GTC Workshop Japan 2011（2011年7月22日）

通信コードを自動生成可能なマルチＧＰＵ向けビジュアルプログラミング環境

Visual Programming Environment with Automatic Network Code Generation for Multi-GPUs

村瀬 正名[1)]，前田 久美子[1)]，土居 意弘[1)]，小松 秀昭[1)]，野田 茂穂[2)]，姫野 龍太郎[2)],
Masana Murase, Kumiko Maeda, Munehiro Doi, Hideaki Komatsu, Shigeho Noda, and Ryutaro Himeno

1) 日本アイ・ビー・エム（株）（〒242-8502 神奈川県大和市下鶴間1623-14）
2) （独）理化学研究所　情報基盤センター　（〒351-0198 埼玉県和光市広沢2-1）

Key Words : *Multi-GPUs, Network Code Generation, Visual Programming*

1. はじめに

High Performance Computing (HPC)をはじめ，金融，医療など様々な分野のアプリケーション性能を単一または複数のGPUを利用して高速化する動きが加速している．

しかしながら，複数のGPUノードを利用したアプリケーション開発では，単一GPU向けのカーネル開発に加えて，(1) ノード間の通信，(2) ホストメモリ・GPUメモリ間の通信の両通信コードを実装する必要がある．本稿では，これら通信コードの自動生成を実現するデータフロースタイルのプログラミング環境を提案する．ステンシルアプリケーション（姫野ベンチマーク[1]）を例にマルチGPU環境における通信コードの自動生成を試みる．

2. マルチGPU向けビジュアルプログラミング

我々はこれまでC, C++, Fortran, C for CUDAなど既存のプログラミング言語を横断的に扱えるビジュアルプログラミング環境を提案してきた[2]．我々が開発中のビジュアルプログラミング環境では，既存のプログラミング言語を用いてライブラリ作成者がコンポーネントを実装し（図1），エンドユーザはこれらコンポーネントを連結させ，データフローを作成することでアプリケーションロジックを表現する．

```
/* CUDA kernel definition */
/// KERNEL cu_jacobi
/// + float [][][] in __DEVICE__
/// -float [][][] out __DEVICE__
/// -float gosa
/// > HIMENO_JACOBI
/// ARCH x86
/// ACCEL cuda
void cu_jacobi ( float *in , float *out ,
float * gosa ) {
```

図1　コンポーネントプログラミング例

3. 通信コード自動生成

本稿では，これまでサポートされてきた通信コード(socket, メモリコピー，ポインタ渡し)に加えて，マルチGPU向けの通信コード生成拡張を行う．通信コードの自動生成において，我々のコンパイラはデータフローグラフ上の明示的パスの解釈，ポートの属性（__DEVICE__など），並列数および通信テンプレートの解釈を行うことで対象アプリケーションの並列化に必要な通信パスを特定する．生成される通信コードはノード間の場合にはMPI, GPUとホストや同一GPU内の場合にはcudaMemcpyが利用される．

4. コード生成例

姫野ベンチマークを用いてコード生成を行った．本実験において新たに姫野ベンチマークをGPUコンポーネントとして設計し，実装した．表1に本コンポーネントの行数を示すと同時に，本環境において通信用バッファの管理コード，通信コードを含む自動生成されたコード行数も併記する．

表1　コンポーネントおよび自動生成コード行数

コンポーネントコード（アノテーション+ロジック）		自動生成されたコード（4並列GPU版）
221	10 (アノテーション)	6,225
	43(本環境用追加)	
	168 (CUDAカーネル)	

5. むすび

本稿では，マルチGPU環境向け並列アプリケーション開発において新たに必要となる二つの通信プログラミング，MPIとPCIe通信の自動生成について，姫野ベンチマークを利用して生成を試み，並列度を変えた場合でも自動生成されることを示した．

参考文献

[1] 姫野ベンチ: http://accc.riken.jp/HPC/HimenoBMT.html
[2] Murase, M. et al.: A Parallel Programming Framework Orchestrating Multiple Languages and Architectures. In Proceedings of ACM CF'11.