Abordagem Paralela da Evolução Diferencial em GPU

Filipo Novo Mór[1], Omar A. C. Cortes[2], César A. M. Marcon[1]

[1]PPGCC – Pontifícia Universidade Católica do Rio Grande do Sul (PUCRS)
Av. Ipiranga, 6681 – Prédio 32 –90619-900 – Porto Alegre – RS – Brazil

[2]Departamento de Informática - Instituto Federal de Educação, Ciência e Tecnologia
Av. Getulio Vargas, 04 - Campus Monte Carlo - 65030-005 - São Luis - MA - Brazil

filipo.mor@acad.pucrs.br, omar@ifma.edu.br, cesar.marcon@pucrs.br

Resumo.*O objetivo deste trabalho é apresentar uma abordagem de paralelização do algoritmo de evolução diferencial em uma unidade de processamento gráfico (GPU). Foi utilizada uma função de benchmark chamada Schwefel, sendo que os testes foram feitos usando a abordagem fatorial 2^k com três fatores. Os resultados indicam speedup em torno de 2 com uma população pequena e em torno de 4 para uma população grande.*

1. Introdução

Algoritmos de evoluçãodiferencial (ED) fazem parte do grupo de algoritmos evolutivos, onde a simulação da evolução de uma população formada por possíveis soluçõesé realizadaobjetivando-se encontrar o melhor conjunto de soluções possível ao problema.Quando o problema a ser resolvido é de natureza contínua, o objetivo é encontrar um vetor *X* tal que se minimize ou maximize uma função *f(x)*. A principal vantagem da ED é que o algoritmo pode tratar funções objetivo que são descontinuas, não-diferenciáveis, não-lineares ou depende de muitas variáveis [Kromer, Platos, Snasel, 2013]. O algoritmo ED é apresentado na Figura 1.

```
1    Inicializa População
2    Avalia(População)
3    Enquanto critério de parada não alcançado
4     Para cada individuo i na população (Pop) faça
5          Selecionar três individuos
6           V ← individuo3 + F * (individuo1 – individuo2)
7           Novo_individuo ← cruzamentoentre V e Pop_i
8           Se o fitness(Novo_individuo) < fitness(Pop_i)
9              Substitui Pop_i por Novo_individuo
10          Fim-se
11       Fim-para
12   Fim-Enquanto
```

Figura 1 - Algoritmo seqüencial de Evolução Diferencial

O funcionamento do algoritmo é realtivamente simples. Para cada individuo da população cria-se um vetor de diferenças *V* na qual *F* é o fator de multiplicação normalmente no intervalo [*0,1*]. Então um novo indivíduo é criado usando-se como genitores o novo individuo*V* e o indivíduo alvo da população Pop_i em um processo semelhante ao do cruzamento discreto dos algoritmos genéticos. Se o novo indivíduo obtido pelo cruzamento apresentar um melhor *fitness*, este substitui o indivíduo alvo *i* na população.

2. Resultados Preliminares

A primeira abordagem implementada foi a utilização das funções de paralelização para GPU do Matlab via *Parallel Computing Toolbox* onde basicamente se transferem os dados da RAM para a GPU. A placa de vídeo usada é uma NVIDIA GTS 450. A CPU é um Intel i7 3.4Ghz com 16 GB de RAM executando Windows 7 64 bits. Os parâmetros testados são dimensão (*Dim*), chamadas à função de avaliação (*Calls*) e tamanho da população (*Pop*) nos seguintes valores respectivamente: {500, 2000}, {10^5, 10^6} e {100, 2000}, representando os níveis baixo e alto (-1, 1) do design de experimentos (2^k). Cada experimento foi executado 30 vezes para assegurar a adesão dos dados à distribuição normal (teorema do limite central).

A Figura 2 apresenta o gráfico de pareto, os efeitos principais dos parâmetros e interação entre os parâmetros. Pelo gráfico de pareto pode-se observar que os parâmetros que mais afetam o speedup são o tamanho da população e, a interação entre dimensão e população. Já no gráfico dos efeitos principais vê-se que o tamanho da população influencia positivamente enquanto que a interação entre dimensão e população diminui o speedup quando os parâmetros são aumentados. A Tabela 1 apresenta os speedups alcançados que variam entre 2.13 e 4.40, sendo que o maior (4.40) foi obtido nas configurações {500, 10^5 e 2000}.

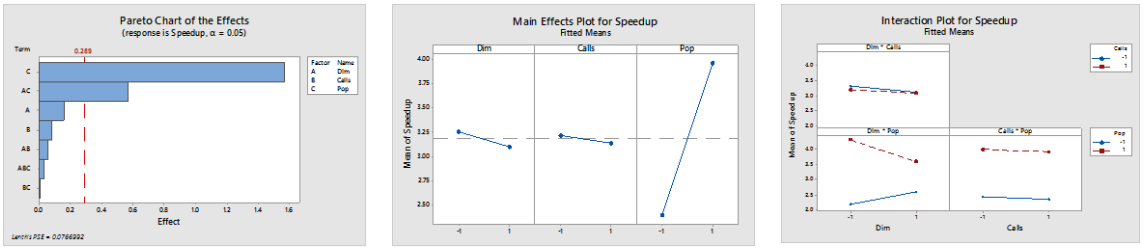

Figura 2 - Influencia dos parâmetros de execução na aceleração obtida.

Tabela 1 – Speedups encontrados por configuração

Dim	Calls	Pop	Speedup	Dim	Calls	Pop	Speedup
-1	-1	-1	2.242328005	-1	-1	+1	**4.402031153**
+1	-1	-1	2.631392488	+1	-1	+1	3.59878617
-1	+1	-1	2.139425192	-1	+1	+1	4.246680717
+1	+1	-1	2.571705205	+1	+1	+1	3.604751521

Considerando que os parâmetros de entrada são definidos pelo problema a ser processado, a identificação deste tipo de relação servirá de insumo ao dimensionamento de novas aplicações baseadas em ED, em especial na otimização de contratos de reseguros. Embora tento obtido razoáveis speedups, espera-se encontrar resultados melhores implementando o código em OpenACC e/ou C-CUDA, já que elas permitem um maior controle da execução em GPU.

Referências

Kromer, P.; Platos, J.; Snasel, V., (2013) "A brief survey of differential evolution on Graphic Processing Units,"IEEE Symposium on Differential Evolution, pp.157-164, 2013