Análise de um algoritmo aproximativo para o problema de escalonamento de tarefas com restrições de precedência em máquinas paralelas idênticas

Elton Lever, Omar Vilca[1], Rosiane de Freitas[1]

[1]Instituto de Computação – Universidade Federal do Amazonas (UFAM)
Manaus, AM – Brasil

{elton,omarlatorre,rosiane}@icomp.ufam.edu.br

Abstract. *This paper presents an unit time precedence constrained scheduling, considering a variable number of identical parallel machines, which is NP-Hard. For this problem, the best approximation ratio was provided by Coffman and Graham's algorithm (1972),* $2 - \frac{2}{m}$*, with* $m \geq 3$ *parallel machines, which has long been the best ratio until recently an approximation algorithm that guarantees schedules which are at most* $2 - \frac{7}{3m+1}$ *factor as long as the optimal, was proposed by Gangal and Ranade (2008), providing a small improvement but with interesting theoretical concepts, whose analysis is presented.*

Resumo. *Neste artigo é apresentado o problema de escalonamento de tarefas com tempos de execução unitários e restrições de precedência, considerando um número variável de máquinas paralelas idênticas, que é NP-difícil. Para este problema, a melhor razão de aproximação era fornecida pelo algoritmo de Coffman e Graham (1972), de fator de aproximação* $2 - \frac{2}{m}$*, para* $m \geq 3$ *máquinas paralelas idênticas, e que perdurou por um longo tempo até que recentemente um algoritmo de fator de aproximação* $2 - \frac{7}{3m+1}$ *foi proposto por Gangal e Ranade (2008), proporcionando uma melhoria pequena mas significativa e com conceitos teóricos interessantes, cuja análise é apresentada.*

1. Introdução

O escalonamento de tarefas com tempos de execução unitários e restrições de precedência, considerando máquinas (ou processadores) paralelas idênticas, consiste em um importante problema de otimização combinatória com aspectos de complexidade computacional de grande interesse. Deste modo, dadas $J = \{J_1, J_2, ...J_j, ..., J_n\}$ tarefas de tempos unitários de processamento, $p_j = 1$, a serem escalonadas em $|P|$ máquinas paralelas idênticas, onde as tarefas $|P| = n$ possuem relações de precedência entre si ($prec$), deseja-se escalonar as n tarefas nas m máquinas de modo a minimizar o *makespan*, ou seja, minimizar o tempo de completude (C_j) da última tarefa a ser escalonada. Usando a notação clássica de 3-campos [Graham et al. 1979], tem-se $P_m|prec, p_j = 1|C_{max}$, ou somente $P|prec, p_j = 1|C_{max}$, para o caso onde o número de máquinas for parte da entrada, ou seja, se tiver um número variável de máquinas, provado ser NP-difícil [Ullman 1975]. Se o número de máquinas for fixo, com $|P| = k$, então, o problema ainda está em aberto e corresponde ao oitavo problema da lista de 12 problemas em aberto publicada no livro *Computers and Intractability* [Garey and Johnson 1979], da qual apenas este problema de escalonamento e o problema de isomorfismo em grafos, permanecem em aberto.

O problema de interesse deste trabalho é o de escalonamento em um número variável de máquinas, para o qual existem diversos algoritmos aproximativos, sendo por muito tempo o melhor deles o proposto por Coffman e Graham, de razão de aproximação $2 - \frac{2}{m}$, para $m \geq 3$ máquinas paralelas idênticas [Coffman and Graham 1972]. Recentemente, um algoritmo de fator de aproximação $2 - \frac{7}{3m+1}$ foi proposto [Gangal and Ranade. 2008], proporcionando uma melhoria pequena mas significativa e com conceitos teóricos interessantes. Este algoritmo é analisado no restante deste artigo.

2. Algoritmo aproximativo para $P|prec, p_j = 1|C_{max}$

Nesta seção serão brevemente apresentados os principais algoritmos aproximativos para $P|prec, p_j = 1|C_{max}$. O algoritmo pioneiro foi proposto em 1961 [Hu 1961], e mostrou que resolve o problema $P|prec, p_j = 1|C_{max}$ em tempo polinomial quando as restrições de precedência das tarefas são árvores enraizadas (*in-tree*, *out-tree*, *in-forest* e *out-forest*). Para tais casos, a complexidade do algoritmo é da ordem de $O(|V|log|V|)$ [Papadimitriou and Yannakakis 1979], onde $|V|$ é o número de vértices, assim o algoritmo oferece um escalonamento ótimo quando $|P| = 2$, porém, para $|P| \geq 3$ o algoritmo dista $2 - \frac{1}{|P|-1}$ do ótimo [Feng 1975]. Em 1972, o algoritmo de Hu foi refinado, ficando conhecido como algoritmo CG [Coffman and Graham 1972], cuja complexidade ficou $O(|V| + |E|)$. Lam & Sethi mostraram que a razão de aproximação do algoritmo de CG quando $|P| \geq 3$ é de $2 - \frac{2}{|P|}$, onde $|P|$ é o número de processadores. Em 1994, foi corrigido um erro na análise de Lam & Sethi e deram com melhoria no limite inferior gerado [Braschi and Trystram 1994]. Trinta e cinco anos mais tarde, em 2008, um algoritmo com melhor aproximação do ótimo que todos os anteriores foi proposto [Gangal and Ranade. 2008] e será apresentado seguir.

2.1. Algoritmo aproximativo de Gangal & Ranade(GR)

O algoritmo de melhor razão de aproximação para o problema $P|prec, p_j = 1|C_{max}$ foi proposto em 2008 [Gangal and Ranade. 2008], onde foi estabelecida uma razão de aproximação de $2 - \frac{7}{3|P|+1}$, onde $|P|$ é a quantidade de processadores para o caso de $|P| > 3$, portanto, melhor que os seus antecessores.

Os passos detalhados do algoritmo são apresentados a seguir (Algoritmo 1). Cada vértice folha será calculado pelo caminho do comprimento mais longo no grafo. N é a quantidade de sucessores de v no caminho mais longo até suas folhas. L é o número de arcos no caminho de v até u, $|P|$ são os processadores e D o maior prazo do caminho. As colunas representam o tamanho do tempo no escalonamento. A equação (1) é utilizada para calcular os prazos $D(v)$ para cada vértice v no sentido inverso da ordem de classificação na ordenação topológica, começando pelas folhas das árvores em que $N(v, D, L)$ é o maior conjunto de vértices (v, u) e, será calculado o menor prazo, tendo um caminho de comprimento pelo menos $L+1$, conectando v até u com $D(v) \leq D$, onde D é o prazo do caminho mais longo no grafo direcionado. O alcance máximo de valores inteiros que $L, D, |N(v, D, L)|$ pode tomar é $|V|$, contudo, a complexidade de tempo do algoritmo é $|V|^4$ para calcular todos prazos.

$$D(v) = D - L - \lceil \frac{N(v, D, L)}{P} \rceil \tag{1}$$

Algoritmo 1: Algoritmo - (Gangal & Ranade)

Entrada: Grafo $G(V, E)$, P: Conjunto de processadores.
inicio
  1 - Inclua um vértice a com arestas $\forall v \in V(G)$
  para cada *vértice $v \in V(G)$* **hacer**
    se *v é vértice sem sucessores* **então**
      Seja $D(v) = \Delta$, onde Δ é um número qualquer, $D(v) =$ comprimento do caminho mais longo em G;
    senão
      Calcule os prazos $D(v)$ para os outros vértices de acordo com a Equação (1);
    fim se
  fin para cada
  2 - Ordene os vértices em ordem não decrescente de acordo com seu prazo, realizando a redução de transitividade ;
  3 - Escalone a no tempo 1 do processador 1;
  para cada *vértice $v \in V(G)$* **hacer**
    enquanto *Reorganiza predecessores (v)* **faça**
      1. Seja t' uma coluna com maior prazo, tal que contém alguns predecessores de v;
      2. Seja A o conjunto de vértices na coluna de $t' - 1$ e coluna t';
      se *todos os vértices em A tiver o mesmo prazo, e A tem no máximo $|P|$ predecessores de v, e coluna t' tem menos de $|P|$ vértices* **então**
        - Mova os predecessores de v para a posição $t' - 1$;
        - Mova outros vértices da coluna $t' - 1$ para a coluna t' se for necessário;
      fim se
    fim enqto
    Seja t o menor tempo em que v pode ser escalonado;
    Escalone v na coluna t livre do primeiro processador disponível.
  fin para cada
fin

Primeiramente, um vértice artificial a é criado e conectado com todos os outros vértices, e se estima prazos $D(v)$ para todos os vértices v, que define o comprimento do caminho mais longo no grafo (G). Os prazos dos outros vértices são, então, calculados utilizando-se a equação (1). Depois de calculado os prazos, o algoritmo realiza uma redução de transitividade no grafo. Tal processo, com a reorganização dos predecessores, onde os vértices que estão na coluna $t' - 1$, com t' tendo algum prazo, não afeta as restrições de precedências, pois as tarefas estão prontas para serem escalonadas na quantidade de processadores disponíveis naquele momento. Se todos os antecessores de v se moverem para a coluna de $t' - 1$, em seguida, v é escalonado na coluna t'. O benefício deste rearranjo é exemplificado na Figura 1, onde é demonstrado que a reorganização dos predecessores no primeiro bloco não é possível, no entanto, no segundo bloco da instância do exemplo percebe-se que o tempo é quebrado, e isto ocorre devido a reorganização das tarefas seguintes. A tarefa j que iniciava no tempo 7 foi para o tempo 6. Por consequência, a tarefa g que estava iniciando no tempo 6 agora inicia no tempo 7. Com isto, as tarefas sucessoras de j passam a ser {n,p,o} e, consequentemente, z diminue um período de tempo, para o inicio da execução. As tarefas l e n podem iniciar no tempo 7, pois, a quantidade de tarefas é menor que de processadores. Como a tarefa i é uma folha e tem baixa prioridade, inicia no tempo 8. A tarefa p vai para o tempo 7 e, consequentemente, z vai para o tempo 8 (terminando assim o escalonamento). Pode-se perceber claramente

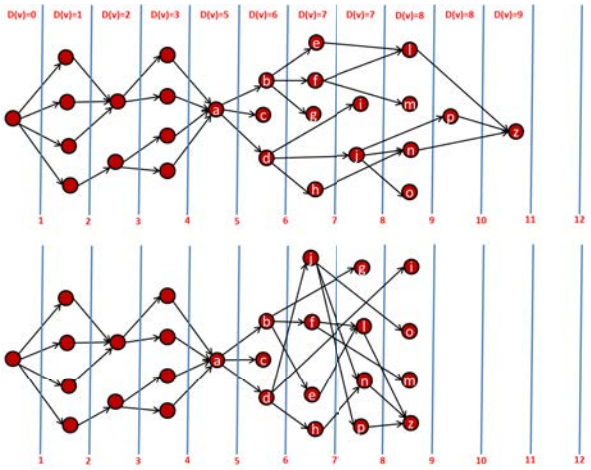

Figura 1. Alg. GR, com reorganização dos predecessores para $|P| = 4$.

que após tal reorganização, o tempo diminui de 11 para 9, mas 9 é o comprimento do caminho mais longo no grafo e, portanto, é um limite inferior. Assim a Figura 1 mostra o escalonamento ótimo para este grafo.

3. Considerações Finais

Neste trabalho foi abordado o algoritmo de melhor razão de aproximação para $P|prec, p_j = 1|C_{max}$, sendo de fator $2 - \frac{7}{3|P|+1}$, considerando $|P| > 3$. É ótimo para 2 máquinas e, para 3, mantém o fator de $2 - \frac{2}{m}$, dado pelo algoritmo de CG.

Referências

Braschi, B. and Trystram, D. (1994). A new insight into the coffman-graham algorithm. *SIAM J. Comput.*, 23:662–669.

Coffman, E. and Graham, R. (1972). *Optimal Scheduling of coffman-graham algorithm for a new order class*. Acta informatica, 1:200-213.

Feng, T., editor (1975). *Parallel Processing, Proceedings of the Sagamore Computer Conference, August 20-23, 1974*, volume 24 of *Lecture Notes in Computer Science*. Springer.

Gangal, D. and Ranade., A. (2008). Precedence constrained scheduling in $(2 - \frac{7}{3p+1})$ para $p \geq 4$ optimal. *Elsevier. Journal of Computer and System Sciences, 74 (2008) 1139-1146.*, Vol:1:1–13.

Garey, M. R. and Johnson, D. S. (1979). *Computers and Intractability: A Guide to the Theory of NP-Completeness*. W. H. Freeman.

Graham, R. L., Lawler, E. L., Lenstra, J. K., and Kan, A. H. G. R. (1979). Optimization and approximation in deterministic sequencing and scheduling: a survey. *Annals of Discrete Mathematics*.

Hu, T. C. (1961). Parallel sequencing and assembly line problems. *IBM Research Center, Yorktotim, New York*, 1(1):8.

Papadimitriou, C. and Yannakakis, M. (1979). *Scheduling interval-ordered tasks*. SIAN J. Comput., 8(3): 405-409.

Ullman, J. (1975). Np-complete scheduling problems. *Journal of Computer and System Sciences*, 10(3):384 – 393.