ТЕОРИЯ И ПРАКТИКА МНОГОПОТОЧНОГО ПРОГРАММИРОВАНИЯ

Тема 1

Введение

Д.ф.-м.н., профессор А.Г. Тормасов
Базовая Кафедра «Теоретическая и Прикладная Информатика», МФТИ

Почему? …

- Производители процессоров «уткнулись в множество стен»:
  - Мощность (выше частота –> больше мощность -> больше потерь на тепло – и как отдавать тепло?)
  - Скорость света (скоро будет так, что за один такт сигнал уже не успевает передаться с одного конца ЦПУ на другой!)
  - Скорость памяти (тормозит выполнение программ, часто количество промахов кэша определяет производительность!)
  - Умозрительное исполнение ограничено (сложность растет экспоненциально с глубиной)

А потому, что…

- Решение: «потроха наружу», и пусть программист заботится о производительности!
  - Примерно так же как во времена начала перехода CISC -> RISC
- Типичный современный компьютер:
  - Много быстрых ядер и гиперпотоков
  - Много кэшей разного уровня
  - Медленная память (штраф за обращение!)
- Написание программы в «традиционном» последовательном стиле на 16 потоковой системе приведет к тому, что…
- от полной мощности системы ваша программа может получить уже около 1/16 ~ 5-6%!

парадигма

- Традиционная: последовательная
  - Я один на машине
  - Мне никто не мешает
  - Думаю только о правильности алгоритма
- Современная: параллельная
  - Меня на машине много (много потоков)
  - Мне мешает ОС, другие программы, я сам и тд
  - Думаю и о корректности работы алгоритма в многопоточном окружении
  - Думаю и об общей скорости работы алгоритма, задействованных устройствах, попадании в кэши и тд
- Практически вся рассматриваемая «математика» появилась в XXI веке!

До курса:

- Умею программировать (слегка)
- Знаю язык C
- Знаю на уровне прикладного программиста
  - Аппаратуру
  - ОС и АПИ системных вызовов
  - Базовые синхронизационные вызовы
- Знаю что такое машина Тьюринга

Содержание курса

- «Аппаратура»
- Классика параллельного программирования(небольшая ее часть, не претендуя на замену других традиционных курсов ПП)
- Теория (математика): теоремы, доказательства, анализ алгоритмов (на корректность, производительность, иногдп в лучшем, в худшем, в среднем,...)
- Практика (программирование): написание реально работающих программ, исследование поведения алгоритмов под нагрузкой(отдельные занятия)
- Все вместе: работа с неблокирующими алгоритмами в программах

Содержание курса

- Основы архитектуры современных микрокомпьютерных систем
  - Только те сведения, которые влияют на производительность программ
  - Способы явного и неявного программного манипулирования компонентами аппаратуры
  - Аппаратура, система команд и языки программирования (C-подобные)
  - Акцент на архитектуру x86
- Некоторые необычные понятия параллельного программирования
  - Уровни абстракции, не-существование «а на самом деле...»
  - Атомарность
  - Время
  - Корректность
- Специфика работы с разделяемой памятью
  - Общие проблемы параллельности и частные проблемы раздельного доступа
  - Методы организации синхронного доступа

Содержание курса

- Математика параллельного программирования
  - Некоторые математические аппараты, применяемые для анализа и моделирования параллельных программ
  - Формализация понятий
    - История, завершенность, сериализованность, линеаризованность, легальность и тд
    - Прогресс, условный и безусловный
- Сравнение примитивов друг с другом, относительная мощность
- Использование задачи о консенсусе для сравнения примитивов синхронизации
- Числа консенсуса для типовых примитивов

Содержание курса

- Свобода – наша цель
  - Свободные, свободные от блокировок, свободные от ожидания и с ограниченным от ожидания прогрессом
  - Эффективность разных свободных алгоритмов

- Невозможность – наш ограничитель
  - Невозможно улучшить примитив
  - Невозможно решить задачу о консенсусе инструкциями чтения записи
  - Невозможно решить задачу о консенсусе для системы со сбоями
  - Невозможно…
    - Но! *Невозможно решить точно* не означает *нерешаемо*! Пример: Ethernet

Содержание курса

- Неблокирующие алгоритмы – наше будущее?
  - Существующие алгоритмы
  - Когда можно применять, и зачем
  - Как реализовать
  - Как создать новый
- Упражнения и задание (TBD)

После курса:

- Уметь анализировать работу параллельных потоков, использующих общую (разделяемую) память
- Понимать проблемы и терминологию параллельного программирования, математический аппарат, применяемый для описания и анализа
- Знать современные подходы к организации параллельных вычислений, в том числе высокопроизводительных
- Знать ограничения алгоритмики и не заниматься «квадратурой круга» или изобретением велосипеда
- Уметь создавать эффективные параллельные алгоритмы и понимать, когда их применение оправдано

Дополнения

- Источники, на базе которых появился курс
  - Потребность в новых алгоритмах и программах
  - Книга Шафита, Херлихая и курсы на ее основе
  - Недостатки имеющихся источников (слишком «молодая» тема)
  - Личный опыт программирования и написания высокопроизводительных программ (как следствие – использование C и x86 для примеров)
- Литература
  - Увы, почти вся англоязычная, причем большинство в виде «зубодробительных» научных статей
  - Готовится пособие по курсу (уже выпущена часть про аппаратуру)
- Практические занятия
  - Задание со списком задач (TBD)
  - Практикум – реализация и анализ какого либо алгоритма
  - Дополнение к курсу в виде библиотеки неблокирующих алгоритмов
- Комментарии, ошибки в материалах, предложения – welcome!
  - tor@parallels.com

НИР

- На кафедре информатики МФТИ ведется работа по методам анализа неблокирующих алгоритмов
- Можно ли построить автоматический метод анализа алгоритмов на корректность и отсутствие неразрешенных гонок (сейчас это крайне сложно, например, анализ неблокирующей хеш таблицы на корректность методом машинной верификации занял 2 года!).
- Новые подходы к неблокирующим алгоритмам, не использующим блокировки шины (CAS) и барьеры
- Защищены (к.ф.-м.н.) 2 первые диссертации по этим темам
- Желающие заниматься – welcome!

(c) А. Тормасов, 2010-11 г.

Базовая кафедра «Теоретическая и Прикладная Информатика» ФУПМ МФТИ
tor@ crec .mipt .ru_