



**CENTRO FEDERAL DE EDUCAÇÃO
TECNOLÓGICA DE MINAS GERAIS**

Diretoria de Pesquisa e Pós-Graduação

**Programa de Mestrado em Modelagem
Matemática e Computacional**

**CONTRIBUIÇÕES PARA A RESOLUÇÃO DO
PROBLEMA DE PROGRAMAÇÃO DE CURSOS
UNIVERSITÁRIOS BASEADA EM CURRÍCULOS VIA
META-HEURÍSTICAS**

Dissertação de Mestrado apresentada ao Programa de Pós-Graduação em Modelagem Matemática e Computacional, como parte dos requisitos exigidos para a obtenção do título de Mestre em Modelagem Matemática e Computacional.

Aluno: Saulo Henrique D'Carlos Barbosa

Orientador: Sérgio Ricardo de Souza

Co-Orientador: Marcone Jamilson Freitas Souza

Belo Horizonte - MG
Abril de 2012

Resumo

Este trabalho trata o Problema de Programação de Cursos Universitários baseada em Currículos. O problema consiste em alocar um período e sala para as aulas de todas as disciplinas, evitando conflitos entre disciplinas de um mesmo currículo. São consideradas as características e instâncias de um relevante evento da área, a Competição Internacional de Programação de Horários (International Timetabling Competition - ITC 2007), essa abordagem se fundamenta pela generalidade das instâncias e também pelos recursos disponíveis para análise comparativa. Como se trata de um problema da classe NP-difícil, a utilização de meta-heurísticas para sua resolução têm se justificado por apresentar boas soluções sem um elevado custo computacional. Dessa forma, propõe-se a resolução do problema via técnicas meta-heurísticas, utilizando-se de estratégias avançadas de resolução, como hibridização de meta-heurísticas e variações das técnicas ainda não testadas para o problema em tela. São feitas abordagens através de estruturas de vizinhanças avançadas, em que movimentos ditos “inteligentes” são aplicados a partir da análise da solução corrente e das restrições envolvidas. São apresentados cinco algoritmos com as principais implementações realizadas, os mesmos são comparados entre si juntamente com os resultados da literatura com objetivo de se obter a análise sobre o comportamento e a capacidade dos algoritmos desenvolvidos. Os experimentos computacionais comprovaram certa eficiência da abordagem realizada mediante os resultados da literatura, além de comprovar a influência positiva de determinadas implementações como da técnica de relaxação aplicada ao ILS, e também dos procedimentos desenvolvidos: Kempeswap e Intra-Inter-Salas. Por fim, são apresentadas as principais conclusões e os trabalhos futuros deste estudo, além de destacar quais são as principais contribuições e os aspectos promissores da pesquisa realizada.

PALAVRAS-CHAVE: Programação de cursos universitários baseada em currículos; ITC 2007; Meta-Heurísticas; Kempeswap; Relaxação; Intra-Inter-Salas.

Abstract

This study addresses the Curriculum Based Course Timetabling Problem. This problem consists in allocating a timeslot and room for the classes of all disciplines, avoiding conflicts between courses of the same curriculum. We considered the characteristics and instances of the (International Competition timetabling - ITC 2007), this approach is founded by the majority of instances and also the resources available for comparative analysis.

Since this is a problem in class NP-hard, the use of meta-heuristics for its resolution have been justified by present good solutions without a high computational cost. Thus, it is proposed to solve the problem via meta-heuristics techniques, using advanced strategies for resolution, such as hybridization of metaheuristics and variations of untested techniques to the problem at hand. Approaches are made by Advanced structures vicinity, wherein said “motions smart” are applied from the analysis of the current solution and constraints involved. The five algorithms which are presented with the main implementations performed, they are compared with the results of the literature in order to obtain the analysis of the behavior and the ability of the algorithms developed. The computational experiments have shown some effectiveness of the approach taken by the results of the literature, and demonstrate the positive influence of certain implementations such as the relaxation technique applied to the ILS, and also developed procedures: Kempeswap Intra-Inter-Room. Finally, are presented the main conclusions and future work of this study, and highlight what are the main contributions and promising aspects of the survey.

PALAVRAS-CHAVE: Curriculum Based Course Timetabling; Metaheuristics; ITC 2007; Kempeswap; Relaxation; Intra-Inter-Room.

Sumário

1	INTRODUÇÃO	1
1.1	Justificativas	2
1.2	Objetivos	3
1.3	Organização do Trabalho	4
2	CARACTERIZAÇÃO DO PROBLEMA	5
2.1	Classificação dos Problemas de Horários Educacionais	6
2.1.1	Programação de Horários em Escolas	7
2.1.2	Programação de Horários de Cursos	7
2.1.3	Programação de Exames	8
2.1.4	Outras Classificações	9
2.2	Viabilidade, Otimalidade e Complexidade	10
2.3	O Problema abordado: <i>Curriculum Based Course Timetabling</i>	11
2.3.1	Caracterização do Problema	12
2.4	Formulação Matemática para o Problema	15
3	RESOLUÇÃO DE PROBLEMAS DE OTIMIZAÇÃO VIA META- HEURÍSTICAS	19
3.1	Otimização Combinatória	20
3.2	Conceito de Vizinhaça e Movimento	21
3.3	Heurísticas	21
3.3.1	Heurísticas de Construção	22
3.3.2	Heurísticas de Refinamento	22
3.4	Meta-heurísticas	24
3.4.1	Grasp	25
3.4.2	Busca Tabu	26
3.4.3	ILS	28
4	TRABALHOS RELACIONADOS	30
4.1	Revisões do Estado da Arte	30
4.2	Trabalhos correlacionados em Quadros de Horários	31
4.3	Trabalhos em Problemas de Programação de Cursos Universitários baseada em Currículos	32
5	ALGORITMOS APLICADOS À RESOLUÇÃO DO PROBLEMA	35
5.1	Representação da Solução	35
5.2	Vizinhaça	36

5.3	Função de Avaliação	39
5.4	Busca Local	39
5.5	Solução Inicial: Algoritmo <i>ConstroiGRASP</i>	40
5.6	ILS	41
5.7	Busca Tabu	42
5.8	O Procedimento <i>Kempswap</i>	43
5.9	O Procedimentos Intra-Inter-Salas	44
5.9.1	Intrasalas	45
5.9.2	Intersalas	48
5.9.3	O Procedimento Desenvolvido	48
5.10	Algoritmos Propostos	49
5.10.1	ILS-RELAX	49
5.10.2	ILS-TABU	51
5.10.3	ILS-KEMPE	52
5.10.4	ILS-INTRA	53
5.10.5	ILS-INTRAK	53
6	EXPERIMENTOS COMPUTACIONAIS	55
6.1	Definições de Parâmetros e Experimentos	55
6.2	ILS-RELAX vs ILS-TABU	57
6.3	ILS-KEMPE vs ILS-INTRA	61
6.4	ILS-INTRAK	64
6.5	Comparações com Resultados da Literatura	68
7	CONCLUSÕES E TRABALHOS FUTUROS	71
A	INSTÂNCIA EXEMPLO	74
B	SOLUÇÃO EXEMPLO	78
C	PUBLICAÇÕES	82
	REFÊRENCIAS	84

Lista de Tabelas

2.1	Descrição das Formulações para o CBCT	15
5.1	Parte da solução $Q1$	45
5.2	Solução $Q1$	46
6.1	Características das instâncias.	56
6.2	Parâmetros das implementações.	57
6.3	Comparações entre ILS*, ILS-RELAX e ILS-TABU.	58
6.4	Comparações entre ILS-RELAX, ILS-KEMPE e ILS-INTRA.	61
6.5	Comparações entre ILS-INTRA e ILS-INTRAK.	64
6.6	Resultados obtidos através do ILS-INTRAK e comparações com os resultados da literatura.	68
6.7	Resultados obtidos através do ILS-INTRAK, desprezando o tempo limite da competição, e, comparações com os melhores resultados encontrados para o problema.	70

Lista de Figuras

3.1	Representação de Ótimos Locais e um Ótimo Global	21
3.2	Princípios básicos que envolvem o funcionamento do ILS	29
5.1	Representação da solução adotada.	35
5.2	Representação MovimentoPeríodo.	36
5.3	Representação MovimentoSala.	36
5.4	Representação MovimentoAula.	37
5.5	Representação MovimentoEstabilidadeSala.	37
5.6	Representação MovimentoTrocaDeAulasESalas.	38
5.7	Representação MovimentoTrocaDeAulasESalas.	38
5.8	Movimento Kempeswap.	44
5.9	G_{Ra} , grafo da sala Ra	46
5.10	G'_{Ra} , grafo da sala Ra após 1ª execução do intrasalas	47
5.11	G''_{Ra} , aperfeiçoamento desenvolvido no Intrasalas.	47
5.12	Intersalas	48
5.13	Fluxograma da execução do ILS-RELAX	50
5.14	Comparações entre fluxogramas do ILS-RELAX e ILS-KEMPE	53
6.1	Média, Erro Mínimo e Erro Máximo: ILS*, ILS-RELAX e ILS-TABU. .	60
6.2	Média, Erro Mínimo e Erro Máximo: ILS-RELAX, ILS-KEMPE e ILS-INTRA.	63
6.3	Média, Erro Mínimo e Erro Máximo: ILS-RELAX, ILS-RELAX- KEMPESWAP e ILS-RELAX-INTRAINTERSALAS.	65
6.4	Gráfico de Probabilidade Empírica, instância <i>comp02</i>	66
6.5	Gráfico de Probabilidade Empírica, instância <i>comp11</i>	67
6.6	Gráfico de Probabilidade Empírica, instância <i>comp12</i>	67

Capítulo 1

INTRODUÇÃO

De acordo com Burke e Petrovic (2002), a classe de problema *Timetabling* (nome dado ao Problema de Programação de Horários - PPH) tem atraído a comunidade científica nos últimos quarenta anos, devido a sua importância e complexidade de resolução (NP-Difícil). Segundo Willemen (2002), o problema abrange todas as atividades que envolvem o processo de construção de quadros de horários, sendo o PPH de grande aplicação nas mais variadas áreas, tais como nas indústrias, hospitais, empresas de transporte, eventos esportivos, até escolas e universidades. Em essência, o problema consiste na atribuição de um número de eventos, cada um com uma série de características, a um número limitado de recursos sujeitos a determinadas restrições (Qu, 2002).

O PPH apresenta como uma de suas principais variações, o Problema de Programação de Horários Educacionais (PPHE). Um importante marco para a classe de problemas do tipo PPHE foi a realização da competição internacional de programação de horários (*International Timetabling Competition* - ITC 2002 e ITC 2007). Nessa competição, o objetivo foi o de apresentar diferentes técnicas para resolução de problemas de PPHE e compará-las. A importância do evento deve-se ao fato de não ser comum nesta classe de problema autores realizarem testes computacionais e comparações entre diferentes técnicas, devido a diversidade de características envolvidas em cada problema. Em sua última edição, o ITC 2007 apresentou três categorias: Programação de Exames, Programação de Cursos Pós-Matrículas e Programação de Cursos baseada em Currículos (*Curriculum based Course Timetabling* - CBCT).

O CBCT é o objeto de estudo neste trabalho, trata-se de um problema geralmente encontrado em universidades que apresentam, como característica, a possibilidade de o aluno escolher quais disciplinas serão cursadas a cada semestre, sendo as possíveis disciplinas limitadas pelo currículo ao qual o aluno pertence. Na variação estudada neste projeto, busca-se alocar um período e uma sala para cada aula de todas as disciplinas, levando-se em conta os conflitos entre disciplinas de um mesmo currículo. O presente estudo visa a utilização não só das instâncias fornecidas pelo sítio do evento, bem como o respeito às regras da competição para a realização dos experimentos e comparações.

Na literatura é possível encontrar diferentes abordagens para resolução de problemas de programação de horários, tais como métodos sequenciais, métodos baseados em restrições e meta-heurísticas. A utilização de meta-heurísticas tem se desta-

cado por apresentar consideráveis resultados, tendo em vista o custo computacional envolvido e a qualidade das soluções alcançadas. Vários trabalhos relevantes que relatam a resolução do PPH via diferentes abordagens meta-heurísticas podem ser encontrados na literatura, dentre eles: (Lü e Hao, 2010) que utiliza Busca Tabu com aperfeiçoamentos como a utilização de perturbações e movimentos avançados de busca; (Chiarandini et al., 2006) apresenta um algoritmo que combina heurísticas de construção com as meta-heurísticas VND e Busca Tabu; (Muller, 2008) que utiliza *Simulated Annealing* e *Great Deluge* aliado a outros métodos de busca local e análise estatística; (Bai et al., 2008) utiliza um *Simulated Annealing* com aperfeiçoamentos e (Irene et al., 2009) que apresenta uma abordagem ao problema através da meta-heurística *Particle Swarm Optimization* (PSO). Uma completa revisão do estado da arte para o PPHE pode ser vista em trabalhos como (Schaerf, 1999), (Burke e Petrovic, 2002) e (Lewis, 2008), nestes é possível notar a importância dos métodos meta-heurísticos na resolução de problemas da classe.

Assim sendo, propõe-se com esse trabalho o estudo sobre a resolução do CBCT via técnicas meta-heurísticas, visando a implementação e o estudo de diferentes abordagens que apresentem certa eficiência e aspectos promissores, agregando à literatura relacionada novas contribuições. Por conseguinte, são apresentados cinco algoritmos que representam a evolução das implementações realizadas em prol da resolução do problema via meta-heurísticas.

Dentre as principais abordagens implementadas e estudadas no trabalho, destaca-se: i) a utilização de um critério de aceitação denominado relaxação, em que soluções de piora são aceitas no ILS; ii) a implementação do Procedimento Kempswap, que realiza movimentos entre cadeias kempe, realizando movimentos bruscos em que a qualidade da solução é preservada; iii) a implementação do Procedimento Intra-Inter-Salas, em que através da representação de um grafo obtido sobre uma solução, é possível obter ciclos mínimos de custo negativo, que representarão na maioria das vezes melhora na solução.

Ao final, o trabalho apresenta a análise e comparação entre as implementações realizadas e também com os principais resultados encontrados na literatura validando as abordagens realizadas.

1.1 Justificativas

O problema de programação de horários educacionais (PPHE), é a denominação dada a uma classe de problemas combinatórios, que envolvem a construção de quadro de horários em ambientes educacionais e apresentam grande relevância na área de Pesquisa Operacional. O problema em questão trata de um tema presente no contexto das instituições educacionais, uma vez que a programação das aulas e exames é realizada no mínimo uma vez ao ano e apresenta grande influência sobre os vários setores (docentes, discentes e administrativo).

Segundo Santos e Souza (2007), o estudo sobre a automatização do PPHE pode ser justificado pelas seguintes razões:

- **Dificuldade de Resolução:** a construção de um quadro de horários que atenda a todos os interesses envolvidos é uma tarefa extremamente difícil. Pode-se afirmar ainda que, frequentemente, a simples construção de um quadro

de horários válido, ou seja, um quadro que atenda somente às restrições básicas, já é uma tarefa complicada;

- **Importância Prática:** como mencionado anteriormente, um quadro de horários que atenda a instituição de ensino nos aspectos propostos, pode melhorar a satisfação do corpo docente, melhorar o aproveitamento dos alunos e possibilitar uma gestão eficiente dos recursos da instituição. Sendo assim, sua resolução tem grande importância prática em âmbito educacional;
- **Importância Teórica:** problemas de PPHE são classificados como NP-difíceis ou NP-completos, ou seja, são problemas de resolução com elevada complexidade computacional. Logo, progressos alcançados na resolução de problemas desse tipo são um dos grandes objetivos das pesquisas coerentes em computação, pesquisa operacional e matemática.

Este trabalho é justificado, também, por tratar o problema de programação de cursos baseada em currículos com a utilização de instâncias da competição internacional de programação de horários ITC 2007. A principal vantagem quanto à utilização destas instâncias está no fato destas possuírem todo aparato necessário para análise comparativa entre diferentes técnicas. Como mencionado anteriormente, a realização de uma competição foi um marco importante para a área, visto que a comparação entre diversas técnicas para um mesmo problema sobre as mesmas condições não era uma prática comum dos pesquisadores. Assim, como novas técnicas foram implementadas e testadas têm-se, de certa forma com essa abordagem um maior poder comparativo e conclusivo sobre o que foi feito.

O trabalho visa também uma contribuição teórica à abordagem. Pretende-se com a realização dessa pesquisa a agregação de novos aspectos relacionados a resolução do problema, aumentando ainda mais o banco de dados (De Cesco et al., 2010) referente aos métodos de solução empregados nas instâncias do ITC 2007.

1.2 Objetivos

O presente trabalho tem como objetivo geral, o estudo da resolução, via meta-heurísticas, do problema de programação de cursos baseada em currículos. Pretende-se a realização de uma abordagem que reúna diferentes aspectos promissores (no sentido de meta-heurísticas, vizinhanças, Busca Local, etc.) em busca da implementação de uma técnica que apresente certa eficiência e robustez, contribuindo na resolução da classe de problemas em tela.

Já os objetivos específicos são apresentados abaixo:

- Pesquisar sobre diferentes técnicas meta-heurísticas aplicáveis ao problema tratado, principalmente no que se refere à hibridização dessas técnicas ou em novas atribuições realizadas que ainda não foram testadas para o problema envolvido;
- Avaliar e pesquisar diferentes estruturas de vizinhança e métodos não convencionais de exploração do espaço busca para resolução do problema. Destaca-se

que resultados relevantes estão sendo obtidos a partir da utilização de movimentos avançados, que são mais caros computacionalmente, porém, apresentam maior qualidade. Dessa forma, novas vizinhanças podem ser criadas ou adaptadas para o problema;

- Realizar análises e comparações que sejam conclusivas quanto ao comportamento da técnica utilizada na resolução do problema e também quanto a sua qualidade frente aos resultados existentes na literatura.

1.3 Organização do Trabalho

O restante deste trabalho está organizado como segue. No próximo capítulo, a caracterização do problema é discutida e detalhada. Alguns aspectos relevantes sobre resolução de problemas via meta-heurísticas são apresentados no capítulo 3. No capítulo 4, são descritos alguns trabalhos relacionados ao tema. As implementações empregadas para a resolução do problema são detalhadas no capítulo 5. No capítulo 6, os resultados obtidos são apresentados juntamente com as comparações com os resultados da literatura. Por fim, é apresentado através do capítulo 7, as conclusões gerais e os trabalhos futuros.

Capítulo 2

CARACTERIZAÇÃO DO PROBLEMA

Qu et al. (2009) afirma que problemas de programação de horários (*Timetabling problems*) podem ser encontrados em várias formas, dentre estas são citadas: o PPHE - *Educational Timetabling*, o escalonamento de horários de enfermeiras - *Nurse Scheduling* (Burke et al., 2004), programação de horários em eventos esportivos - *Sports Timetabling* (Hamiez e Hao, 2001) e programação de horários em problemas de transporte - *Transportation Timetabling* (Kwan, 2004).

A programação de horários é uma atividade administrativa de grande importância em vários segmentos. A construção de quadro de horários em ambientes educacionais é uma tarefa comum que geralmente ocorre periodicamente (semestral, anual, etc.), em que a qualidade do quadro de horários gerado tem grande impacto sobre diferentes partes interessadas, incluindo professores, alunos e administradores. Um quadro de horários é construído realizando a alocação de eventos (aulas, exames ou palestras), considerando os recursos disponíveis (docentes, salas e períodos disponíveis) e atendendo ao máximo as restrições envolvidas.

Resoluções manuais podem ser adotadas para a construção de quadros de horários educacionais, porém essa é uma abordagem que geralmente requer muitas pessoas e/ou muitos dias. Além do que, tais resoluções se baseiam num processo de tentativa e erro que, na maioria das vezes, produzem soluções que não satisfazem a instituição em diversos aspectos.

Burke e Silva (2005) relacionam a dificuldade de resolução do PPHE com a natureza combinatorial presente no mesmo, o que implica que o tamanho do espaço de busca aumente drasticamente, de acordo com o tamanho do problema, tornando-se praticamente impossível explorar explicitamente todas as soluções (com exceção de problemas pequenos). Ressalta-se, ainda, que características como a quantidade de restrições e o critério de avaliação de um quadro de horários podem complicar ainda mais a resolução do problema.

Por razões como essas apresentadas acima, uma atenção especial vem sendo dada à automação do PPHE. A automação do problema representa uma importante e desafiadora área da pesquisa operacional e inteligência artificial, que vem sendo estudada desde o início da década 60 (Gotlieb, 1963). Desde então, o problema é cada vez mais discutido, sendo abordado por diferentes técnicas e sobre diferentes características. Esse fato é evidenciado - entre outros aspectos - pelo surgimento de

uma série de conferências internacionais, como a Conferência Internacional de Automação de Problemas de Horário (*Practice and Theory on Automated Timetabling - PATAT*), juntamente com a Competição Internacional de *Timetabling* (*International Timetabling Competition - ITC*).

Destaca-se que a resolução do PPHE tem características básicas que estão presentes nos diversos modelos de ensino, porém, cada instituição apresenta suas peculiaridades e aspectos próprios que, de certa forma, determinam a grade de horários construída. Devido às características básicas encontradas em cada abordagem, diferentes subclasses do PPHE foram propostas na literatura, a fim de delimitar os problemas encontrados e discutí-los da melhor maneira possível.

Tendo em vista as subclasses existentes, são encontradas várias formulações matemáticas para os problemas, como pode ser verificado em de Werra (1985) e Souza (2000). Contudo, mesmo em problemas que estão em uma mesma subclasse, ainda assim é possível encontrar aspectos próprios de cada instituição, de modo que, não existe nenhuma formulação que possa ser aplicada a todos os casos.

O PPHE, em uma visão geral, pode ser modelado como um problema matemático de Otimização Combinatória de complexidade NP-Completo ou NP-Difícil (Cooper e Kingston, 1996). Várias técnicas podem ser empregadas para a resolução deste problema, como métodos sequenciais, métodos baseados em restrições lógicas, métodos baseados em coloração de grafos, métodos exatos, heurísticas, redes neurais, meta-heurísticas entre outras abordagens. A utilização de técnicas meta-heurísticas para a resolução do problema tem se destacado, afirmação que é comprovada pelo alto número de publicações nas últimas décadas que envolvem a utilização dessas, com certa eficiência para resolução de problemas de programação de horários.

Serão apresentados, nas seções a seguir, alguns dos principais aspectos do problema para este trabalho, quais sejam: as classificações encontradas na literatura para o PPHE, que delimitam classes do problema encontradas em âmbito educacional; uma seção referente a viabilidade, otimalidade e complexidade para o PPHE, em que são apresentadas informações relevantes quanto aos tipos de restrições envolvidas e também sobre o nível de complexidade para resolução do problema; e, por último, uma seção referente à competição internacional de programação de horários, que tem grande importância na área e também apresenta uma variação de problema que é o objeto de estudo deste.

2.1 Classificação dos Problemas de Horários Educacionais

O PPHE é um problema presente na maioria das instituições de ensino. Sendo assim, apresenta muitas variações que refletem as peculiaridades de cada abordagem. Um grande número de variantes deste problema são cada vez mais discutidas na literatura, sendo as principais diferenças entre as variantes, o tipo de instituição envolvida (universidade ou escola) e as restrições que o envolvem.

Schaerf (1999) apresenta uma classificação do PPHE em que três classes definem o problema: Programação de Horários em Escolas (*School Timetabling*), Programação de Horários de Cursos (*Course Timetabling*) e Programação de Exames (*Exam Timetabling*). De acordo com Souza (2000), esta classificação não é absoluta, pois

existem variações do problema que não se enquadram de maneira exata nestas categorias. Contudo, esta classificação é a adotada pela maioria dos pesquisadores.

As categorias definidas por Schaerf (1999), juntamente com outras classificações encontradas na literatura, são apresentadas nas subseções seguintes.

2.1.1 Programação de Horários em Escolas

O Problema de Programação de Horários em Escolas é também conhecido como problema Classe-Professor (*Class-Teacher Problem*). Trata-se do agendamento semanal de encontros classe-professor para uma instituição com características semelhantes a uma escola. Características como a programação semanal, que se repete semestral ou anualmente, aulas ministradas em um mesmo turno e o fato de cada classe apresentar um único conjunto de disciplinas para todos alunos da turma são as típicas dessa classe de problemas.

De acordo com Souza (2000), são dados referentes ao problema: um conjunto de turmas, um conjunto de professores, um conjunto de disciplinas e um conjunto de períodos disponíveis à alocação. Para cada disciplina, tem-se a carga horária (número exigido de períodos semanais), professor que leciona e turma relacionada. Para turmas, tem-se o conjunto de disciplinas da grade curricular. Para professores, tem-se conjunto de disciplinas lecionadas e, em alguns casos, a disponibilidade do professor.

A resolução do problema é dada pela alocação de um período para cada aula de todas as disciplinas, respeitando-se a carga horária de cada disciplina e evitando-se que um professor esteja alocado em mais de uma turma em um mesmo período e, também, que turmas tenham aula com mais de um professor em um mesmo horário.

Vale ressaltar sobre o conceito de turma (ou classe). Neste problema turma é definido como um grupo de alunos que apresenta a mesma grade curricular, cujas as aulas são ministradas em uma sala fixa. O escalonamento de salas não faz parte do problema, pois cada turma já possui sua sala pré-estabelecida, o que significa que o deslocamento é realizado pelo professor (exceto por disciplinas que requerem salas específicas).

Schaerf (1999) relata algumas variantes propostas sobre o problema básico para atender as demandas reais. Dentre essas, a ocorrência de aulas que envolvam duas turmas no mesmo período, a ocorrência de disciplinas que são ministradas por mais de um professor e problemas que envolvam disciplinas que requerem salas especiais. Ressalta-se também que o *School Timetabling* é uma classificação que pode abranger a construção de quadro de horários em universidades, principalmente em universidades que utilizam o conceito de turmas, no sentido apresentado nesta subseção. Um exemplo desta abordagem pode ser visto em Barbosa et al. (2010).

2.1.2 Programação de Horários de Cursos

Trata-se de um problema geralmente encontrado em universidades que apresentam como característica a possibilidade do aluno escolher as disciplinas a serem cursadas a cada semestre, ou seja, um currículo flexível. Este problema apresenta: um conjunto de disciplinas (ex: Álgebra Linear, Engenharia de Software, Compiladores, etc.); um conjunto de currículos (Ciência da Computação, Matemática, Engenharia

Elétrica) em que cada currículo engloba um subgrupo de disciplinas; um conjunto de salas e um conjunto de períodos disponíveis para alocação.

Neste problema, cada aluno pode se matricular apenas nas disciplinas pertencentes ao seu currículo, ou seja, ao contrário do *School Timetabling* não se tem o conceito de turmas e, sim, o de currículos, que determinam as possíveis disciplinas de um aluno. Ressalta-se que uma mesma disciplina pode estar em mais de um currículo, e algumas disciplinas são comuns em vários currículos.

A alocação de salas também faz parte desse problema, pois por não existir turmas, também não se tem o conceito de sala fixa. Sendo assim, o deslocamento é realizado pelos alunos, ao invés do professor. Na alocação de salas, a disponibilidade e a capacidade de cada sala são informações relevantes no processo de construção do quadro de horários. O escalonamento de salas e o conceito de turma são as principais diferenças entre esse problema e a programação de horários em escolas.

Segundo Souza (2000), a solução desse problema consiste em alocar as aulas necessárias para cada disciplina nos horários disponibilizados, respeitando-se a capacidade e disponibilidade das salas envolvidas, de forma que nenhum aluno tenha aulas simultâneas.

Se, por um lado, o problema apresenta maior complexidade, por ser responsável pela alocação de salas, por outro, há uma maior disponibilidade de horários para realização das aulas. Em problemas desta classificação, geralmente os horários podem ser alocados em diferentes turnos (manhã, tarde e noite).

Diversas variações do problema são encontradas na literatura, dentre estas tem-se: problemas que consideram o deslocamento entre campus, problemas que consideram a matrícula dos alunos para evitar sobreposições, problemas com períodos de alocação de diferentes tamanhos, entre outros.

2.1.3 Programação de Exames

Segundo Burke et al. (1997), o Problema de Alocação de Horários em Universidades pode ser dividido em dois problemas: Programação de Exames e Programação de Horários de Cursos. O Problema de Programação de Exames consiste em alocar exames para as disciplinas dos cursos de uma universidade. O problema apresenta: um conjunto de exames (um para cada disciplina), um conjunto de salas e o conjunto de períodos disponíveis para alocação.

O objetivo básico do problema é que a alocação seja realizada de forma que nenhum aluno possua exames simultâneos, ou seja, evitar que os alunos tenham mais de um exame em um mesmo horário. Neste sentido, nota-se que o problema é muito semelhante ao problema de programação de horários de cursos. A principal diferença destes problemas é dentre às restrições envolvidas que, segundo Souza (2000), se distinguem sobretudo pela natureza das mesmas.

Entre as distinções para diferenciação do problema, tem-se as principais restrições, que são destacadas abaixo:

- Podem ocorrer múltiplos exames em uma mesma sala;
- Exames de certas disciplinas não podem preceder o exame de outras;
- Alguns exames devem ocorrer de forma consecutiva;

- Respeitar um limite do número de exames por dia para todos alunos;

Vários tipos de restrições e variações podem ser encontradas no problema. O trabalho de Qu et al. (2009) faz uma revisão sobre o problema e apresenta uma ampla perspectiva sobre o tema. Ressalta-se que este problema não é comumente encontrado nas universidades brasileiras.

2.1.4 Outras Classificações

Como foi mencionado no início dessa seção, a classificação feita por Schaerf (1999) não é absoluta, sendo encontradas na literatura diferentes classificações para este problema, das quais, destaca-se a classificação apresentada por Carter e Laporte (1997), a qual delimita o problema em cinco categorias.

Das categorias apresentadas por Carter e Laporte (1997), três são equivalentes as apresentadas em Schaerf (1999) e as demais são apresentadas abaixo:

- ***Teacher Assignment Problem***: o problema consiste em alocar um professor para cada disciplina a partir de um quadro de horários pré-estabelecido. Informações como disponibilidade e possíveis disciplinas lecionadas por cada professor juntamente, com o quadro de horários estabelecido, são informações prévias do problema.
- ***Classroom Assignment Problem***: o problema consiste em alocar um conjunto de salas para realização das aulas de um quadro de horários pré-estabelecido.

Outra classificação, contendo oito categorias para o PPHE, é apresentada por Reis e Oliveira (2000). Neste trabalho, às seguintes categorias são apresentadas:

- ***Class-Teacher Timetabling***: corresponde à categoria *School Timetabling Problem* definida por Schaerf (1999);
- ***Course Timetabling***: corresponde à categoria *Course Timetabling Problem* definida por Schaerf (1999);
- ***Examination Timetabling***: corresponde à categoria *Exam Timetabling Problem* definida por Schaerf (1999);
- ***Room Assignment***: corresponde à categoria *Classroom Assignment Problem* definida por Carter e Laporte (1997);
- ***Staff Allocation***: corresponde à categoria *Teacher Assignment Problem* definida por Carter e Laporte (1997);
- ***Section Definition***: o problema consiste em definir o número de seções em que um mesmo evento será ofertado, ou seja, o número de vezes que um mesmo evento ocorre. Por exemplo, uma mesma aula de determinada disciplina pode ser dada em três seções, devido ao grande número de cursos em que esta disciplina está presente ou, então, um mesmo exame deve ser dado em duas salas diferentes devido a limitação de espaço;

- ***Student Scheduling***: este problema ocorre quando há eventos com múltiplas seções na grade de horários. Consiste em determinar o número de estudantes para cada evento, e também escalonar, de forma balanceada, cada estudante para uma seção, evitando conflitos.
- ***Invigilator Assignment***: este problema está associado à Programação de Exames, consiste em escalonar fiscais para a realização dos exames, sendo a alocação dos fiscais relacionada ao número de estudantes e as salas envolvidas.

Várias subcategorias ligadas às classificações apresentadas neste trabalho são encontradas devido às peculiaridades institucionais presentes até mesmo em problemas de uma mesma classe. Há de se destacar que o problema abordado neste trabalho é uma subcategoria da classe de *Course Timetabling Problem* (Programação de Cursos), o qual foi melhor descrito na seção 2.1.2.

2.2 Viabilidade, Otimalidade e Complexidade

No que diz respeito à abordagem dada ao PPHE, têm-se dois fatores fundamentais, que são viabilidade e otimalidade, que interferem diretamente na complexidade do problema.

Em alguns casos do PPHE, a resolução consiste em encontrar um quadro de horários que satisfaça todas as restrições envolvidas. Nestes casos, o problema pode ser referenciado como um problema de busca ou viabilidade, no qual o objetivo se limita a obter uma solução (quadro de horários) viável. Quanto à complexidade, este problema é classificado como NP-Completo (Cooper e Kingston, 1996).

O PPHE também pode ser formulado como um problema de otimização. De acordo com Souza (2000), nos problemas deste tipo deseja-se encontrar, dentre todos os quadros de horários que satisfaz a um certo conjunto de restrições ditas fortes (essenciais), aquele, chamado *ótimo*, que minimize uma certa função objetivo, na qual se encontram as restrições fracas (não-essenciais). Nesse caso, o problema é NP-Difícil (Eikelder e Willemen, 2001).

Na literatura, é mais comumente encontrada a abordagem do PPHE como um problema de otimização. Contudo, o problema de viabilidade é considerado aplicando-se a ele técnicas de otimização. Pode-se dizer que a tentativa de se encontrar uma solução viável é tratada como um problema de minimização da distância de viabilidade, sendo esta distância inserida na função de avaliação do problema. Dessa forma, todas as restrições podem ser combinadas em uma única função objetivo, que consiste em uma combinação linear com pesos (valores numéricos).

Os pesos associados a cada restrição são responsáveis por distinguir uma restrição Forte e uma restrição Fraca no problema. Restrições ditas Fortes recebem um peso associado com um alto valor enquanto as restrições Fracas têm pesos de baixo valor associados. A definição dos pesos de cada restrição é uma tarefa árdua, na qual é fundamental que o decisor conheça claramente a relação de importância entre as restrições do quadro de horários a ser construído.

Segundo Santos e Souza (2007), as restrições do problema são os requerimentos desejáveis à grade de horários a ser construída. Dessa forma, é apresentado o conceito sobre restrições Fortes e Fracas:

- **Restrições Fortes:** Estão diretamente ligadas à viabilidade da grade de horários gerada e, portanto, não devem ser negligenciadas, pois não é possível a implementação de quadro de horários que as desrespeitem. Restrições comuns deste tipo são as de não ocorrência de conflitos: em uma mesma sala, em um mesmo horário ou com disciplinas de um mesmo currículo. Restrições Fortes são responsáveis por determinar o espaço de busca de um problema, de forma que apenas soluções que respeitem todas estas restrições são chamadas de soluções **factíveis**;
- **Restrições Fracas:** São responsáveis pela qualidade da grade. Assim, podem ser violadas quando houver extrema necessidade. Ou seja, são aspectos desejáveis que, quando violados, não impedem a implementação do quadro de horários construído. São Restrições Fracas comuns no PPHE : respeitar os horários preferenciais do professor, alocar aulas de forma geminada, evitar grandes deslocamentos entre aulas alocadas, entre outras restrições. Ressalta-se que as restrições Fracas podem apresentar diferentes níveis de importância em um mesmo problema, sendo este refletido no peso atribuído a cada restrição na função de avaliação.

As restrições que envolvem o problema e também a maneira que são classificadas (restrições Fortes ou Fracas) são características particulares de cada abordagem, que estão ligadas diretamente com os aspectos desejáveis e característicos da instituição educacional envolvida. Santos e Souza (2007) definem que em geral, as restrições podem ser divididas em três classes: organizacionais, que desrespeitam a instituição e geralmente estão ligadas à legislação da instituição ou gestão de recursos; pedagógicas, que são considerações importantes para o melhor aproveitamento das aulas; pessoais, que dizem respeito a aspectos pessoais do corpo docente ou discente da instituição.

2.3 O Problema abordado: *Curriculum Based Course Timetabling*

O Problema de Programação de Horários em Universidades é reconhecido como um problema combinatorial de difícil resolução e de grande relevância prática, seja com relação a exames (*Examination Timetabling*) ou a cursos (*Course Timetabling*).

Recentemente, uma competição internacional foi organizada, tendo como foco a resolução de problemas de programação de horários em universidades. Em 2002, a *International Metaheuristic Network* organizou a primeira Competição Internacional de Programação de Horários (*International Timetabling Competition 2002 - ITC 2002*).

Nesta competição, os proponentes devem desenvolver programas para resolver o problema de programação de cursos, sendo a resolução vencedora aquela que apresentar o melhor desempenho médio em um determinado conjunto de instâncias com diferentes atributos. A competição apresenta, como uma de suas características para os problemas abordados, um modelo que incorpore os principais aspectos que são comuns à grande maioria dos problemas da classe envolvida. Dessa forma, os

modelos adotados na competição têm se tornado padrões que estão sendo seguidos por um grande número de pesquisadores. Em alguns casos apenas são acrescentadas novas características específicas de cada instituição, sendo o modelo da competição utilizado em partes.

Em 2007, foi lançada a segunda edição da competição, o ITC 2007. Seguindo as principais características da primeira edição, busca-se avançar a discussão do problema em novos aspectos. Nessa edição, foram consideradas três categorias: Programação de Exames (*Examination Timetabling*), Programação de Cursos Pós-Matrículas (*Post Enrolment based Course Timetabling*), Programação de Cursos baseada em Currículos (*Curriculum based Course Timetabling*). As duas últimas categorias são variações presentes na classe de problemas denominada *Course Timetabling*. O problema alvo deste trabalho trata-se da terceira categoria da competição, a qual será apresentada na subseção a seguir.

A realização de um evento como o ITC é um importante marco para a pesquisa na área, visto que, tradicionalmente, poucos autores realizavam comparações entre diferentes técnicas para um mesmo problema. Até então, o mais comum era a realização de comparações entre soluções geradas manualmente para o problema, sendo este um critério bastante impreciso (Santos e Souza, 2007), devido à diversidade de resoluções encontradas em diferentes instituições de ensino.

2.3.1 Caracterização do Problema

O Problema de Programação de Cursos Universitários baseada em Currículos (*Curriculum based Course Timetabling* - CBCT) é uma variação do PPHE, descrita na terceira modalidade de problemas da competição *International Timetabling Competition 2007* (<http://www.cs.qub.ac.uk/itc2007/>).

De acordo com Di Gaspero e Schaerf (2007), o problema consiste em realizar a programação semanal de disciplinas de vários cursos universitários dentro de um determinado número de períodos e salas disponíveis. Os conflitos entre as disciplinas são determinados pelos currículos em que as mesmas estão envolvidas, além de outras restrições, que determinam a viabilidade e a qualidade de um quadro de horários construído.

De acordo com De Cesco et al. (2010), as principais entidades envolvidas no problema são:

- Dias, Horários e Períodos: dado um número de dias letivos, cada dia é dividido em um dado número de períodos, sendo um horário correspondente ao par dia/período;
- Currículos: um currículo é composto por um grupo de disciplinas que possam ter estudantes em comum e, por isso, as disciplinas de um mesmo currículo não devem ser agendadas em um mesmo período;
- Disciplinas e Professores: cada disciplina apresenta um professor alocado para lecionar (previamente alocado pela instituição); o número de aulas; o número de alunos; o mínimo de dias exigidos na alocação; e, também, um conjunto de horários em que a mesma não está disponível para alocação;

- **Salas:** cada sala apresenta sua respectiva capacidade e, em alguns problemas, sua localização.

Uma solução para o problema é dada pela atribuição de um período e uma sala para todas as aulas de todas as disciplinas.

Conforme descrito na seção 2.1.2, a avaliação de uma solução é feita tendo em vista: (i) as restrições fortes, que são essenciais, pois estão ligadas à viabilidade; e (ii) as restrições fracas, que são almejadas para melhorar a qualidade de uma solução, mas podem ser desrespeitadas, caso não exista outra alternativa. As restrições consideradas pelo problema são descritas em Di Gaspero e Schaerf (2007) e são apresentadas abaixo.

As restrições fortes para o problema são:

- **Restrição Aulas:** determina que todas as aulas de uma disciplina devem ser agendadas e alocadas em períodos distintos. Uma violação ocorre se alguma aula da disciplina não é agendada ou se duas aulas distintas são agendadas no mesmo período;
- **Restrição Conflitos:** as aulas de disciplinas de um mesmo currículo ou aulas de duas ou mais disciplinas que possuem o mesmo professor não devem ser agendadas para um mesmo período. Um período com duas disciplinas conflitantes conta como uma violação. Um período com três disciplinas conflitantes conta como duas violações, sendo uma violação para cada par de disciplinas;
- **Restrição Ocupação das Salas:** duas ou mais aulas distintas não podem ocupar a mesma sala em um mesmo período, sendo considerado como o número de violações cada sala extra encontrada no período conflitante;
- **Restrição Disponibilidade:** caso o professor que leciona a disciplina não tenha disponibilidade em determinado período, nenhuma aula da disciplina deve ser alocada no mesmo. Cada aula agendada em um período com indisponibilidade do professor representa uma violação;

As restrições fracas para o problema são:

- **Restrição Capacidade da Sala:** para cada aula agendada, a sala alocada deve possuir capacidade maior ou igual à exigida pela disciplina. Nesta restrição, para cada estudante a mais que a capacidade da sala, tem-se uma violação;
- **Restrição Mínimo de Dias:** as aulas de uma disciplina devem estar agendadas em um número mínimo de dias pré estabelecido. Para cada dia a menos que o mínimo estabelecido, tem-se uma violação da restrição;
- **Restrição Aulas Isoladas:** aulas de disciplinas de um mesmo currículo devem ser agendadas sempre de maneira adjacente, ou seja, em horários consecutivos de um mesmo dia. Nesta restrição, os horários alocados para um currículo de disciplinas são analisados, e cada disciplina que não é adjacente a nenhuma outra conta como uma violação;

- **Restrição Estabilidade da Sala:** todas as aulas de uma disciplina devem ocorrer em uma mesma sala. Cada sala diferente utilizada por uma disciplina conta como uma violação.

Esta formulação (conjunto de restrições juntamente com os pesos atribuídos a estas) do problema foi concebida com duplo objetivo, qual seja, ser uma abordagem realista e ao mesmo tempo simples e geral o suficiente para atrair muitos pesquisadores. Porém, com o intuito de promover ainda mais a discussão do tema sobre outras perspectivas, em De Cesco et al. (2010) os pesquisadores que estiveram envolvidos na formulação do ITC propõem novas estruturas possíveis para o CBCT, com o objetivo de criar um conjunto de versões padrões do problema, que poderá ser aceito por uma vasta gama de variações.

Para as novas formulações apresentadas por De Cesco et al. (2010), novas restrições Fracas são consideradas:

- **Restrição Janelas:** aulas de disciplinas de um mesmo currículo devem ser agendadas de maneira que não haja janelas de tempo (períodos sem agendamento) entre elas no mesmo dia. Nesta restrição, o número de violações é igual ao tamanho da janela, em períodos;
- **Restrição Mínimo e Máximo de Aulas Por Dia:** para cada currículo, o número de aulas diárias deve estar dentro de um determinado intervalo mínimo e máximo de aulas;
- **Restrição Deslocamentos:** os estudantes devem ter um tempo para se deslocar, quando tiverem duas aulas consecutivas em prédios diferentes. Uma violação é considerada quando duas aulas do mesmo currículo estão agendadas em prédios diferentes e em períodos adjacentes;
- **Restrição Adequação das salas:** determinadas disciplinas exigem recursos especiais (computador, projetor, microscópio, etc.) e estas devem ser alocadas em salas que oferecem estes recursos. Cada aula de uma disciplina alocada em uma sala inadequada representa uma violação;
- **Restrição Agrupamento de Aulas:** as aulas alocadas em um mesmo dia para uma mesma disciplina devem ser agrupadas, sendo um agrupamento a alocação de duas aulas de uma mesma disciplina em horários adjacentes em uma mesma sala. A cada vez que uma disciplina possui mais de uma aula em um mesmo dia, toda aula não agrupada conta como uma violação.

Anteriormente, apenas duas formulações foram consideradas para o problema. A formulação denominada UD1 foi a utilizada no ITC 2002; a formulação UD2 foi a utilizada no ITC 2007. Após este evento, foram propostas três novas formulações, descritas em De Cesco et al. (2010): UD3, que prioriza as restrições que beneficiam o aluno; UD4, que visa a construção de quadro de horários compactos que respeitem a adequação das salas; e UD5, que considera o problema de deslocamento entre prédios. Na Tabela 2.1 são apresentadas as formulações para o CBCT descritas em De Cesco et al. (2010).

Tabela 2.1: Descrição das Formulações para o CBCT

RESTRIÇÕES	Formulações				
	UD1	UD2	UD3	UD4	UD5
<i>Aulas</i>	F	F	F	F	F
<i>Conflitos</i>	F	F	F	F	F
<i>Ocupação das Salas</i>	F	F	F	F	F
<i>Disponibilidade</i>	F	F	F	F	F
<i>Capacidade da Sala</i>	1	1	1	1	1
<i>Restrição Mín. de Dias</i>	5	5	–	1	5
<i>Restrição Aulas Isoladas</i>	1	2	–	–	1
<i>Estabilidade da Sala</i>	–	1	–	–	–
<i>Janelas</i>	–	–	4	1	2
<i>Mín. e Máx. de Aulas P/Dia</i>	–	–	2	1	2
<i>Deslocamentos</i>	–	–	–	–	2
<i>Adequação das salas</i>	–	–	3	F	–
<i>Agrupamento de Aulas</i>	–	–	–	1	–

As restrições fortes do problema recebem altos pesos, que são representados na Tabela 2.1 pela letra F. Em todas as formulações, as restrições fortes do problema são as mesmas, com exceção da formulação UD4, que considera a restrição “Adequação das Salas” como uma restrição forte. É possível notar também que as restrições fracas do problema recebem diferentes pesos em cada formulação, o que determina, de certa forma, diferentes objetivos entre as formulações descritas.

Com os dados e publicações provenientes das competições, juntamente com o trabalho de De Cescio et al. (2010), é possível obter uma ótima “infra-estrutura” necessária para análise comparativa de novas técnicas a serem implementadas para a solução do problema. No endereço de internet <http://tabu.diegm.uniud.it/ctt/>, encontram-se definidos validadores nas diferentes formulações do problema; instância; *lower bounds*; soluções; *ranking* das soluções para cada instância; dentre outras informações com respeito à solução do problema. Dessa forma, mesmo depois da última competição, realizada em 2007, o CBCT continua sendo alvo de novas pesquisas, as quais são comparadas e classificadas através do endereço de internet mencionado.

O presente estudo propõe a utilização da formulação UD2 referente ao ITC 2007. Os resultados parciais alcançados serão comparados com os obtidos na competição. A utilização da formulação UD2 mostra-se útil por ser essa a formulação mais discutida na literatura e, também, por apresentar as principais características envolvidas na maioria dos problemas desta classe.

2.4 Formulação Matemática para o Problema

Várias formulações matemáticas podem ser encontradas na literatura para os diferentes tipos de problemas de Programação de Horários Educacionais. As for-

mulações variam de acordo com a classificação, restrições e aspectos desejados do problema tratado. Dentre estas diversas formulações, podem ser citadas os trabalhos de de Werra (1985), Schaerf (1999), Even et al. (1976) e Souza (2000).

Um modelo matemático para o problema de programação de horários baseada em currículos pode ser visto em Lubbecke e Lach (2010). Trata-se de um modelo relativamente simples que tem sido utilizado com certa eficiência na obtenção de *lower bounds* e *upper bounds*, como é mostrado em Lü et al. (2011). A seguir, é apresentada uma breve descrição do modelo matemático em questão. Maiores detalhes podem ser encontrados em Lubbecke e Lach (2010). As principais entidades envolvidas no modelo são:

- C, CU, R, S, D, P, T são, respectivamente, disciplinas, currículos, salas, diferentes capacidades das salas, dias, períodos e professores;
- $P(c)$ é o subconjunto de períodos P disponíveis para a disciplina c ;
- $C(t)$ é o subconjunto de disciplinas C ministradas pelo professor t ;
- $C_{\geq s}$ é o subconjunto de disciplinas C que requer salas com mais de s assentos;
- $R_{\geq s}$ é o subconjunto de salas R que possuem mais do que s assentos;
- $l(c)$ é o número de aulas para a disciplina c ;
- $mnd(c)$ é o número mínimo de dias com aulas prescritos para uma disciplina c ;
- x são variáveis de decisão binárias indexadas de acordo com os períodos e as disciplinas. Para cada par disciplina/período, a variável x é incluída somente se o professor ministrante estiver disponível para aula no horário p . Dessa forma, a restrição *Indisponibilidades* não é desrespeitada. A disciplina c é ministrada no horário p se $x_{c,p}$ é igual a 1;

A partir das variáveis de decisão x são determinadas cinco variáveis: y, z, w, r e v . O processo de resolução consiste em minimizar uma combinação linear entre essas variáveis sobre condições de penalidade. A combinação linear em questão trata da função objetivo do problema, apresentada a seguir:

$$\sum_{p \in P, s \in S, c \in C \geq s} \alpha \cdot obj_{s,c,p} \cdot y_{s,c,p} + \sum_{c \in C} \beta \cdot w_c + \sum_{cu \in CU, p \in P} \gamma \cdot v_{cu,p} \quad (2.1)$$

O primeiro somatório da função objetivo considera, para cada aula de cada disciplina, o número de estudantes que ultrapassam (caso exista) a quantidade máxima de assentos para a sala alocada. Se o número de estudantes que comparecem a uma aula da disciplina c no período p é maior que a capacidade da sala alocada, o valor $obj_{s,c,p}$ é a diferença entre o número de estudantes atendidos na disciplina e a capacidade máxima da sala. Nesses termos, tem-se ainda a variável $y_{s,c,p}$, que é referente à restrição estabilidade de sala.

O segundo termo representa a restrição *Mínimo de dias Alocados*. O coeficiente β representa o peso atribuído à restrição. Já no terceiro termo, a variável $v_{cu,p}$ representa o número de aulas isoladas em cada currículo.

A formulação matemática completa para o problema é, então:

$$\min \sum_{p \in P, s \in S, c \in C_{\geq s}} \alpha obj_{s,c,p} \cdot y_{s,c,p} + \sum_{c \in C} \beta \cdot w_c + \sum_{cu \in CU, p \in P} \gamma \cdot v_{cu,p} \quad (2.2)$$

Sujeito a:

$$\sum_{p \in P} x_{c,p} = l(c) \quad \forall c \in C \quad (2.3)$$

$$\sum_{c \in C} x_{c,p} \leq |R| \quad \forall p \in P \quad (2.4)$$

$$x_{c,p} - y_{c,p} \geq 0 \quad \forall s \in S, c \in C_{\geq s}, p \in P \quad (2.5)$$

$$\sum_{p \in P} x_{c,p} - y_{c,p} \leq |R_{\geq s}| \quad \forall s \in S, p \in P \quad (2.6)$$

$$\sum_{p \in d} x_{c,p} - z_{c,d} \geq 0 \quad \forall c \in C, d \in D \quad (2.7)$$

$$\sum_{d \in D} z_{c,d} + w_c \geq mnd(c) \quad \forall c \in C \quad (2.8)$$

$$\sum_{c \in cu} x_{c,p} - r_{cu,p} = 0 \quad \forall cu \in CU, p \in P \quad (2.9)$$

$$-r_{cu,p-1} + r_{cu,p} - r_{cu,p+1} - v_{cu,p} \leq 0 \quad \forall cu \in CU, p \in P \quad (2.10)$$

$$\sum_{c \in C(t)} x_{c,p} \leq 1 \quad \forall t \in T, p \in P \quad (2.11)$$

$$x_{c,p} \in \{0, 1\} \quad (2.12)$$

$$y_{s,c,p} \in \{0, 1\} \quad (2.13)$$

$$z_{c,d} \in \{0, 1\} \quad (2.14)$$

$$w_c \in Z_+ \quad (2.15)$$

$$r_{cu,p} \in \{0, 1\} \quad (2.16)$$

$$v_{cu,p} \in \{0, 1\} \quad (2.17)$$

A restrição (2.3) impõem que o número de aulas semanais estipulado para cada disciplina seja obrigatoriamente alocados (restrição *Aulas*). As restrições (2.4, 2.5 e 2.6) são relacionadas às restrições *Ocupação das Salas* e *Capacidade da Sala*, respectivamente. As restrições (2.7 e 2.8) são relacionadas à restrição *Mínimo de Dias*. As restrições (2.9 e 2.10) são relacionadas às restrições *Aulas Isoladas* e *Conflitos*. E, por fim, a restrição (2.11) também é relacionada à restrição *Conflitos*, verificando, contudo, conflitos entre aulas de diferentes disciplinas que são ministradas por um mesmo professor.

Capítulo 3

RESOLUÇÃO DE PROBLEMAS DE OTIMIZAÇÃO VIA META-HEURÍSTICAS

Como o problema alvo deste estudo é considerado um problema de otimização combinatória, sua resolução pode ser dada por diversas técnicas desenvolvidas ao decorrer dos últimos anos, as quais apresentam diferentes níveis de eficiência para cada abordagem realizada. Essas técnicas podem ser classificadas em dois grandes grupos: métodos exatos ou completos e métodos aproximados.

Métodos exatos, garantem a obtenção da melhor solução de um problema para toda instância de tamanho finito em um espaço de tempo limitado. Contudo, para problemas de otimização combinatória classificados como Np-difícil, não são conhecidos algoritmos de resolução em um tempo polinomial. Dessa forma, no problema de programação de horários a resolução via métodos exatos pode exigir tempo computacional exponencial, tornando inviável a aplicação dessas técnicas em determinadas instâncias em que uma solução não pode ser obtida em um tempo computacional aceitável.

Já os métodos aproximados, tem como objetivo a obtenção de boas soluções, não necessariamente as melhores. Esta classificação é composta por métodos heurísticos e meta-heurísticos, que se destacam na resolução de problemas de elevada complexidade, por fornecerem no geral uma boa solução em um tempo computacional razoável.

Os algoritmos aplicados ao PPH geralmente devem considerar um aspecto importante, o tempo de execução. Apesar de alguns problemas terem um tempo considerável para sua resolução, este ainda é um aspecto imprescindível a classe de problemas. Dessa forma, mesmo com a evolução dos métodos exatos com a criação de novas técnicas e abordagens que agilizam a busca por uma solução - algumas até híbridas com a utilização de métodos aproximados - há que se destacar que em aspectos gerais a utilização de métodos aproximados neste problema tem se apresentado com maior eficiência no que diz respeito a tempo de execução/qualidade da solução.

Neste contexto, o presente capítulo visa apresentar alguns conceitos importantes que envolvem a resolução do problema de programação de horários via meta-heurísticas. Os seguintes conceitos são apresentados: Otimização Combinatória;

Vizinhança; Heurísticas; e Meta-heurísticas. Nas subseções Heurísticas e Meta-Heurísticas são apresentadas algumas técnicas que poderão ser ou foram utilizadas para a resolução do problema neste trabalho.

3.1 Otimização Combinatória

O termo otimização é definido por Deb e Talbi (2001) como um processo de encontrar e comparar soluções factíveis, que são ditas boas ou ruins em termos de objetivos almejados, sendo apresentado ao final do processo a melhor solução encontrada.

De acordo com Osman e Kelly (1997), otimização combinatória é o estudo matemático e computacional que consiste em encontrar um ótimo arranjo, agrupamento, ordenação ou seleção para objetos discretos disponíveis em um conjunto finito. Sendo assim, pode-se dizer que são problemas de otimização que apresentam como característica um espaço de soluções finitas e discretas.

Problemas de otimização combinatória geralmente são de fácil entendimento e difícil resolução (Osman e Kelly, 1997). Tal afirmação pode ser comprovada pela quantidade de problemas combinatórios classificados como NP-Completo ou NP-Difícil, alguns problemas podem ser citados como: o Problema do Caixeiro Viajante; variantes de problemas que envolvem escalonamento e agendamento, Roteamento de Veículos e Problemas de Particionamento de Conjuntos.

Formalmente um problema de otimização, para o caso de minimização, pode ser descrito pelas equações abaixo:

$$\min f(x) \quad (3.1)$$

$$\text{sujeito a } s \in S \quad (3.2)$$

Para $f : \mathbb{R}^n \rightarrow \mathbb{R}$ a função objetivo pode ser minimizada e $S \subset \mathbb{R}^n$ a região viável ou região de viabilidade, ou seja, o conjunto finito das possíveis soluções viáveis para o problema em questão. Para resolver um problema de otimização combinatória deve-se encontrar uma solução ótima $s^* \in S$, que atenda à relação:

$$f(s^*) \leq f(s), \forall s \in S \quad (3.3)$$

Sendo assim, pode-se dizer que a solução para o problema é dada pelo menor (ou maior em casos de maximização) valor encontrado na região factível do problema, ou seja, o menor valor que respeite as restrições impostas. Para essa solução dá-se o nome de ótimo global.

Na exploração do espaço de busca pode ser encontrada uma solução que representa um ponto mínimo encontrado em determinada região do espaço de soluções pesquisado, esta solução é denominada ótimo local. Ressalta-se que ótimos locais podem atrapalhar um processo de busca, evitando que uma solução ótima global seja encontrada. Dessa forma, afim de ilustrar um problema de otimização combinatória, é apresentada na Figura 3.1 uma representação do problema a qual deseja-se minimizar uma função $f(\cdot)$ (são representados os ótimos locais e um ótimo Global para a função hipotética).

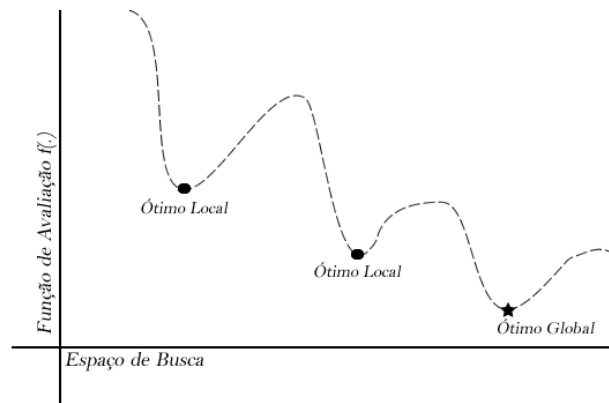


Figura 3.1: Representação de Ótimos Locais e um Ótimo Global

3.2 Conceito de Vizinhança e Movimento

Segundo Souza (2000), sendo o espaço de pesquisa de um problema de otimização definido por S e a função objetivo a se minimizar definida por f . A função N , a qual depende da estrutura do problema tratado, associa a cada solução $s \in S$, sua vizinhança $N(S) \subseteq S$, sendo cada solução $s' \in N(S)$ chamada de vizinho de s . Ou seja, uma vizinhança da solução s pode ser considerada como um conjunto de novas soluções geradas a partir de alterações (movimentos) aplicadas na solução s .

Um movimento, no que diz respeito a geração de vizinhanças, pode ser entendido como um mecanismo que gera um conjunto de vizinhos de determinada solução, modificando um atributo ou um conjunto de atributos pertencentes a instância do problema em que a solução é dada. Sendo assim, um movimento é a função presente na transição de uma solução s em outra $s' \in N(s)$, em apenas uma iteração (Osman e Kelly, 1997). Na prática, movimentos são representados por operações como trocas, inserções e remoções em atributos da solução corrente.

Pode-se dizer que em resoluções que utilizam o conceito de vizinhança, a exploração do espaço de busca do problema tem forte relação com os movimentos adotados e vizinhanças pesquisadas. Assim, busca-se pesquisar diferentes tipos de exploração de vizinhanças, com o objetivo de aumentar a eficiência de determinadas técnicas ao percorrer o espaço de busca.

Nesse sentido Gaspero e Schaerf (2006) definem vizinhanças compostas por soluções que são geradas por mais de um movimento. As soluções em questão são geradas a partir de n tipos de movimentos aplicados em sequência. Este tipo de estratégia permite que movimentos avançados sejam criados visando aumentar a capacidade de busca de uma estrutura de vizinhança. Ressalta-se, que alguns dos movimentos adotados nesse trabalho são combinações entre movimentos básicos diferentes.

3.3 Heurísticas

Heurísticas são procedimentos algorítmicos baseados em modelos intuitivos que procuram uma boa solução a um custo computacional aceitável, sem, no entanto, garantir quão próximo está da solução ótima. Dessa forma, um algoritmo dito

heurístico busca, sem oferecer garantias, soluções de boa qualidade para problemas complexos utilizando uma quantidade não muito grande de recursos - especialmente no aspecto relativo ao tempo gasto para execução.

Abbass e Newton (2001) afirmam que técnicas heurísticas tendem a percorrer o espaço de busca de forma “inteligente”, utilizando um conhecimento prévio do problema como guia para pesquisa em regiões promissoras do espaço de soluções. Dessa forma, pode-se dizer que as heurísticas englobam estratégias, procedimentos e métodos aproximados com o objetivo de produzir, em tempo reduzido, soluções tão próximas quanto possível da solução ótima.

A maioria das heurísticas são técnicas baseadas nos processos de construção ou refinamento de soluções. Sendo assim, as primeiras heurísticas criadas poderiam ser categorizadas de duas maneiras: heurísticas de construção e heurísticas de refinamento, as quais serão descritas nas subseções a seguir.

3.3.1 Heurísticas de Construção

Heurísticas construtivas partem de uma solução vazia, sendo inserido elemento a elemento a cada iteração, até que alguma solução seja construída. A seleção de qual elemento vai ser inserido e em que parte da solução esse será associado, varia de acordo com o critério heurístico do método de construção, o qual, depende do problema abordado.

A heurística de construção mais simples é representada por um método aleatório de construção de soluções. Nesse método as alocações são realizadas de maneira aleatória até que uma solução seja construída. Nas heurísticas clássicas (ou gulosas) os elementos são ordenados de acordo com uma função gulosa, que avalia cada possível inserção, sendo realizada a cada iteração a alocação que apresentar o melhor nível de benefício avaliado.

No geral as heurísticas gulosas - em se tratando de qualidade de soluções geradas - apresentam soluções superiores as heurísticas de construção aleatória, pelo fato da construção aleatória não apresentar nenhuma “inteligência” nas operações realizadas.

Contudo, os métodos de construção gulosa também apresentam desvantagens: i) uma alocação realizada não pode ser desfeita, dessa forma, as últimas alocações tendem em alguns casos, a piorar consideravelmente a solução gerada; ii) as soluções geradas apresentam pouca ou nenhuma diversidade. Essas desvantagens podem ser minimizadas com a introdução de uma componente aleatória juntamente com o aspecto guloso, como pode ser visto na meta-heurística GRASP.

No geral em problemas considerados difíceis, as heurísticas de construção não produzem boas soluções e geralmente são aplicadas juntamente com heurísticas de refinamento ou meta-heurísticas.

3.3.2 Heurísticas de Refinamento

Heurísticas de refinamento, também são chamadas de heurísticas de melhoria ou heurísticas de busca local, constituem uma categoria de técnicas baseadas na noção de vizinhança. Basicamente, essas técnicas partem de uma solução inicial qualquer, explorando um espaço de soluções através de sucessivas soluções vizinhas obtidas

iterativamente até que nenhuma solução de melhora seja encontrada ou outro critério de parada seja satisfeito.

Hertz e Widmer (2003) define heurísticas de refinamento como um procedimento que percorre uma determinada trajetória em um grafo não-orientado $G = (S, E)$, onde S representa o conjunto de soluções s do problema e E o conjunto de arestas (s, s') , com $s' \in N(s)$.

Nas subseções a seguir são descritas duas heurísticas de refinamento. Sendo estas: Método de Descida; e Método Randômico de Descida.

3.3.2.1 Método de Descida

Trata-se de um procedimento que tem como princípio analisar todos os possíveis vizinhos de uma solução s em sua vizinhança $N(s)$, escolhendo, a cada iteração, aquele que apresente uma melhora no valor corrente da função de avaliação. O processo em questão se inicia a partir de uma solução inicial qualquer e é repetido até que a solução corrente não possa mais ser melhorada. O método de descida é apresentado como um pseudocódigo no Algoritmo 1.

Algoritmo 1: Método de Descida

Entrada: $f(\cdot), N(\cdot), s$
 $V = \{s' \in N(s) | f(s') < f(s)\};$
enquanto $|V| > 0$ **faça**
 Selecione $s' \in V$, onde $s' = \arg \min\{f(s') | s' \in V\};$
 $s \leftarrow s';$
 $V = \{s' \in N(s) | f(s') < f(s)\}$
fim
Saída: s

As principais características desse tipo de busca local são: i) analisar **todas** as possíveis soluções da vizinhança adotada a cada iteração; ii) aceitar um vizinho qualquer somente se ele **melhorar estritamente** o valor da melhor solução encontrada até então. Assim sendo, pode-se afirmar que quando a técnica em questão é encerrada, têm-se como solução um ótimo local para vizinhança analisada.

3.3.2.2 Método Randômico de Descida

O método randômico de descida é uma variação do método apresentado na subseção anterior, trata-se de um procedimento que analisa um vizinho qualquer e o aceita somente se ele for estritamente melhor que a solução corrente. Caso o vizinho analisado não apresente um valor de melhora, a solução corrente permanece inalterada e um novo vizinho é gerado. O procedimento é repetido até atingir um número máximo de iterações sem melhora.

A principal vantagem desse procedimento consiste em evitar a busca exaustiva por toda vizinhança, como acontece no método de descida. Característica que pode ser desejável em problemas em que a vizinhança de uma solução é muito extensa. Contudo, este método também fica preso no primeiro mínimo local encontrado.

O método randômico de descida é apresentado como um pseudocódigo no Algo-

ritmo 2.

Algoritmo 2: Método Randômico de Descida

Entrada: $f(\cdot)$, $N(\cdot)$, s , $iterMax$
 $iter \leftarrow 0$;
 $s \leftarrow \text{SoluçãoInicial}()$;
enquanto $iter < iterMax$ **faça**
 $iter \leftarrow iter + 1$;
 Selecione aleatoriamente s' em $N(s)$;
 se $f(s) < f(s_*)$ **então**
 $s_* \leftarrow s$;
 $iter \leftarrow 0$;
 fim
fim
Saída: s_*

Esse procedimento possui uma variação denominada método não descendente randômico, sua principal diferença está em aceitar o vizinho gerado aleatoriamente se ele for melhor ou igual à solução corrente. Esse método possui a capacidade de percorrer em um espaço de busca com movimentos laterais (Selman et al., 1992), sendo promissora sua utilização em problemas em que várias soluções podem ter a mesma avaliação (regiões planas).

3.4 Meta-heurísticas

As heurísticas ditas convencionais (construção e refinamento) são muito específicas para problemas particulares, o que prejudica a eficiência (em alguns casos até a aplicação) dessas em uma classe mais ampla de problemas. A criação e utilização de meta-heurísticas para resolução de problemas de otimização teve início na década de 1980 (Glover, 1986), as pesquisas se iniciaram no sentido de criar novos procedimentos heurísticos com uma certa estrutura teórica e caráter geral para que apresentassem assim, certa flexibilidade.

A reunião de conceitos das áreas de Otimização e Inteligência Artificial, permitiram a criação de métodos “inteligentemente flexíveis”, comumente referenciados como meta-heurísticas. Glover e Kochenberger (2003), define meta-heurística como sendo métodos de resolução que “orquestram” uma interação entre procedimentos de melhoria local e diferentes estratégias de busca, para criar um processo capaz de escapar de ótimos locais e realizar uma pesquisa robusta em um espaço de soluções. Ou seja, meta-heurísticas são de caráter geral e têm condições de escapar de ótimos locais, ao contrário das heurísticas convencionais.

Muitas aplicações que utilizam meta-heurísticas têm mostrado a eficiência e a eficácia dessas técnicas para resolver, principalmente, problemas grandes e complexos (Talbi, 2009). A aplicação dessas técnicas pode ser notada em várias áreas tais como: Aprendizado de Máquinas, Mineração de Dados, Modelagem de Sistemas, Simulações, Problemas de Planejamento, Roteamento, entre outras. Salienta-se, que não existe uma meta-heurística que possa ser considerada melhor com relação a outra, uma vez que o comportamento dessas técnicas está diretamente relacionado com a natureza do problema e os objetivos buscados.

Meta-heurísticas se diferenciam basicamente pela forma em que escapam das armadilhas dos ótimos locais. Quanto à sua classificação, Blum e Roli (2003), categorizam as técnicas de acordo com o número de soluções usadas para explorar o espaço de soluções, o que se dá através de uma única solução ou em conjuntos de soluções (população).

Técnicas que trabalham com uma única solução recebem a denominação de meta-heurísticas de trajetória, que se caracterizam por explorar o espaço de busca através de movimentos aplicados a cada iteração sobre uma solução corrente, gerando outra solução promissora em sua vizinhança. Entre as técnicas que se enquadram nesta categoria pode-se citar: Busca Tabu (Glover e Laguna, 1997); *Simulated Annealing* (Dowsland, 1993); *Variable Neighborhood Search* - VNS (Hansen e Mladenović, 2001); e *Iterated Local Search* - ILS (Lourenço et al., 2003).

Técnicas que utilizam um conjunto de soluções são denominadas como meta-heurísticas populacionais, caracterizadas por realizarem processos de busca que descrevem a evolução de um conjunto de pontos no espaço de soluções. São algumas das técnicas que se enquadram nesta categoria: Algoritmos Genéticos (Melanie, 1998); os Algoritmos Meméticos (Moscato, 1999); e o Algoritmo de Colônia de Formigas (Dorigo et al., 2005).

Segundo Talbi (2009) nos últimos anos, o interesse em hibridização de meta-heurísticas aumentou consideravelmente no campo da pesquisa operacional. Fato que se deve aos recentes melhores resultados encontrados para muitos problemas de otimização obtidos através de algoritmos híbridos. A hibridização de meta-heurísticas tem proporcionado a criação de poderosos algoritmos de busca, os quais são dados por combinações entre diferentes meta-heurísticas ou com diferentes abordagens como: métodos exatos; programação por restrições; e técnicas de aprendizado de máquina.

Neste trabalho são utilizados conceitos de diferentes meta-heurísticas, pode-se dizer assim, que trata-se de uma abordagem híbrida em que aspectos das seguintes técnicas são utilizadas: GRASP, Busca Tabu e ILS. As meta-heurísticas em questão serão apresentadas na subseções abaixo.

3.4.1 Grasp

O GRASP (Greedy Randomized Adaptive Search Procedure - Procedimento de Busca Adaptativa Gulosa e aleatória) é uma meta-heurística iterativa proposta por Feo e Resende (1995), na qual cada iteração conta com duas fases : i) a fase de construção, onde uma solução é gerada elemento a elemento, via um procedimento guloso-aleatório; ii) a fase de busca local, na qual um ótimo local da vizinhança da solução construída é pesquisado.

Como a fase de construção apresenta também um caráter aleatório, sua utilização juntamente com uma busca local pode proporcionar a localização de diferentes ótimos locais para um problema. De certa forma, esse tipo de estratégia possibilita a técnica a escapar da armadilha dos ótimos locais. O método em questão, retorna o melhor resultado encontrado diante de um determinado número de iterações

pré-estabelecidas. Assim, um pseudocódigo é apresentado através do Algoritmo 3.

Algoritmo 3: GRASP

```

enquanto critério de parada não satisfeito faça
     $s' \leftarrow \text{ConstruaSoluçãoGulosaAleatória}(\alpha)$ ;
     $s \leftarrow \text{BuscaLocal}(s')$ ;
    se  $f(s) < f(s_*)$  então
         $s_* \leftarrow s$ ;
    fim
fim
Saída:  $s_*$ 

```

Na fase de construção, uma solução é iterativamente construída, de forma que os elementos candidatos ou possíveis elementos são alocados e ordenados em uma lista C de candidatos. Em cada iteração da técnica é realizada uma alocação de um elemento candidato na solução, e a reordenação da lista de candidatos.

Os processos de ordenação e alocação dos elementos candidatos são baseados em uma função adaptativa gulosa, que estima o nível de benefício alcançado com a seleção de cada um dos elementos presentes na lista C , de maneira que, a cada iteração da fase de construção seja realizada uma alocação que traga mais benefícios a solução.

A meta-heurística é dita adaptativa, pois, os benefícios associados à escolha de cada um dos elementos candidatos estão diretamente ligados entre si, isso porque, de certa forma, os benefícios alcançados com uma nova alocação dependem das alocações já realizadas. Característica que é perceptível a fase de construção do GRASP pelo fato da lista de candidatos ser reordenada a cada iteração.

Segundo Souza (2000), o caráter probabilístico da técnica fica a cargo de que cada elemento ser selecionado de forma aleatória a partir de um subconjunto restrito formado pelos melhores elementos que compõem a lista de candidatos. Esse subconjunto recebe o nome de lista de candidatos restrita (LCR). Salienta-se, que a fase de construção recebe como parâmetro α , que é responsável por definir o número de elementos que compõem o subconjunto LCR. A importância deste parâmetro deve-se ao fato de que seu valor é responsável por controlar o nível de gulosidade e aleatoriedade na execução do método.

Pode-se dizer então que a fase de construção do GRASP tenta reunir os aspectos positivos de construções gulosas e construções aleatórias, gerando uma boa diversidade de soluções que tendem a apresentar uma qualidade razoável. Já a fase de busca local é de grande importância na técnica também, pois, através desta, a diversidade das soluções construídas é explorada em busca de uma solução ótima.

3.4.2 Busca Tabu

A técnica denominada Busca Tabu (*Tabu Search*) foi formalizada em Glover (1989). Trata-se de uma das meta-heurísticas mais difundidas para resolução de problemas de otimização combinatória. Tem como principal aspecto a utilização de uma memória (Lista Tabu) que armazena informações sobre o processo de busca, evitando determinadas soluções.

A Busca Tabu faz uso de uma busca local a cada iteração, que consiste em

explorar um dado espaço de soluções determinado por uma vizinhança analisada, movendo-se obrigatoriamente para o melhor vizinho dentre os pesquisados na respectiva iteração, mesmo que este represente uma piora da solução corrente anterior. A Lista Tabu é aplicada justamente no processo de escolha do melhor vizinho, esta estrutura de memória armazena informações de movimentos proibidos que evitam soluções geradas previamente, busca-se com sua aplicação uma forma de se evitar que o algoritmo não fique preso nos ótimos locais.

Em sua forma clássica, a Lista Tabu, tem a cada nova alteração da solução corrente, um novo movimento (ou solução) correspondente adicionado e o movimento mais antigo eliminado, tendo funcionamento associado ao de uma fila de tamanho fixo. A definição do tamanho da Lista Tabu é de extrema importância, pois, pode determinar um comportamento da técnica mais voltado para diversificação ou para intensificação, em um processo de busca (Talbi, 2009). Com uma Lista Tabu muito grande, uma determinada região de busca será evitada por várias iterações privilegiando a descoberta de novas bacias de atração, enquanto uma Lista Tabu pequena, permite que diferentes ótimos locais (eventualmente um ótimo global) que estejam próximos sejam visitados.

Os movimentos considerados tabu são proibitivos a menos que estes satisfaçam a um critério chamado de critério de aspiração, de forma que, em geral, este critério é que a solução gerada mesmo sendo tabu seja uma solução melhor do que a melhor solução encontrada até então (s_*). De acordo com Souza (2000), o critério aspiração ou também chamado de função aspiração, fundamenta-se no fato de que soluções melhores que s_* corrente, ainda que geradas por movimentos tabu, podem não ter sido visitadas anteriormente. Isso evidencia que a lista tabu pode impedir tanto o retorno de uma solução gerada anteriormente, quanto de soluções ainda não geradas. Desse modo, a lista tabu de movimentos juntamente com a função de aspiração tenta garantir que o algoritmo não fique “preso” e não deixe de visitar determinadas soluções que poderiam não ser visitadas por apresentarem movimentos ditos tabu.

É apresentado através do Algoritmo 4 um pseudocódigo da meta-heurística Busca Tabu básica.

Algoritmo 4: Busca Tabu

```

 $s_0 \leftarrow \text{SoluçãoInicial}();$ 
 $T[] \leftarrow \text{InicializarListaTabu}();$ 
enquanto critério de parada não satisfeito faça
     $s' \leftarrow \text{Melhor solução permitida da vizinhança pesquisada};$ 
    // solução não tabu ou que respeita critério de aceitação;
     $T[] \leftarrow \text{AtualizaListaTabu}();$ 
    se  $s' < s_*$  então
         $s_* \leftarrow s';$ 
    fim
fim
Saída:  $s_*$ 

```

Para interromper o algoritmo podem ser utilizados diferentes critérios de parada. Por exemplo, a busca pode ser encerrada quando é atingido um certo número máximo de iterações sem melhora ou quando o valor da melhor solução chega a um limite inferior conhecido (ou próximo dele). Ao final da execução é retornada a melhor

solução encontrada durante o processo de busca, representada no algoritmo por s_* .

3.4.3 ILS

O ILS (*Iterated Local Search*), foi proposto por Lourenço et al. (2003), e parte do princípio que um ótimo local obtido por uma busca local é dependente da solução de partida. Sendo assim, o procedimento de busca local pode ser melhorado através de sucessivas buscas a partir de diferentes soluções obtidas por meio de perturbações em ótimos locais previamente encontrados.

Basicamente a cada iteração o ILS realiza uma perturbação em uma boa solução (provavelmente um ótimo local) e em seguida realiza uma busca local partindo da solução perturbada, em busca de novas soluções ainda não exploradas. A solução encontrada pela busca local pode ser atribuída como nova solução corrente ou não, de acordo com um critério de aceitação definido no método. Este processo é repetido até que a condição de parada seja atendida.

Para aplicação do ILS na resolução de problemas de otimização, quatro são os principais componentes a serem especificados: i) solução inicial, como será gerada a primeira solução do método; ii) busca local, como será feita a exploração da vizinhança da solução corrente; iii) perturbação, qual o tipo de alteração será feita na solução corrente; iv) e critério de aceitação, quando as soluções serão aceitas e qual o nível de intensidade da perturbação. É apresentado abaixo um pseudocódigo do ILS através do Algoritmo 5.

Algoritmo 5: ILS

```

 $s_0 \leftarrow \text{SoluçãoInicial}();$ 
 $s_* \leftarrow \text{BuscaLocal}(s_0);$ 
enquanto critério de parada não satisfeito faça
     $s' \leftarrow \text{Perturbação}(s_*, \text{nível});$ 
     $s'_* \leftarrow \text{BuscaLocal}(s');$ 
     $s_* \leftarrow \text{CritérioAceitação}(s_*, s'_*, \text{nível});$ 
fim
Saída:  $s_*$ 

```

Um aspecto de grande importância no ILS é a intensidade da perturbação. Ela deve ser forte o suficiente para permitir que a técnica escape de um ótimo local e ao mesmo tempo fraca o suficiente para explorar uma região promissora próxima ao ótimo local corrente. Dessa forma, a intensidade da perturbação é definida pelo nível desta, e por sua vez o nível é definido pelo critério de aceitação.

Geralmente as perturbações começam em um nível baixo (fracas) e a cada vez que um determinado número de iterações ocorre sem nenhuma melhora o nível é incrementado no critério de aceitação. Assim, o objetivo é que as perturbações apresentem em regiões promissoras a característica de intensificação de busca, e quando necessário (ótimo local distante) a característica de diversificação de busca.

Na Figura 3.2 são representados os princípios básicos que envolvem o funcionamento do ILS em um possível comportamento de sua utilização para resolução de um problema de otimização.

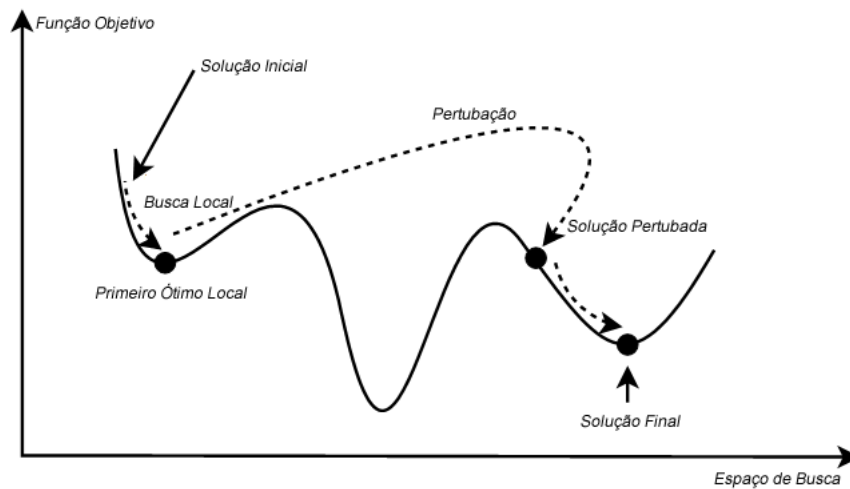


Figura 3.2: Princípios básicos que envolvem o funcionamento do ILS

Através da Figura 3.2 é possível ter uma clara noção sobre o comportamento do ILS. Primeiramente uma solução gerada é melhorada através de uma busca local, em seguida a mesma é perturbada para uma nova região. Novamente através de uma busca local é possível chegar a um novo mínimo local, melhor que o primeiro encontrado. Contudo, caso a busca se encerrasse neste passo o ponto encontrado não seria o ótimo global do problema.

Capítulo 4

TRABALHOS RELACIONADOS

O estudo da automatização do processo de construção de quadro de horários vêm sendo alvo de inúmeras pesquisas nas últimas cinco décadas, sendo consideradas diferentes resoluções, formulações, análises e classificações. Dessa forma, a literatura conta com uma vasta quantidade de trabalhos publicados, destacando-se a realização de eventos específicos sobre o tema, como o PATAT e o ITC. A importância dos trabalhos relacionados advém da necessidade de se justificar determinados aspectos do problema e de sua resolução, os quais podem ser verificados no contexto histórico da abordagem ao tema. Ressalta-se também, o favorecimento na percepção de novas tendências futuras para resolução do problema.

Sendo assim, neste capítulo são apresentados, através de uma breve descrição, alguns dos trabalhos mais relevantes para a realização deste estudo.

4.1 Revisões do Estado da Arte

Primeiramente, são destacados alguns *surveys* relacionados à problemas de programação de horários educacionais.

Em de Werra (1985) foram introduzidos alguns modelos matemáticos para o problema e apresentada uma pesquisa relacionada a abordagem realizada através de coloração de grafos e métodos de rede de fluxo. O mesmo autor apresenta em de Werra (1997) uma revisão sobre os modelos matemáticos e a utilização de métodos exatos e de coloração de grafos em problemas de *timetabling* educacional.

Wren (1996) pesquisou sobre a relação existente entre problemas de escalonamento, programação de horários e roteamento, através de um exemplo do problema do caixeiro viajante. O trabalho apresenta uma pesquisa bibliográfica que comprova a relação detectada entre esses tipos de problema e como a combinação de aspectos de resolução entre esses podem trazer benefícios para as áreas envolvidas.

Carter e Laporte (1997) apresentam uma pesquisa verificando os diferentes métodos de resolução para problemas da classe *Course Timetabling*. As técnicas de resolução foram classificadas e descritas em quatro categorias, sendo: métodos de *cluster*, métodos sequenciais, pesquisa generalizada (heurísticas e meta-heurísticas) e programação baseada em restrições.

Já Schaerf (1999) desenvolve sua pesquisa em torno das classificações delimitadas no problema, como: *School Timetabling*, *Course Timetabling* e *Exam Timetabling*. São verificados e analisados diferentes métodos de resolução para problema de acordo

com cada classificação, sendo apresentadas também possíveis direções futuras para a pesquisa na área. No contexto de classificações para o PPHE ressalta-se também o trabalho de Reis e Oliveira (2000), que apresenta oito classificações baseadas em abordagens realizadas no problema juntamente com uma linguagem de representação para problemas de *Timetabling*.

Em Burke et al. (1997) e Burke e Petrovic (2002), é apresentada uma visão geral sobre a automação do problema de programação de horários em universidades (*Exam Timetabling*, *Course Timetabling*). O estudo baseia-se nas diferentes formas de resolução do problema como: algoritmos híbridos, multi-objetivos, meta-heurísticas, abordagens adaptativas, programação por restrições, entre outras. Direções futuras para pesquisa na área são apontadas nesse trabalho, destacando-se a utilização de abordagens que visam aumentar a generalidade de sistemas de *Timetabling*.

Por sua vez Lewis (2008) apresentou uma pesquisa recente que tem como principal aspecto de discussão a automação do problema de programação de horários em universidades (*Exam Timetabling*, *Course Timetabling*) através de meta-heurísticas. O estudo se concentra nas principais meta-heurísticas utilizadas, dividindo as abordagens em três categorias: algoritmos com um estágio de otimização, algoritmos com dois estágios de otimização e algoritmos que permitem a relaxação de restrições. Outro aspecto desta pesquisa trata-se de uma análise realizada nas restrições do problema e também nas instâncias testes comuns na literatura.

Qu et al. (2009) realiza uma pesquisa em torno do problema de programação de exames, apresentando os principais avanços teóricos e práticos alcançados na última década. O principal objetivo desta pesquisa é destacar as novas tendências e delinear as questões mais importantes encontradas em trabalhos recentes.

4.2 Trabalhos correlacionados em Quadros de Horários

Alguns trabalhos correlacionados ao tema alvo desse projeto foram de grande relevância a esta pesquisa, ainda que esses tratem de diferentes variações e subvariações do PPHE ou instâncias com características próprias. Nesses trabalhos, as técnicas empregadas bem como as análises realizadas, servem como base teórica de possíveis aspectos futuros desse projeto.

Cooper e Kingston (1996) prova que a resolução do problema de alocação de horários apresenta complexidade NP-completo em circunstâncias comuns a classe de problemas. E também, discute sobre perspectivas de pesquisas para superar essa característica.

Souza (2000) trata o problema de programação de horários em escolas, através de várias meta-heurísticas, sendo utilizada uma heurística de busca local baseada em caminhos mínimos desenvolvida neste trabalho. O trabalho descreve também características e classificações do PPHE juntamente com alguns dos principais algoritmos heurísticos da literatura.

Em Qu (2002), é desenvolvido um conceito denominado CBR (raciocínio baseado em casos), para resolução de problemas de programação de cursos. O objetivo geral do trabalho é estudar como a técnica pode ajudar a resolver problemas relacionados através de um conhecimento prévio armazenado em uma base de dados.

Principalmente através de duas maneiras: através da reutilização de soluções de boa qualidade e na escolha de qual técnica heurística será utilizada em diferentes estágios de resolução. Sendo assim, testes são realizados e perspectivas futuras são apresentadas, dentre elas a possível criação de uma hiper-heurística com a utilização do CBR.

Willemen (2002) e Eikelder e Willemen (2001) apresentam uma abordagem ao *School Timetabling* para uma instância real de uma escola holandesa, que possui como diferencial a possibilidade de alunos de mesma turma cursarem algumas disciplinas opcionais. É realizado um estudo sobre a complexidade do problema estudado em relação aos problemas convencionais da classe. Para resolução do problema é desenvolvido uma técnica nomeada de busca em árvore, sendo os resultados computacionais comparados e analisados.

Já Muller (2005) desenvolve, descreve e verifica experimentalmente um algoritmo baseado em programação por restrições, denominado como Algoritmo de Busca Iterativa para Frente (iterative forward search algorithm - IFS), para resolução de problemas de programação de cursos. A fim de analisar a eficiência do algoritmo, o mesmo é aplicado com sucesso em um problema real de grande escala, sendo esse referente a Universidade de Purdue.

Chiarandini et al. (2006) propuseram um algoritmo híbrido para resolução das instâncias de *Course Timetabling* relacionadas a competição internacional de 2002 (ITC 2002). No algoritmo desenvolvido são combinadas várias heurísticas de construção, Busca Tabu, Descida em Vizinhaça Variável (*Variable Neighborhood Descent*, VND) e *Simulated Annealing*. São apresentados testes que validam os parâmetros adotados e também análises comparativas que verificam a eficiência do método, sendo a abordagem realizada superior aos demais resultados encontrados na literatura.

Santos (2007) trata o problema de programação de horários em escolas, a partir de três abordagens diferentes. Primeiramente, resolve o problema através de uma heurística híbrida, baseada em Busca Tabu. Na segunda abordagem o autor utiliza técnicas de Programação Linear Inteira Mista para a obtenção de quadro de horários provadamente ótimos. E em uma terceira abordagem, o autor explora a hibridização entre heurísticas e métodos exatos, encontrando através de uma técnica hibridizada os melhores limites superiores disponíveis.

Em Irene et al. (2009), é realizada uma abordagem ao *Course Timetabling* através de um algoritmo híbrido que combina a meta-heurística Enxame de Partículas (particle swarm optimization - PSO) com programação baseada em restrições. Essa abordagem, tem eficiência validada através de comparações com outros algoritmos híbridos propostos, que envolvem Algoritmo Genético e Busca Local. São utilizados dados reais com base na grade curricular da Universidade de Ciências de Yemen.

4.3 Trabalhos em Problemas de Programação de Cursos Universitários baseada em Currículos

São apresentados alguns trabalhos que realizam a abordagem à problemas de programação de cursos universitários baseada em currículos através de instâncias da competição internacional de 2007 (ITC 2007), assim como é realizado nesse projeto.

Gaspero e Schaerf (2006) propõem um estudo sobre uma nova forma de percorrer o espaço de soluções, denominado *Neighborhood Portfolio Approach*, que consiste na intercalação de técnicas de Busca Local com base em várias combinações de vizinhanças. A abordagem realizada, é baseada em um conjunto de operadores para a combinação de vizinhanças e de uma estratégia de resolução que intercala algoritmos básicos. Ressalta-se que esse trabalho apresenta como aspecto conclusivo a relevante eficiência alcançada através de combinações entre diferentes vizinhanças.

Muller (2008) apresenta uma abordagem híbrida para resolução do problema, que foi testada também nas outras categorias da competição. O algoritmo híbrido combina quatro técnicas diferentes sendo estas: o IFS (Muller, 2005); Great Deluge (Dueck, 1993); Método de descida combinado com recurso denominado CBS - *Conflict-based Statistics* (Muller et al., 2004), que qualifica a busca local; e *Simulated Annealing*. Ressalta-se que essa abordagem foi a vencedora do ITC 2007 para a categoria programação de cursos universitários baseada em currículos e nas demais obteve ótimos resultados.

Em Lü e Hao (2010), é proposto um algoritmo denominado Busca Tabu Adaptativa (Adaptive Tabu Search - ATS) para resolução do problema de programação de cursos universitários baseada em currículos. Destaca-se que a implementação realizada é uma hibridização que combina conceitos da meta-heurística ILS e Busca Tabu, proposta com intuito de garantir um equilíbrio entre diversificação e intensificação. Esta abordagem é dita adaptativa pelo fato de ter parâmetros que se ajustam com a instância e também com o comportamento da técnica ao decorrer da resolução. A técnica em questão, foi utilizada na competição internacional (ITC 2007) ficando em segundo lugar, como pode ser visto em Lu e Hao (2008). Contudo, a técnica foi aprimorada (Lü e Hao, 2010) encontrando os melhores resultados para um bom número de instâncias.

Em De Cescio et al. (2010) são propostas novas formulações para o problema de programação de cursos universitários baseada em currículos, com o objetivo de representar as principais variações encontradas para o problema. O trabalho em questão, é acompanhado de um endereço WEB que proporciona várias ações e informações referentes ao problema, que são de grande importância na análise de novas abordagens ao tema.

Geiger (2010) propõem um algoritmo meta-heurístico, denominado *Threshold Accepting Metaheuristic*, para resolução do problema. O algoritmo desenvolvido apresenta uma Busca Local baseada em um limiar de aceitação, que visa superar os ótimos locais através de aceitações determinísticas de soluções inferiores. Nesse trabalho uma nova vizinhança também é explorada, essa consiste em remover determinadas alocações, realizar uma Busca Local e reinserir as alocações removidas. Vários testes e comparações são realizados para validar os parâmetros atribuídos e a eficiência da técnica. Ressalta-se que uma versão da técnica em questão alcançou o quarto lugar no ITC 2007, como pode ser visto em Geiger (2008).

Em Lü et al. (2011), é apresentada uma análise aprofundada sobre a influência da utilização de diferentes vizinhanças em uma Busca Local. Por meio do problema em questão, é analisada a capacidade de busca de quatro tipos de vizinhança diferentes, através de três critérios de avaliação: percentual de melhora de vizinhos, força de melhoria e etapas da pesquisa. Após a realização de vários experimentos as vizinhanças envolvidas são avaliadas. Ressalta-se que esse trabalho apresenta o

procedimento Kempeswap como uma poderosa técnica de exploração de vizinhanças.

Hao e Benlic (2011) realiza uma abordagem para gerar limites inferiores (*lower bounds*) para o problema em questão. A técnica de resolução utiliza o algoritmo ATS (Lü e Hao, 2010) para subdividir o problema em problemas menores, os quais são solucionados a partir de uma ferramenta que utiliza programação linear inteira (COIN - OR Cbc 2.2.2). Os resultados obtidos foram capazes de melhorar os limites inferiores para 12 das 21 instâncias existentes e provar que o ótimo global foi encontrado em 6 dessas. Esses novos limites inferiores são úteis para estimar a qualidade dos limites superiores obtidos com abordagens diversas como heurísticas.

Capítulo 5

ALGORITMOS APLICADOS À RESOLUÇÃO DO PROBLEMA

Neste capítulo são apresentados os algoritmos e procedimentos desenvolvidos para resolução do problema abordado. Tratam-se de implementações híbridas que combinam características das meta-heurísticas ILS, Busca-Tabu e GRASP juntamente com os procedimentos denominados Kempeswap, Intra-salas e Inter-salas. Dessa forma, são apresentadas nas seções abaixo os principais aspectos de todas as abordagens realizadas juntamente com a descrição de cada algoritmo construído.

5.1 Representação da Solução

O problema apresenta: q disciplinas $d_1...d_q$, p períodos $1, ..., p$ e m salas $s_1...s_m$. Cada disciplina d_i consiste em l_i aulas, agendadas em períodos distintos e atendendo a_i alunos. Cada sala s_j apresenta uma capacidade cap_j em termos de número de lugares. O problema apresenta também g grupos de disciplinas, os quais são chamados currículos. As disciplinas de um currículo são agrupadas por apresentarem estudantes em comum e assim sendo não devem ser agendadas em um mesmo período.

A representação de uma solução (Figura 5.1) é dada por uma matriz T de inteiros $q \times p$, em que $T_{ik} = j$ (com $1 \leq j \leq m$) significa que a disciplina d_i é mostrada na sala r_j no período k e $T_{ik} = -1$ significa que a disciplina d_i não possui nenhuma aula alocada no período k .

SALAS / INDEX																									
Sala rC = 1					Sala rD = 2					Sala rF = 3					Sala rG = 4										
PERÍODO / DISCIPLINA	DIA 1					DIA 2					Dia 3					Dia 4					Dia 5				
	p1	p2	p3	p4	p5	p6	p7	p8	p9	p10	p11	p12	p13	p14	p15	p16	p17	p18	p19	p20	p21	p22	p23	p24	p25
AEDS I	-1	-1	2	-1	-1	2	2	-1	-1	-1	-1	2	-1	-1	-1	2	-1	-1	-1	-1	-1	-1	-1	-1	-1
AEDS II	3	3	-1	-1	-1	3	3	-1	-1	-1	4	4	-1	-1	-1	-1	-1	-1	-1	-1	4	4	-1	-1	-1
AEDS III	-1	-1	3	3	-1	-1	3	-1	-1	-1	3	-1	-1	-1	-1	-1	-1	3	-1	-1	-1	-1	3	3	-1
Algebra Linear	2	2	-1	-1	-1	-1	-1	2	2	-1	-1	-1	-1	2	2	-1	-1	-1	-1	-1	-1	-1	-1	-1	2
Cálculo I	1	1	-1	-1	-1	1	1	-1	-1	-1	-1	-1	-1	-1	-1	1	1	-1	-1	-1	-1	-1	1	1	-1

Figura 5.1: Representação da solução adotada.

Ou seja, uma solução é dada por uma matriz que relaciona disciplina por períodos. Esta matriz é preenchida com valores inteiros que representam em qual sala a disciplina será ministrada. Através desta representação, é possível obter a grade de horário das disciplinas, bem como o escalonamento de salas para as mesmas.

5.2 Vizinhança

Para exploração do espaço de soluções, vários tipos de movimentos foram testados, de forma que a escolha dos movimentos adotados reflete-se fortemente nos resultados obtidos. Após testes empíricos, foram adotados os seguintes movimentos:

- **MovimentoPeríodo** (Figura 5.2): Uma disciplina tem uma de suas aulas selecionada aleatoriamente (Algebra Linear, período 15), esta tem seu período alterado para um outro período disponível (período 18) selecionado também aleatoriamente em que não haja conflito com outras disciplinas do mesmo currículo (um exemplo de conflito ocorre no período 17).

Ex. Currículo
1º P. Sist de Inf. : AEDS I, Algebra, Cálculo I, ...

PERÍODO / DISCIPLINA	DIA 1					DIA 2					Dia 3					Dia 4					Dia 5				
	p1	p2	p3	p4	p5	p6	p7	p8	p9	p10	p11	p12	p13	p14	p15	p16	p17	p18	p19	p20	p21	p22	p23	p24	p25
AEDS I	-1	-1	2	-1	-1	2	2	-1	-1	-1	-1	1	-1	-1	-1	2	-1	-1	-1	-1	-1	-1	-1	-1	-1
AEDS II	3	3	-1	-1	-1	3	3	-1	-1	-1	4	4	-1	-1	-1	-1	-1	-1	-1	-1	4	4	-1	-1	-1
AEDS III	-1	-1	3	3	-1	-1	-1	-1	-1	-1	3	3	-1	-1	-1	-1	-1	3	-1	-1	-1	-1	3	3	-1
Algebra Linear	2	2	-1	-1	-1	-1	-1	2	2	-1	-1	-1	-1	2	2	-1	-1	-1	-1	-1	-1	-1	-1	-1	2
Cálculo I	-1	-1	-1	1	1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	1	1	-1	1	1	-1	-1	1	1	-1

PERÍODO / DISCIPLINA	DIA 1					DIA 2					Dia 3					Dia 4					Dia 5				
	p1	p2	p3	p4	p5	p6	p7	p8	p9	p10	p11	p12	p13	p14	p15	p16	p17	p18	p19	p20	p21	p22	p23	p24	p25
AEDS I	-1	-1	2	-1	-1	2	2	-1	-1	-1	-1	1	-1	-1	-1	2	-1	-1	-1	-1	-1	-1	-1	-1	-1
AEDS II	3	3	-1	-1	-1	3	3	-1	-1	-1	4	4	-1	-1	-1	-1	-1	-1	-1	-1	4	4	-1	-1	-1
AEDS III	-1	-1	3	3	-1	-1	-1	-1	-1	-1	3	3	-1	-1	-1	-1	-1	3	-1	-1	-1	-1	3	3	-1
Algebra Linear	2	2	-1	-1	-1	-1	-1	2	2	-1	-1	-1	-1	2	2	-1	-1	X	2	-1	-1	-1	-1	-1	2
Cálculo I	-1	-1	-1	1	1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	1	1	-1	1	1	-1	-1	1	1	-1

Figura 5.2: Representação MovimentoPeríodo.

- **MovimentoSala** (Figura 5.3): Uma disciplina (AEDS I) tem uma de suas aulas selecionada aleatoriamente (período 12), esta tem sua sala alterada (sala 2) para a primeira sala não conflitante, em termos de ocupação;

PERÍODO / DISCIPLINA	DIA 1					DIA 2					Dia 3					Dia 4					Dia 5				
	p1	p2	p3	p4	p5	p6	p7	p8	p9	p10	p11	p12	p13	p14	p15	p16	p17	p18	p19	p20	p21	p22	p23	p24	p25
AEDS I	-1	-1	2	-1	-1	2	2	-1	-1	-1	-1	1	-1	-1	-1	2	-1	-1	-1	-1	-1	-1	-1	-1	-1
AEDS II	3	3	-1	-1	-1	3	3	-1	-1	-1	4	4	-1	-1	-1	-1	-1	-1	-1	-1	4	4	-1	-1	-1
AEDS III	-1	-1	3	3	-1	-1	-1	-1	-1	-1	3	3	-1	-1	-1	-1	-1	3	-1	-1	-1	-1	3	3	-1
Algebra Linear	2	2	-1	-1	-1	-1	-1	2	2	-1	-1	-1	-1	2	2	-1	-1	-1	-1	-1	-1	-1	-1	-1	2
Cálculo I	-1	-1	-1	1	1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	1	1	-1	1	1	-1	-1	1	1	-1

PERÍODO / DISCIPLINA	DIA 1					DIA 2					Dia 3					Dia 4					Dia 5				
	p1	p2	p3	p4	p5	p6	p7	p8	p9	p10	p11	p12	p13	p14	p15	p16	p17	p18	p19	p20	p21	p22	p23	p24	p25
AEDS I	-1	-1	2	-1	-1	2	2	-1	-1	-1	-1	2	-1	-1	-1	2	-1	-1	-1	-1	-1	-1	-1	-1	-1
AEDS II	3	3	-1	-1	-1	3	3	-1	-1	-1	4	4	-1	-1	-1	-1	-1	-1	-1	-1	4	4	-1	-1	-1
AEDS III	-1	-1	3	3	-1	-1	-1	-1	-1	-1	3	3	-1	-1	-1	-1	-1	3	-1	-1	-1	-1	3	3	-1
Algebra Linear	2	2	-1	-1	-1	-1	-1	2	2	-1	-1	-1	-1	2	2	-1	-1	-1	-1	-1	-1	-1	-1	-1	2
Cálculo I	-1	-1	-1	1	1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	1	1	-1	1	1	-1	-1	1	1	-1

Figura 5.3: Representação MovimentoSala.

- **MovimentoAula** (Figura 5.4): Uma disciplina tem uma de suas aulas selecionada, um novo período para a aula é selecionado também e por fim uma nova sala é escolhida (AEDS 1, período 12 para período 2, sala 2). Todos estes são selecionados de forma aleatória. A aula tem seu período alterado e também uma nova sala é alocada. Caso não haja conflitos, apenas este movimento é realizado. Caso haja, a aula da disciplina conflitante tem uma troca de horários entre os horários envolvidos no movimento (aula do período 2 de Álgebra Linear é movimentada para o período 12);

Ex. Currículo
1º P. Sist de Inf.: AEDS 1, Álgebra, Cálculo I, ...

PERÍODO / DISCIPLINA	DIA 1					DIA 2					Dia 3					Dia 4					Dia 5				
	p1	p2	p3	p4	p5	p6	p7	p8	p9	p10	p11	p12	p13	p14	p15	p16	p17	p18	p19	p20	p21	p22	p23	p24	p25
AEDS I	-1	-1	2	-1	-1	2	2	-1	-1	-1	-1	1	-1	-1	-1	2	-1	-1	-1	-1	-1	-1	-1	-1	-1
AEDS II	3	3	-1	-1	-1	3	3	-1	-1	-1	4	4	-1	-1	-1	-1	-1	-1	-1	-1	4	4	-1	-1	-1
AEDS III	-1	-1	3	3	-1	-1	-1	-1	-1	-1	3	3	-1	-1	-1	-1	3	-1	-1	-1	-1	3	3	-1	-1
Álgebra Linear	2	2	-1	-1	-1	-1	-1	2	2	-1	-1	-1	-1	2	2	-1	-1	-1	-1	-1	-1	-1	-1	-1	2
Cálculo I	-1	-1	-1	1	1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	1	1	-1	1	1	-1	-1	1	1	-1

PERÍODO / DISCIPLINA	DIA 1					DIA 2					Dia 3					Dia 4					Dia 5				
	p1	p2	p3	p4	p5	p6	p7	p8	p9	p10	p11	p12	p13	p14	p15	p16	p17	p18	p19	p20	p21	p22	p23	p24	p25
AEDS I	-1	2	2	-1	-1	2	2	-1	-1	-1	-1	-1	-1	-1	-1	2	-1	-1	-1	-1	-1	-1	-1	-1	-1
AEDS II	3	3	-1	-1	-1	3	3	-1	-1	-1	4	4	-1	-1	-1	-1	-1	-1	-1	-1	4	4	-1	-1	-1
AEDS III	-1	-1	3	3	-1	-1	-1	-1	-1	-1	3	3	-1	-1	-1	-1	3	-1	-1	-1	-1	3	3	-1	-1
Álgebra Linear	2	-1	-1	-1	-1	-1	-1	2	2	-1	-1	2	-1	2	2	-1	-1	-1	-1	-1	-1	-1	-1	-1	2
Cálculo I	-1	-1	-1	1	1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	1	1	-1	1	1	-1	-1	1	1	-1

Figura 5.4: Representação MovimentoAula.

- **MovimentoEstabilidadeSala** (Figura 5.5): Este movimento é utilizado no intuito de diminuir o número de violações a respeito da restrição estabilidade de salas. O movimento consiste em escolher uma nova sala aleatoriamente e atribuir a mesma para todos horários de uma determinada disciplina (AEDS III, da sala 3 para sala 4). Caso haja conflitos de sala após a alocação, as salas das aulas das disciplinas conflitantes são trocadas pelas salas que estavam na disciplina que recebeu o movimento (AEDS II, período 11 e 12, para sala 3);

PERÍODO / DISCIPLINA	DIA 1					DIA 2					Dia 3					Dia 4					Dia 5				
	p1	p2	p3	p4	p5	p6	p7	p8	p9	p10	p11	p12	p13	p14	p15	p16	p17	p18	p19	p20	p21	p22	p23	p24	p25
AEDS I	-1	-1	2	-1	-1	2	2	-1	-1	-1	-1	1	-1	-1	-1	2	-1	-1	-1	-1	-1	-1	-1	-1	-1
AEDS II	3	3	-1	-1	-1	3	3	-1	-1	-1	4	4	-1	-1	-1	-1	-1	-1	-1	-1	4	4	-1	-1	-1
AEDS III	-1	-1	3	3	-1	-1	-1	-1	-1	-1	3	3	-1	-1	-1	-1	3	-1	-1	-1	-1	3	3	-1	-1
Álgebra Linear	2	2	-1	-1	-1	-1	-1	2	2	-1	-1	-1	-1	2	2	-1	-1	-1	-1	-1	-1	-1	-1	-1	2
Cálculo I	-1	-1	-1	1	1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	1	1	-1	1	1	-1	-1	1	1	-1

PERÍODO / DISCIPLINA	DIA 1					DIA 2					Dia 3					Dia 4					Dia 5				
	p1	p2	p3	p4	p5	p6	p7	p8	p9	p10	p11	p12	p13	p14	p15	p16	p17	p18	p19	p20	p21	p22	p23	p24	p25
AEDS I	-1	-1	2	-1	-1	2	2	-1	-1	-1	-1	1	-1	-1	-1	2	-1	-1	-1	-1	-1	-1	-1	-1	-1
AEDS II	3	3	-1	-1	-1	3	3	-1	-1	-1	3	3	-1	-1	-1	-1	-1	-1	-1	-1	4	4	-1	-1	-1
AEDS III	-1	-1	4	4	-1	-1	-1	-1	-1	-1	4	4	-1	-1	-1	-1	4	-1	-1	-1	-1	4	4	-1	-1
Álgebra Linear	2	2	-1	-1	-1	-1	-1	2	2	-1	-1	-1	-1	2	2	-1	-1	-1	-1	-1	-1	-1	-1	-1	2
Cálculo I	-1	-1	-1	1	1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	1	1	-1	1	1	-1	-1	1	1	-1

Figura 5.5: Representação MovimentoEstabilidadeSala.

- **MovimentoCurriculoCompacto** (Figura 5.6): Este movimento é utilizado no intuito de diminuir o número de violações a respeito da restrição aulas

isoladas. Uma aula isolada (ou seja, que não possui aulas do mesmo currículo alocadas de forma adjacente) é escolhida aleatoriamente (AEDS 1, período 12) e a mesma é alocada em um horário adjacente (período 18) a outra aula qualquer do currículo;

Ex. Currículo
1º P. Sist. de Inf. : AEDS 1, Álgebra, Cálculo I, ...

PERÍODO / DISCIPLINA	DIA 1					DIA 2					DIA 3					DIA 4					DIA 5				
	p1	p2	p3	p4	p5	p6	p7	p8	p9	p10	p11	p12	p13	p14	p15	p16	p17	p18	p19	p20	p21	p22	p23	p24	p25
AEDS I	-1	-1	2	-1	-1	2	2	-1	-1	-1	-1	1	-1	-1	-1	2	-1	-1	-1	-1	-1	-1	-1	-1	-1
AEDS II	3	3	-1	-1	-1	3	3	-1	-1	-1	4	4	-1	-1	-1	-1	-1	-1	-1	-1	4	4	-1	-1	-1
AEDS III	-1	-1	3	3	-1	-1	-1	-1	-1	-1	3	3	-1	-1	-1	-1	3	-1	-1	-1	-1	3	3	-1	-1
Álgebra Linear	2	2	-1	-1	-1	-1	-1	2	2	-1	-1	-1	-1	2	2	-1	-1	-1	-1	-1	-1	-1	-1	-1	2
Cálculo I	-1	-1	-1	1	1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	1	1	-1	1	1	-1	-1	1	1	-1

PERÍODO / Currículo	DIA 1					DIA 2					DIA 3					DIA 4					DIA 5				
	p1	p2	p3	p4	p5	p6	p7	p8	p9	p10	p11	p12	p13	p14	p15	p16	p17	p18	p19	p20	p21	p22	p23	p24	p25
1º P. de Sist. de Inf.	Alge L	Alge L	AEDS1	Cálc1	Cálc1	AEDS1	AEDS1	Alge L	Alge L	-1	-1	AEDS1	-1	Alge L	Alge L	Cálc1	Cálc1	-1	Cálc1	Cálc1	-1	-1	Cálc1	Cálc1	Alge L

PERÍODO / DISCIPLINA	DIA 1					DIA 2					DIA 3					DIA 4					DIA 5				
	p1	p2	p3	p4	p5	p6	p7	p8	p9	p10	p11	p12	p13	p14	p15	p16	p17	p18	p19	p20	p21	p22	p23	p24	p25
AEDS I	-1	-1	2	-1	-1	2	2	-1	-1	-1	-1	1	-1	-1	-1	2	-1	1	-1	-1	-1	-1	-1	-1	-1
AEDS II	3	3	-1	-1	-1	3	3	-1	-1	-1	4	4	-1	-1	-1	-1	-1	-1	-1	-1	4	4	-1	-1	-1
AEDS III	-1	-1	3	3	-1	-1	-1	-1	-1	-1	3	3	-1	-1	-1	-1	3	-1	-1	-1	-1	3	3	-1	-1
Álgebra Linear	2	2	-1	-1	-1	-1	-1	2	2	-1	-1	-1	-1	2	2	-1	-1	-1	-1	-1	-1	-1	-1	-1	2
Cálculo I	-1	-1	-1	1	1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	1	1	-1	1	1	-1	-1	1	1	-1



Figura 5.6: Representação MovimentoTrocaDeAulasESalas.

- **MovimentoTrocaDeAulasESalas** (Figura 5.7): Neste movimento são selecionadas duas disciplinas que utilizam a mesma sala em horários diferentes (AEDS 1 e Álgebra Linear, nos períodos 3 e 25 respectivamente). Após escolher as disciplinas envolvidas as mesmas tem seus horários alternados mantendo a alocação para a sala. Este movimento é uma contribuição deste trabalho à resolução do PPHE.

PERÍODO / DISCIPLINA	DIA 1					DIA 2					DIA 3					DIA 4					DIA 5				
	p1	p2	p3	p4	p5	p6	p7	p8	p9	p10	p11	p12	p13	p14	p15	p16	p17	p18	p19	p20	p21	p22	p23	p24	p25
AEDS I	-1	-1	2	-1	-1	2	2	-1	-1	-1	-1	1	-1	-1	-1	2	-1	-1	-1	-1	-1	-1	-1	-1	-1
AEDS II	3	3	-1	-1	-1	3	3	-1	-1	-1	4	4	-1	-1	-1	-1	-1	-1	-1	-1	4	4	-1	-1	-1
AEDS III	-1	-1	3	3	-1	-1	-1	-1	-1	-1	3	3	-1	-1	-1	-1	3	-1	-1	-1	-1	3	3	-1	-1
Álgebra Linear	2	2	-1	-1	-1	-1	-1	2	2	-1	-1	-1	-1	2	2	-1	-1	-1	-1	-1	-1	-1	-1	-1	2
Cálculo I	-1	-1	-1	1	1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	1	1	-1	1	1	-1	-1	1	1	-1

PERÍODO / DISCIPLINA	DIA 1					DIA 2					DIA 3					DIA 4					DIA 5				
	p1	p2	p3	p4	p5	p6	p7	p8	p9	p10	p11	p12	p13	p14	p15	p16	p17	p18	p19	p20	p21	p22	p23	p24	p25
AEDS I	-1	-1	-1	-1	-1	2	2	-1	-1	-1	-1	1	-1	-1	-1	2	-1	-1	-1	-1	-1	-1	-1	-1	2
AEDS II	3	3	-1	-1	-1	3	3	-1	-1	-1	4	4	-1	-1	-1	-1	-1	-1	-1	-1	4	4	-1	-1	-1
AEDS III	-1	-1	3	3	-1	-1	-1	-1	-1	-1	3	3	-1	-1	-1	-1	3	-1	-1	-1	-1	3	3	-1	-1
Álgebra Linear	2	2	2	-1	-1	-1	-1	2	2	-1	-1	-1	-1	2	2	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1
Cálculo I	-1	-1	-1	1	1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	1	1	-1	1	1	-1	-1	1	1	-1



Figura 5.7: Representação MovimentoTrocaDeAulasESalas.

Os movimentos utilizados apresentam como característica, determinadas verificações para algumas restrições, visando evitar a geração de soluções ineficazes e agilizando o processo de busca. Ressalta-se também que alguns movimentos são específicos para diminuir a ocorrência de violações de determinadas restrições, medida que apresentou grande importância nos resultados alcançados.

5.3 Função de Avaliação

A função de avaliação conceitua um quadro de horários analisando a qualidade e à viabilidade de uma solução encontrada. Esta análise é feita de modo a verificar as violações das restrições estabelecidas, restrições estas que são divididas em fortes e fracas, conforme apresentado no capítulo 2 deste trabalho.

O valor dado a violação de uma restrição está diretamente ligado a relevância desta no problema. Dessa forma, as restrições fortes possuem um valor de peso alto para que sejam prioritárias. Nesta implementação as mesmas receberam peso igual à 1000. Já as restrições fracas recebem pesos menores, que se diferenciam de acordo com as prioridades da instituição. Nesta implementação as restrições *R05* e *R08* receberam peso 1, a restrição *R06* peso 5 e a restrição *R07* peso igual a 2. As definições dos pesos também são apresentadas no trabalho de De Cesco et al. (2010).

Sejam: V o conjunto de restrições do problema com elementos $v \in V$ identificados por $v = 1, \dots, \bar{v}$; $f_v(X)$ função de avaliação da restrição R_v ; e w_v o peso da restrição de avaliação $f_v(X)$. A função avaliação, é dada pela equação(5.1) formulada como um problema de otimização, onde o objetivo é minimizar o custo da solução encontrada. A equação que representa a avaliação da função é dada a seguir:

$$\text{Minimizar} \sum_{v=1}^{\bar{v}} [w_v \times f_v(X)] \quad (5.1)$$

A avaliação das soluções é um aspecto crítico na resolução do problema, pelo alto tempo de processamento consumido por estas no processo. Sendo assim, foi utilizada nessa abordagem o conceito de avaliação delta (Burke e Newall, 1999), no qual uma avaliação é realizada a partir da diferença entre a solução corrente e a solução vizinha, esta técnica reduziu consideravelmente o tempo computacional envolvido na resolução.

5.4 Busca Local

Sendo o espaço de busca para o problema muito grande devido a quantidade de combinações e movimentos possíveis na exploração do espaço, a busca local adotada foi baseada no método randômico de descida descrito em Souza (2000). Trata-se de um método que não necessariamente pesquisa toda vizinhança de uma solução - como ocorre no método de descida - dessa forma, evita-se a pesquisa exaustiva que neste problema demandaria um alto tempo de processamento.

O método implementado consiste em analisar um conjunto de vizinhos gerados a partir de movimentos aleatórios, em que o melhor vizinho encontrado é comparado com a solução corrente. A solução vizinha somente será a nova solução corrente caso seja melhor ou igual a anterior. Para geração da vizinhança a cada iteração todos os movimentos descritos na seção 5.2 tem a mesma probabilidade de ocorrerem. O procedimento é repetido por um número fixo de iterações, destacando-se que devido ao vasto espaço de busca a cada iteração são gerados vizinhos de somente uma disciplina.

A disciplina é escolhida a cada iteração de forma aleatória em uma estrutura de dados do tipo lista. Após ser escolhida, a mesma é removida para que todas as disciplinas passem pelo processo de busca local com a mesma frequência. Ressalta-se que quando a lista de disciplinas ficar vazia é preenchida novamente com todas as disciplinas em ordem aleatória, até que o critério de parada seja alcançado. Para esta implementação foi utilizado como critério de parada um número máximo de iterações.

No Algoritmo 6 é apresentado o pseudo-código da busca local utilizada:

Algoritmo 6: Descida Randômica por Disciplina

Entrada: *SolucaoCorrente*, *NumVizinhos*, *MaxIteracoes*
Saída: *SolucaoV*

```

1 início
2   Iteracao  $\leftarrow \emptyset$ ; SAux  $\leftarrow \emptyset$ ; SolucaoV  $\leftarrow$  SolucaoCorrente
3   enquanto (Iteracao  $\leq$  MaxIteracoes) faça
4     Iteracao  $\leftarrow$  Iteracao + 1;
5     Seleciona-se uma disciplina na lista de disciplinas disponíveis para geração da vizinhança;
6     SAux  $\leftarrow$  Melhor vizinho entre os NumVizinhos gerados a partir de SolucaoVizinha; se
7       SAux  $\leq$  SolucaoV então
8         SolucaoV = SAux;
9     fim
10  fim

```

5.5 Solução Inicial: Algoritmo *ConstroiGRASP*

A solução inicial é gerada através de um algoritmo desenvolvido, o qual foi baseado na fase de construção da meta-heurística GRASP (Feo e Resende, 1995). O GRASP é uma meta-heurística que consiste na aplicação iterativa de duas fases: uma fase de construção da solução, via um procedimento guloso-aleatório (lista restrita de candidatos e parâmetro α e outra fase, de busca local, na qual um ótimo local da vizinhança da solução construída é pesquisado. A principal diferença entre o algoritmo desenvolvido e o GRASP, está no fato de que no primeiro apenas uma solução é criada, diferentemente do segundo caso em que várias soluções são criadas e refinadas.

O procedimento implementado neste trabalho realiza as alocações de todas as aulas de cada disciplina uma a uma. A alocação é feita a partir de cada disciplina, de modo que todas as aulas de uma disciplina devem primeiramente ser alocadas para somente depois realizar as alocações da próxima disciplina. Desta forma, a ordem da lista de disciplinas candidatas (LDC) é de suma importância no procedimento. O ordenamento das disciplinas é determinado por uma função que considera:

- o número de currículos que a disciplina está envolvida. Quanto maior o número de currículos em que uma disciplina está envolvida, maior a chance desta apresentar conflitos;
- o número de indisponibilidades de horários da disciplina;
- a quantidade de alunos na disciplina. Uma vez que disciplinas que possuam um alto número de alunos provavelmente terão poucas salas capacitadas.

As alocações são realizadas para todas as disciplinas, respeitando-se o ordenamento da LDC. Para cada disciplina, tem-se o número de aulas a serem alocadas, sendo a alocação feita de forma a testar todas as possíveis salas em todos os possíveis horários. Funções especiais de avaliação (que consideram as mesmas restrições do problema) foram criadas para testar as possíveis alocações. Após isso, uma lista de possíveis alocações candidatas (LPAC) é criada de forma ordenada, de acordo com o valor de avaliação de cada possível alocação.

A partir da LPAC, é criada uma Lista Restrita de Possíveis Alocações Candidatas (LRPAC), sendo o número de elementos dessa última lista definida por um parâmetro α , que varia no intervalo $[0, q]$, sendo q o número máximo de disciplinas a serem alocadas. A LRPAC é composta pelas α melhores possíveis alocações. A seguir, seleciona-se aleatoriamente uma alocação na LRPAC e atribui-se esta alocação à solução que está sendo construída. Como a alocação é escolhida de maneira aleatória, a cada execução uma solução diferente é criada. O procedimento continua até que todas as aulas de todas as disciplinas sejam alocadas.

No Algoritmo 7 é apresentado o pseudo-código do método denominado *ConstroiGrasp*.

Algoritmo 7: ConstroiGRASP

Entrada: α , *TotalAulas*,
Saída: *Solucao*

```

1 início
2   aulasAlocadas  $\leftarrow$  0;
3   Solucao  $\leftarrow$  inicia sem nenhuma aula alocada;
4   LDR  $\leftarrow$  Disciplinas ordenadas pela probabilidade de violação;
5   enquanto (aulasAlocadas < TotalAulas) faça
6     disciplinaEscolhida  $\leftarrow$  primeira posição da LDR;
7     para cada aula da disciplina escolhida faça
8       LPAC  $\leftarrow$  todas as possíveis alocações (sala/período);
9       LPACR  $\leftarrow$  as  $\alpha$  melhores possíveis alocações de LPAC;
10      alocaoEscolhida  $\leftarrow$  alocação escolhida aleatoriamente em LPACR;
11      Solucao  $\leftarrow$  Solucao recebe alocaoEscolhida;
12      aulasAlocadas  $\leftarrow$  aulasAlocadas + 1;
13    fim
14    LDR  $\leftarrow$  LDR - primeira posição
15  fim
16  Retorne Solucao
17 fim
```

5.6 ILS

A meta-heurística ILS (*Iterated Local Search*), segundo o próprio criador Lourenço et al. (2003), é um método de busca iterativa que faz uso de perturbações de solução (alterações na solução corrente). Tem-se como principal objetivo a diversificação da busca, escapando e visitando diferentes ótimos locais.

Quatro são os principais componentes que definem método ILS: geração da solução inicial, busca local, perturbação e critério aceitação. Primeiramente, a geração da solução inicial foi realizada através do algoritmo *ConstroiGrasp*. Já a busca local é realizada através da descida randômica por disciplina, conforme foi descrito na subseção anterior.

A perturbação é realizada de duas maneiras diferentes em dois níveis de intensidade. Os níveis determinam apenas a quantidade de perturbações aplicadas à solução

corrente, sendo nível um igual a duas perturbações e nível dois igual a três perturbações. Quanto às formas de perturbação, na primeira são utilizados os movimentos de geração de vizinhança seguidamente, de forma aleatória, sem garantia de melhora (por se tratar de uma perturbação). Já na segunda forma, um novo procedimento criado é utilizado para perturbar uma solução corrente.

O procedimento criado para a perturbação é denominado *constroiGRASPDisciplina*. Consiste em desalocar todas as aulas de determinada disciplina e alocar novamente utilizando parte do algoritmo de construção de solução inicial para reconstrução da solução. Este movimento apresentou grande importância, pois se trata de uma perturbação forte que de certa forma mantém a qualidade da solução. Ressalta-se que este movimento apresenta uma probabilidade menor de acontecer por apresentar um alto custo computacional.

Quanto ao critério de aceitação, foi utilizada uma condição para aceitar também soluções de piora. Através da utilização de uma variável que delimita o nível de piora juntamente com a avaliação da atual solução corrente, tem-se uma abordagem que, em determinados momentos, facilitará o critério de aceitação permitindo que atribuições de novas soluções correntes sejam realizadas ainda que sejam soluções piores. A esta técnica criada, foi dado o nome de relaxação.

A estrutura bem como o funcionamento do ILS implementado juntamente com a técnica de relaxação poderão ser observados e entendidos melhor nos pseudo-códigos apresentados ao decorrer desse trabalho.

5.7 Busca Tabu

A Busca Tabu - BT (Glover e Laguna, 1997) é uma meta-heurística de trajetória que explora o espaço de soluções se movendo de solução em solução buscando a cada iteração o melhor vizinho. Tem como grande característica a utilização de uma memória flexível para armazenar as informações referentes a determinadas regiões já pesquisadas. A essa memória é dado o nome de Lista Tabu, que tem como função armazenar soluções - ou características destas - que serão evitadas durante um certo número de iterações da busca, a menos que uma determinada função de aspiração seja atendida. A estratégia adotada na BT pode permitir que a busca escape de ótimos locais e eventualmente encontre um ótimo global.

Na BT implementada neste trabalho, a exploração do espaço de soluções foi realizada através dos movimentos já apresentados na seção 5.2. A cada iteração um número de soluções vizinhas de cada disciplina são geradas aleatoriamente a partir das vizinhanças descritas, aquele vizinho que apresentar o melhor valor de avaliação e não for Tabu ou atender a função aspiração, passa a ser a nova solução corrente.

A Lista Tabu desenvolvida é uma matriz com coordenadas referentes a sala, disciplina e período (matriz tridimensional), na qual, a cada nova solução corrente encontrada é atribuído o status de Tabu para as coordenadas dos movimentos que geraram a respectiva solução. Um movimento que gera uma nova solução corrente tem sua respectiva posição na matriz preenchida com o número da iteração que deixará de ser tabu, ou seja, toda posição que está preenchida com um valor maior que o número da iteração corrente recebe o status de Tabu. Pode-se dizer, que essa implementação trata-se de uma implementação eficiente da Lista Tabu, pois, a

complexidade da verificação se um movimento é Tabu ou não, é igual a $O(1)$.

O tamanho da Lista Tabu é de suma importância, pois, representa o número de iterações que um movimento deve permanecer como Tabu. Na abordagem realizada, como várias instâncias de diferentes tamanhos foram testadas, esse parâmetro é ajustado dinamicamente de acordo com o tamanho da instância. Esta abordagem foi realizada prezando a eficiência da técnica sobre as diferentes execuções.

Quanto a função de aceitação foi adotado aspiração por objetivo global, o qual consiste em retirar o status Tabu de um movimento se for produzido uma solução com a melhor avaliação global ainda não alcançada.

5.8 O Procedimento *Kempswap*

O procedimento desenvolvido tem como base o movimento de troca de horários entre duas cadeias Kempe denominado *Kempeswap* (Lü e Hao, 2010). Tendo em vista as disciplinas e os conflitos de currículo, cada solução para o problema CBCT pode ser representada por um grafo G onde os nós são as aulas de cada disciplina e as arestas interligam aulas de um mesmo currículo ou ministradas por um mesmo professor. Uma cadeia Kempe é um conjunto de nós interligados que compõem um subgrafo de G que é obtido a partir de dois horários específicos.

Formalmente, sendo K_1 e K_2 duas cadeias Kempe no subgrafo gerado a partir dos horários h_i e h_j , o movimento *Kempeswap* produz uma solução substituindo h_i com $(h_i \setminus (K_1 \cup K_2)) \cup (h_j \cap (K_1 \cup K_2))$ e h_j com $(h_j \setminus (K_1 \cup K_2)) \cup (h_i \cap (K_1 \cup K_2))$. Por definição pelo menos três disciplinas devem estar envolvidas no movimento, sendo $|K_1| + |K_2| \geq 3$. A Figura 5.8(a), apresenta um subgrafo gerado a partir de dois períodos h_i e h_j , neste são encontradas cinco cadeias Kempe: $K_a = \{D_1, D_2, D_7, D_8\}$, $K_b = \{D_3, D_6, D_9\}$, $K_c = \{D_4, D_{11}, D_{12}\}$, $K_d = \{D_5\}$ e $K_e = \{D_{10}\}$. Um exemplo de *Kempeswap* no subgrafo da figura seria entre K_b e K_c , produzindo uma nova atribuição movendo $\{D_3, D_4, D_6\}$ para h_j e $\{D_9, D_{11}, D_{12}\}$ para h_i (representado na Figura(b)).

Após a realização do movimento entre as aulas das cadeias Kempe selecionadas, é necessário um procedimento para alocação de salas (que ocorre apenas nas aulas movimentadas). Nos trabalhos relacionados pesquisados foram utilizados métodos exatos para esta atribuição, já na implementação deste trabalho é utilizada parte da implementação do método de geração de solução inicial(ControIGRASP) para testar todas as possíveis salas a cada aula, realizando alocações que respeitem ao máximo as restrições do problema.

A troca de períodos entre duplas cadeias Kempe é considerada uma generalização do movimento simples Kempe, que consiste de uma troca de períodos envolvendo apenas uma cadeia que apresente mais de três disciplinas. A utilização do movimento simples Kempe pode ser vista em Chiarandini et al. (2006) e Thompson e Dowsland (1998). Já a utilização do movimento duplo em cadeias Kempe denominado *Kempeswap* é apresentado em Lü e Hao (2010) e Lü et al. (2011), sendo no segundo trabalho apresentada uma comparação entre os movimentos Kempe juntamente com uma análise dos mesmos que comprova a eficiência do movimento duplo.

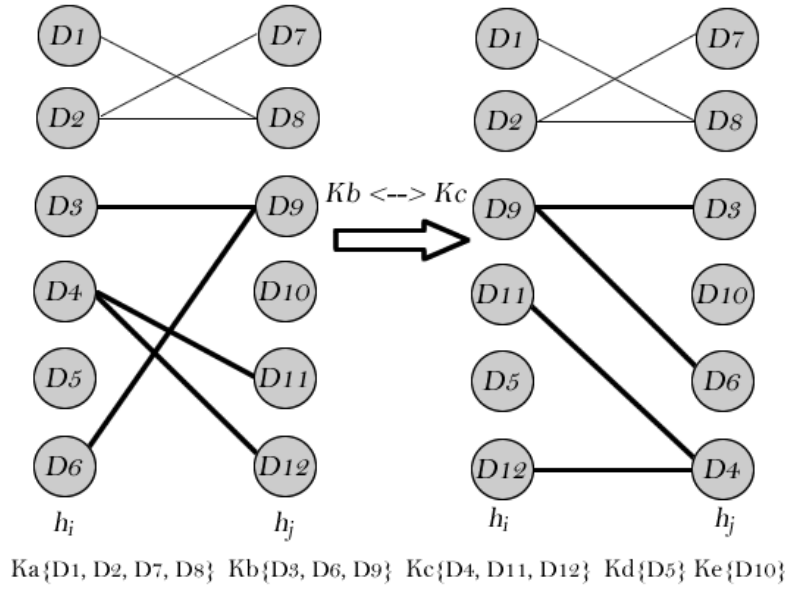


Figura 5.8: Movimento Kempeswap.

Salienta-se que o movimento *Kempeswap* pode gerar novas soluções modificando várias aulas de diferentes disciplinas, mas mantendo de certa forma a qualidade da solução. Dentre os principais aspectos que garantem a afirmação anterior temos o fato de realizar várias alterações de aulas em um mesmo movimento, mas respeitando determinadas restrições a partir dos grafos construídos e também com uma realocação de salas que considera a avaliação de cada alocação realizada.

O Procedimento *Kempeswap* desenvolvido, é uma descida randômica que faz uso somente de movimentos *Kempeswap*. A cada iteração são escolhidos dois horários de forma aleatória e o movimento é aplicado nas cadeias kempe encontradas. São realizados apenas trinta por cento das combinações possíveis entre os diferentes horários que envolvem o problema. Esta medida foi necessária devido ao alto custo computacional que o movimento possui para construção e análise dos grafos envolvidos, o que justifica a abordagem através de uma descida randômica e não de uma descida completa.

5.9 O Procedimentos Intra-Inter-Salas

O Procedimento Intra-Inter-Salas é um procedimento baseado em caminhos mínimos encontrados em grafos que são gerados a partir de uma solução corrente. A abordagem realizada trata-se de uma implementação inspirada no trabalho de Souza (2000) em que um procedimento que encontra caminhos mínimos entre aulas de uma mesma turma é aplicado a um problema de *School Timetabling*.

A implementação realizada pode ser dividida em duas partes distintas: i) grafos são gerados a partir das aulas alocadas para uma determinada sala, nos quais movimentos de transferência de aula são testados em busca de ciclos de custo negativo que indicam uma possível melhora na solução corrente - esta parte do procedimento é denominada Intrasalas; ii) a partir dos grafos gerados na etapa anterior são veri-

ficadas possíveis trocas de horários entre aulas de diferentes salas, possibilitando a execução de um movimento de melhora indicado pelo grafo criado - esta parte do procedimento é denominada Intersalas).

Sendo assim, serão apresentados os “movimentos”: Intrasalas e Intersalas, e também, como o procedimento desenvolvido aplicou esses conceitos no refinamento de soluções.

5.9.1 Intrasalas

Com intuito de exemplificar o funcionamento desta parte do procedimento, considera-se uma solução corrente na qual é possível definir um grafo $G_r(V_r, A_r)$ com todas as aulas de uma mesma sala r , onde V_r é o conjunto dos horários alocados para a sala r e A_r é o conjunto dos arcos orientados definidos conforme a seguir:

$A_r = (k, \bar{k})$: a aula da sala r lecionada no horário k pode ser lecionada também no horário $\bar{k}$;

A cada arco $(k, \bar{k}) \in G_r$ associamos um custo $\Delta f_i(k, \bar{k})$, o qual representa a variação do custo de se transferir o professor p do horário k para o horário $\bar{k}$. Desta forma, o custo é obtido calculando-se a diferença entre os valores da função objetivo, relativa ao professor, antes e depois da execução do possível movimento, isto é:

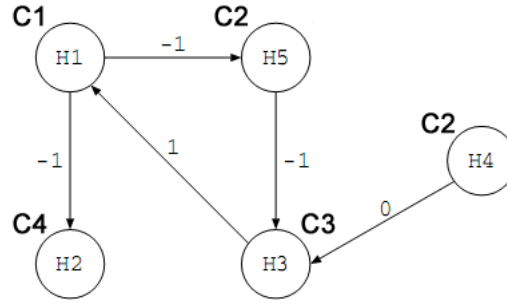
$$\Delta f_i(k, \bar{k}) = f_i(\bar{k}) - f_i(k) \quad (5.2)$$

A Tabela 5.1 mostra um fragmento de uma solução Q_1 , onde cada linha i representa uma disciplina (*course*) ($i = C1, C2, C3, C4$) e cada coluna k um horário ($k = H1, H2, H3, H4, H5$) de um mesmo dia. Cada elemento q_{ik} desta tabela representa uma aula alocada para disciplina, bem como a sala em que a aula ocorrerá. Sendo Ra e Rb as possíveis salas para alocação. Um traço (–) indica que o professor que leciona a disciplina não está disponível em um determinado horário, enquanto uma célula vazia representa a ausência de aulas alocadas. A função objetivo desta solução Q_1 sob a perspectiva de cada disciplina é dada por $F(Q_1) = F(C1) + F(C2) + F(C3) + F(C4) = 0 + 1 + 1 + 0$.

Tabela 5.1: Parte da solução Q_1

	H1	H2	H3	H4	H5	Fi
C1	Ra		Rb	Rb		1
C2	Rb	Rb		Ra	Ra	1
C3		Rb	Ra	Rb	Rb	0
C4	Rb	Ra	Rb	Rb	-	0

A partir do fragmento de solução apresentado, é possível definir o grafo G_{Ra} que representa as aulas alocadas na sala Ra , o qual é representado através da Figura 5.9. Cada horário é representado por um vértice, o qual está associada uma disciplina. O arco $(H1, H5)$ de custo -1, está indicando que caso a aula da disciplina $C1$ seja alterada do horário $H1$ para o horário $H5$, haverá uma diminuição da função objetivo em 1 unidade (os valores das arestas são determinados de acordo com $\Delta f_i()$).

Figura 5.9: G_{Ra} , grafo da sala Ra

Para encontrar uma solução de melhora que represente a diminuição da função objetivo corrente do problema, é suficiente procurar um ciclo de custo negativo no grafo G_{Ra} . A existência de um ciclo garante que todas as aulas alocadas nos horários correspondentes aos vértices afetados serão realocadas, garantindo que a maior parte das restrições do problema serão respeitadas. Para encontrar os caminhos mínimos nos grafos obtidos, foi utilizado o algoritmo de Floyd-Warshall (Floyd, 1962), no qual foi realizada uma pequena alteração, para que possa trabalhar com ciclos negativos. Mais detalhes sobre este algoritmo podem ser vistos em Cormen et al. (1990).

No exemplo apresentado, a sequência de arcos $(H1, H5)$, $(H5, H3)$, $(H3, H1)$ forma um ciclo de custo total $-1 (= -1 + (-1) + 1)$. Percebe-se que várias aulas de diferentes disciplinas serão movimentadas, contudo, todas estão alocadas em uma mesma sala, o que justifica o nome *Intrasalas* dado a essa parte do procedimento.

A Tabela 5.2 representa a nova solução Q'_1 , obtida após aplicação dos movimentos do ciclo de custo negativo representado na Figura 5.9. A função objetivo da nova solução Q'_1 é dada por $F(Q'_1) = F(C1) + F(C2) + F(C3) + F(C4) = 0 + 0 + 0 + 1$.

Tabela 5.2: Solução $Q1$

	H1	H2	H3	H4	H5	Fi
C1			Rb	Rb	Ra	0
C2	Rb	Rb	Ra	Ra		0
C3	Ra	Rb		Rb	Rb	1
C4	Rb	Ra	Rb	Rb	-	0

Após a atualização da solução a partir dos movimentos indicados em um ciclo negativo encontrado, procura-se pela existência de novos ciclos negativos. É construído um novo grafo G'_{Ra} a partir da solução obtida Q'_1 , o qual é representado pela Figura 5.10. Como pode ser visto, para o exemplo considerado não existem mais ciclos de custo negativo no grafo G'_{Ra} , o que representa o fim deste tipo de busca para esta sala. No entanto, esta análise demonstrada foi realizada apenas na sala Ra , sendo possível, para a mesma solução, a análise das outras salas alocadas.

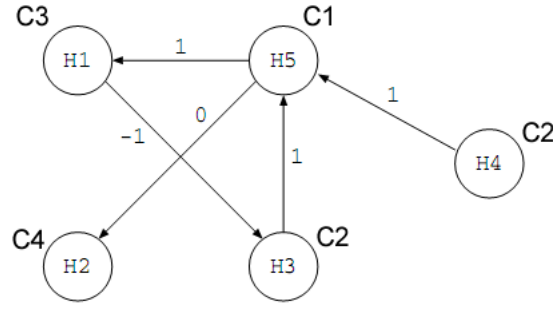


Figura 5.10: G'_{Ra} , grafo da sala Ra após 1ª execução do intrasalas

Um aperfeiçoamento foi desenvolvido no Intrasalas referente a adição de arestas de valor zero em horários que não possuem nenhuma alocação de aula para horários que possuem aulas alocadas. Essa modificação aumentou o poder de busca da técnica, possibilitando movimentos antes não possíveis. Um exemplo pode ser dado através do grafo G''_{Ra} , no qual um novo horário $H6$ sem nenhuma alocação é considerado. O Grafo em questão é representado pela Figura 5.11

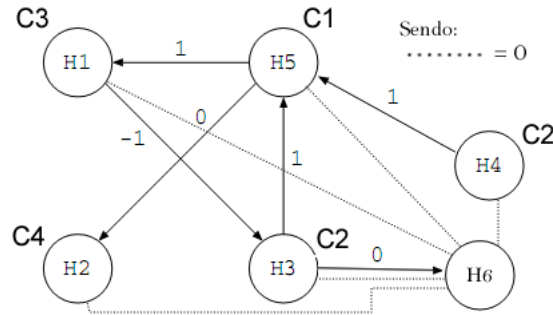


Figura 5.11: G''_{Ra} , aperfeiçoamento desenvolvido no Intrasalas.

Na Figura 5.11 é possível entender melhor qual foi o aperfeiçoamento desenvolvido. Trata-se justamente da adição das arestas representadas através das linhas pontilhadas, as mesmas não indicam movimento algum, por isso têm valor zero. Contudo, possibilitam que ciclos sejam formados a partir de movimentos que envolvam realocações em horários antes não ocupados. No exemplo apresentado, pode-se notar que com o aperfeiçoamento criado, é possível encontrar um ciclo de custo negativo entre os arcos $(H1, H3)$, $(H3, H6)$, $(H6, H1)$. Na figura em questão, destaca-se que haveriam as demais arestas (com respectivos valores calculados) partindo dos horários com aulas para $H6$ (assim como a aresta $(H3, H6)$), representando a transferência da aula para o respectivo horário vago, entretanto, as mesmas foram omitidas para facilitar a compreensão da figura.

É importante ressaltar que a existência de um ciclo negativo pode, todavia, não garantir melhora no valor da função objetivo. Em algumas execuções determinados ciclos podem não corresponderem com seus valores resultantes, devido ao fato do cálculo do valor da aresta ser realizado apenas com a disciplina envolvida, antes e depois de uma possível transferência (cálculo que é realizado dessa forma pelo ganho em tempo computacional). Sendo assim, sempre que um ciclo negativo é executado,

a solução resultante é validada através de uma reavaliação que comprova a melhoria na solução corrente.

5.9.2 Intersalas

Ao final da execução do Intrasalas podem restar, ainda, arcos de custo negativo nos grafos das salas. Contudo, a existência de apenas um arco de custo negativo não garante a alocação de ambos os vértices envolvidos. Em alguns casos, o vértice que não vai ser movimentado não pode ocupar o horário do vértice oposto por haver um conflito entre disciplinas de um mesmo currículo ou ministradas por um mesmo professor. Dessa forma, o movimento Intersalas procura possíveis trocas entre horários de aulas de diferentes salas, em busca de validar a transferência delimitada pela aresta gerada no Intrasalas.

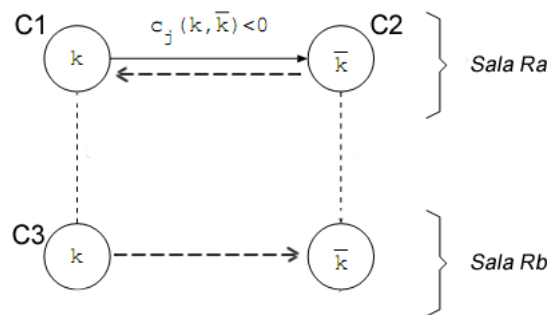


Figura 5.12: Intersalas

Na Figura 5.12, há um arco de custo negativo na sala Ra , de k a $\bar{k}$, indicando que caso a disciplina $C1$ tenha sua aula alterada do horário k para o horário $\bar{k}$ poderá haver uma melhoria na função objetivo. Contudo, a disciplina $C2$ não pode ser transferida do horário k para $\bar{k}$, pois este já está ocupado pela disciplina $C3$ que mesmo sendo de salas diferentes pertence a um mesmo currículo, e por isso não pode ser alocada em um mesmo horário.

A idéia então, é verificar todas as disciplinas relacionadas a $C2$ - seja por currículo ou professor em comum - em possíveis trocas que possibilitem a execução do movimento. No exemplo dado, uma troca de horários entre as disciplinas $C2$ e $C3$ juntamente com a transferência de $C1$ para o horário $\bar{k}$, possibilitariam a melhoria da função objetivo corrente.

Esta parte do procedimento está relacionada ao Intrasalas, visto que, somente é executada quando arestas de custo negativo são encontradas. Quanto ao nome Intersalas é justificado por relacionar aulas de diferentes salas para realização dos movimentos.

5.9.3 O Procedimento Desenvolvido

A partir dos conceitos dos movimentos Intrasalas e Intersalas foi proposto um procedimento iterativo que analisa, a cada vez, as aulas alocadas para uma determinada sala. Para cada sala r é gerado um grafo G_r , no qual é aplicado o algoritmo de Floyd-Warshall em busca de ciclos negativos que tragam melhoria a solução corrente.

Ao final de cada verificação de um grafo é aplicado o Intersalas nas arestas negativas encontradas.

Enquanto houver ciclo de custo negativo que traga melhora na função objetivo, são realizados os respectivos movimentos Intrasalas e Intersalas, atualizando a solução corrente e o grafo G_r . Inexistindo mais tais ciclos no grafo da sala r parte-se para uma nova sala. Esse procedimento é encerrado quando nenhum movimento de melhora for mais possível para todas as salas de uma solução.

5.10 Algoritmos Propostos

Todos os conceitos apresentados até então neste capítulo foram utilizados em diferentes combinações que formam cinco algoritmos distintos, construídos para resolução do problema CBCT. Foram testadas várias combinações entre os procedimentos e conceitos de meta-heurísticas descritos, em busca das implementações realizadas. Os algoritmos propostos são:

- ILS-RELAX;
- ILS-TABU;
- ILS-KEMPE;
- ILS-INTRA;
- ILS-INTRAK;

Outros aspectos como das meta-heurística *Simulated Annealing*, *Multistart* e procedimentos variados para exploração do espaço de busca foram testados também. Contudo, os principais avanços alcançados para resolução do problema com as abordagens realizadas podem ser sintetizadas nos cinco algoritmos, os quais são descritos nas subseções seguintes.

5.10.1 ILS-RELAX

Este algoritmo baseia-se na meta-heurística ILS, para refinamento da solução, juntamente com o procedimento de construção de solução inicial baseado no GRASP. Um aspecto importante desta implementação trata-se de uma modificação no critério de aceitação do ILS, para que, em determinados momentos, o mesmo aceite soluções de piora como nova solução corrente. Essa modificação justifica o nome dado ao algoritmo, por, de certa forma, permitir um relaxamento no critério de aceitação normalmente utilizado no ILS.

É apresentado na Figura 5.13 um fluxograma que destaca a sequência de execução dos principais pontos desta abordagem.

No fluxograma apresentado, é importante esclarecer que *SCurrent*, *SStar* e *Relax* são respectivamente: a solução corrente a cada iteração; a melhor solução encontrada até então; e, variável que controla o nível de piora no critério de aceitação. Nota-se que a partir da solução inicial criada, uma busca local é realizada a fim de refinar a solução encontrada. Em seguida, o ILS continua a exploração do espaço de soluções por meio da aplicação de perturbações, seguidas de buscas locais.

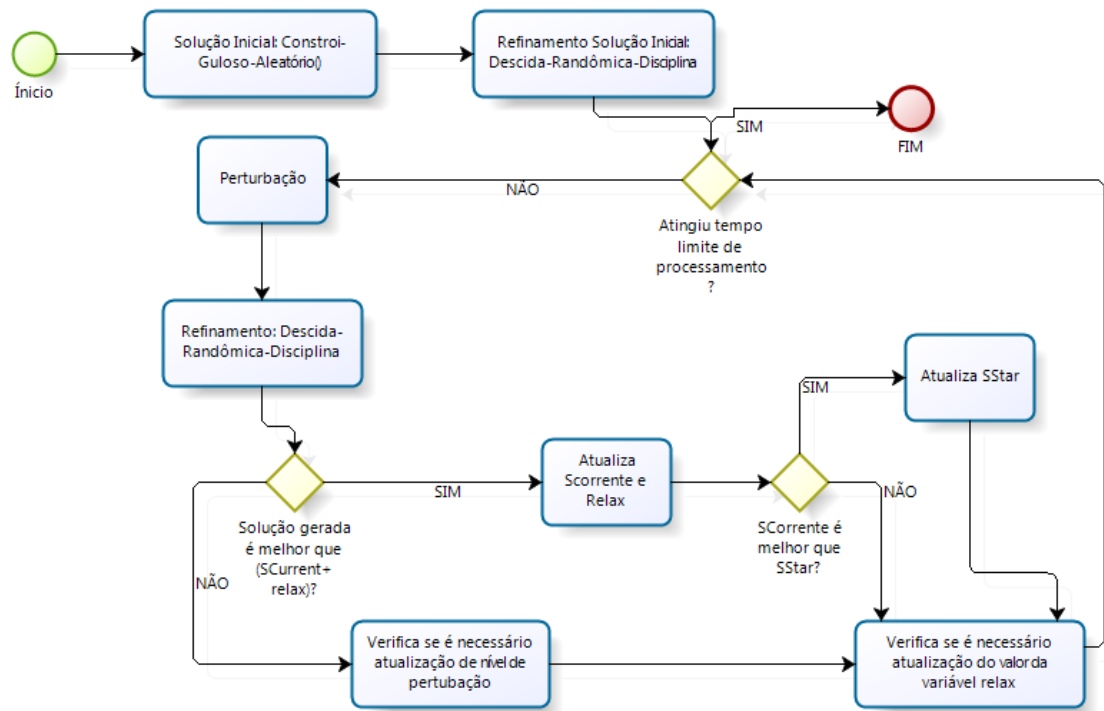


Figura 5.13: Fluxograma da execução do ILS-RELAX

A cada iteração do ILS o tipo de perturbação é selecionado aleatoriamente e o critério de aceitação pode ou não ser atualizado. A variável *relax* trata-se de um possível incremento no valor de avaliação da solução corrente, o qual é atualizado a cada ciclo de iterações. O critério de parada é o tempo de processamento pré-estipulado.

No Algoritmo 8 é apresentado o pseudo-código da implementação realizada. Destaca-se que a variável *SStar* ao final da execução representará a melhor solução encontrada durante o processo de busca. Quanto aos parâmetros do algoritmo desenvolvido, ressalta-se a funcionalidade de cada um:

- *numVizinhos*: número de vizinhos gerados a cada iteração da busca local;
- *maxIt*: número máximo de iterações que a busca local será executada;
- *tempoDeExecucao*: tempo em segundos em que o algoritmo poderá ser executado;
- *iterPorNivel*: número máximo de iterações por nível de perturbação do ILS;
- *numDeNiveis*: número de níveis do ILS. Em cada nível é realizado (*numDeNiveis*+1) movimentos na solução;
- *iterPorRelax*: intervalo de iterações em que o relaxamento é aplicado. Ex: se *iterPorRelax* = 5 a cada cinco iterações a variável *relax* é incrementada;
- *incrementRelax*: valor de incremento da variável *relax*;

Algoritmo 8: ILS-RELAX

Entrada: *numVizinhos*, *maxIt*, *tempoDeExecucao*, *iterPorNivel*, *numDeNiveis*, *iterPorRelax*, *incrementRelax*

Saída: *Solucao*

```

1  início
2  |   relax ← 0;
3  |   nivel ← 1;
4  |   tempo ← 0;
5  |   iteracao, iteracaoSemMelhora ← 0;
6  |   SAux ← solução inicial sem nenhuma aula alocada;           // Solução Auxiliar
7  |   SCurrent ← constróiGulosoAleatorio();                       // Melhor solução Corrente
8  |   SCurrent ← descidaRandomicaPorDisciplina(SAux, numVizinhos, maxIt);
9  |   SStar ← SCurrent;                                           // Melhor solução Global
10 |   enquanto (tempo < tempoDeExecucao) faça
11 |       |   SAux ← perturbação(nivel, SCurrent);
12 |       |   SAux ← descidaRandomicaPorDisciplina(SAux, numVizinhos, maxIt);
13 |       |   se SAux ≤ (SCurrent + relax) então
14 |           |       relax ← 0;
15 |           |       nivel ← 1;
16 |           |       iteracaoSemMelhora ← 0
17 |           |       SCurrent ← SAux;
18 |           |       se SCurrent < SStar então
19 |               |           SStar ← SCurrent;
20 |           |       fim
21 |       |   senão
22 |           |       iteracaoSemMelhora ← iteracaoSemMelhora + 1
23 |           |       se iteracaoSemMelhora resto iterPorNivel == 0 então
24 |               |           nivel ← nivel + 1;
25 |               |           se nivel > numDeNiveis então
26 |                   |               nivel ← 0;
27 |               |           fim
28 |           |       fim
29 |       |   fim
30 |       |   se iteracao resto iterPorRelax == 0 então
31 |           |       relax ← relax + incrementRelax;
32 |       |       fim
33 |       |       iteracao ← iteracao + 1;
34 |   fim
35 |   Retorne SStar
36 fim

```

5.10.2 ILS-TABU

As principais diferenças existentes entre esse algoritmo e o ILS-RELAX, consistem na utilização da meta-heurística Busca Tabu para refinamento da solução após a realização de uma perturbação (substituindo a descida randômica por disciplina) e também na não utilização da “relaxação” no critério de aceitação do ILS. O Algoritmo 9 apresenta o pseudo-código da implementação realizada.

A maioria dos parâmetros deste algoritmo possuem as mesmas funcionalidades descritas para o ILS-RELAX. Contudo, o parâmetro *tamTabu* é responsável por definir o tamanho da lista tabu, mais precisamente para esta implementação, trata-se do número de iterações em que uma alocação recebe o status de Tabu. Destaca-se que *tamTabu* varia de acordo com a instância a ser testada e que em cada iteração os vizinhos são gerados a partir de todas disciplinas, diferentemente da descida randômica por disciplina, que gera vizinhos de apenas uma disciplina a cada iteração.

Esta implementação foi realizada com o objetivo de analisar a combinação entre as estratégias da BT e ILS para percorrer o espaço de busca.

Algoritmo 9: ILS-TABU

Entrada: *numVizinhos*, *maxIt*, *tempoDeExecucao*, *iterPorNivel*, *numDeNiveis*, *tamTabu*
Saída: *Solucao*

```

1  início
2      nivel ← 1;
3      tempo ← 0;
4      iteracaoSemMelhora ← 0;
5      SAux ← solução inicia sem nenhuma aula alocada;           // Solução Auxiliar
6      SStar ← constróiGulosoAleatorio();                         // Melhor solução Global
7      enquanto (tempo < TempoDeExecucao) faça
8          SAux ← perturbacao(nivel, SStar);
9          SAux ← BuscaTabu(SAux, NumVizinhos, MaxIt, tamTabu);
10         se SAux ≤ SStar então
11             SStar ← SStar;
12         senão
13             iteracaoSemMelhora ← iteracaoSemMelhora + 1
14             se iteracaoSemMelhora resto iterPorNivel == 0 então
15                 nivel ← nivel + 1;
16                 se nivel > numDeNiveis então
17                     nivel ← 1;
18             fim
19         fim
20     fim
21 fim
22 Retorne SStar
23 fim

```

5.10.3 ILS-KEMPE

Este algoritmo trata-se do ILS-RELAX aliado ao procedimento denominado *KempeSwap*. O procedimento em questão é utilizado a cada intervalo de iterações, com intuito de explorar um espaço de soluções com movimentos de cadeias *Kempe*. Através da Figura 5.14 é apresentado um comparativo entre parte do fluxograma do ILS-RELAX com e sem o procedimento *KempeSwap*, nesta representação é possível notar em que parte da implementação o procedimento é aplicado.

O único parâmetro adicionado a esta implementação em relação ao ILS-RELAX é *iterPorProcedimento*, trata-se do intervalo de iterações em que o procedimento será aplicado. O pseudo-código deste algoritmo é semelhante ao do ILS-INTRA, com alteração do procedimento a ser aplicado, sendo assim, pode-se assimilar o funcionamento dessa implementação ao Algoritmo 10.

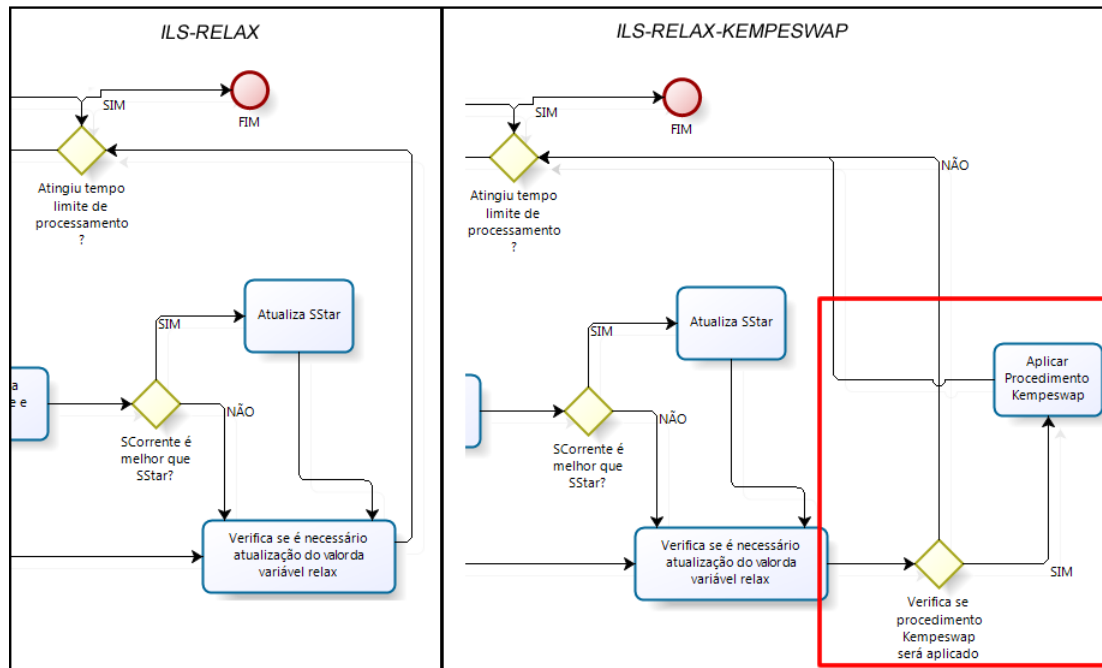


Figura 5.14: Comparações entre fluxogramas do ILS-RELAX e ILS-KEMPE

5.10.4 ILS-INTRA

Esta implementação possui praticamente a mesma estrutura do algoritmo apresentado anteriormente, contudo, o procedimento aplicado é o Intra-Inter-Salas ao invés do Kempswap. Nesta implementação o procedimento também é aplicado a cada intervalo de iterações delimitados pela variável *iterPorProcedimento*. Esse tipo de abordagem justifica-se pelo alto tempo computacional dos procedimentos utilizados, o que inviabiliza a utilização dos mesmos seguidamente.

Através do Algoritmo 10 é apresentado o pseudo-código da implementação realizada.

5.10.5 ILS-INTRAK

Para esta implementação toda estrutura apresentada anteriormente é considerada, porém movimentos do procedimento Kempswap são utilizados como nova forma de perturbação do ILS. Ressalta-se que as perturbações já apresentadas também são consideradas, sendo os movimentos de cadeias kempe apenas mais uma forma, a qual juntamente com as outras, possui chances de ser executada a cada iteração uma vez que as perturbações são escolhidas aleatoriamente.

Essa abordagem, conforme já destacado, deve-se ao fato de que movimentos de cadeias kempe produzem em algumas execuções grandes modificações na solução mantendo a factibilidade, sendo que o valor de avaliação pode ser igualado ou até melhorado através de uma busca local como ocorre no ILS.

Algoritmo 10: ILS-INTRA

Entrada: *numVizinhos*, *maxIt*, *tempoDeExecucao*, *iterPorNivel*, *numDeNiveis*, *iterPorRelax*,
incrementRelax, *iterPorProcedimento*

Saída: *Solucao*

```

1  início
2  |   relax ← 0;
3  |   nivel ← 1;
4  |   tempo ← 0;
5  |   iteracao, iteracaoSemMelhora ← 0;
6  |   SAux ← solução inicial sem nenhuma aula alocada;           // Solução Auxiliar
7  |   SCurrent ← constroiGulosoAleatorio();                       // Melhor solução Corrente
8  |   SCurrent ← descidaRandomicaPorDisciplina(SAux, numVizinhos, maxIt);
9  |   SStar ← SCurrent;                                           // Melhor solução Global
10 |   enquanto (tempo < tempoDeExecucao) faça
11 |       |   SAux ← perturbacao(nivel, SCurrent);
12 |       |   SAux ← descidaRandomicaPorDisciplina(SAux, numVizinhos, maxIt);
13 |       |   se SAux ≤ (SCurrent + relax) então
14 |           |       relax ← 0;
15 |           |       nivel ← 1;
16 |           |       iteracaoSemMelhora ← 0
17 |           |       SCurrent ← SAux;
18 |           |       se SCurrent < SStar então
19 |               |           SStar ← SCurrent;
20 |           |       fim
21 |       |   senão
22 |           |       iteracaoSemMelhora ← iteracaoSemMelhora + 1
23 |           |       se iteracaoSemMelhora resto iterPorNivel == 0 então
24 |               |           nivel ← nivel + 1;
25 |               |           se nivel > numDeNiveis então
26 |                   |               nivel ← 0;
27 |               |           fim
28 |           |       fim
29 |       |   fim
30 |       |   se iteracao resto iterPorRelax == 0 então
31 |           |       relax ← relax + incrementRelax;
32 |       |   fim
33 |       |   se iteracao resto iterPorProcedimento == 0 então
34 |           |       SAux ← IntraRoom – InterRoom(SCurrent);
35 |           |       se SAux < SCurrent então
36 |               |           SCurrent ← SAux;
37 |               |           se SCurrent < S* então
38 |                   |               S* ← SCurrent;
39 |               |           fim
40 |           |       fim
41 |       |   fim
42 |       |   iteracao ← iteracao + 1;
43 |   fim
44 |   Retorne SStar
45 fim

```

Capítulo 6

EXPERIMENTOS COMPUTACIONAIS

Neste capítulo são apresentados os experimentos computacionais realizados com os algoritmos propostos. Foram realizados testes e comparações com diferentes aspectos que envolveram abordagens com características concorrentes, os resultados encontrados na literatura e também desprezando o tempo computacional limitante da competição.

A organização do capítulo é dada pelas seguintes seções:

- **Definição de Parâmetros e Experimentos:** são apresentadas as informações relevantes que envolveram os experimentos, instâncias testadas e parâmetros das implementações;
- **ILS-RELAX vs ILS-TABU:** é realizada uma comparação entre os respectivos algoritmos, visando a análise da influência da relaxação e da Busca Tabu nos algoritmos;
- **ILS-KEMPE vs ILS-INTRA:** é realizada uma comparação entre os respectivos algoritmos, visando a análise da influência dos procedimentos Kempswap e Intra-Inter-Salas;
- **ILS-INTRAK:** são apresentados os resultados do respectivo algoritmo, juntamente com um gráfico de probabilidade empírica que permite uma comparação entre as demais abordagens;
- **Comparações com resultados da literatura:** são apresentados comparações com os resultados encontrados na literatura. Nesta seção, são apresentados também resultados encontrados desprezando o limite de tempo estabelecido pela competição;

6.1 Definições de Parâmetros e Experimentos

As implementações descritas neste trabalho foram feitas sobre a plataforma Java, versão JDK 6.0, utilizando o IDE Netbeans 6.9. Os testes foram realizados em computadores com processador Core 2 Quad 2.2 Ghz, memória RAM de 4GB e

sistema operacional Linux Ubuntu. Foram trinta execuções sobre cada instância, e, para os testes de probabilidade empírica foram escolhidas três instâncias as quais foram executadas cem vezes cada uma.

As instâncias testadas referenciam dados reais da universidade de Udine da Itália, as quais são fornecidas pelo endereço <http://tabu.diegm.uniud.it/ctt/index.php?page=instances>. Tratam-se de diferentes instâncias, com características diversas determinadas pelo número de disciplinas, aulas, salas e currículos. As informações e características referentes a cada instância testada podem ser visualizadas através da Tabela 6.1.

Tabela 6.1: Características das instâncias.

Instância	Disciplinas	Aulas	Salas	Currículos
<i>comp01</i>	30	160	6	14
<i>comp02</i>	82	283	16	70
<i>comp03</i>	72	251	16	68
<i>comp04</i>	79	286	18	57
<i>comp05</i>	54	152	9	139
<i>comp06</i>	108	361	18	70
<i>comp07</i>	131	434	20	77
<i>comp08</i>	86	324	18	61
<i>comp09</i>	76	279	18	75
<i>comp10</i>	115	370	18	67
<i>comp11</i>	30	162	5	13
<i>comp12</i>	88	218	11	150
<i>comp13</i>	82	308	19	66
<i>comp14</i>	85	275	17	60

Cada instância é definida por um arquivo no formato *ctt*¹, neste arquivo todos os dados necessários para construção de um quadro de horários para um determinado problema são fornecidos. Um arquivo *ctt* contém um cabeçalho e quatro seções: disciplinas, salas, turmas e restrições de indisponibilidade. No cabeçalho são fornecidos os números que envolvem o problema (ex: número de salas, número de aulas), e nas demais seções as informações sobre as respectivas. É apresentado como anexo a este trabalho uma instância exemplo do problema (*comp01*).

Quanto aos resultados obtidos, as principais formas utilizadas na comparação desses foram: i) através dos melhores valores encontrados em cada abordagem, juntamente com a média e o desvio padrão; ii) através de gráficos com erros percentuais; iii) através de gráficos de probabilidade empírica. Os gráficos com erros percentuais são obtidos a partir das médias encontrados nas execuções. Para cada instância são considerados o valor médio, o desvio percentual máximo e o desvio percentual mínimo. Destaca-se também que a principal forma de comparação utilizada na competição é a comparação entre melhores valores encontrados em cada abordagem, aspecto que justifica a forma que os resultados foram analisados neste trabalho.

¹<http://satt.diegm.uniud.it/ctt/index.php?page=format>

A definição dos parâmetros que envolvem as implementações, são de suma importância para a eficiência das abordagens realizadas. Cabe salientar, que todos os parâmetros envolvidos foram definidos através de testes empíricos realizados exaustivamente sobre as diversas instâncias envolvidas. Assim sendo, através da Tabela 6.2 são apresentados os parâmetros para cada implementação.

Tabela 6.2: Parâmetros das implementações.

ILS-RELAX	ILS-TABU
$numVizinhos = 15$ $maxIt = 5 \times n^{\circ} \text{ disciplinas}$ $iterPorNivel = 4$ $numDeNiveis = 2$ $iterPorRelax = 3$ $incrementRelax = 1$	$numVizinhos = 30$ $maxIt = 15 \times n^{\circ} \text{ disciplinas}$ $iterPorNivel = 4$ $numDeNiveis = 2$ $tamTabu = 20\% n^{\circ} \text{ disciplinas}$
ILS-KEMPE	ILS-INTRA E INTRAK
$numVizinhos = 20$ $maxIt = 5 \times n^{\circ} \text{ disciplinas}$ $iterPorNivel = 4$ $numDeNiveis = 2$ $iterPorRelax = 3$ $incrementRelax = 1$ $iterPorProcedimento = 30$	$numVizinhos = 10$ $maxIt = 5 \times n^{\circ} \text{ disciplinas}$ $iterPorNivel = 4$ $numDeNiveis = 2$ $iterPorRelax = 3$ $incrementRelax = 1$ $iterPorProcedimento = 50$

Nos parâmetros apresentados na Tabela 6.2 alguns pontos devem ser destacados. Primeiramente, sobre o parâmetro *tamTabu* que trata-se de um parâmetro dinâmico que varia de acordo com a instância testada, esta abordagem foi necessária, visto a variação existente entre as diferentes instâncias. Na implementação com Busca Tabu é possível notar também o aumento do número de iterações de *maxIt*, medida que foi adota para maior eficiência da lista tabu. Já o parâmetro *iterPorProcedimento*, varia entre as implementações com os procedimento Kempeswap e Intra-Inter-Sala, devido ao custo computacional envolvido em cada procedimento.

Por fim, ressalta-se que como critério de parada foi utilizado um tempo máximo de 357 segundos (parâmetro *tempoDeExecucao*), este tempo é calculado de acordo com o *hardware* em que as instâncias serão executadas. O cálculo é feito através de um software fornecido no site http://www.cs.qub.ac.uk/itc2007/index_files/benchmarking.htm e faz parte do regulamento do ITC 2007, sendo estas e outras informações descritas no trabalho de Di Gaspero e Schaerf (2007).

6.2 ILS-RELAX vs ILS-TABU

A utilização de uma variável que permita que soluções de piora sejam aceitas no critério de aceitação do ILS, foi um aprimoramento que apresentou melhoras significativas nos resultados obtidos. Sendo a Busca Tabu uma meta-heurística que tem características de coibir determinados movimentos repetitivos permitindo

também que soluções de piora sejam aceitas, foi realizada uma implementação em que o ILS é combinado com a meta-heurística. As duas implementações foram comparadas, pois, de certa forma, apresentaram um aspecto em comum: permitir soluções de piora, em diferentes estratégias.

As comparações foram feitas a partir das implementações ILS-RELAX, ILS-TABU e também uma implementação similar ao ILS-RELAX sem a utilização do relaxamento (a qual recebe o nome de ILS*). Os testes com ILS* foram realizados com intuito de comprovar o peso da utilização da técnica de relaxação no ILS.

Através da Tabela 6.3 são apresentados os resultados obtidos, sendo estes: melhor valor de avaliação encontrado (**MELHOR VALOR**), valor médio para cada instância (**VALOR MÉDIO**) e o desvio padrão dos resultados obtidos para cada instância (**DESVIO PADRÃO**). Os melhores valores para cada comparação aparecem em negrito na tabela.

Tabela 6.3: Comparações entre ILS*, ILS-RELAX e ILS-TABU.

<i>Instância</i>	MELHOR VALOR			VALOR MÉDIO			DESVIO PADRÃO		
	<i>ILS*</i>	<i>ILS-TABU</i>	<i>ILS-RELAX</i>	<i>ILS*</i>	<i>ILS-TABU</i>	<i>ILS-RELAX</i>	<i>ILS*</i>	<i>ILS-TABU</i>	<i>ILS-RELAX</i>
<i>comp01.ctt</i>	5	5	5	6,20	5,53	5,00	0,85	0,82	0,00
<i>comp02.ctt</i>	125	150	114	160,13	167,42	147,37	24,41	16,14	18,61
<i>comp03.ctt</i>	132	146	97	155,38	163,00	140,10	15,06	9,05	18,10
<i>comp04.ctt</i>	50	57	46	73,07	67,05	55,75	10,70	6,48	6,14
<i>comp05.ctt</i>	391	528	370	1604,93	845,50	1384,03	940,85	180,60	821,74
<i>comp06.ctt</i>	105	111	91	148,20	125,42	118,53	23,51	9,07	13,42
<i>comp07.ctt</i>	107	91	70	137,47	105,08	102,27	21,35	7,99	17,71
<i>comp08.ctt</i>	64	57	49	82,40	69,33	60,27	9,74	8,30	5,79
<i>comp09.ctt</i>	134	134	118	146,40	141,33	135,90	9,81	7,61	8,22
<i>comp10.ctt</i>	77	76	58	113,37	87,08	81,37	19,43	6,98	14,32
<i>comp11.ctt</i>	0	0	0	0,50	1,87	0,00	0,57	2,70	0,00
<i>comp12.ctt</i>	411	467	404	462,53	503,17	444,10	28,53	20,26	27,36
<i>comp13.ctt</i>	93	90	84	113,07	99,08	92,37	8,58	5,65	6,00
<i>comp14.ctt</i>	79	94	71	104,80	109,75	86,97	12,55	10,28	8,65

Na Tabela 6.3 é possível notar que o algoritmo ILS-RELAX encontrou as melhores soluções para todas as instâncias. Nas médias encontradas o algoritmo também se destacou, apenas no desvio padrão o ILS-TABU apresentou valores menores. Apesar da implementação ILS-TABU ter uma variação menor que a ILS-RELAX, apresenta uma eficiência bem menor comprovada pelas comparações entre as médias e os melhores valores encontrados.

Já a implementação ILS* apresentou melhores valores do que a ILS-TABU para uma boa parte das instâncias, o que de certa forma, mostra que a combinação das meta-heurísticas BT e ILS não surtiu o efeito desejado, que era justamente aumentar a eficiência do ILS convencional. Foi possível notar durante as execuções, que a BT, inicialmente - em soluções de baixa qualidade - realiza uma rápida melhora das soluções, principalmente em instâncias em que a factibilidade é difícil de ser alcançada. A técnica se mostra lenta quando chega em um espaço de busca com vários ótimos locais, ou seja, a medida que se aproxima das melhores soluções encontradas, momento em que a técnica ILS-RELAX apresenta ampla vantagem.

Uma possível resposta para este fato é que a BT seja mais lenta por natureza ao passar por regiões promissoras, contudo, é uma técnica que pode ser mais poderosa que o ILS convencional e encontrar soluções ainda não encontradas, aspecto que não pode ser melhor explorado pela limitação de tempo de execução do algoritmo. Outro aspecto agravante nesta implementação é a calibração dos parâmetros da BT, trata-se de uma tarefa extremamente difícil visto que o algoritmo é aplicado em várias instâncias de diferentes tamanhos e também pela natureza do problema que dificulta uma calibração eficiente para o conjunto de testes.

É apresentado também um gráfico com erros percentuais através da Figura 6.1.

O gráfico de erros percentuais evidencia a melhor performance do ILS-RELAX sobre as outras abordagens, além de dimensionar a diferença existente entre os resultados obtidos e os melhores valores encontrados para cada instância na competição.

Através dos resultados apresentados pode-se afirmar que a implementação da relaxação no ILS para este problema surtiu um efeito positivo a ser considerado. A utilização deste artifício no critério de aceitação do ILS é justificada pelo fato do problema estudado ser caracterizado por vários ótimos locais e regiões planas, as quais dificultam a exploração do espaço de soluções. Busca-se com esta técnica, aliada as perturbações, uma forma mais eficaz de percorrer o espaço de busca.

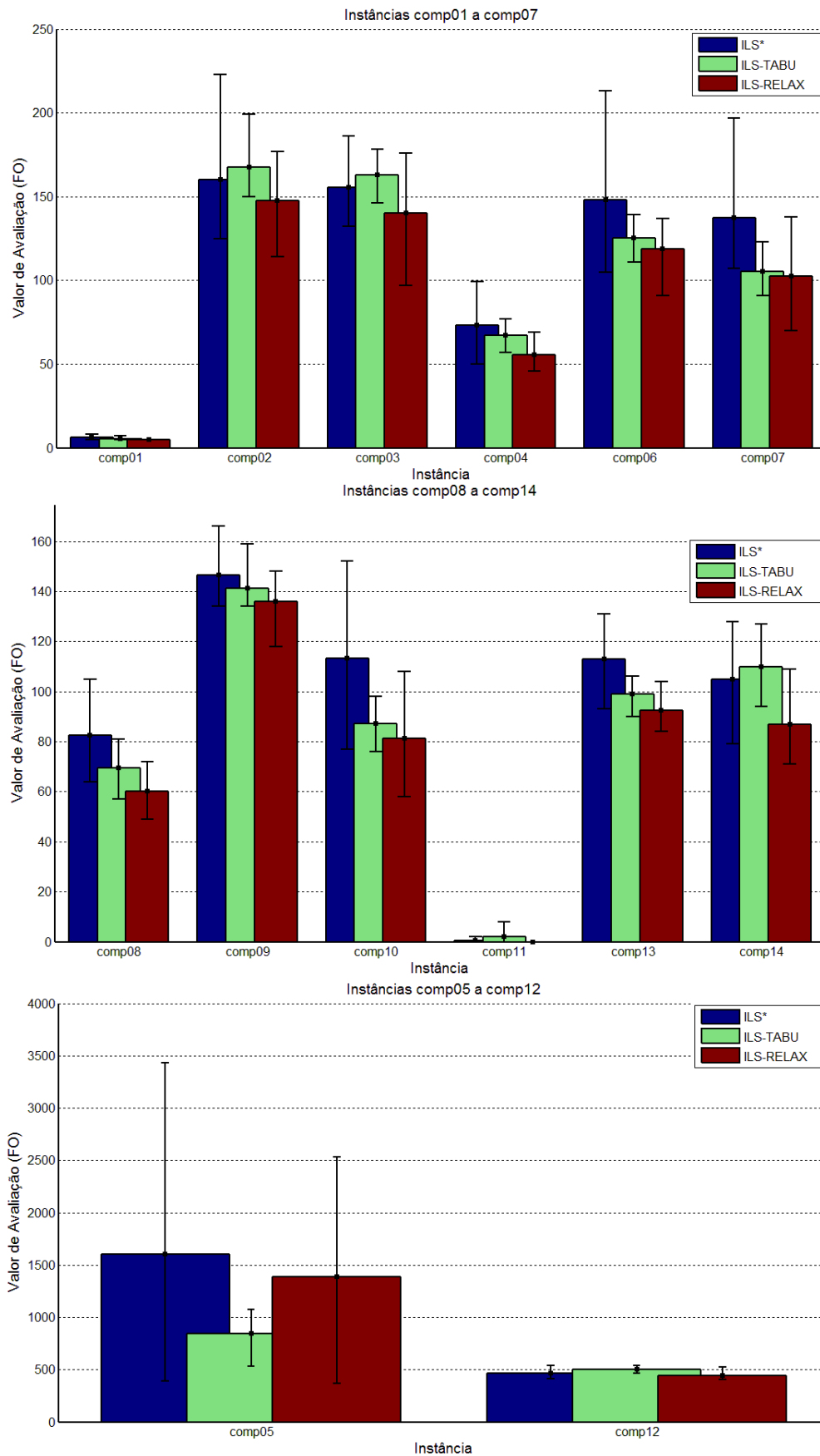


Figura 6.1: Média, Erro Mínimo e Erro Máximo: ILS*, ILS-RELAX e ILS-TABU.

6.3 ILS-KEMPE vs ILS-INTRA

Sendo um dos objetivos desse trabalho o estudo de movimentos avançados que apresentem determinada eficiência em formas não convencionais de geração de vizinhança, buscou-se através do estudo de alguns destes, implementações que melhorem os resultados obtidos a partir do ILS-RELAX. Através da implementação de procedimentos que funcionam como uma busca local realizada com esses movimentos, as seguintes implementações foram realizadas: primeiramente o procedimento Kempeswap e em seguida o procedimento Intra-Inter-Salas.

Como apresentado na seção 5.10, estes procedimentos são aplicados a cada intervalo de iterações devido ao custo computacional envolvido, sendo a partir desta aplicação definidas duas novas abordagens denominadas ILS-KEMPE e ILS-INTRA. Na Tabela 6.4 são apresentados os resultados referentes a essas implementações juntamente com ILS-RELAX, sendo os melhores valores para cada comparação sinalizados em negrito.

Tabela 6.4: Comparações entre ILS-RELAX, ILS-KEMPE e ILS-INTRA.

<i>Instância</i>	MELHOR VALOR			VALOR MÉDIO			DESVIO PADRÃO		
	<i>ILS-RELAX</i>	<i>ILS-KEMPE</i>	<i>ILS-INTRA</i>	<i>ILS-RELAX</i>	<i>ILS-KEMPE</i>	<i>ILS-INTRA</i>	<i>ILS-RELAX</i>	<i>ILS-KEMPE</i>	<i>ILS-INTRA</i>
<i>comp01.ctt</i>	5	5	5	5,00	5,00	5,00	0,00	0,00	0,00
<i>comp02.ctt</i>	114	96	102	147,37	129,60	127,07	18,61	18,32	14,72
<i>comp03.ctt</i>	97	92	96	140,10	129,52	127,59	18,10	17,34	13,12
<i>comp04.ctt</i>	46	39	37	55,75	53,85	50,44	6,14	7,43	6,40
<i>comp05.ctt</i>	370	381	349	1384,03	1306,03	1086,03	821,74	783,29	808,71
<i>comp06.ctt</i>	91	90	87	118,53	112,20	106,43	13,42	11,20	12,76
<i>comp07.ctt</i>	70	70	50	102,27	94,60	84,47	17,71	13,92	15,42
<i>comp08.ctt</i>	49	49	48	60,27	59,03	58,93	5,79	6,47	7,31
<i>comp09.ctt</i>	118	113	110	135,90	130,70	128,20	8,22	7,54	7,30
<i>comp10.ctt</i>	58	53	49	81,37	82,17	70,37	14,32	14,99	12,27
<i>comp11.ctt</i>	0	0	0	0,00	0,00	0,00	0,00	0,00	0,00
<i>comp12.ctt</i>	404	392	388	444,10	435,60	430,67	27,36	35,58	18,96
<i>comp13.ctt</i>	84	79	81	92,37	90,43	88,53	6,00	7,84	4,39
<i>comp14.ctt</i>	71	62	70	86,97	81,37	81,57	8,65	8,40	6,76

Na Tabela 6.4 nota-se que a utilização de ambos procedimentos trouxeram melhoras nos resultados obtidos em comparação com a implementação ILS-RELAX. As melhoras podem ser observadas nos três quesitos de comparação: melhor valor, média e desvio padrão. Sendo assim, a aplicação de tais procedimentos nos algoritmos construídos é validada.

Quanto a abordagem com a utilização do Kempeswap, pode-se afirmar que trata-se de movimentos de troca de períodos entre aulas correlacionadas, no qual várias aulas podem ser alteradas em um mesmo movimento, não onerando a custo da nova solução gerada e muitas vezes visitando espaços de busca ainda não visitados. Sua utilização no procedimento ocorre como uma descida randômica, ou seja, os parâmetros dos movimentos são escolhidos aleatoriamente e executados em um número determinado de iterações.

Já no algoritmo que utiliza os conceitos de movimentos Intra-salas e Inter-salas,

tem-se uma abordagem em que todos os possíveis movimentos de transferência de horário das aulas que ocorrem em uma sala, são testados e valorizados através de um grafo. Grafo, que pode sinalizar uma possível melhora na solução, através de ciclos mínimos de custo negativo, sendo a execução deste ciclo mínimo o movimento intra-salas. O movimento inter-salas realiza verificações nos grafos gerados no intra-salas, para que possíveis trocas de horários entre aulas de salas diferentes possam melhorar a solução corrente. No procedimento construído os dois tipos de movimentos são utilizados em conjunto para verificação de cada sala da solução, ou seja, este procedimento trabalha como uma busca local de descida, na qual todas as salas serão analisadas.

Os dois procedimentos apresentam características diferentes na forma de explorar o espaço de soluções. O procedimento *KempeSwap* realiza vários movimentos de cadeias *Kempe*, e ao final a melhor solução encontrada é retornada. Neste processo várias soluções são geradas e também desprezadas por não apresentarem nenhuma melhora. No procedimento *Intra-Inter-Salas*, em sua principal abordagem várias verificações são realizadas, mas os movimentos só são realizados quando são encontrados ciclos mínimos negativos, dessa forma, um número menor de soluções são geradas, contudo, na maioria das vezes essas soluções representam melhoras na solução corrente.

Analisando essas características em uma comparação entre as abordagens, pode-se dizer que o procedimento *KempeSwap* apresenta um aspecto de maior diversificação da solução, movimentando aulas de diferentes salas e disciplinas. Enquanto o procedimento *Intra-Inter-Salas* apresenta um aspecto de maior intensificação (mais guloso), no qual, a partir da alocação de salas existente, busca-se a melhor alocação de horários para as aulas das disciplinas. Outro aspecto que deve ser ressaltado é que o procedimento *KempeSwap* é mais rápido que o *Intra-Inter-Salas*, por ser um descida randômica e também pelo fato de que o número de verificações para se construir o grafo no *Intra-Salas* é extremamente alto.

Na Figura 6.2 são apresentados: o valor médio, valor mínimo e valor máximo alcançados para cada instância, através dos gráficos de erro percentual. Observando a Figura 6.2 juntamente com a Tabela 6.4 é possível notar que o algoritmo *ILS-INTRA* teve um desempenho superior na maioria das instâncias que o *ILS-KEMPE*. Apesar de mais lenta e também ser utilizada em intervalos de iterações maiores, a implementação com procedimento *Intra-Inter-Salas* deixou o algoritmo mais robusto, destacando-se nos resultados obtidos.

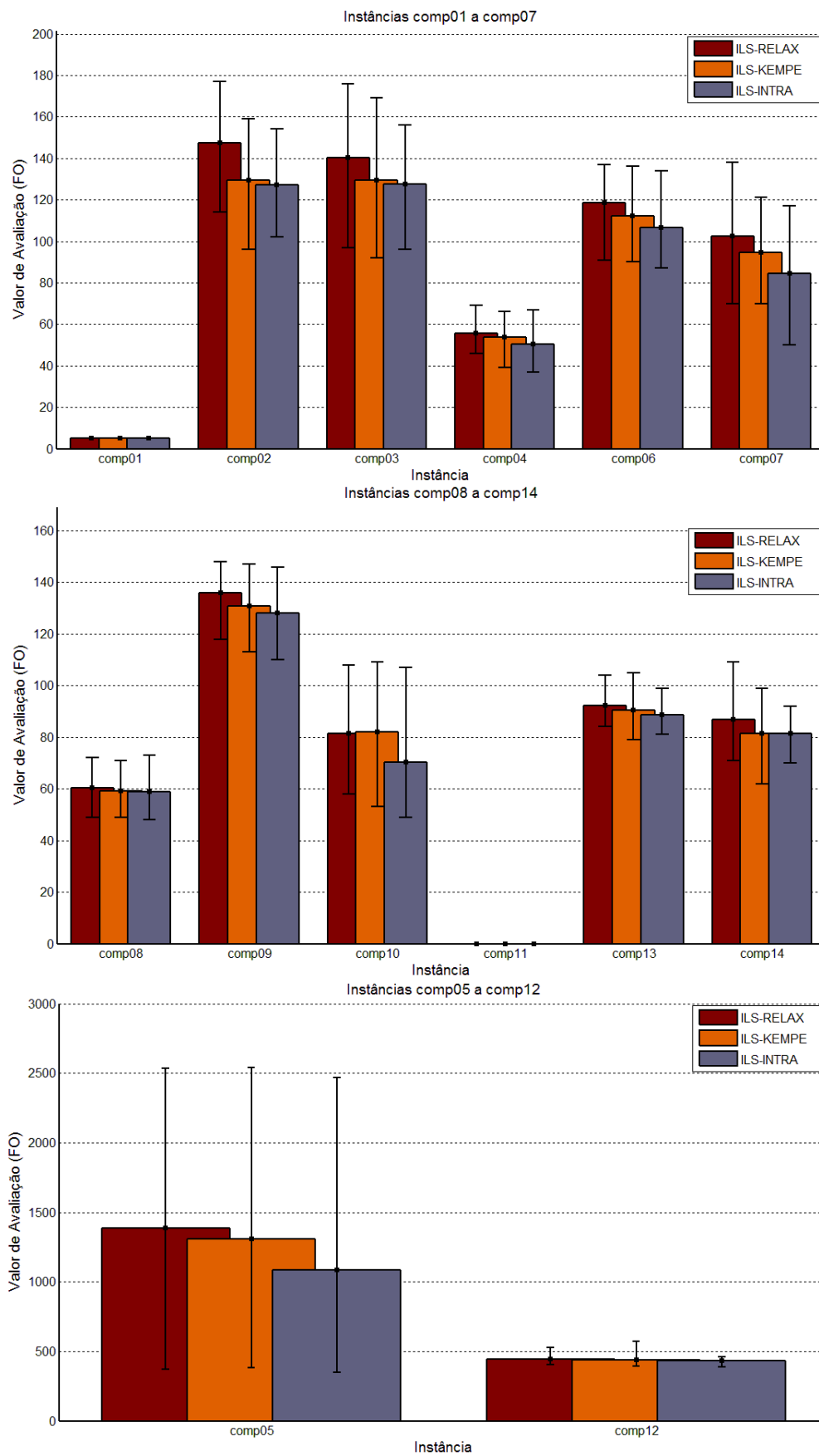


Figura 6.2: Média, Erro Mínimo e Erro Máximo: ILS-RELAX, ILS-KEMPE e ILS-INTRA.

6.4 ILS-INTRAK

Conforme descrito na seção anterior os procedimentos desenvolvidos apresentaram diferentes aspectos, um favorecendo mais a diversificação e outro a intensificação. Através dos resultados foi possível notar que o algoritmo com o procedimento Intra-Inter-Salas foi mais eficiente, contudo, os resultados do algoritmo ILS-KEMPE apresentaram também uma melhora significativa em relação ao ILS-RELAX.

Tendo os dois procedimentos princípios diferentes, em que ambos apresentaram melhora na resolução do problema, ficou evidente que estas características poderiam ser combinadas em busca de melhoras. Em um primeiro momento, os dois procedimentos foram utilizados em conjunto, cada um em um intervalo diferente de iterações. Esta implementação não foi eficiente, devido ao tempo computacional que acabou sendo onerado, o que piorou os resultados obtidos.

Sendo assim, a utilização do Kempeswap no algoritmo ILS-INTRA foi feita na forma de perturbação do ILS, sendo este o algoritmo ILS-INTRAK. O que não sobrecarrega tanto o tempo computacional como a utilização de um procedimento, e pode favorecer a diversificação mantendo a qualidade da solução. Com o objetivo de validar esta implementação foram realizadas comparações entre o ILS-INTRA, as quais são apresentadas na Tabela 6.5.

Tabela 6.5: Comparações entre ILS-INTRA e ILS-INTRAK.

<i>Instância</i>	MELHOR VALOR		VALOR MÉDIO		DESVIO PADRÃO	
	<i>ILS-INTRA</i>	<i>ILS-INTRAK</i>	<i>ILS-INTRA</i>	<i>ILS-INTRAK</i>	<i>ILS-INTRA</i>	<i>ILS-INTRAK</i>
<i>comp01.ctt</i>	5	5	5,00	5,00	0,00	0,00
<i>comp02.ctt</i>	102	85	127,07	120,57	14,72	21,71
<i>comp03.ctt</i>	96	93	127,59	129,79	13,12	16,60
<i>comp04.ctt</i>	37	37	50,44	49,89	6,40	5,75
<i>comp05.ctt</i>	349	363	1086,03	1056,30	808,71	821,47
<i>comp06.ctt</i>	87	85	106,43	105,40	12,76	13,06
<i>comp07.ctt</i>	50	50	84,47	83,70	15,42	15,22
<i>comp08.ctt</i>	48	45	58,93	55,33	7,31	5,31
<i>comp09.ctt</i>	110	108	128,20	126,70	7,30	10,00
<i>comp10.ctt</i>	49	47	70,37	74,87	12,27	13,18
<i>comp11.ctt</i>	0	0	0,00	0,00	0,00	0,00
<i>comp12.ctt</i>	388	386	430,67	429,70	18,96	25,15
<i>comp13.ctt</i>	81	77	88,53	88,23	4,39	6,11
<i>comp14.ctt</i>	70	64	81,57	79,70	6,76	8,09

Através destes resultados é possível notar que existe uma pequena melhora na utilização do ILS-INTRAK. Talvez esta melhora não seja tão brusca pelo fato de que a adição de uma forma de perturbação não altere tanto os resultados, mas ainda assim deve ser considerada. Outro ponto interessante é que a variação entre maior e menor valor encontrado, tornou-se menor no ILS-INTRAK. Aspecto que pode ser melhor visualizado na Figura 6.3, na qual é realizada uma comparação entre as abordagens ILS-KEMPE, ILS-INTRA e ILS-INTRAK.

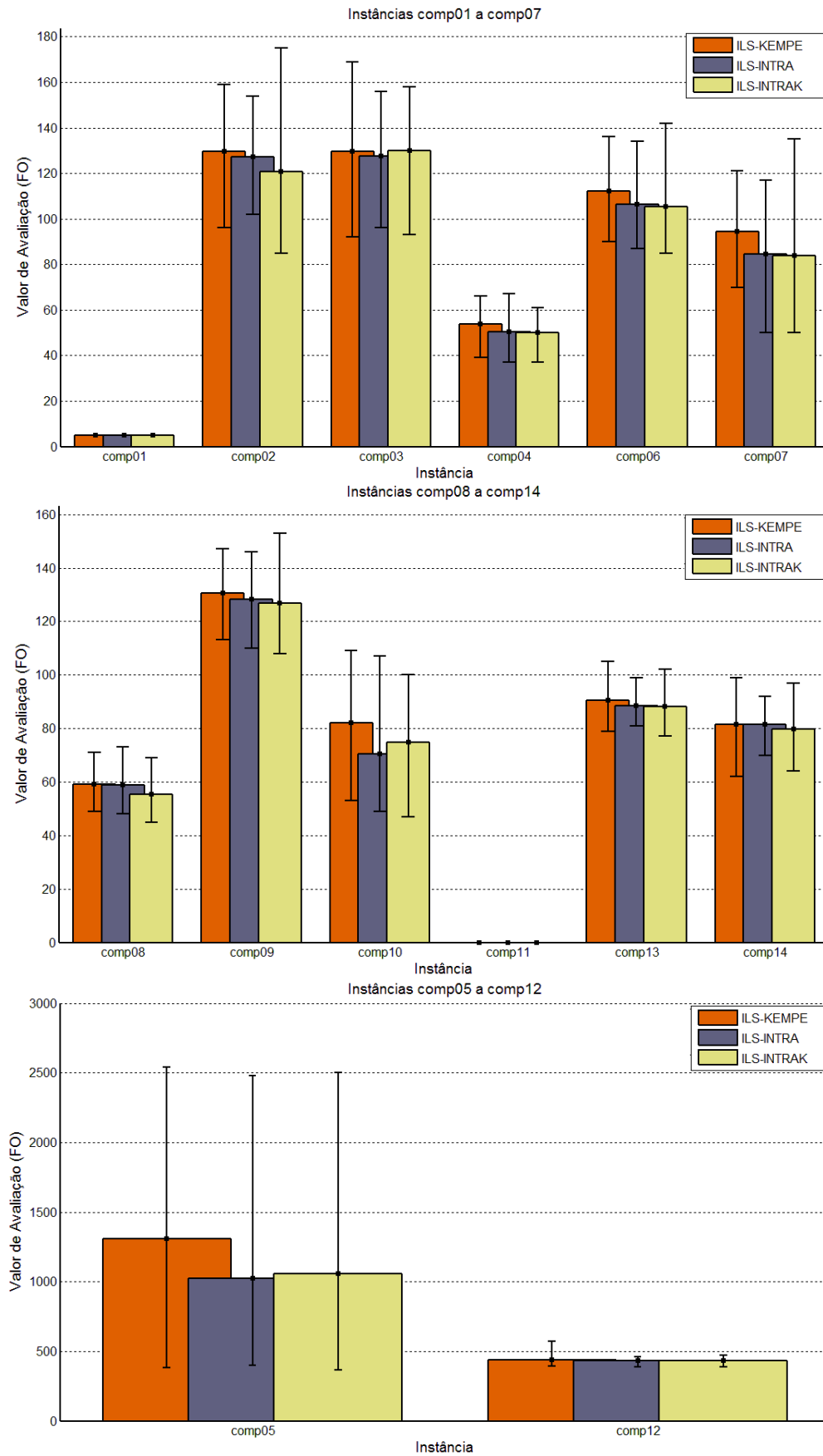


Figura 6.3: Média, Erro Mínimo e Erro Máximo: ILS-RELAX, ILS-RELAX-KEMPESWAP e ILS-RELAX-INTRAINTERALAS.

Um outro tipo de análise foi realizada através de gráficos de probabilidade empírica. Com estes gráficos é possível verificar a eficiência dos algoritmos em relação ao tempo de processamento. Define-se um valor alvo e um tempo limitante para as execuções a serem realizadas, a partir destas é obtido uma porcentagem de execuções em que se conseguiu atingir o valor alvo sem esgotar o tempo limitante.

Foram realizados três testes com instâncias de diferentes características, sendo estas: *comp02*, instância de tamanho médio em que a otimalidade é difícil de ser alcançada; *comp11*, instância pequena em que a solução ótima pode ser alcançada; e *comp12*, instância grande em que a factibilidade é difícil de ser alcançada. Como tempo limitante foi utilizado 357 segundos, o mesmo tempo limitante dos outros testes realizados.

Para a instância *comp02* o valor alvo foi definido como sendo 132 (10% a menos que a média obtida pelo ILS-RELAX), sendo o gráfico de probabilidade empírica das execuções apresentado através da Figura 6.4. É possível notar que o ILS-INTRAK apresentou melhor desempenho, alcançando o valor alvo em mais de sessenta por cento das execuções. Outro fato interessante, é que nos instantes iniciais o ILS-INTRAK foi a abordagem que menos encontrou o valor alvo, devido ao custo computacional do procedimento Intra-inter-salas, ou seja, inicialmente é mais lento mas apresenta maior eficiência ao decorrer do tempo.

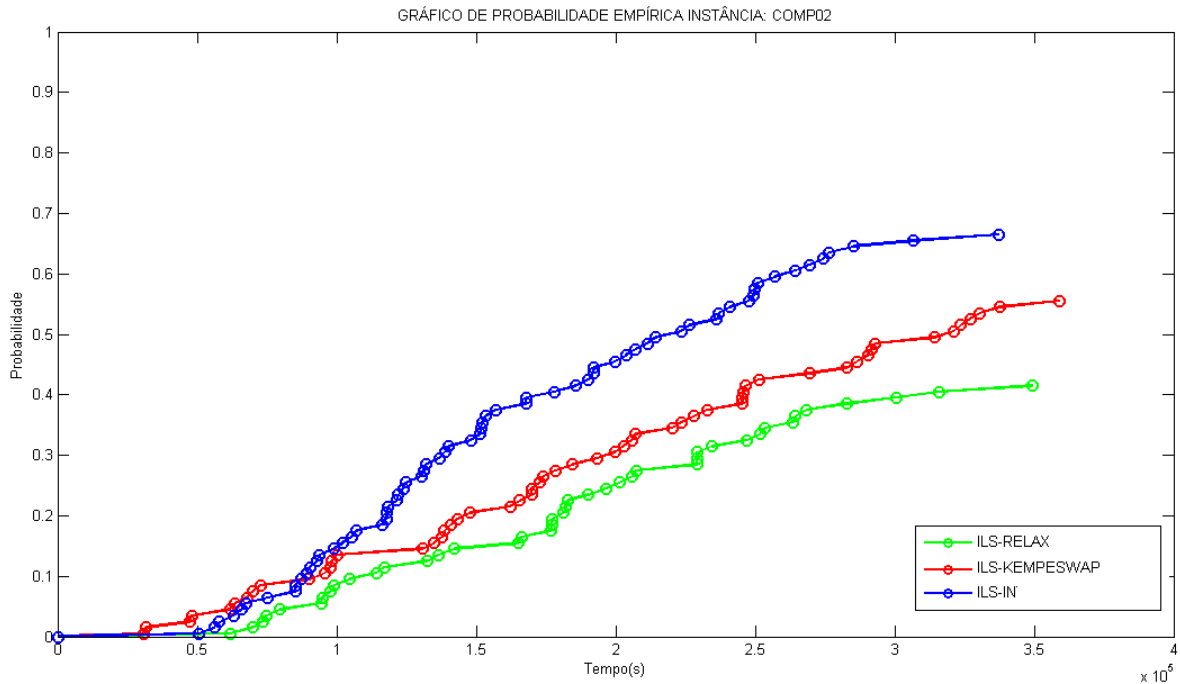


Figura 6.4: Gráfico de Probabilidade Empírica, instância *comp02*.

Para a instância *comp11* o valor alvo foi definido como sendo 0 (valor ótimo), sendo o gráfico de probabilidade empírica das execuções apresentado através da Figura 6.5. Analisando o gráfico é possível afirmar que trata-se de uma instância em que o ILS-RELAX já é suficiente para encontrar o valor ótimo. O peso computacional dos procedimentos Kempeswap e Intra-Inter-salas pode ser notado, uma vez que, enquanto o ILS-RELAX encontra o valor alvo em até $\simeq 28s$ as abordagens desses procedimentos gastam respectivamente $\simeq 32s$ e $\simeq 34s$.

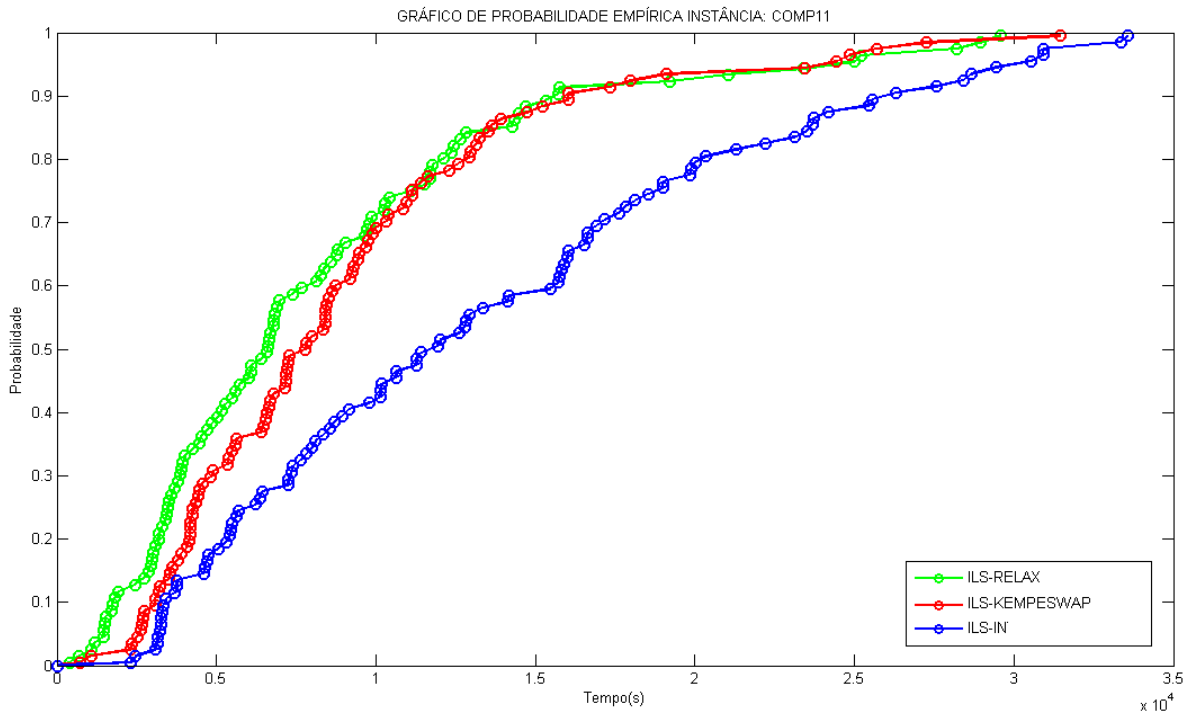


Figura 6.5: Gráfico de Probabilidade Empírica, instância *comp11*.

Para a instância *comp12* o valor alvo foi definido como sendo 488 (10% a mais que a média obtida pelo ILS-RELAX), sendo o gráfico de probabilidade empírica das execuções apresentado através da Figura 6.6.

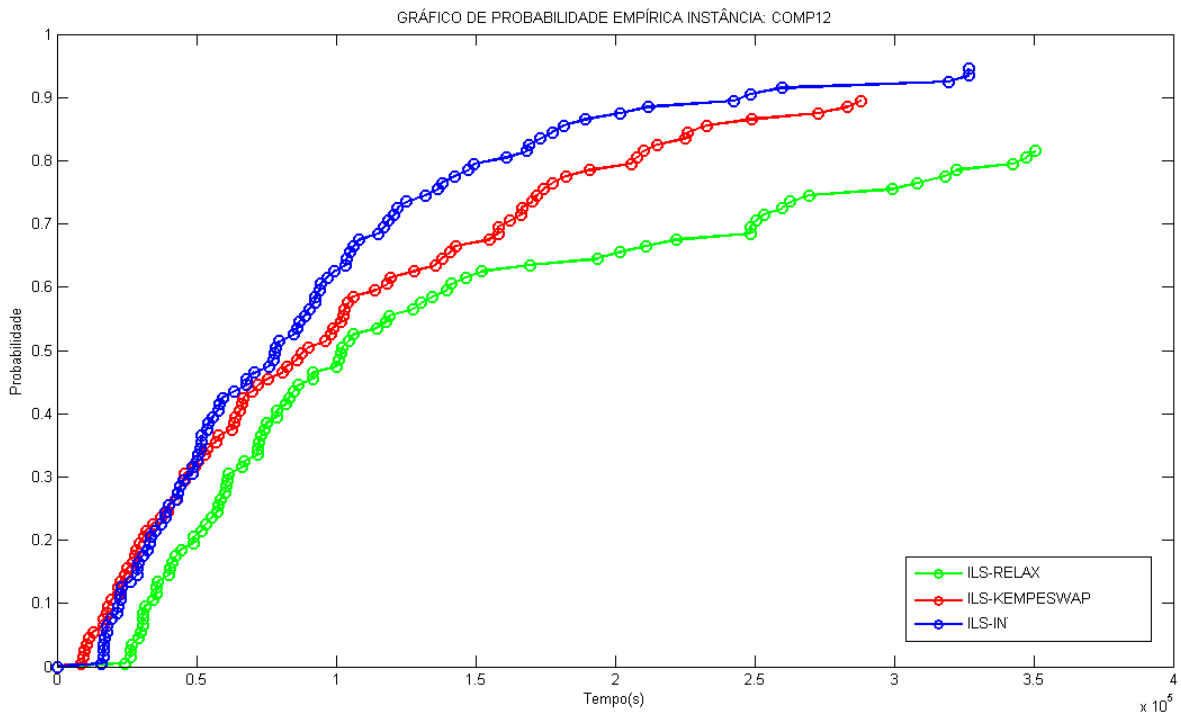


Figura 6.6: Gráfico de Probabilidade Empírica, instância *comp12*.

Neste gráfico é possível notar como os procedimentos implementados auxiliam não só no processo de busca da otimalidade mas também na busca da factibilidade. Para esta instância mesmo nos primeiros segundos, onde os procedimentos tendem a perder pelo custo computacional, o ILS-KEMPE e o ILS-INTRAK foram mais eficientes, devido a complexidade da instância.

A partir de todos os testes realizados até aqui, pode-se concluir que a abordagem mais eficiente entre as que foram propostas, é o ILS-INTRAK. Além de encontrar as melhores soluções para uma boa parte das instâncias, foi a que apresentou melhor comportamento nos gráficos de probabilidade empírica. Na junção do procedimento Intra-Inter-salas ao ILS-RELAX com Kempeswap como perturbação, são destacados pontos como a intensificação e diversificação no processo de busca e também a aceitação de soluções de piora.

6.5 Comparações com Resultados da Literatura

Nesta seção são apresentadas comparações entre os resultados da literatura e os melhores resultados alcançados neste trabalho. Primeiramente, as comparações são realizadas com os resultados do ITC 2007, os quais foram todos alcançados sobre o mesmo fator limitante, o tempo computacional estabelecido pela competição. Dessa forma, na Tabela 6.6 é possível visualizar os resultados dos cinco primeiros colocados da competição juntamente com os resultados alcançados pelo algoritmo ILS-INTRAK, ressalta-se que os resultados ~~risca~~ são maiores ou iguais aos alcançados pela abordagem realizada.

Tabela 6.6: Resultados obtidos através do ILS-INTRAK e comparações com os resultados da literatura.

INSTÂNCIA	Müller (USA)	Lu, Hao (França)	Atsuta et. Al (Japão)	Geiger (Alemanha)	Clark et. Al (Singapura)	ILS-INTRA2
<i>comp01.ctt</i>	5	5	5	5	10	5
<i>comp02.ctt</i>	51	55	50	111	111	85
<i>comp03.ctt</i>	84	71	82	128	119	93
<i>comp04.ctt</i>	37	43	35	72	72	37
<i>comp05.ctt</i>	330	309	312	410	426	363
<i>comp06.ctt</i>	48	53	69	100	130	85
<i>comp07.ctt</i>	20	28	42	57	110	50
<i>comp08.ctt</i>	41	49	40	77	83	45
<i>comp09.ctt</i>	109	105	110	150	139	108
<i>comp10.ctt</i>	16	21	27	71	85	47
<i>comp11.ctt</i>	0	0	0	0	3	0
<i>comp12.ctt</i>	333	343	351	442	408	386
<i>comp13.ctt</i>	66	73	68	98	113	77
<i>comp14.ctt</i>	59	57	59	90	84	64
<i>RANK ITC</i>	1°	2°	3°	4°	5°	-

Analisando a Tabela 6.6 pode-se afirmar que os resultados obtidos foram satisfatórios frente aos resultados do ITC 2007. Destaca-se que: i) na maioria das instân-

cias os valores alcançados foram melhores que os do quarto e quinto colocados; ii) nas instâncias *comp04*, *comp08* e *comp09* conseguiu-se alguns resultados superiores aos do terceiro, segundo e primeiro colocados; iii) em duas instâncias conseguiu-se chegar ao valor ótimo conhecido - *comp01* e *comp11*-. Ressalta-se também, que em todas as resoluções são alcançadas soluções factíveis, uma tarefa difícil em algumas instâncias pela alta probabilidade de conflitos.

Para que a implementação chegasse ao ponto de ser comparada aos primeiros colocados do ITC 2007, pode-se dizer que foi realizado um processo lento e empírico de implementações e testes de diversas técnicas em busca de uma implementação que obtivesse certa eficiência diante dos demais competidores. Há a que se destacar, que as abordagens realizadas pelos primeiros colocados da competição, são implementações bem robustas em que várias técnicas são combinadas e também que alguns desses pesquisadores apresentam grande importância na área de *timetabling*, principalmente para o problema estudado como pode ser visto nos trabalhos de Muller (2005), Lü e Hao (2010) e Geiger (2010).

O tempo computacional limitante utilizado na competição é relativamente baixo considerando a resolução de problemas de alocação de horários. Testes nos quais o tempo é desprezado, ou aumentado significativamente, podem encontrar soluções melhores que as apresentadas na competição. Sendo assim, foram realizados testes com ILS-INTRAK com o tempo limite de cinco horas. Esses resultados são apresentados na Tabela 6.7, juntamente com os melhores resultados encontrados para cada instância no endereço <http://satt.diegm.uniud.it/ctt/>. Os resultados ~~risca~~ são maiores ou iguais aos alcançados pela abordagem realizada

Através do endereço indicado acima, é possível ter acesso as instâncias, validadores de soluções, *lower bounds* e também cadastrar novas soluções geradas para cada instância. O sítio em questão apresenta também um ranking com as melhores soluções geradas para cada instância, de certa forma, a competição continua ocorrendo através desse endereço. Contudo, os resultados apresentados neste endereço já não são limitados pelo tempo computacional da competição, como pode ser visto nos trabalhos de Lubbecke e Lach (2010), Achá e Nieuwenhuis (2012), Abdullah et al. (2009) e Shaker et al. (2010).

Analisando a Tabela 6.7 é possível notar que o desempenho de um algoritmo varia muito a partir da instância que está sendo testada, o que é comprovado pelo fato de que em cada instância tem-se um *ranking* com diferentes colocações e competidores. Ressalta-se também que com a liberação do tempo computacional alguns proponentes resolveram o problema via métodos exatos, obtendo resultados satisfatórios em algumas instâncias.

Quanto aos resultados obtidos nesta comparação, pode-se destacar que: i) em três instâncias foram obtidos os resultados ótimos (*comp01*, *comp04* e *comp11*); ii) em quatro instâncias os resultados obtidos ficaram bem próximos dos melhores (*comp08*, *comp09*, *comp13* e *comp14*); iii) em apenas duas instâncias os resultados obtidos não ficaram entre os cinco melhores pesquisadores.

Por fim, pode-se afirmar que a implementação realizada apresentou certa eficiência nas duas comparações realizadas (respeitando o tempo limitante e desconsiderando o mesmo), ressalta-se que os melhores resultados obtidos foram validados e cadastrados em <http://satt.diegm.uniud.it/ctt/>. Sendo o problema em questão envolvido em uma competição internacional e posteriormente apresentado em

um *Benchmarking* proposto em De Cesco et al. (2010), vários pesquisadores têm atacado o problema pelo conjunto de recursos e dados que facilitam a análise e também o aperfeiçoamento de novas técnicas propostas. O que evidencia o fato de que a criação de uma nova técnica dita “aceitável” para resolução do problema é uma tarefa árdua, devido a quantidade e a qualidade das abordagens já construídas para esse fim.

Tabela 6.7: Resultados obtidos através do ILS-INTRAK, desprezando o tempo limite da competição, e, comparações com os melhores resultados encontrados para o problema.

INSTÂNCIA	1º	2º	3º	4º	5º	ILS-INTRAK
<i>comp01.ctt</i>	5 Andrea Schaerf	5 Tomas Muller	5 M. J. Geiger	5 Lu and Hao	5 A. Abuhamdah	5 -
<i>comp02.ctt</i>	24 Barcelogic Team	29 Lu and Hao	33 Gerald Lach	35 Tomas Muller	56 Andrea Schaerf	48 -
<i>comp03.ctt</i>	66 Tomas Muller	66 Lu and Hao	72 Jakub Marecek	79 Andrea Schaerf	82 S. Abdullah	83 -
<i>comp04.ctt</i>	35 Tomas Muller	35 Lu and Hao	35 Barcelogic Team	38 Andrea Schaerf	43 A. Abuhamdah	35 -
<i>comp05.ctt</i>	290 M. Muehlenthaler	291 Andrea Schaerf	292 Lu and Hao	298 Tomas Muller	318 S. Abdullah	360 -
<i>comp06.ctt</i>	27 Barcelogic Team	37 Tomas Muller	37 Lu and Hao	49 S. Abdullah	94 M. J. Geiger	61 -
<i>comp07.ctt</i>	6 Barcelogic Team	7 Tomas Muller	11 S. Abdullah	13 Lu and Hao	26 Andrea Schaerf	18 -
<i>comp08.ctt</i>	37 S. Abdullah	38 Barcelogic Team	39 Tomas Muller	42 Lu and Hao	42 Andrea Schaerf	39 -
<i>comp09.ctt</i>	96 Lu and Hao	99 Gerald Lach	100 Tomas Muller	103 S. Abdullah	104 Andrea Schaerf	101 -
<i>comp10.ctt</i>	4 Barcelogic Team	7 Tomas Muller	10 Lu and Hao	15 S. Abdullah	27 Andrea Schaerf	31 -
<i>comp11.ctt</i>	0 Andrea Schaerf	0 Tomas Muller	0 M. J. Geiger	0 Lu and Hao	0 S. Abdullah	0 -
<i>comp12.ctt</i>	300 Andrea Schaerf	305 Lu and Hao	310 Tomas Muller	341 S. Abdullah	430 M. J. Geiger	347 -
<i>comp13.ctt</i>	59 Lu and Hao	60 Gerald Lach	61 Tomas Muller	69 S. Abdullah	72 Andrea Schaerf	66 -
<i>comp14.ctt</i>	51 Gerald Lach	51 Lu and Hao	51 Barcelogic Team	57 S. Abdullah	57 Andrea Schaerf	57 -

Capítulo 7

CONCLUSÕES E TRABALHOS FUTUROS

Este trabalho realizou um estudo sobre o problema de alocação de horários em instituições educacionais, mais precisamente abordando o problema denominado Programação de Cursos Universitários baseada em Currículos (*Curriculum Based Course Timetabling*), sendo consideradas instâncias e definições da competição de *Timetabling*, ITC 2007. Foi realizada uma revisão bibliográfica e uma caracterização dos problemas relacionados ao tema, na qual aspectos como otimalidade, viabilidade e complexidade foram discutidos juntamente com as classificações existentes. A resolução de problemas de otimização via meta-heurísticas também foi discutida, destacando os principais conceitos relacionados as técnicas utilizadas nas implementações realizadas.

Foram cinco diferentes algoritmos construídos e aplicados na resolução de quatorze instâncias do problema, sendo as diferentes implementações comparadas e analisadas entre si. A implementação com melhor desempenho encontrado nesses experimentos computacionais, o ILS-INTRAK, foi também comparado com os resultados encontrados na literatura para o problema.

É importante salientar, que o processo de construção de um algoritmo que apresenta certa eficiência na resolução do problema em um tempo computacional tão pequeno, foi realizado a partir de diversos refinamentos nas implementações realizadas. Alguns desses refinamentos podem ser visualizados nesse trabalho na forma dos diferentes algoritmos apresentados, os quais podem ser ordenados de tal maneira que evidencia a evolução dos resultados obtidos.

Mediante os experimentos computacionais realizados, é possível afirmar que o objetivo geral foi alcançado, tendo este estudo uma implementação que combina diferentes aspectos heurísticos e de exploração de vizinhança, apresentando sim, certa eficiência na resolução do problema. Quanto aos objetivos específicos, pode-se dizer que foram alcançados através: i) da implementação da técnica de relaxação no ILS; ii) dos procedimentos Kempswap e Intra-Inter-Salas; iii) das comparações realizadas entre as diferentes técnicas que permitiram a análise a respeito dos diferentes comportamentos apresentados.

A utilização da técnica de relaxação no ILS apresentou grande melhora nos resultados obtidos, comprovando superioridade nas comparações com ILS convencional. Essa modificação realizada no critério de aceitação, permite que a solução corrente

seja alterada também para uma solução de piora, o que aumenta o poder da técnica de superar ótimos locais e também regiões planas. Um dos pontos complicados da técnica desenvolvida foi a calibração dos parâmetros, que apesar de poucos, possuem a difícil tarefa de permitir determinadas pioras no processo de busca, sem se perder por de mais distanciando-se das regiões promissoras.

Os procedimentos Kempeswap e Intra-Inter-Salas apresentam uma característica em comum, ambos são baseados em movimentos avançados que são caracterizados pelo maior custo computacional, compensado pela “inteligência” das movimentações realizadas nas soluções. O procedimento Kempeswap é composto por movimentos de cadeias Kempe representadas através de grafos, os quais são determinados pelas restrições de currículo, dessa forma, em um único movimento várias aulas podem ser alteradas sem alterar a factibilidade da solução e em alguns casos encontrar soluções ainda não encontradas. A eficiência do procedimento foi comprovada através do algoritmo ILS-KEMPE, sendo ressaltado que movimentos de cadeias Kempe vêm apresentando resultados consideráveis para diferentes problemas de distintas classes.

Seguindo a linha característica do procedimento Kempeswap de movimentos inteligentes, foi implementado o procedimento Intra-Inter-Salas, em que os resultados foram validados através da implementação ILS-INTRA. Trata-se de um procedimento que tem como sua principal característica a construção de um grafo com arestas valoradas, em que ciclos mínimos de custo negativo apontarão para várias realocações de aulas, que na maioria das vezes produzirá melhoras na solução.

Os procedimentos Kempeswap e Intra-Inter-Salas podem ser visualizados com diferentes características marcantes, o primeiro apresenta maior diversificação no processo de busca sem onerar por de mais o custo da nova solução. Já no segundo, a intensificação no processo de busca, garante a melhor alocação de aulas a partir de uma alocação de salas presente em uma solução prévia. Um outro ponto que deve ser destacado é que em ambas abordagens ocorre uma mudança de representação da solução (de matriz para grafos), pode ser observado como a exploração do espaço de busca está ligada com a representação adotada. Talvez em alguns casos uma troca de representação da solução aliada a uma nova busca local, pode ser suficiente para encontrar regiões não visitadas.

Pode-se afirmar que o procedimento Intra-Inter-Salas apresenta maior complexidade de execução e implementação, contudo, nas comparações realizadas apresentou melhores soluções que o procedimento Kempeswap. Mas, como os movimentos de cadeia Kempe também tem caráter positivo na resolução do problema, os respectivos foram utilizados na forma de perturbação na implementação ILS-INTRAK, sendo esta a implementação a mais eficiente entre as realizadas.

Outros aspectos com alguma relevância na abordagem realizada foram: a utilização de um construtor baseado em GRASP para geração de uma solução inicial; a descida randômica aleatória, que realiza uma busca local com certa eficiência e rapidez; os movimentos utilizados na exploração da vizinhança, dos quais alguns são específicos para determinadas restrições; as perturbações implementadas, destacando-se a perturbação desenvolvida em que parte da solução é destruída e em seguida construída com parte da técnica de geração de solução inicial.

Como trabalhos futuros podem ser destacados:

- Uma implementação da técnica de relaxação através de parâmetros adapta-

tivos, os quais favorecerão a relaxação em ótimos locais, e, amenizarão sua utilização a medida que a solução corrente se distâncie por de mais da melhor solução encontrada no processo de busca;

- Um estudo mais aprofundado sobre o impacto de diferentes representações da solução em um mesmo processo de busca;
- Novas abordagens a partir dos conceitos contidos nos procedimentos implementados. Principalmente, no que se refere aos movimentos Intra-salas e Inter-salas, abordagens em que são utilizados com mais frequência mas com alguma compensação no custo computacional ou então aliados a meta-heurísticas como Busca-Tabu;
- Aperfeiçoar o movimento Inter-Salas, favorecendo também a diversificação no procedimento Intra-Inter-Salas;

As principais contribuições desse trabalho consistem nas descrições sobre as implementações realizadas, principalmente as que apresentaram maior impacto sobre os resultados obtidos. Sendo a utilização de movimentos de cadeias Kempe uma técnica já consolidada, inclusive para o problema CBCT, pode-se afirmar que a utilização da relaxação no ILS e o procedimento Intra-Inter-Salas são aspectos promissores que apresentaram melhora significativa e poderão ser melhores explorados em novos trabalhos e também em outros problemas.

Por fim, cabe ressaltar que o modelo do problema CBCT juntamente com as instâncias consideradas, são de caráter geral e apresentam características presentes na maioria das universidades. Sendo assim, tendo as implementações certa eficiência na resolução do problema em questão, em trabalhos futuros, poderão ser aplicados em uma modelagem de problema inserida no contexto das universidades brasileiras.

Apêndice A

INSTÂNCIA EXEMPLO

É apresentado neste anexo o conteúdo do arquivo *comp01.ctt*. Trata-se de um arquivo-instância que é apresentado para exemplificar como as informações relativas ao problema são armazenadas e manipuladas para resolução do problema estudado.

O arquivo em questão apresenta o seguinte formato:

- Disciplinas: <disciplinaID> <professor> <numeroAulas> <minimosDiasTrabalhados> <numeroEstudantes>;
- Salas: <salaID> <capacidade>;
- Currículos: <currículoID> <numeroCurrículo> <disciplinaID> ... <disciplinaID>;
- Restrições de Indisponibilidade: <disciplinaID> <dia> <periodoDia>;

Segue o conteúdo do arquivo *comp01.ctt*:

Name: Fis0506-1
Courses: 30
Rooms: 6
Days: 5
Periods_per_day: 6
Curricula: 14
Constraints: 53

COURSES:

c0001 t000 6 4 130
c0002 t001 6 4 75
c0004 t002 7 3 117
c0005 t003 3 3 75
c0014 t004 1 1 65
c0015 t005 8 3 65
c0016 t006 7 3 65
c0017 t007 2 2 65
c0024 t008 4 3 55
c0025 t009 8 3 55
c0078 t010 5 4 55
c0030 t011 5 4 20

c0031 t012 5 4 11
c0032 t013 1 1 31
c0033 t014 6 4 31
c0057 t015 5 4 2
c0058 t016 5 4 2
c0059 t017 6 4 7
c0061 t018 6 4 6
c0062 t019 5 4 10
c0063 t020 6 4 8
c0064 t020 6 4 6
c0065 t021 6 4 5
c0066 t008 6 4 14
c0067 t022 5 4 7
c0068 t023 6 4 9
c0069 t007 6 4 7
c0070 t002 6 4 4
c0071 t001 6 4 10
c0072 t003 6 4 9

ROOMS:

rB 200
rC 100
rE 9
rF 30
rG 20
rS 30

CURRICULA:

q000 4 c0001 c0002 c0004 c0005
q001 4 c0014 c0015 c0016 c0017
q002 4 c0024 c0025 c0001 c0078
q003 3 c0030 c0032 c0033
q004 3 c0031 c0032 c0033
q005 3 c0070 c0066 c0072
q006 2 c0057 c0059
q007 2 c0061 c0062
q008 3 c0065 c0059 c0072
q009 4 c0063 c0064 c0066 c0071
q010 3 c0058 c0063 c0068
q011 3 c0069 c0067 c0068
q012 1 c0004
q013 3 c0062 c0066 c0071

UNAVAILABILITY_CONSTRAINTS:

c0001 4 0
c0001 4 1
c0001 4 2

c0001 4 3
c0001 4 4
c0001 4 5
c0004 0 0
c0004 0 1
c0004 0 2
c0004 0 3
c0004 0 4
c0004 0 5
c0024 3 0
c0024 3 1
c0024 3 2
c0024 3 3
c0024 3 4
c0024 3 5
c0025 2 2
c0025 2 3
c0025 2 4
c0025 2 5
c0025 3 0
c0025 3 1
c0025 3 2
c0025 3 3
c0025 3 4
c0025 3 5
c0033 3 2
c0033 3 3
c0033 3 4
c0033 3 5
c0033 4 0
c0033 4 1
c0033 4 2
c0033 4 3
c0033 4 4
c0033 4 5
c0071 0 0
c0071 0 1
c0071 0 2
c0071 1 0
c0071 1 1
c0071 1 2
c0071 2 0
c0071 2 1
c0071 2 2
c0071 3 0
c0071 3 1
c0071 3 2

c0071 4 0

c0071 4 1

c0071 4 2

END.

Apêndice B

SOLUÇÃO EXEMPLO

É apresentado neste anexo o conteúdo de um arquivo que contém uma solução para instância *comp01.ctt*. Este formato de representação de solução é utilizado para que as soluções geradas possam ser validadas através do mesmo validador da competição ITC 2007.

O arquivo em questão apresenta o seguinte formato:

Para cada alocação tem-se: <disciplinaID> <sala> <dia> <período>

Segue o conteúdo de um arquivo de solução *comp01.ctt* com função de avaliação igual a cinco:

```
c0001 rB 0 1
c0001 rB 0 2
c0001 rB 1 4
c0001 rB 1 5
c0001 rB 2 1
c0001 rB 3 4
c0002 rB 0 0
c0002 rB 1 2
c0002 rB 2 2
c0002 rB 4 2
c0002 rB 4 4
c0002 rB 4 5
c0004 rB 1 0
c0004 rB 1 1
c0004 rB 2 4
c0004 rB 2 5
c0004 rB 3 1
c0004 rB 3 2
c0004 rB 3 3
c0005 rB 2 3
c0005 rB 3 0
c0005 rB 4 3
c0014 rC 3 3
c0015 rB 0 3
c0015 rB 0 4
```

c0015 rB 0 5
c0015 rB 1 3
c0015 rB 2 0
c0015 rB 3 5
c0015 rB 4 0
c0015 rB 4 1
c0016 rC 1 4
c0016 rC 1 5
c0016 rC 2 1
c0016 rC 2 3
c0016 rC 2 4
c0016 rC 2 5
c0016 rC 3 2
c0017 rC 0 2
c0017 rC 3 4
c0024 rC 0 3
c0024 rC 0 4
c0024 rC 1 3
c0024 rC 4 0
c0025 rC 0 0
c0025 rC 1 0
c0025 rC 1 1
c0025 rC 1 2
c0025 rC 2 0
c0025 rC 4 3
c0025 rC 4 4
c0025 rC 4 5
c0078 rC 0 5
c0078 rC 2 2
c0078 rC 3 5
c0078 rC 4 1
c0078 rC 4 2
c0030 rS 0 0
c0030 rS 0 3
c0030 rS 1 1
c0030 rS 2 5
c0030 rS 3 2
c0031 rG 0 0
c0031 rG 1 4
c0031 rG 2 5
c0031 rG 4 0
c0031 rG 4 1
c0032 rS 1 3
c0033 rC 0 1
c0033 rS 0 2
c0033 rS 1 2
c0033 rS 2 4

c0033 rC 3 0
c0033 rC 3 1
c0057 rF 1 0
c0057 rF 2 0
c0057 rF 2 2
c0057 rF 3 0
c0057 rF 4 0
c0058 rG 1 3
c0058 rG 2 1
c0058 rG 3 1
c0058 rG 3 4
c0058 rG 4 5
c0059 rE 1 1
c0059 rE 1 2
c0059 rE 2 1
c0059 rE 2 3
c0059 rE 3 1
c0059 rE 4 1
c0061 rE 0 0
c0061 rE 1 3
c0061 rE 1 4
c0061 rE 1 5
c0061 rE 3 3
c0061 rE 4 3
c0062 rF 0 1
c0062 rF 0 2
c0062 rF 1 2
c0062 rF 3 2
c0062 rF 4 4
c0063 rG 0 1
c0063 rG 1 0
c0063 rG 1 2
c0063 rG 2 3
c0063 rG 3 0
c0063 rG 4 4
c0064 rF 0 3
c0064 rF 0 4
c0064 rF 0 5
c0064 rF 1 3
c0064 rF 3 3
c0064 rF 4 5
c0065 rE 0 3
c0065 rE 2 2
c0065 rE 2 5
c0065 rE 3 4
c0065 rE 3 5
c0065 rE 4 2

c0066 rF 0 0
c0066 rF 1 1
c0066 rF 1 4
c0066 rF 3 1
c0066 rF 4 1
c0066 rF 4 2
c0067 rS 0 4
c0067 rS 2 1
c0067 rS 3 1
c0067 rS 3 4
c0067 rS 4 1
c0068 rG 0 2
c0068 rG 0 3
c0068 rG 1 1
c0068 rG 2 2
c0068 rG 3 2
c0068 rG 3 3
c0069 rS 1 0
c0069 rS 2 0
c0069 rS 3 5
c0069 rS 4 2
c0069 rS 4 4
c0069 rS 4 5
c0070 rS 0 1
c0070 rS 1 5
c0070 rS 2 2
c0070 rS 2 3
c0070 rS 3 0
c0070 rS 4 0
c0071 rF 1 5
c0071 rF 2 4
c0071 rF 2 5
c0071 rF 3 4
c0071 rF 3 5
c0071 rF 4 3
c0072 rE 0 2
c0072 rE 0 4
c0072 rE 0 5
c0072 rE 1 0
c0072 rE 2 4
c0072 rE 3 2
END.

Apêndice C

PUBLICAÇÕES

São listadas neste apêndice as publicações produzidas ao decorrer deste estudo.

Título: Utilização de uma heurística híbrida para resolução do problema de alocação de horários em uma universidade.

Co-autores: Sérgio Ricardo de Souza e Marccone Jamilson Freitas Souza

Evento: XIII Encontro de Modelagem Computacional (EMC 2010)

Local: Nova Friburgo, RJ

Período: 03 a 05 de Novembro de 2010

Título: Resolução do Problema de Programação de Cursos Universitários Baseada em Currículos via uma meta-heurística híbrida GRASP-ILS-RELAXADO

Co-autores: Sérgio Ricardo de Souza e Marccone Jamilson Freitas Souza

Evento: XLIII Simpósio Brasileiro de Pesquisa Operacional

Local: Ubatuba, SP

Período: 15 a 18 de agosto de 2011

Título: Programação de Cursos Universitários Baseada em Currículos via uma meta-heurística híbrida GRASP-ILS utilizando o procedimento Kempeswap

Co-autores: Sérgio Ricardo de Souza e Marcone Jamilson Freitas Souza

Evento : XIV Simpósio de Pesquisa Operacional e Logística da Marinha

Local : Rio de Janeiro, RJ

Período 15 a 16 de setembro de 2011

Título: Programação De Cursos Universitários Baseada Em Currículos via uma meta-heurística híbrida GRASP-ILS-TABU

Co-autores: Sérgio Ricardo de Souza e Marcone Jamilson Freitas Souza

Evento: 32º Congresso Ibero Latino Americano de Métodos Computacionais em Engenharia

Local: Ouro Preto, MG

Período: 13 a 16 de novembro de 2011

Referências Bibliográficas

Abbass, H. e Newton, C. (2001). *Heuristic and Optimization for Knowledge Discovery*. IGI Global, Hershey.

Abdullah, Salwani; Turabieh, Hamza; Mccollum, Barry e Burke, Edmund K. (2009). An investigation of a genetic algorithm and sequential local search approach for curriculum-based course timetabling problems. *Multidisciplinary International Conference on Scheduling : Theory and Applications (MISTA 2009)*.

Achá, Roberto Asín e Nieuwenhuis, Robert. (2012). Curriculum-based course timetabling with sat and maxsat. *Annals of Operations Research*, p. 1–21.

Bai, R.; Burke, E. K.; Kendall, G. e Mccullum, B. (2008). A simulated annealing hyper-heuristic for university course timetabling problem extended abstract. (*Abstract*) *PATAT '06, Proceedings of the 6th International Conference on the Practice and Theory of Automated Timetabling*, p. 345–350.

Barbosa, S. H. D.; Souza, S. R. e Sousa, A. M. (2010). Utilização de uma heurística híbrida para resolução do problema de alocação de horários em uma universidade. *XIII Encontro de Modelagem Computacional (EMC 2010), 2010, Nova Friburgo - RJ. Anais do XIII Encontro de Modelagem Computacional.*, (2010).

Barbosa, S H D.; Souza, S. R e Souza, M. J. F. (2011)a. Programação de cursos universitários baseada em currículos via uma meta-heurística híbrida grasp-ils-tabu. *XXXII Congresso Ibero Latino Americano de Métodos Computacionais em Engenharia*, (2011)a.

Barbosa, S H D.; Souza, S. R e Souza, M. J. F. (2011)b. Programação de cursos universitários baseada em currículos via uma meta-heurística híbrida grasp-ils utilizando o procedimento kempeswap. *XIV Simpósio de Pesquisa Operacional e Logística da Marinha*, (2011)b.

Barbosa, S H D.; Souza, S. R e Souza, M. J. F. (2011)c. Resolução do problema de programação de cursos universitários baseada em currículos via uma meta-heurística híbrida grasp-ils-relaxado. *XLIII Simpósio Brasileiro de Pesquisa Operacional*, (2011)c.

Blum, C. e Roli, A. September(2003). Metaheuristics in combinatorial optimization: Overview and conceptual comparison. *ACM Comput. Surv.*, v. 35, p. 268–308.

Burke, E.; Jackson, K.; Kingston, J. e Weare, R. (1997). Automated university timetabling: The state of the art. *The Computer Journal*, v. 40, p. 565–571.

- Burke, E. e Silva, J. L. (2005). The design of memetic algorithms for scheduling and timetabling problems. Hart, William; Smith, J. e Krasnogor, N., editors, *Recent Advances in Memetic Algorithms*, volume 166 of *Studies in Fuzziness and Soft Computing*, p. 289–311. Springer Berlin / Heidelberg, (2005).
- Burke, E. K.; Causmaecker, P. e Landeghem, H. V. (2004). The state of the art of nurse rostering. *Journal of Scheduling*, v. 7, p. 441–499.
- Burke, E. K. e Newall, J. P. (1999). A multistage evolutionary algorithm for the timetable problem. *IEEE Transactions on Evolutionary Computation*, v. 3, p. 63–74.
- Burke, E. K. e Petrovic, S. (2002). Recent research directions in automated timetabling. *European Journal of Operational Research*, v. 140, p. 266–280.
- Carter, M. W. e Laporte, G. (1997). Recent developments in practical course timetabling. *Practice and Theory of Automated Timetabling*, p. 3–19, (1997).
- Chiarandini, M.; Birattari, M.; Socha, K. e Rossi-Doria, O. October(2006). An effective hybrid algorithm for university course timetabling. *J. of Scheduling*, v. 9, p. 403–432.
- Cooper, T. e Kingston, J. (1996). The complexity of timetable construction problems. Burke, Edmund e Ross, Peter, editors, *Practice and Theory of Automated Timetabling*, volume 1153 of *Lecture Notes in Computer Science*, p. 281–295. Springer Berlin / Heidelberg.
- Cormen, T. H.; Leiserson, C. E. e Rivest, R. L. (1990). *Introduction to Algorithms*. MIT Press and McGraw-Hill, 1 edição.
- De Cescio, F.; Di Gaspero, L. e Schaerf, A. (2010). Benchmarking curriculum-based course timetabling: formulations, data formats, instances, validation, visualization, and results. *Annals of Operations Research*, p. 1–12.
- deWerra, D. (1985). An introduction to timetabling. *European Journal of Operational Research*, v. 19, p. 151–162.
- deWerra, D. (1997). Restricted coloring models for timetabling. *Discrete Mathematics*, v. 166, p. 161 – 170.
- Deb, K. e Talbi, E. (2001). Multi-objective optimization using evolutionary algorithms. *JSAI Workshops*, (2001).
- Di Gaspero, L. e Schaerf, A. (2007). The second international timetabling competition (itc 2007): Curriculum-based course timetabling (track 3).
- Dorigo, M.; Blum, C. e Talbi, E. (2005). Ant colony optimization theory: A survey. *Theoretical Computer Science*, v. 344, p. 243–278.
- Dowsland, K. A. (1993). Simulated Annealing. Reeves, C.R., editor, *Modern Heuristic Techniques for Combinatorial Problems*, Advanced Topics in Computer Science Series, Capítulo 2, p. 20–69. Blackwell Scientific Publications, London.

- Dueck, G. (1993). New optimization heuristics the great deluge algorithm and the record-to-record travel. *Journal of Computational Physics*, v. 104, p. 86–92.
- Eikelder, H. M. M. ten e Willemsen, R. J. (2001). Some complexity aspects of secondary school timetabling problems. *Selected papers from the Third International Conference on Practice and Theory of Automated Timetabling III*, PATAT '00, p. 18–27, London, UK. Springer-Verlag.
- Feo, T. A. e Resende, M. G. C. (1995). Greedy randomized adaptive search procedures. v. 6, p. 109–133.
- Floyd, R. W. (1962). Algorithm 97: Shortest path. *Communications of the ACM*, (1962).
- Gaspero, L. D. e Schaerf, A. (2006). Neighborhood portfolio approach for local search applied to timetabling problems. *Journal of Mathematical Modelling and Algorithms*, v. 5, p. 65–89.
- Geiger, M. J. (2008). An application of the threshold accepting metaheuristic for curriculum based course timetabling. *In Proceedings of the 7th International Conference on the Practice and Theory of Automated Timetabling, PATAT*, p. 19–22.
- Geiger, M. J. (2010). Applying the threshold accepting metaheuristic to curriculum based course timetabling. *Annals of Operations Research*, p. 1–14.
- Glover, F. (1989). Tabu search - part i. *ORSA Journal on Computing*, v. 1, n. 3, p. 190–206.
- Glover, F. e Kochenberger, G. (2003). *Handbook of Metaheuristics*. Springer.
- Glover, F. e Laguna, M. (1997). *Tabu Search*. Kluwer Academic Publishers, Boston.
- Glover, Fred. May(1986). Future paths for integer programming and links to artificial intelligence. *Comput. Oper. Res.*, v. 13, p. 533–549.
- Gotlieb, C.C. (1963). The construction of class-teacher timetables. *of IFIP Congress*, p. pp. 73–77, Amsterdam.
- Hamiez, J. e Hao, J. K. (2001). Solving the sports league scheduling problem with tabu search. *European Conference on Artificial Intelligence*, p. 24–36, (2001).
- Hansen, P. e Mladenović, N. (2001). Variable Neighborhood Search: Principles and Applications. *European Journal of Operational Research*, v. 130, p. 449–467.
- Hao, J. e Benlic, U. (2011). Lower bounds for the itc-2007 curriculum-based course timetabling problem. *European Journal of Operational Research*, v. 212, n. 3, p. 464 – 472.
- Hertz, A. e Widmer, M. (2003). Guidelines for the use of meta-heuristics in combinatorial optimization. *European Journal of Operational Research Society*, v. 151, p. 247–252.

- Irene, H. S. F.; Deris, S.; Hashim, M. e Zaiton, S. (2009). University course timetable planning using hybrid particle swarm optimization. *Proceedings of the first ACM/SIGEVO Summit on Genetic and Evolutionary Computation*, p. 239–246, (2009).
- Kwan, R. (2004). Bus and train driver scheduling. *J. Leung (Ed.), Handbook of scheduling: algorithms, models, and performance.*, p. Chap. 51.
- Lü, Z. e Hao, J. (2010). Adaptive tabu search for course timetabling. *European Journal of Operational Research*, v. 200, n. 1, p. 235 – 244.
- Lewis, R. (2008). A survey of metaheuristic-based techniques for university timetabling problems. *OR Spectrum*, v. 30, p. 167–190.
- Lourenço, H. R.; Martin, O. e Stützle, T. (2003). Iterated Local Search. Glover, F. e Kochenberger, G., editors, *Handbook of Metaheuristics*, p. 321–353. Kluwer Academic Publishers.
- Lu, Z. e Hao, J. (2008). Solving the course timetabling problem with a hybrid heuristic algorithm. *Artificial Intelligence: Methodology, Systems, Applications*, p. 262–273, (2008).
- Lü, Z.; Hao, J. e Glover, F. April(2011). Neighborhood analysis: a case study on curriculum-based course timetabling. *Journal of Heuristics*, v. 17, p. 97–118.
- Lubbecke, M.E. e Lach, G. (2010). Curriculum based course timetabling: new solutions to udine benchmark instances. *Annals of Operations Research*, (2010).
- Melanie, M. (1998). *An Introduction to Genetic Algorithms*. The MIT Press.
- Moscato, P. (1999). Memetic algorithms: a short introduction. *Dublin Core Conference*, (1999).
- Muller, T. *Constraint-based Timetabling*. PhD thesis, Charles University in Prague Faculty of Mathematics and Physics, (2005).
- Muller, T. (2008). Its 2007 solver description: A hybrid approach. *In Proceedings of the 7th International Conference on the Practice and Theory of Automated Timetabling, PATAT*.
- Muller, T.; Barták, R. e Rudova, H. (2004). Conflict-based statistics. *In EU/ME Workshop on Design and Evaluation of Advanced Hybrid Meta-Heuristics*, (2004).
- Osman, I H e Kelly, J P. (1997). Meta-heuristics theory and applications. *Journal of The Operational Research Society*, v. 48, p. 657–657.
- Qu, R. *Case Based Reasoning for Course Timetabling Problems*. PhD thesis, University of Nottingham, (2002).
- Qu, R.; Burke, E.; McCollum, B.; Merlot, L. e Lee, S. (2009). A survey of search methodologies and automated system development for examination timetabling. *Journal of Scheduling*, v. 12, p. 55–89.

- Reis, L. P. e Oliveira, E. (2000). A language for specifying complete timetabling problems. *PATAT2000 Proceedings, August 2000. Burke and Erben*, p. 322–341. Springer, (2000).
- Santos, H. e Souza, M. (2007). Programação de horários em instituições educacionais: Formulações e algoritmos. *Anais do XXXIX Simpósio Brasileiro de Pesquisa Operacional - SBPO, Fortaleza*, p. 2827–2882.
- Santos, H. G. (2007). Formulações e algoritmos para o problema de programação de horários em escolas. Tese de doutorado, Universidade Federal Fluminense, Niterói, RJ.
- Schaerf, A. (1999). A survey of automated timetabling. *Artif. Intell. Rev*, v. 13, n. 2, p. 87–127.
- Selman, Bart; Levesque, Hector J. e Mitchell, D. (1992). A New Method for Solving Hard Satisfiability Problems. *Proceedings of the Tenth National Conference on Artificial Intelligence*, p. 440–446, Menlo Park, California. AAAI Press.
- Shaker, Khalid; Abdullah, Salwani; McCollum, Barry e McMullan, Paul. (2010). Incorporating great deluge with kempe chain neighbourhood structure for the enrolment-based course timetabling problem. *Rough Set and Knowledge Technology*, volume 6401 of *Lecture Notes in Computer Science*, p. 70–77. Springer Berlin / Heidelberg.
- Souza, M. J. F. (2000). Programação de horários em escolas: uma aproximação por metaheurísticas. Tese de doutorado, Programa de Engenharia de Sistemas e Computação, COPPE/UFRJ, Rio de Janeiro.
- Talbi, E. (2009). *Metaheuristics: from design to implementation*, volume 10. John Wiley & Sons.
- Thompson, J. M. e Dowsland, K. A. July(1998). A robust simulated annealing based examination timetabling system. *Comput. Oper. Res.*, v. 25, p. 637–648.
- Willemsen, R. J. (2002). School timetable construction algorithms and complexity. Master's thesis, Technische Universiteit Eindhoven.
- Wren, A. (1996). Scheduling, timetabling and rostering - a special relationship. *Practice and Theory of Automated Timetabling*, volume 1153 of *Lecture Notes in Computer Science*, p. 46–75. Springer Berlin / Heidelberg.