Васильев Михаил Иванович – студент кафедры «Автоматизированные системы управления» ФГБОУ ВПО «Магнитогорский государственный технический университет им. Г.И. Носова». E-mail: Misha462@yandex.ru.
Парсункин Борис Николаевич – д-р техн. наук, действительный член АИН им. А.М. Прохорова, проф. кафедры «Автоматизированные системы управления» ФГБОУ ВПО «Магнитогорский государственный технический университет им. Г.И. Носова». Направление исследований: оптимизация управления технологическими процессами металлургического производства. Тел. 8(3519)298432. E-mail: pksu035@gmail.com.
Андреев Сергей Михайлович – канд. техн. наук, доц., зав. кафедрой «Автоматизированные системы управления» ФГБОУ ВПО «Магнитогорский государственный технический университет им. Г.И. Носова». Направление исследований: оптимизация управления технологическими процессами металлургического производства. Тел. 8(3519)298527. E-mail: pk_su@bk.ru.
Ахметов Тимур Уралович – аспирант кафедры «Автоматизированные системы управления» ФГБОУ ВПО «Магнитогорский государственный технический университет им. Г.И. Носова».

Васильев М.И., Парсункин Б.Н., Андреев С.М., Ахметов Т.У. Моделирование управления давлением в рабочем пространстве промышленных печей при использовании принципа нечёткой логики // Математическое и программное обеспечение систем в промышленной и социальной сферах. – 2014. – №2. – С. 35-45.

Vasiliev, M.I., Parsunkin, B.N., Andreev, S.M. and Akhmetov, T.U., 2014. Pressure control in industrial workspace furnaces using the principle of fuzzy logic. Software of systems in the industrial and social fields, 2: 35-45.

УДК 681.3.06

МАТЕМАТИЧЕСКОЕ И АЛГОРИТМИЧЕСКОЕ ОБЕСПЕЧЕНИЕ ПРОГРАММЫ ИМИТАЦИОННОГО МОДЕЛИРОВАНИЯ ДИНАМИЧЕСКИХ СИСТЕМ

Макарычев П.П., Захарикова Е.Б.

Аннотация. Обсуждаются модели и алгоритмы имитационного моделирования систем и сетей массового обслуживания. Обосновывается разработка программного комплекса с применением языка программирования C++ для математического пакета Mathcad. Приводятся примеры использования программного комплекса.
Ключевые слова. Имитационное моделирование, анализ систем массового обслуживания, стохастическая сеть Петри, технология разработки программного комплекса.

MATHEMATICAL AND ALGORITHMIC SUPPORT OF SOFTWARE FOR SIMULATION DYNAMIC SYSTEMS

Makarychev P.P., Zakharikova E.B.

Abstract. Models and algorithms for simulation of queuing systems and queuing networks are discussed. The development of software using the C++ programming language for mathematical package Mathcad is justified. Examples of using the software are given.
Keywords. Simulation, analysis of queuing systems, stochastic Petri network, software development technology.

Представление модельного времени

Динамическая природа дискретно-событийных имитационных моделей требует отслеживания текущего значения имитационного времени по мере функционирования имитационной модели. Существует два основных подхода к продвижению модельного времени: продвижение времени с постоянным шагом и продвижение времени от события к событию [1]. При использовании продвижения времени от события к событию часы модельного времени в исходном состоянии устанавливаются в «ноль» и определяется время возникновения будущих событий. После этого часы модельного времени переходят на время возникновения ближайшего события, и в этот момент обновляются состояние системы с учетом произошедшего события, а также сведения о времени возникновения будущих событий. Затем часы модельного времени продвигаются ко времени возникновения следующего (нового) ближайшего события, обновляется состояние системы и определяется время будущих событий, и т. д. Процесс продвижения модельного времени от времени возникновения одного события ко времени возникновения другого продолжается до тех пор, пока не будет выполнено какое-либо условие останова, указанное заранее [2].

Модели и алгоритмы источников заявок

Центральное место в перечне вопросов, связанных с имитацией стохастических элементов системы, занимает проблема построения датчиков случайных величин с заданными законами распределения. Базовым алгоритмом, на основе которого могут быть разработаны

алгоритмы и программы моделирования случайных величин с различными законами распределения, является алгоритм стандартной равномерно распределенной случайной величины. Для генерации случайных чисел, распределенных по экспоненциальному закону с параметром λ, использован метод нелинейного функционального преобразования. Для реализации генерирования нормально распределенных случайных чисел использован метод Марсальи – Брея [3]. Аргументами программы генерации заявок

InputStream (*double Tmod, int Ns, vector<int> distInStr, vector<vector<double>> distPar*)

являются время моделирования, количество источников заявок, законы распределения заявок. Возвращаемое программой значение имеет тип *vector<stream>*, где *stream* – структура. Элементы вектора представляют номера источников заявок, а поля структуры – соответствующие сгенерированные динамические массивы времен поступления заявок. Таким образом, снимается ограничение на количество источников и достигается компактность возвращаемого результата. Программа *sum* суммирует значения времен поступления заявок для каждого источника. На выходе получается поток времен заявок в абсолютном времени. Функция *CombineNStream* (*vector<stream> stm*) осуществляет объединение времен поступления заявок из всех источников в один массив, а функция *Identif* идентифицирует номера источников, соответствующих заявкам объединенного массива.

Модели и алгоритмы узлов обслуживания

Алгоритмические модели узлов обслуживания программного комплекса построены с использованием механизма временных стохастических ингибиторных сетей Петри, включая СМО типа *M*/*M*/*m*/0, *M*/*M*/*m*, *M*/*M*/1/*n*, *M*/*M*/*m*/*n* [9, 11, 12]. Модельное представление многоканальной с ограниченным буфером СМО типа *M*/*M*/*m*/*n* в виде временной стохастической сетью Петри приведено на рис. 1.

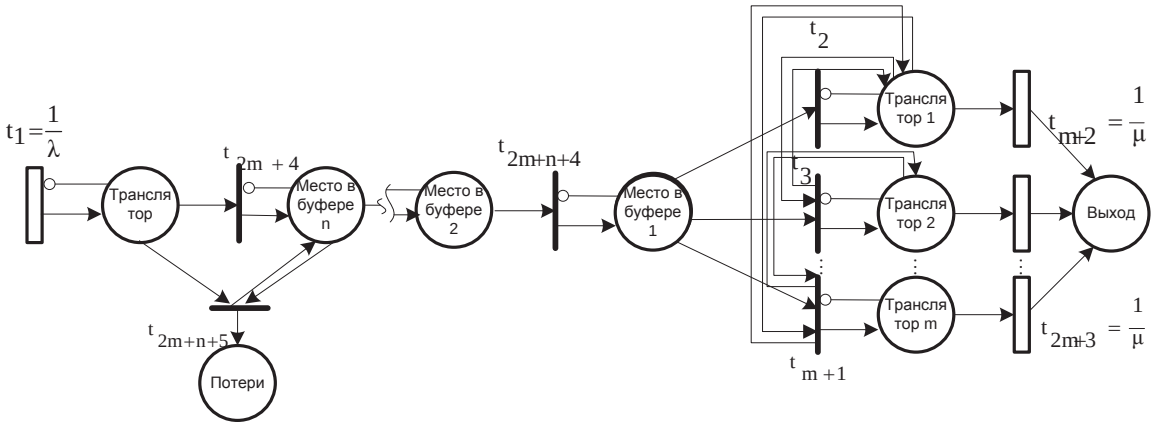

Рис. 1. Стохастическая сеть Петри для СМО типа M/M/m/n

Основная программа комплекса

Obsl(*double T, int Nk, vector<double> tpos, vector<int>nu, vector<vector<double>> distpar, vector<int>dist, vector<int> id, vector<int> typeBuf, vector<int> buf, vector<int>ind*).

Программа функционирует при задании следующих аргументов: времени моделирования, количества параллельно соединенных узлов обслуживания, массивов времен поступления заявок, количества каналов в каждом узле, законов и параметров распределения времен обслуживания заявок в узлах СеМО, идентификаторов заявок, типов и емкости буферов узлов, массивов номеров параллельно соединенных узлов в сети.

В цикле программы *Obsl*() вызывается подпрограмма *OccupyChannel*, осуществляющая выбор узла и канала, в котором будет произведено обслуживание. Для этого используются вспомогательные векторы структуры: *tpb* – время поступления в канал, *tobb* – время ожидания в буфере, *tv* – время выхода обслуженной заявки из канала. Сначала происходит поиск свободных каналов, т.е. таких, где время поступления следующей заявки больше или равно временам выхода предыдущих (*tpos*[*i*]>=*tv*[*j*].*ok*[*uj*].*st*[*x*]). Если свободные узлы найдены, из них случайным образом выбирается один. Если выбранный узел имеет более одного свобод-

ного канала, также случайно выбирается любой из них. При этом вспомогательные массивы, соответствующие выбранным узлу и каналу, очищаются и добавляются данные принятой к обслуживанию заявки.

Если свободных узлов не найдено, определяются их типы буферов. В случае, когда у всех узлов буфер отсутствует, фиксируются потери. Иначе для каждого узла с ограниченным буфером проверяется текущая емкость буфера (длина очереди) и сравнивается с максимальной. Случайный выбор узла (*k*) осуществляется из узлов с типами буфера неограниченный и ограниченный, если текущая емкость буфера позволяет. Выбирается канал (*uz*), при обслуживании в котором время ожидания заявки минимально. Операция вставки в буфер осуществляется функцией *PutBufer*(*tpb*[*k*].*ok*[*uz*].*st, tobb*[*k*].*ok*[*uz*].*st, tv*[*k*].*ok*[*uz*].*st, tpos*,*i*), принимающей в качестве аргументов времена поступления, ожидания и выхода из выбранного канала, а также массив и индекс поступающих заявок. Данная функция удаляет из массивов *tpb, tobb, tv* вышедшие заявки ко времени поступления текущей заявки, рассчитывает время ожидания и записывает данные в соответствующие массивы. Управление возвращается к программе *OccupyChannel*, которая рассчитывает время обработки в зависимости от заданных вида и параметров распределения для узла либо для типа заявки. На основе известных данных вычисляется также время выхода заявки и собирается статистика, включающая времена обработки, ожидания, выхода, количество потерь. Модель структуры СеМО задается в виде ориентированного графа, вершины которого представляют множество возможных узлов обслуживания. Количество узлов обслуживания не ограничено. Возможно задание последовательного, параллельного и смешанного соединения узлов.

Компоненты обработки результатов имитационного моделирования

Общеизвестно, что имеется несколько трактовок определения вероятности. Классический метод основан на определении отношения числа исходов опыта, благоприятствующих определенному событию, к общему числу возможных исходов опыта [4]. Геометрический метод предлагает определение вероятности как отношение меры части к общей мере (длине, площади, времени и т.д.). В данном комплексе вероятность нахождения системы в определенном состоянии трактуется и вычисляется как среднее относительное время пребывания системы в этом состоянии, равное отношению среднего времени пребывания системы в этом состоянии к времени моделирования [5].

Время моделирования выбирается в пятьдесят и более раз больше времени переходного периода, то есть таким образом, чтобы исключить (или по меньшей мере уменьшить) влияние переходного периода на величины характеристик системы. Для определения количества членов в выборке *n* уместно использовать метод оценок Бернулли, применяемый для вероятностных характеристик [6].

При аналитическом моделировании в результате решения системы линейных уравнений основными выходными характеристиками СМО типа *M/M/n/m* являются вероятности нахождения системы в одном из возможных состояний. В комплексе прикладных программ имеется программа *FindP*, которая позволяет оценить эти характеристики. Алгоритм программы основан на анализе загрузки каналов и буфера системы. На рис. 2 приведены графики загрузки каналов $f(t)_0$, $f(t)_1$, $f(t)_2$ и общей загрузки трехканальной СМО $g(t)$.

Так как графики функций имеют ступенчатый характер, то очевидно, что скачки функции $g(t)$ возможны только в тех точках, в которых имеются скачки в функциях $f(t)_0$, $f(t)_1$, $f(t)_2$. Следовательно, для определения значений характеристик в точках t_0, t_1, t_2 можно анализировать только значения функций в точках скачков и в областях, достаточно близких к точкам скачков. Кроме данных характеристик в программном комплексе вычисляются и другие характеристики [7]: средняя длина очереди, среднее время ожидания, среднее время обслуживания.

Технология разработки программного комплекса

В данном разделе приведена последовательность действий для создания пользовательской библиотеки на *C*/*C++* для *Mathcad* при помощи механизма *UserEFI* (интерфейс взаимодействия динамических библиотек *Windows* (*DLL*) и *Mathcad*) [8, 10]. Главная функция имеет имя *Model* и указывается в диалоговом окне *Insert Function* (Вставка функции) в категории (Моделирование). В качестве среды разработки пользовательской библиотеки для *Mathcad* выбрана *Microsoft Visual Studio* 2010. Выбор данной среды разработки неслучаен: именно ее используют разработчики пакета *Mathcad* и именно для компиляторов *Microsoft* в последних версиях *Mathcad* поставляются файлы примеров, заголовочные и *lib*-файлы. Для создания пользовательской функции *Mathcad* на *C*/*C++* требуется реализовать проект: подключить заголовочный и библиотечный файлы из комплекта поставки *Mathcad*; создать и заполнить массив (таблицу) сообщений ошибок, возникающих при вызове пользовательской функции; разработать код функции по правилам механизма *UserEFI*; создать и заполнить структуру, описывающую пользовательскую функцию при подключении к *Mathcad*; разработать код регистрации таблицы сообщений об ошибках и пользовательской функции; создать специальный файл с информацией о пользовательской функции для отображения в диалоге *Insert Function*.

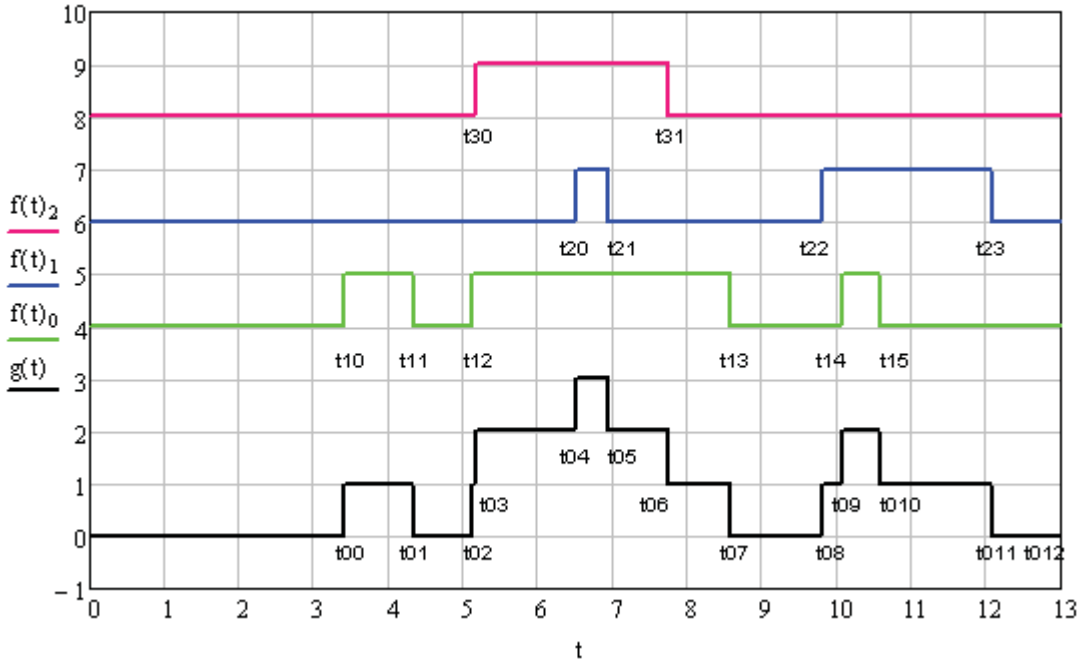

Рис. 2. Графики загрузки трехканальной СМО

Подключение файлов *mcadincl.h* и *mcaduser.lib*

Для создания пользовательских функций пакета *Mathcad* необходимо подключить к созданному проекту заголовочный файл *mcadincl.h* и библиотечный файл *mcaduser.lib*. В файле *mcadincl.h* находятся определения констант, типов и прототипы функций используемых при взаимодействии *Mathcad* и пользовательской библиотеки. Для компиляторов фирмы *Microsoft* он находится в папке *\UserEFI\MicroSft\Include*, вложенной в папку установки *Mathcad*, а файл *mcaduser.lib* — в папке *\UserEFI\MicroSft\Lib*.

Для подключения заголовочного файла *mcadincl.h* необходимо добавить следующие строки в файл *stdafx.h* проекта:

// TODO: reference additional headers your program requires here

#include "…Mathsoft\Mathcad \UserEFI\Microsft\include\mcadincl.h"

Подключение файла *mcaduser.lib* осуществлено путем добавления ссылки на файл в настройках проекта, и редактирования списка путей к *lib*-файлам проектов *C*/*C++* .

Разработанная пользовательская функция осуществляет проверку на правильность входных данных и сигнализирует об ошибке, аналогично встроенным функциям *Mathcad*. Помимо всего прочего, для осуществления проверки необходимо создать таблицу (массив) строк, в которой будет храниться текстовое описание ошибок.

Разработка кода функции

Тип возвращаемого значения главной функции *Model — LRESULT*. Все пользовательские функции для *Mathcad*, работающие по механизму *UserEFI* должны иметь такой тип возвращаемого результата.

Функция *Model* имеет одиннадцать аргументов:

```
LRESULT Model(
        COMPLEXARRAY * const Result,
        const COMPLEXARRAY * const TmodN,
        const COMPLEXARRAY * const Nsk,
  const COMPLEXARRAY * const distInStr,
  const COMPLEXARRAY * const distPar,
  const COMPLEXARRAY * const distCan,
  const COMPLEXARRAY * const parCan,
  const COMPLEXARRAY * const bufer,
  const COMPLEXARRAY * const volumeBufer,
  const COMPLEXARRAY * const numCan,
  const COMPLEXARRAY * const Struct)
```

В механизме *UserEFI* первый аргумент функции (с именем *Result*) – это результат, возвращаемый при вызове функции в *Mathcad*, т.е. тот результат, который получен в рабочем документе *Mathcad*. В данном случае он имеет тип *COMPLEXARRAY * const* (т. е. постоянный указатель на переменную типа *COMPLEXARRAY*). Тип *COMPLEXARRAY* описан в файле *mcadincl.h* как:

```
// complex array type
typedef struct tagCOMPLEXARRAY {
   unsigned int rows;
   unsigned int cols;
   double **hReal;   // hReal[cols][rows],
                // == NULL when the real part is zero
   double **hImag;   // hImag[cols][rows],  == NULL when the imaginary part is zero
} COMPLEXARRAY;
```

Первые два поля структуры хранят количество строк и столбцов в массиве соответственно. Третье и четвертое поля – массивы действительных и мнимых частей элементов массива. Для доступа к действительной части элемента массива в строке с номером *i* и столбце с номером *j* необходимо использовать

$$A\text{->}hReal[j][i]$$

а для доступа к мнимой части:

$$A\text{->}hImag[j][i],$$

где *A* – переменная типа *COMPLEXARRAY*.

Использование типа *COMPLEXARRAY* для первого аргумента функции означает, что в рабочем документе функция будет возвращать матрицу.

Следующие аргументы функции также имеют тип указателя на переменную типа *COMPLEXARRAY* с единственным отличием: они постоянны (*const*), что означает, что изменять их значения запрещено. В начале функции производится проверка на правильность значений аргументов. После всех проверок производится непосредственно запуск процесса моделирования. В конце функции производится возврат нулевого значения (*return* 0;), что указывает на успешное выполнение функции.

Создание структуры с информацией о функции

Для каждой функции пользователя *Mathcad*, созданной по механизму *UserEFI*, требуется заполнить описывающую ее специальную структуру. По информации из этой структуры *Mathcad* определяет имя функции, ее адрес количество и типы передаваемых аргументов и возвращаемого результата.

Для создания такой структуры в файл *model.cpp* после кода функции *Model* добавлены следующие строки:

```
FUNCTIONINFO fiModel =
{
  "model",
"Tm,N,Numbersource,NumberNode,DistSource,ParSource,TypeService,DistNode,ParNode,TypeBuf
,VolumeBud,NumChannel,Struct",
  "Моделирование",
  (LPCFUNCTION) Model,
  COMPLEX_ARRAY,
  10,
  {COMPLEX_ARRAY, COMPLEX_ARRAY, COMPLEX_ARRAY, COMPLEX_ARRAY,
COMPLEX_ARRAY, COMPLEX_ARRAY,COMPLEX_ARRAY, COM-
PLEX_ARRAY,COMPLEX_ARRAY,COMPLEX_ARRAY}
};
```

Данный код создает глобальную переменную типа *FUNCTIONINFO*, который определен в файле *mcadincl.h* как:

```
typedef struct tagFUNCTIONINFO {
   char * lpstrName;
   char * lpstrParameters;
   char * lpstrDescription;
   LPCFUNCTION lpfnMyCFunction;
   long unsigned int returnType;
   unsigned int nArgs;
   long unsigned int argType[MAX_ARGS];
} FUNCTIONINFO;
```

Первое поле структуры – *lpstrName* – имя функции, которое используется в документах *Mathcad* и в диалоге вставки функции. Как было ранее указано, данная функция имеет имя *Model*.

Второе поле структуры – *lpstrParameters* – содержит строку с перечнем аргументов функции. Эта строка отображалась в диалоговом окне *Insert Function* (Вставки функции) в *Mathcad* 7. Начиная с 8-й версии *Mathcad*, перечень аргументов указывается в *xml*-файле описания функций, структура и содержимое которого будут представлены далее.

Третье поле структуры – *lpstrDescription* – содержит строку с описанием функции. Как и предыдущее поле, эта строка отображалась в диалоговом окне *Insert Function* (Вставки функции) в *Mathcad* 7, а с 8-й версии описание функции берется из специального *xml*-файла. Для последних версий *Mathcad* в качестве значения можно задавать пустую строку.

Четвертое поле структуры – *lpfnMyCFunction* – указатель на функцию типа *LPCFUNCTION*, определенного в файле *mcadincl.h* как:

```
typedef LRESULT (* LPCFUNCTION )
  ( void * const, const void * const, ... );
```

Данная запись означает, что пользовательская функция для *Mathcad*, созданная по механизму *UserEFI*, на языке *C++* всегда должна иметь тип результата *LRESULT*, изменяемый первый аргумент, который будет хранить возвращаемый в *Mathcad* результат, и не менее одного неизменяемого аргумента.

Пятое поле структуры *FUNCTIONINFO* – поле *returnType* – это число, характеризующее тип результата, возвращаемого функцией. Значения для каждого из возможных типов

результатов приведены в файле *mcadincl.h*. В данном случае результат вычисления функции имеет тип *COMPLEXARRAY* поэтому используется значение, определенное под именем *COMPLEX_ARRAY*.

Шестое поле – *nArgs* – содержит количество аргументов функции.

Седьмое (последнее) поле – *argType*[] – массив чисел, характеризующих тип аргументов функции (аналогично пятому полю *returnType*). В данном случае используются аргументы типа *COMPLEXARRAY*, поэтому указываются идентификаторы *COMPLEX_ARRAY*.

Подключение пользовательской функции к *Mathcad*

Подключение пользовательской функции по механизму *UserEFI* происходит следующим образом. *Mathcad* загружает все библиотеки (файлы с расширением *dll*) из папки *UserEFI* (находится в папке установки *Mathcad*). При загрузке библиотека должна зарегистрировать при наличии свою таблицу описания ошибок и пользовательских функции для пакета *Mathcad*, определенных в ней. В случае успешной регистрации пользовательских функций они становятся аналогичны встроенным функциям пакета *Mathcad*. Информация для диалога *Insert Function* (Вставки функции), начиная с 8-й версии *Mathcad*, берется, в случае наличия, из специальных *xml*-файлов. Для обработки события загрузки библиотеки и регистрации функции необходимо внести изменения в функцию *DllMain* в файле *model.cpp*:

```
BOOL APIENTRY DllMain(
    HINSTANCE hModule,
    DWORD ul_reason_for_call,
    LPVOID lpReserved)
{
 switch (ul_reason_for_call)
 {
  case DLL_PROCESS_ATTACH:
    {
     if (CreateUserErrorMessageTable(hModule, 8, myErrorMessageTable))
     {
      CreateUserFunction(hModule, &fiConeVolume);
     };
    }
    break;
 }
 return TRUE;
}
```

Функция *DllMain* вызывается операционной системой в различных ситуациях (определяется параметром *ul_reason_for_call*). Параметр *ul_reason_for_call* равен значению *DLL_PROCESS_ATTACH* при загрузке библиотеки. В приведенном выше коде в этом случае производится:

– регистрация таблицы сообщений об ошибках путем вызова функции *CreateUserErrorMessageTable* (первый аргумент функции – *hModule* – дескриптор текущего модуля (библиотеки с пользовательскими функциями); второй – число сообщений об ошибках; третий – указатель на массив, содержащий описание ошибок);

– регистрация пользовательской функции путем обращения к функции *CreateUserFunction* (первый аргумент функции – *hModule* – дескриптор текущего модуля; второй аргумент – указатель на структуру, содержащую информацию о пользовательской функции).

После внесения всех указанных изменений необходимо скомпилировать библиотеку и полученный файл с расширением *dll* (*model.dll*) поместить в папку *UserEFI*, вложенную в папку установки *Mathcad*. После запуска *Mathcad* станет возможным использование разработанного комплекса программ.

Файл описания функции

Для указания информации о функции в диалоговом окне *Insert Function* (Вставка функции) в пакете *Mathcad*, начиная с 8-й версии, необходимо создать специального вида *xml*-файл и поместить его в папку *DOC\FUNCDOC* папки установки *Mathcad*. В данном случае имя файла – *model_RU.xml* и размещаться он будет в папке …*Mathsoft\Mathcad* 15*Doc\Funcdoc*\. Содержимое файла таково:

<?*xml version*="1.0" *encoding*="*UTF*-8"?>
<*FUNCTIONS*>
<*help_file*>*help_model.chm*</*help_file*>
<*function*>
<*name*>*model*</*name*>
<*params*>*TmodN*,
Nsk,distInStr,distPar,distCan,parCan,bufer,volumeBufer,numCan,Struct</*params*>
<*category*>Имитационное моделирование</*category*>
<*description*>Функция *model* программного комплекса имитационного моделирования систем и сетей массового обслуживания «Афелий» является функцией десяти аргументов: *TmodN* (время моделирования, число повторений), *Nsk* (количество источников, узлов, тип обслуживания), *distInStr* (законы распределения для источников заявок), *distPar* (параметры распределения для источников заявок), *distCan* (законы распределения для узлов обслуживания), *parCan* (параметры распределения для узлов обслуживания), *bufer* (типы буферов), *volumeBufer* (емкости буферов), *numCan* (количество каналов в узлах), *Struct* (структура сети).
</*description*>
<*help_topic*>*help_model.chm*</*help_topic*>
</*function*>
</*FUNCTIONS*>

Верхняя строчка в приведенном тексте стандартная для всех *xml*-файлов. На следующей строке открывается корневой элемент *xml*-файла: *FUNCTIONS*. Для файлов описания функций пакета *Mathcad* эта строчка должна присутствовать без изменений. На третьей строке находится элемент *xml*-файла, содержимое которого указывает на *chm*-файл справки, связанной с функциями для *Mathcad*, описываемыми в *xml*-файле. На четвертой строке открывается элемент *function*, который содержит информацию о функции для пакета *Mathcad*. Для функции указывается имя, список параметров, категория функции, описание функции, раздел из файла справки. На десятой и одиннадцатой строках производится закрытие элементов *function* и *FUNCTIONS*. В том случае, если пользовательских функций несколько, то на каждую функцию создается свой элемент *function*. с соответствующими элементами *name, params, category* и т. д.

После формирования *xml*-файла описания функции и его сохранения в требуемой папке, запустим *Mathcad* и обратимся к диалоговому окну *Insert Function* (Вставка функции). Созданная функция *Model* появляется в этом диалоге аналогично встроенным функциям пакета *Mathcad* (рис. 3).

Алгоритм функционирования СМО *M*/*M*/3/3 проиллюстрирован временной стохастической сетью Петри, представленной на рис. 1. Графическая интерпретация генераторов заявок $f_0(t)$, занятости каналов $g_0(t)$, $g_1(t)$, $g_2(t)$, суммарной занятости узла $g_3(t)$, суммарной занятости каналов и буфера $v(t)$ представлена на рис. 4-6.

Заключение

В разработанном программном комплексе модель структуры СеМО задается в виде ориентированного графа. Алгоритмические модели узлов обслуживания построены на основе механизма временных стохастических ингибиторных сетей Петри. Для определения вероятностей нахождения системы в определенном состоянии используется геометрический метод. Алгоритм расчета узловых характеристик сети массового обслуживания обеспечивает

проведение сравнительного анализа результатов имитационного моделирования с результатами решения системы уравнений Колмогорова. С помощью метода оценок Бернулли доказано, что при моделировании стохастических процессов достаточно воспользоваться одной, но достаточно длинной, реализацией. Программный комплекс «Афелий» имеет свидетельство о государственной регистрации программы для ЭВМ № 2013612116.

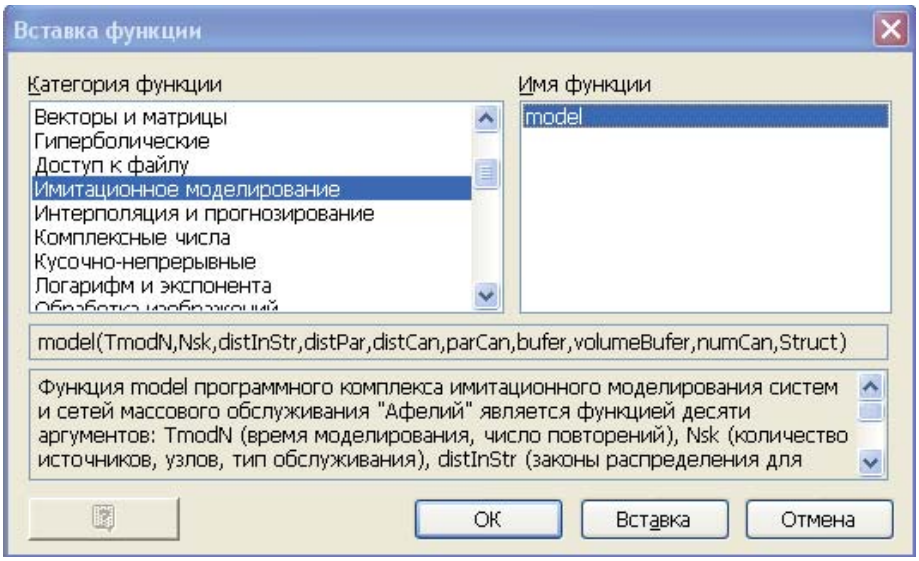

Рис. 3. Диалоговое окно Insert Function с пользовательской функцией

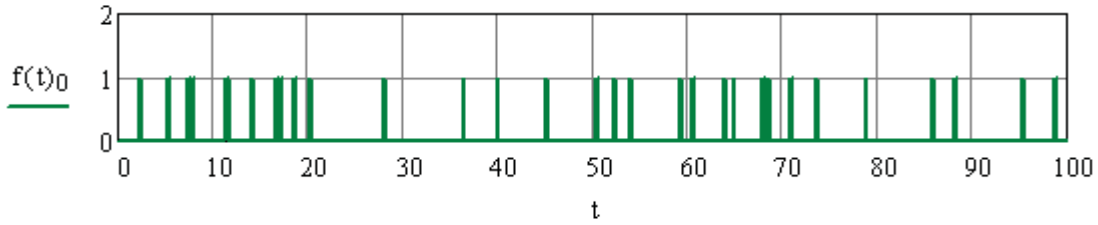

Рис. 4. Графическая интерпретация потока заявок

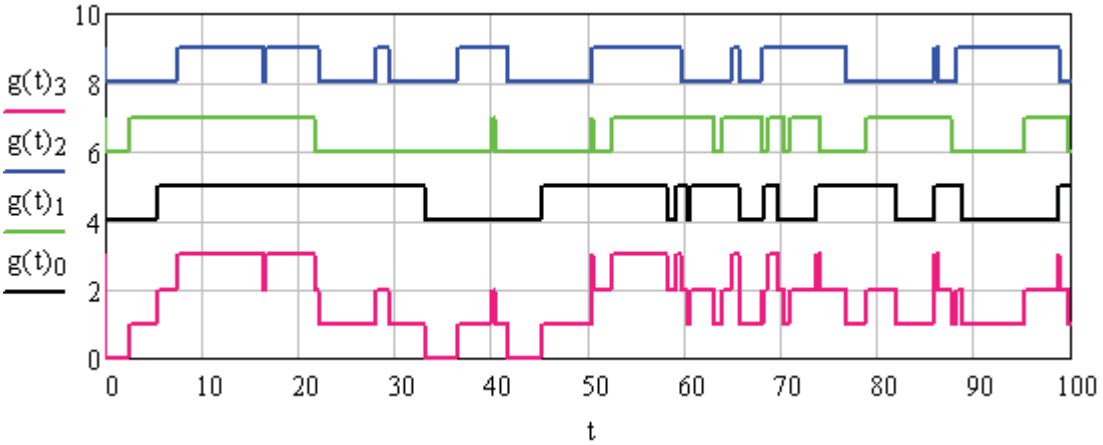

Рис. 5. Графическая интерпретация занятости каналов и суммарной занятости узла

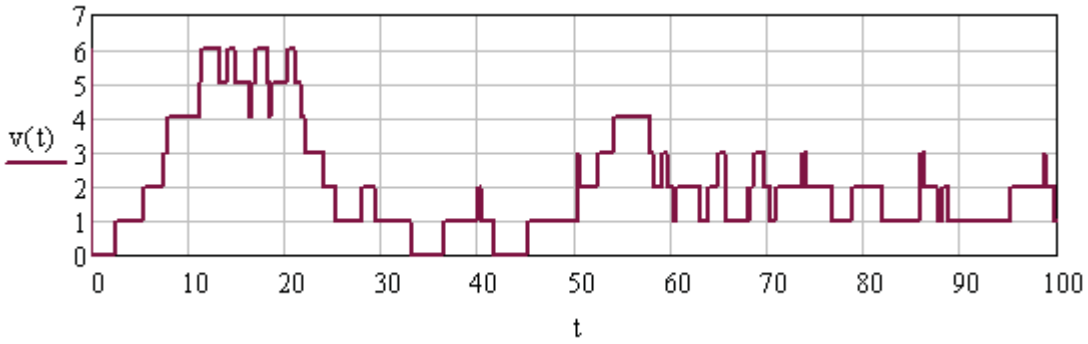

Рис. 6. Графическая интерпретация суммарная занятости каналов и буфера

Список используемых источников

1. *Sheldon, M.R. Simulation / M. Sheldon. – Imprint*: *Academic Press*, 2012. – 328 *p*.
2. Калашников, В.В. Организация моделирования сложных систем / В.В. Калашников. – М.: Знание, 1982. – 62с.
3. *Gentle, J.E. Random number generation and Monte Carlo methods / J.E. Gentle. - New York*: *Statistics and Computing*, 2003. – 381 *p*.
4. Баврин И.И. Теория вероятностей и математическая статистика / И.И.Баврин – М.: Высшая шк., 2005. – 160 с.
5. Климов Г.П. Теория массового обслуживания / Г.П. Климов – Изд-во МГУ, 2011. – 312 с.
6. Шеннон, Р. Имитационное моделирование систем. Искусство и наука / Р. Шеннон. – М.: Мир, 1978. – 418 с.
7. Крылов С.М. Неокибернетика: алгоритмы, математика эволюции и технологии будущего / С.М. Крылов – ООО «Издательство ЛКИ». – 208. – 499 с.
8. Очков, В.И. *Mathcad* 14 для студентов, инженеров и конструкторов / В.И. Очков. – СПб.: *BHV*-Петербург, 2007. – 497 с.
9. Захарикова Е.Б. Разработка программного обеспечения для исследования сетей массового обслуживания / Е.Б. Захарикова // Имитационное моделирование. Теория и практика: материалы *V* (юбилейной) Всероссийской научно-практической конференции по имитационному моделированию и его применению в науке и промышленности. – СПб., 2011. - С. 96-97.
10. Захарикова, Е.Б. Имитационное моделирование систем и сетей массового обслуживания средствами приложения к пакету Mathcad / Е.Б. Захарикова, П.П. Макарычев // Извести высших учебных заведений. Поволжский регион. Технические науки. – 2012. – № 3. – С. 29-37.
11. Пащенко, Д.В. Проблемы построения многопоточной модели программного обеспечения экспертной системы авиационных радиолокационных комплексов // Д.В. Пащенко, Д.А. Крокоз // Известия высших учебных заведений. Поволжский регион. Технические науки. – 2011. – № 2. – С. 21-29.
12. Haas, Peter J. (Peter Jay) Stochastic Petri nets : modelling, stability, simulation / Peter J. Haas. // Springer-Verlag New York, Inc., 2002. – 509 p.

Макарычев Петр Петрович – д-р техн. наук, проф., Почетный работник высшего профессионального образования, заведующий кафедрой «Математическое обеспечение и применение ЭВМ» ФГБОУ ВПО «Пензенский государственный университет». *E-mail*: *makpp@yandex.ru*.
Захарикова Елена Борисовна – к-т техн. наук, программист ООО «Алгоритм-Сервис». *E-mail*: *rabbit7@live.ru*.

Макарычев П.П., Захарикова Е.Б. Математическое и алгоритмическое обеспечение программы имитационного моделирования динамических систем // Математическое и программное обеспечение систем в промышленной и социальной сферах. – 2014. – №2. – С. 45-54.

Makarychev, P.P. and Zakharikova, E.B., 2014. Mathematical and algorithmic support of software for simulation dynamic systems. Software of systems in the industrial and social fields, 2: 45-54.

УДК 519.688

РЕЗУЛЬТАТЫ СРАВНИТЕЛЬНОГО АНАЛИЗА РЕШЕНИЯ МНОГОКРИТЕРИАЛЬНОЙ ЗАДАЧИ ОПТИМИЗАЦИИ ДЛЯ РАСЧЕТА СТРУКТУРЫ ШИХТОВЫХ МАТЕРИАЛОВ ДУГОВОЙ СТАЛЕПЛАВИЛЬНОЙ ПЕЧИ

Логунова О.С., Сибилева Н.С., Павлов В.В.

Аннотация. В работе представлены результаты решения многокритериальной задачи оптимизации определения структуры шихтовых материалов дуговой сталеплавильной печи с помощью трех методов: метод уступок, метол свертки и метод ограничений. Особенностью рассматриваемой задачи является наличие эмпирической системы, включающей целевую функцию, систему ограничений и последовательность двух взаимосвязанных