1 Введение

Причины использования параллельных алгоритм:

1. Сокращение времени решения прикладных задач
2. Обеспечение решения большего количества задач за единицу времени

Кризис эффективности

Благодаря закону Мура увеличение производительности приводило к уменьшению времени выполнения некоторой задачи. Ответ на пределе тактовой частоты - многопроцессорные системы. Однако последовательные программы не могут работать на многопроцессорных системах быстрее, чем на последовательных. В общем случае автоматически сконвертировать последовательную программу в параллельную невозможно - обычно это требует новых алгоритмов.

1.1 Параллельное вычисление

- **Одна ЭВМ с одним ядром**

  Компилятор часто уже генерирует параллельный код, тогда, когда он может. Это расширение SSE - Streaming SIMD extension. Например, операции с векторами или строками параллелизуются.

  - по умолчанию
  - вручную (асссемблерный код или использование istriscics)

- Одна ЭВМ с несколькими процессорами

  Примитивы:

  - Процессоры
  - Потоки
  - Средства межпроцессорного взаимодействия (IPC)
  - Примитивы синхронизации

  Технологии: OpenMP (Open Multi-Processing), Boost threads $\sim$ Java threads, Intel TBB, Java.util.concur Fork/Join Framework, CUDA, OpenCL.

- Несколько ЭВМ с несколькими процессорами - кластерные вычисления MPP (Massive Parallel Processing)

  - MPI (Message Passing Interface) стандарт обмена сообщениями
  - framework MapReduce

  Общие подходы:

  - Общие подходы:
  - ...

  GRID — форма распределённых вычислений, в которой «виртуальный суперкомпьютер» представлен в виде кластеров, соединённых с помощью сети, слабосвязанных гетерогенных компьютеров, работающих вместе для выполнения огромного количества заданий.

- Globus Toolkit
- Unicore
- ...

Cloud Computing - модель обеспечения удобного сетевого доступа по требованию к некоторому общему фонду конфигурируемых вычислительных ресурсов, которые могут быть оперативно предоставлены и освобождены с минимальными эксплуатационными затратами или обращениями к провайдеру.

1.2 Графические процессоры (GPU)

- CUDA (Compute Unified Device Architecture)
- OpenCL (Open Computing Language) OpenGL OpenAL
- Транзисторная память (transactional memory)

1.3 Закон Амдала

$$S_p = \frac{1}{\alpha + \frac{1-\alpha}{p}}$$

где α - доля последовательного кода, p - число процессов.

Содержание