

UNIVERSIDADE FEDERAL RURAL DO RIO DE JANEIRO  
INSTITUTO MULTIDISCIPLINAR

LEONARDO DUARTE DE ALMEIDA DE  
AZEREDO

**Abordagens híbridas em  
meta-heurísticas para o problema de  
programação horária de cursos  
universitários: um estudo de caso**

Prof. Adria Ramos de Lyra, D.Sc.  
Orientador

Nova Iguaçu, Dezembro de 2018

**Abordagens híbridas em meta-heurísticas para o problema de  
programação horária de cursos universitários: um estudo de  
caso**

**Leonardo Duarte de Almeida de Azeredo**

Projeto Final de Curso submetido ao Departamento de Ciência da Computação do Instituto Multidisciplinar da Universidade Federal Rural do Rio de Janeiro como parte dos requisitos necessários para obtenção do grau de Bacharel em Ciência da Computação.

Apresentado por:

Leonardo D. A. de Azeredo

Leonardo Duarte de Almeida de Azeredo

Aprovado por:

Luiz Fernando R. Oliveira

Prof. Carlos Eduardo Ribeiro de Mello, D.Sc.

Juliana Mendes Nascente e Silva Zamith

Prof. Juliana Mendes Nascente e Silva Zamith, D.Sc.

Luiz Fernando dos Reis de Oliveira

Prof. Luiz Fernando dos Reis de Oliveira, M.Sc.

Marcelo Panaro de Moraes Zamith

Prof. Marcelo Panaro de Moraes Zamith, D.Sc.

NOVA IGUAÇU, RJ - BRASIL

Dezembro de 2018

# Agradecimentos

Aos meus pais, cujos os sacrifícios próprios e lutas na vida foram fundamentais para que eu pudesse chegar a qualquer lugar hoje. Obrigado, por tudo.

À professora Adria, pelo apoio e paciência durante esses anos e por acreditar na qualidade do meu trabalho mesmo quando eu mesmo duvidava.

Aos meus amigos pelo incentivo nos momentos de desânimo, em especial aqueles que além de incentivo também dividiram esses mesmos momentos.

## RESUMO

Abordagens híbridas em meta-heurísticas para o problema de programação horária

de cursos universitários: um estudo de caso

Leonardo Duarte de Almeida de Azeredo

Dezembro/2018

Orientador: Adria Ramos de Lyra, D.Sc.

Um problema comum em todas as instituições de ensino do mundo e recorrente a todo período letivo é a definição de grades horárias dos encontros entre professores e alunos. Tal problema, conhecido como Problema de Programação Horária, é NP-difícil mas ainda é majoritariamente tratado de forma manual pelas instituições, levando a desperdício de recursos e frequentemente soluções de baixa qualidade. Este trabalho apresenta o problema no contexto de cursos universitários e desenvolve um estudo de caso com a aplicação de técnicas meta-heurísticas, vastamente usadas na literatura para a solução do problema devido sua complexidade e o desempenho geral superior que tem sido observado das metodologias de busca local em relação a outras. O desempenho destes métodos é avaliado com instâncias reais do Departamento de Ciência da Computação da UFRRJ considerando suas necessidades específicas, como a valorização das preferências dos professores por disciplinas. As técnicas *Simulated annealing* e estratégias híbridas de GRASP+VND e VNS+VND foram experimentadas e, com base nos resultados, outras hibridizações foram investigadas. Os resultados mostram que no estudo de caso os métodos implementados foram capazes de superar facilmente a qualidade das soluções geradas manualmente, segundo os critérios usados pelo departamento, em uma fração do tempo.

## ABSTRACT

Abordagens híbridas em meta-heurísticas para o problema de programação horária

de cursos universitários: um estudo de caso

Leonardo Duarte de Almeida de Azeredo

Dezembro/2018

Advisor: Adria Ramos de Lyra, D.Sc.

*A common problem in all educational institutions in the world and recurrent to every school year is the definition of the timetables with encounters between teachers and students. Such problem, known as timetabling problem, is NP-Hard but still is mostly tackled manually by the schools, leading to waste of resources and frequently low quality solutions. This work present the problem in the context of university courses and develop a case study with the application of metaheuristics techniques, vastly used in the literature for solving the problem because of its complexity and the superior general performance that has been observed of the local search methodologies in relation to others. The performance of these methods is evaluated with real instances from the Departamento de Ciência da Computação of UFRRJ considering its specific needs such as the valorization of the teachers preferences for disciplines. The technique Simulated annealing and hybrid strategies of GRASP+VND and VNS+VND were experimented and, based on the results, other hybridizations were investigated. The results show that in the study case the methods implemented were able to easily surpass the quality of the solutions generated manually, according to the criteria used by the department, in a fraction of the time.*

# **Lista de Figuras**

Figura A.1: Tela principal. Manipulação de arquivos com os dados das instâncias. . . . .	83
Figura A.2: Tela principal. Acesso à edição de dados da instância. . . . .	83
Figura A.3: Tela principal. Acesso à configuração de restrições de tempo. . . . .	84
Figura A.4: Tela principal. Acesso às funções de geração de grades, configurações das estratégias e soluções geradas. . . . .	84
Figura A.5: Lista de professores. . . . .	85
Figura A.6: Edição das preferências de um professor por disciplinas. . . . .	86
Figura A.7: Lista de atividades da instância. . . . .	87
Figura A.8: Edição/Inclusão de uma atividade. . . . .	87
Figura A.9: Lista de todas as restrições da instância. . . . .	88
Figura A.10: Exemplo de edição/inclusão: restrição do mínimo número de dias entre atividades. . . . .	89
Figura A.11: Exemplo de edição/inclusão: restrição sobre horários não disponíveis para um grupo de alunos. . . . .	90
Figura A.12: Interface de configuração e seleção das estratégias de geração. . . .	91
Figura A.13: Interface para a geração das grades horárias (durante o processo de geração). . . . .	91

Figura A.14:Interface para a geração das grades horárias (geração concluída) . . .	92
Figura A.15:Lista de soluções geradas. . . . .	92
Figura A.16:Descrição detalhada dos conflitos e custo de uma solução. . . . .	93
Figura A.17:Visualização de uma grade horária gerada (perspectiva dos grupos de alunos). . . . .	94
Figura A.18:Visualização de uma grade horária gerada (perspectiva dos pro- fessores). . . . .	95
Figura A.19:Visualização de uma grade horária parcical para alocação manual. .	96
Figura A.20:Entrada manual de uma alocação. . . . .	97

# Listas de Tabelas

Tabela 3.1: Um exemplo de identificadores das horas de uma semana . . . . .	22
Tabela 3.2: Representação interna de uma solução com quatro atividades ativas	22
Tabela 4.1: Dimensões das instâncias . . . . .	41
Tabela 4.2: Professores afetados nas restrições extras . . . . .	44
Tabela 4.3: Penalidades individuais por par ( <i>professor, disciplina</i> ) . . . . .	45
Tabela 4.4: Fatores de violação do grupo essencial . . . . .	46
Tabela 4.5: Fatores de violação do grupo importante . . . . .	47
Tabela 4.6: Fatores de violação do grupo desejável . . . . .	47
Tabela 4.7: Custos das soluções manuais para as instâncias. . . . .	48
Tabela 4.8: Legenda dos valores nas tabelas de resultados. . . . .	49
Tabela 4.9: Resultados médios de construções com 100 iterações. Os três melhores valores de cada métrica por coluna de média geral foram grifados. . . . .	51
Tabela 4.10: Resultados médios de construções com 1000 iterações. O melhor valor de cada métrica por coluna de média geral foi grifado. . . . .	52
Tabela 4.11: Resultados médios GRASP + VND com <i>best improving</i> e 100 iterações. Os três melhores valores de cada métrica por coluna de média geral foram grifados. . . . .	53

Tabela 4.12: Resultados médios de GRASP+VND com <i>best improving</i> com 1000 iterações. O melhor valor de cada métrica por coluna de média geral foi grifado. . . . .	55
Tabela 4.13: Resultados médios GRASP + VND com <i>first improving</i> e 100 iterações. Os três melhores valores de cada métrica por coluna de média geral foram grifados. . . . .	56
Tabela 4.14: Resultados médios de GRASP+VND com <i>first improving</i> com 1000 iterações. O melhor valor de cada métrica por coluna de média geral foi grifado. . . . .	57
Tabela 4.15: Resultados médios das execuções de GRASP+VND com construção por grupos e busca local <i>best improving</i> . . . . .	59
Tabela 4.16: Resultados médios do VNS Versão A (perturbação simples em vizinhanças consecutivas) partindo da melhor solução entre 100 aleatórias e realizando VND com <i>best improving</i> . O melhor valor de cada métrica por coluna de média geral foi grifado. . . . .	60
Tabela 4.17: Resultados médios do VNS Versão A (perturbação simples em vizinhanças consecutivas) partindo da melhor solução entre 100 aleatórias e realizando VND com <i>first improving</i> . O melhor valor de cada métrica por coluna de média geral foi grifado. . . . .	60
Tabela 4.18: Resultados médios do VNS Versão B (perturbação incremental em vizinhança aleatória) partindo da melhor solução entre 100 aleatórias e realizando VND com <i>best improving</i> . O melhor valor de cada métrica por coluna de média geral foi grifado. . . . .	61
Tabela 4.19: Resultados médios do VNS Versão B (perturbação incremental em vizinhança aleatória) partindo da melhor solução entre 100 aleatórias e realizando VND com <i>first improving</i> . O melhor valor de cada métrica por coluna de média geral foi grifado. . . . .	61

Tabela 4.20: Resultados médios do VNS Versão A partindo da melhor solução entre 100 aleatórias, realizando VND com <i>best improving</i> . . . . .	62
Tabela 4.21: Resultados médios do VNS Versão A partindo da melhor solução entre 100 aleatórias, realizando VND com <i>first improving</i> . . . . .	62
Tabela 4.22: Resultados médios <i>Simulated annealing</i> a partir da melhor solução entre 100 aleatórias. O melhor valor de cada métrica por coluna de média geral foi grifado. . . . .	63
Tabela 4.23: Resultados médios de VNS versão B com 50 iterações sem melhora aplicado à soluções produzidas por GRASP com construção por grupos de estudantes e VND na busca local. O menor valor de cada métrica por ambas as colunas de médias foram grifados. . . . .	66
Tabela 4.24: Resultados médios de VSN versão B com <i>best improving</i> com solução inicial gerada pela fase construtiva do método GRASP. . . . .	67
Tabela 4.25: Resultados médios de VSN versão B com <i>first improving</i> com solução inicial gerada pela fase construtiva do método GRASP. . . . .	68
Tabela 4.26: Resultados médios de VSN versão B com solução inicial gerada pelo método <i>Simulated annealing</i> . O melhor valor de cada métrica por coluna de média geral foi grifado. A métrica tempo não inclui o tempo de execução do <i>Simulated annealing</i> . . . . .	69
Tabela 4.27: Resultados médios das execuções do método GRASP usando construção por grupos e VNS A com <i>first improving</i> na fase de busca local. . . . .	71
Tabela 4.28: Resultados médios das melhores estratégias, recapitulados. Estratégias que atingiram viabilidade média não nula. . . . .	73
Tabela 4.29: Resultados médios das melhores estratégias, recapitulados. Estratégias que atingiram viabilidade total. . . . .	74

# **Lista de Algoritmos**

1	GRASP . . . . .	25
2	CONSTRUÇÃO A PARTIR DE GRUPOS DE ALUNOS . . . . .	28
3	CONSTRUÇÃO A PARTIR DAS ATIVIDADES . . . . .	30
4	GRASP REATIVO . . . . .	32
5	VNS COM PERTURBAÇÃO SIMPLES EM MÚLTIPLAS VIZINHANÇAS . .	34
6	VNS COM PERTURBAÇÃO INCREMENTAL EM VIZINHANÇAS . . . .	35
7	VND . . . . .	36
8	SIMULATED ANNEALING . . . . .	38

# **Lista de Abreviaturas e Siglas**

AGPL	GNU Afferro General Public License
DCC	Departamento de Ciéncia da Computaçao
FET	Free Timetabling Software
GRASP	Greedy Randomized Adaptive Search Procedures
RGRASP	Reactive Greedy Randomized Adaptive Search Procedures
GNU	GNU's Not Unix
LC	Lista de Candidatos
LRC	Lista Restrita de Candidatos
PPH	Problema de Programação de Horários
PPHU	Problema de Programação de Horários de Cursos Universitários
SA	Simulated Annealing
UFRRJ	Universidade Federal Rural do Rio de Janeiro
VND	Variable Neighborhood Descent
VNS	Variable Neighborhood Search

# Sumário

<b>Agradecimentos</b>	i
<b>Resumo</b>	ii
<b>Abstract</b>	iii
<b>Lista de Figuras</b>	iv
<b>Lista de Tabelas</b>	vi
<b>Lista de Abreviaturas e Siglas</b>	x
<b>1 Introdução</b>	1
1.1 Motivação . . . . .	1
1.2 Objetivo . . . . .	2
1.3 Estrutura . . . . .	3
<b>2 Revisão da Literatura</b>	4
2.1 Problemas de Programação de Horários . . . . .	4
2.2 Problemas de Programação de Horários e o contexto de Instituições de Ensino . . . . .	5

2.3	PPH com múltiplos objetivos: possíveis modelagens . . . . .	8
2.4	Abordagens na literatura . . . . .	9
2.4.1	Trabalhos Relacionados . . . . .	10
<b>3</b>	<b>Metodologia</b>	<b>15</b>
3.1	A ferramenta FET . . . . .	15
3.1.1	Recursos herdados . . . . .	16
3.1.2	Conceitos, terminologias e adaptações . . . . .	16
3.2	Construção de soluções . . . . .	18
3.3	Busca Local . . . . .	18
3.3.1	Método de descida ( <i>Best improving</i> ) . . . . .	18
3.3.2	Método randômico de descida ( <i>First improving</i> ) . . . . .	19
3.3.3	Meta-heurísticas na busca local . . . . .	19
3.4	Modelagem do problema . . . . .	19
3.4.1	Fatores de violação de restrições . . . . .	19
3.4.2	Grupos de restrições e função objetivo . . . . .	20
3.4.3	Função objetivo . . . . .	21
3.4.4	Representação da solução . . . . .	21
3.4.5	Estruturas de vizinhança . . . . .	22
3.4.5.1	<i>Troca dos horários entre pares de atividades de um professor no mesmo dia (N1)</i> . . . . .	23
3.4.5.2	<i>Troca dos horários entre pares de atividades de um professor em dias distintos (N2)</i> . . . . .	23

3.4.5.3	<i>Troca de professores entre sub-atividades que pertencem a um mesmo grupo (N3)</i>	23
3.4.5.4	<i>Troca do professor em uma atividade (N4)</i>	23
3.4.5.5	<i>Troca de professores entre atividades (N5)</i>	23
3.4.5.6	Lidando com grupos de atividades	24
3.5	Métodos implementados	24
3.5.1	GRASP	24
3.5.2	Métodos de Construção	25
3.5.2.1	Versão A: Por grupo de alunos	27
3.5.2.2	Versão B: Por atividades	29
3.5.3	GRASP Reativo	31
3.5.4	VNS	33
3.5.4.1	Versão A: Perturbação simples, vizinhanças em ordem	33
3.5.4.2	Versão B: Perturbação de tamanho incremental em vizinhança aleatória	34
3.5.5	VND	35
3.5.6	<i>Simulated Annealing</i>	37
3.5.7	Explorando o paralelismo dos procedimentos de busca multi-início	38
<b>4</b>	<b>Estudo de caso: DCC/UFRRJ</b>	<b>40</b>
4.1	Descrição geral das instâncias baseadas no departamento	40
4.2	Apuração das preferências dos professores por disciplinas e requisitos	41
4.3	Restrições estudadas	42

4.3.1	Restrições gerais a todas as instâncias . . . . .	42
4.3.2	Restrições específicas a cada instância . . . . .	44
4.3.3	Atendimento às preferência dos professores em forma de restrição	44
4.3.4	Cálculo dos fatores de violação das restrições selecionadas . .	46
4.4	Experimentos e resultados . . . . .	47
4.4.1	Ambiente computacional . . . . .	47
4.4.2	Pesos da função objetivo . . . . .	48
	4.4.2.1 Custo das soluções manuais reais . . . . .	48
4.4.3	Características gerais do experimentos . . . . .	48
4.4.4	Experimentos exploratórios . . . . .	49
	4.4.4.1 Métodos de construção . . . . .	50
	4.4.4.2 GRASP+VND . . . . .	52
	4.4.4.3 VNS Versão A/VNS Versão B . . . . .	59
	4.4.4.4 <i>Simulated annealing</i> . . . . .	62
	4.4.4.5 Comparação e análise . . . . .	64
4.4.5	Experimentos refinados . . . . .	65
	4.4.5.1 GRASP+VND seguida de VNS+VND . . . . .	66
	4.4.5.2 Construção GRASP seguida de VNS+VND . . . .	66
	4.4.5.3 <i>Simulated annealing</i> seguido de VNS+VND . . . .	68
	4.4.5.4 GRASP+VNS+VND . . . . .	69
	4.4.5.5 Comparação e análise . . . . .	72
<b>5</b>	<b>Conclusão e Trabalhos Futuros</b>	<b>75</b>

<b>Referências</b>	<b>77</b>
--------------------	-----------

<b>A A ferramenta de software</b>	<b>82</b>
-----------------------------------	-----------

# Capítulo 1

## Introdução

Os problemas de programação horária (PPH) podem ser definidos como a alocação de uma sequência de elementos à um conjunto de recursos, nos limites de um período prefixado de tempo, satisfazendo determinadas restrições e potencialmente otimizando alguma função matemática [4]. Quando aplicado ao contexto de instituições de ensino o PPH consiste na programação dos encontros (aulas) entre professores e estudantes em ciclos de tempo tipicamente de uma semana [33].

### 1.1 Motivação

A solução manual do PPH normalmente requer grande consumo de recursos de pessoal/tempo e além disso pode ser muito insatisfatória com respeito às restrições e critérios de qualidade de cada instituição [33]. Segundo [41] o problema de alocação de horários escolares é recorrente a todo período letivo nas instituições de ensino e torna-se cada vez mais complexo a medida que aumenta o número de variáveis envolvidas, podendo chegar ao ponto de torná-lo de difícil solução para métodos exatos, devido à explosão combinatória.

A maioria das instituições ainda hoje realiza a elaboração das grades horárias de forma manual, o que se mostra um trabalho custoso e ineficiente e acaba envolvendo grande parte dos funcionários da instituição [15], levando assim a um grande

desperdício de recursos. Soluções computacionais automatizadas podem levar a resultados qualitativamente melhores além de reduzir o número de pessoas envolvidas no processo e o tempo gasto.

Além disso, [26] afirma que a solução obtida por métodos manuais pode ser insatisfatória com relação a vários aspectos sob a perspectiva dos envolvidos na confecção da grade. Um possível exemplo disso é que pode-se julgar interessante que aulas de uma mesma matéria não sejam ministradas em dias consecutivos (devido a possíveis prejuízos na sedimentação do conteúdo) e com a confecção manual da programação isso pode ser difícil de garantir.

Para [5], no contexto das universidades o problema de alocação de horários adquire uma complexidade maior já que a maioria dos professores possuem grande número de opções de disciplinas que estes podem ministrar (sendo, por vezes, possível que todas as disciplinas sejam teoricamente possíveis) e ainda podem ocorrer mudanças a cada novo período. Nesse cenário o problema pode ser resolvido com foco a favorecer as preferências de cada professor.

Por essas razões esse problema, que pertence à classe NP-difícil, frequentemente é trabalhado através de técnicas heurísticas, entre as quais se destacam na literatura aquelas que são chamadas de meta-heurísticas. Tais técnicas apresentam a característica de serem gerais e possuírem mecanismos desenhados para escapar de ótimos locais [26], podendo serem adaptadas a múltiplos problemas.

## 1.2 Objetivo

O presente trabalho tem como objetivo principal adaptar diferentes técnicas meta-heurísticas (um subconjunto selecionado da literatura em trabalhos que lidam com problemas semelhantes) e elaborar uma abordagem de automatização na busca de soluções do PPHU (Problema de Programação de Horários de Cursos em Universidades) com a maximização das preferências dos professores por disciplinas que desejem ministrar, tomando como foco o estudo de caso do Departamento de Ciência da Computação da Universidade Federal Rural do Rio de Janeiro.

Simultaneamente, este trabalho também tem com objetivo a produção de uma ferramenta de software de fácil utilização que seja capaz de atender às necessidades do DCC/UFRRJ na alocação semestral de sua grade horária, além de permitir adaptações customizáveis nas restrições e nos critérios de qualidade para as soluções de maneira simplificada por seus usuários. Tal ferramenta é desenvolvida a partir de um software de código aberto pré-existente ([21]), que foi alterado para dar o suporte necessário às técnicas e objetivos acima descritos.

### 1.3 Estrutura

Este trabalho está organizado da seguinte forma:

O capítulo 1 apresenta a definição simplificada do problema, a justificativa de sua abordagem e os objetivos específicos do presente trabalho.

O capítulo 2 apresenta um levantamento dos conceitos da literatura que tocam o problema de programação de horários no contexto escolar e universitário, assim como trabalhos semelhantes e as técnicas utilizadas por estes.

O capítulo 3 descreve detalhadamente a metodologia deste trabalho, apresentando os conceitos básicos iniciais e seus desenvolvimentos, a descrição das metaheurísticas adaptadas para o problema e seus conceitos principais.

O capítulo 4, define o estudo de caso desenvolvido e, finalmente, os experimentos realizados juntamente com seus resultados.

Por fim, o capítulo 5 apresenta as conclusões deste trabalho e algumas considerações sobre pontos interessantes a serem explorados em trabalhos futuros.

# Capítulo 2

## Revisão da Literatura

Esta seção descreve alguns conceitos fundamentais sobre os problemas de programação de horários em instituições de ensino, sintetizados através da revisão da literatura realizada, além de um levantamento de trabalhos relacionados à metodologia desenvolvida neste trabalho.

### 2.1 Problemas de Programação de Horários

[40], ao avaliar diversos trabalhos anteriores, define o problema de programação de horários como a alocação de um conjunto de recursos a um conjunto de objetos no espaço e no tempo, de forma a satisfazer tanto quanto for possível um conjunto de objetivos e levando em consideração um conjunto de restrições.

[2] descreve alguns conceitos básicos da terminologia da programação de horários:

**Evento (Atividade):** é o elemento a ser alocado. Exemplos são exames ou as aulas de um curso.

**Intervalo de tempo (período):** o intervalo de tempo no qual um evento pode ser alocado.

**Recurso:** requerimentos de eventos. Exemplos são salas, equipamentos, etc.

**Indivíduo:** uma pessoa que tem que comparecer aos eventos.

**Restrição:** as necessidades que devem ser respeitas ao alojar os eventos. Exemplos são a capacidade das salas, um período específico, preferências dos indivíduos, etc.

**Conflito:** situação onde dois ou mais eventos possuem pelo menos um indivíduo em comum e estão alocados no mesmo período.

## **2.2 Problemas de Programação de Horários e o contexto de Instituições de Ensino**

Como exposto em [31], o problema de programação de horários no contexto das instituições de ensino é foco de grande interesse no meio científico por algumas razões. Entre elas:

**Dificuldade de solução:** produzir uma grade horária que satisfaça adequadamente todos os interesses envolvidos não é fácil e até mesmo a construção de uma grade viável qualquer já pode, por vezes, ser desafiador.

**Importância prática:** grades horárias de melhor qualidade se relacionam com a maior satisfação dos professores, com o uso eficiente dos recursos da instituição e com o desempenho acadêmico dos alunos.

**Importância teórica:** variações destes problemas são exemplos da classe dos problemas NP-Completos ou NP-Difíceis.

Um grande número de variações do problema foram observadas e propostas na literatura, diferindo entre si pelo tipo de instituição (escolas de ensino fundamental

e médio ou universidades) e pelo tipo das restrições. [33] classifica essas variações em três tipos principais:

***School Timetabling:*** agendamento semanal em escolas de todas as aulas, evitando que professores sejam alocados a duas aulas no mesmo período.

***Course Timetabling:*** agendamento semanal das aulas de um conjunto de turmas de cursos universitários, minimizando a sobreposição de aulas de turmas que tenham alunos em comum.

***Examination Timetabling:*** agendamento dos exames de um conjunto de cursos universitários, evitando sobreposição de exames de turmas que tenham estudantes em comum e, simultaneamente, espalhando os exames para cada estudante tanto quanto possível.

A variação *School Timetabling* ou Problema de Programação de Horários em Escolas (PPHE) é provavelmente o mais conhecido dos PPH e corresponde, no Brasil, ao encontrado em escolas de ensino fundamental e médio [31]. Aqui, cada professor tem um número determinado de aulas a lecionar para cada turma em um conjunto de períodos e as alocações devem considerar a não existência de conflitos onde cada professor. Cada turma podem ser associados a, no máximo, uma atividade por período de tempo e um conjunto de restrições específicas como, por exemplo, a disponibilidade dos professores deve ser respeitado (o que por si só já coloca o problema na classe NP-Completo) [31].

Já no problema de *Course Timetabling* ou Problema de Programação de Cursos em Universidades (PPHU) um conjunto de cursos é alocado a um número determinado de períodos (ou salas) dentro de uma semana e, ao mesmo tempo, estudantes e professores são associados aos cursos de forma a que os encontros possam acontecer [2]. Nessa variação, restrições como a utilização de recursos restritos e preferências

de professores ou cursos por horários ou salas são consideradas. Além disso, o conceito de turma é mais fraco que no PPHE já que a flexibilidade do currículo é maior e, assim, a variação dos alunos interessados em disciplinas de períodos (do ponto de vista curricular) e até mesmo departamentos diferentes é um fator importante [31].

[31] cita que a dificuldade de produção de grades horárias de qualidade vem, em parte, a própria dificuldade dos tomadores de decisões interessados tem em definir clara e totalmente as características de qualidade que tornam uma grade horária melhor ou pior. Os problemas possuem geralmente múltiplos objetivos simultâneos, que frequentemente são também contraditórios. Com isso, passa a ser um tarefa do analista definir cada um destes objetivos e sua importância relativa.

Tais objetivos se originam nos requerimentos levantados para as grades de horário e são muito dependentes de cada instituição e do contexto educacional local. [31] classifica tais requerimentos de forma geral em três grupos:

**Organizacionais:** oriundos das regras da própria instituição de ensino, como a gestão de recursos e a adequação à legislação. Como exemplo tem-se a alocação de salas, que deve levar em consideração a capacidade, características como multimídia ou laboratórios e etc.

**Pedagógicos:** ligados ao interesse de promover o melhor aproveitamento das aulas, como a duração das aulas (como, por exemplo, dar preferência a uma distribuição homogênea das horas de aula pela semana) e a garantia de intervalos controlados entre as aulas (já que tarefas de casa, por exemplo, seriam melhor desenvolvidas entre uma aula e outra).

**Pessoais:** aqueles relacionados às preferências e interesses pessoais dos professores, como por exemplo a preferência por horários e dias da semana ou menos deslocamentos entre campi.

Assim, como é apontado em [41], o estudo do problema é fortemente dificultado pela variedade de contextos de estudo. Apesar de muito explorado, não existe uma

formulação padrão para o problema, muito menos instâncias de referência para testes comparativos. Cada nova variação do problema que é estudada resultada em novas instâncias com características específicas.

### **2.3 PPH com múltiplos objetivos: possíveis modelagens**

Pelos motivos descritos na seção anterior, [31] observa que os PPHU são exemplos naturais de Problemas com Múltiplos Objetivos (PMOs) e, como tais, podem ser abordados através de técnicas desenhadas especificamente para esse tipo de problema. Nesse sentido, os autores de [23] argumentam que as abordagens habituais a problemas deste tipo, nas quais as múltiplas funções são agregadas em apenas uma através de uma combinação linear, podem ser inadequadas. A dificuldade surge notadamente quando a definição dos coeficientes de penalização referentes aos diferentes objetivos são difíceis de se definir, devido a tendência de uns dominarem os valores de outros ou simplesmente o fato de não serem diretamente comparáveis. Por isso, em seu trabalho, o PPHE é explorado com uma abordagem baseada em algorítimos genéticos de multi-objetivo e com uma modelagem baseada em apenas dois objetivos.

Porem, é importante notar que técnicas específicas a PMOs tem normalmente como resultado não apenas uma solução, mas sim um conjunto delas e cabe ao analista selecionar entre estas a solução com a melhor relação entre os objetivos, de acordo com seus critérios. Tal tarefa pode não ser factível quando o número de objetivos é grande, já que consequentemente o tamanho do conjunto de soluções finais pode crescer muito. Nesses casos, a combinação das diferentes funções em uma única parece uma abordagem mais adequada, principalmente quando o analista conhece bem a relação de importância entre os objetivos [31].

Em alguns casos, como é descrito em [33], o problema de programação de horários consiste em simplesmente encontrar uma grade horária que satisfaça todas as restrições, podendo dessa forma ser formulado como um problema de busca, enquanto em outros casos mais próximos de casos reais ele é melhor definido como um

problema de otimização, onde as restrições são divididas em dois tipos gerais, um destes associado à viabilidade da solução (restrições fortes) e o outro à sua qualidade (restrições fracas). Desse modo, as soluções devem satisfazer todas as restrições fortes e simultaneamente maximizar (ou minimizar) uma única dada função objetivo que é embutida no conjunto de restrições fracas.

[30], ao considerar a modelagem do problema, descreve os grupos de restrições da seguinte forma:

**Fortes:** são aquelas que devem ser satisfeitas a qualquer custo, em detrimento de quaisquer outras, visto que uma grade de horários que as viole não é aceitável. O exemplo mais comum é a não ocorrência de conflitos no tempo ou no espaço. Estas restrições definem o espaço de busca de soluções factíveis. Caso o problema considerado consista em encontrar uma dessas soluções qualquer, então este é chamado de problema de busca.

**Fracas:** são aquelas cuja satisfação é desejável, mas uma solução que não o faça ainda pode ser utilizada na prática. Um exemplo comum destas é a preferência dos professores em qualquer que seja o aspecto. O atendimento destas rescrições é a mensuração da qualidade da grade horária e o que permite a comparação de duas soluções factíveis. Quando o problema consiste em encontrar uma solução que maximize ou minimize a violação das restrições deste tipo o problema é dito um problema de otimização.

## 2.4 Abordagens na literatura

Como descreve [2], diversas técnicas de naturezas distintas foram utilizadas na literatura para resolver diversas partes dos PPH nas últimas décadas. O autor divide as técnicas empregadas em múltiplas categorias gerais: métodos baseados em restrições, abordagens baseadas em grafos, métodos baseados em grupos, abordagens baseadas em população, métodos meta-heurísticos, abordagens multi-critério, hiper-heurísticas, raciocínio baseado em casos, raciocínio baseado no conhecimento e raciocínio baseado em lógica nebulosa.

O presente trabalho se concentrou na pesquisa envolvendo meta-heurísticas, técnicas descritas por [38] como "processos-mestre" que guiam e modificam as operações de outras heurísticas, subordinadas a ele, para produzir mais eficientemente soluções de maior qualidade. Tal processo pode manipular soluções únicas ou conjuntos delas a cada iteração e elas podem estar completas ou não. Adicionalmente, as heurísticas subordinadas podem ser procedimentos de alto ou baixo nível, procedimentos de busca local ou apenas métodos de construção.

Como é explicado em [35], técnicas de busca local tal como as implementadas neste trabalho (e na ferramente de software produzida) são adequadas tanto à tarefa de construir uma grade horária quanto a de fazer a manutenção continua destas. A partir de uma grade gerada por um procedimento qualquer (inclusive manualmente) que contempla determinado conjunto de restrições, tais técnicas podem ser usadas para tomar essa grade como ponto de partida quando os requerimentos mudam (ou até mesmo partes da estrutura da instância, como quando professores são adicionados ou retiradas). Segundo o autor, diante desta perspectiva, os métodos baseados em busca local superam outros métodos usados para essa classe de problemas, como os métodos puramente construtivos e as técnicas exatas.

#### 2.4.1 Trabalhos Relacionados

Esta seção discute brevemente alguns trabalhos anteriores que exploram diferentes variações do PPH, principalmente através de técnicas meta-heurísticas, tanto no contexto de escolas quanto no de universidades. Nos PPH, as salas de aula podem ser tomadas como recursos a serem alocados aos eventos e, portanto, alguns dos trabalhos tratados abaixo estudaram casos em que a alocação de salas fazia parte do problema, apesar do presente estudo não incluí-las.

[32] descreve a implementação de uma solução baseada em Busca Tabu para o PPH aplicado a escolas de ensino médio, onde foi realizada uma intercalação de movimentos de diferentes tipos e usada uma técnica que busca ajustar dinamicamente o peso das restrições fortes durante a busca nomeada "Relaxação Adaptativa".

Em [35] o problema abordado também se refere a uma escola de ensino médio

com muitos requisitos típicos como, por exemplo, carga horária mínima para cada professor, critérios de disponibilidade de horário e compacidade de seus horários entre outros específicos da instituição, como o respeito a uma hierarquia entre os professores com mais e menos tempo de serviço quanto a consideração de suas preferências. O trabalho desenvolve um modelo matemático para o problema, aplicando redução de restrições e cortes. Para o processo de produção de soluções o trabalho explora um conjunto de técnicas: GRASP com a Busca Tabu na fase de busca local, *Simulated annealing* e *Annealing* Microcanônico combinados com uma heurística de melhoria baseada em grafos chamada de Procedimento Intraturmas-Interturmas e Otimização Microcanônica. A mesma técnica de GRASP com Busca Tabu é usada em [36], juntamente com uso do Procedimento Intraturmas-Interturmas para recuperar a viabilidade das soluções quando necessário durante a busca.

Já em [23] o problema (também no contexto de escolas de ensino médio) é tratado de forma distinta, através de uma modelagem de múltiplas funções objetivo e a implementação do algoritmo NSGA (um exemplo da classe dos algoritmos genéticos multi-objetivo). Em sua modelagem, os autores construíram duas funções objetivo de minimização distintas, uma delas orientada às restrições ligadas aos professores e a outra àquelas ligadas às classes.

A técnica de algoritmos genéticos foi usada em diversos estudos de caso na produção das grades horárias em universidades brasileiras. Os estudos sobre no curso de Informática da Universidade Federal do Paraná ([3]), na Faculdade de Engenharia Elétrica da Universidade Federal de Uberlândia ([16]), na Universidade Presidente Antônio Carlos ([22]) e no Departamento de Computação da Universidade Federal de Sergipe ([37]) são exemplos. [3] explora também a evolução cooperativa e compara a técnica com a solução em algoritmos genéticos enquanto [22] aplica um paradigma de paralelismo para heurísticas baseadas em população conhecido como modelo de ilhas ([39]). Uma distinção em [37] é que foi utilizada uma função objetivo de maximização (em vez de minimização) e, assim, o atendimento das restrições fortes e fracas do problema foi modelado sob a perspectiva de bônus ou penalidades.

Por sua vez, [14] faz um estudo comparativo entre uma implementação heurística

com algoritmos genéticos, uma baseada em programação linear inteira binária e uma alternativa mista na qual uma solução é gerada pelo método exato sob um conjunto reduzido de restrições e então é refinada pela técnica heurística (atingindo os melhores resultados em seus experimentos).

Nos trabalhos [6], [29] e [18] (estudos de caso sobre o Departamento de Matemática Aplicada da Universidade Federal do Espírito Santo (UFES), da Faculdade de Engenharia de Produção da Universidade Federal Fluminense (UFF) e dos cursos de pós-graduação do Departamento de Estatística e Informática da Universidade Federal Rural de Pernambuco, respectivamente) o problema de programação de horários foi abordado através da técnica de programação inteira. [18] se destaca por incluir o desenvolvimento de uma módulo *web* que permite a entrada dos dados da instância e os converte para formatos suportados por *solvers* e bibliotecas disponíveis para problemas de programação inteira, facilitando o uso da solução na prática das instituições. Já [7], no PPHE com restrições de compacidade, desenvolve uma abordagem heurística de programação inteira chamada *fix-and-optimize*, combinada com um método VND.

Em [26] foi desenvolvida uma heurística GRASP+VND, fazendo uso de um método que não garante a construção de uma solução viável na fase construtiva do GRASP, mas que, em vez disso, tenta apenas minimizar o número de violações das restrições impostas para a formação do horário.

[41] desenvolve uma implementação de uma heurística VNS/VND para o problema de alocação horária de uma instância reduzida de uma instituição de ensino superior e que se baseia em parte do estudo em [26]. No trabalho em questão, foi utilizada uma função de objetivo diferente do princípio de minimização de penalidades usado por [26]. A função objetivo é de maximização das preferências dos professores por dias e horários específicos da semana e apenas soluções viáveis são consideradas na busca. No entanto, todas as estruturas de vizinhança definidas por [26] são usadas na implementação, sendo acrescida mais uma.

No trabalho de [12] é apresentada a solução desenvolvida pelos autores durante uma competição internacional que foi promovida para estimular pesquisadores a me-

lhorarem as abordagens usadas para o problema PPHE (a ITC2011 - International Timetabling Competition [1]). Devido à natureza internacional da competição o modelo proposto foi genérico, incluindo um conjunto amplo de restrições, e as instâncias utilizadas foram codificadas em um formato proposto por [27] (o XHSTT). A estratégia de resolução do trabalho fez uso do algoritmo construtivo heurístico específico ao PPHE presente em um *solver* chamado KHE ([20]), que tem suporte ao XHSTT, e de uma combinação das técnicas meta-heurísticas *Simulated Annealing* e Busca Local Iterada. A solução proposta foi vencedora da competição.

Em um trabalho posterior ([10]) os autores da solução vencedora na competição descrita no parágrafo anterior aplicam algoritmos baseados em VNS às mesmas instâncias e modelagem do trabalho anterior, superando os resultados obtidos com a combinação *Simulated Annealing*-Busca Local Iterada original.

Também capaz de superar o desempenho da solução vencedora da ITC2011 é a alternativa descrita em [11], que em sequência aos trabalhos [12] e [10] tratou os problemas da competição com variações de *Late acceptance hill-climbing* combinadas, também, ao *Simulated Annealing*.

Outro estudo de caso no contexto universitário brasileiro (sobre o Centro de Ciências Agrárias da Universidade Federal do Espírito Santo) também faz uso da aplicação da técnica *Simulated Annealing*, mas partindo de uma solução inicial puramente gulosa e não genérica ([4]).

Em [19] é apresentado um estudo de caso em um departamento da Universidade Federal Fluminense (UFF) usando a meta-heurística Busca Local Iterada (ILS). A instituição estudada neste trabalho fazia, até então, uso da ferramenta FET para a criação de suas grades horárias e os resultados obtidos com a implementação da ILS desenvolvida se mostraram superiores, sob a perspectiva e critérios da instituição.

É notável que a modelagem baseada no princípio da função objetivo única descrito na seção 2.3 é a mais comum para os PPH em instituições de ensino, quando considerados quase todos os trabalhos levantados. Além disso, de maneira geral, a modelagem mais comum é a de minimização de uma função ponderada de penaliza-

dades ligadas às restrições.

# Capítulo 3

## Metodologia

Todo o desenvolvimento metodológico deste trabalho foi orientado a produzir uma ferramenta de software que pudesse, simultaneamente, atender todas as necessidades do DCC/UFRRJ e também ser tão flexível quanto possível para permitir seu uso na maior variedade de instâncias possível.

### 3.1 A ferramenta FET

O produto de software desenvolvido ao longo deste trabalho foi customizado a partir da ferramenta de código aberto FET, disponível em [21] sob os termos da licença AGPL. O FET é um software livre para geração de grades horárias no contexto escolar, usada internacionalmente e desenvolvido desde 2002.

Apesar de possuir suporte à alocação de espaço, além de tempo, uma grande gama de restrições específicas implementadas e uma interface gráfica que facilita a entrada de dados, a ferramenta não possui suporte ao conceito de alocação de professores, sendo estes necessariamente pré-alocados na entrada. Sendo assim, para muitas instituições, como é o caso do DCC/UFRRJ, essa ferramenta é extremamente limitada.

Ademais, o criador da ferramenta não faz referências muito detalhadas às bases teóricas utilizadas no desenvolvimento do algoritmo implementado (apenas apon-

tando [24], um trabalho que trata o PPH com programação por restrições, como inspiração), além de se limitar à construção de soluções viáveis e não explorar de nenhuma forma as técnicas de busca local amplamente usadas na literatura para essa classe de problemas (seção 2.4). De fato, a implementação realizada na ferramenta se assemelha mais ao conceito de "problema de busca" do que a um "problema de otimização", tal como descritos na seção 2.3.

### 3.1.1 Recursos herdados

Dentre os recursos presentes na ferramenta produzida neste trabalho e que são herança do FET estão (lista não exaustiva):

- a customização de detalhes sobre o domínio de uma instância, tal como número de horas por dia/dias por semana;
- a persistência dos dados das instâncias no formato de arquivo *XML*;
- uma gama diversa de restrições de tempo já implementadas;
- a maior parte da interface gráfica;
- suporte a divisão de atividades em sub-atividades, com durações diferentes definidas em número de horas.

### 3.1.2 Conceitos, terminologias e adaptações

Um série de alterações estruturais foram feitas para adequar o modelo do FET ao novo problema de alocação de professores, desde a modelagem básica das instâncias até a completa substituição dos algoritmos de geração de soluções. Além disso, a fim de simplificar as modificações e limitar o escopo deste trabalho, os recursos de alocação de espaço presentes no FET foram suprimidos.

Muitos dos termos utilizados ao longo deste capítulo foram herdados dos conceitos definidos no ecossistema do software FET. Estes foram mantidos, no processo

de junção entre o modelo de problema usado na ferramenta, a pesquisa teórica realizada e as observações práticas decorrentes do estudo de caso feito durante este trabalho.

Por isso, a fim de evitar confusões entre termos nas próximas seções, convém definir alguns conceitos. O primeiro deles é a ideia *grupo de alunos*, que é qualquer grupo genérico de alunos que pode ser definido em uma instituição. No caso do DCC/UFRRJ, os grupos de alunos foram definidos como os períodos: 1º período, 2º período, etc.

O conceito mais importante é o de *atividade*. Uma atividade aqui é um par de a) **Lista com um ou mais grupos de alunos** e b) **Uma disciplina** e é parte dos dados imutáveis (por ação dos algoritmos de geração) da entrada de uma instância. Assim, o problema a ser solucionado pode ser descrito, nestes termos, como a alocação de um par de a) **professor** e b) **horário da semana** a cada uma das atividades informadas como entrada. No contexto do DCC/UFRJ, uma atividade é o equivalente a uma *turma* de uma determinada disciplina, destinada a algum período.

Uma atividade também pode ser dividida em *sub-atividades*, que são individualmente manipuladas como atividades comuns na maior parte do tempo pelos algoritmos aqui utilizados (com exceção da situação discutida na Seção 3.4.5.6), mas que estão permanentemente associadas como membros de um *grupo de atividades*. No contexto do estudo de caso deste trabalho, a divisão de uma atividade em sub-atividades modela o fato de que, nas instâncias do DCC/UFRRJ consideradas, cada turma oferecida possui duas aulas de duração igual por semana.

Outra característica das instâncias estudadas é que cada grupo de atividades (tal como definido acima) pode ser alocado apenas a um mesmo professor, ou seja, todas as sub-atividades (ou aulas) que compõem uma atividade (ou turma oferecida) devem ser ministradas pelo mesmo professor (o que leva à questão levantada na Seção 3.4.5.3). No entanto, a opção de permitir que a alocação de professores se dê de forma individual por sub-atividade foi incluída na ferramenta para uso futuro em outros tipos de instância.

## 3.2 Construção de soluções

Umas das etapas do problema consiste na construção de uma solução inicial, a partir dos dados da instância. Heurísticas de construção são normalmente a forma mais rápida de se obter soluções viáveis para um problema e tem como objetivo principal fornecerem boas soluções para problemas urgentes ou, como é o caso do uso feito no presente trabalho, soluções iniciais para outras heurísticas de melhoramento [34]. O próprio algoritmo desenvolvido e utilizado pelos autores da ferramenta FET é um exemplo de heurística construtiva aplicada ao problema de programação horária escolar.

Conforme exposto nas seções posteriores, os métodos de construção (Seção 3.5.2) utilizados aqui foram baseados na definição genérica da fase de construção da meta-heurística GRASP (Seção 3.5.1), sendo, por vezes, usados inclusive para a geração de soluções iniciais completamente aleatórias.

## 3.3 Busca Local

Os métodos de busca local são aqueles que exploram o espaço de busca ao redor de uma dada solução inicial, avaliando seus *vizinhos*, ou seja, as soluções produzidas através de alguma estrutura de vizinhança. A estrutura de vizinhança  $N(s)$  de uma solução  $s$  é definida como o conjunto de todas as soluções que podem ser alcançadas com a aplicação de algum tipo de movimento [13]. Aqui, a definição de movimento não é clara de maneira genérica, mas sim depende do domínio do problema. No caso dos problemas de programação horária, alguns dos tipos de movimentos possíveis (e utilizados neste trabalho) estão descrições na Seção 3.4.5.

### 3.3.1 Método de descida (*Best improving*)

Uma técnica simples de busca local percorre todos os vizinhos de uma solução dada alguma estrutura de vizinhança, avaliando os valores da função objetivo em cada um deles e retornando a solução de melhor custo [35].

### 3.3.2 Método randômico de descida (*First improving*)

Outra estratégia de exploração de uma vizinhança qualquer consiste em percorrer os vizinhos de uma solução até que seja encontrado um que tenha melhor valor na função objetivo que a solução inicial. Esta técnica é útil principalmente em situações em que explorar todos os vizinhos seja de alguma forma inviável [35].

### 3.3.3 Meta-heurísticas na busca local

Na literatura existem diversas técnicas de meta-heurística que foram desenhadas para serem usadas em conjunto com técnicas de busca local e foram amplamente exploradas nos trabalhos discutidos no Capítulo 2. A Seção 3.5 descrevem algumas destas técnicas, especificamente as que foram usadas no decorrer deste trabalho e adaptadas ao problema estudado.

## 3.4 Modelagem do problema

Para problemas de otimização multiobjetivo, uma forma de modelagem possível é centrada na definição das restrições do problema. Cada objetivo é contemplado por uma restrição e, para uma dada solução, o nível de violação dessas restrições define a sua qualidade. Assim, para a avaliação de uma dada solução, cada restrição é avaliada individualmente computando suas violações e a função objetivo se reduz a uma de minimização do somatório destas violações. Como descreve [35], uma função de objetivo assim definida tenta medir a diferença entre uma solução real avaliada e uma solução ideal, isto é, uma que não viola nenhuma das restrições.

### 3.4.1 Fatores de violação de restrições

Uma solução pode ser avaliada, na perspectiva de uma dada restrição, computando um *fator de violação* numérico, referente a cada restrição analisada. A função que calcula tal valor é particular para cada restrição específica, mas conceitualmente deve ser definida de tal forma que sua imagem  $I$  satisfaça  $I \in [0, +\infty]$ , onde um fator

de violação igual a zero significa que a solução avaliada não infringe de nenhuma forma a restrição em questão.

Uma forma simples de cálculo dos fatores de violação baseia-se na contagem absoluta das violações pontuais de uma restrição em uma solução e é a forma como a maior parte das restrições já implementadas do FET tiveram seus cálculos definidos, assim como também é a forma como este cálculo foi realizado em [26]. Os detalhes de como esses fatores foram calculados para as restrições utilizadas no estudo de caso da Seção 4 (que incluem tanto implementações herdadas do FET quanto novas) estão expostos nas tabelas da Seção 4.3.4.

### 3.4.2 Grupos de restrições e função objetivo

Problemas de programação horária frequentemente trazem a necessidade de conciliar múltiplos objetivos conflitantes ao avaliar a qualidade de uma grade [35]. Por exemplo, quando deseja-se minimizar a quantidade de dias em que um professor precisa dedicar-se à sala de aula ao mesmo tempo em que se deseja maximizar a distância entre diferentes aulas de uma mesma atividade. Porém, segundo [26], observa-se na literatura que as abordagens mais comuns definem a função objetivo como uma única função que minimiza o somatório das penalidades, para avaliar as soluções.

Por isso, para uma avaliação mais dinâmica das grades horárias, as restrições do problema devem ser separadas em subgrupos, cada um representando uma função linear de penalidades e assegurando um peso distinto [35]. Portanto, tal como definido em [26], as restrições deste trabalho foram divididas em três grupos:

<b>Essenciais</b>	Restrições deste grupo são as mais importantes. Uma solução que viola, mesmo que parcialmente, qualquer restrição neste grupo é considerada inviável.
<b>Importantes</b>	São as restrições que, quando violadas, prejudicam <i>muito</i> a qualidade de uma solução.
<b>Desejáveis</b>	A violação de restrições deste grupo é aceitável, contribuindo para

o prejuízo da qualidade de uma solução, mas em grau muito menor que a do grupo anterior.

Naturalmente, às restrições que pertencem ao grupo de restrições essenciais é atribuído um peso maior, na avaliação das soluções, do às que pertencem ao grupo de importantes, que por sua vez tem peso maior do que as que pertencem ao grupo de restrições desejáveis (Seção 3.4.3).

### 3.4.3 Função objetivo

A função objetivo usada neste trabalho (tal como em [26] e baseado nos conceitos de [35]) foi definida como o somatório ponderado de quatro funções lineares distintas, cada qual contabilizando o somatório dos fatores de violação computados para cada restrição (Seção 3.4.1) referente a cada um dos grupos de restrição, resultando na seguinte equação:

$$f_{obj}(s) = \omega * f_e(s) + \sigma * f_i(s) + \varrho * f_d(s), \quad (3.1)$$

onde,  $\omega$ ,  $\sigma$  e  $\varrho$  são os pesos atribuídos a cada grupo de restrições,  $f_e(s)$  representa o fator de violação total de todas as restrições do grupo essencial,  $f_i(s)$  o fator de violação do grupo das importantes e  $f_d(s)$  do grupo das desejáveis.

Os pesos atribuídos a cada uma das componentes devem ser definidos seguindo uma hierarquia na forma  $\omega >> \sigma > \varrho$ , segundo orientação de [35], garantindo que sejam diretamente proporcionais às importâncias dos grupos de restrições aos quais se referem.

### 3.4.4 Representação da solução

Para fins de compatibilidade com as restrições já implementadas no código do material fonte discutido na Seção 3.1, o estado de uma solução é representado como dois vetores de inteiros de comprimento igual ao número de atividades ativas, onde cada posição identifica uma atividade distinta. O primeiro deles é o vetor de tempos, que registra o identificador do horário alocado a uma dada atividade. O segundo

vetor corresponde ao recurso professor, contendo em cada posição uma referência ao professor alocado à atividade correspondente.

O identificador do horário usado no vetor de tempos é uma representação unificada de dia e horário na forma exemplificada na Tabela 3.1, para uma instância com cinco dias na semana e cinco possíveis horários de início para as atividades.

Tabela 3.1: Um exemplo de identificadores das horas de uma semana

	seg	ter	qua	qui	sex
08:00	0	5	10	15	20
10:00	1	6	11	16	21
12:00	2	7	12	17	22
14:00	3	8	13	18	23
16:00	4	9	14	19	24

Para um conjunto de professores  $P = \{p1, p2, p3, p4\}$ , um conjunto de atividades  $A = \{a1, a2, a3, a4\}$  e a definição de horários da semana do exemplo anterior, a representação interna de uma possível solução pode ser vista da Tabela 3.2.

Tabela 3.2: Representação interna de uma solução com quatro atividades ativas

<b>Vetor de professores</b>	[	$p1, p2, p1, p4$	]
<b>Vetor de tempos</b>	[	0, 5, 10, 5	]

a1	a2	a3	a4
----	----	----	----

### 3.4.5 Estruturas de vizinhança

As estruturas de vizinhança implementadas neste trabalho para uso nos diferentes métodos de busca local implementados são descritas abaixo. As estruturas  $N1$ ,  $N2$  e  $N3$  foram definidas em [26], a estrutura  $N4$  em [41] e a estrutura  $N5$  em [13]. Note que, durante a exploração das vizinhanças é possível que soluções inviáveis sejam visitadas e nenhum esforço para evitar tal acontecimento é feito. Espera-se, nesse modelo, que toda a navegação pelo espaço de soluções em direção a soluções viáveis seja feito com a avaliação da função objetivo.

3.4.5.1 *Troca dos horários entre pares de atividades de um professor no mesmo dia (N1)*

Esta estrutura troca os horários de duas atividades alocadas a cada professor, em um mesmo dia da semana.

3.4.5.2 *Troca dos horários entre pares de atividades de um professor em dias distintos (N2)*

De forma similar à vizinhança anterior, esta é alcançada com a troca dos horários de duas atividades alocadas a um professor, mas em dias distintos da semana.

3.4.5.3 *Troca de professores entre sub-atividades que pertencem a um mesmo grupo (N3)*

Para atividades que possuem mais de uma aula e aceitam múltiplos professores, esta estrutura se da pela troca dos professores alocados, entre cada uma das aulas que compõem a mesma atividade. Esta vizinhança não tem nenhuma utilidade no estudo de caso discutido na Seção 4, já que nas instâncias utilizadas nenhuma das atividades permitia que professores distintos fossem alocados a cada aula individualmente. No entanto, ainda assim foi implementada e está disponível para uso futuro.

3.4.5.4 *Troca do professor em uma atividade (N4)*

Esta estrutura realiza a troca do professor alocado a cada atividade, dentre todos os professores ativos.

3.4.5.5 *Troca de professores entre atividades (N5)*

Como forma de aumentar a diversidade das dimensões do espaço de busca, a estrutura N5 foi adicionada, inspirada na vizinhança *Resource move* utilizada em [13]. Aqui, os professores são trocados entre todos os pares de atividades.

#### 3.4.5.6 *Lidando com grupos de atividades*

A nível de implementação, sub-atividades são tratadas como atividades independentes durante a alocação de horários sem nenhum comprometimento. Porém, quando acontece a alteração do professor alocado a uma atividade, como é o caso das vizinhanças do tipo  $N4$  e  $N5$ , grupos de atividades que não permitem a alocação de professores distintos para suas sub-atividades (como os presentes nas instâncias do DCC/UFRRJ) devem ser tratados de acordo com essa limitação. Assim, quando um professor é atribuído a uma das sub-atividades de um grupo de tal tipo, todas as outras sub-atividades deste grupo também o são.

### 3.5 Métodos implementados

Neste trabalho, foram implementadas adaptações de uma série de métodos metaheurísticos, colhidos da literatura, que tipicamente foram utilizados em variações do problema de geração de grades horárias (Capítulo 2). As seções a seguir descrevem estes métodos.

#### 3.5.1 GRASP

O método GRASP é descrito como um algoritmo iterativo de multi-começo que consiste de duas fases [9]:

1. uma fase inicial de construção gulosa, adaptativa e aleatorizada na qual uma solução inicial é gerada, elemento a elemento.
2. uma fase de busca local que começa da solução inicial construída e encontra um ótimo local em sua vizinhança.

A melhor solução encontrada, ao longo de todas as iterações GRASP realizadas após um critério de parada especificado, é retornada como resultado. Um pseudo-código do método GRASP tal como foi implementado neste trabalho segue no Algoritmo 1.

**Algoritmo 1** GRASP

---

```

1: método GRASP( $I$ ,  $nIter$ ,  $\alpha$ )
2:    $custoMinimo \leftarrow \infty$ 
3:   para  $i \leftarrow 1$  até  $nIter$  faz
4:      $s \leftarrow \text{CONSTRUIRSOLUCAO}(I, \alpha)$ 
5:      $s \leftarrow \text{BUSCALOCAL}(s)$ 
6:     se  $s.custo < custoMinimo$  então
7:        $custoMinimo \leftarrow s.custo$ 
8:        $s^* \leftarrow s$ 
9:     fim se
10:   fim para
11:   retorna  $s^*$ 
12: fim método

```

---

Os parâmetros recebidos pela função GRASP são um objeto contendo os dados da instância ( $I$ ), o número de iterações a serem executadas ( $nIter$ ) e o parâmetro  $\alpha$  utilizado para regular o equilíbrio entre o caráter guloso e o caráter aleatório da fase de construção (detalhada na próxima seção). Inicialmente o custo mínimo é inicializado com o valor infinito (linha 2), na linha 4 uma solução nova é construída a partir dos dados da instância e do parâmetro  $\alpha$  e a seguir (linha 5) alguma das estratégias de busca local implementadas (ou adaptações de outras meta-heurísticas, em caso de hibridização) é alimentada com esta solução, retornando uma nova. O custo da solução resultante da busca local é comparado com o custo mínimo visto durante todas as iterações (no condicional da linha 6) e, quando uma solução melhor é encontrada esta é armazenada como nova solução a ser retornada na linha 8, quando o critério de parada (nesse caso o número de iterações) for alcançado.

### 3.5.2 Métodos de Construção

A fase de construção de uma solução se da iterativamente, um elemento por vez. [9] descreve a fase construtiva do método GRASP da seguinte forma: partindo de uma solução inicialmente vazia, a cada iteração a escolha do próximo candidato a ser adicionado à solução é feita aleatoriamente entre os elementos possíveis em uma lista restrita de candidatos (LRC). Os candidatos incluídos nesta lista são selecionados

da lista de todos os candidatos (LC) possíveis usando como critério o valor da função objetivo avaliada de maneira parcial e incremental, considerando o custo da adição de cada candidato à solução incompleta. A heurística é adaptativa porque a cada iteração os candidatos são avaliados tendo em vista a solução já construída até o momento e é gulosa por limitar o custo máximo do candidato escolhido. O componente probabilístico da heurística é atingindo ao se escolher aleatoriamente um candidato de um lista restrita de melhores candidatos gerados. Essa escolha permite que novas soluções sejam obtidas em cada iteração do GRASP sem necessariamente comprometer o poder do componente adaptativo-guloso.

Segundo [9], existem duas formas de selecionar candidatos para serem incluídos na LRC. Uma delas é a baseada na cardinalidade, ou seja, o número de elementos da lista é pré-fixado e ela passa a ser composta pelos  $p$  melhores candidatos gerados. A segunda forma, que é a que foi utilizada neste trabalho, é a baseada em valor. Por este critério a cardinalidade da lista em cada iteração é definida por um parâmetro  $\alpha \in [0, 1]$  e cada candidato  $c$  é incluído na lista se, e somente se, o valor da função objetivo considerando a adição de  $c$  é tal que:

$$\text{limite} = \text{custo}_{\text{solucao}}^{\text{maximo em LC}} - \alpha * (\text{custo}_{\text{solucao}}^{\text{maximo em LC}} - \text{custo}_{\text{solucao}}^{\text{minimo em LC}}) \quad (3.2)$$

$$LRC = \{ c \mid \text{custo}_{\text{solucao}}(c) \in [\text{custo}_{\text{solucao}}^{\text{minimo em LC}}, \text{limite}] \} \quad (3.3)$$

Dessa forma, quando  $\alpha = 1$ , o algoritmo de construção se comporta de forma totalmente gulosa e quando  $\alpha = 0$ , de forma totalmente aleatória. A regulação do parâmetro  $\alpha$  pode levar a soluções de qualidade distintas, tornando útil encontrar um valor que equilibre o fator aleatório e o guloso no método, de modo a produzir as melhores soluções em cada instância.

Assim como o método construtivo usado em [26], as alternativas de construção utilizadas neste trabalho são tais que é possível que as soluções construídas violem uma ou mais restrições de viabilidade da instância. No entanto, o respeito ao menos às restrições básicas de tempo (professores alocados a mais de uma atividade ao mesmo tempo e grupos de alunos alocados a mais de uma atividade ao mesmo tempo) é garantido após a fase de construção.

As estratégias de construção utilizadas neste trabalho se basearam unicamente na fase construtiva do método GRASP, aqui descrita.

#### 3.5.2.1 *Versão A: Por grupo de alunos*

A primeira alternativa de construção das soluções é feita a partir dos grupos de alunos, iterando sobre a lista de pares possíveis (*grupo, horário*), e computando a lista de candidatos (LC) com elementos na forma (*atividade, professor*).

**Algoritmo 2** CONSTRUÇÃO A PARTIR DE GRUPOS DE ALUNOS

---

```

1: método CONSTRUCAOPORGRUPOS( $I, \alpha$ )
2:    $s \leftarrow$  vazia
3:   para cada  $g$  em VISÃOALEATÓRIA( $I.gruposDeEstudantes$ ) faça
4:     para cada  $t$  em VISÃOALEATÓRIA( $I.slotsDeTempo$ ) faça
5:       se  $g$  está livre no slot  $t$  então
6:          $LC \leftarrow []$ 
7:         para cada  $a$  em atividades não alocadas de  $g$  faça
8:           para cada  $p$  em professores livres no slot  $t$  faça
9:              $LC \leftarrow (a, p)$ 
10:            fim para
11:        fim para
12:        se  $LC.vazia$  então
13:          retorna falha
14:        fim se
15:         $maximo \leftarrow$  CUSTOINCREMENTALMAXIMO( $LC, s$ )
16:         $minimo \leftarrow$  CUSTOINCREMENTALMINIMO( $LC, s$ )
17:         $limite \leftarrow maximo - \alpha * (maximo - minimo)$ 
18:         $LRC \leftarrow \{candidato \in LC \mid s.custoCom(candidato) \leq limite\}$ 
19:         $candidatoEscolhido \leftarrow$  SELECCIONAALEATORIAMENTE( $LRC$ )
20:         $s.adicionaCandidato(candidatoEscolhido)$ 
21:      fim se
22:    fim para
23:    se existem atividades não alocadas para  $g$  então
24:      retorna falha
25:    fim se
26:  fim para
27:  retorna  $s$ 
28: fim método

```

---

A função de construção recebe como parâmetros apenas os dados da instância e o parâmetro de regulação  $\alpha$ . Na linha 2, uma solução vazia, que será preenchida durante o processo de construção, é instanciada. Então, cada grupo de estudante  $g$  da instância (visitado em ordem aleatória) e cada horário possível  $t$  (também visitado de forma independentemente aleatória) em que  $g$  não esteja alocado a nenhuma atividade forma um par  $(g, t)$ , de grupo e tempo, a ser avaliado. Para cada um destes

pares é criada a LC contendo todas as combinações possíveis entre as atividades  $a$  das quais  $g$  faz parte e ainda não foram alocadas e os professores  $p$  existentes (linha 9). Com a lista de candidatos completa, nas linhas 15 e 16 são encontrados, respectivamente, o maior e o menos custo incremental referente à adição de cada um dos candidatos na LC à solução  $s$ . Na linha 17, o limite para o valor do custo é calculado com uso do parâmetro  $\alpha$ , tal como descrito na seção 3.5.2, e na linha 18 é formada a lista restrita de candidatos. Um dos candidatos é então escolhido aleatoriamente e adicionado à solução  $s$ .

Em dois momentos diferentes, durante a execução do método, existe a possibilidade de que a construção falhe. O primeiro é na linha 13, quando para algum par  $(g, t)$  não existe nenhum par  $(a, p)$  possível. O outro momento é na linha 24, quando todos os pares  $(g, t)$  possíveis são considerados mas ainda existirem atividades não alocadas para o grupo  $g$ . Em ambos os casos o método retorna falha e é reiniciado até que tenha sucesso.

#### 3.5.2.2 Versão B: Por atividades

Este método de construção é mais simples, iterando sobre a lista de atividades ativas e computando a lista de candidatos (LC) com seus elementos na forma *(horário, professor)*.

**Algoritmo 3** CONSTRUÇÃO A PARTIR DAS ATIVIDADES

---

```

1: método CONSTRUCAOPORATIVIDADES( $I, \alpha$ )
2:    $s \leftarrow \text{vazia}$ 
3:   para cada  $a$  em VISÃOALEATÓRIA( $I.atividades$ ) faz
4:     para cada  $t$  em VISÃOALEATÓRIA( $I.slotsDeTempo$ ) faz
5:       se  $t$  está livre para todos os grupos de alunos na atividade  $a$  então
6:          $LC \leftarrow []$ 
7:         para cada  $p$  em professores livres no slot  $t$  faz
8:            $LC \sim (t, p)$ 
9:         fim para
10:        fim se
11:      fim para
12:      se  $LC.vazia$  então
13:        retorna falha
14:      fim se
15:       $maximo \leftarrow \text{CUSTOINCREMENTALMAXIMO}(LC, s)$ 
16:       $minimo \leftarrow \text{CUSTOINCREMENTALMINIMO}(LC, s)$ 
17:       $limite \leftarrow maximo - \alpha * (maximo - minimo)$ 
18:       $LRC \leftarrow \{candidato \in LC \mid s.custoCom(candidato) \leq limite\}$ 
19:       $candidatoEscolhido \leftarrow \text{SELECCIONAALEATORIAMENTE}(LRC)$ 
20:       $s.adicionaCandidato(candidatoEscolhido)$ 
21:    fim para
22:  retorna  $s$ 
23: fim método

```

---

Este método também recebe como parâmetros os dados da instância e o parâmetro  $\alpha$ . Na linha 2 a solução vazia é instanciada. Então, cada atividade ativa  $a$  em  $I$  é visitada (em ordem aleatória) e, então, cada horário possível  $t$  (também aleatoriamente visitado) que esteja livre para todo grupo de alunos associado a  $a$ . Para cada um dos apres  $(a, t)$ , de atividade e tempo, assim definidos são avaliados na geração de uma LC com todas as combinações possíveis entre o tempo  $t$  e cada um dos professores  $p$  em  $I$ , adicionadas à LC na linha 8. Com a lista de candidatos completa, as linhas 15 a 20 se comportam exatamente como no método anterior, calculando o custo limite, formando a LRC e selecionando o candidato aleatório.

Diferentemente da estratégia anterior, aqui a construção pode falhar apenas no

caso em que não existir candidato possível para formar a LC, ou seja, quando não houver nenhum tempo  $t$  possível para a atividade  $a$  em que *algum* professor esteja livre. Nesse caso o método retorna falha e é reiniciado.

### 3.5.3 GRASP Reativo

Apesar de sua simplicidade, possuindo apenas um parâmetro a ser ajustado, uma das possíveis limitações do método GRASP, segundo [9], está na falta de memória sobre o histórico de soluções encontradas nas iterações anteriores. Adicionalmente, a regulação do parâmetro  $\alpha$  tem grande impacto sobre a qualidade e diversidade das soluções produzidas na fase de construção, o que se reflete nas soluções finais [8].

Com isso em mente, o método GRASP Reativo foi proposto em [28] como uma modificação do método GRASP clássico, baseada em uma auto-regulação do valor do parâmetro  $\alpha$  ao longo das iterações a partir da avaliação da qualidade das soluções previamente encontradas.

Como se observa no Algoritmo 4, em vez de fixar o valor de  $\alpha$ , este método seleciona aleatoriamente o valor a ser usado em uma lista de valores aceitáveis (linha 10), no início de cada iteração. Inicialmente, a probabilidade de escolha de cada valor é igual (linha 6), mas a cada  $nIterAteAtualizacao$  iterações (linha 19) as probabilidades de escolha são atualizadas de forma a tornar mais prováveis valores de  $\alpha$  que em média levaram a soluções cujo valor é mais próximo do melhor encontrado até então. Os demais valores, entretanto, continuam sendo possíveis.

**Algoritmo 4** GRASP REATIVO

---

```

1: método GRASPREATIVO( $I$ ,  $nIter$ ,  $nIterAteAtualizacao$ )
2:    $alfas \leftarrow \{0.0, 0.1, 0.2, 0.3, 0.4, 0.5, 0.6, 0.7, 0.8, 0.9, 1.0\}$ 
3:    $nUsos \leftarrow somaDeCustos \leftarrow aProb \leftarrow []$ 
4:   para  $j \leftarrow 1$  até  $alfas.tamanho$  faça
5:      $nUsos[j] \leftarrow somaDeCustos[j] \leftarrow 0$ 
6:      $aProb[j] \leftarrow 1 / alfas.tamanho$ 
7:   fim para
8:    $custoMinimo \leftarrow \infty$ 
9:   para  $i \leftarrow 1$  até  $nIter$  faça
10:     $k \leftarrow$  índice aleatório de  $alfas$  com probabilidade  $aProb[k]$ 
11:     $s \leftarrow CONSTRUIRSOLUCAO(I, alfas[k])$ 
12:     $s \leftarrow BUSCALOCAL(s)$ 
13:    se  $s.custo < custoMinimo$  então
14:       $custoMinimo \leftarrow s.custo$ 
15:       $s^* \leftarrow s$ 
16:    fim se
17:     $nUsos[k] ++$ 
18:     $somaDeCustos[k] \leftarrow somaDeCustos[k] + s.custo$ 
19:    se  $(i \bmod nIterAteAtualizacao) = 0$  então
20:       $Q \leftarrow []$ 
21:      para  $j \leftarrow 1$  até  $alfas.tamanho$  faça
22:         $custoMedio \leftarrow somaDeCustos[j] / nUsos[j]$ 
23:         $Q[j] \leftarrow (custoMinimo / custoMedio)$ 
24:      fim para
25:       $somatorioQ \leftarrow \sum_{j=1}^{alfas.tamanho} Q[j]$ 
26:      para  $j \leftarrow 1$  até  $alfas.tamanho$  faça
27:         $aProb[j] \leftarrow Q[j] / somatorioQ$ 
28:      fim para
29:    fim se
30:  fim para
31:  retorna  $s^*$ 
32: fim método

```

---

### 3.5.4 VNS

O método VNS (Busca em Vizinhança Variável) foi proposto em [25] como uma meta-heurística simples e eficiente baseada na troca sistemática entre as estruturas de vizinhança dentro de uma busca local. Segundo os autores, ao contrário da maior parte dos métodos de busca locais existentes na época, o VNS não segue uma trajetória definida, mas sim explora vizinhanças gradativamente mais distantes da solução corrente, aceitando um novo vizinho apenas se uma melhora foi atingida.

Dentre os critérios de parada para o método, o que foi escolhido para o uso neste trabalho foi o de número de iterações sem melhora. Duas abordagens foram implementadas sendo a primeira delas (Seção 3.5.4.1), retirada de [41], tal que realiza movimentos simples (sempre aplica apenas um movimento) na fase de perturbação da solução corrente, variando a cada iteração a vizinhança usada na perturbação. A segunda abordagem (Seção 3.5.4.2) foi uma adaptação proposta aqui: a cada iteração do método uma vizinhança é escolhida de forma aleatória para o uso na perturbação e, durante a iteração, varia-se a intensidade da perturbação aplicada.

#### 3.5.4.1 Versão A: Perturbação simples, vizinhanças em ordem

Essa versão do método VNS foi implementada tal como o pseudocódigo no Algoritmo 5 e corresponde à implementação clássica do método VNS básico descrita em [17]. Além dos dados da instância  $I$  e uma solução inicial  $solucao$ , o método toma como parâmetro o valor  $maxIterSemMelhora$ , que determina sua parada (linha 5). Na linha 8, é sempre aplicado uma perturbação de um único movimento selecionado de forma aleatória na vizinha  $k$  corrente. Alguma técnica de busca local é utilizada na linha 9 para intensificar a solução e um procedimento semelhante ao descrito no método VDN (Seção 3.5.5) é aplicado nas linhas 10 até 15. Os condicionais nas linhas 17 até 22 registram a ocorrência ou não de melhora na solução corrente durante cada iteração.

Como este método fixa a ordem de consideração das estruturas de vizinhança a ordem utilizada para as vizinhanças N1, N2, N3 e N4 foi a mesma definida por [41]

(parcialmente baseada em [26]). Com relação à vizinhança N5 esta foi adicionada ao fim da ordenação, arbitrariamente.

---

**Algoritmo 5** VNS COM PERTURBAÇÃO SIMPLES EM MÚLTIPLAS VIZINHANÇAS

---

```

1: método VNSPERTSIMPLES( $I$ ,  $solucao$ ,  $maxIterSemMelhora$ )
2:    $s^* \leftarrow copia(solucao)$ 
3:    $nIterSemMelhora \leftarrow 0$ 
4:    $menorCustoAnterior \leftarrow s^*.custo$ 
5:   enquanto  $nIterSemMelhora < maxIterSemMelhora$  faça
6:      $k \leftarrow 1$ 
7:     enquanto  $k \leq I.nVizinhancas$  faça
8:        $s' \leftarrow$  vizinho aleatório de  $s^*$  na vizinhança  $k$  com 1 movimento
9:        $s \leftarrow BUSCALOCAL( $s'$ )$ 
10:      se  $s.custo < s^*.custo$  então
11:         $s^* \leftarrow s$ 
12:         $k \leftarrow 1$ 
13:      senão
14:         $k++$ 
15:      fim se
16:    fim enquanto
17:    se  $menorCustoAnterior \leq s^*.custo$  então
18:       $nIterSemMelhora ++$ 
19:    senão
20:       $nIterSemMelhora \leftarrow 0$ 
21:       $menorCustoAnterior \leftarrow s^*.custo$ 
22:    fim se
23:  fim enquanto
24:  retorna  $s^*$ 
25: fim método

```

---

#### 3.5.4.2 Versão B: Perturbação de tamanho incremental em vizinhança aleatória

Já esta versão do método recebe como parâmetro outro valor: o valor  $maxPertuT$  para o tamanho máximo a ser usado na perturbação de cada solução. O critério de parada e seu controle são idênticos ao do algoritmo anterior, mas na estrutura de repetição da linha 8 o valor em questão é o tamanho atual da perturbação a ser aplicada. Na linha 9, então, uma perturbação de tamanho  $m$  é aplicada na solução

$s^*$  na vizinhança fixa  $k$  escolhida aleatoriamente na linha 7 e, nas linhas 11 até 16, a variação do tamanho dessa perturbação é controlada (de uma maneira similar àquela com a qual a variação da vizinhança é controlada no Algoritmo 5) a fim de causar pertubações mais intensas de forma incremental.

---

**Algoritmo 6** VNS COM PERTURBAÇÃO INCREMENTAL EM VIZINHANÇAS

---

```

1: método VNSPERTINCRE( $I$ ,  $solucao$ ,  $maxPertuT$ ,  $maxIterSMelhora$ )
2:    $s^* \leftarrow copia(solucao)$ 
3:    $nIterSemMelhora \leftarrow 0$ 
4:    $menorCustoAnterior \leftarrow s^*.custo$ 
5:   enquanto  $nIterSemMelhora < maxIterSMelhora$  faça
6:      $m \leftarrow 1$ 
7:      $k \leftarrow$  vizinhança escolhida aleatoriamente
8:     enquanto  $m \leq maxPertuT$  faça
9:        $s' \leftarrow$  vizinho aleatório de  $s^*$  em  $k$  com  $m$  movimentos
10:       $s \leftarrow BUSCALOCAL( $s'$ )$ 
11:      se  $s.custo < s^*.custo$  então
12:         $s^* \leftarrow s$ 
13:         $m \leftarrow 1$ 
14:      senão
15:         $m++$ 
16:      fim se
17:    fim enquanto
18:    se  $menorCustoAnterior \leq s^*.custo$  então
19:       $nIterSemMelhora++$ 
20:    senão
21:       $nIterSemMelhora \leftarrow 0$ 
22:       $menorCustoAnterior \leftarrow s^*.custo$ 
23:    fim se
24:  fim enquanto
25:  retorna  $s^*$ 
26: fim método

```

---

### 3.5.5 VND

Como o ótimo local dentro de uma vizinhança pode não corresponder a um ótimo local em outra, durante a execução da busca local pode ser interessante a troca entre

as vizinhanças consideradas [17]. Assim, no método de busca local VND (Decida em Vizinhança Descendente), uma variação do VNS descrita em [17] cada vizinhança possível é explorada consecutivamente através de algum método de busca local e a melhor solução encontrada retornada. O Algoritmo 7 descrevendo o método tal como implementado segue abaixo.

---

**Algoritmo 7** VND

---

```

1: método VND( $I$ ,  $solucao$ )
2:    $s^* \leftarrow copia(solucao)$ 
3:    $k \leftarrow 1$ 
4:   enquanto  $k \leq I.nVizinhancas$  faça
5:      $s \leftarrow BUSCALOCAL_k(s^*)$ 
6:     se  $s.custo < s^*.custo$  então
7:        $s^* \leftarrow s$ 
8:        $k \leftarrow 1$ 
9:     senão
10:     $k++$ 
11:  fim se
12: fim enquanto
13: retorna  $s^*$ 
14: fim método

```

---

A estrutura de repetição iniciada na linha 4 garante que todas as vizinhanças sejam exploradas e, então, na linha 5 alguma técnica de busca local explorando especificamente a vizinhança  $k$  é aplicada sobre a melhor solução encontrada até o momento. O custo do vizinho retornado pela busca local é comparado ao melhor custo conhecido (linha 6) e, quando menor, a solução corrente (que é a melhor encontrada até então) é atualizada. Sempre que um vizinho melhor é encontrado em uma vizinhança qualquer, este é avaliado novamente a partir da primeira (linha 8). O método termina quando a busca local não encontrar vizinhos melhores em nenhuma vizinhança  $k$  para alguma solução, ou seja, quando um ótimo local é atingido com repetição a todas as estruturas.

### 3.5.6 *Simulated Annealing*

Segundo [13], a meta-heurística *Simulated Annealing* é um método probabilístico baseado em uma analogia com a termodinâmica do resfriamento de um conjunto de átomos aquecidos. Como outras das técnicas já descritas aqui, esta também parte de uma solução inicial e realiza um procedimento de busca local, mas este se dá de forma diferente dos demais.

A cada iteração, um vizinho da solução corrente (ou estado atual)  $s$  é gerado aleatoriamente e o novo estado, correspondente a  $s'$  possui uma variação de custo em relação a  $s$ , calculado na linha 14. Variações de melhora são incondicionalmente aceitas como novo estado atual  $s$  (linha 15), porém variações de piora podem ser aceitas com uma probabilidade que varia em função do valor de um parâmetro  $T$  (analogamente chamado de temperatura do sistema). Em todas as iterações, o melhor estado já atingido é armazenado para retorno (linha 18).

Entre os parâmetros passados ao método estão uma taxa de resfriamento  $\alpha$ , usada na linha 8 para definir o valor de  $T$  no início de cada iteração, o parâmetro  $T_0$ , que é o valor inicial atribuído a  $T$  (normalmente um valor muito alto), um parâmetro opcional *nReaquecimentos* (usado no *loop* da linha 4 para reaquecer o sistema quando a temperatura atinge valor igual a zero) e o número *maxIteracoes* de iterações a serem usadas em cada valor de  $T$ . Quando a temperatura chega a zero e o número de reaquecimentos é igual a *nReaquecimentos* o método termina e retorna a melhor solução encontrada.

O parâmetro *maxPertuT* faz parte de uma customização proposta neste trabalho que, em vez de realizar sempre um único movimento na linha 13, escolhe aleatoriamente uma quantidade de movimentos distinta em cada iteração.

**Algoritmo 8** SIMULATED ANNEALING

---

```

1: método SA(solucao,  $\alpha$ , nReaquecimentos, maxIteracoes, maxPerturT,  $T_0$ )
2:    $s \leftarrow s^* \leftarrow \text{copia}(solucao)$ 
3:   reaquecimento  $\leftarrow 0$ 
4:   enquanto reaquecimento  $\leq nReaquecimentos faça
5:      $T \leftarrow T_0$ 
6:      $t \leftarrow 0$ 
7:     enquanto  $T > 0$  faça
8:        $T \leftarrow T_0 * \alpha^t$ 
9:        $i \leftarrow 0$ 
10:      enquanto  $i < maxIteracoes$  faça
11:         $m \leftarrow$  número aleatório entre 1 e maxPerturT inclusive
12:         $k \leftarrow$  vizinhança escolhida aleatoriamente
13:         $s' \leftarrow$  vizinho aleatório de  $s$  em  $k$  com  $m$  movimentos
14:         $\Delta T \leftarrow s.custo - s'.custo$ 
15:        se  $e^{\Delta T/T} >$  valor aleatório entre 0 e 1 então
16:           $s \leftarrow s'$ 
17:          se  $s.custo < s^*.custo$  então
18:             $s^* \leftarrow s$ 
19:          fim se
20:        fim se
21:         $i++$ 
22:      fim enquanto
23:       $t++$ 
24:    fim enquanto
25:    reaquecimento  $\leftarrow$  reaquecimento + 1
26:  fim enquanto
27:  retorna  $s^*$ 
28: fim método$ 
```

---

**3.5.7 Explorando o paralelismo dos procedimentos de busca multi-início**

Métodos iterativos de busca *multi-start*, tal como o método GRASP e suas variações, possuem uma natureza intensamente paralela, já que consistem em múltiplas soluções geradas e exploradas de forma independente, uma a cada iteração. Assim, torna-se tentador explorar a existência generalizada de máquinas com múltiplas unidades de processamento.

Para isso, na ferramenta desenvolvida durante esse trabalho, os métodos iterativos foram implementados de forma a explorarem simultaneamente múltiplas soluções (tantas quanto for o número de processadores disponíveis), através da tecnologia de *multi-thread* amplamente presente atualmente.

O impacto dessa alteração sobre os pseudo-códigos apresentados anteriormente não são apresentados neste texto, pelo motivo de que não interferem na execução dos métodos. Na prática, tais adaptações apenas executam múltiplas iterações, já inherentemente independentes, de forma paralela.

# Capítulo 4

## Estudo de caso: DCC/UFRJ

O Departamento de Ciência da Computação da Universidade Federal Rural do Rio de Janeiro possui uma quantidade relativamente pequena de professores mas, ainda assim, no inicio de todo período letivo a elaboração manual da grade horária é um trabalho árduo que envolve grandes esforços de pessoal. Esta seção descreve o processo de coleta de informação e a definição das instâncias do Curso de Bacharelado em Ciência da Computação (correspondentes aos seis períodos letivos regulares referentes aos anos de 2015, 2016 e 2017) que foram utilizadas nos testes de estratégias apresentados nas próximas seções.

### 4.1 Descrição geral das instâncias baseadas no departamento

A ferramenta resultante da pesquisa aqui realizada é flexível na medida que permite a definição dos detalhes de cada instância quanto a todos as suas dimensões, como o número de dias por semana e de horas em um dia. As instâncias do Departamento de Ciência da Computação consideradas foram modeladas de forma a ter cinco dias por semana e cinto horários de inicio possíveis para o inicio de atividades (08:00, 10:00, 12:00, 14:00 e 16:00) já que as aulas do curso de Bacharelado em Ciência da Computação (ao menos nos dados coletados) sempre ocupam esses horários. Nada no modelo da ferramenta impede que que as instâncias fossem modeladas com definições mais granulares de horários, mas essa simplificação foi

adotada pelo simples fato de ser possível e benigna. A Tabela 4.1 exibe algumas das dimensões em que as instâncias diferem.

	2015-1	2015-2	2016-1	2016-2	2017-1	2017-2
Número de professores	13	14	13	13	15	14
Número de atividades	24	26	26	24	27	25

Tabela 4.1: Dimensões das instâncias

Cada uma das atividades cadastradas em uma instancia foi dividida em duas sub atividades, cada uma representando uma das aulas de uma turma e com duração igual a 1 período de tempo (que na prática tem duração de duas horas reais).

## **4.2 Apuração das preferências dos professores por disciplinas e requisitos**

Os níveis de preferência para cada uma das disciplinas obrigatórias e optativas oferecidas pelo DCC/UFRRJ foram coletados para todos os professores ligados ao departamento durante o desenvolvimento deste trabalho, através de um formulário *on-line*.

O formulário ofereceu, para cada disciplina, as opções equivalentes às descritas na Seção 4.3.3, possibilitando que cada professor indicasse seu nível de preferência por disciplina nos termos do modelo apresentado neste capítulo.

Além disso, a fim de tentar evitar ocasiões em que mais de um professor desejem utilizar o único laboratório presente do instituto com capacidade para turmas grandes, o formulário incluiu, para cada disciplina, um campo que permitia ao professor informar se, caso alocado àquela disciplina, desejava fazer uso desse recurso. As opções foram a) **Não**, b) **Sim, em apenas uma das aulas** e c) **Sim, em ambas as aulas**. Apesar da alocação de espaço físico não ter sido explorada neste trabalho, tal necessidade foi incluída no modelo na forma de uma restrição de tempo, descrita no item 3c da listagem na seção a seguir.

## **4.3 Restrições estudadas**

### **4.3.1 Restrições gerais a todas as instâncias**

Abaixo encontram-se enumeradas as restrições colhidas que se aplicam a todas as instâncias que foram utilizadas para os testes.

1. Grupo das restrições de viabilidade:

- (a) Restrições básicas: Alunos ou professores não podem estar alocados a múltiplas atividades no mesmo horário.
- (b) Períodos 1, 2, 3 e 4 de alunos periodizados só podem ter aulas durante a tarde.
- (c) Períodos 5, 6, 7, 8 e 9 de alunos periodizados só podem ter aulas durante a manhã.
- (d) Não existem aulas durante o horário de almoço (12h até 14h).
- (e) Nenhum professor pode ser alocado em mais do que 3 dias em uma semana.
- (f) Nenhum professor pode ser alocado a mais do que 2 disciplinas.
- (g) Todos os professores ativos devem ser alocados a pelo menos uma disciplina.
- (h) Horários já ocupados, em cada período, por turmas cujos os horários são definidos por outros departamentos não podem ser alocados a nenhuma atividade de tal período.

2. Grupo das restrições macias que pesam mais no cálculo da qualidade:

- (a) As preferências dos professores por ministrarem disciplinas específicas devem ser respeitadas.
- (b) Nenhuma atividade deve ter suas aulas marcadas com uma janela de tempo inferior a 1 dia.

- (c) Nenhum professor deve ter um intervalo entre o horário de início das suas atividades em um dia e o fim destas maior do que 8 horas.
  - (d) Nenhum professor deve ser alocado a atividades de disciplinas para as quais este declarou restrição máxima.
3. Grupo das restrições macias que pesam menos no cálculo da qualidade:
- (a) Nenhum professor deve ser alocado a atividades de disciplinas para as quais este declarou restrição parcial.
  - (b) Nenhum professor deve ser alocado a atividades de disciplinas para as quais este declarou indiferença.
  - (c) Pares de atividades referentes a disciplinas para as quais *algum* professor declarou precisar do uso do laboratório não devem sofrer choques de horário entre si.

Os pares de atividades aos quais a restrição 3c se refere são aquelas que, em cada instância, correspondem às disciplinas da listagem abaixo.

**Com uma só aula no laboratório:** Análise de Algoritmos, Arquitetura de Computadores I, Circuitos Digitais, Engenharia de Software I, Fundamentos de Sistemas, Gerencia de Projetos, Grafos e Algoritmos, Inteligência Artificial, Modelagem de Sistemas, Métodos Numéricos, Otimização Linear, Projeto de Sistemas, Redes de Computadores, Sistemas Distribuídos, Sistemas Operacionais, Tópicos Especiais em Otimização, Otimização Combinatória, Processamento de Linguagem Natural, Mineração de Dados, Processamento de Imagens, Introdução a Sistemas de Recomendação, Computação Gráfica II, Circuitos eletrônicos, Interfaces usuário-máquina, Introdução à computação quântica, Educação e informática, Criptografia, Otimização inteira e Engenharia de software II.

**Com ambas as aulas no laboratório:** Ambas as aulas, Arquitetura de Computadores II, Banco de Dados, Compiladores, Computação Gráfica, Computação

I, Computação II, Computação III, Estrutura de Dados I, Estrutura de Dados II, Linguagens de Programação, Tópicos Especiais em Banco de Dados, Tópicos Especiais em Ciência da Computação, Tópicos Especiais em Inteligência Artificial, Tópicos Especiais em Programação, Programação Paralela e Distribuída, Banco de Dados II, Desenvolvimento de Aplicações e Internet, Sistemas multimídia, Aprendizado de máquina e Informática aplicada ao ensino de matemática.

#### 4.3.2 Restrições específicas a cada instância

Uma prática comum ao elaborar os quadros de horários do DCC nos períodos letivos considerados foi de limitar o número de dias da semana aos quais cada professor doutorando está alocado a apenas no máximo dois, em vez de três (1e). Adicionalmente, professores que estão exercendo algum cargo administrativo durante o período ministram apenas uma disciplina. Assim, uma restrição extra para cada uma destas limitação foi adicionada para cada professor nas condições descritas, no grupo das restrições essenciais. O número de professores afetados em cada instância é mostrado na Tabela 4.2.

	2015-1	2015-2	2016-1	2016-2	2017-1	2017-2
Número de professores limitados a 2 dias	0	0	4	3	2	0
Número de professores limitados a 1 disciplina	0	0	0	0	2	3

Tabela 4.2: Professores afetados nas restrições extras

#### 4.3.3 Atendimento às preferência dos professores em forma de restrição

Além de permitir a alocação automática de professores às atividades, a contabilização das preferências dos professores em ministrar disciplinas específicas é uma importante adição aos algoritmos de programação horária desenvolvidos neste trabalho, atendendo às necessidades dos estudos de caso aqui realizados. Além disso, o enfoque nas preferência dos professores por disciplinas não é comum entre os problemas levantados na literatura, tendo sido encontrado, durante a realização deste trabalho, apenas em [18].

Para integrar o objetivo de atender às preferências dos professores, que em sua natureza é de maximização, ao modelo de minimização de violação de restrições aqui descrito, o nível de respeito de uma solução às preferências por disciplinas foi modelado sob a forma de uma única restrição, pertencente ao grupo de restrições importantes.

Foram definidos cinco níveis de preferência possíveis associando cada professor a cada uma das disciplinas:

**Preferência máxima** Muito interesse em ministrar a disciplina.

**Preferência parcial** Possibilidade mas pouco interesse.

**Indiferença** Sem preferência ou restrição.

**Restrição parcial** Precisa ser avisado com antecedência.

**Restrição máxima** Não há a possibilidade de ministrar a disciplina.

A violação das restrições nesse grupo é calculada de acordo com uma penalidade diferente por nível de preferência, que é aplicada a uma solução cada vez que um professor é alocado a uma disciplina do nível. Estas penalidades (expostas na Tabela 4.3) foram definidas de forma a associar uma maior qualidade (portanto menor custo total) à soluções onde alocações entre professores e as disciplinas que este declarou como mais interessantes sejam mais frequentes. Note que, em uma situação ideal, todos os professores em uma solução estariam alocados exclusivamente à atividades de disciplinas às quais este declarou preferência máxima e, com isso, o valor total da violação das restrições neste grupo seria igual a zero.

Tabela 4.3: Penalidades individuais por par (*professor, disciplina*)

	Penalidade
Preferência máxima	0
Preferência parcial	1
Indiferença	2
Restrição parcial	3
Restrição máxima	4

#### 4.3.4 Cálculo dos fatores de violação das restrições selecionadas

A tabela a seguir mostra como o fator de violação (Seção 3.4.1) de cada restrição da listagem da Seção 4.3 é calculado quando da avaliação da função objetivo sobre uma solução.

Tabela 4.4: Fatores de violação do grupo essencial

Restrição	Fator de violação
1a	Contagem de ocorrências de alocações onde um grupo de aluno OU um professor é alocado a mais de uma atividade em um mesmo horário.
1b	Contagem de alocações dos períodos 1, 2, 3 e 4 fora dos horários permitidos.
1c	Contagem de alocações dos períodos 5, 6, 7, 8 e 9 fora dos horários permitidos.
1d	Contagem de alocações de qualquer atividade alocada ao horário de almoço.
1e	Contagem de professores alocados a mais do que 3 dias em uma semana.
1f	Contagem de professores ativos alocados a mais do que 2 turmas.
1g	Contagem de professores ativos alocados a menos do que 1 turmas.
1h	Contagem de grupos de alunos alocados em horários previamente ocupados por disciplinas de outros departamentos.

Tabela 4.5: Fatores de violação do grupo importante

Restrição	Fator de violação
2a	A razão entre o somatório de professores alocados a cada disciplina (ponderado pelos valores da tabela 4.3) e o total de professores ativos.
2b	Somatório de: Para cada par de aulas de uma mesma atividade: se distância em dias entre as aulas $< 2$ , então $2 - \text{distância}$ .
2c	Somatório de professores alocados a mais de 8 horas em um mesmo dia.
2d	Somatório de alocações de professores à disciplinas a que ele declarou restrição máxima.

Tabela 4.6: Fatores de violação do grupo desejável

Restrição	Fator de violação
3a	Somatório de alocações de professores à disciplinas a que ele declarou restrição parcial.
3b	Somatório de alocações de professores à disciplinas a que ele declarou indiferença.
3c	Somatório de ocorrências de alocações das atividades no mesmo horário.

## 4.4 Experimentos e resultados

### 4.4.1 Ambiente computacional

Os experimentos descritos nesta seção foram realizados em um computador pessoal com 16GB de memória RAM (1600MHz) e um processador com 4 núcleos de 3.4GHz (Intel® Core™ i5-3570).

No entanto, apesar da utilização de uma máquina com múltiplos núcleos em

seu processador, o recurso de paralelismo descrito em 3.5.7 foi desativado a fim de facilitar a execução e comparação dos resultados. Dessa forma, a execução de cada experimento foi realizada de maneira sequencial por um único processo independente para cada instância.

#### 4.4.2 Pesos da função objetivo

Seguindo o estabelecido na Seção 3.4.3, os pesos dos grupos de restrição no cálculo da função objetivo foram fixados em 50, 20 e 1, para o grupo essencial, grupo importante e grupo desejável, respectivamente. É digno de nota que nenhuma dos trabalhos levantados na revisão da literatura exploram profundamente ou dão detalhes de como esses valores foram escolhidos. Logo, neste trabalho eles foram selecionados a partir de testes empíricos superficiais nos quais, para quase todas as instâncias, execuções esporádicas de diferentes métodos foram capazes de produzir soluções viáveis de menor custo que as manuais.

##### 4.4.2.1 Custo das soluções manuais reais

A tabela abaixo mostra o valor dos custos das soluções manuais viáveis que foram usadas pelo DCC para as instâncias estudadas, calculado segundo os peso fixados acima.

Instância	2015-1	2015-2	2016-1	2016-2	2017-1	2017-2	Média
Custo	311.38	274.14	311.46	151.23	97.33	126.14	211,95

Tabela 4.7: Custos das soluções manuais para as instâncias.

#### 4.4.3 Características gerais do experimentos

A fim de tentar observar o comportamento geral das estratégias elaboradas cada uma delas foi executada em múltiplas rodadas e os valores médios foram coletados. Além disso, os valores médios entre todas as instâncias também foram avaliados, com o intuito de entender o desempenho de cada estratégia em uma instância **típica** do

DCC. Este conceito é fundamental neste trabalho pois os experimentos foram realizados com foco em melhor atender o departamento em instâncias futuras e não na obtenção das melhores soluções possíveis para cada instância do passado, individualmente. Esse interesse prático faz também com que, na comparação qualitativa entre duas estratégias, a importância relativa da medida de tempo de execução seja avaliada sob uma perspectiva não linear. Ou seja, estratégias cujas execuções duram horas, por exemplo, são consideradas inferiores quando existe uma estratégia que execute em questão de minutos e leve a soluções viáveis, mesmo que aquelas produzam soluções viáveis de menor custo.

A legenda abaixo mostra o significado dos valores apresentados nas tabelas de resultados.

Nas tabelas	Significado
C	Valor médio entre as rodadas do custo da melhor solução produzida em cada uma.
V	Valor médio entre as rodadas do fator de violação das restrições essências da melhor solução produzida em cada uma.
R	Valor médio entre as rodadas do número médio de falhas/reinícios da construção em cada uma.
T	Valor médio entre as rodadas do tempo total de execução da estratégia, em minutos.
I	Iteração média com ocorreu a última melhora.

Tabela 4.8: Legenda dos valores nas tabelas de resultados.

#### 4.4.4 Experimentos exploratórios

Devido a grande quantidade de técnicas implementadas durante este trabalho e ao fato de cada uma delas possuir diversos parâmetros específicos, foi elaborada uma fase de experimentos exploratórios com um conjunto de estratégias. Nesta fase, um pequeno número de valores possíveis para os parâmetros de cada método foi testado.

Estes primeiros experimentos incluíram as estratégias com o método *Simulated annealing*, a combinação GRASP usando VND na fase de busca local e a técnica VNS, também usando VND para a busca. Além destas, as duas alternativas de construção usadas na fase correspondente do método GRASP também foram investigadas isoladamente. A técnica VND foi sempre utilizada aqui na fase de busca

local pelo motivo de evitar a necessidade de explorar as vizinhanças individualmente, levando a uma quantidade significativamente menor de experimentos.

#### *4.4.4.1 Métodos de construção*

Os primeiros experimentos visam fazer uma comparação entre os dois métodos propostos para a fase construtiva da técnica GRASP. Para isso, ambos foram executados em todas as instâncias sem a realização da fase de busca local nas soluções construídas.

Foram utilizados os seguintes valores fixos para o parâmetro alfa: 0.0 (equivalente à solução aleatória), 0.1, 0.2, 0.3, 0.4, 0.5, 0.6, 0.7, 0.8, 0.9 e 1.0. Além disso, a versão reativa do GRASP também foi executada, com o número de iterações antes da atualização da probabilidade dos valores de alfa escolhidos fixada em 10. Além disso, para todas as execuções o número de iterações foi fixado em 100 e o número de rodadas em 10.

Como estes experimentos não incluem a fase de busca local não são exibidos os valores médios dos custos das soluções, mas em seu lugar o número médio de vezes que o método de construção falha e é reiniciado.

#### 4.4. EXPERIMENTOS E RESULTADOS

51

Instância	Por grupo de estudantes							Por atividades							
	2015-1	2015-2	2016-1	2016-2	2017-1	2017-2	Média	2015-1	2015-2	2016-1	2016-2	2017-1	2017-2	Média	
Alfa: 0,0	V	6,4	6,7	13	9,8	11,1	8,6	9,3	29,8	24,2	24,7	30,9	17,5	17,7	24,1
	R	3,45	1,63	0,48	3,51	2,83	0,44	2,06	0	0	0	0	0	0	<b>0</b>
Alfa: 0,1	V	3,9	4,3	8,1	5,4	6,7	4,5	5,48	24,1	17,6	19,9	25,5	14,1	14	19,2
	R	3,57	1,68	0,4	3,23	2,57	0,45	1,98	0	0	0	0	0	0	<b>0</b>
Alfa: 0,2	V	3,1	2,8	8,6	4,2	5,2	4,4	4,7	21,9	16,6	17,1	22,5	13	12,5	17,3
	R	3,54	1,61	0,46	3,29	2,61	0,48	2	0	0	0	0	0	0	<b>0</b>
Alfa: 0,3	V	3,7	3,3	7	5,2	5,5	4,3	4,8	19,6	14,9	15,3	19	10,5	12,5	15,3
	R	3,5	1,54	0,39	3	2,65	0,48	1,93	0	0	0	0	0	0	<b>0</b>
Alfa: 0,4	V	3,4	3,1	7,4	4,6	5,1	3,7	4,5	16,2	11	12,9	17,4	9,1	10	12,8
	R	3,38	1,53	0,41	3,15	2,36	0,47	1,88	0	0	0	0	0	0	<b>0</b>
Alfa: 0,5	V	2,8	2,9	6,5	4,9	5	3,4	4,2	12,5	8,5	12,3	13,8	8,4	7,6	10,5
	R	3,13	1,61	0,39	2,94	2,58	0,5	1,86	0	0	0	0	0	0	<b>0</b>
Alfa: 0,6	V	2,6	1,4	5,7	3,3	3,5	2,9	3,2	9,3	7	8,9	10,9	5,5	5,8	7,9
	R	2,95	1,47	0,4	2,87	2,13	0,49	1,72	0	0	0	0	0	0	<b>0</b>
Alfa: 0,7	V	2,1	2,1	5,7	3,9	3,4	2,9	3,3	6,9	4,4	7,7	6,3	5,5	3,6	5,7
	R	2,78	1,42	0,37	2,57	2,27	0,42	1,64	0	0	0	0	0	0	<b>0</b>
Alfa: 0,8	V	2,5	1,3	4,8	2,9	2,8	2,2	<b>2,75</b>	4,3	2,9	7,3	4,4	3,9	2,5	4,2
	R	2,5	1,44	0,34	2,25	1,92	0,48	<b>1,49</b>	0	0	0	0	0	0	<b>0</b>
Alfa: 0,9	V	2,4	1,2	4,1	2,8	2,8	2	<b>2,55</b>	4	2,5	5,9	2,8	3,3	1,8	<b>3,4</b>
	R	2,2	1,27	0,36	1,88	1,56	0,41	<b>1,28</b>	0	0	0	0	0	0	<b>0</b>
Alfa: 1,0	V	1,7	0,8	4,6	3	2,4	1,7	<b>2,4</b>	3,6	2,1	5,5	2,7	3	1,1	<b>3</b>
	R	2,22	1,52	0,37	1,42	1,46	0,39	<b>1,23</b>	0	0	0	0	0	0	<b>0</b>
Reativo	V	2,5	1,4	4,9	3,6	3,2	2,6	3	4,8	2,7	7,1	3	3,3	2,3	<b>3,9</b>
	R	3,12	1,59	0,36	2,47	2,09	0,43	1,67	0	0	0	0	0	0	<b>0</b>

Tabela 4.9: Resultados médios de construções com 100 iterações. Os três melhores valores de cada métrica por coluna de média geral foram grifados.

Como é possível observar na tabela 4.9, o método de construção por atividades teve a característica de não precisar ser reiniciado nenhuma vez, com nenhum dos valores para o parâmetro alfa. Por outro lado, ele levou à construção de soluções mais distantes da viabilidade (conceito medido aqui por "V"), quando comparado ao método de construção por grupo de estudantes, em todas as execuções.

Além disso, é possível observar que configurações mais gulosas dos métodos de construção não levaram apenas à distâncias para a viabilidade menores (algo a se esperar já que esta se relaciona diretamente ao custo), mas também a uma frequência de reinício menor.

Com estas observações, os dois valores para alfa com menores valores foram investigados novamente, agora com 1000 iterações. Além deles, o método reativo também foi incluído, pelos motivos de ter atingido valores próximos ao menores e

de que a técnica reativa parece interessante quando se espera encontrar estratégias genéricas que potencialmente possam atender instâncias novas com outras características, no futuro. Por questões práticas o numero de rodadas foi reduzido, a partir daqui, para 3.

Instância	Por grupo de estudantes							Por atividades							
	2015-1	2015-2	2016-1	2016-2	2017-1	2017-2	Média	2015-1	2015-2	2016-1	2016-2	2017-1	2017-2	Média	
Alfa: 0,9	V	1	0,3	3,3	1	2	0,7	<b>1,38</b>	2,7	1,3	4,7	2,7	2,3	1	2,45
	R	2,26	1,27	0,33	1,91	1,6	0,42	1,30	0	0	0	0	0	0	<b>0,00</b>
Alfa: 1,0	V	1,3	0	4,7	2	1,3	1	1,72	2,3	0,7	4,7	2	2	1,7	<b>2,23</b>
	R	2,1	1,5	0,36	1,62	1,47	0,39	<b>1,24</b>	0	0	0	0	0	0	<b>0,00</b>
Reativo	V	1,3	0,3	4,7	2,7	1	1	1,83	2,7	1,7	5,3	2,3	2,3	2	2,72
	R	2,88	1,57	0,38	2,61	2,06	0,45	1,66	0	0	0	0	0	0	<b>0,00</b>

Tabela 4.10: Resultados médios de construções com 1000 iterações. O melhor valor de cada métrica por coluna de média geral foi grifado.

O resultados mostram que o comportamento relativo observado se mantém com o novo número de iterações, para ambos os métodos. As probabilidades de reinício se mantiveram similares, porem analisar o efeito prático da ocorrência de falha e reinício dos métodos não é possível nesse estudo, uma vez que no caso do DCC a probabilidade de ocorrência é baixa. Em termos práticos, a probabilidade de falha do método por grupos é desprezível no caso das instâncias estudadas. Quanto à distância para a viabilidade, o maior número de iterações levou a soluções melhores para ambos os métodos e a superioridade da construção por grupos nesse quesito também se manteve.

#### 4.4.4.2 GRASP+VND

A seguir, experimentos nos mesmos moldes que os anteriores foram executados, mas incluindo a fase de busca local do método GRASP, omitida anteriormente. Como estratégia de busca local foi selecionado a heurística VND e como técnica de busca local foram comparados *best improving* e *first improving* em termos de custos médios encontrados e, assim como antes, a distância para a viabilidade. Ambos os métodos de construção foram executados a fim de investigar o impacto que estes tem na qualidade das soluções encontradas.

A tabela 4.11 mostra os resultados das execuções fazendo uso da busca local *best improving*.

Instância	Por grupo de estudantes						Por atividades							
	2015-1	2015-2	2016-1	2016-2	2017-1	2017-2	Média	2015-1	2015-2	2016-1	2016-2	2017-1	2017-2	Média
C	176,31	133,71	308,15	130,08	143,56	86,52	163,06	174,87	146,33	286,74	157,85	216,44	111,57	182,30
Alfa: 0,0 V	0,3	0,0	2,7	0,7	0,7	0,0	0,73	1,0	0,0	3,3	1,0	1,7	0,0	1,17
T	4,6	4,4	4,2	3,6	4,4	4,1	4,2	6,4	6,0	5,5	4,8	5,1	5,0	5,5
C	154,79	135,29	286,49	98,08	159,78	89,38	153,97	229,97	152,76	276,51	175,54	193,11	101,95	188,31
Alfa: 0,1 V	0,7	0,0	2,0	0,0	0,0	0,0	0,45	1,7	0,0	2,0	1,3	0,7	0,3	1,00
T	3,7	3,7	3,5	2,8	3,5	3,3	3,4	5,7	5,5	4,9	4,4	4,7	4,6	5,0
C	161,15	116,48	228,69	118,23	151,11	105,19	146,81	182,90	117,52	273,44	149,90	188,67	108,52	170,16
Alfa: 0,2 V	0,0	0,0	2,0	0,0	0,3	0,0	0,38	0,7	0,0	2,0	1,3	1,0	0,0	0,83
T	3,6	3,7	3,3	2,8	3,4	3,3	3,4	5,5	5,2	4,8	4,3	4,5	4,4	4,8
C	198,28	126,00	262,46	169,21	191,89	69,10	169,49	173,90	137,14	308,77	120,56	210,33	127,33	179,67
Alfa: 0,3 V	0,0	0,0	2,3	0,0	0,3	0,0	0,43	0,3	0,0	3,0	0,3	1,3	0,3	0,87
T	3,5	3,5	3,3	2,7	3,4	3,2	3,3	5,3	5,0	4,4	4,1	4,2	4,2	4,5
C	148,46	135,24	238,79	86,62	142,11	96,33	141,26	164,64	144,76	243,36	123,13	192,56	99,52	161,33
Alfa: 0,4 V	0,3	0,0	1,0	0,0	0,0	0,0	<b>0,22</b>	0,3	0,0	2,0	0,3	1,3	0,0	<b>0,65</b>
T	3,5	3,4	3,1	2,6	3,2	3,1	3,2	4,9	4,6	4,1	3,9	3,9	3,9	4,2
C	164,23	131,14	279,97	87,77	222,22	89,05	162,40	211,59	145,71	272,79	110,10	185,44	110,05	172,61
Alfa: 0,5 V	0,0	0,0	1,7	0,3	0,0	0,0	<b>0,33</b>	0,7	0,0	2,0	1,0	0,3	0,0	<b>0,67</b>
T	3,3	3,2	3,0	2,5	3,1	2,9	3,0	4,4	4,2	3,8	3,4	3,7	3,5	3,8
C	155,00	99,33	257,67	91,05	181,11	90,05	145,70	171,26	122,24	289,90	162,31	174,33	85,62	167,61
Alfa: 0,6 V	0,0	0,0	2,3	0,0	0,0	0,0	0,38	0,7	0,0	3,3	1,7	1,0	0,0	1,12
T	3,3	3,1	2,8	2,4	2,9	2,8	2,9	4,1	3,7	3,4	3,1	3,3	3,3	3,5
C	175,72	100,05	281,36	92,59	136,11	97,95	147,30	192,62	127,67	279,51	146,10	178,89	78,90	167,28
Alfa: 0,7 V	0,0	0,0	1,7	0,0	0,3	0,0	<b>0,33</b>	0,0	0,0	2,7	1,3	0,3	0,0	<b>0,72</b>
T	3,2	2,9	2,7	2,4	2,8	2,8	2,8	3,6	3,3	3,1	2,7	3,0	2,8	3,1
C	112,72	99,05	231,82	83,05	220,00	69,33	<b>136,00</b>	179,87	125,05	235,69	140,36	164,22	106,10	<b>158,55</b>
Alfa: 0,8 V	0,3	0,0	1,3	0,0	0,0	0,0	<b>0,27</b>	0,3	0,0	2,7	0,3	0,7	0,3	<b>0,72</b>
T	3,0	2,9	2,5	2,3	2,6	2,6	<b>2,7</b>	3,2	2,8	2,8	2,3	2,6	2,6	2,7
C	145,95	75,19	209,05	69,64	115,22	52,57	<b>111,27</b>	194,31	124,57	231,05	85,56	149,56	65,95	<b>141,83</b>
Alfa: 0,9 V	0,0	0,0	2,3	0,0	0,3	0,0	0,43	0,7	0,3	2,0	0,0	1,0	0,0	<b>0,67</b>
T	2,9	2,7	2,4	2,2	2,5	2,6	<b>2,6</b>	2,9	2,2	2,6	1,9	2,3	2,1	<b>2,3</b>
C	131,92	60,57	184,82	61,69	112,44	49,33	<b>100,13</b>	167,03	75,00	246,62	93,31	149,22	60,81	<b>132,00</b>
Alfa: 1,0 V	0,3	0,0	1,7	0,0	1,0	0,0	0,50	1,3	0,0	3,0	1,0	0,7	0,0	1,00
T	2,4	2,3	2,2	2,0	2,1	2,1	<b>2,2</b>	2,4	1,5	2,1	1,4	1,9	1,4	<b>1,8</b>
C	141,85	99,67	256,74	98,54	202,33	66,19	144,22	182,15	144,76	265,23	155,82	182,78	78,57	168,22
Reativo V	0,0	0,0	2,3	1,0	0,0	0,0	0,55	0,3	0,0	2,3	1,3	0,7	0,0	0,77
T	3,1	3,1	3,0	2,6	3,2	3,1	3,0	4,5	3,3	3,1	3,2	3,4	3,5	3,5

Tabela 4.11: Resultados médios GRASP + VND com *best improving* e 100 iterações.

Os três melhores valores de cada métrica por coluna de média geral foram grifados.

Os resultados na tabela 4.11 começam a mostrar que o método de construção por grupos é capaz de levar a soluções com menor custo que a alternativa por atividades. Em todas execuções tanto o custo médio quanto a distância para a viabilidade daquele superaram os valores deste.

Adicionalmente, seguindo o mesmo padrão das soluções construídas no experimento anterior, esses resultados mostram que o método GRASP com a fase construtiva mais próxima do comportamento guloso levam a soluções melhores em média.

No entanto, é possível notar que a produção de soluções viáveis com a combinação GRASP+VND+*Best Improving* não parece ser uma opção viável na prática, levando com frequência a valores de "V" maiores que zero. De fato, até mesmo nas execuções com menores custos o valor médio da violação das restrições de viabilidade foi maior que zero, indicando como a estratégia falhou em produzir uma solução viável em ao menos uma rodada, para mais de uma instância.

Pela perspectiva de tempo de execução os dois métodos de construção são comparáveis, com valores próximos especialmente para as configurações mais gulosas. Quando comparadas as configurações de alfa, em ambos os casos valores maiores de alfa também levam a menores tempos de execução.

Tal como na seção anterior (e pelos mesmos motivos) os dois melhores valores do parâmetro alfa, além da versão reativa, foram executados novamente, com 1000 iterações. Os novos resultados (4.12) da execução com 1000 iterações levaram a custos médios menores e também a valores médios de viabilidade melhores. De fato, para cinco das seis instâncias todas as três execuções utilizando a construção por grupos produziram soluções viáveis. Já nas execuções com o método de construção por atividades o método manteve um desempenho quanto à viabilidade inferior à sua alternativa.

Adicionalmente, os tempos de execução médios forma computados, mostrando certa equivalência na comparação entre os métodos de construção e que a configuração mais gulosa da fase construtiva leva a execuções consideravelmente mais rápidas.

Outra medida colhida foi o número médio da iteração do GRASP em que estratégia visitou uma nova melhor solução. Esses valores mostram que quando procedimento de construção por grupo de estudantes é usado um número menor de iterações é necessário para atingir soluções de menor custo do que no caso do procedimento

por atividades. Outro fenômeno a se observar é que no GRASP Reativo o número de iterações é maior, o que é um preço a se pagar pela vantagem de não se selecionar o valor de alfa.

Instância	Por grupo de estudantes							Por atividades							
	2015-1	2015-2	2016-1	2016-2	2017-1	2017-2	Média	2015-1	2015-2	2016-1	2016-2	2017-1	2017-2	Média	
Alfa: 0,9	C	78,44	49,57	158,31	39,44	88,89	25,48	73,36	118,26	72,05	210,62	37,41	99,89	44,05	97,05
	V	0	0	0,3	0	0	0	<b>0,05</b>	0	0	1	0	0	0	<b>0,17</b>
	I	367	690,3	224,3	233	519,3	317	<b>391,8</b>	357,3	889	299,7	447,3	402,7	758,7	525,8
	T	28,57	43,60	37,91	34,11	39,64	40,49	37,39	29,35	40,01	38,51	29,29	36,51	32,30	34,33
Alfa: 1,0	C	74,72	45,00	155,49	35,03	68,00	28,38	<b>67,77</b>	105,87	58,43	179,23	66,46	107,33	26,14	<b>90,58</b>
	V	0	0	1,7	0	0	0	0,28	0,3	0	1,3	0	0,3	0	0,32
	I	460,7	441,7	565,7	274,7	260,7	437,7	406,9	630	437	312	505,7	670,7	359,7	<b>485,9</b>
	T	15,22	23,24	22,55	20,42	21,01	21,64	<b>20,68</b>	15,20	20,02	20,97	14,14	18,33	14,59	<b>17,21</b>
Reativo	C	103,08	60,00	203,51	44,77	99,56	32,57	90,58	102,54	88,62	215,95	46,79	107,11	31,62	98,77
	V	0	0	2	0	0	0	0,33	0	0	1,7	0	0	0	0,28
	I	570,3	621	654	706	670	440,7	610,3	415,7	771	298,3	619,7	564	300,3	494,8
	T	20,07	32,27	30,01	25,77	30,76	30,38	28,21	26,91	39,58	35,65	32,92	32,66	35,67	33,90

Tabela 4.12: Resultados médios de GRASP+VND com *best improving* com 1000 iterações. O melhor valor de cada métrica por coluna de média geral foi grifado.

A seguir experimentos nos mesmos moldes foram realizados utilizando a busca local *first improving*, com 100 iterações para o conjunto maior de valores de alfa e 1000 iterações para os valores com melhores desempenhos.

#### 4.4. EXPERIMENTOS E RESULTADOS

56

Instância	Por grupo de estudantes							Por atividades						
	2015-1	2015-2	2016-1	2016-2	2017-1	2017-2	Média	2015-1	2015-2	2016-1	2016-2	2017-1	2017-2	Média
C	185,92	130,24	277,05	124,92	178,44	84,33	163,48	212,54	145,95	326,03	163,77	194,22	147,14	198,28
Alfa: 0,0	V	0,7	0	2,7	0,7	0	0,68	1,7	0	3,7	1,3	1,7	0	1,40
	T	2,6	3,6	3,5	2,9	3,8	3,3	2,7	4,0	4,0	3,5	4,0	3,7	3,6
C	187,87	113,95	294,85	199,13	182,11	87,14	177,51	178,69	140,24	311,33	133,21	188,11	107,81	176,57
Alfa: 0,1	V	0,7	0	2,7	0	0	0,57	1	0	3	1	1,3	0,7	1,17
	T	2,1	2,8	2,8	2,1	2,8	2,5	2,4	3,5	3,6	3,2	3,6	3,4	3,3
C	178,62	138,43	316,41	102,23	198,44	98,24	172,06	187,77	125,71	240,38	124,79	176,33	139,52	165,75
Alfa: 0,2	V	0	0	3,3	0,3	0	0,60	0,3	0	2,7	0,7	0,3	0	0,67
	T	2,0	2,7	2,7	2,0	2,7	2,4	2,3	3,3	3,4	3,0	3,3	3,2	3,1
C	189,77	117,76	298,77	91,08	161,22	103,1	160,28	187,1	116,33	287,31	117,82	183,22	92,52	164,05
Alfa: 0,3	V	0,3	0	2,7	0	0,7	0,62	1	0,3	2,7	1	1,7	0	1,12
	T	1,9	2,5	2,5	1,9	2,6	2,3	2,1	3,1	3,2	2,8	3,1	3,0	2,9
C	216,87	121,24	260,97	123,46	210,56	76,1	168,20	262,56	128,24	261,97	130,41	170,67	89,38	173,87
Alfa: 0,4	V	0	0	2	0,3	0	0,38	0,7	0	2	1,3	0,7	0	0,78
	T	1,7	2,2	2,4	1,7	2,5	2,1	1,9	2,8	2,9	2,5	2,8	2,7	2,6
C	184,05	122,62	314,38	174,67	163,22	83,62	173,76	199,9	137,52	300,28	143,38	185,89	112,29	179,88
Alfa: 0,5	V	0	0	3	0	0,7	0,62	1	0,3	2,7	1,3	1,3	0	1,10
	T	1,6	2,0	2,2	1,7	2,3	2,0	1,7	2,3	2,6	2,3	2,5	2,4	2,3
C	137,59	97,48	267,82	63,77	136,11	84,57	131,22	203,15	161,67	272,59	136,15	147,33	115,48	172,73
Alfa: 0,6	V	0	0	2	0	0,3	0,38	1,0	0,0	1,3	0,3	0,7	0,0	0,55
	T	1,5	2,0	2,1	1,5	2,2	1,9	1,6	2,0	2,3	2,0	2,3	2,1	2,0
C	151,74	118,71	256,64	115,87	179,44	83,76	151,03	213,87	143,48	278,49	109,46	215,22	98,52	176,51
Alfa: 0,7	V	0,7	0	1,7	0	0	0,40	1	0	2,3	0,3	0,7	0	0,72
	T	1,4	1,9	2,0	1,5	2,0	1,8	1,5	1,8	2,0	1,7	2,0	1,9	1,8
C	168,28	137,19	233,33	58,1	163,11	62,43	137,07	161,51	108,9	236,03	118,46	166,33	91,24	147,08
Alfa: 0,8	V	0,3	0	1,7	0	0	0,33	0,7	0	2	0,7	0,3	0	0,62
	T	1,4	1,8	1,8	1,5	1,9	1,8	1,3	1,6	1,9	1,5	1,9	1,7	1,7
C	130,05	79	206,77	52,1	142,22	39,86	108,33	173,82	140	233,97	122,36	163,67	74,81	151,44
Alfa: 0,9	V	0	0	1,7	0	0,3	0,33	0,7	0	2	0,7	1,3	0	0,78
	T	1,3	1,7	1,7	1,3	1,8	1,6	1,2	1,5	1,7	1,3	1,6	1,5	1,5
C	105,15	65,05	220,87	65,79	110,78	46,76	102,40	169,49	84,52	251,28	114,72	136,67	89,95	141,11
Alfa: 1,0	V	0	0	2	0	0	0,33	0,3	0,3	3	0,7	0,7	0	0,83
	T	1,1	1,4	1,6	1,3	1,5	1,3	1,1	1,1	1,6	1,1	1,3	1,1	1,2
C	160,56	85,67	252,44	103,18	149,67	81,14	138,78	183,44	128,52	277,21	136,67	143,44	77	157,71
Reativo	V	0,3	0	2	0	0	0,38	1,3	0	3	0,3	0,3	0	0,82
	T	1,6	2,1	2,3	1,7	2,5	2,0	1,8	2,3	2,7	2,5	2,4	2,7	2,4

Tabela 4.13: Resultados médios GRASP + VND com *first improving* e 100 iterações.

Os três melhores valores de cada métrica por coluna de média geral foram grifados.

Instância	Por grupo de estudantes							Por atividades							
	2015-1	2015-2	2016-1	2016-2	2017-1	2017-2	Média	2015-1	2015-2	2016-1	2016-2	2017-1	2017-2	Média	
Alfa: 0,9	C	69,87	53,95	181,03	32,77	73,11	37,29	74,67	108,74	69,81	170,31	45,44	98,56	42,05	89,15
	V	0	0	2	0	0	0	0,33	0	0	1	0	0	0	<b>0,17</b>
	I	494,3	421,3	359	210	599	582,3	<b>444,3</b>	629	547,7	662	605	549	725,3	619,7
	T	22,64	29,26	27,29	21,29	28,44	25,86	25,80	21,25	27,87	27,21	20,63	26,87	24,07	24,65
Alfa: 1,0	C	81,38	43,71	165,03	31,31	80	19,14	<b>70,10</b>	99,49	65,71	178,41	48,36	114,44	25,48	<b>88,65</b>
	V	0	0	1	0	0	0	<b>0,17</b>	0	0	1,7	0	0,3	0	0,33
	I	340,3	598,7	474	464,7	564,3	543	497,5	356,7	783,7	514,7	631,3	532,3	452,3	545,2
	T	18,62	23,53	24,61	18,66	22,23	20,69	<b>21,39</b>	17,63	20,59	23,60	15,93	20,92	17,57	<b>19,37</b>
Reativo	C	98,79	54,9	181,92	46,13	98,22	28,38	84,72	132,92	94,33	216,15	72,1	101,67	44,52	110,28
	V	0	0	1	0	0	0	<b>0,17</b>	0	0	1,7	0	0	0	0,28
	I	461	577	544,3	618,3	543,3	820,3	594,0	359,3	283	393,7	257,7	432	701,3	<b>404,5</b>
	T	27,89	36,01	33,03	26,57	36,35	32,21	32,01	30,80	41,48	40,01	30,87	39,97	36,98	36,69

Tabela 4.14: Resultados médios de GRASP+VND com *first improving* com 1000 iterações. O melhor valor de cada métrica por coluna de média geral foi grifado.

As estratégias usando *first improving* apresentaram um comportamento bem similar às usando *best improving*, tanto quanto à viabilidade quanto aos custos, não sendo clara nenhuma vantagem de uma estratégia sobre a outra.

Quanto à comparação de tempos os comportamentos foram similares aos das estratégias com *best improving*, mas a diferença entre os parâmetros mais e menos gulosos foi menor que no experimento anterior.

Observando as iterações médias de última melhora nota-se que o método reativo requer um número maior de iterações até convergir a melhores soluções. Isso não é surpresa, já que o método alternativo mantém o parâmetro alfa fixo em valores nos quais já é sabido que ele leva a melhores resultados, enquanto o GRASP Reativo realiza um processo de ajuste de tal parâmetro ao longo das iterações.

Diante da avaliação dos resultados anteriores, a tabela abaixo mostra os valores obtidos em um conjunto de experimentos usando apenas alguns nos métodos que obtiveram melhor desempenho. Estes são: a busca local *best improving* e o procedimento de construção por grupo de alunos. O valores para o parâmetro alfa foram mantidos, mas um dos parâmetros do GRASP Reativo, aquele que define o número de iterações até que a atualização da probabilidade de escolha de cada valor de alfa seja atualizada e que foi fixado em 10 nos experimentos até aqui, foi executado

também com valores 20 e 40. O objetivo é investigar se a variação do parâmetro é capaz de levar as soluções do GRASP Reativo a valores mais próximos das do GRASP com configurações mais gulosas, mesmo que em maior número de iterações. Adicionalmente, o número de iterações foi alterado para 600, no caso do método reativo, e 400 nos outros (valores determinados pela aproximação das médias gerais das tabelas anteriores).

Os resultados (4.15) mostram que a variação do parâmetro do no GRASP Reativo não parece gerar uma alteração significativa na qualidade das soluções e que o valor mais alto levou a um número de iterações maior sem ganho em custo.

Quando observa-se as estratégias com alfa igual a 0.9 e 1.0, o novo valor para o número de iterações parece ter levado ao uma perda modesta na qualidade média das soluções. Uma possível explicação pode estar no número de rodadas usado nos experimentos, que pode ter levado a uma variabilidade grande nos resultados médios que foram usados para definir esse parâmetro. Além disso, como o valor de 400 para o número de iterações foi escolhido usado a média por todas as instâncias, este valor é menor do que a média para algumas instâncias individuais.

Instância		2015-1	2015-2	2016-1	2016-2	2017-1	2017-2	Média
Alfa: 0,9	C	89,21	59,62	204,44	53,13	90,67	38,9	89,33
	V	0	0	1,7	0	0	0	0,28
	T	7,93	12,23	10,73	9,62	11,24	11,27	10,50
	I	132,3	186,7	206,7	209,7	166,3	113,7	169,23
Alfa: 1,0	C	78,95	52,33	179,92	30,72	95,44	20,1	76,24
	V	0	0	2	0	0	0	0,33
	T	6,5	9,96	9,95	8,7	8,97	9,28	8,89
	I	207,3	153	182	86	55	367,7	175,17
Reativo com 10	C	96,87	67,9	198,56	45,05	89,11	43,62	90,19
	V	0	0	1,3	0	0	0	0,22
	T	13,86	21,99	20,28	16,51	19,63	19,03	18,55
	I	212	344	496,3	94,7	281	174	267,00
Reativo com 20	C	120,87	67,86	182,62	47,46	95,22	30,95	90,83
	V	0	0	2,3	0	0	0	0,38
	T	13,82	20,96	19,4	16,67	20,51	18,97	18,39
	I	218	216	330,7	183,7	281,3	362	265,28
Reativo com 40	C	125	65,14	172,97	48,33	123,67	46,14	96,88
	V	0	0	1,7	0	0	0	0,28
	T	13,62	20,84	19,49	16,62	19,71	19,09	18,23
	T	331,7	441,3	446,7	404	260	50,3	322,33

Tabela 4.15: Resultados médios das execuções de GRASP+VND com construção por grupos e busca local *best improving*.

#### 4.4.4.3 VNS Versão A/VNS Versão B

As duas variações do método VNS implementadas foram executadas partindo de soluções aleatórias. Para isso, 100 soluções foram geradas em cada rodada usando ambos os métodos construtivos já descritos, com alfa igual a zero, e a melhor entre elas foi usada como estado inicial do método. Cada uma das versões foi combinada com a heurística VND, em um momento usando *best improving* e em outro *first improving*. As tabelas a seguir apresentam os resultados para execuções com três

valores para o critério de parada (o número de iterações sem melhora): 50, 100 e 150. No caso do parâmetro extra presente na versão B da técnica, que define o tamanho máximo da perturbação aplicada a uma solução, este foi fixado em 5.

Instância	Por grupo de estudantes							Por atividades							
	2015-1	2015-2	2016-1	2016-2	2017-1	2017-2	Média	2015-1	2015-2	2016-1	2016-2	2017-1	2017-2	Média	
ISM															
50	C	148,54	101,19	227,08	167,72	132,00	56,00	138,76	180,59	98,90	237,67	65,33	144,44	76,05	133,83
	V	0,0	0,0	1,7	0,7	0,0	0,0	0,40	0,3	0,0	1,7	0,0	1,0	0,0	0,50
	T	4,01	5,06	3,79	3,55	4,44	4,08	<b>4,16</b>	4,48	7,47	5,49	4,24	4,89	4,18	<b>5,13</b>
100	C	110,26	74,76	188,77	54,56	93,00	41,05	<b>93,73</b>	128,41	107,95	193,56	133,10	126,89	76,95	127,81
	V	0,7	0,0	1,0	0,0	0,3	0,0	0,33	0,0	0,3	1,0	0,3	0,3	0,0	<b>0,32</b>
	T	7,18	9,50	12,38	6,70	12,53	7,48	9,30	5,47	9,20	12,94	7,89	9,32	6,55	8,56
150	C	179,15	55,71	174,28	91,67	92,67	34,38	104,64	101,64	47,43	221,44	78,74	150,00	31,19	<b>105,07</b>
	V	0,0	0,0	1,0	0,0	0,0	0,0	<b>0,17</b>	0,3	0,0	1,0	0,3	0,7	0,0	0,38
	T	8,68	10,99	11,42	7,94	10,19	9,63	9,81	11,89	11,74	13,06	10,45	14,47	10,70	12,05

Tabela 4.16: Resultados médios do VNS Versão A (perturbação simples em vizinhanças consecutivas) partindo da melhor solução entre 100 aleatórias e realizando VND com *best improving*. O melhor valor de cada métrica por coluna de média geral foi grifado.

Instância	Por grupo de estudantes							Por atividades							
	2015-1	2015-2	2016-1	2016-2	2017-1	2017-2	Média	2015-1	2015-2	2016-1	2016-2	2017-1	2017-2	Média	
ISM															
50	C	86,72	50,86	217,85	110,10	119,11	24,52	101,53	131,72	68,71	148,49	80,03	97,89	57,10	97,32
	V	0,0	0,0	1,0	0,0	0,3	0,0	0,22	0,3	0,0	1,0	0,0	0,0	0,3	0,27
	T	4,11	5,56	6,38	3,05	5,39	4,15	<b>4,77</b>	3,45	4,47	5,51	5,27	4,33	3,13	<b>4,36</b>
100	C	95,54	83,33	204,49	76,74	122,00	17,86	<b>99,99</b>	83,41	71,00	152,67	52,49	135,44	40,43	89,24
	V	0,0	0,0	1,3	0,0	0,3	0,0	0,27	0,0	0,0	1,3	0,0	0,7	0,0	0,33
	T	5,97	8,56	9,82	5,60	9,18	5,60	7,46	8,32	8,16	10,92	5,96	7,20	7,16	7,95
150	C	92,28	75,38	187,00	77,44	137,22	44,52	102,31	69,56	48,71	176,64	66,69	103,56	49,67	<b>85,81</b>
	V	0,0	0,0	0,7	0,0	0,0	0,0	<b>0,12</b>	0,0	0,0	1,0	0,0	0,3	0,0	<b>0,22</b>
	T	8,85	12,28	9,97	7,32	11,62	9,00	9,84	9,06	13,30	13,36	10,79	12,96	7,93	11,23

Tabela 4.17: Resultados médios do VNS Versão A (perturbação simples em vizinhanças consecutivas) partindo da melhor solução entre 100 aleatórias e realizando VND com *first improving*. O melhor valor de cada métrica por coluna de média geral foi grifado.

#### 4.4. EXPERIMENTOS E RESULTADOS

61

Instância	Por grupo de estudantes							Por atividades						
	2015-1	2015-2	2016-1	2016-2	2017-1	2017-2	Média	2015-1	2015-2	2016-1	2016-2	2017-1	2017-2	Média
<b>ISM</b>														
C	79,72	74,86	141,95	53,03	104,78	31,19	80,92	99,49	61,10	146,08	79,69	57,56	56,33	83,38
50 V	0,0	0,0	1,0	0,0	0,3	0,0	0,22	0,0	0,0	1,0	0,7	0,0	0,0	0,28
T	7,89	10,81	14,95	7,19	10,01	9,65	<b>10,08</b>	7,28	13,76	14,41	6,97	11,45	6,03	<b>9,98</b>
C	45,00	40,10	130,00	35,38	67,56	37,86	<b>59,32</b>	75,49	58,52	239,10	54,56	91,67	38,24	92,93
100 V	0,0	0,0	0,3	0,0	0,0	0,0	<b>0,05</b>	0,3	0,0	1,0	0,0	0,0	0,0	<b>0,22</b>
T	16,97	21,52	20,23	16,74	22,77	12,09	18,39	12,66	15,52	12,87	16,30	16,27	10,17	13,97
C	85,49	39,52	130,92	30,28	66,33	17,86	61,73	70,69	55,05	127,10	48,92	71,11	33,43	<b>67,72</b>
150 V	0,0	0,0	0,3	0,0	0,0	0,0	<b>0,05</b>	0,3	0,0	1,0	0,0	0,0	0,0	<b>0,22</b>
T	18,81	22,14	19,24	23,69	24,55	17,06	20,92	15,50	21,71	27,33	21,63	32,64	20,03	23,14

Tabela 4.18: Resultados médios do VNS Versão B (perturbação incremental em vizinhança aleatória) partindo da melhor solução entre 100 aleatórias e realizando VND com *best improving*. O melhor valor de cada métrica por coluna de média geral foi grifado.

Instância	Por grupo de estudantes							Por atividades						
	2015-1	2015-2	2016-1	2016-2	2017-1	2017-2	Média	2015-1	2015-2	2016-1	2016-2	2017-1	2017-2	Média
<b>ISM</b>														
C	86,15	58,81	182,67	52,33	82,11	28,67	81,79	66,13	52,48	161,64	54,05	59,67	35,67	71,61
50 V	0,0	0,0	0,3	0,0	0,0	0,0	<b>0,05</b>	0,0	0,0	1,0	0,0	0,0	0,0	0,17
T	10,80	10,39	16,62	8,52	12,21	7,83	<b>11,06</b>	7,28	11,50	8,28	5,71	9,61	6,35	<b>8,12</b>
C	62,21	54,43	170,18	21,56	67,33	24,52	66,71	43,44	45,29	126,74	48,92	53,67	20,43	<b>56,42</b>
100 V	0,0	0,0	1,0	0,0	0,0	0,0	0,17	0,0	0,0	1,0	0,0	0,0	0,0	0,17
T	11,59	12,95	16,64	18,22	15,82	9,85	14,18	18,16	13,03	22,73	13,56	20,04	10,98	16,42
C	34,33	48,10	136,85	46,36	62,33	17,86	<b>57,64</b>	66,87	47,10	148,62	54,38	50,56	24,52	65,34
150 V	0,0	0,0	1,0	0,0	0,0	0,0	0,17	0,0	0,0	0,7	0,0	0,0	0,0	<b>0,12</b>
T	14,59	20,34	22,89	26,30	24,88	14,07	20,51	15,88	29,88	17,18	17,73	29,38	16,42	21,08

Tabela 4.19: Resultados médios do VNS Versão B (perturbação incremental em vizinhança aleatória) partindo da melhor solução entre 100 aleatórias e realizando VND com *first improving*. O melhor valor de cada métrica por coluna de média geral foi grifado.

Se avaliados individualmente (nas tabelas 4.16/4.17 e 4.18/4.19), quando usando *best improving* a versão A teve resultados melhores com a construção por grupos enquanto que quando usando *first improving* os valores obtidos com o método de construção por atividades foram ligeiramente melhores (os únicos resultados até então que favorecem este método em detrimento ao baseado em grupos). Já a versão B teve um comportamento similar nesse sentido, mas a diferença entre as formas de construção foram menos pronunciadas. No geral, a busca local *first improving*

parece ser mais promissora em ambas as versões de VNS.

Já a qualidade geral das soluções com a versão B do VNS foi superior à versão A, para todas as estratégias, com a busca local *best improving* e *first improving* e com ambas os métodos de construção. Mas quanto ao consumo de tempo, a versão A é executada mais rapidamente que a versão B, o que pode estar associado ao seu desempenho pior e ao critério de parada usado. A versão B parece ser capaz de escapar de ótimos locais mais frequentemente e, com isso, o processo geral de busca é naturalmente mais longo para um mesmo número de iterações sem melhora.

Levando em consideração a inferioridade da versão A em termos de qualidade de solução mas sua superioridade em termos de tempo de execução, esta foi executada novamente com o critério de parada de 300 iterações sem melhora. No entanto, mesmo dobrando o valor do critério de parada a qualidade das soluções permaneceu inferior mesmo a execuções da versão B com 50 iterações.

Instância	Por grupo de estudantes						Por atividades							
	2015-1	2015-2	2016-1	2016-2	2017-1	2017-2	Média	2015-1	2015-2	2016-1	2016-2	2017-1	2017-2	Média
ISM														
C	90,08	53,48	221,1	71,67	71,78	40,1	91,37	79,59	42,71	174,44	85	121,44	41,05	90,71
300	V	0	0	1	0	0	0,17	0,3	0	1	0	0	0	0,17
T	12,44	13,61	18,26	13,56	15,51	10,07	13,91	9,27	12,71	15,68	12,29	13,84	13,4	12,87

Tabela 4.20: Resultados médios do VNS Versão A partindo da melhor solução entre 100 aleatórias, realizando VND com *best improving*.

Instância	Por grupo de estudantes						Por atividades							
	2015-1	2015-2	2016-1	2016-2	2017-1	2017-2	Média	2015-1	2015-2	2016-1	2016-2	2017-1	2017-2	Média
ISM														
C	69,08	40,43	134,59	48,85	54,00	42,33	64,88	83,46	65,86	228,31	40,03	63,89	48,38	88,32
300	V	0,00	0,00	1,00	0,30	0,00	0	0,22	0,00	0,00	0,30	0,00	0,00	0,05
T	8,82	12,67	12,41	10,62	17,58	14,42	12,75	11,43	12,37	18,59	11,79	15,45	10,91	13,42

Tabela 4.21: Resultados médios do VNS Versão A partindo da melhor solução entre 100 aleatórias, realizando VND com *first improving*.

#### 4.4.4.4 Simulated annealing

O desempenho da implementação de *Simulated annealing* sobre as instâncias do DCC foi avaliado isoladamente partindo de uma solução aleatória, da mesma

forma que nos experimentos com a técnica VNS+VND. O número de iterações por temperatura foi fixado em 100, a temperatura inicial em 1000 e apenas o valor alfa que define a função de taxa de resfriamento foi variado. Tal abordagem que fixa parâmetros e varia apenas um deles é possível para a técnica, não comprometendo suas bases teóricas.

Instância	Por grupo de estudantes							Por atividades							
	2015-1	2015-2	2016-1	2016-2	2017-1	2017-2	Média	2015-1	2015-2	2016-1	2016-2	2017-1	2017-2	Média	
Alfa: 0,99	C	197,64	147,86	406,62	172,77	170,22	79,76	195,81	202,97	171,29	294,54	101,46	200,78	84,52	175,93
	V	1,3	0	4,3	0,3	0,3	0	1,03	1	0,7	2,7	0,3	0,7	0	0,90
	T	0,80	0,81	0,71	0,76	0,88	0,71	<b>0,78</b>	0,78	0,98	0,81	0,97	0,88	0,97	<b>0,90</b>
Alfa: 0,999	C	115,56	53,14	164,21	49,69	106,11	34,38	87,18	116,77	89,05	173,64	49,08	116,67	90,05	105,88
	V	0	0	0,7	0	0,3	0	0,17	0	0	0,7	0	0,3	0	0,17
	T	6,01	6,58	6,18	6,01	6,56	6,30	6,27	5,99	6,77	6,27	6,21	6,58	6,56	6,40
Alfa: 0,9999	C	58,97	58,76	79,69	41,56	50,11	31,19	<b>53,38</b>	84,77	67,76	101,49	43,92	60,89	25,81	<b>64,11</b>
	V	0	0	0	0	0	0	<b>0,00</b>	0	0	0,3	0	0	0	<b>0,05</b>
	T	41,16	45,85	42,71	40,74	39,65	38,95	41,51	39,75	46,16	42,49	40,31	45,26	44,03	43,00

Tabela 4.22: Resultados médios *Simulated annealing* a partir da melhor solução entre 100 aleatórias. O melhor valor de cada métrica por coluna de média geral foi grifado.

Naturalmente, devido à natureza do algoritmo, funções de resfriamento mais lentas levam a tempos de execução mais altos. Quanto à qualidade das soluções, o desempenho quando cada método de construção é usado parece ser similar mas o procedimento por grupos de estudantes permanece apresentando vantagem, tanto em custo como em viabilidade.

O método parece ser, em média, capaz de produzir soluções viáveis para instâncias do DCC com facilidade, mesmo usando um valor para alfa que levou a um tempo médio de execução inferior a 6 minutos. No entanto, para essa configuração, instâncias mais problemáticas (2016-1 e 2017-1) tiveram a violação média de viabilidade maior que zero. Já com o valor de alfa mais próximo a 1 utilizado, a execução tem um tempo médio de aproximadamente 40 minutos, mas foi capaz de produzir soluções viáveis para todas as instâncias, inclusive a 2016-1.

#### 4.4.4.5 Comparação e análise

Quando comparados os métodos de construção o que se mostrou capaz de levar a soluções melhores foi na maioria das vezes o que parte de grupos de alunos. A única exceção em que a qualidade das soluções produzidas com uso do procedimento de construção por atividades se aproximou e por vezes superou ligeiramente a alternativa foi em parte das estratégias com VNS. De forma similar o uso da busca local *best improving* também foi frequentemente mais promissora (sendo uma possível exceção também com a técnica VNS, em certas circunstâncias específicas).

A combinação de GRASP com VND mostrou ser capaz de produzir soluções de qualidade para quase todas as instâncias, superando os custos das soluções manuais e o fazendo em tempos computacionais que variam entre 10 e 20 minutos. No entanto, a estratégia não parece apropriada para lidar com casos com restrições mais severas, como é o caso da instância 2016-1. De fato nenhuma das execuções levou a valores médios de violação de viabilidade iguais a zero para esta instância.

Já quando avalia-se a combinação VNS e VND houve uma diferença clara entre as versões do VNS implementadas. A versão A parece ficar presa em ótimos locais com mais facilidade que a versão B, mas possui um tempo de execução inferior. Assim, quando são usado valores iguais para o critério de parada, a qualidade das soluções produzidas com uso da versão B supera significativamente às da versão A. Por outro lado, o tempo de execução da versão é maior que o dobro do da versão A, o que leva em termos práticos a ser interessante usar valores muito maiores para o critério de parada na versão A. De fato, quando o critério de parada é relaxado a qualidade da solução produzida na versão A começa a se aproximar de quando se usa a versão B, enquanto o tempo de execução permanece inferior. De forma geral, ambas as versões tem bom desempenho e, assim como com o GRASP+VND, superam as soluções manuais com tempos inferiores a 20 minutos, com facilidade. Também como na estratégia anterior, esta tem dificuldades em produzir soluções viáveis para a instância 2016-1.

O método *Simulated annealing*, especialmente, foi capaz de obter soluções viáveis

e de custo médio menor que todos os outros métodos testados (quando usando a construção por grupos), para todas as instâncias. Além disso, os valores produzidos nesses resultados mostram que a instância 2016-1 é especialmente difícil do ponto de vista da busca de soluções viáveis, não apresentando valor médio igual a zero em nenhuma das estratégias executadas, com única exceção para a execução do método *Simulated annealing* com a configuração mais lenta usada. Por outro lado, apesar de produzir soluções de melhor qualidade, a configuração de parâmetros usada no *Simulated annealing* levou ao maior tempo médio de execução entre todas as estratégias (aproximadamente 40 minutos) e como, o experimento foi realizado em apenas 3 rodadas, não existem garantias fortes de que a viabilidade para a instância seja atingida com facilidade. Outro ponto é que esse método, mesmo usando uma configuração de parâmetros que em média leva a uma execução de menos de 7 minutos, levou a soluções de custo muito próximo aos obtidos pelas técnicas anteriores.

Diante destes resultados, é evidente que as combinações de métodos metaheurísticos propostos se mostram capazes de produzir soluções viáveis e de qualidade superior às manuais. No entanto, como mostra o caso da instância 2016-1, é importante observar que existe uma questão clara na relação entre tempo de execução a geração de soluções viáveis. Para instâncias menos difíceis (e mais comuns) do departamento, os métodos aqui apresentados podem ser aplicados para produzir soluções boas em pouco tempo. Por outro lado, mesmo quando uma instância se mostra mais desafiadora nesse sentido os resultados mostram que a técnica *Simulated annealing* pode ser aplicada com parâmetros ajustados de forma a sacrificar tempo e levar a soluções factíveis de qualidade alta.

#### 4.4.5 Experimentos refinados

A partir dos resultados dos experimentos descritos na seção anterior, nesta são propostas algumas outras estratégias híbridas combinando aquelas que obtiveram melhor desempenho segundo critérios de qualidade como tempo de execução aceitável, custo das soluções e facilidade de obtenção de viabilidade.

#### 4.4.5.1 GRASP+VND seguida de VNS+VND

Os primeiros experimentos foram uma combinação entre GRASP+VND e VNS. A melhor combinação de métodos segundo os valores da tabela 4.12, com a construção por grupos de estudantes e busca local *best improving*, foi selecionada e a melhor solução encontrada foi usada como solução inicial do método VNS versão B com VND que, por sua vez, teve um desempenho melhor nos testes anteriores. A tabela 4.23 exibe os resultados e estes mostram que a estratégia supera todas as anteriores em custo médio. A estratégia também foi capaz de produzir viabilidade para todas as instâncias em uma das configurações.

Instância	Best improving							First improving									
	2015-1		2015-2		2016-1		2016-2		2017-1		2017-2		Média	Média			
	C	V	C	V	C	V	C	V	C	V	C	V					
GRASP+VND	Alfa: 0,9	C	49,44	40,14	115,33	21,56	48,56	24,52	49,93		37,64	38,86	82,69	20,21	39,11	17,86	<b>39,40</b>
		V	0,0	0,0	0,7	0,0	0,0	0,0	0,12		0,0	0,0	0,0	0,0	0,0	0,0	<b>0,00</b>
		T	3,50	4,21	6,54	4,21	5,42	4,39	4,71		5,54	4,35	9,31	3,42	6,58	3,63	5,47
Alfa: 1,0	Alfa: 1,0	C	65,15	38,19	131,62	23,28	58,22	18,81	55,88		65,49	38,19	115,69	24,31	57,89	17,86	53,24
		V	0,0	0,0	0,7	0,0	0,0	0,0	0,12		0,0	0,0	0,7	0,0	0,0	0,0	0,12
		T	4,29	6,26	5,12	2,48	3,30	2,72	4,03		4,34	3,63	4,85	2,73	3,81	2,78	<b>3,69</b>
Reativo com 10	Reativo com 10	C	41,49	40,43	118,85	28,56	69,67	17,86	52,81		45,77	39,81	149,54	21,23	71,22	17,86	57,57
		V	0,0	0,0	0,3	0,0	0,0	0,0	0,05		0,0	0,0	0,7	0,0	0,0	0,0	0,12
		T	3,95	4,99	7,65	4,82	3,59	4,71	4,95		3,84	6,31	3,98	3,17	4,07	3,66	4,17
Reativo com 20	Reativo com 20	C	61,05	42,05	137,00	21,56	61,78	19,14	57,10		76,79	40,76	125,56	22,59	44,22	19,14	54,84
		V	0,0	0,0	1,0	0,0	0,0	0,0	0,17		0,0	0,0	1,0	0,0	0,0	0,0	0,17
		T	4,16	5,33	4,36	4,16	6,18	3,35	4,59		3,37	4,04	6,48	3,38	5,96	2,61	4,31
Reativo com 40	Reativo com 40	C	66,69	39,81	121,03	32,85	80,11	32,14	62,11		58,00	40,76	107,54	25,85	81,56	31,19	57,48
		V	0,0	0,0	1,3	0,0	0,0	0,0	0,22		0,0	0,0	0,7	0,0	0,0	0,0	0,12
		T	4,44	4,87	5,06	3,39	4,89	3,07	4,29		5,01	4,29	4,59	2,80	3,75	2,75	3,87

Tabela 4.23: Resultados médios de VNS versão B com 50 iterações sem melhora aplicado à soluções produzidas por GRASP com construção por grupos de estudantes e VND na busca local. O menor valor de cada métrica por ambas as colunas de médias foram grifados.

#### 4.4.5.2 Construção GRASP seguida de VNS+VND

Como, segundo a tabela 4.23, o desempenho do método VNS+VND a partir de uma solução inicial de qualidade alta pareceu ser significativamente melhor do que quando este partiu de uma solução aleatória (produzindo, inclusive, apenas soluções viáveis para 2016-1, em uma das execuções), a seguir ele foi executado a partir da melhor solução produzida pelo método GRASP sem busca local, ou seja, apenas

executando a fase de construção e coletando a melhor solução. As tabelas 4.24 e 4.25 mostram os resultados de execuções usando *best improving* e *first improving* na busca local, respectivamente. Os resultados parecem sugerir que a técnica VNS partindo de soluções de alta qualidade leva a menores tempos de execução mantendo valores interessantes para o custo e a viabilidade médios, mesmo quando a fase de busca local não é executada na produção da solução inicial. Além disso, em múltiplas execuções a viabilidade da instância 2016-1 foi atingida nas três rodadas, o que sugere uma vantagem da estratégia sobre instâncias mais difíceis.

Instância		Por grupo de estudantes							Por atividades							
		2015-1	2015-2	2016-1	2016-2	2017-1	2017-2	Média	2015-1	2015-2	2016-1	2016-2	2017-1	2017-2	Média	
ISM (VNS)	Construção	C	74,54	42,21	132,38	59,42	76,00	17,86	67,07	127,54	48,00	134,62	31,23	100,17	49,79	81,89
		Alfa: 0,9	V	0,0	0,0	0,3	0,0	0,0	0,0	0,0	0,0	0,7	0,0	0,0	0,0	0,12
		T	3,54	6,01	8,18	5,55	5,13	5,08	5,58	3,69	6,45	7,99	7,97	5,25	3,67	5,84
	50	C	47,88	88,86	137,19	99,42	59,67	17,86	75,15	43,27	46,07	188,88	66,35	75,50	19,79	73,31
		Alfa: 1,0	V	0,0	0,0	1,0	0,0	0,0	0,0	0,0	0,0	1,5	0,0	0,0	0,0	0,25
		T	4,46	4,53	6,07	3,94	5,42	4,52	4,82	3,21	6,33	7,95	4,50	6,45	5,28	5,62
	Reativo	C	115,04	40,29	209,15	26,35	66,67	50,36	84,64	99,92	42,71	143,73	56,85	55,67	51,71	75,10
		Reativo	V	0,0	0,0	1,3	0,0	0,0	0,0	0,0	0,0	0,5	0,0	0,0	0,0	0,08
		T	6,38	6,50	8,34	3,78	5,17	5,96	6,02	6,23	6,96	5,06	4,82	4,75	5,21	5,51
100	Construção	C	48,92	48,86	180,12	36,35	82,50	29,79	71,09	66,85	38,36	176,15	24,31	65,17	19,79	65,11
		Alfa: 0,9	V	0,0	0,0	1,33	0,0	0,0	0,0	0,0	0,0	1,0	0,0	0,3	0,0	0,22
		T	10,02	10,53	11,70	6,29	10,92	8,48	9,66	5,77	14,70	12,75	6,85	10,07	9,87	10,00
	100	C	62,96	44,14	152,58	21,23	57,83	17,86	59,43	61,96	40,79	141,46	64,31	68,33	42,64	69,92
		Alfa: 1,0	V	0,0	0,0	0,7	0,0	0,0	0,0	0,0	0,0	2,0	0,0	0,3	0,0	0,38
		T	8,32	7,93	12,57	8,90	10,22	7,51	9,24	7,41	10,03	12,57	9,62	12,10	10,00	10,29
	Reativo	C	57,35	65,14	90,50	41,23	75,67	17,86	57,96	27,35	48,86	135,27	59,19	86,50	19,79	62,83
		Reativo	V	0,0	0,0	0,0	0,0	0,0	0,0	0,0	0,0	0,5	0,0	0,3	0,0	0,13
		T	10,87	12,59	18,08	6,84	8,25	6,93	10,59	10,65	16,42	16,38	7,79	10,96	9,69	11,98

Tabela 4.24: Resultados médios de VSN versão B com *best improving* com solução inicial gerada pela fase construtiva do método GRASP.

## 4.4. EXPERIMENTOS E RESULTADOS

68

ISM (VNS)	Instância	Por grupo de estudantes							Por atividades							
		2015-1	2015-2	2016-1	2016-2	2017-1	2017-2	Média	2015-1	2015-2	2016-1	2016-2	2017-1	2017-2	Média	
50	Alfa: 0,9	C	104,73	47,07	117,12	21,23	52,83	34,57	62,93	42,77	39,79	102,08	64,31	121,17	37,86	68,00
		V	0,0	0,0	0,7	0,0	0,0	0,0	0,12	0,0	0,0	0,0	0,0	0,7	0,0	0,12
		T	5,14	6,85	5,40	4,98	6,36	4,52	5,54	3,77	10,20	7,11	5,07	7,80	3,79	6,29
	Alfa: 1,0	C	42,46	37,86	109,69	66,35	44,33	21,71	53,73	75,85	38,36	161,15	62,50	80,17	34,57	75,43
		V	0,0	0,0	0,3	0,0	0,0	0,0	0,05	0,0	0,0	0,3	0,0	0,3	0,0	0,10
		T	4,67	5,88	6,57	3,31	6,53	3,66	5,10	4,28	3,77	6,43	3,78	6,25	5,29	4,97
	Reativo	C	64,54	40,29	87,58	50,92	79,67	21,71	57,45	49,92	40,79	151,31	59,69	95,00	21,71	69,74
		V	0,0	0,0	0,3	0,0	0,0	0,0	0,05	0,0	0,0	0,7	0,0	0,0	0,0	0,12
		T	7,21	5,16	8,83	3,75	4,29	3,75	5,50	5,31	5,25	8,00	4,77	3,78	4,60	5,29
100	Alfa: 0,9	C	55,85	38,86	118,15	37,88	67,83	21,71	56,71	71,50	38,36	191,15	72,50	55,17	30,71	76,57
		V	0,0	0,0	0,7	0,0	0,0	0,0	0,12	0,0	0,0	0,3	0,0	0,0	0,0	0,05
		T	8,48	9,78	9,27	6,57	6,41	5,23	7,62	8,70	7,44	12,70	8,06	8,00	6,05	8,49
	Alfa: 1,0	C	33,27	40,79	137,65	19,69	61,00	19,79	52,03	83,31	38,86	139,65	55,85	57,50	17,86	65,51
		V	0,0	0,0	0,0	0,0	0,0	0,0	0,00	0,0	0,0	0,3	0,0	0,0	0,0	0,05
		T	7,47	8,80	7,37	6,22	7,23	6,87	7,33	5,69	10,07	9,29	8,12	11,74	6,52	8,57
	Reativo	C	111,69	59,29	153,35	77,88	91,17	17,86	85,21	50,92	37,86	150,85	42,77	81,00	30,71	65,69
		V	0,0	0,0	0,7	0,0	0,0	0,0	0,12	0,0	0,0	0,3	0,0	0,0	0,0	0,05
		T	7,15	10,14	12,30	8,03	7,15	6,40	8,53	6,88	9,36	11,26	8,44	10,31	8,81	9,18

Tabela 4.25: Resultados médios de VSN versão B com *first improving* com solução inicial gerada pela fase construtiva do método GRASP.

### 4.4.5.3 Simulated annealing seguido de VNS+VND

Levando em consideração desempenho geral da técnica VNS+VND nos experimentos anteriores quando sua solução inicial é de qualidade alta, este foi investigado também usando as soluções obtidas pelo *Simulated annealing* como ponto de partida. A tabela 4.26 mostra os resultados e permite ver que quando executada após o *Simulated annealing* a técnica também teve bons resultados, tanto quando o *Simulated annealing* é executado com alfa igual a 0.999 quanto quando ele é igual a 0.999. No entanto, a combinação da construção GRASP com VNS e VND foi capaz de atingir melhor qualidade geral superior em menor tempo.

Alfa (SA)		0,999							0,9999							
		Instância		2015-1	2015-2	2016-1	2016-2	2017-1	2017-2	Média	2015-1		2015-2	2016-1	2016-2	2017-1
ISM (VNS)	Construção	C	47,03	62,38	139,00	52,18	63,67	32,48	66,12	33,77	40,79	117,62	49,92	67,50	17,86	54,58
		Best improving V	0,0	0,0	0,7	0,0	0,3	0,0	0,17	0,0	0,0	0,7	0,0	0,0	0,0	0,12
		T	3,23	3,54	5,06	4,52	4,83	2,60	3,96	3,32	5,10	3,58	2,44	2,89	3,50	3,47
	Construção	C	45,67	40,76	161,23	58,18	67,33	17,86	65,17	35,81	40,29	106,62	49,92	77,00	17,86	54,58
		First improving V	0,0	0,0	1,3	0,0	0,3	0,0	0,27	0,0	0,0	0,7	0,0	0,0	0,0	0,12
		T	2,82	6,18	4,67	4,24	3,69	3,70	4,22	2,89	4,14	6,10	2,18	3,01	2,88	3,53

Tabela 4.26: Resultados médios de VSN versão B com solução inicial gerada pelo método *Simulated annealing*. O melhor valor de cada métrica por coluna de média geral foi grifado. A métrica tempo não inclui o tempo de execução do *Simulated annealing*.

#### 4.4.5.4 GRASP+VNS+VND

Considerando os objetivos práticos deste trabalho em produzir estratégias úteis que possam ser usadas pelo departamento o tempo de execução máximo foi um fator importante a se considerar na seleção das estratégias e, de fato, os experimentos realizados até aqui levaram a soluções de qualidade alta em tempo aceitáveis. No entanto, na instância 2016-1 (e em menor proporção na instância 2017-2) soluções viáveis não foram facilmente encontradas em nenhuma das execuções. Esse fato é evidenciado ao observar-se que o fator de violação médio das restrições essenciais raramente foi igual a zero para esta instância nas execuções com três rodadas. Entre as estratégias que foram capazes de atingir esse objetivo para esta instância estão a configuração mais lenta de *Simulated annealing* e múltiplas execuções do método VNS+VND aplicado a soluções iniciais boas (tabelas 4.23, 4.24 e 4.25). Isso sugeriu que a combinação entre a fase construtiva do GRASP como solução inicial para o método tem maiores chances de lidar com estas instâncias mais problemáticas, do ponto de vista da viabilidade, além de necessitarem de tempo de execução menor que a combinação com *Simulated annealing* para fazê-lo.

Assim, foi proposto o uso da técnica VNS+VND na fase de busca local do método GRASP, a fim de aumentar as chances de soluções viáveis serem produzidas ao se valer da propriedade de múltiplos inícios iterativos. Pela limitação de tempo tal combinação torna proibitiva execução de um grande número de iterações do GRASP

e, por isso, o número de iterações selecionado foi igual a 15, enquanto o número de iterações sem melhora como critério de parada do método VNS foi fixado em 25. Pelos mesmos motivos, a versão A do método VNS (realizando *first improving*) foi escolhida devido ao seu menor tempo de finalização.

No entanto, para a instância 2016-1, mesmo esta estratégia não foi capaz de atingir média igual a zero para a medida de violação de viabilidade em pelo menos uma das execuções em três rodadas. Com isso, com o objetivo de produzir resultados com menos variância, uma nova execução em 10 rodadas foi realizada apenas com esta instância, aumentando também o número de iterações do GRASP para 20 (tabela 4.27).

Os resultados da tabela 4.27 parecem sugerir que a execução do método GRASP com VNS+VND na busca local tem o maior potencial, entre as estratégias estudadas, de gerar soluções viáveis para instâncias do DCC mais problemáticas. Além disso, configurações um pouco mais aleatórias do parâmetro de construção parecem facilitar a obtenção de viabilidade e também, para esta instância, levar aos menores custos produzidos por esta estratégia.

		Médias de 3 execuções						Médias de 10 execuções	
		GRASP com 15 iterações e VNS-A com 25						GRASP com 20 iterações e VNS-A com 25	
Instância		2015-1	2015-2	2016-1	2016-2	2017-1	2017-2	Média	2016-1
Construção									
Alfa: 0,0	C	27,33	41,43	168,92	33,69	51,78	17,86	56,84	Viáveis
	V	0,0	0	0,3	0	0	0,00	0,05	
	T	22,65	27,33	28,35	20,55	26,79	20,72	24,40	
	I	8,3	10	4	9,3	5	4,7	6,88	
Alfa: 0,1	C	51,33	41,14	108,23	22,26	66,11	17,86	51,16	10/10
	V	0,0	0	0,3	0	0	0	0,05	
	T	20,39	29,72	31,27	20,13	25,53	20,46	24,58	
	I	12,7	8,3	8,7	5,3	3,3	9,3	7,93	
Alfa: 0,2	C	66,13	40,48	133,9	30,28	50,33	20,1	56,87	7/10
	V	0,0	0	0,3	0,0	0	0	0,05	
	T	22,43	29,54	30,16	21,06	25,45	21,64	25,05	
	I	9,0	10,7	7,7	8,7	9	3,7	8,13	
Alfa: 0,3	C	38,10	41,1	143,82	22,95	48,56	17,86	52,07	9/10
	V	0,0	0	0,3	0,0	0	0,0	0,05	
	T	23,27	30,12	28,32	19,10	27,2	20,3	24,72	
	I	10,7	11,3	8,3	4,3	6,3	9,3	8,37	
Alfa: 0,4	C	58,77	39,48	115,56	22,26	32,5	24,52	48,85	10/10
	V	0,0	0	0	0,0	0,0	0,0	0,00	
	T	22,84	26,08	29,17	21,15	25,9	19,71	24,14	
	I	5,0	5,7	11	10,3	6	10,7	8,12	
Alfa: 0,5	C	57,41	45,48	162	30,28	63,89	20,43	63,25	10/10
	V	0,0	0	0	0,0	0,0	0,0	0,00	
	T	22,23	26,93	30,18	21,15	25,79	21,20	24,58	
	I	5,3	9,7	9	10,0	5,7	3,0	7,12	
Alfa: 0,6	C	48,05	40,1	113,92	41,03	54,00	25,81	53,82	10/10
	V	0,0	0	0,3	0,0	0,0	0,0	0,05	
	T	20,97	27,53	27,87	20,74	27,44	20,71	24,21	
	I	9,7	8	9	7,0	7,3	8,3	8,22	
Alfa: 0,7	C	36,05	43,95	90,59	29,95	44,89	24,52	44,99	9/10
	V	0,0	0,0	0	0,0	0,0	0,0	0,00	
	T	23,12	27,36	30,61	20,95	27,05	21,09	19,93	
	I	10,3	6,0	7	7,3	4,7	3,0	6,38	
Alfa: 0,8	C	56,10	40,76	164,56	26,87	57,67	17,86	60,64	10/10
	V	0,00	0	0	0	0	0	0,00	
	T	22,45	26,68	26,48	19,27	22,11	17,82	22,47	
	I	5,00	5,70	9,3	6,7	11,00	6	7,28	
Alfa: 0,9	C	50,74	38,52	146,28	23,62	70	17,86	57,84	8/10
	V	0,00	0	0	0	0	0	0,00	
	T	18,42	25,81	27,24	18,32	22,86	19,24	21,98	
	I	11,30	6,30	6	5,7	6,30	5,7	6,88	
Alfa: 1,0	C	44,31	38,52	132,90	18,18	44,89	17,86	49,44	8/10
	V	0,00	0	0,3	0	0	0	0,05	
	T	19,56	25,99	28,36	20,28	23,98	17,32	22,58	
	I	8,30	6,30	7	7,3	5,30	7,7	6,98	

Tabela 4.27: Resultados médios das execuções do método GRASP usando construção por grupos e VNS A com *first improving* na fase de busca local.

#### 4.4.5.5 Comparação e análise

As tabelas 4.28 e 4.29 mostram um subconjunto das estratégias discutidas que inclui algumas daquelas que apresentaram melhor desempenho na relação tempo/-custo/viabilidade. A combinação de GRASP+VND para produzir as soluções iniciais da combinação VNS+VND foi a estratégia que obteve os melhores valores médios para o custo, além de ter também sido capaz de encontrar os menores valores de custo para algumas das instâncias e, ao menos no experimento realizado, produzido apenas soluções viáveis para todas as instâncias.

De forma geral as estratégias envolvendo VNS+VND como fase final foram as que tiveram a maior facilidade em atingir viabilidade, além de o fazerem em tempos de execução baixos (quando a primeira fase é uma técnica rápida, como a construção pura) quando comparados ao GRASP+VND com grande número de iterações e a configurações lentas de *Simulated annealing*.

Para instâncias com características que dificultam a obtenção de soluções viáveis, o que parece não ser comum nesse estudo de caso, a execução de VNS+VND nas iterações do método GRASP com configuração gulosa parece ser muito eficaz em encontrar soluções viáveis e de fazê-lo em menor tempo do que *Simulated annealing* usando parâmetros muito lentos. Essa estratégia também foi capaz de produzir alguns dos menores custos médios entre todas as estratégias selecionadas e o fez em tempo de execução competitivo, especialmente ao se observar que número médio da iteração de ultima melhora sugere que um número menor de iterações (e por tanto tempo) é seria realmente necessário na prática.

Uma possível explicação para o desempenho da técnica VNS+VND na produção de soluções viáveis para instâncias difíceis neste sentido está na natureza nada função objetivo e das restrições. Avaliando as peculiaridades da instância 2016-1 vemos que esta é a mais restrita na restrição de número de dias máximo em que cada professor pode ser alocado. Combinando esse fato ao pequeno número de professores na instância o espaço de soluções parece estar se dispondo de tal forma em que soluções viáveis são esparsas. Assim, a aplicação da técnica VNS ganha signifi-

cativa vantagem, devido aos seus mecanismos centrados no escape de ótimos locais. Essa vantagem se potencializa, ainda, quando a técnica é usada na busca local do método GRASP já que diversos pontos diferentes do espaço de busca são tomados como ponto de partida, aumentando a probabilidade da técnica navegar até soluções viáveis.

		2015-1	2015-2	2016-1	2016-2	2017-1	2017-2	Média
Estratégia								
GRASP+VND	C	98,79	54,9	181,92	46,13	98,22	28,38	84,72
Por grupos	V	0	0	1	0	0	0	0,17
First improving	I	461	577	544,3	618,3	543,3	820,3	594,0
Reativo	T	27,89	36,01	33,03	26,57	36,35	32,21	32,01
GRASP+VND	C	74,72	45,00	155,49	35,03	68,00	28,38	67,77
Por grupos	V	0	0	1,7	0	0	0	0,28
Best improving	I	460,7	441,7	565,7	274,7	260,7	437,7	406,9
Alfa:1,0	T	15,22	23,24	22,55	20,42	21,01	21,64	20,68
VNS-A+VND; 300 iterações	C	69,08	40,43	134,59	48,85	54,00	42,33	64,88
Por grupos	V	0,00	0,00	1,00	0,30	0,00	0	0,22
First improving	T	8,82	12,67	12,41	10,62	17,58	14,42	12,75
Construção GRASP; Alfa: 1,0; Por grupos	C	62,96	44,14	152,58	21,23	57,83	17,86	59,43
VNS-B+VND; 100 iterações	V	0,0	0,0	0,7	0,0	0,0	0,0	0,12
Best improving	T	8,32	7,93	12,57	8,90	10,22	7,51	9,24
Construção GRASP; Reativo; Por grupos	C	64,54	40,29	87,58	50,92	79,67	21,71	57,45
VNS-B+VND; 50 iterações	V	0,0	0,0	0,3	0,0	0,0	0,0	0,05
First improving	T	7,21	5,16	8,83	3,75	4,29	3,75	5,50
VNS-B+VND; 100 iterações	C	43,44	45,29	126,74	48,92	53,67	20,43	56,42
Por atividades	V	0,0	0,0	1,0	0,0	0,0	0,0	0,17
First improving	T	18,16	13,03	22,73	13,56	20,04	10,98	16,42
SA; Alfa: 0,9999	C	33,77	40,79	117,62	49,92	67,50	17,86	54,58
VNS-B; 50 iterações	V	0,0	0,0	0,7	0,0	0,0	0,0	0,12
Best improving	T	3,32	5,10	3,58	2,44	2,89	3,50	3,47
SA; Alfa: 0,9999	C	35,81	40,29	106,62	49,92	77,00	17,86	54,58
VNS-B; 50 iterações	V	0,0	0,0	0,7	0,0	0,0	0,0	0,12
First improving	T	2,89	4,14	6,10	2,18	3,01	2,88	3,53
GRASP+VND; Por grupos; Reativo	C	41,49	40,43	118,85	28,56	69,67	17,86	52,81
VNS-B+VND; 50 iterações	V	0,0	0,0	0,3	0,0	0,0	0,0	0,05
Best improving	T	3,95	4,99	7,65	4,82	3,59	4,71	4,95

Tabela 4.28: Resultados médios das melhores estratégias, recapitulados. Estratégias que atingiram viabilidade média não nula.

Estratégia		2015-1	2015-2	2016-1	2016-2	2017-1	2017-2	Média
Construção GRASP; Reativo; Por grupos	C	57,35	65,14	90,50	41,23	75,67	17,86	57,96
VNS-B+VND; 100 iterações	V	0,0	0,0	0,0	0,0	0,0	0,0	0,00
Best improving	T	10,87	12,59	18,08	6,84	8,25	6,93	10,59
Construção GRASP; Alfa: 1,0; Por grupos	C	33,27	40,79	137,65	19,69	61,00	19,79	52,03
VNS-B+VND; 100 iterações	V	0,0	0,0	0,0	0,0	0,0	0,0	0,00
First improving	T	7,47	8,80	7,37	6,22	7,23	6,87	7,33
SA - Início aleatório	C	58,97	58,76	79,69	41,56	50,11	31,19	53,38
Alfa: 0,9999	V	0	0	0	0	0	0	0,00
Por grupos	T	41,16	45,85	42,71	40,74	39,65	38,95	41,51
GRASP+VNS-A+VND	C	58,77	39,48	115,56	22,26	32,5	24,52	48,85
Alfa: 0,4; 15 iterações	V	0,0	0	0	0,0	0,0	0,0	0,00
VNS com 25 interações	T	22,84	26,08	29,17	21,15	25,9	19,71	24,14
First improving	I	5,0	5,7	11	10,3	6	10,7	8,12
GRASP+VNS-A+VND	C	36,05	43,95	90,59	29,95	44,89	24,52	44,99
Alfa: 0,7; 15 iterações	V	0,0	0,0	0	0,0	0,0	0,0	0,00
VNS com 25 interações	T	23,12	27,36	30,61	20,95	27,05	21,09	19,93
First improving	I	10,3	6,0	7	7,3	4,7	3,0	6,38
GRASP+VND; Por grupos; Alfa: 0,9	C	37,64	38,86	82,69	20,21	39,11	17,86	39,40
VNS-B+VND; 50 iterações	V	0,0	0,0	0,0	0,0	0,0	0,0	0,00
First improving	T	5,54	4,35	9,31	3,42	6,58	3,63	5,47

Tabela 4.29: Resultados médios das melhores estratégias, recapitulados. Estratégias que atingiram viabilidade total.

# Capítulo 5

## Conclusão e Trabalhos Futuros

Este trabalho apresentou o problema de programação de horários no contexto universitário e um estudo de caso sobre o Departamento de Ciência da Computação da Universidade Federal Rural do Rio de Janeiro. O problema, de complexidade NP-difícil, é tipicamente tratado através de técnicas heurísticas e nesse estudo foram descritas e adaptadas as meta-heurísticas GRASP, VND, VNS e SA para produzir soluções de seis instâncias reais do departamento. O desempenho de cada técnica foi avaliado e algumas estratégias combinando foram propostas e estudadas. Além disso, foi produzida uma ferramenta de software de código aberto com interface gráfica que permite a entrada de dados de instâncias e a configuração de estratégias de execução entre as meta-heurísticas implementadas.

Os resultados mostram que diversas estratégias combinando Construção GRASP, GRASP+VND, VNS+VND e *Simulated annealing* foram capazes de produzir soluções viáveis de qualidade superior às soluções manuais utilizadas pelo DCC para todas as instâncias estudadas. A execução no método VNS+VND partindo de soluções boas geradas por outros métodos é especialmente eficiente na produção e soluções viáveis para instâncias difíceis, mas também, quando executado após GRASP+VND, pode levar aos melhores resultados em termos de qualidade das soluções.

Para soluções menos difíceis e mais comuns quase todos os métodos aplicados

foram capazes de produzir soluções viáveis melhores que as manuais em tempos inferiores a 20 minutos, a exceção sendo a técnica de *Simulated annealing* que para algumas instâncias não superou a solução manual, usando configurações de parâmetros mais rápidas. Já para as instâncias mais complicadas, a estratégia mais eficiente na tarefa de produzir viabilidade foi o uso de VNS+VND na fase de busca local do GRASP, estratégia esta que também levou aos segundo e terceiro melhores valores métidos para o custo.

Além disso, as soluções foram geradas em tempo extremamente inferior ao necessário no método manual, o que mostra que as técnicas e o software desenvolvido ao longo deste trabalho são opções de automatização adequadas ao contexto do departamento.

Como trabalhos futuros é sugerido que as estratégias sejam executadas em instâncias maiores de outras instituições para avaliar seus desempenhos principalmente no que tange a capacidade de produzir soluções viáveis em casos com maior número de variáveis e restrições. Explorar a execução paralela das iterações do GRASP em ambientes computacionais multi-núcleo e seus efeitos sobre o tempo total real da execução das estratégias também parece interessante, levando em consideração as necessidades práticas que devem ser consideradas quando o objetivo é produzir uma ferramenta para uso nas instituições. Outro estudo interessante pode se focar na determinação automatizada dos pesos dos grupos de restrições na função objetivo (nas linhas do procedimento de "Relaxação adaptativa" descrito em [32]) e seus efeitos tanto na qualidade final das soluções quanto da capacidade dos algoritmos de adaptarem a novas instâncias sem a necessidade de um estudo caso a caso.

# Referências

- [1] ITC 2011 | International Timetabling Competition 2011 | High School Timetabling Project (HSTT).
- [2] ABDULLAH, S. *Heuristic approaches for university timetabling problems*. PhD thesis, University of Nottingham Nottingham, 2006.
- [3] BORGES, S. K. *Resolução de Timetabling utilizando algoritmos genéticos e evolução cooperativa*. PhD thesis, Universidade Federal do Paraná, 2003.
- [4] CARVALHO, A. S., MARIANO, G. P., KAMPKE, E. H., E MAURI, G. R. SIMULATED ANNEALING APLICADO AO PROBLEMA DE PROGRAMAÇÃO DE HORÁRIOS DO CCA-UFES. *Blucher Marine Engineering Proceedings* 2, 1 (2016), 341–352.
- [5] COSTA, L. C. A. D., ACIOLI, L. F., E SUBRAMANIAN, A. UM MODELO DE PROGRAMAÇÃO INTEIRA PARA o PROBLEMA DE ALOCAÇÃO DE PROFESSORES a TURMAS DO DEPARTAMENTO DE ENGENHARIA DE PRODUÇÃO DE UMA INSTITUIÇÃO DE ENSINO SUPERIOR. *XLIV Simpósio Brasileiro de Pesquisa Operacional* (2012).
- [6] DE ARAUJO, R. L., E SECCHIN, L. D. O problema da elaboração de grade de horários: estudo de caso em uma Universidade Brasileira. *Anais do CMAC Nordeste* (2012).
- [7] DORNELES, A. P., DE ARAUJO, O. C. B., E BURIOL, L. S. A fix-and-optimize heuristic for the high school timetabling problem. *Computers & Operations Research* 52, Part A (dezembro de 2014), 29–38.

- [8] FEO, T. A., e RESENDE, M. G. C. Greedy Randomized Adaptive Search Procedures. *Journal of Global Optimization* 6, 2 (março de 1995), 109–133.
- [9] FESTA, P., e RESENDE, M. G. GRASP: An annotated bibliography. *Essays and surveys on metaheuristics* 6 (2002), 325–367.
- [10] FONSECA, G. H. G., e SANTOS, H. G. Variable Neighborhood Search based algorithms for high school timetabling. *Computers & Operations Research* 52, Part B (dezembro de 2014), 203–208.
- [11] FONSECA, G. H. G., SANTOS, H. G., e CARRANO, E. G. Late acceptance hill-climbing for high school timetabling. *Journal of Scheduling* 19, 4 (agosto de 2016), 453–465.
- [12] FONSECA, G. H. G. D. *Métodos de busca heurística para problemas de programação de horários modelados em XHSTT*. PhD thesis, Universidade Federal de Ouro Preto, 2013.
- [13] FONSECA, G. H. G. D., SANTOS, H. G., TOFFOLO, T. Â. M., BRITO, S. S., e SOUZA, M. J. F. GOAL solver: a hybrid local search based solver for high school timetabling. *Annals of Operations Research* 239, 1 (abril de 2016), 77–97.
- [14] GÓES, A. R. T. *Otimização na distribuição da carga horária de professores: método exato, método heurístico, método misto e interface*. PhD thesis, Universidade Federal do Paraná, 2005.
- [15] GÓES, A. R. T., COSTA, D. M. B., e STEINER, M. T. A. Otimização na programação de horários de professores/turmas: Modelo Matemático, Abordagem Heurística e Método Misto. *Revista Eletrônica Sistemas & Gestão* 5, 1 (2010), 50–66.
- [16] HAMAWAKI, C. D. L. *Geração automática de grade horária usando algoritmos genéticos: o caso da Faculdade de Engenharia Elétrica da UFU*. PhD thesis, Universidade Federal de Uberlândia, novembro de 2005.

- [17] HANSEN, P., E MLADENOVIC, N. An Introduction to Variable Neighborhood Search. In *Meta-Heuristics*. Springer, Boston, MA, 1999, pp. 433–458. DOI: 10.1007/978-1-4615-5775-3\_30.
- [18] HIME, R. Uma aplicação da Programação Inteira no School Timetabling Problem. Projeto Final, UFRPE, 2015.
- [19] JARDIM, A. M., SEMAAN, G. S., E PENNA, P. H. V. UMA HEURÍSTICA PARA O PROBLEMA DE PROGRAMAÇÃO DE HORÁRIOS: UM ESTUDO DE CASO. *XLVIII Simpósio Brasileiro de Pesquisa Operacional* (2016), 12.
- [20] KINGSTON, J. A software library for school timetabling. URL: <http://sydney.edu.au/engineering/it/jeff/khe> (2012).
- [21] LALESCU, L., E DIRR, V. FET. <http://lalescu.ro/liviu/fet/>. Acesso em: Abril/2017.
- [22] LOBO, E. L. M. Uma solução do problema de horário escolar via algoritmo genético paralelo. *Centro Federal de Educação Tecnológica de Minas Gerais* (2005).
- [23] MARCO P. CARRASCO, E MARGARIDA V. PATO. A multiobjective genetic algorithm for the class/teacher timetabling problem | SpringerLink. In *Practice and Theory of Automated Timetabling III* (2001).
- [24] MARTE, M. *Models and Algorithms for School Timetabling*. PhD thesis, lmu, 2002.
- [25] MLADENOVIC, N., E HANSEN, P. Variable neighborhood search. *Computers & Operations Research* 24, 11 (novembro de 1997), 1097–1100.
- [26] OLIVEIRA, J. G. D., VIANNA, D. S., E VIANNA, M. D. F. D. A GRASP+VND HEURISTIC FOR THE SCHOOL TIMETABLING PROBLEM. *Sistemas & Gestão* 7, 3 (novembro de 2012), 326–335.
- [27] POST, G., KINGSTON, J. H., AHMADI, S., DASKALAKI, S., GOGOS, C., KYNGAS, J., NURMI, C., MUSLIU, N., PILLAY, N., SANTOS, H., E OTHERS.

- Xhstt: an xml archive for high school timetabling problems in different countries. *Annals of Operations Research* 218, 1 (2014), 295–301.
- [28] PRAIS, M., E RIBEIRO, C. C. Reactive GRASP: An Application to a Matrix Decomposition Problem in TDMA Traffic Assignment. *INFORMS Journal on Computing* 12, 3 (agosto de 2000), 164–176.
- [29] RIBEIRO, D. M., AIZEMBERG, L., E UCHOA, E. GERAÇÃO DE GRADE DE HORÁRIOS PARA DISCIPLINAS DE UMA INSTITUIÇÃO DE NÍVEL SUPERIOR UTILIZANDO PROGRAMAÇÃO LINEAR INTEIRA MULTI-OBJETIVO. *XLV Simpósio Brasileiro de Pesquisa Operacional* (2013).
- [30] SANTOS, H. G. Formulações e Algoritmos para o Problema de Programação de Horários em Escolas. *Universidade Federal Fluminense, Brasil* (2007).
- [31] SANTOS, H. G., E SOUZA, M. J. F. Programação de horários em instituições educacionais: formulações e algoritmos. *XXXIX SBPO-Simpósio Brasileiro de Pesquisa Operacional*, 1 (2007), 2827–2882.
- [32] SCHAERF, A. *Tabu search techniques for large high-school timetabling problems*. Centrum voor Wiskunde en Informatica, 1996.
- [33] SCHAERF, A. A survey of automated timetabling. *Artificial Intelligence Review* 13, 2 (1999), 87–127.
- [34] SCHNEIDER, J., E KIRKPATRICK, S. Construction Heuristics. In *Stochastic Optimization*, Scientific Computation. Springer, Berlin, Heidelberg, 2006, pp. 59–61. DOI: 10.1007/978-3-540-34560-2\_8.
- [35] SOUZA, M. J. F. *PROGRAMAÇÃO DE HORÁRIOS EM ESCOLAS: UMA APROXIMAÇÃO POR METAHEURÍSTICAS*. PhD thesis, UNIVERSIDADE FEDERAL DO RIO DE JANEIRO, 2000.
- [36] SOUZA, M. J. F., MACULAN, N., E OCHI, L. S. A GRASP-Tabu Search Algorithm for Solving School Timetabling Problems. In *Metaheuristics: Computer Decision-Making*, Applied Optimization. Springer, Boston, MA, 2003, pp. 659–672. DOI: 10.1007/978-1-4757-4137-7\_31.

- [37] VIEIRA, F., E MACEDO, H. Sistema de Alocação de Horários de Cursos Universitários: Um Estudo de Caso no Departamento de Computação da Universidade Federal de Sergipe. *Scientia Plena* 7, 3 (abril de 2011).
- [38] VOSS, S., MARTELLO, S., OSMAN, I. H., E ROUCAIROL, C. *Meta-heuristics: Advances and trends in local search paradigms for optimization*. Springer Science & Business Media, 2012.
- [39] WHITLEY, D., RANA, S., E HECKENDORN, R. B. The Island Model Genetic Algorithm: On Separability, Population Size and Convergence. *Journal of computing and information technology* 7, 1 (março de 1999), 33–47.
- [40] WREN, A. Scheduling, timetabling and rostering — a special relationship? In *Practice and Theory of Automated Timetabling* (1996), E. Burke e P. Ross, Eds., Lecture Notes in Computer Science, Springer Berlin Heidelberg, pp. 46–75.
- [41] XAVIER, B. M., DA SILVA, A. D., VIANNA, D. S., COSTA, H. G., E COELHO, W. B. Proposta de Alocação de Horários de Professores e Turmas em Instituições de Ensino Superior Utilizando uma Heurística VNS/VND. *XLVSBPO* (2013).

# Apêndice A

## A ferramenta de software

A ferramenta de software desenvolvida ao longo deste trabalho, por ter sido construída sobre o código aberto de uma ferramenta pré-existente, tem funcionalidades extensas e possuí diversas características que não são frutos diretos dos estudos aqui realizados, como as interfaces de inserção de dados das instâncias, impressão e exportação de grades horárias produzidas, entre outras.

A seguir são apresentadas algumas das interfaces gráficas que forma alteradas ou criadas no presente trabalho.

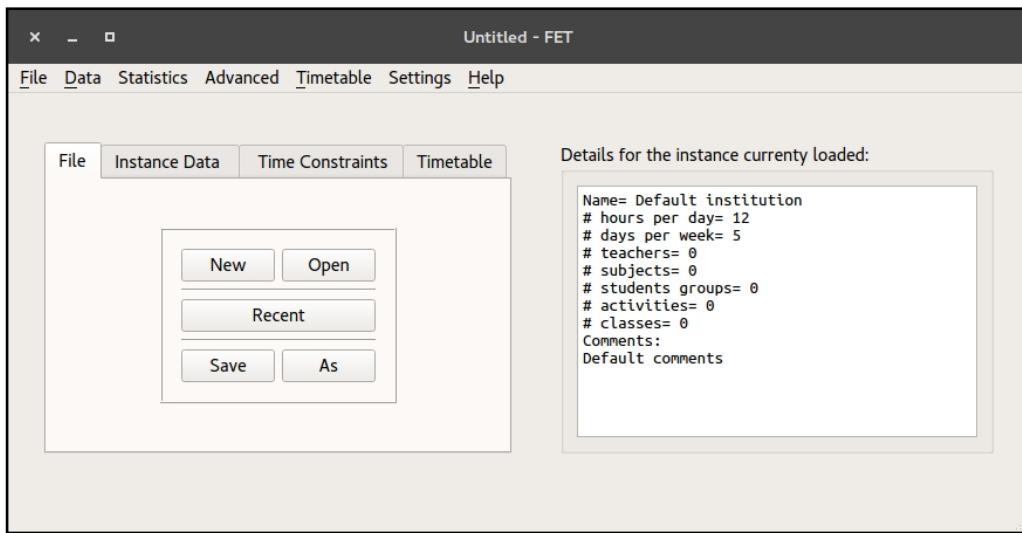


Figura A.1: Tela principal. Manipulação de arquivos com os dados das instâncias.

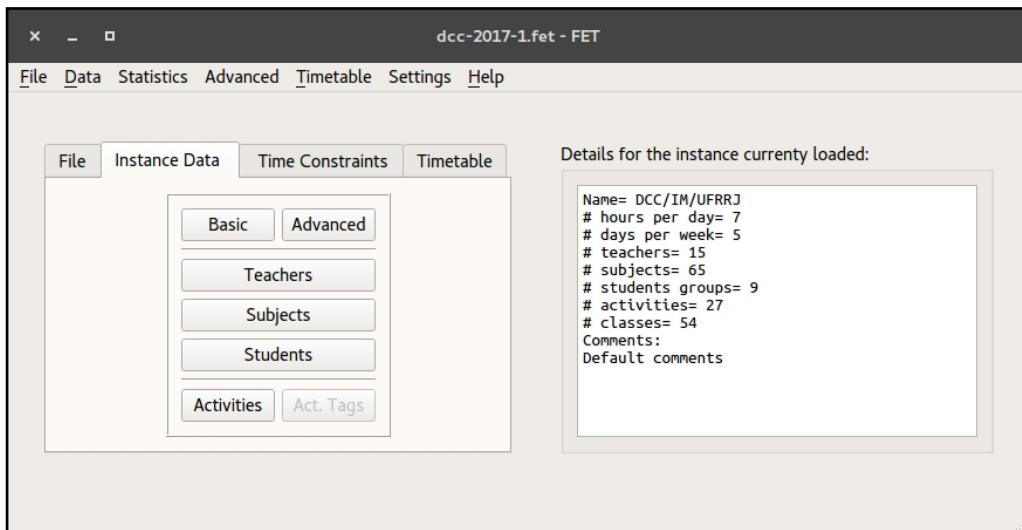


Figura A.2: Tela principal. Acesso à edição de dados da instância.

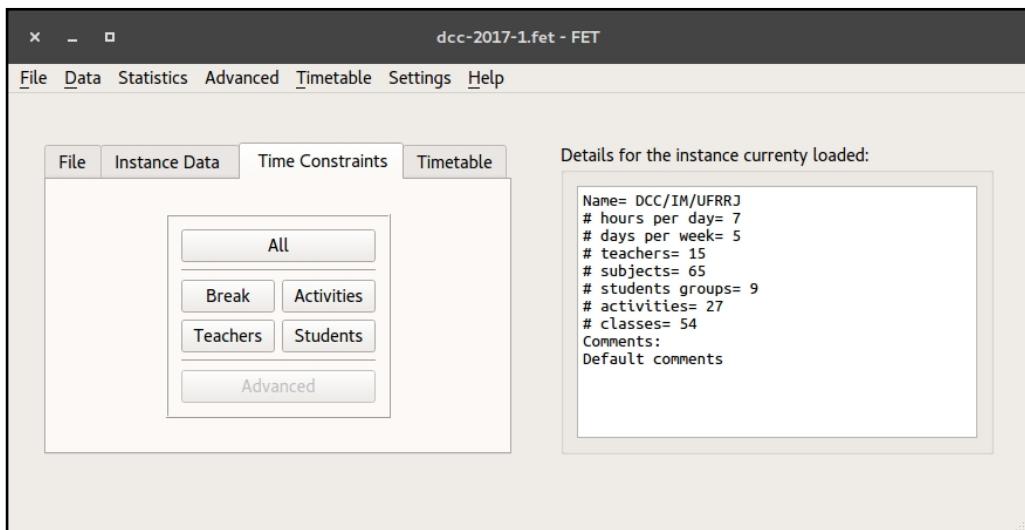


Figura A.3: Tela principal. Acesso à configuração de restrições de tempo.

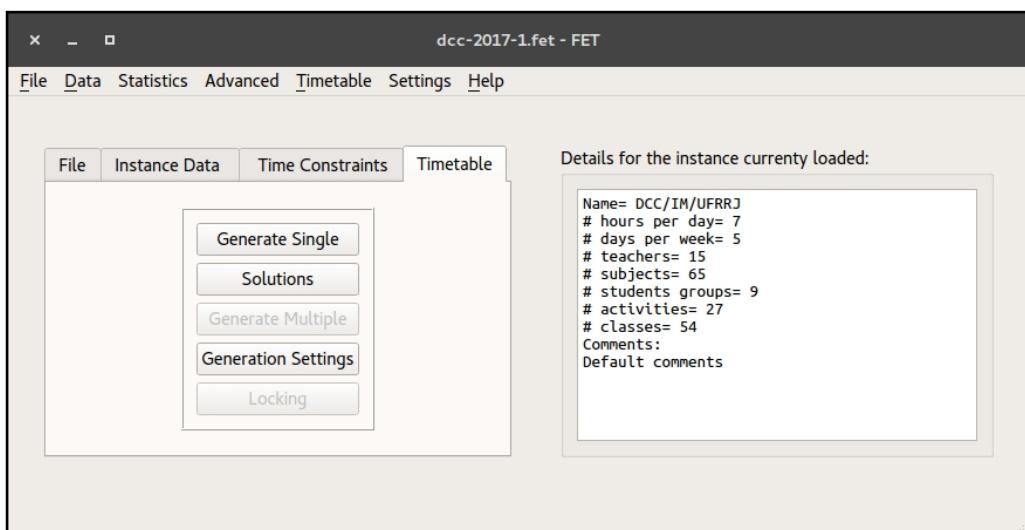


Figura A.4: Tela principal. Acesso às funções de geração de grades, configurações das estratégias e soluções geradas.

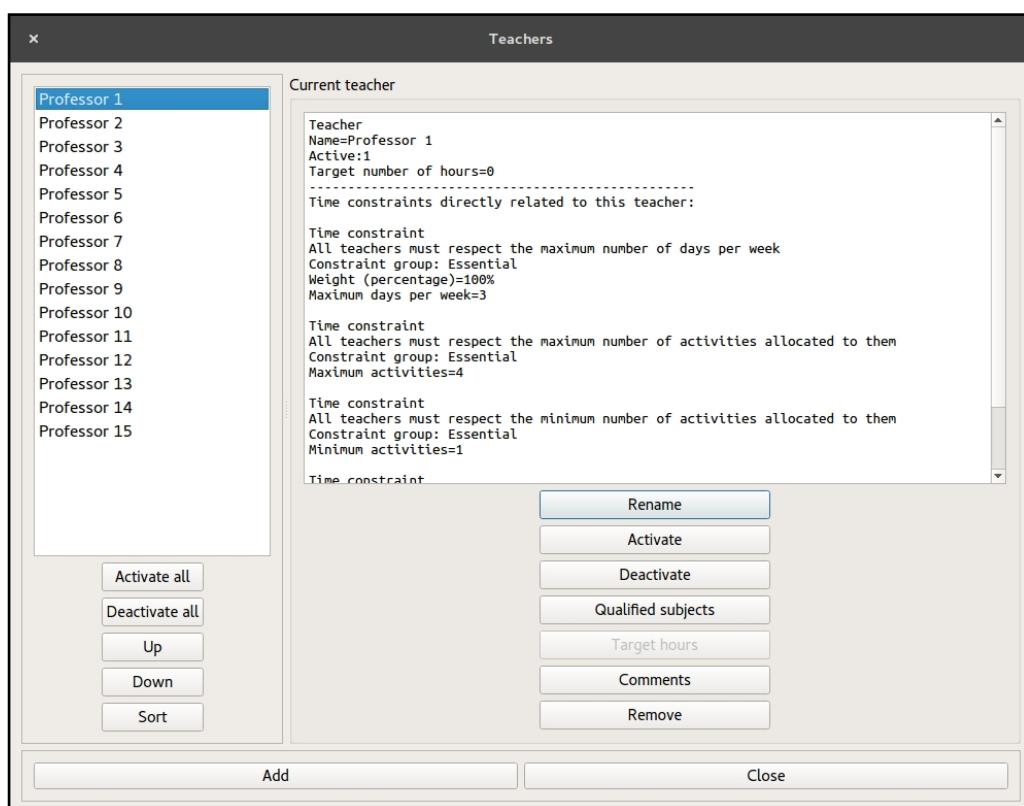


Figura A.5: Lista de professores.

**Teacher subjects qualifications**

Teacher: Professor 8

Subject:	Preference:
<input checked="" type="checkbox"/> Algoritmos Paralelos	Very_Low
<input type="checkbox"/> Análise de Algoritmos	Indifferent
<input type="checkbox"/> Aprendizado de máquina	Indifferent
<input type="checkbox"/> Arquitetura de Computadores I	Indifferent
<input type="checkbox"/> Arquitetura de Computadores II	Indifferent
<input checked="" type="checkbox"/> Arquiteturas Paralelas	Very_Low
<input checked="" type="checkbox"/> Banco de Dados	Very_High
<input checked="" type="checkbox"/> Banco de Dados II	Very_High
<input type="checkbox"/> Circuitos Digitais	Indifferent
<input type="checkbox"/> Circuitos eletrônicos	Indifferent
<input type="checkbox"/> Compiladores	Indifferent
<input checked="" type="checkbox"/> Computadores e Sociedade	Low
<input checked="" type="checkbox"/> Computação Gráfica	Low

OK Cancel

Subject	Preference
Algoritmos Paralelos	Very_Low
Análise de Algoritmos	Indifferent
Aprendizado de máquina	Indifferent
Arquitetura de Computadores I	Indifferent
Arquitetura de Computadores II	Indifferent
Arquiteturas Paralelas	Very_Low
Banco de Dados	Very_High
Banco de Dados II	Very_High
Circuitos Digitais	Indifferent
Circuitos eletrônicos	Indifferent
Compiladores	Indifferent
Computadores e Sociedade	Low
Computação Gráfica	Low

Figura A.6: Edição das preferências de um professor por disciplinas.

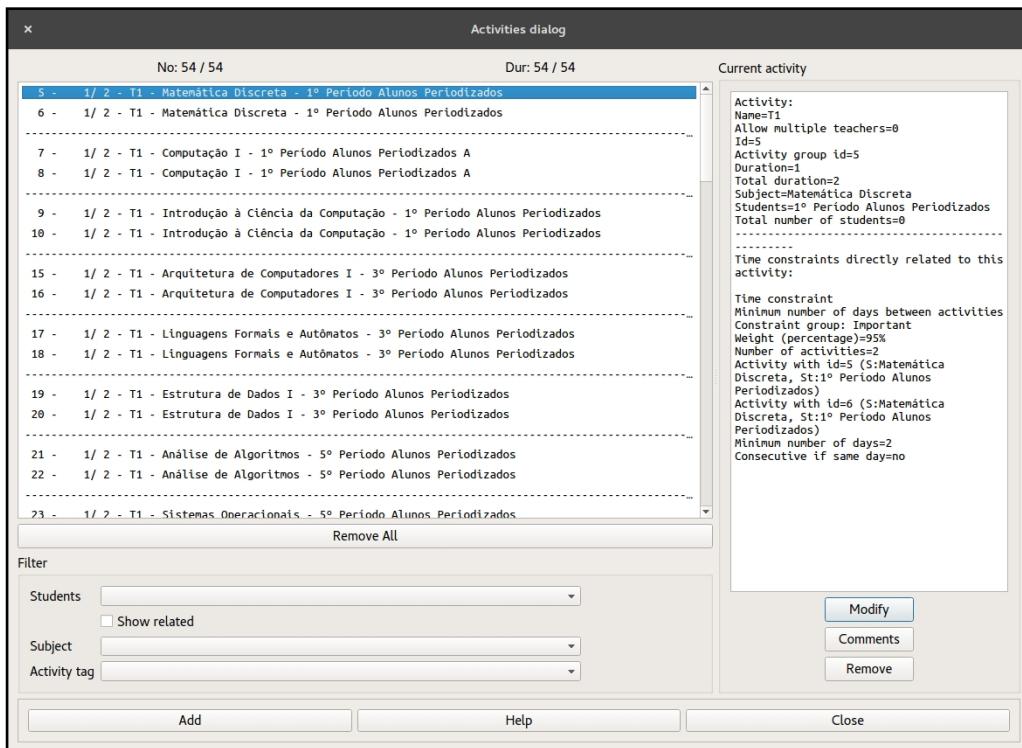


Figura A.7: Lista de atividades da instância.

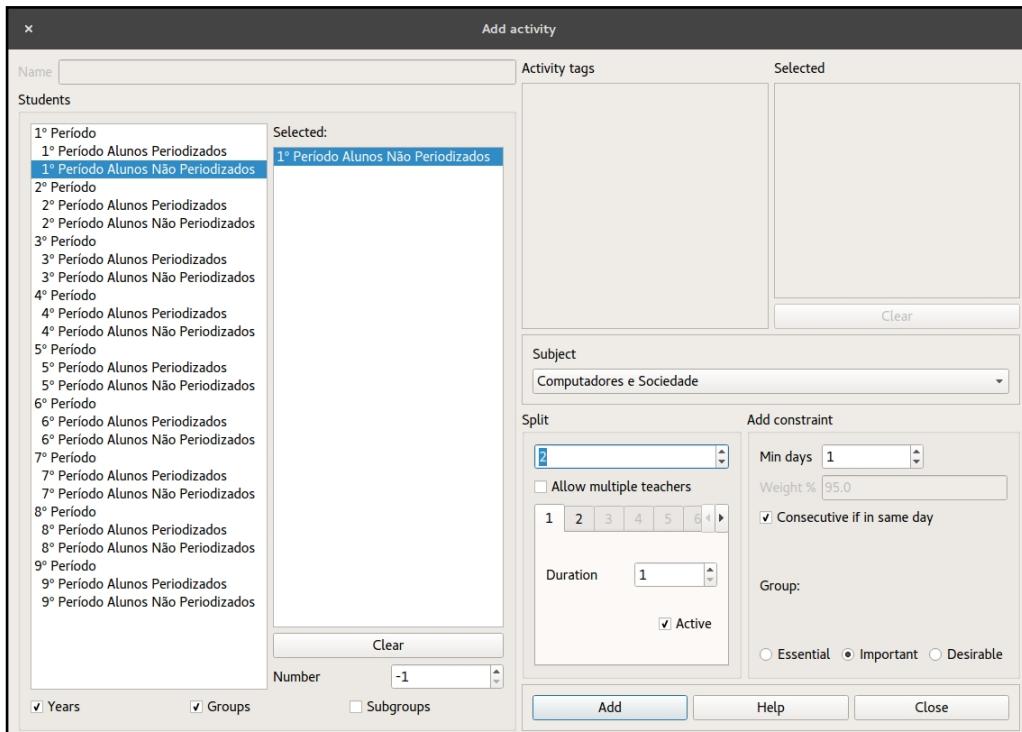


Figura A.8: Edição/Inclusão de uma atividade.

All time constraints

**Basic compulsory constraints (time), G: Essential, WP:100%**

Teachers max days per week, G: Essential, WP:100%, MD:3  
 Teachers max activities, G: Essential, MA:4  
 Teachers min activities, G: Essential, MA:1  
 Students set not available, G: Essential, WP:100%, St:1º Período Alunos Periodizados, NA at: Segunda 08:00; Segunda 10:00; Segunda 18:00; Seg  
 Students set not available, G: Essential, WP:100%, St:2º Período Alunos Periodizados, NA at: Segunda 08:00; Segunda 10:00; Segunda 18:00; Seg  
 Students set not available, G: Essential, WP:100%, St:3º Período Alunos Periodizados, NA at: Segunda 08:00; Segunda 10:00; Segunda 18:00; Seg  
 Students set not available, G: Essential, WP:100%, St:4º Período Alunos Periodizados, NA at: Segunda 08:00; Segunda 10:00; Segunda 18:00; Seg  
 Students set not available, G: Essential, WP:100%, St:5º Período Alunos Periodizados, NA at: Segunda 14:00; Segunda 16:00; Segunda 18:00; Seg  
 Students set not available, G: Essential, WP:100%, St:6º Período Alunos Periodizados, NA at: Segunda 14:00; Segunda 16:00; Segunda 18:00; Seg  
 Students set not available, G: Essential, WP:100%, St:7º Período Alunos Periodizados, NA at: Segunda 14:00; Segunda 16:00; Segunda 18:00; Seg  
 Students set not available, G: Essential, WP:100%, St:8º Período Alunos Periodizados, NA at: Segunda 14:00; Segunda 16:00; Segunda 18:00; Seg  
 Students set not available, G: Essential, WP:100%, St:9º Período Alunos Periodizados, NA at: Segunda 14:00; Segunda 16:00; Segunda 18:00; Seg  
 Break times, G: Essential, WP:100%, B at: Segunda 12:00; Terça 12:00; Quinta 12:00; Sexta 12:00  
 Students set not available, G: Essential, WP:100%, St:1º Período Alunos Não Periodizados, NA at: Segunda 20:00; Terça 20:00; Quarta 20:00; Qui  
 Students set not available, G: Essential, WP:100%, St:2º Período Alunos Não Periodizados, NA at: Segunda 20:00; Terça 20:00; Quarta 20:00; Qui  
 Students set not available, G: Essential, WP:100%, St:3º Período Alunos Não Periodizados, NA at: Segunda 20:00; Terça 20:00; Quarta 20:00; Qui  
 Students set not available, G: Essential, WP:100%, St:4º Período Alunos Não Periodizados, NA at: Segundo 20:00; Terça 20:00; Quarta 20:00; Qui  
 Students set not available, G: Essential, WP:100%, St:5º Período Alunos Não Periodizados, NA at: Segundo 18:00; Segunda 20:00; Terça 18:00; Te  
 Students set not available, G: Essential, WP:100%, St:6º Período Alunos Não Periodizados, NA at: Segunda 18:00; Segunda 20:00; Terça 18:00; Te  
 Students set not available, G: Essential, WP:100%, St:7º Período Alunos Não Periodizados, NA at: Segunda 18:00; Segunda 20:00; Terça 18:00; Te  
 Students set not available, G: Essential, WP:100%, St:8º Período Alunos Não Periodizados, NA at: Segunda 18:00; Segunda 20:00; Terça 18:00; Te  
 Students set not available, G: Essential, WP:100%, St:9º Período Alunos Não Periodizados, NA at: Segunda 18:00; Segunda 20:00; Terça 18:00; Te  
 Min days between activities, G: Important, WP:95%, NA:2, Id:5, Id:6, mD:2, CSD:no  
 Min days between activities, G: Important, WP:95%, NA:2, Id:7, Id:8, mD:2, CSD:no  
 Min days between activities, G: Important, WP:95%, NA:2, Id:9, Id:10, mD:2, CSD:no  
 Min days between activities, G: Important, WP:95%, NA:2, Id:15, Id:16, mD:2, CSD:no  
 Min days between activities, G: Important, WP:95%, NA:2, Id:17, Id:18, mD:2, CSD:no  
 Min days between activities, G: Important, WP:95%, NA:2, Id:19, Id:20, mD:2, CSD:no  
 Min days between activities, G: Important, WP:95%, NA:2, Id:21, Id:22, mD:2, CSD:no  
 Min days between activities, G: Important, WP:95%, NA:2, Id:23, Id:24, mD:2, CSD:no  
 Min days between activities, G: Important, WP:95%, NA:2, Id:25, Id:26, mD:2, CSD:no

Current constraint

These are the basic compulsory constraints (referring to time allocation) for any timetable

Constraint group: Essential

Weight (percentage)=100%

The basic time constraints try to avoid:

- teachers assigned to more than one activity simultaneously
- students assigned to more than one activity simultaneously

Modify Remove

Activate Deactivate

Comments

Up Down

Filter Sorted

Close

Figura A.9: Lista de todas as restrições da instância.

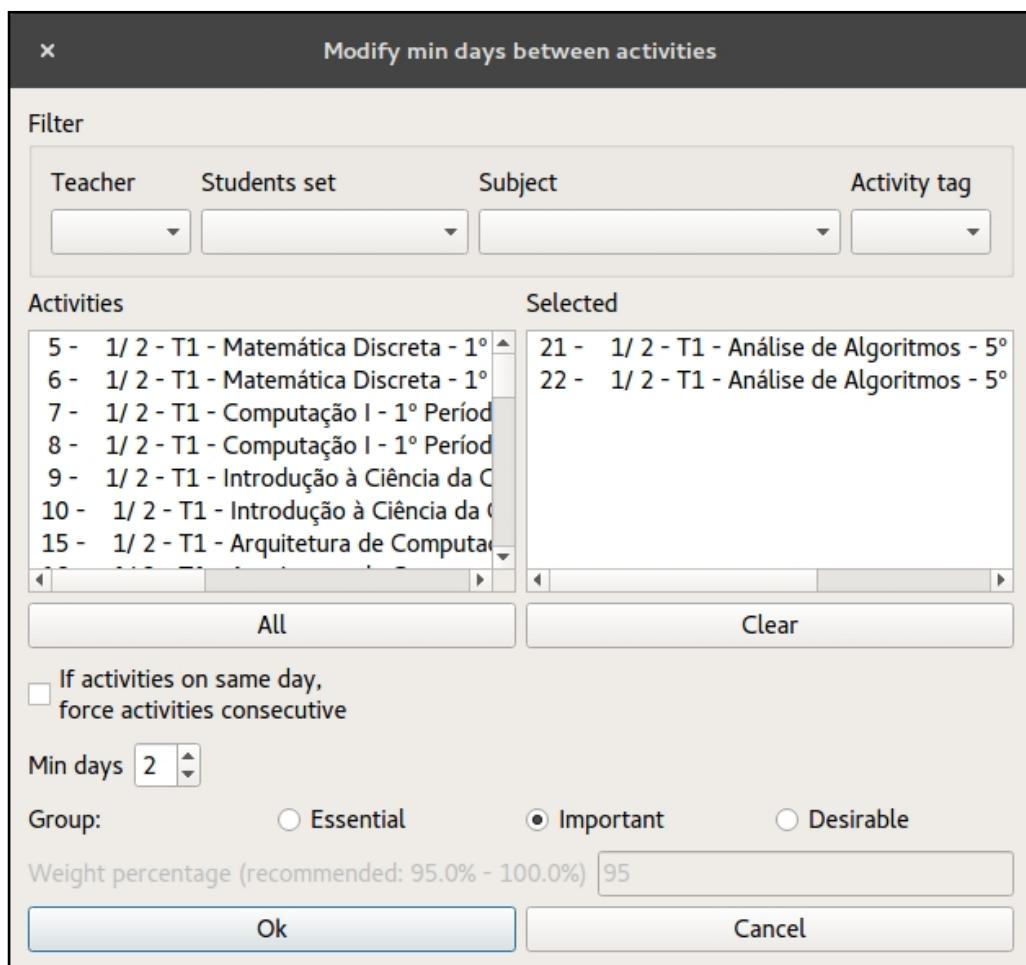


Figura A.10: Exemplo de edição/inclusão: restrição do mínimo número de dias entre atividades.

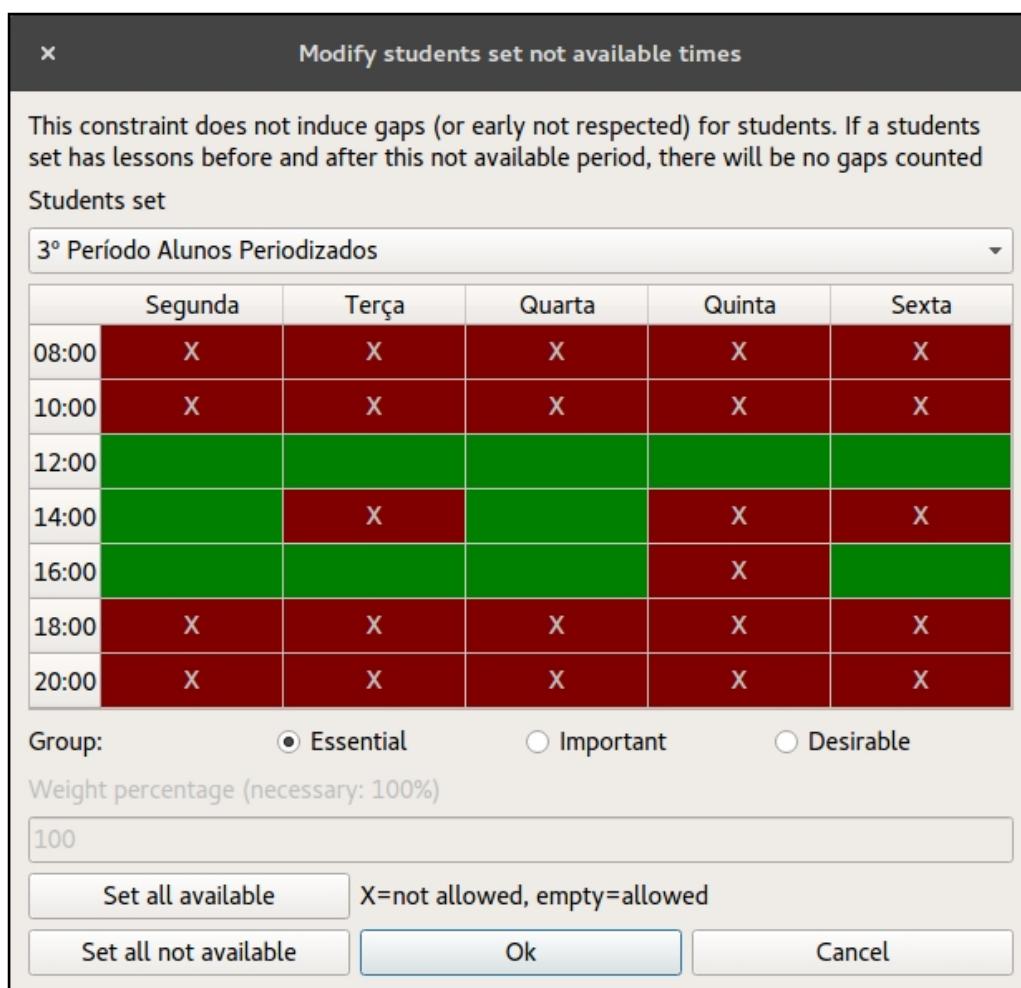


Figura A.11: Exemplo de edição/inclusão: restrição sobre horários não disponíveis para um grupo de alunos.

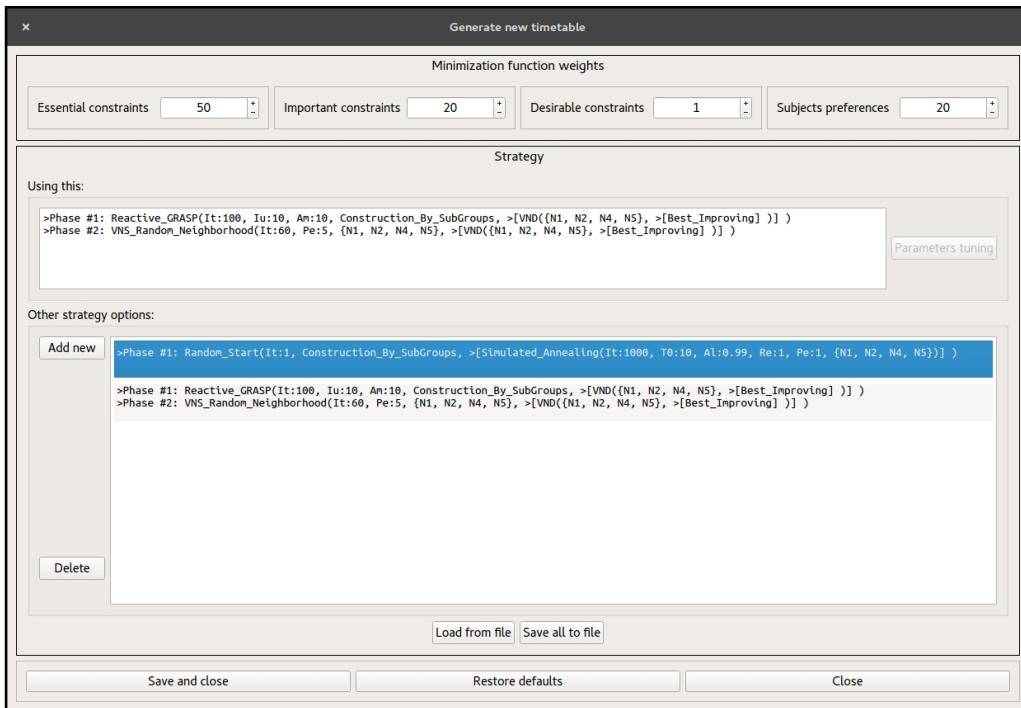


Figura A.12: Interface de configuração e seleção das estratégias de geração.

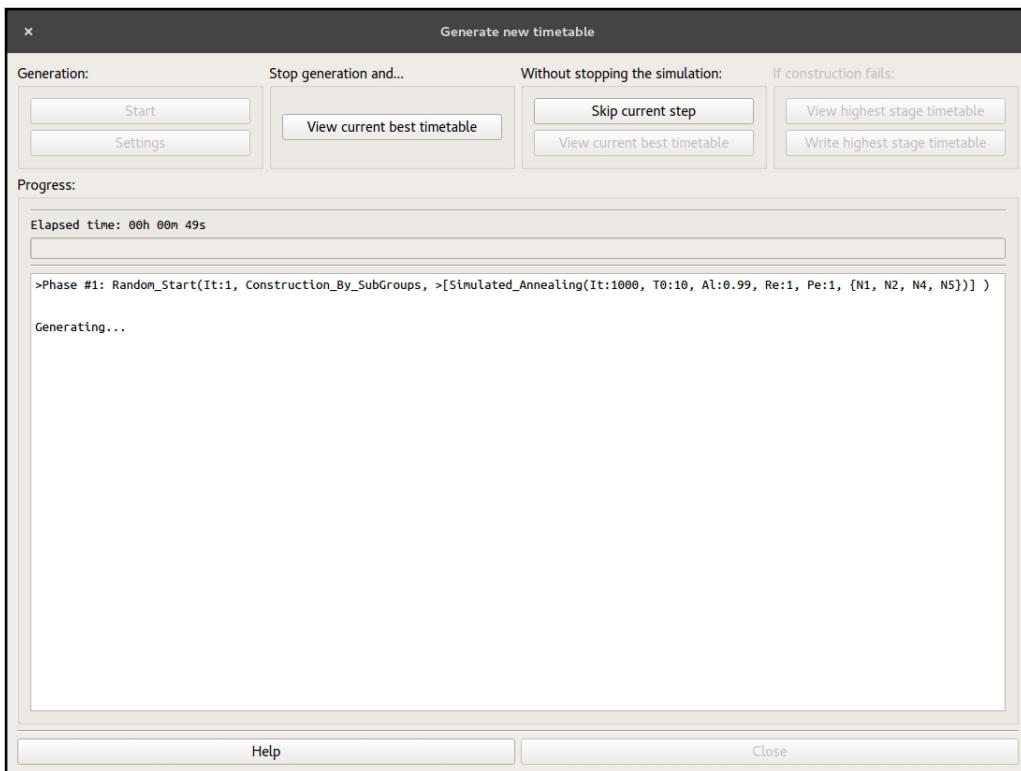


Figura A.13: Interface para a geração das grades horárias (durante o processo de geração).

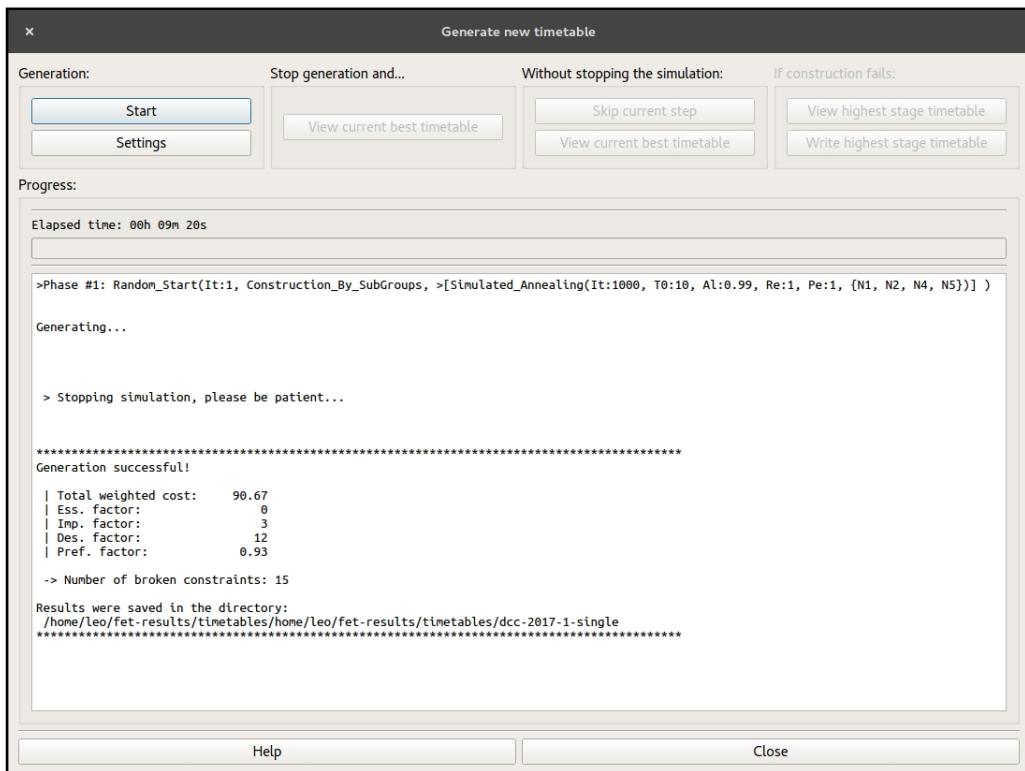


Figura A.14: Interface para a geração das grades horárias (geração concluída).

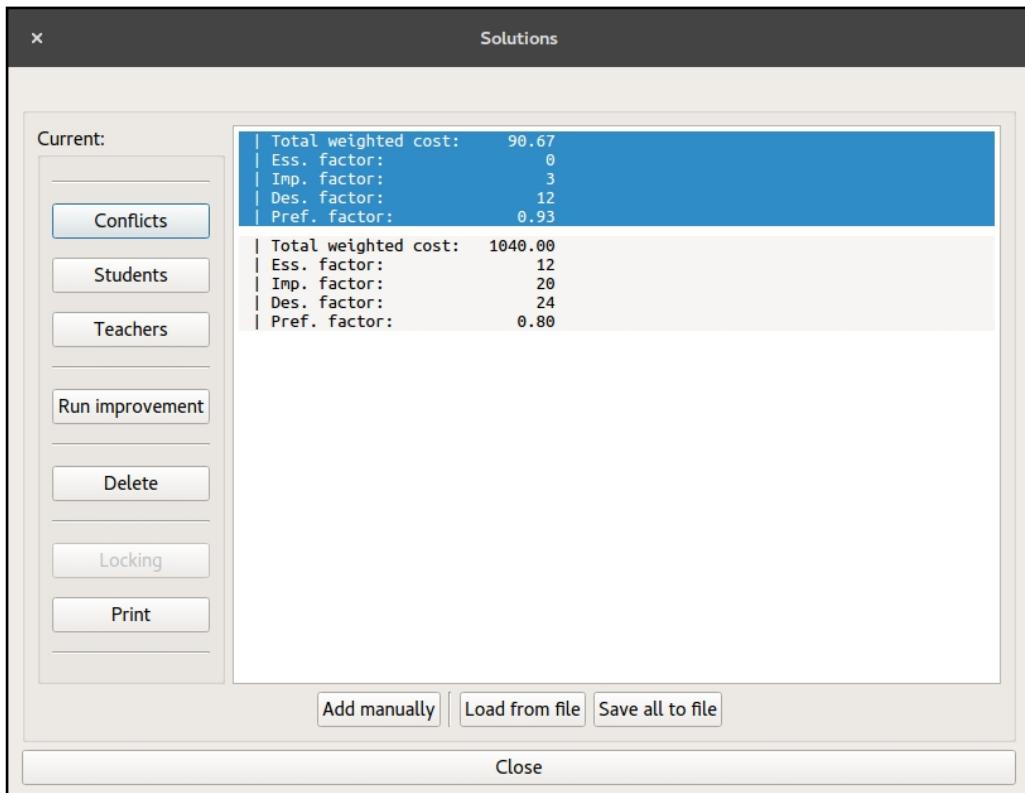


Figura A.15: Lista de soluções geradas.

Conflicts

```

| Total weighted cost:           55
| Essential constraints violation factor:  0
| Important constraints violation factor:  2
| Desirable constraints violation factor:  7
| Preference violation factor:    0.4

* Teachers allocated to 'Very_High' preference: 15      -> Total of subactivities: 50
  ↳ Professor 13: 2 subactivities
  ↳ Professor 14: 4 subactivities
  ↳ Professor 3: 4 subactivities
  ↳ Professor 4: 4 subactivities
  ↳ Professor 1: 2 subactivities
  ↳ Professor 2: 4 subactivities
  ↳ Professor 7: 4 subactivities
  ↳ Professor 8: 2 subactivities
  ↳ Professor 5: 4 subactivities
  ↳ Professor 6: 4 subactivities
  ↳ Professor 11: 4 subactivities
  ↳ Professor 12: 2 subactivities
  ↳ Professor 9: 4 subactivities
  ↳ Professor 10: 2 subactivities
  ↳ Professor 15: 4 subactivities

* Teachers allocated to 'High' preference: 1            -> Total of subactivities: 2
  ↳ Professor 13: 2 subactivities

* Teachers allocated to 'Indifferent' preference: 1     -> Total of subactivities: 2
  ↳ Professor 1: 2 subactivities

* Teachers allocated to 'Low' preference: 0             -> Total of subactivities: 0

* Teachers allocated to 'Very_Low' preference: 0         -> Total of subactivities: 0

#####
Number of broken constraints: 9
Conflicts listing:

  ↳ Important constraints conflicts:
    #1: Time constraint min days between activities broken: activity with id=6 (S:Matemática Discreta, St:1º Período Alunos Periodizados) conflicts with activity with id=5 (S:Matemática Discreta, St:1º Período Alunos Periodizados), being 1 days too close, on days Quinta and Sexta, conflicts cost increase=20.

    #2: Constraint no teachers allocated to subjects of preference 'Indifferent'.

  ↳ Desirable constraints conflicts:
    #1: Constraint no teachers allocated to subjects of preference 'High'.

```

[Close](#)

Figura A.16: Descrição detalhada dos conflitos e custo de uma solução.

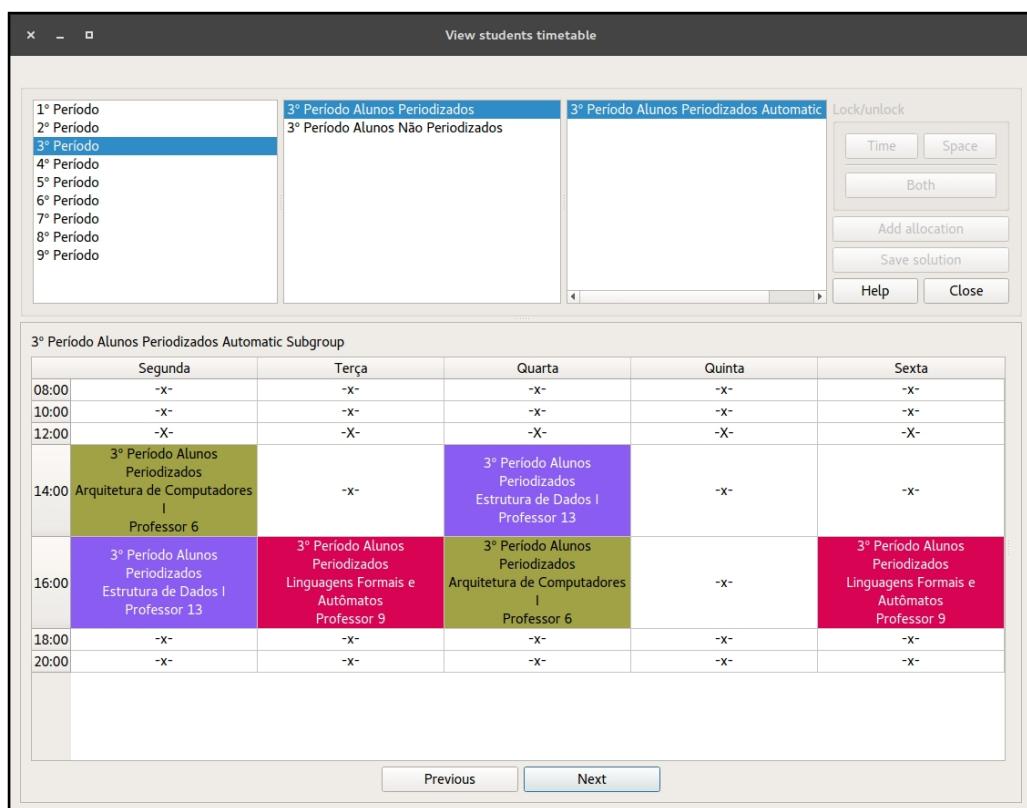


Figura A.17: Visualização de uma grade horária gerada (perspectiva dos grupos de alunos).

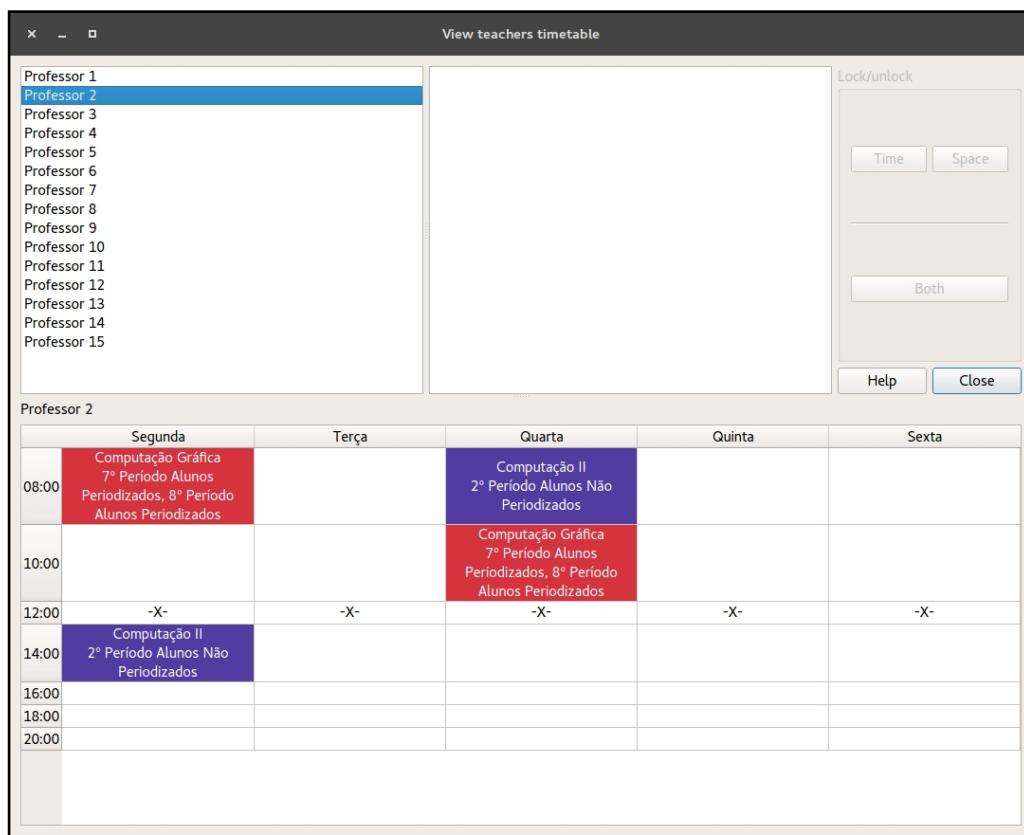


Figura A.18: Visualização de uma grade horária gerada (perspectiva dos professores).

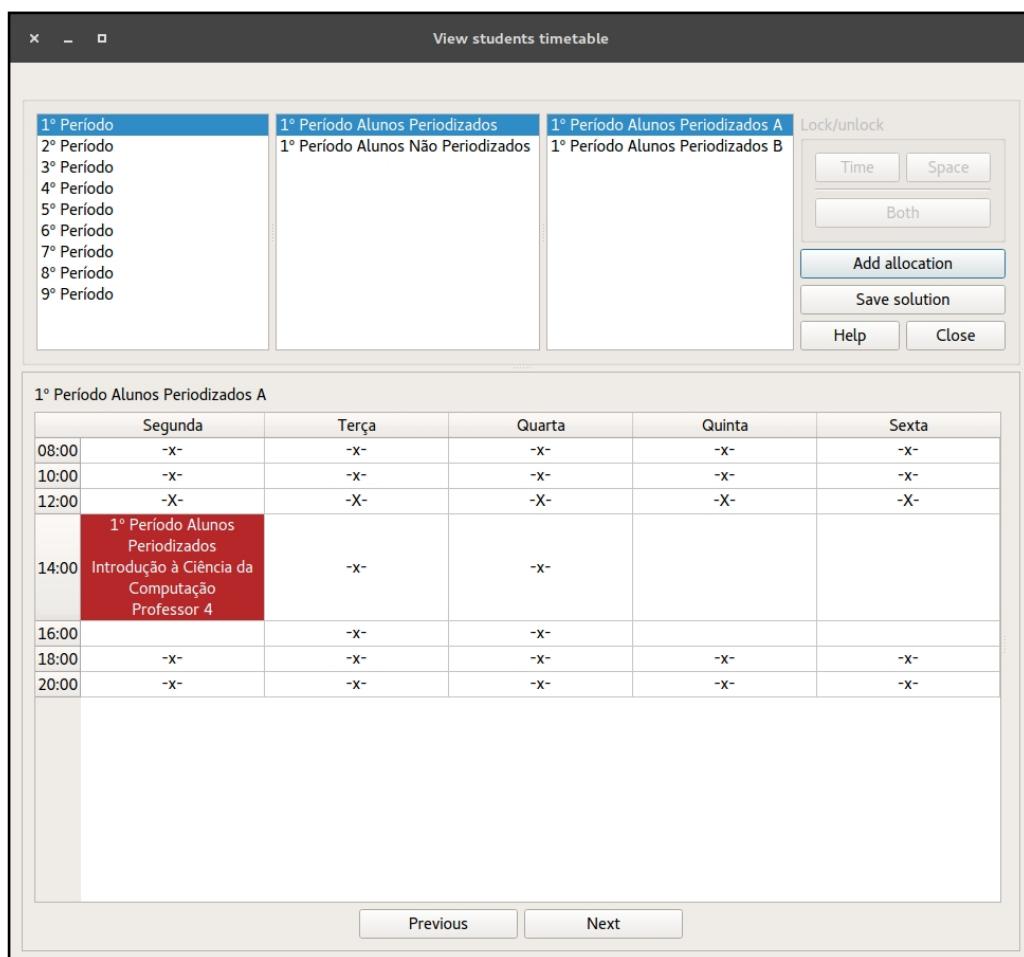


Figura A.19: Visualização de uma grade horária parcical para alocação manual.

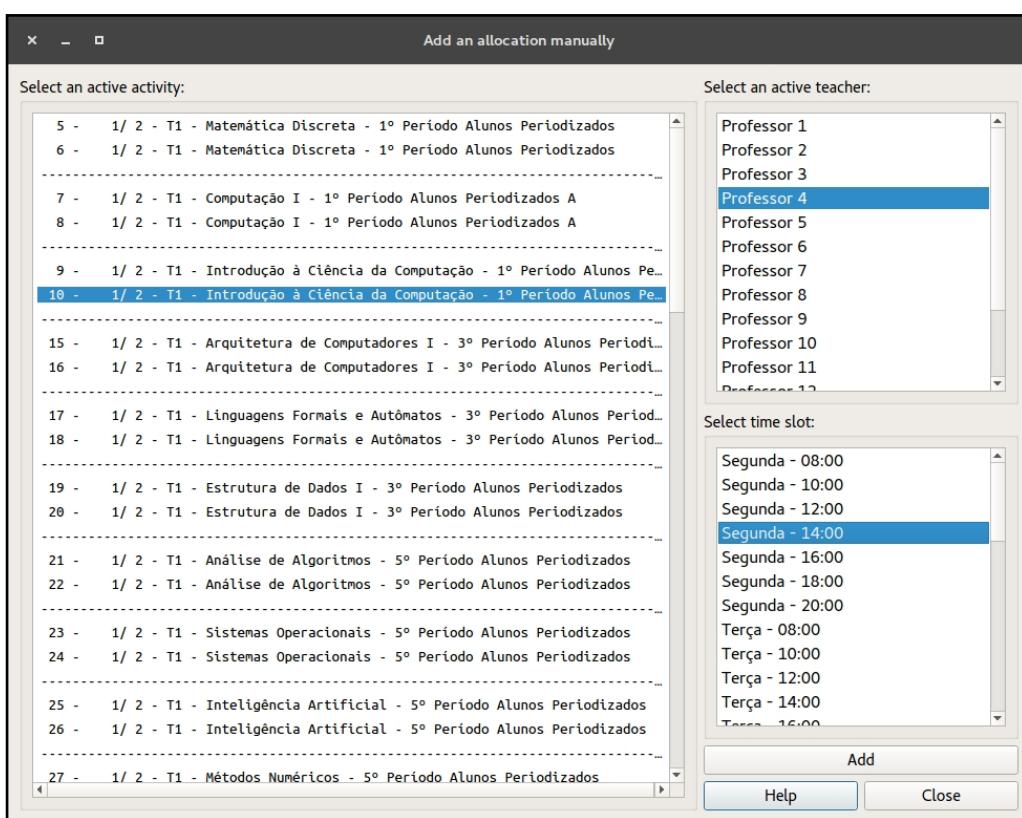


Figura A.20: Entrada manual de uma alocação.