2010 年度 未踏 IT 人材発掘・育成事業 採択案件評価書

1. 担当PM

増井 俊之 PM（慶應義塾大学 環境情報学部 教授）

2. 採択者氏名

チーフクリエータ： 盧 承鐸（東京大学大学院）
コクリエータ： なし

3. 委託金支払額

1,210,000 円

4. テーマ名

大規模配列の GPU 高速処理プログラミングを支援する開発環境

5. 関連Webサイト

なし

6. テーマ概要

本システムは、大規模配列の GPU 高速処理プログラムの開発を支援する、ビジュアルプログラミング環境を提供する。

この環境でユーザは、従来の GPU プログラミングのようにプログラムのコードを直接書くわけではなく、メモリ上の配列を可視化させた「図」と、その「図」に適用される「離散化式」を修正することによってプログラミングを行う。

本環境において各々の「図」は、計算において処理を行う配列を意味しており、この「図」の上の各「ピクセル」は配列の個別の要素を意味している。この「ピクセル」に、

適用される計算スキームの種類のメタファーである「色」を塗った上、各「色」に適用される計算スキームを「離散化式」で書くことによってプログラミングを行う。

システムの特長点として、GPU プログラミングに慣れていない者が苦労する、並列処理部分のプログラムをシステム側が自動的に最適なパフォーマンスが得られるようにチューニングして出力する。これによって、ユーザは GPU の並列処理を意識せず、メモリアクセスや離散化式、つまり計算アルゴリズムにのみ集中することができ、従来の GPU プログラミングより高い生産性で開発をすることができる。

また、実際ユーザが操作する部分はプログラムコードの形でなく、適用される計算スキームとメモリの配置状況を示している。そのため計算機の振る舞いをソースコードで表現しなければならない一般的な配列の計算プログラムでは、符号や添字を間違えるバグを多く生む可能性がある。提案するシステムではそのような間違えを未然に防ぐことができる。

7. 採択理由

近年 GPGPU を用いた高速計算が脚光をあびてきており、様々なシステムで利用されるようになってきているが、提案者の盧君が卒論で CUDA を使った GPGPU プログラミングを行なったところ、まだまだプログラミングが大変だということを痛感したということであった。メモリのブロックを適切に分割する方法やデバッギングなど、GPGPU のプログラミングの難しいところを視覚化によって解決するというのが盧君の提案である。

複雑なプログラミングを簡単に理解するために、これまで様々な「ビジュアルプログラミング」システムが提案されてきており、非専門家向けのプログラミングシステムや音楽のプログラミングなど、広い分野で活用されてきている。

このような手法を GPGPU の並列プログラミングに適用するという考えは自然なものだが、きちんと実装するためには幅広い知識とプログラミングの実力が必要であり、かなりチャレンジングな提案だといえる。盧君はプログラミングコンテストでの受賞経験を持つなどプログラミングの基礎能力が高く、ビジュアルプログラミングと GPGPU プログラミングの両方に精通しているためこのような提案を実現できる可能性が高いと思われる。

誰でも GPGPU プログラミングに挑戦できるようなシステムの開発をめざしてもらいたいと思う。

8. 開発目標

GPU について詳しくないユーザがグラフィカルに GPU プログラミングを行なうためので

きる環境を構築する。プログラミングで必要になる各種の式もグラフィカルに編集できるようにする。

9. 進捗概要

計算したい数式を画面上の GUI で編集することによってビジュアルに自動的に GPGPU プログラミングを行なえるようにしようというのが最初の提案であったが、作成が非常に難しいこと及び適用範囲がかなり限られてしまうことが明らかになったためこの方針は中止し、ビジュアリゼーションやデバッグをサポートする「GPGPU プログラミング支援システム」を作成するという方針に変更した。ビジュアル画面上での編集を行なうことはやめ、結果の表示だけをビジュアルに行なうことにした。

10. プロジェクト評価

前述の方針変更の結果、全くの初心者向けのシステムの構築には失敗したが、C プログラミングに慣れた普通のプログラマにとって GPU の入門となるシステムを作成することには成功した。GPU プログラミングに関する基本的な知識があるユーザであれば、開発されたシステムを用いてビジュアルに GPU 動作のシミュレーションやデバッグを行なうことができる。初期の目標から比べると若干物足りないものになってしまったが、技術的には高度なものが開発できたことは評価したい。

11. 今後の課題

GPGPU プログラミングの難しさのひとつであるデバッグやシミュレーションの問題を解決する方法を本プロジェクトである程度示すことができたため、当初の目標であった総合的なビジュアル開発環境の構築が課題となる。