準粒子乱雑位相近似の GPU コードの開発
Development of GPU code of quasiparticle random-phase approximation

寺崎 順
チェコ工科大学プラハ 実験応用物理学研究所

１．研究目的

今まで課題実施者が用いてきた QRPA ハミルトニアン行列要素を計算するプログラム(Fortran90+MPI)を、OpenACC を用いて GPU 並列計算ができるようにする。

２．研究成果の内容

QRPA ハミルトニアン行列要素数は、10^7 から 10^8 個であり、これらは独立に計算できるので、元プログラムでは、この独立計算を行うループを分割して、MPI により並列計算している。今回は、二つの方法で OpenACC のこのプログラムへの実装を行った。

一つの方法（方法１）は、一 MPI プロセスが担当する行列要素計算ループを OpenACC で並列化するやり方である。このループはファイルへの write を含むため、担当ループ数をいくつかに分離し、このグループを制御するもう一つのループを外郭に設けた。この内郭が元ループであり、ここを OpenACC で並列化する。ただし、write は、この内郭ループの外に移動する。計算した行列要素は配列に格納され、この内郭ループ終了後、行列要素はファイルに書き出される。ここが外郭ループの終点で、次の外郭ループにより OpenACC を用いた行列要素計算が繰り返される。

もう一つの方法（方法 2）は、行列要素計算内の随所で行われる積分計算を OpenACC で並列化するやり方である。外郭ループを設ける必要はない。このやり方では、一つの OpenACC ループ内の計算量は方法 1 よりずっと少なく、OpenACC 並列領域の数は多い。方法 1 では、これらの積分計算はデバイスの一スレッドで行われる。

Cygnus の一ノードは四枚の GPU をもっている。実行時には、一ノード四プロセスで、四 GPU を用いるようにした。つまり一プロセスが一 GPU を占有する。一ノード中の二十個の CPU コアは計算に用いられない。一ノードで、約二万個の GPU コアが用いられるので、CPU コアの非効率性は計算全体の中では、無視できるという考えである。テストサンプルには、^{136}Xe の陽子・中性子準粒子 QRPA 計算を用いた。二ノードを用いて、全行列要素の百分の一を計算し、要した計算時間を表 1 に示す。方法 2 で一つの積分計算の OpenACC ループ数は約 3500 であり、計算に実際に用いられるデバイススレッド数は方法 1 より少ない。このことを考慮しても、方法 1 の方が方法 2 よりはるかに速い。なるべく多くの計算を含み、ループ数の多い外側のループをデバイス並列化するのが、この計算ではよいということがわかった。これは、OpenACC 並列領域の数が少ない方がよいということでもある。

	方法 1	方法 2
推定計算時間（時間）	0.275	5.92

表 1。部分的テスト計算における、MPI+OpenACC を実装したプログラムによる計算時間。5120×8 個の GPU コアが占有された。方法 1 と 2 の違いは OpenACC の実装方法にある。これについては本文を参照のこと。

比較のため、京と Oakforest-PACS で以前行った同じ入力による全行列要素計算時間を表 2 に示す。京では、80000 コア、Oakforest-PACS では、34816 コアが用いられた。これら二機の計算効率は Cygnus に比べると同じ範疇にある。Cygnus 計算では、5120×8 個の GPU コアが用いられたので、同じコア数の計算の比較を考えると、CPU 機の方が、Cygnus(方法 1)よりもおよそ因子 80 ほど効率がよい。

	Cygnus (方法 1)	京	Oakforest-PACS
コア数	40960	80000	34816
計算時間（時間）	70	0.43	1.72

表 2。Cygnus, 京および Oakforest-PACS での全行列要素の計算時間。いずれも Fortran90+MPI のプログラムで行われた。

３．今後の展望

上記のように同じコア数で比較すると GPU 機は CPU 機よりも計算効率が劣る。そこで、GPU 機の利点は何かという問題を考えると、ユーザにとっては最終的な仕事の生産性が重要であるという観点から考察の余地がある。表 3 には、二計算機での全行列要素計算に要する消費割当量と、全ユーザに対する年間総計算割当量(京については不明)を示す。

	Cygnus	Oakforest-PACS
行列要素計算（コア時間）	2800000	35000
計算機の総割当量（コア時間）	8×10^9	5×10^7

表 3. 二計算機の行列要素計算の消費割当量(Cygnus では GPU コア時間)と、それぞれの計算機の全ユーザに対する年間総割当量。後者は異なる計算機利用プログラムのものを含む。

Cygnus は Oakforest-PACS に比べ、約 80 倍のコア時間を要するが、提供される総コア時間は 160 倍である。従って、Oakforest-PACS に比べ 80 倍以上の計算割当量を確保することは可能と思われる。この十分な割当量により、一年間に Cygnus が Oakforest-PACS より多くの計算を行うことは可能である。ここが Cygnus の特筆すべ

き利点であり、用いる意義は十分にある。

昨今 GPU 機は増える傾向にある。計算機の価格や消費電力がこの傾向を後押ししているとすれば、後退することはないであろう。そこで、GPU 機で実行できるプログラムをもつことは、利用できる計算機を増やすことになり、研究の生産性向上にとって好ましいといえる。

４．学際共同利用が果たした役割と意義

高性能並列計算機の利用プログラムでは、生産的計算の開始準備が整っていることが利用条件であることが少なくない。学際共同利用では、プログラム開発のための利用が認められるので、今回の OpenACC 実装の過程で、課題実施者が参照した文献では得られない重要な技術的情報を得ることができた。すなわち、

1. デバイス並列領域内では、ファイルにアクセスできない。これは、デバイスの特性から理解できる。
2. デバイス並列領域内で定義されたプライベート配列は、そこで用いられるプロシジャに引数を通じて正しく渡すことができない。デバイス並列領域内でのプライベート配列に対するメモリー上限がきついのかもしれない。
3. デバイス並列領域内で用いられるプロシジャにモジュール経由で配列を渡すことはできない。従って、この領域内でのプロシジャに配列を渡すためには、そのプロシジャの実引数に共有配列を充てるしかない。

Cygnus のマニュアルに、計算結果が正しくないことがある、との注意書きがあり、ある段階のテストでこの問題に遭遇した。マニュアル中で指示された方法によりこの問題は回避された。

行列要素計算のデバイス並列領域は asynch してあり、ホストの行列要素書き出しと並列に行われる（最初と最後の外郭ループを除く）。実際には、書き出し時間は、行列要素計算時間に比べ無視できるほど小さかった。

５．成果発表

(1) 学術論文、

1.J. Terasaki,
“Modification of Nuclear Matrix Elements of Neutrinoless Double-β decay”
AIP Conf. Proc., to be published.

2. J. Terasaki,
“Cause of discrepancy problem of calculated running sums to nuclear matrix element of two-neutrino double-β decay”

submitted to Phys. Rev. C

(2) 学会発表、
J. Terasaki,
"Modification of Nuclear Matrix Elements of Neutrinoless Double-β decay",
Workshop on Calculation of Double-beta-decay Matrix Elements (MEDEX'22),
Prague, Czech Republic, Jun. 13–17, 2022.

(3) その他、なし。

使用計算機	使用計算機に○	配分リソース※	
		当初配分	追加配分
Cygnus	○	160	0
Oakforest-PACS			
※配分リソースについてはノード時間積をご記入ください。			