Высокопроизводительные параллельные вычисления

Лекция №2

Тема: Моделирование и анализ параллельных вычислений

Козбакова Айнур Холдасовна– PhD, профессор, кафедра «Информационные системы» Факультет «Инжиниринг и информационных технологии» 01.09.2025 г.

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 (оценка эффективности распараллеливания конкретного алгоритма). Другой важный подход состоит в построении оценок максимально возможного ускорения процесса решения задачи конкретного типа (оценка эффективности параллельного способа решения задачи). ля описания существующих информационных зависимостей в выбираемых алгоритмах решения задач может быть использована модель в виде графа "операции-операнды" [4] (дополнительная информация по моделированию параллельных вычислений может быть получена в [5-9]). Для уменьшения сложности излагаемого материала при построении модели будет предполагаться, что время выполнения любых вычислительных операций является одинаковым и равняется 1 (в тех или иных единицах измерения); кроме того, принимается, что передача данных между вычислительными устройствами выполняется мгновенно без каких-либо затрат времени (что может быть справедливо, например, при наличии общей разделяемой памяти в параллельной вычислительной системе).

Рассмотрим *модель вычислений* в виде *графа "операции – операнды"*, которая может использоваться для описания существующих информационных зависимостей в выбираемых алгоритмах решения задач. Будет предполагаться, что *время выполнения* любых вычислительных операций является одинаковым и равняется 1 (в тех или иных единицах измерения). Кроме того, принимается, что передача данных между вычислительными устройствами выполняется мгновенно без каких–либо затрат времени (что может быть справедливо, например, при наличии общей разделяемой памяти в параллельной вычислительной системе).

Представим множество операций, выполняемых в исследуемом алгоритме решения вычислительной задачи, и существующие между операциями информационные зависимости в виде ациклического ориентированного графа G = (V, R), где V = {1,...,|V|}есть множество вершин графа, представляющих выполняемые операции алгоритма, а R есть множество дуг графа.

Операции алгоритма, между которыми нет пути в рамках выбранной схемы вычислений, могут быть выполнены параллельно. 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*)

$H_p = \{(i,P_i,t_i): i \in V\}$,

в котором для каждой операции $i \in V$ указывается номер используемого для выполнения операции процессора P_i и время начала выполнения операции t_i. Для того чтобы *расписание* было реализуемым, необходимо выполнение следующих требований при задании множества H_p:

1. $\forall i,j \in V : t_i = t_j \Rightarrow P_i \neq P_j$, т.е. один и тот же процессор не должен назначаться разным операциям в один и тот же момент;
2. $\forall (i,j) \in R t_j \geq t_i + 1$, т.е. к назначаемому моменту выполнения операции все необходимые данные уже должны быть вычислены.

2.1 Определение времени выполнения параллельного алгоритма

Вычислительная схема алгоритма G совместно с *расписанием* H_p может рассматриваться как модель параллельного алгоритма A_p (G, H_p), исполняемого с использованием p процессоров. Время выполнения параллельного алгоритма определяется максимальным значением времени, применяемым 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 $T_p(G,H_p)$, $T_p(G)$ и T_p могут быть применены 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, обычно называемой ***паракомпьютером***, широко применяется 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 (ускорения времени решения задачи). Очевидно, что

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

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

$$T_1 = \min_{G} T_1(G),$$

то получаемые при такой оценке показатели ускорения будут характеризовать *эффективность* распараллеливания выбранного алгоритма. Для оценки эффективности параллельного решения исследуемой вычислительной 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.

Приведем без доказательства теоретические положения, характеризующие свойства оценок времени выполнения параллельного алгоритма.

Теорема 1. Минимально возможное *время выполнения* параллельного алгоритма определяется длиной максимального пути вычислительной схемы алгоритма, т.е.

$T_\infty(G)=d(G)$.

Теорема 2. Пусть для некоторой вершины вывода в вычислительной схеме алгоритма существует путь из каждой вершины ввода. Кроме того, пусть входная степень вершин схемы (количество входящих дуг) не превышает 2. Тогда минимально возможное время выполнения параллельного алгоритма ограничено снизу значением

$T_\infty(G)=log_2 n,$

где n есть количество вершин ввода в схеме алгоритма.

Теорема 3. При уменьшении числа используемых процессоров *время выполнения* алгоритма увеличивается пропорционально величине уменьшения количества процессоров, т.е.

$$\forall q=cp,\ 0<c<1 \Rightarrow T_p \leq cT_q.$$

Теорема 4. Для любого количества используемых процессоров справедлива следующая верхняя оценка для времени выполнения параллельного алгоритма

$$\forall p \Rightarrow T_p<T_\infty+T_1/p.$$

Теорема 5. Времени выполнения алгоритма, которое сопоставимо с минимально возможным временем T_∞, можно достичь при количестве процессоров порядка $p\sim T_1/T_\infty$, а именно,

$$p \geq T_1/T_\infty \Rightarrow T_p \leq 2T_\infty.$$

При меньшем количестве процессоров *время выполнения* алгоритма не может превышать более чем в 2 раза наилучшее время вычислений при имеющемся числе процессоров, т.е.

$$p < T_1/T_\infty \Rightarrow \frac{T_1}{p} \leq T_p \leq 2fracT_1p.$$

Приведенные утверждения позволяют дать следующие рекомендации по правилам формирования параллельных алгоритмов:

1. при выборе вычислительной схемы алгоритма должен использоваться граф с минимально возможным диаметром (см. теорему 1);
2. для параллельного выполнения целесообразное количество процессоров определяется величиной $p\sim T_1/T_\infty$ (см. теорему 5);
3. *время выполнения* параллельного алгоритма ограничивается сверху величинами, приведенными в теоремах 4 и 5.