UTILIZANDO A TÉCNICA DE BUSCA EM PROFUNDIDADE PARA RESOLVER O PROBLEMA DE SEQUENCIAMENTO DE TAREFAS EM MÁQUINAS PARALELAS

Vívian Ludimila Aguiar Santos[1]; Thales Francisco Mota Carvalho[2]

Resumo: Este trabalho estuda um problema de sequenciamento de tarefas em máquinas paralelas considerando que as tarefas causam certos desgastes das máquinas e tem como objetivo reduzir o tempo final de processamento das tarefas. Para resolução baseia-se no problema de partição de números e utiliza-se um algoritmo de busca em profundidade.

Palavras–chave: Sequenciamento de tarefas. Algoritmos de Busca. Partição de Números.

Introdução

Pode-se perceber o quanto a oferta de materiais e produtos tem crescido nos últimos anos. Assim como, notar que a globalização e a ampliação de vendas pela internet transformou a concorrência de local para mundial. Isso provocou desafios decisivos para que as organizações consigam permanecer firmes no mercado de vendas e sobrevivam em um ambiente de intensa competição. Desta forma, tornou-se necessário às empresas atender as crescentes exigências por parte dos consumidores, como por exemplo, melhor qualidade dos produtos, maior variação de modelos, entregas mais confiáveis e rápidas e menores custos. Para auxiliar as empresas industriais a continuarem engajadas em mercados de dimensões mundiais surgiram os eficientes sistemas de planejamento e controle da produção. É neste contexto que se inserem os problemas de sequenciamento de tarefas os quais visam reduzir o tempo de processamento final em cada máquina, assegurando a alocação eficiente dos recursos, diminuição nos custos da produção, atendimento das datas combinadas para entrega dos produtos, e velocidade nos processos industriais. O problema abordado neste trabalho consiste em processar *n* tarefas em *m* máquinas levando em conta que cada tarefa executada provoca um certo desgaste na máquina. Este desgaste irá diminuir o desempenho da máquina, aumentando o tempo de processamento da tarefa seguinte a ser efetuada. Cada máquina pode processar somente uma tarefa de cada vez e não pode ficar ociosa até a última tarefa atribuída ter sido finalizada. Neste problema busca-se minimizar o máximo tempo de conclusão do processamento das tarefas, também conhecido como *makespan* (RUIZ-TORRES et al., 2013). Além disso, verificou-se a semelhança do presente problema com os problemas de partição de número. Desta forma, para resolução é proposto tratá-lo

1 Docente do IFNMG, Campus Pirapora. Departamento de Informática. E-mail: vivian.santos@ifnmg.edu.br

2 Mestrando em Ciência da Computação, Universidade Federal de Viçosa. E-mail: thales.mota@ufv.br

através de algoritmos de busca em árvore, pois é uma das formas de solucionar as partições de números.

Material e Métodos

De acordo com Korf (2013), o problema de partição de número consiste em dividir um determinado conjunto de números inteiros em uma coleção de subconjuntos, a fim de que a soma dos números em cada subgrupo tenha o valor mais próximo possível. Percebeu-se que este problema é similar ao problema abordado neste trabalho, no qual deseja-se alocar tarefas às máquinas de forma que minimize o tempo total de processamento de cada máquina, buscado assim que a diferença dos tempos entre as máquinas seja a menor possível também. Dessa forma busca-se resolver o presente problema da mesma forma que o autor Korf (2013) solucionou o problema de partição de números, visto que os resultados apresentados por ele são muito animadores. Para isto foi desenvolvido um algoritmo de busca em profundidade e também aplicadas algumas regras de poda que podem melhorar a eficiência do algoritmo. O algoritmo de busca escolhido é o *backtracking*, esta técnica representa um refinamento da busca por enumeração exaustiva, no qual boa parte das soluções pode ser eliminada sem serem explicitamente examinadas. Considere o exemplo a seguir que contém 2 máquinas e 4 tarefas. A Figura 1 apresenta a execução do algoritmo para um dos lados da árvore. Em cada nó é inserido uma tarefa e verificado o novo tempo final de processamento para a máquina. A mesma tarefa é incluída nos dois lados das ramificações de um nó, ou seja, nas duas máquinas. Isto ocorre para que seja verificada qual máquina terá o menor tempo final de processamento após a adição desta tarefa. Os nós folhas, isto é, os últimos nós da ramificação contém uma solução completa, onde todas as tarefas já foram atribuídas. Nota-se na Figura 1 que os nós que estão destacados representam a sequência para a melhor solução encontrada até o momento para este exemplo, na qual o *makespan* tem o valor de 21,3 e a diferença entre os valores de tempo final de processamento entre as máquinas é de 0,2. Vale ressaltar que já se conhece a ordem em que as tarefas são alocadas na máquina por uma relação definida por Ruiz-Torres et al. (2013), diminuindo assim as ramificações da árvore de busca.

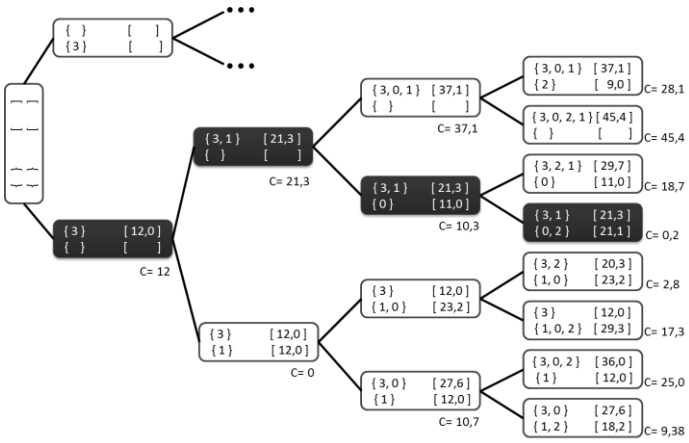

Figura 1: árvore de busca parcial do *backtracking*.

Resultados e Discussão

Uma das melhores medidas de eficiência de um algoritmo de *backtracking* é o número de nós na árvore gerado pelo algoritmo (KONDRAK e BEEK, 1997). Foram implementadas duas regras para tornar o algoritmo eficiente, visitar um número menor de nós e gastar menos tempo para ser executado. Através dos resultados observou-se que as regras realmente tornam o algoritmo eficaz ao comparar o número de nós visitados quando o algoritmo realiza podas e quando não realiza. Em uma instância que contém 20 tarefas notou-se que com as podas a quantidade expandida é de menos de 500 mil. No entanto, se não há podas, são gerados cerca de 2 milhões. A quantidade de nós é em média 15 vezes maior quando não há poda. Esta proporção de diferença entre os nós é maior ainda para as instâncias de 25 tarefas, na qual é cerca de 25 vezes superior o número de nós se não há poda e aproximadamente 37 vezes nós a mais para as instâncias de 30 tarefas que não utilizam regras de corte. Verificou-se também o quanto o algoritmo tornou-se mais rápido com a aplicação dos cortes. Por exemplo para efetuar as instâncias de 20 tarefas não é necessário nenhum segundo para executar cada uma delas. Ao contrário do que acontece quando não existem podas em que são gastos entre 19 e 55 segundos.

Conclusões

Apresentou-se o algoritmo de busca em árvore para a resolução do problema de sequenciamento de tarefas em máquinas paralelas. Para resolução, baseou-se no problema de partição de números e foram aplicados critérios de podas para melhorar o desempenho do algoritmo. Através dos resultados verificou-se que os cortes tornam o algoritmo mais eficiente, diminuindo o número de nós visitados e consequentemente o tempo gasto para encontrar a solução ótima. Contudo, percebe-se que para instâncias com mais 30 tarefas não é viável utilizar a técnica do *backtracking*, mas sim as heurísticas como apresenta Ruiz-Torres et al. (2013).

Referências

Kondrak, Grzegorz e Beek, Peter Van. (1997). A theoretical evalution of selected backtracking algorithms. Artificial Intelligence (89); 365-387.

Korf, R. E. (2013) Multy-Way Number Partitioning. International Joint Conference on Artificial Intelligence (IJCAI -13); 538 – 543.

Ruiz-Torres A. J., Paletta G., Pérez E. (2013) Parallel machine scheduling to minimize the makespan with sequence dependent deteriorating effects. Computers & Operations Research 40; 2051-2061.

Agradecimentos

Os autores agradecem o apoio do IFNMG e da CAPES.