Aplicação da Abstração *MapReduce* na Paralelização de Procedimentos de Otimização

Sabir Ribas, Igor Machado Coelho, Mário Henrique de Paiva Perché
Marcone Jamilson Freitas Souza, David Menotti Gomes
Universidade Federal de Ouro Preto - Departamento de Ciência da Computação
35400-000, Campus Morro do Cruzeiro , Ouro Preto, MG
E-mail: {sabir,imcoelho,mariohpp,marcone,menotti}@iceb.ufop.br .

RESUMO

O crescimento da demanda de sistemas computacionais de auxílio à tomada de decisões acompanha o aumento da complexidade das cadeias produtivas. Tais sistemas normalmente estão relacionados a proporcionar uma visualização objetiva e clara da situação de uma linha de produção, à resolução de um gargalo nos processos de um sistema produtivo ou à melhoria de qualidade de algum produto.

Para resolver problemas combinatórios destacam-se duas abordagens, métodos baseados em heurísticas e técnicas exatas. A vantagem de se adotar metodologias exatas é a garantia da obtenção das soluções ótimas para o problema, o que não acontece no caso de heurísticas. Porém o uso de heurísticas é muito atraente quando a prioridade é o tempo de resposta. Heurísticas são métodos de busca geralmente dotados de alguma inteligência e apresentam boas soluções em um tempo relativamente baixo se comparados com os métodos exatos. Dessa forma, métodos heurísticos passaram a ser vastamente empregados na resolução de problemas reais.

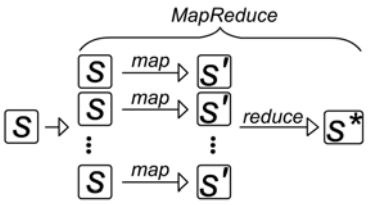
O fato é que as heurísticas e os métodos matemáticos, em suas formas originais, não exploram a capacidade dos processadores e das arquiteturas atuais. O tempo de resolução de problemas de otimização poderia ser reduzido se os algoritmos utilizassem os recursos *multi-core* desses processadores assim como versões de sistemas operacionais especializados em ambientes distribuídos e construção de *clusters* de maneira rápida, como é o caso das distribuições Linux PelicanHPC, Parallel Knoppix.

Como ferramenta de paralelização dos procedimentos, sejam eles heurísticos ou exatos, propõe-se o uso da técnica *MapReduce*. Trata-se de uma abstração simples e poderosa, geralmente aplicada ao processamento ou geração de grandes massas de dados. Dean e Ghemawat [2] apresentam uma visão geral sobre a implementação do *MapReduce* da Google a qual facilita muito o trabalho de seus programadores. Esse modelo foi feito para processar grandes conjuntos de dados de uma maneira massivamente paralela e é baseado nos seguintes fatores [1]: (i) iteração sobre a entrada; (ii) computação sobre cada um dos pares (*chave,valor*) da entrada; (iii) agrupamento de todos os valores intermediários por chaves; (iv) iteração sobre os grupos resultantes; (v) redução de cada grupo. O usuário da biblioteca *MapReduce* expressa a computação como duas funções: *map* e *reduce*. A função *map* recebe um par como entrada e produz um conjunto de pares intermediários também na forma (*chave,valor*). A biblioteca *MapReduce* agrupa todos os valores intermediários associados à mesma chave intermediária e os passa à função *reduce*. A função *reduce* aceita uma chave intermediária e o conjunto de valores relacionados aquela chave. Essa função junta esses valores para formar um conjunto possivelmente menor de valores. Tipicamente, zero ou apenas um valor é produzido pela função *reduce*.

Especificamente para problemas de otimização, onde os itens a serem mapeados não necessitam ser ordenados ou agrupados por chaves, pois o que interessa é a solução e uma forma de se medir sua qualidade, o modelo pode ser simplificado. Usuários especificam as funções *map* e *reduce*. A primeira é responsável por processar um item de um tipo α resultando em um item

do tipo β e a segunda mescla todos os dados intermediários, representados por um conjunto de elementos do tipo β, gerado a partir da aplicação da função *map* a um conjunto de dados do tipo α. A Figura 1 (a) apresenta uma forma de se aplicar tal abstração a problemas de otimização, onde os tipos α e β representam soluções (s e s') e a função *map* pode ser qualquer método de refinamento, o resultado da função *reduce*, s^*, é geralmente a melhor dentre as soluções intermediárias. Programas escritos nesse estilo podem ser automaticamente paralelizados.

Para testar a aplicação da abstração *MapReduce* na paralelização de procedimentos de otimização, este trabalho propõe um procedimento paralelo baseado na metaheurística *Iterated Local Search* [3]. Seu pseudocódigo é apresentado na Figura 1 (b) onde: s_0 é uma solução inicial; s^* é a melhor solução obtida; P é o conjunto de nodos de processamento; A e B são conjuntos de soluções; $s^{'}$ é a solução perturbada; e $s^{''}$ é, potencialmente, um ótimo local obtido pela aplicação da busca local à solução perturbada.



(a)

Procedimento Parallel Iterated Local Search

$s_0 \leftarrow$ GeraSoluçãoInicial
$s^* \leftarrow$ BuscaLocal(s_0)
repita
  $A \leftarrow |P|$ cópias de s^*
  $B \leftarrow \emptyset$
  para *cada* $s \in A$ **faça em paralelo**
    $s' \leftarrow$ Perturbação (s, *histórico*)
    $s'' \leftarrow$ BuscaLocal (s')
    Adicione s'' ao conjunto B
  fim
  $s^* \leftarrow$ CritérioAceitação (s^*, B, *histórico*)
até *critério de parada satisfeito*
return s^*

(b)

Figura 1: (a) Abstração *MapReduce* em problemas de otimização (b) Procedimento PILS

O procedimento PILS foi implementado em C++ usando a biblioteca MapReduce++. Tal biblioteca está sendo desenvolvida neste trabalho e, atualmente, possui recursos de paralelização tanto em *threads* quanto em rede. Seu principal objetivo é oferecer ao usuário todo aparato necessário para o desenvolvimento rápido de aplicações paralelas.

A fase atual do trabalho compreende a etapa de testes e análise de resultados. Os resultados esperados são (1) diminuição no tempo de processamento para se obter soluções de qualidades equivalentes e (2) aumento da qualidade das soluções finais. Centenas de programas baseados em *MapReduce* já haviam sido implementados na Google até 2004 [2], uma empresa altamente voltada à rapidez de resposta aos usuários de seus produtos, assim pode-se dizer que há uma boa chance desta abstração ser muito útil à paralelização de procedimentos de otimização.

Palavras-chave: *MapReduce, Otimização, Algoritmos Paralelos, Heurísticas*

Referências

[1] R. Lämmel, Google's MapReduce Programming Model, Microsoft Corp.

[2] J. Dean e S. Ghemawat, MapReduce: Simplified Data Processing on Large Clusters, In OSDI'04, 6th Symposium on Operating Systems Design and Implementation, Sponsored by USENIX, in cooperation with ACM SIGOPS, pg 137-150, 2004.

[3] H. R. Lourenço, O. C. Martin e T. Stützle, Iterated Local Search. In: F. Glover e G. Kochenberger, (Eds). Handbook of Metaheuristics, Kluwer Academic Publishers, Boston, 2003.