

УДК 681.324

А.В. Скатков, профессор, д-р техн. наук;**Д.Ю. Воронин, магистр; Д.Н. Данильчук, магистр***Севастопольский национальный технический университет**ул. Университетская 33, СевНТУ, Севастополь, Украина, 99053**E-mail: kvf@sevgtu.sebastopol.ua***ИНСТРУМЕНТАЛЬНОЕ СРЕДСТВО РЕШЕНИЯ ОПТИМИЗАЦИОННЫХ ЗАДАЧ НА ОСНОВЕ СОПРЯЖЕНИЯ РАЗНОРОДНЫХ ПРОГРАММНЫХ СРЕД**

Рассматривается инструментальное средство решения оптимизационных задач на основе сопряжения разнородных программных сред. Производится анализ существующих программных систем для решения оптимизационных задач. Доказывается актуальность предлагаемого подхода. Описываются основные аспекты предлагаемого инструментального средства. Приводится пример решения оптимизационной задачи при помощи предлагаемой системы.

В самых различных областях человеческой деятельности широко применяются методы оптимизации [1]. Особую роль оптимизация играет при проектировании и анализе сложных систем. Например, в [2] приводится решение задачи по оптимизации работы узлов коммутации в ортогональной сети.

Целью данной статьи является рассмотрение и решение ряда вопросов, связанных с сопряжением разнородных программных сред. Описываемый подход предлагается использовать для улучшения процесса имитационного моделирования при решении оптимизационных задач.

Описательно постановка задачи по оптимизации работы узлов коммутации [2] может быть представлена следующим образом. Задан фрагмент ориентированной коммутационной сети, которая предназначена для связи множества источников $I = \{i_1, i_2, i_3, i_4, i_5, i_6\}$ с множеством приемников $\Pi = \{p_1, p_2, p_3, p_4, p_5, p_6\}$. В сети выделены два типа источников. Назовем их условно вертикальными $I_B = \{i_1, i_2, i_3\}$ и горизонтальными $I_H = \{i_4, i_5, i_6\}$. Соответственно выделены два типа приемников: горизонтальные $\Pi_H = \{i_1, i_2, i_3\}$ и вертикальные $\Pi_B = \{i_4, i_5, i_6\}$. Описанный фрагмент сети является ортогональным, служит для передачи информационных пакетов сверху-вниз и слева-направо, его структура изображена на рисунке 1.

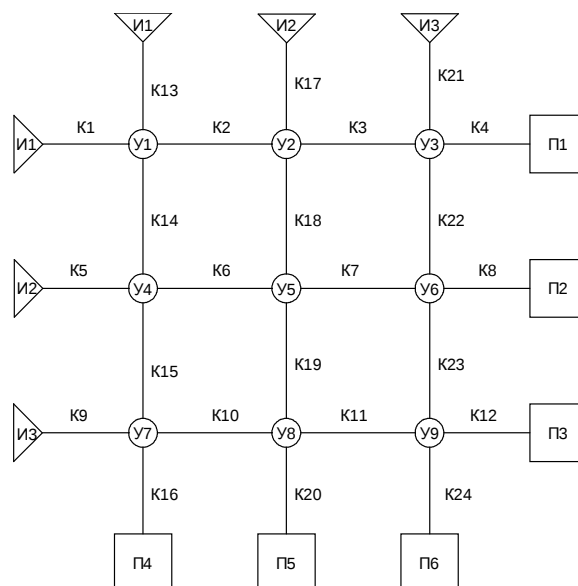


Рисунок 1 – Структура фрагмента ортогональной коммутационной сети (I_i – i -ый источник; Π_i – i -ый приемник; U_i – i -ый узел коммутации; K_i – i -ый локальный канал сети)

Передача информационных пакетов в такой сети осуществляется посредством информационных каналов, которые условно можно разделить на локальные и маршрутные. Маршрутные каналы передачи служат для связи источника с соответствующим приемником, а локальные для связи двух соседних узлов коммутации. В сети имеются узлы коммутации, которые предназначены для соединения двух локальных каналов. Основной особенностью рассматриваемой сети является то, что в каждом узле коммутируются пакеты в двух взаимно противоположных направлениях. Это вызывает конфликт, так как в один и тот же

момент времени через рассматриваемый узел возможно перемещение пакета только в одном направлении, второе направление в это время заблокировано.

Предполагается, что для каждого источника известно число пакетов, которое необходимо передать, и трудоемкость передачи каждого пакета, т.е. необходимые затраты времени. Пропускные способности каналов считаются достаточными, но узел коммутации имеет ограничивающую производительность, т.е. время прохождения пакета через узел коммутации не является пренебрежимо малым и считается прямо пропорциональным трудоемкости передачи пакета.

Узлы коммутации считаются абсолютно надежными, они работают независимо друг от друга и являются детерминировано управляемыми. Прерывания при прохождении пакета через узел коммутации отсутствуют: если пакет захватил узел коммутации, то он его не освободит до тех пор, пока полностью не закончится его передача. С каждым узлом коммутации связаны два настраиваемых параметра: η_i и τ_i ; η_i – частота переключения направления коммутации; τ_i – время между двумя непосредственно следующими переключениями в одном и том же направлении. Для простоты будем считать, что первоначально для всех i : $\tau_i = \tau$. В качестве начального приближения положим, что все $\eta_i = \eta_0$.

Режимы коммутации устанавливаются под воздействием соответствующих команд со стороны устройства управления (УУ). Качество управления оценивается по одному из двух согласованных критериев. Первый – суммарное время доставки всех пакетов, второй – среднее время простоя приемников сети, которое определяют время её реакции. Целевая задача управления – найти такие η_i и τ_i , которые минимизируют для заданного фрагмента сети и заданного множества пакетов выше предложенные критерии.

Таким образом, сложно переоценить важность процесса оптимизации при имитационном моделировании сложных систем, а в дальнейшем и при построении программных комплексов поддержки принятия решений в области синтеза и анализа сложных систем.

В таблице 1 приведена сравнительная характеристика различных программных комплексов, используемых при имитационном моделировании сложных систем.

По данным таблицы 1 видно, что сопряженная среда обладает рядом преимуществ даже по сравнению с наиболее гибкой и современной на сегодняшний день средой имитационного моделирования AnyLogic. Таким образом, разработка такой сопряженной системы является крайне важной и актуальной задачей.

Для сопряжения высокоуровневых систем разработки программ со специализированными системами имитационного моделирования исходя из теории «Операционных систем» известно несколько возможных способов организации взаимодействия. Опишем наиболее простой метод, с точки зрения понимания, основанный на передаче сообщений между приложениями. Рассмотрим общую идею взаимодействия программы, написанной на языке высокого уровня, (далее просто «программы») со средой моделирования GPSS World.

С помощью программы подготавливаются исходные данные для имитационной модели, затем средствами программы вносятся изменения в файл модели. Когда файл с моделью подготовлен, запускается среда моделирования GPSS World, в которую загружается подготовленный файл с моделью и запускается на выполнение. По окончании моделирования, из файла с отчетом в программу импортируются необходимые данные, и среда программирования за ненадобностью закрывается. На рисунке 2 изображены основные этапы взаимодействия программы со средой GPSS World.

Таблица 1 – Сравнительная характеристика программных комплексов для ИМ

Характеристики	Название системы				
	Delphi	C Builder	GPSS World	AnyLogic	Сопряженная система DG
Гибкость языковых конструкций	+	+	–	+	+
Лаконичность	–	–	+	+	+
Универсальность	+	+	–	+	+
Использование готовых библиотек алгоритмов на Паскале и Си	+	+	–	–	+
Создание интерфейса и анимация процесса моделирования	+	+	–	+	+
Использование библиотек моделей написанных на GPSS	–	–	+	–	+
Экспорт и импорт данных из различных приложений	+	+	–	+	+
Расширенные возможности по организации и проведению экспериментов с моделью	–	–	–	+	+
Объектно-ориентированный подход к созданию моделей	+	+	–	+	+

Рассмотрим эти этапы более детально. *Подготовка данных* подразумевает формирование списка переменных, которые представляют собой параметры имитационной модели, а также их конкретных значений для данного запуска модели.

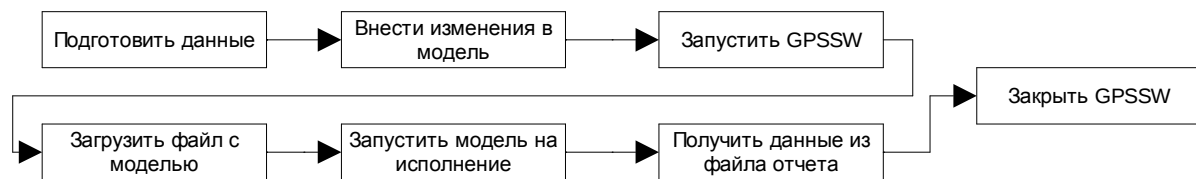


Рисунок 2 — Основные этапы сопряжения Delphi и GPSS

Внесение изменений в модель обозначает изменение текстового файла, содержащего имитационную модель, т.е. замену значений переменных в файле с моделью на значения, предназначенные для текущего запуска модели. Целесообразно сначала сделать копию текстового файла с моделью, и дальнейшие изменения производить не с оригиналом, а с копией. *Запуск среды моделирования GPSSW* представляет собой создание процесса на базе исполнимого файла среды моделирования GPSSW. При запуске исполнимого файла необходимо получить дескриптор процесса и заголовок созданного окна для того, чтобы в дальнейшем знать идентификатор получателя для отправки сообщений. *Загрузка модели* в GPSS начинается с отправки сообщения в окно GPSSW, предназначенного для вызова диалогового окна открытия файла. Затем необходимо получить идентификатор открывшегося диалогового окна. С помощью полученного идентификатора производится отправка двух сообщений диалоговому окну. Первое – предназначено для передачи имени файла с измененной моделью диалоговому окну, второе – дает указание на открытие этого файла. *Запуск модели* производится передачей сообщения окну GPSSW, содержащего указание на компиляцию и запуск модели. Момент окончания моделирования определяется по появлению окна в окне GPSSW, содержащего слово «REPORT». *Получение данных* из отчета производится в два этапа. На первом этапе по аналогии с загрузкой модели, производится сохранение отчета в текстовый файл, который на втором этапе считывается в программу. *Завершение работы* среды GPSSW осуществляется уничтожением дескриптора процесса.

В качестве *примера* реализации описанного подхода рассмотрим задачу об оптимизации процесса конвейерной обработки изделий. Пусть имеется конвейерная лента, на которую поступают детали на обработку с интенсивностью λ . Обработка ведется на двух взаимонезависимых устройствах, для которых заданы времена обработки одного изделия (равны μ_1 и μ_2). Система относится к классу СМО с потерями, так как если оба устройства заняты, а на обработку поступает очередная заявка, то она теряется. Необходимо найти такую интенсивность поступления изделий на конвейер, при которой устройства обработки будут достаточно загружены ($\rho > 0$), а потери заявок будут минимальны.

Целевая фракция может быть представлена в виде

$$\begin{aligned}
 F(c, \rho_1, \rho_2) &= \alpha W_1 - \beta W_2 \rightarrow \max \\
 W_1(\lambda, \mu_1, \mu_2) &= \rho_1(\lambda, \mu_1) + \rho_2(\lambda, \mu_2), \\
 W_2 &= \sum c,
 \end{aligned} \tag{1}$$

где c – потери заявок; ρ_1, ρ_2 – коэффициенты загрузки устройств; λ – интенсивность потока изделий; α, β – коэффициенты важности (компромисс между загрузкой оборудования и потерей заявок)

В рамках примера необходимо построить график зависимости загрузки приборов и количества потерянных заявок от интенсивности поступления деталей. Интенсивность будем вычислять внутри заданного пользователем интервала.

Описательно алгоритм поисковой оптимизации состоит из следующих пунктов.

1. Ввести границы интервала значений интенсивности
2. Пока не превышены границы интервала
 - 2.1. Увеличить счетчик на шаг изменения интенсивности
 - 2.2. Модифицировать параметры модели
 - 2.3. Осуществить прогон модели
 - 2.4. Проанализировать результаты прогона (файл отчета после моделирования)
 - 2.5. Вернуться к пункту 2
3. Построить график зависимости загрузки приборов и количества потерянных заявок от интенсивности поступления деталей
4. Вывести полученные результаты в виде таблицы

Очевидно, что данная задача не может быть эффективно решена только средствами среды GPSSW [3], в связи с тем, что для изменения интенсивности потока заявок необходимо останавливать, модифицировать и перезапускать модель. При выполнении этих процедур многократно вручную значительно увеличивается время проведения экспериментов с моделью. В связи с этим целесообразно использовать предлагаемый подход.

Сначала необходимо разработать модель в нотации GPSSW с учетом особенностей взаимодействия среды Delphi[4] с системой GPSSW. Ниже приведена возможная реализация такой модели.

В виде переменных задаются величины, которые будут передаваться из Delphi, и модифицироваться в процессе построения зависимости. Второй сегмент кода модели представляет собой описание действий с заявкой от её генерации до уничтожения. В третьем сегменте при помощи переменной задается время моделирования.

```

Lost VARIABLE 0
Lambda VARIABLE 2

GENERATE V$Lambda
GATE NU FAC1,ONE
SEIZE FAC1
ADVANCE 23
RELEASE FAC1
TERMINATE
ONE      GATE NU FAC2,TWO
SEIZE FAC2
ADVANCE 17
RELEASE FAC2
TERMINATE
TWO      SAVEVALUE LOST+,1
TERMINATE

GENERATE 1200
SAVEVALUE LOST+,1
TERMINATE 1
START 1

```

Перейдем к разработке приложения в среде программирования Delphi. Во-первых, для передачи данных в модель необходимо заполнить структуру данных типа **TVariablesList**, представляющую собой список изменяемых переменных и их значений. Пусть переменная такого типа имеет имя **MyVariables**, тогда сегмент кода программы необходимый для определения значения этой переменной имеет вид:

```

// задается число передаваемых переменных
MyVariables.Count:=1;
SetLength(MyVariables.Variables,1);
// задаются имена и значения передаваемых переменных
MyVariables.Variables[0].Name:='Lambda';
MyVariables.Variables[0].Value:=inttostr(i);

```

Здесь под переменной *i* понимается значение интенсивности поступления заявок, которое изменяется внутри заданного пользователем интервала с шагом 1.

Для запуска одного прогона модели используется последовательность действий, которая записана в следующем фрагменте текста программы.

```

//Открыть GPSSW
RunGPSS(GPSSW_Path);
//Изменение модели
InsertModelingData(ModelFileName,GPSSModelFileName,MyVariables);
//Выполнение модели
RunModel(GPSSModelFileName,ResultFileName);
//Извлечение данных из отчета
ExtractModelingData(ResultFileName,MyReport);
//Закрытие GPSSW

```

```

StopGPSS;
//Удалить временные файлы
AssignFile(txt,GPSSModelFileName);
Erase(txt);
AssignFile(txt,ResultFileName);
Erase(txt);

```

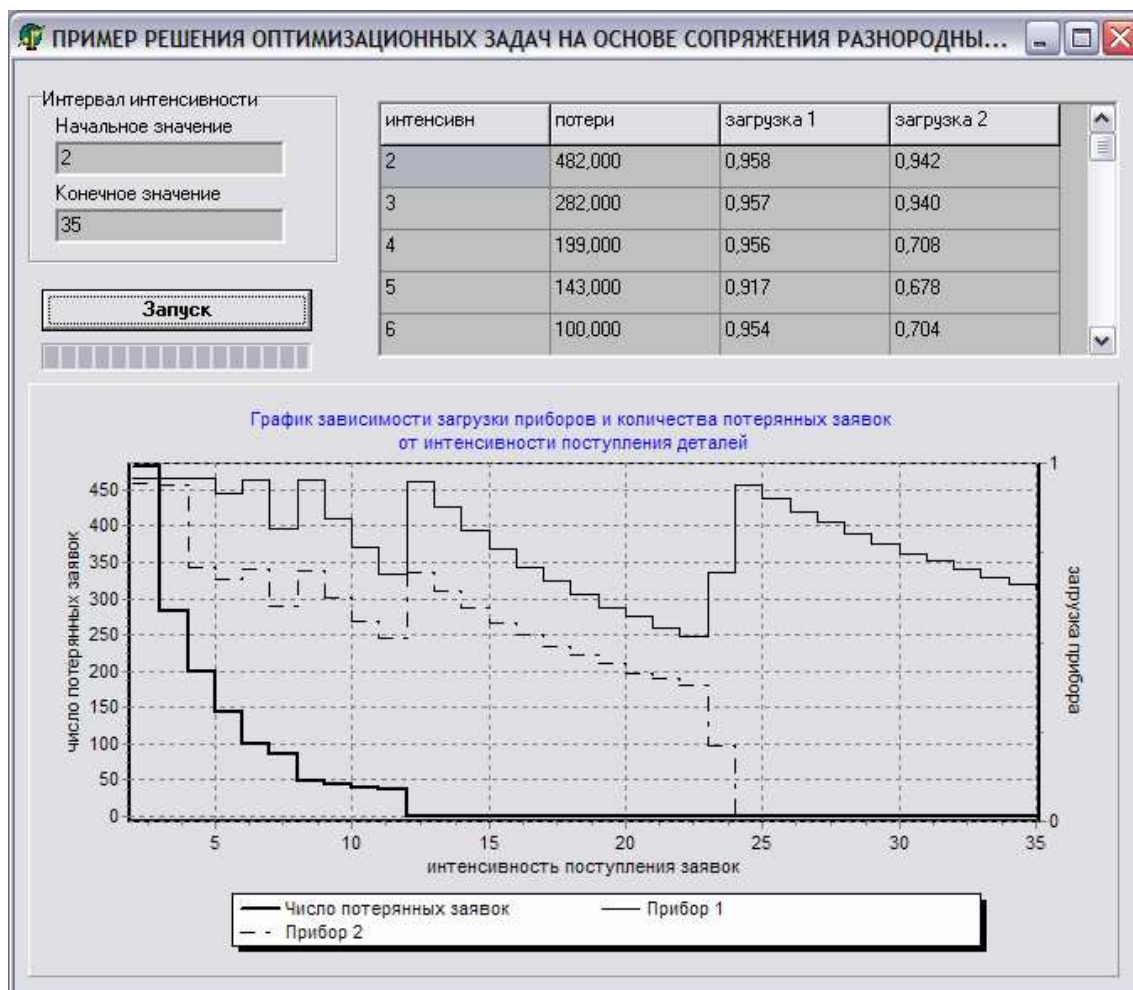


Рисунок 2 – Результаты решения задачи оптимизации работы конвейерной линии

Результат выполнения прогона модели (содержимое отчета) фиксируется в переменной MyReport типа TReport. Из этой переменной необходимо извлечь значение средней длины очереди СМО и перейти к реализации следующего прогона модели.

В результате прогонов модели был получен искомый график и данные, которые представлены в таблице 2.

Таблица 2 – Решение задачи оптимизации работы конвейерной линии ($\alpha = 0,5$; $\beta = 0,001$)

№ итерации	$F(c, \rho_1, \rho_2)$	λ	ρ_1	ρ_2	W_1	W_2
1	0,468	2	0,958	0,942	1,9	482
2	0,667	3	0,957	0,940	1,897	282
3	0,633	4	0,956	0,708	1,664	199
4	0,655	5	0,917	0,678	1,595	143
5	0,729	6	0,954	0,704	1,658	100
6	0,626	7	0,819	0,603	1,422	85
7	0,778	8	0,953	0,701	1,654	49
8	0,691	9	0,846	0,623	1,469	44
9	0,623	10	0,764	0,561	1,325	39
10	0,564	11	0,691	0,510	1,201	36
11	0,821	12	0,949	0,694	1,643	0
12	0,759	13	0,877	0,641	1,518	0

Продолжение таблицы 2

13	0,704	14	0,813	0,595	1,408	0
14	0,656	15	0,760	0,552	1,312	0
15	0,616	16	0,709	0,523	1,232	0
16	0,580	17	0,671	0,490	1,161	0
17	0,548	18	0,633	0,463	1,096	0
18	0,518	19	0,597	0,439	1,036	0
19	0,492	20	0,573	0,411	0,984	0
20	0,468	21	0,539	0,397	0,936	0
31	0,447	22	0,517	0,378	0,895	0
32	0,453	23	0,693	0,212	0,905	0
33	0,469	24	0,939	0	0,939	0
34	0,450	25	0,901	0	0,901	0
35	0,433	26	0,866	0	0,866	0
36	0,417	27	0,834	0	0,834	0
37	0,403	28	0,805	0	0,805	0
38	0,383	29	0,766	0	0,766	0
39	0,374	30	0,748	0	0,748	0
40	0,364	31	0,728	0	0,728	0
41	0,351	32	0,703	0	0,703	0
42	0,341	33	0,681	0	0,681	0
43	0,330	34	0,660	0	0,660	0
44	0,321	35	0,641	0	0,641	0

Из полученных данных видно, что оптимальной для рассматриваемой задачи будет интенсивность 12, так как при ней достигаются достаточно высокие коэффициенты загрузки устройств и в то же время заявки не теряются (значение целевой функции максимально и равно 0,821).

Проанализировав изложенную в этой статье информацию, можно сделать вывод о том, что предлагаемый нами подход позволяет предложить эффективное инструментальное средство решения оптимизационных задач. Посредством сопряжения GPSSW и Delphi был получен программный комплекс, позволяющий использовать библиотеки моделей реализованных на языке GPSS, а также использовать множество готовых оптимизационных методов, реализованных на языке Pascal. У сопряженной системы присутствуют следующие достоинства: лаконичность специализированного языка имитационного моделирования GPSSW, весь широкий спектр возможностей по созданию интерфейса, анимации процесса моделирования, импорта и экспорта данных, управления процессом моделирования и т.п. (от Delphi)

В качестве *дальнейшего развития* предполагается расширить возможности по обработке файлов-отчетов из GPSS и задействовать другие внешние приложения (например пакеты статистической обработки информации или использование таких математических пакетов как MatLab и MathCad).

Библиографический список

1. Реклейтис Г. Оптимизация в технике [В 2-х кн]. Кн. 1 / Г. Реклейтис, А. Рейвиндран, К. Рэгсдел. — М.: Мир, 1986. — 350 с.
2. Скاتков А.В. Имитационная модель поддержки принятия решений по управлению ортогональной коммутационной сетью / А.В. Скатков, Д.Ю. Воронин, Д.Н. Данильчук // Вестник СевГТУ. Сер. Информатика, электроника, связь: сб. науч. тр. — Севастополь, 2006. — Вып. 74. — С. 27–37.
3. Шрайбер Т.Дж. Моделирование на GPSS / Т.Дж. Шрайбер. — М.: Машиностроение, 1980. — 592 с.
4. Архангельский А.Я. Программирование в Delphi 7 / А.Я. Архангельский. — М.: Изд-во ООО «Бином-Пресс», 2003. — 1152 с.

Поступила в редакцию 14.07.2007 г.