ПАРАЛЛЕЛЬНЫЕ ВЫЧИСЛЕНИЯ В МНОГОПРОЦЕССОРНЫХ СИСТЕМАХ

Лю Цзяхуэй

ВВЕДЕНИЕ

Последний период развития микропроцессорной техники можно охарактеризовать сильным опережением быстродействия выполнения операций по отношению ко времени подготовки данных. Особенно названная проблема актуальна для суперкомпьютеров. Параллельная система позволяет считывать одновременно много данных. Наиболее просто эта возможность реализуется в суперкомпьютерах с распределенной памятью. Такими являются кластерные системы [1]. Для того чтобы перенести некоторые разработанные программы с последовательных компьютеров на параллельные, анализируются возможности их распараллеливания. Автоматическое распараллеливание последовательных алгоритмов позволит сократить время разработки параллельных программ.

1. АЛГОРИТМ ПРЕОБРАЗОВАНИЯ АРИФМЕТИЧЕСКОГО ВЫРАЖЕНИЯ ДЛЯ ПАРАЛЛЕЛЬНЫХ СИСТЕМ

Пусть арифметическое выражение состоит из арифметических операций, переменных и констант. Время вычислений арифметического выражения зависит от количества необходимых итераций, поэтому операции суммирования и произведения является главной затратой времени. Этот алгоритм преобразовывает арифметическое выражение на параллельные системы для вычисления. Можно ускорить время вычислений. Пусть система включает главный процессор, который выполняет процедуру анализа выражения и рассылки субвыражений, и обрабатывающие процессоры, выполняющие вычисления субвыражений. На рисунке 1 показан алгоритм преобразования последовательного выражения для параллельного вычисления. Алгоритм состоит из следуюших шагов:

Шаг 1. Запись вычисляемого выражения и исходных данных из файла в главный процессор.

Шаг 2. Главный процессор анализирует данные для вычисления выражения, эти данные хранятся в системных буферах данных. Затем главный процессор рассылает данные обрабатывающим процессорам, для вычисления субвыражений.

Шаг 3. Главный процессор анализирует заданное выражение для поиска субвыражений, которые включают операции суммирования и про-

изведения. Затем он записывает субвыражения в двухмерном массиве с типом “char” и их количество (N).

Рис. 1. Схема алгоритма преобразования последовательного арифметического выражения на параллельные системы

Шаг 4. Если счётчик цикла “i” меньше количества субвыражений “N”, главный процессор посылае очередные субвыражения свободному процессору. Процесс заканчивается тогда, когда все субвыражения будут вычислены.

Шаг 5. Вычисленные значения субвыражений посылаются главному процессору. Главный процессор принимает эти результаты и хранит их в одномерном массиве.

Шаг 6. Главный процессор интегрирует результат вычислений.

Шаг 7. Конец.

2. АНАЛИЗ ЭФФЕКТИВНОСТИ ВЫЧИСЛЕНИЙ

Анализ эффективности проведен для арифметических выражений 1 и 2:

$$\sum_{i=1}^{1000000}(a+b+1)+\sum_{i=1}^{1000000}(c+d)+\sum_{i=1}^{1000000}(e+f)+\sum_{i=1}^{1000000}(g*h)+\sum_{i=1}^{1000000}(k+j)-\sum_{i=1}^{1000000}(m-l) \quad (1)$$

$$\sum_{i=1}^{1000000}(a+b+1)+\sum_{i=1}^{100000}(c+d)+\sum_{i=1}^{10000}(g*h)+\sum_{i=1}^{1000}(a+e)+\prod_{i=1}^{100}(b+d)+\prod_{i=1}^{10}(c-b) \quad (2)$$

Время вычисления на последовательной системе имеет следующий вид:

$$T_{computing\text{-}single} = T_{initial} + N_{iterations} \times t_{unit\text{-}computing} \times N_{subequation} \quad (3)$$

$$T_{initial} = T_{initial\text{-}data} + T_{find\text{-}data} \quad (4)$$

где $T_{computing\text{-}single}$ – время вычисления на последовательной системе. $T_{initial}$ – время инициализации вычислений. $N_{subequation}$ – количество субвыражений для вычисления. $N_{iterations}$ – количество итераций операций суммирования и произведения. $t_{unit\text{-}computing}$ – время соответствующей вычислительной операции в системе. $T_{initial\text{-}data}$ – время инициализации массива и записи данных в системном буфере. $T_{find\text{-}data}$ – время поиска данных в системном буфере данных. Пусть $N_{processors}$ – количество процессоров для вычисления на параллельной системе. $T_{computing\text{-}mult}$ – время вычисления на параллельной системе.

Если $N_{subequation} \leq N_{processors}$, время вычисления на параллельной системе приобретает вид:

$$T_{computing\text{-}mult} = T_{initial} + N_{iterations} \times t_{unit\text{-}computing} \times (N_{subequation} / N_{processors}) \quad (5\text{-}1)$$

На паралленьной системе лишний свободный процессор не используется для вычисления субвыражения, поэтому число активных процессоров для вычисления субвыражений равно числу субвыражений. После упрощения 5-1 получается уравнение 5-2 :

$$T_{computing\text{-}mult} = T_{initial} + N_{iterations} \times t_{unit\text{-}computing} \quad (5\text{-}2)$$

Если $N_{subequation} \square N_{processors}$, время вычисления на параллельной системе приобретает вид:

$$T_{computing\text{-}mult} = T_{initial} + N_{iterations} \times t_{unit\text{-}computing} \times (\text{Integer})\ (N_{subequation} / N_{processors}) \quad (6)$$

Ускорение вычисляется по формуле:

$$R = T_{computing\text{-}single} / T_{computing\text{-}mult} \quad (7)$$

На рисунке 2 показана зависимость времени вычисления от количества процессоров. На рисунке 3 показано ускорение при использовании сети Fast Ethernet. Если количество процессоров для вычисления равняется количеству субвыражений, то параллельные вычисления дают лучшую производительность [3].

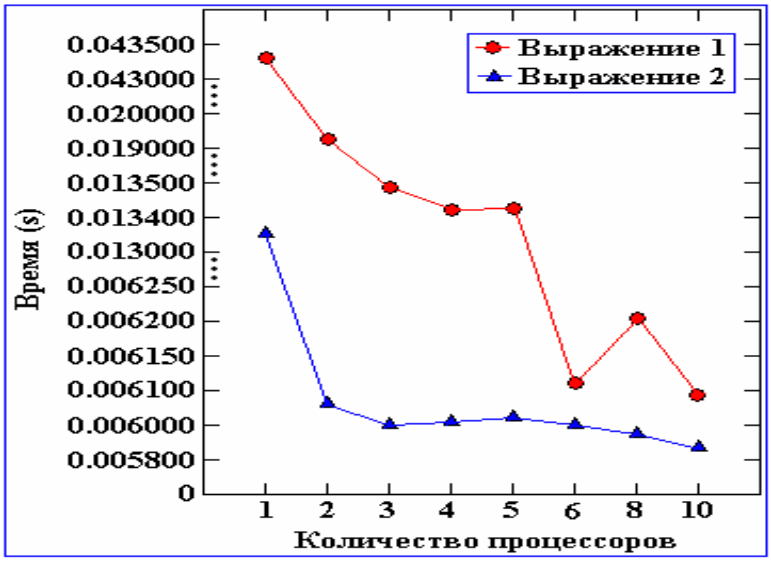

Рис. 2. Зависимость времени вычисления от количества процессоров

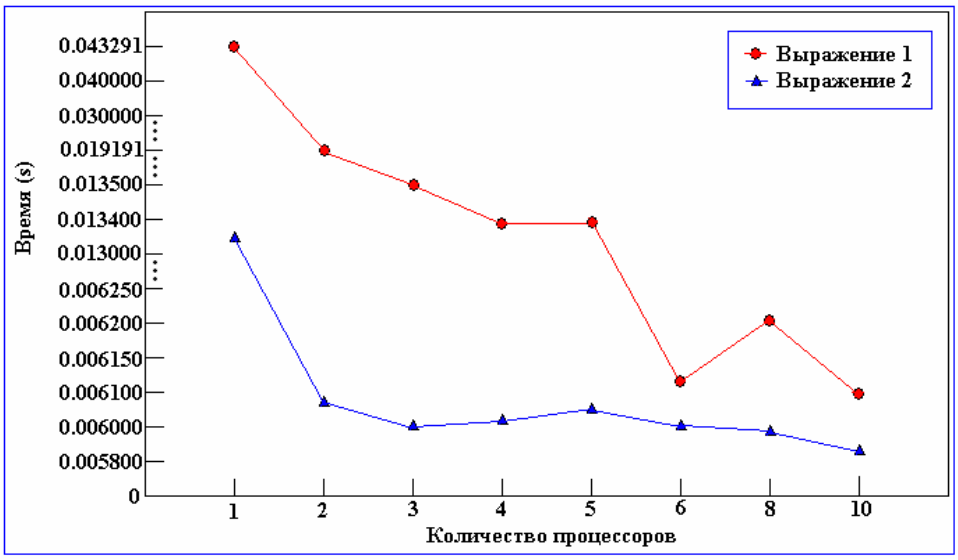

Рис. 3. Ускорение для кластера типа Beowulf с сетью Fast Ethernet

Литература

1. *Буза М.К., Зимянин Л.Ф.* Модели распределенной разделяемой памяти. //II Международная научн. конф. «Сетевые компьютерные технологии» NCT'2005 Сб. тр. – Мн.: Изд.центр БГУ, 2005. – С.15-20.
2. *Барский А. Б.* Параллельные информационные технологии. Учебное пособие. – М.: Интернет–Университет Информационных Технологий; БИНОМ. Лаборатория знаний, 2007. –503 с.
3. *K. Hwang, Z. Xu.* Scalable Parallel Computing: Technology, Architecture, Programming. McGraw-Hill, New York, 1998.