УДК 004.75

УМЕНЬШЕНИЕ ОТКАТОВ ПРИ РАСПРЕДЕЛЕННОМ ОПТИМИСТИЧЕСКОМ МОДЕЛИРОВАНИИ

А.В. Мирецкий, Ю.В. Ладыженский
Донецкий национальный технический университет

Розглянуто два методи зменшення відкотів при розподіленому оптимістичному моделюванні – метод замкненої внутрішньо-компонентної обробки подій, що відстали та метод локальних каскадних відкотів. Показано, що одночасне їх використання дозволяє знизити кількість та глибину відкотів у логічних процесах.

Важным этапом автоматизированного проектирования систем является моделирование. Для сложных объектов чаще всего применяется имитационное моделирование. Для представления проектируемых систем используются дискретные модели, что позволяет применять метод событийного продвижения времени. Известно, что выполнение дискретно-событийного моделирования крупных и сложных систем на одном процессоре обычно требует много времени. Ускорить этот процесс можно, если выполнить декомпозицию модели на логические процессы и распределить эти процессы между процессорами для параллельного запуска, но при этом появляется необходимость в синхронизации моделирующих логических процессов [1]. Существующие алгоритмы оптимистической синхронизации слабо учитывают неоднородности потоков сообщений между компонентами моделируемых систем. Это приводит к выполнению избыточных откатов состояний. Исключение таких откатов позволяет ускорить моделирование. Использование свойств потоков сообщений и поведения компонентов в синхронизации – перспективное направление повышения эффективности распределенного моделирования.

Метод замкнутой внутрикомпонентной обработки отставших событий основан на компонентном представлении моделируемой системы. Объектно-ориентированный анализ сложной системы позволяет представить ее в виде конечного множества компонентов, связанных и взаимодействующих между собой посредством обмена сообщениями (событиями). Модель системы можно представить в виде ориентированного графа $M=(C,L)$, где C – множество компонентов модели M, L –множество связей между компонентами.

Логический процесс LP_n, моделируемый на одном процессоре n, представляется в виде конечного множества компонентов $C_n \subseteq C$.

В [2] введена характеристика lookback (обозначим ее Lb), которая открывает новый источник параллелизма. Основная идея состоит в том, что компоненты могут обрабатывать отставшие события беспорядочно в рамках некоторого интервала модельного времени *[T-Lb, T]* (T – локальное виртуальное время логического процесса), без отката других уже произошедших событий и состояния логического процесса. Для каждого компонента определена lookback-процедура, обрабатывающая отставшие события без отката так, как оно было бы обработано в прошлом, т.е. в момент времени, равный временной отметке отставшего события. Ее реализация зависит от поведения компонента.

Абсолютное значение величины lookback вычисляется каждым компонентом и зависит от функциональности компонента. В [2] показана связь между этой величиной и величиной lookahead, используемой в консервативном моделировании.

В [2] показано, что этот метод позволяет ускорить процесс параллельного моделирования. Прирост производительности в среднем составил 20%.

Метод локальных каскадных откатов

В [2] показано, что есть смысл разработки lookback-процедур только для граничных компонентов, которые имеют связи с внешними логическими процессами, т.к. только они могут получать отставшие сообщения. Таким образом, положительный эффект от lookback-процедур ограничен размером пограничного подмножества компонентов логического процесса.

Разработан метод локальных каскадных откатов (ЛКО) (см. [3]). В нем откаты впервые перенесены с уровня логических процессов на уровень их компонентов. Это позволяет: 1) уменьшить глубину отката в пространстве компонентов, т.к. учитывает обмен сообщениями не только между логическими процессами, но и между их компонентами; 2) отставшие сообщения могут получать не только пограничные компоненты, поэтому область действия lookback-процедур расширяется на все множество компонентов логического процесса. Метод ЛКО дополняет метод замкнутой внутрикомпонентной обработки отставших событий.

В методе ЛКО также используется представление модели в виде графа $G=(C,L)$. Сообщение (событие) есть вектор $e=(y,\phi,c,h)$, где y – момент модельного времени, когда было порождено сообщение; ϕ – момент времени, когда событие должно произойти; c – компонент,

породивший сообщение; h – компонент, которому предназначено сообщение. У каждого компонента $c \in C$ есть свое собственное локальное виртуальное время $LVT^{(c)}$. Локальное виртуальное время логического процесса равно временной метке последнего обработанного события.

Структуры данных, используемые в оптимистическом алгоритме Time Warp [1] распределены между компонентами: SE – общий список событий, предназначенный для синхронизации компонентов внутри логического процесса; $SE^{(c)}$ – список событий компонента c, предназначен для ускорения доступа к событиям, запланированным для выполнения в c; $HE^{(c)}$ – список событий, обработанных компонентом c; $AM(c, c_1)$ – список анти-сообщений для сообщений отправленных из c в смежный с ним c_1.

На рисунке 1 представлено концептуальное описание алгоритма отката компонента c до отметки $ф$.

```
Если ф<LVT(c) Тогда
    1. Откат компонента с до отметки ф;
    LVT(c) := ф;
    2. Для всех смежных с c компонентов с`
        Если с` принадлежит другому логическому процессу Тогда
            Отправить все антисообщения для него по сети;
        Иначе
            Если есть анти-сообщение в AM(c, c`) Тогда
                Выполнить рекурсивный откат с` до минимальной метки анти-сообщения;
                Удалить антисообщения;
            КонецЕсли
        КонецЕсли
    КонецДля
    3. Удалить все события e` из HE(c) для которых y`>LVT(c`);
    Переместить события e` из HE(c) в SE(c) для которых y`<=LVT(c`) и ф`>ф;
    4. Удалить все события e` из SE(c) для которых y`>LVT(c`);
    5. Синхронизировать SE с SE(c);
КонецЕсли
```

Рис. 1. Алгоритм отката в ЛКО

На этапе 1 выполняется откат состояния компонента в его прошлое. На этапе 2 рекурсивно вызывается процедура отката для всех связанных компонентов. На этапах 3 и 4 выполняется отмена обработанных компонентом сообщений. На этапе 5 происходит синхронизация общего списка запланированных событий логического процесса с внутрикомпонентным списком.

Покажем работу алгоритма ЛКО в сравнении с оптимистическим алгоритмом Time Warp. На рисунке 2 представлен пример модели некоторой компьютерной сети, распределенной между 5-ю

процессорами. Процесс LP_3 присылает в процесс LP_1 отставшее событие. В LP_1 начинается откат состояния и отмена событий.

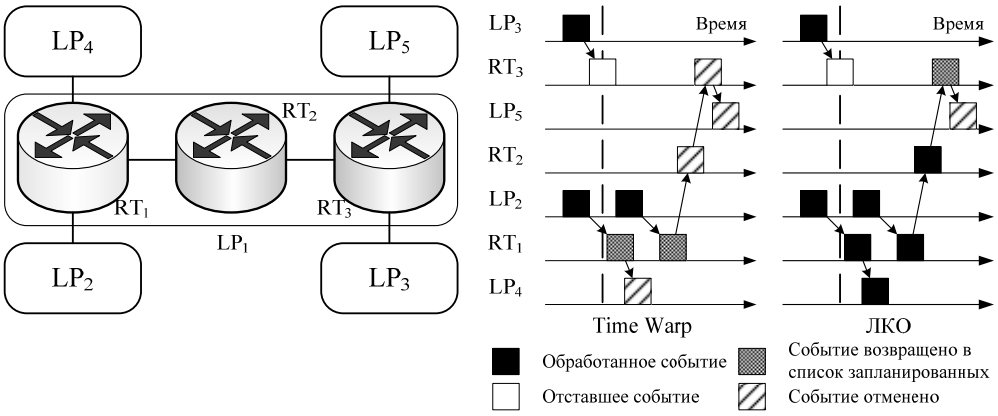

Рис. 2. Сравнение процедуры отката в Time Warp и ЛКО.

Алгоритм Time Warp выполняет откат всего логического процесса, поэтому вызывает отмену событий, отправленных в LP_4 и LP_5. Алгоритм ЛКО выполняет отмену только событий компонента RT_3, что вызывает отмену события, отправленного в LP_5. В рассмотренном примере, все остальные события не отменяются алгоритмом ЛКО. Откат процесса LP_4 не происходит.

Алгоритм ЛКО позволяет уменьшить глубину отката, т.е. уменьшить количество отменяемых событий, количество антисообщений, отправляемых по сети, и количество каскадных откатов удаленных логических процессов.

Экспериментальное исследование ЛКО

Разработана система для моделирования маршрутизации в компьютерных сетях AVMSim, позволяющая использовать любые методы синхронизации логических процессов и исследовать их свойства (см. [4]).

Выполнено исследование поведения алгоритма ЛКО при моделировании замкнутых сетей с очередями разного размера (см. [3]). Для задания размера сети используется обозначение X×Y, где X – количество переключателей, а Y – количество компонентов FCFS. Количество компонентов в модели равно X·(Y+1). Модели разрезались на два логических процесса. Коэффициент уменьшения глубины отката рассчитан по формуле (1) относительно количества компонентов, участвующих в откате в алгоритме Time Warp:

$$K_{кол\text{-}во\ комп.},\% = (1\text{-}2\cdot C_{max}/(X\cdot(Y+1))) \cdot 100, \qquad (1)$$

где C_{max} – максимальное количество компонентов, принявших участие в откате в алгоритме ЛКО. Результаты показаны в таблице 1.

Таблица 1

Результаты экспериментального исследования

Размер сети X×Y	Количество компонентов	Мин. C_{min}	Среднее C_{avg}	Макс. C_{max}	Коэффициент уменьшения, %
12x12	156	1	2,5	21	73,1
24x24	600	1	2,3	34	88,7
36x36	1332	1	2,2	42	93,7

Алгоритм ЛКО позволил уменьшить максимальную глубину отката на ~73-94%, в зависимости от размера сети.

Выводы

Рассмотрен метод замкнутой внутрикомпонентной обработки отставших событий, который позволяет ускорить параллельное/распределенное моделирование в среднем на 20% за счет использования новой характеристики lookback, введенной в [2]. Разработан метод ЛКО, который расширяет возможности метода замкнутой обработки. Совместное использование lookback-процедур и метода ЛКО позволяет уменьшить количество откатов и снизить глубину откатов (количество компонентов, состояние которых возвращается в прошлое). Это приводит к уменьшению ресурсов, задействованных в откате (процессорное время, межпроцессорное взаимодействие).

Библиографический список

1) Fujimoto R.M. Parallel and distributed simulation systems, Wiley Interscience, 2000
2) Chen Gilbert (Gang). New methods for parallel discrete event simulation. // A thesis submitted to the graduate faculty of Rensselaer Polytechnic Institute in partial fulfillment of the requirements for the degree of Doctor of Philosophy Major subject: Computer science. – Troy, New York.: May 2003.
3) Ладыженский Ю.В., Мирецкий А.В. Метод уменьшения глубины каскадных откатов при распределенном оптимистическом моделировании алгоритмов маршрутизации // Научные труды Донецкого национального технического университета. Серия «Информатика, кибернетика и вычислительная техника», 2009. Выпуск 10(153) – Донецк: ДонНТУ. – с. 87-92
4) Ладыженский Ю.В., Мирецкий А.В. Организация системы распределенного дискретно-событийного моделирования с локальными каскадными откатами (avmsim) // Вісник інженерної академії україни №3-4, 2008. – с. 102-109