Метрики параллельных вычислений

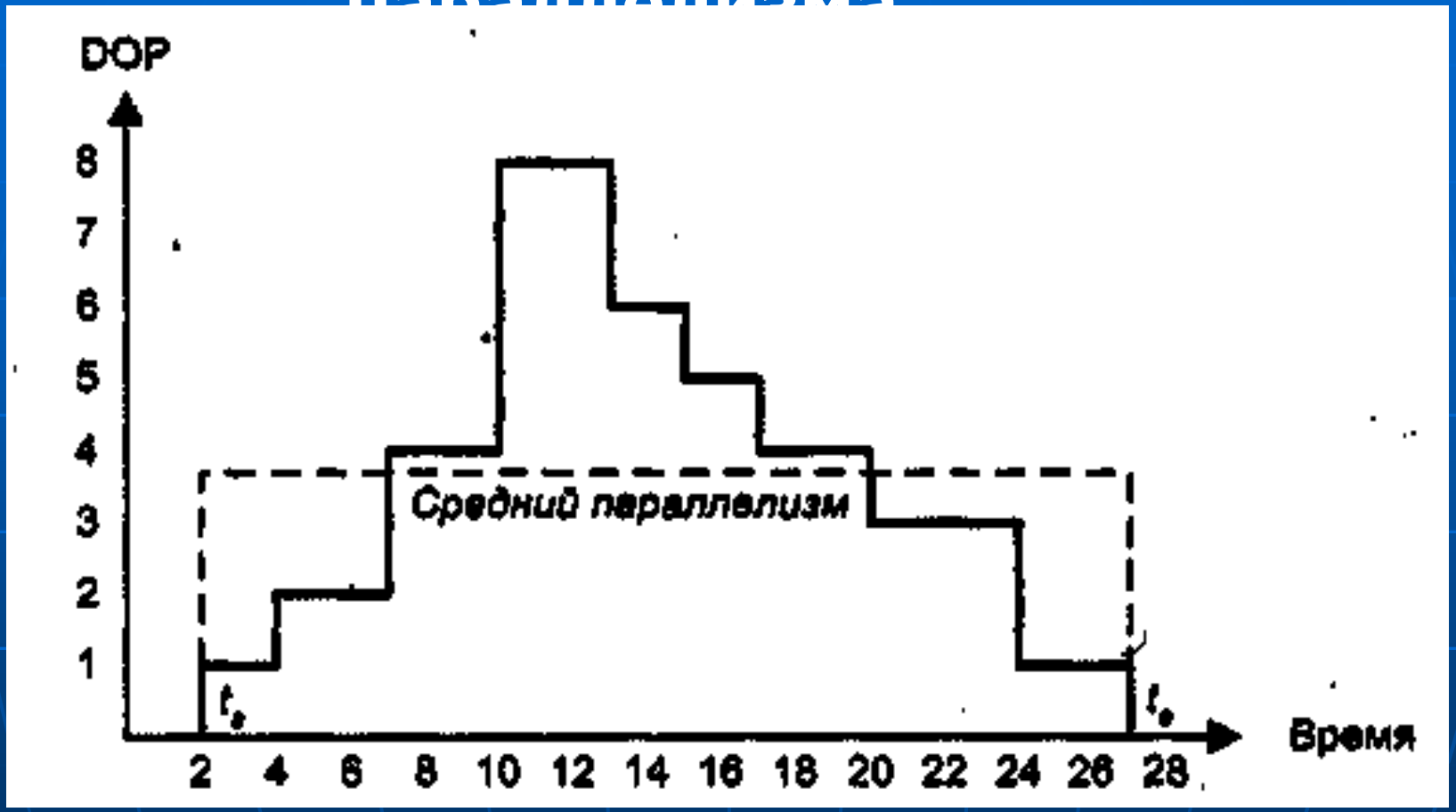
Профиль параллелизма программы

Число процессоров многопроцессорной системы, параллельно участвующих в выполнении программы в каждый момент времени t, определяют понятием ***степень параллелизма***

D(t) (DOP, Degree Of Parallelism)

Графическое представление параметра D(t) как функции времени называют ***профилем параллелизма программы***.

Профиль параллелизма



Изменения в уровне загрузки процессоров за время наблюдения зависят от многих факторов (алгоритма, доступных ресурсов, степени оптимизации, обеспечиваемой компилятором и т. д.)

Общий объём работы

Общий объем вычислительной работы W (команд или вычислений), выполненной начиная со стартового момента **ts** до момента завершения **te**, пропорционален площади под кривой профиля параллелизма:

$$W = \Delta \int_{t_s}^{t_e} D(t)dt.$$

Интеграл часто заменяют дискретным эквивалентом:

$$W = \Delta \sum_{i=1}^{m} i \times t_i,$$

где t_i — общее время, в течение которого $D = i$, а $\sum_{i=1}^{m} t_i = t_e - t_s$ — общее затраченное время.

Средний параллелизм

Средний параллелизм **A** определяется как:

$$A = \frac{1}{t_e - t_s} \int_{t_s}^{t_e} D(t)dt.$$

В дискретной форме это можно записать как:

$$A = \frac{\sum_{i=1}^{m} i \times t_i}{\sum_{i=1}^{m} t_i}.$$

A = (1x5 + 2 x 3 + 3 x 4 + 4 x 6 + 5 x 2 + 6x2 + 8x3) /
(5+ 3 + 4 + 6 + 2 + 2 + 3) = 93/25 = 3,72.

Ускорение

Рассмотрим параллельное выполнение программы со следующими характеристиками:

- **O(n)** — общее число операций (команд), выполненных на n-процессорной системе;
- **T(n)** — время выполнения O(n) операций на n-процессорной системе в виде числа квантов времени.

Примем, что в однопроцессорной системе T(1) = O(1).

Ускорение (speedup), или точнее, среднее ускорение за счет параллельного выполнения программы (без учета коммуникационных издержек) **S(n)** определяется как:

$$S(n) = \frac{T(1)}{T(n)}.$$

Эффективность

Эффективность (efficiency) **E(n)** n-процессорной системы – это ускорение на один процессор, определяемое выражением:

$$E(n) = \frac{S(n)}{n} = \frac{T(1)}{n \times T(n)}.$$

Эффективность обычно отвечает условию 1/n < =E(n) < = n.

Довольно часто организация вычислений на n процессорах связана с существенными издержками. Поэтому имеет смысл ввести понятие ***избыточности*** (redundancy) в виде:

$$R(n) = \frac{O(n)}{O(1)} = \frac{1}{E(n)} - 1.$$

Очевидно, что 1 <= R(n) <= n.

Коэффициент утилизации

Определим еще одно понятие, коэффициент полезного использования или утилизации (utilization) **U(n),** как

$$U(n) = R(n) \times E(n) = \frac{O(n)}{n \times T(n)}.$$

Пример. Пусть наилучший из известных последовательных алгоритмов занимает 8 с, а параллельный алгоритм занимает на пяти процессорах 2 с. Тогда:

S(n) = 8/2 = 4

E(n)= 4/5 = 0,8

R(n)= 1/0,8 - 1=0,25

Если ускорение, достигнутое на n процессорах, равно n, то говорят, что алгоритм показывает ***линейное ускорение***.

В исключительных ситуациях ускорение S(n) может быть больше, чем n. В этих случаях иногда применяют термин ***суперлинейное ускорение.***

Факторы, ограничивающие ускорение

Программные издержки. Параллельным алгоритмам присущи добавочные программные издержки - дополнительные индексные вычисления (декомпозиция данных и распределение их по процессорам; различные виды учетных операций).

Издержки из-за дисбаланса загрузки процессоров. Каждый из процессоров должен быть загружен одинаковым объемом работы, иначе часть процессоров будет ожидать, пока остальные завершат свои операции. Эта ситуация известна как *дисбаланс загрузки.*

Коммуникационные издержки. Любые коммуникации между процессорами снижают ускорение. В плане коммуникационных затрат важен уровень гранулярности, определяющий объем вычислительной работы, выполняемой между коммуникационными фазами алгоритма.

Для уменьшения коммуникационных издержек выгоднее, чтобы вычислительные гранулы были достаточно крупными и доля коммуникаций была меньше.

Качество параллельного выполнения

Еще одним показателем параллельных вычислений служит качество параллельного выполнения программ - характеристика, объединяющая ускорение, эффективность и избыточность.

Качество определяется следующим образом:

$$Q(n) = \frac{S(n) \times E(n)}{R(n)} = \frac{T^3(1)}{n \times T^2(n) \times O(n)}.$$

Поскольку E(n) — это всегда дробь, а R(n) - число между 1 и n, качество **Q(n)** при любых условиях ограничено сверху величиной ускорения S(n).

Закон Амдала

Джин Амдал (Gene Amdahl) — один из разработчиков всемирно известной Системы IBM 360, в своей работе, опубликованной в 1967 году, предложил формулу, отражающую зависимость ускорения вычислений, достигаемого на многопроцессорной ВС, от числа процессоров и соотношения между последовательной и параллельной частями программы.

$$S = \frac{T_s}{T_p}.$$

$$T_p = f \times T_s + \frac{(1-f) \times T_s}{n}.$$

$$S = \frac{T_s}{T_p} = \frac{n}{1 + (n-1) \times f}.$$

$$\lim_{n \to \infty} S = \frac{1}{f}.$$

Закон Густафсона - Барсиса

Известную долю оптимизма в оценку, даваемую законом Амдала, вносят исследования, проведенные Джоном Густафсоном из NASA Ames Research.

В первом приближении объем работы, которая может быть произведена параллельно, возрастает линейно с ростом числа процессоров в системе.

Объем параллельных вычислений увеличивается с ростом количества процессоров в системе (Е. Барсис).

$$S = \frac{T_s}{T_p} = \frac{f \times T_s + n \times (1-f) \times T_s}{f \times T_s + (1-f) \times T_s} = n + (1-n) \times f.$$

Данное выражение известно как ***закон масштабируемого ускорения*** или закон Густафсона (иногда его называют также *законом Густсафсона-Барсиса*),