Лекция 3

Анализ параллельных вычислений

Модели параллельных вычислений

Сравнительный анализ

-

- Различие мнений
- Наличие субъективных факторов:
  - ✔ Разные постановки задач
  - ✔ Разные критерии сравнения
  - ✔ Разные цели оценивания
  - ✔ …

+

Движение вперед!

Принципиальный момент при разработке параллельных алгоритмов- **анализ эффективности использования параллелизма**:

- Оценка эффективности распараллеливания конкретных выбранных методов выполнения вычислений;
- Оценка максимально возможного ускорения процесса решения рассматриваемой задачи (анализ всех возможных способов выполнения вычислений)

Основные оценки эффективности параллельных вычислений

Показатели эффективности вычислительной системы

- Производительность
- Загруженность

Показатели эффективности параллельного алгоритма

- Ускорение
- Эффективность
- Стоимость

Оценка максимально достижимого параллелизма

- Законы Амдала
- Закон Густафсона

Анализ масштабируемости параллельного алгоритма

Показатели эффективности вычислительной системы

- Производительность
- Загруженность

Система функциональных устройств (ФУ)

Ограничения:

1) За операциями стоят разные функции;
2) Все срабатывания одного ФУ одинаковы по времени;
3) Время срабатывания ФУ – не нулевое;
4) Каждое ФУ – простое;
5) ФУ не имеет памяти;
6) Время передачи данных – нулевое.

Пусть:

n – число операций;
T – общее время работы ФУ;
τ - время выполнения одной операции.

Тогда:

$$n = \frac{T}{\tau}$$

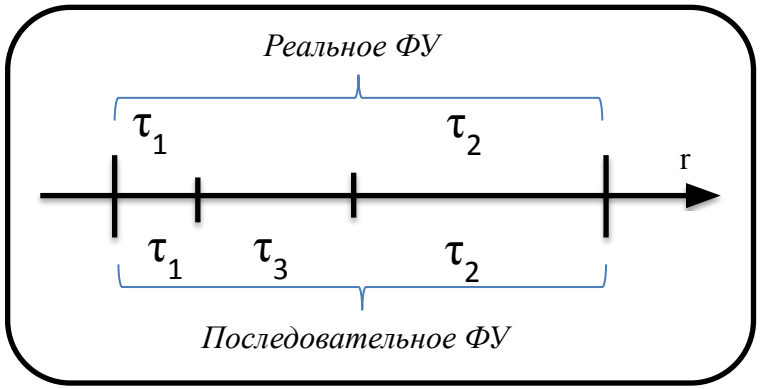
или $n = \lceil \frac{T}{\tau} \rceil - 1$
(при делении с ненулевым остатком).
Но при $T \to \infty$ и $\tau \to 0$:
$\lceil \frac{T}{\tau} \rceil - 1 \to \frac{T}{\tau}$

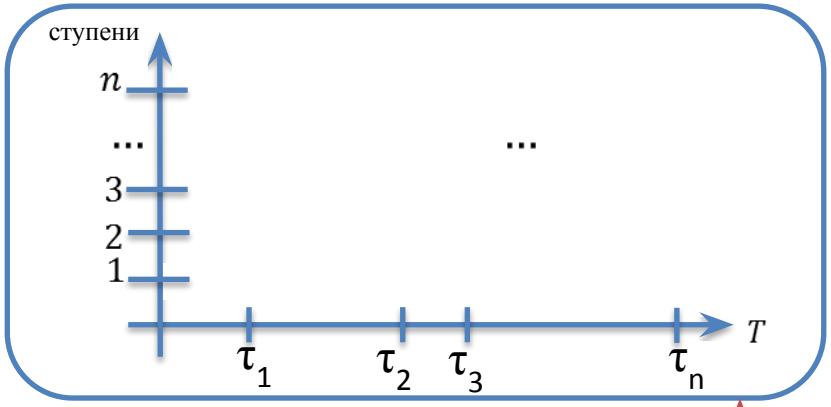
Система функциональных устройств (ФУ)

Стоимость операции: τ (длительность операции)

Стоимость работы: $\sum \tau$ (на последовательном ФУ)

Загруженность ФУ: $p = \frac{\sum \tau}{max \sum \tau} = \frac{\text{стоимость работы на реальном ФУ}}{\text{стоимость работы на последовательном ФУ}}$





Для последовательного ФУ: $p = \frac{\sum \tau}{max \sum \tau} = \frac{\tau_1+\tau_2+\cdots+\tau_n}{\tau_1+\tau_2+\cdots+\tau_n} = \frac{T}{T} = 1$

Для конвейерного ФУ: $p = \frac{\sum \tau \text{ на одной ступени}}{max \sum \tau} = \frac{T}{nT} = \frac{1}{n}$ где $T=\tau_1 + \tau_2 + \cdots + \tau_n$

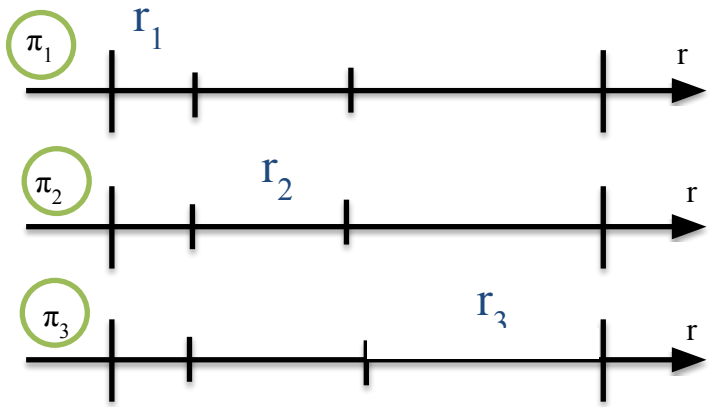
Система функциональных устройств (ФУ)

Пиковая производительность: π - max. количество операций за 1 ед. времени

$$\pi = \sum \pi_i$$

Реальная производительность: r - количество операций за 1 ед. времени

$$r = \sum r_i \qquad (1)$$



Загруженность ФУ: $p_i = \frac{r_i}{\pi_i}$ (2)

Система функциональных устройств (ФУ)

Утверждение 1.

Пусть:

s – количество устройств системы;

π_i – пиковая производительность i-го ФУ, i=1..s;

p_i – загруженность i-го ФУ, i=1..s;

r – реальная производительность.

Тогда:

$$r = \sum_{i=1}^{s} p_i \pi_i.$$

Доказательство.

(1) $p_i = \frac{r_i}{\pi_i} \Rightarrow r_i = p_i \pi_i$

(2) $r = \sum r_i$

$\Rightarrow \quad r = \sum_{i=1}^{s} p_i \pi_i$

Приоритет:

1) Производительность;
2) Загруженность.

Система функциональных устройств (ФУ)

Определение.

Пусть:

s – количество устройств системы;

π_i – пиковая производительность i-го ФУ, i=1..s;

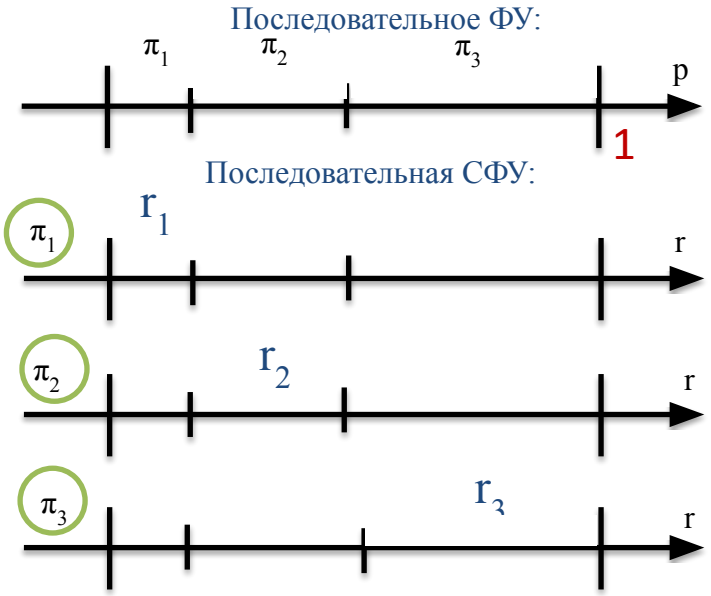
p_i – загруженность i-го ФУ, i=1..s;

r_i – реальная производительность i-го ФУ , i=1..s.

r – реальная производительность системы ФУ.

Тогда:

Загруженность всей системы p:



$$p = \frac{\text{реальная производительность системы}}{\text{реальная производительность последовательного ФУ}} = \frac{\sum \pi_i p_i}{\sum \pi_i \cdot 1}$$

$$p = \frac{\sum \left(p_i \cdot \frac{\pi_i}{\sum \pi_i} \right)}{1} = \sum a_i p_i$$

$$p = \sum_{i=1}^{s} a_i p_i, \quad a_i = \frac{\pi_i}{\sum_{j=1}^{s} \pi_j}.$$

Загруженность системы - взвешенная сумма загруженностей отдельных устройств, т.к.:

$$\sum_{i=1}^{s} a_i = 1, \quad a \geq 0, \quad 1 \leq i \leq s.$$

Система функциональных устройств (ФУ)

Пусть:

s – количество устройств системы.

Тогда при:	**s = 1**	**s = n**
Загруженность ФУ:	$p_i = \frac{\sum \tau}{max \sum \tau} = 1$	$p_i = \frac{\sum \tau}{max \sum \tau} \leq 1$
Загруженность системы:	$p = p_1 = 1$	$0 < p = \sum a_i p_i \leq 1$
Пиковая производительность системы:	$\pi = \pi_1$	$\pi = \pi_1 + \pi_2 + \ldots + \pi_n$
Реальная производительность системы:	$r = \pi$	$r = \sum p_i \pi_i = \frac{\sum \pi_i p_i}{\sum \pi_i \cdot 1} \cdot \sum \pi_i = p\pi$ (p, π)

Показатели эффективности параллельного

- Ускорение = *Ускорение* относительно последовательного выполнения вычислений
- Эффективность = *Эффективность* использования процессоров
- Стоимость = *Стоимость* вычислений

Ускорение (speedup)

Ускорение (speedup) вычислений – это ускорение, получаемое при использовании параллельного алгоритма для *p* процессоров, по сравнению с последовательным вариантом выполнения вычислений:

$$\textbf{Ускорение: } R = \frac{T_{\text{последовательное}}}{\text{T}_{\text{параллельное}}}$$

или

$$R_p(n) = \frac{T_1(n)}{T_p(n)}$$

где:

n – параметр вычислительной сложности решаемой задачи (например, количество входных данных задачи)

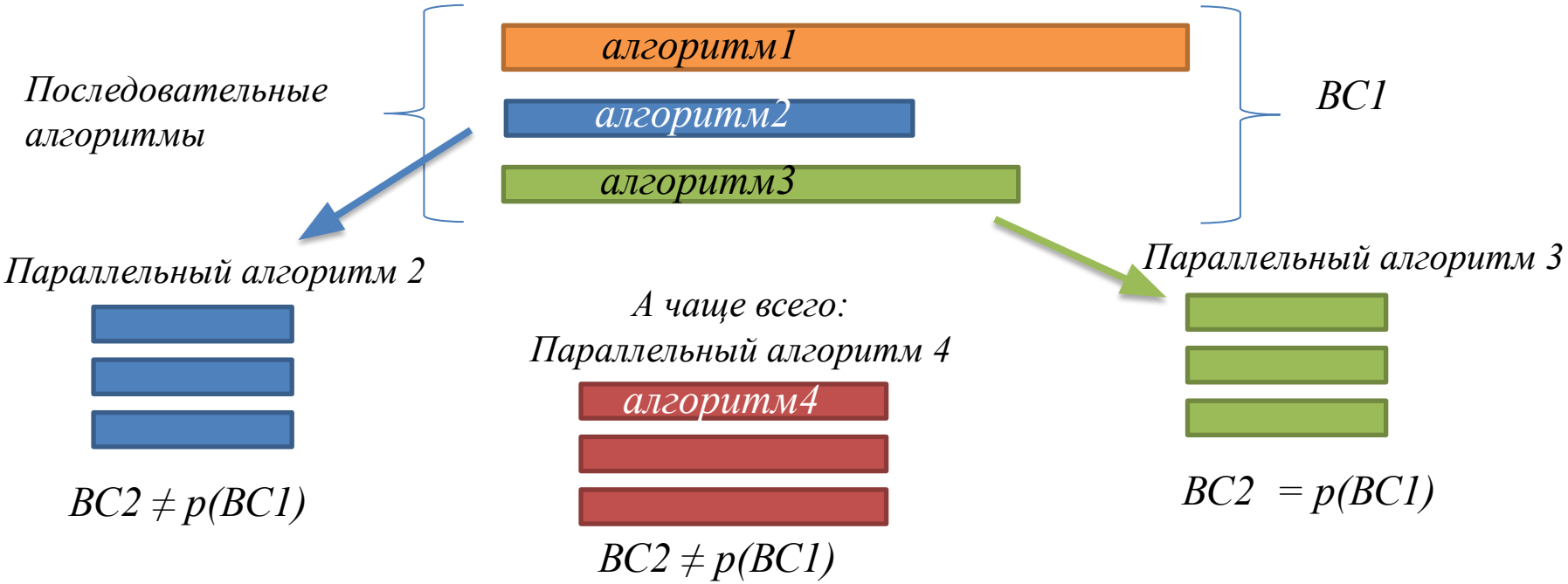
p – число процессоров

Виды ускорений

В зависимости от эталона

Величину ускорения называют ***абсолютной***, если в качестве T_1 берется время выполнения наилучшего последовательного алгоритма.

Величину ускорения называют ***относительной***, если в качестве T_1 берется время выполнения параллельного алгоритма на одном процессоре.



Виды ускорений

В зависимости от величины R

Линейное *(linear)* или *идеальное (ideal)* ускорение имеет место при

$Sp=p$

Сверхлинейное *(superlinear)* ускорение имеет место при

$Sp>p$.

Причины недостижимости этих ускорений

- Неравноправность выполнения последовательной и параллельной программ (например, недостаток оперативной памяти).
- Нелинейный характер зависимости сложности решения задачи от объема обрабатываемых данных.
- Различие вычислительных схем последовательного и параллельного методов.

Эффективность *(efficiency)*

Эффективность (efficiency) – средняя доля времени выполнения параллельного алгоритма, в течение которого процессоры реально используются для решения задачи.

Эффективность: $E = \frac{R}{s}$, где s – число ФУ

или

$$E_p(n) = \frac{T_1(n)}{p \cdot T_p(n)} = \frac{R_p(n)}{p}$$

где:

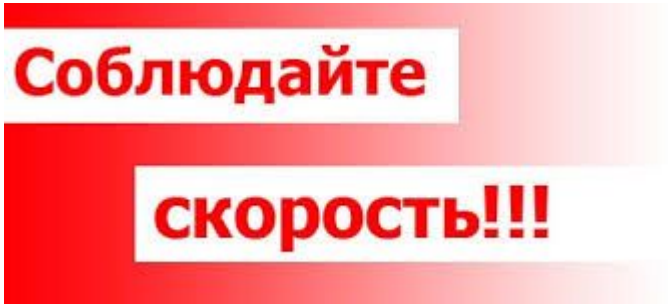
n – параметр вычислительной сложности решаемой задачи (например, количество входных данных задачи)

p – число процессоров

Ускорение vs эффективность

Ускорение и эффективность – 2 стороны одной медали: попытки повышения качества параллельных вычислений по одному из показателей может привести к ухудшению качества по другому показателю.

Тише едешь – дальше будешь?



Стоимость вычислений

Стоимость *(cost)* параллельных вычислений

$$C_p = p \cdot T_p$$

Стоимостно-оптимальный *(cost-optimal)* параллельный алгоритм – алгоритм, стоимость которого является пропорциональной времени выполнения наилучшего последовательного алгоритма.

Можно ли достичь max параллелизма?

- Получение идеальных величин $Rp=p$ для ускорения и $Ep=1$ для эффективности может быть обеспечено не для всех вычислительно трудоемких задач.
- Достижению максимального ускорения может препятствовать существование в выполняемых вычислениях последовательных расчетов, которые не могут быть распараллелены.

Оценки максимально достижимого параллелизма

Ускорение и эффективность на СФУ

Пусть: $\pi_1 \le \pi_2 \le \ldots \le \pi_s$

r – реальная производительность системы ФУ.

Пусть для сравнения есть ФУ – гипотетическое, для которого $\pi = \pi_s$

Тогда:

Ускорение: $R = \dfrac{r}{\pi_s}$ $\Rightarrow$ $R = \dfrac{\sum_{i=1}^{s} p_i \pi_i}{\max\limits_{1 \le i \le s} \pi_i}$.

В системе из одинаковых ФУ ($\pi_1 = \pi_2 = \ldots = \pi_s$, $p_i = 1$): $\mathrm{R} = \dfrac{\sum(1 \cdot \pi_s)}{\pi_s} = s$

В остальных случаях: $\mathrm{R} \le s$

Система функциональных устройств (ФУ)

Утверждение 2.

Пусть:

$$\pi_1 = \pi_2 = \ldots = \pi_s$$

Тогда:

1) $p = \frac{\sum p_i}{s}$ — загруженность системы равна среднему арифметическому загруженностей всех устройств;

2) $r = \sum r_i$ — реальная производительность системы равна сумме реальных производительностей всех устройств;

3) $\pi = s\pi$ — пиковая производительность системы в s раз больше пиковой производительности одного устройства;

4) $R = \sum p_i$ — ускорение системы равно сумме загруженностей всех устройств;

5) $R = \frac{T_i}{T}$ — если система состоит только из простых устройств, то ее ускорение равно отношению времени реализации алгоритма на одном универсальном простом устройстве с той же пиковой производительностью к времени реализации алгоритма на системе.

Граф системы

Как повысить производительность системы?

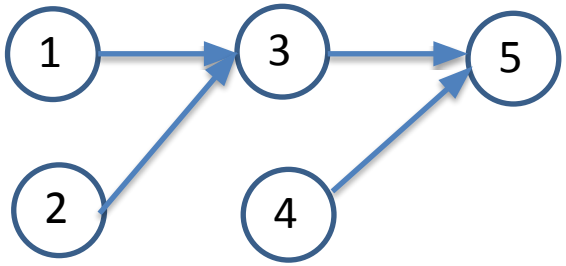
Повысить загруженность системы!

Как повысить загруженность системы?

Повысить загруженность ФУ!

Всегда ли можно повысить загруженность ФУ?

Вершины – ФУ;
Ребра – связь по данным.

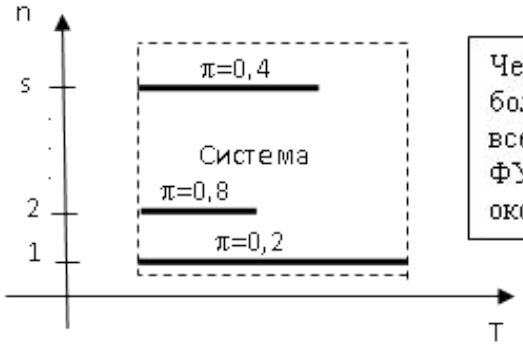


Утверждение 3.
Пусть:

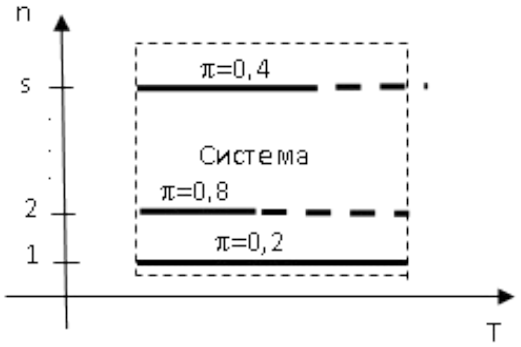
$\pi_1, \pi_2, \ldots, \pi_s$
граф системы связный

Тогда:

$$r_{\max} = s \min_{1 \le i \le s} \pi_i$$



Чем ниже π, тем больше Ti, все остальные ФУi будут ждать окончания ФУ1



Система функциональных устройств (ФУ)

Следствие из утверждения 3.

Пусть:

$\pi_1, \pi_2, \ldots, \pi_s$

граф системы связный

Тогда:

1) $p \le \min p_i$ Загруженность любого устройства не превосходит загруженности самого НЕ производительного устройства.

2) Если $p_i = 1$, то это самое НЕ производительное устройство.

3) Загруженность системы *p* не превосходит:

$$p_{\max} = \frac{s \min\limits_{1 \le i \le s} \pi_i}{\sum\limits_{i=1}^{s} \pi_i}$$

4) Ускорение системы *R* не превосходит:

$$R_{\max} = \frac{s \min\limits_{1 \le i \le s} \pi_i}{\max\limits_{1 \le i \le s} \pi_i}$$

1-й закон Амдала

Джин Амдал
(р. 1922)

Производительность системы определяется

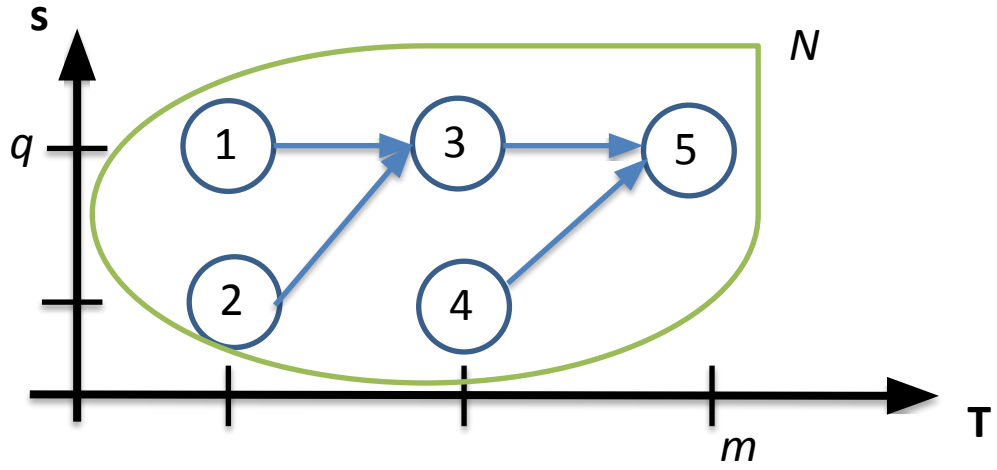
самым непроизводительным ее устройством.

Следствие.

Асимптотически производительность системы максимальна, если

$$\pi_1 = \pi_2 = \ldots = \pi_s$$

Информационный граф



Ускорение системы

$$R_{max} = \frac{N}{m}$$

Утверждение 4.

Пусть:

$\pi_1 = \pi_2 = \ldots = \pi_s = \pi$

T – время реализации алгоритма;

N_i – число операций на i-м ФУ;

Тогда:

Т.к. $p_i = \frac{N_i}{\pi T}$ - загруженность ФУ и $R = \frac{\sum_{i=1}^{s} p_i \pi_i}{\max_{1 \le i \le s} \pi_i}$.

то:

$$R = \frac{\sum_{i=1}^{s} \left(\frac{N_i}{\pi T}\right) \pi}{\pi} = \frac{N}{\pi T}$$

Для одного яруса $t_1 = \frac{1}{\pi}$

Для m ярусов $T = \frac{m}{\pi}$

Следовательно:

$$R_{max} = \frac{N}{m}$$

2-й закон Амдала

Пусть: Доля последовательных вычислений:

$$\beta = \frac{n}{N}, \quad n - \text{число последов. операций}$$

Тогда:

$$R = \frac{s}{\beta s + (1 - \beta)}$$

Доказательство:

Если доля последовательных вычислений: β, то:
βN – число последовательных операций;
$(1 - \beta)N$ – число параллельных операций;
$\frac{(1-\beta)N}{s}$ – число параллельных операций на i-м ФУ;

Пусть все последовательные операции выполняются на 1-м ФУ, тогда:

$$T_1 = \frac{\beta N + (1 - \beta)N/s}{\pi}$$

Т.к. $p_i = \frac{N_i}{\pi \mathrm{T}}$, то: $p_1 = \frac{\beta N + (1-\beta)N/s}{\pi \frac{\beta N + (1-\beta)N/s}{\pi}} = 1$

2-й закон Амдала

Для остальных ФУ:

$$T_i = \frac{(1-\beta)N/s}{\pi} \quad , 2 \le i \le s$$

$$\rho_i = \frac{(1-\beta)N/s}{\beta N + (\beta - 1)N/s}$$

Из формулы: $R = \sum_{i=1}^{s} p_i$ получаем:

$$R = 1 + \sum_{i=2}^{s} \frac{(1-\beta)N/s}{\beta N + (1-\beta)N/s} = \frac{s}{\beta s + (1-\beta)}$$

ч.т.д.

Более привычный вид закона: $S_p \le \frac{1}{f + (1-f)/p} \le S^* = \frac{1}{f}$

2-й закон Амдала

Пример: Ускорение последовательной программы

Пусть имеется последовательная программа, состоящая из двух независимых частей: **A** (75%) и **B** (25%).
Увеличение быстродействия части **B** *в 5 раз* даст ускорение

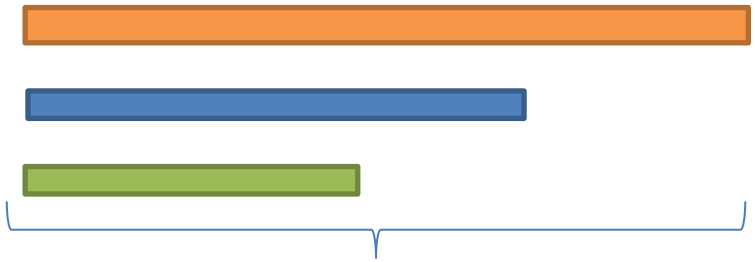
$$R = \frac{1}{f + \frac{(1-f)}{p}} = \frac{p}{fp + (1-f)} = \frac{5}{0{,}75 \cdot 5 + 0{,}25} = \frac{5}{4} = 1{,}25$$

Увеличение быстродействия части **A** *в 2 раза* даст ускорение

$$R = \frac{1}{f + \frac{(1-f)}{p}} = \frac{p}{fp + (1-f)} = \frac{2}{0{,}25 \cdot 2 + 0{,}75} = \frac{2}{1{,}25} = 1{,}6$$

Программа **AB**

Исходная программа
Часть **B** быстрее в 5 раз
Часть **A** быстрее в 2 раза



3-й закон Амдала

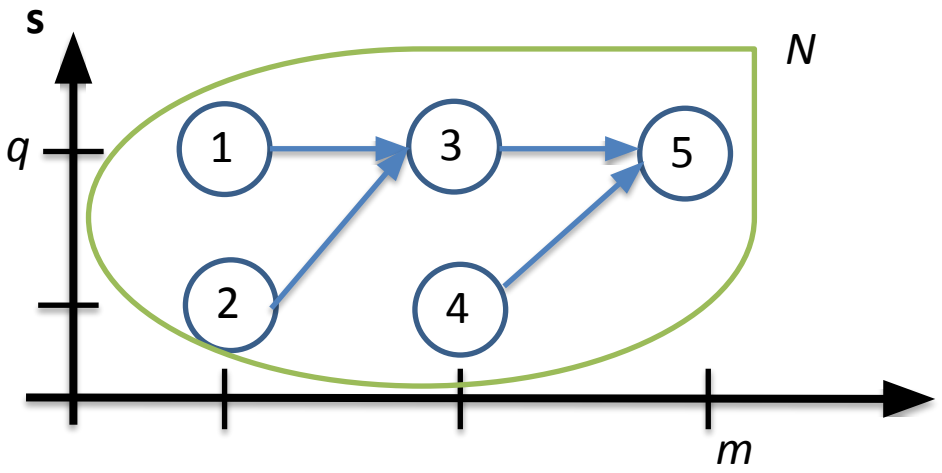
Пусть:

Доля последовательных вычислений: $\beta = \frac{n}{N}$

Тогда: $R \leq \frac{1}{\beta}$

Примечание:

Если n – число последовательный операций, то: $m \geq n$

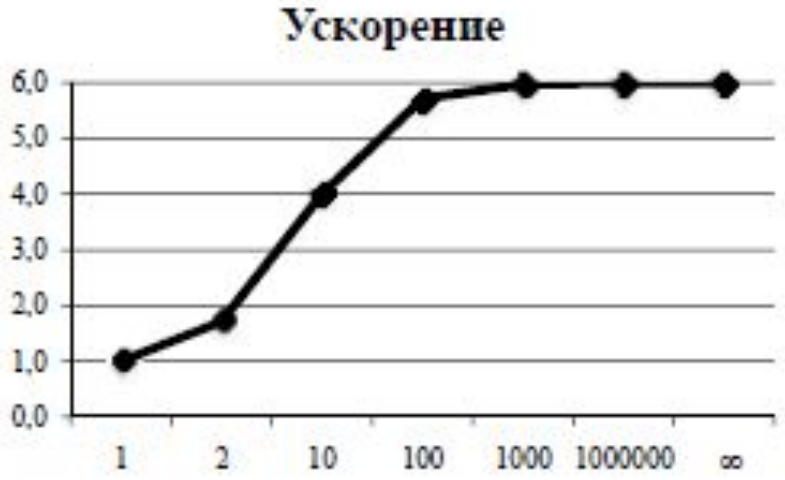


2-й закон Амдала

$$\lim_{p\to\infty} S_p = \frac{1}{f}$$

- Покраска забора (300 досок)
  - *Подготовка* – 30 мин.
    НЕ распараллеливается
  - *Покраска* (одной доски) – 1 мин.
    РАСПАРАЛЛЕЛИВАЕТСЯ
  - *Уборка* – 30 мин.
    НЕ распараллеливается

Количество маляров	Время покраски	
1	30 + 300/1	+ 30 = 360
2	30 + 300/2	+ 30 = 210
10	30 + 300/10	+ 30 = 90
100	30 + 300/100	+ 30 = 63
1000	30 + 300/1000	+ 30 ≅ 60
1000000	30 + 300/1000000	+ 30 ≅ 60



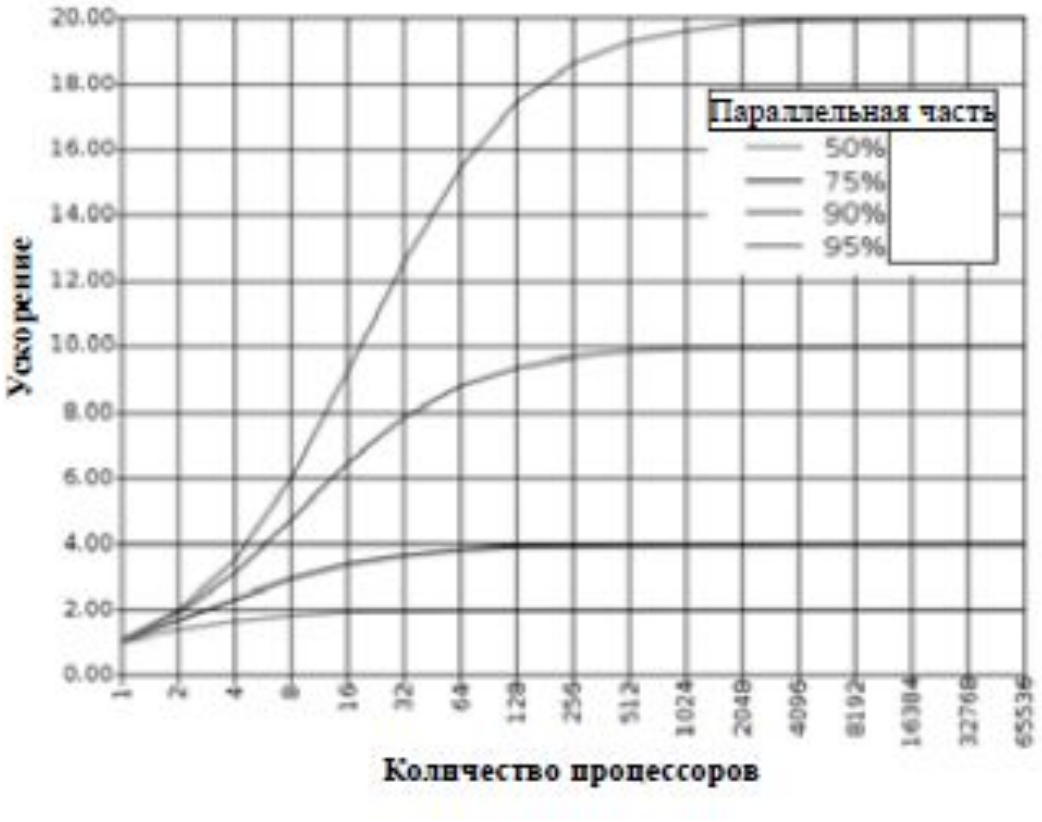
3-й закон Амдала

$$\lim_{p\to\infty} S_p = \frac{1}{f} \qquad \lim_{p\to\infty} E_p = \frac{R_p(n)}{p} = \frac{1/f}{p} = \frac{1}{f\cdot p}$$

- Покраска забора (300 досок)
  - *Подготовка* – 30 мин.
    НЕ распараллеливается
  - *Покраска* (одной доски) – 1 мин.
    РАСПАРАЛЛЕЛИВАЕТСЯ
  - *Уборка* – 30 мин.
    НЕ распараллеливается

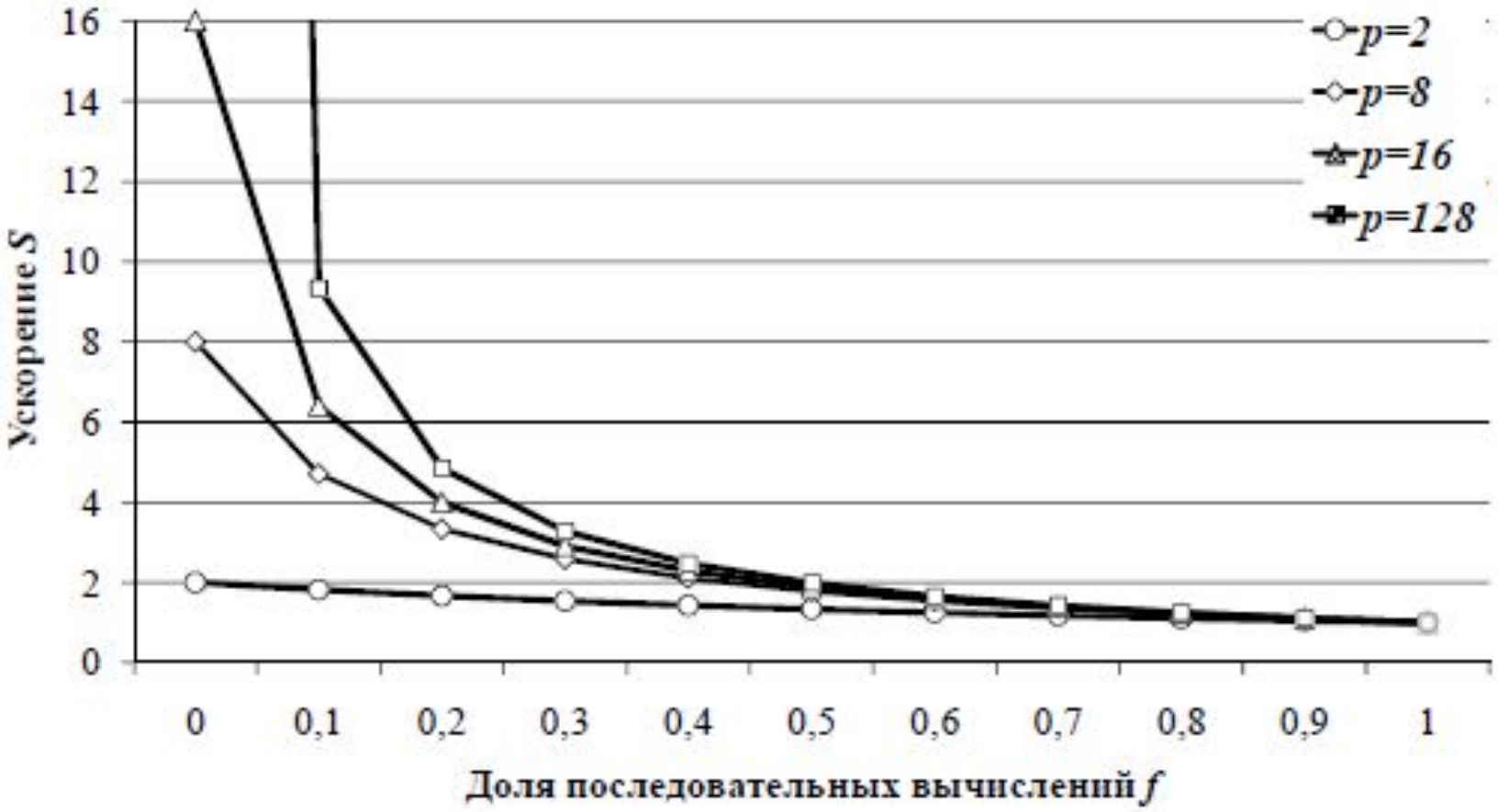
Количество маляров	Время покраски	
1	30 + 300/1	+ 30 = 360
2	30 + 300/2	+ 30 = 210
10	30 + 300/10	+ 30 = 90
100	30 + 300/100	+ 30 = 63
1000	30 + 300/1000	+ 30 ≅ 60
1000000	30 + 300/1000000	+ 30 ≅ 60

3-й закон Амдала



Ускорение параллельной программы зависит не от количества процессоров, а от величины последовательной части программы.

3-й закон Амдала



Закон Густафсона-Барсиса

Вопроса об ускорении на n процессорах

Вопрос о замедлении вычислений при переходе на один процессор

закон Амдала

закон Густафсона

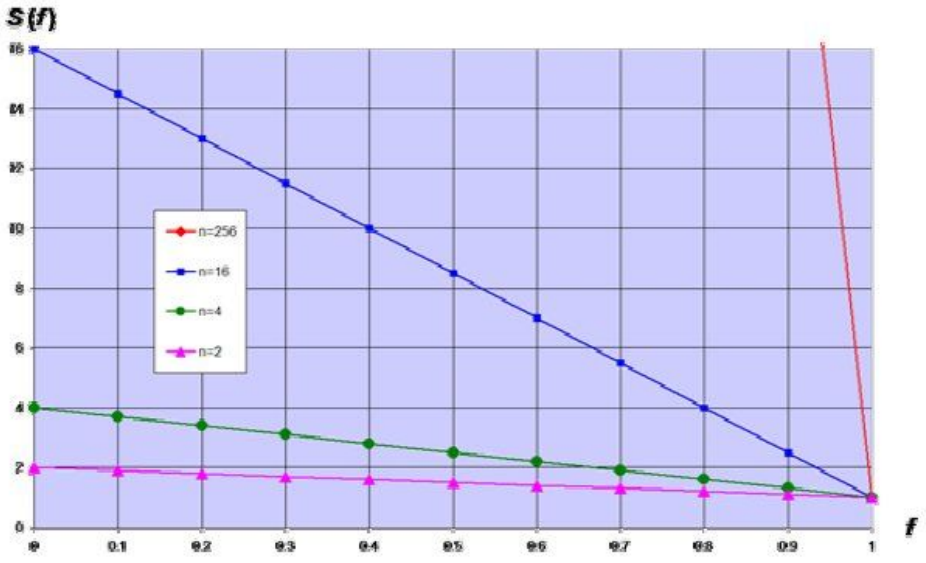
Закон Густафсона-Барсиса

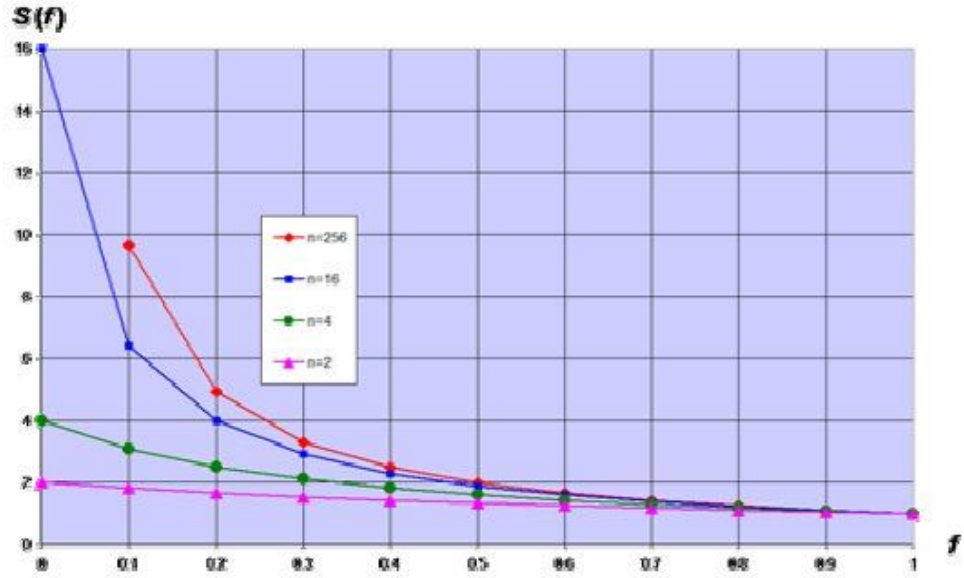
Дж. Густафсон
(р. 1955)

Аналогично за ***f*** примем долю последовательной части программы. Тогда получим
закон масштабируемого ускорения:

$$S(n) = \frac{T(1)}{T(n)} = \frac{f \times T(n) + n \times (1-f) \times T(n)}{f \times T(n) + (1-f) \times T(n)} = n + (1-n) \times f$$

Закон Густафсона-Барсиса против закона Амдала





закон Густафсона

$$S(n) = n+(1-n)\times f$$

закон

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

Т.е. законы Амдала и Густафсона в идентичных условиях дают различные значения ускорения!

Где же ошибка?

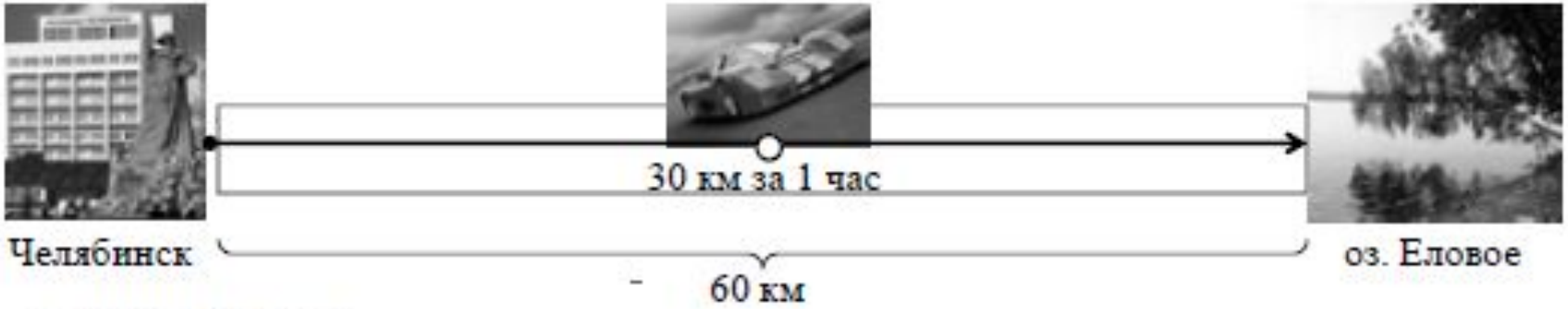
Каковы области применения этих законов?

Ответ!

Увеличение объема решаемой задачи приводит к увеличению доли параллельной части, так как последовательная часть (ввод/вывод, менеджмент потоков, точки синхронизации и т.п.) не изменяется.

Таким образом, уменьшение доли f приводит к перспективным значениям ускорения.

Закон Густафсона-Барсиса против закона Амдала



- Закон Амдала
  - Независимо от того, как быстро будет ехать машина вторую половину пути, невозможно достигнуть средней скорости 90 км/ч до приезда в пункт назначения
    - Например, до приезда даже очень быстрая езда позволит достигнуть лишь средней скорости 60 км/ч.
- Закон Густавсона
  - Независимо от того, как долго или как медленно двигалась машина первую половину пути, при наличии достаточного количества времени и протяженности дороги средняя скорость машины в конечном итоге всегда достигнет значения 90 км/ч
    - Например, при скорости 150 км/ч на второй половине пути по приезде в пункт назначения средняя скорость равна 90 км/ч .

Анализ масштабируемости параллельного алгоритма

Масштабируемость алгоритмов

- Параллельный алгоритм называют *масштабируемым (scalable)*, если при росте числа процессоров он обеспечивает увеличение ускорения при сохранении постоянного уровня эффективности использования процессоров.

- При анализе масштабируемости необходимо учитывать *накладные расходы (total overhead)* на организацию взаимодействия процессоров, синхронизацию параллельных вычислений и др.

Анализ масштабируемости

- Накладные расходы: $T_0 = pT_p - T_1$
- Время решения задачи: $T_p = \frac{T_1 + T_0}{p}$
- Ускорение: $S_p = \frac{T_1}{T_p} = \frac{pT_1}{T_1 + T_0}$
- Эффективность: $E_p = \frac{S_p}{p} = \frac{T_1}{T_1 + T_0} = \frac{1}{1 + \frac{T_0}{T_1}}$

Анализ масштабируемости

- Если сложность решаемой задачи является фиксированной (*T1=const*), то при росте числа процессоров эффективность, как правило, будет убывать за счет роста накладных расходов *T0*.
- При фиксации числа процессоров эффективность использования процессоров можно улучшить путем повышения сложности решаемой задачи *T1*.
- При увеличении числа процессоров в большинстве случаев можно обеспечить определенный уровень эффективности при помощи соответствующего повышения сложности решаемых задач.

Оценка максимально достижимого параллелизма…

- Оценка качества параллельных вычислений предполагает знание *наилучших* (*максимально достижимых*) значений показателей ускорения и эффективности

- Получение идеальных величин $S_p=p$ для ускорения и $E_p=1$ для эффективности может быть обеспечено не для всех вычислительно трудоемких задач