1 Первая лекция (вторая)

Преподлватель: Андрей Горобец.

Сферы применения: можно рассчитывать свойства предметов до того как тот был создан.

На лекции показывали примеры.

g++ - fopenmp функция что это?

рассмотрели unroll.

Классификация параллелизма Флинна.

Флопсы количество произведенных инструкций деленные на время. TPP: пиковая производительность устройства. BW: пропускная способность памяти. AI: флоп на байт вычислительная интенсивность (как сильно нагружен процессор к памяти) TPB: теоретически достижимая производительность. Rmax: фактически достигаемая производительность.

CPI - количество тактов на операцию.

IPC - количество инструкций выполняемых за такт.

FLOP per cucle - количество операций с плавающей тоской.

ILP - конвейерный параллелизм и суперскалярность.

SIMD расширение - параллелизм ядра процессора.

FMA операции (AVX-512), векторные инструкции. + в может производить за один таки последовательно сложение люс умножение. 16 операций за такт.

Ключевой момент - доступ к памяти.

Устройство конвеера. Разные типы конфликтов, которые прерывают конвеер.

Как определяется область видимости переменной в си.

OoOE (out-of-order) процессор растаскивает инструкции.

vLIW - very long instruction word. Компилятор сам распределяет инструкции.

SMT - simultaneous multithreading одновременная многопоточность. Команды из разных потоков выполняются одновременно.

Иерархия памяти.

Поколения памяти - пропускная возможность увеличивается, но задержка оо- швах.

Кэш. Выравнивание из-за кэша.

2 Лекция 2

FUA - fused multiplier addition.

LLC - распределенный кеш.

2.1 Основы стандарта OpenMP (Старый).

макрос для секций

PU - processing unit. С общей памятью.

Характеристики памяти: Пропускная способность и время доступа (латентность).

Неоднородная общая память - NUMA (в разные места разное время доступа и пропускная способность).

Презентацию добавить.

Неоднородность работы.

Определение NUMA раздела с помощью масок.

3 Лекция 3

Имя лектора: Михаил Владимирович Якобовский.

Цель курса - дать понимание перед тем как писать программу имеет ли смысл ее распаралелливать.

Большая часть суперкомпьютеров > 100000

Четыре области применения многопроцессорных систем

1. Сокращение времени решения высокопроизводительных задач.

2. Сокращение времени обработки больших данных.

3. Решение задач в реальном веремени.

4. Создание систем высокой надежности.

Оптически матричный умножитель - хорошая производительность, но низкая точность (специфический класс систем).

На нижнем уровне данные всегда передаются синхронным методом.

В итоге все равно будет передача точка-точка, даже в групповых передачах данных.

За микросекунду процессор выполняет примерно 1000 операций (порядок 10^3).

Ускорение это $S_p = \frac{T_1}{T_p}$

Эффективность $E_p = \frac{S_p}{p}$

Предел масштабируемости (не универсальное определение) - минимальное число процессоров, при котором достигается максимаольное ускорение.

Каскадная схема сложения 8-ми чисел в 1000 раз медленнее на многопроцессорной системе (задача очень маленькая, из-за этого идет замедление, слишком много взаимодействий между процессорами).

T_1^* - Самый быстрый последовательный алгоритм. Относительно него не должно быть сверхлинейного ускорения.

4 Лекция 4

$\tau_c -$ время операции

τ_s время передачи сообщения

4.1 Метод герметрического параллелизма

Задания должны быть связаны только локально, то он эффективен.

$$T_1 = (kn) = \tau_c kn$$

$$T_p(kn) = \tau_c \frac{kn}{p} + 4k\tau_S$$

$$S_p(kn) = p\frac{1}{1+4}$$

Перед тем как выполнить алгоритм нужно потратить еще логарифм от количества узолов передач сообщений.

Статическая балансировка и динамическая балансировка - различается тем что в статической заранее известно какие задачи будут выполняться на каком процессоре.

4.2 Метод коллективного решения

$$T_p = \frac{N}{p}(\tau_c + \tau_S)$$

$$T_p = min(\frac{N}{p}(\tau_c + \tau_S); \frac{\tau_c}{\tau_s})$$