Нижегородский государственный университет им. Н.И. Лобачевского

Проект «Создание системы подготовки высококвалифицированных кадров в области суперкомпьютерных технологий и специализированного программного обеспечения»

Инструменты параллельного программирования для систем с общей памятью

Intel Thread Profiler
Краткое описание
Часть 1: Общие вопросы

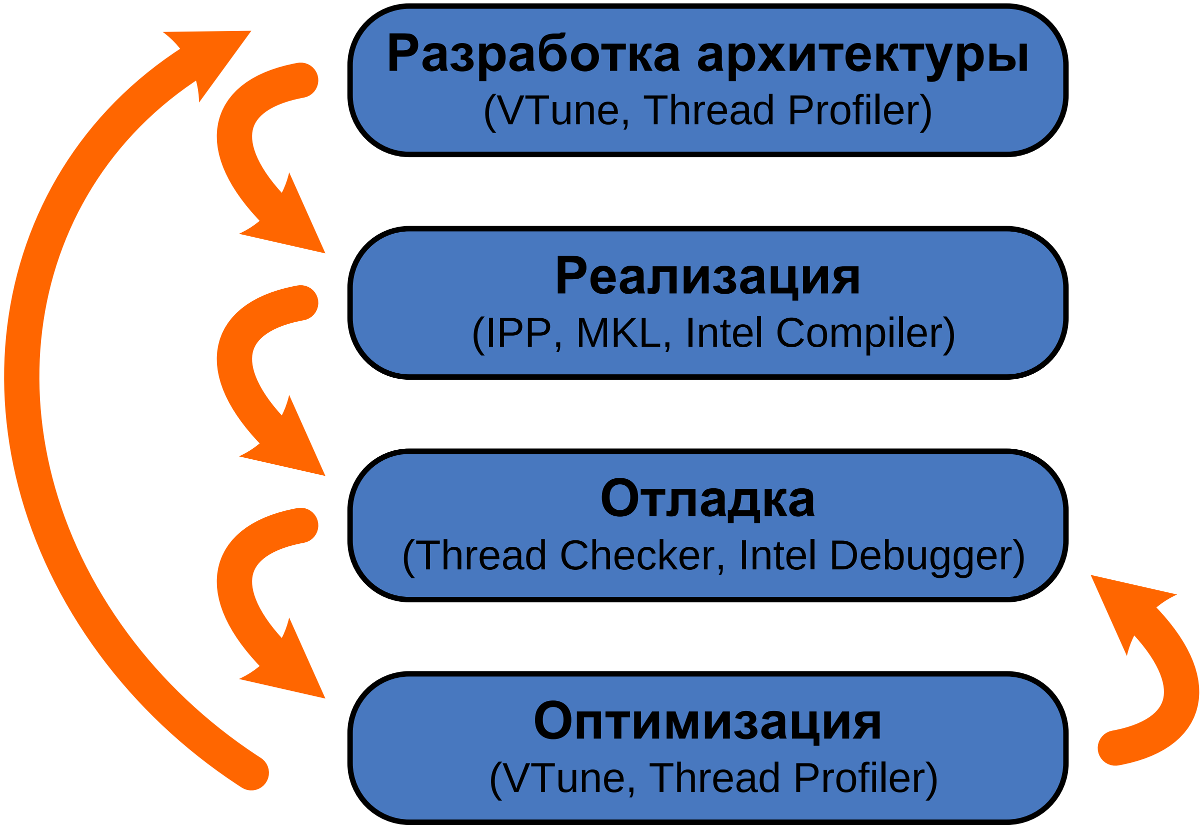
Корняков К.В., Шишков А.В.
Кафедра математического обеспечения ЭВМ

Содержание

- Введение
- Технические характеристики Intel® Thread Profiler
- Основные концепции и понятия профилирования
- Круг решаемых задач
- Общий порядок работы с инструментом

Введение

- Цикл разработки приложения



Введение

- *Профилирование* – процесс анализа производительности приложения и учета потребляемых им ресурсов.
- *Профилировщик* – инструмент, при помощи которого осуществляется профилирование.
- *Трасса (профиль)* – список событий, произошедших в приложении (исполненные инструкции, вызовы функций, вызовы операционной системы).

Оптимизация многопоточных приложений ***существенно*** сложнее, чем последовательных.

Введение

- Intel® Thread Profiler (ITP) предназначен для профилирования многопоточных приложений с целью их последующей оптимизации и настройки.
- Варианты использования:
  - анализ эффективности распределения вычислительной нагрузки между потоками;
  - определение доли накладных расходов на поддержку потоков;
  - выбор оптимальной архитектуры многопоточного приложения (число потоков, алгоритм, примитивы синхронизации);
  - определение наиболее медленных и нуждающихся в оптимизации потоков;
  - анализ масштабируемости приложения;

Технические характеристики

- ITP позволяет профилировать многопоточные приложения, созданные при помощи следующих технологий:
  - OpenMP
  - Windows API threads
  - POSIX threads
- ITP является плагином для VTune Performance Analyzer.

Технические характеристики

Аппаратное обеспечение

- **Минимальные требования**
  – Процессор Pentium 4;
  – ОЗУ 512 МБ;
  – Свободное дисковое пространство 300 МБ.
- **Рекомендуемые**
  – Процессор Pentium 4, поддерживающий технологию Hyper-Threading, или процессор Intel Xeon;
  – ОЗУ 2 ГБ.

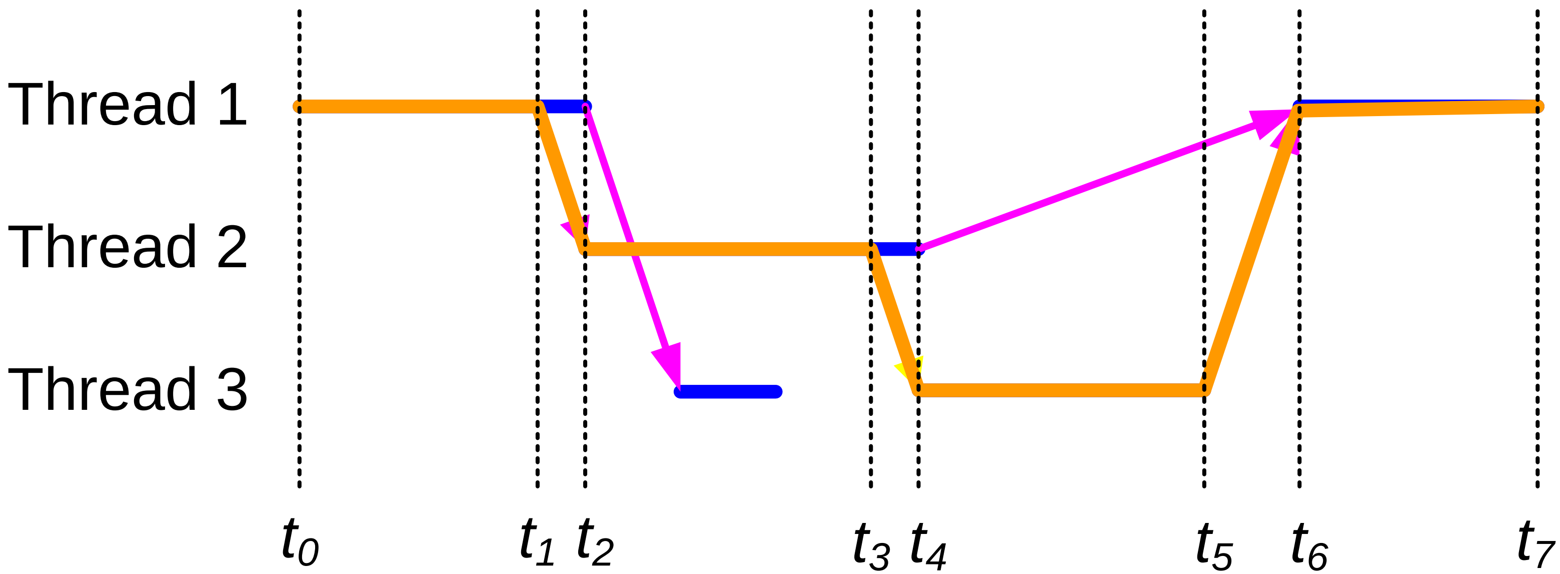
Технические характеристики

Программное обеспечение

- **Минимальные требования**
  - Windows XP Professional, или Windows Server 2003.
  - Internet Explorer 6.0 или выше.
  - Visual C++ 6.0 или выше.
  - Adobe Acrobat Reader.
- **Требуемое программное обеспечение для анализа OpenMP-приложений, и инструментации на уровне исходных кодов**
  - Intel C++ Compiler 8.1 для Windows для архитектуры IA-32;
  - Intel C++ Compiler 9.1 для Windows для архитектуры Intel EM64T;
  - Intel Fortran Compiler для Windows 8.1, Package ID: w_fc_pc_8.1.023 или выше.

Основные концепции и понятия профилирования

- *Поток исполнения* – последовательность команд приложения, исполняемая на процессоре.
- *Критический путь* – последовательность действий, которая определяет насколько быстро может завершиться приложение.



Основные концепции и понятия профилирования

- Состояние потока
  - *Active* (поток исполняется в настоящее время);
  - *Spin* (активное ожидание);
  - *Wait* (неактивное ожидание).
- Типы поведения потока в активном состоянии*:*
  - *Impact* (поток активен и сдерживает выполнение другого потока);
  - *Blocking* (поток находится в состоянии ожидания);
  - *Critical Path* (поток активен, но нет потоков, его ожидающих).
- Уровень параллелизма (concurrency)
  - *No Thread Active*
  - *Serial*
  - *Under Utilized*
  - *Fully Utilized*
  - *Over Utilized*
  - *Overhead*

Основные концепции и понятия профилирования

- Категории времени

Behavior	Processor Utilization				
	Bad			Good	Over Utilized
	Idle 0	Serial $n = 1$	Under $(n < p)$	Full $(n = p)$	$(n > p)$
Impact					
Blocking					
Critical Path					
Overhead					

Рисунок взят из [7]

Круг решаемых задач

Распределение вычислительной нагрузки

- Время работы приложения в значительной степени определяется самым медленным из потоков.
- Пример: стоит задача обработки множества заявок, трудоемкость каждой из которых заранее неизвестна.
- ITP позволяет выявить следующие признаки дисбаланса:
  - большая доля последовательных вычислений
  - значительные различия во времени работы потоков
- Решение должно учитывать специфику решаемой задачи.

Круг решаемых задач

Выбор примитивов синхронизации

- Разработчику важно выбрать наиболее подходящий тип примитива синхронизации для каждой конкретной ситуации (атомарные признаки блокировки, мьютексы, семафоры, критические секции).
- ITP позволяет определить время, затраченное на работу с каждым из примитивов синхронизации. Разработчик может реализовать несколько вариантов и сравнить их между собой.
- Для увеличения глобального счетчика, например, нерационально использовать мьютексы. Гораздо удобнее использовать атомарные функции ОС Windows, такие как **`InterlockedIncrement`**.
- Лучше делать выбор в пользу примитивов синхронизации пользовательского уровня (**`CriticalSection`**, например), поскольку они не генерируют системных вызовов операционной системы.

Круг решаемых задач

Синхронизация между потоками

- Синхронизацию между потоками нужно стараться делать как можно реже. Слишком частое обращение нескольких потоков к разделяемому ресурсу приводит к тому, что большое количество времени потоки простаивают, ожидая освобождения ресурса.
- С помощью ITP определяются объекты, обращение к которым происходит наиболее часто. Затем анализируется насколько затратно столь частое обращение к объекту.
- Если обнаруживается, что производительность существенно страдает, разработчику следует попытаться изменить архитектуру приложения. Хороший пример – замена глобальных переменных локальными.

Круг решаемых задач

Непроизводительные издержки

- Потоки за время своей жизни должны совершать работу гораздо большей сложности, чем сложность их собственного создания, уничтожения и управления ими.
- ITP позволяет определить долю непроизводительных издержек от общего времени работы приложения. Если она оказывается слишком велика, скорее всего, приложение требует перепроектирования.
- Так, например, если создается система, обслуживающая поступающие в режиме реального времени запросы, то лучше содержать пул потоков, вместо того, чтобы каждый раз создавать новый поток для обработки очередного запроса.

Общий порядок работы

- Порядок работы с Intel Thread Profiler включает в себя следующие шаги:
  - инструментация приложения;
  - профилирование приложения;
  - анализ полученной информации.

Общий порядок работы

Инструментация приложения

- *Инструментация* – это встраивание в приложение дополнительных вызовов, при помощи которых профилировщик получает информацию о работе приложения.
- Различают инструментацию исходных кодов приложения (source instrumentation) и бинарную инструментацию (binary instrumentation). ITP поддерживает оба этих способа. В основном используется бинарная инструментация.
- Инструментация исходных кодов используется крайне редко.
  - Бинарная инструментация недоступна
  - Необходимо запустить инструментированное приложение вне ITP

Общий порядок работы

Профилирование приложения

- ITP осуществляет запуск инструментированного приложения, в процессе которого собирает его трассу, в которую включается следующая информация:
  – идентификаторы созданных потоков;
  – время создания и уничтожения каждого конкретного потока;
  – участки кода, в которых происходило создание потоков;
  – количество времени, которое каждый поток провел в состоянии ожидания;
- При профилировании старайтесь следовать следующим советам:
  – Избегайте запускать другие приложения во время профилирования.
  – Осуществите профилирование несколько раз и для анализа выберите тот запуск, когда приложение отработало быстрее всего.

Источники информации

1. «Developing Multithreaded Applications: A Platform Consistent Approach», Intel Corporation, March 2003.
2. «Threading Methodology: Principles and Practices», Intel Corporation, March 2003.
3. «Multi-Core Programming for Academia», Student Workbook, by Intel.
4. «Multi-Core Programming», book by Sh. Akhter and J. Roberts, Intel Press 2006.
5. «Intel® Thread Profiler. Getting Started Guide».
6. «Intel® Thread Profiler. Guide to Sample Code».
7. «Intel® Thread Profiler Help».
8. «Intel® Thread Profiler – краткое описание», материалы по образовательному комплексу «Технологии разработки параллельных программ».