МЕТРИКИ(СТАТИСТИЧЕСКИЙ ПОДХОД)

Определяются как результат многократного выполнения графа программы. Вершины графа – ассемблерные команды, дуги – отображают последовательность выполнения команд.

После трассировки в логе имеем последовательный поток команд и поток данных.

1. **Объем векторизованных вычислений на единицу данных.**

Средний объем вект. Вычислений, который приходится на единицу данных в приложении. Данная метрика вычисляет среднее число векторных операций, приходящихся на один доступ к данным.

Число векторных операций / число векторных операций чтения + записи

Число векторных операций = суммарное кол-во обращений к вычислит векторным операциям

Число вект, операций чтения и записи= суммарное кол-во обращений к вект. Операциям чтения + суммарное кол-во обращений к вект. Операциям записи

1. **Пространственная локализация данных**.

= суммарное кол-во обращений по одному адресу/ Количество команд чтения и записи

1. **Временная локализация данных.**

= количество уникальных адресов, которые запрашивается между двумя соседними запросами к адресу X, включая сам адрес X.

1. **Выравнивание данных.**

Количество команд выравнивания данных / Количество команд чтения и записи

ПИСЬМО ИЗ НИИСИ

Добрый день.

1. Какие векторные операции учитываются при вычислении среднего объема

векторных вычислений? Обычно под векторными вычислительными операциями

понимаются операции сложения и умножения. Именно они используются при

подсчете производительности в единицах Flops.

2. Пространственную локализацию имеет смысл оценивать с точностью до

размера строки кэша 32 байта. Также, интересно знать о количестве

следующих друг за другом обращений, т.е. о соседних адресах,

отличающихся на размер строки кэша.

3. Согласен.

4. Под выравниванием данных предлагаю понимать количество обращений по

адресам не выровненным на размер используемого вектора. Например, при

использовании вектора шириной 128 бит, выровненными считаются обращения

по адресам, кратным 0x10. Остальные признаются невыровненными.

Впрочем, ваша метрика тоже очень полезна интересна, как оценка затрат на

выравнивание данных. Другое дело, что архитектура MIPS MSA поддерживает

невыровенные операции, так что операций выравнивания может не быть в коде.

1. Соогласны.
2. Правильно ли понял- Рассматриваем кластеры адресов и попадание в них(

1 кластер- 0:32 байт, 2 кластер- 33:65 байт, 3 кластер- 66:98 байт,……).

соседние адреса, отличающиеся на размер строки кэша – это соседние кластеры?

1. Согласны
2. согласны

*Для того чтобы переменная, занимающая несколько байт памяти, была выровнена, необходимо, чтобы ее адрес был кратен ее размеру. Так, например, если объявленная в программе переменная имеет тип int32 (4 байта), то для того, чтобы она была выровнена, она должна быть размещена по адресу кратному 4.*

*Если процессор поддерживает невыровненный доступ, то может возникнуть ситуация, когда невыровненное данное находится в соседних словах. В этом случае процессору придется считывать из памяти соседние слова, в которых находится данное, а также произвести “склейку” этих частей. Таким образом, при доступе к невыровненным данным производятся дополнительные арифметические операции чтения данных. Невыровненный доступ порождает дополнительные кеш-промахи в случае, когда считываемая переменная хранится на границе кеш-линеек или на границе виртуальных страниц.*

