PGI プロファイラユーザーガイド

プロファイリングの概要
このドキュメントでは、CUDA、OpenACC、または OpenMP アプリケーションのパフォーマンスを理解して最適化できるようにする PGI プロファイリングツールについて説明します。PGPROF のビジュアルモードには、アプリケーションの CPU および GPU アクティビティのタイムラインが表示され、最適化の機会を特定する自動分析エンジンが含まれています。 PGPROF は、ビジュアルモード（GUI）とコマンドラインモードの両方で使用できます。引数なしでコマンド pgprof を発行すると、PGPROF が視覚的に起動します。それ以外の場合、PGPROF はコマンドラインモードで動作します。

ビジュアル PGI プロファイラと PGI プロファイラは、将来の CUDA リリースで非推奨になることに注意してください。GPU プロファイリングには次世代ツール NVIDIA Nsight Compute を、GPU および CPU のサンプリングとトレースには NVIDIA Nsight Systems を使用することをお勧めします。

NVIDIA Nsight Compute は、CUDA アプリケーション用のインタラクティブなカーネルプロファイラです。ユーザーインターフェイスとコマンドラインツールを介して、詳細なパフォーマンスメトリックと API デバッグを提供します。さらに、そのベースライン機能により、ユーザーはツール内で結果を比較できます。 Nsight Compute は、カスタマイズ可能でデータ駆動型のユーザーインターフェイスとメトリックコレクションを提供し、後処理結果の分析スクリプトで拡張できます。

NVIDIA Nsight Systems は、アプリケーションのアルゴリズムを視覚化するために設計されたシステム全体のパフォーマンス分析ツールであり、最適化する最大の機会を特定し、任意の数量またはサイズの CPU および GPU にわたって効率的にスケーリングするための調整を支援します。大規模サーバーから最小の SoC まで。

ブログの投稿 Visual Profiler と nvprof から Nsight Tools への移行、Visual Profiler と nvprof から Nsight Systems への移行、および Nsight Compute を使用したカーネルの検査では、開発を次世代ツールに移行する方法について説明しています。

https://devblogs.nvidia.com/migrating-nvidia-nsight-tools-nvvp-nvprof/
https://devblogs.nvidia.com/transitioning-nsight-systems-nvidia-visual-profiler-nvprof/
https://devblogs.nvidia.com/using-nsight-compute-to-inspect-your-kernels/

新着情報
プロファイリングツールには、このリリースの一部として以下の変更が含まれています。

* ビジュアル PGI プロファイラと PGI プロファイラを使用すると、デスクトッププラットフォームの非 root ユーザーと非管理ユーザーのトレース機能を使用できます。イベントとメトリックのプロファイリングは、非 root ユーザーと非管理者ユーザーに対して引き続き制限されていることに注意してください。問題と解決策の詳細については、この Web ページを参照してください。
* ビジュアル PGI プロファイラと PGI プロファイラにより、仮想 GPU（vGPU）のトレース機能が可能になります。
* プロファイラは、アプリケーションをトレースするときに CUDA グラフのパフォーマンス特性をオフにしなくなりました。
* ビジュアル PGI プロファイラで OpenMP プロファイリングを有効/無効にするオプションが追加されました。
* 非同期の cuMemset / cudaMemset アクティビティの不正なタイミングの問題を修正しました。

用語

イベントは、デバイスでのカウント可能なアクティビティ、アクション、または発生です。カーネルの実行中に収集される単一のハードウェアカウンター値に対応します。特定の NVIDIA GPU で利用可能なすべてのイベントのリストを表示するには、pgprof --query-events と入力します。

メトリックは、1 つ以上のイベント値から計算されるアプリケーションの特性です。特定の NVIDIA GPU で利用可能なすべてのメトリックのリストを表示するには、pgprof --query-metrics と入力します。メトリックリファレンスを参照することもできます。

第 1 章 プロファイリングのためのアプリケーションの準備

CUDA プロファイリングツールでは、プロファイリングを有効にするためにアプリケーションを変更する必要はありませんただし、簡単な変更や追加を行うことで、使いやすさと効果のプロファイリングを大幅に向上させることができる
このセクションでは、これらの変更と、プロファイリング結果を改善する方法について説明します

1.1 集中的なプロファイリング

デフォルトでは、プロファイリングツールは、アプリケーションの実行全体にわたってプロファイルデータを収集します
ただし、以下で説明するように、通常は、性能が重要なコードの一部またはすべてを含むアプリケーションの領域のみをプロファイルする必要がある
プロファイリングを性能が重要な領域に制限すると、ユーザーとツールの両方が処理する必要のあるプロファイルデータの量が減り、最適化によって性能が最大になるコードに注目が集まります

アプリケーションの領域のプロファイリングが役立つ一般的な状況がいくつかあります

1.アプリケーションは、アルゴリズムのすべてまたは一部の CUDA 実装を含むテストハーネスです
テストハーネスはデータを初期化し、CUDA 関数を呼び出してアルゴリズムを実行し、結果が正しいかどうかを確認します
テストハーネスの使用は、アルゴリズムの変更をすばやく繰り返してテストするための一般的で生産的な方法です
プロファイリング時に、アルゴリズムを実装する CUDA 関数のプロファイルデータを収集しますが、データを初期化したり結果を確認するテストハーネスコードは収集しない

2.アプリケーションは段階的に動作し、各段階で異なるアルゴリズムのセットがアクティブになります
アプリケーションの各フェーズの性能を他のフェーズとは無関係に最適化できる場合、各フェーズを個別にプロファイルして、最適化の取り組みに集中できる
3.アプリケーションには多数の反復で動作するアルゴリズムが含まれていますが、アルゴリズムの性能はそれらの反復間で大幅に変化しない
この場合、反復のサブセットからプロファイルデータを収集できる

プロファイリングをアプリケーションの領域に制限するために、CUDA はプロファイルデータ収集を開始および停止する機能を提供する

cudaProfilerStart()を使用してプロファイリングを開始し、
cudaProfilerStop()を使用してプロファイリングを停止します
(CUDA ドライバーAPI を使用すると、cuProfilerStart()および cuProfilerStop()で同じ機能が得られる)
これらの関数を使用するには、cuda_profiler_api.h(またはドライバーAPI の場合は cudaProfiler.h)を含める必要がある
開始および停止機能を使用する場合は、アプリケーションの開始時にプロファイリングを無効にするようにプロファイリングツールに指示する必要もあります
nvprof の場合は、-profile-from-start off フラグを使用する
ビジュアルプロファイラの場合は、 Settings View の'Start execution with profiling enable'チェックボックスを使用する

1.2 CPU アクティビティの領域のマーキング
ビジュアルプロファイラは、アプリケーションによって行われた CUDA 関数呼び出しのトレースを収集できる
ビジュアルプロファイラはこれらの呼び出しをタイムラインビューに表示し、アプリケーションの各 CPU スレッドが CUDA 関数を呼び出している場所を確認できる
アプリケーションの CPU スレッドが CUDA 関数呼び出し以外で何を行っているかを理解するには、NVIDIA Tools Extension API(NVTX)を使用する
アプリケーションに NVTX マーカーと範囲を追加すると、CPU スレッドがそれらの領域内でいつ実行されているかがタイムラインビューに表示される
nvprof は、NVTX マーカーと範囲もサポートする
マーカーと範囲は、タイムラインの API トレース出力に表示される
要約モードでは、各範囲は、その範囲に関連付けられた CUDA アクティビティとともに表示される

1.3 CPU および CUDA リソースの命名
ビジュアルプロファイラのタイムラインビューには、CPU スレッドと GPU デバイス、コンテキスト、およびストリームのデフォルトの名前が表示される
これらのリソースにカスタム名を使用すると、特に多くのホストスレッド、デバイス、コンテキスト、またはストリームを持つ CUDA アプリケーションの場合、アプリケーションの動作の理解が向上する
NVIDIA Tools Extension API を使用して、CPU および GPU リソースにカスタム名を割り当てることができる
カスタム名がタイムラインビューに表示される
nvprof は NVTX ネーミングもサポートする
CUDA デバイス、コンテキスト、およびストリームの名前は、サマリーモードおよびトレースモードで表示される

スレッド名は要約モードで表示される

1.4 プロファイルデータのフラッシュ(消去)
プロファイリングのオーバーヘッドを削減するため、プロファイリングツールはプロファイル情報を収集して内部バッファに記録する
これらのバッファは、アプリケーションの動作の混乱を避けるため、優先度の低いディスクに非同期でフラッシュされる
まだフラッシュされていないプロファイル情報が失われないように、プロファイリングされるアプリケーションは、終了する前に、すべての GPU 作業が(CUDA 同期呼び出しを使用して)完了していることを確認してから、cudaProfilerStop()または cuProfilerStop()を呼び出す必要がある
これにより、対応するコンテキストのバッファされたプロファイル情報が強制的にフラッシュされる
CUDA アプリケーションにディスプレイまたはメインループを使用して動作するグラフィックが含まれる場合、そのループを実行するスレッドが exit()を呼び出す前に cudaProfilerStop()または cuProfilerStop()を呼び出す必要がある
これらの API の 1 つを呼び出さないと、収集されたプロファイルデータの一部またはすべてが失われる
OpenGL を使用するような一部のグラフィックスアプリケーションでは、エスケープキーを押すとアプリケーションが終了する
終了前に上記の関数を呼び出すことができない場合は、nvprof オプション --timeout を使用するか、ビジュアルプロファイラで「実行タイムアウト」を設定する
プロファイラは、タイムアウトの直前にデータを強制的にフラッシュする

1.5 CUDA Fortran アプリケーションのプロファイリング
PGI CUDA Fortran コンパイラーでコンパイルされた CUDA Fortran アプリケーションは、nvprof および Visual Profiler でプロファイルできる
プロファイラがソースファイルと行情報(カーネルプロファイル分析、グローバルメモリアクセスパターン分析、分岐実行分析など)を必要とする場合は、コンパイル時に「-Mcuda＝lineinfo」オプションを使用する
このオプションは、PGI 2019 バージョン 19.1 以降の Linux 64 ビットターゲットでサポートされる

訳 佐賀大学 今井康貴 imaiy@cc.saga-u.ac.jp