Fizyka wysokich energii: GooFit

Krótki opis usługi

Usługa **GooFit** udostępnia infrastrukturę obliczeniową dedykowaną do wykonywania masowo równoległego przetwarzania danych przy pomocy **GPGP** (***G**eneral **P**urpose **G**raphical **P**rocessing **U**nit*) i przeznaczona jest w pierwszej kolejności do wspomagania końcowego etapu analizy danych fizycznych HEP. Biblioteka GooFit wzorowana jest w dużej mierze na dodatku RooFit, który stworzony został w celu ułatwienia procesu dopasowania modeli teoretycznych do eksperymentalnych danych pomiarowych. RooFit stanowi obecnie integralną część standardowej dystrybucji aplikacji ROOT. Pakiet GooFit rozszerza funkcjonalność biblioteki RooFit o możliwość uruchamiania procedur dopasowania na urządzeniach wyposażonych w procesory graficzne zbudowane w oparciu o model masowego równoległego przetwarzania **CUDA** (***C**ompute **U**nified **D**evice **A**rchitecture*). Technologia CUDA umożliwia programiście bezpośredni dostęp do instrukcji oraz pamięci GPU, co z kolei pozwala na stworzenie aplikacji do rozwiązywania typowych problemów numerycznych przy użyciu procesorów graficznych. Model struktur danych oraz odpowiednie algorytmy do wykonywania typowych operacji na tych strukturach udostępnia w bibliotece GooFit pakiet Thrust. Dzięki zastosowaniu dodatku GooFit użytkownik końcowy nie musi znać żadnych szczegółów technicznych związanych z obsługą GPU i może skupić się całkowicie na własnej analizie fizycznej. Wstępne testy pokazały, że dzięki zastosowaniu biblioteki GooFit możliwe jest przyspieszenie procedury dopasowania od 10 do 1000 razy (w zależności od użytego modelu teoretycznego, który ma zostać dopasowany do danych).

Aktywowanie usługi

Usługa **GooFit** dostępna jest dla użytkowników posiadających ważne konto w Portalu PL-Grid oraz aktywowali następujące usługi:

1. Dostęp do klastra w dedykowanym centrum obliczeniowym (np. ZEUS)
2. Dostęp do maszyny udostępniającej złącze użytkownika (UI) w ośrodku obliczeniowym (CYFRONET, ICM, PCSS)
3. Aktywacja subskrypcji usługi GooFit na platformie dziedzinowej HEPGrid

Aktywacja usług dostępnych w poszczególnych ośrodkach obliczeniowych opisana jest dokładnie w podręczniku użytkownika: Aktywacja_Usług

Ograniczenia w korzystaniu

Użytkownicy korzystający z usługi GooFit powinni w pierwszej kolejności przeczytać poniższy rozdział "**Pierwsze kroki**". Z uwagi na specyfikę zastosowania pakiet GooFit może zostać uruchomiony jedynie na dedykowanej kolejce GPU. Należy również pamiętać, że GooFit wymaga instalacji odpowiedniej wersji pakietu ROOT oraz zestawu narzędzi i bibliotek do programowania równoległego CUDA. Zalecana konfiguracja, która została udostępniona na klastrze ZEUS, to: GooFit v0.4, ROOT v5.34.09 oraz CUDA 5.0.35.

Pierwsze kroki

Poniższa instrukcja dotyczy klastra ZEUS.

Po zalogowaniu status kolejki można sprawdzić przy pomocy następującego polecenia:

[plguser@zeus ~]$ **qstat -Q -f gpgpu**

Użytkownik usługi GooFit powinien zalogować się na maszynę udostępniającą złącze użytkownika, a następnie uruchomić zadanie interaktywne na kolejce GPGPU:

```
~ $ qsub -I -X -q gpgpu -l nodes=1:ppn=1:gpus=1
```

(dokładny opis systemu kolejkowego używanego na klastrze ZEUS można znaleźć tutaj: opis systemu kolejkowania zadań PBS).

Pracę z pakietem GooFit rozpoczynamy od zaimportowania **modułu goofit** (można oczywiście umieścić odpowiednie polecenie w skrypcie powłokowym):

```
~ $ module add goofit
 'gpu/cuda/5.0.35' load complete.
 'tools/root/5.34.09' load complete.
 'goofit/0.4' load complete.
```

Moduł goofit ładuje automatycznie odpowiednie wersje bibliotek CUDA oraz ROOT.

Warto zapamiętać, że zmienna środowiskowa, która wskazuje na katalog zawierający biblioteki GooFit nazywa się **$GOOFIT_ROOT** a jej wartość na klastrze ZEUS zdefiniowana jest jak poniżej:

```
~ $ echo $GOOFIT_ROOT
/software/local/el6/GPU/goofit/0.4
```

Użytkownicy planujący stworzenie swoich własnych plików konfiguracyjnych/kompilacyjnych powinni dodać tę zmienną do kolekcji ścieżek, które będą przeglądane automatycznie przez odpowiednie pliki makefile. Przykładowy plik pozwalający na dołączenie odpowiednich bibliotek i kompilację prostego przykładu podany jest poniżej.

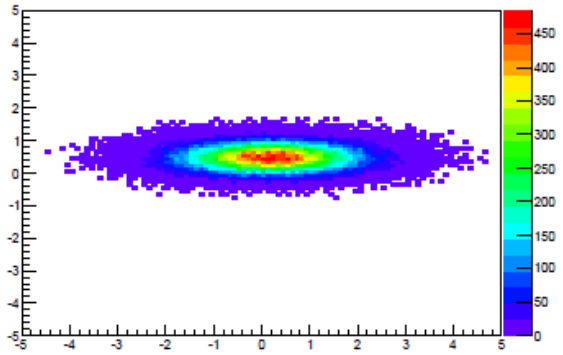
```
# @author Jeremi Piotrowski

CXX=nvcc
LD=g++
CXXFLAGS=-O3 -arch=sm_20

# Ustawienie potrzebne na Zeusie
GOOFITDIR=$(GOOFIT_ROOT)


# Nagłówki CUDA oraz GooFit
INCLUDES = -I$(CUDALOCATION)/include/ -I$(GOOFITDIR) -I$(GOOFITDIR)/rootstuff -I$(GOOFITDIR)/PDFs

# Nagłówki oraz biblioteki ROOT
ROOT_INCLUDES= -I$(shell root-config --incdir)
ROOT_LIBS = $(shell root-config --glibs)

# Biblioteki CUDA oraz GOOFIT
LIBS = -L$(CUDALOCATION)/lib64 -lcudart -L$(GOOFITDIR)/rootstuff -lRootUtils
WRKDIR=$(GOOFITDIR)/wrkdir/

# Nazwa własnej aplikacji
PROGRAM=myfit

all: $(PROGRAM)

# Linkowanie aplikacji
$(PROGRAM): $(PROGRAM).o
	$(LD) $(LIBS) $(ROOT_LIBS) $(WRKDIR)/*.o $^ -o $@
	@echo "Building $(PROGRAM) is done"

# Kompilacja aplikacji
$(PROGRAM).o: $(PROGRAM).cu
	$(CXX) $(INCLUDES) $(CXXFLAGS) $(ROOT_INCLUDES) -c -o $@ $<


clean:
	@rm -f *.o myfit
```

Proces budowy aplikacji wykorzystującej możliwości biblioteki GooFit, którą uruchamiamy na GPU może wyglądać jak poniżej:

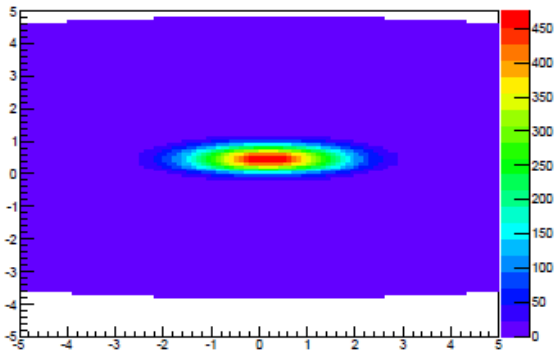
1. Kopiujemy przykładowy kod z katalogu zawierającego kolekcję przykładów dołączonych do pakietu GooFit:
   ```
   ~ $ cp -r $GOOFIT_ROOT/examples/2d_plot/ .
   ```
2. Przechodzimy do skopiowanego katalogu i budujemy aplikację:
   ```
   ~ $ make clean
   ~ $ make GOOFITDIR=$GOOFIT_ROOT
   2d_plot done
   ```
3. Wykonujemy:
   ```
   ~ $ ./2d_plot &> fit_rslt.log
   ~ $ ls
   2d_plot     2d_plot.o  data.png      png.png   xhist.png
   2d_plot.cu  Makefile   fit_rslt.log  pull.png  yhist.png
   ```

Wyprodukowane rysunki można następnie obejrzeć i przeanalizować. Poniżej przedstawiono wybrane wyniki działania przykładowego programu.

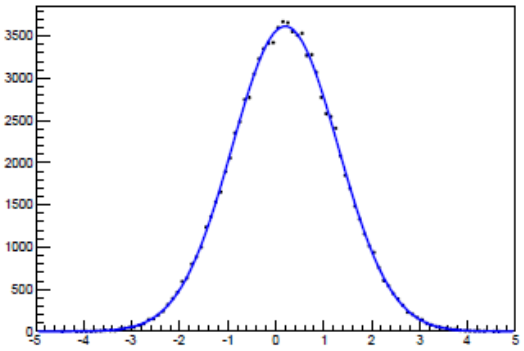
Dane



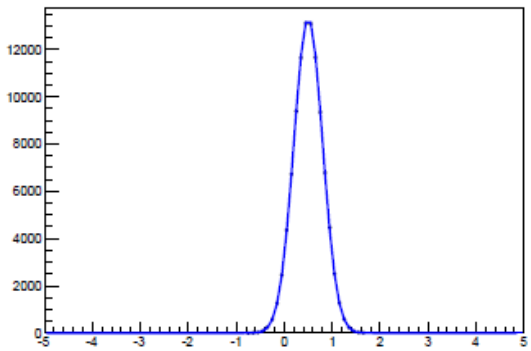
Dopasowanie



Rzut x



Rzut y



W przypadku typowych zastosowań pakietu GooFit możliwe jest stworzenie minimalnej aplikacji wzorcowej (template), która pozwala zwiększyć wydajność pracy z pakietem oraz zmniejsza prawdopodobieństwo błędów. Przygotowanie takiego wzorca rozpoczynamy od:

1. Zdefiniowania wielkości fizycznych, które mierzymy ("obserwable") eksperymentalnie
2. Przygotowania zbioru danych
3. Zdefiniowanie modelu (oraz odpowiednich parametrów), który chcemy użyć w celu dopasowania danych

Przykładowa implementacja przedstawiona jest poniżej. Zaczynamy od zdefiniowania zmiennej, która będzie reprezentować wartości zmierzone eksperymentalnie:

```
// Nazwa zmiennej, limit dolny, limit górny
Variable* xvar = new Variable("xvar", -5, 5);
```

Następnie dane należy zapisać do odpowiedniego kontenera, który użyty zostanie w procesie dopasowania:

```
// Przypisanie zmiennej do zbioru.
// Może być też vector zmiennych.
UnbinnedDataSet data(xvar);
```

```
// Wczytanie danych
std::ifstream plik;
plik.open("dane.txt");
while (!plik.eof()) {
  // Wypełnienie zmiennej wartością.
  plik >> xvar->value;
  // Dodanie eventu do zbioru.
  data.addEvent();
}
```

Następnie definiujemy model, którego chcemy użyć oraz jego parametry:

```
// Nazwa, wartość początkowa, limit dolny, limit górny
Variable* mean = new Variable("mean", 0, -10, 10);
Variable* sigma = new Variable("sigm", 1, 0.5, 1.5);
```

```
// Nazwa, obserwabla, parametry.
// Konstrukcja zależy od konkretnego rozkładu.
GaussianPdf gauss("gauss", xvar, mean, sigma);
```

Ostatecznie wykonujemy procedurę dopasowania, która wykonana zostanie na GPU:

```
// Dane data należą do rozkładu gauss.
gauss.setData(&data);
```

```
// Przekazanie rozkładu z danymi do FitManager'a.
FitManager fitter (&gauss);
// Uruchomienie fitu. Wynik zostanie wypisany na stdin.
fitter.fit();
```

Przykładowa aplikacja wraz z odpowiednim plikiem do kompilacji znajduje się w katalogu **/mnt/gpfs/work/plgrid/groups/plgggoofit**.

Gdzie szukać dalszych informacji?

Więcej informacji użytkownik może znaleźć w oficjalnej dokumentacji pakietu GooFit - dokument można znaleźć tutaj: dokumentacja biblioteki GooFit.
Dodatkowo dostępny jest kompletny kurs dotyczący użycia pakietu GooFit z przykładami o różnym stopniu trudności: kurs użytkownika pakietu GooFit.