УДК 004.08

Д.В. Бабин, С.М. Вороной
Донецкий государственный институт искусственного интеллекта, Украина

Генетический алгоритм построения расписаний для многопроцессорных вычислительных систем

Рассматривается генетический алгоритм построения расписаний для многопроцессорных вычислительных систем. Приводятся результаты экспериментального исследования эффективности алгоритма планирования параллельного выполнения множества взаимосвязанных задач. Определены параметры алгоритма, при которых сокращается время построения оптимальных по выбранным критериям расписаний.

Введение

Многопроцессорная обработка, как способ повышения общей эффективности вычислений, является одним из перспективных направлений развития вычислительных систем. Задачи, которые необходимо решать при планировании параллельных вычислений в общей постановке являются NP полными. Для сокращения времени решения таких задач используют эвристические подходы. В работе [1] рассмотрены итерационные алгоритмы построения и оптимизации расписаний, в работе [2] предложены методы оперативного планирования периодической обработки информации с совмещением циклов в вычислительных устройствах. Жадный подход к построению расписаний предложен в работе [3]. Такой подход далеко не всегда дает оптимальное решение, но во многих случаях получаемое решение оказывается достаточно близким к оптимальному. Алгоритмы локальной оптимизации расписаний, ориентированные на доводку полученного ранее расписания до локального оптимума, приведены в работе [4]. Одним из эвристических подходов к решению задач планирования является использование генетических алгоритмов. В работе [5] рассмотрена возможность использования генетических алгоритмов для задач планирования вычислений, описаны проблемы, возникающие при использовании генетических алгоритмов для решения таких задач.

Постановка задачи

Будем рассматривать задачу построения расписания в следующем варианте постановки. Исходными данными для построения расписания являются:

- множество работ, на котором определено отношение частичного порядка, задающее ограничения на допустимую последовательность выполнения работ;
- множество ресурсов;
- критерий оптимальности расписания (целевая функция) и функция его вычисления по исходно заданным характеристикам работ и ресурсов.

Расписание выполнения заданного множества работ определено, если для каждой работы заданы:

- привязка к одному из ресурсов;
- порядковый номер выполнения на ресурсе.

Привязка – это всюду определенная на множестве работ функция, которая задает распределение работ по ресурсам. Порядок задает ограничения на последовательность выполнения работ в расписании и является отношением частичного порядка, удовлетворяющим условиям ацикличности и транзитивности. Отношение порядка на множестве работ, распределенных на один и тот же ресурс, является отношением полного порядка.

Расписание является корректным, если выполнены следующие ограничения:

- каждая работа должна быть назначена на ресурс и притом лишь на один ресурс;
- частичный порядок, заданный на исходном множестве работ, не должен нарушаться в расписании;
- расписание должно быть беступиковым.

Описание алгоритма

Для однозначного распределения процессов по процессорам необходимо для каждого процесса знать номер процессора, на котором он будет выполняться. Информацию о процессе выполнения программы удобно представлять в виде строки следующего вида:

- количество чисел в строке (длина строки) соответствует количеству процессов;
- каждая позиция строки соответствует одному процессу, порядковый номер которого равен номеру позиции;
- число в позиции строки показывает, на каком процессоре будет выполняться данный процесс.

Например, строка <12212> обозначает, что для упорядоченного списка из 5 процессов выполнение на двухпроцессорной системе будет осуществляться по схеме: 1-й и 4-й процессы будут выполняться на 1-м процессоре, 2-й, 3-й и 5-й – на 2-м процессоре. Имея эту информацию, информацию о порядке выполнения процессов, а также информацию о времени выполнения каждого процесса $t_i, i = 1,...,N$ можно посчитать для каждого из процессов время начала выполнения $t_i^0, i = 1,...,N$.

Целевая функция. В зависимости от выбранного критерия эффективности целевая функция может представлять собой либо время выполнения программы

$$T = \max(t_i^0 + t_i),\ i = 1,...,N,$$

либо загрузку системы

$$P = \sum_{j=1}^{M} p_j,$$

где p_j представляет собой загрузку j-го процессора в системе, рассчитываемую по формуле

$$p_j = \frac{1}{T}\sum_{i=1}^{N} t_i^j.$$

Запись t_i^j обозначает, что i-й процесс будет выполнен на j-м процессоре. Как правило, в качестве критерия эффективности используется время выполнения программы.

Генетический алгоритм выполняется по классической схеме:

- селекция;
- скрещивание;
- мутация.

Начальная популяция генерируется случайным образом, если предполагается построение нового расписания. Если есть вариант расписания и его необходимо улучшить, за основу для начальной популяции берется имеющееся расписание.

Селекция проводится по пропорциональной схеме. Число потомков прямо пропорционально зависит от значения целевой функции для текущей строки.

Скрещивание проводится по следующей схеме:

- из популяции случайным образом выбираются две строки;
- для каждой позиции строки i $(i = 1,...,N)$ с заданной вероятностью строки обмениваются значениями.

В данном случае такой подход оказался эффективнее одноточечного скрещивания.

Для проведения операции мутации в строке случайным образом выбирается позиция, число в которой заменяется выбранным случайным образом числом из интервала $[1, M]$.

Критерием останова алгоритма служит неулучшение целевой функции для лучшей из строк на протяжении заданного числа шагов. Кроме того, после исследования алгоритма можно выбрать количество итераций алгоритма, необходимое для получения оптимального решения, и в дальнейшем останавливать алгоритм после такого числа итераций.

Экспериментальные исследования алгоритма

Для проведения экспериментальных исследований генетический алгоритм был реализован программно. Для проведения тестовых экспериментов был реализован генератор программ, подлежащих планированию.

Получаемые в результате генерации программы содержали от 30 до 100 процессов. Тестирование проводилось для количества процессоров $M = 2,...,8$.

При исследовании алгоритма составления расписания выполнения тестовых программ получены следующие результаты:

- 30 – 50 итераций позволяют получить оптимальное расписание, в дальнейшем улучшения расписания не происходит либо происходит незначительное улучшение;
- оптимальная вероятность скрещивания составляет ≈ 0,5;
- оптимальная вероятность мутации составляет ≈ 0,05;
- качество получаемого решения на 10 – 20 % лучше первоначального.

Полученные показатели справедливы для популяции размером 20. Увеличение размера популяции позволяет получить решение за меньшее количество шагов, но при этом возрастает количество вычислений на каждом шаге, что является неоправданным.

Выводы

Разработанный алгоритм представляет практический интерес при построении планировщиков многопроцессорных систем. В его разработке использован современный генетический подход.

Исследования алгоритма показали эффективность данного подхода в сравнении с итерационными методами [1]. Дополнительным преимуществом данного метода является независимость от структуры вычислительной системы, количество процессоров в системе является входными данными для алгоритма.

К сожалению, данный алгоритм не применим в распределенных многомашинных вычислительных системах, поскольку для них немаловажное влияние на расписание оказывает время, затрачиваемое на пересылку заданий. В дальнейшем возможно усовершенствование данного алгоритма для такого класса вычислительных систем.

Литература

1. Костенко В.А. Оценки сложности и качества различных итерационных алгоритмов построения расписаний // Фак-т ВМиК МГУ им. М.В. Ломоносова. – М. – 2004. – С. 161-167.
2. Хачумов В.М. Периодические расписания с совмещением циклов обработки данных // Труды Первой Всерос. науч. конф. – Москва: МГУ. – 2003. – С. 522-528 // http://lvk.cs.msu.su/mco/
3. Вавинов С.В., Костенко В.А. Параметризованный жадный алгоритм построения статических расписаний // Труды Первой Всерос. науч. конф. – Москва: МГУ. – 2003. – С. 323-324 // http://lvk.cs.msu.su/mco/
4. Калашников А.В., Костенко В.А. Алгоритмы локальной оптимизации расписаний // Труды Первой Всерос. науч. конф. – Москва: МГУ. – 2003. – С. 381-389 // http://lvk.cs.msu.su/mco/
5. Костенко В.А. Возможности генетических алгоритмов для решения задач синтеза архитектур и планирования параллельных вычислений // Труды Третьей Междунар. науч. конф. «Дискретные модели в теории управляющих систем» (Красновидово'98). – М.: Диалог МГУ. – 1998. – С. 53-58.

Д.В. Бабин, С.М. Вороной

Генетичний алгоритм побудови розкладів для багатопроцесорних обчислювальних систем

Розглядається генетичний алгоритм побудови розкладів для багатопроцесорних обчислювальних систем. Наводяться результати експериментального дослідження ефективності алгоритму планування паралельного виконання великої кількості взаємозв'язаних задач. Визначені параметри алгоритму, при яких скорочується час побудови оптимальних за обраними критеріями розкладів.

D.V. Babin, S.M. Voronoy

The Genetic Algorithm Making Timetabless for Multiprocessor Computer Systems

The genetic algorithm making timetables for multiprocessor computer systems is considered. The results of experimental research of the efficiency of algorithm of planning parallel solving many interrelated tasks are given. The parameters of algorithm are determined which allow to reduce the time of making timetables according to the selected criteria.

Статья поступила в редакцию 15.02.2005.