Trabalho apresentado no XXXIX CNMAC, Uberlândia - MG, 2019.

Proceeding Series of the Brazilian Society of Computational and Applied Mathematics

Uma paralelização por OpenMP de método baseado em MOPSO aplicado em estimação de parâmetros

Breno Tiago de Souza Mota[1]
Instituto Politécnico, UERJ, Nova Friburgo, RJ
Sanderson L. Gonzaga de Oliveira[2]
DCC, UFLA, Lavras, MG
Luiz Nélio Henderson Guedes de Oliveira[3]
Instituto Politécnico, UERJ, Nova Friburgo, RJ

Resumo. Problemas de estimação de parâmetros, ao serem tratados como problemas de otimização e resolvidos por método meta-heurístico, exigem uma quantidade grande de vezes o cálculo da função objetivo . A resolução da função objetivo demanda um custo computacional elevado. Neste trabalho, propomos uma versão paralela, via OpenMP, de um método baseado na meta-heurística *Multi-Objective Particle Swarm Optimization*, aplicado no problema de estimação de parâmetros de um importante modelo termodinâmico. Com os resultados obtidos, é comprovada a robustez da metodologia de divisão explícita de trabalho utilizada para paralelizar o método.

Palavras-chave. MOPSO, computação paralela, OpenMP, estimação de parâmetros

1 Introdução

Ao tratar o problema da estimação de parâmetros de um modelo via otimização, alguns aspectos influenciam o tempo computacional para obter a solução. Problemas inversos são extremamente mal postos, não-lineares e multimodais [4]. Dessa forma, não é indicada a utilização de determinados métodos numéricos determinísticos nesse contexto, como o método de Newton ou similares, que são baseados em derivadas.

Em meta-heurísticas populacionais, o principal mecanismo de otimização é a comparação entre os valores de aptidão dos membros da população. Dessa forma, a heurística resultante da aplicação da meta-heurística requer que a função objetivo seja calculada diversas vezes, gerando um custo computacional elevado. A quantidade de dados experimentais para serem ajustados também influencia no custo computacional da heurística.

Nesse contexto, a computação paralela fornece recursos que se destinam a otimizar o processamento de simulações numéricas. Neste trabalho, são mostrados os resultados da

[1]brenotsm1@gmail.com
[2]sanderson@ufla.com
[3]neliohenderson@gmail.com

DOI: 10.5540/03.2020.007.01.0461

utilização da API OpenMP na paralelização de um método baseado na meta-heurística *Multi-Objective Particle Swarm Optimization* (MOPSO). O método paralelo foi utilizado na solução de um problema de estimação de parâmetros de um modelo termodinâmico referente a dois fluidos. A paralelização é baseada na técnica *explicit work sharing*.

Na seção 2, é mostrada uma breve revisão bibliográfica de versões em paralelo da meta-heurística MOPSO e suas respectivas aplicações. Na seção 3, é descrito o modelo termodinâmico que será alvo da estimação de parâmetro. A descrição do processo de paralelização do código é feita na seção 4. A metodologia empregada nos experimentos computacionais é explicada na seção 5. As seções 6 e 7 são destinadas à exposição dos resultados e conclusões, respectivamente.

2 Trabalhos relacionados

Foram publicadas versões paralelas da meta-heurística MOPSO aplicadas em vários problemas. para citar exemplos recentes, Hussain e Fujimoto [5] apresentaram uma implementação paralela de método projetado pela meta-heurística MOPSO baseada em CUDA e GPU aplicada em problemas testes de grandes dimensões e grandes populações de enxames. McDougall e Nokleby [7] utilizaram um esquema de síntese intensiva para verificar melhorias de desempenho na paralelização de um método baseado na meta-heurística MOPSO. O algoritmo paralelo resultante foi aplicado em dois exemplos de *benchmark* para verificar a eficiência da implementação. Li *et al.* [6] propuseram uma versão hibrida de modelos de programação de memória distribuída e compartilhada, adotando um modelo de ilha e dividindo toda a população em várias subespécies. Os experimentos foram realizados em funções testes.

Realizamos uma busca na base de dados bibliográficos Scopus® com as palavras-chave "MOPSO", "*inverse problem*" e "*parallel*". Nessa busca, não foi retornado nenhum documento. Também buscamos na base Scopus® as palavras-chave "MOPSO", "*parameter estimation*" e "*parallel*". Nessa segunda busca, relacionado ao nosso contexto, foi retornada somente a publicação de Yuhui *et al.* [11]. Yuhui *et al.* [11] utilizaram rotinas para programação paralela do Matlab em seus experimentos. Os autores realizaram dois experimentos em casos específicos. No primeiro experimento, foi utilizada uma função teste. No segundo caso, os autores aplicaram um modelo de simulação de chuva-vazão hidrológico Xinanjiang. No experimento com melhor resultado, os autores obtiveram *speedup* menor do que quatro em uma máquina com processador Intel® Core™2, com quatro núcleos de 2,4 GHz de frequência.

3 Problema teste

Estimação de parâmetros está na classe de problemas inversos. Equações cúbicas de estado são modelos termodinâmicos que predizem propriedades (pressão de vapor, densidade do líquido, volume de líquido saturado etc.) de substâncias puras ou mistura [1]. Alguns desses modelos exigem parâmetros numéricos para cada substância.

DOI: 10.5540/03.2020.007.01.0461

A equação de estado de Patel-Teja [8] é um importante modelo aplicável a componentes puros, incluindo hidrocarbonetos, não-hidrocarbonetos e componentes polares (água, amônia, álcoois) e não polares. Essa equação é fortemente recomendada para modelar o comportamento de hidrocarbonetos pesados e componentes polares para as quais outras equações são inadequadas. A equação na forma P - explícita é

$$P = \frac{RT}{v-b} - \frac{a[T]}{v(v+b)+c(v-b)}, \tag{1}$$

em que v é o volume molar, R é a constante universal dos gases, T é a temperatura corrente, a, b e c são funções que necessitam dos parâmetros F e ζ_c. Portanto, para se usar a equação 1, é necessária a temperatura crítica T_c, pressão crítica P_c e outras duas constantes, F e ζ_c, que dependem da substância em análise. Valores ótimos dos parâmetros são encontrados por meio da predição da equação para pressões de saturação e densidades de líquido saturado, em comparação com dados experimentais.

4 Um método MOPSO paralelo baseado na API OpenMP aplicado na estimação de parâmetros

Neste trabalho, foi utilizada a versão de Coello *et al.* [3] da meta-heurística MOPSO. Por ser uma meta-heurística populacional, o processo de otimização é realizado ao se utilizar um conjunto de vetores, que são as variáveis do problema e que, obrigatoriamente, necessitam do cálculo de seus respectivos valores de função. Dessa forma, a parte mais crítica no custo computacional de um problema de estimação de parâmetros é a execução da função objetivo. Assim, o cálculo das funções foi realizado em paralelo neste método. Apesar de o método ser estocástico, foi utilizada a mesma semente para o gerador de números aleatórios, tornando confiável a comparação entre diferentes execuções.

A implementação paralela vincula uma quantidade de vetores da população a determinado processo. Como os experimentos foram realizados em uma máquina com 12 núcleos, a implementação tem 12 funções, em que cada uma dessas funções é executada por apenas um núcleo específico da máquina. Como as computações das funções são independentes, essa ligação, entre processo e núcleo, é feita de maneira explícita por meio da identificação do núcleo da máquina, utilizando a função *omp_get_thread_num()*. Cada núcleo calcula os valores de função de $ITV = \frac{\mathrm{NP}}{N}$ indivíduos da população.

Dessa forma, no laço de repetição paralelizado, foi utilizada a diretiva *#pragma omp parallel shared(exame,F,ITV) private (i)*, em que *exame* é um *array* de tamanho NP com as variáveis do problema, F é um *array* com N funções, e ITV é a quantidade de indivíduos da população a ser tratado por cada núcleo da máquina. Essas estruturas de dados são compartilhadas de maneira explícita entre os núcleos. Somente a variável i é privada porque é responsável por percorrer um intervalo específico do *array* do *exame* que pertencente a cada núcleo.

DOI: 10.5540/03.2020.007.01.0461

5 Descrição dos testes

A implementação do método baseado na meta-heurística MOPSO e sua versão paralela foram escritas na linguagem de programação C++. Foi utilizada a equação *speedup* = $t(1)/t(N)$ como métrica para avaliar os resultados, em que $t(1)$ é o tempo de processamento da versão sequencial do método e $t(N)$ é o tempo de execução da versão paralela utilizando N núcleos.

A máquina em que os experimentos foram realizados contém dois processadores Intel® Xeon® E5-2420 Hexa Core, com frequência de 1.9 GHz. Como descrito, nessa máquina, cada processador é composto por seis núcleos, totalizando 12 núcleos (físicos). Esse processador contém duas *threads* por núcleo. Todavia, a tecnologia *hyper-threading* da máquina foi desabilitada para os experimentos realizados neste trabalho.

A implementação paralela é configurada com o número de indivíduos da população NP. Como a implementação foi baseada na técnica *explicit work sharing*, foi estabelecida uma quantidade de indivíduos da população (NP) de forma que a divisão pela quantidade de núcleos (N) fosse um número inteiro. Como exemplo, nos principais experimentos, foi estabelecida uma população (NP) de 120 vetores, de maneira que se pudesse definir a quantidade de núcleos (N) com números inteiros divisores de 120, isto é, 1, 2, 3, 4, 5, 6, 8, 10 e 12, para se estudar a redução em tempo de execução da versão paralela em relação à sequencial.

Cada experimento foi realizado com um total de 300 iterações da meta-heurística MOPSO. Para uma população de 120 indivíduos, o número de avaliações da função objetivo foi igual a 36.000.

Os testes foram realizados baseando-se no problema de estimação de parâmetros relacionado a duas substâncias [Formamide (CH_3NO) e Acetamide (C_2H_5NO)]. Em ambos os casos, foram utilizados 38 pontos de dados experimentais de cada propriedade termodinâmica. Esses dados foram retirados da publicação de Perry *et al.* [9].

6 Resultados e análise

Na Figura 1, são mostrados os *speedups* da versão paralela em relação à versão sequencial do método baseado na meta-heurística MOPSO projetado para o processo de estimação de parâmetros dos fluidos Formamide (CH_3NO) e Acetamide (C_2H_5NO). Com ambos os fluidos, foi estabelecido NP=120 e N variando de 1 a 6 e 8, 10 e 12. Isso para que cada núcleo tivesse a mesma quantidade de indivíduos da população. Pelo mesmo motivo, foi realizado um experimento com NP=110 e N estabelecido com 1, 2, 5, 10 e 11 com o fluido Formamide (CH_3NO). Esse experimento foi realizado para se verificar se com 11 núcleos seria possível obter um *speedup* maior do que com 10 núcleos, o que não ocorreu.

Na Figura 1, são mostrados os *speedups* obtidos com a versão paralela do método. Como a máquina em que os experimentos foram realizados é composta de dois processadores com seis núcleos cada e por causa da forma em que foi realizada a paralelização do método, as execuções até seis núcleos da versão paralela do método obteve *speedup* quase

DOI: 10.5540/03.2020.007.01.0461

linear em relação à quantidade de núcleos utilizada. Nas execuções com oito núcleos, o *speedup* alcançado ainda foi significativo e os maiores *speedups* foram obtidos com 10 núcleos (8x para a substância Formamide e 7,23x para Acetamide). Com mais de 10 núcleos, os *speedups* obtidos foram menores do que com 10 núcleos. Isso ocorre porque, com mais de 10 núcleos, todos os núcleos passam a estar ocupados, já que também há execução de processos do sistema operacional.

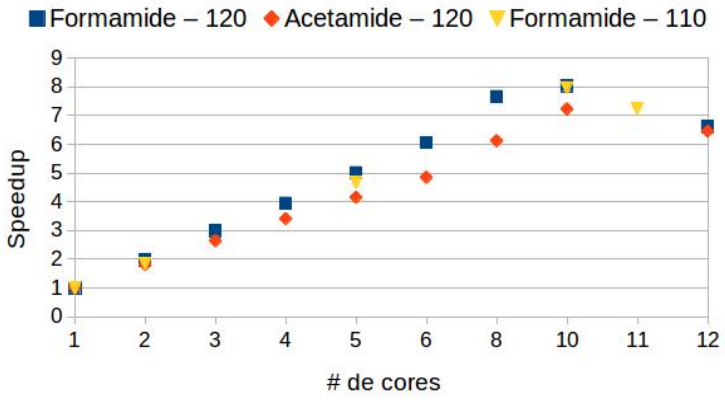

Figura 1: Speedups da versão paralela de método baseado na meta-heurística MOPSO projetado para estimação de parâmetros.

Na Figura 2, são mostrados os tempos de execução dos experimentos, com NP=120, retratados na Figura 1. Em particular, na execução do método com a substância Formamide, o tempo da execução sequencial foi de 80,5 minutos e o tempo da versão paralela com 10 núcleos foi de 10 minutos.

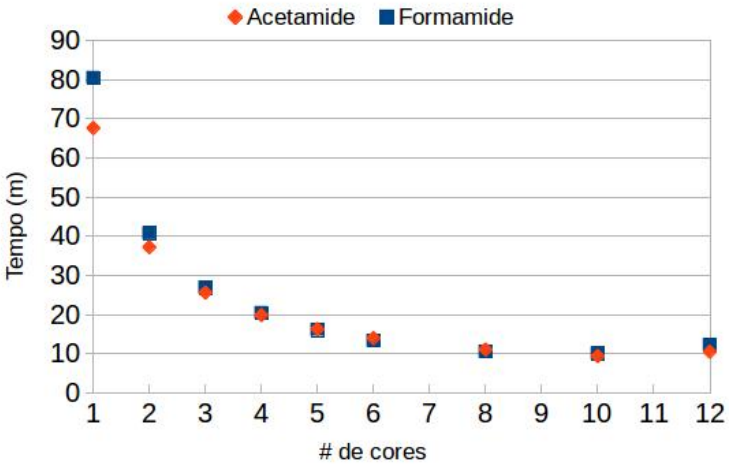

Figura 2: Tempos das versões sequencial e paralela de método baseado na meta-heurística MOPSO projetado para estimação de parâmetros com NP=120.

DOI: 10.5540/03.2020.007.01.0461

7 Conclusões

Neste trabalho, foram mostrados resultados de uma implementação paralela, utilizando o padrão OpenMP, de um método baseado na meta-heurística MOPSO aplicado em um problema de estimação de parâmetros de um modelo termodinâmico. As simulações foram realizadas com duas substâncias puras com 38 pontos de dados experimentais. O método paralelo obteve um *speedup* máximo de 8x e 7,23x, para duas substâncias, em relação à versão sequencial do método. Para se alcançar esses *speedups*, foram utilizados 10 núcleos da máquina.

Em trabalhos futuros, pretendemos utilizar a tecnologia *hyperthreading* em co-processador com micro-arquitetura Intel® Many Integrated Core para verificar a possibilidade de alcançar um speedup maior do que o obtido neste trabalho. Além disso, pretende-se investigar diferentes políticas de vinculação de *threads* para examinar a possibilidade de ganhos de desempenho da implementação paralela do método.

Pelo nosso conhecimento, não há abordagem do problema, de estimação de parâmetros de modelos termodinâmicos por otimização multiobjetivo, na bibliografia especializada. Dessa forma, também pretendemos comparar a metodologia proposta, tanto em sequencial, quanto em paralelo, com o método DE (Differential evolution), para avaliarmos se a metodologia proposta é realmente uma ferramenta válida para esse tipo de problema.

Agradecimentos

O presente trabalho foi realizado com apoio da Coordenação de Aperfeiçoamento de Pessoal de Nível Superior (CAPES) com código de financiamento 001. Os autores agradecem a disponibilidade do computador utilizado nas simulações realizadas neste trabalho à Fundação de Amparo à Pesquisa do Estado do Rio de Janeiro (FAPERJ), que financiou o projeto denominado *Experimentos e Simulações de Processos de Destilação em Alambiques: Um Projeto Voltado para a Melhoria da Qualidade da Cachaça da Região Centro Norte Fluminense*, por meio do edital 33/2012-DCTR – Apoio ao Desenvolvimento Científico e Tecnológico Regional no Estado do Rio de Janeiro no processo de número E-26/112.575/2012.

Referências

[1] Abbott, M. M., & Van Ness, H. C. (1989). *Schaum's outline of theory and problems of thermodynamics*. McGraw-Hill.

[2] Chandra, R., Dagum, L., Kohr, D., Menon, R., Maydan, D., & McDonald, J. (2001). *Parallel programming in OpenMP*. Morgan Kaufmann.

[3] Coello, C. A. C., Pulido, G. T., & Lechuga, M. S. (2004). Handling multiple objectives with particle swarm optimization. *IEEE Transactions on evolutionary computation, 8*(3), 256-279.

[4] Hadamard, J. (1923). *Lectures on Cauchy's problem in linear partial differential equations* (Vol. 37). Yale University Press.

[5] Hussain, M. M., & Fujimoto, N. (2018, July). Parallel Multi-Objective Particle Swarm Optimization for Large Swarm and High Dimensional Problems. In *2018 IEEE Congress on Evolutionary Computation (CEC)* (pp. 1-10). IEEE.

[6] Li, J. Z., Chen, W. N., Zhang, J., & Zhan, Z. H. (2015, December). A parallel implementation of multiobjective particle swarm optimization algorithm based on decomposition. In *2015 IEEE Symposium Series on Computational Intelligence* (pp. 1310-1317). IEEE.

[7] McDougall, R., & Nokleby, S. (2009, August). On the application of multi-objective parallel asynchronous particle swarm optimization to engineering design problems. In *ASME Design Engineering Technical Conferences and Computers in Engineering Conference* (pp. 1-9).

[8] Patel, N. C., & Teja, A. S. (1982). A new cubic equation of state for fluids and fluid mixtures. *Chemical Engineering Science, 37*(3), 463-473.

[9] Perry, R. H., Green, D. W., & Maloney, J. O. (1997). Perry's handbook of chemical engineering. *Perry's Handbook of Chemical Engineering.*

[10] Rodríguez-Gallegos, C. D., Singh, J. P., Yacob Ali, J. M., Gandhi, O., Nalluri, S., Kumar, A., & Panda, S. K. (2019). PV-GO: A multiobjective and robust optimization approach for the grid metallization design of Si-based solar cells and modules. *Progress in Photovoltaics: Research and Applications, 27*(2), 113-135.

[11] Yuhui, W., Xiaohui, L., Yunzhong, J., & Xinshan, S. (2010, July). Parallelization and Performance Test to Multiple Objective Particle Swarm Optimization Algorithm. In 2010 *International Forum on Information Technology and Applications* (Vol. 1, pp. 216-223). IEEE.

