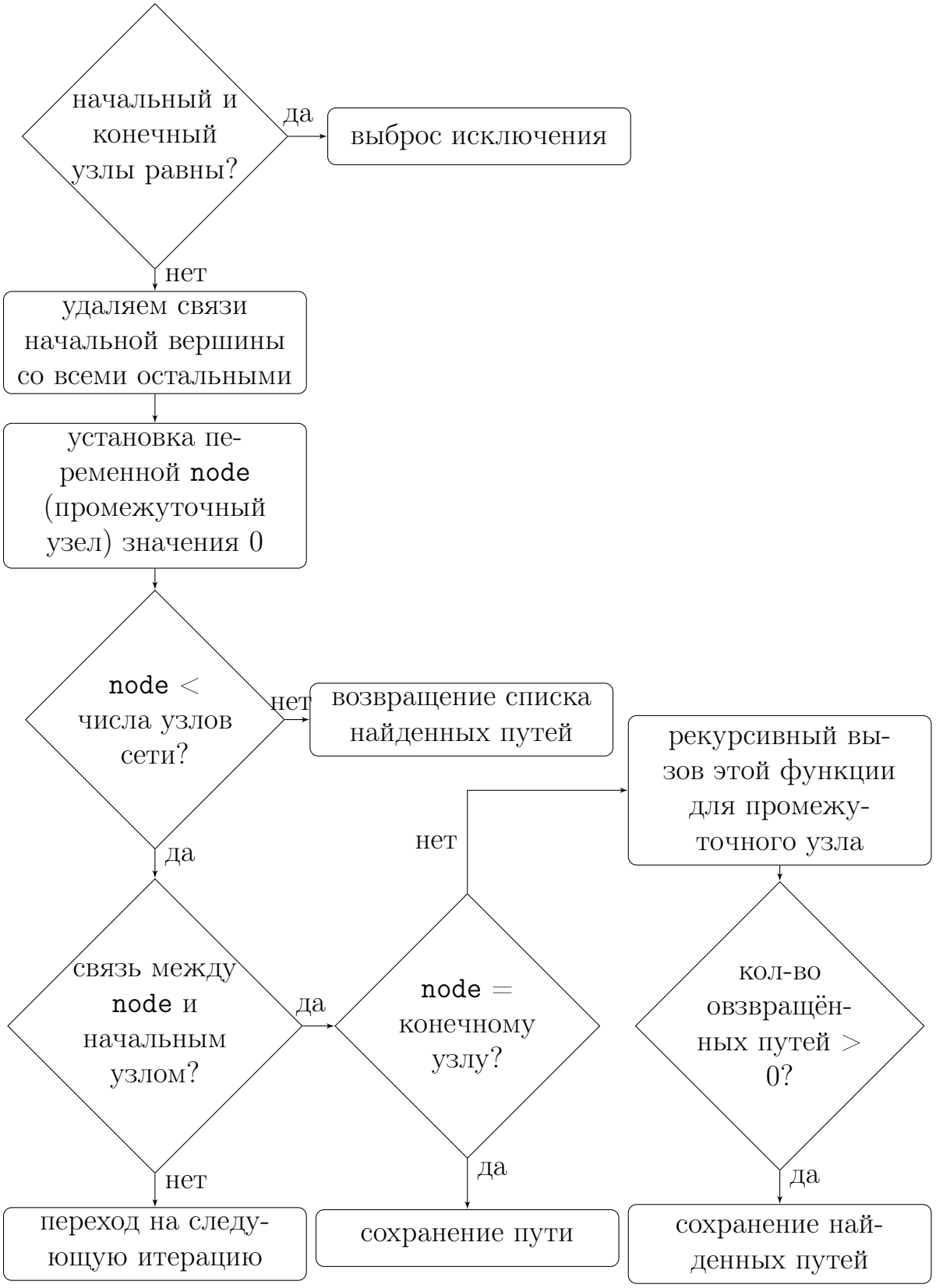

Рис. 2.7. Поиск всех путей между двумя узлами сети (одна итерация)

2. 6. Основные моменты реализации

2. 6. 1. Функция расчёта параметров модели

Функция `ComputeParameters()` (рис. 2.8) основная функция, которая после парсинга конфигурационного файла высчитывает следующие параметры:

- суммарная интенсивность внешних потоков λ_0^m (рис. 2.8, строка 4)
- вероятности поступления сообщений из внешнего источника $P_{0,j}^m$ (рис. 2.8, строка 11)
- коэффициенты передачи e_i^m (рис. 2.8, строка 14)
- общие интенсивности потоков $\lambda_i'^m$ (рис. 2.8, строка 17)
- коэффициенты использования узлов ρ_i^m и коэффициенты загрузки ρ_i (рис. 2.8, строка 20)
- все пути в сети (рис. 2.8, строка 33)

Эти параметры дают достаточно инормации для дальнейших вычислений стационарных и интергральных ВВХ и плотностей распределения сообщений в маршрутах.

2. 6. 2. Составление и решение уравнений баланса

Расширенная матрица системы уравнений баланса получается из матрицы маршрутизации следующим способом:

1. Транспонируем матрицу маршрутизации.
2. Вычёркиваем нулевую строчку.
3. Умножаем нулевой столбец на -1.
4. Переносим нулевой столбец в конец.
5. Проставляем -1 на главной диагонали.

Получаем расширенную матрицу для системы уравнений следующего вида и решаем её с помощью матода Гаусса:

$$\begin{cases} -e_1^m + e_1^m P_{11}^m + \cdots + e_n^m P_{n1}^m = -P_{01}^m \\ \vdots \\ -e_n^m + e_1^m P_{1n}^m + \cdots + e_n^m P_{nn}^m = -P_{0n}^m \end{cases}$$

В программной реализации (рис. 2.8, строка 15) вызывается функция `GetExtendedMatrix(int streamIndex)` (рис. 2.9), которая преобразует матрицу маршрутизации заданного потока.

2. 6. 3. Вычисление стационарных ВВХ

2. 6. 4. Вычисление интегральных ВВХ

•

```
1  void ComputeParameters()
2  {
3    // вычисление суммарной интенсивности внешних потоков
4    Lambda.ForEach(
5          lambdas => Lambda0.AddElement(lambdas.Sum()));
6    RoTotal = new Vector<double>(NodesCount);
7
8    for (int stream = 0; stream < StreamsCount; stream++)
9    {
10         // вычисление входных вероятностей
11         InputIntensity.Add(Lambda[stream].Clone()
12           .InsertElement(0, 0).Divide(Lambda0[stream]));
13         // вычисление коэфициентов передачи
14         E.Add(
15           GaussMethod.Solve(GetExtendedMatrix(stream)));
16         // вычисление общих интенсивностей потоков
17         LambdaBar.Add(
18           E[stream].Multiply(Lambda0[stream]));
19         // вычисление коэффициентов использования узлов
20         RoBar.Add(
21           LambdaBar[stream].DivideElementWise(Mu[stream]));
22         RoTotal = RoTotal.AddElementWise(RoBar.Last());
23
24         if(RoBar[stream].Any(roBar => roBar > 1))
25           throw new ArgumentOutOfRangeException(
26           string.Format("RoBar, stream {0}", stream),
27           "Some Ro' is greater than zero.");
28   }
29   if (RoTotal.Any(roTotal => roTotal > 1))
30         throw new ArgumentOutOfRangeException("RoTotal",
31         "Some RoTotal is greater than zero.");
32
33   AllPaths =
34         GraphHelper.GetAllPaths(
35         RoutingMatrix.Clone().RemoveColumn(0))
36         .OrderBy(path => path.Length).ToList();
37         Paths = AllPaths.Where(
38         path => path.First() == StartNode &&
39         path.Last() == TargetNode).ToList();
40 }
```

Рис. 2.8. Функция расчёта параметров модели

```
1   Matrix<double> GetExtendedMatrix(int streamIndex)
2   {
3     var matrix = GetRoutingMatrix(streamIndex);
4     matrix.Transpose();
5     matrix.RemoveRow(0);
6     matrix.AddColumn(matrix.GetColumn(0).Negate());
7     matrix.RemoveColumn(0);
8     matrix.ReplaceDiagonalElements(-1);
9
10    return matrix;
11  }
```

Рис. 2.9. Функция преобразования мартрицы маршрутизации в расширенную матрицу системы уравнений баланса

```
1   void ComputeStationaryPTC()
2   {
3     for (int i = 0; i < NodesCount; i++)
4     {
5           double sum = 0.0;
6           for (int stream = 0; stream < StreamsCount; stream++)
7           sum += RoBar[stream][i] / Mu[stream][i];
8
9           Ws.AddElement(sum / (1 - RoTotal[i]));
10    }
11
12    for (int stream = 0; stream < StreamsCount; stream++)
13    {
14          Us.Add( Ws.AddElementWise(Mu[stream].Pow(-1)) );
15          Ls.Add(
16            LambdaBar[stream].MultiplyElementWise(Ws) );
17          Ns.Add(
18            LambdaBar[stream].MultiplyElementWise(Us[stream]));
19    }
20  }
```

Рис. 2.10. Функция вычисления стационарных ВВХ

```
1  void ComputeIntegratedPTC()
2  {
3    ComputeTransitionProbabilities();
4
5    Wi = ComputeIntegralChar(Ws);
6    for (int stream = 0; stream < StreamsCount; stream++)
7    {
8          Ui.Add(ComputeIntegralChar(Us[stream]));
9          Li.Add(ComputeIntegralChar(Ls[stream]));
10         Ni.Add(ComputeIntegralChar(Ns[stream]));
11   }
12 }
```

Рис. 2.11. Функция вычисления интегральных ВВХ

```
1  double ComputeIntegralChar(Vector<double> staticChar)
2  {
3    var integralChar =
4          staticChar[StartNode] + staticChar[TargetNode];
5
6    for (int i = 0; i < Paths.Count(); i++)
7    {
8          var temp = 0.0;
9          for (int j=1; j < Paths.ElementAt(i).Count()-1; j++)
10           temp += staticChar[j];
11
12         integralChar += TransitionProbabilities[i] * temp;
13   }
14
15   return integralChar;
16 }
```

Рис. 2.12

```
1  void ComputeTransitionProbabilities()
2    {
3          var matrix = RoutingMatrix.Clone().RemoveColumn(0);
4
5          var pathsProbabilities =
6          Paths.Select(
7          path => path.Where(
8          (item, index) => index < path.Length - 1)
9          .Select((item, index) => new {item, index})
10         .Aggregate(1.0, (accumulate, anon) => accumulate *=
11         matrix[anon.item, path[anon.index + 1]]));
12
13         var s = pathsProbabilities.Sum();
14
15         TransitionProbabilities = pathsProbabilities
16         .Select(prob => prob / s).ToList();
17 }
```

Рис. 2.13. Функция вычисления вероятностей выбора маршрутов

```
1  List<Vector<int>> GetPathsBetween(
2    Matrix<double> matrix, int startNode, int targetNode)
3  {
4    if (startNode == targetNode)
5    throw new ArgumentException(
6    "stratNode and targetNode must be different.");
7
8    var paths = new List<Vector<int>>();
9
10   // exclude routing loops
11   matrix.ForEachRow(row => row[startNode] = 0);
12
13   for (int intermediateNode = 0;
14   intermediateNode < matrix.RowsCount; intermediateNode++)
15   {
16         if (matrix[startNode, intermediateNode] <= 0)
17           continue;
18
19         if (intermediateNode == targetNode)
20           paths.Add(
21           new Vector<int>(new[] { startNode, targetNode }));
22         else
23         {
24           List<Vector<int>> localPaths = GetPathsBetween(
25           matrix.Clone(), intermediateNode, targetNode);
26           if (localPaths.Any())
27                 localPaths.ForEach(path =>
28                 {
29                   path.InsertElement(0, startNode);
30                   paths.Add(path);
31                 });
32           }
33     }
34
35   return paths;
36 }
```

Рис. 2.14. Функция поиска всех путей между двумя вершинами

Глава 3. Модельный эксперимент

Заключение

Приложение А

Структура XML файла для описания модели ИВС

Список литературы

1. Барашин Г.П., Харкевич А.Д., Шнепс М.А. Массовое обслуживание в телефонии. — Москва: Издательство «Наука», 1968. — 246 с.

2. Вишневский В.М. Теоретические основы проектирования компьютерных сетей: монография. — Москва: Техносфера, 2003. — 512 с.

3. Клейнрок Л. Коммуникационные сети (стохастические потоки и задержки сообщений). — Москва: Главная редакция физико-математической литературы изд-ва «Наука», 1970. — 256 с.

4. Клейнрок Л. Теория массового обслуживания. — Москва: Машиностроение, 1979. — 432 с.

5. Клейнрок Л. Вычислительные системы с очередями. — Москва: Издательство «Мир», 1979. — 600 с.

6. Климанов В.П. Методология анализа вероятностно-временных характеристик локальных вычислительных сетей составных топологий на основе аналитического моделирования: автореферат диссертации на соискание учёной степени доктора технических наук. — Москва: Московский энергетический институт, 1993. — 40 с.

7. Климанов В.П., Руделёв Р.А. Моделирование информационных систем. Математические модели для разработки информационных систем: методика и решения: учебное пособие. — Москва: ФГБОУ ВПО МГТУ «СТАНКИН», 2014. — 45 с.

8. Кокс Д.Р., Смит У.Л. Теория очередей. — Москва: Издательство «Мир», 1966. — 218 с.

9. Олифер В.Г., Олифер Н.А. Компьютерные сети. Принципы, технологии, протоколы: учебник для вузов. — 4-е изд. — Санкт-Петербург: Питер, 2010. — 944 с.

10. Писарев В.Н. Применение теории массового обслуживания в задачах инженерно-авиационного обеспечения. — Типография ВВИА имени проф. Н.Е. Жукова, 1965. — 45 с.

11. Таненбаум Э., Уезеролл Д. Компьютерные сети. — 5-е изд. — Санкт-Петербург: Питер, 2012. — 960 с.

12. Хинчин А.Я. Работы по математической теории массового обслуживания — Москва: Физматгиз, 1963. — 236 с.