Aproximação de raízes de funções pelo uso de métodos numéricos paralelizados

Matheus da Silva Menezes
matheus@ufersa.edu.br

Vinícius Samuel Valério de Souza
viniciussamuel@ufersa.edu.br

Daniel Sabino Amorim de Araújo
daniel@ufersa.edu.br

João Paulo Caraú Oliveira *
jpauloco@gmail.com

Raimundo Leandro Andrade Marques*
leanmarqs@hotmail.com

Universidade Federal Rural do Semi-Árido - UFERSA
59015-000, Campus Angicos, RN
www.ufersa.edu.br

RESUMO

No cotidiano profissional e científico, muitas vezes nos deparamos com situações onde necessitamos encontrar a raiz ou zeros de funções [1]. É possível se chegar a tal solução através de métodos diretos, porém, seu uso é limitado somente a funções cujo grau polinomial é menor ou igual a quatro. Para funções acima disso, devemos utilizar métodos iterativos, que mesmo não alcançando a solução exata, fornecem um resultado aproximado, com a precisão esperada pelo problema com o qual estamos lidando [2].

Devido à natureza dessas funções, não é possível determinar um único método para a resolução de todas elas. Mesmo um especialista com vasta experiência pode não ser capaz de decidir a melhor abordagem para um determinado problema, sem que mais de um desses métodos seja testado. Sendo assim, o uso de uma abordagem paralela pode fornecer uma alternativa mais eficiente na determinação do método mais apropriado para função a qual estamos trabalhando.

Diante disso, este trabalho propõe uma abordagem que utiliza paralelismo e os métodos numéricos Bisseção [1], Falsa Posição [1], Newton [1] e Secante [1] para encontrar de maneira eficiente zeros de funções transcendentes.

1. Abordagem Numérica Paralela

O algoritmo proposto neste trabalho utiliza noções de teoria dos jogos especificamente teoria da escolha racional onde aplicada uma lógica (resolução por métodos diferentes) a premissas dadas (número máximo de iterações e a precisão estabelecida) [3], com processos que competem entre si, para determinar o mais rápido. Cada processo utiliza um método distinto para encontrar o zero de uma função qualquer e aquele que for o vencedor, ou seja, o processo que primeiro convergir, será consequentemente o mais apropriado para a função analisada. Também compramos os resultados para soluções de forma paralela e de forma sequencial.

Existem cinco processos atuando no algoritmo. O processo 0 é chamado *Master*, sua função é coordenar o trabalho dos demais processos chamados *Slaves* ou *Worker*[4], sendo que o termo utilizado neste trabalho será *Slave*. Cada *Slave* procura a raiz da função proposta, por meio de um dos quatro métodos escolhidos. O processo de convergência é atingido através dos seguintes passos:

I. O *Master* garante que todos os processos sejam iniciados por meio de uma rotina que cria uma barreira lógica, a qual todos os processos devem atingir, antes que o processamento realmente tenha início, funcionando assim como uma linha de largada.

II. A partir de então os *Slaves* entram em *loop*, procurando o zero da função analisada.

III. Quando um dos processos encontra a raiz ele envia uma mensagem para o mestre e então termina seu laço de execução. Assim que o *Master* recebe a mensagem

* Bolsistas de Iniciação Científica UFERSA

ele envia uma mensagem para os demais *Slaves*, informando que eles também devem parar de realizar cálculos, pós a raiz já foi encontrada.

IV. Caso todos os processos atinjam o número máximo de iterações sem que a raiz seja encontrada, dizemos que a função analisada não convergiu para nenhum dos processos.

A seguir verificaremos os resultados obtidos para análise de algumas funções, quando colocamos um algoritmo sequencial versus um algoritmo paralelo. Todos os testes foram realizados em um computador com Processador Core i5 2ª Geração e 4 Gb RAM, Windows 8, escrita em linguagem C e compilado no Microsoft Visual Studio 2013. Os critérios de parada foram um limite máximo de 1000 iterações, e precisão de 10^{-14}.

FUNÇÃO —›	$x^5 - 2x^4 - 9x^3 + 22x^2 + 4x - 24 = 0$		$\cos^4(2x)\sin^2(\frac{x}{3}) = 0$		$\log_2(x+1) = 0$		$(\sqrt{x} - 5)e^{-x} = 0$	
MÉTODO	Iterações	Tempo	Iterações	Tempo	Iterações	Tempo	Iterações	Tempo
PARALELO								
Bisseção	6,33	0,00014	16,67	0,00028	21,33	0,00039	24,67	0,000575
Falsa Posição	3,67	0,00027	1	0,000131	1	0,00041	16	0,000573
Newton	4,67	0,00014	50	0,010996	4	0,00023*	3,67	0,0000876
Secante	5	0,00011*	1	0,0000417*	1	3,9E-05	5	0,000196*
SEQUENCIAL								
Bisseção	51	1,7*	73	0,8	72	0,7	45	0,8*
Falsa Posição	8	0*	1	1*	1	0,1*	18	0,6*
Newton	5	0*	1000	7,7	1	0*	20	0*
Secante	5	0*	1	0*	1	0,1*	5	0*

Foram realizados vários testes para a mesma função quanto ao tempo de execução e número de iterações, apurando-se a média aritmética desses resultados. Os tempos que estão acompanhados de "*" mostram os processos que convergiram, nos testes em paralelo.

Como podemos verificar através da tabela acima a abordagem paralela leva vantagem sobre a sequencial, pois como o algoritmo irá parar as iterações dos demais métodos quando a raiz for descoberta, ele sempre utilizará o menor número de iterações necessárias para convergir. Para a primeira função, por exemplo, o pior caso foi a bisseção, enquanto no método sequencial ela precisou de 51 iterações e um tempo maior que um segundo, no algoritmo paralelo ela consumiu apenas 6.33 iterações em média e um tempo de poucos milésimos de segundo.

Palavras-chave*: Métodos iterativos, raízes de funções, paralelismo, MPI.*

Referências

[1] Franco, Neide Bertoldi. "Cálculo Numérico", Pearson, São Paulo, abril 2013.

[2] F.F. Campos. "Algoritmos Numéricos". LTC, Rio de Janeiro, 2010.

[3] FIANI, Ronaldo. Teoria dos Jogos para cursos de administração e economia. 2ª Ed. Revisada e atualizada – Rio de Janeiro-RJ; Elsevier, 2006 – 3ª Reimpressão.

[4] Message Passing Interface (MPI), disponível em https://computing.llnl.gov/tutorials/mpi/#What, acesso em 27/02/2014, 23h.

