Estratégias de Paralelização de Algoritmo Genético para Problemas de Roteamento

Felipe Nathan Welter[1]

[1]Centro de Ciências Tecnológicas – Universidade do Estado de Santa Catarina (UDESC)
89.219-710 – Joinville – SC – Brazil

felipenwelter@gmail.com

Resumo. *A aplicação de algoritmo genético como técnica para solução do problema de roteamento de veículos permite a ampla aplicação de estratégias de paralelismo, assim como flexibilidade para diferentes ambientes computacionais. A partir de um estudo de caso realizado em uma mineradora, esse artigo tem por objetivo apresentar uma breve contextualização dos conceitos aplicados e realizar o comparativo com outros modelos de implementação a fim de se identificar possibilidades de melhoria.*

1. Introdução

O Problema de Roteamento de Veículos (VRP, do inglês Vehicle Routing Problem) é um dos problemas mais conhecidos em otimização combinatória. Sua solução consiste em encontrar o melhor conjunto de rotas a serem percorridas por uma frota de veículos atendendo à algumas restrições específicas tal como janela de tempo ou capacidade de carga. Considerando um problema da classe NP-difícil, a busca pela melhor resposta exige grande tempo de processamento e por esse motivo normalmente é solucionado por meio de um algoritmo probabilístico ou pela aplicação de heurísticas que encontram soluções boas o suficiente em tempo reduzido.

Na prática, o VRP pode ter uma série de restrições que o distingue em categorias mais específicas, assim como uma série de abordagens possíveis para solucioná-lo que variam desde métodos exatos como o algoritmo branch-and-bound, heurísticas construtivas ou ainda metaheurísticas como é o caso do Algoritmo Genético, solução adotada pelo autor no estudo de caso XXXXX.

https://www.dcc.ufmg.br/pos/cursos/defesas/1652M.PDF - explica diferentes tipos de problemas de roteamento e solucoes comumente aplicadas

2. Heurísticas e metaheurísticas

Métodos heurísticos são algoritmos exploratórios que buscam a solução de problemas. Em Ciência da Computação, o desenvolvimento de um algoritmo busca atender a duas propriedades: ter um tempo de execução aceitável e encontrar a solução ótima, ou provavelmente boa, para o problema em todos os casos. Um algoritmo heurístico, entretanto, não necessariamente atende uma dessas propriedades, de forma que não se pode garantir que consegue encontrar a melhor solução nem que o fará de maneira rápida em todas as situações [Bueno 2009].

3. Algoritmo Genético

O algoritmo genético é uma das técnicas da computação evolucionária de busca e otimização inspirada na evolução natural das espécies de Darwin. A idéia é criar uma população de indivíduos que vão se reproduzir e competir pela sobrevivência, onde os melhores indivíduos sobrevivem e transferem suas características para novas gerações, até que se alcance uma solução próxima ao resultado ótimo [Pozo et al. 2003]. O desenvolvimento de simulações computacionais de sistemas genéticos teve início na década de 50 e 60 a partir de biólogos e foi John Holland quem iniciou as primeiras pesquisas sobre o tema, publicando, em 1975, o livro *"Adaptation in Natural and Artificial Systems"*. Posteriormente se iniciariam aplicações industriais dos algoritmos genéticos com David E. Goldberg, aluno de Holland, e hoje esses algoritmos são utilizados para solucionar problemas de otimização e aprendizado de máquinas.

Em problemas de alocação dinâmica, a função de aptidão leva em consideração os seguintes fatores: Minimizar a Distância Média de Transporte, diminuir a formação de Alas e o reduzir a ociosidade das máquinas. (TORRES, 2017)

Antes de detalhar o comportamento do algoritmo genético propriamente dito, se faz necessária a compreensão da terminologia adotada para representar as entidades e processos envolvidos. Os termos, análogos aos da biologia, referem-se no algoritmo à construção das possíveis soluções ao problema abordado.

4. Estudo de caso + dificuldades

A operação da frota de mineração consiste no transporte do produto, a céu aberto, desde um ponto de carregamento onde o material é extraído até um ponto de descarregamento, para que se inicie o proceso de separação e transformação do material bruto. Um sistema de despacho e gerenciamento de frotas em mineração realiza o monitoramento e controle de indicadores como posição dos equipamentos, percurso, velocidade, carga entre diversos outros fatores, buscando otimizar o tempo e capacidade de operação reduzindo também os custos de manutenção dos equipamentos. O sistema SmartMine, por exemplo, é aplicado por grandes mineradoras, porém o investimento necessário e o alto custo de infraestrutura não favorecem sua aplicação por minas de menor porte.

O estudo de caso busca aplicar uma solução embarcada de menor custo focada especialmente na otimização de rotas, através da aplicação de um algoritmo genético paralelizado com modelo de ilhas através de programação OpenMP [Neto 2018]. Ainda segundo o autor, a aplicação e inteligência artificial é pouco explorada em sistemas embarcados e com o crescimento do IOT aliado ao baixo custo dos dispositivos, permite sua aplicacção de forma cada vez mais abrangente.

Cada veículo é equipamento com um sistema GPS que fornece suas cordenadas de latitude e longitude e a partir dessa informação é realizado o gatilho para cálculo de rotas em três momentos: fim do carregamento, fim do descarregamento e na região de cruzamento de rotas.

- OpenMP motivo.

- Trabalha com modelo de ilha (granularidade grossa) - Processamento local e tráfego de informações de localização com um servidor central

- Hardware utilizado: junto de hardwares embarcados ARM com Raspberry Pi 2 com 4 núcleos, ARM com Raspberry Pi 3 com 4 núcleos, ARM com Banana Pi com 8 Núcleos. - O sistema operacional do sistema embarcado auxiliar, onde é executado o algoritmo, é de plataforma linux, compilado para atender as necessidades básicas do sistema em ambos equipamentos. Esse sistema não teve nenhuma compilação especial, sendo que os processos executados como padrão são determinados pelas versões de seus fabricantes.

- Tamanho das minas: Mina de pequeno porte, que é caracterizada como de 5 a 40 equipamentos, 4 origens de carregamento de carga, 2 destinos e 8 cruzamentos. Mina de médio porte, com características de variáveis como de 1 a 110 equipamentos, 6 origens de carregamento de carga, 3 destinos e 32 cruzamentos. Mina de grande porte, com uma conAguração de 8 pontos de origem para carregamento, 62 cruzamentos de opções de rotas e com 5 opções de destino com uma variação de 1 a 170.

Limitação: Lembrando que a cerca é um círculo virtual de aproximadamente 30 metros de raio ao redor de um cruzamento e que o caminhão trafega a aproximadamente 35 km/h, o algoritmo genético precisa ser executado em um tempo máxi

5. Análise da abordagem e propostas de melhoria

Os resultados apresentados indicam que conforme o número de equipamentos aumenta, maior é o tempo de execução do algoritmo, mas também maior é o ganho da execução paralela em relação à sequencial. O hardware utilizado é um fator determinante e os melhores resultados foram obtidos no equipamento com maior número de núcleos, o BananaPI M3, alcançando-se speedups de 1.08, 1.30 e 1.20 em minas de pequeno, médio e grande porte respectivamente. Em determinadas situações não foi possível atender à restrição de tempo de processamento de até 1 segundo, o que não permite ao motorista a tomada de ação em tempo hábil.

Visando uma contribuição à proposta apresentada nesse estudo de caso, algumas considerações podem ser discutidas visando a resolução de algumas das limitações apresentadas ou ainda o aumento da eficiência da solução.

Um primeiro aspecto a se observar é em relação ao hardware dos dispositivos embarcados, que precisam ser compatíveis às restrições específicas do cenário apresentado e às limitações orçamentárias da empresa. A aquisição de equipamentos mais potentes, de gerações mais atuais e com maior número de núcleos de processamento poderia trazer um ganho considerável na performance e sem inviabilizar o custo do projeto. Esse ganho de performance foi comprovado pela análise comparativa entre os modelos RasperryPI 2 e 3 apresentado pelo autor, onde o speedup variou de 1,15 para 1,29 nas minas de grande porte. Nesse contexto uma outra alternativa seria a customização do hardware através do aumento do clock do CPU, ou ainda a adoção de compilações especiais para os sistemas operacionais, mantidos originais durante o estudo de caso.

Outro fator limitante pouco explorado pelo autor é a questão da latência de rede no processo de busca das informações das coordenadas dos outros veículos direto com o servidor central. Trantando-se de uma etapa obrigatória para obter um melhor de desempenho da função de avaliação fitness, uma alta latência poderia comprometer o processo como um todo. Nesse contexto, realizar a sincronização das informações em intervalos

regulares de forma assíncrona eliminaria a etapa de sincronização prévia na execução do algoritmo genético.

Além dos aspectos mais voltados ao próprio hardware e camada de rede, uma alteração na estratégia de paralelização em modelo de ilha aplicada no estudo seria, talvez, o aspecto com maior potencial de ganho. O algoritmo genético pode ser adequado para trabalhar em diferentes estruturas computacionais, entre eles: - Master-Slave: as operações de selection, crossover e mutation sáo realizadas pelo mestre, que distribui entre os nós filhos cada indivíduo para cálculo do fitness. Apresenta pouca variação de sua implementação sequencial, mas pode ser útil quando a função fitness conta com operações mais complexas envolvendo diferentes variáveis para atender as restrições de negócio. - Modelo de ilhas: a população é divida em subpopulações que evoluem de forma independente. Uma etapa de migração é incluída no processo para replicar os melhores de cada população entre as demais ilhas. A diferença entre o modelo síncrono e assíncrono é que no modelo síncrono a réplica de indivíduos ocorre de forma fixa após um número específico de gerações, enquanto no modelo assíncrono não há essa restrição. - Modelo em células: cada indivíduo é executado por um dispositivo independente. Para as operações de crossover e mutation há troca de informações entre nós vizinhos, de forma que a topologia de conexões é um fator determinante para o processo, mas que permite o maior nível de paralelismo (REZA). - Modelo hierárquico: permite a mescla de diferentes estruturas em camadas, o que permite grande flexibilidade para se adequar a diferentes ambientes computacionais. Um exemplo poderia trabalhar com o modelo de ilhas em uma população (através de threads) e ao mesmo tempo realizar a troca de informações com outros nós da rede, no modelo em células.

A escolha do autor em utilizar o modelo de ilhas é assertiva visto as especificadades do cenário de mineração. O modelo master-slave não seria uma boa escolha pois as regras de avaliação fitness não são tidas pelo autor como gargalo de processamento, e os modelos em célula e hierárquico teriam sua aplicação mais relacionada a ambientes multi-computadores e não multi-processadores. Entretanto, em relação ao modelo de ilhas, o autor deixa claro que ”a função de migração do algoritmo genético com ilhas para um número maior de threads executadas e pelo tamanho do indivíduo (cromossomo) já ocasiona perda de desempenho”(NETO). Como alternativa para esse cenário CHINES indica que ”withthesamesizeofpopulationandnumberof generations, the Asynchronous Island scheme should be fasterthantheSynchronousIslandone”. Na prática, os melhores indivíduos poderiam ser dispostos em uma memória compartilhada e conforme cada ilha concluísse a avaliação de sua população poderia utilizar os melhores indivíduos disponíveis até o momento, o que não acarretaria perdas de performance ou qualidade da solução.

De forma complementar ainda, a alteração dos parâmetros do algoritmo genético tal como o número de populações e a taxa de migração podem afetar significativamente a velocidade de processamento e a qualidade da solução, portanto precisam ser configurados de acordo com a realidade de cada caso, sugerindo-se ainda a avaliação a partir de experimentações práticas.

Para concluir, a otimização das rotas de transporte descrita no estudo de caso permite a redução de custos operacionais e aumenta a produtividade e eficiência do processo produtivo, tendo ainda um grande diferencial pelo pequeno custo de implantação através

de uma solução embarcada, ao invés das soluções robustas disponíveis no mercado, incompatível com a realidade orçamentária de pequenas mineradoras. A otimização do algoritmo genético, por sua vez, deve considerar não apenas o tempo de processamento, mas principalmente a qualidade da solução, como afirmam CHINESES.

Most work focuses exclusively on the relationship between the speedup on the multi-core and many-core architecture,pursuing a fast execution time, but ignoring the solution quality.Since GAs mostly find only approximate solutions,more attention should be paid to solution quality. The relationship between speedup and architecture should be discussed along with the solution quality (CHINESES)

artigo 1 [Neto 2018].

artigo 2 [Reza 2015].

artigo 3 [Zheng et al. 2014].

Referências

Bueno, F. (2009). *Métodos Heurísticos: Teoria e Implementações*. IFSC. Disponível em: http://bit.ly/30hooR9. Acesso em: 30 de junho de 2019.

Neto, G. D. (2018). Paralelismo de um algoritmo genético, aplicado a otimização de rotas em mineração de modo escalável. Master's thesis, Universidade Federal de Uberlândia, Uberlândia. Disponível em: https://repositorio.ufu.br/bitstream/123456789/25374/3/ParalelismoAlgoritmoGenetico.pdf. Acesso em: 16 out. 2019.

Pozo, A. et al. (2003). *Computação Evolutiva*. Disponível em: http://www.inf.ufpr.br/aurora/tutoriais/Ceapostila.pdf. Acesso em: 7 de julho de 2019.

Reza, Roshani; Karim Sohrabi, M. (2015). Parallel genetic algorithm for shortest path routing problem with collaborative neighbors. *Ciência e Natura*, 37:328 – 333. Disponível em: https://www.redalyc.org/articulo.oa?id=467547683042. Acesso em: 20 out. 2019.

Zheng, L. et al. (2014). Architecture-based design and optimization of genetic algorithms on multi- and many-core systems. *Future Generations Computer Systems*, 38:75 – 91. Disponível em: https://www.sciencedirect.com/science/article/pii/S0167739X13002082. Acesso em: 20 out. 2019.