Dissertação de Mestrado

Universidade Estadual de Campinas (UNICAMP)

Faculdade de Engenharia Elétrica e de Computação (FEEC)

Departamento de Engenharia de Sistemas (DENSIS)

Resolução de um Problema Dinâmico de Programação de Máquinas Paralelas com Custo de Troca de Ferramentas Dependente da Seqüência e Restrições de Tempo

Aluno: Claudio Fabiano Motta Toledo

Orientador: Dr. Luiz Manoel Aguilera

Co-orientador: Prof. Dr. Paulo Morelato França

Banca: Prof. Dr. Antônio Clécio F. Thomaz

Prof. Dr. Rafael S. Mendes

FICHA CATALOGRÁFICA ELABORADA PELA
BIBLIOTECA DA ÁREA DE ENGENHARIA - BAE - UNICAMP

T575r

Toledo, Claudio Fabiano Motta
Resolução de um problema dinâmico de programação de máquinas paralelas com custo de troca de ferramentas dependente da seqüência e restrições de tempo . / Claudio Fabiano Motta Toledo.--Campinas, SP: [s.n.], 1999.

Orientador: Luiz Manoel Aguilera
Dissertação (mestrado) - Universidade Estadual de Campinas, Faculdade de Engenharia Elétrica e de Computação.

1. Programação. 2. Pesquisa operacional. 3. Transportes. 4. Heurística. 5. Otimização matemática. I. Aguilera, Luiz Manoel. II. Universidade Estadual de Campinas. Faculdade de Engenharia Elétrica e de Computação. III. Título.

RESUMO

O Problema de Roteamento de Veículos (PRV) procura estabelecer uma eficiente distribuição de bens de forma a atender as demandas existentes. Os atuais avanços em tecnologia de informação como rádio transmissores, telefonia celular, sistemas de localização via satélite, estão alterando o cenário em que um PRV pode ocorrer permitindo, por exemplo, a atualização de dados e localização de veículos em tempo real. Um PRV será considerado dinâmico caso seus dados não sejam conhecidos pelo usuário a priori e atualizados simultaneamente a determinação ou execução do conjunto de rotas. Um Problema de Roteamento Dinâmico de Veículos (PRDV) será estabelecido e um método de resolução, chamado algoritmo MORSS, será adaptado para resolver instâncias deste PRDV. Em seguida, um Problema Dinâmico de Programação (PDP) também será estabelecido e o algoritmo MORSS adaptado para resolver instâncias deste PDP. Um segundo método, baseado em heurísticas de busca em vizinhança e inserção, também será proposto para resolver as instâncias do PDP. O trabalho se propõe a resolver dois diferentes tipos de problemas dinâmicos procurando avaliar a adaptabilidade e desempenho do algoritmo MORSS enquanto método de resolução. No caso do PDP, o desempenho de um segundo método também é analisado e comparado ao desempenho obtido pelo algoritmo MORSS.

ABSTRACT

The Vehicle Routing Problem (VRP) is the efficient distribution of products in order to attend customer requirement. Recently, the advances in information technology as radio transmission, cellular telephone, localization systems by satellite, are altering the scenarios in that VRP occurs and allowing update of information and vehicle localization occur in real time. The VRP is dynamic if the inputs of the problem are known by the decision-maker and are updated concurrently with the determination or execution of the route's set. A Dynamic Vehicle Routing Problem (DVRP) will be established and a solution's method, called MORSS algorithm, will be adapted to solve DVRP instances. Next, a Dynamic Scheduling Problem (DSP) will be established and the MORSS algorithm will be adapted to solve DSP instances. A second method, based in heuristics of neighborhood search and insertion, will be also proposed to solve DSP instances. This work proposes to solve two different dynamic problems searching to evaluate the MORSS algorithm adaptability and performance as resolution method. In the PDP, the performance of the second method proposed also will be analyzed and compared with the performance obtained by MORSS algorithm.

FINACIAMENTO E RECURSOS COMPUTACIONAIS

Este trabalho contou com o financiamento da Fundação de Amparo à Pesquisa do Estado de São Paulo (FAPESP), processo número 95/8306-1, que forneceu bolsa de mestrado durante o período de 24 meses. Foram utilizados os recursos computacionais do Departamento de Engenharia de Sistemas (DENSIS) da Faculdade de Engenharia Elétrica e de Computação (FEEC) e do Instituto de Automação (IA) da Fundação Centro Tecnológico para Informática (CTI).

AGRADECIMENTOS

Agradeço a todos aqueles que direta ou indiretamente colaboraram para que este trabalho pudesse ser realizado. Agradeço ao meu orientador Dr. Luiz Manoel Aguilera da Fundação Centro Tecnológico para Informática (CTI) pela oportunidade que me concedeu ao aceitar me orientar. Também agradeço ao professor Dr. Paulo Morelato França da Faculdade de Engenharia Elétrica de UNICAMP pela atenção dedicada a este trabalho.

Agradeço ao meus companheiros do Laboratório de Otimização pelo ambiente agradável de trabalho, em especial a Denise Satto, Pablo e Regina, Débora e Alexandre. A outros companheiros da FEEC como João Viana, Marcos, Moacir, Ricardo Valença, Alda e tantos outros. Ao amigo Adriano, agradeço o companheirismo que me dedicou principalmente na reta final deste trabalho

Agradeço o apoio prestado pelos meus amigos Jerônimo, Delano e Oliva. As minhas amigas Gisele, Ana, Patrícia e a todos os outros companheiros do instituto de computação que também torceram por este trabalho.

Agradeço ao meu irmão Anderson pelo exemplo de dedicação, a minha irmã Valdinéia e meu grande amigo Júlio César por acreditarem e me apoiarem sempre. A minhas irmãs Adriana e Marta, meu irmão Jonathas e minha mãe Sônia pelo carinho e força. A minha avó Esther pelo amor e dedicação que sempre teve por mim. A minha tia Shirley por tudo que sempre fez e faz pelo meu sucesso.

Agradeço também a minha sogra Mariângela pelo apoio incondicional que direta ou indiretamente sempre me forneceu. A Tatiane, Viviane, Gabriela e todos de Ribeirão que torcem pelo meu trabalho.

A minha esposa Christiane Neme Campos Toledo eu agradeço por sempre estar ao meu lado. Você é responsável direta por este trabalho estar sendo concluído. As palavras companheira, amiga e esposa juntas não conseguem expressar o significado da sua presença que, desde de 1992, só contribuiu para melhorar simplesmente tudo.

ÍNDICE

Capítulo 4: Resolução de um Problema Dinâmico de Programação de Máquinas Paralelas com Custo de Troca de Ferramentas Dependente da Seqüência e Restrições de Tempo 66

INTRODUÇÃO

Os avanços tecnológicos na área de sistemas informativos estão tornando cada vez mais comum o acesso à informação em tempo real. Os desenvolvimentos de hardware e softwares permitiram a popularização dos computadores do tipo PC, os Notebooks e as Estações de Trabalho. O crescimento da rede mundial de computadores, a comunicação via satélite, a telefonia celular, entre outros meios, podem ser conjugados para criarem situações onde as informações passam a ser obtidas e atualizadas constantemente. O acesso rápido e constante à informação demonstra-se de fundamental importância para empresas ou indústrias no competitivo mercado mundial.

Os atuais avanços tecnológicos estabelecem novos cenários para diversos problemas que deixam de estar inseridos em um contexto estático e passam a ser tratados em um contexto dinâmico. Assim, o contexto dinâmico pode ser entendido como aquele em que todas as informações sobre determinado problema ou situação não são totalmente conhecidas a priori e poderão ser constantemente atualizadas.

Um problema clássico como o de roteamento de veículos já pode ter novos ingredientes adicionados ao seu cenário habitual. O veículo pode estar executando sua rota para atender demandas quando novas demandas passam a ser integradas ao problema ou informações sobre demandas existentes são alteradas. Isto ocorre porque existe tecnologia capaz de permitir uma rápida localização e contato com os veículos, possibilitando mudanças nas rotas em execução. Porém, para determinar as melhores alterações nestas rotas, também devem existir métodos de resolução capazes de atualizá-las levando em conta o cenário dinâmico em que o problema está inserido.

Nesta dissertação, será abordado o contexto dinâmico envolvendo um Problema de Roteamento de Veículos e um Problema de Programação. Desta forma, Um Problema Dinâmico de Roteamento de Veículos (PRDV) e um Problema Dinâmico de Programação (PDP) serão estabelecidos e algumas instâncias serão solucionadas. Para isso, um método de resolução, chamado algoritmo MORSS, será adaptado para resolver estes problemas. O algoritmo MORSS foi desenvolvido para solucionar um problema de roteamento envolvendo o transporte de cargas por navios em uma situação com características dinâmicas.

A primeira adaptação realizada no algoritmo MORSS permitirá sua utilização na resolução do PRDV estabelecido. O objetivo é avaliar a adaptabilidade do método quando soluciona um PRDV diferente daquele para o qual foi idealizado. A segunda adaptação permite avaliar tanto a adaptabilidade quanto o desempenho do método na resolução de um PDP. Para estabelecer um critério de comparação, foi desenvolvido um segundo método baseado em heurísticas de busca em vizinhança e inserção para solucionar o PDP. Os resultados obtidos na resolução das instâncias do PDP por estes dois métodos são comparados e permitem estabelecer uma análise do desempenho destas duas metodologias.

CAPÍTULO 1

PROBLEMA DE ROTEAMENTO DINÂMICO DE VEÍCULOS (PRDV)

1.1 - Introdução

O uso de veículos para distribuir bens ou serviços em diferentes locais surge em diversas situações práticas, tais como na determinação de rotas para entrega de cargas ou no estabelecimento de roteiros para realização de manutenção e inspeção de equipamentos. Estas situações implicam na necessidade de escolha dos veículos mais adequados para realizar as tarefas, bem como na otimização das rotas a serem percorridas.

Estes problemas se enquadram na área de pesquisa denominada de Problema de Roteamento de Veículos (PRV). Os estudos neste campo incluem a representação do problema através de modelos matemáticos seguido do desenvolvimento de métodos para sua resolução. Existem atualmente ferramentas computacionais de apoio a tomada de decisões em Gestão de Sistemas Logísticos ou Gestão de Sistemas de Transporte entre outras, onde os problemas são modelados e resolvidos como um PRV.

Dentro do contexto de um PRV, uma possível classificação consiste em separar os problemas em estáticos e dinâmicos. Um PRV estático é aquele cujas informações disponíveis sobre o problema não são alteradas, seja durante a execução do método que estabelecerá as rotas para os veículos, seja durante a própria execução das respectivas rotas pelos veículos. Por outro lado, um PRV é dinâmico quando as informações disponíveis podem e são alteradas durante a execução do método ou das rotas. Neste caso, os processos de determinação e execução das rotas ocorrem simultaneamente, tornando necessária a existência de métodos capazes de atualizar constantemente as informações disponíveis e adaptar as soluções já encontradas para satisfazer os novos dados do problema.

Atualmente, a área de PRV dinâmico tem sido amplamente pesquisada como resultado dos atuais avanços em Tecnologia de Informação que vão desde um simples rádio transmissor, passando pela telefonia celular até sistemas de localização via satélite; incluindo transmissores, receptores, "Geographic Position System" (GPS), computadores de bordo, etc.

Desta forma, não seria difícil imaginar uma central que controlasse de forma dinâmica uma frota de veículos durante a coleta e entrega, por exemplo, de correspondência expressa. A central receberia em tempo real informações sobre novas correspondências a serem coletadas ou entregues, devendo gerenciar as informações recém chegadas, avaliar as diversas rotas em execução pelos veículos e decidir onde as novas coletas ou entregas

poderiam ser inseridas da melhor forma possível. Ao final deste processo, a central poderia informar aos veículos as novas tarefas e as alterações a serem realizadas nas diversas rotas.

Neste capítulo será realizada uma breve introdução ao Problema de Roteamento de Veículos (seção 1.2), para que em seguida o Problema de Roteamento Dinâmico de Veículos (seção 1.3) seja apresentado. As principais características do problema dinâmico são destacadas (seção 1.4) e, finalmente, alguns métodos de resolução existentes na literatura são mencionados (seção 1.5).

1.2 - Problema de Roteamento de Veículos

O Problema de Roteamento de Veículos (PRV) surge da necessidade de se planejar a distribuição de bens ou a realização de serviços construindo as melhores rotas possíveis de serem executadas por uma frota de veículos. Estas rotas começam em um ou vários depósitos e percorrem um número conhecido de cidades ou clientes, geograficamente dispersos, onde bens ou serviços precisam ser entregues ou realizados.

O problema descrito aparece em várias situações práticas:

- Coleta de correspondência em caixas de correio.
- Rotas para vendedores atenderem clientes.
- Roteiro para técnicos realizarem serviços de manutenção e inspeção de equipamentos.
- Roteiros para ônibus escolares.

Um modelo simplificado de um PRV é formado por uma frota de veículos que parte de um depósito central e atende clientes em várias cidades, executando uma rota específica ou diferenciada para cada veículo.

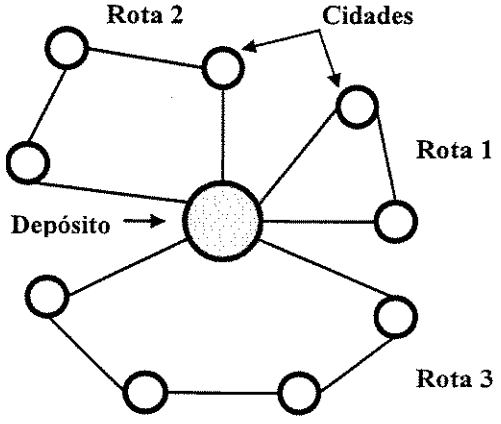


Figura 1.1 – Exemplo de solução para um PRV

Na Figura 1.1 é apresentada uma solução para um modelo simplificado do PRV, onde três rotas distintas são construídas para veículos localizados em um depósito. As rotas não apresentam cidades em comum e os veículos partem do depósito, visitam um certo número de cidade e retornam ao depósito. As seguintes restrições fazem parte ou podem ser adicionadas ao modelo apresentado na Figura 1.1 (Christofides, 1985):

- Todas as rotas construídas para os veículos começam e terminam em um mesmo depósito.
- Cada cidade é visitada por apenas um veículo e uma única vez.
- A frota de veículos pode ser homogênea ou não homogênea. Será homogênea quando todos os veículos forem do mesmo tipo, ou seja, apresentarem idênticas características como capacidade, velocidade, etc.
- Existência de volumes ou pesos associados às cidades. Logo, as rotas designadas aos veículos não poderão incluir cidades cuja soma total destes volumes ou pesos ultrapasse uma capacidade máxima estabelecida para os veículos.
- O número de cidades em um trajeto é limitado. Desta forma, as rotas estabelecidas para cada veículo deverão passar por um número mínimo de cidades e/ou não poderão ser muito extensas, obedecendo um limite máximo de cidades.
- O tempo total gasto em uma rota não poderá ultrapassar um certo limite de tempo. O tempo total gasto em uma rota será o tempo gasto pelo veículo ao deslocar-se entre as cidades adicionado ao tempo de permanência. Este tempo de permanência refere-se ao tempo gasto durante o atendimento aos clientes em cada cidade.
- Pode ser fornecido ao problema intervalos de tempo $[a_i , b_i]$ associados a uma cidade i, onde o veículo não poderá chegar antes de a_i ou sair depois de b_i. São as chamadas restrições do tipo janelas de tempo ("time-windows").
- Por algum motivo inerente ao modelo real, poderão ser estabelecidas restrições de precedências entre as cidades i e j, determinando que o veículo atenda clientes em j antes de passar por i.

Outras restrições não são diretamente adicionadas ao modelo simplificado de um PRV apresentando aspectos diferenciadores mais relevantes, porém o problema a ser resolvido continua sendo de roteamento de veículos. Por exemplo:

- Múltiplos depósitos, permitindo que o veículo retorne para um depósito diferente do qual partiu.
- A construção das rotas pode ser realizada obedecendo uma ordem de urgência no atendimento de cada cliente ou uma freqüência habitual em que surgem tarefas em determinadas cidades.
- Cada veículo pode transportar diferentes tipos de bens que podem ser requisitados em diferentes quantidades pelos clientes. Logo, as rotas construídas procuram combinar da melhor forma possível os tipos de bens e as quantidades requeridas com os modelos de veículos disponíveis e as respectivas capacidades de transporte.
- O problema pode ter como objetivo minimizar o número de veículos a ser adicionado à frota existente para atender os clientes, minimizar o número de clientes não atendidos, minimizar o tempo ou distância total usado nas rotas ou mesmo uma combinação destes objetivos.

Isto posto, o PRV com algumas das características anteriormente mencionadas poderá ser representado através de uma formulação matemática. Existem diversos tipos de formulações conforme as restrições pertinentes ao PRV. A fim de ilustrar, será apresentada uma formulação matemática proposta por Fisher e Jaikumar (1981). Considerem-se as seguintes informações:

- Sejam i=2,..., n os índices das cidades e i=1 o depósito.
- Sejam k=1, ..., m os índices dos veículos.
- Em cada cidade i existe uma demanda q_i e o custo de viajar entre as cidades i e j é c_{ij}. A capacidade de cada veículo k é C_k.
- A principal restrição a ser considerada consiste na realização do roteamento, construindo rotas distintas para cada veículo, que comece e termine no depósito e não ultrapasse a capacidade de cada veículo.
- O problema estará solucionado quando as rotas construídas minimizam o custo total da viagem.

As variáveis de decisão a serem consideradas na formulação matemática são:

$$x_{ijk}=\begin{cases}1 & \text{Se o veículo } k \text{ visita a cidade } j \text{ depois da cidade } i,\\ 0 & \text{Caso contrário.}\end{cases}$$

$$y_{ik}=\begin{cases}1 & \text{Se o cidade } i \text{ é visitada pelo veículo } k,\\ 0 & \text{Caso contrário.}\end{cases}$$

A correspondente formulação matemática será:

$$\text{Min} \sum_{ij} c_{ij} \sum_{k} x_{ijk} \tag{1.1}$$

s.a.

$$\sum_{k} y_{ik} = 1 \qquad i=2,..., n \tag{1.2}$$

$$\sum_{k} y_{ik} = m \qquad i=1, \tag{1.3}$$

$$\sum_{i} q_i y_{ik} \leq C_k \qquad k=1,..., m \tag{1.4}$$

$$\sum_{j} x_{ijk} = \sum_{j} x_{jik} = y_{ik} \qquad i=1,...,n \qquad k=1,..., m \tag{1.5}$$

$$\sum_{i,j \in S} x_{ijk} \leq |S| - 1 \qquad S \subseteq \{2,..., n\} \qquad k=1,..., m \tag{1.6}$$

$$y_{ik} \in \{0, 1\}, x_{ijk} \in \{0, 1\} \qquad i, j=1,...,n \qquad k=1,..., m \tag{1.7}$$

A função objetivo é representada por 1.1 e deixa claro a busca por rotas com custo total mínimo. A restrição 1.2 garante que cada cidade seja visitada por um único veículo, pois

somando-se y_{ik} para todos os veículos k em uma mesma cidade i existirá um único veículo com $y_{ik}=1$ tal que 1.2 seja satisfeita. Como esta restrição deve ser obedecida considerando-se todas as cidades $i=2,\ldots, n$; cada uma estará sendo visitada por apenas um veículo.

A restrição 1.3 estabelece que todos os veículos passem pelo depósito. Neste caso, a soma das variáveis de decisão y_{1k} deve ser igual ao número de veículos m porque todos os veículos visitam o depósito ($i=1$). A restrição 1.4 impõe que as capacidades dos veículos sejam respeitadas porque quando $y_{ik}=1$ teremos uma demanda q_i sendo satisfeita na cidade i. Portanto, somando-se estas demandas em relação a um mesmo veículo k sua capacidade C_k não poderá ser ultrapassada.

A restrição 1.5 assegura que um veículo ao chegar a determinada cidade também parta desta cidade. Isto ocorre porque, se a variável de decisão $y_{ik}=1$, deverá existir uma cidade j qualquer visitada pelo veículo k antes que o mesmo visite a cidade i ($x_{jik}=1$) e também deverá existir outra cidade j qualquer visitada pelo veículo k após deixar a cidade i ($x_{ijk}=1$).

Finalmente, a restrição 1.6 não permite que existam ciclos no trajeto de uma rota. Neste caso, existe um subconjunto S de cidades que são visitadas pelo veículo k e, portanto, apresentam valores x_{ijk} iguais a 1. Se interpretarmos os valores $x_{ijk}=1$ como relativos aos arcos (i, j) pertencentes ao caminho percorrido pelo veículo k, a restrição $|S|-1$ (cardinalidade do conjunto S menos 1) impõe a não existência de ciclos neste caminho.

Conhecidas as características do PRV e estabelecida sua formulação matemática, o próximo passo é levantar os métodos que são empregados na sua resolução. Os PRV's pertencem a classe de problemas NP-Difíceis (Garey e Johnson, 1979) e diversos métodos para solucioná-los foram desenvolvidos, podendo ser separados em exatos e heurísticos.

Os métodos exatos sempre encontram soluções ótimas para estes problemas, porém o gasto computacional envolvido cresce exponencialmente em função do número de clientes. Os métodos heurísticos procuram encontrar soluções ótimas, porém não há garantia de que sejam encontradas. Uma apresentação específica de métodos exatos e heurísticos que solucionam o PRV pode ser encontrado em Christofides (1985) e Laporte (1992).

1.3 - Problema de Roteamento Dinâmico de Veículos

Um PRV procura encontrar distribuições eficientes de bens (mercadorias, passageiros, cargas, etc.) criando rotas, por exemplo, capazes de minimizar o custo total desta distribuição e ainda atender determinadas restrições. As informações necessárias ao processo de construção das rotas são conhecidas antecipadamente e dificilmente alteradas, levando o veículo a executar um trajeto previamente estabelecido e, geralmente, não sujeito a mudanças.

Atualmente, existe uma alteração nos cenários onde um PRV pode ocorrer. Os recentes avanços tecnológicos como telefonia celular, transmissão de dados via satélites e redes de computadores permitem realizar em tempo real as atualizações nos dados disponíveis de um PRV. O processo de construção ou execução das rotas pode estar acontecendo simultaneamente à chegada de novas informações.

O acesso à informação em tempo real era raro ou inexistente há algumas décadas atrás e hoje é cada vez mais comum. Tornam-se inúmeros os problemas passíveis de constantes alterações nos seus dados e, portanto, cada vez menos úteis soluções que fiquem presas a um cenário estático. Para ilustrar, serão apresentados alguns problemas práticos com características dinâmicas encontrados na literatura:

- Uma firma prestadora de serviços (gás, água, eletricidade, etc.) mantém uma rede de pronto atendimento para os casos de falha em seu serviço. As falhas ocorrem de forma aleatória em relação ao tempo (em qualquer instante) e espaço (em qualquer local). A firma dispõe de uma frota de veículos que transporta equipes de reparos. Estas equipes são levadas de uma garagem para os locais onde as falhas no serviço acontecem. As decisões de roteamento da frota são tomadas em tempo real, baseadas na ocorrência das falhas e, talvez, levando em conta alguma caracterização de futuras falhas nos serviços. As equipes de reparo gastam uma quantidade aleatória de tempo resolvendo o problema antes de estarem livres para atender a próxima falha. A firma deverá operar sua frota de forma que minimize o tempo médio gasto para solucionar as falhas no serviço. (Bertisimas e Ryzin, 1991).
- Uma firma de transporte de cargas emprega 5 funcionários encarregados de estabelecer e gerenciar as rotas dos caminhões. Dois funcionários trabalham a noite e são responsáveis por planejar as rotas que serão executadas durante o dia. Os outros três funcionários mantêm em execução as rotas já estabelecidas e, em alguns casos, corrigem as mesmas em resposta a situações imprevistas que possam ocorrem, entrando em contato com os caminhões através de equipamento telefônico instalado nos veículos. Um sistema de apoio a tomada de decisão foi desenvolvido levando em conta o planejamento e as alterações que podem ocorrer nas rotas. (Lysgaard, 1992).
- Uma frota de navios transporta cargas volumosas (como óleo, cimento, metais, carvão, etc.) a partir dos centros de produção (refinarias, fábricas, terminais de embarque, etc.) para locais de consumo (locais de armazenamento, terminais de desembarque, etc.). As demandas nestes pontos de consumo não são conhecidas com antecedência. Além das distâncias a serem percorridas, o tempo de viagem também depende das condições climáticas prevalecentes. A programação e as rotas destes navios devem ser estabelecidas de forma contínua para que possam atender as demandas. (Ronen, 1986).
- Uma frota de veículos deve atender demandas por serviços que chegam por telefone ou outro meio de comunicação a uma central. Essas demandas tornam-se conhecidas em tempo real e precisam ser satisfeitas o mais cedo possível. Na central, o responsável pela tomada de decisões deverá decidir, também em tempo real, qual veículo enviar para atender determinada demanda e como modificar a rota deste veículo (Thompson, 1993).

- Containers são descarregados de navios e carregados em outros navios. O movimento dos containers é feito por guindastes e as informações sobre o movimento das cargas a ser executado pelos guindastes chegam em tempo real. Os movimentos dos guindastes precisam ser otimizados a medida que as informações chegam, ou seja, em tempo real (Aslidis, 1991).

Nas situações apresentadas, o aspecto que se destaca é o não conhecimento, no início do processo, de todas as informações necessárias para a solução do problema. Os dados são dinamicamente renovados, ou seja, durante o decorrer do tempo antigas informações podem ser alteradas e/ou novos dados são incorporados impedindo que se estabeleça uma solução fixa e definitiva. O problema é otimizado baseado nas informações disponíveis em determinado momento e, quando ocorrem mudanças, torna-se necessário encontrar novas soluções em muitos casos partindo-se das soluções anteriormente estabelecidas.

A importância destes problemas está ligada à necessidade das empresas em economizar, ganhar eficiência e aumentar sua competitividade no mercado. Nos países da União Européia em 1990 havia 13 milhões de caminhões operando. Somente na Alemanha, país que detém 10% das toneladas/quilometro transportadas na União Européia, a renda dos fretes alcançou 35 bilhões de dólares. Uma modesta melhoria na eficiência das rotas, minimizando custos nas suas alterações ou ajustes, poderia gerar rentável redução dos gastos (Psaraftis, 1995). Também o setor de serviços como correios, coleta de lixo, transporte de pessoal, entre outros, já têm acesso a tecnologia capaz de permitir que atuem em tempo real, ganhando eficiência e rentabilidade.

Os avanços tecnológicos foram os grandes responsáveis pela existência de cenários dinâmicos em problemas que até então só podiam ser tratados de forma estática. Abaixo apresentamos alguns destes avanços (Psaraftis, 1995):

- **EDI (Eletronic Data Interchange)**: Classificam-se com este nome as ferramentas que permitem transferência eletrônica de informações entre computadores, nas transações comerciais e administrativas, através do estabelecimento de um padrão comum capaz de permitir compatibilidade nas trocas de dados. Na área de logística e distribuição, o uso de EDI aumenta a velocidade de comunicação e controle de todos os aspectos das operações logísticas. São considerados EDI sistemas desenvolvidos para identificação de cargas, como os códigos de barra, e sistemas automáticos para manuseio, ordenamento e controle do fluxo de cargas.
- **GIS (Geographic Information Systems)**: Trata-se de sistemas para mapeamento eletrônico de rodovias capazes de mostrar telas gráficas de alta resolução. Apresenta informações estáticas (como dados geográficos, redes de transporte, etc.) e dinâmicas (posicionamento dos veículos, surgimento de demandas, condições do tráfego, etc.). O desenvolvimento de sistemas do tipo GIS tem recebido atenção especial nos últimos anos na América do Norte, Europa e Japão.
- **IVHS (Intelligent Vehicle-Highway Systems)**: Abrange o desenvolvimento de sistemas para melhor controle do fluxo de veículos durante tráfego intenso. Sistemas IVHS procuram integrar avanços tecnológicos e algoritmos para solucionarem

problemas que vão desde transporte de materiais com altos riscos de locomoção em rodovias até o controle de congestionamento de tráfego.

Todos estes avanços, além de outros que não citamos, procuram tornar viável um rápido processamento de informações, diminuem custos e são capazes de integrar diversos componentes tecnológicos. Desta forma, com os obstáculos técnicos sendo superados, torna-se viável a implementação de sistemas eficientes de roteamento capazes de operar em ambientes dinâmicos. Para o desenvolvimento destes sistemas, os métodos de otimização a serem utilizados ou desenvolvidos devem incluir as características dinâmicas inerentes ao problema.

Isto posto, pode ser estabelecido um conceito mais preciso do que será considerado neste trabalho um Problema de Roteamento Dinâmico de Veículos (PRDV). Um PRV será dinâmico se as informações (entradas de dados) tornam-se conhecidas ou atualizadas ao mesmo tempo em que o conjuntos de rotas está sendo determinado ou executado (Psaraftis, 1995).

Para efeito de comparação, será citado outro conceito de problema dinâmico fornecido por Powell *et al* (1995) que define como dinâmico o problema com um ou mais de seus parâmetros obtidos em função do tempo. Essa definição inclui dois tipos de problema, onde o primeiro tipo envolve a mudança dinâmica de dados, ou seja, é caracterizado por dados que são constantemente alterados. O segundo inclui problemas com dados dependentes do tempo que tornam-se conhecidos antecipadamente. Nessa segunda categoria está incluso o Problema de Roteamento de Veículos com Janelas de Tempo (PRVJT).

No PRVJT existem janelas de tempo ("time-windows") que estabelecem determinadas restrições. Por exemplo, podemos ter janelas de tempo $[a_i , b_i]$ associadas a uma cidade i, com a restrição de que o veículo não poderá atender determinada demanda antes do instante a_i ou depois de b_i. Neste caso, a existência da demanda já é um fato conhecido antecipadamente e a única restrição refere-se a quando poderá ser satisfeita, ou seja, em função do intervalo de tempo $[a_i, b_i]$).

O conceito estabelecido neste trabalho para o PRDV comporta apenas a mudança dinâmica de dados, ou seja, o primeiro tipo de problema dinâmico considerado na definição de Powell *et al* (1995). Desta forma, as restrições do tipo janela de tempo não caracterizam um problema como dinâmico na conceituação de Psaraftis (1995) que será a conceituação considerada no decorrer deste trabalho.

Também previsões probabilísticas ou qualquer outra forma de antecipar o comportamento das variáveis do problema não qualificam um problema como dinâmico, pois permitem antecipar o comportamento do sistema. O PRDV considerará que as informações simplesmente tornam-se conhecidas de forma imediata em relação ao tempo, portanto, apenas quando explicitamente fornecidas passam a ser considerada dentro do cenário em que o problema está inserido.

Para exemplificar a distinção entre problemas estáticos e dinâmicos, dentro dos conceitos estabelecidos, vamos analisar dois problemas: Problema do Caixeiro Viajante Probabilístico (PCVP) e o Problema do Caixeiro Viajante Dinâmico (PCVD).

O Problema do Caixeiro Viajante (PCV) é um famoso problema de otimização combinatória, tanto pela sua simplicidade para ser definido como pela sua dificuldade de ser solucionado. Partindo de sua cidade, um caixeiro viajante deseja visitar várias cidades uma única vez e retornar ao final para sua cidade. Deseja fazer isso com o menor custo de viagem possível.

O PRV pode ser encarado como uma extensão do caixeiro viajante. Por exemplo, seja considerado um conjunto de veículos, cada um com uma capacidade fixa, que precisam visitar um conjunto de clientes para entregar determinados bens. Uma vez designado um conjunto de clientes para cada veículo, estes deverão percorrer os clientes a ele atribuídos. O problema envolvendo a construção de rotas para cada veículo e seu respectivo conjunto de clientes constitui um PCV.

No Problema do Caixeiro Viajante Probabilístico (PCVP), supõe-se que os tempos de viagem sejam determinísticos e correspondam ao custo que deverá ser minimizado. Neste problema, as demandas de cada cidade ocorrem com probabilidade p ou não ocorrem com probabilidade $1-p$. Ao resolver o PCVP, primeiramente será obtida uma rota R, através de todas as cidades, que minimize o tempo total do percurso. Porém, antes de R ser executada, o veículo será informado de quais cidades realmente estão requisitando demanda e quais não estão, ou seja, o veículo recebe antecipadamente a informação de quais cidades apresentam ou não demandas a serem atendidas. Desta forma, uma rota R' será obtida a partir de R, apenas excluindo as cidades onde não há demanda (Jaillet, 1991).

Do exposto e levando-se em conta a definição adotada para um problema dinâmico, conclui-se que o PCVP é um problema estático porque durante a determinação da rota R todas as informações (os tempos de viagem entre as cidades e o comportamento probabilístico das demandas) são conhecidas e não mudam. Além disso, na determinação da rota R' temos acesso às informações de quais cidades apresentam ou não demandas antes da execução da rota pelo veículo.

No Problema do Caixeiro Viajante Dinâmico (PCVD), as demandas de serviços são geradas em diferentes cidades seguindo um processo de Poisson. Os tempos de viagem entre as cidades onde surgem demandas são conhecidos e determinísticos. O caixeiro viajante gasta um tempo também conhecido no atendimento de cada cliente. O problema consiste, em determinar qual é a estratégia de roteamento que minimiza a média, para todas as demandas, do tempo de espera até que o atendimento da demanda seja completado. Também poderíamos desejar saber qual é a estratégia de roteamento que maximiza a média esperada do número de demandas atendidas por unidade de tempo (Psaraftis, 1988).

Este problema é dinâmico, pois parte das entradas necessárias para resolvê-lo (ou seja, quais demandas atuais precisam ser satisfeitas) é revelada simultaneamente com a

determinação da rota. Desta forma, é impossível produzir previamente uma rota ótima a ser executada pelos veículos. No máximo consegue-se determinar uma estratégia que estabeleça qual ação deverá ser tomada em função do estado atual do sistema.

1.4 - Características do PRDV

Abaixo é apresentado um breve sumário das principais diferenças capazes de distinguir um PRDV do PRV estático. As diferenças apresentadas têm implicação imediata nas técnicas de resolução dos PRDV's.:

- **Dimensão do tempo é essencial**: No PRV estático geralmente o tempo é proporcional à distância e não é considerado explicita e separadamente na formulação e resolução do problema. Em toda situação que trata de roteamento dinâmico de veículos, possuindo ou não restrições de tempo, a dimensão do tempo é essencial. No mínimo, torna-se necessário saber a localização dos veículos em determinado instante de tempo para determinação de sua rota, principalmente quando novos dados são fornecidos ou determinadas alterações nas informações passam a ser conhecidas.
- **Problema tem um "final aberto"**: Em um PRV, geralmente, a rota de um veículo começa na garagem, percorre os clientes e retorna para a garagem. É possível, por exemplo, estimar um limite para a duração da rota. No PRDV, as constantes alterações na rota inicial, para atender as novas requisições, não permitem saber quando o veículo poderá retornar ao depósito. As rotas estabelecidas em um contexto dinâmico vinculam-se a determinado cenário ou situação do problema, sendo constantemente readaptadas para atender as mudanças em tempo real que estabelecem outros cenários ou situações para o problema. Assim, dificilmente haverá uma rota fixa, representando uma solução final para o problema, capaz de satisfazer as mudanças que estão por acontecer. Por isso o problema apresenta um "final aberto".
- **Informações futuras podem ser imprecisas ou desconhecidas**: As informações em um PRDV são precisas para eventos próximos de ocorrer e incertas para eventos considerados futuros. Por exemplo, clientes ou demandas que ocorrem em instantes de tempo mais afastados do momento atual têm mais chances de sofrer alterações no seus dados do que aqueles que imediatamente precisam ser atendidos.
- **Eventos recentes são mais importantes**: Como conseqüência do exposto acima, a prioridade será dada às informações já disponíveis e/ou às novas informações relativas a clientes que precisem ser atendidos imediatamente.
- **Mecanismos de atualização de informações são essenciais**: Os dados relativos aos clientes, rotas e veículos são constantemente alterados. Portanto, técnicas de gerenciamento de estrutura de dados que facilitem a atualização das entradas do problema e adeqüem de forma eficiente as mudanças necessárias tornam-se vitais ao desenvolvimento de uma aplicação que resolva um PRDV.
- **Reseqüenciamento e Reatribuição devem ser garantidos**: No PRDV, devido as constantes alterações, a seqüência de uma rota deve ser facilmente reordenada para

permitir eficiente inserção, troca e retirada de clientes na ordem de atendimento. Veículos também podem deixar de atender determinados clientes para os quais foram inicialmente atribuídos, para que sejam reatribuídos a outros. Todas essas tarefas devem ser possíveis e facilmente realizáveis em um ambiente dinâmico.

- **Mecanismos de adiamento indefinido são essenciais**: Adiamento indefinido significa a possibilidade de uma demanda ter seu atendimento adiado indefinidamente, por exemplo, se apresentar uma localização geográfica desfavorável (alto custo associado à viagem, tempo de viagem muito grande, etc.). Mecanismos que permitam adiamento indefinido são necessários, pois uma restrição que obrigasse o pleno atendimento de todas as demandas tornaria a obtenção de soluções impraticável em um PRDV. A necessidade de prioridade para eventos recentes, conforme já mencionado, torna essencial que outras demandas possam sofrer certo adiamento. Porém, determinadas penalidades na função objetivo ou algumas restrições ao não atendimento, após um certo número de adiamentos, podem ser incorporadas à resolução do problema.
- **Funções objetivo podem ser diferentes**: Objetivos como minimizar a distância total percorrida ou a duração total de uma rota que ocorrem em um PRV estático poderão não ter sentido em um cenário dinâmico. No PRDV, as funções objetivo são mais relacionadas com a produtividade. Por exemplo, funções objetivo que procurem maximizar o número médio de demandas atendidas por veículos, minimizar o tempo médio de espera no atendimento dos clientes, minimizar o atraso total na entrega ou recolhimento dos bens, etc.
- **Restrições de tempo podem ser diferentes**: Esta diferença pode ser exemplificada usando as restrições de janela de tempo. Restrições do tipo janela de tempo em um PRDV tendem a ser violadas e por isso não podem ser rígidas. Mesmo no PRV estático com janelas de tempo, estas restrições podem ser violadas acarretando penalidade, por exemplo, na sua função objetivo. A principal diferença é que no PRDV estas restrições podem ser violadas ou constantemente atualizadas com o decorrer do tempo, ou seja, o intervalo $[a_i, b_i]$ pode ser alterado para um novo $[a'_i, b'_i]$ após determinada alteração nos dados do problema.

1.5 Métodos de Resolução Existentes na Literatura

Uma abordagem utilizada no desenvolvimento de métodos que procuram resolver um PRDV consiste em adaptar os desenvolvidos para o PRV estático. Psaraftis (1988) menciona a existência de duas formas de adaptar um algoritmo estático para o caso dinâmico. A primeira procura inicialmente executar o algoritmo estático para os dados disponíveis do PRDV e executá-lo novamente toda vez que ocorrem alterações. Desta forma, um conjunto de novas rotas é gerado a cada atualização de dados, procurando preservar as decisões anteriormente tomadas. Um dos problemas apresentados relaciona-se com o tempo gasto nas repetidas execuções do algoritmo, pois os resultados precisam ser fornecidos em tempo real.

A segunda forma de se adaptar algoritmos estáticos procura manusear as atualizações dinâmicas através de operações locais, envolvendo a execução de uma heurística de inserção depois que o algoritmo estático for executado. Neste contexto, executa-se o algoritmo estático somente para iniciar o processo, em seguida, operações locais encarregam-se de atualizar as rotas quando ocorrem as subseqüentes atualizações nos dados.

O método mais rápido de operações locais é a heurística de inserção, onde uma nova demanda ou cliente é inserido na rota em execução, sem perturbar as seqüências já planejadas. Heurísticas do tipo busca em vizinhança que realizam permutações nas seqüências das demanda ou clientes a serem atendidos também podem ser usadas. Neste caso, as heurísticas do tipo busca em vizinhança são executadas depois que a inserção da nova demanda ou cliente for realizada, já que procuram reseqüênciar as rotas de forma a melhorar o resultado fornecido pelas heurísticas de inserção.

Uma implementação utilizando algoritmo estático é fornecida por Bell *et al* (1983) para o roteamento de uma frota de veículos que precisa distribuir produtos armazenados em um depósito central. A implementação é separada em dois módulos, no primeiro um algoritmo estático gera, usando uma heurística, um conjunto de possíveis rotas para os veículos. Para isso, leva em conta a localização dos clientes, quantidade de demandas e capacidade dos veículos. Ainda neste módulo, é resolvido o problema de selecionar, entre as rotas geradas, aquela que será executada efetivamente pelo veículo. O segundo módulo é responsável por gerenciar as mudanças subseqüentes que ocorrem em tempo real.

Rego e Roucairol (1995) desenvolveram um algoritmo de duas fases para resolver um problema real envolvendo o despacho de caminhões tanque. O problema ocorre em uma companhia que transporta matérias primas localizadas em 7 terminais (pontos de embarque), envolvendo cerca de 200 pontos de entrega através da Europa. Trata-se de uma combinação de problemas de roteamento e programação ("scheduling"), sendo classificado como um "Multi-Terminal Truck Dispatching Problem". Envolve roteamento devido ao estabelecimento das rotas que cada caminhão executará. O problema de programação refere-se ao planejamento do tempo em que cada rota será executada.

Os caminhões são completamente carregados nos pontos de embarque. As cargas são transportadas diretamente do ponto de embarque para o ponto de desembarque, portanto, existem restrições de precedência a serem obedecidas. Trata-se de um problema dinâmico porque o roteamento e a programação são realizados em tempo real e envolvem demandas que não são conhecidas antecipadamente.

Na primeira fase do algoritmo, determina-se uma solução inicial viável usando um método de decomposição. Esta decomposição consiste na resolução de uma seqüência de problemas menores e mais tratáveis, onde o resultado de um problema torna-se a entrada de dados do seguinte. Na segunda fase, o algoritmo começa com a solução obtida durante a primeira fase e a cada iteração a solução é melhorada usando o método heurístico de Busca Tabu.

A Busca Tabu está baseada em movimentos específicos que tentam melhorar duas ou três rotas a cada passo. Os movimentos básicos utilizados consistem na inserção e troca de arcos no grafo representativo do problema. A composição desses movimentos fornece outros movimentos que tornam possível realizar a busca em regiões de soluções inviáveis.

Uma abordagem que não envolve problemas reais, mas voltada para considerações teóricas pode ser encontrada nos trabalhos de Dror *et al* (1989) e Bertsimas e van Ryzin (1991 e 1993). Dror *et al* apresenta um trabalho envolvendo o Problema de Roteamento de Veículos com Demandas Estocásticas (PRVDE). Essa classe de problemas geralmente envolve o roteamento de uma frota de veículos para atender um conjuntos de demandas sob a hipótese de que o tamanho da demanda é uma variável aleatória. Neste trabalho, foi incluída a formulação de um PRVDE dinâmico, onde a demanda atual torna-se conhecida por ocasião da chegada do veículo ao cliente.

O veículo poderá atender a demanda do cliente e partir para o próximo ou não atendê-la, caso o tamanho da demanda esteja além de sua capacidade de atendimento, retornando para o depósito ou indo para outro cliente. Os autores formularam o problema como um processo de decisão seqüencial sob incertezas, onde o objetivo é produzir uma política que determine a melhor ação a ser tomada em função do estado atual do sistema. Nenhuma experiência computacional foi realizada, mas os autores observaram que devido ao número de possíveis estados ser enorme, a solução irá requerer alguma forma de relaxação do problema.

Bertsimas e van Ryzin (1991) estabelecem o "Dynamic Traveling Repairman Problem" (DTRP). Neste problema, demandas por serviços chegam no decorrer do tempo, de acordo com um processo de Poisson com taxa λ. Estas demandas são independentes e estão uniformemente distribuídas em uma região convexa e limitada A. Cada demanda requer uma quantidade de tempo aleatório, independente e uniformemente distribuído, gasto no atendimento e com duração média s. O problema consiste em encontrar políticas para o roteamento dos veículos que minimizem o tempo médio gasto com as demandas que aparecem no sistema.

Usando resultados que envolvem probabilidade, teoria de fila e otimização combinatória; os autores obtiveram interessantes resultados teóricos para esse problema. Na situação envolvendo um único veículo com capacidade ilimitada, os resultados provam que com poucas demandas surgindo no decorrer do tempo (tráfego leve, $\lambda \rightarrow 0$) existe uma política ótima para o roteamento dos veículos. Esta política ótima consiste na localização do veículo no ponto médio da região A e na realização do atendimento das demandas obedecendo a seguinte ordem: Primeira demanda por serviço a chegar, primeira a ser atendida ("First In First Out"). Após o atendimento de cada demanda o veículo retorna para o ponto médio da região.

No caso de muitas demandas no decorrer do tempo (tráfego intenso), os autores demonstram que existem políticas com tempo do sistema finito para todo $\rho = s\lambda$ (fração de

tempo gasto no atendimento local). O tempo do sistema é formado pelo tempo de espera por atendimento da demanda mais o tempo gasto durante o atendimento.

Entre as políticas que se comportam bem no tráfego intenso, destaca-se a de particionamento, a do caixeiro viajante e a da vizinhança mais próxima. A política de particionamento consiste em dividir a região A em n quadrados atendendo cada sub-região seqüencialmente, onde as demandas que surgem em cada quadrado são satisfeitas na ordem do primeiro a chegar, primeiro a ser atendido. As demais políticas são descritas em Bertsimas e van Ryzin (1991), bem como as devidas demonstrações dos resultados mencionados. Em Bertsimas e van Ryzin (1993) os resultados e políticas apresentados anteriormente são estendidos para o caso de múltiplos veículos com capacidade limitada.

No capítulo seguinte será apresentado um algoritmo proposto por Psaraftis (1988) para resolver um PRDV. A escolha deste algoritmo como objeto de estudo desta dissertação teve como critério o fato de ter sido desenvolvido para resolver um problema com característica dinâmica, ou seja, não se trata de uma adaptação de um algoritmo estático. Além disso, diversos aspectos do algoritmo como *Horizonte de Tempo*, *Funções Utilidades de Designação*, entre outros; procuram explorar o contexto dinâmico em que o problema está inserido.

CAPÍTULO 2

MÉTODO DE RESOLUÇÃO PARA O PROBLEMA DE ROTEAMENTO DINÂMICO DE VEÍCULOS

2.1 - Introdução

Neste capítulo será apresentado um método de resolução para o Problema de Roteamento Dinâmico de Veículos (PRDV) chamado algoritmo MORSS. Este método foi selecionado após uma pesquisa bibliográfica, sendo que algumas das metodologias avaliadas foram apresentadas na seção 1.5 do capítulo anterior.

A escolha do algoritmo MORSS baseou-se no fato de ter sido desenvolvido para solucionar um problema com características dinâmicas, ou seja, não foi utilizado inicialmente em um Problema de Roteamento de Veículos (PRV) estático e adapatado, posteriormente, para o caso dinâmico. Além disso, o método apresenta características interessantes na forma de resolução do PRDV, tais como resolução de subproblemas usando a idéia do Horizonte Rolante, avaliação dos custos ou benefícios através de Funções Utilidades de Designação; entre outras.

Psaraftis *et al* (1985) apresentaram o algoritmo MORSS em um sistema computacional desenvolvido para resolver um problema real de roteamento e programação. Este problema envolve o transporte de cargas em navios numa situação de emergência, ou seja, em um cenário de constantes atualizações nos dados disponíveis e não conhecimento prévio de todas as informações.

MORSS significa "MIT Ocean Routing and Scheduling System" e refere-se ao sistema computacional desenvolvido para solucionar este problema. Dentro do sistema MORSS, existe um conjunto de passos responsáveis por executar as principais decisões do método. Este conjunto de passos é chamado algoritmo MORSS e fazem parte de um subsistema dentro do sistema MORSS.

Na seqüência deste capítulo será apresentado, em linhas gerais, o problema que motivou o desenvolvimento do algoritmo (seção 2.2) e o sistema MORSS (seção 2.3). As características principais do método serão comentadas, destacando-se a idéia do Horizonte de Tempo Rolante (seção 2.4), as Funções Utilidades de Designação (seção 2.5) e o Problema de Designação relacionado (seção 2.6). Finalmente, será apresentado um exemplo numérico (seção 2.7) para ilustrar a execução dos principais passos do algoritmo.

2.2 - Problema que Motivou o Método.

MORSS é um programa computacional que resolve um problema de roteamento e programação, ou seja, um problema de designação de cargas a navios em uma situação de emergência tal que:

(a) as cargas devem alcançar seus destinos dentro de um limite de tempo prescrito;
(b) algumas medidas de desempenho do sistema devem ser otimizadas;
(c) determinadas restrições devem ser satisfeitas.

O PRDV utilizado como exemplo na literatura referente ao sistema MORSS (Psaraftis *et al*, 1985, e Psaraftis, 1988) é constituído por uma frota de navios e um conjunto de cargas que tornam-se disponíveis em diferentes instantes. Os navios estão distribuídos em vários pontos do oceano e as cargas permanecem em portos de embarque até que sejam transportadas aos seus respectivos portos de desembarque.

A Figura 2.1 apresenta um possível cenário do problema em determinado instante. Os círculos representam a localização de quatro portos e os retângulos a localização de dois navios (N_1 e N_2) disponíveis. Existem quatro cargas, denotadas por q_i, que precisam ser transportadas dos portos de embarque para os portos de desembarque. O porto de embarque da carga q_i é representado na Figura 2.1 por $+q_i$ e o porto de desembarque por $-q_i$.

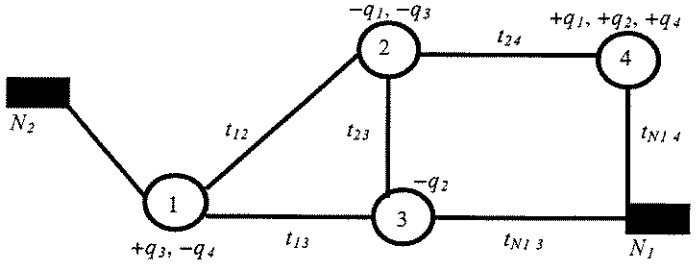


Figura 2.1 - Exemplo de um Cenário do Problema Considerado

Por exemplo, as cargas q_1, q_2 e q_4 estão disponíveis para embarque no porto 4 (assinaladas com $+q_1$, $+q_2$ e $+q_4$) onde a carga q_1 deverá ser desembarcada no porto 2 (assinalada com $-q_1$), a carga q_2 no porto 3 ($-q_2$) e a carga q_4 no porto 1 ($-q_4$). Os arcos ligando os portos apresentam pesos (t_{ij}) que podem significar a distância ou o tempo de viagem entre os mesmos. Os pesos no arcos que ligam navios a alguns portos ($t_{Ni\,j}$) também representam as distâncias ou o tempo de viagem.

O problema é determinístico, ou seja, não existem informações de caráter probabilístico que permitam prever o comportamento do sistema. Por exemplo, não há informação a respeito da freqüência com que novas cargas surgirão ou sobre a probabilidade de onde se localizarão as cargas entre os diversos portos.

As cargas e navios não são homogêneos, logo, existem diferentes tipos de navios e cargas. Desta forma, determinadas cargas somente poderão ser transportadas por certo tipo de navio. As cargas são divisíveis, ou seja, cada navio compatível para o transporte de determinada carga transportará todo seu conteúdo ou apenas uma fração deste.

Os navios podem embarcar o conteúdo de diversas cargas, mas existe um limite relacionado a sua capacidade de transporte que não poderá ser ultrapassado. O número de navios também pode variar, ou seja, navios podem ser acrescentados ou retirados durante a resolução do problema.

As rotas executadas podem conter diversos portos de embarques e desembarques, não sendo obrigatório primeiro realizar o embarque de todas as cargas para depois realizar os desembarques. O problema também não estabelece restrições de precedência que obriguem o embarque e/ou desembarque prévio de determinada carga antes do embarque e/ou desembarque de outra.

As informações relativas às cargas são:

- Porto de embarque (PE): Número do porto onde a carga deverá ser embarcada.
- Porto de desembarque (PD):Número do porto onde a carga deverá ser desembarcada.
- Tempo de Embarque Mínimo (TEMin): Representa o instante de tempo a partir do qual a carga estará disponível no porto para ser embarcada.
- Tempo de Desembarque Mínimo (TDMin): Representa o instante de tempo a partir do qual a carga poderá ser desembarcada no respectivo porto de desembarque.
- Tempo de Desembarque Máximo (TDMax): Representa o instante de tempo máximo no qual a carga deverá ser desembarcada sem que ocorra atraso.

O TEMin e TDMin não poderão ser violados, pois o problema considera que antes do TEMin a carga não estará disponível no PE e antes do TDMin o PD não estará preparado para realizar o desembarque. Muitas vezes o tempo disponível entre o TEMin e TDMax não poderá ser satisfeito devido as constantes alterações que, em um contexto dinâmico, ocorrem no problema. Por isso, entre a possibilidade de não desembarcar a carga por violar seu TDMax ou realizar o desembarque com atraso, o problema permite o desembarque atrasado associando uma penalização a estes desembarques, conforme será explicado na seção 2.5.

Os portos apresentam uma capacidade máxima associada ao número de navios que podem permanecer realizando embarque e/ou desembarque de cargas, ao mesmo tempo em determinado porto. Caso um navio chegue a um certo porto com nova operação de embarque e/ou desembarque, ultrapassando a capacidade operacional do porto, deverá ficar aguardando até que alguma operação em andamento seja concluída. Isto poderá ocasionar congestionamento nos portos.

A função objetivo do problema penaliza os atrasos na entrega das cargas, o baixo aproveitamento dos navios e o congestionamento dos portos. As situações de atraso na

entrega e congestionamento dos portos já foram mencionadas anteriormente. Um baixo aproveitamento dos navios ocorre quando os mesmos executam a maior parte da rota muito abaixo da sua capacidade máxima de transporte ou demoram muito tempo em operações de embarque e desembarque, tornando-se indisponíveis para novas operações durante longos períodos.

O problema a ser resolvido consiste no estabelecimento de rotas para os navios, determinando as melhores atribuições possíveis de cargas aos navios em termos de menor atraso no desembarque das cargas, bom aproveitamento dos navios e não congestionamento dos portos. O caráter dinâmico do problema aparece na necessidade de que as rotas sejam construídas e executadas sem conhecimento prévio de todas as informações relativas ao problema. Afinal, novas informações sobre cargas e navios chegam ao problema em tempo real provocando constantes alterações nos dados.

A descrição do PRDV nesta seção foi realizada em linhas gerais, ou seja, de forma a permitir um melhor entendimento de como MORSS foi desenvolvido para resolver um problema com estas características. Um aprofundamento no problema prático com maiores detalhes das restrições específicas deste problema poderá ser encontrado em Psaraftis *et al* (1985).

2.3 - Sistema e Algoritmo MORSS

O sistema MORSS é composto por quatro subsistemas: READIN, DISPLAY, SEEDS e SCHEDULE. Cada subsistema executa uma função específica que, associadas, formam uma ferramenta computacional capaz de obter soluções para o problema apresentado na seção anterior.

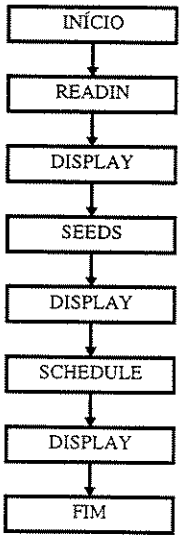


Figura 2.2 - Seqüência de Execução do sistema MORSS

A Figura 2.2 apresenta a seqüência lógica de execução dos diversos subsistemas de MORSS. O subsistema READIN realiza a leitura das informações do problema e inicia as estruturas de dados de todo o sistema. Desempenha um papel de iniciação do sistema, por isso, sua execução precede a dos demais subsistemas. DISPLAY permite a interação do usuário com o sistema. Neste subsistema existem diversos menus capazes de permitir uma visualização dos dados disponíveis ou fornecidos pelo sistema em gráficos e tabelas, facilitando a análise dos mesmos pelo usuário.

SEEDS inicia o processo de designação de cargas aos navios. Isto é feito usando o critério de se atribuir apenas um único tipo de carga para cada navio, determinando as rotas iniciais de cada navio. O subsistema SCHEDULE realiza as posteriores alterações nestas rotas, levando em conta as novas informações que chegam. Desta forma, SEEDS precede SCHEDULE na seqüência de execução apresentada na Figura 2.2. Observa-se que os subsistemas READIN e DISPLAY desempenham papel auxiliar, pois suas funções estão restritas a iniciação do método e interação com o usuário. A resolução efetiva do problema inicia-se com o subsistema SEEDS e concentra-se principalmente em SCHEDULE.

SEEDS tem papel relevante dentro do sistema MORSS porque realiza a primeira designação de cargas aos navios, estabelecendo a rota inicial dos mesmos. Essas rotas são obtidas resolvendo-se um problema de designação inicial de cargas aos navios. Neste problema, o objetivo é maximizar a soma dos benefícios calculados na atribuição das cargas aos navios com a restrição de que apenas um único tipo de carga poderá ser designado para cada navio. O processo de cálculo destes benefícios é feito utilizando uma função matemática chamada Função Utilidade de Designação que será detalhada na seção 2.5 Ao final da execução do subsistema SEEDS, teremos cada navio sendo enviado para determinada região onde será responsável pelo embarque e desembarque de uma única carga.

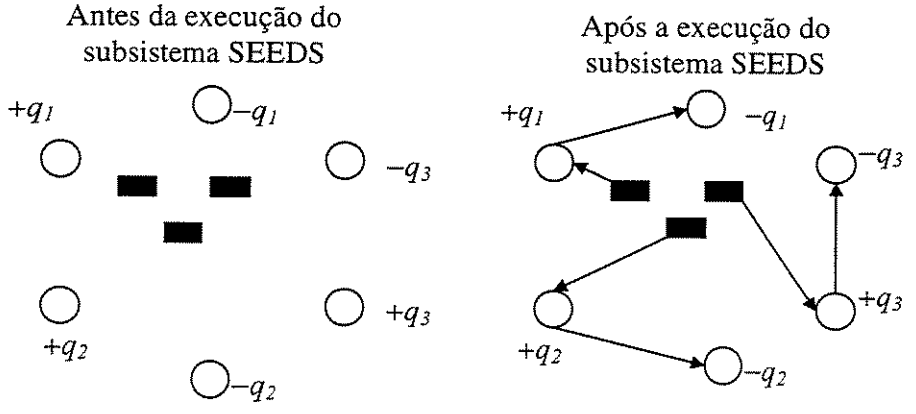


Figura 2.3 - Exemplo de resultado fornecido por SEEDS

A Figura 2.3 ilustra uma atribuição inicial realizada por SEEDS. A três navios serão atribuídas três cargas inicialmente disponíveis, onde a rota de cada navio incluirá um

único tipo de carga. A rotina, ao estabelecer a rota inicial, também acaba determinando a região onde os navios estarão inicialmente localizados.

As decisões tomadas em SEEDS, segundo Psaraftis *et al* (1985), têm influência na determinação das rotas subseqüentes porque as designações das cargas aos navios posteriomente realizadas pelo subsistema SCHEDULE serão baseadas na atribuição inicial realizada em SEEDS. Desta forma, no decorrer da determinação das rotas, os navios tenderão a embarcar e desembarcar cargas próximas da região para a qual foram inicialmente enviados durante a execução de SEEDS.

A Figura 2.4 apresenta uma possível rota obtida a partir daquela determinada na Figura 2.3, com novas cargas q_4, q_5 e q_6 que tornaram-se disponíveis para embarque e desembarque. Neste caso, os veículos que atendiam as cargas q_1, q_2 e q_3 tenderão a concentrar-se nas regiões determinadas inicialmente pelo subsistema SEEDS.

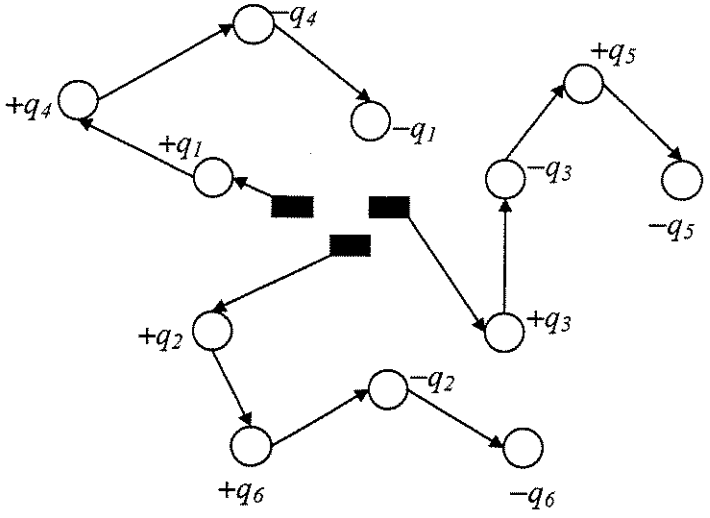


Figura 2.4 - Rotas obtidas por SCHEDULE partindo das rotas fornecidas por SEEDS

O principal subsistema de MORSS é SCHEDULE que executa todos os comandos responsáveis por manipular as restrições e características do problema. Os principais passos executados neste módulo foram apresentados em Psaraftis *et al* (1985) e Psaraftis (1988), onde receberam o nome de algoritmo MORSS. Os métodos que serão apresentados nos capítulos 3 e 4 foram baseados exclusivamente nos passos do algoritmo descrito a seguir.

O algoritmo MORSS baseia-se no conceito do Horizonte Rolante, onde avaliam-se as alterações ocorridas no problema através de sucessivos horizontes ou intervalos de tempo de tamanho fixo. Para realizar esta avaliação, são utilizadas funções chamadas Utilidades de Designação que calculam o benefício de designar determinada carga a determinado navio.

O algoritmo seleciona cargas e navios disponíveis dentro de cada horizonte de tempo considerado e, através das Utilidades de Designação, mede o benefício obtido caso cada carga selecionada fosse inserida na rota de cada navio. Em seguida, um Problema de Designação é estabelecido e resolvido, onde os valores calculados pela Função Utilidade de Designação serão maximizados.

A solução encontrada fornecerá uma tentativa de designação, pois indica quais cargas poderão ser atribuídas aos navios com a garantia de fornecer a melhor resposta ao Problema de Designação. Essa solução será refinada através de critérios que procuram selecionar cargas com menor TEMin e cuja inserção nas rotas não prejudique o transporte das cargas já embarcadas pelos navios, neste e em outros horizontes de tempo. Logo, selecionam-se na solução do Problema de Designação as cargas que interagem favoravelmente com outras já designadas aos navios. Os diversos passos executados pelo método estão descritos abaixo.

ALGORITMO MORSS

Passo 0: Inicie a localização de navios disponíveis.
Inicie uma lista mestre de cargas não designadas.
Selecione o tamanho L do horizonte de tempo.
Selecione uma fração a ($0<a<1$) e ajuste $k=1$, $t_1=0$.

Passo 1: Ajuste o próximo horizonte de tempo $[t_k, t_k+L]$.
Forme uma lista de cargas candidatas para designação (todas as cargas na lista mestre cujos TEMin estão entre t_k e t_k+L).

Passo 2: Calcule as utilidades de designação para todo par cargas candidatas/navios (Detalhes na seção 2.5).

Passo 3: Crie e otimize um Problema de Designação usando utilidades de designação como custo dos arcos. A designação de cargas aos navios obtida como solução do problema estabelece uma tentativa de designação no horizonte de tempo $[t_k, t_k+L]$ (Detalhes na seção 2.6).

Passo 4: Retorne à lista mestre de cargas não designadas:
(i) todas as cargas não designadas pelo Passo 3;
(ii) todas as cargas designadas cujos TEMin's estão entre t_k+aL e t_k+L;
(iii) todas as cargas designadas que interagem desfavoravelmente com outras também designadas neste horizonte de tempo para um

mesmo navio ou com cargas já designadas ao navio em iterações anteriores.

Passo 5: Execute um rolamento do horizonte de tempo. Ajuste t_{k+1} para ser o menor TEMin entre todas as cargas disponíveis na lista mestre. Atualize a localização dos navios em t_{k+1}. Ajuste $k=k+1$ e retorne ao Passo 1.

No Passo 1, as cargas que tornam-se disponíveis no horizonte de tempo estabelecido formam o conjunto de cargas candidatas a serem inseridas nas rotas dos navios. A avaliação de quais cargas candidatas serão atribuídas aos navios é feita calculando-se a utilidade de designação no Passo 2. Este cálculo é realizado simulando-se a inserção de cada carga candidata na rota de cada navio disponível. Os valores calculados serão os benefícios a serem maximizados no Problema de Designação estabelecido no Passo 3.

A iteração desfavorável mencionada no Passo 4 ocorre quando a solução do Problema de Designação indica que mais de uma carga deva ser atribuída ao mesmo navio. Como as utilidades de designação foram calculadas avaliando-se a atribuição de cada carga individualmente, a atribuição conjunto de várias cargas a um mesmo navio poderá afetar o benefício da atribuição de algumas destas cargas. Desta forma, um critério adotado é recalcular a utilidade de designação das demais cargas atribuídas ao mesmo navio, após uma delas ter sido inserida na rota do navio, avaliado se houve redução no novo valor obtido. Maiores detalhes sobre cargas que interagem desfavoravelmente e sobre os critérios de refinamentos adotados serão apresentados nas seções 3.3.3 e 4.5.3.

2.4 - Horizonte Rolante

A idéia do Horizonte Rolante consiste na determinação de uma seqüência de intervalos ou horizontes de tempo, dentro dos quais os dados são avaliados. Inicialmente, é fixado um horizonte de tempo e todas as informações disponíveis ao problema dentro desse período são avaliadas. Em seguida, define-se outro horizonte, sucessivo ao anterior, onde o problema será reavaliado considerando-se as novas informações que se tornam conhecidas.

Ao usar o Horizonte Rolante, procede-se como se o problema dinâmico fosse constantemente "fotografado", ou seja, cria-se um cenário estático dentro de cada horizonte considerado. Porém, como esses intervalos de tempo são sucessivos, as situações e decisões anteriormente tomadas são periodicamente reavaliadas permitindo adaptá-las para que atendam às alterações ocorridas.

Na Figura 2.5, t_k é o tempo corrente (instante atual) considerado na $k^{ésima}$ iteração do algoritmo. Os parâmetros L e a são fornecidos pelo usuário, onde o primeiro determina o tamanho do horizonte de tempo e a, $0<a<1$, fornece uma fração deste horizonte.

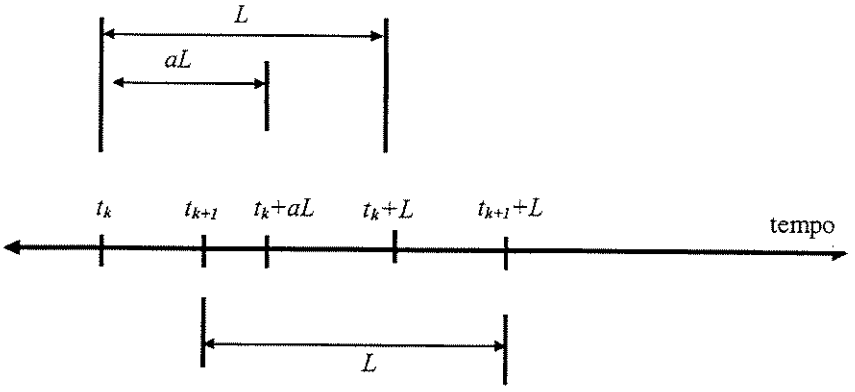


Figura 2.5 - Horizonte Rolante

Dentro de cada horizonte de tempo, MORSS considera as informações relativas às cargas cujo momento do embarque (TEMin) esteja contido num intervalo de tempo partindo do instante atual t_k até t_k+L. Assim, são avaliados não só embarques imediatos, relativos a cargas que estão prontas para embarque em t_k, mas também embarques futuros referentes ao período que vai até t_k+L. Informações além do instante t_k+L não serão consideradas na $k^{ésima}$ iteração do algoritmo. Para cada horizonte de tempo, o algoritmo MORSS define um conjunto de cargas candidatas:

$$C=\{q/q\in Q_k \text{ e TEMin}\in [t_k, t_k+L]\}.$$

O conjunto Q_k contém todas as cargas cujos dados já estão disponíveis na $k^{ésima}$ iteração. O conjunto de cargas candidatas C será formado pelas cargas pertencentes a Q_k cujo Tempo de Embarque Mínimo (TEMin) esteja em $[t_k, t_k+L]$. Depois de estabelecer C, as Utilidades de Designação são calculadas para todas as cargas $q\in C$ e um Problema de Designação é resolvido (Passo 2 e Passo 3 do algoritmo). A solução do Problema de Designação fornecerá o conjunto de cargas que poderão ser atribuídas aos navios, mas apenas aquelas com TEMin$\in [t_k, t_k+aL]$ serão designadas de forma permanente aos navios e, desta forma, inseridas nas respectivas rotas.

O algoritmo MORSS faz uma previsão, ao utilizar o conceito de Horizonte Rolante, pois avalia todas as cargas com TEMin$\in [t_k, t_k+L]$, mas as decisões definitivas são tomadas apenas para os eventos mais próximos de acontecer. Por isso, são designadas de forma permanente apenas as cargas com TEMin$\in [t_k, t_k+aL]$. Assim, coloca-se menos ênfase em dados relativos a eventos futuros (cargas com TEMin$\in [t_k+aL, t_k+L]$), pois o caráter dinâmico do problema aumenta a possibilidade de serem alterados, e priorizam-se os dados referentes a acontecimentos mais imediatos.

Após todos esses procedimentos, o horizonte de tempo sofre uma rolagem na iteração seguinte e o próximo intervalo passará a ser $[t_{k+1}, t_{k+1}+L]$, onde t_{k+1} recebe o valor do menor TEMin das cargas existentes e ainda não atribuídas aos navios. Cargas que pertenceram ao conjunto C formado em $[t_k, t_k+L]$ poderão voltar a ocorrer no conjunto C obtido em $[t_{k+1}, t_{k+1}+L]$, pois existe uma conexão entre os horizontes que se sucedem, conforme pode ser observado na figura 2.5, correspondente ao intervalo $[t_{k+1}, t_k+L]$. Esta conexão não existirá caso não ocorram cargas em $[t_{k+1}, t_k+L]$.

2.5 Função Utilidade de Designação

No passo 2 do algoritmo MORSS é realizado o cálculo da utilidade de designação para todos os pares (carga candidata)/(navios). Calcular a utilidade de designação significa utilizar uma função matemática que avalie o benefício de designar um navio i para determinada carga candidata j. Conforme explicado na seção anterior, esta avaliação será realizada a cada iteração do algoritmo para todas as cargas com TEMin pertencentes ao horizonte de tempo estabelecido. Assim, a inserção de cada carga candidata j na rota de cada navio i é simulada para que o cálculo do benefício desta inserção possa ser realizado. Este cálculo é feito utilizando-se as seguintes funções:

$$U_{ij}=U_{ij}(1)+U_{ij}(2)+U_{ij}(3)+U_{ij}(4). \tag{2.1}$$

Na Equação 2.1 estão as quatro Funções Utilidades de Designação estabelecidas por Psaraftis (1985) cujos conceitos serão apresentados em seguida. A Função Utilidade de Designação U_{ij} avalia o benefício da designação ou atribuição da carga candidata j ao navio i. O valor desta função é o resultado da soma de quatro outras funções encarregadas de medir diferentes benefícios relacionados à atribuição da carga ao navio. Na Equação 2.2, a função $U_{ij}(1)$ avalia o efeito sobre o tempo de entrega da carga candidata j quando designada ao navio i, ou seja, caso inserida na respectiva rota do navio.

$$U_{ij}(1)=V_{min}+(V_{max}-V_{min})\, e^{-2(t/t_0)^b} \tag{2.2}$$

$$t=\begin{cases} t_c-\text{TDMax}, & \text{se } t>0 \\ 0, & \text{caso contrário} \end{cases} \tag{2.3}$$

Os parâmetros V_{min}, V_{max}, t_0 e b são fornecidos pelo usuário e responsáveis por fornecer uma curva de valores adequada para a função U_{ij} (1). A variável t mede o atraso no desembarque da carga, sendo seus valores fornecidos pela Equação 2.3. Nesta equação TDMax é o Tempo de Desembarque Máximo da carga candidata j e t_c o tempo de chegada (data da chegada) do navio i ao Porto de Desembarque (PD) da carga j.

Na Figura 2.6 estão as curvas de valores da função $U_{ij}(1)$, para diferentes valores do parâmetro b quando V_{min}=0, V_{max}=1 e t varia entre 0 e 10. Observa-se que alterações no valor do parâmetro b modificam a curva de valores fornecidas pela função U_{ij}.

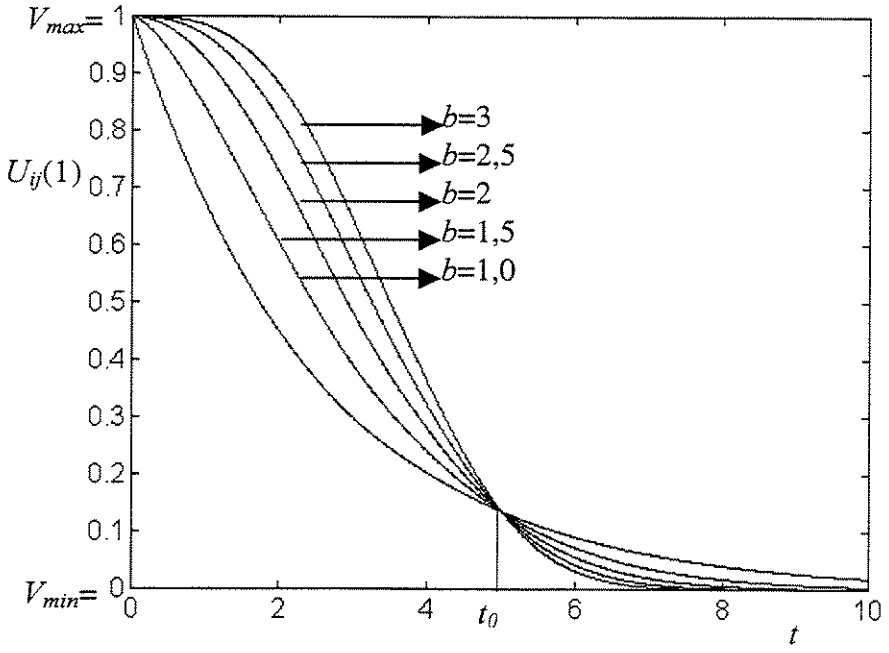


Figura 2.6 - Curvas da função $U_{ij}(1)$ variando-se o parâmetro b

Por exemplo, para b=3 os valores obtidos $U_{ij}(1)$ concentram-se entre 0,9 e 1,0 quando os atrasos (valores de t) variam de 0 até 2. Porém, quando b=1 a função U_{ij} passa a ser mais sensível aos atrasos, fazendo com que o benefício de atribuir o navio i a carga j quando t=2 valha menos que 0,5. Desta forma, aumento nos atrasos poderá ocasionar quedas mais acentuadas ou não poderá ocasionar quedas mais acentuadas nos valores de $U_{ij}(1)$ obtidos dependendo do parâmetro b utilizado.

Também observa-se na Figura 2.6 que os parâmetros V_{max} e V_{min} indicam, respectivamente, o valor máximo e mínimo que podem ser alcançados pela função utilidade de designação. Em todas as curvas obtidas, quanto menor o atraso no desembarque ($t \rightarrow 0$) mais o valor de $U_{ij}(1)$ aproxima-se de V_{max} e, caso ocorra o contrário, o benefício obtido na designação ficará cada vez mais próximo do valor mínimo V_{min}.

Observa-se na Figura 2.6 que a partir do instante TDMax+t_0 os atrasos deixam de ter uma influência muito grande nos valores obtidos para $U_{ij}(1)$, pois os valores de $U_{ij}(1)$ já estão bastante próximos de V_{min}. Logo, o parâmetro t_0 indica o instante de tempo a partir do qual o valor do atraso é tal que a utilidade de atribuir a carga j ao navio i passa a ser mínima.

Por exemplo, na Figura 2.6 temos t_0=5 semanas (meses, dias, horas, etc). Neste caso, se a carga for entregue 2 ou 4 semanas após seu TDMax sua utilidade $U_{ij}(1)$ sofre uma

considerável redução em todas as curvas apresentadas. O aumento no valor do atraso t mostra-se bastante influente nos valores obtidos para $U_{ij}(1)$. Porém, quando este atraso passa de 6 para 8 semanas o valor obtido para $U_{ij}(1)$ em t=8 não sofre uma considerável redução se comparado ao valor obtido quando t=6.

Como o valor t_0 depende de cada carga, Psaraftis *et al.* (1985) sugeriram que t_0 fosse obtido como sendo a quantidade de tempo na qual a influência de V_{min} e V_{max} pesasse sobre $U_{ij}(1)$ da seguinte forma:

$$U_{ij}(1) = V_{\min} + 0.1(V_{\max} - V_{\min}) \tag{2.4}$$

A Figura 2.7 demonstra a evolução da penalidade imposta por $U_{ij}(1)$, proporcionalmente ao aumento dos atrasos em duas curvas distintas. As curvas diferem quantos ao valores de V_{min} e V_{max} pois os valores dos parâmetros b=2 e t_0=5 são os mesmos. Na curva C_1, V_{min}=3 e V_{max}=1 enquanto V_{min}=1 e V_{max}=0 em C_2.

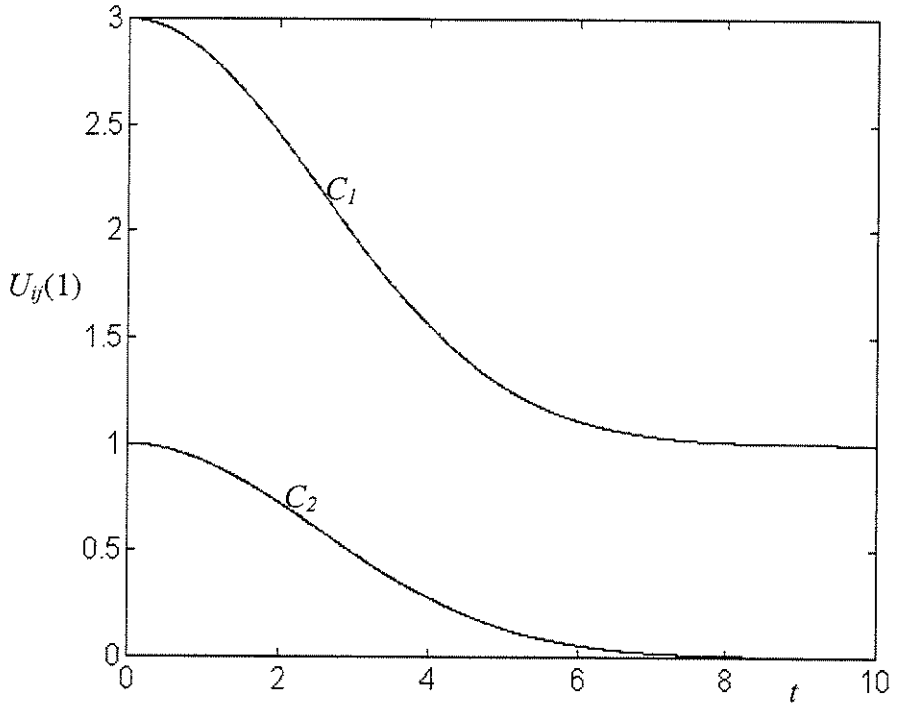


Figura 2.7 - Função Utilidade de Designação em função dos Atrasos

Na curva C_1 a diferença entre os valores de V_{min} e V_{max} é maior que na curva C_2, fazendo com que os U_{ij} obtidos diminuam num intervalo maior de valores. Por outro lado, na curva C_2 a diminuição nos valores de U_{ij} ocorre de forma mais acentuada. Apesar disso, observa-se que para valores de t entre 0 e 2 semanas de atrasos os valores de U_{ij} permanecem elevados com valores entre 3 e 2,5 em C_1 e 1 e 0,7 em C_2. A partir de 4 semanas de atraso, a queda no valor da utilidade das duas curvas mostra-se bastante alta e depois de 8 semanas praticamente não há utilidade em designar o navio à carga.

A função U_{ij} (2) mede o impacto que a designação da carga j ao navio i causa sobre o tempo de entrega das demais cargas $k \in R$, onde R é a rota atualmente executada pelo navio. Assim, avalia-se o efeito da inserção da carga candidata j sobre o embarque e desembarque das cargas já atribuídas ao navio. O cálculo é feito utilizando-se a função:

$$U_{ij}(2) = \sum_{k \in R} \Delta U_{ik}(1) \tag{2.5}$$

onde

$$\Delta U_{ik}(1) = U_{ik}(1) - U^{0}_{ik}(1). \tag{2.6}$$

A Equação 2.5 mede a variação no valor de $U_{ik}(1)$ de todas as cargas k pertencentes a rota R em execução pelo navio i, caso a carga candidata j seja inserida nesta rota. A Equação 2.6 apresenta como a variação de $U_{ik}(1)$ é calculada, onde $U^{0}_{ik}(1)$ é a utilidade de designação da carga k na rota atual R e $U_{ik}(1)$ é a utilidade de designação da carga após inserção da carga candidata j. Esta variação avalia três aspectos da inserção de j:

$\Delta U_{ik}(1)<0$: A inserção de j na rota provoca atraso na entrega da carga k.
$\Delta U_{ik}(1)=0$: A inserção de j não altera o tempo de entrega de k.
$\Delta U_{ik}(1)>0$: A inserção de j provoca uma antecipação na entrega da carga k.

A função $U_{ij}(3)$ mede o efeito da designação da carga j ao navio i em termos de eficiência no uso do navio i. Essa eficiência tem duas dimensões. Primeiro, pode-se desejar que o navio i viaje tão cheio quanto possível. Segundo, pode-se desejar que exista alguma folga de tempo na programação da rota do navio i para que outras cargas possam ser transportadas em iterações futuras. Assim, uma razoável expressão, capaz de medir a eficiência no aproveitamento do navio em termos de capacidade e tempo disponível para transporte de outras cargas, será:

$$U_N = V_N e^{-2(R/K)^C (1 - fF/L)^d}. \tag{2.7}$$

Na Equação 2.7, os termos V_N, c, d e f são parâmetros fornecidos pelo usuário com c, $d>0$ e $0<f<1$. O parâmetro V_N representa o valor máximo obtido no aproveitamento do navio, K é a capacidade máxima do navio e R sua capacidade residual após a carga j ter sido embarcada. O termo F é a folga média de tempo existente na programação das rotas do navio, ou seja, o tempo médio disponível na rota atual para que futuras inserções de cargas possam ser realizadas sem atraso nos embarques e desembarques já programados. Finalmente, L é o tamanho do horizonte de tempo.

A motivação para essa função é que a utilidade alcance seu valor máximo (V_N), em termos de capacidade de transporte, quando R=0. Isto ocorre porque o navio está viajando totalmente carregado, independentemente de existir folga de tempo na programação estabelecida para o navio. Porém, este valor torna-se pequeno (para F constante) a medida que R aumenta. Por outro lado, considerando-se R constante U_N torna-se uma função de F

atingindo seu valor máximo quando $F=L/f$, ou seja, o valor máximo é atingido quando a folga média é igual ao tamanho L do horizonte de tempo.

O navio i terá sua capacidade de transporte e sua folga média de tempo na programação atual alteradas quando sua rota passar a realizar o embarque e desembarque da carga candidata j. Desta forma, a eficiência no uso do navio i deverá ser avaliada nestas duas situações, ou seja, calcula-se U_N após o embarque (U_{Ne}) e desembarque (U_{Nd}) de j. A Equação 2.8 representa o cálculo final da utilidade de designação $U_{ij}(3)$.

$$U_{ij}(3) = U_{Ne} + U_{Nd}. \tag{2.8}$$

Logo, a eficiência no uso do navio i, caso a carga candidata j seja inserida na sua rota atual, será obtida pela soma dos valores encontrados em U_{Ne} e U_{Nd}.

Conforme mencionado na seção 2.2, existe um limite na capacidade operacional dos portos que não poderá ser excedida. Porém, uma grande quantidade de cargas sendo embarcadas e desembarcadas num mesmo porto pode ocasionar congestionamentos, já que navios continuam chegando aos portos sem que suas respectivas operações (embarques e desembarques) possam ser realizadas. Desta forma, a Equação 2.9 apresenta a forma de avaliação da capacidade operacional de um porto p.

$$U_p = \begin{cases} W_p & \text{se } P \text{ for um porto na rota atual} \\ V_p\, e^{-2(mN/P)^l} & \text{caso contrário} \end{cases} \tag{2.9}$$

Se o navio i estiver programado para realizar embarques ou desembarques em p o valor de U_P será máximo e igual a W_p. Caso o porto p necessite ser inserido na rota atual do navio, o valor de U_P irá decrescer, partindo de V_p, quanto maior for o número N de navios programados para visitar p no horizonte de tempo atual. O parâmetro P é a capacidade operacional máxima do porto p, ou seja, o número de operações que poderão ser executadas simultaneamente neste porto. Os parâmetros m e l são constantes não negativas fornecidos pelo usuário. A Equação 2.10 estabelece a avaliação do impacto sobre a capacidade operacional dos portos de embarque (U_{PE}) e desembarque (U_{PD}) ao se inserir a carga candidata j na rota do navio i.

$$U_{ij}(4) = U_{PE} + U_{PD} \tag{2.10}$$

Assim, a função utilidade de designação $U_{ij}(4)$ mede o benefício de inserir a carga candidata em termos de congestionamento nos portos, ou seja, incentiva designações de navios às cargas que utilizem portos com maior folga em sua capacidade operacional ou que já estejam inclusos na rota em execução.

As funções $U_{ij}(3)$ e $U_{ij}(4)$ estão amplamente relacionadas com aspectos peculiares ao problema real para o qual MORSS foi desenvolvido. Por isso, maiores detalhes a respeito

de como estas funções foram estabelecidas e utilizadas podem ser encontrados em Psaraftis *et al.* (1985).

Para este trabalho, o mais relevante é a idéia estabelecida pelo algoritmo ao utilizar diferentes funções matemáticas para avaliar diversas características do problema. A adaptação do método, apresentada no capítulo 3, utiliza as funções $U_{ij}(1)$ e $U_{ij}(2)$ e, outra adaptação, apresentada no capítulo 4, estabelece diferentes funções utilidades de designação. Assim, nem todas estas funções precisam estar presentes na resolução de um PRDV, assim como outras poderiam fazer parte do cálculo de U_{ij}. As peculiaridades do problema real a ser resolvido determinarão quais funções serão incorporadas pelo algoritmo.

2.6 - O Problema de Designação

O Algoritmo MORSS estabelece a cada iteração um conjunto C de cargas candidatas cujos TEMin's estejam dentro do horizonte de tempo $[t_k, t_k+L]$. A inserção de todas as cargas pertencentes a C nos navios é simulada. Esta simulação permite avaliar a utilidade U_{ij} de se inserir uma carga candidata j em cada um dos navios i. Após este procedimento, a decisão de qual carga será designada a determinado navio é tomada através da resolução do seguinte problema de designação:

$$\text{Max} \sum_i \sum_j U_{ij} x_{ij} \tag{2.11}$$

s.a.

$$\sum_i x_{ij} \leq 1, \qquad \text{para todo } j \tag{2.12}$$

$$\sum_j x_{ij} \leq K, \qquad \text{para todo } i \tag{2.13}$$

$$x_{ij} = \begin{cases} 1 & \text{Se o navio } i \text{ é designado para a carga } j \\ 0 & \text{Caso contrário} \end{cases} \tag{2.14}$$

Lembrando o conceito de Utilidade de Designação, percebemos que a função objetivo (2.11) valoriza as designações com maior medida de U_{ij}. Logo, será interessante designar cargas para navios que não atrasem a entrega destas (maior valor de $U_{ij}(1)$), não provoquem atrasos nas cargas anteriormente designadas (maior valor de U_{ij} (2)), valorizem a utilização dos navios (maior valor de $U_{ij}(3)$) e aproveitem os recursos dos portos (maior valor de $U_{ij}(4)$). Desta forma, obteremos valores máximos para a função objetivo (maior valor de U_{ij}).

A capacidade K dos navios, considerada homogênea, deve ser respeitada e isto aparece na restrição 2.13. Também está sendo assumido que cada carga é designada a um único

navio ou que não está designada, conforme estabelece a restrição 2.12. A variável de decisão x_{ij} está representada em 2.14 e poderá assumir o valor 1, caso o navio i seja atribuído à carga j na solução encontrada e 0 caso contrário.

A resolução deste problema ocorre a cada iteração do algoritmo e fornece o conjunto de cargas que maximizam a utilidade de designação aos navios. Após resolver o problema, utilizando algum método específico, a solução encontrada sofrerá um refinamento durante o passo 4 do algoritmo MORSS. Existem duas situações básicas em que a carga não será designada ao navio:

(a) $x_{ij}=0$. Neste caso, não há necessidade de utilizar critérios de refinamento, pois o próprio valor da variável de decisão já descarta a possibilidade da carga j ser transportada pelo navio i.

(b) $x_{ij}=1$ e valor do TEMin da carga j pertencente ao intervalo $[t_k+aL, t_k+L]$. Conforme explicado na seção 2.3, cargas com este valor de TEMin são consideradas eventos futuros e, portanto, não são designadas aos navios e devem retornar ao conjunto C de cargas candidatas.

Quando temos $x_{ij}=1$ e o valor do TEMin da carga entre $[t_k, t_k+ aL]$, duas possibilidades podem acontecer:

(1) $x_{ij}=1$ ocorre no navio i para uma única carga candidata j, ou seja, apenas uma carga candidata j foi designada ao navio i. Neste caso, a designação é efetivada de forma permanente e a carga não fará mais parte do conjunto de cargas candidatas C.

(2) $x_{ij}=1$ ocorre para várias cargas $j=j_1, j_2, j_3, ..., j_p$. Neste caso, a solução encontrada para o problema de designação atribuiu ao navio i a tarefa de transportar mais de uma carga j. O algoritmo selecionará uma carga j_1 com menor valor de TEMin (evento mais próximo do momento atual) e tornará sua designação ao navio i permanente, inserindo tal carga e estabelecendo uma nova rota para o navio. A próxima carga j_2 com menor TEMin terá sua função utilidade de designação U_{ij2} recalculada, considerando a nova rota estabelecida que incluiu j_1. Caso o novo valor U'_{ij2} desta utilidade não diminua mais que um certo parâmetro ρ, fornecido pelo usuário, tal carga também será designada permanentemente ao navio i. Caso contrário, ela retornará ao conjunto C. Este procedimento é repetido para as demais cargas seguindo uma ordem crescente do TEMin.

O parâmetro ρ ajuda no controle das atribuições já que o usuário poderá facilitar o transporte de mais de uma carga numa mesma iteração fornecendo um valor alto para ρ. Este procedimento acaba permitindo a inserção de cargas que poderão provocar atrasos, causar congestionamento nos portos ou ocasionar situações que são penalizadas com U_{ij} apresentando um valor afastado de V_{max}. Por outro lado, se o usuário deseja evitar tais situações ele atribuirá um valor pequeno ao parâmetro ρ.

2.7 - Exemplo Numérico

Nesta seção será apresentado um exemplo numérico da execução do algoritmo MORSS. O objetivo é mostrar como os passos do algoritmo são executados. Considere um conjunto de cargas e navios que tornam-se disponíveis durante a execução do método. Os navios deverão embarcar e desembarcar as cargas de acordo com as rotas estabelecidas pelo algoritmo a cada iteração. Na Figura 2.8, está representado o cenário inicial do problema, onde estão disponíveis as informações sobre três cargas (1, 2 e 3) e dois navios (N_1 e N_2).

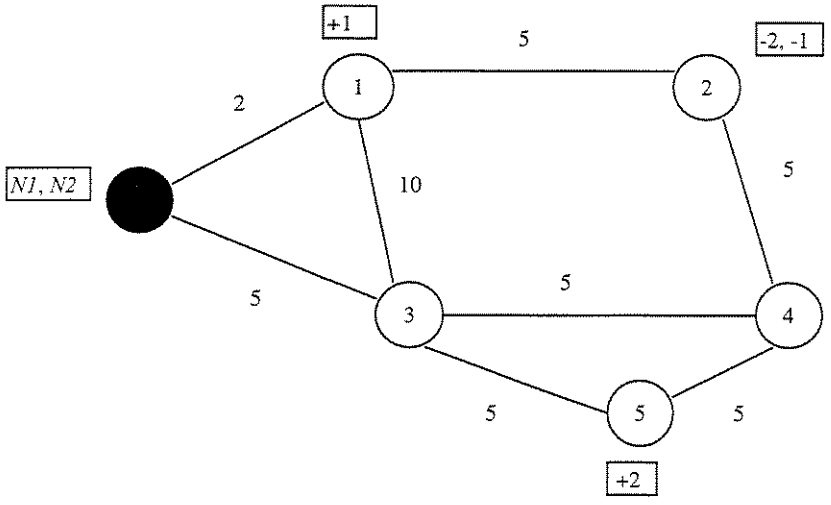


Figura 2.8 - Cenário inicial do problema

Os valores sobre os arcos representam o tempo de viagem em números de dias. Por exemplo, os navios irão levar 10 dias viajando entre os portos 1 e 3, enquanto para percorrer os portos 3 e 4 irão gastar 5 dias. Inicialmente, os navios estão localizados em um porto a 2 dias de viagem do porto1 e cinco dias do porto 3.

Tabela 2.1 - Tabela com dados das cargas

Carga	TEMin	TDMax
1	0	15
2	1	20
3	5	15

A Tabela 2.1 apresenta os dados sobre as cargas. A carga 1 já está disponível para embarque, pois seu TEMin vale 0, ou seja, trata-se do instante atual. Esta mesma carga precisa ser desembarcada no máximo até dia 15 que é o valor de seu TDMax. A carga 2 estará disponível para embarque no dia 1 (TEMin) e deverá ser desembarcada no máximo até o dia 20 (TDMax). Finalmente, a carga 3 apresenta TEMin=5 e TDMax=15. Observe

que não estamos considerando neste exemplo informações relacionadas ao TEMax, além de aspectos como congestionamento dos portos, utilização dos navios, entre outros mencionados no problema prático que motivou o método.

Estão sendo adotados os seguintes valores para os diversos parâmetros do algoritmo: L= 5, V_{min}=0, V_{max}=1, a=0.5, t_0 =5, b_0=2. Procurou-se adotar um intervalo de variação para U_{ij} não muito grande com valores entre V_{min}=0 e V_{max}=1. O método estabelecerá a cada iteração k intervalos de tempo $[t_k, t_k+L]$ de tamanho L=5 unidades de tempo. Desta forma, serão considerados dados relativos às cargas que tornam-se disponíveis neste período. Porém, o método tomará decisões definitivas apenas para cargas que tornam-se disponíveis na primeira metade do intervalo considerado, por isso, foi adotado o valor a=0.5. Os valores t_0 =5, b_0=2 permitem estabelecer uma curva conveniente para U_{ij}. O algoritmo estabelece o horizonte de tempo, considerando o instante inicial t_1=0. Os navios são homogêneos e apresentam capacidade K=2.

Iteração1 e Horizonte de Tempo [0, 5]:

O conjunto de cargas candidatas será $C=\{q/q\in Q_1$ e TEMin$\in [t_1, t_1+L]\}=\{1,2\}$, onde os números 1 e 2 representam as cargas. Em seguida calculam-se o atraso e a utilidade de designação, usando as fórmulas:

$$U_{ij}(1)=V_{min}+(V_{max}-V_{min})\, e^{-2\left(t/t_0\right)^{b}}$$

$$t=\begin{cases} t_c-\text{TDMax}, & \text{se } t>0 \\ 0, & \text{caso contrário} \end{cases}$$

t_c: tempo de chegada da carga j no respectivo PD.

Será assumido que a função utilidade de designação é composta exclusivamente por $U_{ij}(1)$, ou seja, $U_{ij}=U_{ij}(1)$. Desta forma, não está sendo avaliado, através do cálculo de $U_{ij}(2)$, o impacto sobre o tempo de desembarque das cargas que já estão na rota do navio i. Também não serão avaliadas a eficiência no uso do navio ($U_{ij}(3)$) e a capacidade operacional dos portos ($U_{ij}(4)$).

Tabela 2.2 - Cálculo de U_{ij} para o navio 1

Carga	t	U_{1j}
1	0	1
2	0	1
3	0	1

Tabela 2.3 - Cálculo de U_{ij} para o navio 2

Carga	t	U_{2j}
1	0	1
2	0	1
3	5	0,135

As Tabelas 2.2 e 2.3 apresentam os valores de t e os respectivos $U_{ij}=U_{ij}(1)$ para i=1, 2 e j=1, 2 e 3. Por exemplo, caso o navio execute uma rota para embarcar e desembarcar apenas a carga 3 ela será entregue no prazo (t=0) e, consequentemente, o valor da utilidade será $U_{ij}=U_{ij}(1)=V_{max}$=1. O embarque e desembarque desta mesma carga no navio 2 ocorre com um atraso de t=5, logo sua utilidade não será máxima com $U_{ij}=U_{ij}(1)$=0,135. Com os valores das utilidades calculados, é estabelecido o seguinte Problema de Designação:

Min $1x_{11} + 1x_{12} + 1x_{13} + 1x_{21} + 1x_{22} + 0{,}135\ x_{23}$
s. a
$x_{11} + x_{21} \leq 1$
$x_{12} + x_{22} \leq 1$
$x_{13} + x_{23} \leq 1$
$x_{11} + x_{12} + x_{13} \leq 2$
$x_{21} + x_{22} + x_{23} \leq 2$
$x_{ij} \in \{0,1\}$

A solução ótima obtida é: $x_{11}=x_{13}=x_{22}=1$ e $x_{12}=x_{21}=x_{23}=0$. A carga 3 não poderá ser designada efetivamente pois seu TEMin=5$\notin$ [0, 2]=[t_1, t_1+aL]. As cargas 1 e 2 são atribuídas de forma permanente aos navios 1 e 2, pois TEMin=0$\in$ [0, 2] na carga 1 e TEMin=1$\in$ [0, 2] na carga 2.

Os dados relativos às rotas obtidas estão apresentados na Tabela 2.4 e 2.5, onde a coluna Tempo indica as datas em que os navios chegam aos portos. Por exemplo, o navio 2 para embarcar a carga 2 passa pelo porto intermediário 3 no dia 5; embarca a carga dia 10 no porto 5; passa pelo porto intermediário 4 no dia 15 e desembarca a carga 2 sem atrasos dia 20 no porto 2. As operações apresentadas nestas tabelas significam:

- E: Embarque da carga
- P: Passagem por determinado porto intermediário na rota.
- D: Desembarque da carga

Tabela 2.4 - Rota em execução pelo navio 1

Portos	Tempo	Carga	Operação	Atraso
1	2	1	E	0
2	7	1	D	0

Tabela 2.5 - Rota em execução pelo navio 2

Portos	Tempo	Carga	Operação	Atraso
3	5	0	P	0
5	10	2	E	0
4	15	0	P	0
2	20	2	D	0

A Figura 2.9 apresenta o novo cenário do problema. Este cenário contém as rotas que estão sendo executadas pelos navios, as cargas que estão sendo transportadas e as novas cargas que tornaram-se disponíveis.

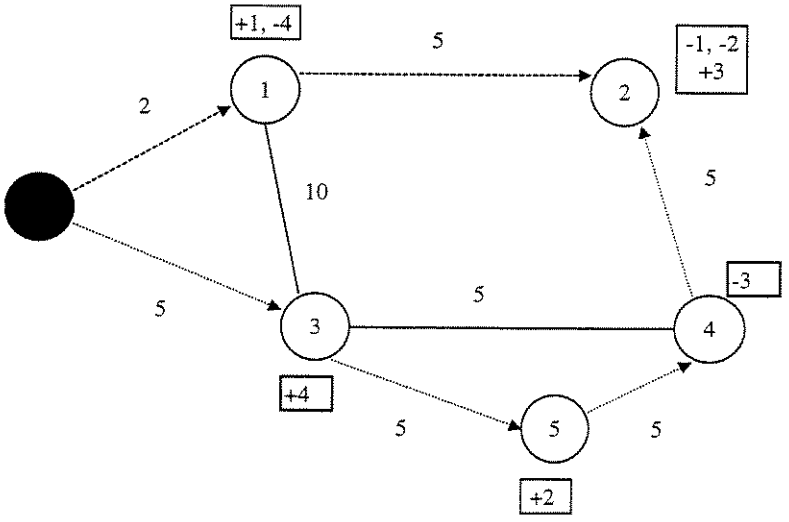


Figura 2.9 - Segundo cenário do problema

Na Figura 2.9, a rota em execução pelo navio 1 é representada pela seta tracejada, enquanto a rota do navio 2 é formada pela seta de pontos. Neste momento, ocorre uma atualização nos dados do problema e a carga 4 é incluída nos dados disponíveis, devendo ser embarcada no porto 3 e desembarcada no porto 1.

Tabela 2.6 - Tabela com dados das cargas

Carga	TEMin	TDMax
3	2	15
4	5	20

A Tabela 2.6 contém as informações relativas às cargas disponíveis que ainda não foram designadas aos navios. Desta forma, permanece como carga disponível para embarque a carga 3, não designada nesta iteração, e é incluída a carga 4 que passa a fazer parte do problema. O algoritmo ajusta $k=k+1=2$ e $t_2=2$ que é o menor valor entre os TEMin das cargas disponíveis.

Iteração2 e Horizonte de tempo [2, 7]:

O conjunto de cargas candidatas neste horizonte de tempo será $C=\{3, 4\}$. No cálculo das utilidade de designação está sendo assumindo que as cargas 3 e 4 foram inseridas no final da rota em execução pelos navios.

Tabela 2.7 - Cálculo de U_{ij} para o navio 1

Carga	t	U_{1j}
3	0	1
4	2	0,7261

Tabela 2.8 - Cálculo de U_{ij} para o navio 2

Carga	T	U_{2j}
3	10	0
4	25	0

Nas Tabelas 2.7 e 2.8, observa-se que o navio 1 é o único que consegue entregar uma carga no prazo. O navio 2, além de não entregar no prazo, obteve um atraso tão elevado que o valor da utilidade foi nulo nas duas cargas. Após o cálculo das utilidades, é estabelecido o Problema de Designação:

Min $1x_{13} + 0{,}7261\ x_{14} + 0x_{23} + 0x_{24}$
s.a
$x_{13} + x_{23} \leq 1$
$x_{14} + x_{24} \leq 1$
$x_{13} + x_{14} \leq 1$
$x_{23} + x_{24} \leq 1$

Observe que as restrições de capacidade apresentam $K=1$ pois ambos os navios apresentam menor capacidade. Isto ocorre porque já que foram designadas aos navios cargas que ainda não foram desembarcadas. Logo, os navios estão com sua capacidade máxima ($K=2$) para transporte de cargas reduzida.

A solução do Problema de Designação será $x_{13}=1$ e $x_{14}=x_{23}=x_{24}=0$. Apenas a carga 3 é atribuída de forma permanente ao navio 1, pois os únicos valores não nulos de utilidade ocorrem nas atribuições relativas a este navio. Para não violar a restrição de capacidade, apenas uma carga será embarcada. Neste caso, será a carga 3 que além de apresentar maior valor de U_{ij} também apresenta TEMin=5$\in$[2, 7]. As rotas em execução passam a ser:

Tabela 2.9 - Rota em execução pelo navio 1

Portos	Tempo	Carga	Operação	Atraso
1	2	1	E	0
2	7	1	D	0
		3	E	0
4	12	3	D	0

Tabela 2.10 - Rota em execução pelo navio 2

Portos	Tempo	Carga	Operação	Atraso
5	10	2	E	0
4	15	0	P	0
2	20	2	D	0

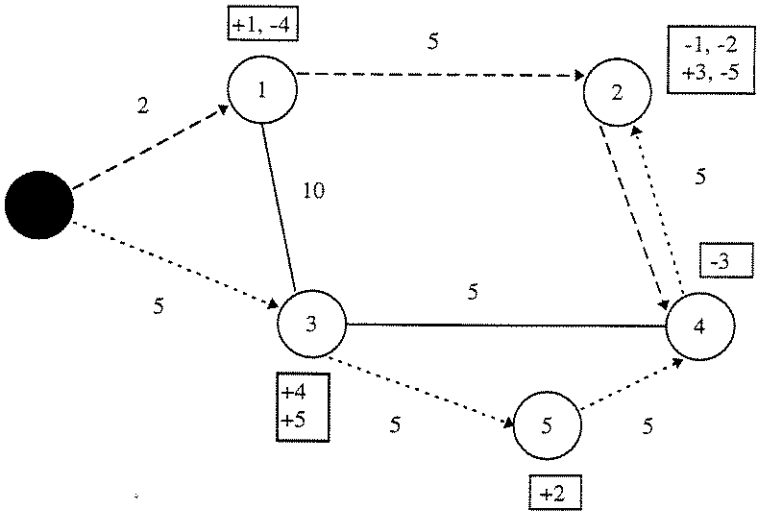


Figura 2.10 - Terceiro cenário do problema

A Figura 2.10 apresenta o percurso a ser executado pelos navios, após as designações realizadas nesta iteração. Também aparecem as cargas já atribuídas e a nova carga 5 que torna-se disponível ao problema.

Tabela 2.11 - Tabela com dados das cargas

Carga	TEMin	TDMax
4	5	20
5	5	20

A Tabela 2.11 apresenta as informações relativas a carga 4, que não foi designada a nenhum navio, e os dados da nova carga 5. O algoritmo ajusta t_3=5 porque é o menor valor entre os TEMin das cargas existentes.

Iteração 3 e Horizonte de tempo [5, 10]:

Cálculo das utilidade de designação:

Tabela 2.12 - Cálculo de U_{ij} para o navio 1

Carga	t	U_{1i}
4	7	0,0198
5	7	0,0198

Tabela 2.13 - Cálculo de U_{ij} para o navio 2

Carga	t	U_{2i}
4	25	0.0
5	0	1.0

Problema de Designação:

Min $0{,}0198\ x_{14} + 0{,}0198\ x_{15} + 0x_{24} + 1x_{25}$
s.a
$x_{14} + x_{24} \leq 1$
$x_{15} + x_{25} \leq 1$
$x_{14} + x_{15} \leq 1$
$x_{24} + x_{25} \leq 1$

O navio 1 ao final da segunda iteração estava no limite de sua capacidade, transportando as cargas 1 e 3. Nesta iteração, ele desembarca a carga 1 no porto 2 e volta a ter capacidade igual a 1. A solução do Problema de Designação será $x_{14}=x_{25}=1$ e $x_{15}=x_{24}=0$. A carga 4 é atribuída ao navio 1 (TEMin=5 na carga 4) e a carga 5 ao navio 2 (TEMin=5 na carga 5). As rotas finais obtidas, supondo que não ocorram mais alterações é:

Tabela 2.14 - Rota em execução pelo navio 1

Porto	Tempo	Carga	Operação	Atraso
2	7	1	D	0
		3	E	0
4	12	3	D	0
3	17	4	E	0
1	27	4	D	7

Tabela 2.15 - Rota em execução pelo navio 2

Portos	Tempo	Carga	Operação	Atraso
3	5	5	E	0
5	10	2	E	0
4	15	0	P	0
2	20	2	D	0
		5	D	0

Percurso executado pelos navios:

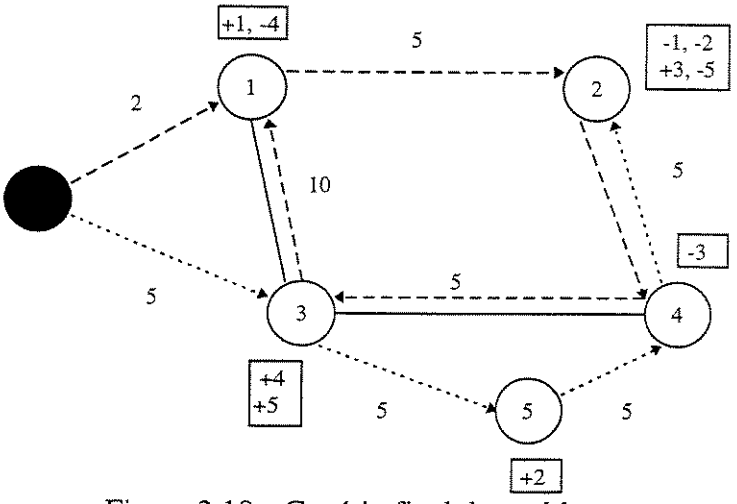


Figura 2.10 - Cenário final do problema

CAPÍTULO 3

RESOLUÇÃO DE UM PROBLEMA DE ROTEAMENTO DINÂMICO DE VEÍCULOS

3.1 - Introdução

Neste capítulo, será abordado o Problema de Roteamento Dinâmico de Veículo (PRDV), seguido das adaptações realizadas no algoritmo MORSS no intuito de resolvê-lo. Ao final, serão apresentados os resultados computacionais e avaliado o comportamento do algoritmo enquanto ferramenta útil na resolução de um conjunto de diferentes instâncias de PRDV´s.

MORSS trata-se de uma ferramenta computacional desenvolvida para solucionar um problema obtido a partir de uma situação prática, envolvendo o transporte em tempo real de diferentes quantidades de cargas por diversos navios. Conforme descrito na seção 2.1 do capítulo anterior, todo o sistema MORSS é composto por quatro subsistemas: READIN, DISPLAY, SEEDS e SCHEDULE.

As adaptações realizadas para resolver os problemas que serão descritos neste capítulo e no capítulo seguinte foram realizadas no subsistema SCHEDULE, onde encontra-se a principal seqüência de comandos de todo o sistema MORSS. Esta seqüência de comandos foi denominada algoritmo MORSS (Psaraftis *et al*, 1985 e 1988) e sua importância reside no gerenciamento das características dinâmicas do problema de logística naval descrito no capítulo 2. Os demais módulos do sistema MORSS, apesar de importantes na resolução do problema em questão, cumprem papel acessório aos comandos executados pelo subsistema SCHEDULE.

O PRDV apresentado neste capítulo tem por objetivo estabelecer um modelo teórico, ou seja, um modelo não restrito às condições peculiares envolvidas em uma situação prática. Desta forma, foi possível criar diferentes instâncias para o PRDV's que será estabelecido na próxima seção.

O capítulo é composto de cinco seções. Na seção 3.2 serão definidas as características do PRDV a ser resolvido, assim como sua apresentação dentro de um contexto dinâmico. As adaptações realizadas no algoritmo MORSS são detalhadas na seção 3.3 e os resultados computacionais obtidos são apresentados e analisados na seção 3.4. As considerações finais sobre o desempenho e adaptabilidade do algoritmo encontram-se na seção 3.5.

3.2 - Estabelecendo um Problema de Roteamento Dinâmico de Veículos

Considere uma frota de veículos em determinada região. A frota não se encontra em um depósito central, desta forma, seus veículos podem estar inicialmente localizados em diferentes pontos. Cada veículo deverá embarcar e desembarcar um conjunto de cargas localizadas em diversos depósitos existentes nessa região. O problema consiste em estabelecer rotas para os veículos, dentro de um contexto dinâmico, minimizando o atraso no desembarque das cargas.

O contexto dinâmico é determinado por novas informações fornecidas ao problema em tempo real e que alteram o cenário inicialmente determinado. Estas novas informações podem referir-se a alterações nos dados disponíveis ou a novos dados que tornaram-se conhecidos. A constante atualização do problema gera a necessidade de mudanças no processo de construção das rotas e exige alterações durante a execução das mesmas pelos veículos. As informações que serão alteradas podem referir-se a veículos, cargas ou depósitos.

Para ilustrar o contexto dinâmico, será fornecida uma representação do problema em termos de cenários. Considere que em dado instante inicial I_0 seja estabelecido o cenário C_0 composto pelo grafo $G_0=(D_0, T_0)$ e conjuntos V_0 e Q_0. O conjunto $V_0=\{v_1,..., v_{|V_0|}\}$ representa os veículos disponíveis e $Q_0=\{q_1,..., q_{|Q_0|}\}$ as cargas existentes, onde $|V_0|$ é a cardinalidade de V_0 e $|Q_0|$ é a cardinalidade de Q_0.

$G_0=(D_0, T_0)$ é um grafo orientado onde $D_0=\{d_1,..., d_{|D_0|}\}$ é o conjunto de depósitos existentes representados pelos nós deste grafo com $|D_0|$ sendo a cardinalidade de D_0. O conjunto $T_0=\{(i,j)/\ \forall i,j \in D_0 \text{ e } i \neq j\ \}$ reúne os arcos (i,j) que representam a ligação entre os depósitos. A cada arco $(i,j) \in T_0$ associamos um peso t_{ij} com $i, j \in D_0$ que indica o tempo de viagem do depósito i até o depósito j. Será considerado que $t_{ij}=t_{ji}$ para todos os depósitos i e j pertencentes aos diversos cenários estabelecidos.

O tempo de viagem inicial dos veículos aos depósitos também está representado no cenário C_0, através dos arcos (v_i, j) com $v_i \in V_0$ e $j \in D_0$ onde t_{vij} representa o tempo de viagem do veículo v_i ao depósito j. De forma geral, todo novo veículo adicionado ao problema aparecerá inicialmente ligado a diversos depósitos através de arcos do tipo (v_i, j).

A Figura 3.1 apresenta um cenário inicial C_0 correspondente ao instante I_0, onde $V_0=\{v_1\}$, $Q_0=\{q_1, q_2\}$ e $D_0=\{d_1, d_2, d_3\}$. O símbolo "$+q$" indica o local onde a carga q deverá ser embarcada e "$-q$" o local de desembarque. Neste cenário, como v_1 não está executando uma rota é apresentado seu tempo de viagem t_{v1d1} e t_{v1d3} relativo aos dois depósitos d_1 e d_3.

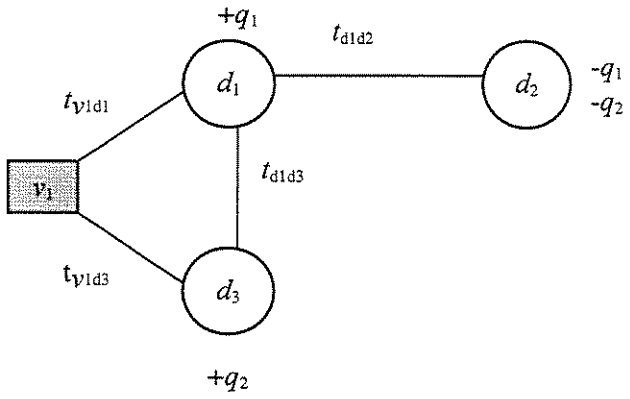


Figura 3.1 - Cenário C_0 correspondente ao instante I_0

Analogamente, define-se um cenário C_1, em um instante I_1 seguinte ao instante I_0, como aquele representado pelo grafo $G_1=(D_1, T_1)$ e pelos conjuntos $V_1=\{v_1, v_{|V1|}\}$ e $Q_1=\{q_1,..., q_{|Q1|}\}$. O cenário C_1 deverá ser diferente de C_0 pois contém as modificações conseqüentes das atualizações nos dados. Logo, o instante de tempo I_1 será aquele no qual ocorreram novas informações e atualizações nos dados existentes, capazes de modificar a situação anteriormente estabelecida. Desta forma, são estabelecidos diversos cenários C_k em diferentes e sucessivos instantes I_k para k=0,1,2...,m; onde I_m seria o instante exato da última alteração de dados no problema.

Dois tipos de alterações serão consideradas no problema proposto, envolvendo os instantes consecutivos I_{k-1} e I_k:

- Alterações paramétricas: Neste caso $|Q_k|=|Q_{k-1}|$ e $|V_k|=|V_{k-1}|$, ou seja, as informações sobre a quantidade de cargas e veículos não foram alteradas e o grafo atual mantém a mesma estrutura do grafo anterior ($G_k=G_{k-1}$). As mudanças no instante I_k ocorrem na qualidade da informação e, neste problema, serão assim consideradas as alterações nos tempos de viagens entre os depósitos e na capacidade de transporte dos veículos.

- Alterações estruturais: As alterações ocorrem na quantidade de dados disponíveis. Neste caso, o número de veículos e cargas existentes no momento anterior podem diminuir, $|V_k|<|V_{k-1}|$ e $|Q_k|<|Q_{k-1}|$, ou aumentar com $|Q_k|>|Q_{k-1}|$ e $|V_k|>|V_{k-1}|$. Quando $|Q_k|<|Q_{k-1}|$ há menos cargas porque muitos embarques foram realizados em I_{k-1}, diminuindo a demanda em I_k. Para $|V_k|<|V_{k-1}|$ alguns veículos disponíveis em I_{k-1} estão indisponíveis em I_k. Se $|Q_k|>|Q_{k-1}|$ ou $|V_k|>|V_{k-1}|$ novas cargas ou veículos passam a fazer parte do problema. No caso de novas cargas, elas podem estar armazenadas em novos depósitos ou em depósitos já existentes. Neste último caso, teremos um novo grafo $G_k\neq G_{k-1}$ com um número de depósitos ($|D_k|$) no instante I_k maior que o número de depósitos($|D_{k-1}|$) existentes no instante anterior I_{k-1}. Assim, quando $|D_k|>|D_{k-1}|$, existirão novos arcos pertencentes a T_k com pelo menos um arco (i, j) apresentando $i\in D_k\backslash D_{k-1}$ ou $j\in D_k\backslash D_{k-1}$.

Na Figura 3.2, o cenário C_1 no instante I_1 apresenta alterações quantitativas em C_0, envolvendo aumento em $V_1=\{v_1, v_2, v_3\}$ com $|V_1|=3$, $Q_1=\{q_1, q_2, q_3, q_4\}$ com $|Q_1|=4$ e $D_1=\{d_1, d_2, d_3, d_4, d_5\}$ com $|D_1|=5$. Desta forma, surgem novos arcos ligando depósitos existentes (arco (d_2,d_3)) e arcos ligando os novos depósitos aos anteriores (arcos (d_3,d_4), (d_2,d_4), (d_1,d_5) e (d_2,d_5)). Também surgem arcos apresentando o tempo de viagem atual dos novos veículos aos depósitos (arcos (v_2,d_2), (v_2,d_4), (v_3,d_5) e (v_3,d_2)). Uma alteração qualitativa ocorre com o surgimento do arco (d_2, d_3) conectando dois depósitos já existentes. Este arco estabelece um novo caminho com um novo tempo de viagem se $t_{d2d3} \neq t_{d2d3} + t_{d2d3}$.

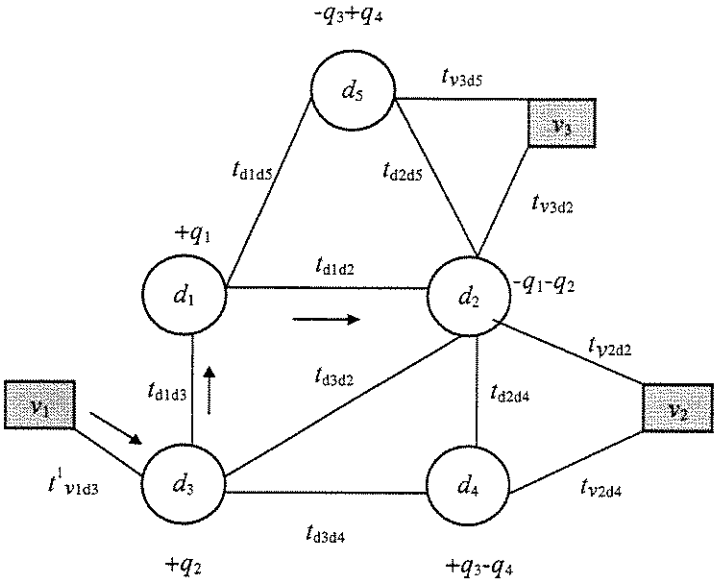


Figura 3.2 - Cenário C_1 correspondente ao instante I_1

O veículo v_1 executa a rota indicada pela seta na Figura 3.2 e seu tempo de viagem ao primeiro depósito desta rota vale t^1_{v1d3} no arco (v_1, d_3). Os demais arcos que representavam o tempo de viagem de v_1 aos outros depósitos em C_0 deixam de existir porque foi estabelecida uma rota para este veículo. Como está sendo considerado o instante I_1 seguinte ao instante I_0 ocorrerá $t^1_{v1d3} < t_{v1d3}$. Isto ocorre porque o veículo na Figura 3.2 já está viajando em direção ao depósito d_3, portanto, o mesmo encontra-se a um tempo de viajem t^1_{v1d3} menor que o inicialmente estabelecido t_{v1d3}.

Isto posto, pode-se visualizar o PRDV proposto como uma sucessão de diversos subproblemas considerados em diferentes e sucessivos instantes. Estes subproblemas devem ser resolvidos de modo a otimizar as rotas em cada cenário C_k relativo ao instante I_k, porém, considerando-se as decisões tomadas nos cenários C_{k-1}, C_{k-2},..., C_1 e C_0 correspondentes aos instantes I_{k-1}, I_{k-2},..., I_1 e I_0. Todos os veículos e cargas considerados nos PRDV's que estão sendo estabelecidos são homogêneos. Portanto, não há diferenças de tipos ou modelos entre eles ou restrições que estabeleçam preferência no transporte de certa carga por determinado veículo.

As cargas são consideradas indivisíveis, logo serão embarcadas em sua totalidade pelo mesmo veículo. Outras informações relativas às cargas:

- **Tempo de Embarque Mínimo (TEMin)**: Representa o instante de tempo a partir do qual a carga estará disponível no depósito para ser embarcada.
- **Tempo de Desembarque Máximo (TDMax)**: Representa o instante de tempo máximo no qual a carga deverá ser desembarcada sem que ocorra atraso.
- **Depósito de embarque (DE)**: Depósito onde a carga deverá ser embarcada.
- **Depósito de desembarque (DD)**: Depósito onde a carga deverá ser desembarcada.

Ao tornar-se disponível em determinado instante I_k, uma carga terá no cenário C_k a localização de seus depósitos de embarque e desembarque. Esses depósitos poderão ser novos (ocorrendo novos nós no grafo G_k) ou depósitos já existentes (nós que já existiam em G_{k-1} no instante I_{k-1}). A restrição relativa ao instante em que cada carga torna-se disponível para embarque, fornecida por seu TEMin, não poderá ser violada. Assim, mesmo que os dados relativos a determinada carga que deverá ser embarcada no futuro já estejam disponíveis, esta não poderá ser efetivamente embarcada antes do seu TEMin. Por outro lado, uma carga poderá ficar aguardando no DE além do seu TEMin até que consiga ser transportada por algum veículo.

As restrições relativas ao TDMax poderão ser violadas. Caso não pudessem ser violadas em situações com muita demanda de cargas e pouca disponibilidade de veículos, várias cargas deixariam de ser embarcadas porque não seria possível realizar sua entrega atendendo o prazo estabelecido nos respectivos TDMax. Portanto, será melhor realizar uma entrega com atraso do que simplesmente não realizá-la e, para inibir atrasos freqüentes, será estabelecida uma penalização no cálculo das utilidades de designação das cargas aos veículos (seção 3.3).

O número de veículos poderá variar, sendo a frota aumentada ou reduzida ao passarmos de um instante para outro. Não será considerada a redução da frota por quebra ou problemas nos veículos, mas a redução relacionada com uma não disponibilidade por estarem operando na sua capacidade máxima de transporte. As informações disponíveis sobre os veículos são sua capacidade K e o tempo de viagem a que se encontram de pelo menos um dos depósitos existentes. As alterações nas rotas dos veículos somente ocorrerão quando os mesmos estiverem passando por determinado depósito, isto é, enquanto o veículo viajar entre dois depósitos sua rota não poderá ser alterada.

Na Figura 3.3, um veículo executa uma rota que passa pelos depósitos d_1, d_2, d_3 e d_4; e precisa executar dois desvios para os depósitos d_5 e d_6, por exemplo, para embarcar carga em d_5 e desembarcar em d_6. Os desvios apresentados na Figura 3.3 (a) não são permitidos porque enquanto viaja entre os depósitos d_1 e d_2 o veículo altera sua rota indo até d_5 , depois entre d_2 e d_3 faz o mesmo para chegar até d_6. Os desvios apresentados na Figura 3.3 (b) são permitidos porque ocorrem quando o veículo está em um depósito e não enquanto o veículo viaja entre eles. Assim, ao chegar no depósito d_1 o veículo desvia sua rota até d_5 e ao atingir o depósito d_2 desvia sua rota até d_6.

(a) Desvio não permitido (b) Desvio permitido

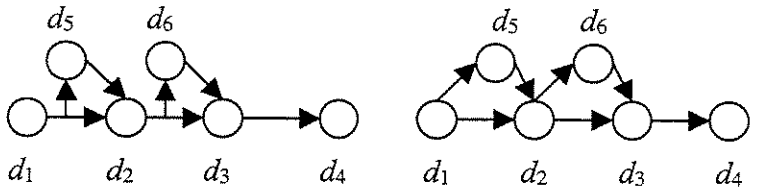


Figura 3.3 – Desvios nas rotas dos veículos

Os depósitos não são separados pelo tipo de operação que realizam. Assim, todos os depósitos estão aptos a realizar embarque e desembarque de cargas. Também não existe um limite para o número de embarques e desembarques a serem realizados nos depósitos, logo, não há limites em sua capacidade operacional. O grafo representativo dos depósitos não será necessariamente um grafo completo, portanto, não haverá obrigatoriamente arcos ligando todos os depósitos. Desta forma, o caminho a ser percorrido entre um DE e o respectivo DD pode envolver a passagem por depósitos intermediários.

O problema a ser resolvido reside na construção de rotas para cada veículo, dentro do contexto dinâmico mencionado, de modo que ao transportar as cargas dos DE's até os DD's os atrasos nos desembarques sejam minimizados. Conforme explicado inicialmente, o contexto dinâmico do problema estabelecido nesta seção aparece da necessidade de determinar as rotas sem conhecimento prévio de todas as informações relativas às cargas, depósitos e veículos. Partindo-se de um instante I_0 e um cenário C_0 inicial, novas informações surgem nos instantes seguintes I_k provocando alterações qualitativas e quantitativas nos dados que estabelecem novos cenários C_k. Essas alterações ocorrem simultaneamente à execução das rotas estabelecidas nos cenários anteriores.

3.3 - Adaptação do Algoritmo MORSS ao Problema Proposto

O algoritmo MORSS é baseado na idéia do *Horizonte Rolante* e nas *Funções Utilidades de Designação*, conforme explicado nas seções 2.3 e 2.4 do capítulo anterior. A cada horizonte de tempo considerado, o impacto que as alterações dos dados provocam nas rotas em execução é avaliado. Isso é feito através de funções matemáticas responsáveis por atribuir um valor ao benefício envolvido na designação de determinada carga a determinado veículo.

Após avaliar os benefícios das possíveis designações de cargas aos veículos, o método estabelece e resolve um *Problema de Designação*. Neste problema o objetivo é estabelecer designações que obedeçam as restrições existentes e maximizem o valor dos benefícios calculados. Finalmente, apenas cargas que forneçam a melhor solução para o problema de

designação e que atendam os critérios de refinamento a serem apresentados (seção 3.3.3) serão efetivamente transportadas, estabelecendo-se as melhores rotas para os veículos.

3.3.1 – Horizonte Rolante

A idéia do *Horizonte Rolante* consiste na avaliação do problema através de intervalos de tempo. A cada iteração k, o algoritmo estabelece um horizonte de tempo $[t_k, t_k+L]$ de tamanho L e resolve um *Problema de Designação* envolvendo as informações disponíveis e relativas a este intervalo. Na iteração seguinte, ocorre uma rolagem do tempo e um novo horizonte é estabelecido. Neste novo horizonte de tempo, as novas informações são avaliadas, porém, levando-se em conta as decisões tomadas nos intervalos anteriores. Maiores detalhes foram apresentados na seção 2.3 do capítulo anterior.

Adaptando o *Horizonte Rolante* ao problema proposto na seção anterior, cada horizonte de tempo estabelecido poderá englobar diversos instantes em que ocorreram atualizações nos dados do problema. Desta forma, o cenário C'_k, referente ao intervalo $[t_k, t_k+L]$ na $k^{\text{ésima}}$ iteração do algoritmo, será formado pela composição de diversos cenários do problema relativos a todos os instantes contidos no horizonte de tempo considerado.

Na Figura 3.4 é ilustrada a adaptação da idéia do Horizonte Rolante utilizada pelo algoritmo MORSS para resolver o PRDV proposto. Neste caso o problema fornece, por exemplo, três cenários (C_1, C_2 e C_3) e os respectivos instantes (I_1, I_2 e I_3) em que ocorreram atualizações nos dados do problema.

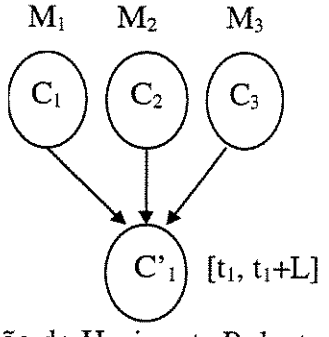


Figura 3.4 - Adaptação do Horizonte Rolante ao problema proposto

O algoritmo utiliza um critério, a ser explicado em seguida, para determinar o tempo mínimo t_1 e , através do tamanho L do horizonte de tempo, fixar seu tempo máximo t_1+L. Quando o método fixa os valores de t_1 e t_1+L o intervalo $[t_1, t_1+ L]$ considerado engloba os três instantes em que ocorreram as atualizações nos dados do problema. Desta forma, um cenário C'_1 relativo a este intervalo é estabelecido sendo formado pela junção dos cenários C_1, C_2 e C_3 correspondentes aos instantes I_1, I_2 e I_3.

Para cada horizonte de tempo estabelecido, o algoritmo MORSS irá formar o conjunto C de cargas candidatas, obedecendo ao seguinte critério:

$$C=\{q/q\in Q_k \text{ com TEMin}\in [t_k, t_k+L] \text{ e TDMax}\geq t_k\}. \qquad (3.1)$$

A Equação 3.1 define o conjunto de cargas candidatas C. Este conjunto contém todas as cargas q disponíveis para embarque cujo TEMin esteja contido no horizonte de tempo estabelecido, ou seja, TEMin$\in[t_k, t_k+L]$ e o TDMax não tenha sido ultrapassado (TDMax$\geq t_k$). Apenas cargas pertencentes a C terão sua atribuição aos veículos avaliadas pelas *Funções Utilidades de Designação*.

O conjunto Q_k contém todas as cargas cujas informações já foram disponibilizadas ao problema, mas que ainda não estão sendo transportadas pelos veículos. Também pertencem a Q_k todas as cargas que foram integradas ao problema pela atualização nos dados ocorrida na $k^{\text{ésima}}$ iteração. Nesta adaptação do algoritmo MORSS, a escolha de t_k será feita adotando-se o seguinte critério:

$$t_k=\min\{\text{TEMin} / \forall q\in Q_k \backslash Q_A\}. \tag{3.2}$$

Pela Equação 3.2, observa-se que o TEMin será escolhido entre as cargas pertencentes a Q_k excetuando-se aquelas contidas em Q_A. O conjunto Q_A contém as cargas cujo TEMin=t_j com j=0, 1, 2, ... ,k-1, ou seja, aquelas cargas que nas iterações anteriores não foram atribuídas aos veículos e apresentavam TEMin=t_{k-1}.

Se as cargas pertencentes a Q_A fossem novamente consideradas na escolha de t_k, ocorreria t_k=TEMin=t_{k-1} e o horizonte de tempo se repetiria, ou seja, não ocorreria um rolamento do horizonte. Nesta adaptação, uma carga deixará de ser entregue apenas quando, em sucessivos horizontes de tempo, ela continuar a permanecer em Q_k até que seu TDMax fique maior que o valor t_k selecionado.

3.3.2 – Função Utilidade de Designação

Depois de estabelecer o horizonte de tempo e o respectivo conjunto C, o algoritmo irá calcular o valor da *Função Utilidade de Designação* para todas as cargas candidatas $q\in C$. Este cálculo é feito simulando-se a inserção de cada carga candidata em cada uma das rotas dos veículos disponíveis. Para cada simulação realizada, o método avalia o benefício que a atribuição de cada carga candidata traz para as rotas dos veículos disponíveis. Este benefício é avaliado através do valor retornado pela *Função Utilidade de Designação*. Conforme exposto no capítulo 2, Psaraftis *et al.* (1985) definiram as seguintes Funções Utilidades de Designação

$$U_{ij}=U_{ij}(1)+U_{ij}(2)+U_{ij}(3)+U_{ij}(4). \tag{3.3}$$

No contexto do PRDV proposto na seção anterior, os termos que compõem a Equação 3.3 podem ser interpretados da seguinte forma:

- $U_{ij}(1)$: Avalia o benefício obtido no tempo de desembarque da carga candidata j quando inserida na rota do veículo i;
- $U_{ij}(2)$: Avalia o benefício obtido no tempo de desembarque das cargas que já estão sendo transportadas pelo veículo i, quando a carga candidata j é inserida na sua rota.

- $U_{ij}(3)$: Avalia o benefício de inserir a carga candidata j na rota do veículo i, procurando garantir um bom aproveitamento da capacidade do veículo e a manutenção de certa flexibilidade para futuras operações de embarque dentro da rota.
- $U_{ij}(4)$: Avalia o benefício de inserir a carga candidata j na rota do veículo i, através do impacto de inserir a carga quando existem filas para embarque ou desembarque nos respectivos depósitos.

Os aspectos avaliados pela função $U_{ij}(3)$ não serão considerados de forma direta no problema que está sendo estabelecido, portanto, tal função não será utilizada na avaliação das possíveis atribuições de cargas aos veículos. Também é suposto que os depósitos possam realizar simultaneamente embarques e desembarques, além de possuírem capacidade ilimitada para realizar tais operações. Logo, a utilidade de designação $U_{ij}(4)$ também não será utilizada na avaliação das atribuições.

O principal objetivo na resolução do problema proposto é entregar as cargas dentro dos respectivos prazos máximos para desembarque. Assim, procurou-se avaliar o benefício na designação das cargas candidatas ao veículos através do impacto sobre o TDMax, tanto da carga candidata a ser inserida na rota quanto daquelas que já estão sendo transportadas. Por isso, utilizou-se a seguinte Função Utilidade de Designação

$$U_{ij} = U_{ij}(1) + U_{ij}(2) \tag{3.4}$$

para calcular a designação de um produto candidato j ao veículo i nesta adaptação. A função $U_{ij}(1)$ avalia o efeito sobre o tempo de entrega da carga candidata j quando inserida na rota do veículo i. Será utilizada a mesma expressão definida no capítulo anterior:

$$U_{ij}(1)=V_{min}+(V_{max}-V_{min})e^{-2[(t/to)]b} \tag{3.5}$$

onde,

$$t = \begin{cases} t_c - \text{TDMax}, & \text{se } t > 0 \\ 0, & \text{caso contrario} \end{cases} \tag{3.6}$$

Nas Equações 3.5 e 3.6, a variável t mede o atraso no desembarque da carga candidata j e t_c o tempo de chegada do veículo i ao Depósito de Desembarque (DD). Os parâmetros V_{min}, V_{max}, t_0 e b são fornecidos pelo usuário e responsáveis por fornecer a curva desejada para a função $U_{ij}(1)$.

$$U_{ij}(2) = \sum_{k \in R} \Delta U_{ik}(1) \tag{3.7}$$

$$\Delta U_{ik}(1)=U^{0}_{ik}(1)-U_{ik}(1) \tag{3.8}$$

Na Equação 3.7, função $U_{ij}(2)$ mede o impacto que a designação da carga j ao veículo i causa sobre o tempo de entrega das demais cargas $k \in R$, onde R é a rota estabelecida na iteração anterior e atualmente em execução pelo veículo. Assim, a Equação 3.8 mostra

como é calculado o impacto da inserção da carga candidata j sobre o embarque e desembarque das cargas já atribuídas ao veículo i.

Este cálculo é realizado da mesma forma que foi definido no capítulo anterior, onde:

- $U^0_{ik}(1)$: Valor da utilidade de designação da carga k na rota em execução R.
- $U_{ik}(1)$: Utilidade de designação da carga k após inserção da carga elegível j na rota.

Neste contexto, a inserção do produto candidato j em R poderá ocasionar as seguintes situações:

- $\Delta U_{ik}(1)<0$: A inserção de j na rota provoca atraso na entrega da carga k.
- $\Delta U_{ik}(1)=0$: A inserção de j não altera o tempo de entrega de k.
- $\Delta U_{ik}(1)>0$: A inserção de j provoca uma antecipação na entrega da carga k.

3.3.3 – Problema de Designação

Depois de calcular as utilidades de designação U_{ij}, o algoritmo resolve o seguinte problema de designação:

$$\text{Max} \sum_{i=1}^{|V|} \sum_{j=1}^{|C|} U_{ij}\, x_{ij} \tag{3.9}$$

s.a.

$$\sum_{i=1}^{|V|} x_{ij} \leq 1, \qquad \forall j=1,2,...|C| \tag{3.10}$$

$$\sum_{j=1}^{|C|} x_{ij} \leq K, \qquad \forall i=1,2,...,|V| \tag{3.11}$$

$$x_{ij} = \begin{cases} 1 & \text{Se o veículo } i \text{ é designado à carga } j \\ 0 & \text{Caso contrário,} \end{cases} \tag{3.12}$$

Os dados do problema de designação apresentado são as cargas $j \in C$ e os veículos disponíveis no intervalo $[t_k, t_k+L]$, ou seja, contidos no conjunto V_k. A função objetivo 3.9 valoriza designações que estabeleçam maior medida na soma das funções utilidade de designação U_{ij}, logo, interessam à solução do problema de designação: designações cargas/veículos que não atrasem a entrega das cargas (maior valor de $U_{ij}(1)$) e não retardem as cargas anteriormente designadas (maior valor de $U_{ij}(2)$).

O PRDV proposto não admite divisões das cargas, ou seja, cada carga é transportada por um único veículo, e isto está assegurado pela restrição 3.10. A capacidade dos veículos é a mesma para toda a frota e não poderá ser ultrapassada, onde a restrição 3.11 garante estas condições.

Nesta adaptação, se apenas um dos veículos não estiver no limite de sua capacidade de transporte, a solução obtida neste caso sempre retornará a atribuição de uma carga a este veículo. Isto é garantido porque utiliza-se $V_{min} \neq 0$, desta forma, não ocorrerá somatório nulo na função objetivo quando $U_{ij}=V_{min}\ \forall i, j$.

O *Problema de Designação* acima é estabelecido a cada iteração do algoritmo MORSS e sua resolução será realizada pelo pacote CPLEX (CPLEX, 1994). CPLEX é um programa de otimização que fornece soluções inteiras e canalizadas, ou seja, retorna soluções em um espaço discreto ou contínuo.

O CPLEX recebe um arquivo com a formulação matemática apresentada acima, processa os dados e retorna uma solução formada pelos valores de x_{ij} que poderão ser iguais a 0 ou 1 (restrição 3.12). Essa solução passará por um refinamento no passo seguinte do algoritmo MORSS que determinará quais cargas efetivamente serão atribuídas aos veículos.

A solução fornecida pelo Problema de Designação indica se uma carga candidata j deve ser atribuída ao veículo i ($x_{ij}=1$) ou se voltará ao conjunto Q_k de cargas disponíveis ($x_{ij}=0$). Porém, o algoritmo não considera que os valores $x_{ij}=1$ sejam suficientes para imediatamente incorporar a carga j à rota do veículo i. Estes valores serão analisados adotando-se diversos critérios que, ao final, indicarão quais cargas serão efetivamente atribuídas aos veículos.

Os critérios adotados nesta adaptação do algoritmo são praticamente os mesmos explicados na seção 2.5 do capítulo anterior. O primeiro passo é retornar ao conjunto Q_k as cargas que apresentam $x_{ij}=0$. Cargas com $x_{ij}=1$ e TEMin$\in (t_k+aL, t_k+L]$ também retornam para Q_k, onde a ($0<a<1$) é um parâmetro fornecido pelo usuário e responsável por restringir o tamanho do horizonte de tempo.

Nos testes computacionais desta adaptação, foi utilizado sempre o valor de $a=0.5$, pois assim o algoritmo analisava as cargas e veículos disponíveis em todo intervalo e designava efetivamente apenas aquelas com $x_{ij}=1$ e TEMin$\in [t_k, t_k+0.5L]$. Desta forma, são priorizadas atribuições que envolvam as primeiras cargas disponíveis nos DE's, ou seja, aquelas cujo valor do TEMin pertença a primeira metade do horizonte de tempo considerado.

A atribuição permanente é feita de imediato, se para um veículo i existe um único $x_{ij}=1$ com TEMin$\in [t_k, t_k+aL]$. Quando um mesmo veículo i tem vários $x_{ij}=1$ com TEMin$\in [t_k, t_k+aL]$, as atribuições de forma definitiva seguem exatamente os mesmos critérios mencionados na seção 2.5 do capítulo 2.

Segundo estes critérios, primeiro é atribuído efetivamente ao veículo i a carga j com $x_{ij}=1$ e menor TEMin entre todas as carga com $x_{ij}=1$. Em seguida, seleciona-se outra carga $q \neq j$ com $x_{iq}=1$ e menor TEMin entre todas as cargas com $x_{iq}=1$. Recalcula-se a utilidade de designação da carga q para a nova rota do veículo i atualizada pela inserção anterior da carga candidata j, sendo encontrado um novo valor U'_{iq}. Se $U_{iq} - U'_{iq} \leq \rho$ a inserção da carga q é realizada, caso contrário, o produto q retorna ao conjunto Q_k.

Este procedimento é repetido para todas as cargas j designadas a um mesmo veículo em uma mesma iteração, ou seja, com x_{ij}=1. O parâmetro ρ é fornecido pelo usuário e determina até onde uma utilidade recalculada U'_{iq} poderá ser menor que a utilidade U_{iq} originalmente calculada.

3.4 - Gerador Aleatório de Instâncias GERAINST

Definido o PRDV a ser resolvido e apresentadas algumas adaptações para que o algoritmo MORSS pudesse resolvê-lo, seguiu-se à implementação computacional do algoritmo. Esta implementação foi realizada utilizando linguagem de programação C, compilador GCC, programa CPLEX, estação Sparc IPX com 64 MB de RAM e sistema operacional SunOS 4.1.1. Esta implementação será tratada no decorrer deste trabalho por rotina VEÍCULOS.

A rotina VEÍCULOS foi testada utilizando um conjunto de instâncias geradas por outro programa chamado GERAINST que será apresentado nesta seção. O código GERAINST, também implementado em linguagem C, cria vários arquivos de dados com informações relativas ao cenário inicial do problema, bem como referentes as suas posteriores alterações.

Assim, GERAINST fornece um arquivo de dados iniciais para o problema (Cenário 0) e um número N de arquivos (Cenário 1, Cenário 2,...,Cenário N) contendo as novas informações que atualizam os dados disponíveis. A Figura 3.5 ilustra os principais procedimentos da rotina GERAINST

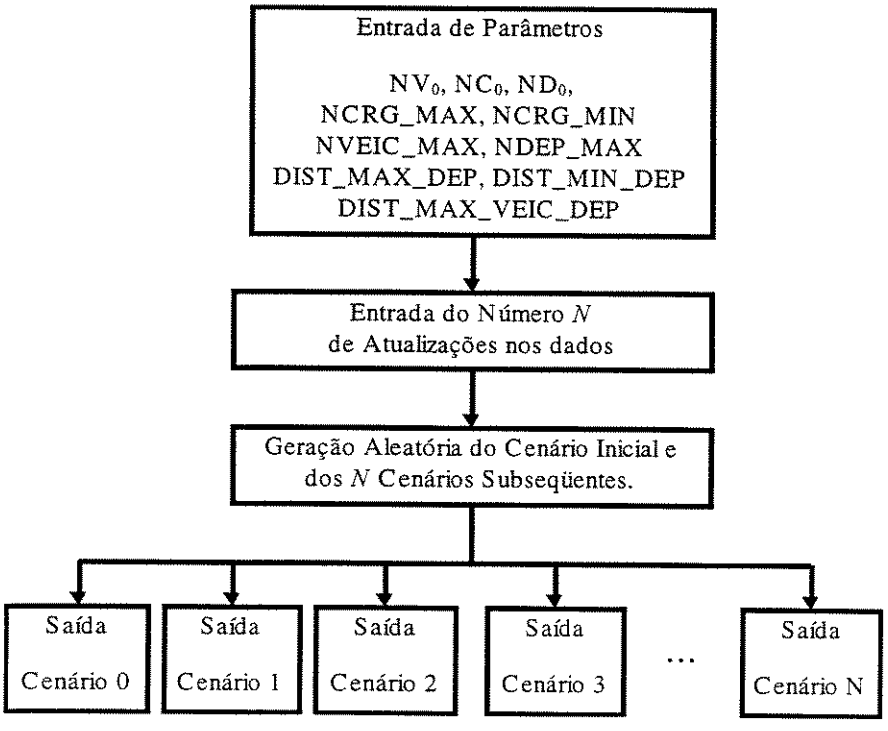


Figura 3.5 - Rotina GERAINST

Inicialmente, são fornecidos os seguintes parâmetros:

- **NV_0, NC_0, ND_0**: Trata-se do número inicial de veículos, cargas e depósitos. Estes são valores fornecidos pelo usuário que farão parte do cenário inicial do problema e, por conseguinte, do primeiro arquivo de dados gerado (Cenário 0). A partir destes números, o problema irá se expandir com novos veículos, cargas e depósitos sendo gerados por GERAINST e armazenados nos arquivos seguintes.

- **NCRG_MAX, NCRG_MIN**: Trata-se do número de cargas máximo e mínimo que poderão ser gerados de forma aleatória em cada novo cenário do problema. Desta forma, evita-se a ocorrência de um número exorbitante ou ínfimo de cargas a cada alteração de dados.

- **NVEIC_MAX, NDEP_MAX**: Trata-se do número de veículos e depósitos máximos a serem gerados aleatoriamente nos arquivos de dados posteriores ao arquivo inicial, ou seja, em Cenário 1, Cenário 2,..., Cenário K.

- **TEMP_MAX_DEP, TEMP_MIN_DEP**: Trata-se do valor máximo e mínimo que os tempos de viagem entre os depósitos poderão assumir.

- **TEMP_MAX_VEIC_DEP**: Trata-se do tempo máximo de viagem que poderá haver inicialmente entre um novo veículo incorporado ao problema e um ou vários depósitos.

Em seguida, o usuário também fornece o número N de cenários a serem criados. A rotina começa a processar os parâmetros fornecidos, gerando os dados e retornando ao final N arquivos com as diversas atualizações do problema.

Cada arquivo de dados contém o número de veículos, cargas e depósitos do problema após a incorporação dos novos dados gerados. Por exemplo, se no cenário 0 temos 3 veículos, 4 cargas e 3 depósitos, no cenário 1 poderemos ter 4 veículos, 7 cargas e 5 depósitos.

Também constam de cada arquivo os tempos de viagens entre os novos depósitos e os depósitos já existentes, os dados das cargas (depósito de embarque, depósito de desembarque, TEMin e TDMax) e o tempo de viagem inicial dos novos veículos aos depósitos.

Foi estabelecida uma independência entre a geração dos dados e a execução da rotina VEÍCULOS, pois primeiro foram criados todos os arquivos de dados usando GERAINST para que, em seguida, a rotina VEÍCULOS fosse executada. Porém, nada impede que a rotina GERAINST fosse executada simultaneamente, fornecendo os arquivos de dados a cada iteração da rotina VEÍCULOS ou após um certo número de iterações da mesma.

Para satisfazer a característica dinâmica do problema, VEÍCULOS não tem acesso prévio às informações geradas. Por exemplo, não pode avaliar as informações do arquivo de dados

Cenário_K+1.doc antes de ter recebido os dados de Cenário_K.doc. Também não tem acesso ao número de arquivos que será fornecido, ou seja, o número de atualizações a serem realizadas não consta dos dados recebidos pela rotina VEÍCULOS.

Estas restrições procuram garantir a execução do código como se as informações fossem obtidas em tempo real. A Figura 3.6 apresenta a execução da rotina VEÍCULOS nas N+1 primeira iterações, onde as entradas são N+1 arquivos de dados do problema.

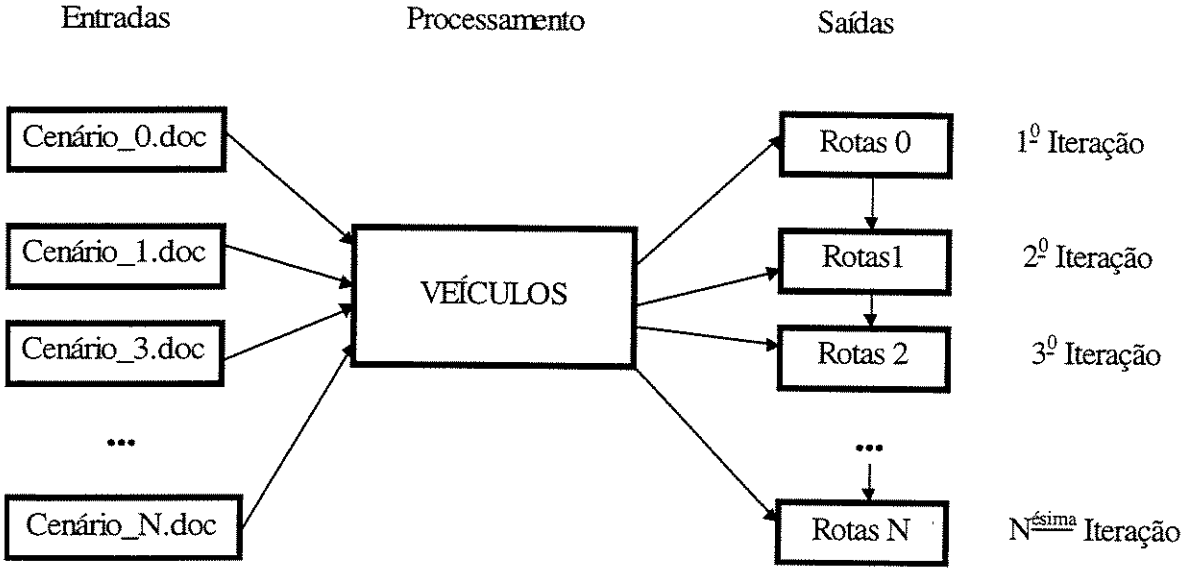


Figura 3.6 - Entrada e saída de dados na rotina VEÍCULOS.

Na primeira iteração, o arquivo de dados iniciais, Cenário_0.doc, é processado e um conjunto de rotas iniciais (Rotas 0) são construídas. Em seguida, ocorre a primeira alteração no cenário inicial através das informações contidas em Cenário_1.doc que, ao serem processadas, retornam outro conjunto de rotas para os veículos (Rotas 1).

A seta ligando Rotas 0 a Rotas 1 indica que as últimas foram construídas a partir das primeiras, ou seja, são mudanças feitas no conjunto de rotas contidas em Rotas 0 para que atendam em tempo real as alterações apresentadas pelo arquivo Cenário_1.doc. Todos estes procedimentos são seguidos até o $(N+1)^{ésimo}$ arquivo de dados ser processado, quando deixam de ocorrer alterações no problema.

Todas as saídas, a partir da $(N+1)^{ésima}$ iteração, passam a trabalhar com dados disponíveis que deixam de sofrer alterações. Todos os testes computacionais realizados para avaliar a rotina VEÍCULOS utilizaram instâncias fornecidas por GERAINST.

3.5 - Instâncias Avaliadas

A implementação computacional do algoritmo MORSS foi executada para diversas instâncias do PRDV definido na seção 3.2 e todas foram numericamente geradas usando a rotina GERAINST. Serão apresentadas nesta seção as características de três instâncias selecionadas para terem seus resultados analisados nas próximas seções.

A Tabela 3.1 apresenta alguns dados numéricos que dimensionam as instâncias I_1, I_2 e I_3. Nos PRDV´s avaliados, não faz sentido falar em número de cargas, veículos e depósitos porque novas cargas, veículos e depósitos tornam-se disponíveis em tempo real fazendo com que estes valores sejam constantemente alterados. Por isso, são colocados como aspectos numéricos os valores totais dos veículos (*TV*), cargas (*TC*) e depósitos (*TD*) e este dimensionamento das instâncias somente ficará estabelecido quando deixarem de ocorrer atualizações nos seus dados.

Tabela 3.1: Aspectos numéricos das Instâncias

I	*TV*	*TC*	*TD*	*TC/TV*	*Med1*	*Med2*
I_1	7	94	22	13,42	16,45	63,70
I_2	15	103	22	6,86	16,43	63,80
I_3	50	163	55	3,26	173,80	101,10

A coluna *TC/TV* mostra a relação carga/veículos ao final das atualizações. A instância I_1 tem 13,42 cargas por veículo, I_2 tem 6,86 cargas por veículo e I_3 tem 3,26 cargas por veículo. Deve-se lembrar que estes quocientes são relativos aos valores finais, pois inicialmente e durante as atualizações o número de cargas por veículo não confirma necessariamente esta proporção.

Também é fornecido o tempo médio de viagem entre os depósitos (*Med1*) e o tempo médio de viagem entre o TEMin e o TDMax das cargas (*Med2*). Neste problema, o tempo será medido em Unidades de Tempo (*UT*) que podem ser horas, dias, semanas, etc.

Verifica-se que os valores na coluna *Med1* são menores que em *Med2*, levando à idéia de que o tempo disponível para embarque e desembarque da carga é suficiente, se comparado com a média de tempo gasto para se ir de um depósito a outro. Por exemplo, na linha I_2 o tempo médio de viagem entre os depósito vale 16,43 UT e a diferença média entre os *TDMax* e *TEMín* das cargas é 63,80 *UT*.

Porém, o veículo nem sempre atenderá determinada demanda imediatamente, precisando adiar seu embarque, ou então deverá embarcá-la retornando em seguida para sua rota original e apenas mais tarde desembarcá-la.

A Tabela 3.2 apresenta as atualizações que ocorrem nas instâncias I_1, I_2 e I_3. Na primeira, quinta e nona colunas há os números dos novos cenários, respectivamente, das instâncias I_1, I_2 e I_3. As colunas com ΔV, ΔC e ΔD apresentam o número de novos veículos, cargas e

depósitos que aparecem a cada atualização de dados. A linha representada por Cen_0 mostra os valores constantes do cenário inicial das instâncias e nas linhas seguintes (Cen_1, Cen_2, ...,Cen_9) aparecem os valores adicionados por nove atualizações nos dados.

Tabela 3.2: Atualização de dados nas instâncias

I_1	ΔV	ΔC	ΔD	I_2	ΔV	ΔC	ΔD	I_3	ΔV	ΔC	ΔD
Cen_0	2	5	5	**Cen_0**	3	5	5	**Cen_0**	8	10	10
Cen_1	0	10	0	**Cen_1**	1	14	2	**Cen_1**	6	25	5
Cen_2	1	5	2	**Cen_2**	1	14	3	**Cen_2**	7	18	4
Cen_3	0	14	2	**Cen_3**	2	10	3	**Cen_3**	4	11	3
Cen_4	0	8	2	**Cen_4**	2	9	2	**Cen_4**	4	19	9
Cen_5	1	7	0	**Cen_5**	2	12	1	**Cen_5**	4	7	3
Cen_6	1	13	1	**Cen_6**	0	7	3	**Cen_6**	6	17	1
Cen_7	1	13	4	**Cen_7**	1	11	0	**Cen_7**	5	16	6
Cen_8	1	11	3	**Cen_8**	2	13	0	**Cen_8**	1	20	6
Cen_9	0	8	3	**Cen_9**	1	8	3	**Cen_9**	5	20	8
Total	7	94	22	**Total**	15	103	22	**Total**	50	163	55

Por exemplo, a instância I_1 apresenta os valores de seu cenário inicial na primeira linha (Cen_0), onde o número inicial de veículos é 2, o número de cargas é 5 e existem 5 depósitos. Na linha seguinte (Cen_1) ocorre a primeira atualização dos dados, onde 10 novas cargas são adicionadas ao problema. Neste caso, não ocorre atualização no número de veículos e depósitos. Na quinta atualização (linha Cen_5), mais um veículo é acrescentado à frota disponível, além de 7 novas cargas. Isto prossegue até a nona atualização (Cen_9) onde ocorrem mais 8 cargas e 3 depósitos.

Na última linha, os valores totais são listados e conferem com aqueles apresentados na Tabela 3.1. Todas as instâncias apresentadas sofrem nove sucessivas atualizações nos dados. Estas atualizações são fornecidas à rotina VEÍCULOS que as executam até que a última ocorra, quando então a rotina passa a trabalhar com um conjunto de informações totalmente conhecidas.

A Tabela 3.3 apresenta o comportamento das atualizações na média, ou seja, ΔV_m, ΔC_m e ΔD_m apresentam o número de veículos, cargas e depósitos totais divido pelo número de cenários existentes.

Tabela 3.3: Atualização média dos dados

Instâncias	ΔV_m	ΔC_m	ΔD_m
I_1	0,7	9,4	2,2
I_2	1,5	10,3	2,2
I_3	5,0	16,3	5,5

Por exemplo, em I_2 há 1,5 veículos chegando em média, incluindo aqueles existentes no cenário inicial, e também ocorrem 10,3 cargas e 2,2 depósitos em média. Esses dados permitem avaliar a característica de cada instância a ser resolvida pela rotina VEÍCULOS.

Na instância I_1, existe um grande número de cargas chegando (ΔC_m=9,4) para um menor número de veículos (ΔV_m=0,7). A instância I_2 apresenta a mesma quantidade média de depósitos (ΔD_m=2,2), mesmo número total de depósito que I_1 (TD=22 na Tabela 3.1) e uma quantidade média de cargas (ΔC_m=10,3) pouco maior que I_1.

Porém, a média de veículos disponíveis em cada cenário de I_2 (ΔV_m=1,5) é bem maior que em I_1 (ΔV_m=0,7). Desta forma, as instâncias I_1 e I_2 permitem avaliar a rotina VEÍCULOS quando para valores próximos de ΔC_m e ΔD_m são disponibilizadas pequenas e altas quantidades de veículos em média.

A instância I_3 permite avaliar o desempenho do algoritmo para um elevado valor médio de todos os dados disponíveis. Neste caso, ocorre um grande número médio de cargas (ΔC_m=16,3) que é acompanhado de uma alta quantidade de veículos por cenário (ΔV_m=5,0), além de um número médio de depósitos maior (ΔD_m=5,5). A instância I_3 impõe a solução de um problema com maior número de cargas e veículos envolvidos em cada cenário.

3.6 - Resultados Computacionais da rotina VEÍCULOS

Um dos objetivos deste capítulo é avaliar a adequação do algoritmo MORSS enquanto ferramenta útil na resolução de diferentes PRDV's. Neste contexto, após utilizar a rotina GERAINST para criar os dados das instâncias I_1, I_2 e I_3, o próximo passo foi executar e avaliar o desempenho de VEÍCULOS.

O algoritmo MORSS necessita de parâmetros que são fornecidos pelo usuário, onde destacam-se a fração a ($0<a<1$) do horizonte de tempo, os parâmetros que irão fornecer a curvatura da função $U_{ij}(1)$ e o limite máximo ρ de redução do valor da função U_{ij} quando recalculada.

Na maioria das instâncias testadas, incluindo as três aqui consideradas, utilizou-se a=0.5, pois permite observar todo o horizonte de tempo e decidir efetivamente apenas para cargas com $TEMin$ na primeira metade do intervalo.

A Figura 3.7 apresenta a curva que descreve o comportamento da função $U_{ij}(1)$ quando a variável t (atraso) oscila entre 0 e 10 UT. Neste caso, foram adotados os valores b=2, t_0=5, V_{min}=1 e V_{max}=3 para os parâmetros de $U_{ij}(1)$. Estes valores são os mesmos utilizados no cálculo das utilidades $U_{ij}(1)$ durante os testes computacionais apresentados neste capítulo.

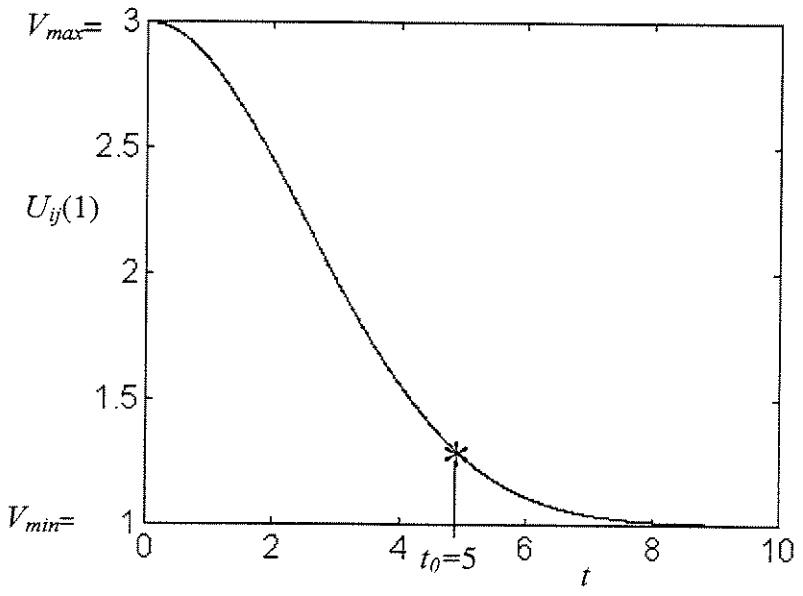


Figura 3.7 - Curva da função $U_{ij}(1)$

Conforme explicado na seção 3.3, a *Função Utilidade de Designação* U_{ij} será calculada utilizando a Equação 3.4, onde $U_{ij}(1)$ será fornecida pela Equação 3.5 e $U_{ij}(2)$ pela Equação 3.7. Quando não há atrasos, ou seja, quando t=0 ocorre $U_{ij}(1)=V_{max}=3$, porém, à medida que os atrasos começam a aumentar, $U_{ij}(1)$ diminui rapidamente até alcançar um valor mínimo quando t_0=5. A partir de t_0, os atrasos não mudam substancialmente o valor de $U_{ij}(1)$ que permanece próximo de $U_{ij}(1)=V_{min}=1$.

Por exemplo, quando ocorre atraso 2 *UT* o valor de $U_{ij}(1)$ está entre 2,5 e 3, enquanto para um atraso de 4 UT este valor fica próximo de 1,5. Neste caso, ocorre uma queda acentuada no valor de $U_{ij}(1)$ correspondente a um aumento de 2 *UT* no atraso. Quando o atraso aumenta de 6 para 8 *UT* o valor obtido para $U_{ij}(1)$ permanece entre 1,0 e 1,5, não havendo uma alteração muito acentuada na utilidade apesar de também haver um aumento de 2 *UT* no atraso.

Quando a solução do problema de designação fornece, para um mesmo veículo i, mais de uma carga j tal que x_{ij}=1 e $TEMin \in [t_k, t_k+aL]$, será adotado o critério explicado na seção 3.3.3. Relembrando, a carga candidata j com x_{ij}=1 e menor $TEMin \in [t_k, t_k+aL]$ será designada efetivamente e as demais cargas $q \neq j$ com x_{iq}=1 e $TEMin \in [t_k, t_k+aL]$ terão o valor de U_{iq} recalculado para a nova rota do veículo i após a inserção de j. Caso o novo valor U'_{iq} seja maior que o original ou não diminua mais que um certo parâmetro ρ definido pelo usuário, a carga q com menor *TEMin* também será designada ao veículo i.

Este procedimento é repetido após a inserção de j até que todas as cargas sejam inseridas ou o valor da utilidade de designação recalculado para todas elas seja maior que ρ. O valor de ρ adotado nesta adaptação é 0,05. Neste caso, permite-se que cargas com pequena

diminuição no valor da utilidade de designação sejam também designadas aos veículos, evitando que retornem ao conjunto de cargas disponíveis.

A Figura 3.8 apresenta as curvas de mudança na porcentagem de cargas entregues no prazo (%Prazo) para diferentes valores do tamanho do horizonte de tempo (L). O símbolo "*" na curva C1 indica o melhor resultado da instância I_1, quando 63,83% das cargas são desembarcas no prazo em um horizonte de tamanho L=6. Observa-se que a curva C1 passa a ter valores constantes a partir de L=20.

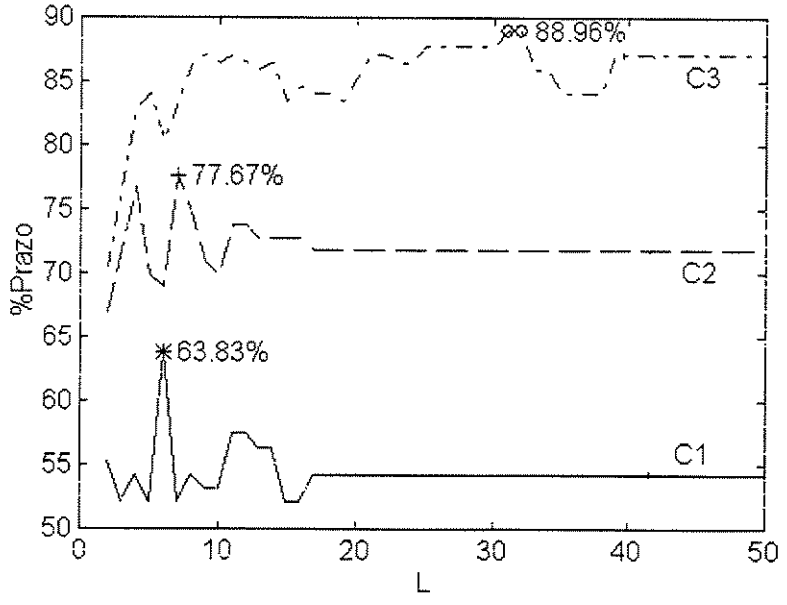


Figura 3.8 – Porcentagem de Cargas Entregues no Prazo

O melhor resultado da instância I_2 está representado pelo símbolo "+" na curva C2 e ocorre para L=7 com 77,67% das cargas no prazo, tornando-se constante a partir de L=20. Finalmente, a instância I_3 tem seu melhor resultado em dois valores de L, representados na curva C3 pelo símbolo "o". Neste caso, a melhor porcentagem de cargas entregues no prazo vale 88,96% e ocorre quando L=31 e L=32. A partir de L=40 a curva passa a ser constante.

Tabela 3.4: Melhores Resultados Obtidos

Instâncias	%Cargas Entregues no Prazo	*TCT*	L^*
I_1	63,83	44"	6
I_2	77,67	49"	7
I_3	88,96	1'17"	31 e 32

O resumo dos melhores resultados encontrados para I_1, I_2 e I_3 mais o respectivo tamanho do horizonte de tempo (L^*) são apresentados na Tabela 3.4. Também é apresentado o Tempo Computacional Total (*TCT*) gasto pela método na resolução de cada instância.

Observa-se a melhor taxa de entrega no prazo em problemas com maior número de veículos disponíveis. Apesar da proporção cargas/veículos não ser a mesma todo o tempo devido a dinâmica do problema, reflete de forma geral, uma situação onde um número reduzido de veículos precisou atender muitas cargas.

No caso da instância I_1, ao final das atualizações, ela apresenta 7 veículos e 94 cargas com 13,42 cargas por veículos e a melhor porcentagem de entrega no prazo de 63,83% em um horizonte de tempo de tamanho 6. Ao contrário, I_2 termina com um total de 15 veículos para 103 cargas apresentando 6,86 cargas por veículo, perfazendo 77,67% das cargas desembarcadas no prazo em L=7. Esses valores demonstram a influência da quantidade de veículos disponíveis no pronto atendimento das cargas (Tabela 3.1).

O *TCT* cresce a medida que aumenta o número de cargas envolvidas em cada instância. Por exemplo, na instância *I1* ocorre *TC*=94 (Tabela 3.1) e a rotina VEÍCULOS leva 44" para estabelecer a última rota desta instância (Tabela 3.4). Em *I3*, ocorre *TC*=163 e o *TCT* aumenta para 1'17".

Os resultados obtidos mostram que a partir de determinado valor de L a porcentagem de cargas entregues no prazo permanece constante. A rotina VEÍCULOS usa a idéia do Horizonte Rolante, onde a cada horizonte de tempo estabelecido é formado um conjunto de cargas candidatas:

$$C=\{q/q\in Q_k,\ \text{TEMin}\in [t_k,\ t_k+L] \text{ e } \text{TDmax}>t_k\}.$$

Alterações no tamanho de L mudam o intervalo $[t_k, t_k+L]$ e, consequentemente, as cargas que compõem o conjunto. Os resultados obtidos demonstraram que, a partir de determinado tamanho de L, as mesmas cargas estarão sempre contidas no conjunto C em cada iteração. Desta forma, o mesmo problema de designação passará a ser resolvido.

Por exemplo, as cargas que pertençam ao conjunto C formado quando L=35, em alguma iteração k durante a resolução da instância I_1, serão as mesmas contidas em C na k^{ésima} iteração quando L=40.

$$\{q/q\in Q_k,\ \text{TEMin}\in [t_k,\ t_k+35] \text{ e } \text{TDMax}>t_k\}=\{q/q\in Q_k,\ \text{TEMin}\in [t_k,\ t_k+45] \text{ e } \text{TDMax}>t_k\}$$

Esta coincidência em C, quando L ultrapassar determinado valor, ocorre porque o horizonte de tempo estabelecido deixa de selecionar cargas, ou seja, as cargas passíveis de avaliação depois de ultrapassado este limite passam a ser sempre as mesmas.

Isto gera soluções idênticas para os problemas de designação, dentro das mesmas iterações do algoritmo, em diferentes horizontes de tempo. Assim, repetem-se as atribuições cargas/veículos levando a mesma porcentagem de cargas entregues no prazo.

3.7 - Resultados Computacionais Calculando L

O fato mais importante observado nos resultados obtidos nestas instâncias e em outras que foram avaliadas é a existência de um tamanho específico para o horizonte de tempo que fornece o melhor resultado na porcentagem de cargas entregues no prazo. Isto conduziu a idéia de que seria possível determinar um tamanho ótimo de horizonte de tempo (L^*) capaz de levar o método a determinar rotas com maior número de desembarques dentro dos prazos exigidos.

Uma explicação para a eficiência destes L^* estaria na formação de cenários mais adequados em $[t_k, t_k+L]$, ou seja, determinados tamanhos de horizonte levariam o método a tomar decisões em subintervalos $[t_k, t_k+aL]$ onde existiriam um conjunto de cargas cuja atribuição aos veículos acabaria sendo mais conveniente. Assim, os conjuntos C formados a cada iteração usando um L^* selecionariam cargas cuja designação aos veículos determinariam rotas mais apropriadas às constantes alterações nos dados.

Os melhores L^* obtidos para I_1, I_2 e I_3 foram alcançados através de várias execuções do método, testando valores incrementais de L até que o melhor fosse encontrado, conforme demonstrado na Figura 3.7. Porém, no contexto dinâmico, nem sempre há possibilidade de se testar soluções para diferentes valores de L até que se alcance o melhor resultado.

Usando apenas a observação dos resultados obtidos e sua relação com os dados disponíveis, tentou-se estabelecer uma forma de calcular L durante a execução do algoritmo. Desta maneira, a rotina teria um modo independente de estabelecer o tamanho do horizonte a cada iteração. Relembrando, o critério para escolha do valor mínimo do horizonte de tempo na $k^{\text{ésima}}$ iteração é $t_k=\min\{TEMin \,/\, \forall q \in Q_k \backslash Q_A\}$, onde Q_k é o conjunto de cargas disponíveis nesta iteração. O conjunto Q_A contém as cargas que nas iterações anteriores não foram atribuídas aos veículos e apresentavam TEMin=t_{k-1}. Isto posto, o valor de L poderia ser determinado como sendo a média do *TEMin* das cargas disponíveis:

$$L_1 = \frac{\sum_{j \in Qk} \text{TEMin}_j}{|Q_k|}. \qquad (3.13)$$

Na Equação 3.13, TEMin$_j$ é o Tempo de Embarque Mínimo da carga $j \in Q_k$. O valor de L_1 será o somatório dos TEMin de todas as cargas ainda não designadas aos veículos, dividida pela quantidade destas cargas na $k^{\text{ésima}}$ iteração. Assim, o intervalo de tempo será limitado inferiormente pelo menor TEMin e superiormente por este valor adicionado da média dos TEMin's ($t_k + L_1$) das cargas disponíveis.

Na Tabela 3.5, a porcentagem de entregas no prazo usando L_1 ficou abaixo do melhor valor encontrado pelo método utilizando L como um valor fixo fornecido pelo usuário. Porém, os resultados com L_1 tem a vantagem de não serem determinados testando valores, mas calculados durante a execução do algoritmo.

Tabela 3.5: Resultados de L_1 comparados aos melhores obtidos com L fixo

Instâncias	*%Cargas Entregues no Prazo usando* L_1	*%Cargas Entregue no Prazo usando L fixo*
I_1	58,51	63,83
I_2	70,87	77,67
I_3	87,12	88,96

Outra forma proposta para o cálculo de L baseia-se no tempo médio de viagem entre os Depósitos de Embarques (DE's) e Depósitos de Desembarques (DD's) das cargas disponíveis. A determinação do valor de t_k continua utilizando o mesmo critério (t_k=min{$TEMin$ / $\forall q \in Q_k \backslash Q_A$}), mas o limite superior do intervalo será obtido pela adição a t_k do valor médio do tempo de viagem entre o DE e DD. Este valor médio é calculado pela expressão

$$L_2 = \frac{\sum_{j \in Qk} t_{(DE\ DD)j}}{|Q_k|}. \qquad (3.14)$$

Na Equação 3.14, O termo $t_{(DE\ DD)j}$ representa o tempo de viagem entre o DE e DD da carga $j \in Q_k$. O valor de L_2 será o somatório do tempo disponível entre o DE e DD das cargas disponíveis dividido pelo número de cargas disponíveis.

A Tabela 3.6 mostra os valores obtidos usando L_2, comparando aos de L_1 e aos melhores valores obtidos com L fixo.

Tabela 3.6: Resultados de L_2 comparados aos de L_1 e L fixo

Instâncias	*% Cargas Entregues no Prazo usando* L_1	*% Cargas Entregues no Prazo usando* L_2	*% Cargas Entregues no Prazo usando L fixo*
I_1	58,51	53,19	63,83
I_2	70,87	74,76	77,67
I_3	87,12	85,10	88,96

A porcentagem de cargas entregues no prazo em L_2 na instância I_2 foi melhor que a obtida usando L_1, porém o valor obtido por L_2 foi menor em I_1 e I_3. Todos os resultados alcançados por L_2 estão abaixo do melhor valor obtido usando horizonte fixo.

A Figura 3.9 compara as porcentagens obtidas usando L_1 e L_2 com os resultados variando-se o tamanho do horizonte L. A reta contínua representa a porcentagem obtida com L_1 e a pontilhado o valor de L_2. Observa-se que em L_1 a porcentagem ficou próxima dos melhores resultados mostrados em C1 e C3 usando L como parâmetro fixo fornecido pelo usuário. A situação muda na curva da instância I_2, quando a porcentagem entregue no prazo obtida com L_1 apresenta-se mais próxima dos melhores resultados obtidos em C2.

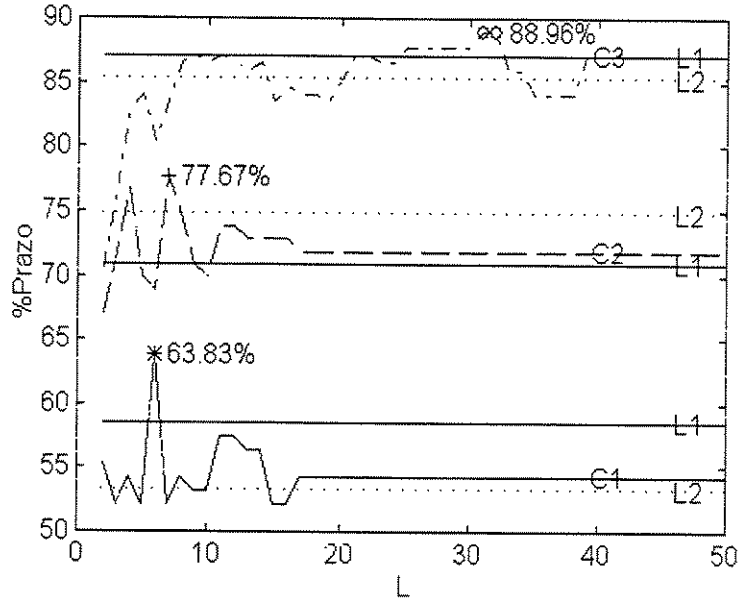


Figura 3.9 – Resultados com L_1 e L_2 comparados aos de L

Para comparar o comportamento do método ao trabalhar com L fixo ou usando L_1 e L_2, serão apresentadas tabelas com os valores utilizados nos diversos horizontes de tempo ao resolver a instância I_2.

As colunas da Tabela 3.7 representam os horizontes de tempo formados a cada iteração e os valores dos veículos, depósitos e cargas disponíveis neste período. As colunas C e CA são, respectivamente, o número de cargas candidatas (C) e de cargas candidatas atribuídas aos veículos de forma definitiva (CA). A coluna CD contém o número de cargas disponíveis, ou seja, todas as cargas do problema menos o somatório daquelas atribuídas aos veículos até a iteração anterior.

Tabela 3.7: Resultados de I_2 usando L=7

Horizonte	*Veículos*	*Depósitos*	*Cargas*	*CD*	*C*	*CA*
[2,9]	3	5	5	5	3	2
[3,10]	4	7	19	17	14	6
[4,11]	5	10	33	25	23	7
[6,13]	7	13	43	28	27	10
[7,14]	9	15	52	27	24	9
[8,15]	11	16	64	30	25	14
[9,16]	11	19	71	23	23	14
[13,20]	12	19	82	20	18	6
[14,21]	14	19	95	27	25	7
[15,22]	15	22	103	28	24	8
[16,23]	15	22	103	20	18	9
[20,27]	15	22	103	11	11	11
Total	15	22	103	-	-	103

Por exemplo, no horizonte de tempo [6,13] o problema apresenta um total de 7 veículos, 13 depósitos e 43 cargas. Existem 28 cargas disponíveis, ou seja, as 43 que no total chegaram ao problema menos 2 atribuídas no intervalo [2,9], 6 atribuídas em [3,10] e 7 em [4,11].

Das cargas disponíveis, 27 apresentam *TEMin* dentro do horizonte de tempo atual formando o conjunto de cargas candidatas, porém o método decide que somente 10 cargas deste conjunto serão embarcadas pelos veículos. Após o nono horizonte, os valores dos veículos, cargas e depósitos tornam-se constantes porque deixam de ocorrer atualizações nos dados do problema, já que foi suposto um número de nove atualizações nos dados do problema.

Na Tabela 3.8 o tamanho do horizontes de tempo varia em função da média do *TEMin* das cargas disponíveis. Por exemplo, na segunda iteração da rotina (segunda linha da Tabela 3.7) o problema apresenta um total de 17 cargas disponíveis (*CD*), onde o cálculo da média dos seus *TEmin*'s (arredondando-se o valor obtido) vale 6 *UT*. Assim, o horizonte de tempo formado nesta iteração será [3,9], já que 4 *UT* é o menor *TEMin* das 17 cargas disponíveis.

Tabela 3.8: Resultados de I_2 calculando L_1

Horizonte	*Veículos*	*Depósitos*	*Cargas*	*CD*	*C*	*CA*	L_1
[2,7]	3	5	5	5	3	2	5
[3,9]	4	7	19	17	10	6	6
[4,12]	5	10	33	25	24	4	8
[6,14]	7	13	43	31	30	12	8
[7,16]	9	15	52	28	28	13	9
[8,18]	11	16	64	27	27	9	10
[9,20]	11	19	71	25	25	15	11
[11,26]	12	19	82	21	21	8	15
[13,29]	14	19	95	26	26	9	16
[14,32]	15	22	103	25	25	11	18
[15,32]	15	22	103	14	14	13	17
[23,46]	15	22	103	1	1	1	23
Total	15	22	103	-	-	103	-

Quando passam a existir 25 cargas disponíveis na iteração seguinte, a média do *TEMin* das cargas será 8 *UT* sendo estabelecido o intervalo [4,12]. A partir do horizonte [7,16], ocorre coincidência entre os valores de *CD* e *C*, demonstrando que o tamanho dos intervalos obtidos por L_1 passam, a partir de determinada iteração, a incluir o *TEMin* de todas as cargas disponíveis.

Observa-se que o tamanho de L_1 aumenta a cada iteração e apenas no intervalo [14,32] o valor de L_1 passa de 18 para 17 em [15,32] na Tabela 3.8. A média dos *TEMin* tendem a crescer porque a idéia é que as novas cargas que chegam apresentem seu *TEMin* próximo ou superior ao momento atual. Não se está considerando a possibilidade de que nas últimas iterações se tornem conhecidas informações sobre cargas que já estavam disponíveis para embarque desde as primeiras iterações.

Isto posto, o valor de L_1 tenderá a crescer ou manter-se elevado no decorrer das atualizações. Um horizonte de tamanho grande aumenta a possibilidade de que o conjunto de cargas disponíveis (*CD*) passe a ser idêntico ao de cargas candidatas (*C*), explicando a coincidência de valores nas colunas *CD* e *C* já a partir do intervalo [6,14].

A Tabela 3.9 apresenta os horizontes de tempo cujo tamanho é a média das distâncias entre os DE e DD das cargas disponíveis. Neste caso, o interessante é a coincidência total entre os valores de *CD* e *C*. Os tamanhos do horizonte obtidos por L_2 revelaram que a média das distâncias acabaram estabelecendo horizontes grandes o bastante para incluir todos os *TEMin* das carga disponíveis a cada iteração.

Tabela 3.9: Resultados de I_2 calculando L_2

Horizonte	Veículos	Depósitos	Cargas	CD	C	CA	L_2
[2,14]	3	5	5	5	5	2	12
[3,18]	4	7	19	17	17	11	15
[4,20]	5	10	33	20	20	10	16
[6,25]	7	13	43	20	20	10	21
[7,24]	9	15	52	19	19	9	17
[8,28]	11	16	64	22	22	14	20
[9,27]	11	19	71	15	15	15	18
[13,30]	12	19	82	11	11	4	17
[14,31]	14	19	95	20	20	11	17
[15,30]	15	22	103	17	17	7	15
[16,31]	15	22	103	10	10	9	15
[23,34]	15	22	103	1	1	1	11
Total	15	22	103	-	-	103	-

3.8 - Conclusões

Neste capítulo foi proposto um PRDV, apresentadas as adaptações realizadas no algoritmo MORSS para solucioná-lo e apresentado os resultados computacionais obtidos. O PRDV proposto foi baseado em um modelo teórico que permitisse avaliar o desempenho do algoritmo MORSS.

As instâncias estabelecidas apresentavam uma dimensão elevada de cargas e veículos envolvidos e, apesar disso, o método comportou-se bem ao manter um bom nível de cargas entregues no prazo. De certa forma, isto demonstra que o método respondeu positivamente ao solucionar um PRDV diferente daquele para o qual foi idealizado.

Os resultados obtidos apresentaram uma relação entre a porcentagem de cargas entregues no prazo e o tamanho do horizonte de tempo utilizado, pois uma melhor porcentagem foi alcançada para determinados valores de L. O inconveniente nesta relação é o fato do tamanho de L ser um parâmetro fornecido pelo usuário à rotina.

Num contexto real e dinâmico, o usuário poderá não saber qual o melhor valor a ser adotado para L já que alguns dados não são conhecidos a priori e estarão sujeitos a constantes alterações. Uma possibilidade seria testar diferentes soluções, a cada iteração, variando L até se encontrar o valor mais adequado para este parâmetro. Porém, este procedimento seria proibitivo em termos de tempo já que novas informações estariam chegando enquanto o usuário estivesse testando diferentes valores de L.

Uma proposta feita neste capítulo foi o uso de fórmulas para o cálculo do valor de L a cada iteração. O cálculo de L forneceu maior independência ao método, pois o valor do parâmetro deixou de ser fornecido pelo usuário e passou a ser obtido segundo determinados critérios. As duas fórmulas utilizadas para determinar L basearam-se mais em observações empíricas dos dados disponíveis do que num estudo voltado para avaliar os diversos fatores influentes em L.

Os resultado obtidos demonstram que as fórmulas utilizadas para o cálculo de L (Equações 3.13 e 3.14) forneceram resultados próximos aos obtidos usando um valor fixo para L. Deve-se ressaltar que o valor fixo de L corresponde ao melhor resultado obtido após terem sido testados diferentes valores de L entre 1 e 40UT, enquanto os fornecidos por L_1 e L_2 foram encontrados em uma única execução do método.

Desta forma, pode-se concluir que os principais resultados deste capítulo foram a apresentação do desempenho desta adaptação do algoritmo MORSS nas instâncias do PRDV estabelecido, a verificação de uma relação entre L e o desempenho do método e uma sugestão para maior independência na obtenção de L através do cálculo de seu valor a cada iteração da rotina.

CAPÍTULO 4

RESOLUÇÃO DE UM PROBLEMA DINÂMICO DE PROGRAMAÇÃO DE MÁQUINAS PARALELAS COM CUSTO DE TROCA DE FERRAMENTAS DEPENDENTE DA SEQÜÊNCIA E RESTRIÇÕES DE TEMPO

4.1 Introdução

Neste capítulo, será apresentada a segunda adaptação realizada no algoritmo MORSS objetivando a resolução de um Problema Dinâmico de Programação de Máquinas Paralelas com Custo de Troca de Ferramentas Dependentes da Seqüência e Restrições de Tempo. Este problema será tratado no decorrer da dissertação simplesmente por Problema Dinâmico de Programação (PDP), onde Programação será adotada como tradução para o termo inglês "scheduling".

O PDP que será definido tem sua motivação em um problema industrial real envolvendo a fabricação de diferentes tipos de embalagens de vidro em um processo de produção semi-contínuo. Estudos já foram realizados para o modelo estático deste problema e o PDP trata de uma extensão para o caso dinâmico.

Um dos objetivos deste capítulo é avaliar a adaptabilidade do algoritmo MORSS, quando utilizado para resolver o PDP, pois este problema apresenta características (custo de troca, reseqüenciamento, etc.) que não estavam presentes no Problema de Roteamento Dinâmico de Veículos. Desta forma, foram realizadas alterações e inovações no método para que este pudesse solucionar adequadamente o PDP proposto.

O desempenho computacional desta adaptação será avaliado através da execução de testes e análise de resultados. Visando estabelecer uma base para comparação dos resultados obtidos, um método chamado Rotina de Busca e Inserção Aleatória (RBI) também será utilizado para resolver o PDP. Este método realiza uma busca mais ampla no espaço de soluções possíveis e apresenta resultados que servem como critério de comparação com o algoritmo MORSS. A comparação com soluções ótimas foi descartada devido a complexidade de se resolver o problema.

Na seção 4.2, será apresentada uma breve abordagem sobre Problemas de Programação. O problema industrial que serviu de motivação para se estabelecer o PPD será explicado na seção 4.3. A definição do PPD a ser resolvido neste capítulo encontra-se na seção 4.4 e em

4.5 explica-se em detalhes as adaptações realizadas no algoritmo MORSS. A RBI será descrita na seção 4.6. Os resultados computacionais são apresentados e analisados na seção 4.7. As conclusões estão relatadas na seção 4.8.

4.2 – Problemas de Programação

Programação é a alocação de recursos para efetuar um conjunto de tarefas (Baker, 1974). Blazewicz *et al.* (1996), define um Problema de Programação como sendo composto pelos conjuntos $T=\{T_1, T_2,..., T_n\}$ de tarefas, $P=\{P_1, P_2,..., P_m\}$ de processadores ou máquinas e $R=\{R_1, R_2,..., R_s\}$ de recursos adicionais. Neste caso, o resultado da Programação consiste na designação dos processadores contidos em P (e possivelmente dos recursos em R) para as tarefas em T, tal que as seguintes condições sejam satisfeitas:

- A todo momento cada processador é designado quando muito para uma tarefa e cada tarefa é processada por no máximo um processador. Porém, estas hipóteses podem ser relaxadas;
- A tarefa T_j é processada no intervalo de tempo $[r_j, \infty)$, onde r_j é o tempo no qual a tarefa T_j torna-se disponível para ser processada;
- Todas as tarefas são completadas pelos processadores;
- Se $T_i \prec T_j$, a tarefa T_j não começará antes de T_i ser completada, pois $\prec$ indica uma relação de precedência em T;
- Uma Programação de tarefas será denotado como *sem interrupções* caso o processamento das mesmas não possa ser interrompido. Será uma Programação de tarefas *com interrupções* caso a tarefa possa ser interrompida e, mais tarde, recomeçada no mesmo ou em outro processador. Neste último caso, o número de interrupções de cada tarefa será finito;
- As restrições de recursos, se houver, são satisfeitas.

Um objetivo na resolução dos Problemas de Programação será encontrar uma designação das tarefas aos processadores, estabelecendo uma programação para os mesmos que minimize ou maximize determinada medida de desempenho. Esta medida poderá ser, por exemplo, o tempo médio de permanência das tarefas no sistema de processamento, o tamanho da programação dos processadores, o atraso no término das tarefas, etc.

No decorrer deste trabalho, será utilizado o termo máquina ao invés de processador e produtos no lugar de tarefas. As máquinas (processadores) poderão ser paralelas ou dedicadas. Máquinas dedicadas são especializadas na produção de certos produtos (execução de certas tarefas) e paralelas são aquelas capazes de efetuar as mesmas funções.

No caso de máquinas paralelas, três tipos de máquinas podem ser citadas:

- **Máquinas Idênticas**: Processam todas os produtos com a mesma velocidade.

- **Máquinas Uniformes**: Têm diferentes velocidades de processamento, mas cada máquina tem velocidade de processamento que independe do produto, ou seja, sua velocidade de processamento é constante.
- **Máquinas não relacionadas**: A velocidade de processamento em cada máquina depende do tipo de produto processado.

Isto posto, um Problema de Programação de Máquinas Paralelas pode ser definido quando há n produtos para serem processados em quaisquer uma das m máquinas existentes, onde supomos que cada produto j necessite de um tempo de processamento T_{ij} em cada máquina i.

Existem aplicações envolvendo Problemas de Programação em diferentes áreas como, por exemplo, linha de produção de uma fábrica, atribuição de códigos ao processador de um computador, alocação de mão de obra para execução de serviços, etc. Os primeiros estudos nesta área foram motivados por problemas de manufatura, desta forma, diversos termos como tarefas, máquinas, recursos, trabalhos são freqüentemente utilizados. Tarefas ou trabalhos disputando recursos ou máquinas podem ter, contudo, naturezas diversas como dinheiro, energia, etc.

4.3 – Problema Industrial

Nesta seção, será apresentado um problema industrial real modelado como um Problema de Programação de Máquinas Paralelas com Custo de Troca de Ferramentas Dependente da Seqüência. Este problema servirá como motivação para o principal objetivo deste capítulo que é definir e resolver um problema de programação dinâmico e bi-critério. Neste caso, procura-se minimizar tanto o custo envolvido como o atraso nos prazos de entrega dos produtos, considerando que novas tarefas chegam ao sistema de produção em tempo real.

O problema industrial envolve a produção de diferentes tipos de embalagens de vidro em um processo de manufatura semi-contínuo. Este problema foi estudado em Aguilera (1992 e 1993) e Gonthier (1990), onde foram definidas diversas restrições envolvidas, propostos métodos de resolução e obtidos resultados computacionais.

A Figura 4.1 apresenta uma visão geral do processo de produção no problema. O processo inicia-se em um forno onde a matéria prima é fundida para, em seguida, ser modelada nas máquinas, conforme o tipo de embalagem a ser produzido. Nesta etapa, surge o problema de programação envolvendo a produção dos diferentes modelos pelas máquinas disponíveis. Na etapa seguinte, as embalagens produzidas passam por um processo de resfriamento, são avaliadas pelo controle de qualidade e, finalmente, empacotadas para serem enviadas aos estoques.

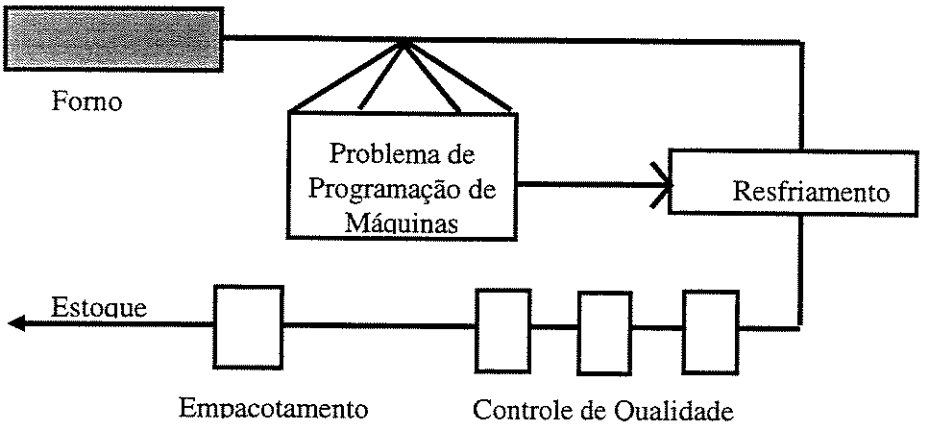


Figura 4.1 - Processo de Produção

A Figura 4.2 apresenta o processo de manufatura semi-contínuo cuja execução ocorre em dois passos. No primeiro, as matérias-primas (areia e soda) são fundidas em um forno e um fluxo contínuo de vidro fundido é enviado às máquinas dispostas de forma paralela. No segundo passo, antes de chegar até as máquinas, o fluxo de vidro é cortado formando bolhas que caem em um primeiro molde onde ganham forma os gargalos das embalagens. Após isso, os moldes seguem para as máquinas que completam a forma das embalagens, conforme o tipo desejado. Cada unidade industrial tem diversas máquinas produzindo simultaneamente diferentes modelos.

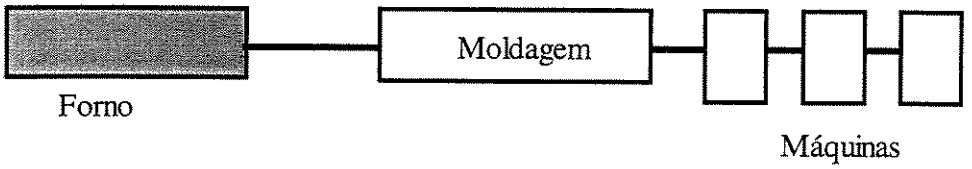


Figura 4.2 - Processo de manufatura semi-contínuo

Neste processo, há um período de troca entre a produção de dois diferentes modelos de embalagens no qual as máquinas permanecem paradas. A troca de modelos é muito complexa, devido às dificuldade operacionais requeridas para mudar os moldes e o tempo necessário para fazer o ajuste nas máquinas. O tempo de troca varia de alguns minutos até várias horas dependendo da quantidade de mudanças necessárias. A Figura 4.3 ilustra a troca de operações.

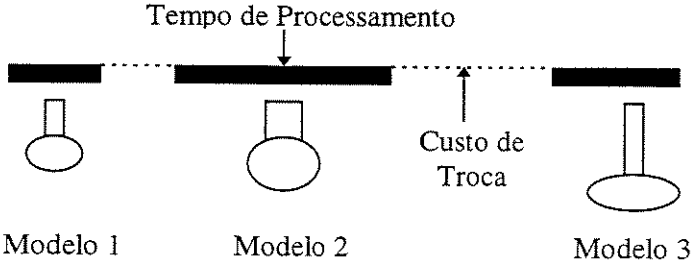


Figura 4.3 - Troca de operações

A complexidade para passar de um modelo de embalagem a outro depende daquele inicialmente produzido pela máquina. Isto acarreta um custo, denominado custo de troca, que pode levar em conta diferentes fatores:

- O desperdício de tempo de produção e fluxo de material pela máquina parada;
- O número de horas de trabalho necessárias para troca de moldes;
- As ferramentas requeridas pelas operações de troca;
- O grau de dificuldade da troca.

Desta forma, o correspondente Problema de Programação em Máquinas Paralelas apresenta custo de troca dependente da seqüência. O termo dependente da seqüência significa que a ordem de passagem dos produtos pela máquina influencia o resultado final do custo de troca de ferramentas. Considerando-se, por exemplo, três produtos i, j e k, teremos $c_{ik} \neq c_{ij}$, logo os custos de uma seqüência $S_1=\{i, j, k\}$ e $S_2=\{i, k, j\}$ são diferentes. Este problema tem, portanto, uma matriz de custo associada onde os valores correspondem aos custos de passagem de um produto i ao produto j.

Na resolução do Problema de Programação, o objetivo é reduzir o custo de troca global além de atender diferentes restrições do modelo como respeitar as datas de entrega dos produtos (*due-dates*), o momento em que os produtos estão disponíveis para serem processados (*ready time*), etc.

O Problema de Programação poderá envolver um conjunto de n embalagens de vidro e apenas uma máquina. Neste caso, cada embalagem passa uma única vez pela máquina, estabelecendo-se um ciclo de produção envolvendo o conjunto de embalagens. Esse tipo de Problema de Programação pode ser solucionado como um Problema do Caixeiro Viajante (Gonthier, 1990), onde a máquina corresponde ao caixeiro e as embalagens são as respectivas cidades. O problema pode consistir em encontrar o ciclo de produção com o menor custo de troca entre os diferentes modelos de embalagens que corresponde à distância entre as cidades.

Quando consideramos várias máquinas paralelas, o problema é modelado como um Problema de Roteamento de Veículo (PRV). Os veículos representam as máquinas, os clientes representam as embalagens a serem processadas, as distâncias entre as cidades são os custos de troca e a capacidade dos veículos é o horizonte de produção em cada máquina.

Um método de duas fases foi adaptado para resolver o caso estático do Problema de Programação em Máquina Paralelas com Custo de Troca de Ferramentas Dependente da Seqüência (Aguilera *et al*, 1995 e 1996, França *et al*, 1996, e Kessous *et al* 1994). Neste caso, primeiro é resolvido um Problema de Designação Generalizado que estabelece o conjunto de embalagens atribuído a cada máquina. Em seguida, é resolvido o Problema de Programação em cada máquina que, neste caso, é modelado pelo Problema do Caixeiro Viajante.

4.4 – Estabelecendo um Problema Dinâmico de Programação de Máquinas Paralelas com Custo de Troca de Ferramentas Dependente da Seqüência e Restrições de Tempo

O problema que será estabelecido e solucionado neste capítulo trata-se de uma extensão do caso estático do Problema de Programação em Máquina Paralelas com Custo de Troca de Ferramentas Dependente da Seqüência, para um contexto dinâmico. O caráter dinâmico do problema será fornecido pela chegada de novos produtos ao sistema de produção em tempo real. Enquanto as máquinas executam a programação estabelecida para os produtos existentes, novos produtos tornam-se disponíveis no decorrer do tempo.

Não há conhecimento prévio de quantos novos produtos poderão chegar e seus dados somente serão conhecidos quando ocorrer a atualização na quantidade de produtos disponíveis. Logo, tais informações poderão ser notificadas ao responsável pela linha de produção simultaneamente à execução ou determinação da seqüência de produtos a serem processados pelas máquinas.

Está sendo suposto que o problema dinâmico ocorre em uma linha de produção nos moldes do problema industrial apresentado. Serão consideradas três máquinas paralelas e não relacionadas, ou seja, todos os produtos poderão ser atribuídos a quaisquer uma das máquinas e o tempo de processamento do produto pode diferir em cada uma delas. O conjunto $M=\{M_1, M_2, M_3\}$ representará tais máquinas. Não ocorrerá aumento no número de máquinas disponíveis ou alterações nas suas características, assim, sempre haverá três máquinas paralelas e não relacionadas.

Os produtos podem chegam a todo instante na linha de produção. Logo, em um instante inicial I_0, existirá um conjunto de produtos $P_0=\{P_{01}, P_{02},..., P_{0r}\}$ a ser designado às máquinas onde P_{0i} é o $i^{ésimo}$ produto disponível no instante I_0. Ao ocorrer a primeira chegada de novos produtos, no instante I_1, será formado outro conjunto $P_1=\{P_{11}, P_{12},..., P_{1s}\}$. Desta forma, a cada instante I_t no qual ocorram novos produtos, haverá um conjunto $P_t=\{P_{t1}, P_{t2},..., P_{tu}\}$ de produtos disponíveis com pelo menos um produto $P_{tk} \notin P_{t-1}$ em I_{t-1}.

Uma vez que o produto esteja disponível para ser designado às máquinas, ou seja, faça parte de algum P_t, não serão consideradas alterações posteriores nos seus dados. As informações associadas a um novo produto P_{tj} são:

- **Tempo de Chegada (r_j)**: Instante no qual o produto P_{tj} estará pronto para ser inserido na linha de produção da máquina (*ready time*).
- **Data de Entrega (d_j)**: Instante em que o produto P_{tj} deverá terminar de ser processado pela máquina. Trata-se do prazo máximo estabelecido para entrega do produto (*due-date*).
- **Tempo de Processamento (p_{ij})**: Quantidade de tempo que o produto P_{tj} permanecerá sendo processado pela máquina M_i (*Processing Time*).
- **Custo de Troca (c_{kj})**: Valor associado ao custo de se trocar o molde do produto P_{tk} para produzir o produto P_{tj}.

A restrição quanto ao tempo de chegada (r_j) não poderá ser violada neste problema, pois não faz sentido inserir na linha de produção um produto que ainda não esteja pronto para ser processado. Por exemplo, as informações sobre um produto P_{tj} tornam-se disponíveis ao problema em I_t, porém, P_{tj} poderá apresentar um $r_j > I_t$. Desta forma, seus dados estão disponíveis ao problema em I_t, mas o início de seu processamento não pode começar antes do instante r_j.

Por outro lado, a restrição quanto a data de entrega (d_j) poderá ser violada. Em um problema dinâmico é mais importante entregar o produto com atraso do que não entregar. Assim, atrasos serão permitidos e o atraso na entrega de um produto P_{tj} será $D_j = max\{C_j - d_j, 0\}$, onde o termo C_j representa o instante de tempo em que o produto terminou de ser processado (*completion time*) e D_j o valor do atraso (*tardiness*).

Neste problema, caso um produto disponível tenha seu processamento concluído antes da data de entrega (d_j) ele será penalizado. A imposição desta penalidade está relacionada ao custo de estoque, isto é, o gasto extra com armazenagem para acomodar estes produtos. Logo, a antecipação na entrega de um produto será $E_j = max\{d_j - C_j, 0\}$ onde E_j é o valor desta antecipação no prazo (*earliness*).

Um dos objetivos do problema será a entrega do produto pela máquina em data próxima a sua data de entrega (d_j), minimizando tanto as antecipações (E_j) como os atrasos (D_j) no término de processamento dos produtos P_{tj}.

Não há restrições de precedência exigindo que determinado produto P_{tk} deva ser processado antes de certo produto P_{tj}. Porém, as dificuldades de passar do processamento de um produto para outro, relacionadas com a troca de moldes e ajuste das máquinas, continuam dependendo do produto que inicialmente encontra-se na máquina (dependente da seqüência). O custo de troca de cada produto em relação aos demais (c_{ik}) será fornecido ao problema quando os mesmos tornam-se disponíveis e, em geral, teremos $c_{ik} \neq c_{jk}$.

Isto posto, o Problema de Programação Dinâmico com Custo de Troca de Ferramentas Dependente da Seqüência pode ser modelado como um Problema de Roteamento Dinâmico de Veículos (PRDV). Neste caso, os custos de troca (c_{jk}) são representados pelas distâncias entre as cidades e as máquinas correspondem aos veículos. O tempo de chegada (r_j) e a data de entrega (d_j) correspondem, respectivamente, ao prazo para embarque e desembarque das cargas.

O objetivo do problema dinâmico definido será alocar os produtos às máquinas e determinar as suas seqüências em cada máquina, procurando minimizar o custo total de troca e as somas das penalidades por atrasos e antecipações na entrega de todos os produtos. Este objetivo deve ser atingido dentro de um contexto dinâmico, ou seja, atendendo as atualizações constantemente realizadas no sistema de produção e levando-se em conta futuras alterações.

As restrições e objetivos, estabelecidos nesta seção, definem o Problema Dinâmico de Programação de Máquinas Paralelas com Custo de Troca Dependente da Seqüência e Restrições de Tempo. Este problema, conforme já mencionado, será tratado simplesmente como Problema Dinâmico de Programação (PDP).

4.5 - Adaptação do Algoritmo MORSS ao Problema Proposto

O Algoritmo MORSS será adaptado para resolver o PDP e, desta forma, o método será avaliado na resolução de um problema que não é originalmente de roteamento de veículos, mas que pode ser modelado como tal. As adaptações foram realizadas no subsistema SCHEDULE, seguindo a proposta apresentada na seção 3.3 do capítulo anterior. Nesta segunda adaptação, foram estabelecidas novas *Funções Utilidades de Designação* e incorporou-se ao algoritmo uma heurística de busca em vizinhança.

O algoritmo MORSS baseia-se na idéia do *Horizonte Rolante* e nas *Funções Utilidades de Designação*. No problema proposto, a cada horizonte de tempo considerado, o algoritmo forma um conjunto de produtos candidatos e avalia o impacto de inserir esses produtos na linha de produção de cada máquina. Em seguida, é resolvido um *Problema de Designação* onde o objetivo será selecionar produtos cuja atribuição às máquinas maximizem o valor das utilidades de designação calculadas. Finalmente, usando a solução deste problema, determinam-se quais produtos serão efetivamente incorporados às máquinas.

4.5.1- Horizonte Rolante e Funções Utilidade de Designação

A idéia do *Horizonte Rolante* consiste em realizar uma decomposição temporal do problema, associando cada iteração do método a um horizonte de tempo. Desta forma, na $k^{\text{ésima}}$ iteração, é definido um horizonte de tempo $[t_k, t_k+L]$ onde $t_k=min\{r_j, \forall j \in P_k\}$ sendo P_k o conjunto de produtos disponíveis nesta iteração e L um horizonte arbitrário. Após fixar o intervalo $[t_k, t_k+L]$ é estabelecido o conjunto de produtos candidatos:

$$P_c=\{j/j \in P_k \text{ e } r_j \in [t_k, t_k+L]\}.$$

Apenas produtos cujas informações estejam disponíveis ao problema, no intervalo de tempo $[t_k, t_k+L]$, serão avaliados para uma possível atribuição às máquinas. Esta avaliação é realizada utilizando-se as *Funções Utilidades de Designação* que medem o benefício de inserir um produto candidato na linha de produção de cada uma das três máquinas.

Assim, cada produto $j \in P_c$ terá o valor de sua *Função Utilidade de Designação* calculada através de uma simulação de sua inserção na linha de produção de cada máquina M_i (i=1, 2 e 3). Basicamente podem ser encontradas três situações:

1. Máquinas sem produtos em processamento ou atribuídos.
2. Máquinas processando um produto sem outros produtos atribuídos.
3. Máquinas processando um produto com outros produtos já atribuídos.

Para cada umas destas três situações será estabelecida uma forma distinta de cálculo do valor da *Função Utilidade de Designação* (U_{ij}) do produto j à máquina i. Independente da situação da máquina, U_{ij} avaliará o benefício da atribuição do produto levando em conta sempre o atraso (D_j) ou a antecipação (E_j) na entrega dos produtos e o custo de troca (c_{kj}) relacionado. Assim, U_{ij} será composta pela soma de duas funções encarregadas de medir o impacto da inserção do produto candidato j, respectivamente, sobre o prazo máximo de entrega (d_j) e sobre o valor total do custo de troca.

4.5.1.1- Máquinas sem produtos atribuídos

A função U_{ij} será expressa, nesta situação, pela Equação 4.1 abaixo:

$$U_{ij}=U_{ij}(1)+U_{ij}(2) \tag{4.1}$$

onde $U_{ij}(1)$ avalia o benefício de se atribuir o produto j à máquina i quanto ao atraso (D_j) ou antecipação (E_j) em relação a d_j. A função $U_{ij}(2)$ avalia este benefício em relação ao custo de troca de inserir o produto j após o produto k na linha de produção da máquina i (c_{kj}).

A Figura 4.4 ilustra a situação em que o produto candidato j encontra a máquina M_i, sem que haja produtos atribuídos à mesma.

Figura 4.4 - Máquina sem produtos em processamento ou atribuídos

Caso fosse inserido, o produto j seria imediatamente processado pela máquina e não haveria custo de troca relacionado. As funções $U_{ij}(1)$ e $U_{ij}(2)$ teriam as seguintes expressões:

$$U_{ij}(1)=\begin{cases} \left[V_{\min}+(V_{\max}-V_{\min})\times e^{-2\times\left(D_j/t_0\right)^{b1}}\right] & \text{se } C_j \geq d_j \quad (4.2) \\ V_{\max}+\left[V_{\min}+(V_{\max}-V_{\min})\times e^{-2\times\left(E_j/t_0\right)^{b1}}\right] & \text{se } C_j < d_j \quad (4.3) \end{cases}$$

$$U_{ij}(2)=\beta \tag{4.4}$$

Conforme apresentado nos dois capítulo anteriores, os parâmetros V_{max}, V_{min}, t_0 e b são estabelecidos pelo usuário e fornecem a curva desejada à função U_{ij} (1). A Equação 4.2 avalia o caso em que o processamento do produto (C_j) termina após d_j, ou seja, com atraso. Assim, os valores de $U_{ij}(1) \in [V_{mim}, V_{max}]$ com o valor máximo V_{max} ocorrendo quando não há atraso (D_j=0), decrescendo à medida que o atraso aumenta e sendo mínimo (V_{min}) para atrasos elevados.

A Equação 4.3 avalia o caso em que o processamento do produto (C_j) termina antes de d_j, ou seja, com antecedência no prazo de entrega. Uma das restrições do PDP definido na

seção 4.4 é que o produto seja entregue o mais próximo possível de seu d_j. Porém, será assumido que o atraso deva receber uma penalidade mais severa do que a antecipação. Isto se justifica pela idéia de que um custo extra para armazenar um produto acabado é menor do que o risco de não entregá-lo no prazo.

Isto posto, a Equação 4.3 deixa claro que os valores de $U_{ij}(1) \in [V_{min}+V_{max}, 2V_{max}]$ estão contidos em um intervalo de valores mais elevado do que o estabelecido quando ocorrem atrasos. O valor mínimo $U_{ij}(1)=V_{max}+V_{min}$ ocorrerá para produtos entregues muito antes do prazo ($E_j>>0$), aumentando à medida que $E_j \rightarrow 0$ e atingindo o máximo quando $E_j=0$ com $U_{ij}(1)=2\times V_{max}$.

A Equação 4.4 atribui $U_{ij}(2)=\beta$ pois não há produto em processamento na máquina. Está sendo suposto que não há um custo de troca relacionado ao produto candidato j, pois a máquina i encontra-se vazia. Assim, o benefício de inserir este produto deverá ser elevado porque não há custo de troca, sendo tal valor um parâmetro (β) fornecido pelo usuário.

4.5.1.2 - Máquina processando um produto sem outros atribuídos.

A Figura 4.5 apresenta a situação em que a máquina processa um produto sem que haja outros na linha de produção. O círculo cinza representa o produto em processo.

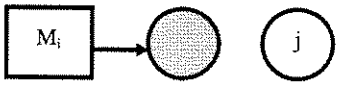


Figura 4.5 – Máquinas processando um produto sem outros atribuídos.

Neste caso, a função U_{ij} também será calculada conforme a Equação 4.1 e terá para $U_{ij}(1)$ a mesma expressão fornecida pelas Equações 4.2 e 4.3. A única diferença na avaliação de E_j ou D_j está no fato do produto candidato j não ser processado imediatamente, devendo aguardar a máquina concluir o produto em processo. Todavia, a inserção de j será imediatamente após o término do produto que esteja sendo processado.

O valor $U_{ij}(2)=\beta$ foi adotado na Equação 4.4 porque não havia produto em processo na máquina i, então o benefício obtido pela atribuição seria máximo em termos de custo de troca. Na situação apresentada pela Figura 4.5, existe um produto em processo e a função $U_{ij}(2)$ terá que avaliar o impacto que a inserção de j provocará sobre o custo de troca.

Um dos objetivos do problema é programar a linha de produção das máquinas com produtos que minimizem o custo total de troca envolvido. Desta forma, quanto menor o custo de troca acrescentado à linha de produção pelo produto candidato, maior será o benefício obtido pela sua inserção. Portanto, o critério de avaliação adotado para o custo de troca é que o valor da *Função Utilidade de Designação* $U_{ij}(2)$ seja o inverso de 1 mais o valor do custo de troca médio da seqüência de produção, após a inserção do produto candidato j.

A Equação 4.5 estabelece que o valor de $U_{ij}(2)$ seja inversamente proporcional a 1 mais o custo de troca c_{kj} entre o produto em processo k e o produto candidato j. Neste caso, existe um único custo de troca relacionando o produto em processo com o candidato.

$$U_{ij}(2)=\frac{\beta}{1+c_{kj}} \tag{4.5}$$

O benefício obtido será máximo (β) quando não há custo de troca relacionado. Para pequenos valores do custo de troca, o benefício obtido estará próximo de β e à medida que aumentar, ele afasta-se deste valor.

4.5.1.3 - Máquina processando um produto com outros atribuídos.

A Figura 4.6 apresenta a situação em que a máquina está processando um produto e existem outros já atribuídos, aguardando o momento de serem processados.

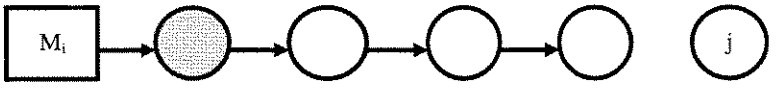


Figura 4.6 – Máquina processando um produto com outros já atribuídos.

Na Figura 4.7, observa-se que a inserção do produto candidato j poderá ocorrer logo após o produto em processo, no final da seqüência ou entre os demais produtos. Desta forma, sua inserção poderá retardar o início do processamento de alguns produtos e alterar o custo de troca total da seqüência.

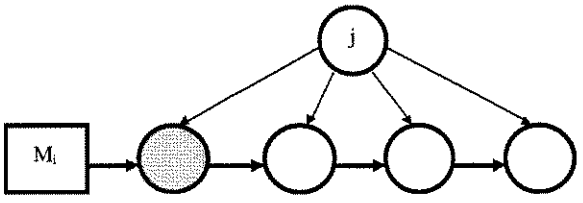


Figura 4.7 - Possíveis inserções do produto candidato j

A Figura 4.8 apresenta alguns possíveis reseqüenciamentos dos produtos após a inserção do produto candidato j.

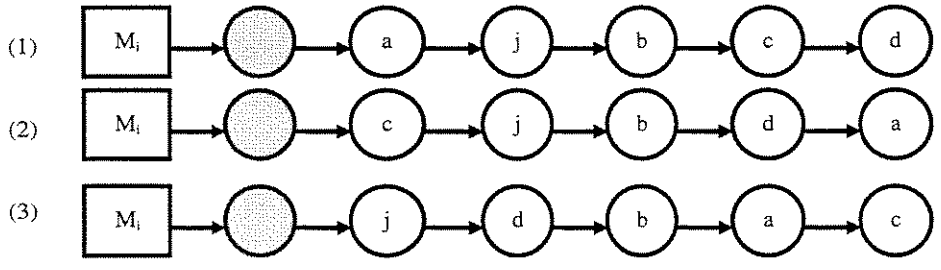


Figura 4.8 - Possíveis reseqüenciamentos após inserir o produto candidato j

Na seqüência (1) o produto j foi inserido e a melhor programação dos produtos, após a inserção de j, não provocou alteração na ordem dos demais produtos. Nas seqüências (2) e (3), estão exemplificadas situações em que a obtenção da melhor programação, após a inserção do produto candidato, exigiu um reseqüênciamento dos demais produtos. Desta forma, com exceção do produto em processo, os demais podem ter sua ordem de processamento alterada para que se obtenha a melhor programação possível.

Neste caso, a função U_{ij} avaliará o impacto da inserção não só no tempo de processamento e custo de troca do produto candidato j, como no tempo de processamento dos demais produtos e no custo de troca de toda a seqüência.

A inserção do produto candidato e os diversos reseqüênciamentos da linha de produção serão obtidos através de métodos heurísticos que serão apresentados na subseção seguinte. No momento, interessa frisar que o valor de U_{ij} será recalculado para cada nova seqüência de produção estabelecida pelas heurísticas. Estas heurísticas terão o objetivo de encontrar a seqüência de produtos com o melhor valor de U_{ij} associado.

Seja $S=p_1, p_2, ..., p_{k-1}, p_j, p_k,..., p_n$; uma seqüência de produção, incluindo o produto candidato j, estabelecida pelas heurísticas para a máquina i. O valor de U_{ij} correspondente a S será:

$$U_{ij}=U_T + U_{ij}(2). \tag{4.6}$$

Desta forma, a Equação 4.6 estabelece a forma de cálculo de U_{ij} quando já existem produtos atribuídos a máquina. O impacto da inserção do produto candidato j sobre sua data de entrega (d_j) e sobre a data de entrega dos demais produtos atribuídos a máquina i será avaliado pela função U_T . O impacto da inserção de j sobre o custo de troca total da seqüência será calculado pela função $U_{ij}(2)$. A função U_T avaliará atrasos e antecipações no processamento de todos os produtos de S usando a expressão

$$U_T = \frac{\sum_{k \in S} U_{ik}(1)}{|S|}. \tag{4.7}$$

Na Equação 4.7, $S=p_1, p_2, ..., p_{k-1}, j, p_k,..., p_n$ é a atual seqüência de produção estabelecida pelas heurísticas já incluindo o produto candidato j. O somatória no numerador retorna o valor da função utilidade $U_{ik}(1)$ de todos os produtos $p_k \in S$. A função $U_{ik}(1)$ é calculada de acordo com os critérios estabelecidos nas Equações 4.2 e 4.3. O valor do somatório é dividido por $|S|$ (cardinalidade de S) que representa o número de produtos existentes na seqüência.

Desta forma, U_T fornece um valor médio das utilidades U_{ik} recalculadas para todos os produtos pertencentes a S. A idéia de usar o valor médio U_T deve-se ao reseqüenciamento realizado pela heurística que provoca alteração no valor $U_{ij}(1)$ do produto candidato e também no valor $U_{ik}(1)$ dos demais produtos $p_k \in S$. Então, neste caso, não interessa saber

apenas o $U_{ij}(1)$ do produto candidato, mas um valor médio da *Função Utilidade de Designação* de todos os produtos da seqüência.

A função $U_{ij}(2)$ avalia o custo total de troca, calculando seu valor para cada seqüência S fornecida pela heurística. Para o cálculo de $U_{ij}(2)$, deve-se obter inicialmente o somatório do custo de troca de todos os produtos em S. Isto é feito calculando-se $\Sigma c_{kl}\ \forall k$ e $\forall l$ com $p_k \in S$ e $p_l \in S$, tal que p_l é o produto imediatamente seguinte a p_k em S. Definido este valor, a função $U_{ij}(2)$ terá a seguinte expressão:

$$U_{ij} = \beta \times \left(1 + \frac{|S| - 1}{\Sigma c_{kj}} \right)^{-1} \tag{4.8}$$

Na Equação 4.8, o valor $|S|$-1 no numerador indica a quantidade de custos de troca presentes na seqüência S. Trata-se de uma extensão da forma de cálculo apresentada pela Equação 4.5 na seção *4.5.1.2*. Na Equação 4.8, o benefício máximo β está multiplicando o inverso do valor 1 mais o custo médio de troca da seqüência porque, neste caso, existem diversos produtos atribuídos à máquina e não apenas o produto candidato *j*.

Desta forma, dependendo de onde o produto candidato *j* for inserido e do reseqüênciamento realizado, diferentes custos médios de troca serão obtidos. O benefício $U_{ij}(2)$ será maior quanto menor for o custo médio de troca calculado.

4.5.2 - Heurísticas de Inserção e de Busca em Vizinhança

Na subseção anterior, o produto candidato *j* encontrava uma máquina processando um produto com outros atribuídos. Foi apresentada a forma de cálculo de U_{ij} para as seqüência obtidas após a inclusão do produto candidato *j*. Também foi comentado que estas seqüências eram fornecidas por uma heurística de inserção e busca em vizinhança. A seguir, será descrita a forma como estas heurísticas fornecem as seqüências.

Uma heurística pode ser entendida como um método capaz de encontrar soluções para determinado problema, porém sem determinar a que distância a solução obtida está da solução ótima. Duas heurísticas, a primeira de Inserção e a segunda de Busca em Vizinhança, foram incorporadas ao algoritmo MORSS permitindo obter um reseqüênciamento da atual linha de produção da máquina *i* (exceto pelo produto em processamento) de forma que ao incluir um produto candidato *j* retorne o melhor valor de U_{ij}.

A heurística de Inserção altera a solução ou seqüência corrente, acrescentando um elemento em diferentes posições e avaliando as mudanças provocadas em cada caso. A heurística de inserção apresentada abaixo acrescenta o produto candidato *j* entre os diversos produtos da seqüência inicial S^0 da máquina *i*, avaliando o benefício obtido para cada inserção realizada. Esta heurística será tratada no decorrer deste trabalho simplesmente por INSERÇÃO e, em linhas gerais, executa os seguintes passos:

Passo 1: Seja j o produto candidato e k o último produto programado na seqüência original S^0 da máquina i. Insira o produto candidato j em S^0, após o produto em processo, obtendo a seqüência S^*. Calcule U^*_{ij} conforme estabelecido pela Equação 4.6.

Passo 2: Insira j depois do produto que o sucedeu na seqüência anterior, obtenha a nova seqüência S e calcule o respectivo U_{ij}.

Passo3: Se $U_{ij}>U^*_{ij}$ faça $S^*=S$ e $U^*_{ij}=U_{ij}$. Se j ainda não foi inserido após k retorne ao Passo 2. Caso contrário, termine a execução e retorne S^* e U^*_{ij}.

Ao final, INSERÇÃO retorna uma seqüência (S^*) com o produto candidato j inserido bem como o valor da *Função Utilidade de Designação* (U^*_{ij}). A heurística de inserção posiciona j na seqüência original de modo que obtenha o melhor valor para U_{ij}. Esta função é calculada pela Equação 4.6, ou seja, somando U_T (Equação 4.7) e $U_{ij}(2)$ (Equação 4.8). Todavia, não foi realizada uma alteração na ordem dos produtos que estavam inicialmente atribuídos à máquina. Tal reseqüênciamento é feito utilizando um método heurístico de Busca em Vizinhança.

A heurística de busca em vizinhança é uma técnica geral composta pelos seguintes elementos (Morton e Pentico, 1993):

(a) Uma solução inicial para o problema de interesse - *semente original.*
(b) Todas as soluções são "vizinhas" da solução original - *vizinhança da semente.*
(c) Um método para selecionar a nova semente (solução melhorada) - *critério de seleção*
(d) Um método para terminar a heurística - *critério de parada.*

Na incorporação deste tipo de heurística ao algoritmo MORSS, a *semente original* será a seqüência de produção de uma máquina, após a inserção do produto candidato $j \in P_c$ pela heurística de inserção explicada anteriormente.

A *vizinhança da semente* será gerada utilizando-se um mecanismo de geração de vizinhanças. Este mecanismo é capaz de fornecer um conjunto de soluções a partir da semente original. Neste trabalho, as sementes são as seqüências de produtos de uma máquina obtidas a partir da seqüência fornecida pela heurística de inserção (considerada a *semente original*). O mecanismo utilizado neste trabalho foi uma adaptação do mecanismo de geração conhecido como "All Pairwise Interchanges" (Evans, 1987) que será tratado por ALLPAIRS.

Em linhas gerais, ALLPAIRS gera uma vizinhança trocando de posição dois produtos da seqüência. Inicialmente, o primeiro produto da seqüência é trocado por todos os seguintes até que realize a troca com o último. Em seguida, o segundo produto é trocado, a partir do terceiro, por todos os seguintes até atingir o último produto da seqüência. O método prossegue desta forma até trocar o penúltimo produto pelo último. Em resumo:

$$S \in \alpha(S^0) \text{ se } S=1, 2,..., k-1, j, k+1,..., j-1, k, j+1,..., n, \text{ para } 1 \leq k < j \leq n; \qquad (4.9)$$

onde S é a seqüência gerada por ALLPAIRS, S^0 é a semente original e $\alpha(S^0)$ sua vizinhança.

Cada seqüência obtida também será avaliada através do cálculo do valor de U_{ij}. O *critério de seleção*, nesta heurística, será a escolha de S com valor de U_{ij} melhor do que U_{ij} calculado inicialmente para S^0. Ocorrendo tal melhora em U_{ij}, a seqüência atual passará a ser a *semente* e assumirá o lugar de S^0 na geração das demais seqüências. O *critério de parada* será o da execução completa de ALLPAIRS sem que nenhuma melhora no valor de U_{ij} tenha ocorrido.

A heurística de busca em vizinhança incorporada ao algoritmo MORSS executará os seguintes passos:

Passo1: Seja $S^0=p_1, p_2,..., p_k, p_j, p_{k+1}, ... p_n$, onde p_i é o $i^{ésimo}$ produto já atribuído à seqüência e j o produto candidato. Calcule U^0_{ij} conforme a Equação 4.6 e faça $S^*=S^0$ e $U^*_{ij}=U^0_{ij}$.

Passo2: Obtenha seqüências $S\in\alpha(S^0)$, conforme estabelecido na Equação 4.9, e calcule U_{ij} pela Equação 4.6. Caso todas as trocas em S^0 já tenham ocorrido, não sendo obtida uma nova seqüência S, execute Passo 4.

Passo3: Se $U_{ij}>U^0_{ij}$ faça $S^0=S$ e $U^0_{ij}=U_{ij}$. Retorne ao Passo 2.

Passo4: Se $S^*\neq S^0$ faça $S^*=S^0$ e $U^*_{ij}=U^0_{ij}$ e retorne ao Passo 2. Caso contrário termine a execução retornando S^* e U^*_{ij}.

Ao final, a heurística acima retorna a melhor seqüência de produção, incluindo o produto candidato j, e o melhor valor para U_{ij}. Desta forma, a execução de INSERÇÃO e ALLPAIRS permitirá a obtenção da melhor programação dos produtos, em termos de *Função Utilidade de Designação*, ao se incluir um produto candidato.

Conforme já mencionado, o algoritmo MORSS simula a inserção de cada produto e avalia o benefício desta inserção através do cálculo de U_{ij}. Nesta adaptação, esta simulação será realizada em dois passos. Primeiro, utiliza-se a heurística de inserção para encontra o melhor valor de U_{ij} avaliando diferentes inserções do produto candidato j na seqüência de produção de cada máquina i. No segundo passo, avalia-se na máquina i um possível reseqüênciamento dos produtos (já incluindo o produto candidato j) que retorne um melhor valor para U_{ij}, através de uma heurística de Busca em Vizinhança. Ao final, todas estas simulações retornam o melhor posicionamento do produto candidato j em cada máquina i e o respectivo valor de U_{ij}.

4.5.3- Problema de Designação

Após simular a inserção de todos os produtos candidatos nas máquinas, calculando as respectivas Utilidades de Designação, o algoritmo resolve um *problema de designação* com a seguinte formulação matemática:

$$\text{Max} \sum_i \sum_j U_{ij} x_{ij} \tag{4.10}$$

s.a.

$$\sum_i x_{ij} \leq 1, \qquad \forall j \in P_c \tag{4.11}$$

$$x_{ij} = \begin{cases} 1, \text{se o produto } j \text{ for atribuído à máquina } i \\ 0 \text{ caso contrário.} \end{cases} \quad \forall i=1, 2, 3 \text{ e } \forall j \in P_c \tag{4.12}$$

A função objetivo (4.10) maximiza as atribuições dos produtos candidatos às máquinas, ou seja, procura obter o maior somatório possível dos U_{ij}'s. A restrição (4.11) impõe que cada produto, se for designado, o seja a uma única máquina.

A restrição (4.12) estabelece os valores possíveis das variáveis de decisão. Logo, se $x_{ij}=1$, o produto j poderá ser designado permanentemente à máquina i. Para $x_{ij}=0$, o produto continuará sendo candidato.

O *Problema de Designação* acima é estabelecido para cada iteração do algoritmo MORSS e sua resolução, nesta adaptação, será realizada pelo pacote computacional CPLEX (CPLEX, 1994). Este programa recebe um arquivo com a formulação matemática apresentada acima, processa os dados e retorna uma solução formada pelos valores de x_{ij}. A solução fornecida pelo CPLEX passará por uma seleção final no algoritmo MORSS que determinará quais produtos efetivamente serão atribuídos às máquinas.

Esta seleção descarta imediatamente produtos com $x_{ij}=0$, realizando a escolha entre aqueles com $x_{ij}=1$. Produtos com $x_{ij}=1$ e $r_j \in (t_k+aL,\ t_k+L]$ também retornarão ao conjunto P_k porque, conforme explicado no capítulo 2, o uso do *Horizonte Rolante* permite avaliar todos os produtos com $r_j \in [t_k,\ t_k+L]$, mas as designações efetivas envolverão apenas produtos cujo $r_j \in [t_k,\ t_k+aL]$. Assim, as decisões definitivas serão tomadas para produtos candidatos que estejam disponíveis para processamento (r_j) próximos ao instante atual (t_k).

Quando ocorre um único $x_{ij}=1$ com $r_j \in [t_k, t_k+aL]$ na máquina i, o produto j será atribuído imediatamente a mesma, passando a fazer parte da linha de produção em execução na máquina. Caso existam outros produtos $p \neq j$ tal que $x_{ip}=1$ e $r_p \in [t_k, t_k+aL]$, será adotado um outro critério para avaliar a atribuição dos produtos, ou seja, calcula-se

$$r_j = \{\ r_p / \ r_p \in [t_k, t_k+aL] \text{ e } x_{ip}=1\} \tag{4.13}$$

A Equação 4.13 estabelece que o produto candidato j com $x_{ij}=1$ e tempo de chegada mais próximo do instante atual, será atribuído de forma definitiva à máquina. Assim, a seqüência de produtos na máquina i será alterada pela inserção do produto candidato j, sendo determinada uma nova linha de produção em execução na máquina.

Para inserir os demais produtos $p \neq j$ com $x_{ip}=1$, seleciona-se um outro produto q utilizando o critério estabelecido na Equação 4.13. O produto selecionado q terá o valor de U_{iq} recalculado considerando a nova seqüência obtida para a máquina i após a inserção definitiva do produto candidato j.

Suponha que U'_{iq} seja o valor recalculado de U_{iq}. Neste caso, se $U_{ip}-U'_{ip} \leq P$ o produto candidato q também será inserido definitivamente na linha de produção da máquina i. O parâmetro P é fornecido pelo usuário e limita o valor do decréscimo em relação a U_{iq}, conforme explicado no capítulo 2. Após a primeira inserção do produto candidato na máquina i, o procedimento descrito para o produto q é repetido em todos os demais produtos p tal que $x_{ip}=1$.

Quando um produto candidato j com $x_{ij}=1$ for definitivamente inserido na máquina i, sua inserção será realizada de acordo com a forma como U_{ij} foi calculado. Este cálculo depende do estado em que se encontra a máquina, logo:

1. Máquina sem produtos atribuídos: O produto j será o primeiro produto a ser processado.
2. Máquina processando um produto sem outros atribuídos: O produto j será processado após o produto em processo.
3. Máquinas processando um produto com outros atribuídos: O produto j e os demais produtos já atribuídos serão programados de acordo com a seqüência que forneceu o melhor valor de U_{ij}. Esta seqüência foi estabelecida pelas heurísticas de Inserção e Busca em Vizinhança.

O modo como o produto candidato j com $x_{ij}=1$ será definitivamente incorporado na seqüência será idêntica à forma como sua *Função Utilidade de Designação* foi calculada, pois o cálculo desta função é feito simulando-se a inserção deste produto na seqüência de produção da máquina.

4.5.4 – Algoritmo MORSS adaptado

Após as explicações de como foram feitas as adaptações no algoritmo MORSS, será apresentado o algoritmo adaptado para o Problema Dinâmico de Programação estabelecido.

Passo 0: Selecione o tamanho do horizonte de tempo L. Selecione uma fração a ($0<a<1$). Ajuste $k=0$ e $t_0=0$. Estabeleça o conjunto de produtos inicialmente disponíveis (P_0).

Passo 1: Ajuste o horizonte de tempo $[t_k, t_k+L]$. Estabeleça o conjunto de produtos candidatos (P_c), ou seja, todos os produtos pertencentes a P_k com $r_j \in [t_k, t_k+L]$.

Passo 2: Calcule o valor das *Funções Utilidade de Designação* U_{ij} para todos os pares (produto candidato j/máquina i), simulando a inserção do produto candidato j na seqüência de produção de cada máquina i. Execute a simulação fazendo:

(i). se a máquina não tem produtos atribuídos, insira o produto candidato j como próximo produto e calcule U_{ij};

(ii). se a máquina está processando um produto sem outros já atribuídos, insira j após o produto em processo e calcule U_{ij};

(iii). se a máquina está processando um produto com outros já atribuídos, execute as heurísticas de Inserção e Busca em Vizinhança para obter o melhor U_{ij}.

Passo 3: Formule e otimize um problema de designação usando os valores obtidos pelas *Funções Utilidades de Designação* como custos. Os resultados obtidos estabelecem uma tentativa de designação no intervalo $[t_k, t_k+L]$.

Passo 4: Retorne as seguintes informações para o conjunto P_k:

(i). todos os produtos candidatos j não designados as máquinas pelo Passo 3,

(ii). todos os produtos candidatos designados pelo Passo 3 cujo $r_j \in (t_k+aL, t_k+L]$,

(iii). todos os produtos candidatos que interagem desfavoravelmente um com o outro.

Passo 5: Desloque o horizonte de tempo, fazendo $t_{k+1}=min\{r_j, \forall j \in P_k\}$. Atualize o tempo de processamento das máquinas para t_{k+1}. Ajuste $k=k+1$ e retorne ao Passo 1.

A cada horizonte de tempo, o algoritmo forma o conjunto de produtos candidatos (Passo 1) e avalia o impacto de inserir produtos nas máquinas, usando as *Funções Utilidades de Desiganação* (Passo 2).

Na adaptação realizada, o Passo 2 passou a incluir um critério de avaliação dos pares (produto candidato j /máquina i) que depende da situação em que a máquina se encontra. Neste passo, as heurísticas poderão ser executadas e as diferentes formas de cálculo adotadas para U_{ij} serão empregadas.

No passo 3 o algoritmo formula e soluciona um problema de designação que, nesta adaptação, será resolvido através do pacote Cplex. Primeiro, a formulação do problema de designação é estabelecida e armazenada em um arquivo. Em seguida, a execução dos demais passos do método é temporariamente interrompida para que o pacote Cplex seja executado. Este programa recebe o arquivo com a formulação do problema de designação e retorna outro arquivo com a solução obtida. Neste momento, a execução do pacote Cplex é encerrada e o Passo 4 é executado.

A solução fornecida pelo pacote Cplex é refinada segundo os critério sintetizados no Passo 4. No critério (iii), está sendo suposto que os produtos possam vir a interagir desfavoravelmente quando ocorre a atribuição de mais de um produto à mesma máquina. Para avaliar isso, o critério adotado é aquele explicado no final da subseção 4.5.3.

Ao final do Passo 4, as decisões de atribuições são definitivamente tomadas e a programação das máquinas com os produtos candidatos atribuídos efetivamente passa a ser executada. No passo 5, ocorre o rolamento do horizonte de tempo para que a próxima iteração seja executada.

A implementação computacional desta adaptação foi realizada utilizando linguagem de programação C, compilador GCC, programa CPLEX, estação Sparc IPX com 64 MB de RAM e sistema operacional SunOS 4.1.1. Esta implementação será tratada no decorrer deste trabalho por rotina MÁQUINAS.

4.6 – Implementação de uma Rotina de Busca e Inserção para Resolução do PDP proposto

A necessidade de comparar os resultados obtidos pela rotina MÁQUINAS com algum outro método, motivou a implementação de um segundo método de resolução. Este método tem por objetivo executar uma busca mais ampla no espaço de soluções possíveis do problema. A Rotina de Busca e Inserção (RBI) procura cumprir esta tarefa, realizando atribuições aleatórias dos produtos às máquinas e otimizando as seqüências estabelecidas utilizando rotinas de Busca em Vizinhança e Inserção.

A RBI estabelece uma programação para todos os produtos cujas informações estejam disponíveis. Inicialmente, as seqüências de produção das máquinas são estabelecidas através de atribuições aleatórias dos produtos. Em seguida, estas seqüências são otimizadas utilizando-se heurísticas de Busca em Vizinhança e Inserção, determinando-se as seqüências de produção que devem ser executadas pelas máquinas.

A cada atualização nos dados do problema, a RBI é novamente executada. O algoritmo executa os seguintes passos:

Passo 0: Recebe os dados iniciais do problema. Ajusta o número de iterações k=1.

Passo 1: Atribua aleatoriamente todos os produtos disponíveis para processamento nas máquinas i=1, 2 e 3.

Passo 2: Seja $S^0=\{S^0_1, S^0_2, S^0_3\}$ o conjunto de seqüências S^0 estabelecidas no Passo 1. Calcule a medida de desempenho inicial U^0_i de cada seqüência S^0_i, i=1, 2 e 3.

Passo 3: Aplicar rotina ALLPAIRS em cada seqüência pertencente a S^0, retornando o conjunto de seqüência $S^1=\{S^1_1, S^1_2, S^1_3\}$ com melhor valor U^1_i para cada seqüência $S^1_i \in S^1$.

Passo 4: Aplicar rotina ALLPAIRS entre as seqüência pertencente a S^1, retornando o conjunto de seqüências $S^2=\{S^2_1, S^2_2, S^2_3\}$ com melhor valor U^2_i para cada seqüência $S^2_i \in S^2$.

Passo 5: Aplicar rotina INSERÇÃO entre as seqüências pertencentes a S^2, retornado conjunto de seqüências $S^3=\{S^3_1, S^3_2, S^3_3\}$ com melhor valor U^3_i para cada seqüência $S^3_i \in S^3$.

Passo 6: Se $U^3_i > U^0_i$ para qualquer i, faça $U^0_i = U^3_i$ para todo i=1, 2 e 3, $S^0=S^1$ e retorne ao Passo 3. Caso contrário, execute Passo 7.

Passo 7: Seja S^3 o conjunto final obtido, onde $S^3_i \in S^3$ são as seqüências finais obtidas que devem ser executadas pelas máquinas i=1, 2 e 3. Atualize os dados do problema.

Se ocorreu atualização nos dados, faça $k=k+1$ e retorne ao Passo1. Caso contrário, Fim.

A atribuição aleatória, adotada no Passo 1, designa produtos às máquinas utilizando um gerador aleatório de 0's e 1's. Uma lista de produtos disponíveis e outra de máquinas existentes são percorridas simultaneamente, atribuindo-se o produto corrente à máquina corrente dependendo do valor gerado para cada associação produto/máquina . Se o gerador aleatório retorna valor 1, a atribuição produto/máquina é realizada. Caso contrário, deve-se continuar percorrendo a lista de produtos e máquinas até que todos os produtos disponíveis estejam em uma das seqüências estabelecidas para as máquinas.

A atribuição aleatória é capaz de gerar diferentes seqüências iniciais, a cada execução do método, para uma mesma instância do PDP proposto. Estas seqüências iniciais são as sementes originais da RBI, pois a partir delas as heurísticas de Busca em Vizinhança e Inserção serão aplicadas. Assim, a vizinhança de busca muda a cada nova execução do método já que as seqüências iniciais são alteradas. Esta mudança na vizinhança de busca faz com que o método possa encontrar diferentes soluções para uma mesma instância do PDP. Por isso, a RBI é executada várias vezes para a mesma instância, permitindo uma procura mais exaustiva pelo espaço de soluções.

Neste trabalho, a heurística RBI foi executada 20 vezes para cada instância avaliada, ou seja, foram obtidas 20 diferentes seqüências iniciais em todos os testes computacionais realizados. A vantagem deste procedimento está na possibilidade de executar o método para um número elevado de diferentes seqüências iniciais, ampliando o espectro de busca e, consequentemente, a diversidade de soluções encontradas.

A desvantagem neste procedimento é que, num contexto dinâmico, o usuário não tem conhecimento prévio de todas as informações do problema. Logo, não dispõe de todas as informações necessárias para resolver 20 vezes todo o problema e avaliar a partir de qual seqüência original a RBI fornece os melhores resultados. Uma alternativa seria realizar as 20 execuções apenas para o conjunto de dados disponíveis. Porém, uma avaliação deste tipo também torna-se proibitiva num contexto dinâmico, pois as atualizações nos dados ocorrerem a todo instante. Desta forma, o usuário não poderia demorar a atualizar as informações, avaliando 20 vezes os dados disponíveis, enquanto novas informações já estão sendo incorporadas ao problema.

No Passo 2, cada seqüência S_i estabelecida pela atribuição aleatória é avaliada através da medida de desempenho U_i. A RBI utiliza o critério de avaliar as seqüências estabelecidas pela atribuição aleatória e pelas heurísticas de Inserção e Busca em Vizinhança. Neste caso, não se avalia o impacto da inserção de determinado produto, mas sim as medidas de desempenho relacionadas com todos os produtos pertencentes à seqüência. Estas medidas de desempenho serão obtidas de forma semelhante ao que foi exposto na seção 4.5 com

$$U_i=U_i(1)+U_i(2). \tag{4.14}$$

O termo $U_i(1)$ avalia o atraso ou a antecipação na data de entrega dos produtos programados na seqüência S_i. O termo $U_i(2)$ avalia o custo de troca dos produtos nesta seqüência. As expressões de $U_i(1)$ e $U_i(2)$ são

$$U_i(1) = \frac{\sum_{j \in S} U_{ij}(1)}{|S_i|} \tag{4.15}$$

$$U_{ij}(1) = \begin{cases} \left[V_{\min} + (V_{\max} - V_{\min}) \times e^{-2 \times (D_j/t_0)^{n1}}\right] & se\ C_j \geq d_j \quad (4.16) \\ V_{\max} + \left[V_{\min} + (V_{\max} - V_{\min}) \times e^{-2 \times (E_j/t_0)^{n1}}\right], & se\ C_j < d_j \quad (4.17) \end{cases}$$

A Equação 4.15 é análoga a Equação 4.7, pois $U_i(1)$ também representa o valor médio das Funções Utilidades de Designação $U_{ij}(1)$ de todos os produtos $j \in S_i$. Nas Equações 4.16 e 4.17 a representação e interpretação de $U_{ij}(1)$ e de seus parâmetros são análogas às realizadas para as Equações 4.2 e 4.3. Desta forma, o valor médio $U_i(1)$ representará o benefício obtido pela atual programação dos produtos em S_i, referente aos atrasos ou antecipações no processamento dos produtos.

A obtenção do valor de $U_i(2)$ está vinculada às três situações possíveis em que uma máquina pode ser encontrada:

$$U_{ij}(2) = \beta \tag{4.18}$$

$$U_{ij}(2) = \frac{\beta}{1 + c_{kj}} \tag{4.19}$$

$$U_{ij} = \beta \times \left(1 + \frac{|S| - 1}{\sum c_{kj}}\right)^{-1} \tag{4.20}$$

A interpretação da Equação 4.18 é análoga a fornecida pela Equação 4.4. Logo, quando não há produto em processo na máquina, não há custo de troca relacionado e a medida de desempenho será máxima (β). A Interpretação da Equação 4.19 é análoga à Equação 4.5, pois refere-se à situação em que há um produto em processo sem outros atribuídos. Por fim, a Equação 4.20 fornece o cálculo da medida de desempenho quando há um produto em processo com outros produtos já atribuídos à máquina. Neste caso, interpreta-se a equação e seus parâmetros da mesma forma que foram interpretados na Equação 4.8.

Estas medidas de desempenho procuram fazer com que a RBI avalie e selecione as seqüências conforme os critérios adotados na rotina MÁQUINAS, onde a inserção dos produtos candidatos e, por conseguinte, as seqüências são avaliadas através das Funções Utilidades de Designação.

No Passo 3, aplica-se em cada uma das seqüências $S^0=\{S^0_1, S^0_2, S^0_3\}$ a rotina ALLPAIRS. Conforme foi explicado, esta heurística realiza uma busca em vizinhança que concentra-se na troca de produtos dentro de uma mesma seqüência. Cada troca resulta numa nova programação dos produtos cujo desempenho será avaliado por U_i. Em linhas gerais, pode ser entendida como:

$$S^1_i \in \alpha(S^0_i) \text{ se } S^1_i = p_1, p_2, \ldots, p_{k-1}, p_j, p_{k+1}, \ldots, p_{j-1}, p_k, p_{j+1}, \ldots, p_n, \text{ para } 1 \leq k < j \leq n \qquad (4.21)$$

onde S^1_i é uma seqüência gerada por ALLPAIRS para a máquina i, S^0_i é a semente original e $\alpha(S^0_i)$ sua vizinhança. Caso determinada seqüência S^1_i obtida apresente um valor U^1_i melhor que o da semente original, passará a ser a seqüência corrente gerando a vizinhança $\alpha(S^1_i)$.

Esta heurística termina quando todas as trocas permitidas forem realizadas nas seqüências, sem que ocorra melhoria em relação à medida de desempenho da seqüência corrente. Desta forma, a rotina ALLPAIRS retorna o conjunto de seqüências $S^1=\{S^1_1, S^1_2, S^1_3\}$ e as respectivas medidas de desempenho U^1_1, U^1_2 e U^1_3.

No Passo 4, a idéia da rotina ALLPAIRS é aplicada para percorrer uma nova vizinhança de soluções, onde novas seqüências são obtidas trocando-se os produtos de duas seqüências diferentes. Sejam S^1_a e S^1_b as seqüências finais estabelecidas no Passo 3 para as máquinas a e b. Estas seqüências são alteradas fazendo:

$$S^2_a \in \alpha(S^1_a) \text{ se } S^2_a = p_1, p_2, \ldots, p_{k-1}, p_j, p_{k+1}, \ldots, p_n \text{ para } 1 \leq k \leq n; \qquad (4.22)$$

$$S^2_b \in \alpha(S^1_b) \text{ se } S^2_b = q_1, q_2, \ldots, q_{j-1}, q_k, q_{j+1}, \ldots, q_m \text{ para } 1 \leq j \leq m\,. \qquad (4.23)$$

onde S^2_a e S^2_b são as seqüências obtidas pela troca realizada entre as máquinas a e b, respectivamente. As seqüências S^1_a e S^1_b são as sementes originais deste procedimento e foram fornecidas pelo Passo 1. Os termos $\alpha(S^1_a)$ e $\alpha(S^1_b)$ indicam as respectivas vizinhanças destas seqüências que estarão sendo percorridas.

As medidas de desempenho U^2_a e U^2_b são calculadas para ambas as seqüências S^2_a e S^2_b. Ocorrendo uma melhoria no valor de U^2_a e U^2_b em relação a medida de desempenho das sementes, estas seqüências passarão a ser geradoras das vizinhanças de busca substituindo S^1_a e S^1_b. O procedimento termina quando todas as trocas, entre as seqüências de todas as máquinas, forem realizadas sem que ocorra melhoria nas medidas de desempenho U^2_1, U^2_2 e U^2_3 obtidas.

No Passo 5, utiliza-se uma heurística que será tratada por INSERÇÃO. Este método irá alterar duas seqüências, inicialmente obtidas no Passo 4, retirando o produto de uma e inserindo na outra em diferentes posições:

$$S^3_a \in \alpha(S^2_a) \text{ se } S^3_a = p_1, p_2, \ldots, p_{k-1}, p_{k+1}, \ldots, p_n, \text{ para } 1 \leq k \leq n \qquad (4.24)$$

$$S^3_b \in \alpha(S^2_b) \text{ se } S^3_b = q_1, q_2, \ldots, q_{j-1}, p_k, q_j, q_{j+1}, \ldots, q_n, \text{ para } 1 \leq j \leq m \qquad (4.25)$$

Trata-se, portanto, de uma extensão da heurística de Inserção apresentada na seção 4.5. Cada produto da seqüência S^3_a será retirado e a nova seqüência obtida S^3_a terá sua medida de desempenho calculada U^3_a. Caso não ocorra melhoria no valor de U^3_a em relação a U^2_a, o produto é inserido novamente na seqüência e retira-se o produto seguinte para novo cálculo de U^3_a. Havendo melhoria em U^3_a, o produto retirado de S^3_a passará a ser inserido em diferentes posições de S^2_b. Ao final, ocorrendo $U^3_a > U^2_a$ e $U^3_b > U^2_b$, as seqüências alteradas S^3_a e S^3_b tornam-se as novas sementes e a procura por melhores soluções passará a ser realizada a partir delas.

No Passo 6, caso pelo menos uma das seqüências finais obtidas apresente melhoria no valor final U^3_i em relação a U^0_i, significa que pelo menos uma máquina i obteve melhor programação da produção (S^3_i) após aplicar-se ALLPAIRS e INSERÇÃO. Desta forma, executa-se novamente os Passos 3 a 6 para $S_0=S_3$. Este procedimento será repetido até que não ocorra melhoria nas medidas de desempenho, ou seja, $U^3_i=U^0_i$ para todo i=1, 2 e 3.

No Passo 7, verifica-se a ocorrência de atualização nos dados do problema, ou seja, se novos produtos chegam ao problema retorna-se ao Passo 1 para nova atribuição aleatória envolvendo estes produtos. Caso não ocorra atualização, as seqüências obtidas tornam-se definitivas e a rotina é encerrada.

4.7 - Resultados Computacionais

Nesta seção, será apresentado e analisado o desempenho da adaptação do algoritmo MORSS ao solucionar diversas instâncias do Problema Dinâmico de Programação (PDP).

4.7.1 – Gerador de Instâncias GERAINST

Os dados do PDP foram obtidos através de um gerador de dados aleatório que segue a mesma estrutura daquele apresentado no capítulo 3. Esta rotina de geração de instâncias (GERAINST) tem a responsabilidade de estabelecer, para cada instância gerada, o cenário inicial (Cenário_0.doc) e os N subseqüentes cenários (Cenário_1.doc, Cenário_2.doc, ... , Cenário_N.doc) obtidos pelas atualizações nos dados disponíveis.

A independência entre o processo de geração dos dados e a execução da rotina MÁQUINAS também foi mantida. GERAINST inicialmente cria todos os arquivos de dados para, em seguida, a rotina MÁQUINAS receber e processar as atualizações uma por vez. Isto evitará que as informações contidas no arquivo Cenário_K.doc sejam conhecidas antes dos dados contidos em Cenário_K-1.doc. O objetivo deste procedimento é garantir a execução da rotina MÁQUINAS como se esta recebesse as informações em tempo real.

A Figura 4.9 mostra que a estrutura de geração de instâncias utilizada difere daquela apresentada no capítulo anterior apenas quanto aos tipos de parâmetros fornecidos pelo usuário:

- **NP_0**: Número inicial de produtos que estará armazenado no arquivo Cenário_0.
- **NP_MAX, NP_MIN**: Número máximo e mínimo de produtos que poderão ser gerados aleatoriamente em cada arquivo.
- **CT_MAX, CT_MIN**: Valor máximo e mínimo que poderá ser gerado aleatoriamente para o custo de troca dos produtos.
- **TP_MAX, TP_MIN**: Valor máximo e mínimo que poderá ser gerado aleatoriamente para o tempo de processamento dos produtos pelas máquinas.

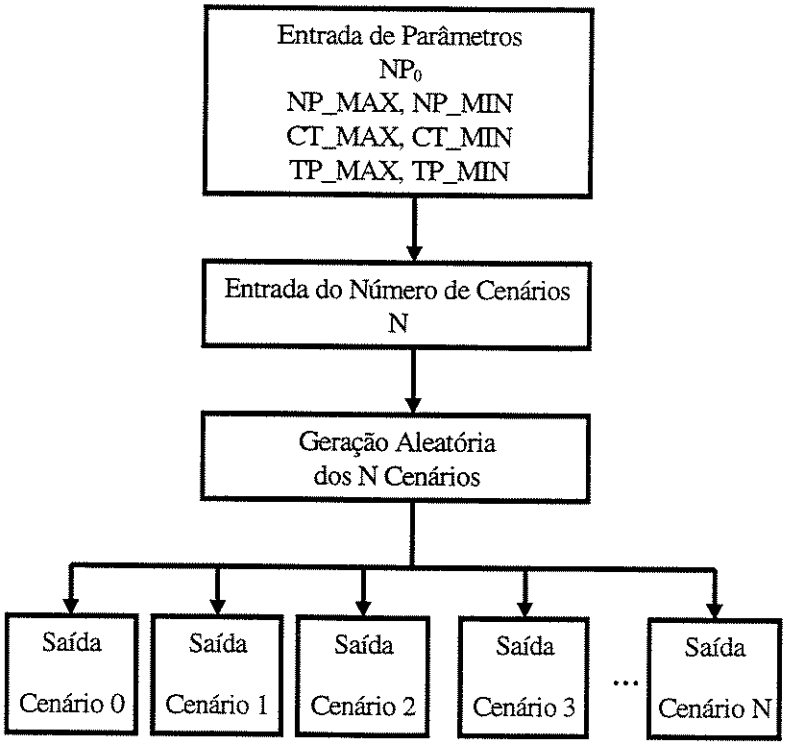


Figura 4.9 - Rotina GERAINST

Cada arquivo de dados contém o número de produtos a serem acrescentados ao problema, os valores dos custos de troca destes novos produtos, o tempo de processamento em cada máquina, o momento em que tornam-se disponíveis para serem processados e o prazo máximo para que sejam processados.

A Figura 4.10 apresenta o processamento dos arquivos de dados fornecidos por GERAINST nas N primeiras iterações da rotina MÁQUINAS. Observa-se novamente que o critério de atualização dos dados na rotina é o mesmo daquele apresentado no capítulo anterior.

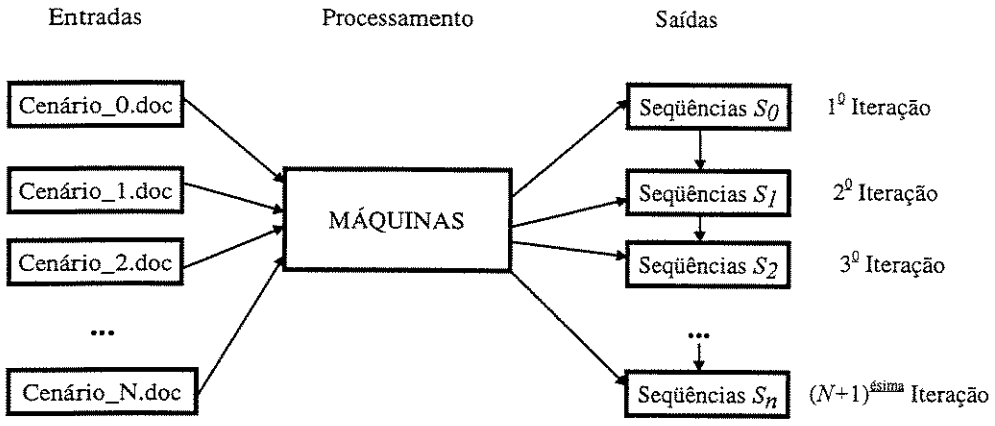


Figura 4.10 - Entrada e saída de dados na rotina MÁQUINAS

Na primeira iteração da rotina MÁQUINAS, o arquivo de dados iniciais (Cenário_0.doc) é processado e o conjunto S_0 que contém as seqüências iniciais das três máquinas do problema é retornado. A primeira alteração é estabelecida pelas informações contidas em Cenário_1.doc que ao serem processadas pela rotina MÁQUINAS, retornam o conjunto de seqüências de produção S_1.

A seta ligando S_0 a S_1 indica que as últimas foram construídas a partir das primeiras, ou seja, trata-se de mudanças realizadas em S_0 para atender as alterações trazidas pelo arquivo Cenário_1.doc. Este procedimento é repetido até o $(N+1)^{ésimo}$ arquivo de dados ser processado, quando deixam de ocorrer alterações no problema. Todas as iterações do método, a partir da iteração $N+1$, passam a trabalhar com um conjunto de dados que deixa de sofrer alterações já que não há mais arquivos de dados para serem processados.

4.7.2 - Instâncias avaliadas

Serão definidas a seguir cinco instâncias do PDP cujos dados foram gerados pela rotina GERAINST. Estas instâncias serão chamadas *I1*, *I2*, *I3*, *I4* e *I5* e os resultados obtidos pela rotina MÁQUINAS serão apresentados e analisados no decorrer deste capítulo.

A Tabela 4.1 lista as principais características que diferenciam as cinco instâncias consideradas. A linha *NTP* apresenta o Número Total de Produtos em cada instância. Nesta linha, ocorrem 46 produtos em *I1*, 80 produtos em *I2* e *I3* e 152 produtos em *I4* e *I5*. Deve-se destacar que, em um contexto dinâmico, o valor do *NTP* representa o número de produtos após todas as atualizações acontecerem. Numa situação real, as atualizações podem ocorrer todo o tempo e não seria possível falar em *NTP*, mas numa quantidade total até que ocorra a próxima alteração. O *NTP* representa um valor acumulado de produtos que passam pelo sistema, pois durante a resolução do problema dinâmico não há todo instante *NTP* produtos demandando processamento.

Tabela 4.1: Aspectos numéricos das Instâncias

Instâncias	*I1*	*I2*	*I3*	*I4*	*I5*
NTP	46	80	80	152	152
NA	20	20	10	20	10
NTP/NA	2,3	4	8	7,6	15,2
CMT	58,21	59,02	58,18	59,18	59,19
CV	0,5	0,49	0,49	0,48	0,48
DP	29,17	29,07	29,12	28,60	28,58
TPM	9,62	9,46	9,13	9,39	9,44
Prazo Médio	63,13	76,88	77,06	110,37	101

A linha *NA* contém o Número de Atualizações ocorridas em cada instância. Assim, acontecem 20 atualizações em *I1*, *I2*, *I4* e 10 nas instâncias *I3* e *I5*. Desta forma, um total de 20 arquivos contendo atualizações de dados foram fornecidos à rotina MÁQUINAS enquanto esta resolvia as instâncias *I1*, *I2* e *I4* e 10 arquivos em *I3* e *I5*. Esses diferentes valores para *NA* significam que existe uma demanda maior de produtos por máquina em algumas instâncias do que em outras.

As diferentes demandas por produtos em cada instância são melhor retratadas pelo valor médio do Número Total de Produtos (*NTP*) que chegam a cada atualização (*NA*), ou seja, pelo quociente *NTP/NA*. Os valores de *NTP/NA* significam, por exemplo, que em *I2* há um volume maior de produtos chegando (em média 8 a cada atualização) do que em *I3* (em média 4 a cada atualização). Neste caso, a rotina MÁQUINA viu-se diante de duas instâncias do PDP com um mesmo número total de produtos (80 produtos) passando pelo sistema, porém em *I2* eles chegam numa freqüência menor do que em *I3*.

Os valores apresentados em *CMT*, *CV* e *DP* são medidas estatísticas relacionadas ao custo de troca dos produtos. O termo *CMT* é o valor do Custo Médio de Troca envolvendo todos os produtos das instâncias, *CV* é o coeficiente de variação e *DP* é o desvio-padrão destes custos. Por exemplo, a instância *I3* tem um *CMT* de 58,18 com um coeficiente de variação de 0,49 e uma dispersão de 29,12. O *CV* mede o grau de heterogeneidade dos valores dos custos de troca e todas as instâncias apresentam $CV \leq 0{,}5$, logo, observa-se uma tendência a homogeneidade dos valores gerados para os custos de troca.

O fator tempo será avaliado em Unidade de Tempo (*UT*), ou seja, uma grandeza positiva sem vínculo a minutos, horas, dias, etc. Isto posto, encontram-se na penúltima linha os Tempos de Processamento Médio (*TPM*) dos produtos nas máquinas. Observando-se os valores do *TPM* na Tabela 4.1, nota-se que os produtos levam em média cerca de 9 *UT* para serem processados.

Na última linha da Tabela 4.1 está o *Prazo Médio* para entrega dos produtos, ou seja, o valor médio computado do instante em que os produtos tornam-se disponíveis (r_j) até a data limite para sua entrega (d_j). Os valores do *Prazo Médio* não variam muito para instâncias com mesmo *NTP*. Produtos que chegam em *I2* e *I3* têm em média cerca de 77 *UT* de prazo

para serem processados e entregues. Nas instâncias *I4* e *I5*, as máquinas devem realizar o processamento dos produtos em até 110 *UT* e 101 *UT*, respectivamente.

Estes valores do *Prazo Médio* indicam que não há uma folga maior nos prazos, apesar do maior volume de produtos chegando ao mesmo tempo. Por exemplo, em *I3* há um número maior de produtos chegando a cada atualização (*NTP/NA*=8) que em *I2* (*NTP/NA*=4) com um *Prazo Médio* de processamento próximos nas duas instâncias. O *Prazo Médio* de *I4* e *I5* também apresentam-se relativamente próximos, apesar de ocorrer uma demanda bem maior de produção em *I5* (*NTP/NA*=15.2) do que em *I4* (*NTP/NA*=7.6).

4.7.3 – Resultados da rotina MÁQUINAS para diferentes valores de L.

Neste capítulo, a rotina MÁQUINAS foi executada usando diferentes valores de L com o objetivo de observar para quais valores o método apresentaria melhor desempenho. Este desempenho foi avaliado em relação ao custo de troca dos produtos e em relação ao número de produtos cujo tempo final de processamento ocorreu dentro do prazo previsto.

A rotina foi executada inicialmente usando L=1 unidade de tempo (UT), desta forma, os conjuntos de produtos candidatos formados e as decisões de atribuição às máquinas envolveram produtos com tempo de chegada nos intervalos $[t_k, t_k+1]$, com o valor de t_k definido a cada $k^{\text{ésima}}$ iteração de MÁQUINAS. Em seguida, a rotina foi novamente executada fazendo L=2UT e trabalhando com intervalos de $[t_k, t_k+2]$ em todas as $k^{\text{ésima}}$ iterações. Este procedimento foi repetido até L=40UT. Ao término da execução, para cada

L considerado, o desempenho da rotina foi avaliado em termos de porcentagem de Produtos Entregues no Prazo (*%PEP*) e Custo Médio de Troca (*CMT*). Estas medidas de desempenho foram calculadas da seguinte forma

$$\%PEP = \frac{|P_d|}{|P_p|} \text{x} 100 \tag{4.26}$$

$$CMT = \frac{\sum_{k,j \in P_p} c_{kj}}{|P_p| - 3} \tag{4.27}$$

Nas Equações 4.26 e 4.27 acima, P_p representa o conjunto de todos os produtos processados ao final da execução da rotina MÁQUINAS. O conjunto P_d é um subconjunto de P_p formado pelos produtos que foram entregues antes ou na data de entrega. A Equação 4.26 representa a porcentagem de produtos cujo processamento foi concluído no máximo até a respectiva data de entrega.

A Equação 4.27 apresenta o valor médio do custo de troca de todos os produtos processados. No numerador está a soma do valor dos custos de troca. O denominador contém o número total dos custos de troca utilizados nas seqüências estabelecidas para as três máquinas. Convém esclarecer que, para determinado número de produtos processados

em cada máquina, o número de custo de troca utilizados é igual ao número de produtos processados menos 1.

Como existem três máquinas, o número de custos de troca utilizados em todas as seqüências de produção é igual ao número de produtos processados menos três. Ao final da execução da rotina MÁQUINAS, para valores de L variando de 1 até $40UT$, será possível identificar qual valor de L retornou a maior *%PEP* e o menor *CMT*.

4.7.3.1 – Resultados Computacionais para Instâncias I1, I2 e I3

A Figura 4.11 apresenta as mudanças na *%PEP* e no *CMT* a medida que a rotina MÁQUINAS é executada para diferentes valores de L. Cada curva representa o desempenho associado às instâncias *I1*, *I2* e *I3*. Os símbolos "*" e a letra maiúscula indicam a localização dos pontos da curva onde ocorrem os melhores resultados. Os resultados de *I4* e *I5* serão apresentados posteriormente, devido a maior dimensão destas instâncias e para permitir melhor visualização gráfica dos resultados obtidos.

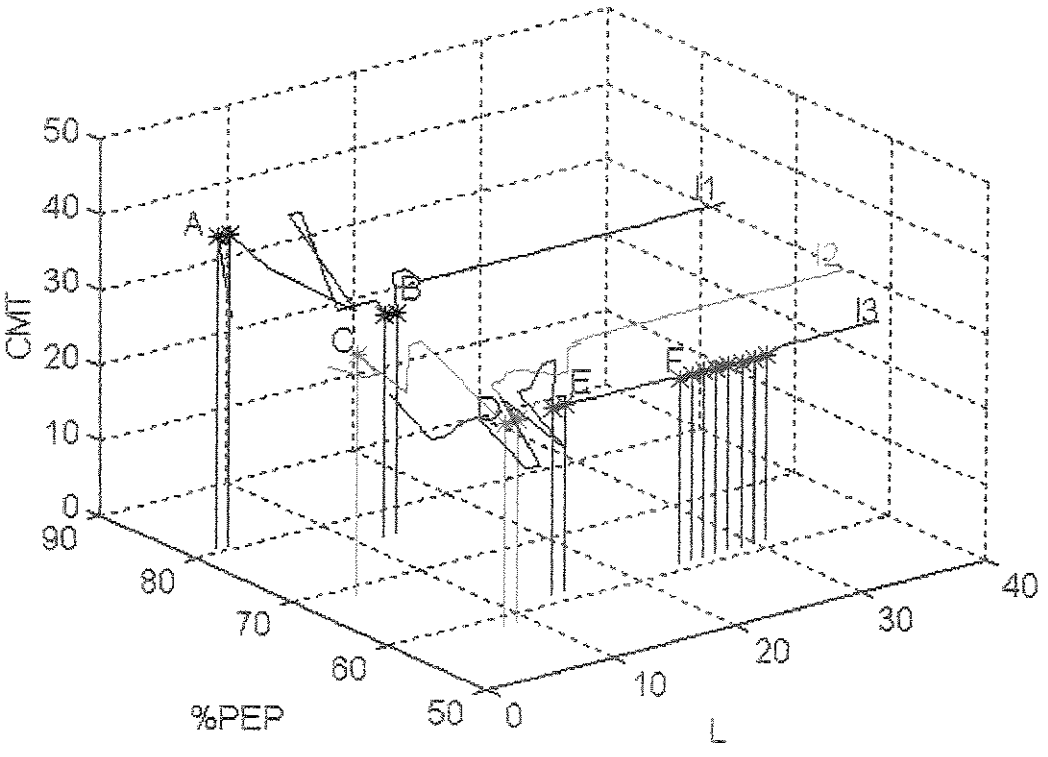


Figura 4.11 – Curvas de desempenho das instâncias *I1*, *I2* e *I3*

As letras A, C e E na Figura 4.11 indicam os locais onde ocorrem os melhores resultados das *%PEP*, respectivamente, em *I1*, *I2* e *I3*. As letras B, D e F indicam o mesmo em relação aos melhores *CMT* de cada instância. Observa-se a existência de melhores resultados em mais de um ponto na mesma curva. Por exemplo, em A ocorre a maior

%PEP para dois valores distintos de *L*. A letra F na curva de *I3* mostra que o menor *CMT* ocorre para sucessivos valores de *L*.

Os pontos com melhores resultados de *%PEP* e *CMT* revelam a existência de um valor *L* capaz de incluir no conjunto de produtos candidatos P_c, a cada iteração da rotina, produtos cujas decisões de atribuição às máquinas são tomadas da melhor forma possível em termos destas duas medidas de desempenho. Desta forma, a escolha de um tamanho adequado de *L* também é relevante nesta adaptação do algoritmo MORSS para obtenção das melhores medidas de desempenho.

Após um certo tamanho de *L*, alterações no seu valor passam a ser irrelevantes porque deixam de afetar as atribuições realizadas em cada horizonte estabelecido. Por exemplo, a partir de *L*=20 em *I2* as atribuições dos produtos às máquinas passam a repetir-se ou, apesar de diferentes, simplesmente não repercutem mais nos valores obtidos para *%PEP* e *CMT* que passam a ser constantes. Este comportamento repetiu-se em todas as instâncias avaliadas, conforme pode ser constatado pelas curvas de desempenho na Figura 4.11.

A curva com os resultados obtidos para *I1* apresenta-se mais elevada em relação a *I2* e *I3*. Isto indica que os valores do *CMT* são melhores em *I2* e *I3* onde o método concentra valores entre 20 e 30, enquanto em *I1* apresenta valores para o custo entre 30 e 40. Por outro lado, os valores da *%PEP* obtidos usando diferentes *L*´s são melhores em *I1* e estão concentrados numa faixa entre 70 e 80% de produtos entregues no prazo. Esta medida de desempenho em *I2* e *I3* não ultrapassa 70%.

A instância *I1* apresenta um *NTP/NA*=2,3, ou seja, seus 46 produtos chegam em média 2 produtos em cada uma das 20 atualizações ocorridas. Estes produtos dispõem de um *Prazo Médio* de 63,13 *UT* para serem processados. As instâncias *I2* e *I3* apresentam, respectivamente, *NTP/NA*=4 e *NTP/NA*=8 com *Prazo Médio* de 76,88 e 77,06 (Tabela 4.1). Estes valores justificam uma *%PEP* mais baixa em *I2* e *I3* do que em *I1*, pois a rotina MÁQUINAS necessita gerenciar um volume médio bem maior de novos produtos chegando a cada atualização nos dados de *I2* e *I3*.

Além disso, observa-se que a maioria dos valores da *%PEP* em *I3* são menores do que em *I2*. Este comportamento é esperado devido ao fato de que 80 produtos deixam de chegar em 20 atualizações em *I2* e passam a chegar em 10 atualizações em *I3*. Isto ocorre sem uma variação considerável no prazo para processamento dos produtos, conforme pode ser constatado pelo valor do *Prazo Médio*=76,88 em *I2* e *Prazo Médio*=77,06 em *I3*.

O CMT mais baixo em *I2* e *I3* explica-se justamente por estas instâncias apresentarem maior volume de novos produtos a cada atualização. Um maior número de produtos chegando a cada iteração fornece mais opções para seleção de produtos com menores custos de troca. Por outro lado, quando chegam poucos produtos uma alteração posterior na seqüência que ocasione uma melhora no custo de troca torna-se mais difícil. Por exemplo, poucos produtos chegando ao sistema de produção a cada iteração pode significar uma demora na chegada de determinado produto *j*. Isto poderá fazer com que produtos da

seqüência de produção em execução, depois dos quais a inserção de *j* apresentaria custo de troca menor, já tenham sido processados pelas máquinas.

A Tabela 4.2 apresenta os valores relativos aos melhores resultados obtidos em cada instância. A coluna Ponto refere-se aos locais nas curvas da Figura 4.11 onde estão assinalados os pontos com os melhores resultados. As letras A, C e E indicam a localização na Figura 4.11 dos pontos com a maior *%PEP* e B, D e F os pontos com o menor *CMT*. Na coluna seguinte está a tripla ordenada com os valores correspondentes a esses pontos. A coluna Δ*%PEP* apresenta a diferença entre a melhor *%PEP* e a *%PEP* obtida no ponto com menor *CMT*. A coluna Δ*CMT* apresenta a diferença entre o valor do *CMT* obtido no ponto com a melhor *%PEP* e o ponto com o menor *CMT*.

Tabela 4.2 – Melhores resultados obtidos para *I1*, *I2* e *I3*

Inst.	*Ponto*	*(%**PEP**;CMT;L)*	*Ponto*	*(%PEP;**CMT**;L)*	*Δ%PEP*	*ΔCMT*
I1	A	(**80,76**; 41,39;2)	B	(76,23;**29,23**;12)	4,53	12,16
		(**80,76**;41,39;3)		(76,23;**29,23**;13)		
I2	C	(**68,63**; 31,77;4)	D	(58,68; **26,84**;8)	9,95	4,93
				(58,68;**26,84**;9)		
I3	E	(**61,57**;24,62;14)	F	(61,28; **24,44**;24)	0,29	0,18
		(**61,57**;24,62;15)		...		
				(61,28;**24,44**;31)		

Por exemplo, na instância *I1* o melhor valor da *%PEP* ocorre em pontos com (***%PEP**;CMT;L*)=(**80,76**;41,39;2) e (***%PEP**;CMT;L*)=(**80,76**;41,39;3). A localização destes dois pontos está assinalada pela letra A na Figura 4.11 e suas coordenadas diferem apenas quanto ao valor de *L*.

Os menores *CMT* de *I1* estão indicados pela letra B na Figura 4.11 e ocorrem em pontos com as coordenadas (76,23;**29,23**;12) e (76,23;**29,23**;13). As variações relativas a *%PEP* e *CMT* são Δ*%PEP*=4,53 e Δ*CMT*=12,16. Desta forma, ocorre uma queda de 4,53% no valor da maior *%PEP* obtida pelo método no ponto (**80,76**;41,39;2) para que o menor *CMT* fosse obtido em (**76,23**;29,23;12), ou seja, houve uma redução no custo de 12,16 em relação ao ponto com melhor *%PEP*.

Pelos valores obtidos nas colunas Δ*%PEP* e Δ*CMT*, observa-se que a melhoria no *CMT* ocasiona uma queda na *%PEP* ou vice-versa. Nos diversos valores de *L* para os quais a rotina MÁQUINAS foi executada, não houve a ocorrência em um mesmo ponto da maior *%PEP* e do menor *CMT*. A princípio, estes dois critérios são concorrentes.

O ponto com melhor compromisso entre *%PEP* e *CMT* foi obtido em *I3*, onde o menor resultado do *CMT* (24,44) foi obtido com queda de apenas 0,29% no melhor valor da *%PEP*. Também observa-se que a redução no *CMT* foi de apenas 0,18. Isto demonstra que, tanto nos pontos assinalados pela letra E quanto naqueles assinalados pela letra F na Tabela 4.2, já havia um bom resultado em termos de *CMT* e *%PEP*.

4.7.3.2 – Resultados Computacionais para instâncias I4 e I5

A Figura 4.12 apresenta as mudanças na *%PEP* e *CMT* quando a rotina MÁQUINAS é executada com diferentes valores de *L* nas instâncias *I4* e *I5*. As letras A e B indicam a localização da maior *%PEP* em *I4*. A letra D indica a maior *%PEP* em *I5*. As letras C e E localizam o menor *CMT*, respectivamente, em *I4* e *I5*.

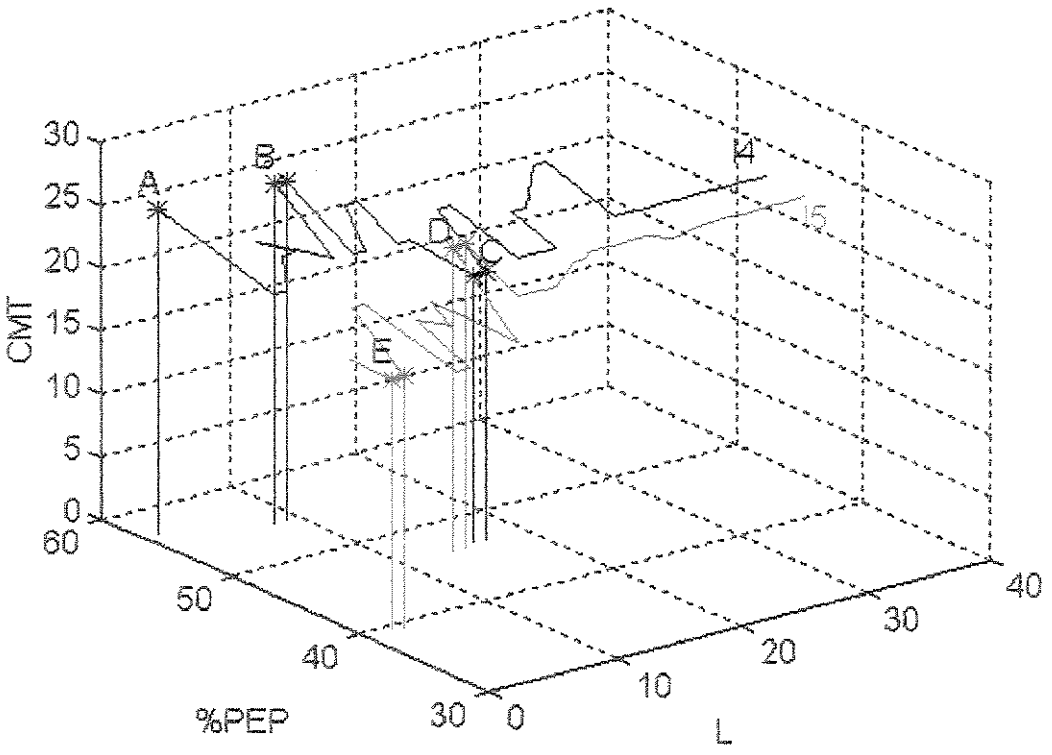


Figura 4.12 - Curvas de desempenho da instâncias *I4* e *I5*

Nestas instâncias, a rotina MÁQUINAS encontra *CMT* próximos para *I4* e *I5* ao ser executada para diferentes valores de *L*. Isto pode ser constatado na Figura 4.12 pela altura semelhante das curvas nestas instâncias, onde a maioria dos *CMT* obtidos estão entre 20 e 25. Porém, os valores obtidos para *%PEP* mostram-se dissociados com *I4* apresentando a maioria dos seus valores entre 50 e 60%, enquanto em *I5* a *%PEP* concentra-se entre 40 e 50%.

Estes valores da *%PEP* em *I4* e *I5* mostram de forma mais contundente a dificuldade encontrada pelo método em manter uma *%PEP* alta quando uma mesma quantidade de produtos (*NTP*=152) deixa de chegar em 20 atualizações (instância *I4*) e passa a chegar em 10 atualizações (instância *I5*).

O número de produtos chegando em média a cada atualização passa de *NTP/NA*=7,6 em *I4* para *NTP/NA*=15,2 em *I5*. Desta forma, existe maior dificuldade em *I5* do que em *I4* para programar dentro dos prazos estabelecidos a produção dos novos produtos que chegam.

A Tabela 4.3 apresenta os valores relativos aos melhores resultados obtidos em *I4* e *I5*. A letra A na coluna *Ponto* refere-se aos pontos na Figura 4.12 onde a instância *I4* apresenta a maior *%PEP*=56,81%, porém deixando de processar 6 produtos. A letra B na Figura 4.12 indica os pontos onde a instância *I4* apresenta maior *%PEP* processando todos os produtos.

Tabela 4.3 – Melhores resultados obtidos para *I4* e *I5*

Inst.	*Ponto*	*(%**PEP**;CMT;L)*	*Ponto*	*(%PEP;**CMT**;L)*	*Δ%PEP*	*ΔCMT*
I4	A	(**56,81**;25,24;1)	-	-	-	-
I4	B	(**54,65**;26,96;8)	C	(46,74;**21,29**;16)	7,91	5,67
		(**54,65**; 26,96;9)		(46,74;**21,29**;17)		
I5	D	(**46,41**;24,14;14)	E	(39,48;**19,94**;2)	6,93	4,2
		(**46,41**;24,14;15)		(39,48;**19,94**;3)		

Portanto, no ponto A, um total de 56,81% dos produtos foram processados dentro do prazo estabelecido, porém 6 produtos deixaram de ser processados. Deve-se lembrar que a *%PEP* é calculada sobre a quantidade de produtos processados ($|P_p|$) e, desta forma, os produtos que não foram processados deixam de entrar no cálculo da *%PEP* (veja Equação 4.26).

A melhor *%PEP* cai para 54,65% quando todos os produtos são processados, conforme valores apresentados pela letra B na Tabela 4.3 e assinalados pela mesma letra na Figura 4.12. Isto significou uma queda de 2,16% no resultado anterior, porém ressalta-se que neste caso foi realizado o processamento de todos os produtos.

A melhor *%PEP*=46,41% em *I5*, assinalada pela letra D na Figura 4.12, foi obtida com o processamento de todos os produtos e ocorre em dois pontos que diferem apenas quanto ao valor de *L*. Comparando-se os valores das melhores *%PEP* em *I4* com o valor obtido em *I5*, observa-se que a diferença foi bastante acentuada.

Os melhores *CMT* em *I4* e *I5* foram encontrados com o processamento de todos os produtos e ocorrem em pontos cujas coordenadas diferem apenas quanto ao valor de *L*. Tanto em *I4* como em *I5* a obtenção dos melhores resultados em termos de *CMT* exigiram uma redução em torno de 7% na melhor %PEP.

Um produto candidato *j* deixa de ser processado à medida que ocorrem sucessivos rolamentos do horizonte de tempo e o valor de t_k em $[t_k, t_k+L]$ passa a ser maior que o respectivo d_j do produto. Neste caso, o prazo para entrega do produto *j* já foi ultrapassado e ele ainda não foi atribuído a nenhuma das máquinas. Portanto, deixará de ser incluído no conjunto dos produtos candidatos e, consequentemente, de ser atribuído às máquinas.

A Tabela 4.4 compara os melhores resultados obtidos pelo método em todas as instâncias. A coluna *#PÑP* lista o número de produtos que deixaram de ser processados para que estes resultados fossem alcançados. A coluna *NTP/NA* apresenta o número de produtos que chegam em média a cada atualização nos dados do problema.

Tabela 4.4 – Comparação das melhores *%PEP* e *CMT* em *I1*, *I2*, *I3*, *I4* e *I5*

Inst.	*NTP/NA*	*%PEP*	*#PÑP*	*CMT*	*#PÑP*
I1	2,3	80,76	0	23,12	0
I2	8	68,63	0	26,84	0
I3	4	61,57	0	24,44	0
I4	7,6	56,81	6	29,16	0
		54,65	0		
I5	15,2	46,41	0	19,94	0

Os resultados obtidos pelo método demonstram a dificuldade em manter uma elevada *%PEP* a medida que *NTP/NA* aumenta. Por outro lado, os melhores *CMT* apresentam-se cada vez mais reduzidos a medida que *NTP/NA* aumenta.

4.7.3.3 –Desempenho das Máquinas

A Tabela 4.5 compara o desempenho das três máquinas quando o método obteve a melhor *%PEP*=80,76 e o menor *CMT*=29,23 na instância *I1*. As quatro colunas seguintes à coluna *M* contêm o número de Produtos Processados ($\#PP_p$), o Tempo Total de Processamento (TTP_p), o Custo de Troca Total (CTT_p) e o Atraso Médio (ATM_p) em cada máquina. O índice *p* indica que estas colunas apresentam os valores relativos ao melhor resultado em termos da *%PEP*. Assim, as quatro últimas colunas apresentam o índice *c* para indicar que apresentam os valores relativos ao melhor resultado em termos do *CMT*.

Tabela 4.5 – Resultados nas máquinas da melhor *%PEP* e *CMT* em *I1*

M	$\#PP_p$	TTP_p	CTT_p	ATM_p	$\#PP_c$	TTP_c	CTT_c	ATM_c
M1	15	149	656	10,86	17	155	472	14,05
M2	12	124	457	10,25	15	149	481	9,93
M3	19	196	667	7,94	14	120	304	8,42

O melhor resultado em termos da *%PPE*=80,76% apresenta um TTP_p de 149UT em *M1*, 124UT em *M2* e 196UT em *M3*. Os valores do CTT_p são de 656 em *M1*, 457 em *M2* e 667 em *M3*. O ATM_p é estabelecido dividindo-se a soma dos atrasos no processamento dos produtos pelo *#PP*. Desta forma, ocorre um ATM_p=10,86*UT* em *M1*, ATM_p=10,25 em *M2* e ATM_p=7,94 em *M3*.

Quando o método obteve o menor valor para o *CMT*=29,23 em *I1*, provocou uma alteração no $\#PP_c$ de cada máquina em relação ao $\#PP_p$ obtido quando alcançou a melhor *%PEP*. Esta alteração ocasionou aumento no TTP_c de *M1* e *M2* e pequena queda em *M3* devido a

redução no $\#PP_p$ por *M3*. O CTT_c sofre expressiva queda passando a 472 em *M1* e 304 em *M3*, acompanhado de um pequeno acréscimo em M2 passando a 481. A redução no valor do CTT_c é justificada pelas novas seqüências de produção estabelecidas, onde as alterações do $\#PP_p$ para $\#PP_c$ demonstram que a troca de produtos entre as máquinas foi capaz de produzir seqüências com menor CTT_c e, por conseguinte, menor *CMT*.

A Tabela 4.6 destaca os mesmos aspectos da Tabela 4.5 para os demais problemas. Os resultados demonstram que uma melhoria nos valores do CTT_c foi obtida com certa variação do $\#PP_p$ para o $\#PP_c$ de cada máquina, seguido pelo aumento no TTP_c em relação ao TTP_p em algumas máquinas.

Tabela 4.6 - Resultados nas máquinas da melhor *%PEP* e *CMT* em *I2*, *I3*, *I4* e *I5*

M – I2	$\#PE_p$	TTP_p	CTT_p	ATM_p	$\#PP_c$	TTP_c	CTT_c	ATM_c
M1	24	223	652	27,79	26	267	681	42,88
M2	30	264	854	32	24	241	626	32,95
M3	26	255	940	35,03	30	272	760	42,73
M – I3	$\#PE_p$	TTP_p	CTT_p	ATM_p	$\#PE_c$	TTP_c	CTT_c	ATM_c
M1	27	250	617	43,14	24	220	608	31,7
M2	28	257	649	46,39	27	228	640	38,55
M3	25	222	630	28,88	29	246	634	41,89
M – I4	$\#PE_p$	TTP_p	CTT_p	ATM_p	$\#PP_c$	TTP_c	CTT_c	ATM_c
M1	54	470	1383	109,68	54	505	1140	121,74
M2	48	464	1343	105,85	48	470	1023	117,52
M3	50	459	1292	111,92	50	464	1009	108,2
M - I5	$\#PE_p$	TTP_p	CTT_p	ATM_p	$\#PP_c$	TTP_c	CTT_c	ATM_c
M1	47	484	1154	141,25	56	554	1161	167,85
M2	59	532	1098	148,37	48	451	940	127,58
M3	46	408	1345	105,63	48	456	871	109,10

Na instância *I3*, a melhoria apresentada no CTT_c (608 em *M1*, 640 em *M2*, 634 em *M3*) em relação ao CTT_p (617 em *M1*, 649 em *M2*, 630 em *M3*) é obtida juntamente com uma redução no TTP_c (220 em *M1*, 228 em *M2*, 246 em *M3*) se comparados ao TTP_p (250 em *M1*, 257 em *M2*, 222 em *M3*). Desta forma, observa-se que o comportamento apresentado em *I1*, *I2*, *I4* e *I5* não é regra sendo o método capaz de obter o melhor *CMT* em *I3* sem precisar aumentar seu TTP_c.

4.7.3.4 – Resultados Computacionais para ρ=0,0 e ρ=0,05

Conforme explicado na seção 4.5, podem existir vários produtos candidatos j com x_{ij}=1 e $r_j \in [t_k, t_k+aL]$ após a resolução do problema de designação. Neste caso, será inserido na máquina i o produto j com menor r, sendo estabelecida uma nova seqüência de produção que será tratada por S_{ij} (seqüência de produção da máquina i após inserção definitiva do produto candidato j).

Os demais produtos $q \neq j$, tais que $x_{iq}=1$, serão inseridos apenas caso sua inserção em S_{ij}, avaliada pela heurística ALLPAIRS, retorne uma seqüência S_{iq} e um novo valor U'_{iq} tal que $U_{iq}-U'_{iq} \leq \rho$. O parâmetro ρ é fornecido pelo usuário e estabelece o valor máximo aceito como redução de U_{iq}.

Nas execuções da rotina MÁQUINAS apresentadas até agora foi adotado o valor ρ=0,05, ou seja, foi permitida numa mesma iteração a inserção de outros produtos candidatos q após a inserção do primeiro produto candidato j. Desta forma, q poderá entrar na seqüência S_{ij} mesmo que esta inserção não forneça um valor para U_{iq} tão bom quanto o que seria obtido caso q fosse inserido na seqüência antes do produto j.

A Tabela 4.7 lista os valores dos melhores resultados para *I1*, *I2* e *I3* usando ρ=0 e ρ=0,05. Na instância *I1* ocorrem os mesmos resultados, exceto pelo *CMT*=26,68 que foi menor quando ρ=0. Nas instâncias *I2* e *I3* os melhores resultados da *%PEP* são um pouco melhor quando ρ=0,05, mas continuam bastante próximos dos alcançados para ρ=0. O *CMT* é menor em *I2* quando ρ=0, porém não deixa de estar próximo do obtido quando ρ=0,05. Em *I3* a situação inverte-se com o menor *CMT* obtido quando ρ=0,05 e a melhor *%PEP* é praticamente a mesma nos dois casos.

Tabela 4.7 – Resultados usando ρ=0 e 0,05 em *I1*, *I2* e *I3*

$Inst^{\rho=0}$	*I1*	*I2*	*I3*
%PEP	80,76	67,49	61,56
L	2 e 3	1	1
#PÑP	0	0	0
CMT	28,76	26,68	25,05
L	12 e 13	12 e 13	26 até 33
#PÑP	0	0	0
$Inst^{\rho=0,05}$	*I1*	*I2*	*I3*
%PEP	80,76	68,63	61,57
L	2 e 3	4	14 e 15
#PÑP	0	0	0
CMT	29, 23	26,84	24,44
L	12 e 13	8 e 9	24 até 31
#PÑP	0	0	0

Os valores da *%PEP* e *CMT* nestas instâncias são próximos e obtidos com todos os produtos processados pelas máquinas (*PÑP*=0). Quando não é permitida inserção de mais de um produto na mesma iteração (ρ=0), a rotina MÁQUINAS obtém os melhores valores da *%PEP* trabalhando com um horizonte de tempo curto (*L*=1 em *I2* e *I3*).

Por outro lado, os melhores *CMT* são obtidos com valores maiores para L que permitem a inclusão de mais produtos no conjunto P_c ao estabelecerem intervalos $[t_k, t_k+L]$ maiores. Maior quantidade de produtos P_c aumenta a variedade de produtos candidatos, cujos custos de troca serão avaliados para uma possível inserção nas máquinas. Intervalos menores

limitam a variedade de produtos candidatos em P_c e, consequentemente, a escolha de menores custos de troca.

A Tabela 4.8 apresenta os melhores resultados de *I4* e *I5* com ρ=0,0 e ρ=0,05. Apesar de não apresentados na tabela, na maioria dos resultados obtidos variando *L* de 1 a 40 e fazendo ρ=0 o *#PÑP* foi diferente de zero tanto em *I4* quanto em *I5*. Em I4, para ρ=0, o *#PÑP*=0 quando L=20 e 21. Nos demais valores de *L*, a rotina deixa de entregar de 1 até 6 produtos. Em *I5*, o *#PÑP*=0 para ρ=0 quando *L*=1, 2 e 3. A instância *I5* deixa de processar entre 1 e 9 produtos nas execuções da rotina MÁQUINAS usando valores de $L \geq 4$.

Tabela 4.8 - Resultados usando ρ=0 e ρ=0,05 em *I4* e *I5*

$Inst^{\rho=0}$	*I4*	*I4*	*I5*	*I5*
%PEP	56,20	51,72	46,65	40,31
L	7	20 e 21	13	1
#PÑP	4	0	5	0
CMT	20,65	25,01	17,67	-
L	18 e 19	20 e 21	2	-
#PÑP	4	0	0	-
$Inst^{\rho=0,05}$	*I4*	*I4*	*I5*	*I5*
%PEP	56,81	54,65	46,41	
L	1	8 e 9	14 e 15	-
#PÑP	6	0	0	-
CMT	21,29	-	19,55	-
L	16 e 17	-	2 e 3	-
#PÑP	0	-	0	-

Analisando os resultados de *I4*, observa-se que a melhor *%PEP* ocorre com ρ=0,05 quando *%PEP*=56,81%, porém a rotina deixa de processar 6 produtos. O melhor resultado para ρ=0 é *%PEP*=56,20% com *#PÑP*=4. Quando ocorre a entrega de todos os produtos, utilizando ρ=0, a melhor *%PEP* vale 51,72%.

Conforme pode ser observado, o processamento de todos os produtos diminuiu o valor da *%PEP*, ou seja, aumentou o atraso. O melhor compromisso entre *#PÑP*=0 e alta *%PEP* ocorre em *I4* com ρ=0,05 quando *%PEP*=54,65%. O melhor compromisso entre *#PÑP*=0 e baixo *CMT* também ocorre quando ρ=0,05 com *CMT*=21,29. O menor *CMT*=20,65 em *I4* é obtido quando ρ=0,0 com *#PÑP*=4, ou seja, deixando de entregar 4 produtos.

Analisado os resultados de *I5*, observa-se que a melhor *%PEP*=46,65 é encontrada quando ρ=0 com *#PÑP*=5. Porém, o melhor compromisso entre *#PÑP*=0 e alta *%PEP* é obtido quando ρ=0,05 com *%PEP*=46,41%. O menor *CMT*=17,67 é obtido quando ρ=0 com

$\#P\tilde{N}P$=0, enquanto CMT=19,55 quando ρ=0,05 com $\#P\tilde{N}P$=0. Desta forma, em *I5* o melhor compromisso entre $\#P\tilde{N}P$=0 e baixo CMT ocorre quando ρ=0.

A inserção de mais de um produto na mesma iteração da rotina MÁQUINAS pressupõe certa degradação no valor do custo de troca e no tempo de processamento, pois utiliza-se o critério U_{iq}-$U'_{iq} \leq \rho$. Os resultados demonstram que a inserção de mais de um produto na mesma iteração não comprometeu o resultado global. A rotina em situações com alto *NTP/NA*, como em *I4* e *I5*, acaba deixando de entregar produtos caso não possa inserir mais que um produto numa mesma iteração. Num contexto dinâmico real, o processamento do produto, ainda que atrasado, pode apresentar relevância muito maior do que seu não processamento.

Pelos resultados apresentados, o objetivo de realizar a entrega de produtos no prazo beneficia-se mais da possibilidade de inserir mais de um produto numa mesma iteração. Os melhores compromissos entre $\#P\tilde{N}P$=0 e maior *%PEP* foram alcançados nas cinco instâncias quando ρ=0,05. Os melhores compromissos entre menor $\#P\tilde{N}P$=0 e *CMT* apresentaram valores próximos nas 5 instâncias, com certa superioridade dos resultados obtidos quando ρ=0,05.

O método apresenta o comportamento esperado, ou seja, inserir vários produtos numa mesma máquina a cada iteração apresenta impacto maior na *%PEP*. A demora na inserção aumenta a possibilidade de atrasos, conforme demonstraram os resultados. Aceitar um valor U_{iq}-$U'_{iq} \leq \rho$ repercutiu negativamente no custo de troca, porém, os resultados não demonstraram uma degradação muito grande nos *CMT's* obtidos quando ρ=0,05 em relação a ρ=0.

4.7.3.5 – Comparação da rotina MÁQUINAS com e sem a heurística ALLPAIRS

O algoritmo MORSS menciona que as cargas ou, neste contexto, produtos candidatos devem ter suas inserções avaliadas pelas *Funções Utilidade de Designação*. O método menciona a forma de cálculo destas funções, mas não define a forma como a inserção do produto candidato deve ser feita. Na adaptação do algoritmo, foi incluída no método uma heurística para Inserção do produto candidato *j* (tratada como INSERÇÃO) que permite encontrar o local de inserção na seqüência de produção da máquina *i* que retorne o melhor valor da *Função Utilidade de Designação* (U_{ij}).

Também foi adicionada ao método uma heurística capaz de realizar uma Busca em Vizinhança (ALLPAIRS) a partir da seqüência estabelecida pela heurística INSERÇÃO. Desta forma, a rotina MÁQUNAS insere o produto candidato *j* na seqüência de produção de determinada máquina *i* usando INSERÇÃO e, em seguida, otimiza esta seqüência utilizando ALLPAIRS (veja seção 4.5).

Os resultados apresentados pela rotina MÁQUINAS serão agora confrontados com aqueles obtidos sem a realização de uma Busca em Vizinhança. Afinal, a rotina de Busca em Vizinhança foi uma modificação no método introduzida por este trabalho. Desta forma,

torna-se conveniente estabelecer uma comparação no desempenho do método com e sem esta modificação.

A Tabela 4.9 apresenta os valores obtidos pela rotina MÁQUINAS sem executar a heurística de busca em vizinhança ALLPAIRS em *I1*, *I2* e *I3*. Além dos resultados relacionados à *%PEP*, *CMT* e aos diferentes valores de *L*, também são apresentados dados relativos ao tempo computacional. A coluna *TCT* lista em termos de minutos e segundos os valores do Tempo Computacional Total para concluir a resolução da instância, ou seja, o tempo gasto pela execução da rotina para estabelecer a última atribuição de produtos às máquinas.

Tabela 4.9 – Resultados de MÁQUINAS sem ALLPAIRS para *I1*, *I2* e *I3*

Inst.	*%PEP*	*L*	*TCT*	*CMT*	*L*	*TCT*
I1	77,71	5	41"	34,79	16–39	40"
I2	61,90	6/7	1'05"/1'08"	25	20–29	50"
I3	59,99	16–39	41"	26,83	10/11	49"/51"

A instância *I2*, por exemplo, atinge seu maior valor de *%PEP*=61,90% quando a rotina é executada usando *L*=6 ou *L*=7. O tempo computacional para *L*=6 foi de 1'05" e para *L*=7 foi de 1'08". O menor *CMT*=25 é obtido usando valores de *L*=20 até 29. Neste caso, o valor *TCT*=50" apresentado é uma média do *TCT* gasto ao executar a rotina MÁQUINAS para cada um desses valores de *L*.

A Tabela 4.10 apresenta as mesmas medidas da Tabela 4.9 utilizando a heurística de Busca em Vizinhança ALLPAIRS.

Tabela 4.10 – Resultados de MÁQUINAS com ALLPAIRS para *I1*, *I2* e *I3*

Inst.	*%PEP*	*TCT*	*L*	*CMT*	*L*	*TCT*
I1	80,76	46"	2/3	29,23	12/13	37"
I2	68,63	1'17"	4	26,84	8/9	1'31"/1'37"
I3	61,57	1'25"	14/15	24,44	24–31	1'26"

Convém esclarecer que nas três instâncias avaliadas não ocorrem produtos não processados, ou seja, as duas formas de execução estabeleceram seqüências nas quais as máquinas processaram todos os produtos que chegaram ao sistema de produção. Comparando os resultados apresentados na Tabela 4.10 com aqueles da Tabela 4.9, observa-se que a rotina MÁQUINAS com ALLPAIRS apresenta desempenho melhor tanto na *%PEP* como no *CMT* na maioria das instâncias. A única exceção ocorre em *I2*, onde o *CMT* é menor quando comparado ao menor valor encontrado pela rotina sem ALLPAIRS.

Na execução de MÁQUINAS sem ALLPAIRS, o *TCT* obtido é menor na grande maioria das instâncias do que usando ALLPAIRS. Por exemplo, *I3* apresenta *%PEP*=59,99% para

L=16 até 39 e um *TCT* médio de 41" ao executar a rotina sem ALLPAIRS (Tabela 4.9). Por outro lado, a rotina com ALLPAIRS obtém para *I3* uma *%PEP*=61,57% melhor, mas gasta um *TCT*=1'25" maior (Tabela 4.10). A única exceção ocorre em *I1* quando a rotina sem ALLPAIRS obtém *CMT*=34,79 com *TCT*=40", enquanto a rotina com ALLPAIRS obtém um melhor *CMT*=29,23 em um menor *TCT*=37".

Os resultados fornecidos pelos métodos são coerentes, pois a rotina ALLPAIRS apesar de permitir um busca maior no espaço de soluções gasta um tempo computacional maior para realizar tal busca. A rotina sem ALLPAIRS percorre um espaço de soluções mais limitado, conforme demonstram as melhores *%PEP* e *CMT* obtidos, porém é menos onerosa em termos de *TCT*.

A Tabela 4.11 apresenta os resultados de *I4* e *I5*. Estas instâncias apresentam uma dimensão total de 152 produtos que passam pelo sistema de produção ao final de todas as atualizações. A rotina MÁQUINAS sem ALLPAIRS deixa de entregar produtos (*PÑP*) quando obtém alguns dos melhores resultados para *%PEP* e *CMT* tanto em *I4* como em *I5*.

Tabela 4.11 – Resultados de MÁQUINAS sem ALLPAIRS para *I4* e *I5*

Inst.	*%PEP*	*L*	*TCT*	*PÑP*	*CMT*	*L*	*TCT*	*PÑP*
I4	45,16	1	2'03"	8	23,17	14/15	1'39"	2
I5	35,36	14/15	2'22"/2'21"	1	20,07	4	2'35"	0

Na primeira linha da Tabela 4.11, estão os melhores resultados obtidos para a *%PEP* e *CMT* em *I4*, porém deixaram de ser processados 8 produtos quando *%PEP*=45,16% e 2 produtos quando *CMT*=23,17. Na linha seguinte, estão os melhores valores relativos a *I5* obtidos com o não processamento de 1 produto quando *%PEP*=35,36% e com o processamento de todos os produtos quando *CMT*=20,07.

A Tabela 4.12 apresenta os resultados obtidos com a execução da rotina utilizando a heurística de Busca em Vizinhança ALLPAIRS. Neste caso, os melhores valores de *%PEP* e *CMT* foram atingidos com todos os produtos sendo processados. Destaca-se também o aumento considerável no Tempo Computacional Total (*TCT*).

Tabela 4.12 – Resultados de MÁQUINAS com ALLPAIRS para *I4*, *I5*

Inst.	*%PEP*	*L*	*TCT*	*PÑP*	*CMT*	*L*	*TCT*	*PÑP*
I4	54,65	8/9	13'56"/13'57"	0	21,29	16/17	14'01"	0
I5	46,41	14/15	22'01"	0	19,55	2/3	19'01"	0

Comparando os resultados obtidos pelos dois métodos apresentados na Tabela 4.11 e Tabela 4.12, observa-se que tanto a melhor *%PEP* como o melhor *CMT* são fornecidos pela rotina MÁQUINAS com ALLPAIRS. Os valores da *%PEP* apresentam uma melhoria mais acentuada, passando de *%PEP*=45,16% obtido pela rotina sem ALLPAIRS (Tabela 4.11) para *%PEP*=54,65% em *I4* utilizando ALLPAIRS (Tabela 4.12). O mesmo ocorre

em *I5*, onde os valores mudam de *%PEP*=35,36% (Tabela 4.11) para *%PEP*=46,41% (Tabela 4.12). Os valores do *CMT* também melhoraram, mas em uma menor proporção, passando de *CMT*=23,17 (Tabela 4.11) para *CMT*=21,29 (Tabela 4.12) em *I4* e de *CMT*=20,07 (Tabela 4.11) para *CMT*=19,55 (Tabela 4.12) em *I5*, respectivamente, com e sem ALPPAIRS.

O *TCT* passa de 2'03" quando *%PEP*=45,16% em *I4*, obtido ao executar MÁQUINAS sem ALLPAIRS (Tabela 4.11), para 13'56" e 13'57" quando obtém *%PEP*=54,65% executando ALLPAIRS (Tabela 4.12). O mesmo ocorre em *I5* onde o tempo de processamento para obter o melhor valor da *%PEP* passa de 2'22' e 2'21" sem ALLPAIRS (Tabela 4.11) para 22'01" com ALLPAIRS (Tabela 4.12). Variação semelhante ocorre em relação ao *CMT*, conforme valores apresentados nestas tabelas.

Estes valores mostram como o tempo computacional poderá crescer durante a resolução de instâncias com dimensão alta ao se utilizar uma heurística de busca em vizinhança. Este aumento no *TCT* pode tornar-se relevante num contexto dinâmico real. O uso da rotina MÁQUINAS sem Busca em Vizinhança seria mais recomendável para problemas que necessitem de respostas rápidas.

Porém, a rotina MÁQUINAS com Busca em Vizinhança apresentou nestas instância, em geral, resultados melhores. O desafio está em unir os bons resultados que uma exploração maior do espaço de soluções fornece com um baixo gasto de tempo computacional durante a obtenção dos mesmos.

4.7.4 –Rotina MÁQUINAS calculando L

Na seção anterior, a rotina MÁQUINAS foi executada várias vezes para uma mesma instâncias utilizando valores de L entre 1 e 40 *UT*. Desta forma, o desempenho do método para cada um dos valores de L utilizados foi avaliado e os melhores resultados apresentados. Estes resultados demonstraram que uma boa escolha de L permite retornar bons resultados em termos de porcentagem de Produtos Entregues no Prazo (*%PEP*) e Custo Médio de Troca (*CMT*).

O usuário, contudo, nem sempre conhece qual o melhor valor a ser empregado para este parâmetro ou possui tempo e informações suficientes para executar toda a rotina testando diferentes valores de L até encontrar a melhor desempenho. Por exemplo, o usuário gastaria um considerável tempo extra para estabelecer a seqüência de produção caso resolvesse avaliar o desempenho obtido testando diferentes valores de L em cada iteração da rotina.

Isto posto, o melhor procedimento seria trabalhar todo o tempo com um valor de L que permita a rotina realizar as melhores programações possíveis. O desafio passa a ser encontrar uma forma de obter tal valor de L que permita ao método tomar as melhores decisões dentro de cada horizonte de tempo.

Um estudo mais aprofundado do problema real envolvido e suas variáveis poderá revelar diversos aspectos que ajudem na definição de uma fórmula ou critério para determinação de L. Isto pode exigir o conhecimento de diversos aspectos inerentes ao problema, sendo um objetivo amplo que foge ao escopo desta dissertação.

Para o PDP estabelecido neste capítulo será sugerida a seguinte expressão que permitirá calcular L simultaneamente à resolução do mesmo.

$$L=\begin{cases} \lambda \dfrac{\sum_{j\in P_{kl}} (t_k - r_j)}{|P_{kl}|}, & 0<\lambda<\lambda_{max} \text{ e } P_{kl}=\{j\in P_k \,/\, t_k-r_j\neq 0\} \quad (4.28) \\ 1, & P_{kl}=\varnothing \quad (4.29) \end{cases}$$

As Equações 4.28 e 4.29 fornecem o critério adotado para calcular L. Na expressão 4.28, t_k é o limite inferior do horizonte de tempo estabelecido na $k^{\text{ésima}}$ iteração da rotina, ou seja, $t_k=\min\{r_j,\ \forall j\in P_k\}$. O conjunto P_{kl} é formado por todos os produtos disponíveis $j\in P_k$ com $t_k-r_j\neq 0$ (veja seção 4.5).

O valor do numerador na Equação 4.28 é o resultado do somatório de $t_k-r_j\neq 0$ e no denominador está a cardinalidade do conjunto P_{kl}, ou seja, o número de produtos disponíveis com $t_k-r_j\neq 0$. A Equação 4.28 fornecerá o valor médio do tempo dos produtos que, apesar de disponíveis, ainda não estão prontos para serem processados pelas máquinas ($t_k-r_j\neq 0$).

O parâmetro $0<\lambda<\lambda_{max}$ é fornecido pelo usuário e indica quanto do valor médio estabelecido será de fato utilizado como valor de L. Para $\lambda=1$ teremos, a cada iteração da rotina, L igual ao valor médio dos $t_k-r_j\neq 0$. Neste caso, em cada horizonte $[t_k,\ t_k+L]$, o conjunto P_c será formado pelos produtos candidatos j imediatamente disponíveis ($t_k=r_j$) ou cujo r_j seja menor ou igual ao instante atual t_k mais o valor médio obtido para os $t_k-r_j\neq 0$ ($r_j\leq t_k+L$).

O usuário poderá desejar incluir no intervalo produtos que estejam disponíveis em até metade do tempo médio $t_k-r_j\neq 0$ fazendo $\lambda=0{,}5$, ou disponíveis em até o dobro deste valor adotando $\lambda=2$. Assim, uma variação de λ poderá fornecer L's que incluam ou excluam r_j's no intervalo $[t_k,\ t_k+L]$. Isto determina um critério para a obtenção de L que deixa de ser fornecido pelo usuário e passa a variar a cada iteração, conforme os valores de λ, t_k e r_j com $j\in P_k$.

A Equação 4.29 retorna o valor 1 quando $j\notin P_{kl}\ \forall j\in P_k$, ou seja, todos os produtos disponíveis j apresentam $t_k=r_j$. Todos os produtos disponíveis, portanto, estão prontos para processamento e o conjunto P_{kl} estará vazio. Neste caso, não faz sentido estabelecer horizontes de tempo com tamanho elevados já que não existem produtos com $t_k\neq r_j$ para serem incluídos. Resumindo, foi adotado o critério de atribuir a L, a cada iteração k, λ vezes

o valor médio das diferenças entre o instante atual e o tempo de chegada dos produtos, cujas informações são disponíveis, mas que ainda não podem ser processados.

4.7.4.1 – Resultados Computacionais para instâncias I1, I2 e I3.

Na Figura 4.13, o símbolo "*" indica os melhores resultados em termos de *%PEP* e *CMT*. As letras A, B e D localizam nas curvas de desempenho os pontos com maiores *%PEP* e as letras C e E localizam os pontos com menores *CMT*.

Inicialmente, a rotina MÁQUINAS foi executada calculando *L* pelo valor médio apresentado na Equação 4.28 fazendo λ=1. Em seguida, este valor médio foi acrescido em 0,1 e a rotina foi novamente executada para λ=1,1. Estes acréscimos foram se sucedendo, fazendo com que ao todo a rotina MÁQUINAS fosse executada 10 vezes utilizando λ=1; 1,1; 1,2;...; 1,9.

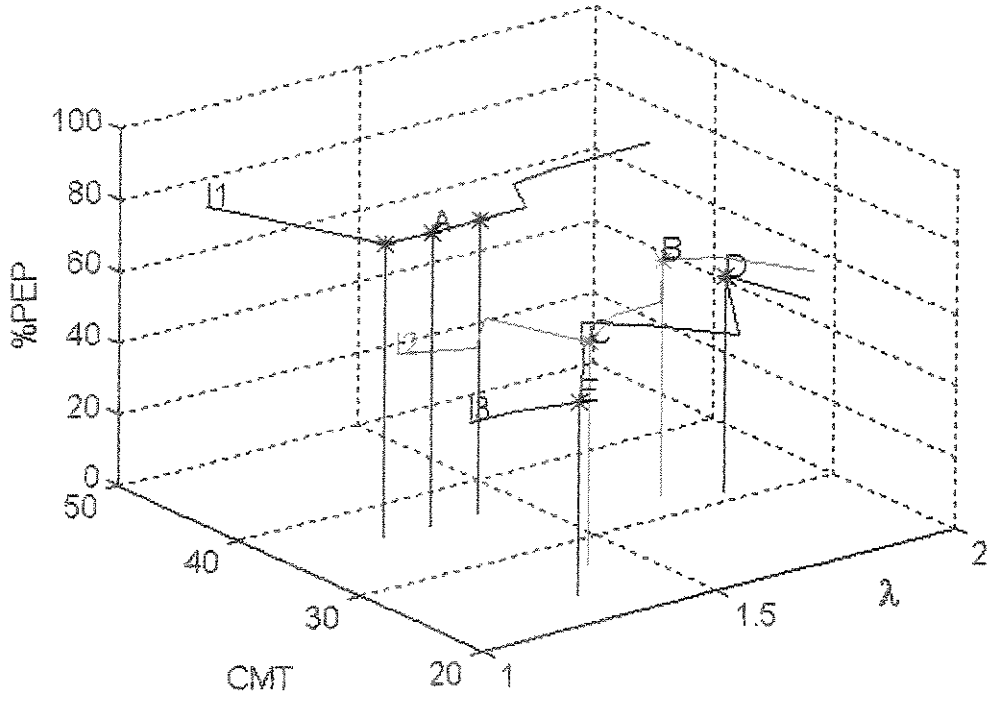


Figura 4.13 – Curvas de desempenho das instâncias *I1*, *I2* e *I3*

A Tabela 4.13 apresenta os melhores resultados obtidos. A coluna Pontos refere-se aos tipos de pontos assinalados na Figura 4.13 pelas letras A, B, C, D e E. Em seguida, são apresentados os valores correspondentes a esses pontos.

Tabela 4.13 – Melhores resultados obtidos para *I1*, *I2* e *I3*

Inst	*Ponto*	*(%**PEP**;CMT; λ)*	*Ponto*	*(%PEP;**CMT**; λ)*	*Δ%PEP*	*ΔCMT*
I1	A	(**82,23**; 36,14; 1,2)	-	-	-	-
		(**82,23**; 36,14; 1,3)				
		(**82,23**; 36,14; 1,4)				
I2	B	(**66,12**; 32,60; 1,7)	C	(63,83; **27,83**; 1,9)	2,29	4,77
I3	D	(**60,34**; 31,10; 1,8)	E	(53,82; **23,69**; 1,3)	6,52	7,41

Na instância *I1*, o melhor resultado obtido pela rotina MÁQUINAS calculando L também estabelece o melhor compromisso entre *%PEP* e *CMT*, ou seja, ao maior valor da *%PEP*=82,23% corresponde o menor *CMT*=36,14. Estes valores de *I1* são repetidos quando o cálculo de L é realizado utilizando λ=1,3 e 1,4.

Desta forma, o melhor resultado encontrado pela rotina MÁQUINAS é repetido ao serem utilizado valores de L que permitam incluir, para avaliação no intervalo $[t_k, t_k+L]$, produtos com r_j que retornem valores até 20% (λ=1,2), 30% (λ=1,2) e 40% (λ=1,4) acima do valor médio dos t_k-$r_j \neq 0$ $\forall j \in P_k$.

Na instância *I2*, a maior *%PEP*=66,12% é alcançada com um *CMT*=32,60 quando λ=1,7, porém, a obtenção do menor *CMT*=27,83 exigiu uma queda na porcentagem de entregas no prazo (*%PEP*=63,83) e foi obtido quando λ=1,9.

Logo, o maior valor da *%PEP* em *I2* foi alcançada quando o valor de L permitiu incluir nos intervalos $[t_k, t_k+L]$ produtos cujo r_j retornavam valores que estavam até 70% acima dos valores médios dos t_k-$r_j \neq 0$ $\forall j \in P_k$. O menor *CMT* foi alcançado com uma liberdade ainda maior sobre o valor r_j, permitindo que estes excedessem em até 90% o valor médio dos t_k-$r_j \neq 0$ $\forall j \in P_k$.

A instância *I3* alcança a maior *%PEP*=60,34% quando λ=1,8 e o menor *CMT*=23,69 quando λ=1,3. Desta forma, o maior valor de *%PEP* foi obtido avaliando produtos no intervalo $[t_k, t_k+L]$ cujo r_j excedesse em até 80% o valor médio dos t_k-$r_j \neq 0$ $\forall j \in P_k$. No caso do melhor valor para o *CMT*, a situação é alterada e o método obtém o melhor resultado ao avaliar produtos cujo r_j excedessem no máximo em 30% o valor médio dos t_k-$r_j \neq 0$ $\forall j \in P_k$.

A Tabela 4.14 apresenta uma comparação entre as melhores *%PEP* obtidas pela rotina MÁQUINAS calculando L e a rotina MÁQUINAS com parâmetro L fornecido pelo usuário. A fim de diferenciar as metodologias, a rotina MÁQUINAS que utiliza um valor fixo para L fornecido pelo usuário será tratada no decorrer deste trabalho como rotina MÁQUINAS com L fixo. Por outro lado, quando o valor de L é obtido a cada iteração pelas equações 4.28 e 4.29, a rotina será chamada MÁQUINAS calculando L.

Tabela 4.14 – MÁQUINAS calculando *L* x MÁQUINAS com *L* fixo: *%PEP*

Máquinas	Inst	*(%PEP;CMT)*	*L*	*#It*	*TCM*	*TCT*
calculando L	I1	(**82,23**; 36,14)	6	22	1,68"	37"
L fixo	*I1*	(**80,76**; 39,56)	3	27	1,70"	46"
calculando L	*I2*	(**66,12**; 32,60)	10	26	3,96"	1'43"
L fixo	*I2*	(**68,63**; 31,77)	4	30	2,56"	1'17"
calculando L	*I3*	(**60,34**; 31,10)	4	22	7,18"	2'38"
L fixo	*I3*	(**61,57**; 24,00)	15	17	5"	1'25"

A coluna *Máquinas* apresenta o tipo de rotina utilizada para solucionar cada instância. A coluna *Inst* indica qual instância foi solucionada pela rotina considerada. A coluna *(%PEP, CMT)* apresenta um par ordenado com o melhor valor da *%PEP* e o correspondente *CMT*. As colunas seguintes listam o tamanho do horizonte *L*, o número de Iterações (*#It*) executadas, o Tempo Computacional Médio por iteração (*TCM*) e o Tempo Computacional Total (*TCT*) em que a melhor *%PEP* foi alcançada.

Algumas colunas da tabela merecem maiores esclarecimentos. Os valores na coluna *L*, relativos às instâncias resolvidas pela rotina MÁQUINAS calculando *L*, apresentam o valor médio de todos os valores obtidos para *L* quando calculado a cada iteração.

O *TCM* refere-se ao tempo computacional gasto em média para que a programação da linha de produção fosse estabelecida a cada iteração pela rotina. *TCT* indica o tempo total de execução da rotina até que a última atribuição de produtos às máquinas tenha ocorrido.

A solução encontrada pela rotina MÁQUINAS calculando *L* em *I1* (segunda linha da Tabela 4.14) supera o resultado obtido por MÁQUINAS com *L* fixo nesta mesma instância (terceira linha da Tabela 4.14), tanto em relação a *%PEP* quanto ao correspondente *CMT*. Os valores do *#It*, *TCM* e *TCT* obtidos por MÁQUINAS calculando *L* para *I1* também foram menores do que aqueles obtidos por MÁQUINAS com *L* fixo para a mesma instância.

Porém, nas instâncias *I2* e *I3* este bom desempenho da rotina MÁQUINAS calculando *L* (quarta e sexta linhas da Tabela 4.14) não se repete e as maiores *%PEP* são obtidas pela rotina MÁQUINAS com *L* fixo (quinta e sétima linhas) com 68,63% e 61,57%. Na instância *I2*, ocorre uma proximidade nos melhores valores obtidos pelas duas formas de execução da rotina MÁQUINAS.

A melhor *%PPE* encontrada utilizando L fixo em *I2* foi 68,63% enquanto o melhor resultado calculando *L* foi *%PPE*=66,12%. O número de iterações (*#It*=26) de *I2* calculando *L* é menor que utilizando *L* fixo (*#It*=30), afinal, o valor médio dos *L*'s calculados vale 10, enquanto *L*=4 foi o valor fixo utilizado pela rotina em *I2*.

O tempo computacional apresentado pelo *TCM* e o *TCT* na Tabela 4.14 são melhores na rotina MÁQUINAS com *L* fixo em *I2* do que calculando *L*. Na instância *I3* repete-se o ocorrido em *I2*, ou seja, a melhor *%PEP* é obtida pela rotina MÁQUINAS utilizando *L* fixo com 61,57% (sétima linha da Tabela 4.14).

Este método também apresenta menor tempo computacional com *TCM*=5" e *TCT*=1'25". O *TCM* para solucionar *I3* utilizando *L* calculado (sexta linha da Tabela 4.14) é 7,18" e o respectivo *TCT* foi de 2'38".

A Tabela 4.15 apresenta uma comparação entre os melhores (menores) valores de *CMT* obtidos pela rotina MÁQUINAS calculando *L* e usando L fixo.

Tabela 4.15 – MÁQUINAS calculando *L* x MÁQUINAS com *L* fixo: *CMT*

Rotina	Inst	*(%PEP; **CMT**)*	*L*	*#It*	*TCM*	*TCT*
L calculado	*I1*	(82,23; **36,14**)	6	22	1,6"	37"
L fixo	*I1*	(76,23;**29,23**)	13	23	1,6"	37"
L calculado	*I2*	(63,83;**27,83**)	10	26	3,88"	1'41"
L fixo	*I2*	(58,68;**26,84**)	9	29	3,34"	1'37"
L calculado	*I3*	(53,82;**23,69**)	6	22	5,72"	2'06"
L fixo	*I3*	(61,28;**24,44**)	31	17	5,05"	1'26"

Na instância *I1* o menor valor de *CMT* (29,23) foi obtido usando *L* fixo igual a 13 em um tempo computacional de 37" (terceira linha da Tabela 4.15). Na instância *I2* o menor valor de *CMT* (26,84) também foi obtido usando *L* fixo igual a 3 em um tempo computacional de 1'37". Na instância *I3* o menor valor de *CMT* (24,69) foi obtido calculando-se *L*, sendo utilizado um tamanho médio do horizonte *L*=6 em um tempo computacional de 2'06". O tempo computacional maior quando *L* é calculado revela o gasto extra de tempo para se estabelecer um novo valor de *L* a cada iteração.

Pelos resultados apresentados nas Tabelas 4.14 e 4.15, os melhores valores da *%PEP*, a medida que aumenta o *NTP/NA* das instâncias, passam a ser determinados pela rotina MÁQUINAS com *L* fixo já que apenas *I1* com *NTP/NA*=2,3 (Tabela 4.1) apresentou a melhor *%PEP* fornecida pela rotina MÁQUINAS calculando *L*.

Por outro lado, os melhores valores do *CMT* são determinados pela rotina MÁQUINAS usando L fixo nas duas instâncias com menores valores do *NTP/NA*. A instâncias *I3* com *NTP/NA*=8,0 (Tabela 4.1) apresenta o melhor resultado do *CMT* determinado pela rotina MÁQUINAS calculando *L*.

Porém, conforme já foi mencionado, os melhores resultados nestas duas metodologias apresentam-se bastante próximos, não havendo uma considerável superioridade nos resultados de uma metodologia em relação a outra.

4.7.4.2 – Resultados Computacionais para Instâncias I4 e I5.

A Figura 4.14 apresenta as curvas de desempenho das instâncias *I4* e *I5* quando MÁQUINAS calcula L para $\lambda \in \{1; 1{,}1; 1.2, 1.3; \ldots; 1{,}9\}$. Na instância *I4* ocorrem 4 resultados que podem ser considerados os melhores estabelecidos por esta rotina.

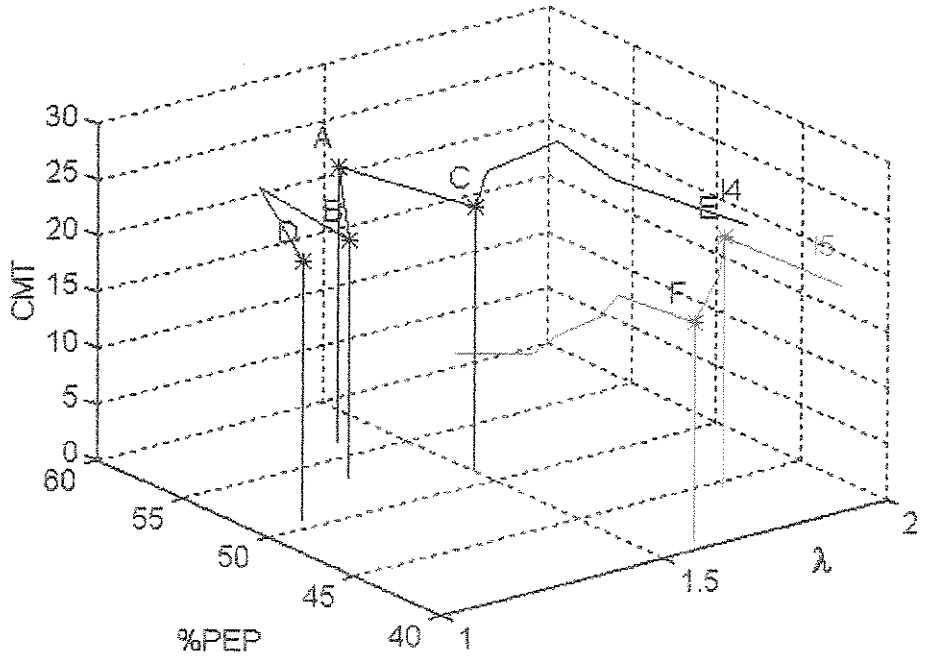


Figura 4.14 – Curvas de desempenho das instâncias *I4* e *I5*

Os valores do *CMT* em *I4* apresentam-se entre 20 e 25 enquanto as *%PEP* concentram-se entre 50% e 55%. Na instância *I5* ocorrem 2 resultados que podem ser considerados os melhores resultados. Observa-se que os valores do *CMT* em *I5* também estão entre 20 e 25, enquanto os valores da *%PEP* concentram-se entre 40% e 45%. Os valores da *%PEP* em *I5* apresentam uma queda acentuada em relação aos valores da *%PEP* obtidos em *I4*.

As letras A e C localizam na curva de desempenho de *I4* os pontos com o melhor resultado de *%PEP*, enquanto B e D localizam os pontos com melhores *CMT*. A letra E localiza o ponto com melhor *%PEP* em *I5* e F o ponto com melhor valor de *CMT*.

A Tabela 4.16 lista os valores relativos aos melhores resultados obtidos calculando-se L. Também há uma coluna indicando o número de produtos que não foram processados (*PÑP*). A instância *I4* obtém os melhores resultados quando deixa de realizar o processamento de alguns produtos.

Neste caso, quando a rotina calcula L e utiliza λ=1,4 e 1,3 são obtidos valores relativos aos dois pontos assinalados pelas letra A e B na Figura 4.14. Estes pontos indicam,

respectivamente, a maior *%PEP*=56,45% e o menor *CMT*=21,08. Estes resultados foram alcançados deixando-se de realizar o processamento de 5 produtos quando *%PEP*=56,45% e não processando 2 produtos quando *CMT*=21,08 (Tabela 4.16).

Tabela 4.16 – Melhores resultados obtidos para *I4* e *I5*

P	*Ponto*	*(%**PEP**;CMT;λ)*	*PÑP*	*Ponto*	*(%PEP;**CMT**;λ)*	*PÑP*	*Δ%PEP*	*ΔCMT*
I4	A	(**56,45**;24,44;1,4)	5	B	(53,28;**21,08**;1,3)	2	3,17	3,36
	C	(**51,26**;23,43;1,5)	0	D	(50,72;**23,17**;1,1)	0	0,54	0,26
I5	E	(**44,32**;22,44;1,9)	0	F	(40,84;**19,38**;1,6)	0	3,46	3,06

A Tabela 4.16 também apresenta os melhores valores de *%PEP* e *CMT* em *I4* quando *PÑP*=0 (terceira linha da tabela) que foram assinalados pelas letras C e D na Figura 4.14. Neste caso, ocorrem *%PEP*=51,26% e *CMT*=23,17 como melhores resultados obtidos usando, respectivamente, λ=1,5 e 1,1. Os valores obtidos indicam que o processamento de todos os produtos leva a uma considerável redução no valor da *%PEP* obtida em C quando comparado ao valor apresentado no ponto A.

Os valores da *Δ%PEP*=0,54 e *ΔCMT*=0,26 em C e D são bem menores do que em A e B com *Δ%PEP*=3,17 e *ΔCMT*=3,36. As soluções estabelecidas para *I4* em C e D, além de obtidas com *PÑP*=0, apresentam um melhor compromisso entre os valores da *%PEP* e *CMT*. As melhores soluções obtidas em *I5* foram encontradas com o processamento de todos os produtos. Destaca-se a queda acentuada no valor da *%PEP* de *I5* em relação à *I4*, demostrando novamente que numa situação de aumento no *NTP/NA* ocorre uma piora nas *%PEP*.

A Tabela 4.17 apresenta a comparação da melhor *%PEP* para *I4* e *I5*, obtida pela rotina MÁQUINAS calculando *L* e MÁQUINAS com *L* fixo. A notação adotada é a mesma estabelecida na Tabela 4.14. Os melhores resultados de *I4* apresentam-se próximos nos dois métodos com *%PEP*=54,65% ao utilizar L fixo e *%PEP*=51,26% calculando *L*. Na instância *I5* a proximidade no valor da melhor *%PEP* obtida pelo dois métodos aumenta, ocorrendo *%PEP*=46,41% com L fixo e *%PEP*=44,32 calculando *L*. Logo, a melhor *%PEP* tanto em *I4* como em *I5* foi encontrada utilizando *L* fixo.

Tabela 4.17 – MÁQUINAS calculando *L* x MÁQUINAS com *L* fixo: *%PEP*

Rotina	Inst	*(%**PEP**;CMT)*	*L*	*#It*	*TCM*	*TCT*
L calculado	*I4*	(**51,26**;23,17)	9	31	36,70"	18'58"
L fixo	*I4*	(**54,65**;26,61)	9	31	28,61"	14'47"
L calculado	*I5*	(**44,32**;22,44)	7	24	56,70"	22'41"
L fixo	*I5*	(**46,41**;23,82)	15	21	1'03"	22'01"

De forma geral, *I4* e *I5* apresentam um gasto de tempo computacional maior devido ao aumento na dimensão do problema. Em *I4* temos *NTP/NA*=7,6 e a rotina MÁQUINAS calculando *L* gasta um *TCT*=18'58" contra um *TCT*=14'47" usando L fixo. Na instância *I5*

há *NTP/NA*=15,2 produtos chegando em média nas 10 atualizações realizadas (Tabela 4.1) e o método gasta um *TCT*=22'41" ao calcular *L* e um *TCT*=22'01" ao utilizar *L* fixo. Este maior volume de novos produtos chegando ao sistema de produção obriga o método, tanto calculando *L* quanto com *L* fixo, a estabelecer seqüências de produção maiores em cada máquina. Isto acarreta um maior gasto de tempo computacional para avaliar as possíveis inserções e reseqüênciamentos.

A rotina MÁQUINAS leva mais tempo calculando *L* do que quando utiliza *L* fixo, devido ao tempo computacional extra envolvido na determinação de *L* a cada iteração pelas Equações 4.28 e 4.29. O cálculo de *L* em *I4* e *I5* estabelece, de forma geral, valores menores para *L* do que os utilizados por *L* fixo ao obter os melhores resultados. Assim, a rotina que calcula *L* acaba executando mais iterações, levando o método a realização de um número maior de programações e, por conseguinte, a um gasto de tempo maior na obtenção dos resultados.

A Tabela 4.18 apresenta uma comparação entre os melhores (menores) valores do *CMT* obtidos pela rotina MÁQUINAS (calculando *L* e usando *L* fixo). Na instância *I4*, o menor *CMT*=21,23 foi obtido utilizando um valor fixo de *L* igual a 17 e levando um *TCT*=14'01". Na instância *I5*, o menor *CMT*=19,38 foi obtido em um *TCT*=19'38" e foram calculados valores de *L* que, na média, apresentaram-se em torno de *L*=6.

Tabela 4.18 – MÁQUINAS calculando *L* x MÁQUINAS com *L* fixo: *CMT*

Rotina	Inst	*(%PEP;**CMT**)*	*L*	*#It*	*TCM*	*TCT*
L calculado	*I4*	(50,72;**23,17**)	7	32	28,56"	15'14"
L fixo	*I4*	(46,74;**21,29**)	17	28	30,03"	14'01"
L calculado	*I5*	(40,84;**19,38**)	6	23	51,21"	19'38"
L fixo	*I5*	(39,48;**19,94**)	3	28	40,07"	19'01"

Pelos resultados apresentados nas Tabelas 4.17 e 4.18, os melhores valores da *%PEP* são determinados pela rotina MÁQUINAS com *L* fixo, onde as duas instâncias apresentam elevado valor de *NTP/NA* com *NTP/NA*=7.6 em *I4* e *NTP/NA*=15.2 em *I5* (Tabela 4.1). Por outro lado, os melhores valores do *CMT* são determinados pela rotina MÁQUINAS com *L* fixo em *I4* e calculando *L* em *I5*. Neste caso, novamente o melhor *CMT* na instância com maior *NTP/NA* (*I5*) foi encontrado pela rotina MÁQUINAS calculando *L*.

O critério de cálculo de *L* apresentado depende de um parâmetro λ fornecido pelo usuário, mas a escolha de λ vincula-se a um critério ligado aos produtos candidatos e não a uma opção livremente adotada pelo usuário.

Assim, o valor de *L* passa a ser obtido durante a execução do método considerando o cenário de produção envolvido a cada iteração. O tempo computacional apresenta-se menor na rotina que utiliza *L* fixo.

Porém, o tempo computacional gasto a mais no cálculo de L não representa um empecilho por que este valor não está muito acima do tempo gasto executando a rotina com L fixo. Uma forma mais elaborada de cálculo de L poderá ocasionar economia no tempo computacional envolvido.

Os resultados apresentados em todas as instâncias indicam que a rotina MÁQUINAS calculando L superou MÁQUINAS com L fixo apenas na instância *I1* onde o *NTP/NA*=2,3. A medida que *NTP/NA* aumenta nas demais instâncias os melhores resultados são obtidos pela rotina MÁQUINAS com L fixo. Porém, comparando-se os valores das melhores *%PEP* nas duas metodologias observa-se uma proximidade nos resultados alcançados.

A rotina MÁQUINAS calculando L obteve o melhor *CMT* nas instâncias *I3* e *I5*, ou seja, nas duas instância onde ocorrem 10 atualizações nos dados. Isto conduz à idéia de que o cálculo de L comporta-se melhor na obtenção dos menores *CMT* quando novos produtos tornam-se disponíveis mais rapidamente ao sistema de produção. De forma geral, os valores obtidos para o CMT pelas duas metodologias apresentam-se próximos o que não permite apontar determinado método como extremamente vantajoso.

Caso o usuário disponha de conhecimento suficiente para estabelecer valor de L que fornecerá os melhores resultados, torna-se conveniente o uso da rotina MÁQUINAS com L fixo.

Se o usuário não dispõe deste conhecimento o cálculo de L demonstra-se, pelos resultados obtidos, uma alternativa conveniente. Afinal, L foi determinado levando-se em conta informações disponíveis a cada iteração e segundo um critério previamente definido nas Equações 4.28 e 4.29.

4.7.5 – Comparação dos resultados de MÁQUINAS x RBI

A Rotina de Busca e Inserção (RBI) foi apresentada na seção 4.6 e, conforme explicado, seu objetivo é solucionar o Problema Dinâmico de Programação (PDP) estabelecido neste capítulo. Para isso, utiliza atribuições aleatórias dos produtos às máquinas e heurísticas de Inserção e Busca em Vizinhança. Seus resultados servirão como critério de comparação para a rotina MÁQUINAS calculando L e para rotina MÁQUINAS com L fixo.

Nos testes realizados, RBI foi executada partindo de 20 diferentes atribuições iniciais envolvendo os produtos disponíveis na primeira iteração do método e as três máquinas do problema. A atribuição de produtos às máquinas realizada na primeira iteração da RBI é chamada de semente inicial porque a programação dos demais produtos que chegam ao sistema de produção serão construídas a partir dela.

Desta forma, a execução do método utilizando sementes iniciais distintas poderá levar à construção de diferentes seqüências de produção e, por conseguinte, a diferentes medidas de desempenho.

4.7.5.1 – Resultados da RBI relativos às instâncias I1, I2 e I3

A Figura 4.15 apresenta a curva com o desempenho da RBI em termos de *%PEP* e *CMT*, ao ser executada para cada uma das 20 sementes iniciais. Estas sementes estão enumeradas no eixo representado pela letra μ. As letras maiúsculas indicam os locais das curvas onde estão os melhores resultados obtidos. Os pontos da curva com os melhores resultados estão representados pelo símbolo "*".

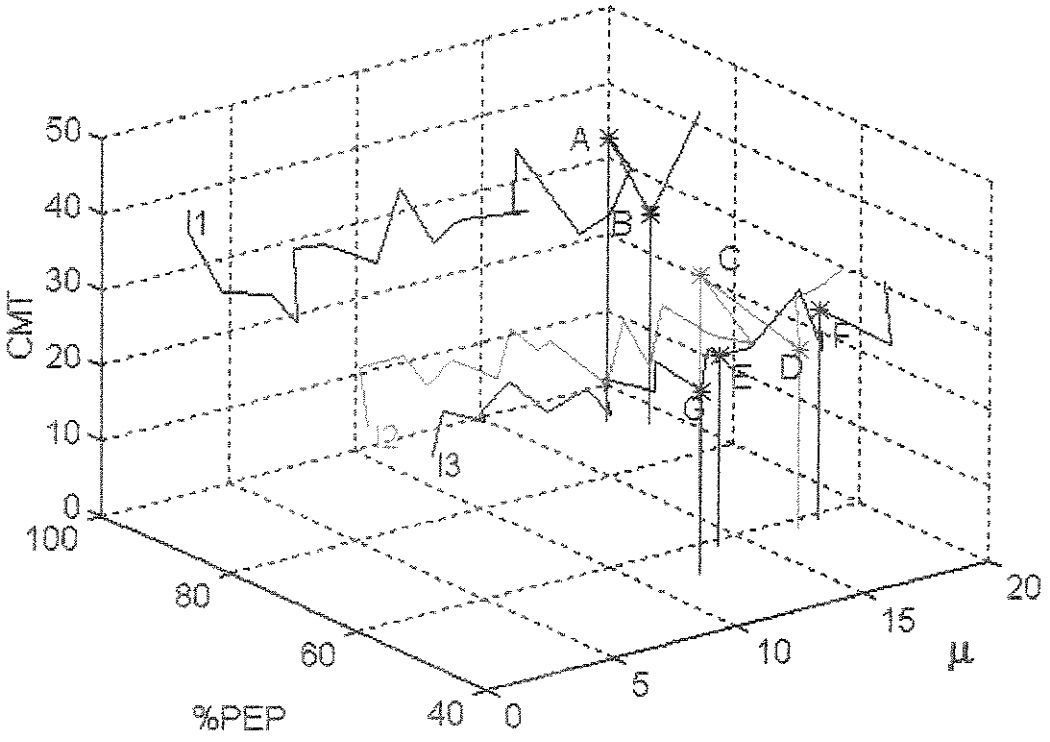


Figura 4.15 –Curva de desempenho da RBI nas instâncias *I1*, *I2* e *I3*.

A curva de desempenho da instância *I1* está localizada em uma região com maiores valores de *%PEP* e *CMT*, ficando bastante concentrada entre 30 e 40 para o *CMT* e entre 80% e 100% para a *%PEP*. Esta curva demonstra que nas 20 execuções realizadas o método forneceu valores elevados de *%PEP* para *I1*. Porém, os valores obtidos para o *CMT* também são elevados o que não é conveniente.

A curva de *I2* apresenta uma concentração de $\%PEP \in [60, 80]$ e $CMT \in [20, 30]$. A maioria dos valores de *I3* apresenta $\%PEP \in [40, 60]$ e $CMT \in [20, 30]$. Os valores obtidos para *I2* e *I3* retratam, conforme já foi observado nas seções 4.7.3 e 4.7.4, a dificuldade de manter uma boa *%PEP* quando aumenta o número de produtos que chegam ao sistema de produção.

Na instância *I2* ocorre *NTP/NA*=2,3, enquanto *I3* tem 4 produtos chegando a cada atualização (Tabela 4.1). A dificuldade causada por este aumento novamente está demonstrada pelos valores obtidos pelo método para a *%PEP* que passam em *I3* a concentrar-se num intervalo de valores menores.

A Tabela 4.19 contém os valores dos melhores resultados obtidos pela RBI. A coluna *Inst* apresenta as instâncias avaliadas, a coluna *Ponto* indica as letras que localizam, na Figura 4.15, os pontos onde foram obtidos os melhores valores em termos da *%PEP* e do *CMT*. A coluna *(%PEP;CMT;μ)* contém as coordenadas relativas aos pontos com os melhores resultados. As linhas relativas às letras A, C e E apresentam as melhores *%PEP* e às letras B, D e G os melhores *CMT*.

Tabela 4.19 – Melhores resultados obtidos para *I1*, *I2* e *I3*

Inst	*Ponto*	*(%**PEP**;CMT;μ)*	*Ponto*	*(%PEP;**CMT**;μ)*	*Δ%PEP*	*ΔCMT*
I1	A	(**92,32**; 37,37,18)	B	(89,49;**27,53**;19)	2,83	9,65
I2	C	(**69,69**;29,71,16)	D	(57,97;**23,55**;17)	11,72	6,16
I3	E	(**58,79**;25,05,14)	G	(53,70;**24,00**;12)	5,09	1,05
		(**58,79**;25,05,18)				

Por exemplo, a instância *I1* apresenta os melhores resultados em dois pontos distintos. No ponto A da Figura 4.15 , está o melhor valor da *%PEP*=92,32% seguido do correspondente *CMT*=37,37 obtidos quando o método estava executando a semente de número μ=18. No ponto indicado por B na Figura 4.15 está o melhor *CMT*=27,53 com a respectiva *%PEP*=89,49% obtidos quando o método executou a semente μ=17.

A Tabela 4.19 também fornece a variação na *%PEP* e no *CMT* entre os pontos destes melhores valores. Desta forma, a obtenção do melhor resultado em termos de *CMT* na instância *I1* significará uma redução de 2,83% no valor da melhor *%PEP*, fato que é negativo, porém representará uma redução no *CMT* de 9,65.

Os melhores valores nestas três instâncias não são encontrados nas primeiras sementes iniciais testadas pela RBI, aparecendo apenas quando μ=12,14,16,17 e 18. Assim, no mínimo 11 sementes iniciais foram avaliadas sem que fosse possível estabelecer, a partir delas, seqüências de produção que retornassem ao final os melhores resultados das medidas de desempenho.

4.7.5.2 – Resultados de MÁQUINAS x RBI em I1, I2 e I3

A Figura 4.16 apresenta os pontos (*%PEP*,*CMT*) obtidos ao final da execução de RBI, rotina MÁQUINAS com *L* fixo e rotina MÁQUINAS calculando *L* para a instância *I1*. As letras maiúsculas indicam os pontos com os melhores resultados alcançados por cada método.

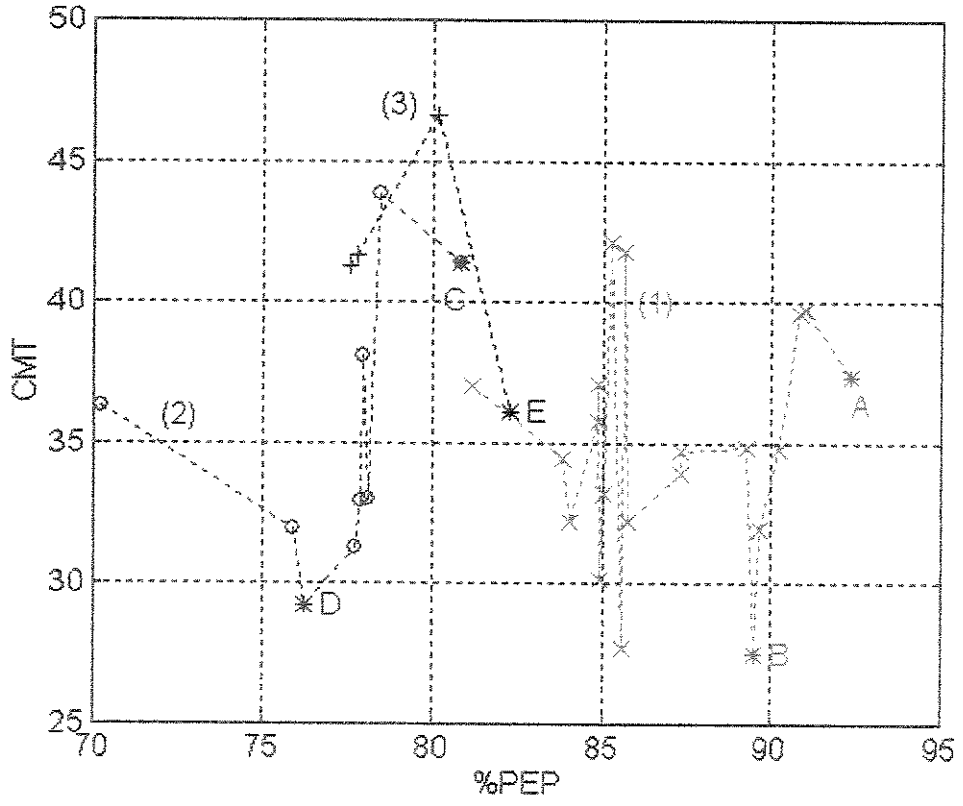


Figura 4.16 – Resultados fornecidos por MÁQUINAS e RBI em *I1*.

O tracejado (1) liga o conjunto de pontos fornecidos pela RBI, onde o símbolo "x" representa o ponto (*%PEP,CMT*) obtido ao final de cada execução do método para as 20 semente iniciais. O tracejado (2) liga os pontos fornecidos pela rotina MÁQUINAS com L fixo, onde o símbolo "o" representa o ponto (*%PEP,CMT*) fornecido por este método para cada L=1 até 40*UT*. O tracejado (3) liga os pontos obtidos pela rotina MÁQUINAS calculando L e o símbolo "+" representa o ponto (*%PEP,CMT*) obtido para cada λ=1; 1,1; 1,2,...,1,9.

A RBI (tracejado (1)) apresenta uma concentração maior de *%PEP* entre os valores 85 e 90% e *CMT* entre 30 e 40, conforme já havia sido observado na Figura 4.15. Os melhores valores de *%PEP* obtidos por RBI alcançam valores entre 90 e 95%. Os menores *CMT* são obtidos em dois pontos e situam-se na faixa de valores entre 25 e 30.

A rotina MÁQUINAS com L fixo apresenta um total de 9 diferentes resultados correspondentes a execução da rotina para valores fixos de L a partir de 1 até 40*UT*. Ocorre uma diversidade baixa de resultados, onde de 40 execuções houve uma concentração em apenas 9 diferentes pontos (*%PEP*; *CMT*). A maioria dos valores de *%PEP* situam-se entre 75 e 80% e apenas a melhor *%PEP* fica acima dos 80%. Os valores do *CMT* apresentam-se mais dispersos num intervalo de valores entre 30 e 40 com apenas o menor *CMT* obtido abaixo de 30.

A rotina MÁQUINAS calculando *L*, conforme explicado na seção 4.6, foi avaliada nas 5 instâncias consideradas utilizando diferentes valores de λ=1;1,1;1,2;...;1,9. Desta forma, a cada λ testado, a rotina estabelecia uma solução para *I1*.

Os pontos assinalados na Figura 4.16 também demonstram uma baixa variedade nos resultados obtidos, onde dos 10 valores de λ testados foram obtidos apenas 4 resultados distintos. Ocorrem três diferentes *%PEP* entre 75 e 80% e apenas o melhor resultado ultrapassa os 80% de produtos entregues no prazo. O *CMT* mostrou-se bastante elevado em todo os pontos, ficando acima de 40 para todos os valores com exceção do melhor resultado em termos do *CMT*. Destaca-se nestes resultados o fato de ser o único método que encontrou a melhor *%PEP* e *CMT* no mesmo ponto E (solução de compromisso).

A Tabela 4.20 compara os melhores resultados obtidos usando a RBI, rotina MÁQUINAS com *L* fixo e MÁQUINAS calculando *L* quando executadas na instância *I1*. As letras *A* e *B* na Tabela 4.20 referem-se a localização na Figura 4.16, respectivamente, das melhores *%PEP*=92,32% e *CMT*=27,53 obtidos pela RBI. Os pontos *C* e *D* representam, respectivamente, as melhores *%PEP*=80,76% e *CMT*=29,23 obtidos pela rotina MÁQUINAS usando *L* fixo. No ponto E estão a melhor *%PEP*=82,23 e o melhor *CMT*=36,14 obtidos pela rotina MÁQUINAS calculando *L*.

Tabela 4.20 – Comparação dos melhores resultados em *I1*

Ponto	*(%**PEP**;CMT)*	*#It*	*TCM*	*TCT*	Ponto	*(%PEP;**CMT**)*	*#It*	*TCM*	*TCT*
A	(**92,32**; 37,37)	21	0,14"	3"	*B*	(89,49;**27,53**)	21	0,28"	6"
C	(**80,76**; 41,39)	27	1,70"	46"	*D*	(76,23;**29,23**)	23	1,6"	37"
E	(**82,23**; 36,14)	22	1,68"	37"		-	-	-	-

Os valores apresentados na Tabela 4.20 confirmam o que já podia ser observado na Figura 4.16, ou seja, os melhores resultados em termos de *%PEP* e *CMT* são obtidos pela RBI. A diferença de valores é maior quando compara-se as *%PEP*, pois RBI consegue um resultado acima de 90% enquanto MÁQUINAS com *L* fixo ou calculando *L* não chegam a 83% dos produtos entregues no prazo.

Quando são comparados os melhores valores em termos do *CMT*, a diferença entre o melhor valor obtido pela rotina MÁQUINAS com *L* fixo e RBI não é tão alta e ambos os métodos conseguem resultados menores que 30. A rotina MÁQUINAS calculando *L* não apresenta um bom desempenho para o *CMT*, não conseguindo sequer ficar abaixo de 35 no melhor resultado obtido.

Na RBI, o número de iterações (*#It*) é sempre igual ao número de atualizações realizadas na instância mais a iteração que realiza a programação inicial dos produtos nas máquinas. Isto acontece porque RBI altera as seqüências à medida que ocorrem as atualizações, ao contrário da rotina MÁQUINAS que realiza esta alteração no contexto do horizonte de tempo rolante.

Está sendo suposto que as atualizações ocorrem de forma contínua nos três métodos, ou seja, ocorrem n atualizações e os três métodos recebem estas informações nas n primeiras iterações. Como RBI retorna sempre uma seqüência de produção para todos os produtos disponíveis, sua execução termina na $(n+1)^{ésima}$ iteração.

A rotina MÁQUINAS atualiza a linha de produção apenas com os produtos candidatos que foram efetivamente atribuídos às máquinas ao final de cada iteração. Podem sobrar produtos disponíveis numa mesma iteração caso nem todos os produtos candidatos tenham sido atribuídos às máquinas, ou caso o conjuntos de produtos candidatos seja diferente do conjunto de produtos disponíveis. Desta forma, se ocorrem n sucessivas atualizações nos dados da instância, o *#It* das rotinas MÁQUINAS calculando L e utilizando L fixo será no mínimo igual a $n+1$.

O *TCM* apresenta-se bastante reduzido na RBI que leva menos de 1' para estabelecer as seqüências de produção quando obtém a melhor *%PEP* ou o melhor *CMT*. RBI também gasta um *TCT*=3" para obter a melhor *%PEP* e 6" para alcançar o melhor *CMT*. A rotina MÁQUINAS com L fixo gasta *TCM*=1,70"e *TCM*=1,6" quando obtém, respectivamente, a melhor *%PEP* e *CMT*. A execução deste método para resolver toda *I1* leva *TCT*=46" durante a obtenção da melhor *%PEP* e *TCT*=37" quando obtém o melhor *CMT*. A rotina MÁQUINAS calculando L gasta um tempo computacional pouco maior que a rotina MÁQUINAS com L fixo, levando um *TCM*=1,68" e *TCT*=37"para obter num mesmo ponto a maior *%PEP* e o menor *CMT*.

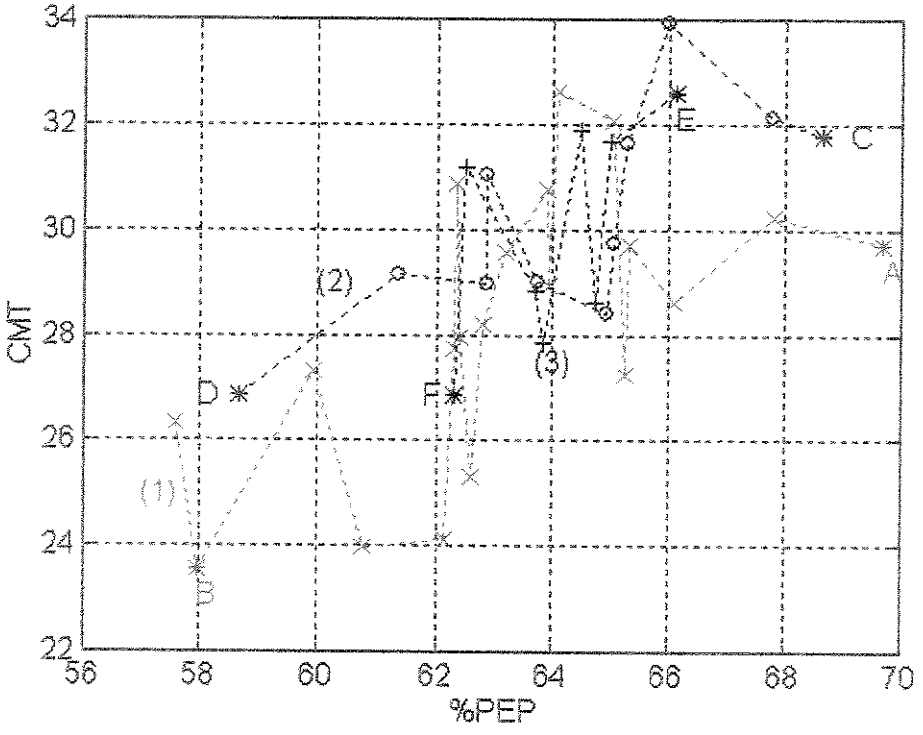


Figura 4.17 – Resultados fornecidos por MÁQUINAS e RBI em *I2*.

A Figura 4.17 apresenta o resultados dos métodos para a instância *I2*. A notação é a mesma utilizada na Figura 4.16. Os resultados apresentam uma redução global no *CMT* que não ultrapassa o limite máximo de 34 e queda na *%PEP* que não excede 70%. As três formas de resolução concentram pontos em uma faixa de valores entre 62 e 66% na *%PEP* e 28 e 32 no *CMT*. Não ocorre uma dissociação dos pontos como em *I1*, onde o conjunto de pontos em cada método aparecia em regiões do plano quase que totalmente separadas.

O comportamento dos três métodos revela a dificuldade de aumentar a *%PEP* mantendo um *CMT* baixo, pois todos os melhores *CMT* (pontos B, D e F) são obtidos com baixa *%PEP*. Analogamente, nas três formas de resolução as melhores *%PEP* (pontos A, C e E) ocorrem com elevado *CMT*.

A RBI destaca-se na obtenção de bons valores para *CMT*, alcançando o melhor *CMT* em relação aos demais e obtendo em outros 4 pontos resultados menores que os apresentados pela rotina MÁQUINAS com *L* fixo e *L* calculado. Isto significa que, partindo de pelo menos 4 diferentes atribuições aleatórias, RBI apresentou *CMT* diferentes e menores que os melhores obtidos pelos demais métodos.

Nos valores obtidos para *%PEP*, ocorre forte concentração de *%PEP* entre 62 e 66% nos três casos. Vale mencionar o bom desempenho da rotina MÁQUINAS calculando *L* que consegue um melhor resultado (ponto E) que ultrapassou 66%. A rotina MÁQUINAS com *L* fixo conseguiu, para determinados valores de *L*, ultrapassar 66% e obteve como melhor resultado um valor acima de 68% (ponto C). A RBI atingiu a melhor *%PEP* (ponto A) entre todos os métodos com 69,69%. Além disso, A RBI obteve a melhor *%PEP* com o menor *CMT* associado em relação as melhores *%PEP* obtidas pelos demais métodos.

A Tabela 4.21 apresenta os valores dos melhores resultados, representados pelos pontos A, B, C, D, E e F na Figura 4.17. Os números apresentam uma proximidade entre os valores obtidos pelos três métodos para a *%PEP* (pontos A, C e E) e a superioridade do resultado encontrado pela RBI no *CMT* (ponto B).

Tabela 4.21 – Comparação dos melhores resultados em *I2*

Ponto	*(%**PEP**;CMT)*	*#It*	*TCM*	*TCT*	Ponto	*(%PEP;**CMT**)*	*#It*	*TCM*	*TCT*
A	(**69,69**;29,71)	21	3,33"	1'10"	B	(57,97;**23,55**)	21	4,14"	1'27"
C	(**68,63**;31,77)	30	2,56"	1'17"	D	(58,68;**26,84**)	29	3,34"	1'37"
E	(**66,12**;32,60)	26	3,96"	1'43"	F	(63,83;**27,83**)	26	3,88"	1'41"

A coluna com o *TCM* mostra que a rotina MÁQUINAS com *L* fixo leva em média 2,56" para estabelecer as seqüências em cada iteração quando obtém o maior valor de *%PEP* (68,63%). A rotina MÁQUINAS apresenta *TCM*=3,96" que está próximo ao valor obtido por RBI que leva 3,33" para executar cada iteração. A RBI continua sendo mais eficiente no tempo total gasto para resolver toda a instância *I2* com *TCT*=1'10" para obter o maior valor de *%PEP* (69,69%). Estes valores na rotina MÁQUINAS com *L* fixo (*TCT*=1'17") ou

rotina MÁQUINAS calculando L (TCT=1'43") são maiores que os de RBI, pois executam um $\#It$ maior que RBI.

Na execução que retorna o melhor CMT, as seqüências são estabelecidas em média mais rapidamente tanto pela rotina MÁQUINAS com L fixo (TCM=3,34") como pela rotina MÁQUINAS calculando L (TCM=3,88"). RBI demora mais na construção das seqüências (TCM=4,14") já que atribui todos os produtos que chegam às máquinas, porém, acaba estabelecendo a seqüência final mais rapidamente (TCT=1'27") do que os demais métodos.

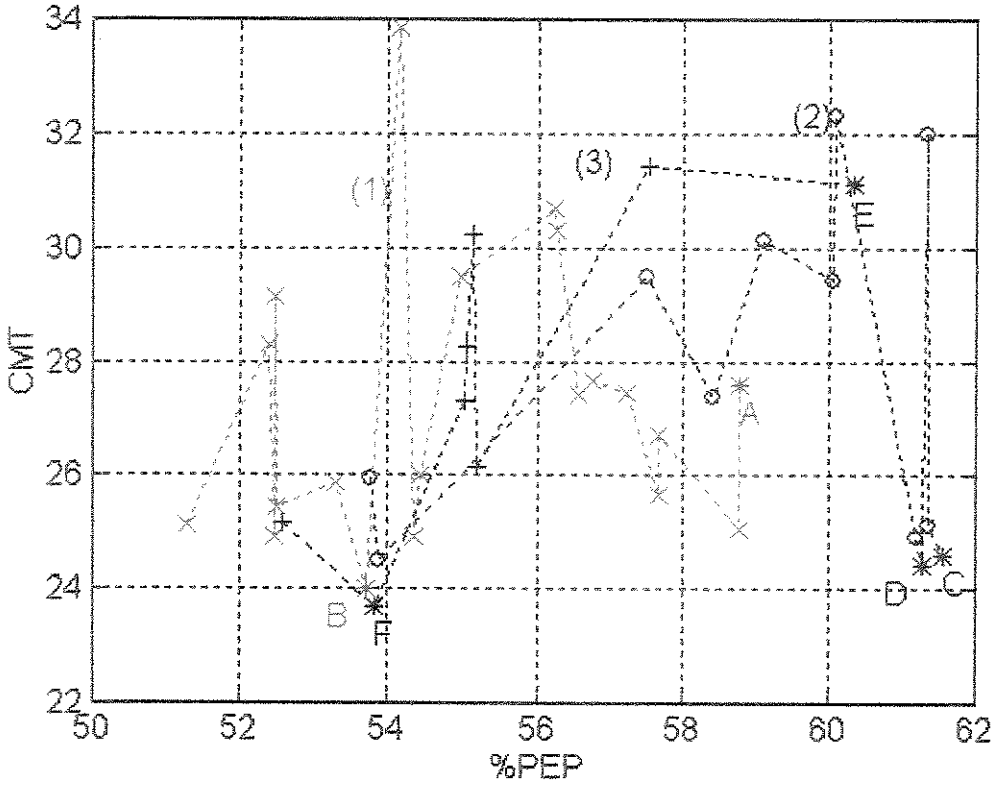


Figura 4.18 – Resultados fornecidos pela rotina MÁQUINAS e RBI em *I3*.

A Figura 4.18 apresenta o desempenho das três formas de resolução para a instância *I3*, onde o tracejado (1) liga os pontos da RBI, o tracejado (2) liga os pontos da rotina MÁQUINAS usando L fixo e o tracejado (3) liga os pontos da rotina MÁQUINAS calculando L. A instância *I3* apresenta o mesmo NTP=80 que *I2*, porém estes produtos chegam ao sistema de produção com NTP/NA=8. Assim, o desempenho acima retrata a reação dos métodos em um cenário mais dinâmico que o anterior.

Todos os métodos apresentam em *I3* uma redução na *%PEP* que não ultrapassa 62% em nenhum dos resultados obtidos. Porém, o maior destaque está na existência de apenas dois valores de *%PEP* obtidos por RBI que ficaram acima de 58%. Estes valores ultrapassam 58%, mas não chegam a 60% de produtos entregues no prazo. O melhor resultado de *%PEP* (ponto E) apresentado pela rotina MÁQUINAS calculando L supera RBI (ponto A),

ultrapassando 60% de *%PEP*. O melhor valor entre todos os método para *%PEP* é obtido pela rotina MÁQUINAS com *L* fixo que ficou próximo de 62% (ponto C).

A maioria dos valores obtidos pela rotina MÁQUINAS, calculando *L* e com *L* fixo, apresentam *CMT* entre 26 e 32. A RBI consegue manter a maioria de seus *CMT* entre 24 e 28. O melhor *CMT* encontrado entre os três métodos ficou abaixo de 24 e foi alcançado pela rotina MÁQUINAS calculando *L* (ponto F). Porém, os melhores *CMT* obtidos pelas demais rotinas encontram-se próximos de 24 (pontos D, C e F).

Apesar do melhor valor em termos de *CMT* ocorrer com a rotina MÁQUINAS calculando *L*, o melhor resultado em termos de *%PEP* obtido por MÁQUINAS com *L* fixo releva-se como solução de compromisso entre *%PEP* e *CMT*. Isto é demonstrado pelo ponto C na Figura 4.18, pois o *CMT* correspondente apresenta um valor próximo a 24.

A Tabela 4.22 contém os valores relacionados aos melhores resultados de *I3*. Conforme mencionado, a melhor *%PEP*=61,57% (ponto C) não ultrapassa 62% e foi obtida pela rotina MÁQUINAS com *L* fixo. O valor da *%PEP* nos três casos não estão afastados e a proximidade maior destes valores ocorre entre a rotina MÁQUINAS calculando *L* (ponto E com *%PEP*= 60,34) e a rotina MÁQUINAS com *L* fixo.

Tabela 4.22 – Comparação dos melhores resultados em *I3*

Ponto	*(%**PEP**;CMT)*	*#It*	*TCM*	*TCT*	Ponto	*(%PEP;**CMT**)*	*#It*	*TCM*	*TCT*
A	(**58,79**;25,05)	11	7,81"	1'26"	B	(53,70;**24,00**)	11	5,18"	57"
C	(**61,57**;24,62)	17	5"	1'25"	C	(61,28;**24,44**)	17	5,05"	1'26"
E	(**60,34**;31,10)	22	7,18"	2'38"	F	(53,82;**23,69**)	22	5,72"	2'06"

Os valores dos melhores *CMT* também são próximos nos três métodos de resolução, onde o menor valor é encontrado por MÁQUINAS calculando *L* com *CMT*=23,69. A solução fornecida pelo ponto C, em MÁQUINAS com *L* fixo, apresenta o melhor compromisso entre *%PEP* e *CMT*, se comparada as demais soluções apresentadas na Tabela 4.22. Esta solução de compromisso também estabelece as seqüências de produção a cada iteração com o menor *TCM*=5" e *TCT*=1'25". O tempo computacional gasto pelos três métodos para obter as melhores soluções, de forma geral, estão próximos. A RBI apresenta menor *TCT*=57" e MÁQUINAS calculando *L* o menor *TCM*=1'26" na obtenção dos menores *CMT*.

4.7.5.3 – Resultados da RBI relativos às instâncias I4 e I5

A Figura 4.19 compara os resultados retornados por RBI quando soluciona *I4* e *I5*. As instâncias *I4* e *I5* apresentam um *NTP*=152, onde em *I4* ocorrem 20 atualizações nos dados e em *I5* ocorrem 10 atualizações.

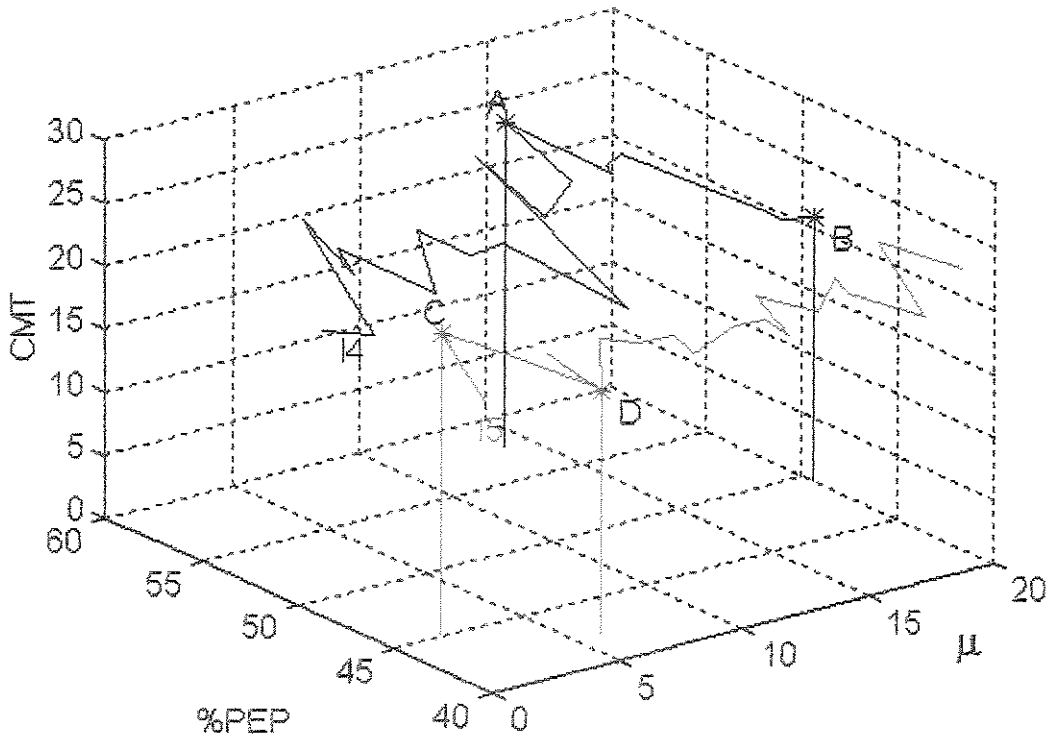


Figura 4.19 – Curvas de desempenho da RBI nas instâncias *I4* e *I5*

Nas instâncias *I4* e *I5* é verificado um comportamento das curvas de resultados semelhante ao ocorrido com *I2* e *I3*. As alturas semelhantes das curvas de *I4* e *I5* revelam um desempenho parecido em termos de *CMT*, porém a dissociação das mesmas mostra a diferença de desempenho em termos de *%PEP*.

Os valores do *CMT* estão entre 20 e 25 tanto na instância *I4* quanto em *I5*. As *%PEP* obtidas concentram-se entre 50 e 55% em *I4* enquanto em *I5* aparecem numa faixa mais reduzida com valores entre 40 e 45%. A dificuldade ao se passar de *NTP/NA*=7,6 em *I4* para *NTP/NA*=15,2 em *I5* (Tabela 4.1), devido ao maior volume de produtos chegando em média a cada iteração, aparece no menor valor da *%PEP* obtido por RBI na instância *I5*.

A Tabela 4.23 apresenta os valores correspondentes aos pontos da Figura 4.19. Os pontos A e C refletem a separação existente entre os resultados obtidos para *%PEP* em *I4* (*%PEP*=57,27%) e *I5* (*%PEP*=45,27%). A variação entre os melhores resultados de *%PEP* entre *I4* e *I5* ultrapassa 10% nos pontos A e C, o mesmo ocorrendo com os valores da *%PEP* apresentados em B (*%PEP*=49,36%) e D (*%PEP*=42,28%). Por outro lado, o valor dos melhores *CMT* em *I4* (*CMT*=20,87) no ponto B e *I5* (*%PEP*=19,31) no ponto D exemplificam o desempenho semelhante em termos de *CMT*, conforme foi observado pela altura semelhante nas curvas de desempenho na Figura 4.19.

Tabela 4.23 – Melhores resultados obtidos para *I4* e *I5*

Inst	*Ponto*	*(%PEP;CMT;μ)*	*Ponto*	*(%PEP;CMT;μ)*	*Δ%PEP*	*ΔCMT*
I4	A	(**57,27**;23,79;14)	B	(49,36;**20,87**;20)	7,91	2,92
I5	C	(**45,27**;23,79,2)	D	(42,28;**19,31**;6)	2,99	4,48

O melhor *CMT*=20,87 em *I4* é obtido com uma redução de *ΔCMT*=2,92 e uma queda de *Δ%PEP*=7,91% em relação ao resultado da melhor *%PEP*=57,27%. Na instância *I5* há uma queda menor na *Δ%PEP*=2,99% para se obter o menor *CMT*=19,31 com *ΔCMT*=4,48.

4.7.5.4 – Resultados de MÁQUINAS x RBI em I4 e I5

A Figura 4.20 compara o desempenho da rotina MÁQUINAS e RBI na resolução de *I4*. A linha tracejada (1) liga os pontos obtidos por RBI, a linha (2) os pontos da rotina MÁQUINAS com *L* fixo e a linha (3) liga os pontos da rotina MÁQUINAS calculando *L*. Alguns pontos apresentados em MÁQUINAS calculando *L* e com *L* fixo foram obtidos sem que todos os produtos tivessem sido processados. Em RBI os resultados foram obtidos com o processamento de todos os 152 produtos.

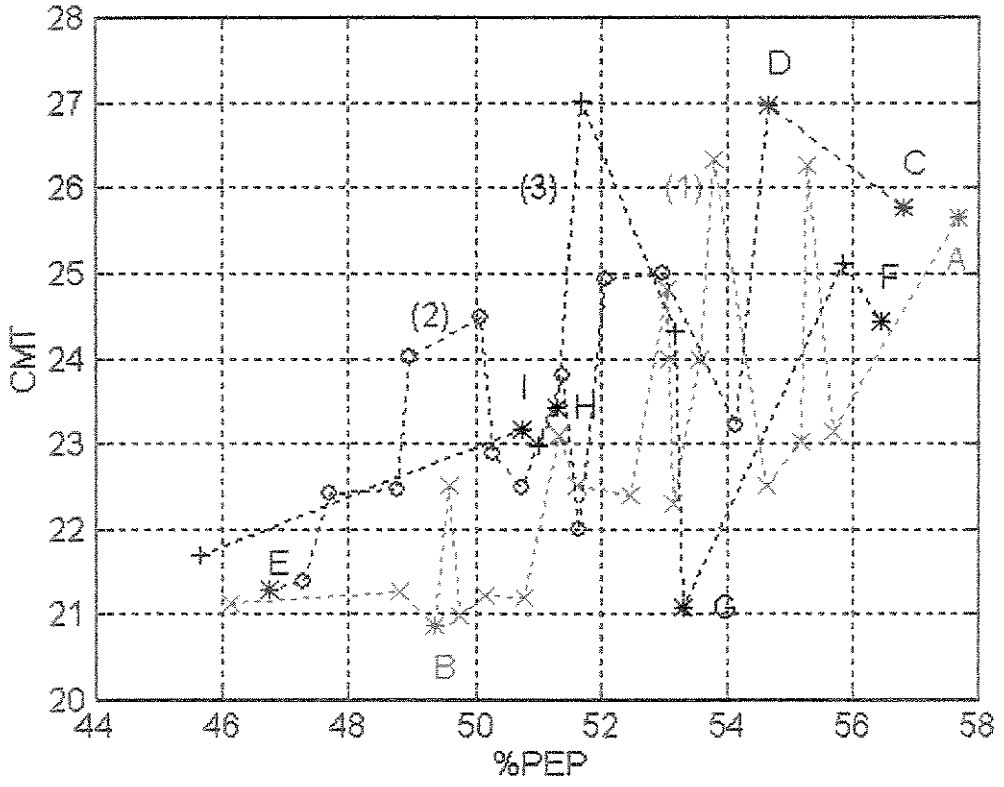


Figura 4.20 – Resultados fornecidos por MÁQUINAS e RBI em *I4*.

Os melhores resultados obtidos nos três métodos para a *%PEP* estão representados pelos pontos A, C e F, alcançando valores entre 56 e 58%. O ponto D revela um segundo valor

para a *%PEP* em *I4* obtido com a entrega de todos os produtos, pois a *%PEP* em C é alcançada pela rotina MÁQUINAS com *L* fixo deixando de processar 6 produtos.

O desempenho das rotinas em termos da *%PEP* deve ser analisado juntamente com o número de produtos processados, principalmente se estas instâncias apresentam um número elevado de produtos. Isto ocorre porque, conforme já mencionado, a medida de desempenho *%PEP* (Equação 4.26) é obtida no universo de produtos que efetivamente foram atribuídos às máquinas. Não entra no cálculo da *%PEP* os produtos que não foram atribuídos, ou seja, aqueles cuja data de entrega passou sem que tivessem sido processados.

Os valores do *CMT* obtidos pelas três rotinas concentram-se entre 22 e 25. A RBI obtém cinco pontos com *CMT* entre 20 e 21, próximos do melhor resultado alcançado que ficou abaixo de 20 (ponto B). A rotina MÁQUINAS calculando *L* apresenta o *CMT* que mais se aproxima do melhor *CMT* da RBI, enquanto a rotina MÁQUINAS com *L* fixo apresenta apenas dois pontos com *CMT* abaixo de 22.

A Tabela 4.24 lista os melhores resultados de *I4*, levando em conta o número de produtos processados. Os pontos A e B indicam, respectivamente, a melhor *%PEP* e *CMT* da RBI que alcança estes resultados realizando o processamento de todos os produtos. Os pontos C, D e E indicam os melhores resultados da rotina MÁQUINAS com *L* fixo.

Nesta rotina, o ponto C indica a melhor *%PEP*=56,81%, obtida deixando de processar 6 produtos. Os pontos D e E representam, respectivamente, a melhor *%PEP*=54,65% e *CMT*=21,29, obtidos juntamente com o processamento de todos os produtos. Para obter a melhor *%PEP*=54,65% no ponto D ocorre uma queda de 2,16% em relação a *%PEP*=56,81% no ponto C.

Tabela 4.24 – Comparação dos melhores resultados em *I4*

Ponto	*(%**PEP**;CMT)*	*#It*	*TCM*	*TCT*	Ponto	*(%PEP;**CMT**)*	*#It*	*TCM*	*TCT*
A	(**57,27**;23,79)	21	37,71"	13'12"	*B*	(49,36;**20,87**)	21	44,38"	15'32"
C	(**56,81**;25,24)	30	19,5"	9'45"	-	-	-	-	-
D	(**54,65**;26,96)	31	28,61"	14'47"	E	(46,74;**21,29**)	28	30,03"	14'01"
F	(**56,45**;24,44)	30	26,73"	13'22"	G	(53,38;**21,08**)	30	49,19"	17'13"
H	(**51,26**;23,43)	31	36,70"	18'58"	I	(50,72;**23,17**)	32	28,56"	15'14"

A rotina MÁQUINAS calculando *L* alcança a melhor *%PEP*=56,45% no ponto F com o não processamento de 5 produtos e o melhor *CMT*=21,08 no ponto G deixando de processar 2 produtos. Os melhores resultados de *%PEP*=51,26% e *CMT*=23,17 com o processamento de todos os produtos estão representados na Tabela 4.24 e na Figura 4.20 pelos pontos H e I. Os valores nestes pontos revelam uma queda de 5,19% na *%PEP*=51,26% e um aumento de 2,09 no *CMT*=21,08.

A RBI fornece os melhores resultados entre todos os métodos com *%PEP*=57,27% (ponto A) e *CMT*=20,87 (ponto B), realizando o processamento de todos os produtos. Os melhores

resultados da rotina MÁQUINAS calculando L e usando L fixo, quando deixam de processar alguns produtos (pontos C, F e G), apresentam-se próximos daqueles obtidos por RBI. Porém, quando a rotina MÁQUINAS realiza a entrega de todos os produtos, a diferença entre os melhores resultados obtidos usando L fixo (pontos D e E) e calculando L (pontos H e I) aumenta em relação aos melhores valores da RBI (pontos A e B).

Os menores *TCM*=19,5" e *TCT*=9'45", utilizados para obter a melhor *%PEP*, pertencem a rotina MÁQUINAS com L fixo quando deixa de processar 6 produtos (ponto C na Tabela 4.24). Porém, considerando-se o processamento de todos os produtos, o menor *TCM*=28,61" é obtido pela rotina MÁQUINAS com L fixo (ponto D) e o menor *TCT*=13'12" é da RBI (ponto A). Para alcançar o menor *CMT* juntamente com o processamento de todos os produtos, a rotina MÁQUINAS calculando L gasta um *TCM*=28,56" e a rotina MÁQUINAS com L fixo leva um *TCT*=14'01".

A Figura 4.21 apresenta o desempenho dos métodos na resolução de *I5*. O intervalo de variação da *%PEP* apresenta uma redução em relação ao apresentado na Figura 4.20. Esta redução fica em torno de 10%, pois o maior valor possível de *%PEP* não ultrapassa 48% contra 58% em *I4*. No eixo das abscissas, o *CMT* também varia em um intervalo menor que em *I4*, não ultrapassando 25.

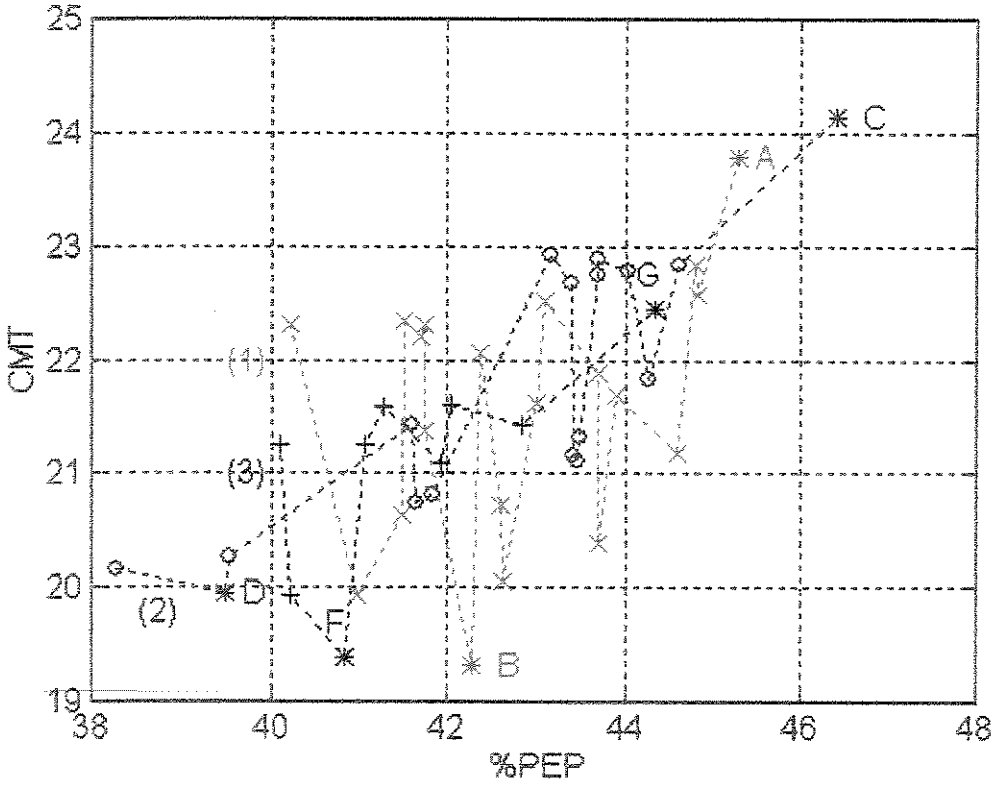


Figura 4.21 – Resultados fornecidos por MÁQUINAS e RBI em *I5*.

Todos dos métodos retornam *%PEP* mais concentradas entre 40 e 44%, enquanto o *CMT* concentra valores entre 20 e 23. Desta forma, constata-se em todos os métodos que um aumento no *NTP/NA*=15,2 em *I5* contra *NTP/NA*=7,6 em *I4* (Tabela 4.1) provoca redução na *%PEP* devido o aumento no volume de produtos chegando a cada iteração.

Todos os melhores resultados para *CMT* ficam abaixo de 20 (pontos B, F e D) onde o melhor desempenho é obtido pela RBI (ponto B na Figura 4.21). As melhores *%PEP* da RBI e MÁQUINAS calculando *L* são próximas e estão acima de 44%, porém o melhor desempenho foi de MÁQUINAS com *L* fixo que ultrapassa 46% dos produtos entregues no prazo.

A Tabela 4.25 apresenta os melhores resultados assinalados pelos pontos de A, B, C, D e E na Figura 4.21. Os dados da Tabela 4.25 revelam que os melhores resultados apresentam valores próximos nos três métodos, tanto em relação a *%PEP* como no *CMT*. A melhor *%PEP* entre todos os métodos é 46,41%, obtida pela rotina MÁQUINAS com *L* fixo (ponto C).

A rotina MÁQUINAS calculando *L* e RBI alcançam, respectivamente, 45,27% e 44,32% como melhor valor da *%PEP* (pontos A e E). O melhor resultado do *CMT* é obtido por RBI com *CMT*=19,31 (ponto B), porém a rotina MÁQUINAS calculando *L* obtém *CMT*=19,38 (ponto F) e a rotina MÁQUINAS com *L* fixo alcança *CMT*=19,94 (ponto D).

Tabela 4.25 – Comparação dos melhores resultados em *I5*

Ponto	*(%**PEP**;CMT)*	*#I*	*TCM*	*TCT*	Ponto	*(%PEP;**CMT**)*	*#I*	TCM	*TCT*
A	(**45,27**;23,79)	21	1'10"	12'59"	B	(42,28;**19,31**)	11	1'06"	12'05"
C	(**46,41**;24,14)	21	1'03"	22'01'	*D*	(39,48;**19,94**)	28	40,07"	19'01"
E	(**44,32**;22,44)	24	56,70"	22'41'	*F*	(40,84;**19,38**)	23	51,21"	19'38"

Para alcançar a melhor *%PEP*, a rotina MÁQUINAS calculando *L* gasta o menor *TCM*=56,70" enquanto RBI obtém o menor *TCT*=12'59". O melhor *CMT* é obtido no menor *TCM*=40,04" pela rotina MÁQUINAS com *L* fixo e menor *TCT*=12'05" pela RBI.

4.7.6 – Comparações dos Resultados de MÁQUINAS x RBI: 20 Instâncias

Apresentamos a seguir uma análise do desempenho destes três métodos em 20 outras instâncias. Estas instâncias serão divididas em dois grupos de 10 instâncias, onde no primeiro grupo estarão aquelas cujos novos produtos chegam ao sistema de produção em 20 sucessivas atualizações nos dados. No segundo grupo de instâncias estarão aquelas cujos novos produtos chegam ao sistema de produção através de 10 sucessivas atualizações.

A Figura 4.22 apresenta as melhores *%PEP* obtidas pelos três métodos nas 10 instâncias onde em cada uma ocorre 20 atualizações nos dados. No eixo das abscissas está a escala de variação do número total de produtos em cada instância e o eixo das ordenados apresenta as melhores *%PEP* obtidas. O símbolo "o" indica os melhores resultados de RBI na resolução

de cada instância avaliada, o símbolo "*" indica o mesmo em relação a rotina MÁQUINAS com L fixo e o símbolo "+" em relação aos resultados da rotina MÁQUINAS calculando L.

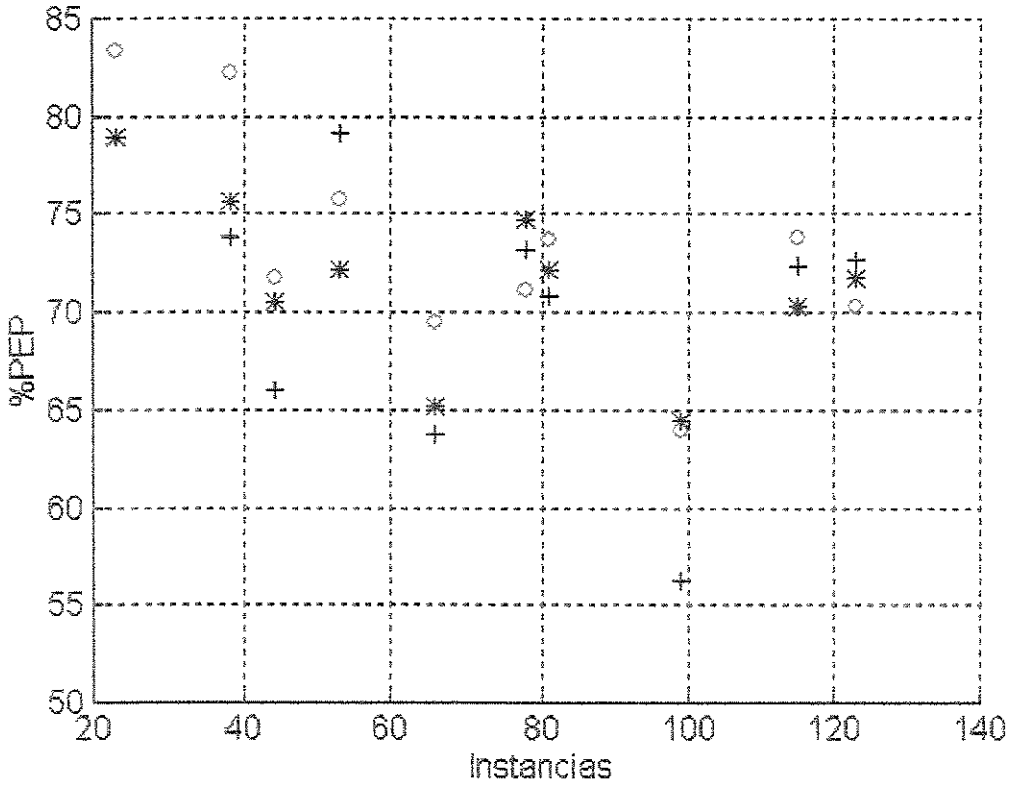


Figura 4.22 – Melhores resultados de *%PEP* em 10 instâncias com 20 atualizações

A *%PEP* decresce quando aumenta o número de produtos total das instâncias, demonstrando a dificuldade encontrada pelos três métodos em cumprir os prazos de entrega. A RBI obteve a melhor *%PEP* em 6 instâncias e a rotina MÁQUINA calculando L e usando L fixo em 2 instâncias cada.

A Figura 4.23 apresenta os melhores *CMT* obtidos pelos três métodos na resolução das 10 instâncias que sofreram 20 atualizações. Observa-se que os *CMT* diminuem a medida que aumenta o número de produtos, ou seja, maior volume de produtos permite estabelecer linhas de produção com menores *CMT*.

Isto ocorre porque são estabelecidas seqüências de produção de maior tamanho, desta forma, existe mais chance de inserir um novo produto e realizar resequênciamentos que, ao alterarem as posições dos mesmos, estabeleçam uma seqüência final com menor *CMT* envolvido.

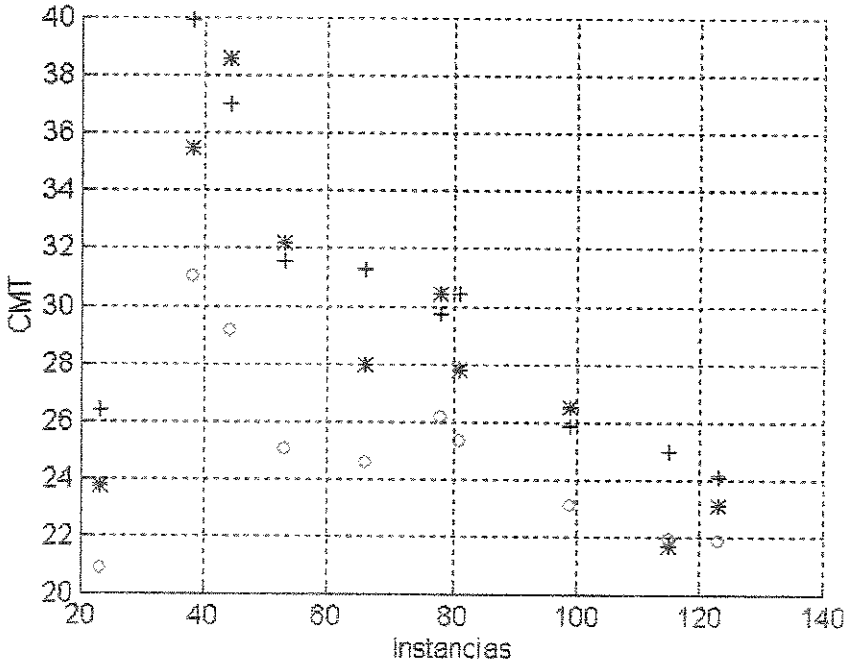


Figura 4.23 – Melhores resultados de *CMT* de 10 instâncias com 20 atualizações

Os três métodos apresentam maior diferença nos melhores resultados obtidos para os *CMT* nas quatro instâncias com *NTP* entre 20 e 60. Nas demais instâncias, estes valores tornam-se mais próximos a medida que aumenta o *NTP*. Os menores *CMT* são fornecidos pela RBI, exceto em uma instância com *NTP* entre 100 e 120, cujo melhor resultado é fornecido pela rotina MÁQUINA com *L* fixo.

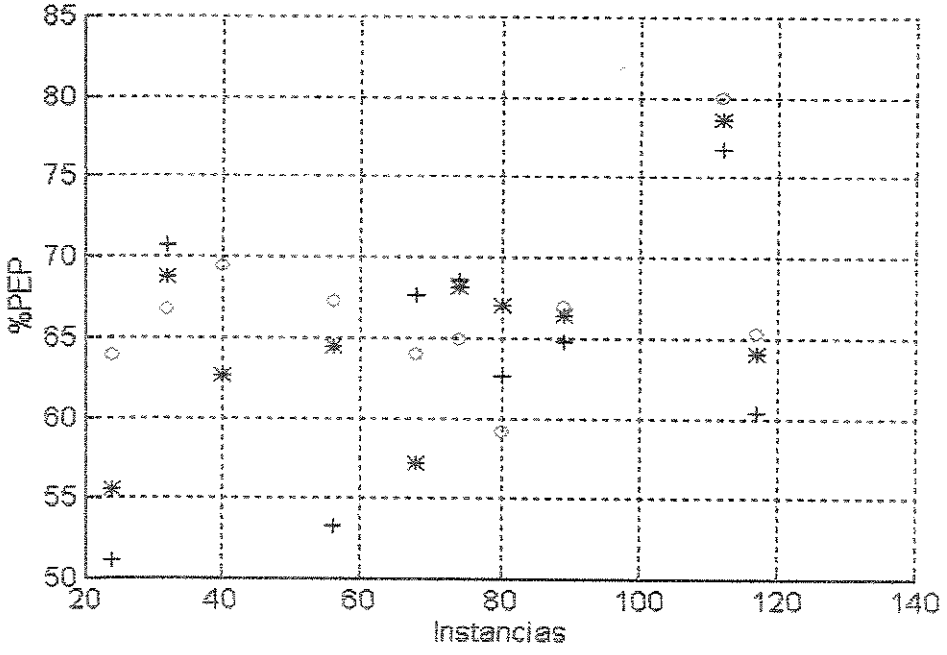


Figura 4.24 – Melhores resultados de *%PEP* de 10 instâncias com 10 atualizações

A Figura 4.24 apresenta os resultados obtidos pelos três métodos na resolução de 10 instâncias com 10 atualizações. O objetivo é avaliar instâncias com *NTP* semelhantes às anteriores, mas com atualizações ocorrendo em menor número de modo que aumente o volume de produtos chegando a cada atualização nos dados.

Exceto por uma instância com *NTP* entre 100 e 120 onde a *%PEP* superou 80%, as demais instâncias apresentam menor *%PEP* que na Figura 4.22. Conforme foi analisado nas cinco instâncias consideradas inicialmente, a dificuldade de gerenciar problemas que apresentem maior *NTP/NA* ocasiona queda na *%PEP*. Desta forma, os três métodos apresentam dificuldade em realizar as entregas no prazo quando mais produtos tornam-se disponíveis ao mesmo tempo.

A RBI supera os demais métodos em 6 instâncias, enquanto MÁQUINAS calculando L alcança a melhor *%PEP* em três instâncias, apresentando um desempenho melhor que a rotina MÁQUINAS com L fixo. Esta rotina obtém o melhor resultado em apenas uma instância com *NTP*=80 e praticamente empata com a rotina MÁQUINAS calculando L em uma instância com *NTP* próximo de 80. Porém, a rotina MÁQUINAS com L fixo obtém na maioria das instâncias o segundo melhor resultado.

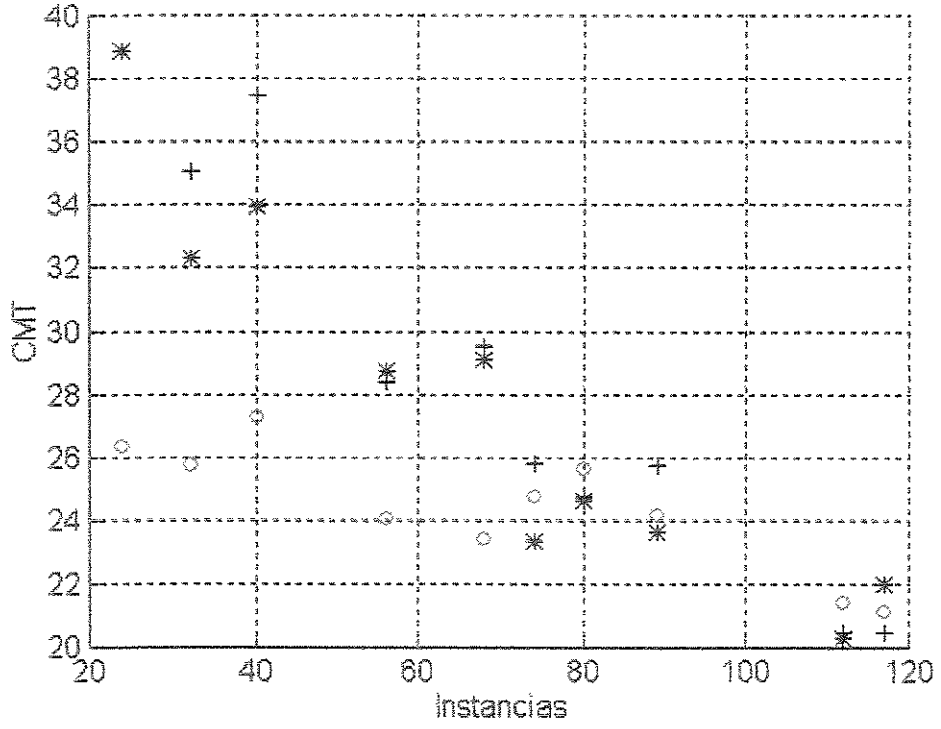


Figura 4.25 – Melhores resultados do *CMT* para 10 instâncias com 10 atualizações

A Figura 4.25 apresenta os melhores *CMT* obtidos pelo métodos nas 10 instâncias que sofrem 10 atualizações. Novamente a proximidade dos valores obtidos é marcante para as instâncias com *NTP* maior que 60 produtos. A RBI consegue supremacia nos resultados para instâncias com *NTP* menor que 60, obtendo o menor *CMT* em apenas uma instância

com *NTP* entre 60 e 80. Nas instâncias com *NTP* maior que 80, as rotinas MÁQUINAS com *L* fixo e calculando *L* obtêm os menores *CMT*.

A rotina MÁQUINA com *L* fixo consegue melhores resultados em 4 instâncias. A rotina MÁQUINAS calculando *L* obtém o menor *CMT* na instância de maior dimensão. Os resultados fornecidos pelos três métodos nas duas instâncias com *NTP* entre 100 e 120 estão bastante próximos, concentrando-se entre os valores 20 e 22.

A Tabela 4.26 apresenta os métodos e a quantidade de vezes em que alcançaram os melhores resultados nas 20 instâncias avaliadas nesta seção. A RBI obteve o melhor resultado em termos de *%PEP* em 12 instâncias e o melhor *CMT* em 14 instâncias. A rotina MÁQUINAS com *L* fixo alcançou o melhor resultado de *%PEP* em 3 instâncias e o melhor *CMT* em 5 instâncias. A rotina MÁQUINAS calculando *L* alcançou a melhor *%PEP* em 5 instâncias e apenas em 1 instância conseguiu o melhor *CMT*.

Tabela 4.26 –Quantidade de melhores resultados obtidos por cada método

Métodos	%PEP	CMT
RBI	12	14
Máquinas com L fixo	3	5
Máquinas com L calculado	5	1
Total	20	20

Os valores apresentados na Tabela 4.26 assim como os resultados apresentados nas cinco instâncias da seção anterior mostram a superioridade de desempenho da RBI. Esta rotina realiza uma busca mais ampla no espaço de soluções. Os reseqüênciamentos realizados pela heurística de busca em vizinhança e inserção, permitindo inclusive a troca de produtos entre as máquinas, podem ser destacados como aspectos fundamentais para o bom desempenho deste método. A rotina MÁQUINAS com *L* fixo e *L* calculado carecem de maior busca no espaço de soluções já que a heurística de busca em vizinhança adotada não permite reseqüênciamento dos produtos que envolvam troca dos mesmos entre as máquinas.

4.8 - Conclusões

Neste capítulo, foi definido um Problema Dinâmico de Programação de Máquinas Paralelas com Custo de Troca de Ferramentas Dependente da Seqüência e Restrições de Tempo. Este Problema Dinâmico de Programação (PDP) originou-se de um Problema Industrial envolvendo a fabricação de embalagens de vidro. Para criar as diversas instâncias do PDP considerado, um gerador aleatório de instâncias foi implementado.

A solução destas instâncias envolveu a implementação de dois métodos. O primeiro método foi uma adaptação do algoritmo MORSS para o PDP que passou a ser tratada como rotina

MÁQUINAS. Para estabelecer um critério de comparação com os resultados obtidos pela rotina MÁQUINAS, foi desenvolvido um segundo método que solucionasse o PDP percorrendo de forma mais ampla o espaço de soluções possíveis. Este método foi chamado Rotina de Busca e Inserção (RBI) porque otimiza as seqüências de produção através de heurísticas de busca em vizinhança e inserção.

Durante a adaptação do algoritmo MORSS foram introduzidas mudanças no método. A primeira foi a incorporação de uma heurística de busca em vizinhança. Esta heurística permitiu uma melhora no desempenho do algoritmo porque, toda vez que uma possível inserção do produto candidato nas máquinas fosse avaliada, ela realizava um reseqüênciamento da linha de produção.

Os resultados obtidos revelaram que, para um valor adequado de L, a rotina apresentava melhor resultado nas medidas de desempenho consideradas. A exemplo do que foi feito no capítulo 3, foi adicionado ao algoritmo um critério que permitiu o cálculo do tamanho do horizonte de tempo L a cada iteração da rotina. Desta forma, pretendia-se uma independência em relação ao fornecimento do valor de L pelo usuário.

Assim, foram obtidos e comparados resultados relativos à rotina MÁQUINAS utilizando um valor fixo para L, à rotina MÁQUINAS calculando o valor de L a cada iteração e à RBI. Todos os resultados apresentados não permitem afirmar um melhor ou pior desempenho para determinado método, pois não foi avaliado um número considerável de instâncias.

Nas instâncias avaliadas, a RBI apresenta o melhor desempenho na maioria dos casos, tanto na obtenção do menor *CMT* quanto na obtenção das maiores *%PEP*. A rotina MÁQUINAS calculando L e a rotina MÁQUINAS com L fixo conseguiram melhores resultados em algumas instâncias.

A rotina MÁQUINAS com L fixo demonstrou que esta adaptação do algoritmo MORSS foi capaz de obter melhores resultados em certas instâncias para determinados valores de L. Porém, esta dependência do parâmetro L em uma situação real torna-se problemática porque um bom desempenho do método dependeria do usuário ter o conhecimento de qual tamanho do horizonte de tempo deve usar.

A rotina MÁQUINAS calculando L torna-se uma alternativa a esta situação. Apesar da forma de cálculo de L utilizada neste trabalho ser bastante empírica, o comportamento desta rotina na resolução das diversas instâncias foi bastante animador. Afinal, esta rotina também conseguiu superar os demais métodos em algumas instâncias.

A RBI mostrou-se bastante eficiente na obtenção de bons resultados. Isto justifica-se porque tal método realiza uma busca pelo espaço de soluções muito maior que a rotina MÁQUINAS. As atribuições dos produtos às máquinas são realizadas pela RBI de forma aleatória e, a partir delas, a linha de produção estabelecida é otimizada. Diferentes resultados podem ser obtidos dependendo da atribuição inicial estabelecida e das

atualizações que ocorrem. Numa situação real, não seria possível prever qual atribuição aleatória permitiria estabelecer as melhores seqüências de produção.

Apesar desta forma de atribuição, este método utiliza heurísticas de inserção e busca em vizinhança que permitem avaliar diversos reseqüênciamentos. Estes reseqüênciamentos conseguem compensar a atribuição aleatória já que alteram a seqüência inicial obtendo outra com melhor medida de desempenho. Este procedimento justifica o bom desempenho desta rotina nas instâncias avaliadas. A incorporação de um critério mais inteligente de atribuição poderá ocasionar uma melhoria nos resultados da RBI.

Os resultados obtidos demonstram que o algoritmo MORSS adaptado para o contexto de um PDP necessita de uma busca mais eficiente pelo espaço de soluções. A incorporação de uma heurística de busca em vizinhança permitiu uma melhoria no desempenho do método se comparada à execução sem esta modificação. A dependência deste método em relação a escolha de um adequado valor de L também prejudica seu desempenho, pois não são todos os valores de L que permitem estabelecer as melhores seqüências de produção.

Por outro lado, o algoritmo MORSS sugere alternativas de como gerenciar problemas dinâmicos através da idéia do horizonte rolante e das funções utilidades de designação. Estas idéias podem ser melhor exploradas no contexto de um PDP, assim como em outras situações que envolvam problema com caráter dinâmico.

Uma possibilidade para trabalhos futuros seria a utilização de heurísticas que realizem buscas eficientes pelo espaço de soluções, principalmente, gastando pouco tempo computacional. A idéia do horizonte rolante pode ser aproveitada para gerar subproblemas a serem solucionados utilizando tais heurísticas.

Ao invés de solucionar um problema de designação e simular inserções de produtos candidatos, conforme ocorre no algoritmo MORSS, ou realizar as atribuições de forma aleatória, como na RBI, um critério mais eficiente de atribuição pode ser utilizado de acordo com as características do problema a ser resolvido.

CONCLUSÕES

Nesta dissertação, foi apresentado como os atuais avanços em tecnologia de informação permitiram que problemas considerados estáticos passassem a ser reformulados como problemas dinâmicos. Esta mudança no cenário de diversos problemas fez surgir a necessidade do desenvolvimento de novas metodologias. Estas metodologias devem ser capazes de solucionar problemas para os quais não se tem conhecimento prévio de todas as informações e que estão sujeitos a constantes atualizações nos dados.

Um método de resolução chamado algoritmo MORSS foi apresentado e utilizado na resolução de dois diferentes problemas dinâmicos. O primeiro problema estabelecido e resolvido foi um Problema de Roteamento Dinâmico de Veículos (PRDV) e o segundo trata-se de um Problema Dinâmico de Programação de Máquinas Paralelas com Custo de Troca Dependente da Seqüência e Restrições de Tempo. Assim, um primeiro resultado deste trabalho foi a adaptação do algoritmo MORSS para que solucionasse estes dois diferentes problemas dinâmicos.

As instâncias do PRDV proposto foram solucionadas de forma satisfatória pela adaptação do algoritmo MORSS. Estas instâncias envolveram uma quantidade elevada de veículos e cargas que tornaram-se disponíveis através de diversas atualizações. Nesta etapa, foi observada uma relação entre o tamanho do horizonte de tempo (L) utilizado e os melhores resultados obtidos. Assim, foram propostas duas formas de se calcular o valor de L que permitiram obter este valor a cada iteração do método, sem a interferência do usuário.

Os resultados obtidos utilizando L como parâmetro e calculando o mesmo a cada iteração estiveram bastante próximos nas instâncias consideradas. Nenhuma conclusão sobre o desempenho do método pode ser assumida já que não foi avaliado um número razoável de instâncias. Porém, os resultado obtidos foram satisfatórios, revelando um bom desempenho da adaptação realizada no método. A relação verificada entre os melhores resultados e o parâmetro L serve como motivação para um estudo mais elaborado de critérios que determinem o valor de L.

O Problema Dinâmico de Programação de Máquinas Paralelas com Custo de Troca Dependente da Seqüência e Restrições de Tempo exigiu maiores adaptações no algoritmo. Assim, neste trabalho, foi incluído no algoritmo MORSS uma heurística de busca em vizinhança responsável por otimizar as seqüências de produção obtidas a cada iteração. Também foram propostas novas Funções Utilidade de Designação capazes de avaliar as restrições relativas ao Custo de Troca.

A inclusão da heurística de busca em vizinhança mostrou-se bastante satisfatória, pois os resultados obtidos superaram aqueles obtidos quando o método foi executado sem tal heurística. Isto revela que o algoritmo MORSS carece de um passo em que execute uma

busca mais ampla pelo espaço de soluções. A avaliação do problema através do horizonte rolante e das cargas através das funções utilidade, assim como, a resolução do problema de designação não são suficientes para garantir que a atribuição realizada seja a melhor no decorrer das atualizações.

Neste problema, também foi verificada uma relação entre valor adotado para o parâmetro L e a obtenção de melhores medidas de desempenho. Uma forma de cálculo de L a cada iteração foi sugerida, permitindo modificar o método para que deixasse de receber este valor como um parâmetro fornecido pelo usuário.

A Rotina de Busca e Inserção (RBI) foi utilizada como método comparativo para os resultados obtidos usando a adaptação do algoritmo MORSS. Apesar de possuir um critério aleatório de atribuição dos produtos às máquinas, a RBI realiza reseqüênciamentos que incluiu a troca de produtos entre as máquinas. Desta forma, os resultados obtidos por RBI foram melhores na grande maioria das instâncias avaliadas.

As idéias fornecidas pelo algoritmo MORSS, como a avaliação do problema através de horizontes de tempo, podem ser úteis já que evitam a tomada de decisões que comprometam futuras atribuições. Por outro lado, uma busca eficiente pelo espaço de soluções poderá economizar tempo computacional e permitir a obtenção de reseqüênciamentos que estabeleçam melhores seqüências de produção. Desta forma, uma interseção da forma de gerenciamento proposta pelo algoritmo MORSS com eficientes heurísticas de busca e inserção poderão gerar métodos bastante adequados à resolução de problemas inseridos em um contexto dinâmico.

REFERÊNCIAS

Aguilera, L.M., Z. Binder, A. Gonthier e K.J. Heeramun (1992). Scheduling Problem with Changeover Costs in Industrial Applications. Symposium on Information Control Problems in Manufacturing Technology, INCON'92. Toronto, Canada.

Aguilera, L.M. (1993). Ordonnancement de Production avec Couts de Changements Dependant de la Sequence. *These de doctorat de l'Institut National Polytechinique de Grenoble*. Grenoble, France.

Aguilera, L.M., Z. Binder, J.J. Guimarães Ramos e S.H. Takara (1995). Parallel Machines Scheduling with Sequence-Dependent Changeover Costs in an Industrial Application. Em: *11º ISPE/IEE/IFAC International Conference on CAD/CAM, Robotics and Factories of the Future, CARS & FOF'95*. Proceedings pg. 703-708. Pereira, Colombia.

Aguilera L.M., Z. Binder, F. Hanada e J.J. Guimarães Ramos (1996). Non-Identical Parallel Machines Scheduling with Sequence-Dependente Changeover-Costs in an Industrial Application. CESA´96/IMACS/IEE-SMC.Proceedings pg. 559-564. Lile, France.

Aslidis, A. (1991). Minimization of Overstowage in Containership Operations. *Operational Research'90* **18**, pg. 457-471.

Baker, K.R. (1974). Introduction to Sequencing and Scheduling. Wiley.

Bell, W., L.M. Dalberto, M.L. Fisher, A.J. Greenfield, R. Jaikumar, P. Kedia, R.G. Macj e P.J. Prutzman (1983). Improving the Distrubution of Industrial Gases with an On-Line Computarized Routing and Scheduling Optimizer. *Interfaces* **13**, pg. 4-23.

Bertsimas, D.J. e G.V. Ryzin (1991). A Stochastic and Dynamic Vehicle Routing Problem in the Euclidean Plane. *Operations Reseach* **39**, No. 4, pg. 601-615.

Bertisimas, D.J. e G.V. Ryzin (1993). Stochastic and Dynamic Vehicle Routing in the Euclidean Plane with Multiple Capacited Vehicles. *Operations Research* **41**, pp. 60-75.

Blazewicz, J., K. Ecker, G. Schmidt e J. Weglarz (1993). *Scheduling in Computer and Manufacturing Systems*. Spring-Verlag, New York.

Chirstofides, N. (1985). Vehicle Routing. Em: *The Traveling Salesman Problem* (E.L. Lawler, J. K. Lenstra, A. H. G. Rinnooy Kan, D. B. Shmoys), pg. 431-449. John Wiley & Sons.

CPLEX Optimization. Version 3.0 (1994). *Using the CPLEX® Callable Library*, CPLEX Copyrights©1989-1994.

Dror, M., G. Laporte and P. Trudeau (1989). Vehicle Routing with Stochastic Demands: Properties and Solution Frameworks. *Transportation Science* **23**, pg. 166-176.

Evans, J.R. (1987). Structural Analysis of Local Search Heuristic in Combinatorial Optimization. Computers & Operations Research **14**, pg. 466-474.

França P.M., M. Gendreau, G. Laporte, F.M. Müller (1996). A Tabu Search Heuristic for the Multi Processor Scheduling Problem with Sequence Dependent Setup Times. *International Journal of Production Economics.*

Fisher, M.L., e R. Jaikumar (1981). A Generalized Assignment Heuristic for Vehicle Routing. *Networks* **11**, pg. 109-124.

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

Gonthier, A. (1990). De l'Optimization des Coûts de Changement de Fabrication vers l'Atelier Logiciel d'Ordonnancement (ALDO). Thèse de doctorat de l'Institut National Polytechnique de Grenoble.

Jaillet, P. (1991). Probabilistic Routing Problems in the Plane. *Operational Reseach'90* **29**, No. 10, pg. 675-688.

Kessous K., L.M. Aguilera, Z. Binder e H. Song (1994). A Decision Aid Approach to Solve Scheduling Problems in an Industrial Application.. IFIP/WG5.7'94. Porto Alegre, Brasil. Proceedings IFIP Transactions B-19, North Holland, ISSN 0926-5481, pg. 207-215.

Laporte, G. (1992). The Vehicle Routing Problem: An Overview of Exact and Approximate Algorithms. *European Journal of Operational Research* **59**, pg. 345-358.

Lygaard J. (1992). Dynamic Transportation Networks in Vehicle Routing and Scheduling. *Intefaces* **22**, pg. 45-55.

Morton, T.E e D.W. Pentico (1993). Heuristic Scheduling Systems: with Applications to Production Systems and Project Management. New York, J. Wiley.

Powell, W.B., P. Jaillet e A. Odoni (1995). Stochastic and Dynamic Networks and Routing. Em: *Network Routing*, pg. 141-295. Elsevier Science, Amsterdam.

Psaraftis, H.N., J.B. Orlin, D. Bienstock e P.M. Thompson (1985). Analysis and Solution Algorithms of Sealift Routing and Scheduling Problems: Final Report. *Working Paper* No. 1700-85, Sloan School of Management, MIT.

Psaraftis, H.N. (1988). Dynamic Vehicle Routing Problems. Em: *Vehicle Routing: Methods and Studies* (B. Golden e A. Assad), pg.223-248. Elsevier Science, North-Holland.

Psaraftis H.N. (1995). Dynamic Vehicle Routing: Status and Prospects. *Annals of Operations Reseach* **61**, pg. 143-164.

Rego, C. e C. Roucairol (1995). Using Tabu Search for Solving a Dynamic Multi-Terminal Truck Dispatching Problem. *European Journal of Operational Research* **83**, pg. 411-429.

Ronen, D. (1986). Short Term Scheduling of Vessels for Shipping Bulk or Semi-Bulk Commodities Originating in a Single Area. *Operations Reseach* **34**, pg. 164-173.

Thompson, P.M. e H.N. Psaraftis (1993). Cyclic Transfer Algorithms for Multivehicle Routing and Scheduling Problems. *Operations Research* **41**, pg. 935-946.

Anexo1
Critérios de Atribuição de RBI e Máquinas

1.1 – Introdução

Neste anexo, os critérios de atribuição dos produtos às máquinas realizada pela rotina MÁQUINAS e RBI será apresentada através de diagramas. O objetivo é enfatizar a diferença existente entre as estratégias de atribuições dos produtos e otimização das seqüências de produção em cada método.

Os resultados que serão apresentados referem-se à instância *I3* quando o melhor resultado é obtido para a medida de desempenho *%PEP*, tanto pela rotina MÁQUINAS com L fixo quanto por RBI. A instância *I3* foi escolhida porque sua resolução envolve uma quantidade razoável de produtos (*NTP*=80) que chegam em um total de 10 atualizações nos dados.

1.2 Atribuições Realizadas

As atribuições realizadas por cada método serão apresentadas através de diagramas que contém as seqüências de produção estabelecidas a cada iteração dos métodos. A primeira seqüência apresentada será formada a partir do conjunto de produtos inicialmente disponíveis. Em seguida, as seqüências relativas a quatro atualizações nos dados também serão apresentadas.

Deve-se relembrar que não está sendo considerada uma unidade de tempo específica, ou seja, estaremos considerando o tempo simplesmente em Unidade de Tempo (UT) que poderá ser minutos, segundos, etc.

A Tabela A1.1 contém algumas informações relativas aos produtos inicialmente disponíveis, onde a coluna *Prod* contém a enumeração de cada um destes produtos. As colunas do tipo p_{ij} apresentam o tempo de processamento do produto i na máquina j, r_j representa o tempo de chegada e d_j a data de entrega do produto j.

Tabela A1.1 – Dados dos produtos disponíveis inicialmente

Prod	p_{1j}	p_{2j}	p_{3j}	r_j	d_j
1	11	11	7	9	49
2	6	14	10	5	33
3	13	7	7	7	47
4	6	14	6	3	31

Por exemplo, o produto 4 terá um tempo de processamento igual a 6UT na máquina 1, 14UT na máquina 2 e 10UT na máquina 3. Este produto torna-se disponível ao sistema de produção quanto o tempo vale 3UT e sua data de entrega ocorrerá no instante 31UT.

A Figura A1.1 e a Tabela A1.2 apresentam os resultados obtidos pela rotina Máquinas com L fixo envolvendo os dados inicialmente disponíveis. Na Figura A1.1 o tamanho de cada retângulo corresponde ao tempo de processamento do produto e o número assinalado corresponde ao produto na Tabela A1.2. Retângulos sem números correspondem a um período em que a máquina está parada. Na Tabela A1.2, a coluna C corresponde ao conjunto de cargas candidatas e PA o conjunto de produtos atribuídos. A coluna C_j corresponde ao tempo em que o produto termina de ser processado e d_j a correspondente data de entrega.

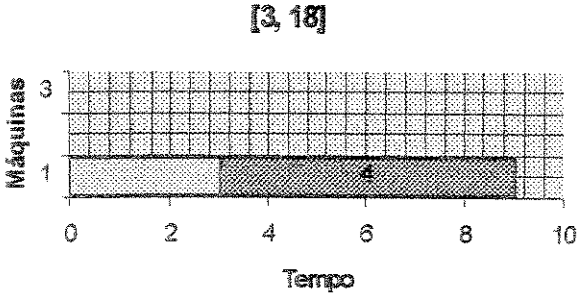


Figura A1.1 – Digrama correspondente ao intervalo [3, 18]

Tabela A1.2 – Atribuições em [3, 18]

C	PA	C_j	d_j
1	1		
2	2		
3	3		
4	4	9	31

Nesta iteração, a rotina atribuiu apenas o produto 4 à máquina 1, ou seja, de todos os produtos contidos no conjunto de cargas candidatas apenas 4 foi atribuído efetivamente. A Figura A1.2 apresenta as atribuições realizadas pela RBI. Esta rotina atribui aleatoriamente todos os produtos disponíveis às máquinas e, posteriormente, realiza uma otimização nas seqüências obtidas. A Tabela A1.3 lista os valores obtidos para o tempo final de processamento e as respectivas datas de entrega.

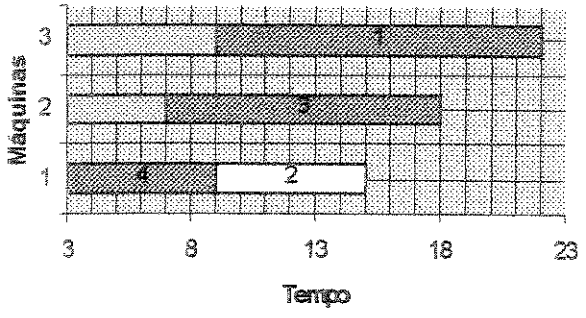


Figura A1.2 – Digrama correspondente a 1º iteração da RBI

Tabela A1.3 – Atribuições da 1° iteração

PA	C_j	d_j
1	22	49
2	15	33
3	18	47
4	9	31

Neste ponto, observa-se uma das principais diferenças entre as duas metodologias, pois enquanto MÁQUINAS realiza toda uma avaliação e designa apenas 1 produto à máquina 1, RBI já aloca todos os produtos disponíveis às três máquinas.

A Tabela A1.4 apresenta alguns dados correspondentes aos produtos que tornaram-se disponíveis na iteração seguinte dos dois métodos. Conforme foi explicado no capítulo 4, as atualizações nos dados da instância *I3* ocorrerão nas dez primeiras iterações dos dois métodos.

Tabela A1.4 – Dados dos produtos disponíveis inicialmente

Prod	p_{1j}	p_{2j}	p_{3j}	r_j	d_j
5	8	11	8	12	57
6	7	6	11	8	35
7	14	9	6	4	25

As atribuições obtidas pela rotina MÁQUINA com L fixo, quando avalia os produtos no horizonte de tempo [4, 19], estão na Figura A1.3 e Tabela A1.5.

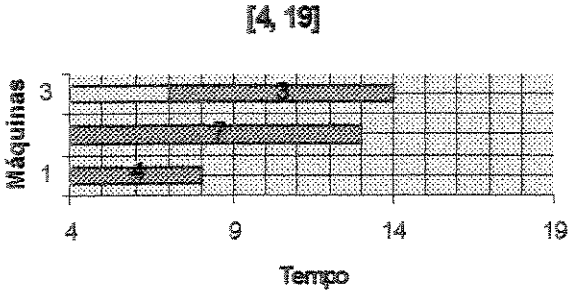


Figura A1.3 – Digrama correspondente ao intervalo [4,19]

Tabela A1.5 – Atribuições em [4, 19]

C	PA	C_j	d_j
1			
2			
3	3	11	47
5			
6			
7	7	13	25

Nesta iteração, a rotina atribui efetivamente os produtos candidatos 3 e 7 que são entregues antes do correspondente prazo. Os resultados obtidos pela RBI estão na Figura A1.4 e Tabela A1.6.

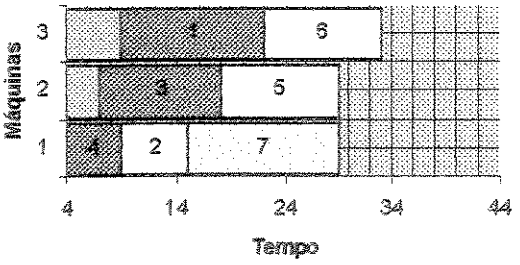


Figura A1.4 – Digrama correspondente a 2º iteração da RBI

Tabela A1.6 – Atribuições da 2° iteração

PA	C_j	d_j
5	29	57
6	33	35
7	29	25

Verifica-se um atraso de 2UT na atribuição do produto 7 à máquina 1 e um tempo final de processamento próximo do prazo na atribuição do produto 6 à máquina 3. A Tabela A1.7 apresenta os dados dos novos produtos na 3º iteração dos métodos.

Tabela A1.7 – Dados dos produtos disponíveis inicialmente

Prod	p_{1j}	p_{2j}	p_{3j}	r_j	d_j
8	6	9	6	11	89
9	9	8	11	5	77
10	8	13	8	5	65
11	11	14	9	11	95
12	12	9	6	5	89
13	7	8	9	9	81
14	8	13	10	7	85
15	13	12	5	9	69
16	12	9	8	11	95

A Figura A1.5 e a Tabela A1.8 apresentam as atribuições da rotina Máquinas.

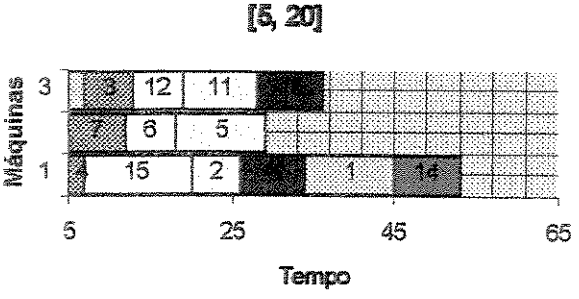


Figura A1.5 – Digrama correspondente ao intervalo [5, 20]

Tabela A1.8 – Atribuições em [5, 20]

C	*PA*	C_j	d_j
1			
2			
5			
6			
8			
9			
10	10	36	65
11	11	24	95
12	12	17	89
13			
14	14	55	85
15	15	22	69
16	16	25	95

Os resultados obtidos pela RBI na 3º iteração estão apresentado na Figura A1.6 e Tabela A1.9.

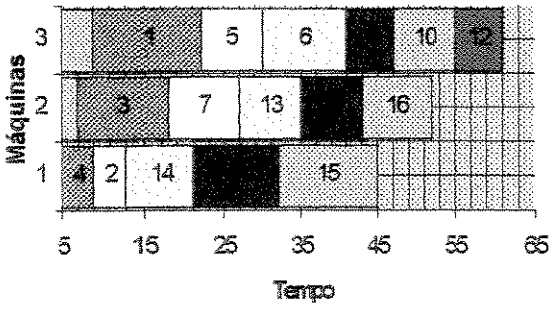


Figura A1.6 - Digrama correspondente a 3º iteração da RBI

Tabela A1.9 – Atribuições da 3° iteração

PA	C_j	d_j
8	39	89
9	46	77
10	47	65
11	34	95
12	53	89
13	37	81
14	23	85
15	45	69
16	55	95

Neste ponto, outra diferença existente nas metodologias pode ser observada. Enquanto a rotina Máquinas mantém os produtos nas máquinas para as quais foram atribuídos no decorrer das iterações, a RBI troca os produtos entre as máquinas. Por exemplo, o produto 5 e 7 na Figura A1.4 estão atribuídos, respectivamente, às máquinas 2 e 1 e na Figura A1.6 a RBI atribui o produto 5 à máquina 3 e o produto 7 à máquina 2.

Na quarta atualização, apenas o produto 17 torna-se disponível ao problema e a Tabela A1.10 apresenta os dados correspondentes.

Tabela A1.10 – Dados dos produtos disponíveis inicialmente

Prod	p_{1j}	p_{2j}	p_{3j}	rj	dj
17	10	11	12	12	28

A Figura A1.7 e Tabela A1.11 apresentam os resultados de MÁQUINAS.

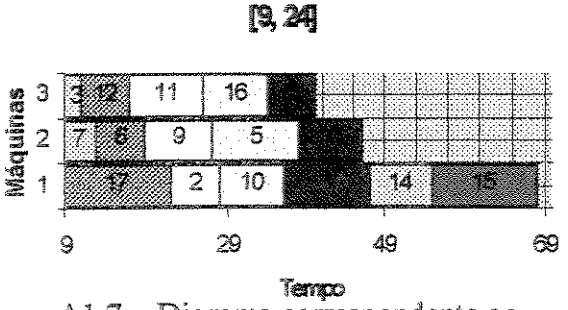


Figura A1.7 – Digrama correspondente ao intervalo [9, 24]

Tabela A1.11– Atribuições em [9, 24]

C	*PA*	C_j	d_j
8	8	40	89
9	9	27	77
13	13	40	81
17	17	11	12

Os resultados obtidos por RBI estão na Figura A1.8 e Tabela A1.12.

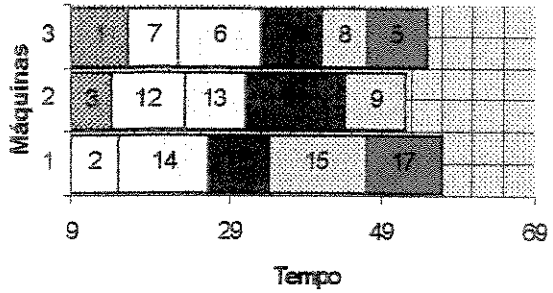


Figura A1.8 – Digrama correspondente a 4º iteração da RBI

Tabela A1.12 – Atribuições da 4° iteração

PA	C_i	d_i
17	55	12

A Tabela A1.12 apresenta os dados da quinta atualização da instância *I3*.

Tabela A1.12 – Dados dos produtos disponíveis inicialmente

$Prod$	p_{1i}	p_{2i}	P_{3i}	r_i	d_i
18	12	9	12	17	108
19	11	8	9	21	100
20	12	11	8	13	98
21	9	10	11	19	92
22	12	9	8	21	106
23	13	8	11	17	96
23	12	5	12	19	104
25	7	8	11	15	112
26	10	7	10	19	110
27	9	12	13	21	100
28	6	5	10	17	90

Os resultados obtidos por MÁQUINAS estão na Figura A1.9 e Tabela A1.13.

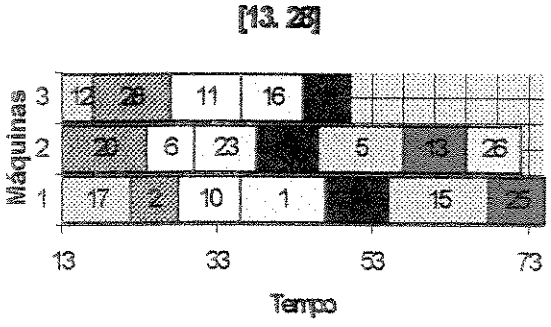


Figura A1.9 – Digrama correspondente ao intervalo [13, 28]

Tabela A1.13 - Atribuições em [13, 28]

C	PA	C_i	d_i
18	18		
19			
20	20	24	98
21			
22			
23	23	38	96
25			
25	25	75	112
26	26	72	110
27			
28	28	27	90

Resultado da RBI estão representados na Figura A1.10 e Tabela A1.14.

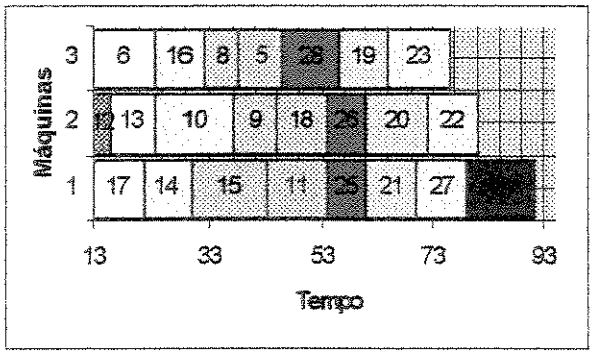


Figura A1.10 - Digrama correspondente a 5º iteração da RBI

Tabela A1.14 - Atribuições da 5° iteração

C	C_j	d_j
18	54	108
19	65	100
20	72	98
21	70	92
22	81	106
23	76	96
24	91	104
25	61	112
26	61	110
27	79	100
28	56	90

Desta forma, as cinco primeiras atualizações da instância *I3* foram apresentadas, assim como, as atribuições obtidas pela rotina MÁQUINAS com L fixo e RBI.

1.3 – Conclusões

Os resultados apresentados procuram estabelecer uma breve exemplificação do funcionamento das rotinas MÁQUIANAS com L fixo e RBI, principalmente, quanto a forma de atribuição dos produtos.

Percebe-se uma alocação de produtos mais imediata na RBI enquanto MÁQUINAS procura investir numa avaliação prévia que não permite uma atribuição imediata de todos os produtos disponíveis às máquinas.

Nos resultados obtidos no capítulo 4, observou-se que as heurística de busca em vizinhança e inserção utilizadas em RBI justificam o bom desempenho desta rotina compensando as atribuições iniciais realizadas de forma aleatória. Por outro lado, a rotina MÁQUINA careceu de tal propriedade, principalmente, por não permitir troca de produtos entre as máquinas o que limita o espaço de busca quando realiza-se otimiza-se as seqüências de produção.

Anexo 2
Tópicos da Implementação Computacional

A2.1 - Introdução

O objetivo deste anexo é apresentar uma visualização geral de como as rotinas implementadas foram executadas. Deve-se lembrar que a implementação computacional destas rotinas foi realizada utilizando linguagem de programação C, compilador GCC, programa CPLEX, estação Sparc IPX com 64 MB de RAM e sistema operacional SunOS 4.1.1.

Tanto a rotina VEÍCULOS apresentada no capítulo 3 como MÁQUINAS no capítulo 4 solucionam instâncias fornecidas por uma rotina encarregada de gerar instâncias chamada GERAINST. No caso do capítulo 3, a rotina fornece instâncias do PRDV estabelecido enquanto que no capítulo 4 estas instâncias referem-se ao PDP proposto. No caso do PDP, também foi implementada a RBI que serviu como critério de comparação ao desempenho de MÁQUINAS. Assim, uma visualização de como a RBI foi executada também será apresentada.

A2.2 - Execução da rotina VEÍCULOS

A Figura 1.1 apresenta, em linhas gerais, todo o processo de execução. Inicialmente, todas as instâncias são geradas para que, em seguida, uma a uma sejam solucionadas por VEÍCULOS. A cada iteração da rotina VEÍCULOS um arquivo de dados é criado com a formulação matemática de um problema de designação. Este arquivo é enviado para o programa CPLEX que soluciona o problema e retorna um outro arquivo de dados com tal solução. Desta forma, a rotina VEÍCULOS é interrompida a cada iteração para execução do CPLEX.

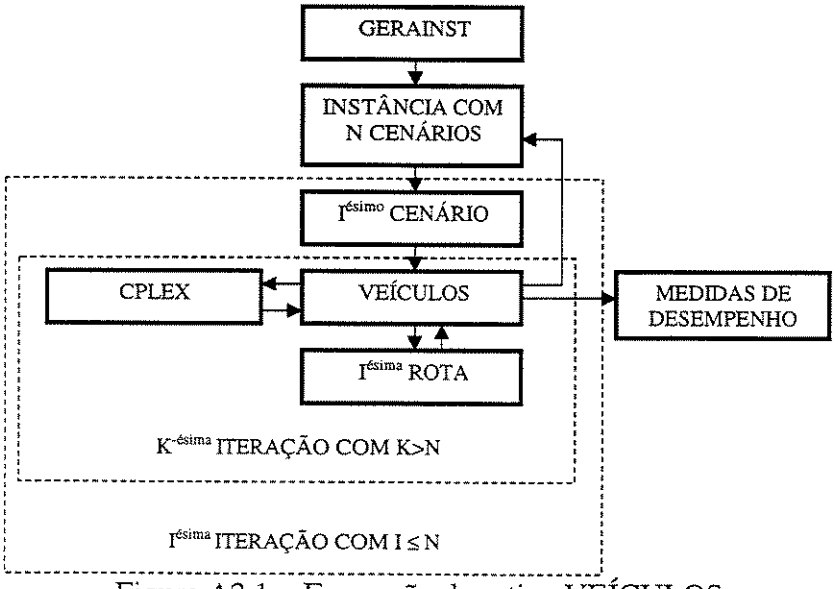


Figura A2.1 – Execução da rotina VEÍCULOS

Para executar a rotina VEÍCULOS, a rotina GERAINST estabelecia inicialmente uma instância do PRDV a ser solucionada. Esta instância, conforme explicado no capítulo 3, era composta por diversos arquivos chamados cenários, responsáveis por alterar o cenário atual do problema, ou seja, responsáveis por simular alterações em tempo real. Desta forma, para uma instância composta de N cenários, a rotina VEÍCULOS lê um arquivo com as alterações a cada I^{ésima} iteração.

Este procedimento é repetido enquanto houver alterações nos cenários. Conforme mencionado, a cada iteração o método executa o programa CPLEX enviando um arquivo com o problema de designação e recebendo um arquivo com a correspondente solução. O método fornece, ao final da iteração, a rota dos veículos que deve ser executada levando-se em conta o cenário atual estabelecido para o PRDV. Quando não há mais arquivos do tipo cenário para serem lidos, a rotina passa a ser executada sem que os dados sejam alterados. Ao final, o método retorna as medidas de desempenho apresentadas no capítulo 3 que avaliam globalmente seu desempenho.

A2.3 - Execução da rotina MÁQUINAS

A Figura A2.2 apresenta, em linhas gerais, todo o processo de execução da rotina MÁQUINAS. O procedimento nesta rotina é análogo ao apresentando na seção anterior onde a diferença básica está no tipo de problema, que neste caso trata-se de um Problema Dinâmico de Programação.

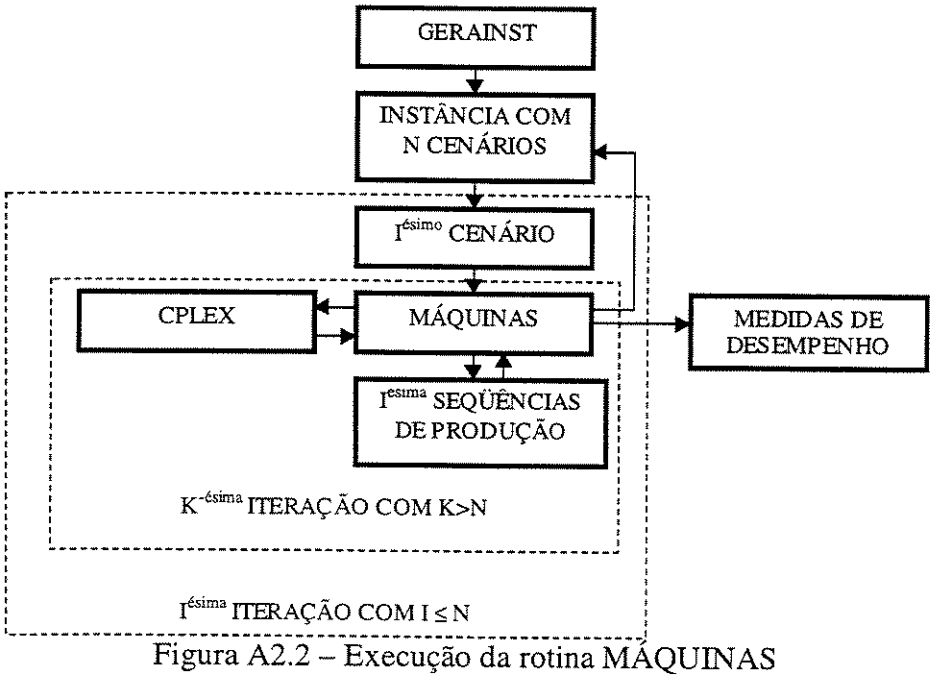


Figura A2.2 – Execução da rotina MÁQUINAS

Neste caso, a rotina retorna a seqüência de produção de cada máquina a cada iteração, baseada nas alterações fornecida pelos arquivos de dados que compõem cada instância. As

medidas de desempenho foram apresentadas no capítulo 4 e também procuram avaliar o desempenho da rotina globalmente.

A2.4 – Execução da RBI

A RBI executa a atribuição aleatória de todos os produtos às máquinas para em seguida executar heurísticas de inserção e busca em vizinhança que otimizem as seqüências estabelecidas. A Figura A2.3 apresenta, em linhas gerais, a forma de execução desta rotina.

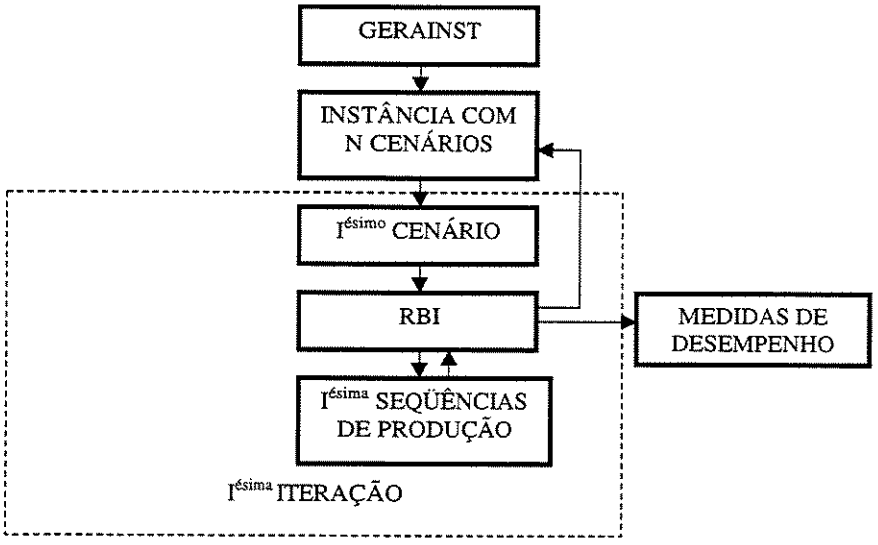


Figura A2.3 - Execução da RBI

A RBI realiza a leitura das atualizações na instância do PDP da mesma forma que a rotina MÁQUINAS. Porém, a cada atualização todos os produtos são imediatamente atribuídos às máquinas fazendo com que o número máximo de iterações seja igual ao número N de alterações ocorridas. As seqüências estabelecidas ao final de cada iteração incluirão todos os produtos que chegam ao sistema de produção e ao final as mesmas medidas de desempenho utilizadas em MÁQUINAS são empregadas para avaliar o comportamento global de RBI.

Conclusão

Nesta anexo foi apresentada uma visualização global de como as rotinas VEÍCULOS, MÁQUINAS e RBI foram executadas durante a resolução de determinada instância. Observou-se que todas estas rotinas estão conectadas a GERAINST, responsável por estabelecer as instâncias a serem avaliadas. Além disso, todas as rotinas fornecem medidas de desempenho que procuram avaliar os respectivos desempenhos.

A rotina VEÍCULOS a cada iteração fornece rotas para os veículos que, numa situação real, deveriam ser colocadas em execução. A RBI e MÁQUINAS fornecem, a cada iteração, as

seqüências de produção a serem executadas pelas máquinas que, num situação real, também seriam colocadas em execução imediatamente.

Observa-se que o procedimento de alteração nos dados dos problemas é o mesmo nas três rotinas. Desta forma, a simulação de um contexto dinâmico foi realizada da mesma maneira em todos os métodos. Isso permitiu uma comparação do comportamento dos métodos quando executados para diferentes instâncias e quando comparou-se a RBI e MÁQUINAS.

ANEXO 3
Publicações

Neste anexo, estão contidos dois artigos que resultaram diretamente deste trabalho. O primeiro artigo chama-se "An Approach to Solve the Dynamic Vehicle Routing Problem" cujos autores foram Claudio Fabiano Motta Toledo, Luiz Manoel Aguilera e Paulo Morelato França. Este artigo foi aceito no " 3rd Portugues Conference on Automatic Control" (CONTROLO'98) realizado nos dias 9, 10 e 11 de setembro de 1998 em Coimbra, Portugal. O segundo artigo chama-se "An Approach to Solve the Dynamic Parallel Machines Scheduling Problem with Sequence-Dependent Changeover Costs" cujos autores foram Claudio Fabiano Motta Toledo, Luiz Manoel Aguilera e Paulo Morelato França. Este artigo foi submetido ao "15th International Conference on CAD/CAM, Robotics & Factories of the Future", CARS & FOF'99", a ser realizado nos dias 18, 19 e 20 de Agosto de 199 em Campinas.

AN APPROACH TO SOLVE THE DYNAMIC VEHICLE ROUTING PROBLEM

Claudio Fabiano Motta Toledo[1]; Luiz Manoel Aguilera[2] and Paulo Morelato França[1]

1 *Universidade Estadual de Campinas - UNICAMP*
Faculdade de Engenharia Elétrica e de Computação - FEEC
Caixa Postal 6101 – CEP 13081-970 – Campinas - SP - Brazil
E-mail: claudio@densis.fee.unicamp.br franca@densis.fee.unicamp.br

2 *Fundação Centro Tecnológico para Informática - C.T.I.*
Instituto de Automação - I.A.
Rodovia D. Pedro I (SP-65) Km 143,6 - Caixa Postal 6162
CEP 13081-970 - Campinas, SP, Brazil - FAX: + 55 19 240 2029
E-mail: aguilera@ia.cti.br

Abstract: The classical Vehicle Routing Problem (VRP) is the efficient distribution of products in order to attend customer requirements. The problem involves route optimization subject to restrictions such as vehicle capacity, route conditions, time to deliver the products. The inputs in this case are previously known and do not take into account changes during their execution. The Vehicle Routing Problem is dynamic if the inputs of the problem are known to the decision-maker or are updated concurrently with the determination of the set of routes. This paper will formulate a DVRP and solve it using the MORSS Algorithm. This algorithm was developed to solve problems where cargo ships should be routed in an emergency situation. We are adapting different MORSS features for a Dynamic Vehicle Routing Problem and evaluating their performance.

Keywords: Scheduling Algorithms, Dynamic Vehicle Routing, Routing Algorithms, Manufacturing Systems.

1. INTRODUCTION

The advances in the area of Mobile Information Systems, including the communication and information technologies by Satellite Communications, Cellular Radios, Traffic Information Broadcasting, Route Guidance and Urban Traffic, allow changing routes at the same time that new information arrives or existent data is updated.

These technological advances require development of new methods and algorithms for the route optimization, taking into account the dynamic characteristics of the routing problem (Psaraftis, 1995).

The classical Vehicle Routing Problem (VRP) aims the efficient distribution of products in order to attend customer requirements. The problem involves route optimization subject to restrictions such as vehicle capacity, route conditions, time to deliver the products. The inputs in this case are previously known and do not take into account changes during their execution. The VRP is a NP-Hard problem (Garey and Johnson, 1979).

The Vehicle Routing Problem is dynamic if the inputs of the problem are known to the decision-maker or are updated concurrently with the determination of the set of routes. In the Dynamic Vehicle Routing Problem (DVRP) the path followed by the vehicles may be changed and requires that the inputs be updated, consequently changing the routing time.

This paper will formulate a DVRP and solve it using MORSS Algorithm. This algorithm was developed by Psaraftis, *et al.* (1985), to solve the problem as to where cargo ships should be routed in an emergency situation. In this work different features of the algorithm was implemented in the aim to solve dynamic vehicle routing problems and evaluate their performance.

Other approaches to solve DVRP were developed by Bertsimas and Ryzin (1991, 1993) using stochastic concepts. Rego and Roucairol (1995) presented an approach using Tabu Search for solving a dynamic multi-terminal truck dispatching problem. A resolution of a practical problem on dynamic transportation networks in vehicle routing and scheduling is presented by Lysgaard (1992).

In section 2, we formulate the problem and show its features. The MORSS Algorithm is presented in section 3. Sections 4 and 5 present respectively the results and conclusions.

2. PROBLEM DEFINITION

Let S_0 be the scenery represented by the graph G_0 with sets V_0 and Q_0 in instant I_0 (Figure 1). Consider the undirected graph $G_0=(N_0, T_0)$ where $N_0=\{1,...,n_0\}$ is the warehouses set and T_0 is the distance set. Let t_{ij} be the travel time between warehouses i and j, such that, $i,j \in D_0$ and $(i,j) \in T_0$. The vehicles fleet is represented by set $V_0=\{1,...,v_0\}$ and the cargo set is represented by $Q_0=\{1,..., q_0\}$. In this context, $+q_i$ is the pickup point of cargo i and $-q_i$ is the respective delivery point.

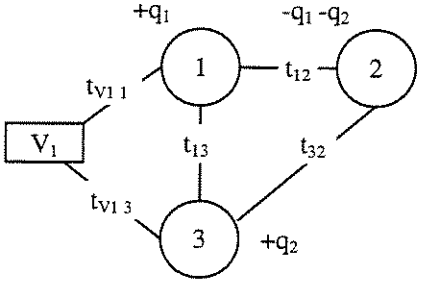


Fig. 1. Scenery S_0

Let S_1 be the scenery represented by the graph G_1, with sets V_1 and Q_1 in the instant I_1, immediatly after instant I_0 (Figure 2). The arrival of new information provokes the changes from one scenery to other. These new information can be the arrival of cargoes in the warehouses, changes in the number of vehicles in the fleet, etc.

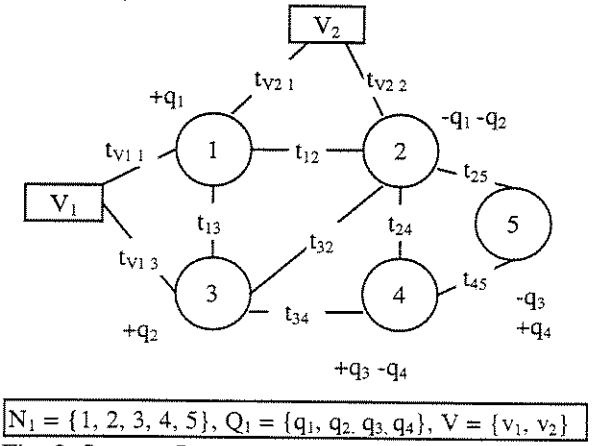


Fig. 2. Scenery S_1

If $q_1=q_0$ or $v_1=v_0$ the information about cargoes and vehicles will not to change ($G_1=G_0$, $V_1=V_0$ and $Q_1=Q_0$). If $q_1<q_0$ or $v_1<v_0$, then either the requirements about cargoes were canceled in T_1, or vehicles were broken so the vehicle fleet was reduced. If $q_1>q_0$ or $v_1>v_0$, then new cargoes arrived and could be placed in existent warehouses or in new warehouses. In order to represent these dynamic changes of the problem, in this second case, we had to extend the graph $N_1=\{1,..., n_1\}$ with $n_1>n_0$ and new arcs $(i,j)\in A_1$ in which $i\in N_1-N_0$ or $j\in N_1-N_0$.

In this problem it is considered that vehicles and cargoes are homogeneous and there are no different kinds of cargoes. Vehicles can take any cargoes if the capacities K are respected. Each cargo has as input the Earliest Pickup Time (EPT), the Latest Delivery Time (LDT), the Origin Warehouse (OW) and the Destiny Warehouse (DW). The cargoes constraint EPT can not be violated. The constraint LDT could be violated but it will imply in penalties. Information about vehicles, cargoes, warehouse number and warehouse distances (t_{ij}) can be updated while the route is being performed. The problem concerns to make efficient routes for each subsequent and different scenery S_0, S_1,...,S_k, obtained by the updates that take place in the respectively instants I_0, I_1,...,I_k.

3. AN APPROACH TO SOLVE THE DVRP USING THE MORSS ALGORITHM

Psarafts, *et al.* (1985), developed the MORSS algorithm to solve an operational routing of ships and cargoes in emergency situations. We describe below an approach to solve the DVRP defined in session 2, by using this Algorithm. The main contribution of this work consists of using this algorithm in another problem and evaluating their performances.

Step 0: Initialize locations of available vehicles. Initialize "master list" of unassigned cargoes. Select length L of individual time horizons. Select fraction a ($0<a<1$). Set $k=1$, $t_1=0$.

Step 1: Set up next horizon $[t_k, t_k+L]$. Form list of cargoes eligible for assignment (all cargoes in master list whose EPT's are between t_k and t_k+L).

Step 2: Calculate assignment utilities for all eligible cargo/vehicle pairs.

Step 3: Form and optimize a transportation network using assignment utilities as arc costs. Resulting assignment forms the "tentative assignment" for $[t_k, t_k+aL]$.

Step 4: Return to "master list" of unassigned cargoes (i) all unassigned cargoes in the Step 3, (ii) all tentatively assigned cargoes whose EPT's are between $[t_k+aL, t_k+L]$ and (iii) all tentatively assigned cargoes which "interact unfavorably" with one another or with cargoes assigned at previous iterations.

Step 5: "Roll" time horizon. Set t_{k+1}=min(lowest EPT of cargoes in master list of unassigned cargoes after input update occurs). Update vehicle locations at t_{k+1}. Set $k=k+1$ and go to Step 1.

The MORSS Algorithm is based on the rolling horizon principle and assignment utility functions. At each time horizon, the algorithm takes the cargoes and vehicles existent and evaluates the impact of inserting cargoes in the routes using mathematical functions. Afterwards, MORSS establishes and solves an assignment problem taking into account assignment utilities that group the benefits of assigning one cargo to a vehicle. Finally, only cargoes with best assign utility and more up-to-date information are assigned to vehicles (Psaraftis, 1988).

The Rolling Horizon Principle consists of sharing the horizon length L in intervals a ($0\le a\le 1$). At each time t_k, current time of the k^{th} iteration, the MORSS algorithm will define the eligible cargoes set:

$$EC=\{q/q\in Q_k \text{ and } q_{EPT}\in [t_k, t_k+L]\} \quad (1)$$

The assignment utility is a mathematical function that determines the impact over the route when assigning one cargo to a vehicle. For each time horizon, after selecting the cargoes, the assignment utility is calculated for all eligible cargoes $q\in EC$, but only cargoes such as $q_{EPT}\in [tk, tk+aL]$ will be assigned permanently. This procedure allows for prioritizing cargoes with earliest pickup time (whose $EPT\in [t_k, t_k+aL]$).

The algorithm calculates assignment utility for all vehicle/cargo pairs. We used the assignment utility function provided by Eq.2. This function is formed by Eq.3, Eq.4 and Eq.5.

$$U_{ij}=U_{ij}(1)+U_{ij}(2) \quad (2)$$

$$U_{ij}(1)=V_{min}+(V_{max}-V_{min})e^{-2[(t/to)]b} \quad (3)$$

The variable t=min{0, t_a-LDT of cargo j} is the tardiness where t_a is the arrival time of the vehicle i with cargo j at the Destiny Warehouse (DW). The values Vmin, Vmax, t_0 and b are input parameters defined by the decision-maker. Eq.3 represents the tardiness impact over the utility of assigned cargo j to vehicle i. If the cargo j arrive in DW on time, then t=0 and $U_{ij}(1)=V_{max}$, otherwise, the utility function will decrease until $U_{ij}(1)=V_{min}$ for $t_a>>$LDT.

$$U_{ij}(2)=\Sigma_{k\in R}\,\Delta U_{ik}(1) \quad (4)$$

$$\Delta U_{ik}(1)=U^{*}_{ik}(1)-U_{ik}(1) \quad (5)$$

The function value $U_{ik}(1)$ is the cargo k utility assignment in actual route and $U^{*}_{ik}(1)$ is the function value for cargo k utility assignment after the dynamic arrival of cargo j in the route. Eq.4 measures the total impact of insert cargo k over the existents cargoes in the route. Eq.5 measures the differences in the assignment utility value (Eq.2) before and after the insertion of cargo k in the existent route over the cargo j tardiness. The following cases may happen:

$\Delta U_{ik}(1)<0$: Cargo j insertion reduces the assignment utility of cargo k. Cargo k delivery time increases after cargo j insertion.

$\Delta U_{ik}(1)=0$: Cargo j insertion unchanged the cargo k assignment utility. Cargo k delivery time is the same after cargo j insertion.

$\Delta U_{ik}(1)>0$: Cargo j insertion increases the cargo k assignment utility. Cargo k delivery time is reduced after cargo j insertion.

After calculating the assignment utility, the algorithm solves an assignment problem:

$$\text{Max } \Sigma_i\Sigma_j\, u_{ij}x_{ij} \quad (6)$$

Subject to

$$\Sigma_i\, x_{ij}\leq 1 \qquad \text{for all j} \quad (7)$$

$$\Sigma_j\, x_{ij}\leq K \qquad \text{for all i} \quad (8)$$

$$x_{ij}=\{0,1\} \qquad \text{for all i and j} \quad (9)$$

Where:

$$x_{ij} = \begin{cases} 1 & \text{If the vehicle i is assigned to cargo j} \\ 0 & \text{otherwise,} \end{cases}$$

The problem formulation aims to obtain the best assignment of cargoes to vehicle in terms of utility function (Eq.6). The restriction provided by Eq.7 avoids that the same cargo travels in difference vehicles according to problem definition in section 2. The vehicle capacity restriction is represented by Eq.8 and the solutions values are in Eq.9. The problem inputs are the cargoes $q_j\in$ EC and vehicles available in $[t_k, t_k+L]$. We used a linear programming software package to solve the assignment problem.

If the solution provides $x_{ij}=1$, vehicle i would be assigned permanently to cargo j, otherwise the cargo will continue to be eligible. The cargoes with $x_{ij}=1$ can not be assigned if their EPT$\in[t_k+aL, t_k+L]$ or if cargo j have an "unfavorably interaction" with other cargoes. We consider that this "unfavorably interaction" occurs when vehicle i is assigned for two or more cargoes. In this case, the cargo assignment with small EPT will be permanently incorporated in the route of vehicle i.

Other cargoes assigned to this vehicle will have the assignment utility recalculated for the new route. The assignment to vehicle i of other cargoes j with $x_{ij}=1$, will happen only if the insertion of these cargoes does not violate the initial assignment utility value. This value is defined by the decision-maker.

4. COMPUTATIONAL RESULTS

In this section, we present the computational results of MORSS Algorithm for three numerical DVRP.

Table 1: Problems and numerical features

P	TV	TC	TW	Av_1	Av_2
P1	10	64	17	18.269	16.285
P2	23	112	39	19.095	12.931
P3	85	177	55	17.506	24.929

Table 1 presents the numerical features of each problem. In the dynamic problems there is no sense in talking about number of cargoes in the problem because new cargoes arrive in real time. Therefore, it is considered the total number of vehicles (TV), warehouses (TW), and cargoes (TC) after every updates of the problem takes place. It is also included the average travel time between warehouses (Av_1) and the average time between earliest pickup time and latest delivery time (Av_2) for every cargoes. Problem P3 is the only whose $Av_2>Av_1$, the others have an average deadline less than the average warehouses travel time.

Table 2: Numerical updates of the Problem 2

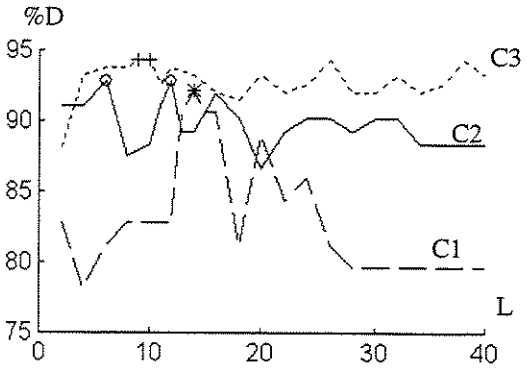
	P20	P21	P22	P23	P24	Total
ΔV	10	0	0	7	6	23
ΔW	7	12	10	2	8	39
ΔC	23	25	15	22	27	112

Table 2 illustrates the updates than happen in the problem P2 as function of the rolling horizon. The columns represent the initial problem scenery P20 and the subsequent updates P21, P22, P23 and P24. The rows ΔV (vehicles), ΔW (warehouses) and ΔC (cargoes) represent the variation in the data when new information arriving on each update.

In conformity with the explained in section 2, in the first scenery (P20) the problem has 10 vehicles, 7 warehouses and 23 cargoes. This scenery changes when updates occurs producing a new scenery (P21) with 12 new warehouses and 25 new cargoes. The data are actualized until the fourth update (P24), in which 6 new vehicles, 27 new cargoes and 8 new warehouses take place. The same procedure, with different values, is used in the problem P1 and P3. Only 4 updates in its original scenery is considered in the 3 problems.

It is important to also evaluate the adequacy of the MORSS algorithm in the solution of different dynamic problems. An analysis of the algorithm's behavior in these three different problems (P1, P2, P3) showed that there were variations over the percentage of delivery cargoes on time when the time horizon length L changed. In each problem was tested the percentage of the cargoes delivery for time horizon length 2 until 40.

Problem P1 present the best percentage value 92.19 (symbol "*" in the curve C1) when the time horizon length L=14. The problem P2, the best percentage 92.86 (symbol "o" in the curve C2) occurs in two time horizons length: L=6 and L=12. Problem P3 also has your best percentage value 94.35 (symbol "+" in the curve C3) for two different time horizon length L=9 and L=10 (Figure1).



%D: Percentage of cargo delivery on time
L: Time horiozon length

Fig. 1. MORSS Solutions. 2D representation of the percentage of the cargoes delivery on time when the time horizon length change in each problem.

Based on this results we conclude that a best time horizon length L_b exists for each problem. This occurs because the algorithm give the best sets of cargoes eligible and vehicles ready for use in each different time interval. Times horizons interval with length L_b let the algorithm give more satisfactory scenarios S_0, S_1,..., S_k in the instants I_0, I_1, ..., I_k, in conformity with idea explained in section 2.

Therefore, the algorithm looks at a more adequate scenery in $[t_k, t_k+ L_b]$ and makes decisions in the most guaranteed subinterval $[t_k, t_k+aL_b]$. For instance, the problem P3 has two L_b, L_{b1}=9 and L_{b2}=10. This happens because either one of the intervals, $[t_k, t_k+L_{b1}]$ or $[t_k, t_k+L_{b2}]$, is established in each iteration to obtain good scenarios.

Problems were more sensible to the time horizon length than utility function parameters during computational tests. The three problems, P1, P2 and P3, were tested with the same utility functions parameters whose values were V_{min}=0, V_{max}=1, t_0=5 and b=2.

Table 3: Morss algorithm results – cargoes delivery

Problem	% Delivery On time	% Delay	Lb
P1	92.19	7.81	14
P2	92.86	7.14	6 and 12
P3	94.35	5.65	9 and 10

Table 3 present the time horizon length L_b with the best percentage of cargoes delivery on time and delay.

Table 4: P1 performance (L_b=14)

H	V	W	C	EC	AC
[2,16]	5	4	27	8	1
[5;19]	5	5	35	17	3
[6,20]	7	9	37	30	4
[9,23]	8	13	54	28	6
[13,27]	10	17	64	34	8
[14,28]	10	17	64	35	9
[15,29]	10	17	64	27	7
[17,31]	10	17	64	23	7
[20,34]	10	17	64	19	5
[26,40]	10	17	64	14	6
[29,43]	10	17	64	8	5
[30,40]	10	17	64	5	2
Total	10	17	64	-	64

Table 5: P2 performance (L_b=12)

H	V	W	C	EC	AC
[3,15]	10	7	23	10	3
[4,16]	10	19	48	9	3
[6,18]	10	29	63	22	4
[7,19]	17	31	85	33	7
[11,23]	23	39	112	46	11
[13,25]	23	39	112	57	22
[16,28]	23	39	112	40	19
[19,31]	23	39	112	39	10
[21,33]	23	39	112	33	5
[25,37]	23	39	112	28	16
[26,38]	23	39	112	12	7
Total	23	39	112	-	112

Table 6: P3 performance (L_b=10)

H	V	W	C	EC	AC
[7,17]	33	32	81	20	5
[8,18]	50	43	115	46	18
[9,19]	66	44	120	50	8
[12,22]	76	53	167	43	11
[13,23]	85	55	177	82	16
[14,24]	85	55	177	71	10
[15,25]	85	55	177	67	8
[17,27]	85	55	177	62	14
[18,28]	85	55	177	60	15
[19,29]	85	55	177	52	12
[20,30]	85	55	177	35	6
[25,35]	85	55	177	43	13
[26,36]	85	55	177	30	11
[28,38]	85	55	177	19	3
[30,40]	85	55	177	16	5
Total	85	55	177	-	177

Table 4, 5 and 6 presents the performance of the problems P1, P2 and P3, respectively, when the best time horizon length (L_b) is used. Columns represents total vehicles (V), warehouses (W) and cargoes (C). Rows represent the correspondent time horizon interval. The number of eligible (EC) available and assignment cargoes (AC) permanently assigned to vehicles, in each time horizon, are also in the last two columns respectively.

The problems present the initial configuration in the first time horizon interval. In the next four time horizons updates occur in the problem and consequently the number in the columns V, W and C increase. For instance, problem P2 in the first interval time [3, 15] presents V=10, W=7 and C=23 corresponding to column P20 in the Table 2. In the next interval, [4, 16] the updates occur with V=10, W=19 and C=48 that correspond to the values ΔV=0, ΔW=12, ΔC=25 in the column P21 in Table 2. After interval [13, 25] there are no updates and the algorithm works with the total available data.

The CPU time in using a SUN station with the Cplex linear program for the trasportation problem solution is 31 seconds in problem P1, 48 seconds in problem P2 and 2 minutes and 31 seconds in problem P3.

5. CONCLUSION

This paper presents an approach to solving a Dynamic Vehicle Routing Problem based on the MORSS algorithm. The main interest in studying this problem is a result of the new trends in mobile information systems areas that allows a real-time monitoring of a vehicle fleet. This work evaluates the MORSS algorithm behavior solving a DVRP with special features. Computational results showed the relation between time horizon length and the performance of routes. Future work will consist in adapting this method to solve a Dynamic Parallel Machines Scheduling Problem with Changeover Costs using the same mathematical model.

ACKNOWLEDGEMENTS

The work at "Fundação Centro Tecnológico para Informática" (C.T.I.) was supported by "Fundação de Amparo à Pesquisa do Estado de São Paulo" (FAPESP) grant 95/8306-1. This work is part of the Short Term Production Planning project (STERMPROPLAN) on the Keep-In-Touch program (KIT No.137).

REFERENCES

Bertsimas, D.J. and G. van Ryzin (1991). Stochastic and Dynamic Vehicle Routing Problem in the Euclidean Plane. *Operations Research* **39**, 601-615.

Bertsimas, D.J. and G. van Ryzin (1993). Stochastic and Dynamic Vehicle Routing Problem in the Euclidean Plane with Multiple Capacitated Vehicles. *Operations Research* **41**, 60-76.

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

Lysgaard, J. (1992). Dynamic Transportation Networks in Vehicle Routing and Scheduling. *Interfaces* **22,** 45-55.

Psaraftis, H. N., J. B. Orlin, D. Bienstock and P. M. Thompson (1985). Analysis and Solution Algorithms of Sealift Routing and Scheduling Problems: *Final report, Working Paper No. 1700-85, Sloan School of Management, MIT.*

Psaraftis, H.N. (1988). Dynamic Vehicle Routing Problems. In: *Vehicle Routing: Methods and Studies* (B.L. Golden and A. A. Assad, (Ed.)), 223-248. Elsevier Science Publishers B. V., North-Holland.

Psaraftis, H.N. (1995). Dynamic Vehicle Routing: Status and Prospects. Annals of Operations Research, 61, 143-164.

Rego, C. and C. Roucairol (1995). Using Tabu Search for Solving a Dynamic multi-terminal truck dispatching problem. *European Journal of Operational Reserach* **83**, 411-429.

AN APPROACH TO SOLVE THE DYNAMIC PARALLEL MACHINES SCHEDULING PROBLEM WITH SEQUENCE-DEPENDENT CHANGEOVER-COSTS

Claudio Fabiano Motta Toledo[1]; Luiz Manoel Aguilera[2] and Paulo Morelato França[1]

1 *Universidade Estadual de Campinas - UNICAMP*
Faculdade de Engenharia Elétrica e de Computação - FEEC
CEP 13081-970 - Caixa Postal 6101 - Campinas, SP, Brazil
E-mail: claudio@densis.fee.unicamp.br Franca@densis.fee.unicamp.br

2 *Fundação Centro Tecnológico para Informática - C.T.I.*
Instituto de Automação - I.A.
Caixa Postal 6162 - CEP 13081-970 - Campinas, SP, Brazil
FAX: + 55 19 746 6028 - E-mail: aguilera@ia.cti.br

Abstract: This paper presents an approach to solving the Dynamic Parallel Machines Scheduling Problem with Sequence-Dependent Changeover-Costs. Several research works have been already tackled to solve the static case of the problem. This work establishes a dynamic case of the problem in which the information about the products was not previously known and proposes to adapt an algorithm to decide it. The objective is to solve a bi-criteria scheduling problem to minimize changeover costs and respect the due-date constraints in the dynamic context.

Keywords: Scheduling Algorithms, Dynamic Vehicle Routing, Routing Algorithms, Manufacturing Systems, Dynamic Scheduling.

1. INTRODUCTION

Scheduling is the allocation of resources over time to perform a collection of jobs (Blazewicz, *et al.*, 1993). Scheduling on Parallel Machines arises when a set of jobs is given, each requiring processing on one of the machines. This problem occurs in several areas such as computing, production, manufacturing, vehicle routing, etc. In this work, a Dynamic Parallel Machines Scheduling Problem with Sequence-Dependent Changeover-Costs is established and a solution approach is presented. This problem is modeled by a Dynamic Vehicle Routing Problem and solved using a MORSS Algorithm adaptation.

The classical Vehicle Routing Problem (VRP) aims for the efficient distribution of products in order to attend customer requirements. The problem involves route optimization subject to restrictions such as vehicle capacity, route conditions and time to deliver the

products. The inputs in this case are previously known and do not take into account changes during their execution. The VRP is a NP-Hard problem (Garey and Johnson, 1979).

The Vehicle Routing Problem is dynamic if the inputs of the problem are known to the decision-maker or are updated concurrently with the determination of the set of routes. The MORSS Algorithm was developed by Psaraftis, *et al.* (1985), to solve a Dynamic Vehicle Routing Problem where cargo ships should be routed in an emergency situation.

2. PROBLEM DEFINITION

The problem consists of the scheduling of different kinds of products in the context of a semi-continuous manufacturing process. A changeover period exists between the production of two different types of products and during this period the machines are idle. There are changeover-costs that take into account different parameters issued by the production characteristics, for example, speed variation of the molding machine, variation on the amount of material flux and variation on the product model shapes. The changeover-costs between all the products characterizes a changeover cost matrix (Gonthier, 1990).

The Three Parallel Machine Scheduling Problem with Sequence-Dependent Changeover Costs can be modeled by the Vehicle Routing Problem (VRP). In this case, the vehicles represent the machines; the customer orders represent the products to be made; the intercities distances represent the changeover-costs and the vehicle capacity is the production horizon in each machine. A two-phase method was adapted to solve the static parallel machines scheduling problem which represents a part of the global problem (Aguilera, 1993; Aguilera, *et al.*, 1995 and Aguilera, *et al.*, 1996; França *et al.*, 1996 and Mendes *et al.*, 1999) an extension of this problem to the dynamic case will be described and solved.

The Three Parallel Machine Problem with Sequence-Dependent Changeover Costs is dynamic if the input of the problem (products information) is known to the decision-maker or is updated concurrently with the products sequence construction or execution. In the initial instant I_0 there is information relating to the available products set P_0. In the subsequent instants I_k ($k>0$) there will be information about the available product sets P_k relating to new products or updates on previous data.

The Dynamic Parallel Machines Scheduling Problem with Sequence-Dependent Changeover Costs can be modeled by a Dynamic Vehicle Routing Problem. In this case, the changeover costs and the machines are respectively the intercities distances and vehicles. The products arrival time will be cargoes pickup time, due-date will be the deadline for cargoes delivery, processing time will be effective cargoes delivery time and processing time will be vehicles capacity. The products sequence construction will correspond to the routes determination in the DVRP.

The products information considerated by the problem is the arrival time or ready time (r_j), due-date (d_j), processing time (p_j) and completion time (C_j). The ready time is the instant when the product will be ready for inserting in the machine. The due-date is the latest possible time when the product shall leave the machines. The processing time is the amount of time that the product will stay in each machine. The completion time is the time to finish the product resulting from the evaluated scheduling.

Each product has only one job, and there are no precedent or technological constraints that require some jobs to be completed before others begin. Job preemption is forbidden, and the difficulties in passing onto a second job depend on the preceding job in the machine (sequence-dependent). There are changeover-costs associated with to passing from one product to another (changeover-cost matrix). The objective is to minimize the total changeover-costs and complete each product process on time taking into account the dynamic characteristics of the problem.

3. AN APPROACH TO SOLVE THE DPMP USING THE MORSS ALGORITHM

Psaraftis, *et al.* (1985), developed the MORSS algorithm to solve an operational routing of ships and cargoes in emergency situations. According to what was explained in section 2, cargoes will correspond to products and ships to machines. The main contribution of this work consists of adapting this algorithm to solve the Dynamic Three Parallel Machines Scheduling Problem with Sequence-Dependent and evaluating its performance. The algorithm steps are described below.

Step 0: Initialize machines process. Initialize the list of the available products (P_0). Select length L of individual time horizons. Select fraction a ($0<a<1$). Set $k=1$, $t_0=0$.

Step 1: Set up next horizon $[t_k, t_k+L]$. Form list of eligible products (EP) for assignment (all products in the list of available products (P_k) of k^{th} iteration whose ready times (r_j) belong $[t_k, t_k+L]$).

Step 2: Calculate assignment utilities for all eligible product/machine pairs.

Step 3: Form and optimize an assigned problem using assignment utilities as arc costs. Resulting assignment forms

the "tentative assignment" for $[t_k, t_k+L]$.

Step 4: Return to list of available products (i) all unassigned products by Step 3, (ii) all tentatively assigned products whose r_j's are between t_k+aL and t_k+L and (iii) all tentatively assigned products that "unfavorably interact" with one another or with products assigned at previous iterations.

Step 5: "Roll" time horizon. Make t_{k+1}=min(lowest r_j of products in the list of available products). Update the machines process time at t_{k+1}. Set k=k+1 and go to Step 1.

The MORSS Algorithm is based on the rolling horizon principle and assignment utility functions. At each time horizon, the algorithm takes the products and machines existent and evaluates the impact of inserting products in the machines sequence using mathematical functions. Afterwards, MORSS establishes and solves an assignment problem taking into account assignment utilities that group the benefits of assigning one product to a machine. Finally, only products with best assign utility and more up-to-date information are assigned to machines (Psaraftis, 1988).

The Rolling Horizon Principle is the decomposition of the problem by time horizons of length L where the problem decisions are evaluated. However, the problem's final decision takes place in a subinterval of this time horizon. The subinterval has a length aL where a (0<a<1) is a user defined parameter input. At each time t_k, current time of the k^{th} iteration, the MORSS algorithm will define the eligible products set (EP) from the available products set (P_k). The EP set is formed by the products available whose ready time r_p belongs to each time horizon:

$$EP=\{p/p\in P_k \text{ and } r_p\in [t_k, t_k+L]\}$$

The assignment utility is a mathematical function that measures the impact over the machine sequence of production when assigning one product p∈EP to a machine. After the time horizon and the corresponding EP set are established, the MORSS algorithm will be simulated, for each product p∈EP, insertions in the machines sequence of production. At this point, the best position of product p and the change in the sequence of production is made using an insertion heuristic and a local search heuristic.

The effects of one product j∈EP insertion in determined machine i is measured by the assignment utility function U_{ij}. Three situations would occur when the insertion of product j is simulated in machine i: (i) machine i doesn't have not products scheduled; (ii) machine i is processing one product scheduled without other products scheduled; (iii) machine i is processing one product and already has other products scheduled.

$$U_{ij}=U_{ij}(1)+U_{ij}(2) \quad (1)$$

Eq.1 evaluates the impact of eligible product j insertion in the machine i over the due dates ($U_{ij}(1)$) and changeover costs ($U_{ij}(2)$) features. This equation will be used when the machine is stopped and when the machine has only one product scheduled. The values Vmin, Vmax, t_0 and b are input parameters defined by the decision-maker.

$$U_{ij}(1)=\begin{cases} \left[V_{min}+(V_{max}-V_{min})\times e^{-2\times (Dj/to)^{b1}}\right] & (2) \\ V_{max}+\left[V_{min}+(V_{max}-V_{min})\times e^{-2\times (Ej/to)^{b1}}\right] & (3) \end{cases}$$

Eq. 2 return the assignment utility when $D_j=\max\{C_j-d_j, 0\}$, where D_j is the tardiness. If $D_j=0$ the benefit of assigned the product j to machine i is maximum ($U_{ij}(1)=V_{max}$). Otherwise, the utility function will decrease until $U_{ij}(1)=V_{min}$ for $C_j>>d_j$. Eq.3 return the same value when occurs $E_j=\max\{d_j-E_j, 0\}$, where E_j is the earliness. If $E_j=0$ the Eq.3 return $U_{ij}(1)=V_{max}+V_{min}$, otherwise, the utility function will decrease until $U_{ij}(1)=V_{max}+V_{min}$ for $d_j>>E_j$. The tardiness in the products processes will be penalized harder than when earliness occurs. The earliness would, for instance, cause extra costs in the products storage. However, in this work, it will be assumed that it is more costly not to process one product on time than to store it.

$$U_{ij}(2)=\beta \quad (4)$$

If the machines are stopped, the Eq. 4 provides the value for the function $U_{ij}(2)$. The $U_{ij}(2)$ value will be maximum and provided by the user input parameter β.

$$U_{ij}(2)=\frac{\beta}{c_{kj}} \quad (5)$$

Eq. 5 provides the value for $U_{ij}(2)$ when there is one product in process without others scheduled in machine i. The criterion adopted in this work will be that the benefit of inserting the product eligible j in terms of changeover costs will be the inverse of changeover cost (c_{kj}) involved in.

$$U_{ij}=U_T(1)+U_{ij}(2) \quad (6)$$

Eq.6 presents the assignment utility function U_{ij} of inserting the product eligible j when there is more than one product already scheduled in machine i.

$$U_T=\frac{\sum_{k\in S} U_{ik}(1)}{|S|} \quad (7)$$

Eq. 7 evaluates the tardiness of products sequence S obtained after j insertion. This equation divided the sum of $U_{ik}(1)$ calculated for each product $k \in S$ by the number of products belong S. The value provided by Eq. 7 is the $U_{ij}(1)$ average of the sequence.

$$U_{ij}(2) = \beta \frac{|S|-1}{\sum c_{kj}} \quad (8)$$

Eq. 8 evaluates the changeover costs of sequence S. The total changeover cost of sequence S ($\sum c_{kj}$) is divided by the products number of sequence S (|S|-1) resulting in the average changeover cost of sequence S. This average value is inverted and multiplied by the maximum value β. The criterion adopted in Eq.8 is an extension of that used in Eq.5.

If the machine i is processing one product and already has other products scheduled, the cargo $j \in EP$ insertion will be simulated over the sequence of products S already attributed previously to the machine. Let S_{ij} be the machine i sequence after eligible product j insertion takes place. In this work, two heuristics were adapted to MORSS algorithm. The first is an insertion heuristic that executes the following steps:

Step 1: Let j the eligible product and k the last product scheduled in the original sequence S^0 of the machine i. Insert j in S^0 after the product in processing, provided the sequence S'. Calculated U'_{ij} by Eq. 6.

Step 2: Insert j after the next product in S^0 and obtained the new sequence S. Calculated the respective value of U_{ij}.

Step 3: If $U_{ij} > U'_{ij}$ then $S' = S$ and $U'_{ij} = U_{ij}$. If j is not inserted after product k yet, then return to Step 2. Otherwise, return S', U'_{ij} and Stop.

After execution of insertion heuristic, the local search heuristic is also execution over the sequence provided by insertion heuristic. The local search heuristic adapted is the All Pairwise Interchange (ALLPAIRS). Let $S^0 = p_1, p_2, \ldots, p_j, p_{j+1}, \ldots, p_k, p_{k+1}, \ldots, p_n$ a products sequence. ALLPAIRS executes permutation and obtained new sequences by the criterion $S \in \alpha(S^0)$ if $S = p_1, p_2, \ldots, p_{k-1}, p_j, p_{k+1}, \ldots, p_{j-1}, p_k, p_{j+1}, \ldots, p_n$, for $1 \le k < j \le n$. The heuristic ALLPAIRS used in algorithm MORSS execute the following steps:

Step 1: Let $S^0 = p_1, p_2, \ldots, p_k, p_j, p_{k+1}, \ldots p_n$, where p_i is the $i^{ésimo}$ product already assigned to sequence and j is the eligible product. Calculate U^0_{ij} by Eq. 6 and make $S' = S^0$ and $U'_{ij} = U^0_{ij}$.

Step2: Take the sequence $S \in \alpha(S^0)$ if $S = p_1, p_2, \ldots, p_{k-1}, p_j, p_{k+1}, \ldots, p_{j-1}, p_k, p_{j+1}, \ldots, p_n$, for $1 \le k < j \le n$ and calculate U_{ij} by Eq. 6. If all permutation of products belong S^0 already take place, without a new sequence S has be obtained, go to Step 4.

Step3: If $U_{ij} > U^0_{ij}$ them $S^0 = S$ and $U^0_{ij} = U_{ij}$. Go to Step 2.

Step4: If $S' \neq S^0$ them $S' = S^0$ and $U'_{ij} = U^0_{ij}$, go to Step 2. Otherwise stop the execution and return S' and U'_{ij}.

At the end, ALLPAIRS will return to machine i and product j the best sequence with a compromise solution of changeover-costs and tardiness. It will also return the corresponding best assignment utility function value U_{ij}. This procedure is repeated for each product $j \in EP$ in the three machines.

After calculating the best assignment utility function value, the algorithm solves an assignment problem:

$$\text{Max } \Sigma_i \Sigma_j U_{ij} x_{ij} \quad (9)$$

Subject to

$$\Sigma_i x_{ij} \le 1 \quad \text{for all j} \quad (10)$$

$$\Sigma_j p_{ij} x_{ij} \le K \quad \text{for all i} \quad (11)$$

$$x_{ij} = \{0,1\} \quad \text{for all i and j} \quad (12)$$

where:

$$x_{ij} = \begin{cases} 1 & \text{If the product i is assigned to cargo j} \\ 0 & \text{otherwise,} \end{cases}$$

The problem formulation aims to obtain the best assignment of products to machines in terms of assignment utility function value (Eq.9). The restriction provided by Eq.10 avoids the same product processes in different machines. The machines capacity restriction is represented by Eq.11 and the solutions values are in Eq.12. A CPLEX (1994) programming software is used to solve this assignment problem. If the solution provides $x_{ij}=1$, product j may be assigned permanently to machine i, otherwise the product will continue to be eligible.

The products j with $x_{ij}=1$ can't be assigned if their ready time (r_j) belongs to $[t_k+aL, t_k+L]$ or "unfavorable interaction" takes place. The "unfavorable interaction" occurs when two or more products are assigned for the same machine i. In this case, the product assignment with small r_j will be permanently incorporated in the machine i sequence of products.

The other products assigned to this machine will have the assignment utility U_{ij} recalculated for the new sequence according to the three possible machines situation. The assignment to machine i of the other products l ($l \neq j$) with $x_{il}=1$ will happen only if the insertion of these products does not reduce the initial assignment utility less than a parameter defined by the decision-maker.

The insertion of product j with value U_{ij} in machine i take place because $x_{ij}=1$ and $r_j \in [t_k, t_k+aL]$ producing

the new sequence S_{ij}. Another insertion of product l with $x_{il}=1$ and $r_l \in [t_k, t_k+aL]$ will take place only if after the insertion of l in sequence S_{ij}, the local search heuristic provides a sequence S_{il} so that the associated U_{il} is larger than the previously obtained value U_{ij}. If $U_{il}<U_{ij}$ and the value $(U_{ij}-U_{il}) \leq \rho$ the l insertion will also take place. The parameter ρ is defined by the user.

4. COMPUTATIONAL RESULTS

In this section, the computational results of MORSS Algorithm adaptation for three numerical Dynamic Three Parallel Machines with Sequence-Dependent Scheduling Problem are presented.

Table1: Problems and numerical features

Prob	P1	P2	P3
TNP	46	80	80
NU	20	20	10
NP/NU	2.3	4	8
AVE	63.13	76.88	77.06

Table 1 presents the numerical features of each problem. In the dynamic problems there is no sense in talking about number of products in the problem because new products arrive in real time. Therefore, it is considered the Total Number of Products (TNP) after each update of the problem takes place. The row NU presents the Number of Updates that occurs in each problem. NP/NU presents the Number of Products that arrive on average at each update. AVE is the average value of difference between due date and ready time of all products. For instance, 80 products arrive through 20 updates in P2. In this case, 4 products arrive in average at each update and the average deadline for the machines to end the products process is 76 TU. The TU means Time Unit and will be the time measure considered.

The method performance will be available by percentage of products delivered on time (%D) and by the average of total changeover cost (AC) in the machines scheduled products sequences.

$$\%D = \frac{|P_d|}{|P_p|} \tag{13}$$

$$AC = \frac{\sum_{k,j \in P_p} c_{kj}}{|P_p| - 3} \tag{14}$$

The Eq. 13 provides %D where P_p is the total products processed and P_d is the products processed on time. Eq. 14 provide AC where the total value of products changeover costs scheduled in machines is divided by the number of changeover costs used in the three machines.

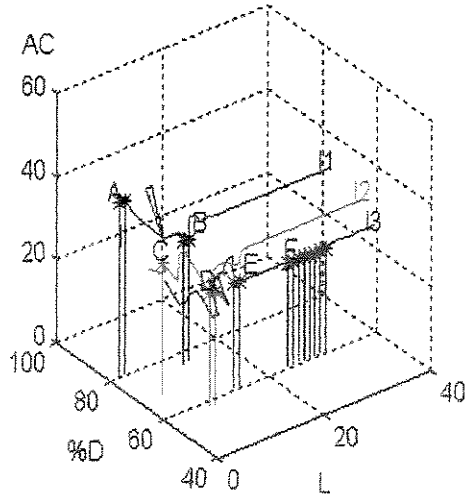


Fig. 1. MORSS Algorithm performance. 3D representation of method performance in problems P1, P2 and P3 when the time horizon length change.

In Figure 1 the letters A, C and E present points with %D best values of problems P1, P2 and P3 respectively. The letters B, D and F present points with AC best values of problems P1, P2 and P3 respectively. Problem P1 presents the highest curve because it has worse AC performance for different L values. P2 and P3 have similar heights because they present similar AC values. The P2 and P3 heights are less than P1 because they have better performance of AC. In another case, P1 presents best performance results of %D than P2 and P3. The %D of P2 and P1 not exceed 70% while P1 has best %D more than 80%.

Table 2: Best values of %D and AC

Prob.	(%D; AC;L)	(%D;AC;L)
P1	(**80,76**; 41,39;2);	(76,23;**29,23**;12)
	(**80,76**;41,39;3)	(76,23;**29,23**;13)
P2	(**68,63**; 31,77;4)	(58,68; **26,84**;8)
		(58,68;**26,84**;9)
P3	(**61,57**;24,62;14)	(61,28; **24,44**;24)
	(**61,57**;24,62;15)	...
		(61,28;**24,44**;31)

In problem P3, for instance, the best value of %D obtained by the method occurs in the ordered triple (**61,57**; 24.62; 14). The best value of AC obtained

occurs in the ordered triple (61.28; 24.44; 24) and this value is repeated for L=25 until L=31. The localization of theses points is showed by letter E and F in Figure 1. Based on these results we conclude that the best result depends on an adequate choice of horizon length (L') value. This occurs because the algorithm looks at a more adequate scenery in $[t_k, t_k+L']$ and makes decisions in the most guaranteed subinterval $[t_k, t_k+aL']$. Parameter L is a decision-maker input. In a real dynamic situation, the decision-maker doesn't have time to evaluate different results obtained using different L values. In this work, we propose a criterion to calculate the L value at each iteration. The aim is to establish a way to obtain L with minor users interference.

$$L = \begin{cases} \lambda \dfrac{\sum_{j \in P_a} (t_k - r_j)}{|P_a|} & \text{If } P_a=\{j \in P_k \,/\, t_k - r_j \neq 0\} \\ 1, & \text{Otherwise } P_{kl}=\varnothing \end{cases} \quad (15)$$

Eq. 15 provides the criterion of calculations adopted. This equation presents the value $t_k=\min\{r_j, \forall j \in P_a\}$ as lower limit of time horizon established in the $k^{ésima}$ method iteration, where P_a has all available products $j \in P_k$ such as $t_k-r_j \neq 0$. The sum of all $t_k-r_j \neq 0$ with $j \in P_k$ is divided by the number of products available. The average value obtained is multiplied by the user parameter λ such as $0<\lambda<\lambda_{max}$. For $\lambda=1$ the horizon length L calculated in each iteration of method will be the average value of $t_k-r_j \neq 0$ with $j \in P_k$. In this case, the interval $[t_k, t_k+L]$ will include only products whose r_j-t_k belong to the value average calculate in Eq.15. If $\lambda=1.2$, for instance, the value of L will be allowed to include products such that $r_j - t_k$ exceeds the average value until 20%

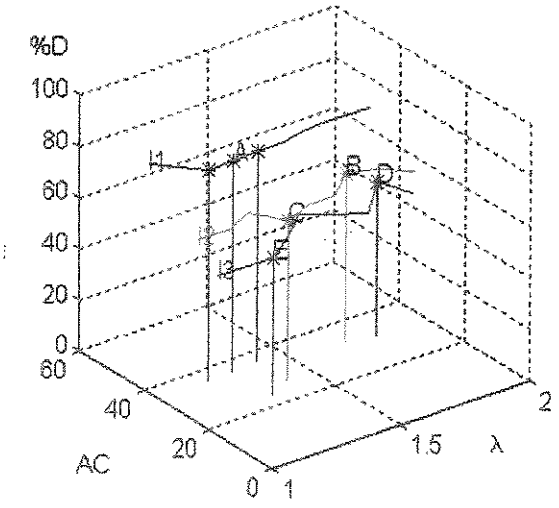


Fig. 2. MORSS Algorithm performance. 3D representation of method performance in problems P1, P2 and P3 when the time horizon length is calculated for different values of λ.

The problems performance was available using λ={1, 1.1,1.2,...,1.9}. In P1 the letter A shows points with the best value of %D and AV. The letter B and D show the points with best results of %D in problems P2 and P3, respectively. The letter C and E show the points with best results of AC in P2 and P3. The problem P1 presents again the best performance of %D for different λ values while P2 and P3 have better performance in AC than P1.

Table 3: Best values of %D obtained

Prob.	(%D; AC;λ)	(%D;AC;L)
P1	(**82.23**; 36.14; 1.2); (**82.23**; 36.14; 1.3); (**82.23**; 36.14; 1.4);	(**80.76**; 41.39;2); (**80.76**;41.39;3)
P2	(**66.12**; 32.60; 1.7)	(**68.63**; 31.77;4)
P3	(**60.34**; 31.10; 1.8)	(**61.57**;24.62;14) (**61.57**;24.62;15)

Table 3 compares the best results of %D obtained when the MORSS algorithm adaptation calculate L value using Eq.15 (first column of Table 3) with the best result obtained without calculating L (second column of Table 3). In P2, for instance, the best result of %D=66.12 when the method calculating L is obtained using λ=1.7. In this case, the result is reached if 70% of average value in Eq.15 is exceeded. The best result of %D=68.63 is reached when the parameter L=4 provided by decision-maker is adopted in all method iteration. In P1 the method has better performance when L is calculated. This situation alters in P2 and P3 when the method that without calculating L obtained the better performance of %D. In the three analyzed problems, the resulting difference indicates that calculating L has better performance in the problem with a lesser dimension (TNP=46 in P1). If the dimension is increased the performance is better when L is not calculated (TNP=80 in P2 and P3).

Table 4: Computational Time

Prob.	ACT(λ)	TCT(λ)	ACT(L)	TCT(L)
P1	1.68"	37"	1.70"	46"
P2	3.96"	1'43"	2.56"	1'17"
P3	7.18"	2'38"	5"	1'25"

Table 4 shows the computational time to obtain the best

results. The Average Computational Time (ACT) is the time expense for to method establish machines product sequences in each iteration. The Total Computational Time (TCT) is the time spent by the method to solve the problem. The computational time in P2 and P3 is great when L needs to be calculated. For instance, the problem P3 spent in average ACT(λ)=7.18" to establish one sequence when calculate L, while the method that not calculate L expensive ACT(L)=5". The TCT(λ)=2'38" also is greater than TCT(L)=1'25" when the method calculate L. The exception occurs in P1 where the method with L calculated to spend less time.

Table 5: Best values of AC obtained

Prob.	(%D; **AC**;λ)	(%D;**AC**;L)
P1	(82.23; **36.14**; 1.2);	(76,23 **29.23**;12);
	(82.23; **36.14**; 1.3);	(58,68;**26.84**;8)
	(82.23; **36.14**; 1.4);	
P2	(63.83; **27.83**; 1.9)	(58,68;**26.84**;8)
		(58,68;**26.84**;9)
P3	(53.82; **23.69**; 1.8)	(61,28;**24.44**;24)
		...
		(61,28;**24.44**;24)

Table 5 compares the best results of AC obtained when the MORSS algorithm adaptation calculates L value (first column) with the best result obtained without calculating L (second column). In these three problems, the resulting difference indicates that calculate L has better performance in problem P3 with larger products arriving in each iteration NP/NU=8. In problems where NP/NU decrease the performance is better when L is not calculated as in P2 with NP/NU=8 and P3 with NP/NU=2.3 (see Table 1).

Table 6: Computational Time

Prob.	ACT(λ)	TCT(λ)	ACT(L)	TCT(L)
P1	1,6"	37"	1,66"	37"
P2	3,88"	1'41"	3,34"	1'37"
P3	5,72"	2'06"	5,05"	1'26"

Table 6 presents the computational time to for search the best AC. In this case, the necessity of calculating L to spend more computational time than the another three problems present. The difference in results between MORSS algorithm adaptation calculating L and without calculating L occurs because the criterion adopted for calculating L is not the ideal. This criterion doesn't obtain the best results, but the results were better a few times and were near the best in another times.

5. CONCLUSIONS

The main contribution of this paper consists in adapting the MORSS algorithm behavior for solving the established problem. Three other proposed contributions were presented in this work. The first was the adaptation of insertion and local search heuristics in the algorithm for available the eligible products insertion. The second was the establishment of different utility functions from those presented original in Psaraftis *et al* (1985). Finaly, in this work was also propose a criterion to calculate the time horizon length at each iteration of the method. In this way, the parameter L will be not a decision-maker input. The results of method calculating L and without calculating L were presented when solving three difference kinds of problems. The calculation criterion adopted provides the best results in few situations but could be refined to provide better results.

Computational tests in another problems instances were realized, but the results obtained not provided sufficient information about what method presets best performance. Moreover, the L value obtained by calculation will need more refined studies about the real problem involved. The Eq. 15 is only an empirical suggestion for illustrated this idea.

ACKNOWLEDGEMENTS

This research work was made in cooperation CTI/Unicamp and funding by "Fundação de Amparo à Pesquisa do Estado de São Paulo" (FAPESP) grant 95/8306-1.

REFERENCES

Aguilera L.M. (1993). Ordonnancement de production avec couts de changements dependant de la sequence. *Thèse de Doctorat de l'Institut National Polytechnique de Grenoble.* Grenoble, France.

Aguilera L.M., Z. Binder, J.J. Guimarães Ramos and S. H. Takara (1995). Parallel Machines Scheduling with Sequence-Dependent Changeover Costs in an Industrial Application. In: *11th ISPE/IEE/IFAC International Conference on CAD/CAM, Robotics and Factories of the Future, CARS&FOF'95.* Proceedings pp. 703-708. Pereira, Colombia.

Aguilera L.M., Z. Binder, F. Hanada and J.J. Guimarães Ramos (1996). Non-Identical Parallel Machines Scheduling with Sequence-Dependent Changeover Costs in an Industrial Application. In: *Symposium on Discrete Events and Manufacturing Systems Computational Engineering in Systems Applications, CESA'96/IMACS/IEE-SMC.* Proceedings pp. 559-564. Lille, France.

Blazewicz, J., K. Ecker, G. Schmidt and J. Weglarz (1993). *Scheduling in Computer and Manufacturing Systems*. Springer-Verlag, New York.

CPLEX Optimization, Inc. (1994). *Using the CPLEX Callable Library*.

França P.M., M. Gendreau, G. Laporte, F. M. Müller (1996). A Tabu Search Heuristic for the Multi Processor Scheduling Problem with Sequence Dependent Setup Times. *International Journal of Production Economics* 43, pp.79-89.

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

Gonthier, A. (1990). De L'Optimisation des Coûts de Changement de Fabrication vers l'Atelier Logiciel Décentralise Ordonnancement (ALDO). *Thèse de Doctorat de l'Institut National Polytechnique de Grenoble*. Grenoble, France.

Mendes, A.S., F.M. Müller, P.M. França and P. Moscato (1999). Comparing Meta-Heuristic Approaches for Parallel Machine Scheduling Problems with Sequence Dependent Setup Times. Submitted: CARS&FOF' 99.

Psaraftis, H.N., J.B. Orlin, D. Bienstock and P.M. Thompson (1985). Analysis and Solution Algorithms of Sealift Routing and Scheduling Problems: Final report, *Working Paper* No. 1700-85, Sloan School of Management, MIT.

Psaraftis, H.N. (1988). Dynamic Vehicle Routing Problems. In: *Vehicle Routing: Methods and Studies* (B.L. Golden and A.A. Assad, (Ed.)), pp. 223-248. Elsevier Science Publishers B. V., North-Holland.