УДК 681.324

М.А. Волк, Р.Н. Гридель, М. Ал Шиблак

Харьковский национальный университет радиоэлектроники, Харьков

АНАЛИЗ РАСПРЕДЕЛЕННЫХ ИМИТАЦИОННЫХ МОДЕЛЕЙ С ОПТИМИСТИЧЕСКИМИ АЛГОРИТМАМИ СИНХРОНИЗАЦИИ

Рассматривается математическая модель функционирования распределенной имитационной модели под управлением оптимистических алгоритмов синхронизации модельного времени. Модель учитывает следующие основные параметры: продвижение модельного времени, объемы памяти, время обслуживания моделей, время простоя ресурсов. Предложенная модель может быть использована при анализе распределенных имитационных моделей, создании эффективных методов распределения ресурсов.

Ключевые слова: *распределенная имитационная модель, оптимистические алгоритмы синхронизации.*

Введение

В настоящее время наблюдается значительный интерес к распределенному имитационному моделированию, позволяющему значительно расширить класс реализуемых имитационных моделей за счет использования большого количества удаленных вычислительных ресурсов, к которым можно отнести локальные и глобальные компьютерные сети, суперкомпьютеры, кластера.

Одной из основных задач при использовании такой дорогостоящей техники является рациональное управление распределением вычислительных ресурсов при проведении масштабных имитационных экспериментов.

Существует ряд работ в области исследования распределенных программ с целью их эффективного выполнения, среди которых можно выделить два основных направления. Первое из них используется в тех случаях, когда в проблемной области, для которой выполняется построение модели, существуют традиционные средства моделирования. Например, в теории массового обслуживания широкое распространение получили программные системы моделирования на базе языка GPSS [1]. Для исследования дискретных динамических систем более широкого класса – аппарат сетей Петри [2]. В этих случаях разработано множество методических рекомендаций, научных аналитических публикаций, которые помогут разработчику выбрать нужные средства моделирования [3]. Второе направление ориентировано на природу имитационной модели как представителя того или иного методологического подхода или использует идею о том, что любая имитационная модель представляет собой программный продукт. К таким работам следует отнести [4 – 8].

Основа подхода, который будет использоваться в настоящей работе заложена в трудах Т.В. Вознесенской [6, 7], которые обобщают модель взаимодействия двух имитационных моделей под управлением оптимистических алгоритмов синхронизации, которые были разработаны D. Mitra и I. Mitrani [5]. Дальнейшее развитие он получил в работах украинских ученых [9 – 11]. В работе [11] приведены результаты исследования существующих методов анализа распределенных имитационных моделей, работающих под управлением различных алгоритмов синхронизации, в глобальных вычислительных системах, объединяющих разнородные ресурсы. Предложена новая математическая модель процесса функционирования распределенных симуляторов, учитывающая временные характеристики исполнения и передачи сообщений в частных моделях и динамическое изменение объемов памяти распределенной модели. В работе [12] приведены модификации модели для случая консервативных алгоритмов синхронизации. В данной статье обобщенная модель, полученная в работе [11], рассматривается для распределенных имитационных моделей, работающих под управлением оптимистических алгоритмов синхронизации.

Модель функционирования распределенной имитационной модели

Процесс функционирования распределенной имитационной модели заключается в исполнении локальных активностей (подпрограмм, процедур, функций) на удаленных процессорах. Активности модели могут вызывать выполнение других активностей, например, по работе с данными (посылка сообщений, сохранение внутреннего состояния), которые, в свою очередь, увеличивают процессорное время по обслуживанию частной модели. При этом, на каждом шаге моделирования может произойти увеличение объема памяти, занимаемого частной моделью. Увеличение памяти обусловлено состояниями внутренних переменных модели, изменениями в объектах операционной системы, обслуживающих программную модель, создаваемыми буферами данных для

обмена сообщениями между моделями. В обобщенную модель включены параметры, учитывающие динамику изменения объемов памяти частных имитационных моделей и реальное время, затрачиваемое на выполнение активностей менеджера памяти по выполнению операций с данными [11].

При продвижении модельного времени t_i^n n-й модели на i-м шаге моделирования происходит сохранение состояния модели, что приводит к росту памяти, занимаемой данными программной частной моделью V_i^n. На операции с памятью затрачивается процессорное время TR_i^n, которое состоит из времени сохранения состояния модели t_i^s, времени передачи сообщения от частной модели n частной модели k (t_i^m). Время t_i^m в общем случае является функцией, зависящей от поведения активности передачи данных δ_i^{nk} и размера передаваемых данных ($buff_i^{nk}$). В свою очередь время, затраченное на выполнение всей имитационной модели на i-м шаге моделирования ΔTR_i определяется как максимум среди суммы времен исполнения поведенческой активности φ_i^n и времени выполнения операций с данными TR_i^n. В результате, получаем следующую математическую модель, описывающую развитие распределенной имитационной модели (1), где TR_i – реальное время, затраченное на выполнение i шагов процесса моделирования всей имитационной модели; TR_i^n – реальное время, затраченное n-й частной моделью на i-м шаге моделирования на работу с данными; t_i^s – время сохранения состояния модели на i-м шаге моделирования; t_i^m – время передачи данных; $time(\varphi_i^n)$ – время исполнения поведенческой активности φ_i^n; a^{nk} – вероятность посылки сообщения от частной модели n к частной модели k; N – количество частных моделей; TR_i^M – время работы служб системы моделирования на i-м шаге моделирования; t_i^n – локальное модельное время n-й частной модели на i-м шаге моделирования; T_i – глобальное модельное время на i-м шаге моделирования; ξ_i^n – значение времени, характеризующее внутреннюю работу процессов между двумя соседними шагами моделирования; V_i^n – объем памяти, занимаемый данными n-й частной моделью на i-м шаге моделирования; V_i^{ns} – объем дампа памяти для n-й частной модели на i-м шаге моделирования; ΔV_i^n – приращение объема памяти n-й частной модели на на i-м шаге моделирования; V_i – объем памяти, занимаемой всей распределенной имитационной моделью на i-м шаге моделирования; TR_0, TR_0^n, t_0^n, V_0^n, V_0 – начальные значения соответствующих параметров.

$$
\begin{cases}
TR_0 = 0;\ TR_0^n = 0; \\
\Delta TR_i^n = \sum\limits_{k=1}^{N} a^{nk} \cdot t_i^m(\delta_i^{nk}, sizeof(buff_i^{nk})) + \\
\qquad + \begin{cases} t_i^s, \text{если } t_i^n = T_i; \\ 0, \text{если } t_i^n \neq T_i; \end{cases} \\
TR_{i+1}^n = TR_i^n + \Delta TR_i^n; \\
TR_{i+1} = TR_i + \max\limits_{n=\overline{1,N}}(time(\varphi_i^n) + \Delta TR_i^n) + TR_i^M; \\
t_0^n = 0;\ t_{i+1}^n = \begin{cases} t_i^n + \xi_i^n, \text{если } t_i^n = T_i \\ t_i^n, \text{если } t_i^n \neq T_i \end{cases}; \\
V_0^n = sizeof(\Lambda^n \cup \Delta^n);\ V_0 = \sum\limits_{n=1}^{N} V_0^n; \\
\Delta V_i^n = \begin{cases} V_i^{ns}, \text{если } t_i^n = T_i; \\ 0, \text{если } t_i^n \neq T_i; \end{cases} \quad V_{i+1}^n = V_i^n + \Delta V_i^n; \\
V_{i+1} = V_i + \sum\limits_{n=1}^{N} \Delta V_i^n; \quad T_0 = 0; \\
T_{i+1} = \min\limits_{n=\overline{1,N}}(t_{i+1}^n);\ n = \overline{1, N}.
\end{cases} \qquad (1)
$$

Параметры T_i, V_i, TR_i вычисляются независимо друг от друга отдельными процедурами. Следовательно, возможна реализация параллельного вычисления этих параметров как на системах с общей (SMP) так и раздельной памятью (MMP).

Кроме того, отсутствие количественных данных о каком-либо члене выражения (1) (например, времени, которое характеризует внутреннюю работу процессов между двумя соседними шагами моделирования), повлечет за собой невозможность вычисления только одного из них.

Модификация модели для оптимистических алгоритмов синхронизации

В общем случае, для анализа оптимистических алгоритмов можно использовать систему выражений (1). Однако, ряд модификаций оптимистических алгоритмов приводят к ее изменению.

Особенностью оптимистических алгоритмов является возможность откатов моделей в прошлое, для организации которого необходимо сохранение состояний модели (выполнение дампов памяти). Это приводит к постоянному росту памяти в моменты активизации частной модели.

Выделим из выражения (1) часть, отвечающую за рост объема памяти:

$$\begin{cases} \Delta V_i^n = \begin{cases} V_i^{ns}, \text{если } t_i^n = T_i \\ 0, \text{если } t_i^n \neq T_i \end{cases}; \\ V_{i+1}^n = V_i^n + \Delta V_i^n; \quad V_{i+1} = V_i + \sum_{n=1}^{N} \Delta V_i^n. \end{cases} \tag{2}$$

При большом количестве шагов моделирования увеличение объема памяти приводит к переполнению физической или виртуальной памяти вычислительного ресурса. Так, при использовании наиболее известного оптимистического алгоритма, разработанного Джефферсоном (Time Warp), в момент получения частной моделью события, имеющего временную отметку меньшую, чем уже обработанные события, частная модель выполняет откат и обрабатывает эти события повторно в хронологическом порядке. Откатываясь назад, частная модель восстанавливает состояние, которое было до обработки событий (используются контрольные дампы памяти) и отказывается от сообщений, отправленных "откаченными" событиями. Следовательно, для частной модели наблюдается постоянный рост памяти за счет выполнения дампа памяти и формирования новых сообщений. Во время отката необходимо не только вернуть состояние частной модели, что соответствует загрузке одного из контрольных дампов памяти, но и очистить память, занимаемую дампами памяти, созданными после контрольного.

Механизм антисообщений в алгоритме Time Warp позволяет уничтожать в памяти необработанные сообщения, созданные после контрольного дампа памяти, либо, если частная модель-получатель уже обработала его, активирует процесс отката для частной модели-получателя.

Результатом реализации этого алгоритма является постоянное изменение объемов памяти частной модели. Для отражения этих процессов предлагается модифицировать выражение (2) следующим образом:

$$\begin{cases} V_0^n = \text{sizeof}(\Lambda^n \cup \Delta^n); \quad V_0 = \sum_{n=1}^{N} V_0^n; \\ \Delta V_i^n = \begin{cases} V_i^{ns} + V_i^{nm} - \sum_{j=1}^{M^n} \tau_j V_{ij}^{nm}, \text{если } t_i^n = T_i; \\ 0, \text{если } t_i^n \neq T_i; \end{cases} \\ V_{i+1}^n = V_i^n + \Delta V_i^n - \sum_k \Delta V_k^n, \ \forall k \in [q,i]; \\ V_{i+1} = V_i + \sum_{n=1}^{N} \Delta V_i^n, \end{cases} \tag{3}$$

где V_i^{nm} – объем сообщений (или антисообщений), возникающих в результате выполнения активностей частной модели на i-м шаге моделирования и помещенных в очередь сообщений; M^n – количество сообщений в локальной очереди сообщений n-й частной модели; V_{ij}^{nm} – объем памяти, занимаемый j-м сообщением в этой очереди; $\tau_j \in [0,1]$ – бинарный признак удаления сообщения из очереди; q – шаг моделирования, до состояния которого частная модель осуществляет откат ($q < i$); индекс k перечисляет все удаляемые дампы памяти, совершенные для данной частной модели между шагами моделирования i и q.

Признак удаления сообщения из очереди τ_j будет равен 1 в следующих случаях:

1) при удалении обработанного сообщения из очереди, то есть если временная метка сообщения равна T_i и сообщение обслужено частной моделью, для которого это сообщение предназначалось;

2) при удалении сообщений во время отката частной модели для всех сообщений, чья временная метка больше временной метки, соответствующей откату.

Несмотря на то, что в выражении (2) параметр V_{i+1}^n уменьшается в случае отката модельного времени, в динамике, при увеличении GVT этот параметр растет в силу того, что в памяти сохраняются все дампы памяти, выполненные до этого момента времени (GVT).

В связи с этим, предложен ряд модификаций оптимистических алгоритмов синхронизации, ограничивающих рост памяти.

Для Time Warp систем, для которых отмечается чрезмерное "забегание" вперёд частных моделей, что ведет к серьёзным затратам памяти и длинным откатам, применяют ограничения этих "забеганий". Вводятся "перемещаемые" в модельном времени окна, определяемые как временной интервал [GVT, GVT+T_w], где T_w – это параметр, задаваемый пользователем либо вычисляемый в процессе имитации. Частные модели обрабатывают только те события, временные метки которых находятся в данном временном интервале. Ограничение на продвижение модельного времени частной имитационной модели приводит к простою вычислительного ресурса, что обсуждалось при расмотрении консервативных алгоритмов синхронизации распределенных имитационных моделей. В этом случае становиться возможным применения выражения (3) для оценки простоя вычислительного ресурса. В этом случае можно использовать выражение (3) с модифицированным условием, учитывающим существование временных окон:

$$\begin{cases} TP_0^n = 0; \\ TP_{i+1}^n = \begin{cases} TP_i^n, \text{если } (t_i^n = T_i \\ \qquad \text{или } t_i^n > GVT + Tw); \\ TP_i^n + \max_{n'}(\text{time}(\varphi_i^{n'}) + \Delta TR_i^{n'}), \\ n' = \overline{1,N} / n' = n, \text{если } t_i^n \neq T_i. \end{cases} \end{cases} \tag{4}$$

Еще одна модификация метода заключается в задержке отправки сообщения до того момента, когда появляется гарантия, что она не состоится позже отката (GVT продвигается до модельного времени, на которое было запланировано событие, что исключает необходимость в антисообщениях и исключаются каскадированные откаты). Известны подходы, использующие "просмотр назад" (lookback), технику прямой отмены, которая иногда используется для срочной отмены некорректных сообщений.

Существуют ряд путей решения проблемы больших затрат памяти для хранения дампов памяти. Наиболее известные из них: выполнение отката с той целью, чтобы освободить память; сохранение текущего состояния реже, чем сохранение после каждого события; задание периодов сохранения в начале моделирования, освобождение памяти, занятой векторами состояний, даже в том случае, если их временная метка больше, чем GVT, и т.д.

Учитывая возможность реализации вышеперечисленных способов управления памятью, предлагается следующая обобщенная математическая модель, отражающая динамику изменения объемов памяти при реализации распределенных имитационных моделей, использующих оптимистические алгоритмы синхронизации:

$$\begin{cases} V_0^n = \text{sizeof}(\Lambda^n \cup \Delta^n); \\ V_0 = \sum\limits_{n=1}^{N} V_0^n; \\ \Delta V_i^n = \begin{cases} V_i^{ns} + V_i^{nm} - \sum\limits_{j=1}^{M^n} \tau_j V_{ij}^{nm}, \text{если } t_i^n = T_i; \\ 0, \text{если } t_i^n \neq T_i; \end{cases} \\ \Delta V_i'^n = \sum\limits_{j=1}^{i} V_j^n, (\forall j, j < i_{GVT} \text{ или } j > i_n'); \\ V_{i+1}^n = V_i^n + \Delta V_i^n - \sum\limits_k \Delta V_k^n - \Delta V_i'^n, \forall k \in [q, i]; \\ V_{i+1} = V_i + \sum\limits_{n=1}^{N} \Delta V_i^n - \Delta V_i'^n, \end{cases} \quad (5)$$

где $\Delta V_i'^n$ – объемы памяти, освобождаемые n-й частной моделью при передвижении нижней границы GVT или откатах памяти, i_{GVT} – шаг моделирования, на котором была достигнута нижняя граница GVT, i_n' – шаг моделирования, до временной метки которого осуществляется откат частной модели n.

В случае реализации отката удаляются все дампы памяти, выполненные частной моделью между шагами моделирования, временные метки которых больше значения модельного времени, до которого осуществляется откат. В этом случае, вычисление значения освобождаемой памяти n-й частной модели $\Delta V_i'^n$ предлагается производить согласно следующему выражению:

$$\Delta V_i'^n = \sum_{j=1}^{i} V_j^n, \forall j, \tau_j > \tau_{i'}, \quad (6)$$

где τ – временная метка дампа памяти, то есть уменьшение объема памяти для n-й частной модели при реализации отката модельного времени определяется как сумма объемов дампов памяти частной модели, временная метка которых τ_j больше временной метки $\tau_{i'}$ дампа памяти частной модели, выполненного на шаге моделирования i', до которого осуществляется откат.

При продвижении нижней границы модельного времени GVT объем освобождаемой памяти определяется как сумма объемов дампов памяти, временная метка которых меньше значения GVT, и равен

$$\Delta V_i'^n = \sum_{j=1}^{i} V_j^n, \forall j, \tau_j < \tau_{GVT}, \quad (7)$$

то есть из совокупности дампов памяти частной модели будут удалены все выполненные дампы памяти n-й частной модели, временная метка τ_j которых меньше временной метки нижней границы глобального модельного времени τ_{GVT}.

Обобщенная математическая модель функционирования распределенной имитационной модели под управлением оптимистических алгоритмов синхронизации должна учитывать описанные наиболее известные алгоритмы, а также дополнительные временные параметры и параметры объемов памяти, появление которых обуславливается применением этих алгоритмов. Отметим, что эти параметры должны учитываться моделью, но при их отсутствии (невозможности их получения), не должны влиять на другие параметры модели.

С учетом вышеизложенного, предлагается обобщенная математическая модель функционирования распределенной имитационной модели под управлением оптимистических алгоритмов синхронизации – следующее выражение (8):

$$TR_0 = 0; TR_0^n = 0; TP_0^n = 0;$$

$$\Delta TR_i^n = \sum_{k=1}^{N} a^{nk} \cdot t_i^m(\delta_i^{nk}, \text{sizeof}(buff_i^{nk})) + \begin{cases} t_i^s, \text{если } t_i^n = T_i; \\ 0, \text{если } t_i^n \neq T_i; \end{cases}$$

$$TR_{i+1}^n = TR_i^n + \Delta TR_i^n; \quad TR_{i+1} = TR_i + TR_i^M + \\ + \max_{n=\overline{1,N}}(\text{time}(\phi_i^n) + \Delta TR_i^n) + \max(\underline{\text{time}}_{j=\overline{1,N}}(\Delta V_i'^j));$$

$$TP_{i+1}^n = \begin{cases} TP_i^n, \text{если}(t_i^n = T_i \text{ или } t_i^n > GVT + Tw); \\ TP_i^n + \max\limits_{n'}(\text{time}(\phi_i^{n'}) + \Delta TR_i^{n'}), \\ \quad n' = \overline{1,N} / n' = n, \text{если } t_i^n \neq T_i; \end{cases}$$

$$t_0^n = 0; \quad t_{i+1}^n = \begin{cases} t_i^n + \xi_i^n, \text{если } t_i^n = T_i; \\ t_i^n, \text{если } t_i^n \neq T_i; \\ t_{i'}^n, \text{если } \exists i'_n; \end{cases} \qquad (8)$$

$$V_0^n = \text{sizeof}(\Lambda^n \cup \Delta^n); \; V_0 = \sum_{n=1}^{N} V_0^n;$$

$$\Delta V_i^n = \begin{cases} V_i^{ns} + V_i^{nm} - \sum_{j=1}^{M^n} \tau_j V_{ij}^{nm}, \text{если } t_i^n = T_i; \\ 0, \text{если } t_i^n \neq T_i; \end{cases}$$

$$\Delta V_i'^n = \sum_{j=1}^{i} V_j^n, (\forall j, j < i_{GVT} \vee j > i'_n);$$

$$V_{i+1}^n = V_i^n + \Delta V_i^n - \sum_k \Delta V_k^n - \Delta V_i'^n, \; \forall k \in [q, i]; \; T_0 = 0;$$

$$V_{i+1} = V_i + \sum_{n=1}^{N} \Delta V_i^n - \Delta V_i'^n; T_{i+1} = \min_{n=\overline{1,N}} (t_{i+1}^n); \; n = \overline{1, N},$$

где $\max(\underset{j=\overline{1,N}}{time}(\Delta V_i'^j))$ – время, затрачиваемое системой моделирования на процедуру отката или удаления дампов памяти, выполненных ранее временной метки GVT: определяется как максимальное время удаления дампов памяти среди частных локальных моделей.

Выводы

Получена математическая модель функционирования распределенной имитационной модели с оптимистическим алгоритмом синхронизации частных моделей (выражение (8)), которая учитывает временные затраты и затраты памяти на выполнение сохранений состояний частных моделей (дампы памяти), реализацию откатов, обслуживание потоков сообщений и антисообщений.

Данная модель может быть использована в методах и средствах анализа распределенных имитационных моделей, при построении автоматизированных систем анализа распределенных имитационных моделей, планировщиков распределенных вычислительных систем, при создании и исполнении распределенных имитационных моделей, параллельном и распределенном программировании.

Список литературы

1. Томашевский В. Имитационное моделирование в среде GPSS / В. Томашевский, Е. Жданова. – М.: Бестселлер, 2003. – 416 с.

2. Юдицкий С.А. Метод анализа конфигураций организационных систем на сетях Петри / С.А. Юдицкий, И.А. Мурадян // УБС. – 2007. – №16. – С. 163-170.

3. Крэйн М. Введение в регенеративный метод анализа моделей / М. Крэйн, О. Лемуан. – М.: Наука, 1982.– 104с.

4. Окольнишников В.В. Разработка средств распределенного имитационного моделирования для многопроцессорных вычислительных систем: диссертация д-ра техн. наук: 05.13.18 / Окольнишников В.В. – Новосибирск, 2006. – 227 с. РГБ ОД71:07-5/433.

5. Mitra D. Analysis and optimum performance of two message-passing parallel processors synchronized by rollback / D. Mitra, I. Mitrani // Performance'84. – 1984. – P. 35-50.

6. Вознесенская Т.В. Математическая модель алгоритмов синхронизации времени для распределённого имитационного моделирования / Т.В. Вознесенская // Программные системы и инструменты: Тематический сборник факультета ВМиК МГУ им. Ломоносова. – №1. – С. 56-66.

7. Вознесенская Т.В. Математическая модель для анализа производительности распределенных систем имитационного моделирования / Т.В. Вознесенская // Искуственный интеллект (Донецк). – 2002. – №2. – С. 74-78.

8. Миков А.И. Программные средства оптимизации распределенного имитационного эксперимента / А.И. Миков, Е.Б. Замятина, А.А. Козлов // Научный сервис в сети Интернет: масштабируемость, параллельность, эффективность: Труды Всероссийской суперкомпьютерной конференции (21-26 сентября 2009 г., г. Новороссийск). – М.: Изд-во МГУ, 2009. – 524 с.

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

10. Ладыженский Ю.В. Математическая модель динамического алгоритма продвижения времени для распределенного логического моделирования цифровых систем / Ю.В. Ладыженский, Г.А. Тесленко // Наукові праці Донецького національного технічного університету. Серія: Інформатика, кібернетика та обчислювальна техніка. – Донецк, 2008. – №9. – С. 55-62.

11. Волк М.А. Анализ распределенных имитационных моделей в гетерогенных вычислительных системах / М.А. Волк // Науковий вісник Чернівецького національного університету імені Юрія Федьковича. Серія: Комп'ютерні системи та компоненти. – Том 1, випуск 2. – Чернівці: ЧНУ, 2010. – С. 35-39.

12. Волк М.А. Анализ распределенных имитационных моделей с консервативными алгоритмами синхронизации / М.А. Волк, М.А.Филимончук, Муаз Ал Шиблак, Р.Н. Гридель // Збірник наукових праць Харківського університету Повітряних Сил. – Х.: ХУ ПС, 2012. – Вип. 1(30). – С. 95-98.

Надійшла до редколегії 21.11.2012

Рецензент: д-р техн. наук, проф. О.Ф. Михаль, Харьковский национальный университет радиоэлектроники, Харьков.

АНАЛІЗ РОЗПОДІЛЕНИХ ІМІТАЦІЙНИХ МОДЕЛЕЙ С ОПТИМИСТИЧЕСКИМИ АЛГОРИТМАМИ СИНХРОНІЗАЦІЇ

М.О. Волк, М. Аль Шиблак, Р.М. Гридель

Розглядається математична модель функціонування розподіленої імітаційної моделі під управлінням оптимистических алгоритмів синхронізації модельного часу. Модель враховує наступні основні параметри: просування модельного часу, обсяги пам'яті, час обслуговування моделей, час простою ресурсів. Запропонована модель може бути використана при аналізі розподілених імітаційних моделей, створення ефективних методів розподілу ресурсів.

Ключові слова: *розподілена імітаційна модель, оптимістичні алгоритми синхронізації.*

ANALYSIS OF SIMULATION MODEL WITH OPTIMISTIC SYNCHRONIZATION ALGORITHMS

M.O. Volk, M. Al Shiblak, R.M. Gridel

A mathematical model of the distributed simulation model for the running of optimistic synchronization algorithms of model time is considered. The model takes the following key parameters: the promotion of model time, memory, service time models, idle resources. The proposed model can be used in the analysis of distributed simulation models, developing effective methods of allocating resources.

***Keywords**: distributed simulation model, optimistic synchronization algorithms.*