Обзор области параллельных вычислений

Востокин Сергей Владимирович

План

- Суть параллельного программирования
- Краткий обзор аппаратного обеспечения
- Приложения и стили программирования
- Средства параллельного программирования, примеры, самостоятельная работа
  - Кластер «Сергей Королёв»
  - Система Templet Web

Суть параллельного программирования

Что представляет собой параллельная программа

- Несколько последовательных программ (процессов, потоков выполнения), совместно выполняющих задачу
- Последовательная программа — один поток управления
- Параллельная программа — несколько потоков управления

Что требуется для организации совместной работы

- Обмен данными
  - Разделяемые переменные
  - Обмен сообщениями
- Синхронизация
  - Взаимное исключение
  - Условная синхронизация

Когда появилось параллельное программирование

- В 1960-е годы изобретены *каналы* или *контроллеры устройств*
- Канал взаимодействует с процессором по *прерыванию*
- Многопроцессорные компьютеры
- Части программы выполняются в непредсказуемом порядке — возникает необходимость синхронизации

Возможности и трудности

- Потенциально получаем существенное ускорение вычислений ~10 ~100 ~1000 раз
- Необходимо разработать корректную синхронизацию
- Организовать взаимодействие процессов
- Накладные расходы на синхронизацию и взаимодействие не должны превышать выигрыш от распараллеливания

О чем не было сказано

- Кроме *императивных* (явный параллелизм) программ имеются декларативные программы — *функциональные* и *логические* (неявный параллелизм)
- Асинхронное программирование (условие → действие)
- Векторные и конвейерные вычисления

Краткий обзор аппаратного обеспечения параллельных вычислений

Процессор и кэш-память

Мультипроцессор с разделяемой памятью

Мультикомпьютер с распределенной памятью

Параллельные архитектуры

- Суперскалярные процессоры (несколько декодеров, конвейеризация, динамическое планирование)
- VLIW процессоры статическое планирование
- PLD — Программируемые Логические Интегральные Схемы
- SSE – Streaming SIMD Extensions
- SMT - Simultaneous multithreading
- Многоядерные процессоры
- Сопроцессоры — NVIDIA Tesla, ATI FireStream, Intel MIC (Xeon Phi)
- COW – cluster of workstations
- Суперкомпьютеры
- Грид-системы

Производительность параллельных вычислительных систем

- Flops (флопс) — FLoating-point Operations Per Second
- Операций с плавающей точкой в секунду
- Мега флопс — 10^6 флопс
- Гига флопс — 10^9 флопс
- Тера флопс — 10^{12} флопс
- Пета флопс — 10^{15} флопс
- Экза флопс — 10^{18} флопс

Примеры значений производительности

- Intel Core i5-2500K 3.3-3.7 ГГц (2011) — 105,5-118 Гфлопс
- СК Сергей Королёв, СГАУ – 21,3 Тфлопс
- СК Ломоносов, МГУ – 2,575 Пфлопс (22)
- СК Тяньхэ-2 — 54,902 Пфлопс (1)
- Грид BOINC - >8,5 Пфлопс
- Грид Bitcoin - > 1053 Пфлопс одинарной точности
- Ещё — см. сайт top500.org

Приложения и стили программирования

Классы приложений

- Многопоточные системы
- Распределённые системы
- Синхронные параллельные вычисления

Многопоточные системы

- Оконные системы, GUI
- Многопроцессорные операционные системы и системы разделения времени
- Системы реального времени, управляющие техническими объектами

  Причина — организовать код в виде набора потоков проще, чем в виде большой последовательной программы

Распределённые вычисления

- Сетевые файловые серверы
- Распределённые системы баз данных
- Web-серверы
- Системы, объединяющие компоненты производства
- Отказоустойчивые системы

  Причины — интеграция систем, удалённый доступ к данным, повышение надёжности системы

Синхронные параллельные вычисления

- Научные вычисления — математическое моделирование физических процессов в машиностроении, физике, науках о Земле, астрономии, медицине

  CAD/CAE: FEA, CFD, MBD, optimization
- Графика, обработка и синтез изображений
- Сложные комбинаторные или оптимизационные задачи, экономическое моделирование

  Причина — ускорение вычислений, решение задач большей размерности

Парадигмы в параллельном программировании

- Итеративный параллелизм
- Рекурсивный параллелизм
- Производители — потребители
- Клиенты и серверы
- Взаимодействующие равные

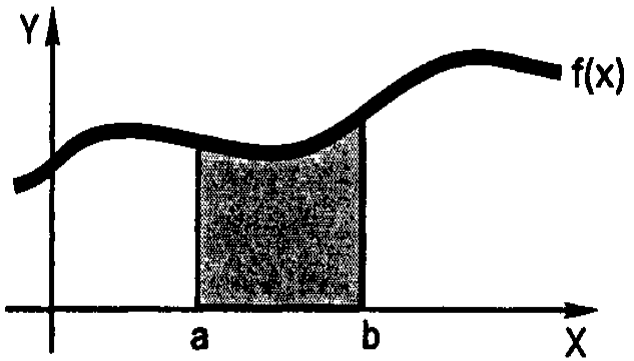
Итеративный параллелизм: умножение матриц (1/2)

```
co [i = 0 to n-1] { # параллельное вычисление строк
  for [j = 0 to n-1] {
    c[i,j] = 0.0;
    for [k = 0 to n-1]
      c[i,j] = c[i,j] + a[i,k]*b[k,j];
  }
}
```

Итеративный параллелизм: умножение матриц (2/2)

```
process worker[w = 1 to P] {   # полосы параллельно
  int first = (w-1) * n/P;     # первая строка полосы
  int last = first + n/P - 1; # последняя строка полосы
  for [i = first to last] {
    for [j = 0 to n-1] {
      c[i,j] = 0.0;
      for [k = 0 to n-1]
        c[i,j] = c[i,j] + a[i,k]*b[k,j];
    }
  }
}
```

Рекурсивный параллелизм: алгоритм адаптивной квадратуры (1/3)




```
double fleft = f(a), fright, area = 0.0;
double width = (b-a) / INTERVALS;
for [x = (a + width) to b by width] {
  fright = f(x);
  area = area + (fleft + fright) * width / 2;
  fleft = fright;
}
```

Рекурсивный параллелизм: алгоритм адаптивной квадратуры (2/3)

```
double quad(double left,right,fleft,fright,lrarea) {
  double mid = (left + right) / 2;
  double fmid = f(mid);
  double larea = (fleft+fmid) * (mid-left) / 2;
  double rarea = (fmid+fright) * (right-mid) / 2;
  if (abs((larea+rarea) - lrarea) > EPSILON) {
    # рекурсия для интегрирования обоих значений
    larea = quad(left, mid, fleft, fmid, larea);
    rarea = quad(mid, right, fmid, fright, rarea);
  }
  return (larea + rarea);
}
```

Рекурсивный параллелизм: алгоритм адаптивной квадратуры (3/3)

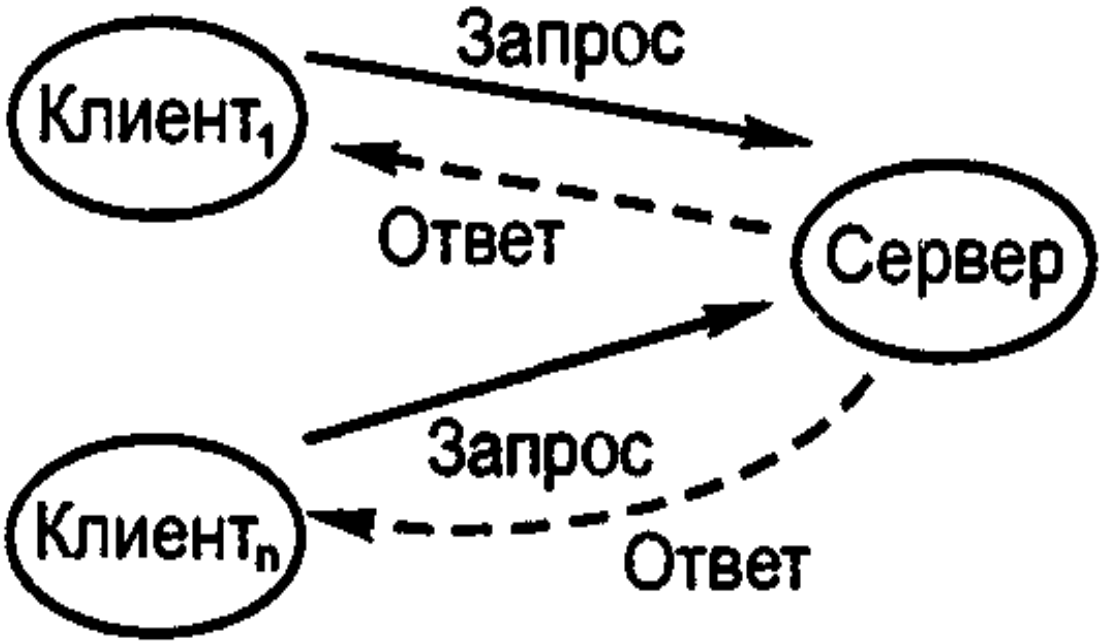
```
area = quad(a, b, f(a), f(b), (f(a)+f(b))*(b-a)/2);
```

```
co larea = quad(left, mid, fleft, fmid, larea);
// rarea = quad(mid, right, fmid, fright, rarea);
oc
```

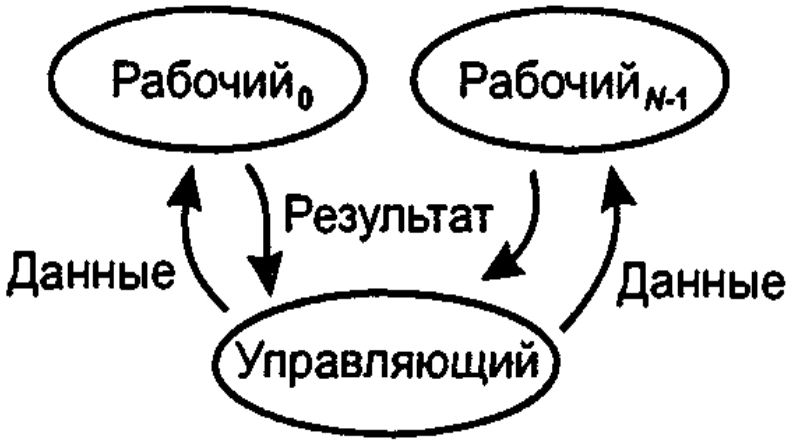
Производители — потребители: конвейер команд в Unix

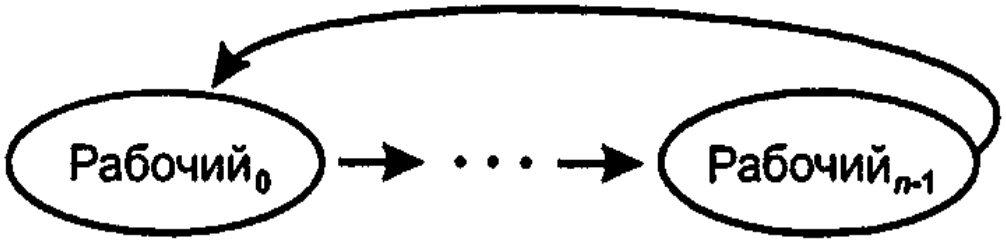
```
sed -f Script $* | tbl |eqn | groff Macros -
```

Клиенты и сервер



Взаимодействующие равные: распределённое умножение матриц





Другие презентации

Владимир Воеводин. "Суперкомпьютеры: незаметные гиганты". (http://tvkultura.ru/)

http://parallel.ru/info — сайт по суперкомпьютингу МГУ

Intuit.ru — лекции по параллельным вычислениям

Средства параллельного программирования

На чём пишутся параллельные программы

Уровни

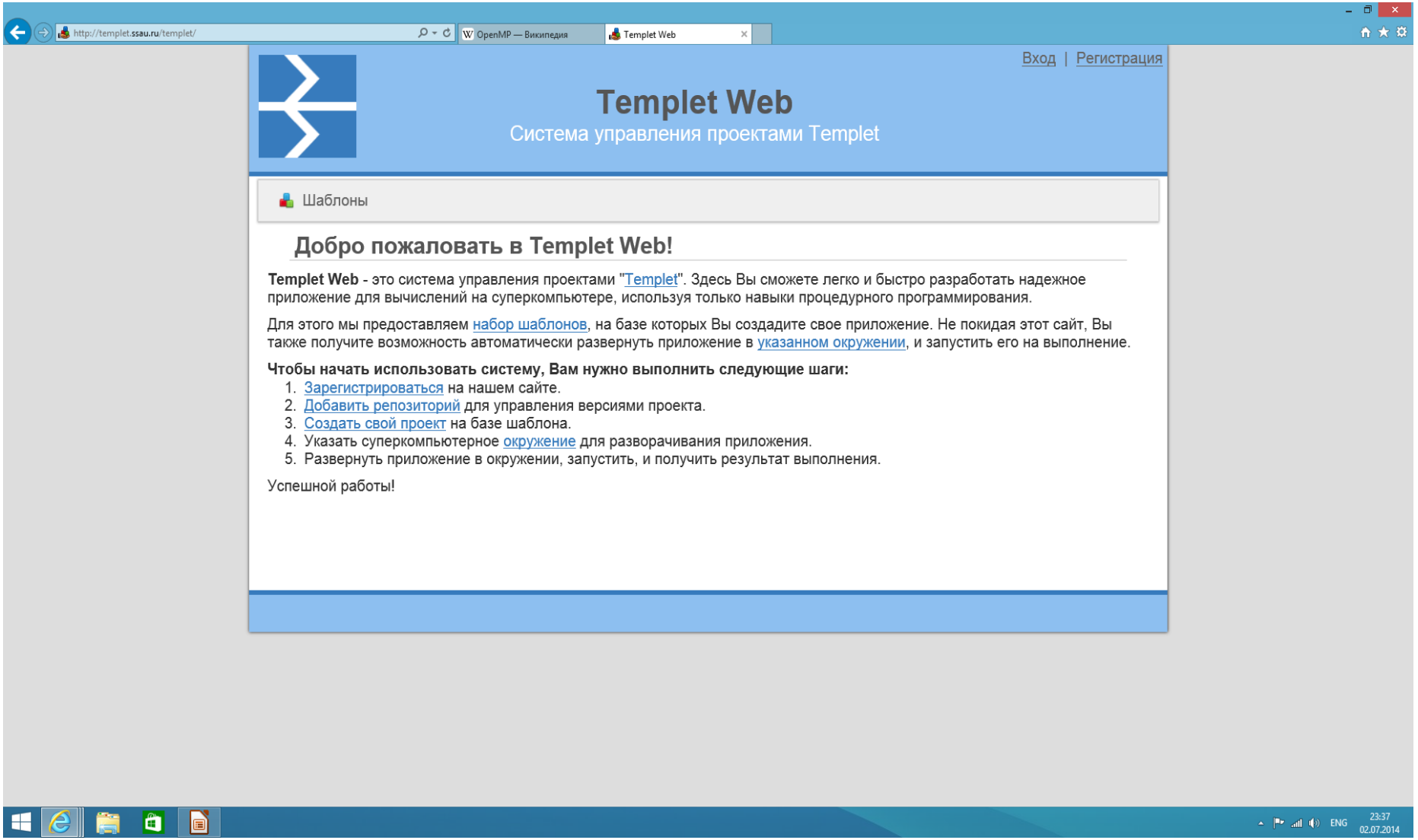
- Программирование GPGPU — CUDA, OpenCL, OpenACC, MS C++ AMP
- API операционных систем — Windows API, POSIX
- RTL -библиотеки языков программирования — C++11, Java, C#
- Дополнительные библиотеки — Intel TBB, MS CCR, Boost, POCO C++ libraries, Qt4 Threads
- Распределённое программирование — MPI, PVM
- RTL -библиотеки с поддержкой компилятора — OpenMP, Cilk Plus
- Фреймворки — Apache Hadoop, BOINC

Инструменты

- MS Visual Studio 2010 Prof, Ulti, 2012 — All (OpenMP)
- Intel Parallel Studio -Intel Parallel Composer, Intel C++ Compiler (OpenMP, Intel Cluster OpenMP, Cilk Plus)
- GCC — 4.2 (OpenMP)

Суперкомпьютер «Сергей Королёв» Самарского государственного аэрокосмического университета (hpc.ssau.ru)

Система организации вычислений через web-интерфейс (templet.ssau.ru/templet)



Инструменты, используемые в самостоятельной работе

- Проект под управлением Subversion
- Microsoft Visual Studio
- Клиент Subversion
- Проект под управлением Templet Web

Порядок выполнения самостоятельной работы

- Настройка учётной записи в системе Templet Web, настройка проекта в системе контроля версий SVN
- Реализация параллельной программы с использованием API операционной системы
- Реализация параллельной программы по тому же алгоритму с использованием OpenMP / MPI
- Добавление в программы кода для анализа производительности. Построение зависимости ускорения параллельной программы от размера решаемой задачи по результатам вычислительных экспериментах
- Оформление отчёта