v.13, n.2, p.271-281, mai.-ago. 2006

ALGORITMOS GENÉTICOS E COMPUTAÇÃO PARALELA PARA PROBLEMAS DE ROTEIRIZAÇÃO DE VEÍCULOS COM JANELAS DE TEMPO E ENTREGAS FRACIONADAS

Guilherme Guidolin de Campos
Hugo Tsugunobu Yoshida Yoshizaki

Departamento de Engenharia de Produção, Escola Politécnica da USP,
Av. Prof. Almeida Prado, Travessa 2, n. 128, 2º andar,
Cidade Universitária, CEP 05508-900, São Paulo, SP,
e-mails: guilherme.campos@bain.com, hugo@usp.br

Patrícia Prado Belfiore

Departamento de Engenharia de Produção, Centro Universitário da FEI,
Av. Humberto de Alencar Castelo Branco, 3972, Assunção,
CEP 09850-901, São Bernardo do Campo, SP,
e-mail: patricia.belfiore@labfin.com.br

Recebido em 25/8/2005
Aceito em 08/6/2006

Resumo

O presente trabalho propõe a utilização de metaheurísticas e computação paralela para a resolução de um problema real de roteirização de veículos com frota heterogênea, janelas de tempo e entregas fracionadas, no qual a demanda dos clientes pode ser maior que a capacidade dos veículos. O problema consiste na determinação de um conjunto de rotas econômicas que devem atender à necessidade de cada cliente respeitando todas as restrições. A estratégia adotada para a resolução do problema consiste na utilização de uma adaptação da heurística construtiva proposta por Clarke e Wright (1964) como solução inicial. Posteriormente, implementa-se um algoritmo genético paralelo que é resolvido com o auxílio de um cluster de computadores, com o objetivo de explorar novos espaços de soluções. Os resultados obtidos demonstram que a heurística construtiva básica apresenta resultados satisfatórios para o problema, mas pode ser melhorada substancialmente com o uso de técnicas mais sofisticadas. A aplicação do algoritmo genético paralelo de múltiplas populações com solução inicial, que apresentou os melhores resultados, proporciona redução no custo total da operação da ordem de 10%, em relação à heurística construtiva, e 13%, quando comparada às soluções utilizadas originalmente pela empresa.

Palavras-chave: problema de roteirização de veículos, janelas de tempo, entregas fracionadas, metaheurísticas.

1. Introdução

O problema de roteirização de veículos com entrega fracionada (*Split Deliveries Vehicle Routing Problem – SDVRP*) foi introduzido recentemente na literatura por Dror e Trudeau (1989, 1990), que apresentaram a formulação matemática do problema e analisaram as economias que podem ser geradas quando se permite que um cliente possa ser atendido por mais de um veículo, tanto em relação ao número de veículos quanto à distância total percorrida.

Segundo Dror e Trudeau, o *SDVRP* é uma relaxação do problema clássico de roteirização de veículos, mas permanece *NP*-completo. Os autores demonstraram que quando as distâncias satisfazem a desigualdade do triângulo, existe uma solução ótima para o problema, de modo que nenhum par de rotas tenha dois ou mais vértices em comum.

O *SDVRPTW* é uma generalização do problema de roteirização de veículos com entregas fracionadas (*SDVRP*), adicionando restrições de janelas de tempo. Ho e Haugland (2004) demonstraram que a propriedade de desigualdade do triângulo de Dror e Trudeau (1989, 1990) também vale para o *SDVRPTW*.

O objetivo deste trabalho consistiu na elaboração de heurísticas e metaheurísticas para um problema real de roteirização de veículos com janelas de tempo e entregas fracionadas (*Split Deliveries Vehicle Routing Problem*

with Time Windows – SDVRPTW), de forma a otimizar o sistema de transporte. Primeiramente, implementa-se uma adaptação da heurística de Clarke e Wright (1964). A seguir, implementa-se a heurística Meta-RaPS de Moraga et al. (2001). A partir das heurísticas anteriores, é implementada a metaheurística Algoritmos Genéticos com base nos trabalhos de Bjarnadottir (2004), Cantú-Paz (1999) e Ombuki et al. (2006). Finalmente, aplica-se o Algoritmo Genético paralelo com base nos trabalhos de Hwang e Briggs (1984), Navaux (1989) e Cantú-Paz (1999). Os métodos foram aplicados em um sistema de abastecimento de lojas de um dos maiores grupos varejistas do Brasil.

O problema estudado neste trabalho consistiu na determinação de um conjunto de rotas econômicas que deveria atender à necessidade de abastecimento de cada uma das lojas do grupo respeitando todas as restrições, principalmente janelas de tempo, duração da jornada e frota heterogênea.

A estratégia adotada para a resolução do problema consiste na utilização de uma adaptação da heurística construtiva proposta por Clarke e Wright (1964) como solução inicial. Posteriormente, são utilizados alguns algoritmos mais sofisticados buscando-se melhorias, dentre eles o algoritmo genético paralelo resolvido com o auxílio de um *cluster* de computadores. Estes métodos de solução são, então, aplicados ao problema prático de abastecimento das lojas do grupo.

A seqüência do artigo está detalhada a seguir. O item 2 apresenta a definição do problema. Já o item 3 apresenta os métodos de solução implementados. Os resultados encontram-se no item 4 e as conclusões e futuras pesquisas no item 5.

2. Definição do problema

O presente trabalho trata de um problema de transporte com decisões de seleção de veículos, consolidação das cargas e roteirização, a partir de um Centro de Distribuição, e que atenda às restrições de abastecimento das lojas com menor custo possível. O problema se enquadra na categoria de Problemas de Roteirização de Veículos com Janelas de Tempo e Entregas Fracionadas (*SDVRPTW*) e tem como característica uma frota de veículos heterogênea. A definição completa e detalhada do problema está listada abaixo.

A partir de um único centro de distribuição, são atendidos N clientes a cada dia. O centro de distribuição atende um total de 323 lojas em todo o Brasil, atuando em 11 Estados brasileiros. Os produtos são distribuídos por uma frota de veículos heterogênea e ilimitada. O serviço de transporte é terceirizado. Cada tipo de veículo tem uma capacidade C_v. A demanda de cada cliente i no dia t é d_{it} e a maioria das lojas possui demanda superior à capacidade do maior caminhão capaz de atendê-la. As lojas possuem restrições de janela de tempo e de recebimento de determinados tipos de veículos. Os veículos saem e retornam ao depósito.

Dada a demanda das lojas em um determinado dia, o objetivo do modelo é decidir como alocar os veículos às lojas, se a consolidação de cargas será de lotação (uma única loja) ou fracionada (várias lojas), determinar a quantidade a ser entregue em cada veículo para cada loja e qual o melhor roteiro no caso de rota fracionada, de forma a minimizar o custo total de distribuição. As restrições estão listadas abaixo:

- Atender à demanda das lojas respeitando as janelas de tempo;
- Cada tipo de veículo tem uma capacidade C_v;
- Algumas lojas possuem restrições quanto ao tipo de veículo; e
- Todos os veículos iniciam e terminam seu trajeto no depósito.

As principais hipóteses e características do problema são:

- A demanda das lojas é determinística;
- A frota de veículos é heterogênea;
- A quantidade de veículos disponíveis diariamente é ilimitada;
- Os clientes podem pertencer a mais de um roteiro; e
- A demanda dos clientes pode ser maior que a capacidade dos veículos.

A formulação matemática do modelo está listada a seguir.

Considerando-se um conjunto de N clientes a serem atendidos: em um determinado dia, a demanda d_i de cada cliente $i \in \{1,2,..., n\}$ deve ser atendida; cada cliente possui uma restrição de janela de tempo $[a_i, b_1]$, tal que $a_i \leq b_i$, que corresponde, respectivamente, ao horário inicial e final em que pode ser iniciado o atendimento; o tempo de atendimento s_i representa o tempo de descarga dos veículos (tempo médio de todos os processos administrativos, burocráticos e de operação); um roteiro pode ser formado por uma única loja ou por várias lojas; e cada rota será atendida por um determinado veículo v que possui uma capacidade máxima C_v.

O deslocamento do nó i, $i = 0,..., n$ até um nó j, $j = 1,..., n + 1$ requer um tempo de viagem t_{ij} e uma distância percorrida d_{ij}. Os pontos 0 e $n + 1$ representam o depósito. As matrizes de tempos de viagem e de distâncias são simétricas, ou seja, $t_{ij} = t_{ji}$ e $d_{ij} = d_{ji}$.

As variáveis de decisão do modelo são:

- $x_{ij}^v = 1$, se j é atendido após i pelo veículo v
  0, caso contrário;

- T_i^v = tempo de início de atendimento do cliente i pelo veículo v, $i = 1,..., n$ $v = 1,..., m$; e
- y_i^v = fração de demanda do cliente i entregue pelo veículo v;

O objetivo do modelo é minimizar o custo total de transporte (CT), de forma que a demanda de todos os clientes seja atendida e as demais restrições do problema sejam respeitadas. A formulação completa do problema está detalhada a seguir com base nos trabalhos de Dror e Trudeau (1990), Ho e Haugland (2004) e Golden et al. (1984).

$$\min \sum_{j=1}^{n} \sum_{v=1}^{m} CT^v x_{0j}^v \quad (1)$$

As restrições do modelo são:

$$\sum_{j=1}^{n} x_{0j}^v = 1 \qquad v = 1, 2,..., m \quad (R1)$$

A restrição (1) garante que cada veículo saia do depósito e chegue a um determinado cliente.

$$\sum_{i=0}^{n} x_{ip}^v - \sum_{j=0}^{n} x_{pj}^v = 0 \quad p = 0,... n; \quad v = 1, 2,..., m \quad (R2)$$

A restrição (2) é a restrição de conservação dos fluxos de entrada e saída; garante que cada veículo saia de um determinado cliente e retorne ao depósito.

$$\sum_{v=1}^{m} y_i^v = 1, \qquad i = 1, 2,..., n \quad (R3)$$

A restrição (3) garante que a demanda total de cada cliente será atendida.

$$\sum_{i=1}^{n} d_i y_i^v \leq C_v \qquad v = 1, 2,..., m \quad (R4)$$

A restrição (4) garante que a capacidade de cada veículo não será excedida.

$$y_i^v \leq \sum_{j=0}^{n} x_{ji}^v \quad i = 1,..., n; \quad v = 1,..., m \quad (R5)$$

A restrição (5) garante que a demanda de cada cliente será atendida somente se um determinado veículo passar por aquele ponto. Nota-se que, adicionando à restrição (5) o somatório de todos os veículos e, combinando-a com a Equação 3, obtém-se a restrição $\sum_{v=1}^{m} \sum_{i=0}^{n} x_{ij}^v \geq$, $j = 0,..., n$ que garante que cada vértice seja visitado pelo menos uma vez, por pelo menos por um veículo.

$$T_i^v + s_i + t_{ij} - M_{ij}(1 - x_{ij}^v) \leq T_j^k \quad (R6)$$
$$i = 1,..., n; j = 1,..., n; v = 1,... m$$

A Equação 6 impõe o horário mínimo de início de atendimento do cliente j em uma determinada rota e também garante que não haja formação de *subtours*. A constante M_{ij} é um número suficientemente grande, por exemplo, $M_{ij} = b_i + t_{ij} - a_j$.

$$a_i \leq T_i^v \leq b_i \qquad i = 1, 2,..., N \quad (R7)$$

A restrição (7) garante que todos os clientes são atendidos dentro de sua janela de tempo.

Há também a restrição quanto ao tipo de veículo (8), sendo que três conjuntos são considerados:

V_1 = {carreta, truck, toco e leve}; (R8)
V_2 = {truck, toco e leve}; e
V_3 = {carreta e truck.

Algumas lojas i aceitam todos os tipos de veículos ($i \in V_1$). Muitas lojas i não aceitam o maior veículo (carreta), devido às restrições operacionais, como tamanho das docas e falta de espaço para manobra ($i \in V_2$). Outras lojas i aceitam apenas carreta e *truck* ($i \in V_3$).

$$y_i^v \geq 0 \quad i = 1,..., n; \qquad v = 1,..., m \quad (R9)$$
$$T_i \geq 0 \quad i = 1,..., n$$

A Equação 9 garante que as variáveis de decisão Q_i^v e T_i^v sejam positivas.

$$x_{ij}^v \in \{0,1\} \quad i = 0,..., n; \quad j = 0,..., n; \quad (R10)$$
$$v = 1,..., m$$

Finalmente a Equação 10 garante que as variáveis de decisão x_{ij}^v sejam binárias.

3. Método de solução

Este item está dividido em quatro partes, de acordo com os métodos utilizados. A primeira parte corresponde à implementação da heurística de Clarke e Wright (1964) adaptada. A segunda parte corresponde à heurística Meta-RaPS de Moraga et al. (2001), que é uma forma simples de se melhorar o desempenho de uma heurística construtiva.

Na terceira parte, descreve-se a metaheurística Algoritmos Genéticos com base nos trabalhos de Bjarnadottir (2004), Cantú-Paz (1999) e Ombuki et al. (2006). Finalmente, na quarta parte, é explicado como o Algoritmo Genético foi adaptado para funcionar de forma paralela em um *cluster*. Os algoritmos genéticos paralelos foram baseados nos trabalhos de Hwang e Briggs (1984), Navaux (1989) e Cantú-Paz (1999).

3.1 Adaptação da heurística de Clarke e Wright (1964)

Este trabalho implementou uma adaptação da heurística de Clarke e Wright (1964), baseada na versão paralela, como solução inicial para o problema. As adaptações em função da heurística original são listadas a seguir.

3.1.1 Demanda

Conforme mencionado anteriormente, a maioria das lojas possui demanda superior à capacidade dos veículos. Neste sentido, foi implementado um algoritmo para suprir a demanda de cada loja com caminhões lotados, visitando, exclusivamente, aquela loja até que a demanda remanescente seja menor do que a capacidade do maior veículo capaz de atendê-la. Assim, garante-se que a demanda de cada loja roteirizada pela heurística seja menor do que a capacidade dos caminhões, permitindo que os métodos pesquisados sejam utilizados sem maiores problemas. Na prática, este método é bastante intuitivo, pois enviar veículos grandes com carga completa é a forma de transporte que minimiza o custo unitário por mercadoria entregue.

3.1.2 Cálculo das economias

O objetivo da heurística original de Clarke e Wright é minimizar a distância total percorrida em todas as rotas, assim, a economia corresponde à distância economizada ao juntar duas lojas numa mesma rota. No entanto, neste problema estudado, em vez da distância, considera-se o custo de frete pago às empresas transportadoras, que é função de um custo fixo do veículo mais um custo variável em função da distância percorrida.

3.1.3 Frota heterogênea

Ao contrário do que ocorre na heurística básica de Clarke e Wright, no modelo estudado, a frota de veículos é heterogênea. Para lidar com esse fato, o algoritmo funciona da seguinte maneira:

- Assim como no algoritmo original, inicialmente são criadas rotas individuais para atender cada uma das lojas;
- A cada rota, é atribuído o menor caminhão capaz de atender à demanda da respectiva loja; e
- Quando duas lojas são avaliadas para se juntar em uma mesma rota, caso a demanda das duas combinadas ultrapasse a capacidade do veículo atual, o menor veículo capaz de atender a essa demanda é designado para a rota.

3.2 Método Meta-RaPS

Uma forma de se incrementar a qualidade de uma heurística construtiva, transformando-a numa metaheurística, é pelo método Meta-RaPS. Segundo Moraga et al. (2001), incorporar elementos aleatórios em uma heurística pode aumentar significativamente seu desempenho. Este procedimento, além de possuir baixo tempo de execução quando comparado a outras metaheurísticas, é extremamente simples, podendo ser utilizado em problemas práticos de roteamento de veículos.

Descreve-se a seguir o procedimento Meta-RaPS para resolução de problemas. O Meta-RaPS integra regras de prioridade (ou regras heurísticas), elementos aleatórios e amostragem. A cada iteração, este procedimento constrói soluções viáveis pela utilização de regras de prioridade em uma ordem aleatória. Após um número de iterações, a melhor solução encontrada é selecionada. Segundo Moraga et al. (2001), a aplicação do Meta-RaPS em qualquer problema, como um procedimento geral, consiste em:

1 - Estudar a estrutura do problema e compreender de forma clara as variáveis, as restrições e a otimização necessárias para resolver o problema;

2 - Encontrar regras heurísticas apropriadas à construção de soluções viáveis;

3 - Criar uma lista das próximas atividades disponíveis. Usando a heurística construtiva e a estrutura do problema, selecionar a próxima atividade. Um parâmetro percentual de restrição pode ser usado como um mecanismo para diminuir ou expandir a lista de atividades disponíveis;

4 - Modificar a regra de priorização introduzindo aleatoriedade de acordo com dois critérios: em alguns ou todos os passos da regra de priorização se possível; na combinação de regras de priorização;

5 - Rodar a heurística. A cada iteração, a regra de prioridade e sua versão modificada são combinadas produzindo diferentes soluções viáveis. Após um número de iterações, a melhor solução encontrada é mantida;

6 - No final de cada iteração, o procedimento pode adotar uma técnica de melhoria na solução encontrada; e

7 - O Meta-RaPS pode utilizar diversos mecanismos para interromper a busca. O critério mais simples usado é estabelecer um número fixo de iterações, depois do qual a melhor solução encontrada é mantida.

Este procedimento bastante simples permite que soluções, ligeiramente diferentes daquela gerada pela heurística construtiva original, sejam testadas de maneira bastante rápida. Assim, amplia-se significativamente o universo de soluções exploradas, permitindo que resultados melhores sejam obtidos.

Moraga et al. (2001) propõem três parâmetros básicos para controlar o funcionamento do Meta-RaPS:

- **Prioridade** – é um fator que varia de 0 a 1 e representa a probabilidade da regra heurística original ser utilizada. Uma prioridade de 0,25 significa que 25% dos arcos testados para a formação de um roteiro serão escolhidos a partir da regra heurística original, que no caso do Clarke e Wright é a lista de economias. O demais 75% serão escolhidos aleatoriamente entre a lista restrita criada pelo algoritmo;
- **Restrição** – este parâmetro controla justamente o tamanho da lista restrita na qual o algoritmo buscará o próximo arco a ser incluído no roteiro. Um valor de 5% para este parâmetro significa que a lista restrita contará com todos os arcos cuja economia seja até 5% menor do que

aquela proposta pela regra heurística (que representa a maior economia possível); e

- **Melhoria** – o último parâmetro que controla este algoritmo determina a probabilidade de um procedimento de melhoria ser utilizado no final de cada iteração. Assim, se este valor for de 20%, a cada 5 soluções geradas, uma será modificada pelas melhorias.

Na versão utilizada do Meta-RaPS, optou-se por uma estrutura ainda mais simples de seleção da próxima atividade que requer apenas uma variável de controle, tornando mais fácil sua implementação e entendimento.

Define-se uma variável chamada de P_H que representa a probabilidade de, em um dado momento, a regra heurística ser utilizada na junção de dois nós para formar uma rota. Esta variável é estabelecida no início do programa como um valor entre 0 e 1. A cada passo do algoritmo de Clarke e Wright, o programa escolhe um número aleatório também entre 0 e 1. Caso este número seja menor do que P_H, a regra heurística é utilizada na formação da próxima rota. Caso este valor seja maior do que P_H, aquela atividade é ignorada e passa-se à próxima atividade da lista.

Assim, quanto maior o valor de P_H, mais parecida com a solução original, proposta pela heurística de Clarke e Wright, a solução formada tenderá a ser. Analogamente, quanto menor este valor, maior a probabilidade das duas soluções encontradas serem diferentes. No caso extremo de $P_H = 1$, a solução obtida será idêntica a de Clarke e Wright. Além disso, optou-se por não implementar um procedimento de melhoria para ser utilizado no final do processo, sendo este um potencial campo para pesquisas futuras.

3.3 Algoritmos genéticos

Os métodos de solução implementados utilizando os Algoritmos Genéticos (AG) foram baseados nos trabalhos de Bjarnadottir (2004), Cantú-Paz (1999) e Ombuki et al. (2006).

Uma das etapas mais importantes na implementação de um AG é a escolha da estrutura de codificação de uma solução ou do DNA de um indivíduo. Esta estrutura deve ser capaz de armazenar toda a informação necessária para representar de maneira precisa uma determinada solução.

No entanto, o material genético armazenado em um indivíduo não precisa necessariamente representar de forma direta a solução. A informação armazenada no DNA constitui o genótipo do indivíduo, enquanto a solução final representa o seu fenótipo. Para passar do genótipo para o fenótipo, o algoritmo conta com um conjunto de regras e procedimentos que lhe permite fazer esta transição, assim como na natureza, o material genético é convertido no "corpo" do indivíduo.

Como se trata de um problema de roteirização, a solução que se deseja obter é um conjunto de rotas que atenda à demanda das lojas de modo a minimizar o custo da operação, respeitando suas restrições. Desta forma, a estrutura da solução deve conter a informação necessária para a construção destas rotas. No entanto, neste problema, cada loja pode ser visitada por vários caminhões, mesmo que sua demanda possa ser atendida por apenas um. Assim, além de determinar a seqüência de lojas a ser atendida em cada rota, deve-se determinar a quantidade a ser entregue em cada uma delas. Para representar o problema estruturado desta forma, utilizou-se a seguinte codificação:

Parâmetros:

- N – número de indivíduos numa população;
- R – número máximo de rotas em uma solução;
- T – tamanho máximo de uma rota em número de lojas visitadas;
- L – número de lojas a serem atendidas; e
- K – tipos de caminhões disponíveis.

Índices:

- i – número do elemento na população, i = 1,...,N;
- j - número da rota na solução, j = 1,...,R; e
- k - posição da loja na rota, k = 1,...,T.

Variáveis:

- Pop_{ijk} – loja visitada na posição k da rota j na solução i, Pop_{ijk} = 0,...,L; e
- Pop_{ij0} – tipo de veículo utilizado para atender à rota j na solução i, Pop_{ij0} = 1,...,K.

Esta estrutura de codificação permitiu representar em um vetor solução toda a informação necessária para o estabelecimento das rotas e dos veículos que irão percorrê-las, no entanto, não representa a quantidade que será entregue para cada uma das lojas. Isto ocorre, pois se optou por manter a atribuição da demanda a cada loja como uma característica fenotípica determinada por um algoritmo simples de alocação. Desta forma, manteve-se reduzido o tamanho do vetor solução e garantiu-se um processo de evolução mais simples e rápido.

A alocação da quantidade entregue em cada parada de um veículo é determinada pelo seguinte algoritmo da Figura 1.

Obedecendo ao algoritmo da Figura 1, pode-se determinar, além da seqüência de lojas visitadas em cada rota, a quantidade entregue em cada parada. Este valor fica armazenado na variável Q_{ijk}.

Uma vez definida a estrutura básica de codificação, o próximo passo no AG é a geração de uma população inicial de indivíduos ou soluções. Neste trabalho, adotaram-se duas técnicas diferentes para elaboração destes indivíduos. Uma delas consiste na geração de valores aleatórios para cada posição de DNA do indivíduo, o que gera soluções iniciais muito ruins, mas garante uma

ampla exploração do espaço de soluções possíveis. Outra abordagem é o uso de uma solução inicial obtida pela heurística de Clarke e Wright ou Meta-RaPS. Este segundo método garante a convergência muito mais rápida do algoritmo, mas pode conduzir a ótimos locais e limitar sua capacidade de encontrar soluções melhores.

Outro elemento fundamental no algoritmo genético é o processo de seleção dos indivíduos que irão se recombinar para dar origem à nova população. Existem muitos métodos para se fazer esta seleção, cada qual com vantagens e desvantagens. No desenvolvimento deste trabalho, dois dos métodos mais populares foram implementados, sendo que um deles apresentou resultados significativamente melhores nos testes e, por isso, foi o escolhido.

O primeiro método testado consiste na atribuição a cada individuo de uma probabilidade de ser selecionado como "pai" proporcional a sua função utilidade. Assim, indivíduos que apresentam melhores soluções para o problema possuem mais chances de se reproduzir, mas todos os indivíduos possuem alguma probabilidade. Este método é interessante, pois garante uma amplitude de recombinações bastante grande possibilitando o surgimento de indivíduos bem diversificados. No entanto, esta abordagem faz com que indivíduos ruins gerem muitos descendentes, diminuindo o número de boas soluções em cada geração.

O segundo método considerado baseia-se na seleção apenas dos melhores indivíduos da população, para dar origem à geração seguinte. Para tanto, todos os indivíduos de uma população são classificados em ordem decrescente, de acordo com sua utilidade, e apenas os x melhores são escolhidos para formar o grupo dos futuros pais de cada elemento da próxima geração x, que é um parâmetro de algoritmo estabelecido pelo usuário. Cada elemento deste grupo tem igual probabilidade de ser escolhido.

Este segundo método, conhecido como elitismo, possibilita o surgimento de um maior número de indivíduos de qualidade e, assim, maior velocidade na obtenção de boas soluções. Em todos os testes realizados, este método apresentou melhores resultados e, por isso, foi adotado. A escolha do parâmetro x não apresentou impacto significativo no desempenho do algoritmo e seu valor foi fixado em 5.

O passo seguinte nos AGs é a recombinação dos indivíduos selecionados (por qualquer dos métodos descritos), para dar origem a uma nova população. Neste ponto, os algoritmos genéticos apresentam uma forma significantemente diferente dos demais métodos de soluções de gerar novos indivíduos. Esta função é realizada por um operador chamado de *crossover*.

O *crossover*, assim como na biologia, permite o intercâmbio de material genético potencialmente útil na formação de novas soluções. Para controlar a taxa de utilização deste operador, existe um parâmetro Pc que estabelece a probabilidade de que ele seja utilizado. Caso este operador não seja utilizado para combinar duas soluções para gerar uma terceira, é feita uma cópia idêntica a um dos pais escolhidos aleatoriamente, o que representa uma reprodução assexuada. A Figura 2 descreve o funcionamento do *crossover*.

O processo de geração de novos elementos é seguido por outro gerador baseado na biologia: a mutação. No contexto deste problema, foram estabelecidas três modalidades de mutação usadas em conjunto por modificar as soluções geradas pela recombinação. A primeira forma de mutação consiste na simples alteração da informação contida em um lócus ou variável do problema. Neste caso, um valor aleatório, representando uma loja ou um tipo de caminhão, dependendo da posição do vetor, é escolhido aleatoriamente com uma probabilidade Pm. Segundo Cantú-Paz (1999), esta probabilidade deve ser bastante baixa para garantir um bom desempenho. Caso este valor seja muito alto, várias mutações ocorrem ao mesmo tempo, destruindo soluções boas. No modelo, o valor adotado para Pm foi 0,01%.

O segundo tipo de mutação implementado consiste na troca de posição de duas lojas numa mesma rota. Assim, o conjunto de lojas visitadas por um determinado caminhão é mantido, alterando apenas a seqüência na qual

- Para cada rota de uma solução, começando da primeira:
  - Inicie com o caminhão vazio:
  - Para cada loja da rota, enquanto houver capacidade ociosa no veículo:
    - Se a demanda remanescente da loja for menor do que a capacidade ociosa:
      - Atenda à toda a demanda da loja.
    - Caso contrário:
      - Entregue toda a capacidade disponível e termine a rota.

Figura 1. Algoritmo de alocação da quantidade entregue a cada loja da rota.

- Caso seja escolhido *crossover* com probabilidade = Pc:
  - Escolha dois indivíduos para serem os pais (XeY);
  - Escolha um ponto "z" no vetor solução para efetuar a quebra;
  - Para j = 1 até z e k = 0 até t:
    - Copie os valores de popx,j,k;
  - Para j = z + 1 até r e k = 0 até t:
    - Copie os valores de popy,j,k.

Figura 2. Funcionamento do *crossover*.

elas são atendidas. Esta operação permite que conjuntos promissores de lojas sejam colocados em uma seqüência que otimize seu custo de atendimento e tenham suas restrições de janela de tempo respeitadas.

O terceiro e último tipo de mutação proposto neste modelo é a troca de lojas de uma rota para outra. Esta geração permite a formação de rotas atendendo conjuntos diferentes de lojas, buscando sua otimização. Estes dois últimos tipos de mutação são implementados da forma exposta na Figura 3, de acordo com uma probabilidade Pt:

Logo após o processo de geração da nova população, uma rotina simples de correção é aplicada às soluções obtidas para eliminar algumas características indesejáveis. Este procedimento percorre todas as rotas formadas em cada uma das soluções, fazendo-se o seguinte:

- Retiram-se da rota atendida por um determinado veículo as lojas que não podem recebê-lo;
- Retiram-se da rota as lojas que não estão recebendo nenhuma carga; e
- Elimina-se a demanda entregue em posições vazias do vetor solução.

Tal rotina elimina problemas existentes nas soluções construídas a partir das recombinações e mutações, garantindo um menor número de soluções inviáveis formadas a cada geração.

Uma vez que a nova população esteja totalmente constituída, o próximo passo no algoritmo é a avaliação de utilidade de cada um de seus indivíduos, que neste caso representa o custo total para atender à demanda. A utilidade de um indivíduo será o somatório do custo de se percorrer todas as rotas que compõem a solução.

No entanto, como o algoritmo genético baseia-se na recombinação e mutação aleatórias, são geradas constantemente soluções que não atendem às restrições do problema. Assim, para fazer com que o algoritmo convirja para soluções viáveis, atribuiu-se uma penalidade que é somada ao custo total de uma solução para cada restrição que não é atendida. Esta penalidade irá diminuir a utilidade daquelas soluções que não atendem a todas as restrições, diminuindo, assim, sua probabilidade de gerar descendente. Por outro lado, os indivíduos que respeitarem um maior número de restrições serão beneficiados e criarão um maior número de descendentes, garantindo a convergência para soluções implementáveis.

- Para cada variável $pop_{i,j,k}$ com uma probabilidade = Pt;
  - Escolha uma das duas formas de troca com 50% de probabilidade para cada:
    - Caso seja escolhida a troca na mesma rota aleatoriamente:
      - Escolha outra posição na rota aleatoriamente;
      - Inverta suas posições;
    - Caso seja escolhida a troca entre rotas:
      - Escolha outra rota aleatoriamente;
      - Escolha uma posição aleatória nesta rota;
      - Inverta suas posições.

Figura 3. Implementação da mutação.

Penalidades variáveis também são adotadas de acordo com a quantidade de restrições desrespeitadas. Quanto maior o número de restrições desrespeitadas de uma determinada categoria, maior a penalidade atribuída a cada uma delas. Conseqüentemente, quando há um pequeno número de restrições não sendo atendidas, a penalidade marginal diminui. Este mecanismo permite que soluções fortemente inviáveis convirjam rapidamente para soluções viáveis, dado o alto custo associado às penalidades. Da mesma forma, permite que soluções viáveis violem algumas restrições para explorar melhor a sua vizinhança na busca de melhorias.

Todos estes conceitos foram utilizados no cálculo da função utilidade para as seguintes restrições:

- A demanda de uma loja não é totalmente satisfeita;
- A duração de uma rota ultrapassa a jornada máxima de trabalho; e
- As janelas de recebimento não são respeitadas.

Para cada uma destas restrições, foi estabelecida uma penalidade associada cujo valor relativo irá afetar o comportamento do algoritmo até que ele convirja para uma solução viável. Daí em diante, o valor destas penalidades não terá grande impacto em seu desempenho.

O critério de parada utilizado no algoritmo genético foi um número fixo de iterações. Como a cada iteração várias soluções diferentes são testadas, este parâmetro pode afetar significativamente o desempenho do algoritmo. Caso um número pequeno de iterações seja definido, pode não haver tempo suficiente para que boas soluções sejam encontradas. Por outro lado, se forem permitidas muitas iterações, corre-se o risco de desperdiçar muito tempo de processamento sem que melhorias sejam obtidas. Para calibrar este parâmetro, foram feitos vários testes variando o número de iterações. Como o algoritmo genético trabalha com aleatoriedade, muitas vezes, o resultado obtido para um mesmo número de iterações é diferente em duas rodadas. Mesmo assim, a correlação entre as duas variáveis é bastante alta. Observou-se que, a partir de um determinado número de iterações, a qualidade da solução não melhora muito. Na prática, optou-se por estabelecer um limite de 10.000 iterações como critério de parada do algoritmo.

3.4 Algoritmos genéticos paralelos

Os resultados obtidos pelo AG foram bastante satisfatórios, no entanto, espera-se obter melhoras significativas por meio de sua implementação na versão paralela. Pas-

sa-se agora a descrever como este algoritmo foi paralelizado, a forma como foi implementado e os resultados obtidos. Discutem-se os aspectos mais relevantes e todas as adaptações feitas no algoritmo para que este funcionasse em paralelo.

Os métodos de solução implementados, utilizando os Algoritmos Genéticos Paralelos, foram baseados nos trabalhos de Hwang e Briggs (1984), Navaux (1989) e Cantú-Paz (1999).

Existem duas maneiras básicas de se implementar um algoritmo genético de forma paralela, uma delas, trabalhando com uma única população que tem a avaliação dos indivíduos distribuída entres os processadores disponíveis (método global), e outra, na qual cada nó recebe uma subpopulação que, eventualmente, troca indivíduos com as outras (método das ilhas). Neste trabalho, adotou-se o segundo método por ele permitir ao algoritmo explorar um universo maior de indivíduos, possibilitando o aparecimento de soluções melhores num tempo razoável e diminuindo a probabilidade de que convirja para ótimos locais.

A adaptação da versão seqüencial do algoritmo para sua forma paralela foi feita com o auxílio de funções de comunicação entre computadores disponíveis na biblioteca padrão MPI. Assim, como os outros métodos de solução utilizados neste trabalho, todo o algoritmo foi programado na linguagem C.

A paralelização ocorre da seguinte maneira: no início da execução, um dos nós do *cluster*, com o qual o usuário tem interface (aqui chamado de nó 0), faz a leitura do arquivo que contém os dados do problema e, em seguida, aciona os demais nós pelo comando MPI_Init(). A partir daí, cada nó executa um comando para saber quantos nós compõem a rede e qual é o seu número neste conjunto.

Deste momento em diante, cada nó atua de maneira independente, realizando todas as etapas do algoritmo genético tradicional. Quando o algoritmo atinge um número pré-estabelecido de gerações, é executada uma rotina que realiza a migração dos melhores indivíduos da população de um determinado nó para um nó adjacente. Assim, caso existam 5 nós (0, 1, 2, 3, 4) os indivíduos migrarão de 0 para 1, de 1 para 2,... e de 4 para 0, completando o ciclo. Este procedimento é repetido toda vez que o número de gerações alcançar este número fixado de gerações.

No final, todos os nós enviam sua melhor solução para o nó 0, que irá compará-las e exibir como resultado final a solução que apresentar o menor custo total.

A característica do AG Paralelo de trabalhar com múltiplas populações simultaneamente permite uma ampla exploração do espaço de soluções, aumentando as chances de se obter boas soluções. Ao mesmo tempo, a migração de indivíduos de uma população para outra garante que nenhum nó perca muito tempo trabalhando com uma população cujo resultado seja ruim, pois esta será melhorada com a chegada de indivíduos de populações vizinhas. Além disso, o fato de cada nó trabalhar a maior parte do tempo de forma independente acarreta um baixo tráfego de rede, proporcionando *speedups* bastante elevados.

Combinando estas características com a capacidade de processamento fornecida pelos 60 nós do *cluster*, foram obtidos ótimos resultados, que serão apresentados no próximo item.

4. Dados de entrada do modelo

O estudo foi realizado com base em um centro de distribuição, localizado em Osasco (SP), que atende um total de 302 lojas distribuídas em todo o Estado de São Paulo. A maioria das lojas é atendida diariamente.

Todas as entregas de mercadorias são feitas por uma frota terceirizada, composta por três veículos: carreta, *truck* e leve. As capacidades dos veículos encontram-se na Tabela 1.

Com relação às restrições de tipo de veículo, a maioria das lojas (mais de 90%) não recebe carreta e aceita os demais tipos de veículos, devido às restrições operacionais.

O frete pago às empresas transportadoras é calculado com base em regiões geográficas pré-definidas. Cada região possui um custo fixo por viagem, aqui chamado C_i. Caso o caminhão seja designado para atender mais de uma loja na mesma viagem, além do custo C_i (proporcional à loja com maior custo), é cobrado um custo extra para cada outra loja atendida ($Delta_i$). Se as lojas estiverem em regiões cujos custos fixos por viagem sejam diferentes entre si, cobra-se o maior valor. Desta forma, o custo de atender a uma determinada rota pode ser definido como:

$$CR = \max(C_1, C_2,...,C_n) + (N-1)\cdot\max(Delta_1, Delta_2,...,Delta_n) \quad (2)$$

em que N representa o número de lojas atendidas pela rota e CR o custo total da rota.

Para o cálculo das distâncias e dos tempos entre as lojas e o Centro de Distribuição, foi necessária a localização de cada ponto. As coordenadas de cada loja e do centro de distribuição foram obtidas a partir de seus respectivos endereços, com o auxílio de um site de localização (www.computador.com.br). Como as coordenadas estão

Tabela 1. Capacidade dos veículos.

Tipo de veículo	Peso (t)	Volume (m^3)	Paletes
Carreta	22.000	49	28
Truck	11.000	26	14
Leve	4.500	13	7

em latitude e longitude, calcula-se a distância geodésica, conforme Ballou (2001). Para corrigir a distorção entre o valor teórico e o valor prático das distâncias, utilizou-se um fator de correção de 1,27.

A velocidade dos trechos foi determinada com base nos dados históricos do grupo. Comparou-se a distância real percorrida entre dois pontos e o tempo total de percurso (sem incluir os tempos de descarga). A equação que melhor representa esses dados é:

$$V = \min(0{,}0000007 \cdot d^3 - 0{,}008 d^2 + 0{,}3096 \cdot d + 16{,}952,\ 60) \quad (3)$$

Muitas lojas possuem demanda superior à capacidade do maior caminhão, resultando em rotas com carga completa. A Tabela 2 apresenta a demanda de algumas lojas (número de paletes) em um determinado dia.

As janelas de tempo da maioria das lojas são largas (o intervalo de entrega é, em média, de 7,3 horas).

Os dados completos do modelo, além dos resultados para um determinado dia (quarta-feira), podem ser encontrados no site http://br.geocities.com/patricia.belfiore/metaheuristica.html.

5. Análise dos resultados

Todos os métodos de solução descritos no item anterior foram aplicados à resolução do problema de abastecimento de um dos maiores grupos varejistas do Brasil para um conjunto de dados selecionados, representando um dia típico de operação. Para que os resultados possam ser comparados, o modelo proposto (dados de latitude e longitude, cálculo da distância, velocidades, frete, etc.) foi aplicado à solução obtida pela empresa. O dia foi escolhido juntamente com a gerência da empresa por ser uma data representativa para as operações no restante do ano. Além desta data básica, foram analisados outros 6 dias na mesma semana para verificar o impacto na qualidade das soluções decorrentes da mudança no volume, tendo em vista que nos finais de semana este é, normalmente, bem mais reduzido.

Aplicando cada um dos algoritmos propostos, para um determinado dia, e comparando com a solução original da empresa, obtêm-se as economias listadas na Figura 4.

A diferença nos custos da operação obtida pelo Algoritmo Genético Paralelo, que apresentou os melhores resultados entre os métodos testados, foi de aproximadamente R$ 5.100,00 ou 13% em apenas um dia. Se esta economia for projetada para os demais dias do ano, considerando 7 dias por semana, os ganhos alcançam R$ 1,6 milhão por ano, o que justifica o investimento em métodos mais eficazes de roteirização.

Como pode ser observado pela Tabela 3, os métodos de solução propostos, além de proporcionarem uma melhoria significativa no custo total da operação, minimizam o número total de viagens.

Tabela 2. Demanda das lojas.

Loja	Paletes
2	8,0
3	10,5
4	9,5
16	16,0
19	4,0
23	26,0
84	4,0
90	21,0
99	33,0
100	97,5
101	64,5
117	7,5
120	108,5

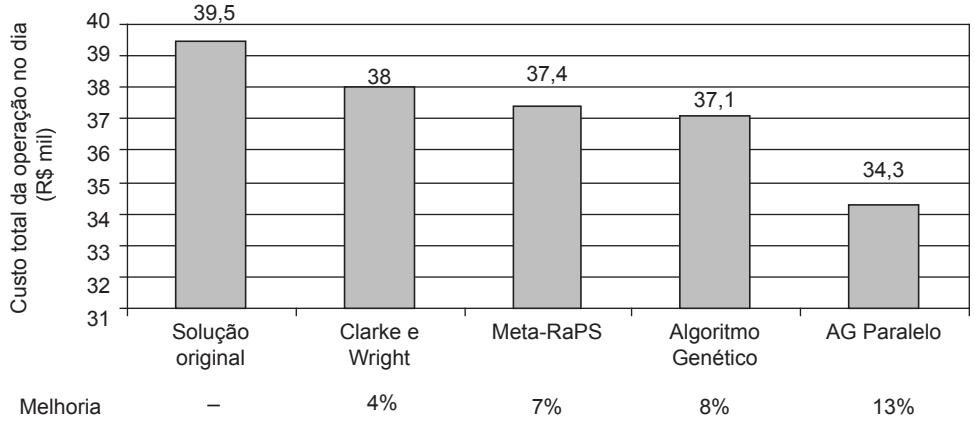

Figura 4. Comparação dos resultados obtidos pelas diferentes soluções.

O tempo médio de processamento em cada rodada varia bastante de acordo com o método de solução utilizado. Para os algoritmos genéticos, este tempo é diretamente proporcional ao número de iterações e varia de acordo com o número de lojas a serem atendidas. No caso da heurística construtiva, ele depende apenas do tamanho do problema, já que é executado em apenas uma iteração. Finalmente, para o Meta-RaPS, este tempo depende do número de iterações e do tamanho do problema, mas é significativamente menor do que no caso dos genéticos. A Tabela 4 apresenta o tempo gasto na resolução dos modelos para cada um dos métodos e dias da semana.

Os roteiros completos de entrega apresentam no máximo 5 lojas. Como a demanda de muitas lojas é superior à capacidade do maior caminhão, entrega-se carga completa.

6. Conclusões e futuras pesquisas

O presente trabalho apresentou um método científico para a resolução de um problema de abastecimento das lojas de um dos maiores grupos varejistas do Brasil, resultando em melhorias significativas em relação ao processo atual. Estas melhorias puderam ser obtidas graças à formulação do problema, de modo a incorporar as restrições operacionais enfrentadas na prática, e à combinação de diferentes técnicas de solução, buscando aproveitar as melhores características de cada uma delas.

Além de apresentar uma forma estruturada de resolução de problemas de roteirização de veículos, o trabalho introduziu o conceito de Computação Paralela, que vem sendo cada vez mais utilizado em diversas áreas do conhecimento, e como pode ser implementado na solução deste tipo de problema. De fato, a combinação de algoritmos eficientes, com a capacidade de processamento proporcionada pela computação paralela garante a obtenção de soluções muito boas num intervalo de tempo compatível com as necessidades da operação.

Este trabalho permite concluir que a utilização de heurísticas construtivas é uma forma simples e rápida de se obter melhorias nas soluções quando comparadas àquelas obtidas empiricamente, fato que ainda ocorre freqüentemente nas empresas. A adoção de metaheurísticas mais sofisticadas e, principalmente, da computação paralela, no entanto, é uma abordagem interessante, pois permite encontrar soluções ainda melhores. Este fato é particularmente útil quando a empresa possui operações de grande porte, em que cada pequeno ganho percentual na qualidade da solução pode significar grandes quantias no resultado financeiro.

Para os casos mais simples, a adaptação da heurística construtiva de Clarke e Wright se mostrou uma ferramenta extremamente prática, tanto pela simplicidade conceitual do método das economias e facilidade de implementação, flexibilidade para se adaptar a diversas restrições operacionais, quanto pelo baixo tempo de processamento. Outro ponto importante é a possibilidade de obter rápidas melhorias nesta heurística pelo algoritmo Meta-RaPS, que a transforma em uma metaheurística pela introdução de elementos aleatórios no processo de formação de rotas. Já nos casos mais complexos, os algoritmos genéticos que permitem a realização de buscas mais amplas no espaço de soluções viáveis e o uso de *clusters* para a paralelização do código e aumento do poder de processamento se mostraram ferramentas bastante poderosas, viabilizando a solução de problemas difíceis e de grande porte.

Finalmente, é importante indicar que os métodos utilizados neste trabalho podem ser objeto de novas pesquisas para que sejam ainda mais aprofundados e adaptados a novos problemas práticos. Estes mesmos métodos podem ser adaptados para sua futura utilização em diferentes categorias de produtos. Além disso, todo trabalho aqui apresentado pode ser implementado com maior abrangência geográfica em problemas com múltiplos depósitos. Alguns novos critérios e restrições também poderiam ser adicionados em futuras implementações, dentre eles a priorização de pedidos.

Tabela 3. Comparação das diferentes soluções.

Método de Solução	Custo total	Número de viagens
Original	R$ 39.500,00	135
Clarke e Wright	R$ 38.000,00	130
Meta-RaPS	R$ 37.500,00	132
Algoritmo genético	R$ 37.100,00	128
AG Paralelo	R$ 34.400,00	125

Tabela 4. Tempos de processamento.

Método	Tempo de processamento (s)						
	Dom.	2ª	3ª	4ª	5ª	6ª	Sab.
Clarke e Wright	0,6	3,1	20,0	9,0	14,4	12,2	9,0
Meta RaPS	12,2	111,7	392,8	265,8	361,3	291,4	231,7
AG	388,8	1.398,0	1.888,8	1.342,8	1.581,0	1.808,4	1.533,6
AG Paralelo	627,0	1.019,0	2.698,0	1.503,0	1.439,0	1.312,0	1.113,3

Referências Bibliográficas

BALLOU, R. H. **Gerenciamento da Cadeia de Suprimentos:** Planejamento, Organização e Logística Empresarial. 4. ed. São Paulo: Bookman, 2001. 532 p.

BJARNADÓTTIR, Á. S. **Solving the Vehicle Routing Problem with Genetic Algorithms**. 2004. 127 f... Dissertação (Mestrado em Engenharia) – Informatics and Mathematical Modelling, Technical University of Denmark, Odense, 2004. Disponível em <http://www2.imm.dtu.dk/pubdb/views/edoc_download.php/3183/pdf/imm3183.pdf>.

CANTÚ-PAZ, E. Implementing Fast and Flexible Parallel Genetic Algorithms. In: Chalmers, L. (Ed.). **Practical Handbook of Genetic Algorithms**. Boca Raton, FL: CRC Press, v. 3, p. 65-84, 1999.

CLARKE, G.; WRIGHT, J. W. Scheduling of Vehicles from a Central Depot to a Number of Delivery Points. **Operations Research**, v. 12, n. 4, p. 568-581, 1964.

DROR, M.; TRUDEAU, P. Savings by Split Delivery Routing. **Transportation Science**, v. 23, n. 2, p. 141-145, 1989.

DROR, M.; TRUDEAU, P. Split Delivery Routing. **Naval Research Logistics**, v. 37, n. 3, p. 383-402, 1990.

GOLDEN, B. L.; ASSAD, A.; LEVY, L.; GHEYSENS, F. The fleet size and mix vehicle routing problem. **Computers & Operations Research**, v. 11, n. 1, p. 49-65, 1984.

HO, S. C.; HAUGLAND, D. A tabu search heuristic for the vehicle routing problem with time windows and split deliveries. **Computers & Operations Research**, v. 31, n. 12, p. 1947-1964, 2004.

HWANG, K.; BRIGGS, F. A. **Computer Architecture and Parallel Processing**. 2. ed. New York: McGraw-Hill International Editions, 1984.

MORAGA, R. J., et al. Solving the Vehicle Routing Problem Using the Meta-RaPS Approach. In: INTERNATIONAL CONFERENCE ON COMPUTERS AND INDUSTRIAL ENGINEERING, 29, 2001, Montreal. **Proceedings of 29th ICCIE**. Montreal: ICCIE, 2001. p. 1-6.

NAVAUX, P. O. A. Introdução ao Processamento Paralelo. **RBC - Revista Brasileira de Computação**, v. 5, n. 2, p. 31-43, 1989.

OMBUKI, B.; ROSS, B. J.; HANSHAR, F. Multi-objective Genetic Algorithms for Vehicle Routing Problem with Time Windows. **Applied Intelligence**, v. 24, n. 1, p. 17-30, 2006.

GENETIC ALGORITHMS AND PARALLEL COMPUTING FOR A VEHICLE ROUTING PROBLEM WITH TIME WINDOWS AND SPLIT DELIVERIES

Abstract

The present work considers the use of metaheuristics and parallel computing to solve a real problem of vehicle routing involving a heterogeneous fleet, time windows and split deliveries, in which customer demand can exceed vehicle capacity. The problem consists of determining a set of economical routes that meet each customer's needs while still being subject to all the constraints. The strategy adopted to solve the problem consists of an adaptation of the constructive heuristics proposed by Clarke & Wright (1964) as the initial solution. More sophisticated algorithms are then applied to achieve improvements, such as parallel genetic algorithms supported by a cluster of computers. The results indicate that the basic constructive heuristic provides satisfactory results for the problem, but that it can be improved through the use of more sophisticated techniques. The use of the parallel genetic algorithm with multiple populations and an initial solution, which presented the best results, reduced the total operational costs by about 10% compared with the constructive heuristic, and by 13% when compared with the company's original solutions.

Keywords: vehicle routing problem, time windows, split deliveries, metaheuristics.