

課題名 (タイトル) :

FX100 を用いたチェックスイート性能評価

利用者氏名 :

○南 一生*, 黒田 明義*, 熊畑 清*, 宮本 健吾*

理研での所属研究室名 :

*理化学研究所 計算科学研究機構 運用技術部門 チューニング技術チーム

1. 本課題の研究の背景、目的、関係するプロジェクトとの関係

本課題では、スーパーコンピュータ「京」(以下「京」と記す) 開発時に作成したチェックスイートやサンプルアプリケーションコードを用いて、FX100 での性能解析を行い、システム性能の評価、チューニング効果の評価、チェックスイートの問題点の洗い出し、新たなチェックスイートの試作などを行うことを目的に利用を行った。

ここで用いたチェックスイートとは、HPC で利用される計算科学アプリケーションから、主要な計算箇所をループ単位の粒度で切り出したものである。切り出しに用いたアプリケーションは、各分野から、アルゴリズム特性に応じて幅広く選択されている。また用途に応じて、単体チェックスイート、並列チェックスイートに分類出来る。

チェックスイートは、「京」開発当初、システム評価並びにアプリケーションのチューニング指針を得ることを目的に構築された。「京」の運用では、システム(OS, ミドルウェア等)のエンハンス/メンテナンスに伴うアプリケーションの性能劣化を「未然に防止する」「検知する」ことを目的とし、以下の機能整備を行った。

- (1) 複数システム(マシン, コンパイラ)動作
- (2) 計算結果のバリデーションチェック機能
- (3) 性能検証の比較結果レポート機能
- (4) 一括実行機能
- (5) ジョブ多重度管理機能
- (6) 評価カーネル選択機能

現在開発が進められているポスト「京」のスーパーコンピュータ開発では、最新システムで測定

された性能をもとに次世代システムの性能評価を行う。このため、現段階で最新のシステムであるFX100を使用した性能評価が重要である。本課題では、現状のチェックスイートを用いて、性能評価を行い、そこから問題点を洗い出し、チェックスイートの再整備を行った。更に実運用で発生した問題の解決に向けたテスト計算を実施した。

2. 具体的な利用内容、計算方法

(1) NTChem チュートリアルサンプルの調査

NTChem は、理化学研究所計算科学研究機構にて開発された分子科学計算ソフトウェアである。課題を実施しているチューニング技術チームでは、本年度からNTChemを含むAICSソフトウェアの開発サポートやユーザビリティの向上を目的とした調査をすべく、「京」並びにHOKUSAI GWMPCCを使用してチュートリアルサンプル問題のテスト実行を試みた。

今回、調査を行ったサンプルは、Hands-On の資料に記載されているインタラクティブなインプット作成補助プログラム ntprep を用いて作成した、水分子の計算である。Hands-On では、水 1 分子のデータを作成するが、ここでは、コスト解析のために 1,2,4,8 分子パターンを作成し、水分子の DFT 一点計算を行った。

(2) FrontFlow/blue カーネルの評価

FrontFlow/blue(以降 FFB)は東京大学革新的シミュレーション研究センターで開発された汎用熱流体シミュレーションコードである。

本アプリケーションはものづくりの現場で実用に供されており、今年度も引き続き幾つかの主要演算部分(カーネルと呼称)に着眼して「京」と比し

での性能評価ならびに高度化の検討を行った。

(3) ADV-K カーネルの評価

ADV-K は、東洋大学を中心に開発中の構造解析プログラムである。HOKUSAI GWMPCC とは別の FX100 システム上において、1 ノード当たり 2 プロセスの並列実行状態での、ADV-K カーネルの PA 情報を採取しようとする、Segmentation fault が発生し、PA 情報を採取することができないという問題があった。この問題が、ユーザプログラムの問題かシステムの問題かを切り分けるため、HOKUSAI GWMPCC でも同様に Segmentation fault が発生するか否かの確認試験を行った。

3. 結果

(1) NTChem チュートリアルサンプルの調査

生成した水分子のチュートリアルサンプルを「京」で実行したところ、MPI 並列数 2proc.で、10[s]程度で終了する問題となっており、リーズナブルな問題設定であった。但し、自動生成されたスクリプトを見ると、内部で 7 つのモジュールが組み合わされて実行されており、負荷の大きい箇所を見いだすことが困難であった。このため水分子を増やして実行を行うと、DFT を用いた SCF 計算を行うロードモジュールの負荷が問題サイズに応じて大きくなることが分かり、この区間の性能を取得した。水分子を増やした時の本区間のプロファイル情報は以下の通りである。

表 「京」で測定した主要モジュールの性能

#mol	#node	Time[s]	MFLOPS	FLOPS%	sched_yield
8	12	713.4	12483	0.929%	85.2%
4	12	163.1	10025	0.746%	85.1%
2	12	6.970	2577.7	0.192%	77.9%
1	12	2.622	923.7	0.069%	72.3%

実行時間のぶれが大きい、水分子を増やすことで、実行時間がおおよそ $O(N^3)$ で増加している。しかし 90%近い時間がスレッドの受け渡しを担う

システム処理で占められており、この問題を確認するため、FX100 でも同様の測定を行った。本アプリケーションは、バイナリ配布されているため、HOKUSAI GWMPCC で計測する際には、「京」用に提供されたロードモジュールを使用した。主な測定は、6nodes, 12processes にて行った。

表 HOKUSAI GWMPCC での主要モジュールの性能

#mol	#node	Time[s]	MFLOPS	FLOPS%	sched_yield
8	12	1672	9715.0	0.171%	82.7%
4	12	120.9	5228.1	0.092%	93.3%
2	12	25.07	1203.0	0.021%	92.3%
1	12	21.17	277.79	0.005%	92.6%
1	2	4.201	657.91	0.069%	83.6%

性能は、HOKUSAI GWMPCC でも同様の傾向が見られた。スレッドの受け渡しのシステム処理時間が全体的に「京」と比べて増大しており、同じプロセス数でも実行時間が多めに出ていることがわかる。測定には「京」用のバイナリを用いたが、プロファイルを見ると 16(-1)threads で動作しており、想定した thread 並列数で動いていることを確認した。

(2) FrontFlow/blue カーネルの評価

本年度は、①コンパイラによる自動スレッド並列化がうまく働いていなかったいくつかのカーネルの改善、ならびに②if 分を含むループの性能改善の検討を行った。

①についてであるが、FFB では OpenMP などには用いず、コンパイラによる自動スレッド並列化(及び、一部については OCL 指示子による自動並列化の促進)によりスレッド並列を行っている。しかし、一部のカーネルでは、コンパイラがデータ依存性を見切れず、スレッド並列化が適応されないケースや、回転数が著しく少ないループにスレッド並列化が適応され、実行時には回転数不足により逐次で実行されるといったトラブルが起きていた。ここではループ構造書き換えと OCL 指示子の利用により各ループが適切にスレッド並列化されるよう修正した。ループ構造書き換えによりメ

モリアクセス効率が向上したこともあり、スレッド数倍以上の性能向上を得た。

②については、特定の要素にだけ追加の演算を行い、値を修正するという処理を行うループが、テストで用いている入力データでは全ての要素で修正演算は不要であるにも関わらず多大な時間を消費していたという問題についての対処である。当初の実装では、全要素についてループを回しループ内で if 文により修正演算の必要性を判定するという構造であった。要素ごとの修正演算の必要性はプログラム実行を通じて不変であるため、予め修正演算を必要とする物だけ集めておき、修正演算が必要となるものにだけループを回すというリストベクトル変換を行うように修正し、テストで用いている入力データでは実行時間が無視できる程度にまで改善した。

(3) ADV-K カーネルの評価

HOKUSAI GWMPIC 上において、1 ノード当たり 2 プロセスの並列実行状態での、ADV-K カーネルの PA 情報の採取を試みた。その結果、HOKUSAI GWMPIC 上でも、他の FX100 システム上と同様に、Segmentation fault が発生し、PA 情報を採取することができないという問題が発生した

4. まとめ

「京」で構築されたチェックスイートやテストコード、テストサンプルを用いて、HOKUSAI GWMPIC での評価を行った。これらの解析を詳細に進めることで、今回新たにチェックスイート側の課題がいくつか顕在化した。FrontFlow/blue のカーネルでの結果はその代表である。また AICS ソフトウェアのサンプルプログラムを動かすことで、性能の問題点の切り出しができ、今後新たなカーネルの整備に役立てることができた。また ADV-K カーネルを用いた評価により、システムで発生した問題の確認にも HOKUSAI GWMPIC を用いて検証することができた。

今回の課題実施によって得られた知見をもとに、更なる詳細の解析を行うとともに、より汎用かつ有用なチェックスイートの整備に力を入れる予定である。

6. 利用がなかった場合の理由

該当なし

5. 今後の計画・展望