Анализ эффективности параллельных алгоритмов и программ

Востокин Сергей Владимирович

План

- Понятия ускорения и эффективности
- Модель операции-операнды, расчёт ускорения и эффективности по графу вычислительного процесса
- Пример 1: Алгоритм каскадного суммирования
- Пример 2: Модифицированный алгоритм каскадного суммирования
- Теоретический предел ускорения. Закон Амдала
- Понятие масштабируемости. Закон Густафсона - Барсиса

Понятия ускорения и эффективности

Определение ускорения

- **Ускорение** (*Speedup*) определяет, во сколько раз уменьшается время счета алгоритма на параллельном компьютере по сравнению со временем счета на последовательном компьютере

$$S = \frac{T_1}{T_p}$$

Какие значения принимает ускорение

- **Линейное** (идеальное) ускорение получается, когда на N процессорах программа работает в N раз быстрее, чем на 1 процессоре
- Обычно ускорение менее чем линейное (***sublinear***)
- Иногда наблюдается **сверхлинейное** (***superlinear***) ускорение программ

Определение эффективности

- **Эффективность** (*Efficiency*)— характеристика того, насколько хорошо параллельная программа использует дополнительные процессоры

$$E = \frac{T_1}{p \cdot T_p} = \frac{S}{p}$$

Какие значения принимает эффективность (1/2)

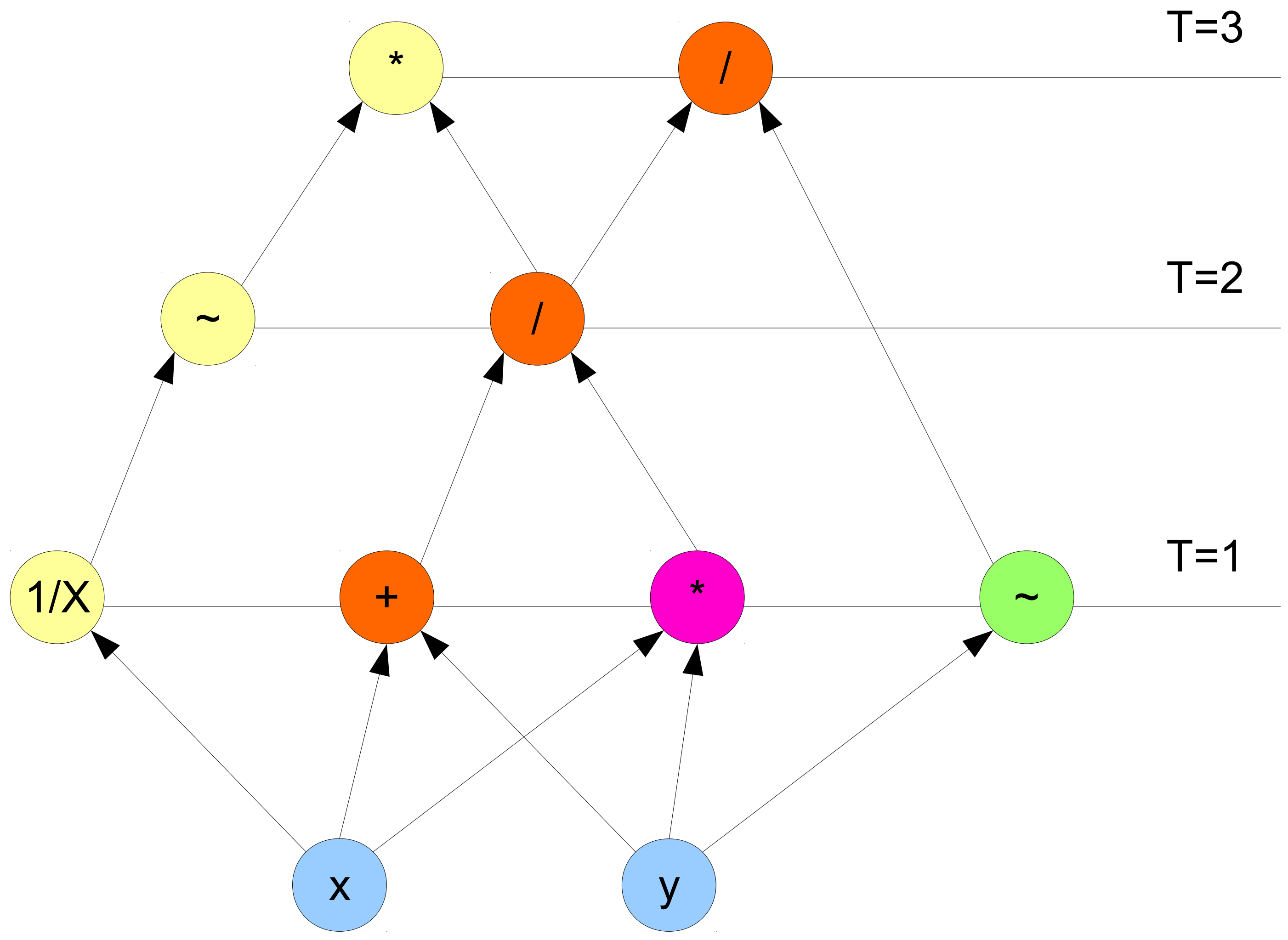
- Если программа имеет линейное ускорение, то её эффективность равна 1
- При сверхлинейном ускорении эффективность >1, при обычном – она <1
- Часто с увеличением числа процессоров эффективность снижается
- Параллельная программа часто эффективна при решении задач больших размеров

Какие значения принимает эффективность (2/2)

- У ***масштабируемых*** программ эффективность постоянна в широком диапазоне количеств процессоров и размеров задач
- Часто для достижения лучшего результата перед распараллеливанием необходимо переработать исходный последовательный алгоритм
- ***Абсолютная эффективность*** и ***абсолютное ускорение*** оцениваются по времени работы лучшего последовательного алгоритма

Модель операции-операнды, расчёт ускорения и эффективности по графу вычислительного процесса

Граф модели операции-операнды



Модель (1/2)

- Информационно-логическая структура программы представляется в виде графа

$$G = (V, R)$$

$$V \subset \mathbb{N} \quad R \subset V \times V$$

$$\tilde{V} = V \setminus \{ v \in V : \nexists x \in V : (x, v) \in R \}$$

Модель (2/2)

- Расписание выполнения программы

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

$$\forall i, j \in V : t_i = t_j \rightarrow P_i \neq P_j$$

$$\forall (i, j) \in R : t_i < t_j$$

Выражение ускорения и эффективности через модель

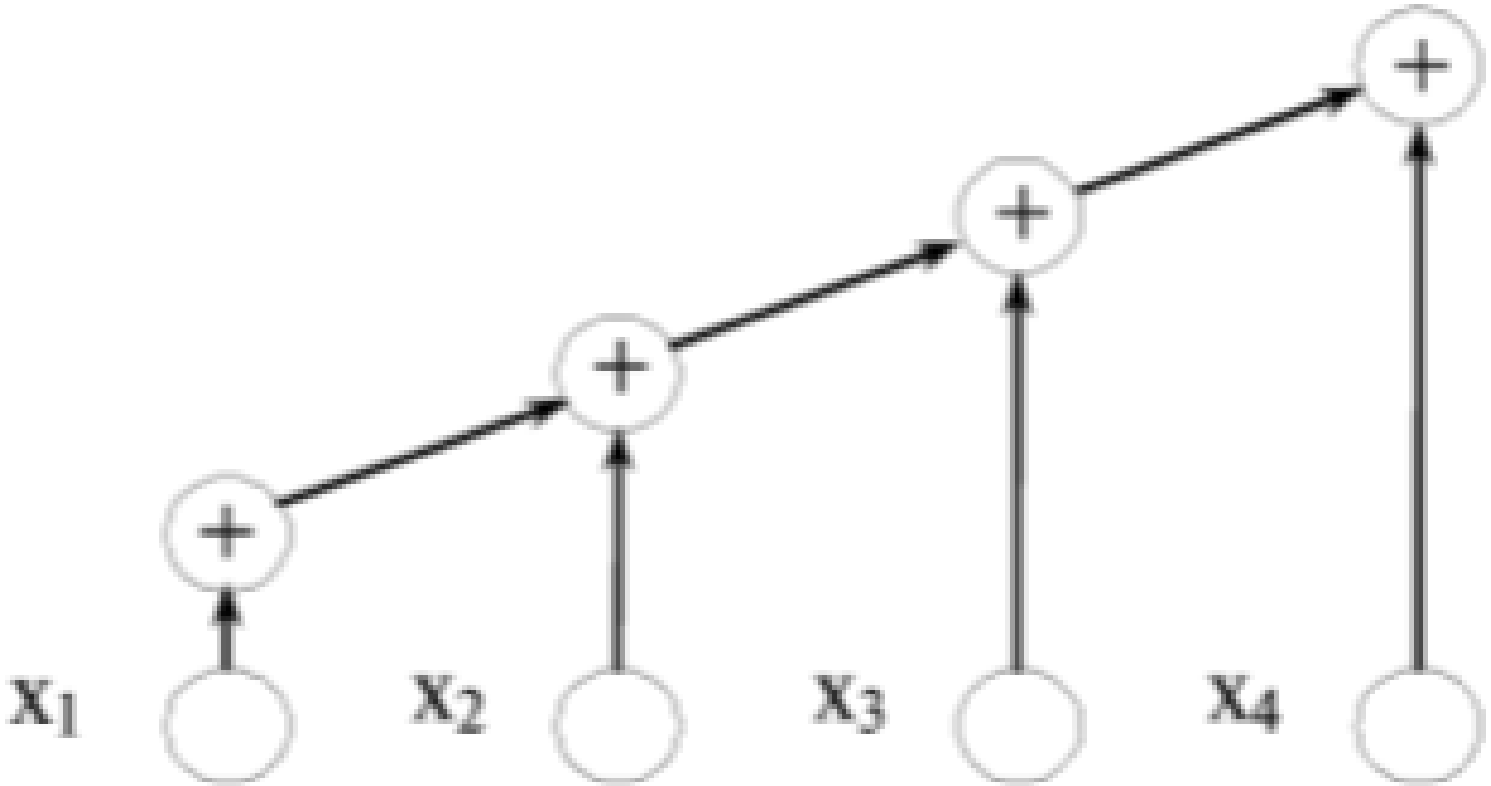
- Обозначим время выполнения операции как ***t***, время выполнения расписания как ***T(.)***, а ***d(.)*** – диаметр графа, тогда

$$T_1 = |\widetilde{V}| \cdot t \qquad \min_{H_p} T_p = d(G) \cdot t$$

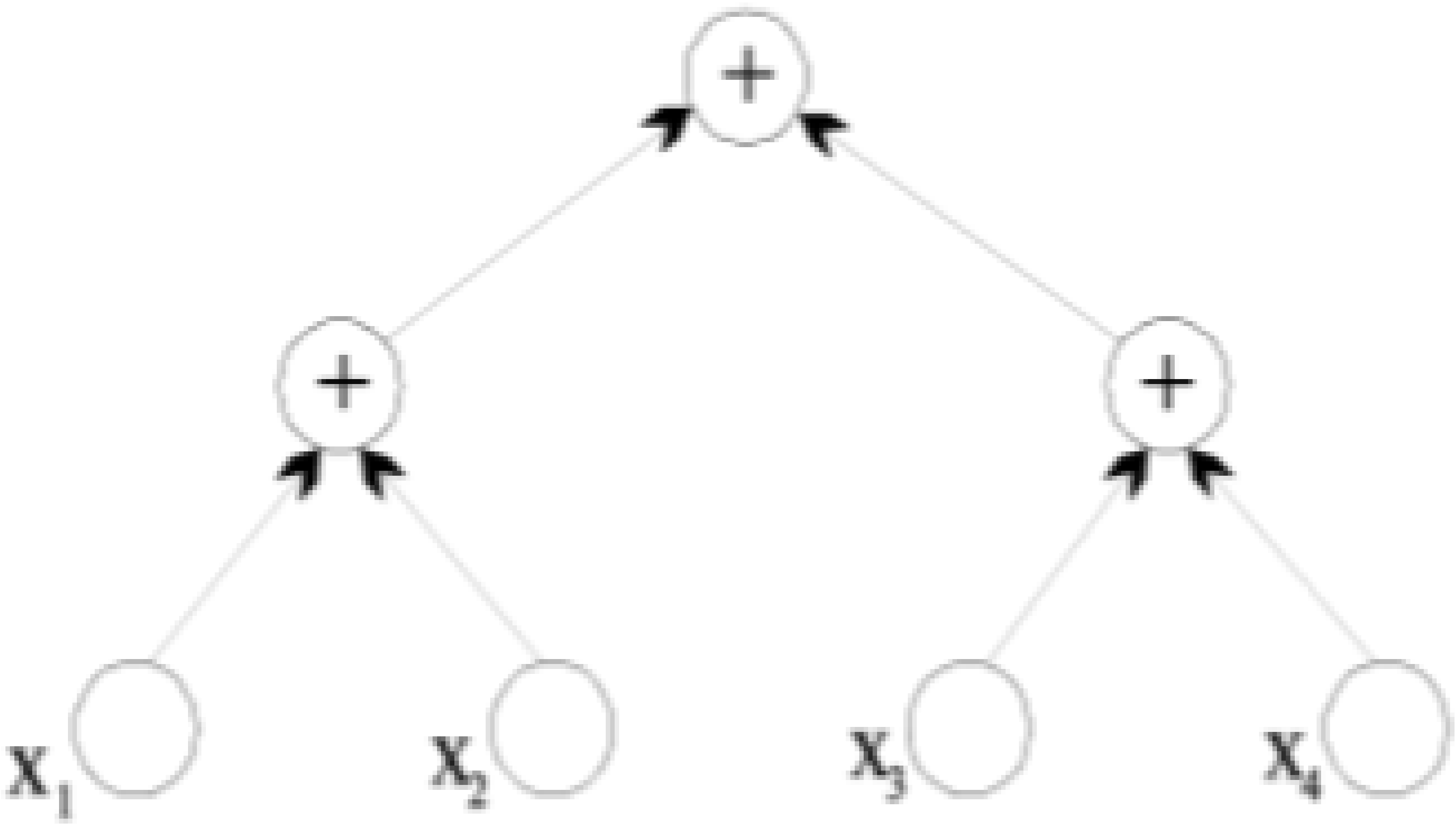
$$S_p = \frac{|\widetilde{V}|}{T(H_p)} \qquad E_p = \frac{|\widetilde{V}|}{T(H_p) \cdot p}$$

Пример 1: Алгоритм каскадного суммирования

Исходный алгоритм в виде модели операции-операнды



Преобразованный алгоритм в виде модели операции-операнды



Параметры ускорения и эффективности алгоритма

$$T_1 = N - 1 \qquad T_p = \log_2(N)$$

$$S = (N-1)/\log_2(N)$$

$$E = (N-1)/(N/2 \cdot \log_2(N))$$

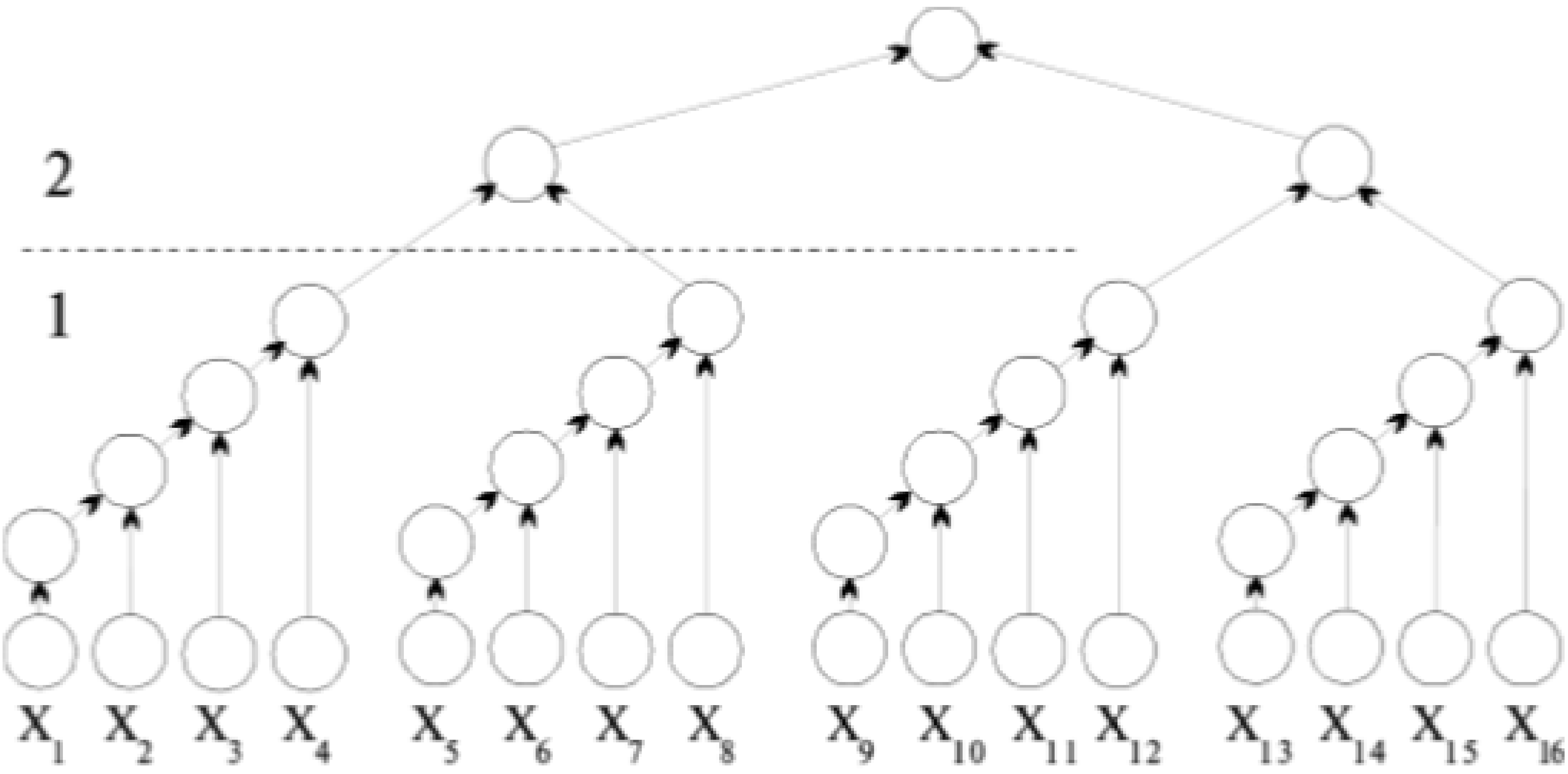
Предельные значения ускорения и эффективности алгоритма

- Хотя ускорение увеличивается при увеличении размерности задачи, эффективность стремится к 0

$$\lim_{p \to \infty} S = \infty \qquad \lim_{p \to \infty} E = 0$$

Пример 2: Модифицированный алгоритм каскадного суммирования

Модифицированная каскадная схема суммирования



Параметры ускорения и эффективности модифицированного алгоритма

- Операции сгруппированы по $log_2 N$ элемента в $N / log_2 N$ групп

$$T_p = \log_2(N) + \log_2(N/\log_2(N)) \approx 2 \cdot \log_2(N)$$

$$S = (N-1)/(2 \cdot \log_2(N))$$

$$E = (N-1)/(N/\log_2(N) \cdot 2 \cdot \log_2(N)) = (N-1)/2 \cdot N$$

Предельные значения ускорения и эффективности модифицированного алгоритма

$$\lim_{p \to \infty} S = \infty \qquad \lim_{p \to \infty} E = \frac{1}{2}$$

Теоретический предел ускорения. Закон Амдала

Постановка задачи

- Предположим, что имеется некоторая задача ***фиксированной размерности***
- Пусть для её решения доступно произвольное количество процессоров параллельного компьютера
- Вопросы:
  - Насколько можно ускорить решение задачи по сравнению с однопроцессорным компьютером
  - Если доступно заданное количество процессоров, то на сколько ускорится решение задачи

Исходные данные (1/3)

- В любом алгоритме есть последовательная часть: ввод, инициализация параллельного выполнения, вывод, особенности алгоритма
- Обозначим через ***A*** время выполнения последовательной части алгоритма
- Тогда время выполнения всего алгоритма на одном процессоре составит

$$T_1 = A + B$$

Исходные данные (2/3)

- При выполнении той же программы на ***p*** процессорах параллельного компьютера время выполнения оставшейся части кода ***B*** сократится в ***p*** раз и составит

$$T_p = A + \frac{B}{p}$$

Исходные данные (3/3)

- Обозначим часть времени, не поддающуюся распараллеливанию, в общем времени счёта исходной последовательной программы через ***f***

$$f = \frac{A}{A+B}$$

Вывод формулы (1/2)

- По определению ускорение равно

$$S = \frac{T_1}{T_p} = \frac{A+B}{A+\frac{B}{p}} = \frac{1}{\frac{A}{A+B}+\frac{B}{p(A+B)}} = \frac{1}{f+\frac{1-f}{p}}$$

Вывод формулы (2/2)

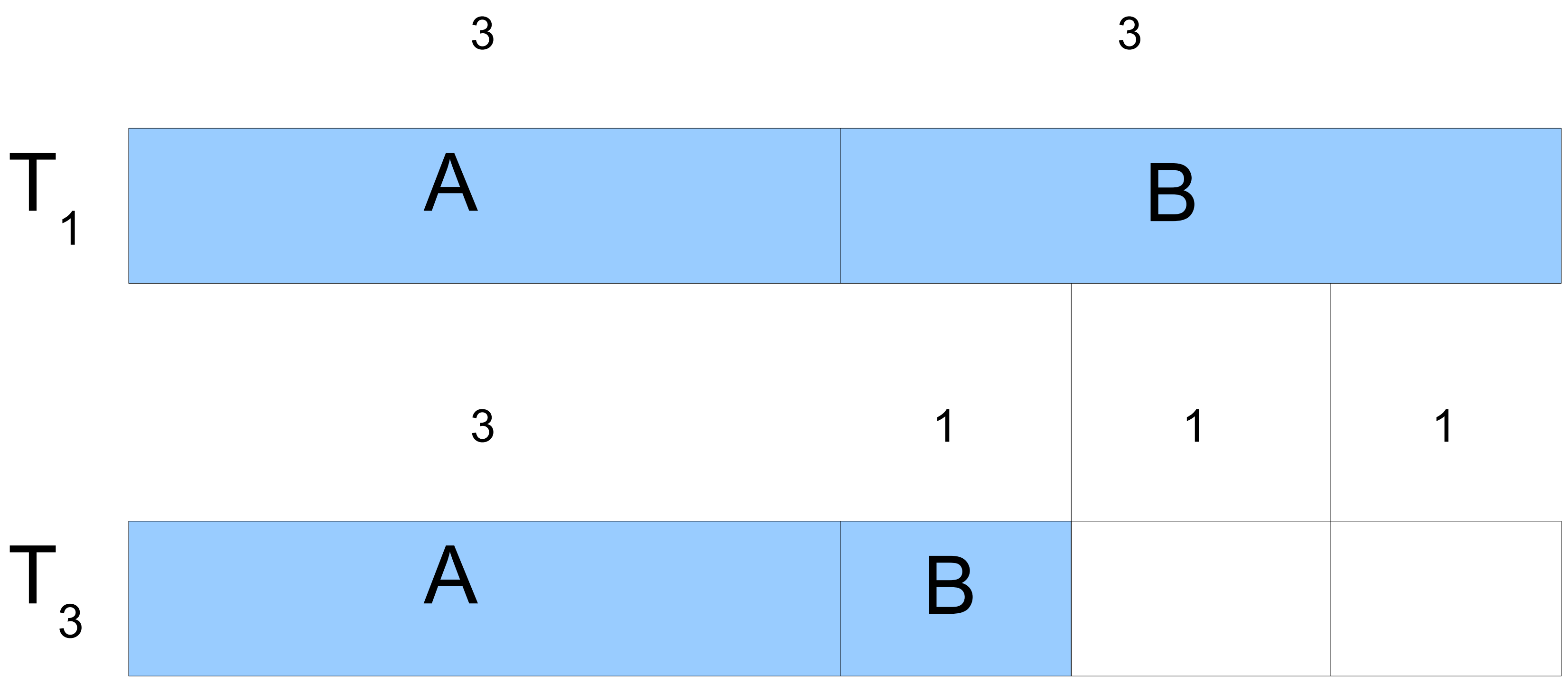
- В итоге, для заданного ***p*** и в предельном случае ускорение составит

$$S(p) = \frac{1}{f + \frac{1-f}{p}}, N = const, T = var$$

$$S(p) \underset{p \to \infty}{\to} 1/f$$

Пример 1 (1/2)

- Пусть имеется 3 процессора и времена выполнения частей программы соотносятся как показано на рисунке



Пример 1 (2/2)

- Определим ускорение по закону Амдала

$$f=\frac{1}{2}$$

$$S(3)=\frac{1}{\frac{1}{2}+\frac{1}{2}\cdot\frac{1}{3}}=\frac{1}{\frac{1}{2}+\frac{1}{6}}=\frac{1}{\frac{3}{6}+\frac{1}{6}}=\frac{6}{4}=\frac{3}{2}=1,5$$

Пример 2

- Для алгоритма каскадного суммирования найти максимальное ускорение по закону Амдала

$$A = \log_2(N), \quad A + B = N - 1$$

$$f = \frac{\log_2(N)}{N-1} \rightarrow S_{max} = \frac{1}{f} = \frac{N-1}{\log_2(N)}$$

Библиографическая справка

- Amdahl, Gene M. "Validity of the single processor approach to achieving large scale computing capabilities." Proceedings of the April 18-20, 1967, spring joint computer conference. ACM, 1967.

Понятие масштабируемости.
Закон Густафсона - Барсиса

Постановка задачи

- Пусть нам требуется решить задачу ***произвольной размерности*** за ***фиксированное время***
- Имеется произвольное количество процессоров ***p*** для решения задачи
- Вопрос

  - как определить ускорение в этом случае

Исходные данные (1/3)

- Каждый из ***p*** процессоров во время решения задачи тратит некоторую часть своего времени ***A*** на последовательные операции ввод, вывод, синхронизацию с другими процессорами

$$T_p = A + B$$

Исходные данные (2/3)

- Длительность последовательной части ***A*** не зависит от числа работающих процессоров
- Тогда время счёта для одного процессора, когда все исходные параллельные части будут исполнятся последовательно будет равно

$$T_1 = A + p \cdot B$$

Исходные данные (3/3)

- Обозначим часть времени, занимаемую последовательными операциями, во времени счёта ***каждого процессора*** через

$$a = \frac{A}{A+B}$$

Вывод формулы (1/2)

- По определению ускорения

$$S=\frac{A+p\cdot B}{A+B}=\frac{A}{A+B}+p\frac{B}{A+B}$$

$$S=a+p\left(1-a\right)=a+p-p\cdot a=p-a\left(p-1\right)$$

Вывод формулы (2/2)

- В итоге, для заданного ***p*** ускорение составит

$$S(p) = p - a(p-1), T = const, N = var, N \propto p$$

Графическая иллюстрация сравнения законов Амдала и Густафсона (1/3)

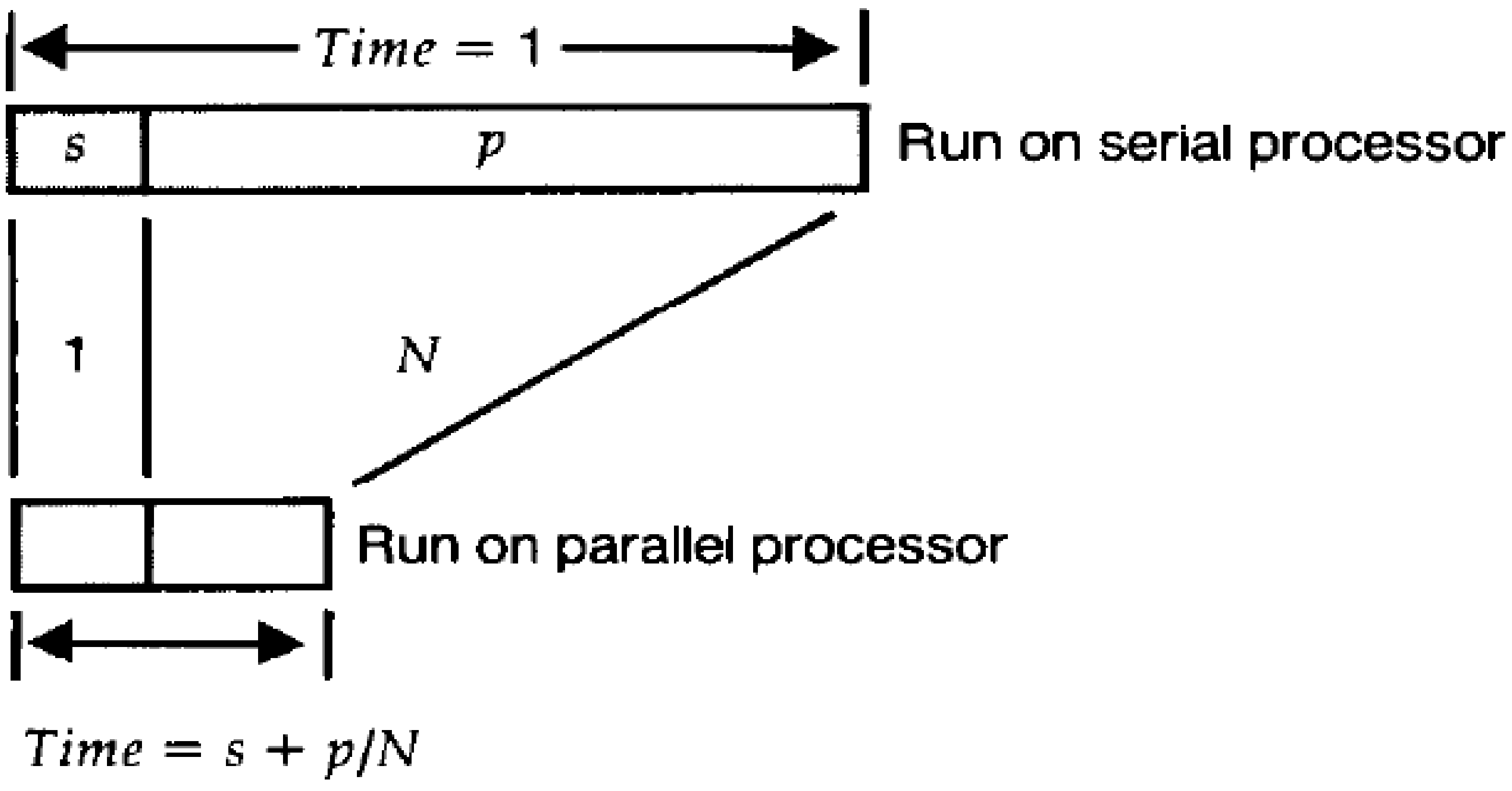


FIGURE 2a. **Fixed-Sized Model for** $Speedup = 1/(s + p/N)$

Графическая иллюстрация сравнения законов Амдала и Густафсона (2/3)

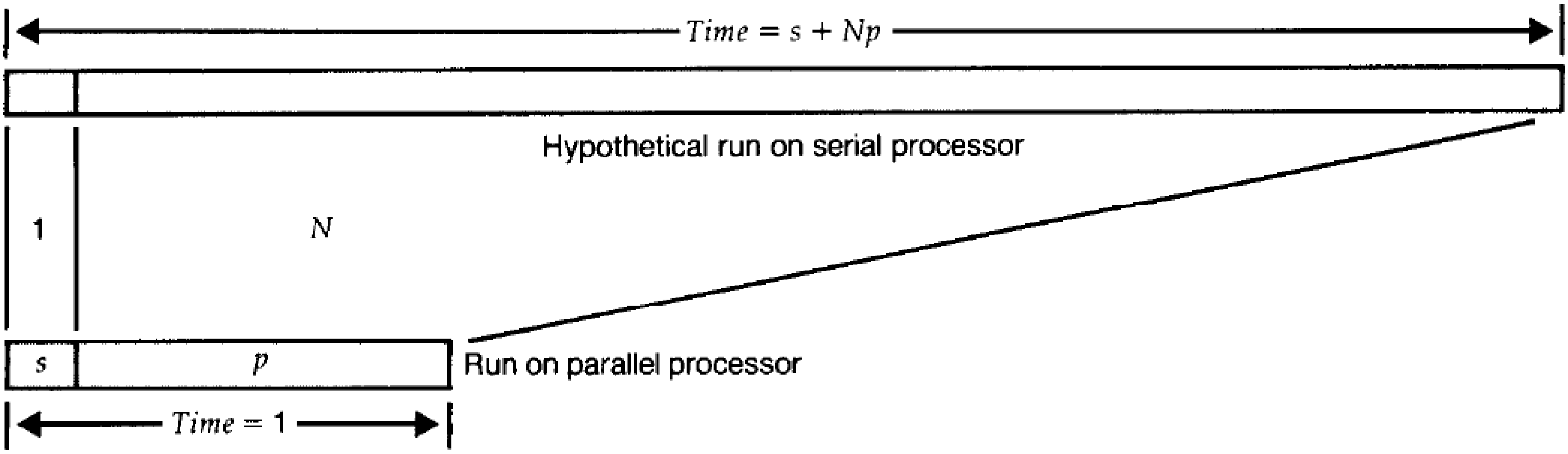
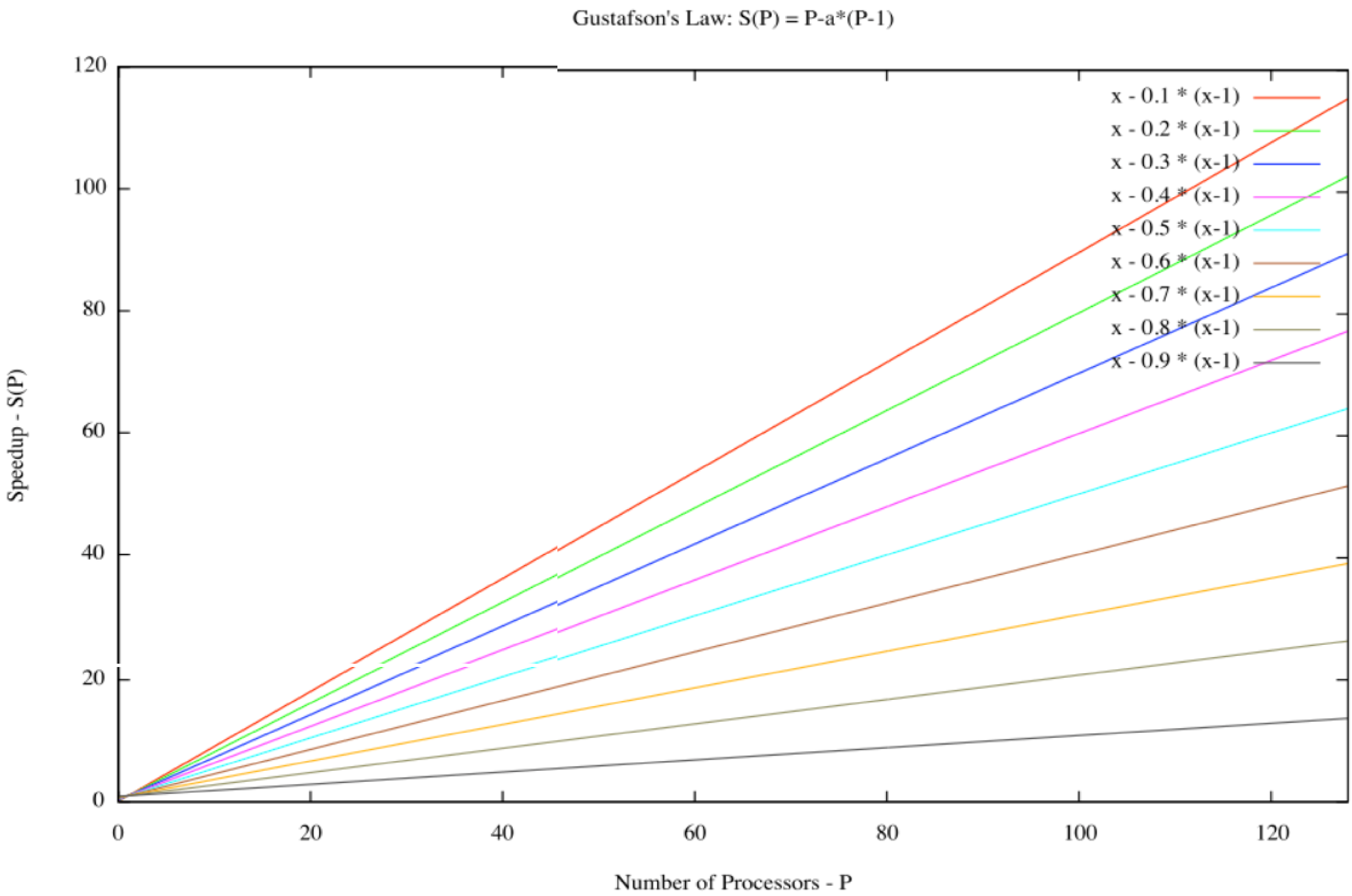


FIGURE 2b. **Scaled-Sized Model for** $Speedup = s + Np$

Графическая иллюстрация сравнения законов Амдала и Густафсона (3/3)



Библиографическая справка

- Gustafson, John L. "Reevaluating Amdahl's law." Communications of the ACM 31.5 (1988): 532-533.

Выводы

- Основными характеристиками параллельного алгоритма (программы) являются ускорение и эффективность
- Существует теоретический предел ускорения времени счёта для программы с данными фиксированного размера
- Однако для некоторых (масштабируемых) программ добавление процессоров позволяет решить задачу большего размера за тоже время, что и в исходной конфигурации параллельного компьютера