APLICAÇÃO DE ALGORITMOS GENÉTICOS PARALELOS A PROBLEMAS DE GRANDE ESCALA DE *VMI – VENDOR MANAGED INVENTORY*

Nelson Standerski
Paperless P&D Ltda.
Av. Prof. Lineu Prestes 2242 - CIETEC - Cidade Universitária
05508-000 São Paulo SP Tel (11) 3039 8307
nelson@paperless.com.br

Resumo

Este trabalho mostra que técnicas de programação paralela e sistemas computacionais paralelos de baixo custo (*beowulf clusters*) podem ser aplicados com sucesso para solução de problemas logísticos de grande escala. Foi estudado um exemplo de reabastecimento ótimo de estoques com entrega direta, onde a programação de entregas é feita pelo fornecedor (VMI – *vendor managed inventory*).

A metodologia proposta utilizou a formulação clássica de reposição ótima de estoques e algoritmos genéticos paralelizados. O modelo desenvolvido aplica-se a entregas diretas com veículos de carga cheia (*full truckload*) para 4.000 clientes e horizonte de planejamento virtualmente infinito. O modelo foi programado em Fortran e MPI e rodado num *beowulf cluster* de 6 nós. Foram obtidas acelerações quase-lineares no tempo de processamento.

O desempenho computacional obtido e os baixos custos dos equipamentos viabilizam sua aplicação no planejamento diário de operações logísticas de grande escala.

palavras-chave logística, VMI, metaheurísticas, programação paralela, *beowulf cluster*.

Abstract

This paper shows that parallel programming techniques and low cost beowulf clusters can be applied to find practical solutions of large-scale logistics problems. It was studied the optimal direct delivery schedule where the vendor is responsible to monitoring the client's inventory (VMI – vendor managed inventory).

The proposed methodology used the classical formulation of optimal inventory re-supply, and parallel genetic algorithms. It was modeled a case with 4.000 customers and a virtually infinite time horizon. It was assumed full truckload vehicles. The model was programmed with Fortran and MPI, and was run on a beowulf cluster of 6 nodes. The resulting speedup was near linear.

The computational performance obtained makes it possible to apply this method to the daily planning logistics operations.

keywords logistics, VMI, metaheuristics, parallel programming, beowulf cluster.

1. Introdução

Há diversos problemas de otimização logística, de grande importância prática e econômica, mas cuja solução numérica é muito difícil. Por exemplo, o problema de reabastecimento ótimo de estoques envolvendo uma grande quantidade de locais e de restrições operacionais. Os estudos encontrados na literatura especializada para essa classe de problema (ver p.ex. [1] e [2]) tratam, entretanto, de modelos com uma quantidade pequena de variáveis e, quando aplicados a modelos de grande escala, não produzem soluções.

O estudo de reabastecimento ótimo não tem um método único de solução. Pequenas alterações na forma de cálculo da função objetivo podem levar a ferramentas de solução completamente diferentes. É usual o emprego de técnicas exatas de busca como MIP (*mixed-integer programming*) e seus derivados (*branch-and-bound*, *branch-and-cut* etc). Mas nem sempre se consegue boa aderência desses modelos a problemas reais de logística: muitas vezes as funções envolvidas não são lineares, ou nem mesmo têm uma descrição analítica explícita.

Mesmo admitindo-se que o problema possa ser modelado satisfatoriamente unicamente com funções lineares, inteiras ou binárias, algoritmos MIP podem não ser eficientes quando aplicados a modelos de grande escala. Isso ocorre porque o número de alternativas a serem analisadas cresce exponencialmente com o tamanho do modelo e, nessa situação, essas técnicas apresentam praticamente o mesmo desempenho de uma busca exaustiva. Adicionalmente nem sempre encontram uma solução viável inicial.

O objetivo principal deste estudo é desenvolver uma metodologia para busca de soluções ótimas, ou sub-ótimas, para problemas de grande escala de VMI. A metodologia proposta apresenta algumas simplificações no modelo de VMI, que entretanto não descaracterizaram a aplicação prática da solução.

Uma simplificação importante é a da entrega direta. Admite-se uma frota de veículos operando com carga cheia (*full truckload*) e atendendo a um único cliente em cada viagem. Essa operação é uma simplificação do problema geral, mas de grande importância prática na distribuição de derivados de petróleo e de gases industriais. No caso específico do Brasil, onde há grandes distâncias rodoviárias, os custos de transportes são significativos, e representam uma parcela significativa do custo final do produto.

Mesmo o problema simplificado de entrega direta ainda é um problema combinatório de difícil solução numérica. Recentemente computação de alto desempenho, técnicas de programação paralela e algoritmos de otimização têm sido aplicados na solução de uma série de problemas práticos em diversas áreas (manufatura, telecomunicações etc), e bons resultados têm sido obtidos (ver p.ex. [3]).

Beowulf clusters são sistemas de processamento paralelo baseados em componentes de micro-computadores de baixo custo. *Beowulf clusters* são aplicáveis a problemas altamente paralelizáveis, com relativamente pouca comunicação entre os processos (“modelos ilha”, “modelos de alta granularidade”). O conceito de software livre (*free software*, www.gnu.org) é também amplamente vantajoso neste caso, devido ao número de cópias de componentes software a serem instaladas num cluster de muitos nós.

O desenvolvimento de aplicações logísticas em ambiente de processamento paralelo é relativamente inédito no Brasil e mesmo no mundo. Somente recentemente estão aparecendo *solvers* comerciais com capacidade de processamento paralelo. Ainda são raras as publicações a respeito. Técnicas de programação paralela permitem não apenas maior rapidez nos resultados, o que é um fator crítico numa aplicação operacional, mas também soluções mais precisas. E também permite montar modelos mais complexos com grande quantidade de variáveis (clientes) e de restrições (janelas de tempo, capacidades de armazenamento e de transportes, horas de *rush* etc).

2. VMI - gerenciamento de estoques pelo fornecedor

VMI – *vendor managed inventory* – é uma prática onde os fornecedores são responsáveis pelo gerenciamento dos estoques de seus clientes, e devem garantir que nunca haja desabastecimento do produto. Essa prática tem-se desenvolvido bastante ultimamente devido ao barateamento dos custos de monitoramento remoto de estoques por telemetria.

Na operação convencional (sem VMI), o cliente coloca seus pedidos periodicamente, após consultar os seus estoques. Via de regra isso ocorre no início de cada semana ou mês, e usualmente quando os níveis de estoque já estão baixos ou mesmo nulos. O resultado é que o fornecedor recebe em determinados dias da semana ou do mês uma grande quantidade de pedidos a serem atendidos, e muitos com urgência. A operação convencional é uma situação normalmente crítica: usualmente ocorre desabastecimento no cliente, ou o fornecedor tem uma frota de veículos superdimensionada.

Com VMI os clientes não fazem pedidos. Seus estoques são estimados pelo fornecedor por telemetria ou por previsão estatística. Automaticamente, quando os níveis de estoque atingem determinado nível, o fornecedor realiza uma entrega. Dessa forma, o cliente nunca fica desabastecido e o fornecedor pode programar as suas entregas com antecipação, o que permite utilizar mais racionalmente a sua frota.

As indústrias onde o retorno econômico pela implantação de VMI é maior são aquelas onde os custos de distribuição (transportes, estoques, armazenagem) constituem parcela significativa do custo final do produto. Por exemplo, indústrias químicas, companhias petrolíferas, empresas de distribuição de derivados de petróleo e de gases industriais, grandes atacadistas e varejistas, incluindo redes de supermercados, indústrias de bebidas e alimentícias.

O problema de roteirização sem que haja desabastecimento nos clientes é conhecido por IRP – *inventory routing problem* – e é uma das ferramentas quantitativas básicas de VMI.

No IRP o fornecedor, que é responsável pela programação dos transportes, objetiva minimizar seus custos de transporte, garantindo entretanto que não haja desabastecimento nos clientes. A flexibilidade de decidir quando e quanto do produto deverá ser entregue pode reduzir substancialmente os custos de transporte, principalmente em se tratando de uma base grande de clientes.

A freqüência de atendimento é inversamente proporcional à quantidade entregue em cada visita. Alta freqüência de atendimento implica baixos volumes por visita e, portanto maiores custos de transporte. Inversamente uma baixa freqüência de atendimento leva a quantidades maiores por entrega, reduzindo os custos com transporte (veículo cheio).

Um caso particular de IRP, mas de grande interesse prático, é o IRPDD – *Inventory Routing Problem with Direct Deliveries* – problema de roteirização com reabastecimento e entregas diretas. Neste caso a entrega é feita unicamente a um cliente, utilizando um caminhão com carga cheia. IRPDD permite grandes simplificações no problema geral de IRP, e tem aplicação prática de grande importância na distribuição de derivados de petróleo e de gases industriais.

A entrega a um cliente pode também ser pensada como uma rota de entrega a um *cluster* de clientes com características semelhantes [1]. Assim o IRPDD pode ser visto como uma primeira etapa na solução de um IRP. Conhecido o momento em que o produto deverá ser entregue no *cluster* de clientes, uma segunda etapa faz a otimização da programação de entregas, levando em conta restrições operacionais dos veículos e dos locais. A definição de um *cluster* de clientes deve ser feita levando em conta as características geográficas e logísticas (capacidade de armazenamento, taxas de consumo do produto).

Outras hipóteses simplificadoras são admitidas que, entretanto não limitam o método propriamente dito: (a) modelos lineares de cálculo de custos de transporte e estoque, (b) único produto, (c) taxa de consumo do produto com distribuição uniforme, (d) tempos de carga e

descarga constantes, (e) sem janelas de tempo e de horas de *rush* e (f) único local de carregamento.

O período ideal de análise de um IRPDD é o longo prazo. Entretanto, quanto maior o período, maiores as incertezas estatísticas nos dados de entrada. A operação diária de distribuição logística tem natureza dinâmica: um veículo pode quebrar ou alguma taxa de consumo no cliente pode mudar abruptamente. Por isso é conveniente o modelo ser rodado periodicamente como dados atualizados.

Com base nesses dados, o modelo é formulado como um problema de otimização composto por uma função objetivo e uma série de restrições operacionais (velocidades, capacidade dos veículos, distâncias, estoques mínimo e máximo etc). Admite-se também que o tempo de viagem é linearmente proporcional à distância e independente do veículo e do período em que o trajeto é realizado.

A variação do nível de estoque em cada cliente está apresentada esquematicamente na Figura 1 abaixo. As funções lineares obtidas são resultados diretos da hipótese de uniformidade na taxa de consumo e da quantidade fixa transportada em cada veículo.

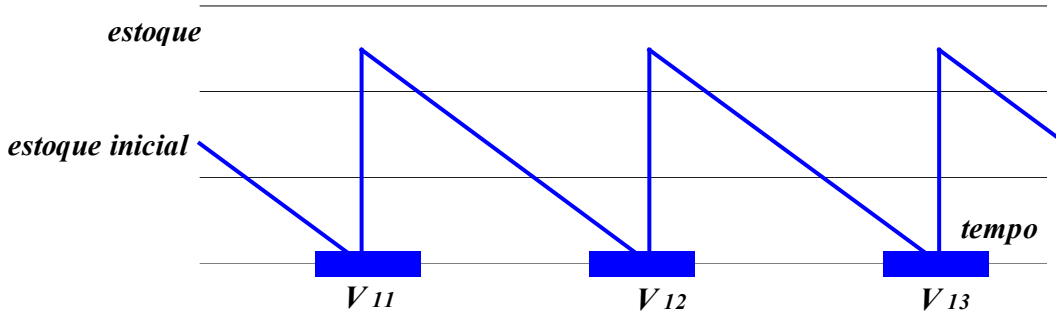


Figura 1 – Reabastecimento ótimo: Um local

O tempo ótimo de visita (estoque = 0) para um determinado cliente i é determinado pelas expressões:

$$\mathbf{V_{ij}} = \mathbf{I^0_i} / \mathbf{u_i}\,, \quad \text{para } j = 1$$

$$\mathbf{V_{ij}} = \mathbf{Q} / \mathbf{u_i} + \mathbf{V_{ij-1}}\,, \quad \text{para } j > 1$$

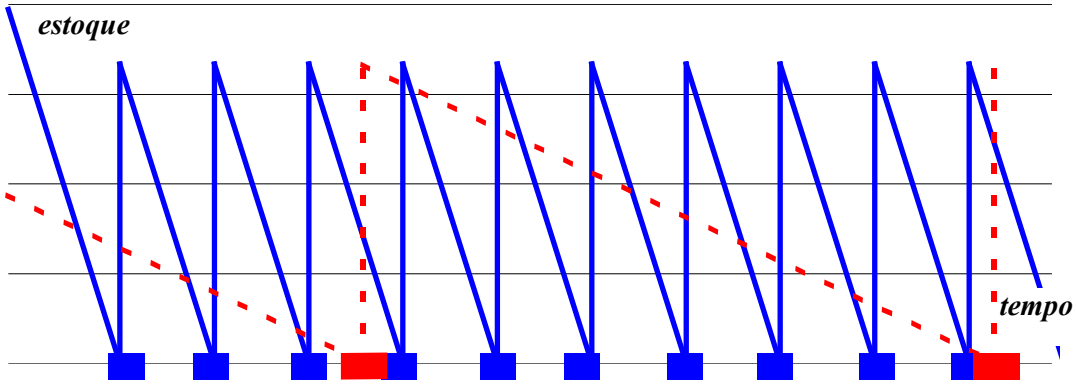
onde $\mathbf{V_{ij}}$ é o tempo da visita j no cliente i, $\mathbf{Q}$ a capacidade de transporte dos veículos, $\mathbf{u_i}$ a taxa de consumo e $\mathbf{I^0_i}$ o estoque inicial no cliente i.

A Figura 1 anterior também mostra o intervalo de tempo em que o veículo está em viagem. O tempo de viagem é admitido igual na ida e na volta, e é igual a distância do armazém até o cliente dividido pela velocidade média no percurso.

A Figura 2 a seguir mostra a variação no nível de estoques para 2 locais de reabastecimento.

Figura 2 – Reabastecimento ótimo: Dois locais

Observa-se na Figura 2 que nos dois locais as restrições de estoque estão sendo atendidas. Em pelo menos uma visita há sobreposição dos tempos em que os veículos estarão operando, e dois veículos devem ser necessários.



Do ponto de vista logístico seria muito interessante reduzir o número de veículos necessários, ou seja, que com apenas um veículo fosse possível realizar as programações de entregas dos dois locais, sem, entretanto ocasionar desabastecimento em nenhum local.

Como seria possível fazer isso ? Uma possibilidade seria o de antecipar algumas entregas, de forma a não se ter sobreposição no uso dos veículos. A antecipação da entrega está limitada superiormente pela capacidade máxima de armazenamento. A postergação da entrega não é possível, pois haveria desabastecimento.

A Figura 3 a seguir exemplifica o caso de antecipação da entrega para a redução do número de veículos.

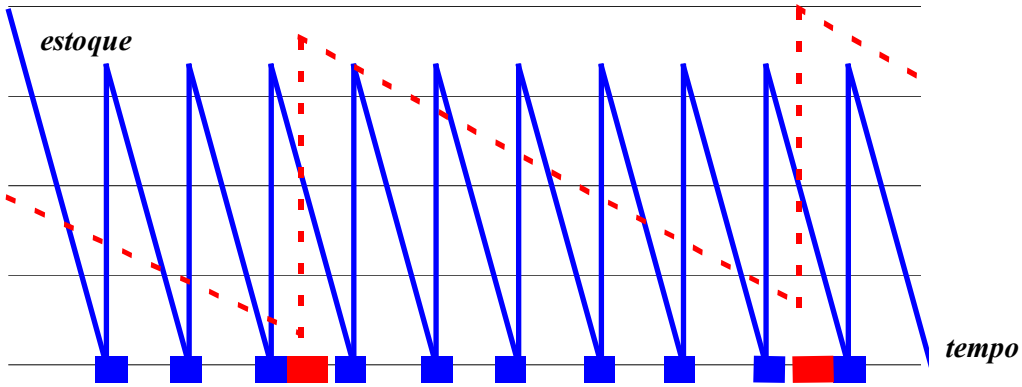


Figura 3 – Antecipação de reabastecimento para reduzir número de veículos

Assim o tempo de visita seria subtraído de uma variável a_{ij} para evitar sobreposição na utilização de veículos, resultando:

$$\mathbf{t_{ij} = V_{ij} - a_{ij}}$$

onde $\mathbf{t_{ij}}$ é o tempo de entrega j no cliente i.

A variável $\mathbf{a_{ij}}$ deve atender às seguintes restrições para todos os índices i e j:

$\mathbf{r_1}$ *Garantia de abastecimento nos clientes:*

$$\mathbf{a_{ij} \geq 0}$$

$\mathbf{r_2}$ *Capacidade máxima de armazenamento nos clientes*

$$\mathbf{a_{ij}\ u_i\ + Q\ \leq Cap_i \quad => \quad a_{ij}\ \leq\ (\ Cap_i - Q\ )\ /\ u_i}$$

onde $\mathbf{Cap_i}$ é a capacidade máxima de armazenamento no cliente i, $\mathbf{u_i}$ sua taxa de consumo e **Q** a capacidade de transporte dos veículos.

$\mathbf{r_3}$ *Integridade temporal da sequencia de entregas*

$$\mathbf{t_{ij-1}} \leq \mathbf{t_{ij}} \leq \mathbf{t_{ij+1}}$$

substituindo-se $\mathbf{t_{ij}} = \mathbf{V_{ij}} - \mathbf{a_{ij}}$, obtem-se:

$$\mathbf{V_{ij}} - \mathbf{V_{ij-1}} + \mathbf{a_{ij-1}} \geq \mathbf{a_{ij}} \geq \mathbf{V_{ij}} - \mathbf{V_{ij+1}} + \mathbf{a_{ij+1}}$$

Numa situação de apenas dois clientes, a solução é encontrada por simples inspeção. Porém no caso de muitos clientes (4.000) as possibilidades de combinação são praticamente infinitas.

Uma estratégia de otimização interessante do ponto de vista logístico é a de se aceitar inicialmente uma frota que atenda todos os clientes sem desabastecimento, e em seguida buscar soluções em que a sobreposição do uso dos veículos seja minimizada. Dessa forma minimizam-se os custos fixos da frota. Os custos operacionais dos veículos (combustível, lubrificantes etc) deverão ser iguais em todas as soluções, pois os veículos operam em entrega direta (IRPDD).

Esta estratégia de solução do problema permite que pelo menos uma solução viável exista. Não é possível fixar um determinado número de veículos para atendimento dos clientes, pois poderia ocorrer desabastecimento.

O tamanho da frota varia ao longo do tempo, e sua variação medida ao longo do horizonte de projeto será utilizada como função objetivo. Essa função não é representável explicitamente por meio de fórmulas analíticas.

A função objetivo poderia ainda incorporar o custo adicional com estoques nos clientes, que deverá ocorrer devido à entrega antecipada do produto. O modelo pode neste caso verificar se em termos de custo é melhor uma frota maior e estoques menores nos clientes, ou o contrário. A análise incluindo custos com estoques não foi feita neste estudo, porém é facilmente implementável.

O modelo proposto de IRPDD atende implicitamente as restrições operacionais sem que haja necessidade de se utilizar funções de penalidades. Isso pode ser feito no futuro para incorporar outras restrições no modelo.

As taxas de consumo do produto nos clientes são na verdade variáveis aleatórias. Em muitos casos taxas uniformes representam bem a realidade, em outros não. O modelo pode ser ampliado para uma distribuição qualquer de taxa de consumo, por exemplo, a partir de dados fornecidos por um módulo de previsão estatístico de demanda. Será mais trabalhoso, porém possível construir diagramas semelhantes aos das Figuras 1, 2 e 3 e estabelecer limites para a antecipação das entregas. Outro aspecto importante é que o modelo deverá estar sendo rodado periodicamente, e alterações dos níveis de estoques estarão sendo incorporadas no modelo praticamente em tempo real, o que reduz a importância das previsões estatísticas. A modelagem proposta é exata e tem boa aderência a problemas reais de IRPDD.

3. Algoritmos genéticos

A escolha de um algoritmo para solução de um problema de otimização depende muito do tipo de modelagem feita. O modelo de IRPDD construído na seção anterior compõe-se de variáveis reais, limitadas inferiormente e superiormente, e de uma função objetivo (variabilidade do tamanho da frota) não representável analiticamente de forma explícita. Esse tipo de problema pode ser modelado com algoritmos genéticos (AGs). AGs têm sido aplicados a problemas de planejamento de transportes e bons resultados têm sido obtidos [4].

AGs são baseados no processo de seleção natural, onde sobrevivem numa população os indivíduos mais adequados segundo algum critério de aptidão (*fitness function*). As informações sobre as características de cada indivíduo (solução) são armazenadas em estruturas denominadas cromossomos, que por sua vez são compostos por seqüência de genes.

Inicialmente é gerada aleatoriamente uma população de indivíduos (soluções). Cada indivíduo é avaliado segundo um critério de aptidão (p.ex. menor custo de transporte, menor tamanho da frota). Então é selecionada a parcela dos indivíduos da população de melhor desempenho, e por meio de modificações (mutação) e combinação (*crossover*) de suas características, é gerada uma nova população de indivíduos.

O processo de criação de novas gerações de soluções continua até que seja atingido um critério de parada, p.ex., número de gerações, tempo de processamento, invariabilidade dos indivíduos de uma população etc.

AGs são de formulação simples e implementação rápida. Além disso apresentam soluções robustas e estáveis, o que é particularmente adequado para sua aplicação em problemas de grande escala. Um fator importante para uma convergência rápida de AGs é a representação utilizada para os cromossomos e dos operadores genéticos. Uma descrição cromossômica com convergência rápida é aquela que não gera soluções inviáveis e nem seja redundante.

A representação cromossômica proposta por Kemenade e Kok [4] é particularmente eficaz quando aplicada a problemas de IRPDD. No caso do modelamento desenvolvido neste estudo é utilizada a variável $\mathbf{a_{ij}}$, que é o valor do tempo de adiantamento da entrega j ao cliente i. O tempo ótimo de visita $\mathbf{V_{ij}}$ é invariável e portanto não influi na solução final. A descrição bidimensional para os cromossomos da população resulta:

	1	*2*	*3*	...	*N clientes*
1	$\mathbf{a_{11}}$	$\mathbf{a_{12}}$	$\mathbf{a_{13}}$	...	$\mathbf{a_{1N}}$
...					
M visitas	$\mathbf{a_{M1}}$	$\mathbf{a_{M2}}$	$\mathbf{a_{M3}}$	...	$\mathbf{a_{MN}}$

Essa descrição de cromossomos utiliza uma descrição bidimensional com variáveis reais para os genes ao invés de vetores com valores binários ou inteiros, como é usual na aplicação de AGs em problemas de transportes. Assim, no lugar de representar períodos (dias, semanas, meses), utiliza-se a escala real do tempo, em números reais (p.ex. 1,45 dia). A grande vantagem dessa representação é o da maior precisão dos eventos de tempo, e também de expandir o horizonte de análise para virtualmente qualquer valor. O limitante de tempo (horizonte de projeto) será o valor mínimo da última visita agendada entre todos os clientes.

Janelas de tempo (*time windows*) podem ser incorporadas sem muitas dificuldades ao modelo. Por exemplo:

data	***dia da semana***	***hora do dia***	***representação real em dias***
28/7/2003	*segunda-feira*	*0*	*0*
28/7/2003	*segunda-feira*	*14 – 16*	*0,583 – 0,691*
30/7/2003	*quarta-feira*	*8 – 10*	*2,333 – 2,417*

1/8/2003	*sexta-feira*	*12 – 14*	*4,5 – 4,583*
4/8/2003	*segunda-feira*	*14 – 16*	*7,583 – 7,691*
etc	*...*	*...*	*...*

Assim bastaria impor que $\mathbf{t_{ij}}$ pertença aos intervalos de tempo representados pelos valores apresentados na última coluna.

Genes descritos por números reais oferecem uma quantidade muito maior de alternativas viáveis, e indicam com mais precisão a influência do gene no nível de aptidão do indivíduo.

Uma característica importante dos modelos desenvolvidos com AGs é que sempre se tem uma solução inicial. No caso deste estudo qualquer valor de $\mathbf{a_{ij}}$ que atenda às restrições $\mathbf{r_1}$, $\mathbf{r_2}$ e $\mathbf{r_3}$ (ver seção 2) fornece uma solução inicial viável de programação de entregas, ainda que não otimizada. A solução inicial foi implementada sorteando-se com distribuição uniforme os valores de $\mathbf{a_{ij}}$ dentro do intervalo acima.

Outros parâmetros de AGs também são importantes para uma convergência mais rápida do algoritmo: tamanho da população e os operadores genéticos de combinação, mutação e migração. Não há uma regra específica para determinar os valores mais adequados para esses parâmetros. É usual na literatura de AGs experimentar diversos valores alternativos e selecionar os valores que produzem os melhores resultados para o modelo específico em estudo. Também é importante realizar uma análise de sensibilidade de cada parâmetro individualmente, para verificar uma possível melhora no desempenho dos algoritmos. É possível ainda incorporar ao modelo outras restrições pela utilização penalidades na função objetivo. Neste caso consegue mais flexibilidade de modelagem, que porém resulta num processo de convergência mais lento.

A Figura 4 a seguir mostra o esquema do processo geracional utilizado no modelo de AG.

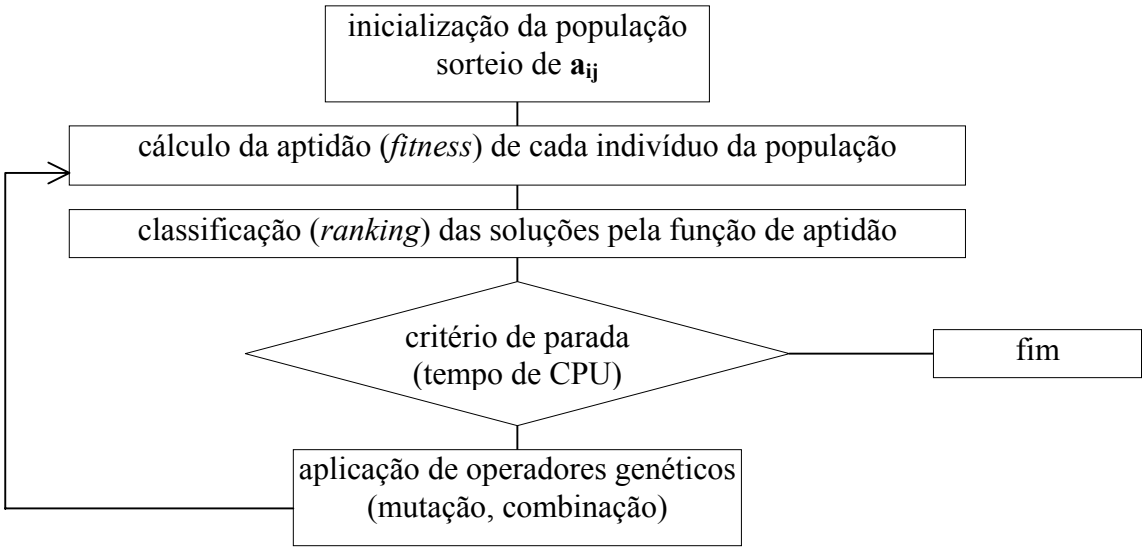


Figura 4 – Processo geracional do algoritmo genético

4. Paralelização

Algoritmos genéticos oferecem boas possibilidades de paralelização. Por exemplo, dividindo-se a população em subgrupos e executando cada subproblema em processadores diferentes. Depois de determinado número de gerações, as subpopulações são comparadas entre si, e os melhores indivíduos (soluções) são intercambiados para evitar o desenvolvimento de populações endêmicas de baixo desempenho.

A busca prossegue independentemente em cada processador e, após um determinado número de gerações, um novo intercâmbio de indivíduos é realizado entre as populações (operador genético de migração). Esse processo é repetido até que um critério de parada final seja atingido.

O processo de paralelização descrito acima aumenta a velocidade da busca numa proporção aproximadamente linear (*linear speedup*). Eventualmente a velocidade de convergência pode ser maior ainda (*superlinear speedup*) pois a busca por AG envolve muitos parâmetros de caráter aleatório.

O modelo de paralelização proposto tem as características de um modelo tipo ilha. A granulometria (tempo de processamento / tempo de comunicação) nos modelos ilha é em princípio grande. Em populações pequenas ou um número grande de processadores, a granulometria diminui (modelo de difusão). Neste caso a eficiência da paralelização diminui, pois as subpopulações resultam muito pequenas.

Aumentando-se a complexidade do modelo de IRPDD proposto, também aumentará sua granulometria. Isso ocorre por que a incorporação de restrições adicionais (janelas de tempo, velocidades variáveis etc) aumentará o tempo de cálculo da função de adaptação, enquanto o tempo de comunicação (intercâmbio de indivíduos) permanece o mesmo.

Modelos tipo ilha são bastante estáveis pois cada processador opera independentemente um do outro. A eventual interrupção de um processador não causará a interrupção da busca global, principalmente no caso de um número grande de processadores.

Uma estratégia de implementação de um modelo paralelizado de grande granulometria é a utilização das bibliotecas de troca de mensagens MPI – *Message Passing Interface*. MPI é uma proposta muito utilizada na indústria para troca de mensagens em máquinas paralelas com memória distribuída. MPI oferece grande portabilidade das aplicações entre computadores paralelos de diferentes fabricantes e arquiteturas.

As bibliotecas são acessadas diretamente por linguagens de programação (Fortran, C, etc) e é composta por uma série de subrotinas para troca de mensagens de (a) sincronização de eventos, (b) envio e recebimento de dados, (c) finalização de processos etc. A implementação de heurísticas paralelizadas utilizando MPI é bastante difundida.

5. Beowulf cluster

Até há pouco tempo atrás, rodar modelos paralelizados, como o IRPDD desenvolvido nas seções anteriores, exigiria equipamentos especializados e de custo muito alto (centenas de milhares de dólares). Eventualmente os ganhos econômicos obtidos com o melhor planejamento logístico poderiam não compensar os investimentos necessários na aquisição de sistemas computacionais sofisticados.

Um conceito novo de hardware para computação paralela, surgido recentemente, é o *beowulf cluster*, um sistema para cálculo numérico paralelizado constituído por componentes de PCs de série, que são produzidos em grande escala e de custo unitário baixo. *Beowulf clusters* compõem-se normalmente de dezenas de motherboards convencionais, processadores Pentium III, memória SRAM, winchesters e placas de rede de 10/100 Mbips. Cada conjunto composto por uma motherboard, um processador e placa de rede forma um nó. *Beowulf clusters* podem ter mais de 200 nós com um desempenho equivalente a de supercomputadores extremamente rápidos.

O desempenho de um *beowulf cluster* é praticamente proporcional ao investimento realizado no número de nós. Sua arquitetura é flexível e altamente escalável. É um sistema estável: os nós operam independentemente uns dos outros, e no caso de falha de um nó a operação global do sistema não é comprometida. *Beowulf clusters* são montados em racks, e não necessitam de instalações sofisticadas (salas limpas, ar condicionado).

Beowulf clusters são muito apropriados para problemas de cálculo numérico altamente paralelizável, como modelos de previsão do tempo, simulações moleculares, análise estrutural não linear etc.

Beowulf clusters são normalmente utilizados com os chamados 'softwares livres' (*free software*), disponíveis para *download* pela internet e em grande parte gratuita. A base instalada de softwares livres tem crescido bastante ultimamente, basicamente devido ao sucesso do Linux, um sistema operacional estável, aberto e de bom desempenho. Alguns dos softwares livres distribuídos gratuitamente e de interesse para este estudo são o sistema operacional Linux, compiladores (Fortran, C, C++, Java), bibliotecas de troca de mensagens (MPI, PVM) e bibliotecas de cálculo numérico (*Scalapack*). Isso permite uma grande economia em licenças de software, o que é muito significativo em termos de custos em um *cluster* com muitos nós. A combinação de placas e processadores de baixo custo com componentes de software livre (Linux, MPI, compiladores) permite obter uma configuração de alto desempenho computacional a baixo custo.

Para os exemplos testados neste trabalho utilizou-se o *beowulf cluster* Pingüim do LCCA – Laboratório de Computação Avançada da USP – Universidade de São Paulo. Esse *cluster* é formado atualmente por 6 nós, cada um com um processador AMD Athlon 1.1GHz, 768MB RAM e 56 GB de disco rígido. Os nós estão interligados por uma rede ethernet de 100 Mbits. O sistema operacional utilizado é o Linux, compilador Fortran G77 e biblioteca de troca de mensagens MPI-LAM.

6. Experimentos

Para testar o modelo paralelizado de IRPDD de grande escala, montou-se um exemplo com base na cidade de São Paulo para atendimento a 4.000 locais de reabastecimento. Esse é um problema típico de reabastecimento de grande escala e, como já foi mencionado anteriormente, tem grande aplicação no planejamento da distribuição de derivados de petróleo (p.ex. postos de gasolina), gases industriais, supermercados, distribuição de autopeças.

As coordenadas geográficas (latitude e longitude) foram sorteadas aleatoriamente com distribuição uniforme dentro do perímetro da cidade. Os demais dados logísticos foram também sorteados aleatoriamente. Foram utilizados adimensionais relacionados de forma coerente entre si, e representando situações reais de transporte. Os valores de capacidade máxima de estoque, estoque inicial e taxa de consumo por unidade de tempo para cada local de reabastecimento foram sorteados utilizando distribuições uniformes.

As distâncias foram estimadas pela distância euclidiana entre os locais mais um fator (25%) para corrigir desvios de direção. Admitiu-se velocidade média de 20 Km/h, constante para todos os locais e em qualquer período do dia. A capacidade de cada veículo da frota foi fixada em 10.000, quantidade adimensional proporcionalmente coerente com os outros valores adimensionais admitidos. Foi considerado o planejamento de entregas em 20 visitas seqüenciais a cada um dos 4.000 clientes.

A qualidade da solução é medida pela variabilidade do número de veículos necessários ao longo do tempo. Quanto menor a variabilidade, melhor a solução (menor frota fixa). A distribuição ao longo do tempo da frota é calculada verificando-se os períodos de tempo onde ocorre uso simultâneo de veículos.

O modelo de algoritmo genético utilizou uma população de 600 indivíduos (soluções). Os indíviduos da população são classificados segundo o seu desempenho de adaptação. Uma nova geração cromossômica é formada da seguinte forma:

- os 8 melhores indivíduos passam inalterados para a nova geração,
- os genes dos 25% melhores indivíduos (sem incluir os 8 melhores) sofrem mutação,
- 50% dos indivíduos da nova população são gerados por combinação da geração anterior,
- os 25% restantes da população são gerados aleatoriamente.

Na mutação cada um dos genes sofre uma alteração de 10% no seu valor. O sinal da variação (+ ou -) é sorteado aleatoriamente com distribuição uniforme. O operador de combinação gera um novo cromossomo a partir de dois cromossomos sorteados aleatoriamente entre os 75% melhores da população. Cada novo gene do novo cromossomo é também escolhido aleatoriamente entre os genes de mesmo índices (i,j) do par de cromossomos formadores.É verificado em cada novo gene gerado se a variável $\mathbf{a_{ij}}$ atende às restrições $\mathbf{r_1}$, $\mathbf{r_2}$ e $\mathbf{r_3}$ (ver seção 2). Caso isso não ocorra, a variável $\mathbf{a_{ij}}$ é alterada para o mais próximo valor viável.

A paralelização do processo de otimização se inicia com a divisão da população em subpopulações de tamanhos iguais, que são executadas seqüencialmente e de forma independente em cada processador do *beowulf cluster*. A cada cinco gerações, as soluções obtidas em cada uma das subpopulações são comparadas entre si, e uma nova subpopulação contendo as melhores soluções é gerada. Essa subpopulação 'ótima' é enviada aos processadores para novo ciclo geracional.

Foram realizados testes com um número variável de CPUs (1, 2, 3, 4 e 6) e os resultados estão apresentados nas Figuras 5 e 6. A Figura 5 mostra que a utilização de um número maior de CPUs fornece melhores soluções. Com uma única CPU obtem-se uma variabilidade total da frota de $5{,}34 \times 10^{-7}$ (veículosXsegundos), que corresponde a uma frota máxima de 64 veículos. Com 6 CPUs a variabilidade reduz-se a $4{,}66 \times 10^{-7}$ (veículosXsegundos), que corresponde a uma frota máxima de 60 veículos. O tempo de processamento também se reduz: de 5 horas (1 CPU) para 2 horas (6 CPUs).

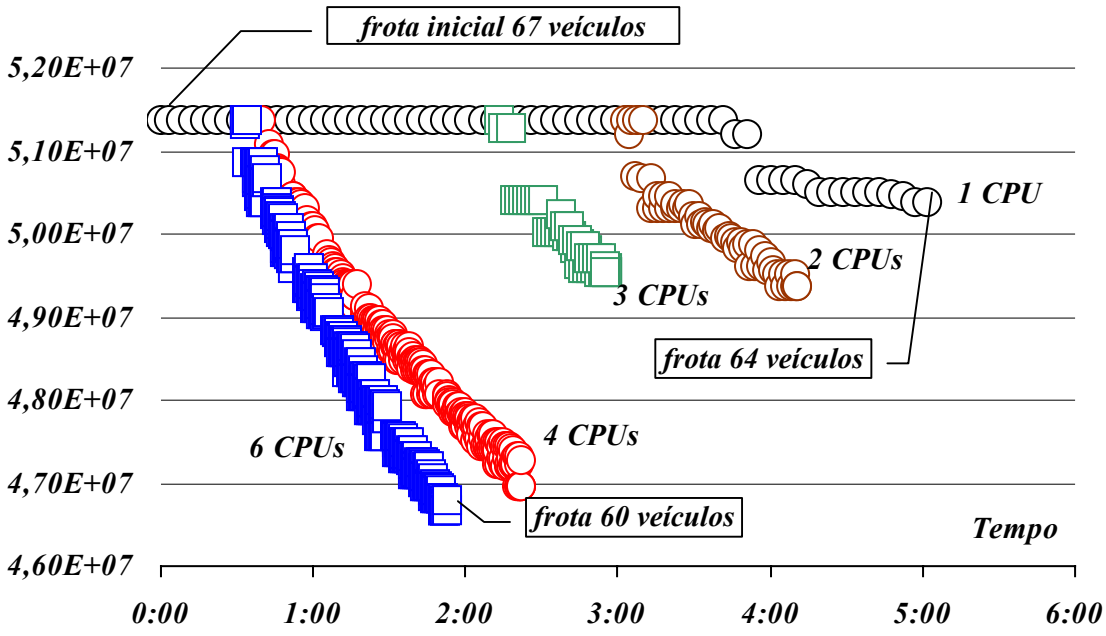


Figura 5 – Variabilidade do tamanho da frota (veículosXsegundos) em função do tempo total de processamento

A Figura 6 abaixo mostra as acelerações obtidas na busca da solução ótima com o número de CPUs. Foram considerados dois valores de função objetivo: 6% e a 8% da melhor solução obtida nos experimentos. Para comparação, inclui-se no gráfico o comportamento linear (linha tracejada).

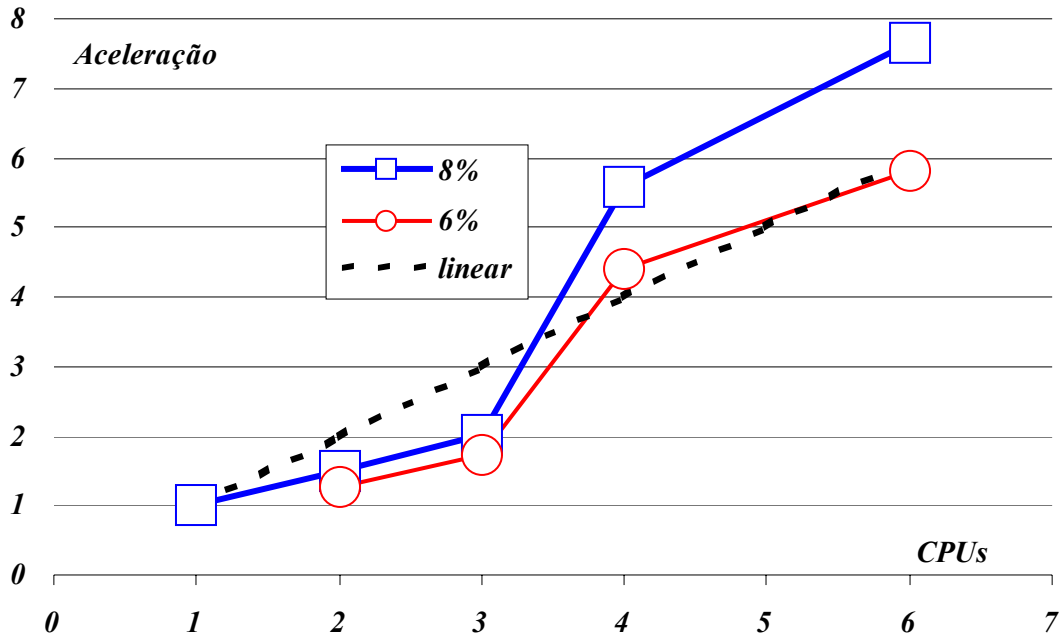


Figura 6 – Aceleração do tempo de busca (speedup)

Observa-se na Figura 6, que as curvas de 6% e 8% tem comportamente aproximadamente linear. Como previu-se anteriormente (ver seção 4), o tempo total gasto na busca de soluções ótimas é linearmente proporcional ao número de CPUs. Em alguns casos, tem-se mesmo um comportamento superlinear, o que é não é incomum em modelos heurísticos paralelizados [3].

7. Conclusões

Os resultados obtidos nos experimentos com um modelo IRPDD de grande escala mostram o potencial de aplicação de técnicas de programação paralela em problemas de logística. Amplia-se significativamente o escopo de análise de problemas IRPDD, de algumas dezenas de clientes para milhares, e do horizonte de planejamento de dias e semanas, para um período virtualmente infinito.

A razão básica dos bons resultados obtidos está nas características de alta granularidade do modelo IRPDD, que permite uma redução aproximadamente linear do tempo de processamento com o número de CPUs. E a solução proposta também é viável economicamente: *beowulf cluster* e softwares livres conjugam alto desempenho computacional e baixos custos.

O modelo de IRPDD utilizado foi simplificado, porém sua aplicabilidade não foi descaracterizada. Esse modelo é uma ferramenta que pode fornecer bons indicadores para o planejamento de rotas de abastecimento. A sofisticação do modelo não deverá inviabilizar a metodologia proposta. Pelo contrário, o aumento da complexidade do modelo resultará num modelo de granulometria ainda maior, exigindo unicamente uma maior quantidade de nós com maior poder de processamento. O tamanho do modelo (clientes, visitas, população) do IRPDD a ser solucionado é limitado apenas pelos recursos computacionais disponíveis nos nós.

Outros desenvolvimentos são ainda necessários para se calibrar o modelo, como por exemplo, análise do tamanho da população e dos parâmetros dos operadores genéticos (mutação,

combinação). Além disso, as possibilidades do paralelismo que *beowulf clusters* oferecem fazem com todas as hipóteses usuais de modelamento devam ser reavaliadas, visando uma eficiência de busca ainda maior.

Assim ferramentas como técnicas de programação paralela operando em conjunto com *beowulf cluster* e software livre mostraram ser eficientes para a solução de problemas logísticos de grande escala, e podem ser aplicadas em vários problemas de grande importância econômica.

Referências bibliográficas

[1] Campbell, A., Clarke, L., Savelsbergh, M., *An inventory routing problem*. The Logistics Institute, Georgia Institute of Technology, Abril, 1999.
(www.tli.gatech.edu/research/papers/files/misc/9916.pdf)

[2] Kleywegt, A. Nori, V., Savelsbergh, M. *The stochastic inventory routing problem with Direct Deliveries*. School of Industrial and Systems Engineering, Georgia Institute of Technology, Junho, 1999.
(www.tli.gatech.edu/research/papers/files/misc/9901.pdf)

[3] Cung, V.-D., Martins, S.L., Ribeiro, C.C., Roucariol, C. *Strategies for the parallel implementation of metaheuristics*. in Essays and Surveys in Metaheuristics (C.C. Ribeiro d P. Hansen editores), 263-308, Kluwer, 2001.
(www-di.inf.puc-rio.br/~celso/artigos/parmeta.pdf)

[4] van Kemenade, C.H.M., Kok, J.N. An evolutionary approach to time constrained routing problems. CWI, Centrum voor Wiskunde Informatica, Amsterdã, Holanda, 1995, ISSN 0169-118X, 12p. (www.cwi.nl/ftp/CWIReports/AP/CS-R9547.ps.Z)

Agradecimentos

Este trabalho foi desenvolvido com apoio da FAPESP Fundação de Amparo a Pesquisa do Estado de São Paulo – Programa PIPE, contrato número 01/03196-6. Agradecemos pelas discussões técnicas às professoras doutoras Cíntia Rigão Scrich (UNIP/Campinas) e Lúcia F. de Almeida Guimarães (PUC/Campinas) e aos professores doutores Alfredo Goldman vel Lejbman (IME/USP) e Marco Antônio Brinati (POLI/USP).