2. Критерии и параметры оценки параллелизма

Процесс программирования можно рассматривать как многоэтапный переход от задачи к представлению выбранного метода и способа ее решения на конкретном языке программирования [13, 21]. Показатели, характеризующие качество разработанной программы, степень достижения предъявляемых к программе требований (времени вычисления, объему требуемой памяти и др.) в большой степени определяются методом решения задачи, выбором языка программирования и собственно уровнем программистского искусства, которое заключается в умелом использовании языковых средств для точного представления метода решения задачи и построение эффективной программы.

Для оценивания сложности и качества параллельных программ используется целый ряд критериев и параметров, от которых они зависят.

2.1. Коэффициент ускорения

Коэффициент ускорения имеет принципиальное значение и обычно определяется как отношение времени выполнения параллельной программы на одном компьютере $t(1)$ ко времени ее выполнения на ВС с N компьютерами или процессорами: $t(N), C(N) = \frac{t(1)}{t(N)}$.

Согласно закону Амдала предельное ускорение решения задачи в параллельной форме на ВС с N узлами определяется соотношением: $C(N) = \frac{1}{(\alpha + \frac{(1-\alpha)}{N})}$, где α - доля операций в программе, которые выполняются последовательно. Предельное ускорение при неограниченном N, очевидно, равно $\frac{1}{\alpha}$. В реальности ускорение меньше предельного из-за затрат времени

на управление параллельным выполнением программы на ВС, реализацию обменов данными и д.р.

На практике более важно понять, как ведет себя ускорение в зависимости от сложности задачи и количества компьютеров (процессоров) в ВС.

Определение. Коэффициент ускорения $C(x_1, x_2,.., x_k, N)$ есть функция, определяющая ускорение решения задачи в параллельной форме в зависимости от параметров $x_1, x_2,.., x_k$ сложности задачи и количества узлов ВС.

Для многих задач часто достаточно просто определить предельное значение $C(x_1, x_2,.., x_k, N)$, если предполагать, что количество компьютеров или узлов ВС не ограничено (в этом случае не возникает задержек при выполнении параллельной программы из-за отсутствия необходимых ресурсов) и не учитывать системные издержки на организацию выполнения параллельной программы на ВС (обменные взаимодействия, управление и др. [11, 12]). Именно при этих предположениях обычно определяется коэффициент ускорения для многих методов и параллельных программ.

В качестве параметров $x_1, x_2,.., x_l$, характеризующих вычислительную сложность задачи, во многих случаях используются параметры, определяющие размерность задачи, которые одновременно являются и аргументами программы, представляющей метод ее решения.

Рассмотрим пример простой задачи вычисления значений *n!*, используя для этого параллельный метод разбиения отрезка *[1÷n]* пополам, по программе:

$Fact(i, j) = if\ i = j\ then\ 1\ else\ Fact\left(i, \left\lfloor \frac{i+j}{2} \right\rfloor\right) \times Fact\left(\left\lfloor \frac{i+j}{2} \right\rfloor + 1, j\right)$, где $\lfloor a \rfloor$ – ближайшее целое к *a*. Очевидно, $F(1, n) = n!$.

Если попытаться реализовать процесс вычисления значения *n!* по этой программе, используя все возникающие при этом возможности

распараллеливания, нетрудно показать, что этот процесс следует очевидной схеме (рис. 3):

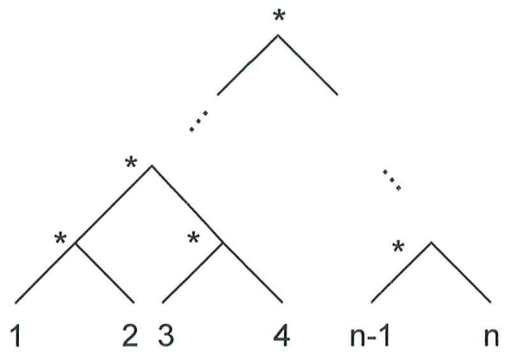

Рис. 3. Схема параллельного вычисления *n!*.

Параметр *n* – аргумент приведенной выше функции – характеризует вычислительную сложность задачи, а коэффициент *C(n)* (при неограниченном числе узлов ВС) ведет себя как $O\left(\frac{n}{log_2 n}\right)$ и при неограниченном увеличении n стремится к бесконечности.

Таким образом, с усложнением задачи (увеличением *n*) можно неограниченно сокращать время выполнения этой программы при условии неограниченных ресурсов.

Можно привести множество примеров других задач (перемножение матриц, решение систем линейных уравнений, численное интегрирование и др.), для которых с увеличением их сложности коэффициент ускорения возрастает неограниченно [11, 20].

Параллельные программы, у которых коэффициент ускорения увеличивается неограниченно с увеличением сложности задачи, в [11] названы методами или программами с неограниченным параллелизмом.

2.2. Глубина и степень распараллеливания

В [11] введен еще один параметр, который определяет глубину распараллеливания метода (или представляющего его алгоритма) и который интуитивно характеризует в среднем вычислительную сложность

компонентов параллельной программы, которые рассматриваются как самостоятельные ее части и могут выполняться одновременно. В литературе этот параметр часто называется зернистость параллелизма.

Определение. Глубиной распараллеливание d метода назовем усредненную вычислительную сложность компонентов, которые рассматриваются в параллельной программе как самостоятельные процессы, идентифицируемые и планируемые при ее выполнении на ВС.

Обратную к d величину будем определять как степень распараллеливания, которая характеризует усредненное количество компонентов параллельной программы, выполняемых одновременно.

Рассмотрим простой пример перемножения квадратных матриц размера n и определим предельный коэффициент ускорения процесса параллельного решения этой задачи для различных d и неограниченного количества компьютеров в ВС.

1. d_1 – сложность вычисления одной строки результирующей матрицы, а степень распараллеливания – одновременное вычисления всех строк матрицы: $C(n,d_1)=\dfrac{n^2\left(n\times t_{ум}+(n-1)\times t_{сл}\right)}{n\left(n\times t_{ум}+(n-1)\times t_{сл}\right)}=n$, где $t_{ум}$ и $t_{сл}$ - время выполнения операций умножения и сложения соответственно.

2. d_2 – сложность вычисления одного элемента результирующей матрицы, а степень распараллеливания – одновременное вычисление всех элементов результирующей матрицы: $C(n,d_2)=\dfrac{n^2\left(n\times t_{ум}+(n-1)\times t_{сл}\right)}{n\times t_{ум}+(n-1)\times t_{сл}}=n^2$

3. d_3 – сложность выполнения операции умножения, а степень распараллеливания – одновременное вычисление выполнение всех операций умножения при одновременном вычислении всех элементов результирующей матрицы: $C(n,d_3)=\dfrac{n^2\left(n\times t_{ум}+(n-1)\times t_{сл}\right)}{t_{ум}+(n-1)\times t_{сл}}=O\left(r_1\times n^3\right)$, где r_1 – некоторая константа, $r_1\le 1$, *а* $O(x)$ – означает близкое к x значение.

4. d_4 – усредненная сложность выполнения операции умножения и параллельного вычисления суммы $\sum_{k=1}^{n} M'[i,k] \times M''[k,j]$, при вычислении элементов результирующей матрицы, степень распараллеливания определяется исходя из условия одновременного выполнения всех операций умножения при последующем параллельном вычислении приведенной выше суммы делением отрезка [1÷n] пополам (см. пример параллельного вычисления функции факториал выше): $C(n,d_4) = \frac{n^2\left(n \times t_{ум} + (n-1) \times t_{сл}\right)}{t_{ум} + log_n n \times t_{сл}} = O\left(r_2 \times n^3\right)$, где $r_2 \leq r_1$.

Нетрудно видеть, что во всех четырех случаях коэффициент ускорения неограниченно растет с увеличением n. Однако, производная (ускорение) этого роста различна для различных d, причем, например, для d_1, d_2, d_3 коэффициент ускорения растет как $O(n)$, $O(n^2)$ и $O(r_1 \times n^3)$ соответственно. Заметим, что степень распараллеливания ведет себя как неубывающая функция в зависимости от величины $\frac{1}{d}$ и всегда существует предельное ее значение, определенное предельной и всегда ограниченной предельной глубиной распараллеливания.

Варьирование степени распараллеливания задачи является чрезвычайно важным при оптимизации процесса выполнения ее на ВС с позиции минимизации времени выполнения и использования ресурсов [11, 12]. Это может происходить как на стадии разработки параллельной программы и ее статическом планировании на ВС, так и на стадии выполнения, когда динамически варьируется степень распараллеливания с целью увеличения фронта готовых к выполнению процессов и , как следствие, увеличения загруженности ВС [12].

2.3. *Интенсивность обменных взаимодействий*

Эффективность параллельной работы ВС существенно зависит от пропускной способности коммуникаций и интенсивности обменных взаимодействий между ее компонентами в процессе выполнения параллельной программы. Поэтому важно понять, каким образом при изменении степени распараллеливания будет изменяться интенсивность обменных взаимодействий при выполнении программы на ВС.

Определение. Определим интенсивность обменных взаимодействий для параллельной программы при глубине распараллеливания d как функцию $\lambda(\bar{x},d,N)=\frac{n_{обмен.}}{t(N)}$, где $n_{обмен.}$ - количество обменных взаимодействий при выполнении параллельной программы сложности $\bar{x}=x_1,x_2,..,x_k$ на ВС с N компьютерами (процессорами).

Если количество узлов ВС N таково, что задержек при выполнении параллельной программы не существует из-за недостатка компьютеров или процессоров в ВС, то будем обозначать $\lambda(\bar{x},d)$ как предельное значение интенсивности обменов.

Легко проверить, что для рассмотренных примеров параллельного вычисления $n!$ и перемножения матриц, если полагать, что они осуществляются на ВС с разделенной памятью, $\lambda(\bar{x},d)$ ведет себя в зависимости от d и $\bar{x}$ практически так же, как и степень распараллеливания.

График на рис. 4. иллюстрирует общий характер реальной зависимости времени выполнения параллельной программы $t(N)$ для больших значений N на ВС с разделенной памятью от степени распараллеливания [11].

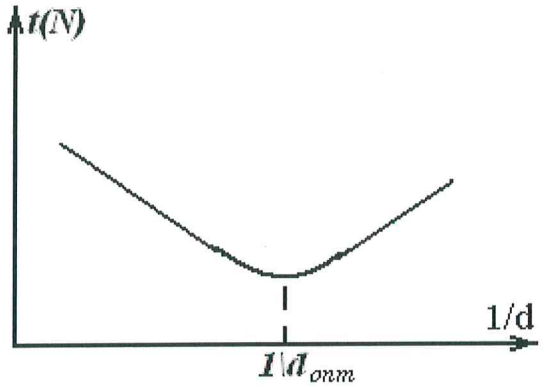

Рис. 4. Общий характер зависимости *t(N)* от распараллеливания

Увеличение степени распараллеливания до определенного предела $\frac{1}{d_{опт}}$ временя выполнения параллельной программы сначала уменьшается за счет большей загрузки узлов ВС. Однако при этом увеличивается интенсивность обменов между узлами ВС и при $d < d_{опт.}$ начинается естественное замедление вычислений из-за увеличения задержек на реализацию обменов.

Может показаться, что реализация параллельных вычислений на ВС с общей памятью устраняет проблему снижения эффективности их работы из-за обменных взаимодействий между компьютерами. Однако, достаточно рассмотреть пример вычисления значений функции *n!* в абсолютно параллельной форме согласно рисунку 1, чтобы понять, что при реализации этой стратегии к общей памяти могут одновременно обращаться порядка $\frac{n}{2}$ команд. С увеличением *n* ограниченная пропускная способность системы «процессоры - память» станет узким местом, существенно уменьшая эффект от распараллеливания.

Более того, для параллельной реализации программы, как правило, требуется гораздо больший объем оперативной памяти (в рассмотренном примере порядка $\frac{n}{2}$ ячеек памяти требуется для хранения промежуточных результатов в отличие от нескольких ячеек, необходимых для выполнения *n!* последовательной программой).

2.4. *Коэффициент использования ресурсов*

На практике при организации параллельных вычислений важно знать не только коэффициент ускорения, получаемый при выполнении параллельной программы на реальной ВС, но и то, как при этом используются ее ресурсы.

Для определения эффективности использования ресурсов обычно используют отношение $C(x_1, x_2, ..., x_m, N)/N$, причем часто считают, что это отношение не может быть больше единицы, полагая, что ускорение не может быть больше N – количества узлов, процессоров и компьютеров ВС.

В действительности, это не так, поскольку для сложных задач, требующих большого объема памяти, ускорение может быть больше N. Это связано с тем, что для сложных задач время выполнения программы на одном компьютере (предполагая, что его память не больше объема памяти всех компьютеров ВС) может существенно увеличится из-за большой интенсивности обменов между оперативной памятью и дисками коэффициент ускорения может быть больше N. Это в реальности наблюдается даже на простых задачах перемножения матриц и решения систем линейных уравнений большого размера [11, 20].

Более точное представление об использовании реальных ресурсов ВС дает усредненное значение из загруженности на интервале T выполнения параллельной программы: $\sum_{i=1}^{N} \frac{1}{T} \int_0^T Li(t)dt$, где $Li(t)$, $i = 1,2,..N_K$, загруженность i-го ресурса, отнесенного к множеству ресурсов N_K типа k [12]. Обычно о загруженности ВС судят по загруженности только ее вычислительных узлов, что часто оправдано для вычислительных задач.

Интересно проследить общий характер изменения показателя эффективности использования ресурсов ВС для задачи заданной сложности с увеличением – количества компьютеров (процессоров) ВС.

Очевидно, если не изменяется глубина распараллеливания и сохраняется общая схема организации выполнения параллельной программы на ВС, этот показатель сначала должен увеличиваться (точнее, не уменьшаться) с увеличением N всегда будет существовать некоторое оптимальное значение N, больше которого уже нельзя уменьшить время выполнения программы (нужна задача большей сложности или требуется уменьшение глубины распараллеливания). Это вытекает из анализа изменения коэффициента ускорения в зависимости от N [11, 20].

Общий вывод из изложенного в этом пункте состоит в том, что при построении эффективных параллельных программ необходимо не только серьезное внимание уделять разработке параллельных методов решения задач, но и уметь их анализировать и «приспосабливать» к «масштабу» ВС и ее техническим возможностям с тем, чтобы достигался максимальный эффект.