Электронный научно-технический журнал Октябрь 2010
http://www.bru.mogilev.by

МЕТОД ИМИТАЦИОННОГО МОДЕЛИРОВАНИЯ РАБОТЫ УЗЛОВ ЛОКАЛЬНОЙ ВЫЧИСЛИТЕЛЬНОЙ СЕТИ

А.Ю. Алексейчикова, **А.И. Якимов, К.В. Захарченков**

В статье рассматривается метод построения имитационной модели локальной вычислительной сети (ЛВС) LocalNet, обеспечивающей возможность оценки загруженности отдельных компонентов узлов сети. Для построения имитационной модели использована среда визуального моделирования ModelDesigner.

Ключевые слова: визуальные средства имитационного моделирования, узел локальной вычислительной сети

1. Состав и структура имитационной модели локальной вычислительной сети

В состав модели входят компоненты для:

а) задания параметров оборудования узла сети;

б) просмотра журнала последовательности обращений к компонентам узла;

в) задания параметров запуска модели (квант времени, количество запросов);

г) задания параметров запросов (средние значения и отклонения количества операций, объема данных в ОЗУ и на жестком диске, объема данных, передаваемых по сети, времени вывода данных) для разных видов запросов (ввода, обработки, корректировки, печати, удаления информации, запроса модуля формирования отчета, проведения математических расчетов);

д) анализа средней загруженности всего узла и отдельных его компонентов (ЦПУ и ОЗУ, жесткого диска, устройства вывода, сети).

2. Метод построения имитационной модели локальной вычислительной сети

Узел локальной вычислительной сети представляет собой совокупность оборудования и программных компонент. Упрощенно оборудование включает в себя центральный процессор и ОЗУ (*Core*), внешнюю память (*HDD*), устройство вывода (*Output*) и сетевой адаптер (*Net*) [1,2].

Схема взаимодействия основных компонентов модели представлена на *рисунке* 1.

На *рисунке* 1 применены следующие обозначения:

GenerateQuery – генерация заявок на узле.

CreateQuery – создать запрос.

QueryProcessing – обработать запрос.

CoreProcessing – обработка средствами ЦПУ и ОЗУ.

HDDProcessing – обработка запросов, требующих ресурсов жесткого диска.

OutputProcessing – обработка устройством вывода.

NetProcessing – обработка запроса сетевым адаптером.

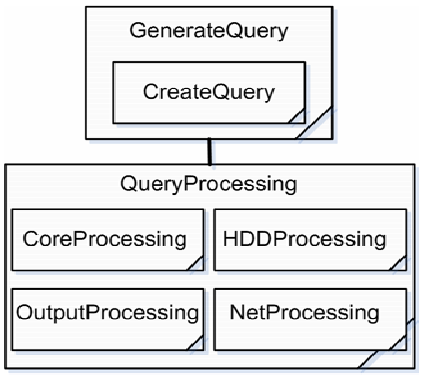

Рис. 1 – Схема взаимодействия основных компонентов модели

Имитационная модель содержит следующие структуры данных

Tdistribution – распределение, определяется следующими параметрами [3]:

mean – среднее значение,

standardDerivation(der) – дисперсия,

type – вид распределения.

TKindQuery – вид запроса, может принимать следующие значения [2]:

qrMVV – запрос модуля ввода информации осуществляет следующие действия: обращение к информационной базе данных, поиск свободного места, проверка связей, занесение в ИБД.

qrMOB – запрос модуля обработки информации обращается к информационной базе данных, ищет заданную первую запись, обрабатывает ее, ищет вторую заданную запись, обрабатывает ее, …, ищет N-ю заданную запись, обрабатывает ее.

qrMKO – запрос модуля корректировки информации обращается к информационной базе данных, ищет заданную запись, модифицирует ее атрибуты, проверяет связи, заносит исправленную запись в ИБД.

qrMYD – запрос модуля удаления информации обращается к информационной базе данных, ищет заданную запись(и), проверяет связи, удаляет искомую запись.

qrMOT – запрос модуля формирования отчета обращается к информационной базе данных, ищет N заданных записей, обрабатывает и формирует отчет.

qrMPE – запрос модуля печати информации печатает отчет, к информационной базе данных не обращается.

qrMCch – запрос модуля проведения математических расчетов к информационной базе данных не обращается и работает только на центральный процессор.

TQueryStatus – статус запроса, может принимать следующие значения: *qsNoProcessing, qsCPU, qsRAM, qsHDD, qsNet, qsOutput, qsSwap, qsLVS.*

TKindResource – вид ресурса, может принимать следующие значения [1,2]:

krCPU = 0 – ЦПУ,

krRAM = 1 – ОЗУ,

krHDD = 2 – жесткий диск,

krNet = 3 – сеть,

krOutput = 4 – устройство вывода.

TQuery (q) – запрос, определяется следующими параметрами:

Ncpu – количество операций,

Qmem – объем данных в ОЗУ,

Qsize – объем данных на жестком диске,

Qnet – объем данных, передаваемых по сети,
Tout – время вывода данных,
inRAM – взаимодействие данных с ОЗУ:
0 – HDD,
1 – ОЗУ,
2 – выгрузка в ОЗУ;
status– положение запроса,
number – порядковый номер запроса,
timeCreate – время создания.

Рассмотренные выше характеристики запроса зависят от типа запроса. Для запросов каждого вида отдельно задаются средние значения и отклонения объемов используемых ресурсов. Конкретные значения параметров запросов определяются во время выполнения с учетом заданного закона распределения.

TProbability – вероятность возникновения запросов определенного типа:
MVV – вероятность возникновения запроса модуля ввода информации,
MOB – вероятность возникновения запроса модуля обработки информации,
MKO – вероятность возникновения запроса модуля корректировки информации,
MYD – вероятность возникновения запроса модуля удаления информации,
MOT – вероятность возникновения запроса модуля формирования отчета,
MPE – вероятность возникновения запроса модуля печати информации,
MCch – вероятность возникновения запроса модуля проведения математических расчетов.
Core – вероятность возникновения запроса о переходе на обработку процессором,
HDD – вероятность возникновения запроса о переходе на обработку жестким диском,
Output – вероятность возникновения запроса о переходе на обработку устройством вывода,
Net – вероятность возникновения запроса о переходе на обработку сетевым адаптером.

TResourceStatistic – статистика использования ресурса, определяется следующими параметрами:
totalLoad – время полезного функционирования – сумма периодов времени активной работы ресурса, секунды,
totalProcessing – время работы модели, секунды. Вместе с *totalLoad* позволяет определить долю полезно используемого времени.
countQuery – количество обработанных ресурсом запросов.

TCore – ядро (состоит из ЦПУ и ОЗУ), определяется следующими параметрами:
CPUspeed – скорость процессора,
TimeSlice – квант времени,
RAMspeed – скорость чтения/записи ОЗУ, МБ/с,
RAMsize – объем ОЗУ,
RAMused – количество уже занятой оперативной памяти,
info – статистика по ресурсу.

THDD – жесткий диск, определяется следующими параметрами:
speed – скорость чтения/записи жесткого диска,

size – объем жесткого диска,
TimeSlice – квант времени,
info – статистика по ресурсу.

TOutput – устройство вывода, определяется следующими параметрами:
info – статистика по ресурсу.

TNet – сетевой адаптер, определяется следующими параметрами:
speed – скорость передачи данных (по умолчанию 10),
TimeSlice – квант времени,

info – статистика по ресурсу.

3. Формализация основных компонентов модели

CreateQuery – процесс, который моделирует создание запроса со значениями занимаемых ресурсов, характерными для данного вида запроса. При этом сначала определяется тип генерируемого запроса на основании данных о вероятностях возникновения запросов разных видов:

$$p2\begin{cases} > p1, k = \text{qrMCch}; \\ \le p1,\ p2 += MKO \begin{cases} > p1, k = \text{qrMKO}; \\ \le p1,\ p2 += MOB \begin{cases} > p1, k = \text{qrMOB}; \\ \le p1,\ p2 += MOT \begin{cases} > p1, k = \text{qrMOT}; \\ \le p1, p2 += MPE \begin{cases} > p1, k = \text{qrMPE}; \\ \le p1, p2 += MVV \begin{cases} > p1, k = \text{qrMVV}; \\ \le \text{p1,k=qrMYD}. \end{cases} \end{cases} \end{cases} \end{cases} \end{cases}$$

Далее определяются параметры запроса *q (Ncpu, Qmem, Qsize, Qnet, Tout)* на основании известных характеристик распределения для выбранного на предыдущем этапе вида запроса.

CQueryProcessing – процесс, моделирующий обработку запроса на узле. Находясь в очереди, запрос ожидает обслуживания, пока не произойдет полное завершение обслуживания; затем определяется, нуждается ли запрос в обработке, и отправляется к выбранному следующим образом ресурсу:

$$\text{p1 =Positive,p2=Core}\begin{cases}\geq p1 and Ncpu>0, core\_push(q), status=qsCPU;\\ p2+=HDD\begin{cases}\geq p1 and Qsize>0, hdd\_push(q), status=qsHDD;\\ p2+=Net\begin{cases}\geq p1 and Qnet>0, net\_push(q), status=qsNet;\\ p2+=Output\begin{cases}\geq p1 and Tout>0, out\_push(q), status=qsOutput;\\ Ncpu=0 and Qsize=0\\ and Qnet=0 and qTout=0,\begin{bmatrix}пополнить Статистику,\\ удалить Запрос;\end{bmatrix}\end{cases}\end{cases}\end{cases}\text{status=qsNoProcessing.}$$

CCoreProcessing – процесс, моделирующий обработку запроса средствами ЦПУ и ОЗУ в течении одного кванта времени; отражает работу центрального процессора узла.

$$inRAM\begin{cases}>0, q=c-q[i], erase(c\_q[i]), NCpu\begin{cases}>speed*slice,\begin{bmatrix}Ncpu-=speed*slice,\\ c\_push(q),\\ Load+=slice;\end{bmatrix}\\ \leq speed*slice,\begin{bmatrix}status=CSNo\,\text{Pr}oces\sin g,\\ RAM-=Qmem,\\ Ncpu=0,\\ t=Ncpu/(speed*slice),\\ Load+=t,\\ \text{Pr}oc+=time-Temp+t,\\ Temp=t,\\ countQ+=1;\end{bmatrix}\end{cases}\\ =0\begin{bmatrix}c\_q[i].status=qsSwap,\\ hdd\_push(c\_q[i]),\\ inRAM=2;\end{bmatrix}\end{cases} i=\overline{0,size-1},1.$$

После выбора очередного запроса из очереди осуществляется проверка нахождения блоков данных данного запроса в ОЗУ. Если блоки находятся на жестком диске,

то идет обращение к HDD на выгрузку блоков в ОЗУ), а затем данный запрос ставится опять в очередь. Если блоки находятся в ОЗУ, то осуществляется непосредственное выполнение запроса на ЦП. При этом рассчитывается время выполнения запроса и осуществляется постановка запроса в очередь, при этом оставшееся количество операций уменьшается на величину, равную произведению кванта времени на быстродействие центрального процессора. Если обработка произошла за время, меньшее кванта, происходит завершение обработки запроса центральным процессором.

COutputProcessing – процесс, моделирующий обработку запроса устройством вывода. После выбора очередного запроса из очереди происходит выполнение запроса устройством вывода в течение требуемого времени.

$$size\begin{cases}\neq 0,\begin{bmatrix} q = query[0], query.erase(query.begin), \\ status = qsNo\,\mathrm{Pr}\,oces\sin g, countQ+=1, \\ Load+=Tout, \mathrm{Pr}\,oc+=time-timeTemp+Tout, \\ timeTemp=Tout, Tout=0. \end{bmatrix} \\ = 0, возврат; \end{cases}$$

CHDDProcessing – процесс, моделирующий обработку запроса, требующего ресурсов жесткого диска. После выбора очередного запроса из очереди осуществляется выполнение запроса на жестком диске. При этом рассчитывается объем считываемых или записываемых данных в зависимости от оставшегося объема данных запроса: в размере целого кванта, при этом оставшийся объем данных запроса уменьшается на величину, равную кванту записываемых или считываемых данных, и осуществляется постановка запроса в очередь; либо объемом меньше кванта, и происходит завершение обработки запроса на жестком диске.

$$size\begin{cases}\neq 0, q = query[0], query.erase(query.begin), status\begin{cases}=qsSwap,\begin{bmatrix} i=0, t=0, \\ tmp=query[i], i++, tmp\begin{cases}=q, continue \\ \neq q, tmp.inRAM\begin{cases}=1,\begin{bmatrix} t+=Qmem/speed, \\ RAMused-=Qmem, \\ tmp.inRAM=0; \end{bmatrix} & RAMused+Qmem \\ & >RAMsize \end{cases}\end{cases} \\ t+=Qmem/speed, RAMused+=Qmem, status=qsCPU, inRAM=1, Load+=t; \end{bmatrix} \\ \neq qsSwap, Qsize\begin{cases}>speed*slice, Qsize-=speed*slice, push\_back(q), Load+=slice; \\ \leq speed*slice,\begin{bmatrix} t=Qsize/(speed*slice), \\ Load+=t, \mathrm{Pr}\,oc+=time-timeTemp+t, \\ timeTemp=t, countQ+=1, \\ Qsize=0, status=qsNo\,\mathrm{Pr}\,oces\sin g. \end{bmatrix}\end{cases}\end{cases} \\ = 0, возврат; \end{cases}$$

CNetProcessing – процесс, моделирующий обработку запроса сетевым адаптером. После выбора очередного запроса из очереди осуществляется передача запроса сетевому адаптеру. При этом рассчитывается объем передаваемых или принимаемых данных в зависимости от оставшегося объема данных: в течении целого кванта, при этом объем оставшихся данных для передачи уменьшается на величину, равную переданной, и происходит постановка запроса в очередь; либо в течении промежутка времени, меньшего кванта, и происходит завершение обработки запроса.

$$\begin{cases} size == 0, возврат; \\ size \neq 0: \\ q = query[0], query.erase(query.begin), \begin{cases} Qnet > slice*speed: \\ Qnet -= slice*speed, \\ push\_back(q), Load += slice; \end{cases} \\ \qquad\qquad\qquad\qquad\qquad\qquad \begin{cases} Qnet \le slice*speed: \\ t = Qnet/(slice*speed), \\ Load += t, \Pr oc += time - timeTemp + t, \\ timeTemp = t, countQ += 1, \\ status = qsNo\,\Pr oces\sin g, Qnet = 0. \end{cases} \end{cases}$$

Разработанный метод построения имитационной модели ЛВС позволяет задаться параметрами оборудования узла и примерным уровнем его нагрузки, чтобы получить уровень загруженности ресурсов узла. Результаты такого моделирования могут быть применены для рационального выбора параметров узла ЛВС.

Литература

1. *Демиденко О.М., Максимей И.В.* Проектное моделирование вычислительного процесса в локальных вычислительных сетях. - Мн.: Белорусская наука, 2001. - 252 с.
2. *Демиденко О.М.* Технология мониторинга и адаптации вычислительного процесса под рабочую нагрузку на локальную вычислительную сеть / *О.М. Демиденко*. - Мн.: Белорусская наука, 2002. - 193 с.
3. *Максимей И.В.* Имитационное моделирование на ЭВМ. - М.: Радио и связь, 1988. - 232 с.: ил.

Алексейчикова Анна Юрьевна
Студентка электротехнического факультета
Белорусско-Российский университет, г. Могилев
Тел.: +375(295) 46-50-03
E-mail: aleks.anna.ur@gmail.com
Якимов Анатолий Иванович
Доцент кафедры «Автоматизированные системы управления», канд. техн. наук.
Белорусско-Российский Университет, г. Могилев
Тел.: +375(447)16-38-16
E-mail: ykm@tut.by
Захарченков Константин Васильевич
Старший преподаватель кафедры «Автоматизированные системы управления»
Белорусско-Российский Университет, г. Могилев
Тел.: +375(297)46-67-98
E-mail: zaharchenkovkv@mail.ru