Математическая модель динамического алгоритма продвижения времени для распределенного логического моделирования цифровых систем

Ладыженский Ю.В., Тесленко Г.А.
Донецкий национальный технический университет
ly@cs.dgtu.donetsk.ua

Abstract

Ladyzhensky Y.V., Teslenko G.A. A mathematical model of dynamic algorithm of time advancement for the distributed logic simulation of digital systems. *A mathematical model of dynamic algorithm of time advancement is proposed. Experimental investigation results of the mathematical model are discussed.*

Введение

Современные методы верификации цифровых систем с использованием моделирования характеризуются большими временными затратами и последовательным способом выполнения операций.

Использование алгоритмов параллельного и распределенного вычислений [1,2] является перспективным направлениям для ускорения моделирования. Создание на их основе эффективных по быстродействию параллельных и распределенных программных и аппаратных средств моделирования цифровых устройств является актуальной научно-технической проблемой.

Целью данной работы является разработка математической модели динамического алгоритма продвижения времени для распределенного логического моделирования, сравнение его с другими алгоритмами продвижения времени.

Распределенное логическое моделирование

Имитационное моделирование является важным инструментом при верификации проектов цифровых систем. Оно используется для функционального и временного моделирования проекта и проводится с целью выявления ошибок, возникающих на стадиях проектирования.

Распределенная система моделирования состоит из процессов, которые взаимодействуют друг с другом путем передачи сообщений [2]. Каждый процесс моделирует часть физической системы и представляет собой логический процесс (ЛП). В течение времени моделирования ЛП генерируют события, посылают их другим логическим процессам и получают события от других процессов. Распределенное моделирование считается корректным, если каждый логический процесс обрабатывает события в порядке возрастания значений их временных меток. Существует два типа алгоритмов синхронизации вычислений, консервативные и оптимистические, которые широко используются при распределенном моделировании.

Распределенная система может состоять из рабочих станций, объединенных в локальную сеть (рис.1). Исходная модель цифровой системы на языке аппаратного описания преобразуется компилятором в список соединений.

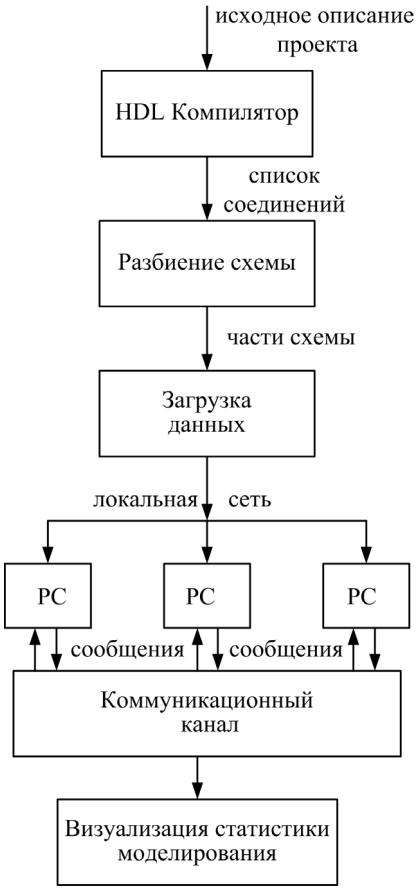

Рисунок 1 - Структура распределенной системы моделирования

Полученный список затем используется блоком разбиения для разрезания исходного графа схемы на подграфы. В результате разрезания полученные части схемы распределяются для моделирования между специализированными рабочими станциями в сети (РС).

Построение математической модели

Рассматриваемая далее математическая модель для динамического алгоритма продвижения времени является обобщением модели, предложенной в [3].

Пусть в распределенной системе есть N взаимодействующих процессов, которые выполняются на N процессорах. За системой ведется наблюдение в дискретные моменты времени $i=1,2,..n$, которые интерпретируются как шаги развития модели. В момент i состояние системы определяется значением X_i, характеризующим текущее модельное время. Значение X_i может быть скаляром в случае единого модельное время, либо вектором, когда каждый процесс имеет свои собственные локальные часы. На каждом шаге значение X_i изменяется в соответствии с выбранным алгоритмом продвижения времени. На любом шаге процессы могут обмениваться сообщениями, *m-й* процесс на *i-ом* шаге посылает сообщение *n-му* процессу с вероятностью $\alpha_{mn}^{i}(m,n=\overline{1,N})$. Событие посылки или приема сообщения считается мгновенным. Приращение локальных часов (затраты на внутреннюю работу) на каждом шаге характеризуется величиной $\xi^n, n=\overline{1,N}.$ Перечисленные параметры характеризуют имитационную модель, выбранный алгоритм синхронизации (консервативный, оптимистический или динамический) определяет способ продвижения модельного времени.

Будем считать, что на шаге с номером i завершается процесс имитационного моделирования. Для определения времени, которое система может достичь за i шагов, введем следующую характеристику

$$T(i,M(X_i))=\max(M(x_i^1),M(x_i^2),\ldots,M(x_i^N)), \qquad (1)$$

где i – номер шага, для которого определяется значение модельного времени;

$M(x_i^n)$, $n=\overline{1,N}$ – математическое ожидание модельного времени n-го процесса на i-м шаге моделирования.

Шаг i необходимо выбирать достаточно большим, чтобы алгоритм продвижения времени успел продемонстрировать свои свойства. Критерием эффективности при сравнении различных алгоритмов является конечное значение модельного времени. Лучшим считается тот процесс, который за определенное количество шагов способен продвинуть модельное время до большего значения.

Особенности динамического алгоритма для распределенного моделирования

Особенностью динамического протокола, используемого в системе моделирования, является возможность переключения от консервативного поведения к оптимистическому и обратно в процессе моделирования. Логический процесс автоматически изменят свой тип на консервативный в случае возникновения частых откатов, либо на оптимистический в случае частых блокировок при моделировании [2].

Переключение в оптимистический режим синхронизации выполняется путем простого изменения типа логического процесса. Переключение в консервативный режим предполагает в начале использование промежуточного режима работы, в котором обрабатываются только безопасные и антисообщения. Так как при оптимистической синхронизации в списке состояний могут находиться события, которые потенциально могут привести к откату в других логических процессах, то перед переключением необходимо гарантировать корректное состояние логического процесса. В промежуточном режиме выполняется откат ошибочных событий с помощью антисообщений, новые состояния не сохраняются в очередь, т.к. обрабатываются только безопасные события. В результате очередь состояний очищается, что гарантирует корректное состояние логического процесса перед его переключением в консервативный режим.

Программный метод интерпретации алгоритмов не является единственным способом решения задач. Можно применять структурную реализацию алгоритма. Такая реализация заключается в создании из набора функциональных блоков и связей между ними структурной модели, обеспечивающей эквивалентное отображение схемы алгоритма и процесса обработки данных. Алгоритм реализуется за счет продвижения данных и их преобразования по путям обработки (асинхронно или при тактировании) от входов к выходам схемы, на которых получается результат. Команды для выполнения отдельных операций могут отсутствовать. Это обеспечивает распределение вычислительного процесса не только во времени, но и в пространстве. Естественным образом достигается высокая степень параллелизма. Необходимый набор функциональных блоков можно реализовать на микросхемах программируемой логики, которые

в настоящее время широко применяются для построения цифровых систем.

Структурная реализация динамического протокола синхронизации в виде специализированного процессора для распределенного моделирования рассматривается в [4].

Модель динамического алгоритма продвижения времени

В консервативном алгоритме продвижение модельного времени для процесса возможно только в том случае, если гарантировано отсутствие сообщений с меньшим значением модельного времени [5]. Для любого *n*-го процесса определим множество $M^n \subset \{1,2,...N\}$ процессов, которые могут послать ему сообщение. Тогда функцию $S^n(i)$ консервативного продвижения времени на *n-м* процессе для *i-го* шага можно описать следующим уравнением (2):

$$S^n(i)=\begin{cases}\xi_i^n, \ \textit{если} \ \forall m\in M^n : x_i^n+\xi_i^n \le x_i^m+\xi_i^m \\ 0, \ \textit{иначе}\end{cases} \quad (2)$$

Выражение (2) определяет величину, на которую *n-й* процесс может продвинуть свое модельное временя в соответствии с консервативным алгоритмом: продвижение на величину ξ_i^n для процесса *n* возможно только если гарантировано, что от процесса *m* не будет получено сообщения с меньшим модельным временем $x_i^m+\xi_i^m$.

В оптимистическом алгоритме допускается существование отстающих сообщений [5], в этом случае процесс должен выполнить откат до времени полученного сообщения. Поскольку процесс может получить несколько отстающих сообщений, то необходимо выполнить откат до минимального из них. Независимо от того, выполнялся ли откат, или нет, процесс продвигает свои часы на величину $\xi^n, n=\overline{1,N}.$

Для возможности организации отката в оптимистическом алгоритме необходимо выполнять сохранение состояний процессов с некоторым периодом $\rho^n, n=\overline{1,N}$. При этом значение периода ρ^n=1 соответствует классическому оптимистическому алгоритму синхронизации. Пусть $t^n, n=\overline{1,N}$ - затраты на сохранение одного состояния *n*-м процессом, выраженные в модельном времени, тогда функцию $F^n(i)$ оптимистического продвижения модельного времени для *n-го* процесса на *i-м* шага можно описать уравнением (3).

$$\begin{cases} F^n(i)=\xi^n-\max\limits_{m=\overline{1,N}}(\alpha_{mn}^i\nu_i^{mn})-W(i)t^n \\ \nu_i^{mn}=\begin{cases}x_i^n-x_i^m, \ \textit{если} \ \ x_i^n>x_i^m \\ 0, \ \textit{если} \ \ x_i^n\le x_i^m\end{cases} \\ W(i)=\begin{cases}1, \textit{если} \ i \bmod \rho=0 \\ 0, \textit{иначе}\end{cases}\end{cases} \quad (3)$$

Здесь ν_i^{mn} - величина отката, при передаче сообщения от *n-го* к *m-му* процессу, W(i) – функция, определяющая необходимость сохранения состояния на *i-м* шаге имитационного моделирования. Для построения модели динамического алгоритма введем следующие обозначения:

$I_i^{cons\to optim}$ – случайное событие переключения алгоритма синхронизации из консервативного в оптимистический на *i-м* шаге;

$I_i^{optim\to cons}$ – случайное событие переключения алгоритма синхронизации из оптимистического в консервативный *i-м* шаге.

Используя выражения для функций консервативного и оптимистического продвижения времени (2), (3) запишем закон продвижения модельного времени для динамического алгоритма:

$$x_{i+1}^n=\begin{cases}x_i^n+F^n(i)I_i^{cons\to optim}+S^n(i)I_i^{optim\to cons} \\ x_0^n=0 \\ n=\overline{1,N}\end{cases} \quad (4)$$

Для оценки случайных событий $I_i^{optim\to cons}$ и $I_i^{cons\to optim}$ введем соответствующие величины:

$\beta_i, 0\le\beta_i\le 1$ – вероятность переключения оптимистического алгоритма в консервативный на *i-м* шаге для всех процессов системы;

$\gamma_i, 0\le\gamma_i\le 1-$ вероятность переключения консервативного алгоритма в оптимистический на *i-м* шаге для всех процессов системы, причем

$$\gamma_i=1-\beta_i$$

При $\beta_i=1$ модель полностью соответствует консервативному алгоритму, а при $\gamma_i=1$ - оптимистическому.

Получим математические ожидания для уравнений (4):

$$M(x_{i+1}^n)=\begin{cases}M(x_i^n)+F^n(i)\gamma_i+S^n(i)\beta_i \\ M(x_0^n)=0 \\ n=\overline{1,N}\end{cases} \quad (5)$$

Таким образом, на каждом шаге выполняется усреднение величин продвижения времени, полученных согласно оптимистическому и консервативному алгоритмам.

Рассмотрим процесс продвижения времени в системе (4). При работе консервативного алгоритма модельное время всей системы T_i обязательно будет двигаться вперед. Действительно, на консервативном шаге работы алгоритма хотя бы для одного процесса *n* будет выполняться условие $x_i^n + \xi_i^n \le x_i^m + \xi_i^m$. Поэтому, хотя бы одному процессу разрешено приращение локальных часов, и согласно выражению (2) значение T_i будет увеличиваться. При работе оптимистического алгоритма доказать обязательное продвижение модельного времени всей системы невозможно, так как в крайнем случае, если состояния не сохраняются, $\forall n : \rho^n = \infty$, при откате каждый раз будет происходить возврат в начальное состояние $T_0 = 0$.

Таким образом, необходимым условием продвижения времени при использовании динамического алгоритма является сохранение состояний, $\forall n : \rho^n \neq \infty$, при оптимистическом продвижении времени.

Результаты экспериментов

С целью исследования свойств предложенной математической модели и сравнения ее эффективности с классическими алгоритмами были проведены эксперименты. Получены зависимости продвижения модельного времени от количества процессоров, интенсивности обмена сообщениями в системе, вероятности переключения в консервативный алгоритм.

Рассмотрим результаты исследования модели динамического алгоритма продвижения времени. Во всех экспериментах период сохранения состояний ρ^n для всех процессов принят равным 1, т.е. сохранение выполняется на каждом шаге.

Результаты эксперимента 1 показывают модельное время, продвинутое различными алгоритмами за 200 шагов. Исследования проводились для консервативного, оптимистического и динамического алгоритмов (рис. 2).

Были выбраны следующие параметры модели:

количество процессов *N*=3;

приращение модельного времени на процессах определяются вектором ξ^n = (2, 10, 25);

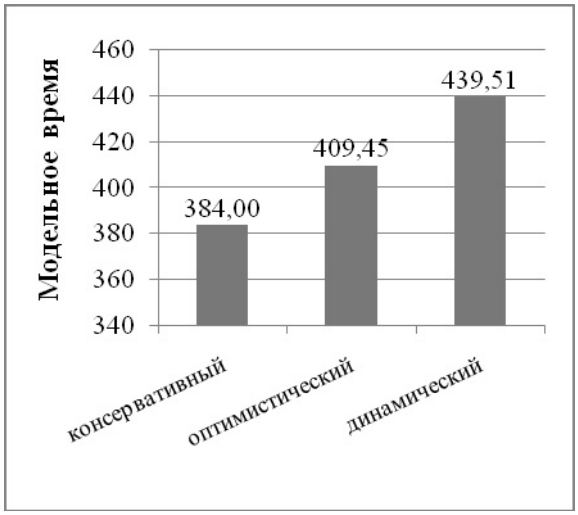

Рисунок 2 – Сравнение модельного времени для различных алгоритмов

затраты на сохранение состояний определяются вектором t^n =(0.5;5;12.5);

вероятности передачи сообщения на *i-м* шаге

$$\alpha_i = \begin{vmatrix} 0 & 0.1 & 0.1 \\ 0.1 & 0 & 0.1 \\ 0.1 & 0.1 & 0 \end{vmatrix}$$

Такая матрица соответствует низкой интенсивности обмена сообщений между процессами.

Для выбора типа алгоритма продвижения времени были заданы величины β_i и γ_i. Значения β_i =1; γ_i =0 соответствуют классическому консервативному алгоритму, значения β_i =0; γ_i =1 – классическому оптимистическому. Для динамического алгоритма были выбраны следующие значения: β_i =0,5; γ_i =0,5. Это означает, что 50% от общего времени имитационного моделирования составляет работа консервативного алгоритма, и 50% – работа оптимистического.

Результаты эксперимента показывают, что при низкой интенсивности обмена сообщений лучшие результаты показывает динамический алгоритм продвижения времени. Результаты динамического алгоритма на 14.3% выше, чем у консервативного и на 7.3% выше, чем у оптимистического. Худший результат по сравнению с остальными показал консервативный алгоритм синхронизации.

Результаты эксперимента 2 показывают модельное время, продвинутое различными алгоритмами за 200 шагов. Как и для первого эксперимента, исследования проводились для

консервативного, оптимистического и динамического алгоритмов (рис. 3).

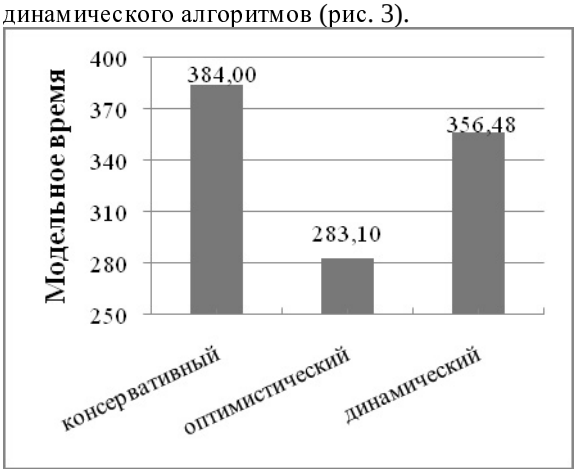

Рисунок 3 – Сравнение модельного времени для различных алгоритмов

Были выбраны следующие параметры модели:

количество процессов *N*=3;

приращение модельного времени на процессах определяются вектором ξ^n = (2, 10, 25);

затраты на сохранение состояний определяются вектором t^n =(0.5;5;12.5);

вероятности передачи сообщения на *i-м* шаге

$$\alpha_i = \begin{vmatrix} 0 & 0.9 & 0.9 \\ 0.9 & 0 & 0.9 \\ 0.9 & 0.9 & 0 \end{vmatrix}$$

Такая матрица соответствует высокой интенсивности обменом сообщений между процессами.

Для динамического алгоритма были выбраны следующие значения вероятностей переключения: β_i =0.5; γ_i =0.5.

По результатам эксперимента видно, что при высокой интенсивности передачи сообщений худший результат, из-за большого количества откатов, показал оптимистический алгоритм. Лучшие результаты показал консервативный алгоритм. Динамический алгоритм оказался на 7.8% хуже чем консервативный, и на 25.8% лучше чем оптимистический.

Результаты эксперимента 3 показывают модельное время, продвинутое различными алгоритмами за 200 шагов в зависимости от количества процессоров. В сравнении участвовали консервативный, оптимистический и динамический алгоритмы (рис. 4).

Входные параметры модели следующие:

количество процессов *N*=2..8;

приращение модельного времени на процессах определяются вектором ξ^n =(2; 10; 25; 15; 25; 10; 30; 25);

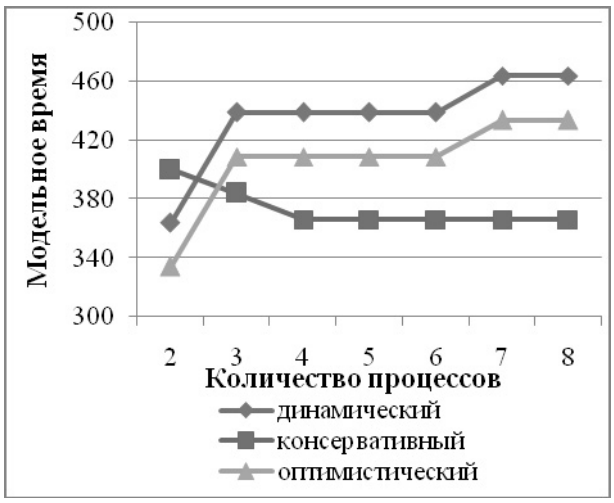

Рисунок 4 – Зависимость модельного времени разных алгоритмов от количества процессов

затраты на сохранение состояний определяются вектором t^n =(0.5; 5; 12.5; 7.5; 12.5; 5; 15; 12.5);

вероятности передачи сообщения на *i-м* шаге

$$\alpha = \begin{vmatrix} 0 & 0.1 & 0.1 \\ 0.1 & 0 & 0.1 \\ 0.1 & 0.1 & 0 \end{vmatrix}$$

Для консервативного алгоритма продвижения времени были заданы величины $\beta_i = 1$ и $\gamma_i = 0.$ Для оптимистического $\beta_i = 0$, $\gamma_i = 1.$ Для динамического алгоритма были выбраны следующие значения: β_i =0.5; γ_i =0.5.

В результате проведения эксперимента установлено, что при использовании двух процессов лучшие результаты показывает консервативный алгоритм. Преимущество консервативного алгоритма над оптимистическим 19.7%, над динамическим 9.9%. На 3-х и более процессах лучшим является динамический алгоритм. Преимущество над консервативным алгоритмом составляет 14.3% – 26.7%, преимущество над оптимистическим составляет 7%.

Результаты эксперимента 4 показывают модельное время, продвинутое различными алгоритмами за 200 шагов для различных значений вероятности переключения β_i (рис. 5).

Были выбраны следующие параметры модели:

количество процессов *N*=3;

приращение модельного времени на процессах ξ^n = (2, 10, 25);

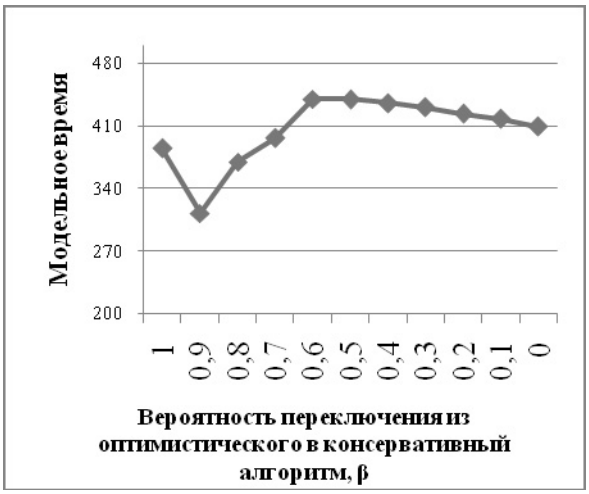

Рисунок 5 – Зависимость модельного времени динамического алгоритма от вероятности переключения

затраты на сохранение состояний t^n =(0.5;5;12.5);

вероятности передачи сообщения на *i-м* шаге

$$\alpha = \begin{vmatrix} 0 & 0.1 & 0.1 \\ 0.1 & 0 & 0.1 \\ 0.1 & 0.1 & 0 \end{vmatrix}$$

Для консервативного алгоритма продвижения времени были заданы величины $\beta_i = 1$ и $\gamma_i = 0.$ Для оптимистического $\beta_i = 0$, $\gamma_i = 1.$ Для динамического алгоритма были выбраны следующие значения: β_i =0.5; γ_i =0.5.

На рис. 5 крайняя левая точка графика соответствует классическому консервативному алгоритму, а крайняя правая точка – классическому оптимистическому.

Анализ результатов эксперимента показывает, что лучшие показатели модельного времени достигаются при значении $\beta_i = (0.5,0.6).$

Эксперимент 5 демонстрирует модельное время, продвинутое различными алгоритмами за 200 шагов в зависимости от количества процессоров. В сравнении участвовали консервативный, оптимистический и динамический алгоритмы (рис. 6).

Входные параметры модели следующие:

количество процессов *N*=2..1000;

приращение модельного времени на процессах определяются вектором ξ^n =(2; 10; 25; 15; 25), *n*=1..5;

для *n*=6..1000 значения ξ^n повторяются с периодом 5.

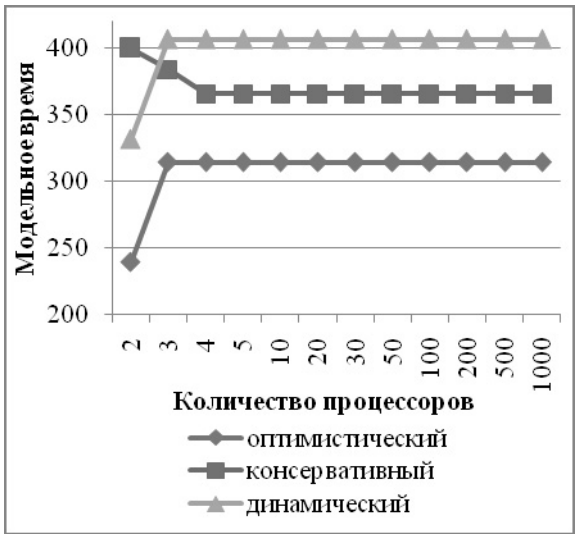

Рисунок 6 – Зависимость модельного времени алгоритмов продвижения времени от количества процессов

затраты на сохранение состояний определяются вектором t^n =(1; 5; 12.5; 7.5; 12.5), *n*=1..5;

для *n*=6..1000 значения t^n повторяются с периодом 5.

вероятности передачи сообщения на *i-м* шаге

$$\alpha_i = \begin{vmatrix} 0 & 0.1 & 0.1 \\ 0.1 & 0 & 0.1 \\ 0.1 & 0.1 & 0 \end{vmatrix}$$

Для консервативного алгоритма продвижения времени были заданы величины $\beta_i = 1$ и $\gamma_i = 0.$ Для оптимистического $\beta_i = 0$, $\gamma_i = 1.$ Для динамического алгоритма были выбраны следующие значения: β_i =0.65; γ_i =0.35.

В результате проведения эксперимента установлено, что при использовании двух процессов лучшие результаты показывает консервативный алгоритм. Преимущество консервативного алгоритма над оптимистическим составило 67.1%, над динамическим – 20.6%. На 3-х и более процессах лучшим является динамический алгоритм. Преимущество над консервативным алгоритмом составляет 5.9% – 11.1%, преимущество над оптимистическим составляет 29.3%.

Эксперимент 6 показывает модельное время, продвинутое различными алгоритмами за 200 шагов для различной интенсивности передачи сообщений в системе моделирования (рис. 7).

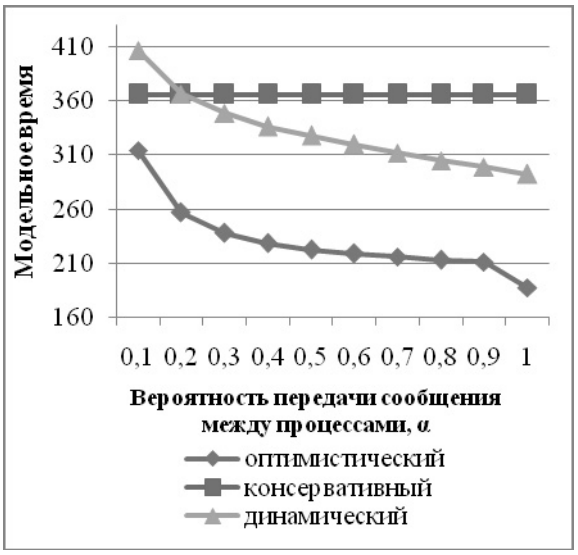

Рисунок 7 – Зависимость модельного времени алгоритмов продвижения времени от интенсивности обмена сообщениями

Были выбраны следующие параметры модели:

количество процессов *N*=4;

приращение модельного времени на процессах определяются вектором ξ^n = (2; 10; 25; 15);

затраты на сохранение состояний определяются вектором t^n =(1;10;25; 7.5);

вероятности передачи сообщения на *i-м* шаге $\alpha_{mn}^i(m,n=\overline{1,N})$ =0.1, 0.2,…1

Для консервативного алгоритма продвижения времени были заданы величины $\beta_i = 1$ и $\gamma_i = 0.$ Для оптимистического $\beta_i = 0$, $\gamma_i = 1.$ Для динамического алгоритма были выбраны следующие значения: β_i =0.65; γ_i =0.65.

Анализ результатов эксперимента показывает, что с увеличением интенсивности обмена сообщениями эффективность оптимистического и динамического алгоритмов уменьшается из-за возрастающих откатов. Эффективность консервативного алгоритма остается неизменной, т.к. математическая модель консервативного алгоритма не учитывает вероятность передачи сообщения между процессами.

Эксперимент 7 показывает модельное время, продвинутое различными алгоритмами за 200 шагов для различной интенсивности сообщений (рис. 8).

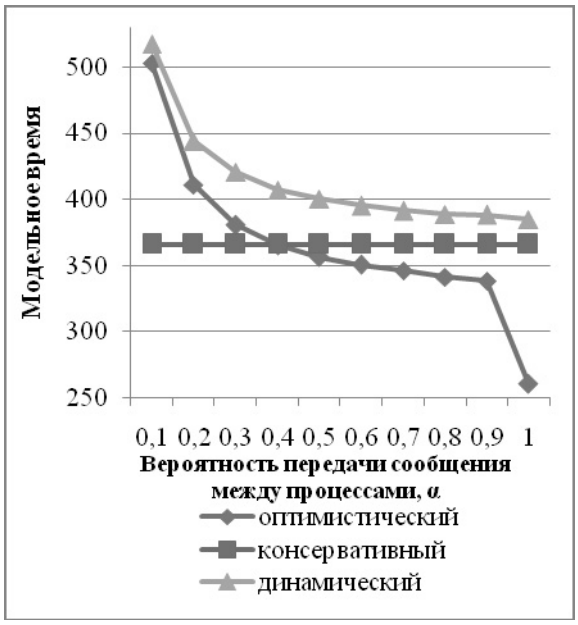

Рисунок 8 – Зависимость модельного времени алгоритмов продвижения времени от интенсивности обмена сообщениями

Были выбраны следующие параметры модели:

количество процессов *N*=4;

приращение модельного времени на процессах определяются вектором ξ^n = (2; 10; 25; 15);

затраты на сохранение состояний определяются вектором t^n =(0.4;2;5; 3);

вероятности передачи сообщения на *i-м* шаге $\alpha_{mn}^i(m,n=\overline{1,N})$ =0.1, 0.2,…1

Для консервативного алгоритма продвижения времени были заданы величины $\beta_i = 1$ и $\gamma_i = 0.$ Для оптимистического $\beta_i = 0$, $\gamma_i = 1.$ Для динамического алгоритма были выбраны следующие значения: β_i =0.65; γ_i =0.35.

Результаты показывают, что уменьшение затрат на сохранение состояний приводит к повышению эффективности динамического и оптимистического алгоритмов.

Эксперимент 8 демонстрирует модельное время, продвинутое различными алгоритмами за 200 шагов в зависимости от количества процессоров. При сравнении рассматривались консервативный, оптимистический и динамический алгоритмы (рис. 9).

Входные параметры модели следующие:

количество процессов *N*=2..1000;

приращение модельного времени на процессах определяются вектором ξ^n =(1; 1; 2; 2; 2), *n*=1..5;

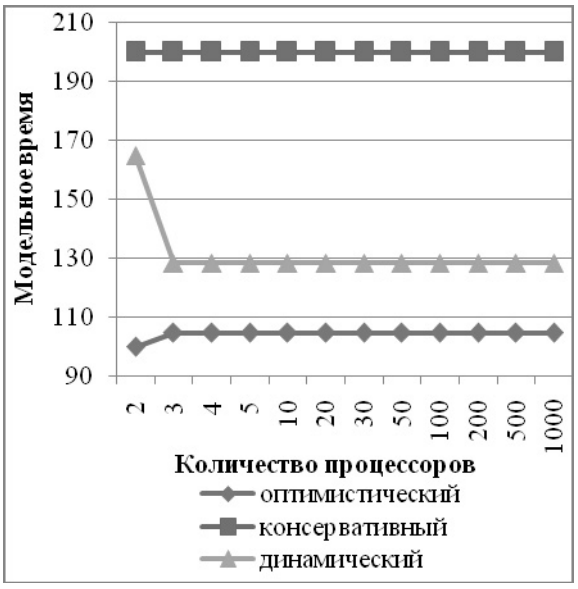

Рисунок 9 – Зависимость модельного времени алгоритмов продвижения времени от количества процессов

для n=6..1000 значения ξ^n повторяются с периодом 5.

затраты на сохранение состояний определяются вектором t^n =(0.5; 0.5; 1; 1; 1), n=1..5;

для n=6..1000 значения t^n повторяются с периодом 5.

вероятности передачи сообщения на i-м шаге

$$\alpha = \begin{vmatrix} 0 & 0.1 & 0.1 \\ 0.1 & 0 & 0.1 \\ 0.1 & 0.1 & 0 \end{vmatrix}$$

Для консервативного алгоритма продвижения времени были заданы величины $\beta_i = 1$ и $\gamma_i = 0.$ Для оптимистического $\beta_i = 0$, $\gamma_i = 1.$ Для динамического алгоритма были выбраны следующие значения: β_i =0.65; γ_i =0,35.

При сравнении результатов экспериментов 5 и 8 видно, что при малом разбросе приращений локальных часов ($\Delta\xi^n$ =1) динамический алгоритм менее эффективен, чем консервативный. Преимущество консервативного алгоритма над динамическим составляет 55.6%.

Заключение

Предложена математическая модель динамического алгоритма продвижения времени для распределенного логического моделирования. Выполнено экспериментальное исследование математической модели. Получены сравнительные характеристики модели динамического алгоритма с классическими консервативным и оптимистическим алгоритмами. Результаты исследования математических моделей показывают, что использование динамического алгоритма эффективно при:

1) низкой интенсивности передачи сообщений в системе моделирования;
2) большом разбросе приращений локальных часов;
3) равновероятной вероятности переключения алгоритмов синхронизации;
4) использование в системе моделирования трех и более процессов;

В дальнейшем планируется соотнести результаты исследования математической модели динамического алгоритма с экспериментальными результатами в программной системе распределенного логического моделирования.

Литература

1. Ladyzhensky Y.V., Popoff Y.V. Software system for event-driven logic simulation // IEEE EWDWT, Odessa, September 15-19, 2005, p.119-122

2. C.J.R. Shi, D.Lungeanu. Distributed simulation of VLSI circuits via lookahead-free self-adaptive optimistic and conservative synchronization. In Proc. ICAAD, pages 500-504, Nov 1999.

3. Вознесенская Т.В. Исследование эффективности методов синхронизации времени для распределенного имитационного моделирования // Труды Всероссийской научной конференции "Высокопроизводительные вычисления и их приложения" (30 октября - 2 ноября 2000 г., г. Черноголовка). -М.: Изд-во МГУ, 2000, с. 208-211.

4. Ладыженский Ю.В., Тесленко Г.А. Аппаратный метод повышения эффективности алгоритмов распределенного логического моделирования цифровых систем. // Наукові праці Донецького національного технічного університету. Серія: «Обчислювальна техніка та автоматизація». Випуск 106 – Донецьк: ДонНТУ, 2006. – 220с. – С.77-81.

5. A.Ferscha. Parallel and distributed simulation of discrete event system. Parallel and Distributed Computing Handbook, McGraw-Hill, 1995.