ФРТК

iSCALARE

intel

Лаборатория суперкомпьютерных технологий для биомедицины, фармакологии и малоразмерных структур

Общие вопросы моделирования: требования, терминология

Григорий Речистов

grigory.rechistov@phystech.edu

18.02.2013

На прошлой лекции:

Симуляторы изменили принципы проектирования вычислительных систем

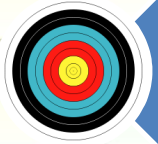
Они стали незаменимы при совместной разработке аппаратуры и ПО

Компьютеры позволили нам решать многие проблемы, которых до момента их создания не существовало

Главная «фишка» моделирования – частичное устранение сложности изучаемой системы, связанной с теми её аспектами функционирования, которые нас не интересуют

Однако, сами симуляторы как программы обладают своими особенностями, которые приходится преодолевать

Требования к симуляторным решениям



Точность

Скорость

Совместимость со сторонними системами

Способность решать задачу пользователя

Точность

- Точность должна быть достаточной для выполнения целей, поставленной перед моделью
- Излишняя точность => медленная работа, дольше разработка, больше ошибок

Точность

- Функциональная точность – модель отвечает на внешние воздействия так же, как и реальная система
- Точность предсказаний – получаемые качественные/количественные характеристики адекватны реальности

Откуда берутся данные для создания модели

- Есть спецификации => создаётся модель, удовлетворяющая им
- Есть модель для «предыдущего» устройства => она модифицируется под новую спецификацию
- Иногда приходится писать модель устройства с помощью «reverse engineering» анализа

Как проверить корректность модели

- Функциональная корректность – программа бежит и не падает
  - «Только в наших мечтах, мистер Виггльз»
- Сравнение с другой моделью того же устройства
  - Функциональная модель vs потактовая
  - Разные поколения одной модели
- Сравнение с реальной аппаратурой
  - Если она уже существует

На чём проверять корректность модели

- Полноплатформенный симулятор – на загрузке ОС
- Симулятор уровня приложения – на работе приложений
- Оба варианта (или неполные модели) – на трассах приложения (историях событий: чтение/запись в память, внешние прерывания и т.д.). При этом ищутся отличия в эволюции состояния модели

Как **верифицировать** корректность модели

- Формальная верификация – строгое доказательство, что система функционирует согласно спецификации
- Требует исследования всех сценариев функционирования системы
- Пример: инструкция IA-32 длиной до 16 байт => требуется перебрать 2^128 комбинаций входных последовательностей
- **Сложный вопрос**

Как измерять скорость симуляторов?

1. Относительная скорость симуляции относительно реального исполнения системы

- Применима, если есть реальная система, с которой проводится сравнение
- Специальный случай «целевая архитектура == хозяйская» (target == host)
- Чаще всего модель исполняется медленнее реальной аппаратуры

Как измерять скорость симуляторов?

1. Относительная скорость симуляции относительно реального исполнения системы

Относительное замедление	Тип модели
1…5	Функциональная, использующая аппаратное ускорение
1…100	Функциональная, с двоичной трансляцией
100…1000	Функциональная, с интерпретацией
10000…100000	Потактовая

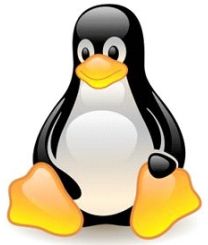
Как измерять скорость симуляторов?

2. MIPS – mega instructions per second

- Под инструкциями понимаются моделируемые инструкции
- Позволяет сравнивать исполнение симулятора на различных этапах его работы
- Позволяет сравнивать симуляторы между собой

Как измерять скорость симуляторов?

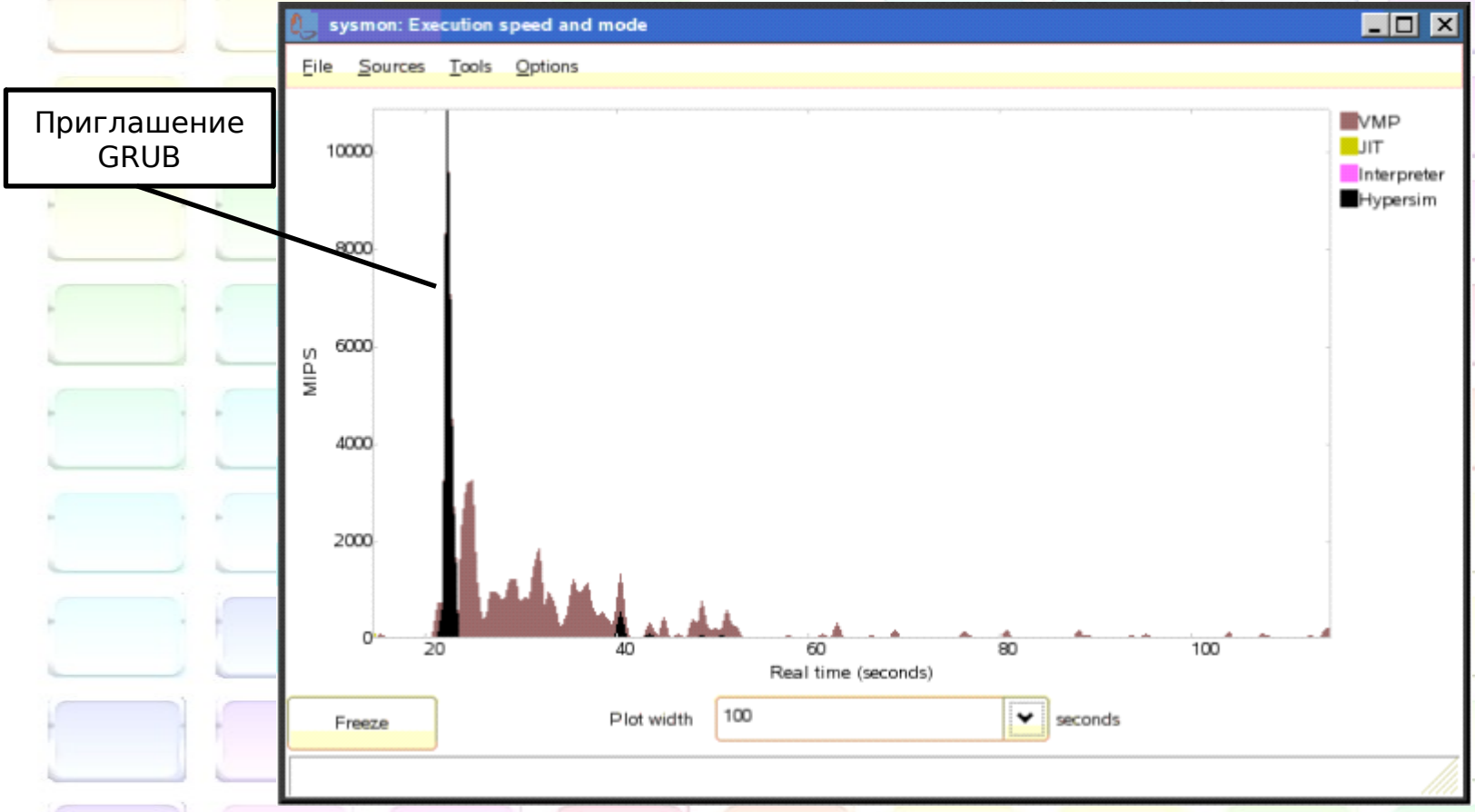
Пример: загрузка Linux



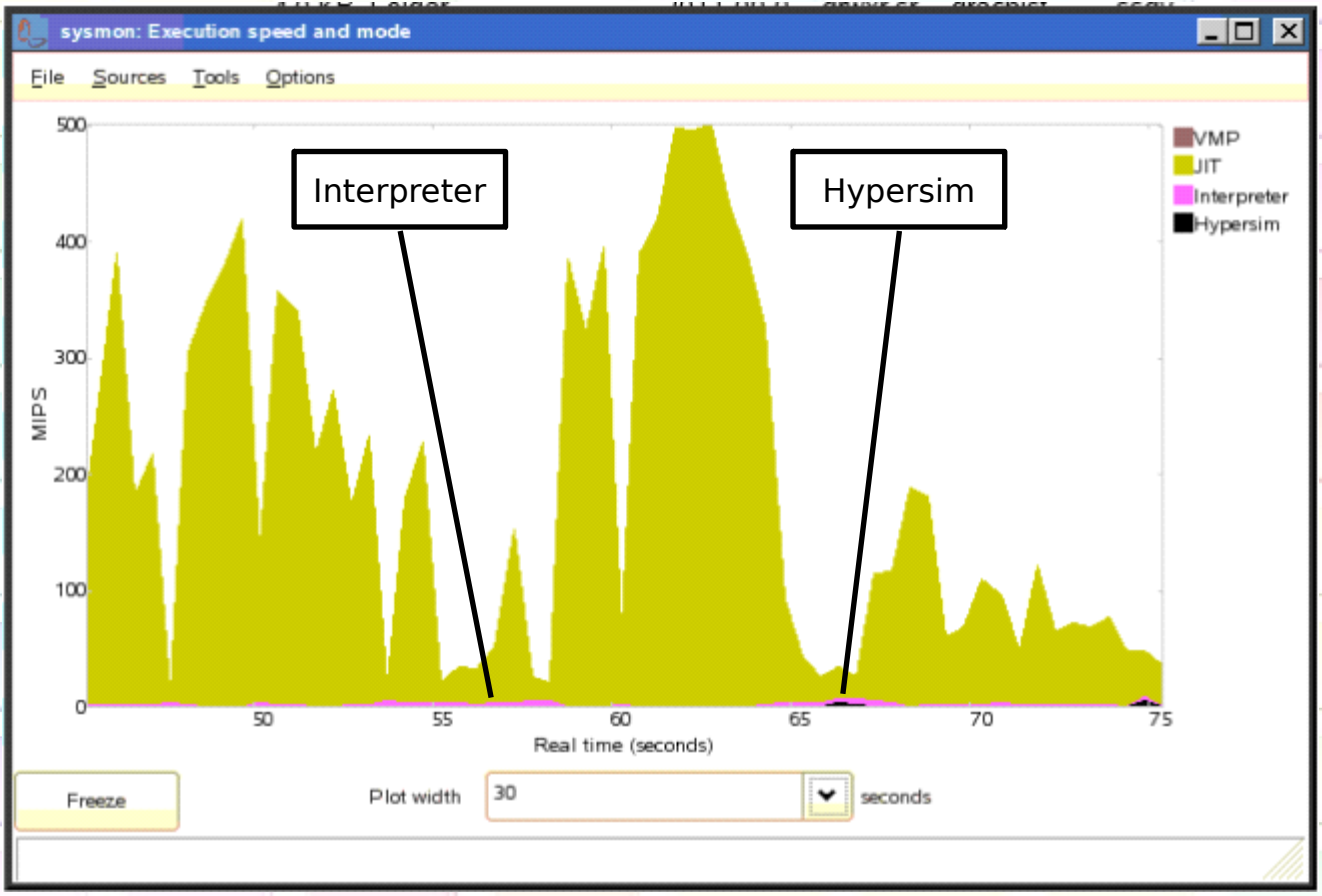
ОС Red Hat Enterprise Linux 5 (64 бит)

Симулятор Wind River Simics 4.6

Как измерять скорость симуляторов?



Как измерять скорость симуляторов?



Как повысить скорость симулятора?

Это мы будем обсуждать на многих последующих занятиях ☺

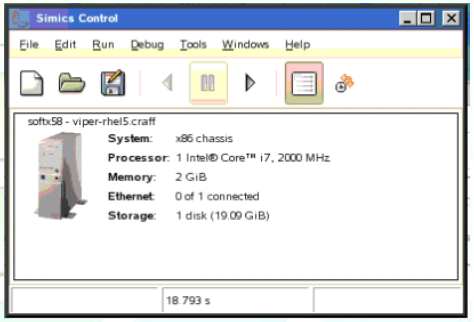
Эффективная симуляция, эффективные простои, аппаратная поддержка, параллельное исполнение…

Совместимость

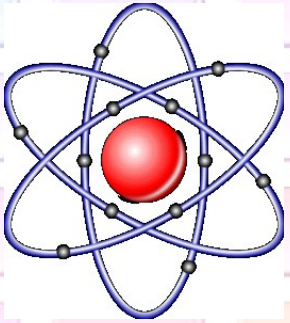
- Существует множество утилит, программ и форматов, знакомых пользователям, и которыми они хотели бы продолжать пользоваться в дальнейшем
  - Отладчики, среды разработки, анализаторы трасс, ...
  - Форматы: образы дисков, трасс, содержимого памяти...
- Ожидаемые от симулятора возможности
  - Автоматизация/расширяемость с помощью динамических языков: Perl, Python, ...
  - Автоматическая/фоновая работа без человеческого вмешательства: без GUI, регулярное тестирование, ...

Структура симулятора

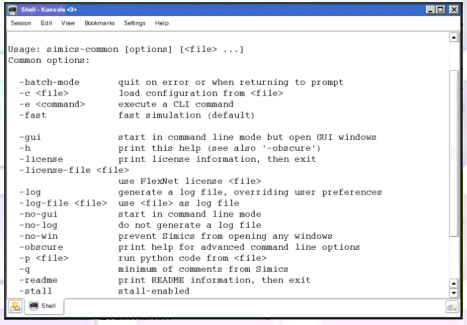
Графический интерфейс



Симуляционное ядро



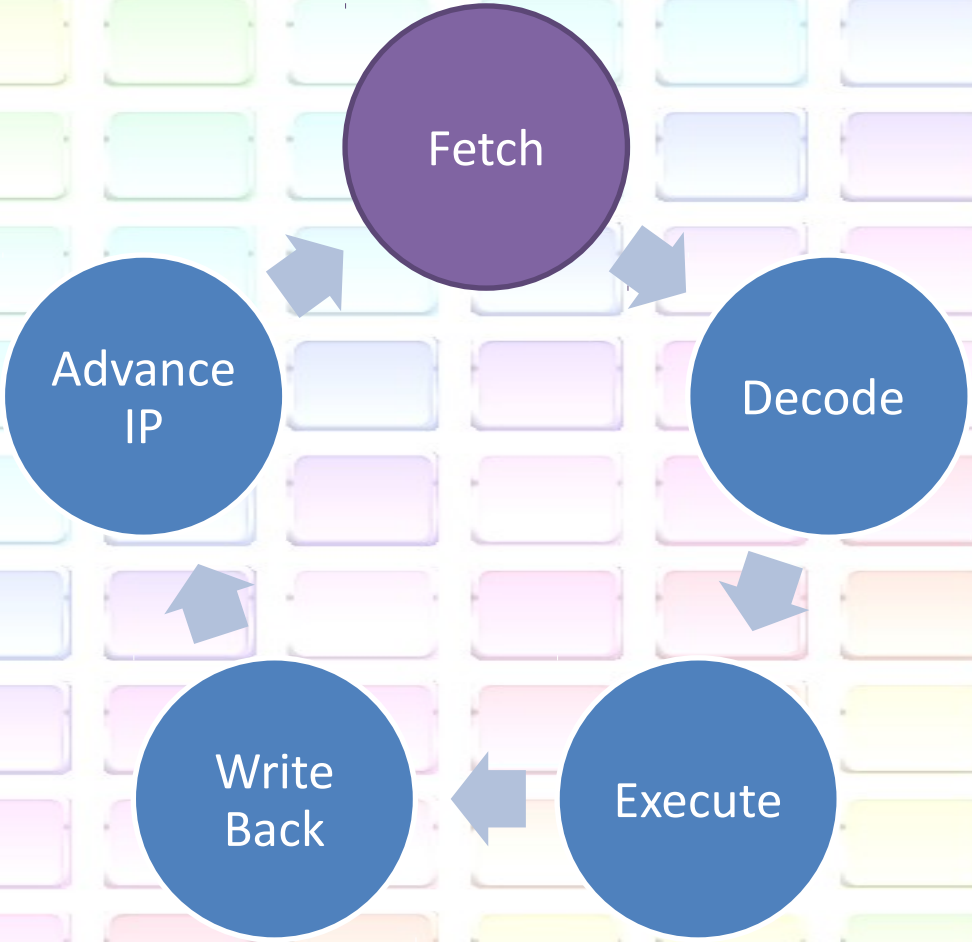
Интерфейс командной строки



Интерпретатор скриптов

Итоги

На следующей лекции: интерпретация



Спасибо за внимание!

Все материалы курса выкладываются на сайте лаборатории:
http://iscalare.mipt.ru/material/course_materials/