WALTER MARCONDES FILHO

DESENVOLVIMENTO E APLICAÇÃO DE ALGORITMOS HEURÍSTICOS AO PROBLEMA DE ALOCAÇÃO DE ESPAÇO FÍSICO EM UNIVERSIDADE

MARINGÁ

2008

WALTER MARCONDES FILHO

DESENVOLVIMENTO E APLICAÇÃO DE ALGORITMOS HEURÍSTICOS AO PROBLEMA DE ALOCAÇÃO DE ESPAÇO FÍSICO EM UNIVERSIDADE

Dissertação apresentada ao Programa de Pós-Graduação em Ciência da Computação da Universidade Estadual de Maringá, como requisito parcial para obtenção do grau de Mestre em Ciência da Computação.

Orientador: Prof. Dr. Ademir Aparecido Constantino.

MARINGÁ

2008

Dados Internacionais de Catalogação-na-Publicação (CIP)
(Biblioteca Central - UEM, Maringá – PR., Brasil)

M321d

Marcondes Filho, Walter
Desenvolvimento e aplicação de algoritmos heurísticos ao problema de alocação de espaço físico em universidade / Walter Marcondes Filho. -- Maringá : [s.n.], 2008.
65 f. : il., figs.

Orientador : Prof. Dr. Ademir Aparecido Constantino.
Dissertação (mestrado) - Universidade Estadual de Maringá, Programa de Pós-Graduação em Ciência da Computação, 2008.

1. Problema de alocação de salas. 2. Heurística. 3. Otimização combinatória. I. Universidade Estadual de Maringá. Programa de Pós-Graduação em Ciência da Computação. II. Título.

CDD 22.ed. 005.741

WALTER MARCONDES FILHO

DESENVOLVIMENTO E APLICAÇÃO DE ALGORITMOS HEURÍSTICOS AO PROBLEMA DE ALOCAÇÃO DE ESPAÇO FÍSICO EM UNIVERSIDADE

Dissertação apresentada ao Programa de Pós-Graduação em Ciência da Computação da Universidade Estadual de Maringá, como requisito parcial para obtenção do grau de Mestre em Ciência da Computação.

Aprovado em 17 / 12 / 2008.

BANCA EXAMINADORA

Prof. Dr. Ademir Aparecido Constantino
Universidade Estadual de Maringá – DIN/UEM

Prof. Dr. Anderson Faustino Silva
Universidade Estadual de Maringá – DIN/UEM

Prof. Dr. Marcone Jamilson Freitas Souza
Universidade Federal de Ouro Preto – ICEB/UFOP

Dedico este trabalho

À minha esposa Valdirene Machado Marcondes, ao meu filho Kauã Machado Marcondes e à minha filha Ruany Machado Marcondes.

Agradecimentos

A Deus acima de tudo.

À minha esposa Valdirene pelo incentivo, carinho e amor ao longo deste período.

Aos meus pais que estiveram me apoiando em todos os momentos.

Ao professor Ademir Aparecido Constantino, meus sinceros agradecimentos, não apenas pela orientação firme e segura demonstrada na elaboração deste trabalho, mas também pelo incentivo, oportunidade, confiança e amizade nesses anos de convivência.

A todos os professores e funcionários do Programa de Pós-Graduação em Ciência da Computação da Universidade Estadual de Maringá.

À Universidade Estadual de Maringá, à Pró-Reitoria de Administração e ao Núcleo de Processamento de Dados que permitiram o meu afastamento para cursar o mestrado.

A todos que direta ou indiretamente, contribuíram para o desenvolvimento e conclusão deste trabalho.

Há homens que lutam um dia e são bons.
Há outros que lutam um ano e são melhores.
Há os que lutam muitos anos e são muito bons.
Porém, há os que lutam toda a vida.
Estes são os imprescindíveis.
(Bertolt Brecht)

Resumo

O Problema de Alocação de Salas em uma instituição acadêmica consiste em distribuir turmas para as devidas salas, respeitando uma série de restrições operacionais e preferências. Neste trabalho é proposta a aplicação de três algoritmos heurísticos. O primeiro consiste na resolução sucessiva de problemas de designação e o segundo na resolução sucessiva de problemas de designação com gargalo, ambos com três fases cada. O terceiro algoritmo é baseado na meta-heurística Busca em Vizinhança Variável. Os testes foram realizados com dados reais de uma universidade e os resultados alcançados pelos três algoritmos foram comparados entre si e com resultados utilizados pela instituição. O primeiro algoritmo, baseado no problema de designação, apresentou os melhores resultados em relação à qualidade da solução e eficiência.

Palavras-chave: Problema de Alocação de Salas, Heurística, Otimização Combinatória.

Abstract

The Classroom Assignment Problem in an academic institution consists in distributing classes for the due rooms, respecting a series of operational restrictions and preferences. In this work, the application of three heuristic algorithms is proposed. The first one consists in a successive resolution of assignment problem and the second in a successive resolution of bottleneck assignment problem, both with three phases each. The third algorithm is based on Variable Neighborhood Search metaheuristic. The tests were realized with real data of a public university and the results reached by the three algorithms were compared among themselves and with those used by the institution. The first algorithm, based on the assignment problem, presented the best results in relation to the quality of the solution and efficiency.
Keywords: Classroom Assignment Problem, Heuristics, Combinatorial Optimization.

Lista de Ilustrações

Lista de Tabelas

Lista de Abreviaturas e Siglas

ASAP	*Automated Scheduling, Optimisation and Planning*
CAP	*Classroom Assignment Problem*
PAS	Problema de Alocação de Salas
PATAT	*Practice and Theory of Automated Timetabling*
VND	*Variable Neighborhood Descent*
VNS	*Variable Neighborhood Search*
PL	Programação Linear
PPL	Problema de Programação Linear
UEM	Universidade Estadual de Maringá

Sumário

CAPÍTULO

1

Introdução

1.1. Considerações Iniciais

A alocação de espaço físico é um problema de otimização que surge na maioria das instituições acadêmicas. O problema consiste em distribuir pessoas (professores, alunos, técnicos, etc.) e outros recursos (equipamentos, aulas, etc.) a espaços físicos atendendo uma série de restrições e preferências. O desafio é gerar soluções ótimas em tempo aceitável do ponto de vista prático. A alocação não eficiente de espaço e recursos pode resultar em custos operacionais, transtornos logísticos, insatisfação, entre outros. Devido à complexidade de resolução do problema, a resolução de forma manual pode ser ineficiente em função do não atendimento de todas as restrições e preferências. Além disso, a má alocação das salas pode causar insatisfação por parte dos usuários (Burke e Varley, 1997).

O Problema de Alocação de Salas (PAS) segundo Schaefer (1999) é a distribuição de aulas para as devidas salas com horários previamente estabelecidos, respeitando-se uma série de restrições, tais como espaço físico e infra-estrutura adequada das salas, acessibilidade para os alunos, entre outras. As restrições ou necessidades de recursos dificultam a distribuição de salas, portanto, a solução manual deste problema é um processo demorado que requer vários dias e pode gerar transtornos para o início das aulas por falta de alocação, transferências, fluxo acentuado de alunos em deslocamento e alocações incorretas em relação à distância e tempo percorrido para acesso entre uma sala e outra.

O PAS é um problema de otimização combinatória que pode ser encontrado na literatura. Conforme Even, Itai e Shamir (1976), Bardadyn (1996) e Carter e Tovey (1992), o PAS faz parte de um contexto mais geral denominado de programação de horários de cursos universitários (*course timetabling*).

Para o problema estudado existe uma conferência internacional sobre prática e teoria de automatização de *timetabling* (PATAT), que acontece a cada dois anos disponibilizando um fórum para troca de idéias. Outro exemplo é o grupo do projeto ASAP da Universidade de Nottingham UK, que desenvolve uma pesquisa sobre técnicas heurísticas para resolução da distribuição efetiva em escritórios e instituições de ensino de grande porte.

Em grande parte dos casos estudados, o problema tem sido classificado como NP-difícil, o que na prática, pode inviabilizar sua resolução por processo manual ou por algoritmos exatos para grande volume de dados, pois o espaço de soluções cresce exponencialmente não sendo possível encontrar uma solução ótima em tempo razoável. Nesses casos, uma alternativa viável é a resolução por técnicas heurísticas ou algoritmos aproximativos que, apesar de não encontrarem soluções ótimas podem gerar boas soluções com um custo computacional razoável e adequadas às necessidades da aplicação (Bardadyn, 1996).

Na literatura, encontram-se vários trabalhos para resolução do PAS por meio de algoritmos heurísticos: Castro (2003), Sávio (2005) e Beyrouthy *et al.* (2006) utilizam o algoritmo *Simulated Annealing*; Souza, Martins e Araújo (2002a) apresentaram um algoritmo que combina as técnicas de *Simulated Annealing* e *Busca Tabu*; Souza, Martins e Araújo (2002b) propõem um método baseado em trocas sistemáticas de vizinhança, conhecido como Pesquisa em Vizinhança Variável (*Variable Neighborhood Search*, VNS); Landa (2003) desenvolve um algoritmo híbrido, adaptando heurísticas conhecidas (Melhoria da Iteratividade, *Simulated Annealing*, *Busca Tabu* e Algoritmos Genéticos) baseado no modelo de busca local cooperativa; Lico (2006) resolve o problema por meio de um grafo bipartido e gera uma solução construtiva com o Algoritmo Húngaro.

Existem situações em que o problema é resolvido por algoritmos exatos, como é o caso de Steiner *et al.* (2000) que obtém resultados com utilização do Algoritmo Húngaro; Lopes e Schoeffel (2002) apresentam uma solução por modelos de programação matemática.

Apesar da existência de alguns trabalhos relacionados na literatura, existe uma grande dificuldade de aproveitamento de modelos e algoritmos em função das particularidades tratadas em cada caso. As principais variações podem ser observadas nas definições dos objetivos e restrições do problema.

Este trabalho propõe três algoritmos heurísticos para a resolução do PAS. Os dois primeiros utilizam os modelos do problema de designação e designação com gargalo, respectivamente, com três fases. O terceiro algoritmo utiliza uma solução inicial da obtida pela fase 1 do primeiro algoritmo e aplica um algoritmo baseado na meta-heurística VNS.

A hipótese é de que os resultados dos algoritmos propostos em comparação com o método de alocação manual da instituição acadêmica, consigam demonstrar uma melhor solução para o PAS, provendo assim, uma melhora na qualidade da alocação com um tempo computacional aceitável, viabilizando sua utilização na instituição acadêmica.

1.2. Objetivos

Os objetivos deste trabalho estão divididos em: objetivo geral e objetivos específicos.

1.2.1. Objetivo Geral

O objetivo geral do trabalho é desenvolver três algoritmos heurísticos para resolução do Problema de Alocação de Salas em uma instituição acadêmica.

1.2.2. Objetivos Específicos

Para o êxito do desenvolvimento deste trabalho e alcance do objetivo geral têm-se os seguintes objetivos específicos:

- Realizar uma revisão bibliográfica sobre o Problema de Alocação de Salas;
- Projetar e implementar algoritmos heurísticos com base nos modelos do problema de designação e designação com gargalo, e na meta-heurística VNS para a resolução do Problema de Alocação de Salas;
- Apresentar um estudo comparativo entre os resultados obtidos por meio dos algoritmos propostos e o método manual da instituição acadêmica;

1.3. Estrutura do Trabalho

Com o objetivo de demonstrar os procedimentos utilizados na condução das atividades desta pesquisa é destacado o roteiro de desenvolvimento do trabalho. O trabalho está dividido em seis capítulos. No primeiro capítulo, têm-se a introdução, objetivos e estrutura do trabalho. O

segundo capítulo apresenta a fundamentação teórica com estudo de temas relacionados à Otimização Combinatória. O terceiro capítulo apresenta uma revisão da literatura sobre o problema de alocação de salas, incluindo modelos e algoritmos para resolução do problema. O quarto capítulo descreve os algoritmos propostos. No quinto capítulo são apresentados e analisados os resultados obtidos. Finalmente, o sexto capítulo apresenta as conclusões do trabalho e recomendações para trabalhos futuros. A Figura 1.1. ilustra a estrutura do trabalho:

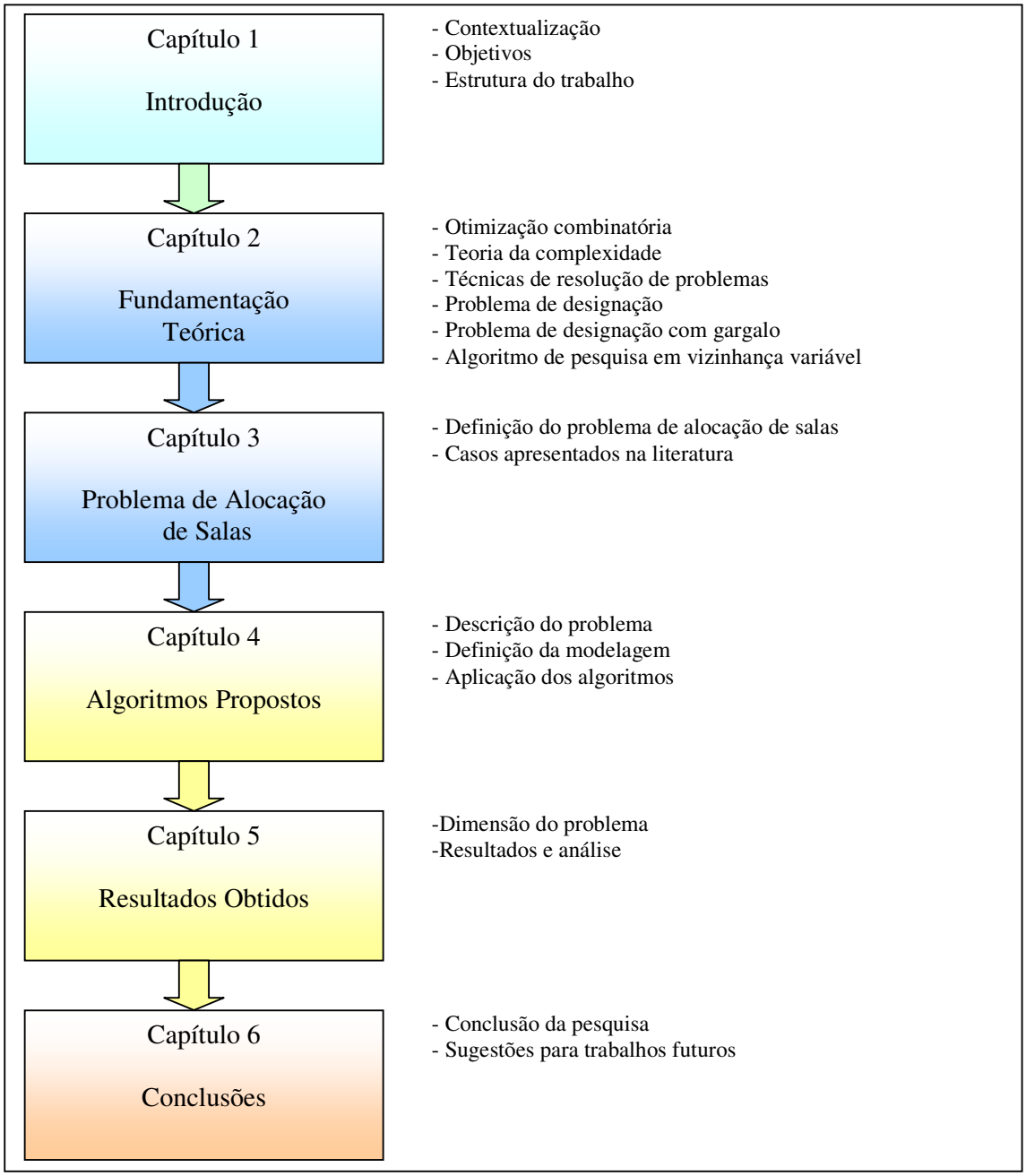


Figura 1.1. Estrutura do trabalho

CAPÍTULO

2

Fundamentação Teórica

2.1. Considerações Iniciais

Este capítulo apresenta uma revisão da literatura sobre temas relacionados aos algoritmos propostos para dar embasamento teórico ao estudo de otimização. Assim, define-se uma base conceitual sobre otimização combinatória, teoria da complexidade e técnicas de resolução de problemas. Também é descrito o problema de designação, problema de designação com gargalo e a meta-heurística VNS.

2.2. Otimização Combinatória

Com a busca de otimizar os processos produtivos e gerenciais para um melhor aproveitamento de recursos e conseqüentemente a diminuição de custos, organizações têm utilizado ferramentas da Pesquisa Operacional e métodos de solução de problemas da Otimização Combinatória.

Segundo Moreira (2007), a Pesquisa Operacional baseia-se em métodos científicos para resolver problemas de como conduzir e coordenar certas operações em diversas organizações. Desta forma, a Pesquisa Operacional é uma área de estudos em que são aplicados métodos analíticos procurando obter a melhor solução para um determinado problema, com o objetivo de ajudar os executivos nas tomadas de decisões.

Em The Guide (2008), com a utilização da modelagem matemática para analisar situações mais complexas, a Pesquisa Operacional permite aos executivos tomar decisões

mais efetivas, construir sistemas mais produtivos, obter previsões de resultados e estimativas de risco com dados mais completos e ferramentas modernas de técnicas de decisão.

Para Cordenonsi (2008) a Pesquisa Operacional é o conjunto de modelos, técnicas, algoritmos e resultados eficientes para problemas de otimização em aplicações práticas nas organizações. O autor ainda relata o termo Otimização Combinatória como um ramo da matemática e da ciência da computação que analisa problemas de otimização em geral dentro de um limite de tempo. Os problemas de otimização podem ter diferentes resultados podendo ser avaliados e comparados, efetuando a otimização. Para a otimização pretende-se encontrar um valor ótimo para um determinado problema, respeitando determinadas restrições com um custo associado. Assim, o objetivo da otimização combinatória é encontrar um resultado onde o custo seja mínimo.

Para resolver um problema de Otimização Combinatória Cordenonsi (2008) relata a possibilidade de combinar ou enumerar todas as possíveis soluções, criando-se todos os subconjuntos existentes a partir do conjunto inicial e das regras de restrições. Esta técnica de combinar todas as soluções e ainda escolher aquela de menor custo, torna-se inviável para aplicações práticas, já que o número de soluções pode ser muito grande. Para ilustrar esse fato, o autor mostra um problema que consiste em separar alguns elementos de um determinado conjunto, em que cada elemento possui um peso e um valor. Estes elementos têm que ser armazenados em uma caixa que possui uma restrição em relação ao peso que ela pode carregar. Desta forma é necessário encontrar um subconjunto de elementos que maximiza o valor que será transportado sem ultrapassar a restrição de peso da caixa. Considerando que existam cinco itens no conjunto, e que o peso máximo que a caixa suporta é 10 kg, conforme abaixo:

A	[valor = 10,	peso = 3]
B	[valor = 3,	peso = 4]
C	[valor = 7,	peso = 6]
D	[valor = 9,	peso = 4]
E	[valor = 2,	peso = 2]

Para Cordenonsi (2008) uma das formas de resolver é testar todas as combinações possíveis. Assim temos os seguintes resultados:

S1.	[A]	v.10	p.3	**S17.**	**[A,B,D]**	**v.22**	**p.11**
S2.	[B]	v.3	p.4	S18.	[A,B,E]	v.15	p.9
S3.	[C]	v.7	p.6	**S19.**	**[A,C,D]**	**v.26**	**p.13**
S4.	[D]	v.9	p.4	**S20.**	**[A,C,E]**	**v.19**	**p.11**
S5.	[E]	v.2	p.2	S21.	[A,D,E]	v.21	p.9
S6.	[A,B]	v.13	p.7	**S22.**	**[B,C,D]**	**v.19**	**p.14**
S7.	[A,C]	v.17	p.9	**S23.**	**[B,C,E]**	**v.12**	**p.12**

S8.	[A,D]	v.19	p.7	S24.	[B,D,E]	v.14	p.10
S9.	[A,E]	v.12	p.5	**S25.**	**[C,D,E]**	**v.18**	**p.12**
S10.	[B,C]	v.10	p.10	**S26.**	**[A,B,C,D]**	**v.29**	**p.17**
S11.	[B,D]	v.12	p.8	**S27.**	**[A,B,C,E]**	**v.22**	**p.15**
S12.	[B,E]	v.5	p.6	**S28.**	**[A,B,D,E]**	**v.24**	**p.13**
S13.	[C,D]	v.16	p.10	**S29.**	**[A,C,D,E]**	**v.28**	**p.15**
S14.	[C,E]	v.9	p.8	**S30.**	**[B,C,D,E]**	**v.21**	**p.16**
S15.	[D,E]	v.11	p.6	**S31.**	**[A,B,C,D,E]**	**v.31**	**p.19**
S16.	**[A,B,C]**	**v.20**	**p.13**				

As soluções em negrito não satisfazem a restrição do problema, pois o peso é maior que a capacidade da caixa. A solução ótima do problema é a 21, com valor 21 para um peso de 9 kg. Este é um problema clássico da Otimização Combinatória denominado problema da mochila (Diaz *et al.*, 1996), cujo número de soluções é dado por:

$$\text{num\_solucoes} = \sum_{i=1}^{n-1} C_{n,i} + 1 \tag{2.1}$$

onde n é o número de elementos no conjunto e C a operação matemática de combinação de n elementos de um determinado conjunto tomado de p a p, sem importar a ordem, dada por:

$$C_{n,p} = \frac{n!}{p!(n-p)!} \tag{2.2}$$

Assim, Cordenonsi (2008) apresenta o calculo do número de possíveis soluções para o problema apresentado como:

$$\text{num\_solucoes} = \frac{5!}{1!(5-1)!} + \frac{5!}{2!(5-2)!} + \frac{5!}{3!(5-3)!} + \frac{5!}{4!(5-4)!} + 1 \tag{2.3}$$

$$\text{num\_solucoes} = \frac{120}{24} + \frac{120}{12} + \frac{120}{12} + \frac{120}{24} + 1 \tag{2.4}$$

$$\text{num\_solucoes} = 5+10+10+5+1 = 31 \tag{2.5}$$

O referido autor ainda demonstra que, por se tratar de um problema combinatório, o número de soluções cresce rapidamente à medida que o número de itens a serem alocados também aumente. A Tabela 2.1 mostra o número de soluções para alguns valores do número n de itens.

Tabela 2.1 – Número de soluções possíveis para o problema da mochila.

Valor de *n*	Número de soluções
5	31
10	1023
20	1048575
50	$1.125 * 10^{15}$
100	$1.269 * 10^{30}$
150	$1.427 * 10^{45}$

Portanto, mesmo utilizando computadores de grande capacidade de processamento, a exposição de todas possíveis soluções para um problema razoável se torna inviável, necessitando de outras técnicas computacionais mais apuradas para resolver problemas práticos.

Conforme Goldbarg e Luna (2005) existe uma grande quantidade de problemas da vida real que podem ser problemas de Otimização Combinatória:

- Problema de particionamento: utilizado para recuperação de informações em banco de dados, alocação de tripulação, distribuição do tráfego de comunicações em satélites, alocações de serviços de emergência e roteamento de petroleiros;
- Problema da árvore geradora mínima: utilizado no projeto de redes de comunicações, roteamento de veículos com função multiobjetivo;
- Problema de roteamento de veículos: utilizado para definir escalas de tripulação, programação de tarefas e tripulação, rotear veículos com restrições;
- Problemas de fluxo: utilizado principalmente em redes de transporte de energia, telecomunicações e escala de motoristas;
- Problema do caixeiro viajante: utilizado na descoberta de rotas de custo mínimo.

Sabe-se que o PAS (Problema de Alocação de Salas) pertence à classe de problemas de Otimização Combinatória (Even, Itai e Shamir, 1976).

Dessa forma, os problemas de Otimização Combinatória são de grande interesse para a melhoria dos processos de negócio, pois proporcionam a redução de custos das organizações. Portanto, o desenvolvimento de modelos e algoritmos na resolução desses problemas práticos é um desafio constante para muitos pesquisadores na investigação científica.

2.3. Teoria da Complexidade

Estabelecer um conjunto de instruções que solucionem certo problema, não significa ter encontrado a solução ideal, pois o objetivo de um algoritmo deve ser solucionar um problema

de forma rápida e econômica. Rápida em relação à realidade para o fim prático do algoritmo e econômica, correspondendo à limitação dos recursos humanos e computacionais. Para analisar um algoritmo, faz-se necessário estabelecer os recursos em termos de memória, tempo de processamento e natureza das operações que o algoritmo irá requerer para desenvolver suas operações (Goldbarg e Luna, 2005).

Segundo Malagutti (2002), a complexidade computacional é o estudo da eficiência dos algoritmos. Do ponto de vista prático, de nada adianta um algoritmo perfeito se sua implementação computacional demora um tempo relativamente alto para ser processada. Assim, uma tarefa simples pode ter um tempo excessivamente longo, de meses ou mesmo anos para sua solução, gerando um algoritmo ineficiente devido ao tempo computacional inaceitável.

Malagutti (2002) ainda relata que para medir a eficiência de um algoritmo é usado o tempo teórico que o programa leva para encontrar uma resposta em função dos dados de entrada. Este cálculo é associado a uma unidade de tempo para cada operação básica que o algoritmo executa. Se o tempo com relação aos dados de entrada for de ordem polinomial, o programa é considerado rápido e se o tempo for de ordem exponencial o programa é considerado lento. Entretanto, existem algoritmos que estão entre estas duas ordens. A Figura 2.1 demonstra a ordem de complexidade.

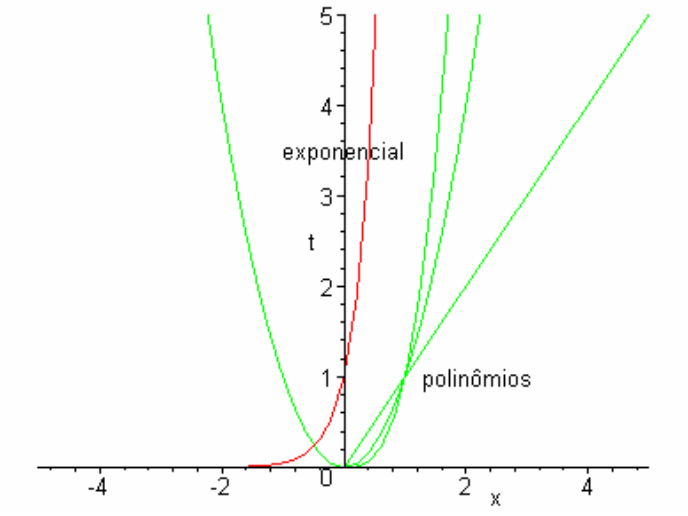


Figura 2.1. Gráfico da ordem de complexidade. (Malagutti, 2002)

A Teoria da Complexidade trata de assuntos da teoria da computação referente à análise da complexidade de algoritmos por meio de considerações matemáticas. Essa teoria torna possível a classificação eficiente da classe em que o problema de otimização combinatória se encontra, podendo determinar o grau de dificuldade para resolução do

problema. Essa classificação é conhecida como teoria de NP-completude para os problemas de Otimização Combinatória. Sua complexidade divide-se em quatro classes: P (*polynomial-time*), NP (*nondeterministic polynomial-time)* , NP-difícil e NP-completo (Garey e Johnson, 1979 e Papadimitriou e Steiglitz, 1982).

Para Viana (1998), antes de caracterizar as quatro classes de problemas, é necessário saber o que é um algoritmo determinístico e um não-determinístico. Dado o fluxo de controle de um algoritmo, entende-se por algoritmo determinístico, se a cada passo que ele der, o próximo passo apresentar apenas uma solução, enquanto que os algoritmos não determinísticos fazem uma escolha aleatória do próximo passo, entre um número fixo de possibilidades, e, conseqüentemente, o fluxo do algoritmo depende da alternativa selecionada gerando um número finito de soluções.

Segundo Cormen *et al.* (2002) e Viana (1998), os problemas pertencentes à classe P são ditos tratáveis computacionalmente quando existe um algoritmo de complexidade polinomial que o resolva, já os demais são considerados intratáveis, ou seja, o número de computações executadas no melhor algoritmo conhecido para o problema cresce exponencialmente em função do tamanho da instância. Uma regra para escolher um algoritmo eficiente é que a função $f(n)$ que estima o tempo de execução não ultrapasse os limites desejados. Por exemplo, sendo n o tamanho de instâncias de entrada do problema, se $f(n) > 10^{10}$ é motivo de preocupação, pois máquinas convencionais com unidade de tempo para instruções e operações da ordem de 10^{-6} segundos, requereriam um tempo de execução em torno de três horas. Se a ordem fosse $f(n) > 10^{14}$ o tempo de execução ultrapassaria três anos.

Conforme Goodrich e Tamassia (2004) as classes de problemas podem ser definidas como:

- P (*Polinomial Time*): problemas que podem ser resolvidos por algoritmos determinísticos polinomiais em função do tamanho da instância, a complexidade do problema cresce polinomialmente em função do tamanho da instância.
- NP (*NonDeterministic Polinomial Time*): problemas que podem ser resolvidos por algoritmos não-determinísticos polinomiais no tamanho da instância. A complexidade do problema cresce exponencialmente em função do tamanho da instância.
- NP-difícil: compostos por problemas caracterizados por haver redução em tempo polinomial a partir de todo problema pertencente à classe NP.
- NP-completo: problemas pertencentes à interseção das classes NP e NP-difícil.

A Figura 2.2 demonstra as classes de problemas:

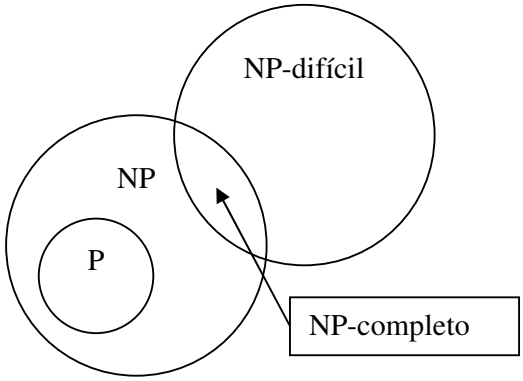


Figura 2.2. Relação entre as classes de problemas.(Cormen et al., 2002)

Campello e Maculan (1994), Bardadyn (1996) e Goldbarg e Luna (2005), relatam que a classe de problemas NP-difícil é um grupo importante de problemas de otimização, para a qual não são conhecidos algoritmos de solução eficiente com complexidade de tempo polinomial em função do tamanho da instância de entrada. Diversos problemas NP-difíceis possuem uma relação com outras áreas de pesquisa e aplicações importantes nas organizações. Pode-se citar o problema do caixeiro viajante, problema geral de roteamento de veículos, problema de particionamento, problema de recobrimento, problema de número cromático em grafos, problema de cobertura de conjunto, problema da mochila, PAS, entre outros.

Com relação à dificuldade para resolução de variados e importantes problemas de Otimização Combinatória, na literatura é proposto o desenvolvimento de algoritmos que forneçam boas soluções em um tempo aceitável.

2.4. Técnicas de Resolução de Problemas

Como descrito na seção 2.2, os problemas de Otimização Combinatória possuem um grande universo de dados, com um número muito extenso de combinações, tornando inviável a análise de todas as possíveis soluções, visto que o tempo computacional para uma análise completa seria extremamente longo. Deste modo, problemas de otimização requerem técnicas específicas.

Segundo Goldbarg e Luna (2005), para muitos problemas de otimização, a obtenção de algoritmos eficientes é difícil. Uma alternativa é a modelagem desses problemas, facilitando a obtenção de algoritmos mais eficientes. Os modelos são representações

simplificadas que preservam para determinadas situações uma equivalência da realidade, tornando os problemas mais fáceis e claros.

O processo de tradução deve ser adequado para identificar os elementos fundamentais da questão e transportá-los para uma representação capaz de ser manipulada por métodos de solução, ou seja, os algoritmos. As dificuldades dos processos de tradução e solução são de naturezas diferentes. Assim, à medida que a tradução produz uma representação mais ou menos tratável pelos métodos existentes, a utilização do modelo é definido. Uma representação simbólica para resolução de problemas pode ser observada na Figura 2.3.

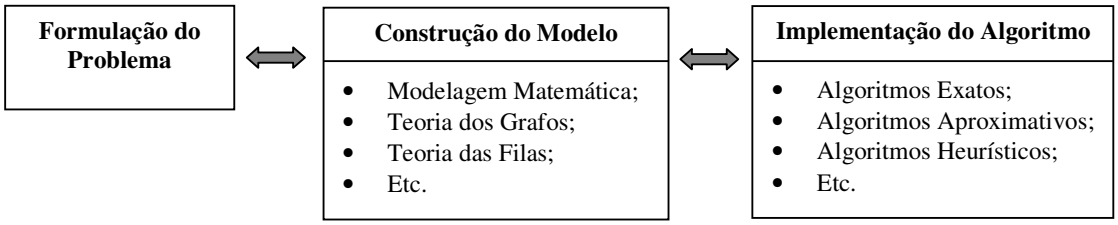


Figura 2.3. Processo de resolução de problemas.(Goldbarg e Luna, 2005)(adaptado).

Conforme Cormen *et al.* (2002), Goldbarg e Luna (2005), Osman (1991) e Viana (1998), podem ser encontradas várias técnicas de resolução de problemas de Otimização Combinatória, a Figura 2.4 demonstra alguns dos principais métodos de resolução, que podem ser classificados de forma geral conforme o tipo de algoritmo, podendo ser:

- Algoritmos Exatos: garantem encontrar uma solução ótima para o problema, mas nem sempre é possível em tempo aceitável. As técnicas mais freqüentemente utilizadas podem ser programação dinâmica, *branch-and-bound*, *branch-and-cut* e etc.
- Algoritmos heurísticos: são procedimentos para resolver problemas por meio de um enfoque intuitivo, em geral racional, no qual a estrutura do problema possa ser interpretada e explorada inteligentemente para obter uma solução razoável em tempo aceitável. Podem ser construtivas ou de melhoramento, não têm prova de convergência e não garantem encontrar a solução ótima,
- Meta-heurística: são procedimentos heurísticos para exploração do espaço de soluções além da otimalidade local, explorando também boas soluções em novas regiões promissoras.
- Algoritmos aproximativos: são aqueles que possuem propriedades de convergência, mas não garantem chegar a uma solução ótima. Assim a solução produzida está dentro de um fator próximo à solução ótima.

- Programação linear e programação inteira: utilizam modelos matemáticos para otimização de uma função linear, denominada de função objetivo, respeitando um conjunto de restrições lineares. Quando todas as variáveis são contínuas, o modelo é dito de programação linear. Se, no entanto, alguma variável assumir valor inteiro, o modelo é dito de programação inteira.

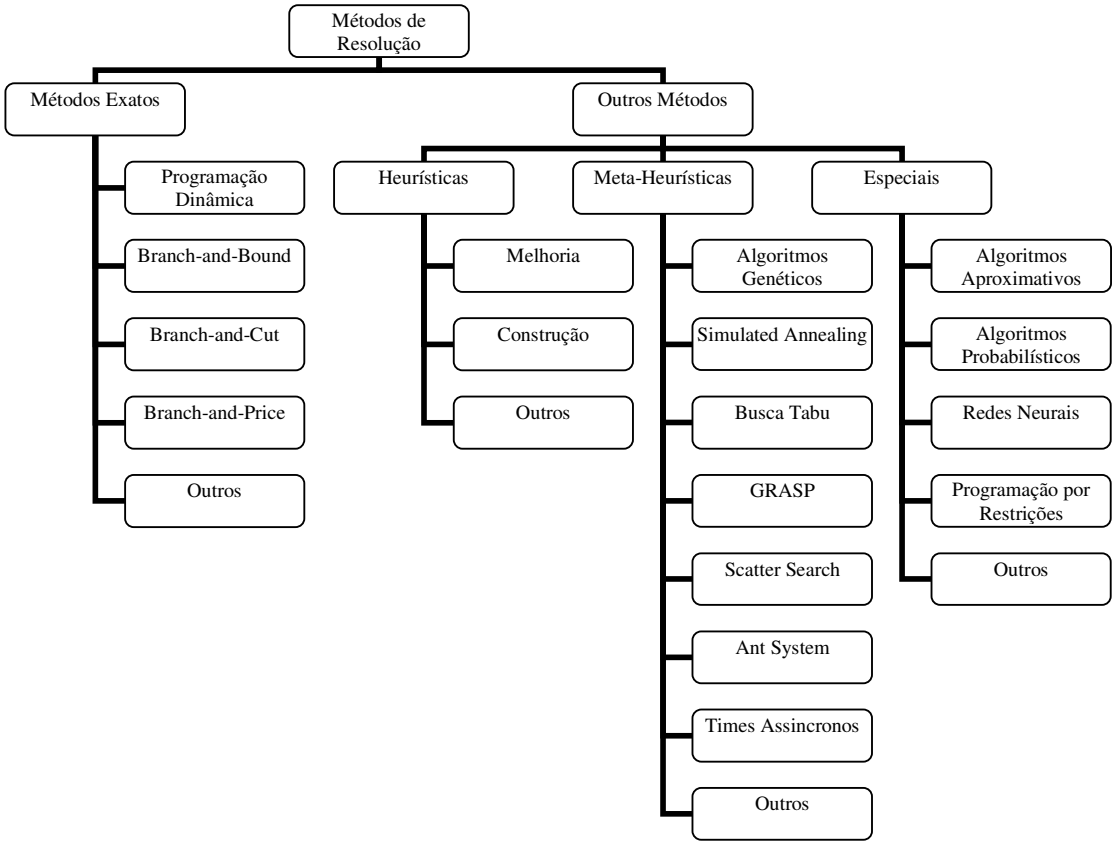


Figura 2.4. Métodos de resolução de problemas

De acordo com Vieira (2006), para resolução de problemas de Otimização Combinatória são utilizados principalmente algoritmos exatos e heurísticos. Se o algoritmo encontra a solução ótima para o problema, isto é, o valor da função objetivo é o melhor que qualquer outro valor encontrado, então temos um algoritmo exato. Mas na prática ocorre a existência de muitos problemas cujo número de soluções cresce de forma exponencial na medida em que aumenta o tamanho do problema. Assim, um algoritmo que não garanta encontrar a solução ótima, mas uma solução viável, isto é, que se aproxime da solução ótima com um tempo computacional aceitável é o mais indicado para resolução desses tipos de problemas de crescimento exponencial.

2.4.1. Programação Linear

Segundo Moreira (2007), a Programação Linear (PL) é um dos modelos matemáticos mais populares, estruturados para resolver problemas que apresentam variáveis que possam ser medidas e cujos relacionamentos possam ser expressos por meio de equações lineares. Na PL existe uma combinação de variáveis que deve ser maximizada ou minimizada. Essa combinação pode ser a expressão do custo de operações industriais, tempo gasto para execução de certas atividades, lucro atingido com a venda de produtos, etc. O autor ainda relata que, durante a formulação do problema, a combinação de variáveis é expressa na forma de uma expressão matemática, conhecida como função objetivo. Assim, um modelo de programação matemática pode ser estruturado para maximizar ou minimizar o resultado da função objetivo. A função objetivo é formulada com base em certo número de restrições ou limitações do mundo real, configuradas pelos próprios dados do problema. Portanto, o objetivo da PL é maximizar ou minimizar a função objetivo e ao mesmo tempo obedecer a todas as restrições.

De acordo com Goldbarg e Luna (2005), a PL é um caso particular dos modelos de otimização em que as variáveis são contínuas e apresentam comportamento linear, tanto em relação às restrições quanto a função objetivo.

Pode-se formular de forma geral o Problema de Programação Linear (PPL) como segue:

Otimizar $z = \sum_{j=1}^{n} c_j x_j$

Sujeito a:

$$\sum_{j=1}^{n} a_{ij} x_j = b_i \qquad i = 1,2,...,m \tag{2.6}$$

$$x_j \geq 0,\ j = 1,2,...,n$$

Com as seguintes notações:

$M=\{1,2,...,m\}$, o conjunto dos índices das restrições do problema;

$N=\{1,2,...,n\}$, o conjunto dos índices das variáveis.

$A=\{a_{ij}\} \equiv$ matriz de restrições;

$A_j \equiv j$-ésima coluna de A;

$x=(x_j), j \in N$ / vetor coluna de n componentes;

$c=(c_j), j \in N$ / vetor linha de n componentes;

$d=(d_j), i \in N$ / vetor coluna de m componentes;

O termo otimizar é utilizado na forma geral do PPL, para genericamente, representar as possibilidades de maximizar ou minimizar a função objetivo. O problema consiste em: dados a matriz *A* e os vetores *b* e *c*, achar o vetor de variáveis contínuas *x* que satisfaça ao conjunto de restrições e que otimize o valor do critério *z* (Goldbarg e Luna, 2005).

Andrade (2007) relata que a PL tornou-se uma das ferramentas mais eficazes para estudos de gestão: organização de transportes, determinação de política de estoques, estudos de fluxos de caixa e investimentos, estudos de sistemas de informações, além dos tradicionais problemas de produção e mistura de componentes.

2.4.2. Heurística e Meta-heurística

De acordo com Cordenonsi (2008), uma heurística é um procedimento desenvolvido por meio de um modelo cognitivo, baseado em regras e na experiência dos desenvolvedores. Ao contrário dos métodos exatos, que buscam encontrar uma forma algorítmica de achar uma solução ótima através da combinação ou busca de todas as soluções possíveis, as heurísticas normalmente tendem a apresentar certo grau de conhecimento acerca do comportamento do problema, gerando um número muito menor de soluções. Ainda conforme o autor, os métodos heurísticos englobam estratégias, procedimentos e métodos aproximativos com o objetivo de encontrar uma boa solução, mesmo que não seja a ótima, em um tempo computacional razoável.

Segundo Rich e Knight (1993) uma heurística é um procedimento para resolver eficientemente muitos problemas difíceis, geralmente é necessário comprometer as exigências de mobilidade e sistematicidade e construir uma estrutura de controle que não garanta encontrar a melhor resposta, mas que quase sempre encontre uma resposta muito boa. A heurística então, é uma técnica que melhora a eficiência de um processo de busca, possivelmente sacrificando pretensões de completeza.

Para Cordenonsi (2008), as heurísticas podem ser divididas, em termos pedagógicos, em construtivas e de melhoramento.

Conforme Cordenonsi (2008) e Vieira (2006), um algoritmo heurístico construtivo, inicia com uma solução vazia e passo a passo agrega componentes (tarefas, arestas, vértices, variáveis, entre outras) à solução do problema, e somente ao final das iterações do algoritmo se obtém uma solução, cujo valor se torna limitante para procura de novas soluções. Algoritmos construtivos não possuem nenhum esquema de *backtracking*, ou seja, após inserir um resultado, não é possível retirá-lo.

Já as heurísticas de melhoramento iniciam com uma solução inicial e trabalham no melhoramento da solução atual, por meio da realização de passos sucessivos. Estes passos realizam a exclusão e inclusão de novos resultados, de forma a pesquisar a vizinhança da solução em busca de uma melhor qualidade. O termo vizinhança se refere aos novos espaços de busca das soluções, que podem ser alcançadas por meio de um movimento. Por movimento em um espaço de busca, entende-se a aplicação de uma regra ou função que altere a solução atual, gerando uma nova solução. As heurísticas de melhoramento possuem um processo de parada, normalmente, quando nenhum outro movimento melhora o resultado atual, o que é considerado um ótimo local. O ótimo local pode ser o ótimo global, ou seja, a melhor solução possível para o problema, mas não há garantias em relação a este fato, como em todos os procedimentos heurísticos. Assim, pode acontecer do algoritmo parar e exibir como resposta um ótimo local, ignorando as possibilidades que poderiam levar o algoritmo a mais tarde, encontrar um ótimo global. A Figura 2.5. ilustra esta última situação.

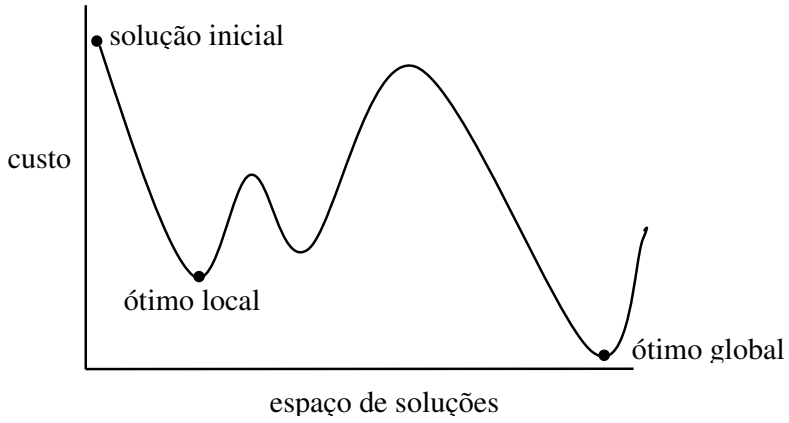


Figura 2.5. Ótimo local e ótimo global.(Cordenonsi, 2008)

Para escapar das armadilhas impostas pelos ótimos locais, foram desenvolvidas técnicas chamadas meta-heurísticas. Uma meta-heurística é um processo iterativo ou de refinamento de soluções do problema que organiza e direciona a heurística pela combinação de diferentes conceitos, podendo manipular uma solução completa, incompleta ou um conjunto de soluções, tendo como objetivo explorar características de boas soluções e até novas regiões promissoras, saindo de um ótimo local (Vieira, 2006).

Dentre as principais meta-heurísticas existentes na literatura, pode-se citar os algoritmos genéticos (Goldberg, 1989), algoritmos meméticos (Moscato, 1989), *Ant Colony Systems* (Dorigo, 1991), *Simulated Annealing* (Kirkpatrick, Gellati, e Vecchi, 1983), Busca Tabu (Glover e Laguna, 1993), *Greedy Randomized Adaptive Procedure* (Feo e Resende,

1994), *Variable Neighborhood Search* (VNS) e *Variable Neighborhood Descent* (VND) (Mladenovic e Hansen, 1999), Redes Neurais (Potvin, 1993), entre outras.

2.5. Problema de Designação

O problema designação é um caso bem definido de um problema de programação linear, clássico de Otimização Combinatória. Segundo Moreira (2007), o problema de designação envolve a atribuição de pessoas a lugares, de tarefas a máquinas, entre outros. Cada atribuição tem uma variável de decisão associada como, por exemplo, o custo, tempo e o lucro. Desenvolve-se uma função objetivo representando o custo total, ou lucro total, ou tempo despendido, a qual deve ser, dependendo do caso, maximizada ou minimizada.

Moreira (2007) relata que, em determinados casos do problema de designação, é necessário utilizar o emparelhamento entre os recursos a serem designados e objetos de designação. Desse modo, se houver mais recursos que objetos de designação, por exemplo, mais máquinas do que trabalhos, a formulação não se altera. Entretanto, o excesso de recursos não será designado. Mas se ocorrer o contrário, isto é, se houver mais objetos de designação do que recurso a designar, uma solução é acrescentar recursos fictícios. Assim, o algoritmo poderá ser executado, mas os objetos de designação que forem atribuídos aos recursos fictícios não serão atendidos.

Conforme Andrade (2007) e Salomão (2005), um exemplo do problema de designação é alocar uma quantidade n de tarefas a m máquinas. Cada tarefa i (i =1, 2,…, n) alocada a uma máquina j (j = 1, 2,…, m) tem um custo c_{ij}. O objetivo do modelo é designar para cada máquina a tarefa adequada, de modo a minimizar o custo total. Caso alguma tarefa não possa ser executada em alguma máquina, o custo correspondente deverá ser um valor muito elevado para impedir essa designação. Um fator importante para resolução do modelo, é a necessidade de tornar a matriz quadrada, adicionando tarefas ou máquinas fictícias, dependendo da relação $n > m$ ou $n < m$. Um exemplo do problema de designação pode ser observado na Figura 2.6.

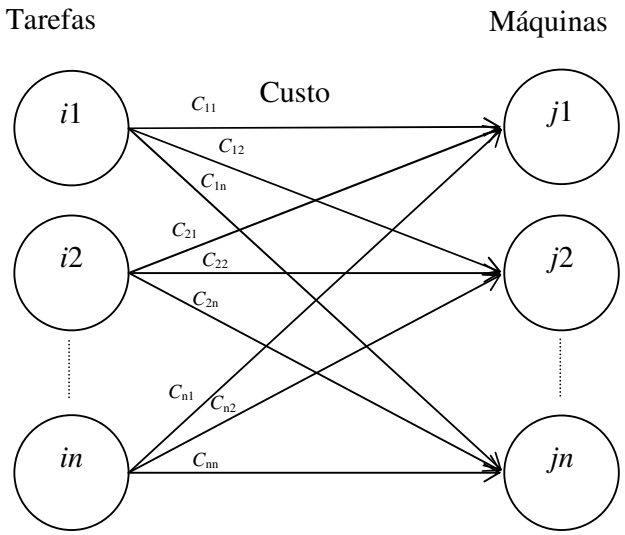


Figura 2.6. Problema de Designação (Andrade, 2007) (adaptado)

Pode-se formular o problema de designação como segue:

$$\text{Min } z = \sum_{i=1}^{n}\sum_{j=1}^{n} c_{ij}.x_{ij}$$

Sujeito a:

$$\sum_{i=1}^{n} x_{ij} = 1 \quad \forall\, j = 1,...,n \tag{2.7}$$

$$\sum_{j=1}^{n} x_{ij} = 1 \quad \forall\, i = 1,...,n$$

$$x_{ij} \in \{0,1\} \qquad \forall i, j = 1,...,n$$

Com as seguintes notações:

c_{ij} o custo associado de atribuir a tarefa *i* à máquina *j*.

r_{ij} um inteiro positivo que representa o tempo que a máquina *j* leva para realizar a tarefa *i*, se a máquina *j* estiver alocada para a tarefa *i.*

b_j um inteiro positivo que representa a capacidade total da máquina *j*.

x_{ij} uma variável definida por:

$$x_{ij} = \begin{cases} 1 \text{ se a tarefa } i \text{ é designada à máquina } j \\ 0 \text{ caso contrário.} \end{cases}$$

Salomão (2005) relata a grande importância prática do problema de designação, que pode estar presente desde um simples planejamento diário de tarefas de uma pessoa, à distribuição de tarefas entre processadores de um sistema de computação paralela (Bokhari, 1987), planejamento de tarefas do telescópio espacial ROSAT (Nowakovski, Schwarzler e Triesch, 1999), entre outros.

Ainda segundo o autor, muitos outros problemas têm sido apresentados na literatura como aplicações do problema de designação, dentre os quais podem ser citados:

- Problema de alocação de salas de aulas (Luan e Yao, 1996);
- Problema de roteamento de veículos (Baker e Sheasby, 1999);
- Problema de alocação de pacientes em vôos médicos (Ruland, 1999);
- Problema de carregamento de caminhões (Pigatti, 2003) e
- Problema de recuperação de blocos de dados a partir de discos paralelos (Aerts, Korst e Spieksma, 2003).

Ainda segundo o autor, existem muitos problemas de decisão que, caso não sejam diretamente problemas de designação, contêm um problema de designação como um subproblema.

O problema de designação é interessante do ponto de vista de pesquisa, pois pode gerar soluções de boa qualidade e com o mínimo de tempo computacional, ou seja, pode ter solução em tempo polinomial (Salomão, 2005).

2.6. Problema de Designação com Gargalo

Conforme Toffolo, Souza e Silva (2005) o problema de designação com gargalo é um modelo apresentado por Carraresi e Gallo (1984) para resolver o problema de rodízio da tripulação. Atribuindo, para cada jornada de trabalho um peso, que represente uma medida do custo da jornada para a tripulação, o peso pode ser dado pela duração da jornada ou pelo tempo gasto pela tripulação para se deslocar da garagem até o ponto onde a jornada se inicia, entre outros. O problema tem como objetivo encontrar um conjunto de jornadas balanceadas ao longo de um dado intervalo de tempo e pode ser formulado por meio do seguinte problema de gargalo: minimizar a soma dos pesos das jornadas designadas à tripulação, que tem o pior custo. Logo, o algoritmo deve encontrar *m* seqüências de jornadas tal que a carga máxima de trabalho seja minimizada.

Constantino (1997) descreve o problema de escala para condutores de ônibus abordado por Carraresi e Gallo (1984). Nesse trabalho é dado um número de *w* de condutores,

um conjunto de turnos que se repetem todos os dias e um horizonte de planejamento de *m* dias. Cada turno recebe um peso p_{ki} (k=1, ..., m; i=1, ..., w) igual ao tempo de duração do turno *i* no dia *k*. O objetivo é construir *w* seqüências individuais de turnos de *m* dias, um turno para cada dia, de maneira que a carga de trabalho fique igualmente distribuída entre os condutores. Portanto, o problema de designação com gargalo é um modelo de atribuição com uma função objetivo do tipo min - max.

Pode-se formular o problema de designação com gargalo como segue:

Min y

Sujeito a:

$$\sum_{i=1}^{w} x_{ij} = 1; \qquad j = 1,..., w; \tag{2.8}$$

$$\sum_{j=1}^{w} x_{ij} = 1; \qquad i = 1,..., w;$$

$$c_{ij} x_{ij} \leq y$$

$$x_{ij} \in \{0,1\} \qquad i, j = 1,..., w;$$

Com as seguintes notações:

c_{ij} o custo associado de atribuir a tarefa *i* ao agente *j*.

x_{ij} uma variável definida por:

$$x_{ij} = \begin{cases} 1 \text{ se a tarefa } i \text{ é designada ao agente } j. \\ 0 \text{ caso contrário.} \end{cases}$$

De acordo com Constantino (1997) e Toffolo *et al.* (2005) o problema de designação com gargalo apresenta uma heurística que resolve problemas de assinalamento com gargalo, que podem facilmente ser ampliáveis para resolução de outros problemas que necessitam de uma designação de forma balanceada, com o mínino de tempo computacional.

2.7. Algoritmo de Pesquisa em Vizinhança Variável

Boaventura-Netto (2003) define a meta-heurística VNS proposta por Mladenovic (1995) e Mladenovic e Hansen (1999) como uma técnica de busca local baseada na definição de um conjunto de vizinhanças a serem utilizadas na busca, seja em uma ordem determinada ou por

seleção aleatória a cada ocorrência adequadamente definida. Assim, define-se um conjunto de vizinhanças N = {Nk, k = 1, ..., kmax} e uma solução inicial x como inicialização. Conforme o autor o algoritmo básico é como segue:

1. $k \leftarrow 1$;
2. enquanto $k \leq k_{max}$ fazer
    a) gerar aleatoriamente um ponto x' na vizinhança k de x;
    b) aplicar uma busca local a partir de x'; seja x" o ótimo local obtido;
    c) se houve melhora de valor, fazer $x \leftarrow x$" e continuar a busca;
    senão, fazer $k \leftarrow k + 1$;

Segundo Boaventura-Netto (2003) pode-se incluir outros critérios de parada, tais como número máximo de iterações ou tempo de processamento. Para o caso da escolha de melhor solução, também é possível utilizar subida-descida, se for admitido um resultado pior. A determinação de x' pode ser feita por meio da comparação de um número dado de soluções geradas aleatoriamente na vizinhança em uso. A troca de vizinhanças pode ser feita ainda por passos modulares ao invés de passos simples, o módulo podendo ainda ser mudado aleatoriamente. A busca local utilizada, por outro lado, pode se beneficiar, de forma independente, das próprias estratégias de troca de vizinhança da VNS.

Para Boaventura-Netto (2003), o VNS é uma técnica que oferece amplas possibilidades de uso como no problema do caixeiro viajante, problema das p-medianas, estudo de árvores relacionadas a fórmulas químicas, problema da coloração de grafos, problema de roteamento, entre outros.

De acordo com Mladenovic e Hansen (2008), o sucesso dessa meta-heurística deve-se principalmente pelas seguintes características:

- Simplicidade: é uma meta-heurística fundamentada em um princípio simples e claro.
- Coerência: os vários passos da heurística para problemas particulares seguem naturalmente o princípio da meta-heurística.
- Eficiência: é uma heurística eficiente, provendo soluções ótimas ou próximas a ótimas, para a maioria dos problemas práticos.
- Efetividade: Para problemas particulares tem condições de prover resultados ótimos ou próximos aos ótimos, com um tempo de uso de CPU moderado.
- Robustez: é efetiva e eficiente para vários tipos de problemas.
- Amigável: é fácil de entender e de usar com poucos parâmetros.

- Inovação: preferivelmente o princípio de eficiência da meta-heurística e efetividade de heurística deve conduzir para novas aplicações. Pode ser utilizada em qualquer tipo de aplicação.
- Generalidade: proporciona bons resultados para uma grande variedade de problemas.
- Interatividade: permite ao usuário incorporar seu conhecimento prévio para melhorar o processo de resolução.
- Multiplicidade: é capaz de apresentar várias soluções próximas a ótima, para a escolha daquela mais conveniente para o usuário.

2.7.1. Busca Local

Busca local em problemas de otimização é uma técnica que constitui uma família de técnicas baseadas em noção de vizinhança. Seja *S* o espaço de pesquisa de um problema de otimização e *f* a função objetivo a minimizar. Assim, uma função *N* é associada a cada solução viável $s \in S$, sua vizinhança $N(s) \subseteq S$. Cada solução $s' \in N(s)$ é chamada de vizinho de *s*. Denomina-se movimento a modificação *m* que transforma *s* em *s*', que esteja em sua vizinhança (Souza, 2000).

Uma heurística de busca local clássica, segundo Souza (2000), é o método de descida. Este método se caracteriza em analisar todos os possíveis vizinhos de uma solução *s* em sua vizinhança *N*(*s*), escolhendo aquele com menor valor para a função objetivo e que melhore o valor da melhor solução até então obtida. O método para assim que um mínimo local é encontrado.

2.8. Considerações Finais

Este capítulo apresentou, um estudo sobre otimização combinatória e as principais técnicas heurísticas referenciadas ao longo do trabalho, para uma melhor compreensão do assunto.

Além disso, demonstrou que a técnica de combinar todas as soluções e ainda escolher aquela de menor custo pode torna-se inviável em relação ao tempo computacional e o grande número de soluções. Sendo necessárias técnicas mais apuradas como, por exemplo, os métodos heurísticos e meta-heurísticas.

CAPÍTULO

3

Problema de Alocação de Salas

3.1. Considerações Iniciais

Este capítulo apresenta uma definição e complexidade computacional do PAS, bem como os casos e modelos apresentados na literatura.

3.2. Definição do Problema de Alocação de Salas

Segundo Bardadyn (1996), o PAS é um problema de otimização combinatória derivado do problema de programação de cursos universitários (*course timetabling*). Para Souza (2000) o problema de programação de cursos universitários consiste em especificar uma seqüência de encontros entre docentes e discentes elaborando um quadro de horários para um determinado período, normalmente durante a semana, satisfazendo restrições de vários tipos.

Conforme Schaefer (1999), já com o quadro de horários de aulas e turmas previamente estabelecidos, o PAS é a distribuição de aulas para as devidas salas, respeitando-se um conjunto de restrições, que podem dificultar a distribuição de salas e até inviabilizar sua solução manual, quanto à existência de um grande número de aulas a serem alocadas.

Para Luan e Yao (1996), o PAS está relacionado com um problema de calendário de aulas, que deve atribuir a cada aula com certas necessidades, uma sala com possíveis limitações em função de seu espaço físico. Como o número de disciplinas ofertadas pode torna-se cada vez maior, a tarefa de alocação de salas torna-se um processo complexo e

exaustivo para um especialista humano, devido à possibilidade de um número limitado de salas disponíveis e a dificuldade para uma boa otimização.

Pertencendo a classe NP-difícil como demonstrado em Carter e Tovey (1992), as exigências e limitações tornam o PAS um problema de otimização combinatória muito complexo. Uma solução ótima para o problema é maximizar a quantidade de salas sem conflitos, ou seja, duas aulas não devem ser atribuídas à mesma sala, ao mesmo tempo. A abordagem mais simples para resolver o PAS é fazer uma pesquisa exaustiva, o que garantirá a otimalidade da solução final. No entanto, a dimensão do espaço de pesquisa para aplicações do mundo real pode excluir esta possibilidade, pois, qualquer algoritmo que tenta encontrar a solução ideal exata globalmente, será impraticável para um grande volume de informação devido a seu tempo computacional proibitivo em termos práticos. Assim, a melhor opção para resolução do PAS é utilizar algoritmos heurísticos, que pode gerar boas soluções com um tempo computacional aceitável (Luan e Yao, 1996).

Conforme Bardadyn (1996), Carter e Tovey (1992) e Landa (2003), a complexidade de muitos problemas de otimização combinatória é descrito por funções exponenciais não sendo conhecido nenhum algoritmo exato em tempo polinomial para resolver esta classe de problemas. Assim, técnicas heurísticas ou algoritmos aproximativos são freqüentemente aplicados por produzir soluções de alta qualidade dentro um tempo razoável.

3.3. Casos Apresentados na Literatura

Na literatura, encontram-se várias aplicações para resolução do PAS por meio de algoritmos heurísticos.

Luan e Yao (1996) propõem uma abordagem baseada em algoritmo genético para o PAS. Uma representação bidimensional do cromossomo é usada no algoritmo genético, que emprega um operador genético nos cromossomos com base em uma coluna que contém os potenciais blocos de um edifício. O algoritmo foi testado em um caso real, da Academia da Força de Defesa Australiana onde a alocação era realizada por método manual por um especialista humano, possuindo como recurso 22 salas de aulas para 153 turmas, programadas em 343 aulas. A distribuição das turmas para as respectivas salas são programadas para uma semana, contendo cada dia 11 períodos de aula, assim, existem 55 períodos na semana. Os autores utilizaram as seguintes restrições:

1. O cronograma de aulas é fixo. Esta condicionante tem um grande impacto sobre as maneiras de gerar as primeiras soluções individuais. Também afeta a execução de importantes operadores do algoritmo genético, tais como crossover e de mutações.
2. O tamanho de uma turma deve ser menor ou igual ao tamanho da sala para a aula. Esta é uma exigência rígida, isto é, não pode ser comprometida.
3. As mesmas aulas agendadas para uma turma devem ser dispostas na mesma sala, quando possível. Esta é uma condicionante flexível, ou seja, ele pode ser ignorado se as circunstâncias não o permitirem.

Um critério importante para os autores é o balanceamento das turmas com relação às salas, como exemplo, em uma turma com 10 alunos não alocá-los em uma sala com 120 lugares. A modelagem do cromossomo utilizado por Luan e Yao (1996), é uma matriz bi-dimensional composta pelo número de períodos na semana e pelo número de salas. Cada entrada da matriz representa a aula que é oferecida em período para uma turma. Uma entrada vazia significa que não foi atribuída nenhuma turma para a sala no momento. A atribuição de turmas para as salas podem ser alteradas para minimizar o custo da função. O custo é composto por duas partes: a primeira parte é a diferença entre o tamanho da turma e o tamanho da sala de aula, e a segunda parte, uma mesma turma estar em diferentes salas em vários períodos. Como pode ser observado na formulação matemática abaixo:

Minimizar $custo = custo_1 + custo_2$ (3.1)

$$custo_1 = \sum_{i \in J} \sum_{j \in J} (r_i - l_{ij}) \quad (3.2)$$

Sujeito a:

$$\sum_{i \in J} l_{ij} \leq r, l_{ij} \quad (3.3)$$

$$custo_2 = A * (n_{lr} - n_l)$$

Sendo:

i = para uma sala

j = para uma turma.

$I = (1,2,..., m)$, para um conjunto de salas.

$J = (1,2 ,..., n)$, para um conjunto de turmas.

$(r1, r2, ..., rm)$, para o tamanho das salas.

l_{ij} = para o custo de alocar uma sala i para uma turma j em determinador período.

A = coeficiente ajustável para penalidade.

n_{lr} = para o número de combinações da turma com mesma sala.

$n_{l=}$ para o número de turmas.

Os experimentos de Luan e Yao (1996) mostraram que os resultados do algoritmo genético são melhores do que os produzidos pelo perito usando o mesmo conjunto de restrições e critérios.

Souza, Martins e Araújo (2002a) têm como objetivo resolver o PAS respeitando uma série de requisitos, classificados em requisitos rígidos aos quais devem ser satisfeitos e em requisitos flexíveis cujo atendimento é desejável:

1. Requisitos rígidos ou essenciais.
   a) Em uma mesma sala e horário não pode haver mais de uma aula.
   b) Uma sala não pode receber uma turma cuja quantidade de alunos seja superior a sua capacidade.
   c) Algumas salas têm alguns horários previamente reservados para a realização de outras atividades e nesses horários ficam indisponíveis.
2. Requisitos flexíveis ou não essenciais.
   a) Certas salas têm restrições de uso e a utilização delas deve ser evitada sempre que possível.
   b) Sempre que possível, alocar os alunos de um mesmo curso e período em uma mesma sala.
   c) Utilizar o espaço das salas eficientemente, isto é, evitar alocar aulas de turmas pequenas em salas de maior capacidade.
   d) Se possível, cada uma das salas deve ser deixada vazia em pelo menos um horário ao longo do dia, de forma a possibilitar sua limpeza.

Ainda com Souza, Martins e Araújo (2002a), uma solução é representada por uma matriz $S=(s_{ij})_{mxn}$, onde m representa o número de horários reservados para realização da aulas e n o número de salas disponíveis. Em cada célula s_{ij} é colocado o número da turma t alocada ao horário i e sala j. Uma célula vazia indica que a sala j está desocupada no horário i. Para Souza, Martins e Araújo (2002a) uma solução s pode ser medida com base em duas componentes, uma de inviabilidade $g(s)$, a qual mede o não atendimento aos requisitos rígidos, e outra de qualidade $h(s)$, a qual avalia o não atendimento aos requisitos considerados flexíveis. A função objetivo f que associa cada solução s do espaço de soluções a um número real $f(s)$ deve ser minimizada. Como segue:

$$\text{Minimizar} \quad f(s) = g(s) + h(s) \tag{3.4}$$

A componente $g(s)$ especifica o nível de inviabilidade de uma solução s, é avaliada por:

$$g(s) = \sum_{k=1}^{k} \alpha_k I_k \tag{3.5}$$

Onde:

K = número de medidas de inviabilidade*;*
I_k = valor da *k-ésima* medida de inviabilidade;
α_κ = peso associado à *k-ésima* medida de inviabilidade.

$h(s)$ especifica a qualidade de uma solução *s,* é avaliada por:

$$h(s) = \sum_{\int=1}^{L} \beta_\int Q_\int \tag{3.6}$$

Onde:

L= número de medidas de qualidade*;*
Q_l = valor da *l-ésima* medida de qualidade;
β_l = peso associado *l-ésima* medida qualidade.

Assim, Souza, Martins e Araújo (2002a) relatam uma experiência com a utilização das técnicas *Simulated Annealing* e Busca Tabu na resolução do PAS e propõem uma metodologia que combina essas duas técnicas, em que uma solução inicial é construída com base na fase de construção do método *GRASP* (Feo e Resende, 1995). Essa solução é então submetida ao método *Simulated Annealing* e a solução resultante deste é refinada pelo método Busca Tabu. A idéia de combinar o mecanismo de construção *GRASP* e os métodos *Simulated Annealing* e Busca Tabu surgiu após a verificação dos seguintes fatos: em uma solução inicial de baixa qualidade, o método Busca Tabu necessitava de muito tempo de processamento para alcançar uma boa solução; com uma boa solução inicial o método Busca Tabu conseguia encontrar soluções de melhor qualidade do que aquelas obtidas pelo método *Simulated Annealing*; apenas o mecanismo de construção GRASP não era suficiente para gerar uma boa solução inicial e o método *Simulated Annealing* conseguia gerar uma boa solução inicial com um baixo custo computacional. A base de dados utilizada foi a do Instituto de Ciências Exatas e Biológicas (ICEB) da Universidade Federal de Ouro Preto (UFOP), com 20 salas de aula teórica, que recebe em média, 1200 alunos por semestre e oferece aproximadamente 250 turmas de disciplinas nos horários matutino, vespertino e noturno.

A técnica *Simulated Annealing* é utilizada na resolução do PAS no Instituto de Ciências Exatas e Biológicas (ICEB) da Universidade Federal de Ouro Preto (UFOP). A função objetivo utilizada é mesma proposta por Souza, Martins e Araújo (2002a), que visa principalmente a eliminação de inviabilidades, no caso, a alocação de turmas em salas cujas capacidades não as comportem e encontrar uma utilização eficiente do espaço minimizando a

quantidade de carteiras não utilizadas nas salas, além de tentar agrupar todas as aulas semanais de uma dada turma em uma mesma sala de aula. Em relação aos resultados, é proposta a utilização de uma técnica híbrida *Simulated Annealing* e Busca Tabu testada em Souza, Martins e Araújo (2002a) que se mostrou superior às técnicas *Simulated Annealing* e Busca Tabu utilizadas isoladamente (Castro, 2003).

Sávio (2005) desenvolve um estudo e implementação para o PAS, mediante a técnica *Simulated Annealing*, com o propósito de aceitar movimentos de piora para escapar de ótimos locais. O algoritmo consegue produzir bons resultados ao atender a maioria dos requisitos e eliminar um número satisfatório de inviabilidades. O autor utilizou como forma de avaliar uma alocação a observação crítica dos requisitos rígidos e flexíveis conforme descrito em Souza, Martins e Araújo (2002a). Para implementação do algoritmo, foram consideradas 3 (três) instâncias de testes, o caso real do Instituto de Ciências Exatas e Biológicas (ICEB) da Universidade Federal de Ouro Preto (UFOP) e outras duas instâncias fictícias. O algoritmo demonstrou uma flexibilidade para executar o programa em outras instâncias.

Souza, Martins e Araújo (2002b) propõem a mesma técnica utilizada para a elaboração do método VNS aplicado neste trabalho, diferenciando nos requisitos e na estrutura de vizinhança. Os referidos autores descrevem que o método VNS possui fácil implementação e que requer a definição de algumas estruturas de vizinhança, ao contrário de outras heurísticas, como Busca Tabu e *Simulated Annealing*, que requerem diversos parâmetros. Assim, o método é aplicado no Instituto de Ciências Exatas e Biológicas (ICEB) da Universidade Federal de Ouro Preto (UFOP). Geram uma solução inicial com base na fase de construção do método *GRASP,* que é então submetida ao método VNS, para uma solução *s,* atingir uma solução *s*' são utilizados dois tipos de movimento: realocação e troca, definindo três estruturas diferentes de vizinhança: $N^{(1)}(s)$, $N^{(2)}(s)$ e $N^{(3)}(s)$. O movimento de realocação consiste em realocar as aulas de uma dada turma e sala a outra sala que esteja vazia nos horários das aulas. Para a realização desse movimento é exigido que a sala que receberá as aulas de uma turma esteja disponível nos horários das aulas. O conjunto de todos os vizinhos de *s* gerados por meio de movimentos de realocação define a estrutura de vizinhança $N^{(1)}(s)$. O movimento de troca consiste em trocar de sala as aulas de duas turmas realizadas em um mesmo bloco de horários. Para realização desse movimento exige-se que nos horários envolvidos as salas estejam vazias ou com aulas apenas das turmas relacionadas com a operação. Assim, o conjunto de todos os vizinhos de s gerados a partir de movimentos de troca define a estrutura de vizinhança $N^{(2)}(s)$. Considera-se a terceira estrutura de vizinhança, $N^{(3)}(s)$, na qual diz-se que uma solução $s' \in N(s)$ é um vizinho de *s* se ela pode ser acessada a partir destra por meio

de um movimento ou de realocação ou de troca. Duas versões da fase de busca local deste método são testadas por Souza, Martins e Araújo (2002b). Na primeira versão, faz-se uma busca local usando o VND. Na segunda, procura-se apenas o melhor vizinho na estrutura de vizinhança corrente da solução em análise.

Uma investigação sobre aplicação de técnicas heurísticas para resolver o problema de alocação de espaço em instituições acadêmicas é proposta por Landa (2003). Com o objetivo de distribuir salas disponíveis entre um grupo de entidades (equipe de funcionários, estudantes de pesquisa, laboratórios de informática, etc.) de modo que o espaço seja utilizado de forma eficaz e os requisitos flexíveis sejam satisfeitos tanto quanto possíveis. O autor utiliza os seguintes requisitos:

1. Uma sala não pode receber uma entidade cuja quantidade seja superior a sua capacidade. Requisito rígido.
2. Conforme o tipo de entidade, não deve compartilhar a sala com outras diferentes entidades. Requisito rígido ou flexível.
3. Algumas entidades devem ser alocadas de preferência para salas específicas. Requisito flexível.
4. Uma entidade específica deve ser alocada adjacente a outra. Requisito rígido ou flexível.
5. Uma entidade específica deve ser alocada longe de outra entidade ou de uma certa sala. Requisito rígido ou flexível.
6. Duas entidades específicas devem ser alocadas na mesma sala. Requisito flexível.
7. Um grupo de pessoas ou entidades deveriam ser alocadas próximas. Requisito flexível.

A formulação matemática proposta por Landa (2003), pode ser observada baixo:

Minimizar $F(x) = (F1(x) + F2(x))$ (3.7)

Sujeito a:

$$\sum_{i=1}^{m} x(i, j) = 1 \quad j = 1,2,...,n \tag{3.8}$$

$$Z(k) = \text{verdadeiro } k = 1,2,...,h$$

Onde:

$$F1(x) = \sum_{i=1}^{m} (WP(i) + OP(i) \tag{3.9}$$

$$F2(x) = \sum_{r=1}^{s} SCP(r) \tag{3.10}$$

A equação (3.9) e (3.10) controla o valor dos requisitos flexíveis respectivamente. $WP(i)$ expressa a penalidade se estiver sobrando capacidade na sala, enquanto $OP(i)$ expressa a penalidade se não houver capacidade na sala.

Para que i_{th} seja alocado com sobra de espaço na sala:

$$c(i) > \sum_{j=1}^{n} w(j)x(i, j) \tag{3.11}$$

Então a penalidade é determinada por:

$$WP(i)= c(i) - \sum_{j=1}^{n} w(j)x(i, j) \tag{3.12}$$

Para que i_{th} seja alocado com falta de espaço na sala:

$$c(i) < \sum_{j=1}^{n} w(j)x(i, j) \tag{3.13}$$

Então a penalidade é determinada por:

$$OP(i)= 2\left(\sum_{j=1}^{n} w(j)x(i, j) - c(i) \right) \tag{3.14}$$

Para:

$SCP(r)$ = penalidade aplicada para r_{th} em relação aos requisitos flexíveis.

m = número de salas disponíveis.

n = número de entidades para alocar.

h = número de requisitos rígidos, para $Z(k)$ = verdadeiro.

s = número de requisitos flexíveis, para $Z(r)$ = verdadeiro.

$c(i)$ = capacidade ou tamanho da sala i

$w(j)$ = exigência espacial de entidade j

$x(i,j)$ = 1 se a entidade j é alocada para a sala i, 0 caso contrário.

Uma solução para Landa (2003) é representada por um vetor $\pi = [\pi(1), \pi(2),..., \pi(j)]$, onde cada elemento $\pi(j) \in \{1,2,...,m\}$ para $j = 1,2,...,n$ indica a sala para o qual a entidade de j_{th} foi alocada. O autor Propõe e compara várias heurísticas para criação da solução inicial e exploração da vizinhança. Desenvolve um algoritmo híbrido, adaptando heurísticas conhecidas (Melhoria da Iteratividade, Simulated Annealing, Busca Tabu e Algoritmos Genéticos) baseado no modelo de procura local cooperativa. Este modelo promove a cooperação de uma população local por meio de mecanismos para compartilhar a informação durante a procura. O algoritmo é aplicado na Universidade de Nottingham, Universidade de Nottingham Trent e na Universidade de Wolverhampton.

Beyrouthy et al. (2006) desenvolvem um estudo que revela que em muitas instituições as salas não foram ocupadas de forma eficaz, principalmente, pela falta de utilização em relação às horas disponíveis de aula. Isto motiva o autor a duas metas: entender os fatores que geram a baixa utilização das salas e desenvolver métodos para prover um melhor planejamento do espaço. O objetivo é maximizar a utilização de salas de uma instituição acadêmica, incluindo salas para conferências e seminários, permitindo satisfazer aos seguintes requisitos rígidos:

1. O total de alunos de uma turma não deve exceder a capacidade da sala e a turma alocada para uma sala não deve exceder o seu horário;
2. Duas ou mais turmas não podem compartilhar a mesma sala no mesmo horário.

A intenção do autor é calcular a utilização de tempo disponível de horas aulas que realmente são utilizadas em termos de percentuais:

$$\text{Utilização} = \frac{horas\ aulas\ utilizadas}{horas\ aulas\ disponíveis} \tag{3.15}$$

O número total de horas aulas é determinado por B:

$$B = \sum_{i,t} s_{i,t} \tag{3.16}$$

Desde que:

$$S_{i,t} \leq C_i \qquad \text{para todo } i,t$$

Onde, C_i é a capacidade da sala i, e $S_{i,t}$ o número de estudantes alocados em uma sala i em um período de aula t.

O número máximo de horas aulas é determinado por B_M, quando $B \leq B_M$:

$$B_M = \sum_{i,t} c_i \tag{3.17}$$

Geralmente, a utilização está definida por meio de "ocupações" e "freqüências", a ocupação $O_{i,t}$ da sala i no tempo t é o uso fracionário naquele momento.

$$O_{i,t} = \frac{S_{i,t}}{C_i} \tag{3.18}$$

A ocupação O_i de uma sala i, está definida como uma alocação má definida para todo período de aula ocupado. Suponha que para a sala i o número de horas aulas é t_i e o número de horas aulas ocupada é t_i^{occ}.

$$O_i = \frac{1}{t_i^{occ}} \sum_t O_{i,t} \tag{3.19}$$

O uso da freqüência F_i, para uma determinada sala i, está definida como a fração de

seu período de aula.

$$F_i = \frac{t_i^{occ}}{t_i} \tag{3.20}$$

U_i é o produto da ocupação e freqüência da utilização da sala *i*.

$$U_i = F_i\, O_i \tag{3.21}$$

Assim:

$$U_i = \frac{\sum_t S_{i,t}}{\sum_t C_i}$$

Para, obter um utilização global é necessário combinar as utilizações de salas diferentes. Foi levado em consideração a média ponderada para as salas, determinado por U^W.

$$U^W = \frac{\sum_i W_i U_i}{\sum_i W_i} \tag{3.22}$$

O peso em ralação a capacidade de quarto, $W_i = C_i$, foi levado também em consideração:

$$U = \frac{\sum_{i,t} s_{i,t}}{\sum_{i,t} c_i} = \frac{B}{B_M} \tag{3.23}$$

A freqüência global de *F,* de uma solução é calculada como:

$$F = \frac{período\ usados}{período\ disponíveis} \tag{3.24}$$

Assim, um algoritmo construtivo é usado por Beyrouthy et al. (2006) para produzir uma solução inicial, em seguida, para melhorar a solução usa-se uma heurística de busca local aplicando o Simulated Annealing. Como exemplo do problema baseado em dados reais os autores aplicaram a técnica proposta na Universidade de Sydney, Austrália.

Com utilização de programação matemática e grafos, Lico (2006) utiliza uma abordagem e modelagem do problema semelhante à escolhida para resolução do problema de designação e designação com gargalo propostos neste trabalho, diferenciando no tipo de aulas para alocação, nos requisitos e principalmente por não se tratar de uma aplicação voltada para o usuário final, e sim, um experimento acadêmico com características apenas construtivas. Lico (2006) resolve o problema de designação, tendo como objetivo encontrar um conjunto de atribuições que possam respeitar se possível um conjunto de pré-requisitos como segue:

1. Uma sala deve ser grande o suficiente para comportar todos os alunos matriculados na disciplina/turma;

2. Uma sala deve atender a necessidade de acessibilidade para alunos deficientes matriculados na disciplina/turma;
3. Uma sala deve atender a necessidade de infra-estrutura para o lecionamento de uma disciplina (por exemplo, o uso de laboratórios);
4. Manter uma distância mínima entre a sala alocada à uma disciplina e o ponto de concentração do curso que a possui.

A modelagem do problema ainda conforme Lico (2006) é demonstrada em forma de um grafo bipartido, onde os vértices de um dos lados do grafo representam o conjunto de recursos disponíveis e o outro o conjunto de entidades existentes. Assim, a função objetivo que determina os custos gerados baseia-se no valor quantitativo de cada parte que compõe o custo e o multiplica pelo seu peso relacional, somando, no final, todos os resultados destas multiplicações e produzindo assim o custo da aresta, conforme equação (3.25):

$$c_{ij} = \sum_{k \in C} Q(i,j)_k P_k \tag{3.25}$$

Onde:

c_{ij} = representa o custo de se atribuir a turma i à sala j;

k = índice para cada custo ou característica;

C = conjunto de características procuradas;

$Q(i,j)$ = valor quantitativo da característica k para os vértices i da turma e o vértice j da sala;

P = peso da característica.

Assim, Lico (2006) a partir da modelagem utiliza um Algoritmo Húngaro para a otimização do processo de alocação do espaço físico entre todas as disciplinas oferecidas aos alunos. O Algoritmo Húngaro utilizado teve apenas características construtivas, não foi abordado o desenvolvimento do algoritmo heurístico melhorativo. Como estudo de caso foram utilizados dados da Universidade Estadual de Maringá (UEM).

Para alguns casos particulares, como os relatados em Steiner *et al.* (2000) e Lopes e Schoeffel (2002), o PAS pode ser resolvido em tempo polinomial.

Assim, Steiner *et al.* (2000) alocam turmas às salas com objetivo de minimizar uma matriz de custo do problema de designação, utilizando um Algoritmo Húngaro. Para a instância de teste foram utilizados apenas os departamentos de informática, matemática, física, química, estatística e desenho da Universidade Federal do Paraná (UFPR).

Lopes e Schoeffel (2002) desenvolveram um sistema para a alocação de salas de aulas para a Universidade Regional de Blumenau (FURB) utilizando apenas cinco turmas e cinco

salas, respeitando-se algumas restrições:

1. Capacidade das salas em relação ao número de alunos de uma turma;
2. Localização do bloco didático;
3. Condições de acesso para deficientes físicos;
4. Necessidades de recursos especiais para as aulas.

O modelo de designação é utilizado por Lopes e Schoeffel (2002), visam minimizar uma função objetivo, decorrente de cálculos com pesos atribuídos as restrições em relação a cada turma alocada em uma sala, gerando uma matriz de valores ou custos. Assim, uma solução é obtida por meio do algoritmo problema de designação baseado na proposta de Shamblin e Stevens (1979) proveniente da pesquisa operacional.

Como se pode observar nos casos apresentados na literatura, os autores consideram o PAS um problema de otimização com multicritérios, e utilizam para determinar a qualidade de uma alocação uma série de requisitos e restrições que devem ser satisfeitos. Contudo, não há uma padronização dos requisitos e restrições a serem considerados, pois, estes podem variar para atender as necessidades específicas de diferentes instituições.

3.4. Considerações Finais

Este capítulo apresentou uma definição do PAS como um problema prático da área de Otimização Combinatória, com o objetivo de designação de aulas com horários e turmas previamente definidos em salas, respeitando um conjunto de restrições.

O PAS é classificado como NP-difícil, sendo necessárias técnicas mais inteligentes do que simplesmente enumerar todas as possíveis soluções e armazenar a de menor custo. Portanto, na literatura muitos autores utilizam técnicas heurísticas e meta-heurísticas, conseguindo bons resultados em tempo computacional aceitável.

CAPÍTULO

4

Algoritmos Propostos

4.1. Considerações Iniciais

Este capítulo apresenta três algoritmos heurísticos para a resolução do PAS. O primeiro deles utiliza o modelo do problema de designação, o segundo o modelo de designação com gargalo e o terceiro a meta-herística VNS.

4.2. Descrição do Problema

Este trabalho tem como referência a Universidade Estadual de Maringá, que por razões administrativas é dividida em centros administrativos, que por sua vez reúnem os departamentos afetos. Os departamentos são responsáveis pelo oferecimento de disciplinas e coordenação de cursos de sua competência. O método de gestão acadêmica utilizada pela instituição é o regime seriado anual, cujas disciplinas estão dispostas em série, com oferta anual. Assim, o aluno ao matricular-se está automaticamente matriculado em todas as disciplinas da série do ano letivo.

A instituição oferece 49 (quarenta e nove) cursos de graduação, com aproximadamente 2.500 (duas mil e quinhentas) disciplinas/turmas ofertadas pelos 34 (trinta e quatro) departamentos divididos em 7 (sete) centros administrativos, a fim de atender, aproximadamente, 16.500 (dezesseis mil e quinhentos) alunos de graduação matriculados, utilizando quase 200 salas para aulas teóricas, além das salas especiais ou laboratórios de aulas práticas. As aulas práticas que requerem laboratórios e salas especiais são alocadas

pelos próprios departamentos nos quais os cursos estão lotados e não fazem parte do PAS.

Apesar dessa divisão administrativa, a alocação das salas de aulas é de competência da administração central da instituição. A quantidade de salas e os horários de aulas são determinados pelos colegiados de curso para atender aos alunos matriculados. Estes horários de aulas são enviados à área de gestão acadêmica central para o processo de alocação das turmas às salas. Normalmente, são enviados meses antes do início do período letivo, mas podem ocorrer transferências e ajustes de alunos alguns dias antes do início das aulas. Esta situação agrava o problema, pois, alocações anteriores devem ser modificadas e normalmente ocorrem transtornos nessa realocação de turmas, gerando um problema administrativo operacional.

Para o processo de alocação de salas, surgem restrições ou necessidades de recursos que dificultam a distribuição de salas. Assim, devem ser observados vários requisitos:

1. Não pode haver mais de uma aula em uma mesma sala e horário, exceto as turmas com aulas teóricas, resultante da junção de turmas de aulas práticas;
2. A acessibilidade da turma em relação à sala de aula deve ser atendida quando houver alunos com necessidades especiais presentes na turma;
3. Exceto as disciplinas indicadas pela coordenação do curso, uma sala não pode receber uma turma cuja quantidade de alunos seja superior à sua capacidade;
4. Cada curso deve possuir uma área geográfica para suas atividades acadêmicas, servindo como referência para alocação de salas de aulas para as disciplinas ofertadas. O objetivo é concentrar as turmas de um mesmo curso dentro de uma área geográfica do campus.
5. Alocar turmas em salas numeradas de acordo com ordem da turma na série do curso;
6. Agrupar todas as aulas semanais de uma turma preferencialmente em uma mesma sala de aula.

O objetivo é alocar todas as turmas de todas as disciplinas e cursos em salas de aulas distribuídas pelo campus dentro dos horários predefinidos, maximizando a concentração de alunos de um mesmo curso dentro de uma área geográfica, minimizando, com isso, o deslocamento de alunos dentro de campus e atendendo às restrições supracitadas. Diante da complexidade e quantidade de possíveis alocações, o presente trabalho propõe a aplicação de algoritmos heurísticos para a resolução deste Problema de Alocação de Salas, que permitam a otimização das atividades operacionais e logísticas da instituição acadêmica.

Além de otimizar o uso do recurso físico, esta proposta visa atender aspectos logísticos relacionados ao deslocamento dos alunos entre salas, aspectos sociais relacionados com o

agrupamento dos alunos dentro de um mesmo espaço geográfico e aspectos operacionais relacionados com o agrupamento e seqüência das numerações das salas conforme a série da turma.

4.3. Definições

Esta seção apresenta algumas definições que serão usadas na descrição dos algoritmos nas próximas seções.

Define-se como *módulo* um intervalo de horário no qual normalmente as aulas geminadas são ministradas. Em cada dia da semana existem 6 (seis) módulos, totalizando 34 (trinta quatro) módulos durante a semana, conforme a Tabela 4.1. Esses 34 módulos são suficientes para fazer o planejamento de alocação de espaço físico para o ano letivo inteiro, uma vez que as alocações serão as mesmas para todas as semanas durante todo o ano letivo. Portanto, basta achar a solução para uma semana.

Tabela 4.1. Definição dos módulos para uma alocação semanal.

Período	Horário	Dias da semana					
		Segunda	Terça	Quarta	Quinta	Sexta	Sábado
Matutino	07:45 - 09:15	1	7	13	19	25	31
	09:30 - 11:45	2	8	14	20	26	32
Vespertino	13:30 - 15:10	3	9	15	21	27	33
	15:30 - 17:10	4	10	16	22	28	34
Noturno	19:30 - 21:10	5	11	17	23	29	-
	21:30 - 23:00	6	12	18	24	30	-

Considerar as seguintes notações para os índices:

m =1, ..., M para os módulos, sendo M=34;

k = 1, ..., K para os cursos;

t = 1, ..., T_m para as turmas com horário no módulo m;

s = 1, ..., S_k para as séries de um curso k;

l = 1, ..., L para as salas.

Compreende-se por bloco didático um edifício ou um aglomerado de salas de aulas. Normalmente, os centros administrativos possuem alguns blocos didáticos de uso prioritário

para os seus cursos. Para cada bloco didático é associada uma coordenada cartesiana (posição central do bloco) denominado de *ponto do bloco*.

Uma característica desejável é tanto o agrupamento de todas as aulas semanais de uma dada turma em uma mesma sala de aula, como também o agrupamento das salas ocupadas pelo mesmo curso e série dentro de uma delimitação geográfica. Denomina-se de *ponto gravitacional* um ponto em coordenadas cartesianas ou um escalar. A função do ponto gravitacional é servir de referência para agrupar cursos, séries e turmas dentro de um espaço geográfico. São considerados três tipos de pontos gravitacionais: em relação ao curso, à série e à turma, sendo assim identificados por PGC_k, PGS_s e PGT_t, respectivamente. Cada PGC_k, corresponde a coordenadas cartesianas extraídas de uma imagem do *layout* do campus, conforme Figura 4.1. Já os pontos gravitacionais PGS_s e PGT_t correspondem ao número de uma sala de aula. Esses valores são utilizados para tentar agrupar as séries e turmas seguindo a ordem de numeração das salas, ou seja, uma série inicial alocada em salas com numeração menor que as séries posteriores. Os pontos gravitacionais são inicializados empiricamente, porém, eles são auto-ajustados durante as execuções dos algoritmos.

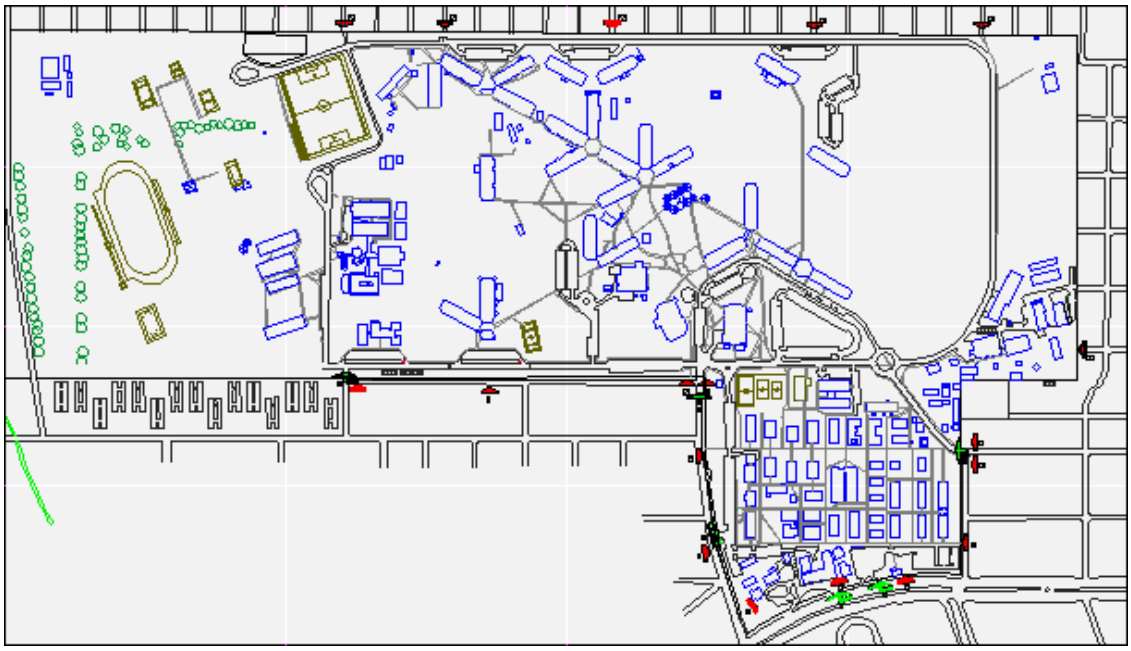


Figura 4.1. Layout do campus sede – Universidade Estadual de Maringá

4.4. Algoritmos Propostos

Esta seção descreve três algoritmos heurísticos. Os dois primeiros são divididos em três fases, sendo que o primeiro algoritmo (PAS-D) é baseado na resolução sucessiva do problema de

designação adaptado ao problema em estudo, enquanto que o segundo algoritmo (PAS-DG) é baseado no problema de designação com gargalo. O terceiro algoritmo (PAS-VNS) é baseado na meta-heurística VNS utilizando a solução inicial obtida pela primeira fase do primeiro algoritmo.

4.4.1. Algoritmo PAS-D

Este algoritmo baseia-se na resolução sucessiva do problema de designação linear. O problema de designação linear é um problema clássico de programação linear equivalente ao emparelhamento perfeito de custo mínimo em um grafo bipartido. Para a resolução do problema foi implementado o algoritmo proposto por Carpaneto e Toth (1987) que combina o método Húngaro e o método do Menor Caminho Aumentante.

Para cada módulo *m* é criada uma instância do problema de designação. A formulação do problema de designação pode ser descrito como:

$$\text{Min } z_m = \sum_{t=1}^{T_m} \sum_{l=1}^{L} c_{tl}^m . x_{tl}^m$$

$$\text{s.a.:} \quad \sum_{t=1}^{T_m} x_{tl}^m = 1, \quad l = 1,..., L \tag{4.1}$$

$$\sum_{l=1}^{L} x_{tl}^m = 1, \quad t = 1,..., T_m$$

$$x_{tl}^m \in \{0,1\}, \quad t = 1,..., T_m, \quad l = 1,..., \text{L}$$

Sendo:

c_{tl}^m = custo da designação da turma *t* à sala *l* para todas as turmas *t* com horário dentro do módulo *m.*

L = número de salas.

T_m = número de turmas *t* com horário dentro do módulo *m.*

x_{tl}^m = 1 se a turma *t* for designada à sala *l* e 0 em caso contrário.

O algoritmo proposto divide-se em três fases executadas em seqüência. O modelo de designação é utilizado em todas as fases, conforme descrito a seguir.

4.4.1.1 Fase 1

Esta fase consiste em resolver *M* problemas de designação. Cada problema de designação é definido pela matriz quadrada $C = [c_{tl}]$; entretanto, o número de turmas pode ser menor que o

número de salas disponíveis. Assim, será considerado $T_m = T_m^{Real} + T_m^{Ficticia} = L$, sendo T_m^{Real} o número real de turmas existentes e $T_m^{Ficticia}$ o número de turmas fictícias criadas para tornar a matriz quadrada. Portanto, a matriz de custo pode ser dividida em duas partes, conforme Figura 4.3, sendo que seus elementos são definidos da seguinte forma:

Parte I: Para $t=1,2,...,\ T_m^{Real}$ e $l=1,2,...,L$, temos $c_{tl} = f(t,l)$, sendo que f é a função que define o custo de cada alocação.

Parte II: Para $t= T_m^{Real} +1$, ..., L (representando turmas fictícias) e $l=1,2,...,L$, c_{tl} é o custo da alocação de uma turma fictícia para uma sala, sendo neste caso $c_{tl} = \infty$.

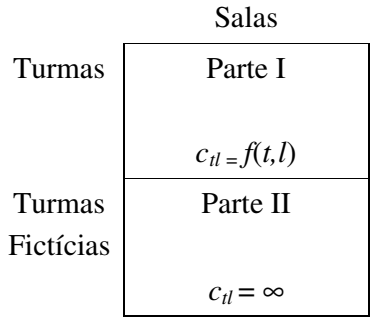


Figura 4.2. Estrutura da matriz de custos

Na formação da Parte I, a função $f(t,l)$ é definida como:

$$f(t,l) = \begin{cases} d_1(t,l) \text{ se a sala } l \in SC(t) \text{ e } Tam(t) \le Cap(l) \\ d_1(t,l) + p_1 \text{ se a sala } l \in SC(t) \text{ e } Tam(t) > Cap(l) \\ d_1(t,l) + p_2 \text{ se a sala } l \notin SC(t) \text{ e } Tam(t) \le Cap(l) \\ d_1(t,l) + p_1 + p_2 \text{ se a sala } l \notin SC(t) \text{ e } Tam(t) > Cap(l) \end{cases} \quad (4.2)$$

Sendo que:

$d_1(t,l)$ = distância euclidiana do PGC_k relacionado com a turma t ao ponto do bloco relacionado com a sala l.

SC(t) é o conjunto das salas de aulas com acessibilidade para a turma t é de uso prioritário dos cursos lotados no centro administrativo ao qual a turma t está lotada.

$Tam(t)$ é o número de alunos na turma t.

$Cap(l)$ é o número de alunos que a sala j comporta.

p_1 é uma penalidade aplicada quando o tamanho da sala não atende a necessidade da turma. O valor foi definido empiricamente por $2x10^3$.

p_2 é uma penalidade definida como a maior distância entre os blocos didáticos que pertençam ao mesmo centro administrativo no qual a turma t está lotada. O objetivo dessa penalidade é forçar que a turma t seja alocada em uma sala l pertencente a $\boldsymbol{SC}(t)$.

Uma iteração desta fase compreende a resolução dos M problemas de designação. Na primeira iteração o PGC_k é definido empiricamente, normalmente um ponto próximo ao bloco didático desejável para o curso. Para as iterações seguintes, o PGC_k recebe o ponto médio das coordenadas entre todos os blocos didáticos utilizados para o curso k na iteração anterior. Este procedimento é repetido até que o PGC_k de todos cursos não sejam modificados.

4.4.1.2 Fase 2

A idéia desta fase é agrupar as turmas da mesma série de um curso seguindo a ordem de numeração das salas, por exemplo, turmas da primeira série alocadas em salas com numeração menor que as turmas da série seguinte.

A estrutura de matriz de custo utilizada nesta fase é igual a da fase anterior, embora a formação dos custos seja um pouco diferente conforme segue:

$$f(t,l) = \begin{cases} d_1(t,l) + d_2(t,l) \text{ se a sala } l \in SC(t) \text{ e } Tam(t) \leq Cap(l) \\ d_1(t,l) + d_2(t,l) + p_1 \text{ se a sala } l \in SC(t) \text{ e } Tam(t) > Cap(l) \\ d_1(t,l) + d_2(t,l) + p_2 \text{ se a sala } l \notin SC(t) \text{ e } Tam(t) \leq Cap(l) \\ d_1(t,l) + d_2(t,l) + p_1 + p_2 \text{ se a sala } l \notin SC(t) \text{ e } Tam(t) > Cap(l) \end{cases} \quad (4.3)$$

Sendo que:

$d_2(t,l) = |PGS_s - Num(l)|$, considerando PGS_s o ponto gravitacional da série s na qual a turma t está vinculada e $Num(l)$ o número da sala l.

Uma iteração desta fase também resolve M problemas de designação. Na primeira iteração $PGS_s = s$, s=1,.., S_k , para o curso k vinculado à turma t. Nas iterações seguintes, PGS_s será o valor médio dos números de todas as salas alocadas para a série s. Este procedimento é repetido até que o PGS_s de todas as séries de todos os cursos não seja modificado.

4.4.1.3 Fase 3

O objetivo desta fase é rearranjar as turmas agrupadas na fase 2 seguindo uma ordem de correspondência da turma em relação à numeração da sala; por exemplo, se a turma de número 1 foi alocada na sala 101, então é desejável que a turma 2 seja alocada na sala 102.

Assim como na fase 2, a estrutura da matriz de custo utilizada é igual à da fase 1, com os custos definidos conforme segue:

$$f(t,l) = \begin{cases} d_1(t,l) + d_2(t,l) + d_3(t,l) \text{ se a sala } l \in SC(t) \text{ e } Tam(t) \leq Cap(l) \\ d_1(t,l) + d_2(t,l) + d_3(t,l) + p_1 \text{ se a sala } l \in SC(t) \text{ e } Tam(t) > Cap(l) \\ d_1(t,l) + d_2(t,l) + d_3(t,l) + p_2 \text{ se a sala } l \notin SC(t) \text{ e } Tam(t) \leq Cap(l) \\ d_1(t,l) + d_2(t,l) + d_3(t,l) + p_1 + p_2 \text{ se a sala } l \notin SC(t) \text{ e } Tam(t) > Cap(l) \end{cases} \quad (4.4)$$

Sendo que:

$d_3(t,l) = |PGT_t - Num(l)|$, considerando PGT_t o ponto gravitacional da turma t e $Num(l)$ o número da sala l.

Uma iteração desta fase também consiste em resolver M problemas de designação. Na primeira iteração $PGT_t = t$. Nas iterações seguintes, PGT_t será o valor médio dos números de todas as salas alocadas para os M módulos da iteração anterior. Este procedimento é repetido até que PGT_t de todas as turmas de todos os cursos não seja modificado.

4.4.2. Algoritmo PAS-DG

Este algoritmo é equivalente ao algoritmo PAS-D, com a diferença que o modelo de designação linear é substituído pelo modelo de designação com gargalo. Para a resolução do problema foi implementado o algoritmo apresentado por Carraresi e Gallo (1984). A formulação do problema de designação com gargalo é descrita como:

Min Z_m

$$\text{s.a.:} \quad \sum_{t=1}^{T_m} x_{tl}^m = 1, \quad l = 1,...,L$$

$$\sum_{l=1}^{L} x_{tl}^m = 1, \quad t = 1,...,T_m \quad (4.5)$$

$$c_{tl}^m x_{tl}^m \leq Z_m, \quad t = 1,...,T_m, \quad l = 1,...,\text{L}$$

$$x_{tl}^m \in \{0,1\}, \quad t = 1,...,T_m, \quad l = 1,...,\text{L}$$

A matriz de custos $[c_{tl}]$ é definida da mesma forma como foi realizada no algoritmo anterior. Enquanto o modelo de designação linear minimiza a soma dos custos de todas as designações, o modelo de designação com gargalo minimiza o custo da maior designação.

4.4.3. Algoritmo PAS-VNS

Este algoritmo é baseado na meta-heurística VNS, o qual explora o espaço de soluções por meio de trocas sistemáticas de estruturas de vizinhanças. Supõe-se que existam R vizinhanças para o problema, $N_1, N_2,..., N_R$. Assim, se a primeira solução não for aprimorante, então a

próxima vizinhança é explorada; caso contrário o algoritmo retorna à vizinhança N_1 e o procedimento recomeça em busca de uma melhor solução. O algoritmo para quando não há mehoria na última vizinhança N_r.

Este algoritmo inicia com uma solução obtida pela fase 1 do algoritmo PAS-D. Foram definidas quatro estruturas de vizinhanças N_r, (r = 1, 2, 3 e 4), sendo que cada vizinhança N_r é obtida explorando os módulos de horários da semana. Realiza-se uma escolha aleatória de um vizinho *s'* dentro da vizinhança $N_r(s)$ da solução *s* corrente. À esse vizinho *s'*, é aplicado o método de busca local, com a estratégia de melhoria iterativa de descida completa, ou seja, todas as soluções vizinhas de *s'* são avaliadas usando a mesma vizinhança $N(s')$, selecionando a melhor solução aprimorante. A função de avaliação de uma solução *s*, $f(s)$, a ser minimizada, é definida como:

$$f(s) = \sum_{m=1}^{34} \sum_{t=1}^{t_m} \sum_{l=1}^{L} c_{tl}^{m} . x_{tl}^{m} \qquad (4.6)$$

Sendo que:

c_{tl}^{m} = uma matriz de custo definida como na fase 3 do algoritmo PAS-D.

x_{tl}^{m} =1 se a turma *t* for designada à sala *l* no módulo *m* e 0 em caso contrário.

Desta forma, se a solução *s*" for melhor que a solução *s* corrente, a busca continua de *s*" iniciando novamente a partir da primeira estrutura de vizinhança $N_1(s)$, caso contrário, continua-se da próxima estrutura de vizinhança $N_{r+1}(s)$.

O critério de parada é acionado quando em cada iteração do algoritmo, todas as vizinhanças são exploradas e o algoritmo pára quando não ocorrer melhoria dentro de um número *MaxInt* de iterações. Foi experimentado *MaxInt*=3.

As estruturas de vizinhanças utilizadas foram:

1. N_1 - para cada módulo *m*, *m*=1, ..., *M,* e para cada bloco didático utilizado na solução: consiste em selecionar uma sala do bloco didático utilizado na solução e efetuar a troca das turmas para uma sala ociosa do mesmo bloco, se for possível.
2. N_2 - para cada módulo *m*, *m*=1, ..., *M,* e para cada bloco didático utilizado na solução: consiste em selecionar uma sala do mesmo bloco utilizada na solução e realizar a troca de turmas entre as duas salas, se for possível.
3. N_3 - para cada módulo *m*, *m*=1, ..., *M,* e para cada curso com aulas no módulo *m*: consiste em selecionar uma sala utilizada pelo mesmo curso (independente do bloco didático) e realizar a troca de turmas entre as duas salas, se for possível.

4. N_4 - para cada módulo m, m=1, ..., M, e para cada série de um curso com aulas no módulo m: consiste em selecionar uma sala utilizada pela série do mesmo curso (independente do bloco didático) e realizar a troca de turmas entre estas salas, se for possível.

O pseudocódigo do algoritmo VNS é representado na Figura 4.3.

```
1. Seja s0 uma solução inicial e R o número de estruturas diferentes de vizinhança;
2. s ← s0; {Solução corrente}
3.enquanto (critério de parada não satisfeito) faça
4.        r ← 1; {Tipo de estrutura de vizinhança}
5.       enquanto (r ≤ R) faça
6.                Gere um vizinho qualquer s' ∈ Nr(s);
7.                s'' ← BuscaLocal(s');
8.                se f(s'') < f(s)
9.                         então s ← s'';
10.                                r ← 1;
11.                        senão r ← r + 1;
12.               fim-se;
13.      fim-enquanto;
14. fim-enquanto;
15. Retorne s;
```

Figura 4.3. Algoritmo VNS

4.5. Considerações Finais

Pode-se observar neste capítulo uma descrição detalhada do problema, as definições importantes para o entendimento da modelagem e a formulação matemática adaptada para os algoritmos apresentados.

Os dados para referência e aplicação dos algoritmos, são de característica prática com um grande volume de informações da Universidade Estadual de Maringá.

CAPÍTULO

5

Resultados Obtidos

5.1. Considerações Iniciais

Todos os testes computacionais foram realizados em um microcomputador PC AMD Atlhon. 2.4 MHz, com 1 GB de RAM e executado sob o sistema operacional Windows XP.

Para a implementação dos algoritmos foi utilizada a linguagem Object Pascal do Delphi 5.0 e o Sistema Gerenciador de Banco de Dados Mysql 5.0 como banco de dados para armazenamento das soluções.

5.2. Dimensão do Problema

Com objetivo de demonstrar uma dimensão aproximada do problema da instituição de ensino superior pública UEM – Universidade Estadual de Maringá, a Tabela 5.1 apresenta um quadro geral das estruturas físicas e humanas.

Tabela 5.1. Quadro geral da UEM.

Descrição	Número
Centros administrativos	7
Departamentos	34
Cursos	49
Turmas	2.500
Salas de aula teóricas	192
Laboratórios	190
Docentes	1390
Discente	16.320

Os algoritmos foram testados com dados reais de três anos letivos consecutivos. As características dos dados utilizados estão apresentadas na Tabela 5.2.

Tabela 5.2. Características das instâncias avaliadas

Ano	Número de Cursos	Número de salas teóricas	Número de módulo de horas das turmas	Número de alunos
2006	47	170	3.927	15.270
2007	48	192	4.016	16.530
2008	49	192	3.978	16.320

5.3. Resultados e Análise dos Resultados

Esta seção aborda os resultados computacionais dos algoritmos propostos e suas análises.

5.3.1. Resultados do Algoritmo PAS-D.

As Tabelas 5.3, 5.4 e 5.5 apresentam alguns dos resultados do algoritmo PAS-D, em suas três fases.

Um dos objetivos do algoritmo PAS-D na fase 1 é maximizar a concentração de alunos de um mesmo curso dentro de uma área geográfica denominada ponto gravitacional PGC. Conforme pode ser observado na Tabela 5.3, a distribuição do bloco didático para o curso 1, conseguiu um bom agrupamento em apenas dois blocos didáticos, mantendo quase todas as aulas no bloco didático E46, e apenas uma no bloco didático 8.

Tabela 5.3. Resultado do algoritmo PAS-D, fase 1

Curso	Disciplina	Série	Turma	Ano	Dia	Módulo	Bloco didático	Sala
1	5	1	1	2008	2	1	**E46**	8
1	1	1	2	2008	2	1	**E46**	10
1	358	1	2	2008	2	1	**E46**	6
1	199	1	3	2008	2	1	**E46**	7
1	2447	2	1	2008	2	1	8	3
1	2451	3	1	2008	2	1	**E46**	5
1	2457	4	1	2008	2	1	**E46**	2
1	2456	4	1	2008	2	1	**E46**	3
1	2455	4	1	2008	2	1	**E46**	4
1	1	1	1	2008	2	2	**E46**	10
1	5	1	2	2008	2	2	**E46**	8
1	1170	1	3	2008	2	2	**E46**	1

Para o algoritmo PAS-D na fase2, o ponto gravitacional *PGS* deve agrupar as séries seguindo a ordem de numeração das salas. Como pode ser observado na Tabela 5.4, o algoritmo designou para a série 1 as salas com menor numeração, ou seja, as salas 1, 2, 3 e 4, e para a série 4 as salas com maior numeração 7, 8 e 10. Com isto houve um bom agrupamento para o ponto gravitacional PGS.

Tabela 5.4. Resultado do algoritmo PAS-D, fase 2

Curso	Disciplina	Série	Turma	Ano	Dia	Módulo	Bloco didático	Sala
1	5	**1**	1	2008	2	1	**E46**	**2**
1	1	**1**	2	2008	2	1	**E46**	**1**
1	358	**1**	2	2008	2	1	**E46**	**4**
1	199	**1**	3	2008	2	1	**E46**	**3**
1	2447	2	1	2008	2	1	8	3
1	2451	3	1	2008	2	1	E46	5
1	2457	**4**	1	2008	2	1	**E46**	**7**
1	2456	**4**	1	2008	2	1	**E46**	**8**
1	2455	**4**	1	2008	2	1	**E46**	**10**
1	1	1	1	2008	2	2	E46	1
1	5	1	2	2008	2	2	E46	2
1	1170	1	3	2008	2	2	E46	4

Na fase3, do algoritmo PAS-D, o ponto gravitacional *PGT* deve tentar agrupar as turmas dentro das séries, também seguindo a ordem de numeração das salas. A Tabela 5.5 mostra que o algoritmo conseguiu bons resultados, alocando de forma ordenada as salas nas turmas e

séries.

Tabela 5.5. Resultado do algoritmo PAS-D, fase 3.

Curso	Disciplina	Série	Turma	Ano	Dia	Módulo	Bloco didático	Sala
1	5	**1**	**1**	2008	2	1	**E46**	**1**
1	1	**1**	**2**	2008	2	1	**E46**	**2**
1	358	**1**	**2**	2008	2	1	**E46**	**3**
1	199	**1**	**3**	2008	2	1	**E46**	**4**
1	2447	2	1	2008	2	1	8	3
1	2451	3	1	2008	2	1	E46	5
1	2457	**4**	**1**	2008	2	1	**E46**	**7**
1	2456	**4**	**1**	2008	2	1	**E46**	**8**
1	2455	**4**	**1**	2008	2	1	**E46**	**10**
1	1	1	1	2008	2	2	E46	1
1	5	1	2	2008	2	2	E46	2
1	1170	1	3	2008	2	2	E46	4

5.3.2. Resultados do Algoritmo PAS-DG

As Tabelas 5.6, 5.7 e 5.8 apresentam alguns dos resultados das três fases do algoritmo PAS-DG.

Conforme pode ser observado na Tabela 5.6, o algoritmo PAS_DG na fase 1 não conseguiu um agrupamento adequado para o ponto gravitacional PGC, distribuindo as turmas entre seis blocos didáticos: 22, 26, 27,.38, D67 e HU.

Tabela 5.6. Resultado do algoritmo PAS-DG, fase 1

Curso	Disciplina	Série	Turma	Ano	Dia	Módulo	Bloco didático	Sala
1	5	1	1	2008	2	1	**38**	3
1	1	1	2	2008	2	1	22	7
1	358	1	2	2008	2	1	**38**	2
1	199	1	3	2008	2	1	27	7
1	2447	2	1	2008	2	1	D67	6
1	2451	3	1	2008	2	1	26	2
1	2457	4	1	2008	2	1	26	3
1	2456	4	1	2008	2	1	HU	1
1	2455	4	1	2008	2	1	HU	2
1	1	1	1	2008	2	2	**38**	3
1	5	1	2	2008	2	2	22	7
1	1170	1	3	2008	2	2	**38**	2

Na Tabela 5.7, os resultados do algoritmo PAS_DG na fase 2 não foram bons em relação ao ponto gravitacional PGS, pois as salas alocadas não ficaram em ordenação com as séries.

Tabela 5.7. Resultado do algoritmo PAS-DG, fase 2

Curso	Disciplina	Série	Turma	Ano	Dia	Módulo	Bloco didático	Sala
1	5	**1**	1	2008	2	1	38	**2**
1	1	**1**	2	2008	2	1	22	**7**
1	358	**1**	2	2008	2	1	27	**3**
1	199	**1**	3	2008	2	1	27	**7**
1	2447	2	1	2008	2	1	E46	3
1	2451	3	1	2008	2	1	26	2
1	2457	4	1	2008	2	1	26	3
1	2456	4	1	2008	2	1	G56	8
1	2455	4	1	2008	2	1	HU	1
1	1	1	1	2008	2	2	27	2
1	5	1	2	2008	2	2	38	4
1	1170	1	3	2008	2	2	E46	3

A Tabela 5.8 mostra que o algoritmo PAS-DG fase 3, para o ponto gravitacional *PGT,* não conseguiu alocar de forma ordenada as salas nas turmas e séries.

Tabela 5.8. Resultado do algoritmo PAS-DG, fase 3.

Curso	Disciplina	Série	Turma	Ano	Dia	Módulo	Bloco didático	Sala
1	5	**1**	**1**	2008	2	1	38	**1**
1	1	**1**	**2**	2008	2	1	38	**2**
1	358	**1**	**2**	2008	2	1	G56	**4**
1	199	**1**	**3**	2008	2	1	G56	**3**
1	2447	2	1	2008	2	1	8	3
1	2451	3	1	2008	2	1	D67	5
1	2457	4	1	2008	2	1	D67	7
1	2456	4	1	2008	2	1	G56	8
1	2455	4	1	2008	2	1	HU	10
1	1	1	1	2008	2	2	4	1
1	5	1	2	2008	2	2	38	2
1	1170	1	3	2008	2	2	33	3

5.3.3. Resultados do Algoritmo PAS-VNS

A Tabela 5.9 apresenta alguns dos resultados do algoritmo PAS-VNS, onde se pode verificar que os mesmos foram satisfatórios para os pontos gravitacionais PGC, PGS e PGT, conseguiu um bom agrupamento para o ponto gravitacional PGC em apenas três blocos didáticos, mantendo principalmente o agrupamento no bloco didático E46 e ordenando as salas conforme as turmas e séries.

Tabela 5.9. Resultado do algoritmo PAS-VNS.

Curso	Disciplina	Série	Turma	Ano	Dia	Módulo	Bloco didático	Sala
1	5	**1**	**1**	2008	2	1	**E46**	**1**
1	1	**1**	**2**	2008	2	1	**E46**	**3**
1	358	**1**	**2**	2008	2	1	**E46**	**2**
1	199	1	3	2008	2	1	D34	1
1	2447	2	1	2008	2	1	D67	6
1	2451	3	1	2008	2	1	**E46**	**5**
1	2457	4	1	2008	2	1	**E46**	**7**
1	2456	4	1	2008	2	1	**E46**	**8**
1	2455	4	1	2008	2	1	**E46**	**6**
1	1	1	1	2008	2	2	**E46**	1
1	5	1	2	2008	2	2	**E46**	2
1	1170	1	3	2008	2	2	**E46**	6

5.3.4. Método Manual

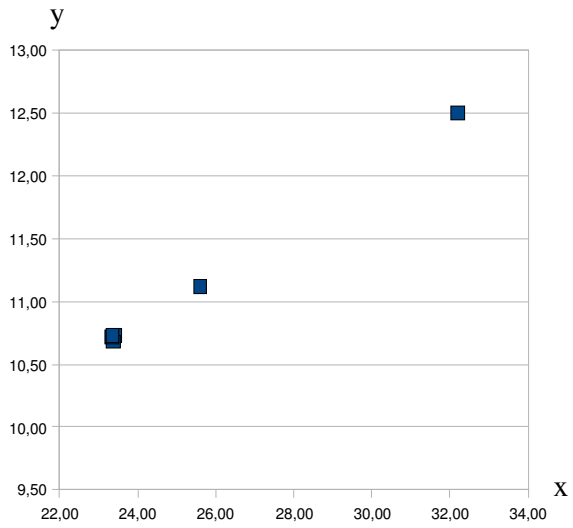
Na Tabela 5.10 são apresentados alguns dos resultados do método manual utilizado pela instituição. Considerando os pontos gravitacionais PGC, PGS e PGT, os resultados do método manual são de certa forma razoáveis, mantendo a distribuição em apenas três blocos didáticos com um maior agrupamento no bloco didático E46 e ordenamento das salas para as turmas e séries.

Tabela 5.10. Resultado do método manual, utilizado pela instituição.

Curso	Disciplina	Série	Turma	Ano	Dia	Módulo	Bloco didático	Sala
1	5	**1**	**1**	2008	2	1	**E46**	**1**
1	1	**1**	**2**	2008	2	1	**E46**	**2**
1	358	1	2	2008	2	1	E34	10
1	199	1	3	2008	2	1	D67	8
1	2447	2	1	2008	2	1	E46	3
1	2451	**3**	**1**	2008	2	1	**E46**	**4**
1	2457	**4**	**1**	2008	2	1	**E46**	**5**
1	2456	**4**	**1**	2008	2	1	**E46**	**6**
1	2455	**4**	**1**	2008	2	1	**E46**	**7**
1	1	1	1	2008	2	2	E46	1
1	5	1	2	2008	2	2	E46	2
1	1170	1	3	2008	2	2	33	2

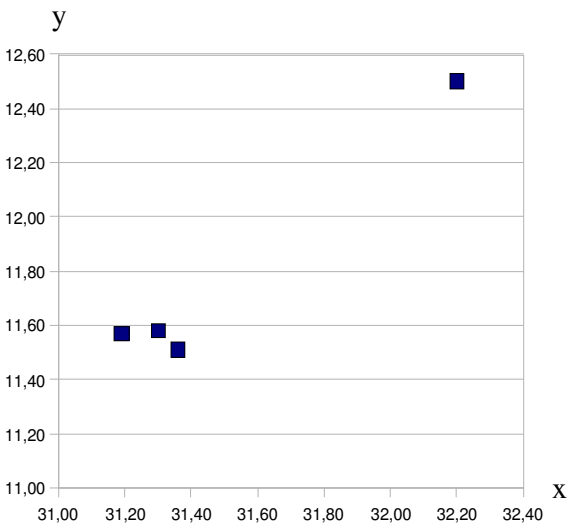
5.3.5. Resultados Gerais

As Figuras 5.1 e 5.2 ilustram a escolha iterativa do ponto gravitacional de um curso durante a fase 1 do algoritmo PAS-D e algoritmo PAS-DG. As figuras mostram que a partir de um ponto gravitacional PGC inicializado empiricamente, são auto-ajustadas as coordenadas durante cada iteração de execução do algoritmo. Esses dados se referem ao curso ciências econômicas, na alocação de 2008.



Iteração	x	y
-	32,20	12,50
0	25,60	11,12
1	23,41	10,73
2	23,32	10,72
3	23,32	10,72
4	23,37	10,68
5	23,36	10,72
6	23,36	10,73

Figura 5.1. Algoritmo PAS-D, ponto gravitacional PGC, ano 2008



Iteração	x	y
-	32,20	12,50
0	31,19	11,57
1	31,30	11,58
2	31,36	11,51

Figura 5.2. Algoritmo PAS-DG, ponto gravitacional PGC, ano 2008

As Tabelas 5.11, 5.12, e 5.13 apresentam os resultados dos algoritmos propostos aplicados para as três instâncias. A coluna Tempo refere-se ao custo computacional para a execução do algoritmo em cada fase. A coluna *Custo total* é resultado da aplicação da função objetivo (equação 4.4) definida na seção 4.4.3. As alocações que atenderam a restrição de capacidade da sala são contabilizadas na coluna de *Alocações favoráveis*, caso contrário, são contabilizadas na coluna *Alocação desfavoráveis*. A coluna *Iterações* corresponde à quantidade de vezes que cada fase foi executada que permitiu melhorar a solução.

Tabela 5.11. Resultados dos três algoritmos sobre os dados de 2006.

Algoritmo	Fase	Tempo hh:mm:ss	Custo total	Alocações favoráveis	Alocações desfavoráveis	Iterações
PAS-D	Fase 1	00:21:08	2.015.496	3.595	332	8
	Fase 2	00:04:52	1.751.583	3.595	332	2
	Fase 3	00:17:27	1.598.637	3.595	332	3
PAS-DG	Fase 1	00:12:18	2.632.801	3.586	341	4
	Fase 2	00:07:15	2.549.259	3.587	340	2
	Fase 3	00:18:05	2.507.436	3.591	336	3
PAS-VNS	Solução inicial	00:21:08	2.015.496	3.595	332	-
	Busca local	00:26:02	1.731.850	3.595	332	3

Tabela 5.12. Resultados dos três algoritmos sobre os dados de 2007

Algoritmo	Fase	Tempo hh:mm:ss	Custo total	Alocações favoráveis	Alocações desfavoráveis	Iterações
PAS-D	Fase 1	00:12:10	1.979.099	3.708	308	6
	Fase 2	00:08:15	1.733.254	3.708	308	3
	Fase 3	00:18:41	1.581.791	3.708	308	3
PAS-DG	Fase 1	00:06:11	2.589.698	3.705	311	2
	Fase 2	00:07:13	2.529.129	3.705	311	2
	Fase 3	00:22:05	2.479.152	3.706	310	4
PAS-VNS	Solução inicial	00:12:10	1.979.099	3.708	308	-
	Busca local	00:35:27	1.693.173	3.708	308	4

Tabela 5.13. Resultados dos três algoritmos sobre os dados de 2008

Algoritmo	Fase	Tempo hh:mm:ss	Custo total	Alocações favoráveis	Alocações desfavoráveis	Iterações
PAS-D	Fase 1	00:11:30	1.960.146	3.677	301	6
	Fase 2	00:09:17	1.697.943	3.677	301	3
	Fase 3	00:17:09	1.580.312	3.677	301	3
PAS-DG	Fase 1	00:06:02	2.589.036	3.666	312	2
	Fase 2	00:05:43	2.506.241	3.669	309	2
	Fase 3	00:18:22	2.493.253	3.671	307	3
PAS-VNS	Solução inicial	00:11:30	1.960.146	3.677	301	-
	Busca local	00:34:47	1.690.372	3.677	301	4

As Tabelas 5.14, 5.15 e 5.16 resumem os resultados alcançados pelos três algoritmos, para efeito de comparação, além de incluir os resultados da solução obtida manualmente pela instituição. Além dos custos calculados pela função objetivo, essas tabelas apresentam as somas das distâncias entre as salas alocadas e o ponto gravitacional de cada curso. Observa-se que essas distâncias são calculadas com base em um desenho do *layout* do campus.

Tabela 5.14. Comparação entre os métodos de alocação para 2006

Método de alocação	Custo total	Alocações favoráveis	Alocações desfavoráveis	Soma das distâncias	Distância mínima	Distância média	Distância máxima
PAS-D	1.598.637	3.595	332	432.855	0	158	1.680
PAS-DG	2.507.436	3.591	336	1.006.023	0	270	1.640
PAS-VNS	1.731.850	3.595	332	577.379	0	223	1.710
Alocação manual	2.394.585	3.293	634	1.011.494	0	269	1.760

Tabela 5.15. Comparação entre os métodos de alocação para 2007

Método de alocação	Custo total	Alocações favoráveis	Alocações desfavoráveis	Soma das distâncias	Distância mínima	Distância média	Distância máxima
PAS-D	1.581.791	3.708	308	506.620	0	172	1.762
PAS-DG	2.479.152	3.706	310	1.152.620	0	289	1.724
PAS-VNS	1.693.173	3.708	308	673.983	0	227	1.762
Alocação manual	2.361.845	3.385	631	1.100.367	0	276	1.775

Tabela 5.16. Comparação entre os métodos de alocação para 2008

Método de alocação	Custo total	Alocações favoráveis	Alocações desfavoráveis	Soma das distâncias	Distância mínima	Distância média	Distância máxima
PAS-D	1.580.312	3.677	301	504.543	0	167	1724
PAS-DG	2.493.253	3.671	307	1.149.892	0	286	1724
PAS-VNS	1.690.372	3.677	301	673.057	0	225	1724
Alocação manual	2.360.918	3.348	630	1.099.214	0	273	1775

Nas Figuras 5.3, 5.4, 5.5 e 5.6 são mostrados os valores dos custos finais encontrados pelos algoritmos propostos e pelo método manual para cada um dos cursos de graduação, referentes à alocação de 2008. Pode-se perceber que o resultado do algoritmo PAS-D demonstra seis cursos com mais problemas de alocação, o algoritmo PAS-DG com quatorze cursos, o algoritmo PAS-VNS oito cursos e o método manual com o pior resultado, demonstra dezesseis cursos com mais problemas.

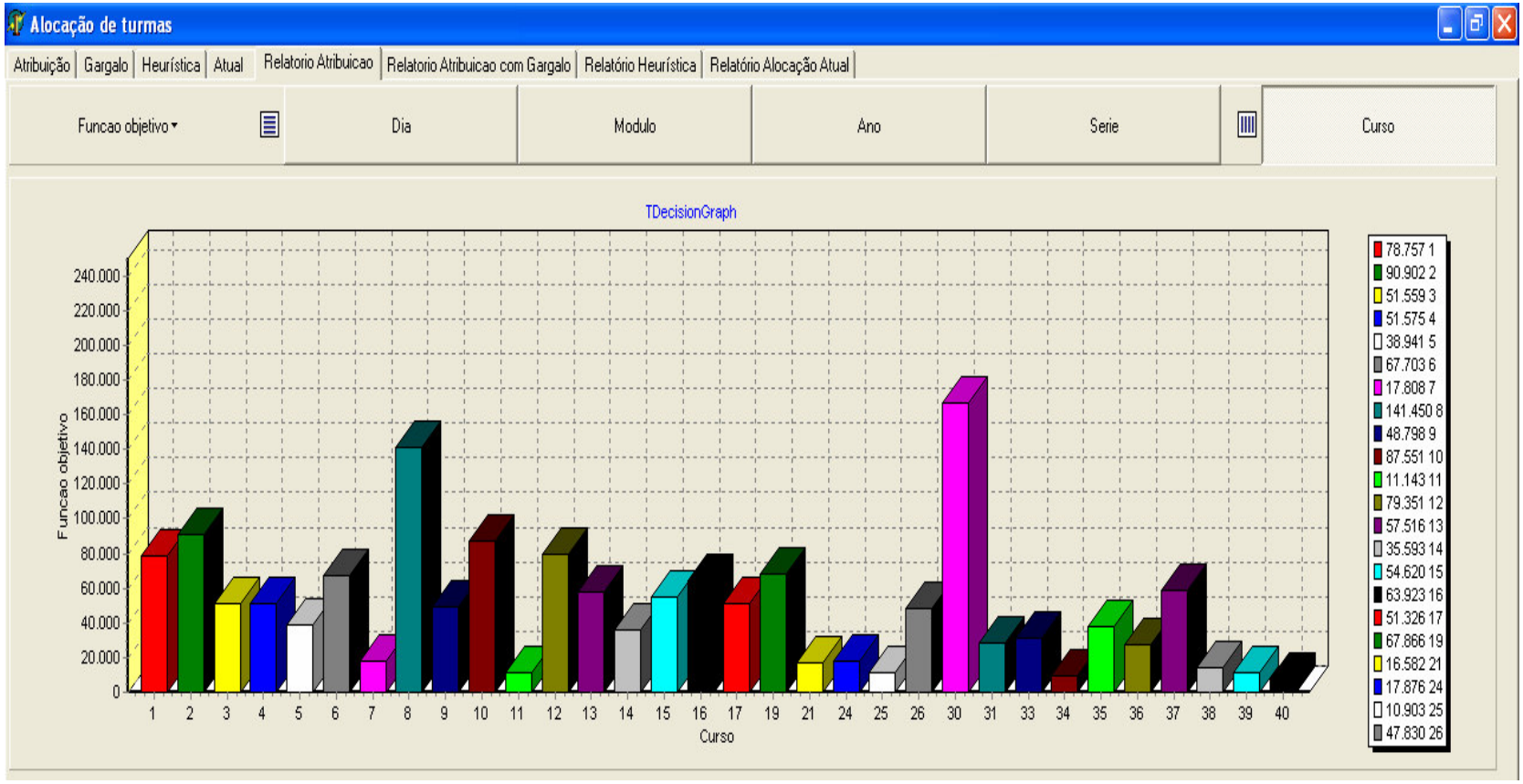


Figura 5.3. Gráfico de alocação de cursos do algoritmo PAS-D, ano 2008

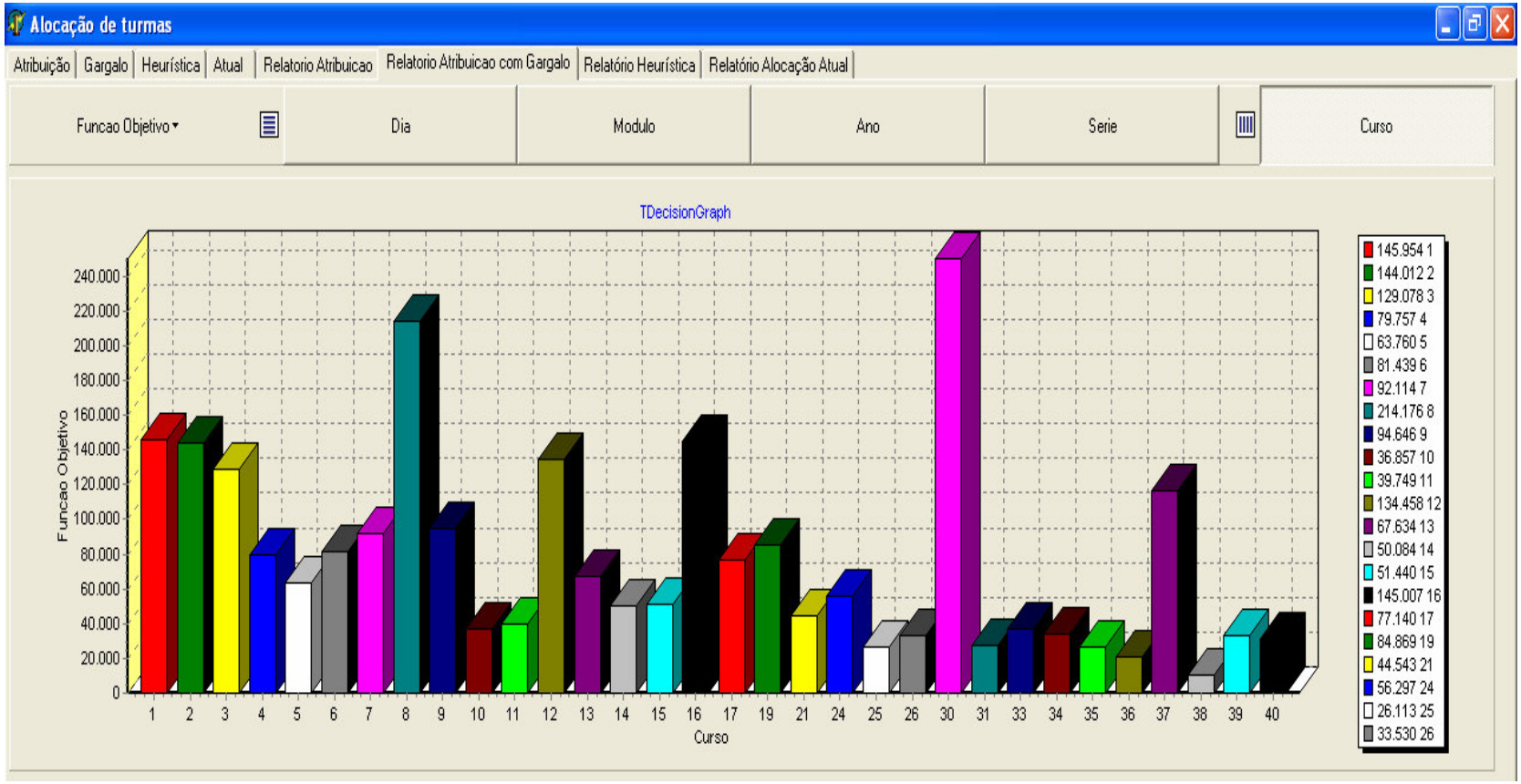


Figura 5.4. Gráfico de alocação de cursos do algoritmo PAS-DG, ano 2008

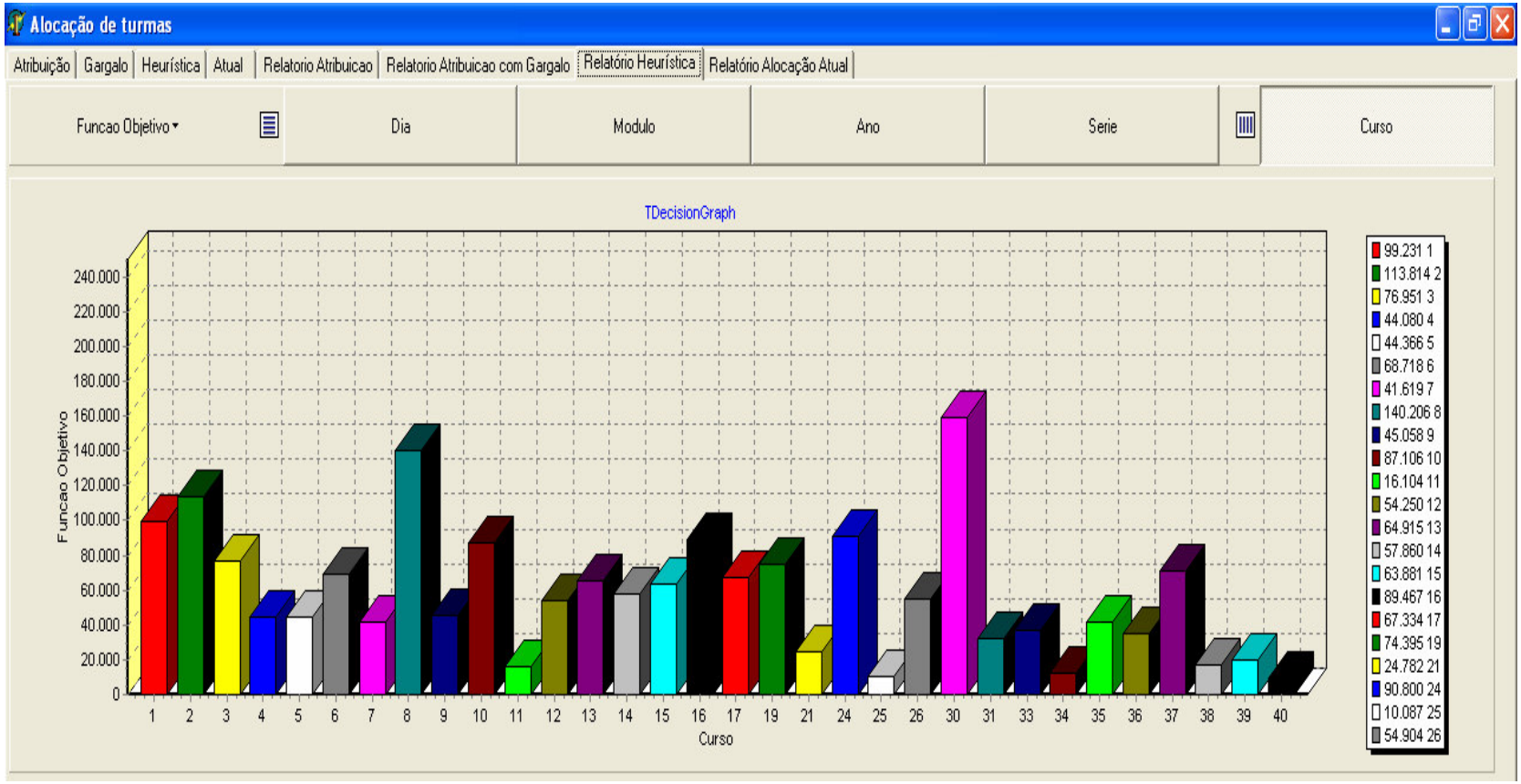


Figura 5.5. Gráfico de alocação de cursos do algoritmo PAS-VNS, ano 2008

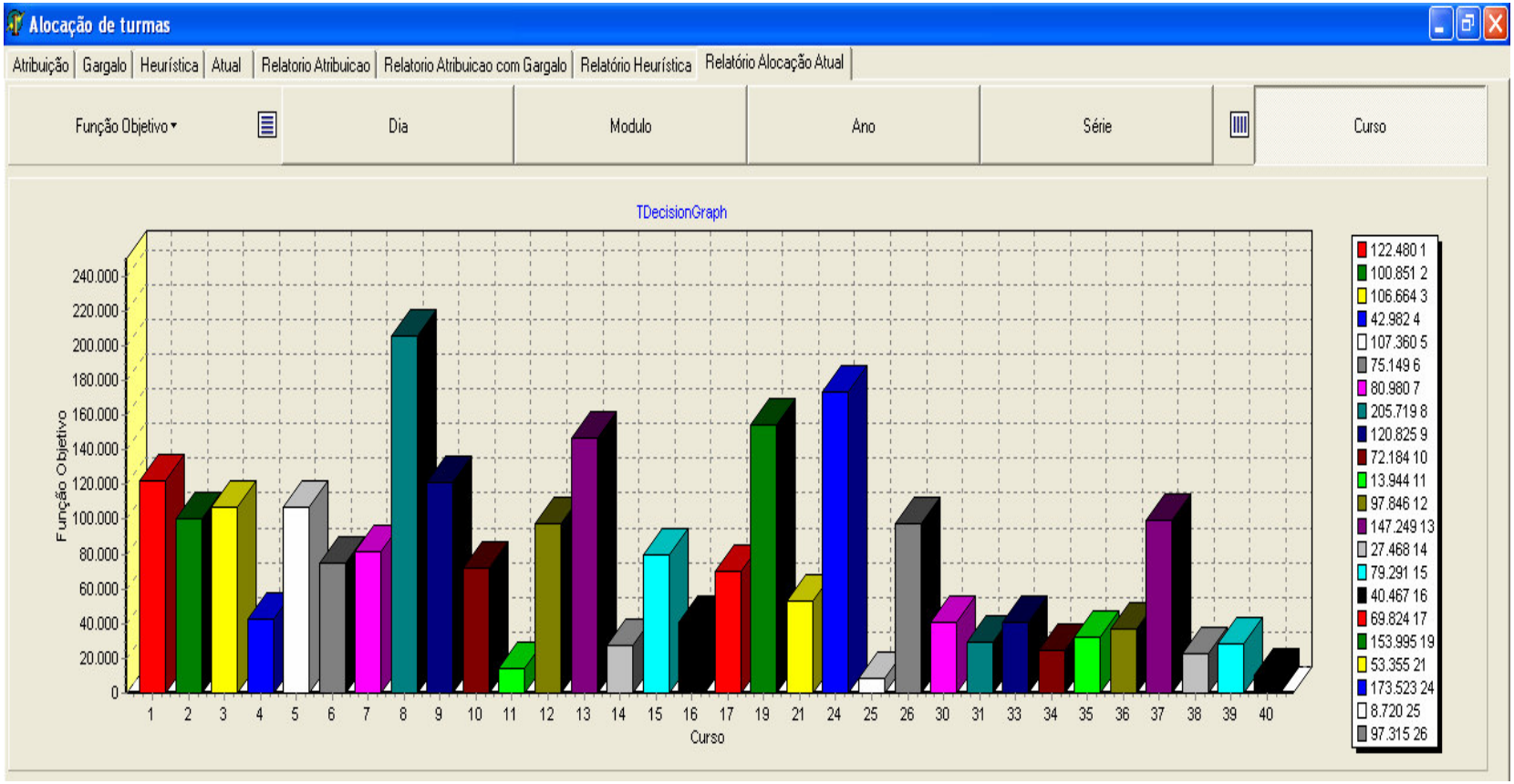


Figura 5.6. Gráfico de alocação manual, ano 2008

5.4. Considerações Finais

Como foram observados, os algoritmos PAS-D e PAS-VNS obtiveram os melhores resultados. A comparação desses resultados com a solução obtida manualmente pela instituição demonstra uma melhoria significativa na qualidade das soluções, principalmente pela redução de alocações desfavoráveis.

Capítulo

6

Conclusão

Este trabalho apresentou três novos algoritmos heurísticos e realizou um estudo comparativo entre os mesmos. Os três algoritmos resolveram instâncias do PAS de grande dimensão em um tempo computacional aceitável e com uma solução de boa qualidade, visto que seria difícil conseguir soluções melhores usando algoritmos exatos em tempo computacional menor. Observe que o algoritmo PAS-DG teve o maior valor no custo total, visto que esse problema é do tipo min-max, ou seja, minimiza o maior valor das alocações e não a soma dos custos de todas as alocações.

O algoritmo PAS-D gerou uma solução com melhor qualidade em relação aos outros dois algoritmos propostos. Seu desempenho computacional foi satisfatório, comprovando sua viabilidade e efetividade. O tempo computacional de aproximadamente de 30 a 40 minutos pode ser aceitável, uma vez que a resolução manual do problema pode consumir dias ou semanas de trabalho.

Os dois primeiros algoritmos são bastante flexíveis em relação às mudanças de requisitos e prioridades, pois basta aplicar tais critérios na formação da matriz de custo para cada problema de designação.

6.1. Sugestões para Trabalhos Futuros

Para trabalhos futuros, propõe-se a modificação do algoritmo PAS-VNS substituindo-se o método de busca local pelo algoritmo VND.

Propõe-se, também investigar outras meta-heurísticas, como Busca Tabu, Algoritmo Genético e *Simulated Annealing*, bem como desenvolver um modelo de programação matemática para o problema.

Referências Bibliográficas

AERTS, J.; KORST, J.; SPIEKSMA, F. Approximation of a retrieval problem for parallel disks. *Lecture Notes in Computer Science*, v. 2653, pp. 178-188, 2003.

ANDRADE, E.L. *Introdução à Pesquisa Operacional: Métodos e modelos para análise de decisões*. Rio de Janeiro: Livros Técnicos e Científicos, 2007.

BAKER, B.M.; SHEASBY, J. Accelerating the convergency of subgradient optimisation. *European Journal of Operational Research*, v. 117, pp. 136-144, 1999.

BARDADYN, V.A. Computer-Aided School and University Timetabling: The New Wave. *Lecture Notes in Computer Science*, v. 1153, pp. 22-45, 1996.

BEYROUTHY, C. et al. *Towards Improving the Utilisation of University Teaching Space.* PATAT, 2006, p. 103-122.

BOAVENTURA-NETTO, P.O. Combinatorial instruments in the design of a heuristic for the quadratic assignment problem. *Pesq. Oper.* v. 23, n. 3, pp. 383-402, 2003.

BOKHARI, S.H. *Assignment Problems in Parallel and Distributed Computing*. Boston, USA: Kluwer Academic Publisher, 1987.

BURKE, E. K.; VARLEY, D.B. Space allocation: an analysis of higher education requirements. *Lecture Notes in Computer Science*, v. 1408, pp. 20-36, 1997.

CAMPELO, R.E.; MACULAN, N. *Algoritmos e Heurísticas: desenvolvimento e avaliação de performance* . Rio de Janeiro: Editora da Universidade Federal de Fluminense, 1994.

CARPANETO, G.; TOTH, P. Primal-dual algorithms for the assignment problem. *Discrete Applied Mathematics*, n. 18, pp. 137-153, 1987.

CARRARESI, P.; GALLO G. A multi-level bottleneck assignment approach to the bus drivers' rostering problem. *European Journal of Operations Research*, v. 16, pp. 163-173, 1984.

CARTER, M.W.; TOVEY, C. A. When is the Classroom Assignment Problem Hard? *Operations Research*, v. 40, pp. 28-41, 1992.

CASTRO, O.M. *Resolução do Problema de alocação de sala de aula via Simulation Annealing*. Monografia (Graduação em Ciência da Computação) - Universidade Federal de Ouro Preto, Ouro Preto, 2003.

CONSTANTINO, A.A. *Otimização de Escala de Trabalho para Condutores de Trem: Sequenciamento de Tarefas e Alocação Baseada em Preferência Declarada*. Tese (Doutorado em Engenharia da Produção) - Universidade Federal de Santa Catarina, Florianópolis, 1997.

CORDENONSI, A.Z. *Ambientes, Objetos e Dialogicidade: uma estratégia de ensino superior em heurísticas e metaheurísticas*. Tese (Doutorado em Informática na Educação) - Universidade Federal do Rio Grande do Sul, Porto Alegre, 2008.

CORMEN, T.H. et al. *Algoritmos: Teoria e Prática*. Rio de Janeiro: Campus, 2002.

DIAZ, A.F. et al. *Optimización Heurística y Redes Neuronales*. Madrid: Paraninfo, 1996.

DORIGO, M. *Optimization, Learning and Natural Algoritms*. Phd thesis. Dipartameto di Elettronica, Politecnico di Milano, IT, 1991.

EVEN, S.; ITAI, A.; SHAMIR, A. On the complexity of timetabling and multicommodity flow problems. *SIAM Journal of Computation*, v. 5, pp. 691-703, 1976.

FEO, T.; RESENDE, M. A Greed Randomized Adaptive Search Procedure for Maximum Independente Set. *Operations Research,* v. 42, pp. 860-879, 1994.

GAREY, M.R.; JOHNSON, D.S. *Computers and Intractability: A Guide to the Theory of NP-Completeness*. San Francisco: Freeman. 1979.

GLOVER, F.; LAGUNA, M. Tabu Search. In: REEVES, Colin (Ed.). *Modern Heuristic Techniques*. Oxford: Blackwell, 1993. pp. 70-150.

GOLDBARG, M.C.; LUNA, H.P.L. *Otimização combinatória e programação linear: modelos e algoritmos*. 2. ed. Rio de Janeiro: Elsevier, 2005.

GOODRICH, M.T.; TAMASSIA, R. *Projeto de Algoritmos: fundamentos, análise e exemplos da Internet*. Porto Alegre: Bookman, 2004.

GORDBERG, D.E. *Genetic Algorithms in Search, Optimization, and Machine Learning*. Addison-Wesley, 1989.

KIRKPATRICK, S.; GELLATI, C.D.;VECCHI, M. Optimization by Simulated Annealing. *Science*, v. 220, pp. 671-680, 1983.

LANDA, S.J.D. *Metaheuristics and multiobjective approaches for space allocation*. Thesis University of Nottingham, School of Computer Science and Information Technology, Nottingham UK, 2003.

LICO, G.C.P. *Modelo e Algoritmo para Alocação de Espaço Físico*. Monografia (Graduação em Ciência da Computação) - Universidade Estadual de Maringá, Maringá, 2006.

LOPES, M.C.; SCHOEFFEL, P. Um método de alocação para o problema de reservas de sala de aula. In: *Anais do Congresso Brasileiro de Geoquímica; II Congresso Brasileiro de Computação*, Blumenau, 2002.

LUAN, F.; YAO, X. *Solving real-world lecture room assignment problems by genetic algorithms, Complex Systems - From Local Interactions to Global Phenomena*. Amsterdam: IOS Press, 1996. pp.148-160.

MALAGUTTI, P.L. *A inteligência artificial no ensino médio –construção de computadores que se comportam como humanos*. Bienal da SBM UFMG, 2002.

MLADENOVIC, N. *A variable neighborhood algorithm – a new metaheuristic for optimization combinatorial*. Abstract of papers presented at Optimization Days, Montreal 12, 1995.

MLADENOVIC, N.; HANSEN, P. Variable neighborhood search. Editorial. *European Journal of Operational Research,* v. 191, pp. 593-595, 2008.

_____. Variable Neighborhood Search: Methods and Recent Applications. In: *Third Metaheuristics International Conference*, Angra dos Reis, Brasil, 1999.

MOREIRA, D.A. *Pesquisa Operacional: curso introdutório*. São Paulo: Thomson Learning, 2007.

MOSCATO, P. *On Evolution, Search, Optimization, Genetic Algorithms and Matial Arts: Towards Memetic Algorithms*, Technical Report, Caltech Concurrent Computation Program, C3P Report 826, 1989.

NOWAKOVSKI, J.; SCHWÄRZLER, W.; TRIESCH, E. Using the generalized assignment problem in scheduling the ROSAT space telescope. *European Journal of Operational Research*, v. 112, n. 3, pp. 531-541, 1999.

OSMAN, I. Heuristics for Combinatorial Optimization Problems: Developments and New Directions. In: *Proceedings of the First Seminar on Information Technology and Applications*, Marfield Conference Centre, 1991.

PAPADIMITRIOU, C.H.; STEIGLITZ, K. *Combinatorial Optimization Algotitms and Complexity.* Printice Hall, 1982.

PIGATTI, A.A. *Modelos e algoritmos para o problema de alocação generalizada e aplicações*. Dissertação (Mestrado em Ciência da Computação) - Pontifícia Universidade Católica do Rio de Janeiro, Rio de Janeiro, 2003.

POTVIN, J.Y. The Traveling Salesman Problem: A Neural Network Perspective. *ORSA Journal on Computing*, v. 5, pp. 328-348, 1993.

RICH, E.; KNIGHT, K. *Inteligência Artificial. São Paulo*: Makron Books, 1993.

RULAND, K.S. A model for aeromedical routing and scheduling. *International Transactions in Operational Research*, v. 6, n. 1, pp. 57-73, 1999.

SALOMÃO, S.N.A. *Métodos de Geração de Colunas para Problemas de Atribuição*. Tese (Doutorado em Computação Aplicada) - Instituto Nacional de Pesquisas Espaciais, São José dos Campos, 2005.

SÁVIO, A. N. S. *Estudo e Implementação, Mediante Recozimento Simulado do Problema de Alocação de Salas*. Monografia (Graduação em Ciência da Computação) - Universidade Federal de Lavras, Lavras, 2005.

SCHAEFER, A. A survey of automated timetabling. *Artificial Intelligence Review*. v. 13, pp. 87-127, 1999.

SHAMBLIN, J. E.; STEVENS, G. T. *Pesquisa Operacional: uma abordagem básica. São Paulo*: Atlas, 1979.

SOUZA, M.J.F. *Programação de Horários em Escolas: Uma Aproximação por Metaheurísticas.* Tese (Doutorado em Engenharia) - Universidade Federal do Rio de Janeiro, Rio de Janeiro, 2000.

SOUZA, M.J.F.; MARTINS, A.X.; ARAÚJO, C.R. Uma metaheurística híbrida baseada em Simulated Annealing e Busca Tabu para o Problema de Alocação de Salas. In: Proceedings XI *Congreso Latino-Ibero-Americano de Investigación de Operaciones y Sistemas*, Concepción, 2002a. 1 CD-ROM.

_____. Métodos de Pesquisa em Vizinhança Variável Aplicados ao Problema de Alocação de Salas. In: *Anais do XXII Encontro Nacional de Engenharia de Produção*, Curitiba, 2002b. 1 CD-ROM.

STEINER, M. T. A. *et al.* Designação ótima de turmas de alunos para salas de aulas nas instituições de ensino. In: *Resumos do X Congreso Latino-Ibero-Americano de Investigación de Operaciones y Sistemas*, Santos, 2000. v. 1, pp. 282.

THE GUIDE to operacional research.
Disponível em <http://www.theorsociety.com/Science_of_Better/htdocs/prospect/index.asp>. Acesso em: 17 mar. 2008.

TOFFOLO, T.Â.M.; SOUZA, M.J.F.; SILVA, G.P. Algoritmos Simulated Annealing e Iterated Local Search Aplicados à Resolução do Problema de Rodízio de Tripulações de Ônibus Urbano. In: *Anais do XIII Seminário de Iniciação Científica da Universidade Federal de Ouro Preto*, Ouro Preto: UFOP, 2005. v. 1. pp. 1-1.

VIANA, G.V.R. *Meta-heurísticas e programação paralela em otimização combinatória.* Fortaleza: EUFC, 1998.

VIEIRA, L.E. *Algoritmo Evolutivo para o Problema do Caixeiro Viajante com Demandas Heterogêneas.* Dissertação (Mestrado em Engenharia da Produção) - Universidade Federal de Santa Maria, Santa Maria, 2006.