

УДК 004.75

ГЛОБАЛЬНОЕ ВИРТУАЛЬНОЕ ВРЕМЯ В СИСТЕМЕ РАСПРЕДЕЛЁННОГО ЛОГИЧЕСКОГО МОДЕЛИРОВАНИЯ ЦИФРОВЫХ СИСТЕМ С ДИНАМИЧЕСКОЙ ТОПОЛОГИЕЙ СВЯЗЕЙ ПРОЦЕССОРОВ

Балута А.В., Попов Ю.В.

*Донецкий национальный технический университет
кафедра прикладной математики и информатики
a.mag@front.ru*

Рассмотрены особенности иерархической архитектуры вычислительной сети для распределённого логического моделирования цифровых систем, допускающей использование ненадежных процессоров. Приведен алгоритм вычисления глобального виртуального времени как одного из ключевых параметров при моделировании с учетом архитектуры вычислительной сети.

Введение

При моделировании цифровых логических систем большой размерности возникает необходимость использования распределённых вычислений. При этом необходимо учитывать возможность возникновения неравномерной нагрузки на процессоры при моделировании [1, 5]. Также в условиях использования нескольких моделирующих процессоров в сети нужно учитывать внезапные отключения компьютеров в сети либо потерю связи между некоторыми из них.

Различные варианты решений задач, связанных с использованием многоуровневой архитектуры моделирующих систем и более эффективного распараллеливания вычислений при распределённом моделировании рассматриваются в работах Theodoropoulos и Logan [2], Ferscha и Tripathi [3].

Объект исследований данной статьи – организация вычислительной сети для распределенного логического моделирования цифровых систем с динамической топологией связей процессоров. Предмет исследования – организация алгоритма вычисления глобального виртуального времени при оптимистическом алгоритме синхронизации моделирующих процессоров. Цели и задачи исследований, рассмотренных в статье, включают получение алгоритма вычисления глобального виртуального времени в системе распределенного логического моделирования цифровых систем и определение особенностей его работы по сравнению с традиционной архитектурой вычислительной сети.

Архитектура вычислительной сети

В разработанной системе используется модификация предложенного в [2] подхода к организации моделирующей системы на основе управления сферами влияния. Система моделирования состоит из нескольких моделирующих процессоров, соединенных в сеть. Логические процессоры (ЛП) выполняют моделирование части схемы, входящей в их сферу влияния. Коммуникационные логические процессоры (КЛП) используются для обеспечения связи между ЛП (см. рис. 1б). В вычислительной сети имеется администратор процесса моделирования (АПМ), содержащий базу данных всех процессоров, доступных в сети. На основании параметров, вычисляемых при моделировании для подчиненных процессоров, каждый КЛП может выполнять перераспределение сфер влияния на подчиненных процессорах или запрашивать у АПМ свободный КЛП для передачи ему группы подчиненных процессоров, что приводит к генерации дерева КЛП (рис. 1).

Функции КЛП: определение процессора, которому должно быть передано событие в системе при моделировании в соответствии с назначенными сферами влияния; сбор статистики при моделировании; обеспечение возможности возврата системы к предыдущим состояниям в случае отключения подчиненного процессора.

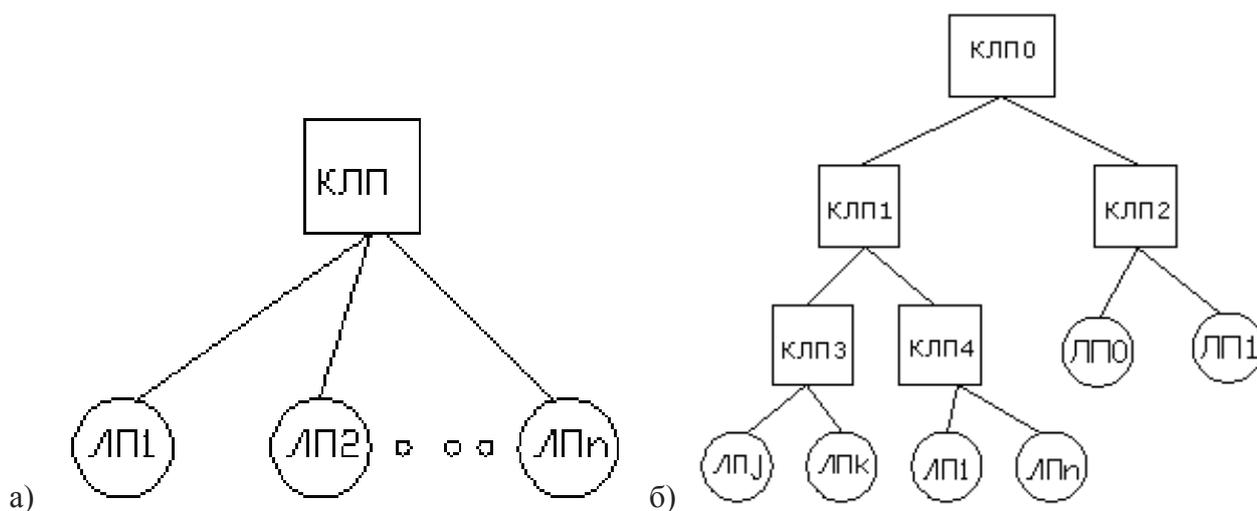


Рисунок 1. а) начальное распределение; б) генерация дерева КЛП

Поскольку разработанная система ориентирована на использование ненадежных процессоров, которые могут отключаться при моделировании, то в системе должна быть реализована возможность выполнять откаты к одному из состояний, предшествующих моменту отключения процессора (рис. 2). Поэтому моделирование на ЛП является оптимистическим и допускает возникновение откатов [1]. В КЛП используется данная особенность оптимистического протокола синхронизации для выполнения отката в группе подчиненных процессоров в связи с перераспределением сфер влияния.



Рисунок 2. Перераспределение сферы влияния при отключении процессора

Вычисление глобального виртуального времени

Глобальное виртуальное время (Global Virtual Time – GVT) – это параметр, вычисляемый для момента модельного времени T . GVT – это минимальное значение LVT на всех процессорах и минимум среди временных меток всех сообщений, которые в момент T находятся в сети [3, 4]. Никакое событие не может уменьшить LVT на любом из ЛП на значение, меньше чем GVT.

GVT используется для освобождения памяти от хранения устаревших состояний системы. Любое обработанное событие e , временная метка которого $t(e) < GVT(T)$ является примененным безвозвратно [3] и, следовательно, для события e уже не нужно хранить историю изменения состояния моделируемой системы.

При использовании традиционной топологии связи ЛП определение глобального состояния системы производится благодаря получению распределенного снимка состояния [4]. Получение снимка состояния системы производится с помощью маркера, рассылаемого по сети, благодаря чему организуется процесс вычисления локального состояния на всех процессорах в сети [4]. Вычисление глобального состояния системы выполняется на ЛП и занимает дополнительную память и процессорное время.

При иерархической организации вычислительной сети требуется алгоритм вычисления GVT, учитывающий особенности такой архитектуры. В процедуре вычисления GVT в системе участвуют

```

(1)  если (запрос_на_вычисление_LGVT) то
(2)    CVT:=∞;
(3)    для всех процессоров ∈ БД
        получен_ответ:=false;
(4)    для всех процессоров ∈ БД
        отправить_запрос_на_вычисление_LGVT;
(5)    если возникло_внешнее_событие (e@t) то
(6)      если t<CVT то CVT:=t;
(7)    если для всех процессоров ∈ БД получен_ответ=true то
        LGVT:=(min(∇ ответов_от_процессоров), CVT);
(8)    если ∃родительский_КЛП то
        отправить_LGVT_на_родительский_КЛП
(9)    иначе
(10)     если LGVT=∞ то
            завершить_моделирование
(11)     иначе
            GVT:=LGVT;
            перейти_на_шаг (2);
        end;
(12)  end;
(13)  end;
    
```

Рисунок 3. Алгоритм вычисления GVT на КЛП

КЛП и ЛП. Алгоритм вычисления GVT на КЛП приведен на рис. 3.

Исходя из структуры системы (см. рис. 1, б) каждый КЛП может вычислить местное GVT (Local GVT - LGVT) в пределах своего поддерева. Тогда GVT во всей сети – это местное GVT, вычисленное на корневом КЛП (КЛП0 на рис. 1, б).

В начале моделирования корневой КЛП отправляет всем своим подчиненным процессорам (КЛП и ЛП) запрос на вычисление LGVT. Этот запрос также содержит текущее значение GVT, которое в начале моделирования равно 0 (рис. 4). Каждый КЛП, получив такой запрос, также рассылает его своим подчиненным ЛП/КЛП.

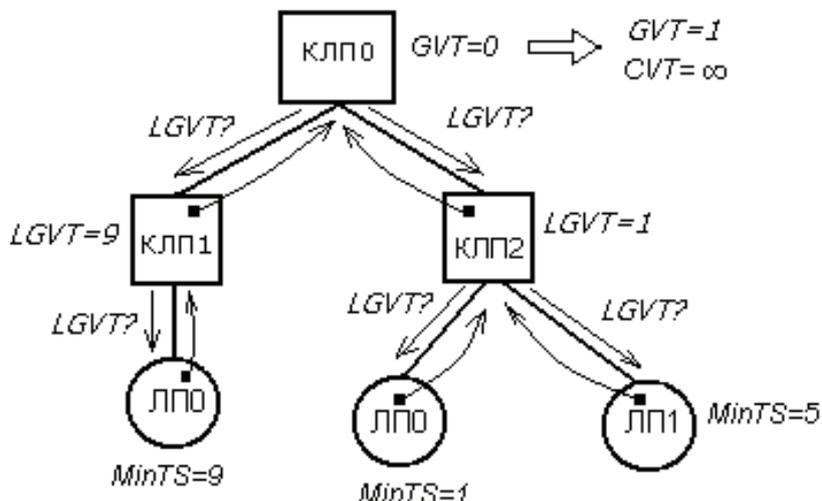


Рисунок 4. Начальный этап вычисления GVT

Когда на ЛП приходит такой запрос, ЛП вычисляет своё местное LGVT как минимум среди значения времени следующего события в списке событий и местного виртуального времени (LVT) (1):

$$LGVT(ЛП) = \min(LEVL.время_следующего_события, LVT). \tag{1}$$

LVT на ЛП - это минимум среди значений виртуального времени VT и времени наступления следующего события в списке событий. ЛП отправляет посчитанное значение LGVT запросившему родительскому КЛП, при этом используя присланное с запросом значение GVT для очистки истории состояний системы раньше GVT. Также GVT используется на ЛП для вычисления LVTH (2):

$$LVTH (ЛП_i) = GVT + \text{Const.} \quad (2)$$

КЛП, выполнив рассылку запросов на вычисление LGVT, ожидает, когда придут ответы от всех подчиненных процессоров. При этом продолжается выполнение моделирования в системе и через КЛП передаются сообщения о событиях в узлах схемы (рис. 5). Во время ожидания прихода всех ответов КЛП фиксирует временные метки всех этих событий на канальных часах CVT и хранит значение минимальной из них (3):

$$CVT(КЛП_i) = \min_k(ts(e_k)), e_k \in [t1; t2] \quad (3)$$

где промежуток времени [t1; t2] означает время, прошедшее от отправки запроса на вычисление LGVT до прихода ответов от всех процессоров по этому запросу.

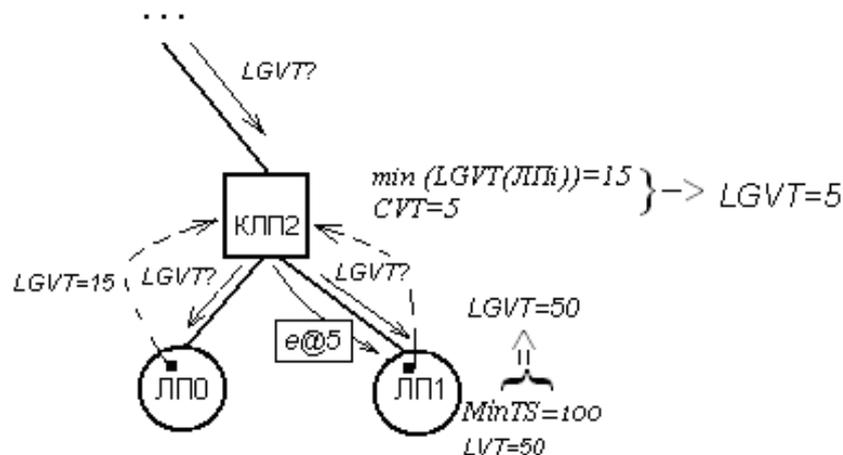


Рисунок 5. Изменение времени на канальных часах и пересчет LGVT

Когда на КЛП_i пришли все ответы на запросы вычисления LGVT от подчиненных процессоров, КЛП_i вычисляет значение своего местного LGVT.

После вычисления LGVT КЛП_i отправляет его родительскому КЛП в ответ на полученный ранее запрос. Когда корневой КЛП получает все ответы от своих подчиненных процессоров, то посчитанное им LGVT является глобальным виртуальным временем во всей системе GVT. Когда корневой КЛП вычислил GVT, он снова рассылает запрос на вычисление LGVT подчиненным процессорам и таким образом инициируется следующий этап пересчета LGVT на всех процессорах в системе и вычисление GVT выполняется заново.

Когда все события на ЛП обработаны, значение LGVT на ЛП становится равным бесконечности. Если после пересчета LGVT на КЛП_i становится равным бесконечности, то это означает, что в пределах поддерева КЛП_i уже нет событий для обработки. Когда GVT становится равным бесконечности, то это означает, что моделирование завершено. Корневой КЛП в этом случае выполняет рассылку своим подчиненным процессорам команды завершения моделирования, которая далее рассылается всем КЛП и ЛП в системе вниз по дереву процессоров.

Выводы

Предложен новый алгоритм вычисления GVT, учитывающий иерархическую организацию связей между процессорами в вычислительной сети. Вычисление глобального виртуального времени в системе используется для экономии памяти при хранении истории предыдущих состояний системы и для определения завершения выполнения задания моделирования.

Разработанный алгоритм вычисления GVT имеет ряд преимуществ по сравнению с вычислением глобального состояния в распределённой системе с традиционной архитектурой связей процессоров:

- не требуется пересылка локальных состояний ЛП по сети и использование памяти для

- хранения этих состояний, за счет этого процедура вычисления GVT в системе использует меньше оперативной памяти;
- не требуется подтверждений при передаче каждого события по сети, что приводит к уменьшению трафика; подтверждения при передаче событий в традиционной архитектуре использовались для получения непротиворечивых срезов состояний распределённой системы [4];
 - уменьшается нагрузка на ЛП, поскольку вычислением GVT в большей степени заняты КЛП.

Литература

- [1] Ладыженский Ю.В., Попов Ю.В. Исследование комбинированного протокола синхронизации логических процессоров при распределенном событийном моделировании цифровых устройств // Моделирование и компьютерная графика: Материалы 1-й международной научно-технической конференции, г. Донецк, 04-07 октября 2005. – Донецк, ДонНТУ, 2005. – С. 165-169.
- [2] Theodoropoulos G., Logan B. An approach to interest management and dynamic load balancing in distributed simulation // 2001 European Simulation Interoperability Workshop.– London: University of Westminster, 2001. – p. 566–571.
- [3] Fersha A., Tripathi S.K. Parallel and Distributed Simulation of Discrete Event Systems // Hardbound of Parallel and Distributed Computing.– McGraw-Hill, 1995.
- [4] Mattern F. Efficient Algorithms for Distributed Snapshots and Global Virtual Time Approximation // Journal of Parallel and Distributed Computing Vol. 18, No 4, 1993. – p. 423-434.
- [5] Marurngsith W., Roland N. Specification-based Verification in a Distributed Shared Memory Simulation Model // SIMULATION vol. 86, No 4, 2010. – p. 229-245.