UNIVERSIDADE FEDERAL DE GOIÁS
INSTITUTO DE INFORMÁTICA

JEAN PAULO MARTINS

O Problema do Agendamento Semanal de Aulas

Goiânia
2010

Jean Paulo Martins

O Problema do Agendamento Semanal de Aulas

Dissertação apresentada ao Programa de Pós–Graduação do Instituto de Informática da Universidade Federal de Goiás, como requisito parcial para obtenção do título de Mestre em Computação.

Área de concentração: Otimização.

Orientador: Prof. Humberto José Longo

Goiânia
2010

Jean Paulo Martins

O Problema do Agendamento Semanal de Aulas

Dissertação defendida no Programa de Pós–Graduação do Instituto de Informática da Universidade Federal de Goiás como requisito parcial para obtenção do título de Mestre em Computação, aprovada em 16 de Julho de 2010, pela Banca Examinadora constituída pelos professores:

Prof. Humberto José Longo
Instituto de Informática – UFG
Presidente da Banca

Prof. Cláudio Nogueira de Meneses
Universidade Federal do ABC – UFABC

Prof. Plínio de Sá Leitão
Instituto de Informática – UFG

Jean Paulo Martins

Graduou–se em Ciências da Computação na UFG - Universidade Federal de Goiás. Durante sua graduação, foi monitor no departamento de Matemática da UFG e pesquisador do CNPq em um trabalho de iniciação científica no Instituto de Informática.

Dedico este trabalho a meus pais: José Maria Martins e Maria das Dores Martins, e minhas duas irmãs: Bárbara Cristina Martins e Luciane Maria Martins.

Agradecimentos

É imprescindível agradecer à minha família, pelo apoio e confiança incondicionais, sem os quais tudo teria sido mais difícil.

Gostaria também, de agradecer ao professor Humberto José Longo pela prontidão e a disponibilidade em sempre ajudar e aconselhar as direções do trabalho, ao professor Cláudio Nogueira de Meneses pelo apoio e confiança que foram decisivos.

Agradeço também ao professor Plínio de Sá Leitão pelo auxílio na obtenção dos dados utilizados na resolução do *Problema do Agendamento Semanal de Aulas*, assim como pelos esclarecimentos sobre o problema e à colega Carine Rodrigues da Costa pelas sugestões e ajuda nas correções do texto.

Por fim, gostaria de agradecer à UFG pelo apoio financeiro sem o qual o desenvolvimento deste trabalho não teria se tornado possível.

– Vida, você pode odiá-la ou ignorá-la mas gostar dela é impossível.

Marvin, O Guia dos Mochileiros das galáxias,
Douglas Adams.

Resumo

P. Martins, Jean. **O Problema do Agendamento Semanal de Aulas**. Goiânia, 2010. 83p. Dissertação de Mestrado. Instituto de Informática, Universidade Federal de Goiás.

O *Agendamento Semanal de Aulas* é um problema de difícil resolução enfrentado em grande maioria das instituições de ensino. Assim como os demais problemas de *timetabling*, possui como característica principal a sua natureza associativa, ou seja, sua resolução envolve a associação entre uma certa quantidade de recursos e eventos que utilizarão tais recursos. Especificamente em relação ao problema em questão, as aulas a serem ministradas podem ser caracterizadas como eventos, enquanto que a carga horária dos professores envolvidos podem ser vistas como recursos disponíveis (*Programação de Horários de Aulas*). Técnicas e métodos de grande relevância na ciência da computação estão relacionados na pesquisa e na solução destes tipos de problemas, contudo, a utilização de tais tecnologias no cotidiano de escolas e universidades ainda é pequena. Neste contexto, propõe-se uma abordagem para a resolução de *Problemas de Programação de Horários*, incluindo o *Problema de Alocação de Professores a Disciplinas*, e utiliza-se o Instituto de Informática da Universidade Federal de Goiás como um estudo de caso para tal.

Palavras–chave

otimização combinatória, *curriculum-based course timetabling*, meta-heurísticas

Abstract

P. Martins, Jean. **Teacher Assignment and Course Scheduling**. Goiânia, 2010. 83p. MSc. Dissertation. Instituto de Informática, Universidade Federal de Goiás.

The Course Scheduling is a hard resolution problem, found in most of the learning institutions. Just like the others timetabling problems, the Course Scheduling have a strong associative characteristic, that means that its resolution is made of associations between events and resources. In the educational case, the lectures are events, while the teachers workload are resources. Techniques and methods have being used on the solution of these kind of problems, however is small the number of universities using software based solutions. This work is a starting point to software based solutions applied to the Federal University of Goiás.

Keywords

combinatorial optimization, *curriculum-based course timetabling*, metaheuristics

Sumário

Lista de Figuras

Lista de Tabelas

Lista de Algoritmos

CAPÍTULO 1

Introdução

Os problemas de *programação de horários* (*Timetabling*) são problemas de alocação que constituem uma área de intensa atividade de pesquisa com aplicações em diversos contextos. A criação de tabelas de horários para jogos e a organização de escalas de horário de trabalho, são exemplos de situações que descrevem esse tipo de problema em ambiente esportivo e empresarial, respectivamente.

Uma característica marcante, compartilhada por diversas variantes de problemas de *timetable* é a sua natureza fortemente associativa, ou seja, soluções para tais problemas são obtidas a partir de associações entre um número de *eventos* (ex.: aulas, palestras, jogos), e um número limitado de *recursos* (ex.: tempo, pessoas, salas) [35]. Sendo o objetivo principal, determinar como utilizar os recursos da melhor forma possível.

A elaboração de quadros de horários, apesar de bem definida, não é uma tarefa simples, e se torna mais complicada à medida que crescem o número de associações e as limitações impostas aos recursos. Um exemplo clássico de problema de *timetable* é encontrado em instituições de ensino, quanto ao *agendamento de aulas*. Para esta variante, existe uma grande quantidade de trabalhos desenvolvidos [1, 4, 5] e até mesmo competições, como a *International Timetabling Competition*[1], que têm como objetivo estimular o crescimento e a popularização de soluções automatizadas (*software*) para o problema.

Contudo, apesar desse esforço, ainda existe uma grande distância entre o que é desenvolvido e o que, de fato, é utilizado na prática. Cada instituição possui peculiaridades que, em certos casos, impedem, ou, no melhor dos casos, dificultam a utilização de *softwares* não específicos. Como raramente algum esforço é estabelecido para o desenvolvimento de soluções específicas esse distanciamento tende a aumentar.

É justamente sobre esse ponto que se direciona o foco deste trabalho, ou seja, o objetivo é, a partir dos diversos métodos de resolução e trabalhos relacionados aos problemas de programação de horários em instituições de ensino, possibilitar o desenvolvimento

[1] `http://www.cs.qub.ac.uk/itc2007/index.htm`

de um *software* específico que atenda aos requisitos encontrados na Universidade Federal de Goiás – UFG.

O problema geral de programação de horários em instituições de ensino será referenciado ao longo do texto como *Problema do Agendamento Semanal de Aulas*, ou abreviadamente PASA. Neste caso, as aulas (definidas pela associação entre um professor, uma disciplina e um horário) são os *eventos* a serem agendados; e as cargas horárias dos professores e os horários semanais são vistos como *recursos* disponíveis.

A resolução de um PASA tem como resultado soluções que podem ser descritas através de quadros de horários que relacionem cada professor a um conjunto de horários, nos quais serão ministradas as disciplinas de responsabilidade do mesmo. A Tabela 1.1 exemplifica uma solução para um problema fictício, no qual existem cinco disciplinas a serem ministradas e três professores disponíveis para ministrá-las.

Horários	**Dias da semana**				
	2^a	**3^a**	**4^a**	**5^a**	**6^a**
08:00-09:40		Algoritmos ***prof. A***		Algoritmos ***prof. A***	
10:00-11:40					
14:00-15:40	Cálculo 1 ***prof. B***	Lógica ***prof. B***	Cálculo 1 ***prof. B***	Lógica ***prof. B***	Inglês ***prof. C***
16:00-17:40	Algoritmos ***prof. A***	Inglês ***prof. C***	Física ***prof. C***		Física ***prof. C***

Tabela 1.1: *Agendamento semanal para cinco disciplinas.*

Em um problema real de tamanho razoável o número de professores e disciplinas envolvidas é bem maior, além disso, diversas outras situações contribuem para tornar a programação de horários uma tarefa tediosa e sujeita a erros. Professores podem se ausentar (férias, problemas de saúde, pós-graduação), alunos que por algum motivo estejam fora do fluxo curricular (reprovações, não adequação ao fluxo sugerido) podem necessitar de situações especiais, e várias outras situações podem ser consideradas, o que colabora para o aumento da dificuldade de resolução do problema.

A aplicação de métodos automatizados (*software*), neste caso, pode trazer consigo grandes vantagens, pois a satisfação dos interesses dos docentes, da instituição e dos alunos, pode ser atingida de forma mais rápida e exigindo menor esforço humano. Considerando as proporções do problema enfrentado na UFG: cerca de 124 cursos de graduação e aproximadamente 17.000 alunos; não é difícil vislumbrar a importância que esse tipo de iniciativa teria na universidade.

O PASA é um problema bastante abordado na literatura, os primeiros trabalhos datam da década de 70 [17], e os métodos de resolução utilizados desde então vêm acompanhando a evolução da área de *Pesquisa Operacional* e a modernização dos modelos de ensino.

No Capítulo 2 inicia-se uma revisão de alguns métodos recorrentemente utilizados na resolução do PASA, além de uma breve descrição das principais definições relacionadas à *Otimização Combinatória* que são utilizadas ao longo do texto. Métodos exatos como *Branch-and-Bound* e *Branch-and-Cut* são descritos na seção 2.4, seguidos pela descrição de algumas meta-heurísticas bem conhecidas como *Busca Tabu* e *Algoritmos Genéticos* na seção 2.5.

Já no Capítulo 3, o PASA é dividido em categorias de acordo com as características da instituição de ensino em questão. Problemas como a *Programação de Horários em Escolas* (seção 3.1) e a *Programação de Horários em Universidades* (seção 3.2) são descritos de forma detalhada, constituindo a base para a resolução do problema enfrentado na UFG.

No Capítulo 4, o problema é visto por uma visão mais especializada e o problema enfrentado na UFG é descrito através de problemas menores , de acordo com os quais uma abordagem de resolução é proposta. Em sequência, no Capítulo 5, a estrutura acadêmica da UFG é descrita, juntamente com os detalhes pertinentes ao INF e a especificação de uma instância de teste e de alguns resultados. Considerações finais quanto ao algoritmo proposto e trabalhos futuros são descritos no Capítulo 6.

CAPÍTULO 2

Técnicas

Diversas técnicas foram desenvolvidas, estudadas e adaptadas ao longo dos anos, visando a solução eficiente de problemas de programação de horários. As primeiras, baseavam-se principalmente em heurísticas desenvolvidas para a utilização na resolução manual, porém, com o aumento da importância de tais problemas, técnicas mais aprimoradas começaram a ser utilizadas.

Duas categorias podem ser identificadas quanto à classificação de tais técnicas. A primeira, composta por *métodos exatos*, tem como objetivo principal encontrar as melhores soluções para o problema. Enquanto que a segunda categoria, composta por métodos *heurísticos* e *meta-heurísticos*, tem como objetivo a obtenção de boas soluções, não necessariamente as melhores.

O objetivo deste capítulo é oferecer uma revisão das principais técnicas de resolução de problemas de programação de horários referenciadas na literatura, citando exemplos pertencentes a ambas as categorias descritas anteriormente, e formando a base para os conceitos utilizados nos capítulos subsequentes.

Algumas definições importantes relacionadas à *Otimização Combinatória* são descritas na Seção 2.1, seguidas pela descrição de alguns métodos exatos na Seção 2.4 e algumas *meta-heurísticas* na Seção 2.5. Enquanto que a Seção 2.6, considera algumas formas de se combinar *métodos exatos* e *meta-heurísticas* na resolução de um mesmo problema.

2.1 Definições

Definição 2.1 (Instância) *Uma instância em um problema de otimização é um par* (S, f)*, onde* S *é um espaço de soluções e* $f : S \to \mathbb{R}$ *é uma função de custo que associa a cada solução pertencente a* S *um valor real.* □

Definição 2.2 (Solução ótima) *Uma solução ótima* x^**, para um problema de otimização na forma de minimização, satisfaz:* $f(x^*) \leq f(x)$, $\forall x \in S$. □

Dadas as definições 2.1 e 2.2 os problemas de otimização são divididos em duas categorias: aqueles representados com variáveis *contínuas*, definição 2.3, e aqueles representados com variáveis *discretas*, definição 2.4.

Definição 2.3 (Otimização contínua) *Na resolução de um problema de otimização contínua o objetivo é encontrar uma solução ótima $x^* \in S$, sendo S um conjunto incontável.* □

Nos problemas de otimização *contínua*, em geral, as soluções são representadas por um conjunto de números reais, desta forma, a resolução deste tipo de problema tem como objetivo encontrar um conjunto de números reais que representem uma *solução ótima* para o problema.

Definição 2.4 (Otimização combinatória) *Na resolução de um problema de otimização combinatória, o objetivo é encontrar uma solução ótima $x^* \in S$, sendo S um conjunto contavelmente infinito.* □

Nos problemas de otimização *discreta*, mais comumente chamados: problemas de *otimização combinatória*, as soluções são representadas por um conjunto de números inteiros e, desta forma, o objetivo é encontrar valores em um conjunto finito, ou contavelmente infinito, que representem um ótimo global.

Estes dois tipos de problemas possuem estratégias de solução, em grande parte, divergentes. Porém, existe um certo limiar no qual as características se confundem. Nesta seção o foco está direcionado a técnicas de otimização combinatória, contudo, alguns conceitos importantes relacionados à otimização contínua são necessários e também são descritos.

A seguir, outras definições também utilizadas ao longo do texto, baseadas principalmente em [37] e [40].

Definição 2.5 (Função de vizinhança) *Uma função de vizinhança $N : S \rightarrow 2^S$, especifica para cada solução $x \in S$, um conjunto $N(x) \subseteq S$ chamado vizinhança de x. A cardinalidade de $N(x)$ é chamada de tamanho da vizinhança de x, e uma solução x' é dita vizinha de x se $x' \in N(x)$.* □

Definição 2.6 (Ótimo local) *Um Ótimo Local em relação a N é uma solução $\hat{x} \in S$ tal que: $f(\hat{x}) \leq f(x),\ \forall x \in N(\hat{x})$.* □

Definição 2.7 (Grafo de vizinhança) *O grafo de vizinhança de uma instância (S, f), associado a uma função de vizinhança N, é um grafo direcionado $G = (V, A)$. O conjunto de vértices V é dado pelo conjunto S de soluções, e o conjunto de arcos A é definido de tal forma que $(i, j) \in A$ se e somente se $x_j \in N(x_i)$* □

Definição 2.8 (Limite inferior) *Um limite inferior para um problema, ou Lower Bound, é um valor menor ou igual ao custo associado a função objetivo de uma solução ótima para o problema. Ou seja, considerando o espaço de solução S, define-se: $lb(S) \leq f(x^*)$, $x^* \in S$.*

Definição 2.9 (Limite superior) *Um limite superior para um problema, ou Upper Bound, é um valor maior ou igual ao custo associado a função objetivo de uma solução ótima para o problema. Ou seja, considerando o espaço de solução S, define-se: $ub(S) \geq f(x^*)$, $x^* \in S$.*

Definição 2.10 (Algoritmo guloso) *Um Algoritmo Guloso faz sempre a escolha que parece melhor no momento. Isto é, ele se baseia em escolhas localmente ótimas, na esperança que estas escolhas levem a uma solução globalmente ótima [13].* □

Definição 2.11 (Problema de busca) *A resolução de um Problema de Busca tem como objetivo a obtenção de uma solução viável para o problema, independentemente da qualidade dessa solução.* □

Definição 2.12 (Problema de otimização) *A resolução de um Problema de Otimização têm como objetivo a obtenção de uma solução viável e de boa qualidade para um problema. Dependendo do tipo de abordagem utilizada essa solução pode ser garantidamente um ótimo global, ou não.* □

2.2 Programação Linear

Um *Problema de Programação Linear*, ou abreviadamente PPL, é o problema de minimizar o custo de uma *função objetivo* linear, f, de acordo com soluções $x \in S$, sendo S definido por um conjunto de restrições também lineares.

$$\min f(x), \quad x \in S, \quad S \subseteq \mathbb{R}^n_+. \tag{2-1}$$

O PPL é um problema de otimização *contínua* devido a $S \subseteq \mathbb{R}^n_+$, pois o conjunto $\mathbb{R}^n_+$ não é contável (ver definição 2.3).

2.3 Programação Linear Inteira

Um *Problema de Programação Linear Inteira*, ou abreviadamente PPLI, é definido de forma semelhante ao PPL, exceto pela restrição de integralidade $S \subseteq \mathbb{Z}^n_+$, assim como descrito pela formulação 2-2. Pelo mesmo motivo, o PPLI é definido como um problema de otimização *combinatória* (ver definição 2.4).

$$\min f(x), \quad x \in S, \quad S \subseteq \mathbb{Z}^n_+. \tag{2-2}$$

Definição 2.13 *A Relaxação Linear de um PPLI consiste em um PPL obtido através relaxação da restrição de integralidade.*

Desta forma, pode parecer tentador solucionar o PPLI através de uma *relaxação linear* seguida pelo arredondamento das variáveis não-inteiras. Contudo, apesar de útil, essa não é uma solução imediata para os problemas de PPLI, de modo que a solução obtida por esse processo pode até mesmo nem ser factível.

Alguns métodos que podem ser utilizados na resolução do PPLI estão descritos nas seções 2.4 e 2.5, respectivamente *Métodos Exatos* e *Meta-heurísticas*.

2.4 Métodos Exatos

Grande parte dos métodos exatos baseiam-se na enumeração *inteligente* do espaço de soluções. A inteligência neste caso, está relacionada ao fato de que, na realidade, nem todas as soluções são avaliadas, mas existe a garantia de que se uma solução não é avaliada é porque ela não é uma solução ótima.

Esta seção enfatiza métodos baseados em enumeração *inteligente* do tipo *Branch-and-Bound*. Para tal, uma revisão do algoritmo de *Branch-and-Bound* é feita na Seção 2.4.1, seguida pela descrição geral dos métodos de *Planos de Corte*, na Seção 2.4.2 e pela combinação de ambos, que resulta no método *Branch-and-Cut*, descrito na Seção 2.4.3.

2.4.1 *Branch-and-Bound*

Algoritmos do tipo *Branch-and-Bound* seguem a abordagem “Dividir para conquistar”, desta forma, seu funcionamento é baseado em sucessivos particionamentos do espaço de soluções, etapa chamada *Branching*. Como forma de evitar a exploração de todo espaço de soluções, o método faz uso de *limites inferiores* (definição 2.8) associados a soluções para os subproblemas, através das quais o método consegue selecionar as soluções *promissoras* e excluir as *não-promissoras*, etapa chamada *Bound* [40].

Desta forma, uma das etapas de construção de um algoritmo *Branch-and-Bound* é a determinação de um *limite inferior* para o problema. A seguir, tem-se a definição do algoritmo de *Branching* responsável pelo particionamento do espaço de soluções. Nessa etapa, restrições são adicionadas ao problema de forma a dividir o espaço de soluções em subconjuntos, que serão tratados de forma independente em etapas posteriores.

Exemplo, seja x^* uma solução ótima para a *relaxação linear* (definição 2.13) do `PPLI`, e seja y_i o valor não-inteiro da variável x_i de x^*. Existem, neste caso, dois subproblemas, mutuamente exclusivos, que devem ser avaliados para que uma solução inteira ótima seja encontrada, os quais decorrem da adição das restrições 2-3 e 2-4, respectivamente.

$$x_i \leq \lfloor y_i \rfloor \tag{2-3}$$

$$x_i \geq \lceil y_i \rceil \tag{2-4}$$

Cada um desses subproblemas pode ser tão difícil quanto o problema original, o que sugere solucioná-los pelo mesmo processo, a divisão em subproblemas ainda mais restritos. Esta etapa de divisão leva a uma árvore de subproblemas a serem avaliados, cada subproblema é considerado um *nó ativo* na árvore *Branch-and-Bound* [2].

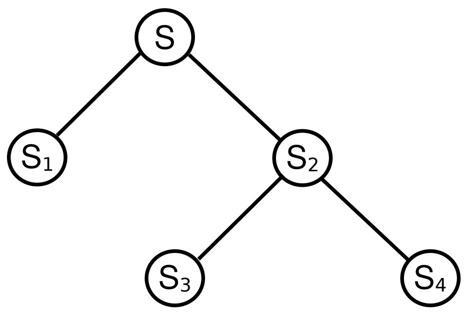

Figura 2.1: *Árvore Branch-and-Bound, de subproblemas.*

No decorrer da execução do algoritmo, alguns subproblemas são solucionados de forma completa e soluções inteiras são obtidas, o valor da melhor solução inteira até então encontrada é considerado como *limite superior* (definição 2.9) para o problema geral, e essa solução é armazenada.

Através da comparação dos *limites inferiores* obtidos para os subproblemas com o *limite superior* é possível decidir se determinado subproblema, precisa, ou não, ser avaliado. Ou seja, se o *limite inferior* para um subproblema S_i for maior que o atual *limite superior* o subproblema S_i não precisa ser avaliado pois certamente não levará a um ótimo global.

Assim que a árvore *Branch-and-Bound* estiver vazia o processo de busca é finalizado e a melhor solução inteira encontrada considerada como um ótimo global. Caso nenhuma solução inteira tenha sido encontrada considera-se que o problema não possui solução inteira factível.

2.4.2 Planos de corte

Considerando um `PPLI` e a sua respectiva *relaxação linear* (definição 2.13), tem-se que a principal ideia por trás dos métodos de planos de corte é a solução do `PPLI` através da solução de uma sequência de *relaxações lineares*.

Primeiramente, soluciona-se a relaxação linear do `PPLI` e uma solução ótima x^* é encontrada. Se x^* é uma solução inteira, então ela também é uma solução ótima para o `PPLI`. Caso contrário, adiciona-se uma restrição ao problema, de tal forma que todas as soluções inteiras para `PPLI` a satisfaçam, mas que x^* não satisfaça, ou seja, a solução x^* é excluída do espaço de soluções. O processo é, então, repetido e o espaço de soluções se torna cada vez mais restrito, até que uma solução inteira ótima, para o problema original, possa ser encontrada [2, 40].

Evidentemente, a eficiência do método depende fortemente da escolha das restrições a serem adicionadas, sendo que o algoritmo de planos de *Cortes de Gomory* [26] foi um dos primeiros para esse problema.

2.4.3 *Branch-and-Cut*

O *Branch-and-Cut* é uma variante do *Branch-and-Bound* que utiliza os métodos de planos de corte na resolução dos subproblemas. Ou seja, na etapa de *Branching*, além da inserção das restrições de particionamento, 2-3 e 2-4, também são inseridas as restrições de corte.

É esperado, a partir dessa combinação, que mais soluções possam ser identificadas como não-promissoras, o que implica na necessidade de uma menor quantidade de iterações até que uma solução ótima possa ser encontrada.

2.5 Meta-heurísticas

Meta-heurísticas são métodos heurísticos independentes de problemas específicos. Uma meta-heurística pode ser utilizada na resolução de problemas pertencentes a diversos contextos, pois oferece o raciocínio, ou abordagem a ser seguida que não se relaciona aos detalhes do problema.

As meta-heurísticas são utilizadas, geralmente, para atacar problemas de otimização de grande porte, para os quais os métodos exatos falham em ser efetivos ou eficientes [39]. Existem duas categorias principais nas quais é possível organizar as meta-heurísticas: *Heurísticas Construtivas* e *Heurísticas de Melhoria*.

As *Heurísticas Construtivas* têm como objetivo criar uma solução viável completa para o problema. Esse processo geralmente é dividido em várias etapas, sendo que em cada uma delas uma solução parcial é estendida até a obtenção de uma solução completa. As *Heurísticas de Melhoria* têm como objetivo encontrar soluções de alta qualidade, e se utilizam dos mais variados mecanismos para atingir esse objetivo.

Meta-heurísticas como *Busca Local* (Seção 2.5.1), *Simulated Annealing* (Seção 2.5.2), *Busca Tabu* (Seção 2.5.3), GRASP (Seção 2.5.4) e *Algoritmos Genéticos* (Seção 2.5.5) pertencem à segunda categoria e incorporam em seu processo *Heurísticas Construtivas*.

2.5.1 Busca Local

A *Busca Local* é uma meta-heurística de melhoria, que em sua forma mais simples tem como objetivo encontrar *ótimos locais*. Partindo de uma *solução inicial* x, é feita uma busca no subconjunto $N(x) \subset S$ que contém as *soluções vizinhas* de x, e a melhor solução, dentre as melhores que x, é escolhida. Esse processo se repete enquanto soluções melhores forem obtidas. Caso nenhuma solução melhor que a atual seja encontrada, tem-se um *ótimo local* e a busca termina. Embora seja possível que o *ótimo local* encontrado também seja um *ótimo global*, isso não é garantido pelo método.

O conceito chave na *Busca Local* é a função de vizinhança. Ela especifica para cada solução quais soluções são, em algum aspecto próximas, e a partir disso direciona a busca. Tal proximidade, geralmente, é medida pela quantidade de variáveis com valores iguais nas soluções em questão, contudo, diferentes métricas podem ser utilizadas. Em cada etapa, um conjunto $N(x)$ é avaliado, e alguma solução $x' \in N(x)$ é escolhida. Esse processo é chamado: movimentação pelo espaço de soluções.

Existem diferentes estratégias de movimentação pelo espaço de soluções e que visam acelerar o processo de busca. A mais conhecida é a utilizada pelo algoritmo de melhoria iterativa[1], que na verdade é o algoritmo básico de *Busca Local*.

Algoritmo 2.1: Melhoria Iterativa

```
s ← alguma solução inicial
repita
  | gerar x' ∈ N(x)
  | se f(x') < f(x) então
  |   | s ← x'
  | fim
até f(x') ≥ f(x) para todo x' ∈ N(x)
```

Em alguns casos, ao se avaliar a vizinhança de uma solução, várias soluções promissoras, ou seja, com melhor custo que a atual são encontradas. Nestes casos alguma

[1] *Iterative Improvement Algorithm*, também conhecido como *Hill Climbing Algorithm*

regra de seleção (*Pivoting Rule*) deve ser utilizada como forma de decidir qual solução escolher para a próxima iteração.

Duas regras de seleção bem conhecidas são: *Primeira melhoria* (*First Improvement*) e *Maior melhoria* (*best improvement*). Na primeira abordagem, assim que uma solução de menor custo que a atual é encontrada a iteração termina e o restante da vizinhança nem mesmo precisa ser avaliado. A segunda abordagem, pelo contrário, avalia todas as soluções vizinhas e escolhe a solução que traga maior melhoria, só então dá continuidade à busca.

É a partir desse processo de movimentação no espaço de soluções que a *Busca Local* pode encontrar boas soluções. No entanto, em alguns casos, ele não é forte o suficiente ao ponto de garantir boas soluções, pois a escolha de soluções melhores a cada passo pode levar a ótimos locais de qualidade muito baixa. A figura 2.2 descreve essa situação: a solução atual *A* é a melhor em sua vizinhança, e desta forma a *Busca Local* é finalizada retornando *A*, no entanto, existe uma solução *B* de qualidade bem superior (supondo um problema de minimização) que não será explorada.

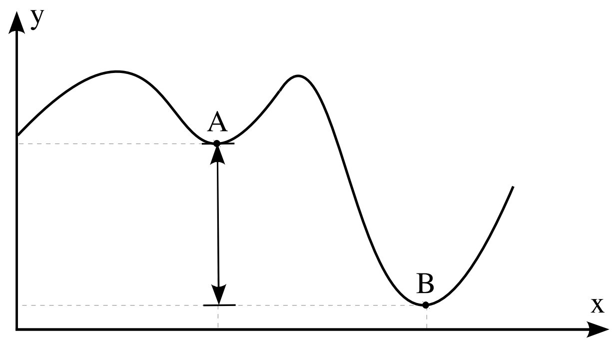

Figura 2.2: *Busca local interrompida na solução A, um ótimo local de baixa qualidade. O Eixo y indica o valor da função objetivo e o eixo x o espaço de soluções.*

Com o objetivo de evitar ótimos locais de baixa qualidade e consequentemente aumentar a qualidade das soluções encontradas, diversas meta-heurísticas baseadas em *Busca Local* foram desenvolvidas. As seções 2.5.2, 2.5.3, 2.5.4 e 2.5.5 se concentram na descrição de algumas delas.

2.5.2 *Simulated Annealing*

Simulated Annealing é uma meta-heurística que utiliza probabilidades no momento da escolha da solução vizinha para tentar escapar de ótimos locais de baixa qualidade. Antes de mais detalhes, o contexto em que essa técnica se aplica é descrito, como forma de deixar claro sua origem e o porquê de seu sucesso.

Considere o caso de uma *Heurística de Melhoria*, onde a solução vizinha de x seja selecionada uniformemente e de forma aleatória, da vizinhança $N(x)$. Desta forma, cada solução vizinha x' tem a probabilidade $|N(x)|^{-1}$ de ser escolhida, sendo que a primeira com custo melhor do que x assumirá como solução atual (*first-improvement*).

Se for permitido, durante esse processo de escolha, que soluções não-melhores sejam aceitas, ou seja, a primeira sorteada toma lugar como solução atual, o método provavelmente conseguirá extrapolar a busca para além do primeiro *ótimo local* encontrado, no entanto, a convergência em direção a bons *ótimos locais* ficará comprometida (*Busca Aleatória – Random Search*).

Como forma de diminuir os problemas de uma *Busca Aleatória* e aproveitar os benefícios da escolha de soluções não melhores, diversos algoritmos foram desenvolvidos. O algoritmo *Limite de Aceitação* (*Threshold Accepting*), por exemplo, permite que soluções não-melhores sejam aceitas, desde que, o fator de deterioração da qualidade da solução não ultrapasse um determinado limite t, ou seja, $f(x') - f(x) < t$.

Uma boa opção, em algoritmos como esse, é relacionar o valor de t à iteração na qual ele é inspecionado, de forma a ser possível controlar a diversidade das soluções no decorrer do processo. Avalia-se um grande espaço de soluções no início (alta deterioração permitida, alto valor de t) e se concentrar a pequenas e promissoras áreas do espaço de soluções no final (baixa deterioração permitida, baixo valor de t).

O *Simulated Annealing* é uma variação do algoritmo de *Limite de Aceitação*, neste caso, porém, o limite t é uma variável aleatória. Uma descrição mais detalhada pode ser vista no algoritmo 2.2.

Algoritmo 2.2: Simulated Annealing

```
x ← alguma solução inicial;
k ← 1 ;
repita
    gerar x' ∈ N(x);
    se f(x') ≤ f(x) então
        x ← x';
    senão
        se e^((f(x)−f(x'))/c_k) > random[0,1) então
            x ← x';
        fim
    fim
    k ← k+1;
até critério de parada;
```

A partir de uma solução inicial o grafo de vizinhança é percorrido, e a cada iteração uma solução x' é escolhida do conjunto $N(x)$. Caso x' seja melhor que x:

x' substitui x, assim como descrito para a *Busca Local*, caso contrário: x' substitui x com probabilidade $e^{(f(x)-f(x'))/c_k}$, onde $c_1, c_2, \ldots, c_k, \ldots$ é uma sequência não crescente de valores. Como implicação do valor c_k a aceitação de soluções piores diminui à medida que se passam as iterações, de modo que a solução tende a estabilizar quando o valor c_k chega próximo a zero. Exemplos de aplicação desta meta-heurística podem ser encontrados em [16, 31].

2.5.3 Busca Tabu

Assim como *Simulated Annealing*, *Busca Tabu* é uma meta-heurística baseada em *Busca Local*. Contudo, diferentemente da *Simulated Annealing*, *Busca Tabu* utiliza uma regra de seleção baseada em *best improvement*, em que a melhor solução *na vizinhança*, $x \in N(x)$, é escolhida para substituir a atual, independentemente dessa solução ser, ou não, melhor que a atual.

Essa abordagem permite à busca ir além de alguns ótimos locais, porém, em alguns casos, uma busca cíclica pode acontecer. Exemplificando, escolhe-se a melhor solução dentre as vizinhas de x e encontra-se $x' \in N(x)$, se em algum passo posterior a melhor solução vizinha for alguma das soluções já visitadas, neste caso x ou x', a busca entrará em um ciclo. Para evitar esse comportamento indesejável, a *Busca Tabu* insere o conceito de soluções "tabu", ou seja, soluções visitadas recentemente ficam proibidas de serem visitadas por uma certa quantidade de iterações, evitando, assim, caminhos cíclicos no grafo de vizinhança.

A partir de uma solução inicial x e uma lista tabu T, a melhor solução x' não-tabu (solução permitida) na vizinhança de x é escolhida para a busca na próxima iteração. Essa solução também é inserida na lista tabu, de forma que, por uma certa quantidade de iterações, retornar até ela não seja permitido. É importante ressaltar que a etapa de atualização da lista tabu envolve tanto a adição de novas soluções, quanto a manutenção do tamanho da lista, visto que uma lista pequena pode não evitar ciclos e uma lista muito grande pode restringir de forma exagerada a busca.

Algoritmo 2.3: Tabu Search

$T \leftarrow []$;
$x \leftarrow$ *alguma solução inicial*;
$\hat{x} \leftarrow x$;
repita
  encontrar a melhor $x' \in N(x) \setminus T$;
  se $f(x') < f(\hat{x})$ **então**
    $\hat{x} \leftarrow x'$;
  fim
  $x \leftarrow x'$;
  atualizar lista tabu T;
até *critério de parada*;

Uma estrutura de dados bastante utilizada como lista tabu é uma fila encadeada circular de tamanho máximo pré-fixado. Caso o tamanho máximo não tenha sido alcançado, adiciona-se os movimentos ao final da estrutura, caso contrário, remove-se o primeiro movimento, e então o novo movimento é inserido. Uma explicação mais detalhada sobre *Busca Tabu* pode ser encontrada em [24, 25] e exemplos de aplicação a problemas de programação de horários em [28, 29].

2.5.4 GRASP

Uma estratégia para a obtenção de ótimos locais de melhor qualidade é processar o algoritmo múltiplas vezes. Uma extensão trivial do algoritmo de *Melhoria Iterativa* baseado nessa ideia é o algoritmo de *Múltiplas Inicializações Aleatórias* (*Random Restart*), no qual o algoritmo básico é executado várias vezes, partindo, em cada uma delas, de uma solução inicial aleatória e diferente das demais.

Essa abordagem aumenta o poder da *Busca Local*, porém, ela encontra dificuldades em alcançar ótimos locais de boa qualidade, isso em grande parte, se deve ao fato de o algoritmo de *Melhoria Iterativa* ser fortemente dependente da qualidade da solução inicial. Sob o objetivo de melhorar a qualidade das soluções encontradas, através da geração de melhores soluções iniciais, tem-se o GRASP - *Greedy Randomized Adaptive Search Procedure* [19].

O GRASP é uma meta-heurística de múltiplas inicializações (*Multistart*), sendo que, a cada iteração do algoritmo, uma solução inicial gulosa e aleatória é construída e sobre ela efetuada uma *Busca Local*. As soluções iniciais são geradas de forma gulosa e aleatória, visando aumentar a variedade enquanto a *Busca Local* é utilizada na melhoria. O processo se repete, encontrando diferentes *ótimos locais* até que algum critério de parada seja satisfeito.

Algoritmo 2.4: GRASP

repita
  $x \leftarrow Greedy\_Randomized\_Construction()$;
  $x \leftarrow Busca\_Local(x)$;
  atualizar a melhor solução;
até *critério de parada*;

2.5.5 Algoritmos Genéticos

Os algoritmos genéticos se baseiam na melhoria de soluções através de mecanismos inspirados em genética evolucionária e seleção natural. Assim como no caso real, existem recombinações e mutações que são efetuadas em soluções para o problema, que representam os indivíduos da população. Os dois mecanismos principais neste tipo de abordagem são a *Seleção* e a *Variação*.

A *Seleção* funciona seguindo um processo chamado sobrevivência do mais apto, no qual os indivíduos mais adaptados ao meio em que vivem têm mais chances de sobreviver e deixar descendentes. Como resultado a população como um todo move-se em direção a uma maior aptidão em relação ao *habitat*.

A *Variação* é o processo, a partir do qual novos indivíduos podem ser inseridos na população, ela pode ocorrer devido a recombinação genética oriunda do processo reprodutivo ou através de mutações de origem aleatória e imprevisível. Enquanto a reprodução tem o efeito de misturar o material genético, apenas as mutações são capazes de criar novas informações genéticas.

Os algoritmos genéticos se baseiam na evolução natural. Neste caso, porém, cada indivíduo representa uma solução para o problema e a aptidão do indivíduo é dada pelo valor da *função objetivo*. O algoritmo 2.5 inicia com a geração de n soluções iniciais, esse conjunto forma a população inicial X_0. O algoritmo gera repetidamente uma população de tamanho n, sendo que cada população X_t é obtida a partir da população anterior X_{t-1}, através da seleção dos melhores indivíduos. A partir de cada solução X_t obtida, novos indivíduos são criados, através de recombinação e/ou mutação, e adicionados à população.

Algoritmo 2.5: Genetic Local Search

$X_0 \leftarrow$ *conjunto de n soluções*;
Alterar cada $x \in X_0$ *por melhoria iterativa*;
repita
  Selecionar $X_t \subseteq X_{t-1}$;
  Aumentar X_t *acrescentando os descendentes*;
  Alterar cada $x \in X_t$ *por melhoria iterativa*;
até *critério de parada*;

O algoritmo 2.5 avalia a cada iteração um conjunto de soluções, de modo que no final do processamento tem-se um conjunto de soluções de qualidade semelhante.

2.6 Métodos híbridos

Devido às características inerentes às meta-heurísticas e aos métodos-exatos, a decisão de escolha entre um ou outro deve sempre ser resultado de ponderação e priorização em relação a tempo e garantia de otimalidade. As meta-heurísticas podem apresentar bons resultados em um tempo relativamente pequeno, porém, a solução obtida não traz garantia de otimalidade. Os métodos exatos garantem a otimalidade da solução encontrada, contudo, pode ser necessário muito tempo para que essa garantia seja obtida. Tendo isso em vista é que a combinação de ambos se tornou uma opção interessante.

Ao decorrer desta seção são descritas e categorizadas algumas formas de combinação encontradas na literatura, as quais constituem uma terceira categoria de métodos de solução para problemas de otimização combinatória, os chamados métodos híbridos. A figura 2.3 descreve a classificação dos tipos de combinações descritas em [41], que identificou duas categorias principais, as *Combinações Colaborativas* e as *Integrativas*, descrita a seguir.

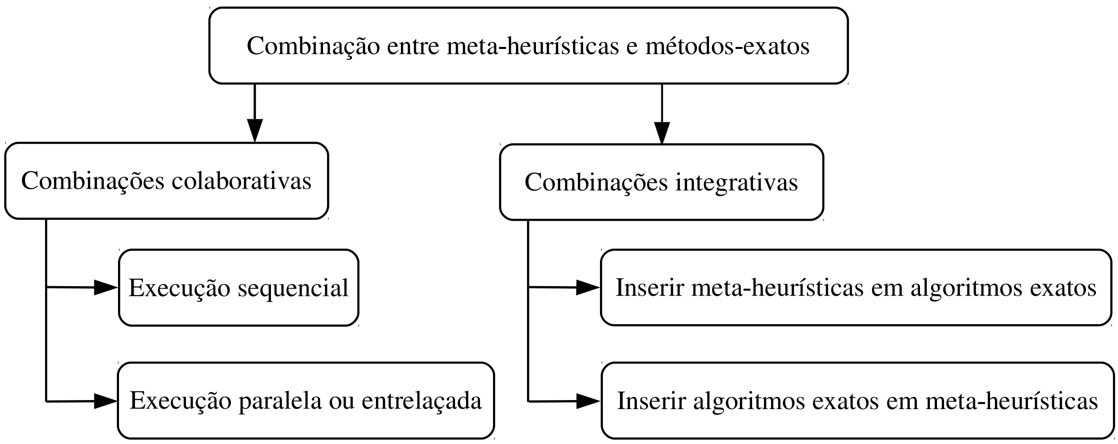

Figura 2.3: *Classificação de combinações entre meta-heurísticas e algoritmos exatos.*

2.6.1 Combinações colaborativas

Neste esquema de execução, os algoritmos trocam informações mas não fazem parte um do outro, ou seja, são executados sequencialmente ou de forma entrelaçada ou paralela.

Execução sequencial

Neste tipo de colaboração, o método-exato é executado antes da meta-heurística ou o contrário, e nesses casos, é difícil dizer se a primeira técnica está trabalhando como inicialização para a segunda ou se a segunda está trabalhando como um pós-processamento dos resultados da primeira.

Um exemplo interessante deste tipo de abordagem pode ser encontrado em [18], onde uma relaxação do problema original é resolvida de forma ótima e então essa solução é ajustada para atuar como solução inicial para a meta-heurística subsequente. Em alguns casos, a solução obtida a partir da relaxação é facilmente ajustável, tornando esse esquema de resolução interessante.

Execução entrelaçada ou paralela

Diferentemente da forma sequencial, os algoritmos exatos e heurísticos podem ser executados de forma paralela. Apesar desse tipo de combinação ser menos frequente, existem alguns casos de sucesso, dentre os quais, pode ser citado, como exemplo, o caso dos *Times Assíncronos* ou *A-Teams*, proposto em [48, 49].

Um *A-Team* é uma arquitetura para solução de problemas, que consiste de um conjunto de agentes e memórias, sendo que cada um dos agentes é um algoritmo de otimização que trabalhará sobre o problema. A ideia básica dos *A-Teams* é que os agentes trabalhem de forma assíncrona e autônoma (paralelismo) no conjunto de memórias compartilhadas, modificando, adicionando soluções, removendo soluções ou alterando-as.

2.6.2 Combinações integrativas

Nas combinações integrativas, geralmente, uma técnica é inserida em outra, atuando como algo semelhante a um componente. Assim, define-se o algoritmo principal, que pode ser um algoritmo exato ou uma meta-heurística, e ao menos um algoritmo, seja exato ou não, trabalhando como componente.

Inserindo meta-heurísticas em algoritmos exatos

Existem diversas formas conhecidas de se integrar meta-heurísticas em algoritmos exatos. Uma lista, não-exaustiva, de tais abordagens, está descrita a seguir:

- Utilização de meta-heurísticas para a obtenção de soluções iniciais e limites em abordagens baseadas em *Branch-and-Bound* [52];

- Meta-heurísticas para geração de cortes. Em algoritmos *Branch-and-Cut* a separação dinâmica de planos de corte, respectivamente, é, algumas vezes, feito por heurísticas, visando acelerar o processo de otimização [27];
- *Branch-and-Bound* inspirado em *Busca Local*. Aplica uma abordagem inspirada em *Busca Local*, ou seja, avaliação da vizinhança de soluções, como estratégia de seleção dos nós em um algoritmo *Branch-and-Bound* tradicional [22].

Inserindo algoritmos exatos em meta-heurísticas

- Resolvendo, de forma exata, problemas relaxados. Além de explorar as relaxações para obter soluções iniciais promissoras, essa abordagem pode ser de grande benefício à heurística, servindo como guia a etapas como busca pela vizinhança, recombinação, mutação [8, 42].
- Busca exata em grandes vizinhanças. Se as vizinhanças forem escolhidas de forma apropriada, o melhor vizinho pode ser buscado de forma eficiente, mesmo em vizinhanças muito grandes. Técnicas deste tipo são conhecidas como *Very Large-Scale Neighborhood Search* (VLSN) [3, 10].

2.7 Considerações finais

Neste capítulo alguns métodos de utilidade na resolução de um `PPLI` foram descritos: meta-heurísticas como exemplo de métodos não-exatos e algoritmos exatos como *Branch-and-Bound* e *Branch-and-Cut*.

Vários detalhes importantes para a eficácia dos métodos descritos foram suprimidos pois fogem ao escopo deste texto. Maiores detalhes podem ser obtidos em referências como [2, 40, 49].

CAPÍTULO 3

Problemas de programação de horários

O termo *programação de horários* se refere ao conjunto de atividades relacionadas à criação de quadros de horários (*timetable*) que, independentemente do contexto de aplicação, têm como foco principal relacionar atividades a horários e locais, de modo a tornar plausível, tanto quanto possível, a participação das pessoas nas atividades em questão [51]. Assim, o objetivo geral é criar quadros de horários que não contenham duas ou mais atividades simultâneas relacionadas a algum participante das atividades.

O *Problema de Agendamento Semanal de Aulas*, abreviadamente `PASA`, representa, neste trabalho, os problemas de programação de horários em instituições de ensino e pode ser definido como a necessidade de se relacionar disciplinas, horários, professores, estudantes e salas, de modo que nenhum dos envolvidos esteja associado a mais de uma atividade em horários simultâneos [17].

Além de evitar tais conflitos, diversas outras restrições podem ser consideradas, dependendo das necessidades da instituição envolvida. Em [45] foram identificadas três categorias de requisitos que implicam em restrições à solução do problema. Essas categorias estão descritas a seguir, juntamente com alguns exemplos.

1. **Requisitos organizacionais**, que se referem à instituição de ensino: alocação de salas, laboratórios, anfiteatros, atendimento à legislação, carga horária semanal de cada professor, carga horária total das disciplinas;
2. **Requisitos pedagógicos**, que visam o bom aproveitamento das aulas: duração e distribuição das aulas ao longo da semana, intervalos que evitem aulas da mesma disciplina em dias consecutivos, ou em períodos consecutivos;
3. **Requisitos pessoais**, decorrentes das necessidades ou preferências dos membros do corpo docente: preferências de horários (matutino, vespertino, noturno), preferências por determinados dias da semana, localização das aulas, preferência por dois horários consecutivos para uma mesma disciplina.

Diferentes tipos de requisitos impõem diferentes tipos de restrições ao problema. Um quadro de horários que viole alguma restrição quanto a conflitos de horários, por exemplo, não é viável (ou factível), pois não consiste em uma solução válida para o problema e não atende às necessidades dos envolvidos. Por outro lado, uma solução que atribua a determinado professor o trabalho no período matutino, mesmo que o professor prefira não trabalhar em tal horário, continua sendo uma solução viável.

Desta forma, de acordo com a importância e o papel das restrições, elas podem ser divididas em duas categorias, segundo a capacidade que possuem de inviabilizar, ou não, uma solução [34]:

1. **Restrições fortes**, que obrigatoriamente devem ser satisfeitas: Essa definição é baseada no fato de quadros de horários somente serem considerados viáveis, caso respeitem todas as restrições deste tipo. Um exemplo comum de restrição forte é a necessidade de inexistência de conflitos entre os horários associados a determinado professor.
2. **Restrições fracas**, que não são obrigatórias, mas desejáveis: Elas não interferem na factibilidade das soluções, contudo, são úteis como indicador da qualidade das mesmas. Um exemplo de restrição fraca é a quantidade de dias por semana em que são divididas as aulas de uma disciplina. É interessante, como requisito pedagógico, que *aulas* (definição 3.1) de uma disciplina não sejam ministradas em dias consecutivos. Contudo, uma solução que não possua essa característica continua sendo viável, apesar de apresentar uma baixa qualidade, segundo esse critério.

Definição 3.1 *Uma Aula, no contexto do* PASA, *é definida como uma associação entre uma disciplina, um professor e um horário.* □

Esses dois tipos de restrições são de fundamental importância quando métodos de otimização são utilizados para a resolução do PASA. As *restrições fortes* são responsáveis por restringir o conjunto de soluções factíveis, limitando, assim, a quantidade de soluções a serem avaliadas. Já as *restrições fracas*, são utilizadas para quantificar a qualidade das soluções e seguem critérios de avaliação mais específicos a cada instância do problema.

Em contraposição à importância que possuem em processos de otimização, as restrições consideradas em cada instância do problema também contribuem para a complexidade de resolução do mesmo. Desta forma, a escolha das restrições se torna uma etapa importante da criação do método de resolução do problema.

são, em sua maioria, pertencentes à classe de problemas $\mathcal{NP}$-Completo, ou $\mathcal{NP}$-Difícil (problemas de otimização), um cuidado adicional é necessário na escolha das restrições e resolução do problema, com o objetivo de favorecer, o quanto possível, o tempo de execução do método a ser utilizado.

É devido a escolha de diferentes restrições que diferentes tipos de problemas de programação de horários em instituições de ensino foram propostos na literatura. Segundo essas variações, algumas sub-categorias foram definidas e, seguindo a notação utilizada em [46] e [45], algumas delas estão descritas nas seções 3.1 (*Programação de Horários em Escolas*), 3.2 (*Programação de Horários em Universidades*) e 3.3 (*Programação de Horários de Exames*).

3.1 Programação de Horários em Escolas

O *Problema de Programação de Horários em Escolas* (*School Timetabling*), abreviadamente PPHE, também conhecido como modelo *Turma/Professor* (definição 3.2), é uma situação tipicamente encontrada no ensino brasileiro de primeiro e segundo graus, onde, geralmente, cada aluno está associado a apenas uma turma e cada turma possui seus horários e salas de aula previamente fixados.

Definição 3.2 *Uma Turma, no contexto do* PPHE*, é definida por um conjunto de alunos matriculados ao mesmo conjunto de disciplinas e que, geralmente, possuem o mesmo nível de escolaridade.* □

O problema consiste em associar cada disciplina a algum horário, de modo que o professor responsável não esteja associado a mais de uma *turma* por horário e que as *turmas* não estejam associadas a mais de uma *aula* por horário.

Formalmente, tem-se a necessidade de associação entre um conjunto de t turmas, $T = \{1,\ldots,t\}$, e um conjunto de p professores, $P = \{1,\ldots,p\}$; a um conjunto de h horários, $H = \{1,\ldots,h\}$, distribuídos ao longo de uma semana.

A quantidade de aulas que deve ser associada entre *turmas* e professores deve respeitar a um certo valor, previamente definido. Em [15], esses valores foram armazenados em uma matriz $R_{p\times t}$, sendo p a quantidade de professores e t a quantidade de turmas. Cada entrada r_{ij}, na matriz $R_{p\times t}$, indica a quantidade de aulas que devem ser ministradas pelo professor i à turma j, com $i \in P$ e $j \in T$.

Mais detalhes sobre os PPHE são apresentados nas próximas seções. A seção 3.1.1 descreve uma versão simplificada do problema, que pode ser solucionada em tempo polinomial, enquanto que as seções 3.1.2 e 3.1.3 descrevem uma versão completa para o problema, modelado como problema de busca e problema de otimização, respectivamente.

3.1.1 Versão simplificada do problema

O problema aqui descrito é simplificado, visto que não inclui restrições relacionadas à indisponibilidade de professores. Para esta versão do problema, existem algoritmos de complexidade de tempo polinomial no tamanho da entrada, como os propostos em [15, 17].

Uma modelagem de *Programação Linear Inteira* (PLI) para este problema, pode ser elaborada a partir de uma matriz binária tridimensional $X_{p \times t \times h}$, com valores definidos pela equação 3-1, assim como especificado pela restrição 3-5:

$$x_{ijk} = \begin{cases} 1 & \text{se existe aula para a turma } j\text{, com o professor } i\text{, no horário } k, \\ 0 & \text{caso contrário.} \end{cases} \tag{3-1}$$

O ponto importante, é garantir o cumprimento das restrições descritas pela matriz $R_{p \times t}$, ou seja, cada professor $i \in P$ deve ministrar necessariamente r_{ij} aulas à turma $j \in T$. A partir desta configuração, o problema de busca pode ser formulado, como:

$$\begin{aligned}
\text{Encontrar} \quad & x_{ijk}, & & i \in P,\ j \in T,\ k \in H; \\
\text{sujeito a:} \quad & \sum_{k=1}^{h} x_{ijk} = r_{ij}, & & i \in P,\ j \in T; & (3\text{-}2) \\
& \sum_{i=1}^{p} x_{ijk} \leq 1, & & j \in T,\ k \in H; & (3\text{-}3) \\
& \sum_{j=1}^{t} x_{ijk} \leq 1, & & i \in P,\ k \in H; & (3\text{-}4) \\
& x_{ijk} \in \{0,1\}, & & i \in P,\ j \in T,\ k \in H. & (3\text{-}5)
\end{aligned}$$

A restrição 3-2 especifica que a quantidade de horários associados entre um professor i e uma turma j deve ser igual ao valor r_{ij} estabelecido pela matriz de restrições $R_{p \times t}$. Já a restrição 3-3 restringe a no máximo 1 a quantidade de aulas associadas a cada turma j, em cada horário k. De forma semelhante, a restrição 3-4 restringe a no máximo 1 a quantidade de aulas associadas a cada professor i, em cada horário k.

Para esta versão do problema sempre existe solução viável (provado em [17]), a menos que seja exigido de algum professor ministrar mais aulas do que a quantidade h de horários disponíveis, ou caso alguma turma esteja associada a mais professores que a quantidade h.

Para excluir tal situação, algumas restrições podem ser adicionadas ao problema, a restrição 3-6, limita a uma quantidade h o número de turmas associadas a cada professor $i \in P$ e a restrição 3-7, limita a uma quantidade h o número de professores associados com cada turma $j \in T$.

$$\sum_{j=1}^{t} r_{ij} \leq h, \quad i \in P; \tag{3-6}$$

$$\sum_{i=1}^{p} r_{ij} \leq h, \quad j \in T. \tag{3-7}$$

3.1.2 Problema de busca

Ao considerar as restrições relacionadas à indisponibilidade de professores, o problema se torna mais próximo de situações encontradas nas instituições de ensino. Na formulação descrita em [32] duas matrizes binárias adicionais, $IP_{p\times h}$ e $IT_{t\times h}$, são utilizadas.

$IP_{p\times h}$ é a matriz de indisponibilidade dos professores, com valores definidos pela equação 3-8, enquanto $IT_{t\times h}$, indica os horários livre nos quais determinada turma pode ser associada a uma aula, cada entrada nessa matriz é definida pela equação 3-9.

$$IP_{ik} = \begin{cases} 1 & \text{se o professor } i \text{ está disponível no horário } k, \\ 0 & \text{caso contrário.} \end{cases} \tag{3-8}$$

$$IT_{jk} = \begin{cases} 1 & \text{se a turma } j \text{ está disponível no horário } k, \\ 0 & \text{caso contrário.} \end{cases} \tag{3-9}$$

A nova formulação introduz as restrições de disponibilidade através da restrição 3-10, que limita os horários das aulas de acordo com a disponibilidade da turma indicada por IT_{jk}, e da restrição 3-11, que limita os horários das aulas de acordo com a disponibilidade do professor, indicada por IP_{ik}.

$$\begin{array}{lll}
\text{Encontrar} & x_{ijk}, & i \in T,\ j \in P,\ k \in H; \\
\text{sujeito a:} & \displaystyle\sum_{k=1}^{h} x_{ijk} = r_{ij}, & i \in P,\ j \in T; \\
& \displaystyle\sum_{i=1}^{p} x_{ijk} \leq IT_{jk}, & j \in T,\ k \in H; \qquad (3\text{-}10) \\
& \displaystyle\sum_{j=1}^{t} x_{ijk} \leq IP_{ik}, & i \in P,\ k \in H; \qquad (3\text{-}11) \\
& x_{ijk} \in \{0,1\}, & i \in P,\ j \in T,\ k \in H.
\end{array}$$

Esta versão do problema foi demonstrada pertencer à classe de problemas

$\mathcal{NP}$-Completo em [17], utilizando para isso uma redução a partir do `3-SAT`.

3.1.3 Problema de otimização

A resolução de problemas de otimização requer a quantificação da qualidade das soluções. Essa responsabilidade fica a cargo da *função objetivo*, pois é através do valor obtido nessa função que as soluções são comparadas. Existem diversas funções objetivo definidas para o PPHE, a seguir está descrita a versão encontrada em [32]:

$$f(x) = \sum_{i=1}^{p}\sum_{j=1}^{t}\sum_{k=1}^{h} s_{ijk}x_{ijk}. \tag{3-12}$$

Neste caso, $S_{p\times t\times h}$ é uma matriz tridimensional de preferências, que possui, em cada entrada s_{ijk}, com $i \in P$, $j \in T$ e $k \in H$, um valor que indica a satisfação obtida pelo agendamento de uma aula no horário k, ministrada pelo professor i à turma j. Quanto maior o valor de s_{ijk} menor a respectiva aptidão, desta forma a função 3-12 deve ser utilizada em um problema de minimização,

3.1.4 Variantes do problema

Diversas variantes do problema descrito na seção 3.1.2 foram propostas na literatura, como forma de lidar com problemas reais. Algumas se aplicam somente a casos bem específicos, outras, porém, tornaram-se amplamente aceitas e utilizadas em estudos posteriores. A seguir, é apresentado um breve resumo das peculiaridades de algumas dessas variantes.

Aulas simultâneas

Alguns tipos de disciplinas permitem que diferentes turmas participem conjuntamente de uma mesma aula. Disciplinas como educação física, por exemplo, pertencem a essa categoria, obviamente, nenhuma das turmas envolvidas pode, então, ser atribuída a outra aula no mesmo horário [53].

Programação baseada em disciplinas

Em algumas situações, caso relatado em [12], as aulas estão associadas apenas a disciplinas, ou seja, existe o compromisso entre uma turma e um assunto de interesse, mas não necessariamente com algum professor específico, consequentemente, uma disciplina pode ser ministrada por mais de um professor ao longo do período letivo.

Salas especiais

A disponibilidade de salas não é levada em consideração na seção 3.1.2 porque, nesse tipo de problema, assume-se que cada turma possui sua própria sala, como, geralmente, ocorre nos cursos do primeiro e segundo grau brasileiros. No entanto, existem casos nos quais salas especias são necessárias, por exemplo, para aulas de música e laboratório, dentre outros. Como o número de salas especias é limitado, existe a necessidade de limitar a quantidade de aulas em salas especiais em um mesmo período.

3.2 Programação de Horários em Universidades

O *Problema de Programação de Horários em Universidades* (*University Course Timetabling*), abreviadamente PPHU, consiste em escalonar um conjunto de *aulas*, para cada disciplina, em um dado número de salas e horários. Contudo, diferentemente dos PPHE, o conceito de *turma* é menos importante, visto a possibilidade dos alunos personalizarem seus currículos. Desta forma, a ênfase é dada às disciplinas, de modo que disciplinas com estudantes em comum não podem ser associadas a um mesmo horário.

Outras diferenças importantes se dão quanto às dimensões do problema. Em universidades, a restrição quanto a disponibilidade de recursos é maior, mais disciplinas precisam ser alocadas a uma quantidade limitada de salas, o que difere do caso PPHE, onde, geralmente, cada *turma* possui sua própria sala.

Dependendo da estrutura acadêmica da instituição, existem duas subcategorias de problemas pertencentes ao PPHU, que se diferenciam, principalmente, pela forma com que a matrícula dos alunos é considerada:

1. **Programação de Horários pós-matrícula**[1]: Coloca as opções dos alunos em primeiro plano. Após os alunos se matricularem nas disciplinas e eventos de seu interesse, os horários são criados de forma que seja possível, aos alunos, participarem do eventos aos quais se matricularam.
2. **Programação de Horários baseada em currículo**[2]: É dada ênfase ao currículo do curso a ser seguido. Assim, a programação semanal é determinada de acordo com as disciplinas sugeridas pelo currículo.

A grande maioria das instituições de ensino superior se enquadra em uma dessas categorias, sendo que a segunda, *Programação de Cursos baseada em currículo*, atende melhor aos padrões do estudo de caso descrito no capítulo 5 e será o foco do restante desta seção.

[1] *Post Enrolment based Course Timetabling*
[2] *Curriculum based Course Timetabling*

3.2.1 Problema de Programação de Horários baseada em currículo

O *Problema de Programação de Horários baseado em Currículo*, ou abreviadamente PPHBC, considera um conjunto de d disciplinas, $D = \{1, \ldots, d\}$, sendo que, à cada disciplina $j \in D$ deve ser associada uma quantidade q_j de horários. Existem c *curricula* (definição 3.3), $C = \{1, \ldots, c\}$, tal que, disciplinas pertencentes ao mesmo *curricula* C_l, com $l \in C$, não podem ser ministradas simultaneamente.

Definição 3.3 *Um Curricula, no contexto do PPHU, é definido por um conjunto de disciplinas para as quais existem estudantes em comum. Ou seja, se um estudante está matriculado em uma disciplina de um Curricula, ele provavelmente estará matriculado em todas as outras. Pode ser visto como o fluxo de disciplinas sugerido para cada período do curso.* □

Existem ainda, h horários, $H = \{1, \ldots, h\}$, sendo que a cada horário k está associado um valor a_k indicando o número máximo de aulas que podem estar associadas a este horário. Ou seja, como existe uma quantidade limitada de salas, em cada horário k não podem ser ministradas mais aulas do que o número de salas disponíveis, a_k.

Mais detalhes sobre o PPHU do tipo PPHBC são apresentados nas próximas seções. A seção 3.2.2 descreve a formulação de um problema de busca, enquanto que na seção 3.2.3 o problema é descrito através de um modelo de otimização. Existem ainda, diversas peculiaridades que podem ser consideradas e que trazem modificações significativas à resolução do problema, algumas delas apresentadas na seção 3.2.4.

3.2.2 Problema de busca

Existem várias formulações possíveis para o PPHBC. A versão descrita a seguir foi proposta em [15], e utiliza como forma de representação uma matriz bidimensional $X_{d \times h}$, com entradas definidas pela equação 3-13:

$$x_{jk} = \begin{cases} 1 & \text{se uma aula da disciplina } j \text{ está associada ao horário } k, \\ 0 & \text{caso contrário.} \end{cases} \tag{3-13}$$

A restrição 3-14 garante que cada disciplina seja composta pelo número correto de aulas, a restrição 3-15 garante que, em cada horário, não existam mais aulas do que salas disponíveis e a restrição 3-16 previne disciplinas pertencentes ao mesmo *curricula* serem associadas ao mesmo horário.

$$
\begin{array}{llll}
\text{Encontrar} & x_{jk}, & j \in D,\ k \in H; & \\
\text{sujeito a:} & \displaystyle\sum_{k=1}^{h} x_{jk} = q_j, & j \in D; & \text{(3-14)} \\
 & \displaystyle\sum_{j=1}^{d} x_{jk} \leq a_k, & k \in H; & \text{(3-15)} \\
 & \displaystyle\sum_{j \in C_l} x_{jk} \leq 1, & l \in C,\ k \in H; & \text{(3-16)} \\
 & x_{jk} \in \{0,1\}, & j \in D,\ k \in H. &
\end{array}
$$

3.2.3 Problema de otimização

A adição da função objetivo 3-17, sugerida em [15], à formulação descrita na seção 3.2.2, descreve um problema de otimização. Sendo s_{jk} o grau de satisfação em se associar uma aula da disciplina $j \in D$ ao período $k \in H$ e x_{jk} conforme descrito em 3-13. Quanto maior s_{jk} menor a satisfação, desta forma o problema deve ser definido como de minimização.

$$f(x) = \sum_{j=1}^{d} \sum_{k=1}^{h} s_{jk} x_{jk}. \qquad \text{(3-17)}$$

Existem diversas formas de se obter a satisfação s_{jk}. Em [50] foi utilizada uma matriz $M_{d \times d}$, com entradas m_{ij}, $i \in D$ e $j \in D$, definidas como a soma da quantidade de estudantes matriculados às disciplinas i e j. A partir de m_{ij}, ou seja, a partir da quantidade de alunos envolvidos em cada par de disciplinas, foi estimada a insatisfação causada no caso de tais disciplinas estarem em conflito de horários.

Outras formas de avaliar a satisfação e, consequentemente, a qualidade da solução podem ser escolhidas. Para tal, as peculiaridades de cada instância do problema podem ser exploradas. Na seção 3.2.4 algumas dessas peculiaridades são descritas.

3.2.4 Variantes do problema

Nesta seção algumas das mais comuns variações do PPHBC são discutidas, dentre elas: *Múltiplas seções e o subproblema de agrupamento*, *Períodos de tamanho variável* e *Subproblema de atribuição de salas*.

Múltiplas seções e o subproblema de agrupamento

Existem casos em que uma disciplina é escolhida por uma quantidade elevada de alunos de diferentes *curricula*. Nestes casos é interessante que a disciplina seja dividida em múltiplas seções, o que pode ajudar na diminuição dos conflitos na tabela de horários.

Suponha o currículo $C_1 = \{d_1, d_2\}$ e o currículo $C_2 = \{d_1, d_3\}$, suponha também que uma aula da disciplina d_2 seja ministrada no horário h_1 e uma aula da disciplina d_3 no horário h_2. Neste caso, aulas da disciplina d_1 não poderiam ser ministradas nos horários h_1 e h_2, pois a disciplina d_1 pertence a C_1 e a C_2 o que geraria um conflito.

No entanto, se a disciplina d_1 for dividida em duas seções, d_1^1 e d_1^2, e os *curricula* alterados para $C_1 = \{d_1^1, d_2\}$ e $C_2 = \{d_1^2, d_3\}$, permancendo o horário h_1 ocupado por d_2 e o horário h_2 ocupado por d_3. Tem-se que, d_1^1 não pode ocupar h_1, contudo, d_1^1 pode ocupar h_2 sem criar um conflito com o currícula C_2. O mesmo ocorrendo, respectivamente, para o caso de d_1^2 e o horário h_1.

A separação de disciplinas em múltiplas seções tem como consequência um outro problema, chamado *subproblema de agrupamento* (*grouping subproblem*) e tem como objetivo atribuir os estudantes a uma determinada seção (turma), de uma disciplina. Este problema já foi abordado em [1, 28, 33, 50].

As abordagens que utilizam o subproblema de agrupamento, solucionam o problema geral em duas fases. Em [28], por exemplo, os dois problemas são resolvidos em sequência para cada iteração, primeiramente o PPHBC e em seguida o subproblema de agrupamento.

Períodos de tamanho variável

Até então foram considerados apenas casos onde todas as aulas possuíam o tamanho de um período. Em outras abordagens, no entanto, podem existir aulas de tamanhos variados. Em [20], por exemplo, aulas podem durar um, dois ou três períodos, e assim, para cada aula, o horário de início e a quantidade de períodos devem ser considerados. Em [29] um problema ainda mais geral é abordado, onde é permitido que as aulas de determinada disciplina possuam tamanhos diferentes.

Subproblema de atribuição de salas

Consiste em, dada uma tabela de horários fixa, associar aulas às respectivas salas. Em [20], este problema é resolvido através de uma redução para o Problema de Atribuição Quadrático (*Quadratic Assignment Problem*), e em [6] é feita uma análise detalhada do problema com a descrição de várias formulações e variantes, além de demonstrar em quais casos o problema pertence à classe $\mathcal{P}$ e em quais ele pertence à classe $\mathcal{NP}$-Completo.

3.3 Programação de Horários de Exames

Os *Problemas de Programação de Horários de Exames*, ou abreviadamente PPHE, consideram um conjunto de alunos que precisam realizar uma ou mais avaliações, distribuídas em um conjunto de horários que, possivelmente, extendem-se por mais de uma semana [45]. O objetivo é agendar as avaliações para dias e horários que evitem conflitos para os alunos, ou seja, nenhum aluno pode ter mais de uma avaliação agendada para o mesmo par, dia-horário.

Esse, é um problema de difícil resolução encontrado em instituições de ensino, e de fato, é um problema bem similar ao descrito na seção 3.2. Porém, algumas distinções são amplamente utilizadas na identificação do problema, as principais são apresentadas a seguir:

- Somente uma avaliação por disciplina;
- Os conflitos de horários entre alunos e avaliações devem ser tratados como restrições fortes. Alunos não podem deixar de comparecer a nenhuma das avaliações conflitantes;
- No máximo uma avaliação por dia para cada estudante, e não muitas avaliações em dias consecutivos;
- Pode haver mais de uma avaliação por sala, contendo diferentes turmas.

Formalmente, define-se um conjunto de d disciplinas, $D = \{1, \ldots, d\}$, sendo considerado apenas uma avaliação para cada disciplina. Existem também, *c curricula de avaliações* $C = \{1, \ldots, c\}$, sendo que cada $C_l \in C$ representa um conjunto de avaliações que não podem ser escalonadas a horários simultâneos (ver definição 3.3). Existe ainda, um conjunto de h períodos, $H = \{1, \ldots, h\}$, sendo que e_k indica o número máximo de exames que podem ser associadas ao período $k \in H$.

Nas seções a seguir, mais detalhes são apresentados sobre o PPHE: o problema de busca é formulado na seção 3.3.1, seguido pelo respectivo problema de otimização, descrito na seção 3.3.2, e das variações para este problema encontradas na literatura, seção 3.3.3.

3.3.1 Problema de busca

O problema de busca para o PPHE é semelhante ao descrito em 3.2.2. Utilizando-se uma matriz bidimensional $X_{d \times h}$, onde cada entrada x_{jk} é descrita pela equação 3-18:

$$x_{jk} = \begin{cases} 1 & \text{se existir uma avaliação da disciplina } j \text{ no horário } k, \\ 0 & \text{caso contrário.} \end{cases} \tag{3-18}$$

$$\begin{aligned}
\text{Encontrar} \quad & x_{jk}, && j \in D,\ k \in H; \\
\text{sujeito a:} \quad & \sum_{k=1}^{h} x_{jk} = 1, && j \in D; && \text{(3-19)} \\
& \sum_{j=1}^{d} x_{jk} \leq e_k, && k \in H; && \text{(3-20)} \\
& \sum_{j \in C_l} x_{jk} \leq 1, && l \in C,\ k \in H; && \text{(3-21)} \\
& x_{jk} \in \{0,1\}, && j \in D,\ k \in H.
\end{aligned}$$

A restrição 3-19 garante que apenas um horário esteja associado ao exame da disciplina j, já a restrição 3-20, tem como função, limitar a quantidade de avaliações que podem ser agendadas para determinado horário. Por último, a restrição 3-21, garante que avaliações de disciplinas pertencentes ao mesmo *curricula* $C_l \in C$ não sejam agendadas em horários conflitantes.

3.3.2 Problema de otimização

Assim como feito nas seções anteriores, modelar um problema de otimização requer a quantificação da qualidade das soluções. Neste tipo de problema, isso pode ser feito de diversas formas, a utilização da função 3-22, mede a quantidade de avaliações em horários consecutivos [46], considerando um problema de minimização ela pode ser utilizada evitar soluções com essas características:

$$f(x) = \sum_{k=1}^{h-1} \sum_{l=1}^{c} \sum_{i,j \in C_l} x_{ik} x_{jk+1}. \tag{3-22}$$

No entanto ela não é uma função linear, o que pode ser um problema dependendo da abordagem de resolução escolhida. Uma alternativa, seria a utilização de um função objetivo que contasse as ocorrências de avaliações em horários conflitantes com o objetivo de minimizar a quantidade de conflitos. A função objetivo 3-23 representa esse caso:

$$f(x) = \sum_{j \in C_l} \sum_{k=1}^{h} x_{jk}, \ l \in C. \tag{3-23}$$

3.3.3 Variantes do problema

Nesta seção, são descritas duas variantes do PPHE, relacionadas à quantidade de avaliações por sala e a minimização da quantidade de horários utilizados para agendar todas avaliações.

Atribuição de salas

As avaliações precisam ser associadas a salas, fato que depende da quantidade de alunos e da capacidade das salas. Algumas abordagens (ver [5]) permitem somente uma avaliação por sala em um dado período, e, nestes casos, existe algoritmo com complexidade polinomial no tamanho da entrada para o problema [46]. Por outro lado, se a quantidade de avaliações por sala for maior, o problema de decisão passa a pertencer à classe $\mathcal{NP}$-Completo, podendo ser visto como uma generalização do *Bin Packing* [23].

Minimizar o tamanho da sessão

Pode ser interessante minimizar a quantidade de períodos necessários para completar todas as avaliações. Nestes casos, o número de horários deve ser considerado para o cálculo da função objetivo, com o objetivo de minimizar a quantidade de horários utilizados.

3.4 Considerações finais

Neste capítulo, diversos tipos de problemas de programação de horários em instituições de ensino foram descritos. Maiores detalhes quanto a cada um dos problemas descritos e a complexidade dos mesmos podem ser obtidos nas referências [6, 32, 46, 47].

CAPÍTULO 4

Abordagem Proposta

Os problemas de alocação de horários, são, em sua maioria, problemas pertencentes à classe $\mathcal{NP}$-Difícil [11, 17], desta forma, solucioná-los através da utilização de métodos exatos (ver seção 2.4) ainda é um desafio para instâncias de tamanho razoável. De acordo com esse aspecto, a utilização de *meta-heurísticas* ou *métodos híbridos* se tornam boas opções.

Neste capítulo, enfatiza-se o uso de *métodos híbridos* (ver seção 2.6) para a resolução do PASA. Um *Problema de Programação de Horários em Universidades* (PPHU) do tipo baseado em currículos (ver seção 3.2.1) é tratado aqui, sem, no entanto, contemplar a alocação de salas de aulas (detalhes no capítulo 5).

Partindo desse pressuposto, o problema consiste em estipular quais professores ministrarão quais disciplinas, e em quais horários. O que pode ser obtido a partir da resolução de dois subproblemas menores, descritos a seguir.

O primeiro subproblema, descrito na seção 4.1, e aqui chamado *Problema Professor* $\times$ *Disciplina*, ou abreviadamente PAPD, representa a necessidade de se alocar disciplinas a professores, de modo a respeitar as preferências dos professores em relação ao turno da disciplina e a área a que ela pertence.

O segundo subproblema, descrito na seção 4.2, e aqui chamado *Problema Professor-Disciplina* $\times$ *Horário*, abreviadamente PAPDH, representa, dado um conjunto de professores associados a disciplinas, a necessidade de se escolher os horários para tais disciplinas, de modo a atender as restrições de disponibilidade dos professores e as restrições pedagógicas de interesse, além de tornar possível o cumprimento do fluxo curricular sugerido pelos currículos dos cursos.

Cada um dos subproblemas descritos é tratado como um PPLI de minimização, onde o objetivo final é a obtenção de soluções factíveis e de boa qualidade. Mais detalhes sobre a forma de obtenção de tais soluções são descritos nas seções 4.1 e 4.2, em relação ao PAPD e o PAPDH, respectivamente.

4.1 Problema Professor × Disciplina

O *Problema de Alocação Professor × Disciplina* (PAPD), conhecido na literatura como *Teacher Assignment*, é uma especialização do *Problema Generalizado de Atribuição* (*Generalized Assignment Problema*), ou abreviadamente PGA. O PGA é reconhecidamente um problema pertencente à classe $\mathcal{NP}$-Difícil [44], o que permite o mesmo ser concluído para o PAPD. A seção a seguir descreve o PGA e a especialização que o transforma no PAPD, em seguida o PAPD é descrito de acordo com os termos específicos encontrados na UFG.

4.1.1 Problema Generalizado de Atribuição e o PAPD

Dado um conjunto $D = \{1, \ldots, d\}$ e um conjunto $P = \{1, \ldots, p\}$, o objetivo do PGA é determinar uma associação, de mínimo custo, entre itens $j \in D$ e itens $i \in P$, de modo que cada j esteja associado a um único i, e que sejam respeitadas as restrições de disponibilidade de cada item i.

Tais restrições de disponibilidade podem ser descritas da seguinte forma: associar um item $j \in D$ a um item $i \in P$ tem como consequência um custo s_{ij} e consome uma quantidade q_{ij} dos recursos do item i. O máximo de recursos disponível para utilização é definido por max_i.

Considerando uma matriz $X_{p \times d}$ com entradas, x_{ij}, definidas de acordo com a equação 4-1, o PGA pode ser definido como um PPLI de minimização, com a formulação descrita pelas equações 4-2 a 4-5.

$$x_{ij} = \begin{cases} 1 & \text{se o item } j \in D \text{ está associado ao item } i \in P, \\ 0 & \text{caso contrário.} \end{cases} \tag{4-1}$$

$$\begin{aligned} \text{Minimizar} \quad & f(x) = \sum_{i \in P} \sum_{j \in D} s_{ij} x_{ij}, && && (4\text{-}2) \\ \text{sujeito a:} \quad & \sum_{j \in D} q_{ij} x_{ij} \leq max_i, && \forall\, i \in P; && (4\text{-}3) \\ & \sum_{i \in P} x_{ij} = 1, && \forall\, j \in D; && (4\text{-}4) \\ & x_{ij} \in \{0, 1\}, && \forall\, j \in D, \forall\, i \in P. && (4\text{-}5) \end{aligned}$$

Considerando-se o conjunto P como um conjunto de professores e o conjunto D como um conjunto de disciplinas, o PGA se reduz ao PAPD. Neste caso, o custo s_{ij} é considerado como a satisfação relacionada à associação da disciplina j ao professor i, ou seja, a preferência do professor i pela disciplina j (quanto menor esse valor maior a satisfação). O valor q_{ij} iria se referir a carga horária semanal da disciplina j quanto a sua associação com o professor i, no entanto, no PAPD esse valor independe do professor i e a notação pode ser alterada para q_j. O valor max_i se refere a carga horária máxima de trabalho estipulada para cada professor i.

Outra diferença entre tais problemas é que, no caso do PAPD, existe uma restrição adicional que visa garantir que cada professor esteja associado a uma carga de trabalho mínima maior que zero. Considerando-se a carga horária mínima exigida como min_i, adiciona-se a seguinte restrição ao PGA:

$$\sum_{j \in D} q_j x_{ij} \geq min_i, \quad \forall\, i \in P; \tag{4-6}$$

A partir dessas considerações o objetivo do PAPD é atribuir a cada professor um conjunto de disciplinas que esteja de acordo com a carga de trabalho e que minimize as situações em que disciplinas não preferidas sejam associadas a determinados professores. A formulação a seguir descreve formalmente essas ambições.

$$\text{Minimizar} \quad f(x) = \sum_{i \in P} \sum_{j \in D} s_{ij} x_{ij}, \tag{4-7}$$

$$\text{sujeito a:} \quad \sum_{j \in D} q_j x_{ij} \geq min_i, \quad \forall\, i \in P; \tag{4-8}$$

$$\sum_{j \in D} q_j x_{ij} \leq max_i, \quad \forall\, i \in P; \tag{4-9}$$

$$\sum_{i \in P} x_{ij} = 1, \quad \forall\, j \in D; \tag{4-10}$$

$$x_{ij} \in \{0,1\}, \quad \forall\, i \in P, \forall\, j \in D. \tag{4-11}$$

Como no PAPD existe a necessidade de associar cada uma das disciplinas a algum professor, somente haverá uma solução válida para o problema caso a seguinte desigualdade seja satisfeita:

$$\sum_{i \in P} min_i \leq \sum_{j \in D} q_j \leq \sum_{i \in P} max_i \tag{4-12}$$

Ou seja, a quantidade de horas a serem ministradas deve ser no máximo igual a quantidade total de horas de trabalho associada aos professores. A seguir, um resumo das principais características do PAPD, baseadas nas definições feitas anteriormente.

1. A cada professor $i \in P$, está associado um par de valores indicando a carga horária semanal mínima (restrição 4-8) e máxima (restrição 4-9) de trabalho que o mesmo deve cumprir;

2. A cada disciplina $j \in D$, está associado um valor q_j, que representa a carga horária semanal que deve ser ministrada à disciplina j;
3. Cada disciplina, $j \in D$, deve estar associada a apenas um professor, $i \in P$ (restrição 4-10);
4. Cada valor s_{ij}, representa a satisfação obtida da associação entre o professor i e a disciplina j. Quanto menor o valor s_{ij}, maior a preferência associada e consequentemente menor o valor da função objetivo (função 4-7).

4.1.2 Resolução do problema

O desenvolvimento das soluções para o `PAPD` seguiu uma abordagem *híbrida* do tipo *colaborativa e sequencial* (ver seção 2.6.1). Nesta seção são descritos os detalhes dessa implementação, que contém, de forma geral, alguns algoritmos para geração de soluções iniciais e um algoritmo de *Busca Tabu Iterada* para melhoria das soluções, além de uma versão utilizando o algoritmo de *Branch-and-Cut* implementado no pacote `GLPK` [36].

Nas próximas seções, tem-se a definição da estrutura utilizada na resolução do problema (seção 4.1.3), seguida da descrição de algoritmos para a construção de soluções iniciais (seção 4.1.4), e a descrição do algoritmo de melhoria que foi implementado (seção 4.1.5).

4.1.3 Estruturação do problema

Antes de uma descrição mais aprofundada em relação às heurísticas implementadas, alguma fundamentação se faz necessária. Deste modo, esta seção tem como objetivo definir a estrutura da implementação, servindo como base para o entendimento dos algoritmos descritos nas seções seguintes.

Inicialmente, é interessante definir a estrutura de armazenamento de uma solução para o `PAPD`. De acordo com o descrito na seção 4.1, a estrutura do problema precisa relacionar cada professor a um conjunto de disciplinas, o que pode ser reconhecido como um grafo bipartido $G = (V, A)$, onde o conjunto de vértices é definido como $V = P \cup D$ e o conjunto de arestas define as associações entre professores e disciplinas. A figura 4.1 representa um grafo bipartido relacionando o conjunto de professores ao conjunto de disciplinas.

A cada aresta a_{ij} do grafo G, está associado um custo, assim como definido na seção 4.1. No `PAPD`, esses custos são os valores relacionados à satisfação entre o professor e as disciplinas a ele atribuídas. A soma de custos associado a cada professor é armazenada em c_j, com $j \in P$.

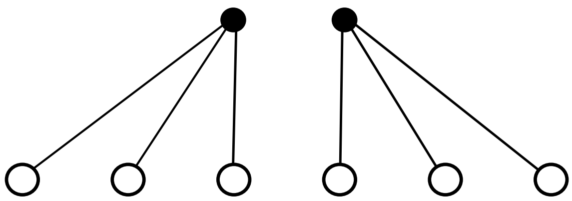

Figura 4.1: *Grafo G representando uma solução para o* PAPD. *Professores são representado por vértices pretos e disciplinas por vértices brancos.*

A partir do vetor c podem ser obtidas duas informações importantes, utilizadas nas demais seções. O valor da função objetivo e a identificação dos professores de maior e menor custo à solução.

O valor da função objetivo é facilmente obtido através da função 4-13, enquanto que o professor de maior e menor custo são especificados, respectivamente, pelas definições 4-14 e 4-15:

$$f(x) = \sum_{i \in P} c_i, \tag{4-13}$$

$$m \in P : \quad c_m \geq c_i, \quad \forall\, i \in P, \tag{4-14}$$

$$n \in P : \quad c_n \leq c_i, \quad \forall\, i \in P. \tag{4-15}$$

4.1.4 Construção de uma Solução Inicial

A geração da solução inicial é uma etapa fundamental para a qualidade das soluções obtidas por meta-heurísticas, pois a partir de boas soluções iniciais o algoritmo tem maior probabilidade de encontrar soluções de boa qualidade. Três algoritmos foram propostos para a geração de soluções iniciais para o PAPD; a seguir cada um deles é descrito em ordem cronológica de desenvolvimento.

Algoritmo Guloso

O algoritmo guloso aqui descrito se baseia em um algoritmo chamado *melhor escolha* que: dada uma disciplina, retorna o melhor professor ao qual tal disciplina pode ser associada no momento.

O funcionamento do algoritmo *melhor escolha* é simples. Dada uma disciplina j o algoritmo avaliará qual aresta a_{ij}, entre o professor i e a disciplina j, causará o menor aumento no valor da função objetivo. Essa aresta será, então, adicionada à solução.

A partir desse procedimento, o algoritmo geral se resume a determinar iterativamente a *melhor escolha* para cada uma das disciplinas $j \in D$, garantindo, além disso, que professores que já tenham atingido a carga de trabalho máxima não sejam considerados no processo. Concluindo então, com uma solução factível para o problema.

Relaxação Linear

A relaxação linear permite a obtenção, rápida, de soluções não factíveis para o PAPD em questão. Tais soluções podem apresentar, em algumas das entradas da matriz $X_{p \times d}$, valores não inteiros e esse é o fato que as tornam inviáveis para o PAPD. O que torna essa abordagem interessante para a geração de soluções iniciais, é que tais soluções podem ser transformadas, no caso do PAPD, em soluções inteiras de forma eficiente. Processo efetuado por um *algoritmo de arredondamento*.

O GLPK foi utilizado para a resolução da relaxação linear, e dois *algoritmos de arredondamento* foram implementados[1]:

- **Escolha ponderada**: Utiliza os valores não-inteiros encontrados na matriz $X_{p \times d}$ como auxílio à discretização. Ou seja, dado o caso em que a entrada x_{ij}, representando a associação da disciplina j ao professor i, possua valor não-inteiro entre 0 e 1, existirá, necessariamente, outras entradas x_{kj}, com $k \in P$, com valor não-inteiro, pois caso contrário a restrição 4-10 não seria respeitada e a solução em questão não seria factível. Assumindo esse pressuposto, o algoritmo se resume a alteração de todos os valores não-inteiros da matriz $X_{p \times d}$, o que é feito da seguinte forma: para cada disciplina $j \in D$, a entrada x_{ij}, $\forall\, i \in P$, com maior valor receberá valor 1, todas as demais entradas receberão valor 0.
- **Algoritmo guloso**: Se baseia no algoritmo *melhor escolha* descrito anteriormente. Neste caso a solução não-inteira é avaliada e a cada entrada x_{ij} não-inteira o algoritmo de *melhor escolha* é utilizado para indicar a qual professor associar a disciplina j, tem-se então que $x_{ij} = 1$ e todas as outras entradas x_{kj}, $\forall\, k \in P$ e $k \neq i$, são alteradas para 0.

Programação Linear Inteira

Um algoritmo *Branch-and-Cut* foi implementado utilizando o `CPLEX 9.0`. Essa opção pode ser utilizada para a obtenção de uma solução ótima para o problema. Porém, para instâncias grandes, o tempo necessário para a obtenção desta solução pode ser muito grande, o que sugere a utilização do algoritmo baseado em relaxação linear, ou o algoritmo guloso, seguido pelo processo de melhoria heurística descrito na seção a seguir.

[1]Nenhum desses arredondamentos garante soluções inteiras factíveis para o PAPD.

4.1.5 Melhoria da solução

Após a geração de uma solução inicial, tem-se como etapa subsequente o processo de melhoria da solução obtida. Esta etapa é utilizada como forma de obter, partindo da solução inicial, uma boa solução, o que é feito através de algum processo heurístico.

O processo de melhoria foi implementado como uma *Busca Tabu Iterada*, a qual se diferencia da *Busca Tabu* descrita na seção 2.5.3 pela utilização de um processo adicional de fuga de ótimos locais. Na *Busca Tabu Iterada* as informações obtidas ao decorrer de busca são utilizadas na determinação de uma nova solução, que será utilizada para o início de uma nova iteração assim que um ótimo local seja encontrado.

Essa nova solução é gerada a partir de um *algoritmo de diversificação*. Este tem como objetivo favorecer a exploração de diferentes áreas do espaço de soluções. Para isso, a solução gerada, nesta etapa, deve ser formada, preferencialmente, por elementos não contidos nas soluções exploradas anteriormente.

Desta forma, o objetivo do algoritmo se resume a obter uma solução que não possua determinadas características (*características ruins*). Para definir o conceito de *característica ruim* utilizado neste procedimento foram utilizados os custos c_i, $i \in P$, assim como definidos na seção 4.1.3. Ou seja, o conjunto de arestas relacionadas ao professor com maior custo representará a *característica ruim* da solução e, deste modo, é função do *algoritmo de diversificação* gerar uma solução que não contenha tais arestas.

A cada nova iteração da *Busca Tabu* um novo ótimo local pode ser encontrado e, assim que isso ocorra, um novo conjunto de arestas (*característica* ruim) pode ser adicionado ao conjunto de arestas ignoradas. Esse acúmulo evita que soluções previamente geradas pelo *algoritmo de diversificação* sejam geradas novamente, o que aumenta a diversidade das soluções exploradas no espaço de soluções [24, 25].

Esse procedimento, como pôde ser observado de forma experimental, obtém sucesso por algumas iterações, no entanto, após algumas iterações a quantidade de arestas ignoradas se torna grande o suficiente para que as soluções geradas não sejam de boa qualidade, o que prejudica o desempenho da *Busca Tabu*.

Visando tornar o *algoritmo de diversificação* menos restritivo, um outro procedimento foi implementado, o qual tem como função principal remover do conjunto de arestas ignoradas as arestas presentes em soluções que são ótimos locais. Experimentalmente foi observado que essa abordagem contribui para uma melhor qualidade das soluções geradas a longo prazo, favorecendo a *Busca Tabu*.

É importante salientar que ser uma característica ruim de um ótimo local não exclui as arestas associadas a esta *característica* de, possivelmente, estarem presentes em um ótimo global (ou solução de menor custo). Por este motivo, tais arestas somente são excluídas do processo de geração da solução pelo *algoritmo de diversificação*, podendo, no entanto, serem novamente avaliadas e adicionadas durante a *Busca Tabu*, se assim for conveniente.

Algoritmo 4.1: Busca Tabu, etapa de melhoria

```
T ← [];
x ← solução inicial gerada por Heurística Construtiva;
x̂ ← x;
estabilidade ← 0;
não_melhorou ← 0;
repita
    encontrar a melhor x' ∈ N(x) \ T;
    se f(x') < f(x̂) então
        x̂ ← x';
        adicionar x̂ a T;
        não_melhorou ← 0;
    senão
        estabilidade ← estabilidade + 1;
        se estabilidade = estabilidade_limite então
            diversificação(x');
            estabilidade ← 0;
            não_melhorou ← não_melhorou + 1;
        fim
    fim
    x ← x';
até não_melhorou = limite.
```

A partir de uma solução inicial x, a busca se inicia na vizinhança não-tabu de x, indicada por $N(x) \setminus T$, caso uma quantidade de iterações igual a *estabilidade_limite* decorra sem melhorias, entra em ação o *algoritmo diversificação*. Se mesmo assim nenhuma melhoria ocorrer em uma quantidade de iterações *limite*, a *Busca Tabu* é encerrada, ou por ter encontrado algum *ótimo global*, ou por estar presa em algum *ótimo local* $\hat{x}$.

4.2 Problema Professor/Disciplina $\times$ Horário

Esta seção descreve a estrutura geral do PAPDH e o porquê das restrições consideradas na seção 4.2.1 e a descrição de detalhes do modelo implementado na seção 4.2.2.

4.2.1 Estrutura geral do PAPDH

O Problema de alocação *Professor/Disciplina* × *Horário*, abreviadamente PAPDH, se refere ao problema de agendar os horários semanais de um conjunto de disciplinas. Tais disciplinas estão, através da resolução do PAPD, previamente associadas a professores e os horários disponíveis a cada uma delas sofrem algumas restrições de acordo com condições que devem ser observadas.

Dentre tais condições, serão descritas nas próximas seções os conflitos intra-*curricula*, conflitos inter-*curricula*, indisponibilidades, adequação ao turno do curso, alunos em conclusão do curso, aulas em horários consecutivos e aulas em dias consecutivos.

Conflitos intra-*curricula*

A partir da definição 3.3 conclui-se que duas ou mais disciplinas são classificadas como pertencentes a um mesmo *curricula* se e somente se possuírem a seguinte característica: se um aluno está matriculado a uma disciplina deste *Curricula*, ele também estará, provavelmente, matriculado em todas as outras do mesmo currículo.

Essa definição tem como consequência o fato de que disciplinas pertencentes ao mesmo *Curricula* não poderem ser ministradas simultaneamente. Essa é uma *restrição forte* visto que o não cumprimento torna a solução não-factível.

Conflitos inter-*Curricula*

Disciplinas pertencentes a diferentes *Curricula* podem ser ministradas simultaneamente na maioria dos casos, pois pouco provavelmente alunos estarão matriculados em disciplinas de diferentes *Curricula*.

O conflito, neste caso, pode ocorrer caso duas disciplinas ministradas simultaneamente estejam associadas ao mesmo professor. Este caso também deve ser tratado como uma *restrição forte*, pelo mesmo motivo anteriormente descrito.

Indisponibilidades

Existem situações em que alguns professores podem estar indisponíveis para o trabalho em sala de aula. Tais situações podem ocorrer devido a diferentes condições, um exemplo comum são as reuniões de cunho administrativo-pedagógico que são agendadas no início do semestre letivo.

Desta forma, cada professor possui um conjunto de pares *dia-período* em que o mesmo não estará disponível e portanto não deve ser associado a tais horários. A disponibilidade dos professores deve ser tratada como uma *restrição forte*.

Adequação ao turno do curso

Cada curso de graduação possui um turno no qual as disciplinas devem ser ministradas. Em alguns casos esse turno é preferencial e algumas disciplinas podem ser ministradas fora do turno padrão, contudo, em outros casos ele é fixo e não deve ser extrapolado. Essa informação é definida no currículo de cada curso.

Apesar dessa obrigatoriedade de horários, a adequação ao turno não será tratada neste PAPDH como uma *restrição forte*. Essa decisão se deve ao fato de que: de acordo com a quantidade de dias letivos por semana, quantidade de disciplinas e carga horária das mesmas, pode não existir solução factível e o problema ficaria então sem solução.

Assim sendo, a adequação ao turno do curso será tratada como uma *restrição fraca*, sendo que o objetivo na resolução neste PAPDH é encontrar a solução que minimize a quantidade de disciplinas associadas a horários fora do turno preferencial de cada curso.

Alunos em iminência de conclusão de curso

Em alguns casos, alunos nos últimos semestres do curso necessitam de disciplinas que estão fora do fluxo sugerido pelo curso em questão. Essa situação ocorre por diversos motivos, contudo, é de interesse da UFG e do aluno que a conclusão do curso possa ser alcançada no tempo previsto, ou o quanto antes.

Para atingir esse objetivo existe uma restrição específica no PAPDH. Cada aluno nessa situação é representado por um conjunto de disciplinas R_l, as quais não podem ser oferecidas simultaneamente para que o respectivo aluno tenha a oportunidade de se matricular em todas e possivelmente concluir o curso.

As restrições implicadas por tais conjuntos de disciplinas têm papel semelhantes ao das restrições implicadas pelos *curricula* de cada curso, fato que pode ser observado pela semelhança entre as restrições que as implementam (ver restrições 4-23 e 4-25).

É importante notar que o total de restrições implicadas pelo conjunto de *curricula* é limitada, enquanto que as restrições descritas nesta seção podem variar bastante. Dependendo da quantidade de conjuntos de disciplinas sendo considerados pode não existir solução factível, nestes casos, cabe ao responsável pela alocação de horários priorizar alguns conjuntos em detrimento a outros.

Existem formas de melhor tratar essa situação, no entanto, não foi possível implementá-la neste trabalho, ficando como sugestão para possíveis trabalhos futuros.

Aulas em horários consecutivos

Por motivos pedagógicos, escalonar duas aulas de uma mesma disciplina para horários consecutivos não é uma situação indicada. Deste modo soluções para o PAPDH não devem permitir esse tipo de situação, o que é feito através da implementação de uma *restrição forte* que limita a quantidade de aulas por dia, para cada disciplina, a no máximo igual a 1.

Aulas em dias consecutivos

Agendar aulas de uma mesma disciplina em dias consecutivos também não é uma característica bem vista em termos pedagógicos, deste modo existe uma *restrição forte* no PAPDH que impede esse tipo de situação. Isso é feito através da limitação de que a quantidade de aulas, para cada disciplina, em dois dias consecutivos seja no máximo igual a 1.

4.2.2 Resolução do problema

A implementação do PAPDH foi baseada em um algoritmo *Branch-and-Cut*[2]. Não foram implementados métodos heurísticos para este subproblema, uma vez que os testes demonstraram bom desempenho quanto ao tempo gasto na obtenção de soluções ótimas para o problema.

Antes de uma descrição do modelo implementado alguns detalhes precisam ser definidos, principalmente quanto a representação da instância do problema e detalhes intrínsecos ao problema na UFG.

Dado um conjunto $D = \{1, \ldots, d\}$, de disciplinas cada uma relacionada a uma quantidade T_j de turmas, um conjunto $H = \{1, \ldots, h\}$, de horários e um conjunto $C = \{1, \ldots, c\}$, de *curricula*, o objetivo é determinar uma associação, de mínimo custo, entre disciplinas $j \in D$ e horários $k \in H$, de modo que aulas de uma turma-disciplina de um mesmo *curricula* $C_l \in C$ não sejam agendadas em horários simultâneos, e que disciplinas de um professor i, representadas por D_i, também não estejam agendadas em horários simultâneos. Também devem ser garantidas todas as restrições descritas na Seção 4.2.1.

A solução é representada por uma matriz $X_{T_j \times d \times h}$, sendo T_j a quantidade de turmas de uma disciplina j e h a quantidade de horários disponíveis para alocação (desconsiderando inviabilidades, ou seja, supondo uma semana com 5 dias letivos e 6 horários de aula por dia tem-se um total de $5 \times 6 = 30$ horários). Cada entrada na matriz $X_{d \times h}$ é definida pela equação 4-28:

$$x_{qjk} = \begin{cases} 1 & \text{se a turma } q \text{ da disciplina } j \in D \text{ está associada ao horário } k \in H, \\ 0 & \text{caso contrário.} \end{cases} \tag{4-16}$$

[2] Uma versão utilizando GLPK e uma versão utilizando `CPLEX 9.0`.

A adequação ao turno é uma medida de grande importância para esta abordagem, pois é através dela que a qualidade da solução será quantificada. Cada disciplina j possui um valor que indica o turno preferencial, $tp_j \in \{1, 2, 3\}$, no qual ela deve ser ministrada (1 - Matutino, 2 - Vespertino, 3 - Noturno). Como os horários estão em escala semanal e o turno preferencial em escala diária, um pequeno cálculo deve ser feito para que seja possível comparar o turno preferencial com o turno do horário que está sendo avaliado.

A função 4-17 auxilia na obtenção do turno do horário k, enquanto que a adequação, ou satisfação para seguir a notação usada em outras seções, é definida pela equação 4-18.

$$t(k) = \left\lfloor \frac{k \mod (\text{número de horários por dia})}{\text{número de horários por turno}} \right\rfloor, \; k \in H; \tag{4-17}$$

$$s_{jk} = \begin{cases} 0 & \text{se } t(k) = tp_j, \\ \sigma & tp_j \in \{1,2\} \text{ e } t(k) = 3, \\ \omega & tp_j \in \{1,2\} \text{ e } t(k) \in \{1,2\}. \end{cases} \tag{4-18}$$

Os valores σ e ω são constantes, sendo que $\sigma > \omega$. Um curso noturno não pode ser associado a um horário diurno, por isso a penalidade σ para esses casos deve ser alta. Já para cursos diurnos, associar uma disciplina em período não ideal é bastante comum em cursos de turno integral. Desta forma, o valor ω também penaliza o custo da solução, porém com menor impacto do que σ.

A formulação completa para o PAPDH está descrita de 4-20 a 4-27, com a *satisfação* aparecendo como multiplicador na função objetivo 4-20. A restrição 4-21 implementa o quesito indisponibilidade descrito anteriormente, para tal, é utilizada uma matriz $I_{d \times h}$, sendo que cada entrada nessa matriz é definida pela equação 4-19:

$$I_{jk} = \begin{cases} 0 & \text{se o professor associado à disciplina } j \text{ está disponível no horário } k, \\ 1 & \text{caso contrário.} \end{cases} \tag{4-19}$$

A restrição 4-22 evita conflitos inter-*curricula*, permitindo que apenas uma disciplina de determinado professor esteja associada a um certo horário k por vez. Enquanto as restrições 4-23 e 4-24 constituem uma forma de evitar conflitos intra-*curricula*, mantendo a possibilidade de duas turmas de uma mesma disciplina serem ministradas em horários simultâneos. O vetor T armazena em cada entrada T_j a quantidade de turmas associadas à disciplina j.

A restrição 4-25 garante que disciplinas pendentes de alunos formandos não sejam alocadas a horários simultâneos, enquanto que a restrição 4-26 evita que aulas de uma mesma disciplina sejam ministradas em períodos consecutivos e também que aulas de uma mesma disciplina sejam ministradas em dias consecutivos (nessa restrição n é o número de períodos por dia). Por fim, a restrição 4-27 garante que a quantidade necessária de aulas sejam ministradas semanalmente, o valor β é a carga horária por aula.

$$
\begin{aligned}
\text{Minimizar} \quad & \sum_{j=1}^{d}\sum_{q=1}^{T_j}\sum_{k=1}^{h} s_{jk}x_{qjk}, && && (4\text{-}20)\\
\text{sujeito a:} \quad & \sum_{j=1}^{d}\sum_{q=1}^{T_j}\sum_{k=1}^{h} I_{jk}x_{qjk} = 0, && && (4\text{-}21)\\
& \sum_{j\in D_i}\sum_{q=1}^{T_j} x_{qjk} \le 1, && \forall i \in P, \forall k \in H; && (4\text{-}22)\\
& \sum_{j\in C_l}\sum_{q=1}^{T_j} x_{qjk} \le \sum_{j\in C_l} T_j, && \forall l \in C, \forall k \in H; && (4\text{-}23)\\
& \sum_{q=1}^{T_j} x_{qjk} \le T_j && \forall j \in D, \forall k \in H; && (4\text{-}24)\\
& \sum_{j\in R_l}\sum_{q=1}^{T_j} x_{qjk} \le 1, && \forall l \in R, \forall k \in H; && (4\text{-}25)\\
& \sum_{k=u}^{u+2n}\sum_{q=1}^{T_j} x_{qjk} \le T_j, && j \in D, u = 1+(v-1)n, && (4\text{-}26)\\
& && v = 1,\ldots,dias; && \\
& \sum_{k=1}^{h}\sum_{q=1}^{T_j} \beta x_{qjk} = q_i, && j \in D; && (4\text{-}27)\\
& x_{qjk} \in \{0,1\}, && j \in D,\ k \in H,\ q \in T_j. &&
\end{aligned}
$$

Essa formulação no entanto contempla alguns detalhes referentes ao tratamento de múltiplas turmas que ainda não foram implementados. Os resultados descritos na seção 5.3.3, se referem à formulação mais simples apresentada abaixo, em que:

$$
x_{jk} = \begin{cases} 1 & \text{se a disciplina } j \in D \text{ está associada ao horário } k \in H, \\ 0 & \text{caso contrário.} \end{cases} \qquad (4\text{-}28)
$$

As principais diferenças ocorrem nas restrições 4-29, 4-30, 4-31 e 4-32. Nestes casos, foi considerada apenas uma turma por disciplina. Duas versões de um algoritmo *Branch-and-Cut* foram implementadas para esta formulação.

$$
\begin{aligned}
\text{Minimizar} \quad & \sum_{j=1}^{d}\sum_{k=1}^{h} s_{jk}x_{jk}, & & \\
\text{sujeito a:} \quad & \sum_{j=1}^{d}\sum_{k=1}^{h} I_{jk}x_{jk} = 0, & & \\
& \sum_{j \in D_i} x_{jk} \leq 1, & \forall\, i \in P, \forall\, k \in H; & \quad (4\text{-}29) \\
& \sum_{j \in C_l} x_{jk} \leq 1, & \forall\, l \in C, \forall\, k \in H; & \quad (4\text{-}30) \\
& \sum_{j \in R_l} x_{jk} \leq 1, & \forall\, l \in R, \forall\, k \in H; & \quad (4\text{-}31) \\
& \sum_{k=u}^{u+2n} x_{jk} \leq 1, & j \in D, u = 1 + (v-1)n, & \quad (4\text{-}32) \\
& & v = 1, \ldots, dias; & \\
& \sum_{k=1}^{h} \beta x_{jk} = q_i, & j \in D; & \\
& x_{jk} \in \{0, 1\}, & j \in D,\ k \in H,\ . &
\end{aligned}
$$

4.3 Considerações finais

Neste capítulo foram propostas abordagens para a resolução do `PAPD` e do `PAPDH`. Para o `PAPD`, uma *Busca Tabu Iterada* e duas versões de um algoritmo *Branch-and-Cut* foram implementadas para a resolução do problema descrito na Seção 4.1. Já para o `PAPDH` apenas o algoritmo *Branch-and-Cut* foi implementado, em duas versões, visando a resolução do problema descrito na Seçao 4.2.

CAPÍTULO 5

Estudo de Caso

5.1 O Problema na UFG

No processo seletivo `2010-1` a UFG ofereceu 5.459 vagas em cursos de graduação, distribuídas entre 124 cursos presenciais e seis cursos na modalidade à distância. O total de alunos nos cursos presenciais, dentre calouros e veteranos, segundo dados de 2009, aproxima-se de 17.000. Esses números fornecem uma boa ideia da dimensão do problema enfrentado na UFG quanto a programação de horários de aulas.

Inicia-se com uma visão geral das estruturas organizacionais da instituição e que são de interesse a este trabalho (seção 5.1.1), seguida pela descrição da estrutura dos cursos de graduação (seção 5.1.2), da oferta de disciplinas (seção 5.1.3) e da descrição do processo de agendamento de aulas (seção 5.1.4).

5.1.1 Estrutura organizacional da UFG

Esta seção descreve a estrutura organizacional da UFG e especifica as responsabilidades associadas a cada um dos órgãos participantes do processo de agendamento de aulas, informações obtidas em [43].

A estrutura organizacional da UFG é composta por uma administração central constituída dos seguintes órgãos: *Reitoria*, Conselho Universitário de Ensino, Conselho Universitário de Extensão e Cultura e de Curadores. Sendo a *Reitoria* de maior interesse ao contexto desta seção.

A *Reitoria* é o órgão responsável pelas funções administrativas da instituição. O planejamento e o relacionamento com os interesses da comunidade (interna e externa) são exemplos dessas funções. A *Reitoria* é composta pelo Gabinete do Reitor, as Pró-Reitorias e ainda as coordenadorias e assessorias especiais, os órgãos suplementares, os campus do interior e os órgãos administrativas.

Existem diferentes Pró-Reitorias, cada uma sendo responsável por uma área de atuação, cabendo a elas assessorar a *Reitoria* no estabelecimento de sua política de atuação. Atualmente tem-se: Pró-Reitoria de Graduação (PROGRAD), Pró-Reitoria de

Pesquisa e Pós-Graduação (PRPPG), Pró-Reitoria de Administração e Finanças (PROAD), Pró-Reitoria de Desenvolvimento Institucional e Recursos Humanos (PRODIRH), Pró-Reitoria de Extensão e Pró-Reitoria de Cultura (PROEC).

Para o propósito deste trabalho, a PROGRAD[1] é a mais importante pois trata de aspectos relacionadas à graduação. Em texto extraído de [43], a função da PROGRAD é definida como:

> *Coordenar e desenvolver meios adequados para assegurar o alto nível de ensino da graduação e a plena integração da comunidade com a Universidade.*

Com esse objetivo, existem dois órgãos vinculados a PROGRAD: o Centro de Seleção (CS), responsável por executar o processo de seleção para o ingresso de novos alunos, bem como outros processos seletivos; e o Departamento de Assuntos Acadêmicos (DAA) que realiza o controle acadêmico dos alunos e ex-alunos da graduação, e dos dados e informações quanto ao desempenho acadêmico.

5.1.2 Estrutura dos cursos de graduação

Os cursos de graduação da UFG conferem graus acadêmicos de nível superior, nas modalidades: bacharelado, licenciatura ou que assegurem o exercício profissional. O *Regulamento Geral dos Cursos de Graduação* (RGCG) organiza as atividades acadêmicas em semestres letivos, constituídos de no mínimo cem dias letivos por semestre, nos quais devem ser ministradas disciplinas do Núcleo Comum (NC), Núcleo Específico (NE) e Núcleo Livre (NL).

O projeto pedagógico que especifica o conjunto de disciplinas pertencentes a cada um desses núcleos, assim como as diretrizes básicas de cada curso e outras situações de ensino-aprendizagem relacionadas à formação do aluno são especificados pelo currículo do curso de graduação.

Nas seções a seguir define-se as características de cada um dos grupos de conteúdos citados anteriormente: NC, NE e NL; assim como uma descrição do processo de oferta de disciplinas.

Núcleo Comum (NC)

O Núcleo Comum é composto por um conjunto de conteúdos (disciplinas) que podem ser compartilhados entre diferentes cursos de uma mesma área. A disciplina de *Cálculo* é um bom exemplo deste caso, pois é ministrada aos cursos de Matemática, Ciências da Computação, Engenharias, dentre outros.

[1] http://prograd.ufg.br

Em tal situação, disciplinas são ministradas por uma *Unidade Acadêmica* (UA) a cursos vinculados a outras unidades. Pelo lado da UA responsável pela disciplina, utiliza-se o termo *disciplina de serviço*, enquanto que a UA destino das aulas utiliza o termo *disciplina externa* para indicar uma disciplina que faz parte do currículo e que é ministrada por outra UA.

O NC é ministrado através de disciplinas obrigatórias definidas pelo currículo de cada curso e possuem carga horária que deve ser limitada a 70% do total de disciplinas do currículo.

Núcleo Específico (NE)

O Núcleo Específico é o conjunto de conteúdos que especificializa a formação do profissional, pois, diferentemente das disciplinas do NC, são específicas a cada curso. A disciplina *Teoria da Computação* é um bom exemplo de disciplina de NE, pois é ministrada somente ao curso de Ciências da Computação.

Disciplinas do NE podem ser optativas ou obrigatórias, informação que deve ser definida no currículo do curso e devem ocupar no mínimo 20% da carga horária total necessária para integralização curricular.

Núcleo Livre (NL)

O Núcleo Livre é o conjunto de conteúdos que objetiva garantir liberdade ao alunos para ampliar sua formação. Disciplinas deste tipo não fazem parte da matriz curricular definida para o curso, pois elas podem ser escolhidas livremente, pelo aluno, dentre todas as disponíveis, nesta categoria, no âmbito da Universidade.

A carga horária total do NL deve ocupar no mínimo 5% do total de carga horária do curso.

5.1.3 Oferta de disciplinas

A unidade acadêmica deve definir a cada semestre letivo, em data estabelecida no calendário acadêmico, disciplinas que serão por ela oferecidas, especificando as turmas e seus respectivos horários, bem como o número de vagas ofertadas por núcleo, curso e turma.

Alterações dos horários de aulas após a matrícula no semestre letivo, somente serão efetuadas mediante anuência dos alunos matriculados, do professor da disciplina e da coordenadoria do curso. Sendo que é facultada às unidades acadêmicas o cumprimento, ou não, da oferta de disciplinas que não alcancem o número mínimo de cinco alunos inscritos.

A matrícula é feita pelos alunos e não sofre interferência das unidades responsáveis, exclusos alguns casos:

- Não é permitida a inscrição em disciplinas com horários simultâneos.
- O total de carga horária semanal em disciplinas do NC e NE não pode ser superior a trinta horas, e quarenta horas para os cursos de período integral.

5.1.4 O agendamento semanal de aulas

O agendamento de aulas na UFG segue uma hierarquia que relaciona, basicamente, as unidades acadêmicas (UA) e a PROGRAD. Cada UA tem liberdade para definir: as disciplinas a serem ofertadas, os horários em que as mesmas serão ministradas e os professores responsáveis por ministrá-las. No entanto, devido às disciplinas de NC essa liberdade se torna dependente de outras UA's.

A interdependência entre as UA's é um dos pontos cruciais do problema enfrentado na UFG. Nesta seção, essa e outras características são descritas, como forma de identificar as características que tornam tal problema diferente do modelo geral de PPHU, descrito na seção 3.2.

O processo geral de agendamento se inicia nas UA's. Cada uma das unidades se responsabiliza por decidir os horários de aulas de todas as disciplinas ofertadas e previstas pelo currículo do curso (horários baseados em currículo, ver a seção 3.2.1), assim como o professor responsável por cada uma das disciplinas.

Em se tratando de disciplinas do NE, esse é um processo isolado a cada UA, pois os professores em questão geralmente estão associados à própria unidade. No entanto, quando uma UA precisa decidir os horários de disciplinas do NC o processo se complica, pois, neste caso, a decisão dos horários está submetida à disponibilidade de um professor de outra UA, com disponibilidades não conhecidas pela unidade que está requisitando os seus serviços.

Devido a essa interseção entre os conteúdos de diferentes cursos, uma parte do agendamento de horários é efetuada através de um trabalho conjunto entre as UA's envolvidas, visando o estabelecimento de horários de aulas factíveis para ambas.

Outra etapa importante do agendamento de aulas é a alocação de salas. Definir quais salas serão utilizadas para cada aula também é responsabilidade das UA's. No entanto, nos casos em que a UA não possui suas próprias salas esse é um processo complicado, visto que é difícil para elas ter a visão geral e saber quais salas utilizar sem que conflitos sejam criados em relação à utilização da mesma sala por outras unidades.

Considerando-se o panorama atual, verifica-se que grande parte dos cursos de graduação não possuem salas próprias e a tendência, devido ao crescimento previsto para a UFG, é que cada vez mais haja o compartilhamento de salas entre diferentes cursos.

A avaliação dos agendamentos criados pelas UA's, principalmente em relação a salas de aulas, requer um ponto centralizador e que tenha a visão geral da Universidade, uma responsabilidade atualmente atribuída à PROGRAD. Ela não é responsável pelo agendamento de aulas, contudo, é um ponto central de avaliação que pode ser notificado quanto a conflitos de alocação de salas e conflitos de horários de aulas.

5.2 O Problema no INF/UFG

Nesta seção a instância do PASA referente ao Instituto de Informática da UFG é descrita. Características dos cursos associados ao INF, como por exemplo turno e turno preferencial são exemplificadas, além de outros detalhes como *disciplinas externas* ao instituto, *disciplinas de serviço* e demais informações pertinentes à resolução do problema.

O objetivo desta seção é descrever de forma prática os itens da instância do INF utilizados como dados de teste para a abordagem proposta no capítulo 4. Para isso os seguinte quesitos serão descritos: turnos das disciplinas (seção 5.2.1), tipos de disciplinas (seção 5.2.2), carga horária de trabalho (seção 5.2.3), preferências dos professores (seção 5.2.4), indisponibilidades dos professores (seção 5.2.5) e alunos em conclusão de curso (seção 5.2.6).

5.2.1 Turno preferencial e turno obrigatório

Os cursos de graduação associados ao INF podem ser divididos em dois grupos, o primeiro grupo constitui-se de cursos que possuem *turno obrigatório*, ou seja, as disciplinas descritas pelos currículos de tais cursos devem ser ministradas exclusivamente em turnos pré-especificados. Por exemplo, disciplinas de cursos como Engenharia de Software e Sistemas de Informação, salvo raras exceções, devem ser ministradas integralmente no período noturno. Tais cursos serão considerados neste trabalho como de *turno obrigatório*.

O segundo grupo é constituído apenas por cursos definidos como de período integral, o que implica que as disciplinas associadas a ele podem ser ministradas em qualquer um dos períodos do dia: matutino, vespertino ou noturno. No momento, o único curso deste grupo (no INF) é o de Ciências da Computação.

Para este curso, o INF aloca preferencialmente, o turno vespertino para os períodos ímpares e o turno matutino para os períodos pares. Por exemplo, turmas do primeiro período do curso possuem aulas, preferencialmente, no turno vespertino, enquanto que turmas do segundo período possuem aulas, preferencialmente, no turno matutino.

O que torna essa divisão importante para a resolução do problema é que em um curso integral, nem todos os períodos do dia são ocupados por aulas, mas dependendo de qual semestre letivo a turma está associada existe um turno preferencial para o qual as aulas dessa turma devem ser escalonadas.

Tal situação é definida como *turno preferencial*, e é mais flexível que a situação dos cursos noturnos. Pode ocorrer que uma turma prevista para o turno matutino tenha aulas no turno vespertino e isso não é uma violação do estatuto do curso (previsto pela definição de curso de período integral). Enquanto que para um curso noturno essa mudança de turno não é algo previsto.

Concluindo, considera-se para o caso do `INF`, que cursos de turno noturno possuem turno obrigatório, enquanto cursos de período integral possuem turno preferencialmente diurno. Para cada disciplina descrita nos dados de entrada existe um campo indicando a qual tipo de turno ela pertence.

5.2.2 Tipos de disciplinas

Dependendo a qual núcleo determinada disciplina pertence, Núcleo Comum (`NC`), Núcleo Específico (`NE`) ou Núcleo Livre (`NL`) ela é definida no `INF` como pertencente a uma de três categorias: *disciplinas internas*, *disciplinas de serviço*, *disciplinas externas*.

As *disciplinas internas* são disciplinas pertencentes ao `NE` do curso, neste caso o professor responsável por ministrá-la está, geralmente, associado ao próprio `INF`. Já as *disciplinas de serviço* e as *disciplinas externas* fazem parte do `NC`, a diferença, neste caso, está relacionado ao professor responsável pela disciplina.

No primeiro caso, a disciplina será ministrada por um professor associado ao `INF` a cursos externos ao instituto (*ex.*: Agronomia, Engenharia Elétrica). No segundo caso, a disciplina é ministrada para cursos do `INF` por um professor associado a outro instituto (*ex.*: Instituto de Matemática e Estatística – `IME`, Instituto de Física – `IF`).

Cada disciplina deve ser classificada em uma dessas categorias. Essa informação é utilizada durante as duas etapas de resolução do problema. *Disciplinas externas* não entram na resolução do `PAPD` e *disciplinas de serviço* não são consideradas na resolução do `PAPDH`.

5.2.3 Professores e carga horária de trabalho

Existem diferentes regimes de trabalho aos quais um professor pode estar relacionado, para cada um desses regimes estão definidas as cargas horárias que o professor deve cumprir. Esses valores são estipulados como uma média anual de trabalho, ou seja, supondo uma carga horária prevista como 8 horas-aula semanais, caso esse professor não ministre aulas no primeiro semestre ele deverá cumprir 16 horas-aula semanais no segundo semestre, como forma de alcançar a média anual.

A cada professor presente nos dados de entrada estão associados dois valores indicando a carga horária mínima e máxima desejada para o semestre em questão. Caso um valor fixo de carga horária seja necessário para o semestre, o mínimo e o máximo devem ser iguais. Essa informações são utilizadas na resolução do PAPD.

5.2.4 Professores e preferências

Assim como descrito na seção 4.1, o objetivo do PAPD é associar professores a disciplinas, de modo a atender o máximo possível as preferências dos professores. Para que isso seja viável é necessária uma indicação, por parte dos professores, quanto a quais disciplinas eles gostariam de ministrar.

Cada professor estabelece para cada uma das disciplinas de seu interesse um valor dentre 1, 2 ou 3 que indica sua preferência (inversamente proporcional ao valor, 1 é a maior) por determinada disciplina. Essa informação é utilizada na resolução do PAPD.

5.2.5 Professores e indisponibilidades

Existem diversos motivos que podem implicar em indisponibilidade de professores em relação a algum dia-horário. As reuniões pedagógico-administrativas que ocorrem periodicamente é um desses motivos.

Para evitar que aulas sejam agendadas em períodos de indisponibilidade essa informação deve ser indicada pelos professores. Cada um deles deve estabelecer em quais horários semanais não estarão disponíveis para o trabalho em sala de aula. Essa informação será utilizada na resolução do PAPDH.

5.2.6 Alunos em iminência de conclusão de curso

Em alguns casos, alunos nos últimos períodos do curso necessitam de disciplinas que estão fora do fluxo sugerido pelo curso em questão. Essa situação pode ocorrer por diversos motivos, contudo, é de interesse do INF que o aluno consiga concluir o curso no tempo previsto, ou o quanto antes possível. Essa informação é obtida da coordenadoria de cada curso ou por iniciativa dos alunos interessados.

5.3 Experimentos e resultados

Os testes computacionais foram implementados utilizando linguagem C++, avaliados sobre uma plataforma Intel(R) Core(TM) 2 Duo, CPU E7400@2.80GHz 3,25GB

RAM e efetuados sobre uma instância de teste obtida junto aos responsáveis pelo agendamento de aulas no `INF`. Tal instância foi criada a partir das disciplinas *internas* e *de serviço* dos cursos de graduação do `INF`. A limitação a esse conjunto de disciplinas, permite ao `PAPDH` ser solucionado de acordo exclusivamente aos interesses do `INF`. Caso disciplinas *externas* fossem consideradas existiria a necessidade de interação entre as alocações de horários entre diferentes *Unidades Acadêmicas*, situação não tratada neste trabalho.

Desta forma, a instância `A`, em questão, descreve um conjunto de 54 professores (carga horária de trabalho máxima disponível semanalmente de 353 horas), 103 disciplinas de turno *obrigatório* ou *preferencial* (carga horária de trabalho requisitada de 353 horas), 18 *curricula* e 35 restrições adicionais referentes a disciplinas de alunos formandos (seção 5.2.6).

Nas seções a seguir diversos pontos observados a partir dos experimentos são comentados, assim como a descrição de alguns resultados obtidos. Na seção 5.3.1 alguns importantes detalhes sobre o conjunto de preferências professor-disciplina são descritos, seguidos na seção 5.3.2 dos resultados obtidos na resolução do `PAPD` e na seção 5.3.3 os resultados e comentários em relação à resolução do `PAPDH`.

5.3.1 Preferências

As preferências entre professores e disciplinas foram indicadas pelos próprios professores. Cada professor associou um valor dentre $\{1, 2, 3\}$ para cada disciplina de seu interesse, sendo 1 alta preferência, 2 média preferência e 3 média-baixa preferência. Para a instância `A`, em questão, 331 preferências foram especificadas pelos professores, ou seja, uma média de 3 professores demonstraram interesse por cada disciplina.

O que pôde ser observado das preferências especificadas é que não houve uma boa distribuição da quantidade de disciplinas selecionadas por cada professor. Alguns determinaram preferências para uma grande quantidade de disciplinas, enquanto outros professores nem mesmo indicaram disciplinas o suficiente para atingir a sua carga horária de trabalho.

Esse tipo de distribuição tem um grande impacto na resolução do problema. Caso um professor especifique apenas disciplinas de alta preferência, por exemplo, ele será beneficiado em relação a outros professores que tenham especificado um amplo conjunto de disciplinas. Isso ocorre porque os métodos de resolução favorecem associações de baixo custo à solução, desta forma, se um professor especifica poucas disciplinas e somente de alta preferência, todas as outras relações entre ele e outras disciplinas implicarão em alto custo a solução e, dificilmente, serão relacionadas. Tais disciplinas serão, deste modo, associadas a outros professores, provavelmente àqueles que a consideraram de média ou média-baixa preferência.

Nessa situação uma má distribuição pode ocorrer, visto que, professores com poucas preferências especificadas ficarão com suas disciplinas de alta preferência, enquanto professores com muitas preferências especificadas ficarão, na maioria dos casos de conflitos de interesses por disciplinas, com as de baixa preferência.

Considere o seguinte exemplo, dois professores: p_1 e p_2, e quatro disciplinas $\{d_1, d_2, d_3, d_4\}$, sendo que cada professor deve ministrar duas disciplinas. A Tabela 5.1 representa as preferências entre os professores e todas as disciplinas, onde $\varphi = 90$ é um valor alto para os casos em que a preferência não foi especificada pelo professor.

Neste caso, o professor p_1 somente especificou as duas disciplinas que gostaria de ministrar e atribuiu a elas alta preferência. Já o professor p_2, também tem interesse pelas mesmas duas disciplinas, porém, especificou disciplinas alternativas a serem escolhidas.

	d_1	d_2	d_3	d_4
p_1	1	1	φ	φ
p_2	1	1	2	3

Tabela 5.1: *Exemplo de má distribuição de preferências.*

Neste tipo de situação uma *solução ótima* sempre relacionará as duas primeiras disciplinas ao primeiro professor e as duas outras ao segundo professor, porque essa é a situação em que se obtém o menor custo para a solução. A Tabela 5.2 descreve todas as possíveis associações e os respectivos valores da função objetivo, comprovando que, nesse tipo de situação, o professor p_2 sempre ficará (em uma solução ótima) com disciplinas de baixa preferência. Vale lembrar que o valor da função objetivo é calculado através da soma das preferências associadas aos pares professor-disciplina existentes na solução.

		d_1	d_2	d_3	d_4	$f(x)$
S_1	p_1	×	×			7
	p_2			×	×	
S_2	p_1	×		×		95
	p_2		×		×	
S_3	p_1	×			×	94
	p_2		×	×		
S_4	p_1		×	×		95
	p_2	×			×	
S_5	p_1		×		×	94
	p_2	×		×		
S_6	p_1			×	×	182
	p_2	×	×			

Tabela 5.2: *Exemplo de má distribuição de preferências.*

A partir dessa consideração, uma instância B foi elaborada, para qual as preferências foram redistribuídas seguindo um critério mais rígido. À cada professor, foram estabelecidas preferências para um conjunto de 9 disciplinas (obrigatoriamente: 3 com *alta preferência*, 3 com *média preferência* e 3 com *média-baixa preferência*). O objetivo dos testes com a instância B é constatar a influência dos dados relativos às preferências na qualidade da solução gerada para o PAPD.

A quantidade de preferências especificadas, neste caso, subiu para 448, e a média de preferências por disciplina se alterou para 4. O importante nesta modificação da instância é que a distribuição das preferências se tornou mais consistente, o que de certo modo facilitou a resolução do problema e implicou na obtenção de melhores soluções (segundo o atendimento das preferências).

A Tabela 5.3 compara a qualidade das soluções obtidas para as instâncias A e B. As colunas **1**, **2** e **3** indicam a quantidade de relações professor-disciplina com preferências 1, 2 e 3, respectivamente. A coluna φ indica a quantidade de relações estabelecidas entre pares professor-disciplina sem preferências pré-definidas (caso em que um professor é associado a uma disciplina que ele não indicou entre suas preferidas: quanto mais relações desse tipo menor a qualidade da solução). A última coluna indica o valor da função objetivo para a solução em questão.

Na solução A (referente à instância A), existe uma quantidade alta de relações professor-disciplina de *alta preferência*, no entanto, também existe uma quantidade alta de relações de baixa qualidade. Já na solução B (referente à instância B), a situação se modifica, a quantidade de relações professor-disciplina de *alta preferência* diminui, contudo, a quantidade de disciplinas indicadas na quarta coluna também sofre decréscimo indicando uma melhoria na solução.

	Preferências				
	1	**2**	**3**	φ	**f(x)**
Solução A	74	13	6	10	1018
Solução B	57	12	30	3	424

Tabela 5.3: *Quantidade de disciplinas em cada categoria de preferências. Solução 1 e 2 são soluções para a primeira e segunda instâncias, respectivamente.*

Essa comparação deixa explícita a importância da qualidade dos dados de entrada na obtenção de boas soluções, quanto mais coerentemente as preferências entre professores e disciplinas forem descritas, mais provavelmente boas soluções serão obtidas para o PAPD e consequentemente para o PAPDH.

5.3.2 PAPD

Em relação aos métodos implementados alguns testes foram executados como forma de avaliar itens como tempo e a qualidade das soluções geradas. As cinco abordagens implementadas para o PAPD foram consideradas, três algoritmos heurísticos que se diferenciam pela forma de geração de uma solução inicial: Solução inicial gulosa (SIG), Solução inicial baseada em relaxação linear e arredondamento guloso de variáveis não-inteiras (SIRLG) e Solução inicial baseada em relaxação linear e arredondamento baseado em escolha ponderada de variáveis não-inteiras (SIRLEP). E duas implementações de um algoritmo de *Branch-and-Cut* utilizando os softwares CPLEX 9.0 e GLPK.

A Tabelas 5.4 e 5.5 descrevem o desempenho de cada um desses algoritmos em relação à resolução do problema representado pela instância A e pela instância B, respectivamente. É importante ressaltar que o *Branch-and-Cut* GLPK não foi executado até o final, um limite de tempo foi estipulado de acordo com o tempo gasto pelas abordagens heurísticas.

	Preferências					
	1	**2**	**3**	φ	$f(x)$	**Tempo(s)**
SIRLG	55	18	2	28	2617	114,42
SIG	59	13	5	26	2440	68,96
SIRLEP	39	29	9	25	2375	31,68
GLPK	65	12	8	18	1733	60,14[1]
CPLEX	74	13	6	10	1018	3,84

Tabela 5.4: *Soluções para o PAPD da instância A.*

	Preferências					
	1	**2**	**3**	φ	$f(x)$	**Tempo(s)**
SIRLG	32	28	12	30	2825	56,06
SIRLEP	39	29	9	25	2375	31,68
SIG	39	30	14	19	1852	40,71
GLPK	39	38	14	11	1148	60,17[1]
CPLEX	58	30	12	3	424	0,31

Tabela 5.5: *Soluções para o PAPD da instância B.*

Como pode ser observado, os métodos exatos obtiveram melhores resultados que as meta-heurísticas implementadas. No entanto, muitas possibilidades de melhoria para tais algoritmos (baseadas no PGA) não foram exploradas, visto a limitação de tempo [18, 38]. Contudo, em alguns casos, como visto mais adiante nesta seção, os algoritmos heurísticos obtiveram melhores resultados.

5.3.3 PAPDH

Neste caso, a diferença entre o tempo de execução do algoritmo implementado utilizando CPLEX 9.0 e o GLPK não foi perceptível, apesar de relativamente grande. A Tabela 5.6 descreve o tempo gasto por cada uma dessas implementações na resolução da instância A, é considerado que uma solução ótima para o PAPD foi obtida previamente.

[1]Execução com limite de tempo.

Algoritmo	f(x)	Tempo (s)
`GLPK`	140217	0,203
`CPLEX 9.0`	140217	0,062

Tabela 5.6: *Soluções 1 e 2, para a instância `A`, foram obtidas utilizando `GLPK` e `CPLEX 9.0`, respectivamente.*

O `PAPDH` e o problema das múltiplas turmas

Os valores altos para a função objetivo se devem ao fato de muitas disciplinas estarem com uma alta carga horária, e desta forma não haver espaço dentro do turno preferencial da disciplina. Como cada aula alocada fora do turno preferencial adiciona uma alta penalização ao valor da função objetivo, pode-se concluir pelo alto valor obtido que existem aulas fora do turno preferencial.

Essa situação se deve a uma inconsistência do modelo na representação da situação real do `INF`, tal problema foi identificado tardiamente e ainda não pôde ser corrigido. O que ocorre é que disciplinas subdivididas em turmas constituem um caso especial em que disciplinas do mesmo *curricula* podem ser alocadas a horários simultâneos. A Tabela 5.7 exemplifica a situação em que a disciplina *Algoritmos* foi dividida em duas turmas: A e B, as quais foram alocadas para o mesmo horário e dia, terça-feira das 8 às $9:40$hs. Esta situação não causa conflitos (desde que as aulas sejam em salas diferentes), pois o conjunto de alunos matriculados em cada uma das turmas não possuem interseção.

<table>
<tr><th></th><th></th><th colspan="5">Dias da semana</th></tr>
<tr><th></th><th></th><th>2^a</th><th>3^a</th><th>4^a</th><th>5^a</th><th>6^a</th></tr>
<tr><th rowspan="2">Horários</th><td rowspan="2">08:00-09:40</td><td rowspan="2"></td><td>Algoritmos(A)
prof. A</td><td rowspan="2"></td><td>Lógica(B)
prof. A</td><td></td></tr>
<tr><td>Algoritmos(B)
prof. B</td><td>Lógica(A)
prof. B</td><td></td></tr>
</table>

Tabela 5.7: *Disciplinas e múltiplas turmas.*

Como tal situação não foi tratada pela abordagem descrita na seção 4, ocorre que em alguns casos não existem horários suficientes para a alocação de todas as aulas de determinados *curricula*. Como a alocação fora de tais horários é contabilizada na função objetivo, tem-se altos valores para ela. A Tabela 5.8 descreve uma alocação realizada pelo *software* na situação de múltiplas turmas descrita acima.

Dias da semana

Horários	2^a	3^a	4^a	5^a	6^a
15:40 17:50			Lógica(B) ***prof. D***		Lógica(B) ***prof. D***
18:50 20:20	Cálc. 1 ***prof. B***	Lógica(A) ***prof. B***	Cálc. 1 ***prof. B***	Lógica(A) ***prof. B***	Inglês ***prof. C***
20:30 22:00	Algoritmos ***prof. A***	Inglês ***prof. C***	Física ***prof. A***	Algoritmos ***prof. A***	Física ***prof. C***

Tabela 5.8: *Por não permitir a alocação em horários simultâneos de disciplinas com múltiplas turmas, não houve espaço suficiente para alocar todas as aulas em turno noturno e algumas foram alocadas no vespertino, penalizando a função objetivo.*

Considerando a questão de múltiplas turmas, as disciplinas poderiam ser ministradas por diferentes professores e salas em horários simultâneos e deste modo, haveria horários para todas, assim como descrito pela Tabela 5.9.

Dias da semana

<table>
<tr><th>Horários</th><th>2^a</th><th>3^a</th><th>4^a</th><th>5^a</th><th>6^a</th></tr>
<tr><td rowspan="2">18:50
20:20</td><td rowspan="2">Cálc. 1
prof. D</td><td>Lógica(A)
prof. B</td><td rowspan="2">Cálc. 1
prof. D</td><td>Lógica(A)
prof. B</td><td rowspan="2">Inglês
prof. C</td></tr>
<tr><td>Lógica(B)
prof. E</td><td>Lógica(B)
prof. E</td></tr>
<tr><td>20:30
22:00</td><td>Algoritmos
prof. A</td><td>Inglês
prof. C</td><td>Física
prof. A</td><td>Algoritmos
prof. A</td><td>Física
prof. C</td></tr>
</table>

Tabela 5.9: *Múltiplas turmas em horários simultâneos. Tabela gerada manualmente.*

O caso de múltiplas turmas é um requisito de extrema importância, desta forma a utilização prática da abordagem proposta, quanto à resolução do `PAPDH`, fica condicionada à correção da situação descrita.

O `PAPDH` sem múltiplas turmas

Como forma de descrever uma solução completa para o problema (sem múltiplas turmas), uma instância `C` foi criada. Para a criação de tal instância foram removidas todas as múltiplas turmas das disciplinas, essa alteração diminui o total de horas-aula a serem ministradas, desta forma, foram necessárias mudanças na carga horária de trabalho mínima de cada professor com o objetivo de atender à restrição de viabilidade 4-12.

As modificações foram feitas sobre a instância original do problema (instância `A`), e se basearam no seguinte critério: professores com carga horária máxima maior ou igual a 12 tiveram sua carga horária mínima diminúida em 4 horas semanais, professores com carga horária máxima igual a 10 tiveram sua carga horária mínima diminuída para 8, os demais professores não tiveram suas cargas horárias alteradas. Estas alterações poderiam ter sido feitas segundo outros critérios, no entanto, como o objetivo é utilizar a instância `C` para comprovar a utilizade da abordagem proposta para o `PAPDH`, este não é um fator de muita importância.

A Tabela 5.10 descreve o resultado dos algoritmos propostos na resolução do `PAPD` dessa nova instância. O que pôde ser observado, é que as abordagens heurísticas obtiveram melhores resultados para esta instância do que para as outras, provavelmente como consequência da alteração das cargas horárias dos professores.

Nas duas primeiras instâncias, as cargas horárias mínimas e máximas de cada professor eram iguais, o que dificultava a resolução do problema de forma heurística (observado empiricamente). Nesta nova instância, nem todos os valores de carga horária mínima e máxima são iguais e uma melhoria ocorreu, de fato o algoritmo `SIRLG` até mesmo obteve melhor resultado que o `GLPK` e em um tempo bem menor.

	Preferências					
	1	**2**	**3**	φ	$f(x)$	**Tempo(s)**
`SIRLEP`	57	12	5	15	1446	142,67
`SIG`	57	13	4	15	1445	116,75
`GLPK`	62	7	6	14	1354	140,15*
`SIRLG`	59	14	2	14	1353	57,73
`CPLEX`	63	12	4	10	999	5,625

Tabela 5.10: *Soluções para o `PAPD` da instância `C`.*

Já a Tabela 5.11, descreve os resultados na resolução do `PAPDH`. O interessante a ser observado é que os valores da função objetivo são bem menores que os descritos na Tabela 5.6. Isto ocorre porque neste caso não existiram penalizações quanto a alocações de aulas fora do turno dos cursos, confirmando que o problema com a instância anterior era realmente a questão de disciplinas em múltiplas turmas.

Algoritmo	**f(x)**	**Tempo (s)**
GLPK	128	0,281
CPLEX 9.0	128	0,062

Tabela 5.11: *Resultados utilizando GLPK e CPLEX 9.0 na resolução do PAPDH da instância C.*

Alunos Formandos

Os últimos experimentos realizados ao PAPDH se direcionaram às restrições oriundas de requisições de alunos formandos. Os dados disponíveis para os testes descrevem um conjunto de 35 alunos do curso de Ciências da Computação, cada um deles com um conjunto de disciplinas as quais não deveriam ser alocadas a horários simultâneos, como forma de permitir a eles concluir o curso no tempo previsto.

Testes foram realizados com as instâncias A e C, e os resultados podem ser observados, respectivamente, nas Tabelas 5.12 e 5.13.

Algoritmo	**f(x)**	**Tempo (s)**
GLPK	140217	0,32
CPLEX 9.0	140217	0,09

Tabela 5.12: *Resultados utilizando GLPK e CPLEX 9.0 na resolução do PAPDH da instância A.*

Algoritmo	**f(x)**	**Tempo (s)**
GLPK	128	0,61
CPLEX 9.0	128	0,09

Tabela 5.13: *Resultados utilizando GLPK e CPLEX 9.0 na resolução do PAPDH da instância C.*

O interessante a ser observado nestas duas tabelas é que, independentemente da instância considerada, todas as restrições de alunos formandos puderam ser atendidas, sem alteração da qualidade da solução, como pode ser observado pelo não acréscimo do valor da função objetivo. É importante, no entanto, ressaltar que o conjunto de disciplinas considerado não contém as disciplinas *externas* do INF, que em determinados *curricula* possuem alta carga horária.

5.4 Considerações finais

Neste capítulo foram descritas as peculiaridades do processo de agendamento de aulas na UFG, englobando aspectos como *oferta de disciplinas*, *tipos de disciplinas* e a estrutura organizacional da instituição. Na Seção 5.3 os resultados obtidos com as abordagens sugeridas no Capítulo 4 são descritos, assim como alguns pontos críticos que devem ser tratados em trabalhos futuros.

CAPÍTULO 6

Conclusões

Este trabalho foi desenvolvido com o intuito de estimular a pesquisa em relação aos problemas de programação de horários na UFG e propor uma abordagem eficaz na resolução do problema de alocação de disciplinas e alocação de horários. Os algoritmos propostos para a resolução do problema foram desenvolvidos de acordo com detalhes específicos do agendamento de aulas na UFG. A divisão em subproblemas terá como consequência a descentralização de grande parte do processo, a idéia principal é, neste caso, que cada unidade acadêmica se responsabilize pela alocação de professores e alocação de horários, sendo a alocação de salas (problema não tratado neste trabalho) uma etapa posterior.

Obviamente, toda essa integração envolve detalhes que extrapolam o foco do trabalho descrito neste texto, no entanto, acredita-se que os problemas aqui abordados são um bom ponto de partida para diversas pesquisas relacionadas à resolução automatizada do problema de alocação geral, enfrentado na UFG. Existe, no entanto, a necessidade de se avaliar a robustez e a adequação da abordagem proposta (divisão em subproblemas) através de testes com instâncias maiores do problema.

Em um cenário ideal, cada unidade acadêmica utilizaria ferramentas computacionais para o auxílio na elaboração dos seus quadros de horários, especificando dias e horários para todas as aulas de cada disciplina. Em seguida um ponto centralizador seria utilizado para a alocação de salas de aulas à todos os cursos, considerando os vários detalhes envolvidos nesse problema. Em uma situação bastante sofisticada, e dependente da qualidade das soluções obtidas para os problemas, o *software* centralizador poderia até mesmo, em caso de de conflitos, requisitar diretamente ao *software* da unidade em questão a elaboração de um outro quadro de horário mais flexível.

Apesar de fictício, esse cenário não é impossível de ser alcançado, cada pequena parte de um sistema tão complexo possui qualificação para se tornar foco de pesquisas, bastando para isso, pessoas e investimentos a longo prazo. Dentro do contexto deste trabalho, é possível descrever alguns pontos de interesse para trabalhos futuros.

Diversos algoritmos foram implementados para a resolução do PAPD, porém, ao longo do processo de desenvolvimento de tais algoritmos uma importante informação esteve ausente, o fato do PAPD ser uma especialização do *Problema Generalizado de Atribuição* (PGA). Desta forma uma abordagem heurística foi implementada, a qual pode ser útil na obtenção de boas soluções, mas que no entanto, não utilizou do conhecimento existente para os algoritmos de resolução do PGA. Ou seja, seria interessante a aplicação de métodos originalmente desenvolvidos para a resolução do PGA, na resolução do PAPD, ou o inverso, explorar a utilização de algoritmos desenvolvidos para o PAPD na resolução de instâncias do PGA, tanto nas meta-heurísticas quanto nos algoritmos exatos [38, 7].

O PGA possui um conjunto de instâncias de teste que poderiam ser utilizadas na avaliação dos algoritmos, o que é um ponto importante, visto que no caso do PAPD as instâncias tiveram que ser criadas, um processo bastante tedioso. Instâncias do PGA estão disponíveis no *site* `http://people.brunel.ac.uk/~mastjjb/jeb/orlib/gapinfo.html`.

Os algoritmos exatos para o PAPD e o PAPDH foram implementados em duas versões, utilizando para isso o GLPK e o CPLEX 9.0. A versão utilizando CPLEX 9.0 se mostrou eficaz na resolução dos dois problemas, obtendo soluções ótimas em pouco tempo. Já a versão implementada utilizando GLPK, obteve desempenho insatisfatório na resolução do PAPD, o tempo necessário para a obtenção de uma solução ótima foi muito grande para as instâncias testadas, porém, no caso do PAPDH o desempenho foi satisfatório.

Especificamente em relação ao PAPDH também existem diversos pontos que poderiam ser atacados em trabalhos futuros, um dos principais é a questão do objetivo da otimização. Neste trabalho o problema foi atacado através de um único objetivo: minimizar a quantidade aulas alocadas fora do turno do curso. No entanto, diversos outros objetivos poderiam ter sido considerados em uma abordagem multi-objetivo: garantir o mínimo de aulas fora dos turnos previstos para o curso, garantir o mínimo de conflitos para alunos formandos, garantir o atendimento de requisitos pedagógicos, dentre outros. Essa abordagem permitiria maior flexibilidade à resolução das instâncias, podendo ser implementada através de relaxação lagrangeana [21].

Este trabalho oferece uma visão geral dos problemas de programação de horários, descreve categorias de problemas bem conhecidos, alguns métodos importantes na resolução do problema e oferece uma abordagem de resolução para os subproblemas de alocação de professores e alocação de horários enfrentado na UFG. Espera-se que com o estudo aqui descrito este trabalho possa ser continuado, de forma a atingir maturidade suficiente para estimular o desenvolvimento e a utilização de *software* na resolução do problema do agendamento semanal de aulas na UFG.

Referências Bibliográficas

[1] Aubin, J.; Ferland, J. **A large scale timetabling problem.** *Computers & Operations Research*, 16(1):67–77, 1989.

[2] Bertsimas, D.; Tsitsiklis, J. **Introduction to Linear Optimization**. Athena Scientific Belmont, MA, 1997.

[3] Burke, E.; Cowling, P.; Keuthen, R. **Effective local and guided variable neighbourhood search methods for the asymmetric travelling salesman problem**. *Applications of Evolutionary Computing*, p. 203–212.

[4] Burke, E.; Elliman, D.; Weare, R. **A genetic algorithm based university timetabling system**. In: *East-West Conference on Computer Technologies in Education, Crimea, Ukraine*, p. 35–40, 1994.

[5] Carter, M.; Laporte, G.; Chinneck, J. **A general examination scheduling system**. *Interfaces*, 24(3):109–120, 1994.

[6] Carter, M.; Tovey, C. **When is the classroom assignment problem hard?** *Operations Research*, p. 28–39, 1992.

[7] Cattrysse, D.; Salomon, M.; Van Wassenhove, L. N. **A set partitioning heuristic for the generalised assignemnt problem**. *European Journal of Operational Research*, 72(1):167–174, 1994.

[8] Chu, P.; Beasley, J. **A genetic algorithm for the multidimensional knapsack problem**. *Journal of Heuristics*, 4(1):63–86, 1998.

[9] Come, D.; Ross, P.; Fang, H. **Fast practical evolutionary timetabling**. *Lecture Notes in Computer Science*, 865:250–263, 1994.

[10] Congram, R. K. **Polynomially searchable exponential neighbourhoods for sequencing problems in combinatorial optimisation**. PhD thesis, University of Southampton, Department of Mathematics, 2000.

[11] Cooper, T.; Kingston, J. **The complexity of timetable construction problems**. In: *Practice and Theory of Automated Timetabling*, volume 1153 de **Lecture Notes in Computer Science**, p. 281–295. Springer-Verlag, 1996.

[12] Cooper, T.; Kingston, J. **The solution of real instances of the timetabling problem**. *The Computer Journal*, 36(7):645–653, 1993.

[13] Cormen, T. H.; Leiserson, C. E.; Rivest, R. L.; Stein, C. **Introduction to Algorithms**. MIT Press, 2nd edition, 2001.

[14] Dantzig, G.; Orden, A.; Wolfe, P. **The generalized simplex method for minimizing a linear form under linear inequality restraints**. *The basic George B. Dantzig*, p. 33, 2003.

[15] De Werra, D. **An introduction to timetabling**. *European Journal of Operational Research*, 19(2):151–162, 1985.

[16] Eglese, R.; Rand, G. **Conference seminar timetabling**. *Journal of the Operational Research Society*, 38(7):591–598, 1987.

[17] Even, S.; Itai, A.; Shamir, A. **On the complexity of timetable and multicommodity flow problems**. *SIAM Journal on Computing*, 5(4):691–703, 1976.

[18] Feltl, H.; Raidl, G. **An improved hybrid genetic algorithm for the generalized assignment problem**. In: *Proceedings of the 2004 ACM Symposium on Applied Computing*, p. 990–995, 2004.

[19] Feo, T.; Resende, M. **Greedy randomized adaptive search procedures**. *Journal of Global Optimization*, 6(2):109–133, 1995.

[20] Ferland, J.; Roy, S. **Timetabling problem for university as assignment of activities to resources**. *Computers & Operations Research*, 12(2):207–218, 1985.

[21] Fischer, M. L. **The Lagrangian relaxation method for solving integer programming problems**. *Management Science*, 50(12):1861–1871, 2004.

[22] Fischetti, M.; Lodi, A. **Local branching**. *Mathematical Programming*, 98(1):23–47, 2003.

[23] Garey, M.; Johnson, D. **Computers and intractability: A guide to $\mathcal{NP}$-completeness**, 1979.

[24] Glover, F. **Tabu search–part I**. *INFORMS Journal on Computing*, 1(3):190, 1989.

[25] Glover, F. **Tabu search–part II**. *INFORMS Journal on Computing*, 2(1):4, 1990.

[26] Gomory, R. **Outline of an algorithm for integer solutions to linear programs**. *Bulletin of the American Mathematical Society*, 64(5):275–278, 1958.

[27] GR Filho, L. **Constructive genetic algorithm and column generation: an application to graph coloring**. In: *Proc. Asia Pacific Operarions Research Symposium APORS*, 2000.

[28] Hertz, A. **Tabu search for large scale timetabling problems**. *European Journal of Operational Research*, 54(1):39–47, 1991.

[29] Hertz, A. **Finding a feasible course schedule using tabu search**. *Discrete Applied Mathematics*, 35(3):255–270, 1992.

[30] Hopcroft, J.; Karp, R. **An $n^{5/2}$ Algorithm for Maximum Matchings in Bipartite Graphs**. *SIAM Journal on Computing*, 2:225, 1973.

[31] Johnson, D.; Aragon, C.; McGeoch, L.; Schevon, C. **Optimization by simulated annealing: an experimental evaluation; part II, graph coloring and number partitioning**. *Operations Research*, p. 378–406, 1991.

[32] Junginger, W. **Timetabling in germany: a survey**. *Interfaces*, p. 66–74, 1986.

[33] Laporte, G.; Desroches, S. **The problem of assigning students to course sections in a large engineering school.** *Computers & Operations Research*, 13(4):387–394, 1986.

[34] Lara, B. **Alocação de professores em instituições de ensino superior: Um modelo matemático para o problema de único campus e para o multicampi**. *SBPO*, 2007.

[35] Lü, Z.; Hao, J. **Adaptive tabu search for course timetabling**. *European Journal of Operational Research*, 200(1):235–244, 2010.

[36] Makhorin, A. **GNU Linear Programming Kit**. Department for Applied Informatics, Moscow Aviation Institute, Moscow, Russia, 2008.

[37] Michiels, W.; Aarts, E.; Korst, J. **Theoretical aspects of local search**. Springer-Verlag New York Inc, 2007.

[38] Osman, I. **Heuristics for the generalised assignment problem: simulated annealing and tabu search approaches**. *OR Spectrum*, 17(4):211–225, 1995.

[39] Osman, I.; Laporte, G. **Metaheuristics: A bibliography**. *Annals of Operations Research*, 63(5):511–623, 1996.

[40] Papadimitriou, C.; Steiglitz, K. **Combinatorial optimization: algorithms and complexity**. Dover Pubns, 1998.

[41] Puchinger, J.; Raidl, G. **Combining metaheuristics and exact algorithms in combinatorial optimization: A survey and classification**. *Artificial Intelligence and Knowledge Engineering Applications: a Bioinspired Approach*, p. 41–53, 2005.

[42] Raidl, G. **An improved genetic algorithm for the multiconstrained 0-1 knapsack problem**. In: *Proceedings of the 5th IEEE International Conference on Evolutionary Computation*, p. 207–211, 1998.

[43] Rosa, D. E. G.; de Oliveira, E. F. **Guia do Estudante, Graduação**. Universidade Federal de Goiás, 2010.

[44] Sahni, S.; Gonzalez, T. **P-complete approximation problems**. *Journal of the ACM (JACM)*, 23(3):555–565, 1976.

[45] Santos, H. G.; Souza, M. J. F. **Programação de horários em instituições educacionais: Formulações e algoritmos**. *SBPO*, 2007.

[46] Schaerf, A. **A survey of automated timetabling**. *Artificial Intelligence Review*, 13(2):87–127, 1999.

[47] Schmidt, G.; Strohlein, T. **Timetable construction-an annotated bibliography**. *The Computer Journal*, 23(4):307–316, 1980.

[48] Talukdar, S.; Baerentzen, L.; Gove, A.; De Souza, P. **Asynchronous teams: Co-operation schemes for autonomous agents**. *Journal of Heuristics*, 4(4):295–321, 1998.

[49] Talukdar, S.; Murthy, S.; Akkiraju, R. **Asynchronous teams**. *Handbook of Metaheuristics*, p. 537–556, 2003.

[50] Tripathy, A. **Computerised decision aid for timetabling: a case analysis**. *Discrete applied mathematics*, 35(3):313–323, 1992.

[51] Willemen, R.; Aarts, P.; Lenstra, J. **School timetable construction: Algorithms and complexity**. *Technische Universiteit Eindhoven*, 2002.

[52] Woodruff, D. **A chunking based selection strategy for integrating meta-heuristics with branch and bound**. In: *MIC-97: Metaheuristics International Conference*, p. 499–511, 1999.

[53] Yoshikawa, M.; Kaneko, K.; Yamanouchi, T.; Watanabe, M. **A constraint-based high school scheduling system**. *IEEE Expert*, 11(1):63–72, 1996.