GRAPE-12(公開版)

牧野

2015/2/10 研究室セミナー

概要

- ここしばらく色々なしがらみもあって汎用メニーコアの「コデザイン」にかかわっていた。
- とてもとても大変で、こんな大変なことは早く止めたいと強く思った。
- 来年度にはなんか研究費とれるかもしれなので (特に根拠はない)、もうちょっと大変でないことを考えよう。

話の構成

- メニーコア MIMD マシンにおけるチューニングについて少しだけ (ほぼ愚痴)
- 問題の原因
- 仮想的な例
- ではどうするか？

メニーコアMIMDマシンにおけるチューニングについて少しだけ(ほぼ愚痴)

- ここのところ MIMD の機械を色々みたり聞いたりした
- チューニングはとても大変だし、大変なことをやった結果性能がでるとも限らない。

問題の例

- なんかよくわかんないけどどう頑張ってもピークがでない

メニーコアMIMDマシンにおけるチューニングについて少しだけ(ほぼ愚痴)

- ここのところ MIMD の機械を色々みたり聞いたりした
- ○○○○○○○ とか ○○○○○○
- チューニングはとても大変だし、大変なことをやった結果性能がでるとも限らない。

問題の例

- なんかよくわかんないけどどう頑張ってもピークがでない
- なんかよくわかんないけどどう頑張ってもピークがでない

メニーコアMIMDマシンにおけるチューニングについて少しだけ(ほぼ愚痴)

- ここのところ MIMD の機械を色々みたり聞いたりした
- ○○○○○○○ とか ○○○○○○
- チューニングはとても大変だし、大変なことをやった結果性能がでるとも限らない。

問題の例

- なんかよくわかんないけどどう頑張ってもピークがでない
- なんかよくわかんないけどどう頑張ってもピークがでない
- なんかよくわかんないけどどう頑張ってもピークがでない

メニーコアMIMDマシンにおけるチューニングについて少しだけ(ほぼ愚痴)

- ここのところ MIMD の機械を色々みたり聞いたりした
- ○○○○○○○ とか ○○○○○○
- チューニングはとても大変だし、大変なことをやった結果性能がでるとも限らない。

問題の例

- なんかよくわかんないけどどう頑張ってもピークがでない
- なんかよくわかんないけどどう頑張ってもピークがでない
- なんかよくわかんないけどどう頑張ってもピークがでない
- なんかよくわかんないけどどう頑張ってもピークがでない

メニーコアMIMDマシンにおけるチューニングについて少しだけ(ほぼ愚痴)

- ここのところ MIMD の機械を色々みたり聞いたりした
- ○○○○○○○ とか ○○○○○○
- チューニングはとても大変だし、大変なことをやった結果性能がでるとも限らない。

問題の例

- なんかよくわかんないけどどう頑張ってもピークがでない
- なんかよくわかんないけどどう頑張ってもピークがでない
- なんかよくわかんないけどどう頑張ってもピークがでない
- なんかよくわかんないけどどう頑張ってもピークがでない
- なんかよくわかんないけどどう頑張ってもピークがでない

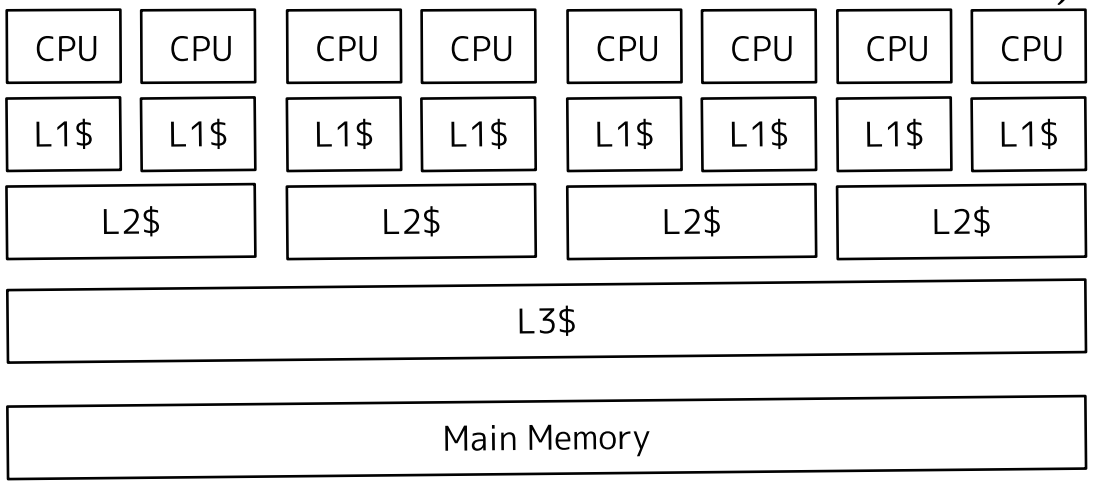
問題の原因

もちろん「よくわからない」わけだが、、、
非常に色々あり、簡単にはいえない。
例:

- 最内側ループが短かすぎて、オーバーヘッドが大きい
- 最内側ループが長すぎて、命令キャッシュにはいらない
- L1D のレジスタへのバンド幅がそもそも足りない
- L1D と L2D の間のバンド幅が不足する
- コア間同期に時間がかかりすぎる
- スレッド間同期に時間がかかりすぎる
- 共有キャッシュにアクセスが集中するとなにかが起こる？

仮想的な例

(実マシンじゃないということで御了解を)



階層キャッシュの機械を考える。

- (図とは違うけど) 16 コアが L2 共有、さらに 16 グループが L3 共有としよう。面倒なのでコア内 SIMD は考えない。コア内 4 スレッド
- これに行列乗算させてみる。行列サイズはどの方向も十分大きいとする。

「理論的に」最適なアルゴリズム

- 「最適」は、メインメモリの読み書きが最小になるという意味とする。
- $C = AB$ とすると、A をキャッシュ全体を使って載るサイズぎりぎりの部分正方行列にわけて、階層キャッシュの一番合計容量の多いところに分散して格納し、その部分行列と掛け算する B の部分行列を 1 列 (よりももうちょっと多い必要ある) づつ処理
- 問題:
  - A の分散のさせかた
  - これで性能でるかどうか

単純な方法とその問題点

- A(の部分行列) をコア数だけ横に分割。例えば 1024x1024 なら 1024x4 に
- これを長さ 1024 の縦ベクトル b と乗算

問題点

- 同じ b を全コアが必要とする
- これはキャッシュに大きな負荷を掛ける。同じところをつながっている全コアがアクセスする。違うところをアクセスするより遅くなる可能性大
- SMT だとスレッド数のほうが行列の行数よりも大きいかも。スレッド間で A, b を分割する必要が発生

キャッシュに対して優しい方法とその問題点

- A をさらに 16x16 に分割。1 列を共有 L2 のグループが担当
- b も L2 以降は 16 分割されるので、L2 から上のアクセスは減る。

問題点

- グループ間で総和をとる必要がある。L3 にデータ書き込んで合計すると、結局 L3 のアクセスが増える
- 総和ではレイテンシも問題になり、なかなか性能でない

実際のところ、、、

- 某社某プロセッサでは、A をもうちょっと小さくして重複して持つほうがよかった模様。それで現在実行効率は××(伏せ字)
- Haswell ではまだ DGEMM ピークの 8 割くらいしかでてない？
- Kepler DGEMM はそれでも 93% までいったらしい。Fermi が××だったのでそこはまあ考えて作ったと思われる。
- ちなみに、GRAPE-DR はメモリバンド幅的にも内部実装的にも最適なアルゴリズムが動く。ブロック間の総和がパイプライン的かつ演算コアと並列に動作するので、DGEMM で理論ピークの 97% を実現している。

何がいいたいかというと

- 階層キャッシュの MIMD って、単純なことをやらせるだけでも性能出すのはものすごく大変
- たかが DGEMM でこんな話で、他のもっと複雑な計算だと、、、
- 個別のアプリケーション・アーキテクチャに対してなんとかしようとするのは貴重な人的資源の浪費。

何がいいたいかというと

- 階層キャッシュの MIMD って、単純なことをやらせるだけでも性能出すのはものすごく大変
- たかが DGEMM でこんな話で、他のもっと複雑な計算だと、、、
- 個別のアプリケーション・アーキテクチャに対してなんとかしようとするのは貴重な人的資源の浪費。

ではどうするか

- 階層キャッシュと MIMD メニーコアという、予測のきかないものを組み合わせた上で予測できる性能をだす、というアプローチ自体が無理
- ほぼフラットなメモリ、SIMD メニーコア、コアと並行動作するコア間およびコア・メモリ間ネットワークで机上で精度の高い性能予測を可能にすると話は簡単
- それって GRAPE-DR とか加速部とかのこと？
- もちろんそう。

GRAPE-DR (及び加速部) の「問題点」

- GRAPE-DR は、チップ内ネットワークに制限が多く、チップ間ネットワークはそもそも考慮されてなくて適用範囲が狭いという問題はあった
- 加速部はそれは改善した (つもりである) が、電力性能が「圧倒的に高くはない」という問題がある
- これは GRAPE-DR にもある問題

電力性能 (大体の数字)

GRAPE-6	250nm	3GF/W
GRAPE-DR	90nm	4GF/W
GRAPE-X	28nm	25GF/W
Fermi	40nm	2GF/W?
Kepler	28nm	5GF/W?
AMD Hawaii	28nm	11GF/W?

- GPU と 2 倍ちょっとしか違わない。
- 競争力をもてる期間が短い (または「ない」)。

理論的限界はどのへんか

- 「倍精度演算器だけの消費電力」が理論限界
- これは多分 GRAPE-X の 3-4 倍。動作電圧とかであと 2 倍くらいは稼ぐ余地あり
- 電力性能を 10 倍あげられれば 10 年寿命が延びる。5 倍なら 7 年。

どうやって現在の GRAPE-DR/X からさらに3倍稼ぐか

- 演算以外の「すべて」を半減とかもっと減らす必要がある
- 単純なのは、演算器だけを倍に増やして、行列乗算だけはできるように何か考えること
- もっと違う考え方もあるかも。

DGEMM以外は？

- もちろん、他のアプリケーションのほうが重要。
- 粒子系、不規則格子、その他
- 粒子系は基本的に距離計算＋関数評価＋ウェイトかけて総和だとすれば、なんかそういう回路をつけてもいいのかも
- 不規則格子ももうメッシュレスで、、、

GRAPE-12

- TSMC の 28ULP でDGEMM 性能チップレベル 70GF/W を目標にする
- 粒子間相互作用とかならさらに倍以上 (単純に単精度ピークを 2 倍にするだけでは無理) を目標にする
- 予算的に 16ULP とか使えればさらに 2 倍を目指す
- 話としては16ULPでシステムレベルで 80GF/W、GPU で 5nm のものより上を目標。
- 詳細はこれから、、、