Paralelização do Algoritmo *Fast Non-Dominated Sorting* para o Ranqueamento de Cidades de Acordo com Informações do Clima

Vinícius A. Barros[1], **Luiz H. Nunes**[1,2], **Paulo S. L. de Souza**[1], **Júlio C. Estrella**[1]

[1]Instituto de Ciências Matemática e de Computação – Universidade de São Paulo (USP)
Avenida Trabalhador São-carlense, 400 – 13566-590 – São Carlos - SP – Brazil

[2]Instituto Federal de São Paulo (IFSP)
Araraquara – SP – Brasil

{viniciusaires}@usp.br, {lhnunes,pssouza,jcezar}@icmc.usp.br

Resumo. *Algoritmos de otimização multiobjetivo são utilizados para solucionar problemas que envolvam mais de um critério a ser otimizado e geralmente apresentam mais de uma solução ótima. Grandes conjuntos de dados são coletados por sensores, o que torna necessária a utilização de algoritmos paralelos para obtenção de um melhor desempenho. Este trabalho apresenta uma abordagem paralela para o algoritmo de otimização multiobjetivo Fast Non-Dominated Sorting, com a finalidade de diminuir o seu tempo de resposta quando aplicado a um grande volume de dados. Os resultados obtidos constatam ganhos de performance do algoritmo implementado além de apresentar as limitações encontradas.*

1. Introdução

Nos últimos anos, dispositivos móveis e sensores têm aumentado a sua capacidade de processamento e precisão na coleta de dados. Diferentes aplicações são desenvolvidas com base nas informações coletadas por esses aparelhos, como por exemplo, aplicações nas áreas médicas, controle do trânsito e agricultura [Atzori et al. 2010].

Aplicações como ViSIoT [Nunes et al. 2016] e CASSARAM [Perera et al. 2013] visam fornecer mecanismos para gerenciar as informações desses aparelhos para desenvolver tais aplicações. Entretanto, diversos desafios são enfrentados pela utilização de sensores para a coleta de dados, dentre eles, o grande número de dispositivos disponíveis, o volume de dados gerados e a capacidade de processá-los em um menor tempo computacional [Li et al. 2015].

Este artigo apresenta uma proposta paralela do algoritmo para otimização multiobjetivo *Fast Non-Dominated Sorting*. Objetiva-se, desta forma, obter um menor tempo de resposta no ranqueamento de cidades de acordo com informações do clima. A base de dados utilizada neste projeto foi retirada da plataforma *Open Weather Map*[1], nela estão contidas informações climáticas de aproximadamente 200 mil cidades.

2. Contextualização

O problema abordado consiste no ranqueamento de cidades de acordo com características climáticas. A base de dados utilizada contém informações de cidades referentes a umidade

[1]https://openweathermap.org/

relativa, temperatura mínima e máxima e a velocidade do vento.

Este trabalho se caracteriza como um problema de otimização multiobjetivo pois para o ranqueamento de cidades, diversos fatores de maximização e minimização podem ser aplicados para determinar se uma cidade apresenta condições climáticas melhores em relação as demais. A seguir é descrito o algoritmo utilizado para o ranqueamento das cidades o *Fast Non-Dominated Sorting*, apresentando também os conceitos relacionados ao ótimo de pareto e a proposta paralela do algoritmo.

2.1. Ótimo de Pareto

Uma caraterística importante de problemas que envolvam otimização multiobjetivo é a raridade dos casos em que um único ponto é capaz de otimizar todas as funções objetivas. Em grande parte dos casos, um conjunto de soluções que apresentam diferentes *trade-offs* entre seus objetivos são usadas para solucionar os problemas em detrimento de uma única solução. Relações de dominância de Pareto são utilizadas para comparar soluções. Ao conjunto de soluções ótimas de um problema é dado o nome de conjunto de soluções ótimas de Pareto ou soluções não dominadas [Jaimes et al. 2009].

2.2. *Fast Non-Dominated Sorting*

A fim de identificar o conjunto de soluções ótimas, cada possível solução deve ser comparada com todas as outras soluções com a finalidade de identificar suas relações de dominância [Deb et al. 2002].

O algoritmo *Fast Non-Dominated Sorting* tem uma complexidade de $O(MN)$ comparações para cada uma das soluções, em que M é o número total de objetivos. Quando este processo é contínuo e realiza a busca em todos os membros do primeiro nível de não dominância, a complexidade do algoritmo é $O(MN^2)$ [Deb et al. 2002].

O Algoritmo 1 apresenta o pseudocódigo do algoritmo *Fast Non-Dominated Sorting*. Observa-se que existe um grande número de comparações para a definição das soluções pareto ótimas. Em casos que lidam com um grande volume de dados, o algoritmo tem um comportamento lento devido a sua complexidade assintótica.

Algoritmo 1: *Fast Non-Dominated Sorting*

```
 1: for p ∈ P do
 2:     S_p ← 0
 3:     n_p ← 0
 4:     for q ∈ P do
 5:         if p ≺ q then
 6:             S_p ← S_p ∪ {q}
 7:         else
 8:             if q ≺ p then
 9:                 n_p ← n_p + 1
10:     if n_p = 0 then
11:         p_rank ← 1
12:         F_i = F_i ∪ {p}
13: i ← 1
14: while F_i ≠ 0 do
15:     Q ← 0
16:     for p ∈ F_i do
17:         for q ∈ S_p do
18:             n_p ← n_p − 1
19:             if n_p = 0 then
20:                 q_rank ← i + 1
21:                 Q ← Q ∪ {q}
22:         i ← i + 1
23:         F_i ← Q
```

2.3. Proposta Paralela

Alguns trabalhos presentes na literatura apresentam abordagens utilizando paralelismo híbrido com GPU, como apresentado em [Smutnicki et al. 2014] e [Santander-Jiménez and Vega-Rodríguez 2015].

A proposta de implementação paralela do algoritmo *Fast Non-Dominated Sorting* foi desenvolvida em Java *threads* e *sockets*. A aplicação destas técnicas de paralelização aborda os paradigmas de memória compartilhada e distribuída. A divisão de tarefas foi feita a partir da criação de processos para realização das comparações de dominância e não dominância entre cada uma das cidades. Para execução dos experimentos foram definidos três casos de teste: memória compartilhada, memória distribuída e o modelo memória compartilhada e distribuída. O planejamento dos experimentos foi da seguinte maneira:

1. Memória Compartilhada: uma máquina com 2, 4 e 8 processadores;
2. Memória Distribuída: um único processo com 2, 4, 8 máquinas;
3. Memória Compartilhada e Memória Distribuída:
   (a) Três máquinas com 2, 4 e 8 processadores;
   (b) Cinco máquinas com 2, 4 e 8 processadores; e
   (c) Nove máquinas com 2, 4 e 8 processadores.

Todas as máquinas utilizadas para o experimento tem a mesma configuração de *hardware*. Cada máquina utilizada apresenta a seguinte configuração: processador Intel Core I7 LGA - 1150 e 32 GB de RAM.

3. Resultados

A Figura 1 apresenta o *speedup* obtido para todos os experimentos executados. Foram executadas 10 repetições para cada experimento e mensurado o tempo de execução. Observa-se que os maiores ganhos de *speedup* foram com a utilização de 1 ou 2 máquinas com 8 processadores cada.

A partir de 4 ou mais máquinas o desempenho começa cair devido ao *overhead* de comunicação em virtude do maior número de máquinas utilizadas. Ainda na Figura 1 é apresentada a eficiência obtida para todos os experimentos executados. Semelhante ao comportamento obtido pela análise do *speedup*, a eficiência do algoritmo apresenta seus melhores índices nos experimentos com 1 ou 2 máquinas.

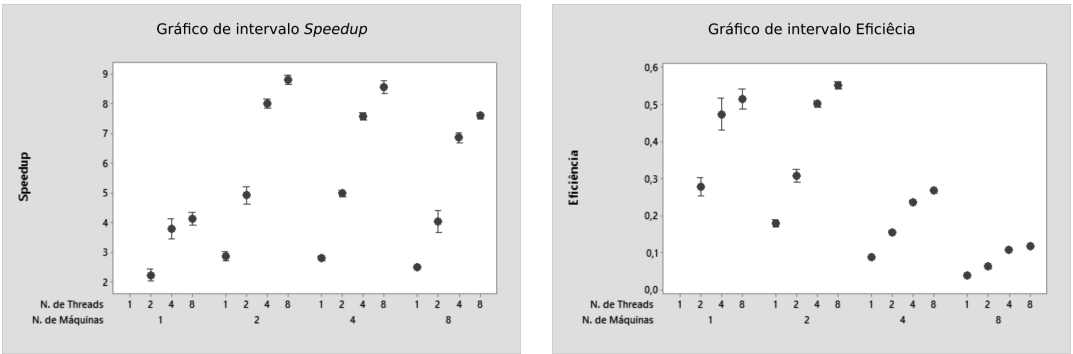

Figura 1. Resultado *speedup* e eficiência.

A Tabela 1 mostra de forma ordenada os resultados obtidos pelos experimentos executados. A ordenação da mesma é em função do maior *speedup* e eficiência obtida. Observa-se que os melhores experimentos executados foram com 2 máquinas e 8 processadores e 1 máquina e 8 processadores. O caso em que se obteve pior desempenho foi com 8 máquinas com apenas um processo cada.

Tabela 1. Tabela ordenada com os melhores experimentos executados.

Paradigma	N. de Máquinas	N. de Threads	Speedup	Eficiência	Custo
Memória Compartilhada e Distribuída	2	8	8,82	0,55	2
Memória Compartilhada	1	8	4,12	0,52	1
Memória Compartilhada e Distribuída	2	4	8,02	0,5	2
Memória Compartilhada	1	4	3,79	0,47	1
Memória Compartilhada e Distribuída	2	2	4,92	0,31	2
Memória Compartilhada	1	2	2,22	0,28	1
Memória Compartilhada e Distribuída	4	8	8,57	0,27	4
Memória Compartilhada e Distribuída	4	4	7,58	0,24	4
Memória Distribuída	2	1	2,88	0,18	2
Memória Compartilhada e Distribuída	4	2	4,99	0,16	4
Memória Compartilhada e Distribuída	8	8	7,6	0,12	8
Memória Compartilhada e Distribuída	8	4	6,86	0,11	8
Memória Distribuída	4	1	2,8	0,09	4
Memória Compartilhada e Distribuída	8	2	4,04	0,06	8
Memória Distribuída	8	1	2,5	0,04	8

4. Considerações Finais

Ao analisar os resultados obtidos com a aplicação das métricas de avaliação de desempenho *speedup* e eficiência, obtém-se as seguintes análises: a eficiência diminui com o aumento do número de máquinas, devido o maior número de comunicações realizadas; O *speedup* e a eficiência aumentam consideravelmente com o uso de um maior o número de *threads*; O desempenho diminui quanto menor for o número de *threads*. Dado um limiar de número de máquinas e *threads*, o desempenho diminui devido aos pontos de sincronização, condições de disputa e uma quantidade maior de comunicações.

Referências

Atzori, L., Iera, A., and Morabito, G. (2010). The internet of things: A survey. *Computer networks*, 54(15):2787–2805.

Deb, K., Pratap, A., Agarwal, S., and Meyarivan, T. (2002). A fast and elitist multiobjective genetic algorithm: Nsga-ii. *IEEE transactions on evolutionary computation*, 6(2):182–197.

Jaimes, A. L., Martınez, S. Z., and Coello, C. A. C. (2009). An introduction to multiobjective optimization techniques. *Optimization in Polymer Processing*, pages 29–57.

Li, S., Xu, L. D., and Zhao, S. (2015). The internet of things: a survey. *Information Systems Frontiers*, 17(2):243–259.

Nunes, L. H., Estrella, J. C., Nakamura, L. H. V., Libardi, R. M. D. O., Ferreira, C. H. G., Jorge, L., Perera, C., and Reiff-Marganiec, S. (2016). A distributed sensor data search platform for internet of things environments. *International Journal of Services Computing (IJSC)*, 4(1):1–12.

Perera, C., Zaslavsky, A., Christen, P., Compton, M., and Georgakopoulos, D. (2013). Context-aware sensor search, selection and ranking model for internet of things middleware. In *Proceedings of the 2013 IEEE 14th International Conference on Mobile Data Management - Volume 01*, MDM '13, pages 314–322, Washington, DC, USA. IEEE Computer Society.

Santander-Jiménez, S. and Vega-Rodríguez, M. A. (2015). A hybrid approach to parallelize a fast non-dominated sorting genetic algorithm for phylogenetic inference. *Concurrency and Computation: Practice and Experience*, 27(3):702–734.

Smutnicki, C., Rudy, J., and Zelazny, D. (2014). Very fast non-dominated sorting. *Decision Making in Manufacturing and Services*, 8(1-2):13–23.