для подогрева дизельного топлива в баке, и нагревательные элементы, установленные на топливной магистрали, и фильтрах очистки топлива [3].

Термогенераторы на кожухе поддона картера, за счет возникающей разности температур на пластинах, вырабатывают электрическую энергию, обеспечивающую питание нагревательных элементов, устанавливаемых на приборах системы питания (рис. 2).

Рис. 2. Размещение элементов системы предпускового подогрева на автомобиле Урал-4320-31

- 1 – электронагреватель в термоколпаке; 2 – подогреватель гибкий ленточный;
3 – подогреватель бандажный на фильтре грубой очистки топлива;
4 – подогреватель бандажный на фильтре тонкой очистки топлива

Согласно результатам проведенной технико-экономической оценки эффективности определено, что использование разработанной системы на автомобилях многоцелевого назначения частей и соединений постоянной готовности позволит сократить общее время приведения техники в готовность к использованию на 15–20%.

1. Левин А.Ф. Эксплуатация электрооборудования автомобилей зимой. М.: Транспорт, 1971.
2. Чижков Ю.П. Электрооборудование автомобилей. Учебник./Ю.П.Чижков, С.В. Акимов М.: За рулем, 2007.
3. Николаенко А.А. Разработка технического решения по облегчению пуска и работы двигателя Урал 4320.31в условиях низких температур. Дипломный проект. Военный учебно-научный центр Сухопутных войск. Омск. 2012

Уласень А.Ф., Уласень А.А.

Разработка параллельных алгоритмов при решении сложных научно-технических задач

*Военная академия ВПВО ВС РФ
(Россия, Смоленск)*

doi: 10.18411/sr-10-02-2018-11

idsp: 000001:sr-10-02-2018-11

Аннотация

В статье рассматриваются особенности программирования на высокопроизводительных вычислительных системах и обосновывается эффективность организации параллельных вычислений при увеличении сложности решаемых научно-технических задач.

Ключевые слова: алгоритм, вычисление, программа, параллелизм, задача.

При разработке параллельных алгоритмов решения сложных научно-технических задач принципиальным моментом является анализ эффективности использования параллелизма, состоящий обычно в оценке получаемого ускорения процесса вычислений – сокращения времени решения задачи. Формирование подобных оценок ускорения может осуществляться применительно к выбранному вычислительному алгоритму, т. е. оценка эффективности распараллеливания конкретного алгоритма. Другой важный подход может состоять в построении оценок максимально возможного ускорения процесса решения задачи конкретного типа, т. е. оценка эффективности параллельного способа решения задачи. Модель вычислений в виде графа «операции–операнды» может использоваться для описания существующих информационных зависимостей в алгоритмах решения задач, что позволяет построить оценки эффективности максимально возможного параллелизма, которые могут быть получены в результате анализа имеющихся моделей вычислений[1,2].

Множество операций, выполняемых в исследуемом алгоритме решения вычислительной задачи, и существующие между операциями информационные зависимости возможно представить в виде ациклического ориентированного графа $G=(V, R)$. Здесь $V=\{1, \dots, |V|\}$ – множество вершин графа, представляющих выполняемые операции алгоритма, а R – множество дуг графа, при этом дуга $r=(i, j)$ принадлежит графу только, если операция j использует результат выполнения операции i .

Можно утверждать, что для выполнения выбранного алгоритма решения задачи могут быть использованы разные схемы вычислений и построены соответственно разные вычислительные модели. При этом разные схемы вычислений обладают различными возможностями для распараллеливания и тем самым при построении модели вычислений может быть поставлена задача выбора наиболее подходящей для параллельного исполнения вычислительной схемы алгоритма. В рассматриваемой вычислительной модели алгоритма вершины без входных дуг могут использоваться для задания операций ввода, а вершины без выходных дуг – для операций вывода. Обозначим через V множество вершин графа без вершин ввода, а через $d(G)$ – диаметр (длину максимального пути) графа. Операции алгоритма, между которыми нет пути в рамках выбранной схемы вычислений, могут быть выполнены параллельно[3].

Возможный способ описания параллельного выполнения алгоритма может состоять в следующем.

Пусть P есть количество процессоров, используемых для выполнения алгоритма. Тогда для параллельного выполнения вычислений необходимо задать множество-расписание $H_p = \{(i, P, t): i \in V\}$, в котором для каждой операции $i \in V$ указывается номер используемого для выполнения операции процессора P_i и время начала выполнения операции t_i .

Вычислительная схема алгоритма G совместно с расписанием H_p может рассматриваться как модель параллельного алгоритма $A_p(G, H_p)$, исполняемого с использованием процессоров. Время выполнения параллельного алгоритма определяется максимальным значением времени, используемым в расписании,

$$T_p(G, H_p) = \max_{i \in V} (t_i + 1).$$

Для выбранной схемы вычислений желательно использование расписания, обеспечивающего минимальное время исполнения алгоритма,

$$T_p(G) = \min_{H_p} T_p(G, H_p).$$

Уменьшение времени выполнения может быть обеспечено и путем подбора наилучшей вычислительной схемы

$$T_p = \min_G T_p(G).$$

Оценки $Tr(G, Np)$, $Tr(G)$ и Tr могут быть использованы в качестве показателей времени выполнения параллельного алгоритма. Кроме того, для анализа максимально возможного параллелизма можно определить оценку наиболее быстрого исполнения алгоритма:

$$T_{\infty} = \min_{p \geq 1} T_p.$$

Оценку T_{∞} можно рассматривать как минимально возможное время выполнения параллельного алгоритма при использовании неограниченного количества процессоров (концепция вычислительной системы с бесконечным количеством процессоров широко используется при теоретическом анализе параллельных вычислений).

Оценка T_1 определяет время выполнения алгоритма при использовании одного процессора и представляет тем самым время выполнения последовательного варианта алгоритма решения задачи. Построение подобной оценки является важной задачей при анализе параллельных алгоритмов, поскольку она необходима для определения эффекта использования параллелизма, т. е. ускорения времени решения задачи. Очевидно, что

$$T_1(G) = |\bar{V}|,$$

где $|\bar{V}|$ есть количество вершин вычислительной схемы G без вершин ввода. Важно отметить, что если при определении оценки T_1 ограничиться рассмотрением только одного выбранного алгоритма решения задачи и использовать величину

$$T_1 = \min_G T_1(G),$$

то получаемые при использовании такой оценки показатели ускорения будут характеризовать эффективность распараллеливания выбранного алгоритма. Для оценки эффективности параллельного решения исследуемой задачи время последовательного решения следует определять с учетом различных последовательных алгоритмов, т. е. использовать величину $T_1^* = \min T_1$, где операция минимума берется по множеству всех возможных последовательных алгоритмов решения данной задачи.

В свою очередь, если в программе относительно немного последовательных операций, но доступные процессоры разнородны по своей производительности, то будет такой момент, когда кто-то из них еще трудится, а кто-то уже все сделал и бесполезно простаивает в ожидании. Если разброс в производительности процессоров большой, то и эффективность всей системы при равномерной загрузке будет крайне низкой. Наряду с этим, если все процессоры одинаковы и выполнили свою работу, но результатами чаще всего надо обмениваться для продолжения вычислений, а на передачу данных уходит время, в это время процессоры опять простаивают. Кроме указанных, есть еще большое количество факторов, влияющих на эффективность выполнения параллельных программ, причем все они действуют одновременно, а значит, все в той или иной степени должны учитываться при распараллеливании [4].

Таким образом, разработка параллельных программ является весьма трудоемким процессом, поэтому, прежде чем приступить к этой работе, важно правильно оценить как ожидаемый эффект от распараллеливания, так и трудоемкость выполнения этой работы. Очевидно, что без распараллеливания не обойтись при программировании алгоритмов решения задач управления боевыми действиями, поскольку дальнейшее совершенствование средств нападения, увеличение объема обрабатываемой информации, уменьшение времени на решения задач управления требуют использования многопроцессорных систем с параллельной обработкой данных.

Однако необходимо отметить, что для небольших задач зачастую оказывается, что параллельная версия работает медленнее, чем однопроцессорная. В этой связи естественным способом решения проблем параллельного программирования для высокопроизводительных вычислительных систем является создание стандартов как для

вычислительной техники, так и для программного обеспечения. При этом требуется выделить класс задач, которые необходимо решать на высокопроизводительной системе, а также обеспечить конструирование самой системы для выделенного класса таких задач. Также следует помнить, что необходимо создать подходящее математическое обеспечение для конкретного класса задач на выбранной системе. И, наконец, написать программу для данной конкретной задачи с учетом перечисленных факторов.

1. Основы параллельных вычислений для многопроцессорных вычислительных систем: Гергель В. П., Стронгин Р. Г. – Н.Новгород: ННГУ, 2003. – 302 с.
2. Высокопроизводительные вычисления: Баденко В. Л. – СПб.: Изд-во Политехн. ун-та, 2010. – 180 с.
3. Уласень, А. Ф. Научно-методические основы обеспечения надежности информационно-управляющих систем на основе супер-ЭВМ: монография / А. Ф. Уласень. – Смоленск: ВА ВПВО ВС РФ, 2016. – 220 с.
4. Константиновский В. М., Лопашинов П. М., Шалин С. А. Средства разработки и отладки программного обеспечения для систем, работающих в режиме реального времени. ВСПЭ, серия РЛТ, 2000, вып. 1, С. 7 21.