Arm MAP

Wprowadzenie do Arm MAP

Arm MAP to profiler, analizujący zachowanie aplikacji pod kątem wydajności i zużycia różnych zasobów, umożliwiający interaktywną analizę profilu wykonania programu, **dostosowany do pracy z aplikacjami rozproszonymi i wielowątkowymi**.

Niniejsze wprowadzenie zakłada znajomość: Tryb graficzny (przypomnienie).

Kiedy i po co używać MAP (wskazówki)?

MAP jest to profiler - stosujemy go w procesie optymalizacji, w celu poznania zachowania naszego kodu. Do procesu optymalizacji przystępujemy w momencie gdy posiadamy poprawnie działający (kompletny) program. Jego zastosowanie można rozbić na dwa kroki:

1. tworzenie *profilu wykonania* aplikacji
2. interaktywna analiza profilu

Wyniki zebrane przez MAP mogą pozwolić zlokalizować nam różne problemy dotyczące wydajności naszego kodu. Wymaga to naszej analizy i umiejętności wyciągania wniosków z zebranych wyników - program sam za nas niczego nie zrobi. Tyle i aż tyle.

Jak uruchomić MAP?

Rozważmy przykładową aplikację, uruchamianą pod MPI:

```
mpiexec -n 4  ./mmult4_c.exe 1024
```

W celu utworzenia profilu wystarczy (w konsoli):

- załadować pakiet arm-forge

```
module add plgrid/tools/arm-forge
```

- uruchomić program w ten sam sposób, **dodając na początku komendy "map –profile"**

```
map --profile   mpiexec -n 4  ./mmult4_c.exe 1024
```

Nastąpi wykonanie programu, po czym zostanie zapisany plik z profilem (o automatycznie wygenerowanej nazwie), z rozszerzeniem **`.map`** na końcu. Jeśli chcemy samemu wybrać nazwę pliku z profilem, należy dodać opcję **-o**: "`map --profile -o <nazwa-pliku>  (...komenda...)`".

- **nie trzeba żadnej rekompilacji**

Zaleca się jednak, aby program był skompilowany z flagą **-g**; wtedy otrzyma się informacje o czasie spędzonym w poszczególnym częściach kodu - będą one bardziej przydatne. Oprócz tego, niektóre optymalizacje (inlinowanie funkcji, nie zapisywanie ramki stosu) może spowodować nieczytelne wyniki. W takim wypadku warto spróbować kompilacji z "`-fno-omit-frame-pointer`"

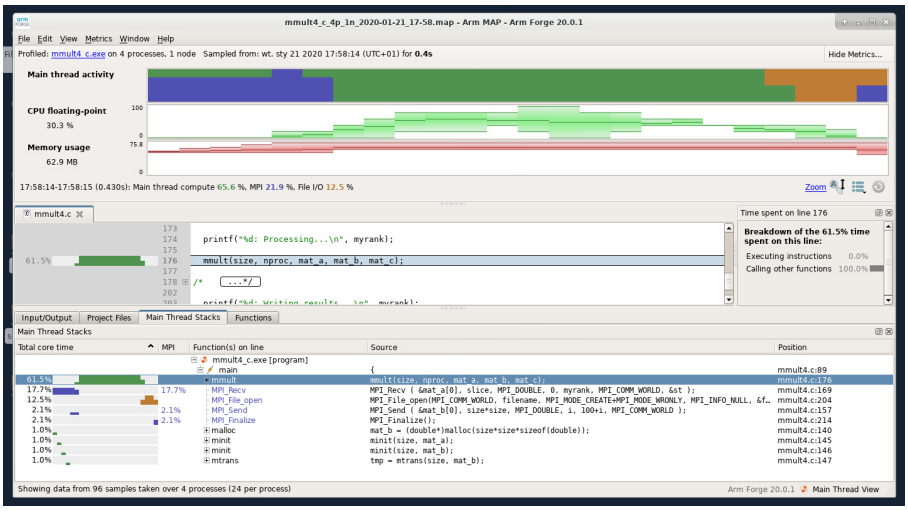
W celu obejrzenia profilu wystarczy (pracując w trybie graficznym):

- uruchomić **map** na pliku z profilem wykonania

```
map mmult4_c_4p_1n_2020-01-21_17-58.map
```

Otworzy się wtedy właściwe GUI programu MAP, w którym będziemy mogli przeanalizować zebrane wyniki.

Podstawowe funkcjonalności



- czas spędzony w funkcjach (wedle ścieżki wywołania - stack)
- czas spędzony w funkcjach (płaski profile - suma wszystkich wywołań), rozróżnienie inclusive i exclusive (self) time
- kolorowanie czasu, wedle aktywności: obliczenia, komunikacja (MPI), I/O
- możliwe inne widoki (metryki)
- zoomowanie na konkretny wycinek czasu

Dodatkowe uwagi

- User Guide (wersja 20.2),
- możliwość dobrania interwału, co ile profiler będzie próbkował naszą aplikację
- można wykonać map bez opcji --profile; wtedy otworzy się GUI, wykona program i od razu otworzą się wyniki; zalecamy stosowanie tego podejścia, tylko dla zadań o małej liczbie CPU lub gdy na analizę spędzamy tylko kilka chwil; plik .map ma tę przewagę, że zostaje na dysku i będziemy mogli go ponownie otworzyć (a wystarczy do tego tylko jeden CPU)

zalecenie:

- wykonywać MAP z profilem do pliku (w trybie sbatch)
- analizować w trybie interaktywnym (sesja graficzna) na małej liczbie CPU
  (chodzi o to aby nie zajmować zasobów → w trakcie analizy wyników, rdzenie prawie niczego nie liczą)