Интернет-журнал «Науковедение» ISSN 2223-5167 http://naukovedenie.ru/
Том 7, №2 (2015) http://naukovedenie.ru/index.php?p=vol7-2
URL статьи: http://naukovedenie.ru/PDF/58EVN215.pdf
DOI: 10.15862/58EVN215 (http://dx.doi.org/10.15862/58EVN215)

УДК 004.94

Удалов Алексей Владимирович
ФГБОУ ВПО «Тверской государственный технический университет»
Россия, Тверь[1]
Аспирант кафедры «Электронные вычислительные машины»
E-mail: yduck@yandex.ru

Разработка и оценка эффективности последовательно-параллельного алгоритма продвижения модельного времени имитационной распределённой модели цифровой электронной техники, построенной с использованием D-сетей Петри

[1] г. Тверь, ул. Тамары Ильиной, д. 32А, кв. 80

Аннотация. Сегодня для моделирования устройств цифровой электронной техники наиболее перспективными представляются распределённые имитационные модели, построенные с использованием сетей Петри. Предлагаемые распределённые модели отличаются от классических (монолитных) тем, что состоят из взаимодействующих автономных частей (компонент) не только на этапе построения, но и в ходе имитационного эксперимента и отладки модели.

Проведенные исследования показали, что использование такого подхода к реализации компоненты распределённой модели способствует значительному повышению быстродействия моделей сложных цифровых устройств и эффективности хранения их описания в формате XML. При этом распределённые модели требуют больше оперативной памяти для выполнения имитационного эксперимента.

Применение технологии .NET Remoting и соответствующего алгоритмического обеспечения при построении распределённой системы имитационного моделирования позволяет эффективно реализовать имитационный эксперимент с моделью, компоненты которой размещаются в нескольких приложениях одного компьютера или на разных компьютерах сети. Однако экспериментальные исследования показали, что при этом значительно снижается быстродействие распределённой модели. Поэтому повышение быстродействия распределённой модели, компоненты которой размещены в разных адресных пространствах, является актуальной задачей.

Традиционным способом уменьшения задержек при передаче сообщений по сети является организация параллельной работы отдельных подпрограмм алгоритма работы модели. Разработан параллельный и последовательно-параллельный алгоритмы продвижения модельного времени. Проведены экспериментальные исследования эффективности обоих алгоритмов.

Предложенный последовательно-параллельный алгоритм продвижения модельного времени позволяет значительно повысить быстродействие моделей при размещении их компонент в разных адресных пространствах.

Ключевые слова: монолитные модели; распределённые модели; цифровая электронная техника; сети Петри; быстродействие моделей; компонента распределённой модели; распределённое моделирование; параллелизм; многопоточность; синхронизация потоков.

Ссылка для цитирования этой статьи:

Удалов А.В. Разработка и оценка эффективности последовательно-параллельного алгоритма продвижения модельного времени имитационной распределённой модели цифровой электронной техники, построенной с использованием D-сетей Петри // Интернет-журнал «НАУКОВЕДЕНИЕ» Том 7, №2 (2015) http://naukovedenie.ru/PDF/58EVN215.pdf (доступ свободный). Загл. с экрана. Яз. рус., англ. DOI: 10.15862/58EVN215

Сегодня для моделирования устройств цифровой электронной техники наиболее перспективными представляются распределённые имитационные модели, построенные с использованием сетей Петри. Этот математический аппарат, по сравнению с используемой традиционно теорией автоматов, обладает более богатыми аналитическими возможностями и большей моделирующей мощностью. Предлагаемые распределённые модели отличаются от классических (монолитных) тем, что состоят из взаимодействующих автономных частей (компонент) не только на этапе построения, но и в ходе имитационного эксперимента и отладки модели. Отличительной особенностью таких моделей является подход к реализации её компоненты, которая представляет собой совокупность взаимодействующих посредством обмена сообщениями функционального модуля (монолитной модели) и его внешнего представителя в другой компоненте [1].

Проведенные исследования показали, что использование разработанного подхода к реализации компоненты распределённой модели способствует значительному повышению быстродействия моделей сложных цифровых устройств и эффективности хранения их описания в формате XML. При этом распределённые модели требуют больше оперативной памяти для выполнения имитационного эксперимента [2].

Применение предложенного подхода [3] к использованию технологии .NET Remoting и соответствующего алгоритмического обеспечения при построении распределённой системы имитационного моделирования позволяет эффективно реализовать имитационный эксперимент с моделью, компоненты которой размещаются в нескольких приложениях одного компьютера или на разных компьютерах сети. Однако экспериментальные исследования показали, что быстродействие распределённой модели, компоненты которой размещены в двух приложениях, ниже, чем в едином адресном пространстве. Более того, при размещении компонент модели на разных компьютерах сети её быстродействие снижается ещё более значительно. Поэтому повышение быстродействия распределённой модели, компоненты которой размещены в разных адресных пространствах, является актуальной задачей [2].

Классическим способом уменьшения задержек при передаче сообщений по сети является организация параллельной работы отдельных подпрограмм алгоритма работы модели [4, 5]. Под параллельной работой обычно понимают одновременное (или квазиодновременное, в случае разделения времени одного процессора) выполнение нескольких подпрограмм. Таким образом, устранение ожидания одной подпрограммой завершения другой, не блокирующей её выполнения, подпрограммой, обычно позволяет повысить быстродействие и скрыть задержки при передаче по каналам связи.

Организация параллельной работы требует решения, как минимум, трех задач: декомпозиции алгоритма на ряд параллельно выполняемых подпрограмм, организации их взаимодействия, синхронизации [6]. Для того чтобы выявить возможные подходы к решению этих задач, необходимо тщательно проанализировать принципы функционирования модели, механизмы синхронизации событий её компонент и продвижения модельного времени.

Алгоритм функционирования распределённой модели предусматривает продвижение модельного времени каждой подчинённой компоненты до наступления ближайшего события модели в целом (LBTS – Lower Bound on the Time Stamp). Этот интервал модельного времени определяется как наименьший из интервалов до ближайшего события каждой компоненты распределённой модели.

Следует заметить, что каждая подчиненная компонента, как и модель в целом (т.е. главная компонента), обладает собственным локальным модельным временем. Продвижение модельного времени в главной компоненте происходит только после того, как изменится

модельное время в каждой её подчиненной компоненте. По завершению продвижения модельного времени всех компонент выполняется продвижение глобального модельного времени. Такой принцип управления временем обычно называется консервативным с переменным шагом [7].

Рисунок 1. *Алгоритм функционирования распределенной модели (составлено автором)*

Функционирование модели (рисунок 1) предусматривает две подпрограммы – определения времени наступления ближайшего события и продвижения модельного времени каждой компоненты, в которых взаимодействие с компонентами производится при помощи цикла традиционным образом – последовательно. То есть, сначала выполняется подпрограмма определения времени наступления ближайшего события в одной компоненте и только по ее завершении эта подпрограмма выполняется для следующей компоненты. Аналогично производится и продвижение модельного времени (рисунок 2).

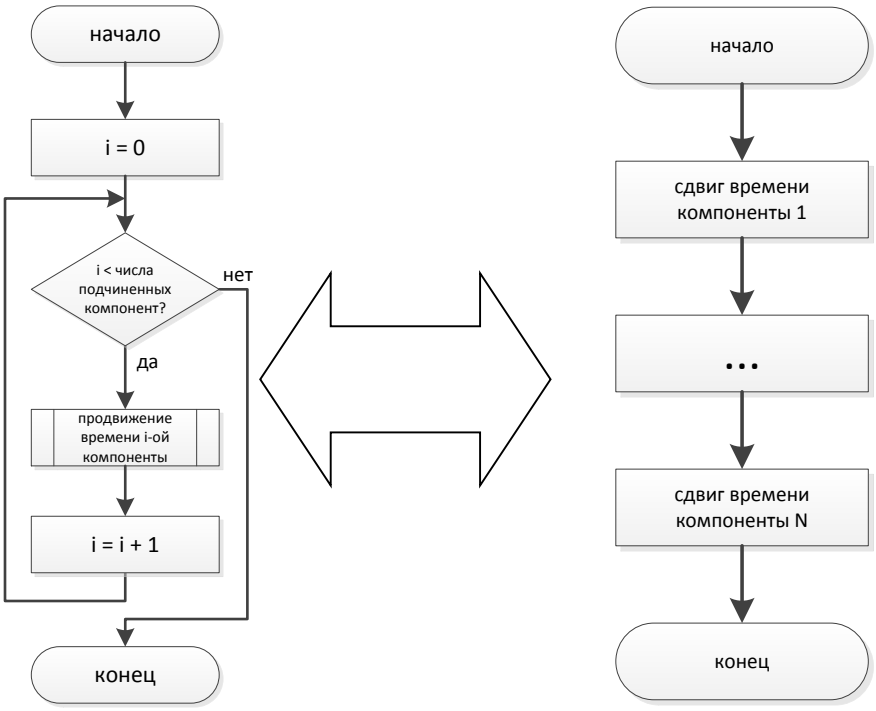

Рисунок 2. *Последовательное выполнение подпрограмм продвижения модельного времени (составлено автором)*

Наиболее простым подходом при решении задачи декомпозиции для организации параллельной работы подпрограмм алгоритма является использование модульной структуры распределённой модели. Иными словами, функционирование распределенной модели может быть представлено как совокупность подпрограмм определения времени наступления ближайшего события и продвижения модельного времени для составляющих её компонент, организовать работу которых можно параллельно. Взаимодействие же между такими подпрограммами в потоках может быть реализовано посредством уже описанного механизма подписки и реакции на происходящие в компонентах распределенной модели события.

Синхронизация подпрограмм алгоритма функционирования использует принцип взаимного исключения и может быть организована аналогично принципу синхронизации событий и продвижения модельного времени распределённой модели. То есть, продвижение глобального времени блокируется до тех пор, пока подпрограммы продвижения модельного времени всех компонент не будут завершены (рисунок 3). Реализация оповещения побочными потоками главного потока приложения об их завершении может быть выполнена посредством реакции на событие или изменения состояния счетчика завершивших работу подпрограмм, расположенного в общей для потоков выполнения памяти. Подпрограмма синхронизации выполняется каждый раз при завершении очередного потока и должна быть использована одновременно не более чем одним потоком выполнения. Другие завершившиеся потоки в таком случае будут ожидать завершения подпрограммы синхронизации и осуществят её вызов только после этого [8].

Программная реализация описанного подхода подразумевает запуск каждой подпрограммы получения времени ближайшего события (или продвижения модельного времени) в отдельно создаваемом потоке выполнения. Кроме того, для реализации принципа взаимного исключения потребуется использовать одно из стандартных средств синхронизации потоков, например, критическую секцию.

Следует заметить, что продвижение модельного времени приводит к изменению внутренних состояний компонент распределённой модели. Поэтому в случае возникновения одновременных событий (с одинаковыми временными метками) при параллельной работе подпрограмм продвижения модельного времени возникают состязания (гонки) потоков при изменении значений соответствующих переменных в памяти. Важной особенностью моделирования цифровой электронной техники является адекватное воспроизведение естественного параллелизма происходящих в таких устройствах событий и гонок сигналов. Поэтому такая особенность параллельной работы компонент распределенной модели не является нежелательной и лишь повышает адекватность моделирования.

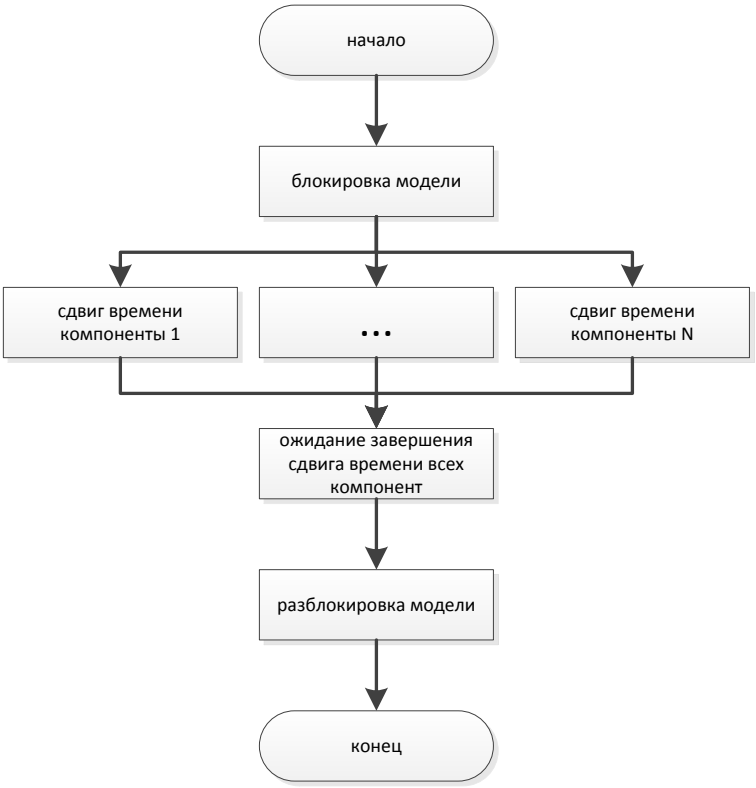

Рисунок 3. *Параллельное выполнение подпрограмм продвижения модельного времени (составлено автором)*

Проведённые в соответствие с методикой [2] экспериментальные исследования быстродействия модели, функционирующей в соответствии с предложенным подходом, позволили получить следующие результаты (рисунок 4 и таблица 1).

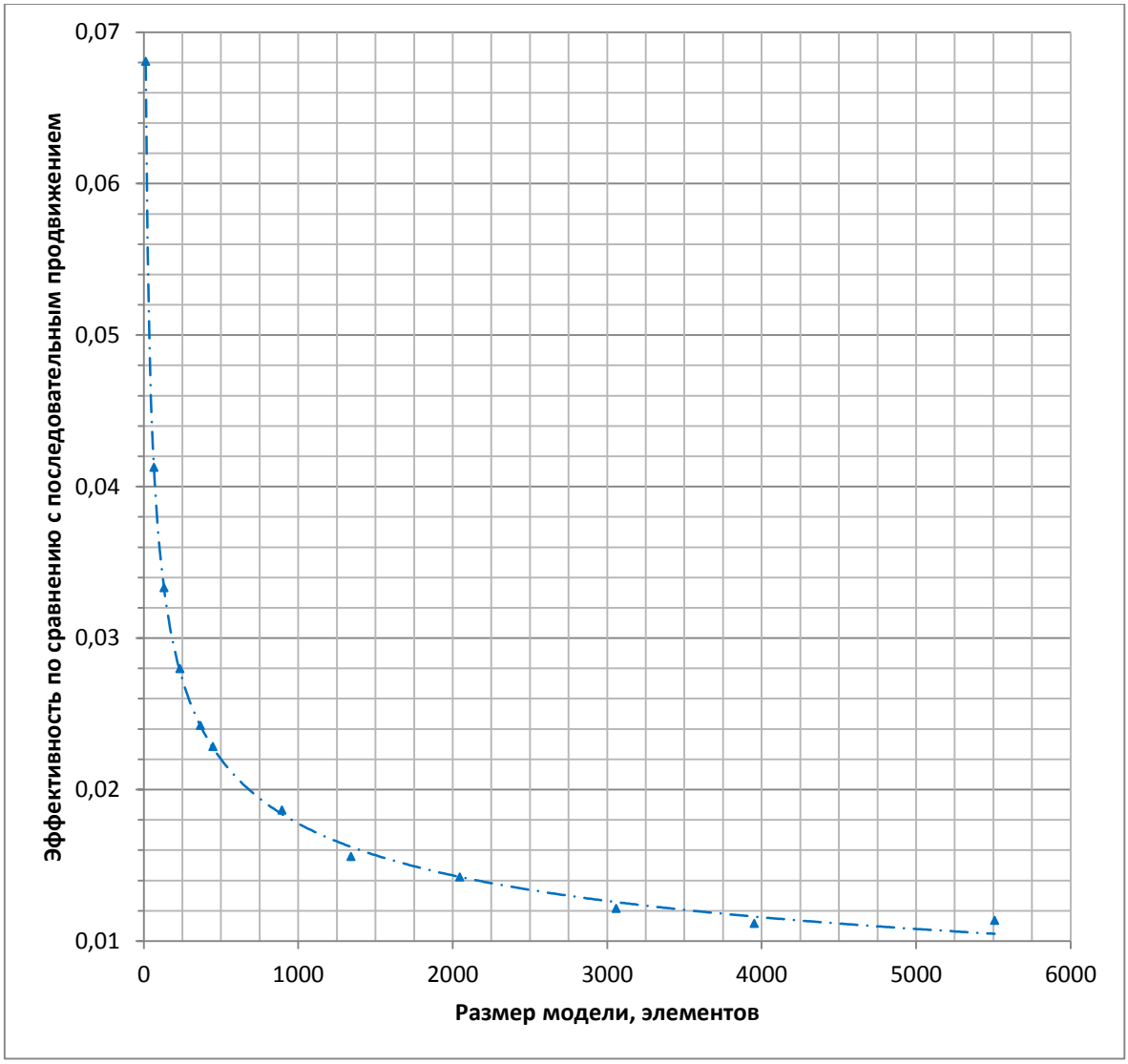

Рисунок 4. *При использовании параллельного продвижения модельного времени быстродействие снижается по сравнению с последовательным (составлено автором)*

Таблица 1

Эффективность использования параллельного продвижения модельного времени по сравнению с последовательным (составлено автором)

Реальный объект (цифровое устройство)	Размер модели, элементов	$E_{быстр.пар}$
К155ЛА3	13	0,0681
К155ТР2	66	0,0413
2хК155ТР2	132	0,0333
К155ТМ2	234	0,0280
К155ТМ2, К155ТР2	366	0,0242
К155ТВ1	447	0,0228
2хК155ТВ1	894	0,0186
3хК155ТВ1	1341	0,0156
К155ИП3	2045	0,0142
К155ИЕ7	3058	0,0122
К155ИЕ7, 2хК155ТВ1	3952	0,7962
К155ИР13	5509	0,8954

Снижение быстродействия модели обусловлено следующими причинами:

- время выполнения подпрограммы продвижения модельного времени каждой компоненты распределенной модели значительно меньше времени, необходимого на выделение необходимых ресурсов и подготовку нового потока к исполнению;
- чрезмерное превышение числа создаваемых потоков над количеством одновременно выполняемых приводит к слишком частому переключению контекста выполнения и значительному времени ожидания неактивных потоков.

С целью устранения влияния этих факторов было пересмотрено решение задачи декомпозиции алгоритма функционирования распределенной модели. Основной идеей модифицированного подхода стало агрегирование подпрограмм продвижения модельного времени компонент распределенной модели в группы, выполняемые параллельно. Число таких групп определяется количеством процессоров (процессорных ядер) компьютера. Подпрограммы, составляющие одну группу, выполняются последовательно (рисунок 5). Такой алгоритм продвижения времени получил название последовательно-параллельного. При его использовании увеличивается время выполнения каждого потока, причем число таких потоков уменьшается до количества максимально возможного для одновременного выполнения, а время ожидания неактивных потоков значительно сокращается. Устранить последнее полностью не представляется возможным вследствие влияния прерывающих контекст выполнения системных процессов и исполняющихся других прикладных программ.

Взаимодействие подпрограмм продвижения модельного времени компонент распределенной модели не претерпело каких-либо значительных изменений. Синхронизация потоков была незначительно доработана посредством организации вызова соответствующей подпрограммы только при завершении работы группы подпрограмм продвижения модельного времени. Это позволило снизить количество вызовов подпрограммы синхронизации потоков и тем самым снизить общее время выполнения.

Программная реализация была модифицирована с целью уменьшения накладных расходов при работе с потоками следующим образом:

- применение пула потоков, содержащего необходимое количество уже готовых к работе потоков;
- повторное использование ресурсов завершивших работу потоков.

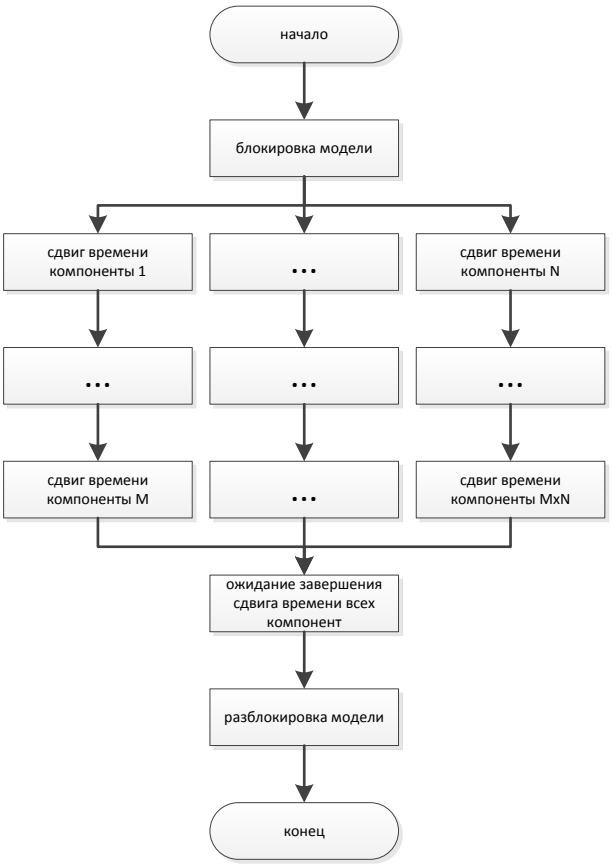

Рисунок 5. *Последовательно-параллельное выполнение подпрограмм продвижения модельного времени (составлено автором)*

Экспериментальная оценка быстродействия распределенной модели, функционирующей с использованием предложенного последовательно-параллельного подхода, показала его более высокую эффективность по сравнению с традиционным (последовательным) способом продвижения модельного времени как в одном и нескольких приложениях одного компьютера, так и на компьютерах в сети (рисунки 6, 7 и таблицы 2, 3). Проведенные исследования позволяют сформулировать следующие выводы о параллельной работе подпрограмм и её использовании при моделировании цифровой электронной техники при помощи распределённых имитационных моделей.

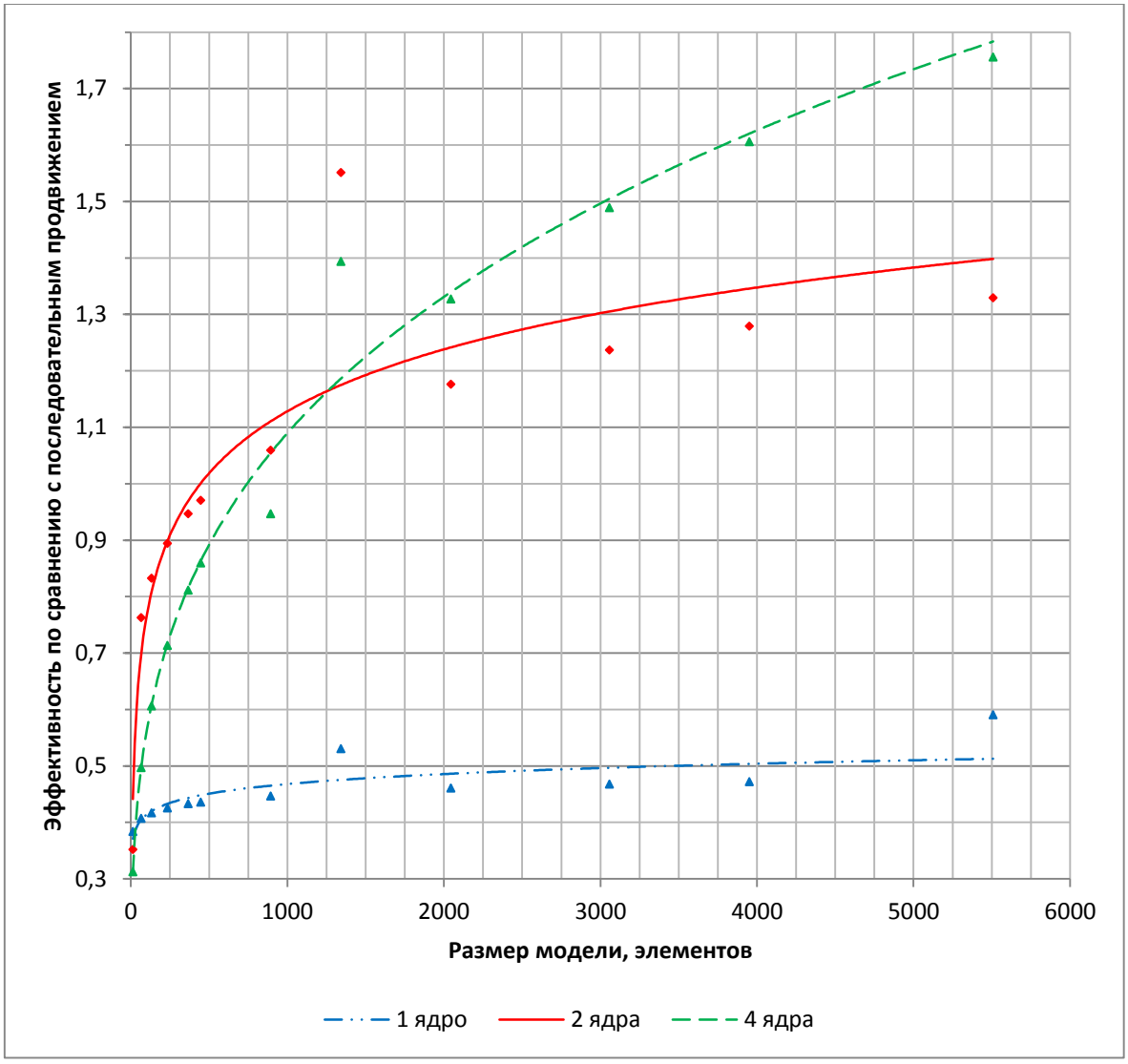

Рисунок 6. *При использовании последовательно-параллельного продвижения модельного времени быстродействие значительно повышается по сравнению с последовательным (составлено автором)*

Таблица 2

Эффективность использования последовательно-параллельного продвижения модельного времени по сравнению с последовательным (составлено автором)

Реальный объект (цифровое устройство)	Размер модели, элементов	$E_{быстр.пос-пар-1}$	$E_{быстр.пос-пар-2}$	$E_{быстр.пос-пар-4}$
К155ЛА3	13	0,3839	0,3521	0,3124
К155ТР2	66	0,4070	0,7630	0,4972
2хК155ТР2	132	0,4173	0,8326	0,6062
К155ТМ2	234	0,4258	0,8946	0,7138
К155ТМ2, К155ТР2	366	0,4332	0,9468	0,8117
К155ТВ1	447	0,4359	0,9708	0,8596
2хК155ТВ1	894	0,4467	1,0593	0,9471
3хК155ТВ1	1341	0,5305	1,5516	1,3941
К155ИП3	2045	0,4609	1,1762	1,3274
К155ИЕ7	3058	0,4681	1,2371	1,4894
К155ИЕ7, 2хК155ТВ1	3952	0,4721	1,2793	1,6061
К155ИР13	5509	0,5909	1,3295	1,7557

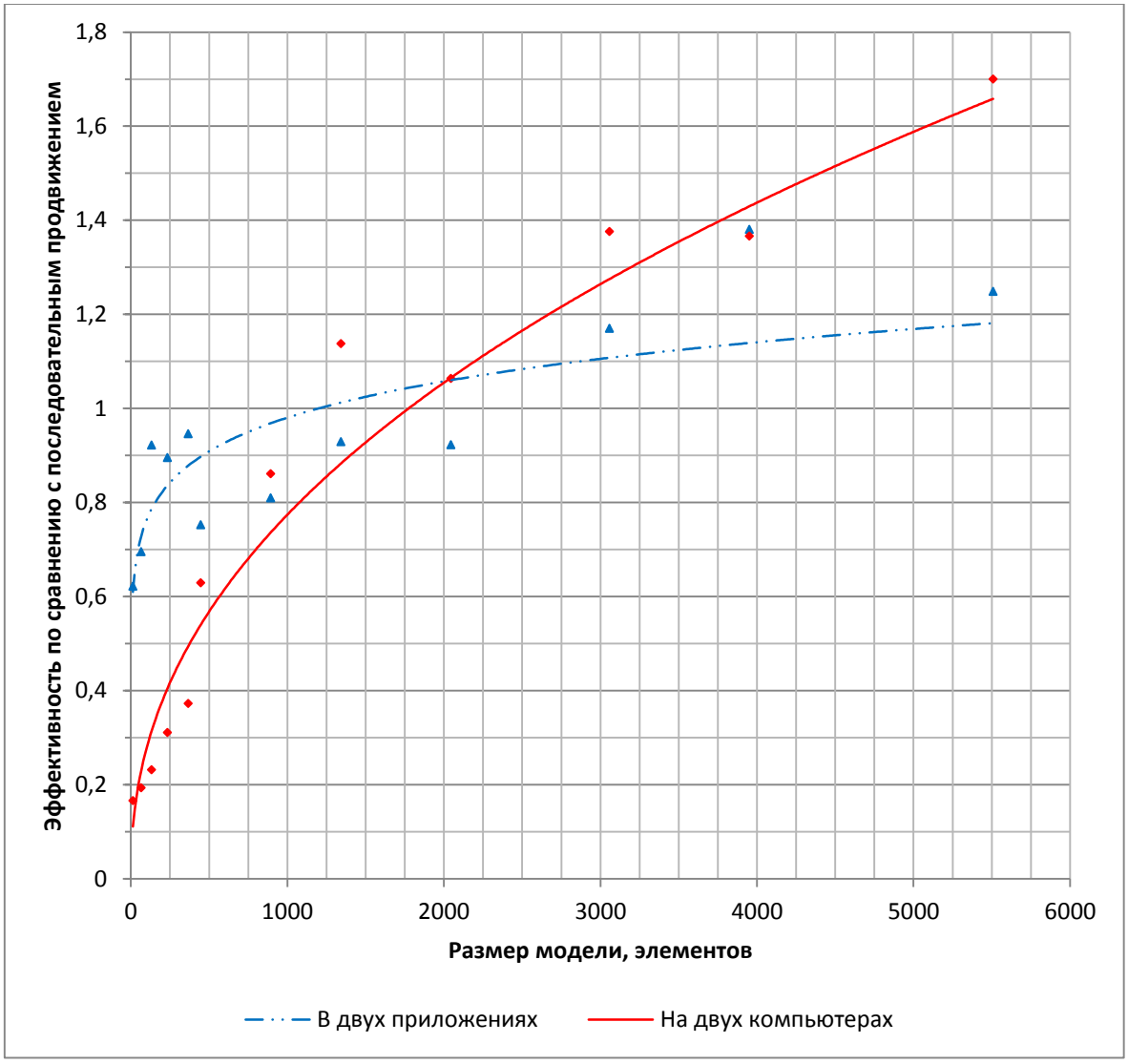

Рисунок 7. *Использование последовательно-параллельного продвижения модельного времени позволяет повысить быстродействие модели и при размещении её компонент в разных адресных пространствах (составлено автором)*

Таблица 3

Эффективность использования последовательно-параллельного продвижения модельного времени по сравнению с последовательным при размещении её компонент в разных адресных пространствах (составлено автором)

Реальный объект (цифровое устройство)	Размер модели, элементов	$E_{быстр.пос-пар-ДП}$	$E_{быстр.пос-пар-ДК}$
К155ЛА3	13	0,6221	0,1667
К155ТР2	66	0,6957	0,1935
2хК155ТР2	132	0,9224	0,2317
К155ТМ2	234	0,8954	0,3111
К155ТМ2, К155ТР2	366	0,9464	0,3730
К155ТВ1	447	0,7530	0,6292
2хК155ТВ1	894	0,8098	0,8615
3хК155ТВ1	1341	0,9292	1,1382
К155ИП3	2045	0,9229	1,0641
К155ИЕ7	3058	1,1702	1,3764
К155ИЕ7, 2хК155ТВ1	3952	1,3810	1,3665
К155ИР13	5509	1,2492	1,7005

Во-первых, последовательно работающие подпрограммы, синхронизируемые общим контекстом посредством виртуального (в случае одного компьютера) или сетевого соединения, обладают значительно меньшим быстродействием по сравнению с последовательной их работой в едином адресном пространстве и тем более меньшим быстродействием по сравнению с их параллельной работой. Более низкое быстродействие связано не только с накладными расходами на формирование и разбор пакетов для обеспечения взаимодействия подпрограмм посредством передачи сообщений, но и ожиданием одной подпрограммой завершения другой, не блокирующей её выполнение. Еще более значительные задержки вызваны передачей сформированных пакетов по каналам связи [9].

Кроме того, увеличение количества компьютеров, на которых функционируют подпрограммы, синхронизируемые общим контекстом выполнения посредством передачи сообщений по сети, приведет к еще более значительному снижению быстродействия из-за роста числа сетевых соединений, каждое из которых несет в себе упомянутые задержки.

Иногда накладные расходы первого типа могут быть снижены в случае использования какого-либо другого способа взаимодействия между подпрограммами (их потоками), например, общей памяти или именованных каналов. Однако при разработке цифровой электронной техники, как правило, не возникает необходимости в размещении компонент её модели в разных приложениях одного компьютера, поскольку использование такого подхода не улучшит быстродействие и не снизит информационную нагрузку по сравнению с работой модели в одном приложении. Поэтому использование такого режима работы нецелесообразно. Несмотря на это, использование виртуального соединения как производной унифицированного взаимодействия помогает обеспечить этот режим работы [10].

Во-вторых, в едином адресном пространстве параллельная работа подпрограмм не всегда более эффективна по сравнению с последовательной. Как правило, быстродействие

зависит в значительной степени от количества и времени выполнения таких подпрограмм (потоков), а также числа процессоров (процессорных ядер) компьютера. В меньшей степени влияние оказывает конкретная программная реализация параллельной работы. Например, параллельная работа большого числа подпрограмм при коротком интервале времени их исполнения обычно отрицательно влияет на быстродействие.

В-третьих, быстродействие параллельно работающих подпрограмм, как правило, ниже, чем функционирующих последовательно, при наличии одного процессора (процессорного ядра) даже с использованием технологий виртуальной многопоточности (например, Hyper Threading). Это связано с тем, что в этом случае квазипараллельная работа реализуется через разделение времени одного процессора. Степень влияния технологий виртуальной многопоточности на быстродействие зависит от особенностей их реализации и выполняемой задачи и может носить как положительный, так и отрицательный характер.

В-четвертых, влияние параллельной работы на быстродействие значительным образом зависит от множества факторов, таких как:

- технические характеристики используемого в компьютере процессора и его архитектуры;
- версия операционной системы и планировщика процессов и потоков;
- настройки BIOS и операционной системы;
- номенклатура запущенных фоновых процессов и приложений и степень их мгновенного использования ресурсов системы;
- организация, пропускная способность и загруженность каналов связи;
- соотношение технических характеристик компьютеров, на которых выполняются подпрограммы.

Выводы по использованию последовательно-параллельного алгоритма продвижения модельного времени компонент распределённой модели при моделировании цифровой электронной техники можно сформулировать следующим образом.

Во-первых, быстродействие моделей малой сложности (до 500 элементов для 2-ядерного процессора и до 750 элементов для 4-ядерного) наиболее высоко при использовании последовательного алгоритма продвижения модельного времени распределённой модели (традиционного подхода).

Во-вторых, функционирование распределенных моделей средней и высокой сложности характеризуется наилучшим быстродействием при использовании последовательно-параллельного алгоритма продвижения модельного времени. Причем для работы с моделями средней сложности рекомендуется использовать компьютеры с небольшим числом процессоров (процессорных ядер). Увеличение числа процессоров непрерывно повышает быстродействие сложных моделей.

Таким образом, разработанный автором подход позволяет снизить влияние задержек при формировании, разборе и передаче сообщений по сети, а также значительно повысить быстродействие распределённых моделей сложной цифровой электронной техники при размещении их компонент, как в сети, так и на одном компьютере. Полученные автором результаты экспериментальных и теоретических исследований можно использовать как для решения задач моделирования, так и других задач, требующих организации параллельного и распределённого взаимодействия.

ЛИТЕРАТУРА

1. Применение распределенных моделей для повышения эффективности САПР цифровых электронных устройств. / А.В. Удалов, А.А. Веселов // Тверь: ТвГТУ. Сборник научных трудов магистрантов и аспирантов. 2012. №1. С. 30-33.
2. Удалов А.В. Экспериментальное исследование характеристик распределённых моделей цифровой электронной техники, построенных с использованием D-сетей Петри // Интернет-журнал «Науковедение», 2015 Том 7 №1 [Электронный ресурс] - М.: Науковедение, 2015. - Режим доступа: http://naukovedenie.ru/PDF/76TVN115.pdf, свободный. - Загл. с экрана. - Яз. рус., англ.
3. Удалов А.В. Использование технологий .NET FRAMEWORK для реализации сетевого взаимодействия компонент распределенной модели цифровой электронной техники / М.: Научтехлитиздат. Приборы и системы. Управление, контроль, диагностика. 2014. №8. С. 10-14.
4. Tanenbaum, A. Distributed Systems: Principles and Paradigms. Second Edition. / A. Tanenbaum, M. Steen. Prentice Hall, 2006. 704 p.
5. Танненбаум Э. Современные операционные системы. 3-е изд.: пер. с англ. СПб: Питер, 2010. 1120 с.
6. Хьюз К. Параллельное и распределённое программирование с использованием C++.: пер. с англ. М.: Изд. дом «Вильямс», 2004. 672 с.
7. Окольнишников В.В. Представление времени в имитационном моделировании // Вычисл. технологии. – 2005. – Т. 10, № 5. – С. 57 – 80.
8. Эндрюс Г. Основы многопоточного, параллельного и распределённого программирования: пер. с англ. М.: Изд. дом «Вильямс», 2003. 512 с.
9. Олифер В.Г., Олифер Н.А. Компьютерные сети. Принципы, технологии, протоколы. 4-е изд. СПб: Питер, 2010. 944 с.
10. Кровчик Э. Сетевое программирование в .NET для профессионалов: пер. с англ. М.: Изд. дом «Лори», 2005. 417 с.

Статья рецензирована членами редколлегии журнала.

Udalov Aleksey Vladimirovich
Federal national budgetary educational institution of higher professional education
Tver State Technical University
Russia, Tver
E-mail: yduck@yandex.ru

Development and a efficiency evaluation of successively-parallel algorithm of modeling time promotion of digital electronic equipment distributed simulation model, constructed using D-Petri Nets

Abstract. Today a distributed simulation models, constructed using Petri Nets, are the most promising for the simulation of digital electronic devices. Proposed distributed model differs from the classical (monolithic) in that consists of interacting autonomous parts (components) not only at the stage of design, but also during the simulation and debug model.

Examinations have shown that the use of this approach to the implementation of a distributed model component contributes significantly to increasing the performance of models of complex a digital electronic equipment and the storage efficiency of their description in XML. Besides distributed models require more RAM to run a simulation.

The use of .NET Remoting technology and related algorithmic support at development of a distributed simulation system allows effectively implement simulations with the models, components of which are located in different applications of one computer or on different computers on the network. However, experimental studies have shown that this significantly decreases the performance of distributed model. Therefore, increasing the performance of the distributed model, the components of which are located in different address spaces, is an vital task.

The traditional way of hiding delays in the transmission of messages over the network is the organization of parallel operation of individual routines of algorithm of the model simulation. A parallel and a successively-parallel algorithms of model time promotion are developed. Experimental studies of the efficiency of both algorithms are carried out.

Proposed successively-parallel algorithm of model time promotion allows significantly improve the performance of the models, components of which are placed in different address spaces.

Keywords: monolithic models; distributed models; digital electronic equipment; Petri nets; models performance; distributed model component; distributed modeling; parallelism; multithreading; threads synchronization.

REFERENCES

1. Primenenie raspredelennykh modeley dlya povysheniya effektivnosti SAPR tsifrovykh elektronnykh ustroystv. / A.V. Udalov, A.A. Veselov // Tver': TvGTU. Sbornik nauchnykh trudov magistrantov i aspirantov. 2012. №1. S. 30-33.
2. Udalov A.V. Eksperimental'noe issledovanie kharakteristik raspredelennykh modeley tsifrovoy elektronnoy tekhniki, postroennykh s ispol'zovaniem D-setey Petri // Internet-zhurnal «Naukovedenie», 2015 Tom 7 №1 [Elektronnyy resurs] - M.: Naukovedenie, 2015. - Rezhim dostupa: http://naukovedenie.ru/PDF/76TVN115.pdf, svobodnyy. - Zagl. s ekrana. - Yaz. rus., angl.
3. Udalov A.V. Ispol'zovanie tekhnologiy .NET FRAMEWORK dlya realizatsii setevogo vzaimodeystviya komponent raspredelennoy modeli tsifrovoy elektronnoy tekhniki / M.: Nauchtekhlitizdat. Pribory i sistemy. Upravlenie, kontrol', diagnostika. 2014. №8. S. 10-14.
4. Tanenbaum, A. Distributed Systems: Principles and Paradigms. Second Edition. / A. Tanenbaum, M. Steen. Prentice Hall, 2006. 704 p.
5. Tannenbaum E. Sovremennye operatsionnye sistemy. 3-e izd.: per. s angl. SPb: Piter, 2010. 1120 s.
6. Kh'yuz K. Parallel'noe i raspredelennoe programmirovanie s ispol'zovaniem C++.: per. s angl. M.: Izd. dom «Vil'yams», 2004. 672 s.
7. Okol'nishnikov V.V. Predstavlenie vremeni v imitatsionnom modelirovanii // Vychisl. tekhnologii. – 2005. – T. 10, № 5. – S. 57 – 80.
8. Endryus G. Osnovy mnogopotochnogo, parallel'nogo i raspredelennogo programmirovaniya: per. s angl. M.: Izd. dom «Vil'yams», 2003. 512 s.
9. Olifer V.G., Olifer N.A. Komp'yuternye seti. Printsipy, tekhnologii, protokoly. 4-e izd. SPb: Piter, 2010. 944 s.
10. Krovchik E. Setevoe programmirovanie v .NET dlya professionalov: per. s angl. M.: Izd. dom «Lori», 2005. 417 s.