Uso de *Threads* para o Planejamento de Segunda Ordem das Redes Geodésicas.

Vinicius Nonnenmacher, Ismael Érique Koch, Fabrício dos Reis Furtado, Rodrigo Righi, Luiz Gonzaga Jr.

[1]Programa de Pós-Graduação em Computação Aplicada (PIPCA)
Universidade do Vale do Rio dos Sinos (UNISINOS) – São Leopoldo, RS – Brasil

vnonnenmacher@gmail.com, isma.koch@gmail.com, fabricio14@gmail.com

rrrighi@unisinos.br, lgonzagajr@gmail.com

Resumo. *O presente trabalho demonstra um estudo feito na paralelização da meta-heurística da colônia artificial de abelhas (ABC) aplicada ao problema de planejamento de segunda ordem de uma rede geodésica planimétrica. A partir de um programa desenvolvido sequencialmente em python, foram realizados experimentos utilizando processamento paralelo de uma das fases do algoritmo.*

1. Introdução

O uso de técnicas para processamento de alto desempenho com fins práticos é utilizado em larga escala [Mattson et al. 2004]. Com o objetivo de verificar a extensão dos benefícios do uso de paralelismo ao algoritmo ABC aplicado ao planejamento de segunda ordem das redes geodésicas, este artigo descreve a implementação e análise do problema utilizando a paralelização de dados através de *threads*.

O planejamento de segunda ordem das redes geodésicas [Grafarend and Sansò 2012] é um problema conhecido na área da geodesia. As redes geodésicas são redes de triângulos medidos com exatidão podendo usar técnicas de levantamento terrestres ou geodesia espacial. Seus vértices são materializados em locais estratégicos a fim de obter melhor precisão e atender a finalidade desejada. Essas redes são utilizadas para serviços baseados em localização como mapeamento, geoinformação, registro de terras etc. Uma rede geodésica estabelecida deve ter alta precisão e confiabilidade. Para isso, no planejamento da mesma, deve-se estimar os melhores pesos para a matriz peso, onde podem ser aplicadas meta-heurísticas de minimização numérica.

O algoritmo Colônia Artificial de Abelhas (Artificial Bee Colony - ABC) foi implementado em *python* para geração dos valores da Matriz Peso a fim de minimizar $d^T * d$. O procedimento para cálculo segue abaixo, sendo P, a Matriz Peso a ter seus valores estimados pela ABC dentro do espaço de busca, e A e Q_X, a Matriz Design e a Matriz Criteria calculada com pesos originais da rede, respectivamente.

$$Q_{XM} = (A^T \cdot P \cdot A)$$
$$Q^{-1}_{R_{XM}} = Q^T_{XM} \cdot (Q_{XM} \cdot Q^T_{XM})^{-1}$$
$$D = Q^{-1}_{R_{XM}} - Q_X$$
$$d = vector(D)$$

$$d^T \cdot d \rightarrow \min$$

Onde, ^{T} é a transposta da respectiva matriz, $^{-1}$ indica a inversão da matriz e $vector$ representa a extração da diagonal de uma matriz e sua transposição em um vetor.

2. Implementação e Experimentos

O ABC é um algoritmo de busca que reproduz N ciclos otimizando a solução através de uma meta-heuristica. O *loop* principal do algoritmo executa diferentes fases, verificando e otimizando o *fitness* das soluções a cada iteração até um número pré-definido de ciclos. A função descrita no capítulo 1 é executada dentro da fase das abelhas trabalhadoras onde o objetivo é verificar e melhorar as soluções em um vetor de soluções.

Com o objetivo de melhorar o tempo de execução da fase das abelhas trabalhadoras, cada *thread* foi programada de forma a receber por parâmetro a posição inicial e final do intervalo o qual a *thread* deve processar. O vetor de soluções é acessado por cada *thread*, que computa e salva os resultados na mesma posição do vetor. Assim, a cada iteração do algoritmo, ao entrar na fase das abelhas trabalhadoras, o programa cria *N threads*, e processa paralelamente o vetor de soluções, salvando as novas soluções no mesmo. Foram realizados experimentos com diferentes números de *threads* e o tempo de execução foi mensurado por meio da função *time* do *python*.

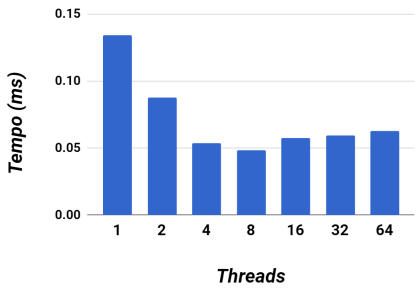

Figura 1. Tempo de execução da função por número de threads.

Os resultados podem ser verificados na Figura 1 que demonstra o tempo de execução da função com diferente número de *threads*.

3. Conclusão

Para otimizar o tempo de execução do algoritmo ABC aplicado as redes geodésicas de segunda ordem, este trabalho propôs a paralelização da fase das abelhas trabalhadoras distribuindo o processamento de um vetor de soluções em *N threads*. Os resultados demonstram que o tempo de execução do método foi reduzido até o uso de 8 *threads* gerando um *speed up* sublinear. Devido ao tempo perdido em comunicação e sincronismo, é possível verificar na prática que não compensa a utilização de mais que 8 *threads*.

Referências

Grafarend, E. and Sansò, F. (2012). *Optimization and Design of Geodetic Networks*. Springer Berlin Heidelberg.

Mattson, T., Sanders, B., and Massingill, B. (2004). *Patterns for Parallel Programming*. Software Patterns Series. Pearson Education.