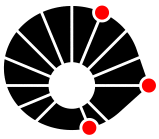

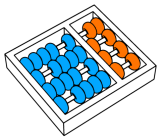

Modelo PLI para alocação de professores do Instituto de Computação da Unicamp

R. C. de Freitas *F. L. Usberti*

Relatório Técnico - IC-PFG-19-55
Projeto Final de Graduação
2019 - Dezembro

UNIVERSIDADE ESTADUAL DE CAMPINAS
INSTITUTO DE COMPUTAÇÃO

Modelo PLI para alocação de professores do Instituto de Computação da Unicamp

Rodrigo Ceccato de Freitas* Fábio Luiz Usberti†

Resumo

A construção de bons quadros de horário é de extrema importância para o uso adequado de recursos e satisfação dos integrantes de uma organização. Esse tipo de tabelamento é utilizado em diversos ambientes, como escolas, universidades, empresas, organização de eventos ou linhas de transporte público. Dada essa diversidade, mesmo dentro de um único campo prático, cada modelagem precisa considerar características específicas da instituição onde é aplicada. Adicionalmente, diversas restrições tem aspectos complicados de se expressar matematicamente, fazendo com que resolvedores automatizados se beneficiem de interfaces interativas para ajustes posteriores e facilidade de re-execução. Este trabalho apresenta uma solução automatizada e interativa para a alocação de professores em disciplinas para o Instituto de Computação da Unicamp. Essa solução permite que o usuário altere o quadro com facilidade, utiliza os dados originais de forma direta e é solucionado, com o resolvedor *Gurobi*, em menos de meio segundo. Neste projeto é levado em conta restrições de horários dos docentes, preferência de cada docente em lecionar cada disciplina e horário no qual cada curso é ministrado.

1 Introdução

Diversos modelos e implementações de soluções para o problema de tabelamento de horários foram e continuam sendo formulados [1]. Isso porque há diversos problemas práticos correspondentes às modelagens propostas, como agendamento de horários para alunos, professores, funcionários [2], linhas de transporte público [3, 4], entre outros [5].

Por se tratar de um problema de aplicações abrangentes, há diversas terminologias e variações. Além disso, há variações dentro de um mesmo campo prático, como modelos que além de elaborar horários não conflitantes, levam em conta recursos físicos disponíveis para a execução das atividades, como número de salas de aula ou quantidade de veículos [6]. Dessa forma, há diversas maneiras de se classificar um modelo de tabelamento de horários.

Em diversos casos, os problemas de tabelamento de horário são resolvidos manualmente [7–10]. Isso acontece porque os esforços empregados para a resolução do problema nem

*Graduando em Engenharia de Computação, Instituto de Computção, Universidade Estadual de Campinas.

†Professor Doutor do Instituto de Computação, Universidade Estadual de Campinas. Av. Albert Einstein, 1251. Cidade Universitária. Caixa Postal 6176, CEP 13083-852, Campinas, SP - Brasil.

sempre são facilmente transmitidos para uma representação formal de restrições. Também pode ocorrer de parte das restrições terem caráter quase subjetivo, o que exige que o usuário possa alterar o modelo em tempo de execução. Isso faz com que, em boa parte dos casos, *solvers* comerciais não se adaptem às particularidades de cada caso [11].

Apesar de sua difícil modelagem, a solução manual, em muitos casos, está chegando ao limite de sua capacidade, ao mesmo tempo em que o volume de dados a ser interpretado é crescente. Dessa forma, é cada vez mais interessante a adoção de formas automatizadas de se realizar o tabelamento de horários. Outra vantagem do uso de sistemas computacionais nesse tipo de problema é a capacidade de se levar em conta um grande número de restrições. Com isso, é possível explorar de forma mais conveniente os recursos disponíveis.

Historicamente, há momentos de pico em tópicos de *timetabling* [12], relacionados a novas modelagens, algoritmos e aumento de poder computacional, tornando viáveis propostas anteriores [11]. Em diversas asserções da literatura, os sistemas automatizados, ou simplesmente de auxílio visual para tabelamento, foram propostos em momentos nos quais o trabalho era feito de forma totalmente manual, sendo lento, custoso e frequentemente não-ótimo. Em certos casos, a parte mais computacionalmente intensa do problema, isto é, resolver o modelo em busca de otimalidade ou factibilidade, não é o maior desafio [8]. Na verdade, a adversidade está em integrar o modelo ao caso de uso da instituição, tanto no recebimento dos dados quanto nas restrições e construção da interface com o usuário. Nessas circunstâncias, há trabalhos que compilam informações de grande utilidade prática para se construir uma proposta para a própria corporação [13]. E, em boa parte dos casos, as propostas que passaram a ser de fato utilizadas foram construídas estritamente para uma dada organização [14, 15].

1.1 Terminologias

Os tipos de problemas de tabelamento de horários podem receber terminologias que variam de acordo com o autor. Ainda assim, há uma coerência razoável nos diferentes grupos formados. Dessa forma, a tabela 1 compila alguns modelos e suas descrições, com uma divisão fortemente baseada na proposta de classificação feita por Schaerf [6].

Vale notar que há algumas restrições comuns aos tipos de problema. A grande maioria das formulações são NP-completo [4, 7, 9, 10, 15–24] por conta das restrições. Essas são divididas, por boa parte dos autores [12, 17–19, 21, 25–37, 37–47], em *hard constraints* e *soft constraints*. As *soft constraints* são desejáveis, mas não obrigatórias, correspondendo, por

Tabela 1: Categorias de problema de tabelamento de horário

Horário escolar	Horário semanal de todas as aulas de uma escola, evitando que dois professores sejam alocados a uma mesma turma no mesmo horário e vice-versa.
Horário de curso	Horário semanal dos cursos de uma universidade, minimizando a intersecção de aulas que são atendidas por alunos em comum.
Horário de exames	Agendamento de provas em datas ao longo do período letivo, evitando conflito de horário e proximidade de datas para cada aluno.

exemplo, à ausência de espaços sem atividade entre as aulas de um dado docente. Essa característica geralmente é desejável, mas sua ausência não inviabiliza o quadro gerado. Já as restrições chamadas de *hard constraints* devem obrigatoriamente ser contempladas e correspondem a aspectos que definem a viabilidade do tabelamento. Não permitir que um professor seja alocado a duas aulas distintas em um mesmo horário é um exemplo desse tipo de restrição.

1.2 Trabalhos relacionados

Conforme informado no início da Seção 1, há diversos trabalhos com diferentes abordagens para o tabelamento de horários. As principais variações mostradas na Seção 1.1 incluem: tempo de solução, presença de interação com o usuário, número de restrições e natureza da instituição.

Pistori et. al [48] propõem aplicação colaborativa chamada SICH (Sistema Integrado de Confecção de Horários). Essa aplicação não utiliza nenhum resolvedor automático para a formação do quadro, ainda assim, reduziu o tempo necessário para se realizar o tabelamento na instituição onde foi aplicado (Universidade Católica Dom Bosco) de três meses para poucas semanas.

Queiroz e Nepomuceno [7] resolvem um modelo de PLI (programação linear inteira) para alocação de disciplinas no curso de Ciência da Computação da Universidade de Fortaleza. Foram obtidos bons resultados rapidamente com o resolvedor *CPLEX*.

Além das métricas citadas em outros trabalhos, há outras abordagens interessantes que utilizam outra perspectiva para mensurar qualidade. Alves B. e Laura S. [47], por exemplo, elaboraram um modelo de PLI para aplicação de recursos em Instituições de Ensino Superior no Brasil. Nesse caso, o modelo soluciona a alocação da carga horária de cursos de graduação, de acordo com as exigências do Ministério da Educação. Essas exigências compreendem três dimensões, envolvendo a organização didático-pedagógica, o corpo docente e as instalações.

Apesar de diversos trabalhos resolverem em tempo razoável e encontrarem uma solução ótima para problemas de *timetabling*, este é NP-Completo [12,17–19,21,25–37,37–47]. Sendo assim, há algumas considerações de performance que devem ser feitas. Há, por exemplo, exigências que requerem um grande número de restrições, como, por exemplo, para garantir compacidade dos horários dos docentes [33]. Dentro desse tipo de problema, há trabalhos que tratam especialmente desse tipo de restrição, ou seja, estudam formas de incorporar a busca por tabelamentos com poucos períodos intermediários sem atividade [10, 49].

Tendo em conta os modelos supracitados, é importante notabilizar que boa parte da literatura não utiliza resolvedores de modelos de PLI para buscar soluções. Em especial, há diversos exemplos de uso de rotinas de coloração de grafos [1,20], algoritmos genéticos [15,42] e outras modelagens com grafos [3], usando, por exemplo, o problema de fluxo e transporte em redes [1, 6, 12, 31, 33, 41, 50, 51].

Assim, nota-se que há uma grande pluralidade de modelagens e implementações, muitas com resultados extremamente promissores. Apesar disso, como comentam B. McCollum et al. [35], infelizmente, boa parte desses resultados não são acompanhados de códigos ou exemplos de instância.

1.3 Objetivo

Este trabalho propõe um programa interativo que auxilia na construção do modelo PLI correspondente à alocação de professores em disciplinas e exibe o resultado de forma amigável após chamar um resolvedor. A solução proposta leva em conta características da formulação de horários do Instituto de Computação. A ideia é fazer com que o modelo corresponda da forma mais conveniente possível aos problemas endereçados pelas pessoas que atualmente resolvem este problema manualmente.

2 Metodologia

Foram construídos modelos que contêm as restrições presentes na elaboração do quadro de horários dos professores do IC-Unicamp. A formulação adotada foi um modelo de PLI (ver Seção 2.3), que posteriormente foi aprimorado com características coletadas sobre a Instituição (ver Seção 2.1 e 2.4). Complementarmente, foi proposto um terceiro modelo que remove a tripla indexação da variável de decisão, possibilitando a criação separada de variáveis para a alocação professor-disciplina e disciplina-horário, modelagem potencialmente mais eficiente (ver Seção 2.5). Adicionalmente, foi construída uma aplicação em Python que, a partir de uma planilha, constrói um dicionário com as informações necessárias para o modelo de PLI, monta esse modelo e chama o resolvedor *Gurobi*[1] para buscar soluções para o mesmo. Além disso, essa aplicação apresenta uma interface gráfica para o usuário visualizar o resultado e, se desejar, fazer alterações para encontrar diferentes respostas ou mesmo viabilizar entradas infactíveis. Os detalhes e decisões de implementação estão detalhados na Seção 2.2.

2.1 Coleta de características

Para adequar o modelo ao Instituto de Computação da Unicamp, foi realizada uma reunião com os responsáveis pelo tabelamento manual. Desse encontro foi possível investigar quais restrições o modelo deveria incorporar para, de forma automática, levar em conta considerações tomadas para montar o quadro de horários do Instituto. Além disso, esse contato mostrou o formato como os dados que compõe os parâmetros do modelo são armazenados e estruturados, permitindo assim adequar a implementação de acordo.

Dessa forma, o modelo inicial, apresentado na Seção 2.3, que contém as restrições mais genéricas presentes em problemas dessa natureza, foi acrescido de restrições específicas do caso tratado. Essa segunda versão do modelo está explicitada na Seção 2.4.

2.1.1 Alocação manual do IC

No Instituto de Computação, os horários das disciplinas de graduação são relativamente fixos, enquanto eletivas tem ligeira flexibilidade e disciplinas de pós-graduação tem grande liberdade de alocação de horário. Apesar disso, devido à quantidade de salas de aula

[1] https://www.gurobi.com/

disponíveis para matérias de pós-graduação, apenas três dessas disciplinas podem ser ministradas em um mesmo horário.

Outra consideração é que, caso a preferência não possa ser atendida, a disciplina ainda deve ser alocada levando-se em conta a especialidade dos professores. Dessa forma, é interessante considerar alocações anteriores e avaliações dos docentes ministrando as matérias em questão, para se avaliar alterações interessantes para o modelo.

Ademais, no IC é feita uma medida de créditos correspondente às horas de aula ministradas. Com isso, é desejado que cada professor lecione uma quantidade de créditos dentro de um determinado intervalo, definido pelos créditos cumpridos nos semestres anteriores.

Finalmente, é preciso respeitar restrição de distribuição de horário de disciplina: por exemplo, *Matéria A* só pode ser ministrada *segunda-feira e quarta-feira de manhã* ou *terça-feira e quinta-feira à tarde*.

A principal mudança associada à consideração dessas observações é a alteração da variável de decisão, que ganha uma nova indexação: o horário t no qual a disciplina d é dada. Porém, como as disciplinas não podem ser livremente aglutinadas ou espalhadas, isto é, ainda que os horários sejam respeitados, uma determinada matéria não pode ser dada contiguamente durante seis horas, por exemplo. Por isso, a variável de decisão, ao invés de ser indexada como x_{pdt} é indexada com $x_{pdc_{dh}}$, de modo que c_{dh} é índice do grupo de horários do conjunto de *timeslots*, formado pelos grupos de *slots* de tempo nos quais ela pode ser ministrada. Ou seja, ao invés de se alocar cada hora-aula, alocam-se grupos de horário pré-determinados.

2.2 Implementação

Conforme indicado na Seção 1, a praticidade de aplicações que montam quadro de horários está fortemente relacionada à sua compatibilidade e integração com os sistemas já utilizados nas instituições que as usarão. Por isso, como no IC os dados de preferências por horário e matéria são armazenados em documentos de planilha ($.CSV$), a aplicação proposta lê diretamente este formato. O formato dessas planilhas (com lista de disciplinas, arco de preferências e arco de horários) está exemplificado na Tabela 2. Esse formato é diretamente compatível com o registro usado atualmente pelo IC para representar essa informação, de modo que a solução proposta pode ser usada com quase nenhuma alteração dos dados.

Em seguida, utilizando o modelo proposto na Seção 2.5, são construídas as restrições que são passadas ao resolvedor *Gurobi*. Este foi escolhido por apresentar bons resultados quando comparados com *softwares* resolvedores similares para modelos de PLI que representam problemas de *timetabling* [21]. A arquitetura do programa está ilustrada na Figura 1.

Para a construção das restrições do modelo PLI proposto (Seção 2.5) o tempo foi discretizado a partir de cada horário de aula, como mostra a Figura 2. Estendendo esse raciocínio para a semana toda (Figura 3), cada horário de disciplina pode ser representado por um vetor binário de 32 posições.

Como há, nos modelos propostos, restrições para evitar que dois professores sejam alocados para ministrar disciplinas em um mesmo horário, é preciso determinar quais quadros de horários tem algum tipo de intersecção. Para evitar sobrecarregar o modelo com verificação, através de variáveis de decisão, a cada *slot* de tempo, é feito pré-processamento

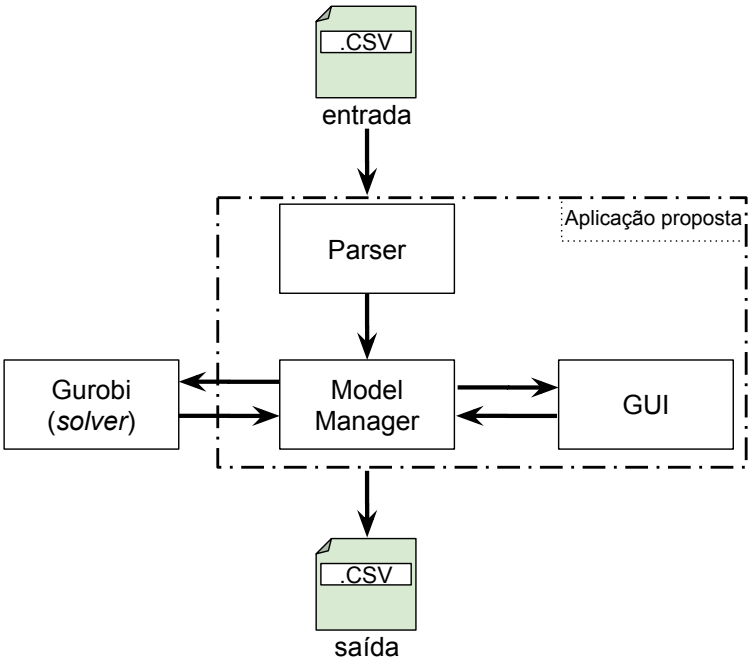

Figura 1: Arquitetura do programa proposto. O *parser* lê os dados fornecidos pelo usuário e os entrega ao *Model Manager*, que constrói o modelo de PLI, além de conter funções para modificação do mesmo. Com isso, com o módulo de interface gráfica (GUI), o usuário pode interagir com o modelo, pedindo alterações ao *Model Manager* e visualizando o resultado. Para a solução do modelo é chamado o resolvedor do *Gurobi*.

para determinar incompatibilidade. Com isso, primeiro são identificados quadros iguais entre cursos, que são indexados uma única vez, como ilustra a Figura 5. Por fim, as restrições que evitam conflito de horário são montadas baseadas nos pares de quadros que têm intersecção, descobertos no pré-processamento. Este processo está ilustrado nas Figuras 4 e 5.

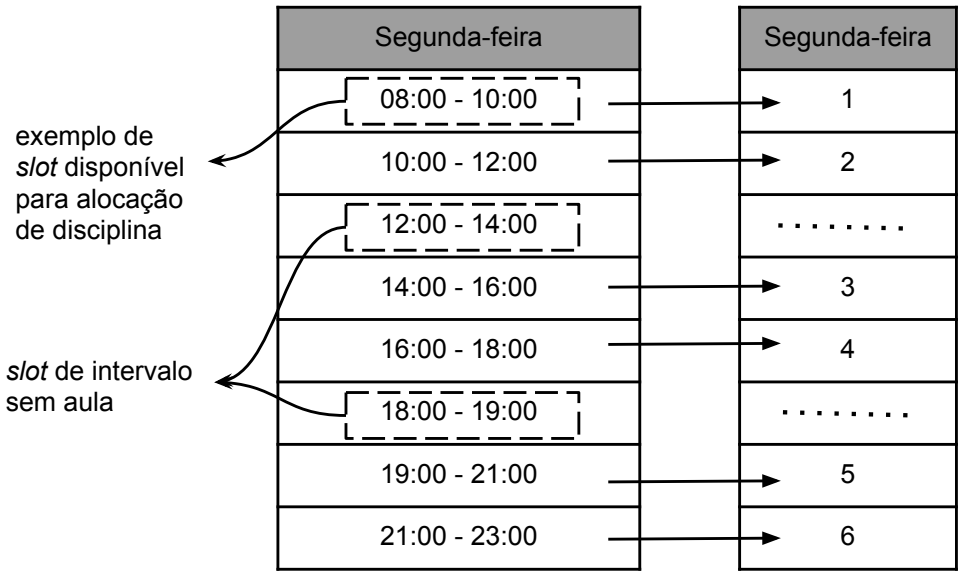

Figura 2: Modo como é feita a discretização do tempo para um dia letivo. Como a unidade mínima de aula dura duas horas, cada *slot* desse é identificado por um número inteiro correspondente à sua posição na semana. Períodos sem aula não são considerados.

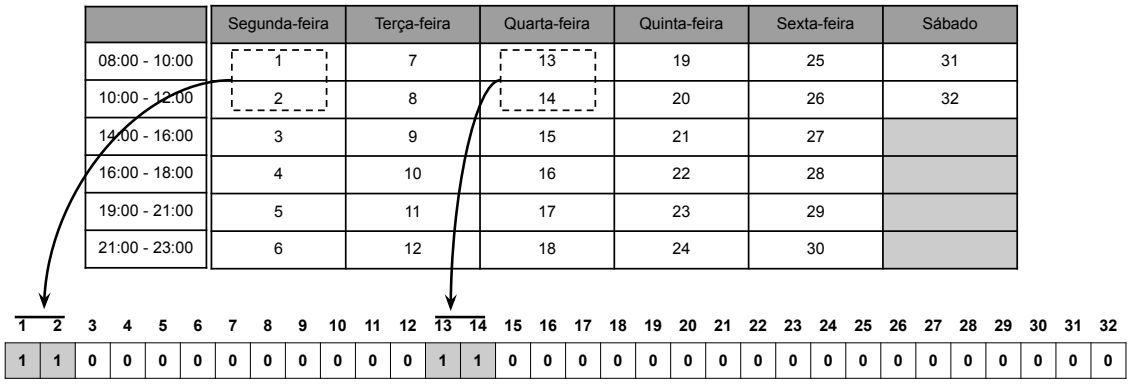

Figura 3: Exemplo de representação de um curso que é ministrado de segunda-feira e quarta-feira das 08h00 horas até as 12h00. Na implementação adotada neste trabalho, o *timeframe* do curso é então escrito como um vetor de binários com o valor *1* nas posições correspondentes aos momentos de aula e *0*, caso contrário.

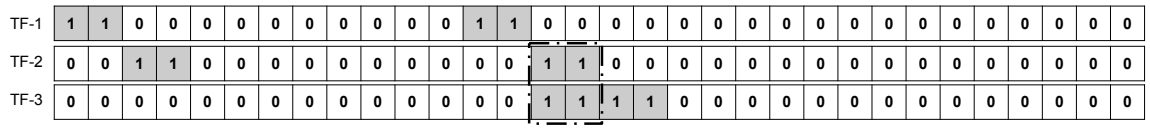

Figura 4: Exemplo de determinação de intersecção (ou conflito) entre *timeframes*. Nos três quadros desta figura, há apenas conflito entre *TF-2* e *TF-3*. Isso significa que um mesmo docente não pode ministrar disciplinas nesses dois quadros simultaneamente.

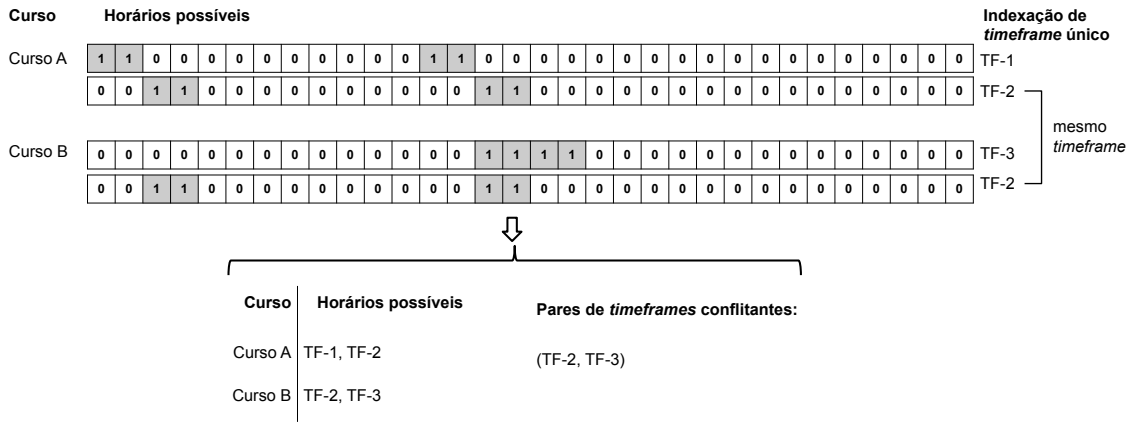

Figura 5: Exemplo de pré-processamento utilizado para identificar quadros de horários conflitantes. Como na entrada cada curso é associado a um conjunto de *timeframes* válidos, o programa primeiro monta uma lista com todos os *timeframes* únicos. Isso facilita o cálculo de horários conflitantes, que precisa ser feito checando par a par quais possuem horários em comum. Finalmente, é indicado quais os índices de quadro de horário correspondentes a cada disciplina e são listados todos os pares de horário conflitante.

Tabela 2: Exemplo de planilha de entrada contendo todas as disciplinas que devem ser ministradas. Nota-se, por exemplo, que uma disciplina pode não ter um horário associado, como "Coordenar MC001". Além disso, uma disciplina pode ter mais de um quadro de oferecimento, como "MC003". Nesse caso, o modelo escolherá exatamente um *timeframe* para a disciplina.

Disciplinas	Arco docente-disciplina-preferência			Arco disciplina-horário	
Coordenar MC001	Jean	Coordenar MC001	3	Coordenar MC001	
MC001	Jean	MC001	3	MC001	2:14 2:15 6:14 6:15
MC002	Franz	MC002	3	MC002	3:08 3:09 5:08 5:09
MC003	Albert	MC002	3	MC003	2:14 2:15 6:14 6:15
⋮	Franz	MC003	1	MC003	3:08 3:09 5:08 5:09
	⋮	⋮	⋮	⋮	⋮

Finalmente, para a construção da interface gráfica, foi utilizada a biblioteca *Tkinter* [52]. O resultado final está expresso na Seção 3.

2.3 Modelo PLI preliminar

Os parâmetros do modelo, utilizados para gerar o conjunto de restrições correspondentes ao modelo de PLI, estão descritos abaixo.

Parâmetros:

$$P \tag{1a}$$

$$D \tag{1b}$$

$$t \in T = \{0...32\} \tag{1c}$$

$$b_{dp} \in \{0, 1, 2, 3\} \tag{1d}$$

$$Hd_t \in \{0, 1\} \tag{1e}$$

$$HP_{pt}^{sim} \in \{0, 1\} \tag{1f}$$

$$HP_{pt}^{nao} \in \{0, 1\} \tag{1g}$$

$$C \in \mathbb{N} \tag{1h}$$

$$C_d \in \mathbb{N} \tag{1i}$$

Variáveis de decisão:

$$x_{pd} \begin{cases} 1, & \text{se professor } p \in P \text{ leciona } d \in D \\ 0, & \text{c.c.} \end{cases}$$

Restrições:

$$\sum_{p} x_{pd} = 1, \forall d \in D \tag{1j}$$

$$\sum_{d \in DT_t} x_{pd} \leq 1, \forall t, \forall p \in P \tag{1k}$$

$$x_{pd} = 0, \forall p \in P, \forall d \in D : \ b_{dp} = 0 \tag{1l}$$

$$\sum_{d \in DT_t} x_{pd} = 0, \forall p \in P, \forall t : \ HP_{pt}^{nao} = 1 \tag{1m}$$

$$\sum_{d \in D} x_{pd} * C_d \leq C, \forall p \in P \tag{1n}$$

$$DT_t \subseteq D \tag{1o}$$

Função objetivo:

$$\text{MAX } Z = \sum_{p \in P} \sum_{d \in D} x_{pd} * b_{dp} \tag{1p}$$

Os conjuntos de professores e disciplinas a serem associados estão representados em (1a) e (1b). Já o conjunto (1d) descreve o benefício de se alocar o professor $p \in P$ à disciplina $d \in D$. Os parâmetros dados em (1e) assumem valores binários que informam se a disciplina $d \in D$ é oferecida no tempo $t \in \{1, ..., 32\}$, no qual cada valor de t representa um horário letivo da semana, que é dividida em seis horários por dia útil com dois horários adicionais nos sábados.

(1f) e (1g) indicam os horários desejados e indesejados para cada docente, respectivamente. Já (1h) indica o número máximo de créditos que podem ser ministrados por um docente e (1i) mostra quantos créditos cada disciplina contém.

A restrição (1j) garante que toda disciplina é ministrada por exatamente um docente. (1k) faz com que nenhum professor tenha que ministrar duas disciplinas distintas no mesmo *timeslot*. (1l) faz com que nenhuma disciplina seja alocada a um professor que não tem interesse em ministrá-la. (1m) impede a alocação de disciplinas que tenham intersecção com a proibição de horário de um docente. Por fim, (1n) impede que um professor lecione mais horas-aula do que é desejado.

2.4 Modelo PLI: considerações para o Instituto de Computação

Variáveis de decisão:

$$x_{pdc_{dh}} \begin{cases} 1, & \text{se } p \in P \text{ leciona } d \in D,\ d \text{ ministrada nos tempos } c_{dh} \in C_{dh} \\ 0, & \text{c.c.} \end{cases}$$

Parâmetros:

$$P \tag{2a}$$

$$D \tag{2b}$$

$$b_{dp} \in \{0, 1, 2, 3\} \tag{2c}$$

$$Hd_t \in \{0, 1\} \tag{2d}$$

$$HP_{pt}^{sim} \in \{0, 1\} \tag{2e}$$

$$HP_{pt}^{nao} \in \{0, 1\} \tag{2f}$$

$$C \in \mathbb{N} \tag{2g}$$

$$C_d \in \mathbb{N} \tag{2h}$$

$$C_{pos} \subset C \tag{2i}$$

$$C_{dh} \in \{0, 1\}^{32}, \forall c \in C \tag{2j}$$

$$C_{dh_{ct}} \in C_{dh}, \forall c \in C, \forall t \in T \tag{2k}$$

$$DT_t \subseteq D \tag{2l}$$

Restrições:

$$\sum_{p} \sum_{c_{dh}} x_{pdc_{dh}} = 1, \forall d \in D \tag{2m}$$

$$\sum_{d \in DT_t,} x_{pdc_{dh}} \leq 1, \forall t, \forall p \in P \tag{2n}$$

$$x_{pdc_{ch}} = 0, \forall p \in P, \forall d \in D : \ b_{dp} = 0 \tag{2o}$$

$$\sum_{d \in DT_t} x_{pd_{ch}} = 0, \forall p \in P, \forall t : \ HP_{pt}^{nao} = 1 \tag{2p}$$

$$\sum_{d \in D} x_{pd_c dh} * C_d \leq C, \forall p \in P \tag{2q}$$

$$\sum_{c_{dhct} \in C_{dh_{ct}}} c_{dhct} \leq 3, \forall t, \forall p \in P \tag{2r}$$

Função objetivo:

$$\text{MAX } Z = \sum_{p \in P} \sum_{d \in D} \sum_{c_{dh} \in C_{dh}} x_{pdc_{dh}} * b_{dp} \tag{2s}$$

Com este modelo, cada disciplina está associada a um conjunto de quadros de horário. Assim, as variáveis de decisão também indexam quando a matéria será ministrada.

Agora, a restrição (2m) garante que cada disciplina seja ministrada em exatamente um horário e por exatamente um professor. Já (2n) faz com que nenhum docente seja alocado a mais de um disciplina no mesmo horário. A somatória está indexada pelo conjunto de arcos disciplina-horário que tem intersecção (2l).

As demais restrições são análogas as do modelo anterior, com a diferença de que agora é levado em conta o quadro de horário no qual a disciplina é ministrada. A única exceção é a restrição (2r), que limita em três o número de disciplinas que compartilham um mesmo horário, para um conjunto de matérias. Isso corresponde às disciplinas de pós-graduação que tem disponível três salas para serem ministradas.

2.5 Modelo PLI implementado: alocação de quadro de horário

Parâmetros:

$$P \tag{3a}$$

$$D \tag{3b}$$

$$Q \tag{3c}$$

Variáveis de decisão:

$$x_{pd} \begin{cases} 1, & \text{se } p \in P \text{ leciona } d \in D, \\ 0, & \text{c.c.} \end{cases} \tag{3d}$$

$$y_{dq} \begin{cases} 1, & \text{se } d \in D \text{ está alocado no horário } q \in Q, \\ 0, & \text{c.c.} \end{cases} \tag{3e}$$

$$z_{pq} \begin{cases} 1, & \text{se } p \in P \text{ ministra discilpina no horário } q \in Q, \\ 0, & \text{c.c.} \end{cases} \tag{3f}$$

Restrições:

$$\sum_{p \in P} x_{pd} = 1, \ \forall d \in D \tag{3g}$$

$$\sum_{q \in Q_d} y_{dq} = 1, \ \forall d \in D \tag{3h}$$

$$\sum_{d \in D} x_{pd} * C_d \leq C_p, \ \forall p \in P \tag{3i}$$

$$z_{pq} = 0 \ \forall p \in P, \forall q \in \tilde{Q}_p \tag{3j}$$

$$x_{pd} = 0 \ \forall p \in P, \forall d \in \tilde{D}_p \tag{3k}$$

$$z_{pq'} + Z_{pq''} \leq 1, \ \forall p \in P, \forall (q', q'') \in Qconflito \tag{3l}$$

Com esse modelo, o parâmetro (3c) contém informações sobre quais quadros de horário podem ser utilizados por cada disciplina. (3g) assegura que toda matéria é lecionada por exatamente um mestre. (3h) certifica que todo curso é atribuído a um *timeframe*. A variável de decisão (3f) faz o acoplamento entre o docente e o horário da disciplina que este ministra. Isso é necessário porque variáveis de decisão distintas alocam professores em matérias e matérias em horários, mas os *timeframes* precisam ser respeitados para cada mestre. (3j) e (3k) fazem com que nenhuma disciplina seja atribuída a um docente que não deseja ministrá-la ou não possa por proibição de *timeslot*. (3l) garante que não haja horários conflitantes para cada titular.

Os pares (q', q'') correspondem a todos os pares de *timeframe* que têm qualquer tipo de intersecção. Na implementação proposta, isso é determinado no pré-processamento.

3 Resultados

Todos os experimentos foram realizados utilizando o processador Intel® Core™ i7-3537U CPU @ 2.00GHz x 4, 6GB de RAM DDR3 @ 1333 MHz, Gurobi 8.1.1 (licença acadêmica), Python 3.7.3 e Ubuntu 19.04 64-bit com ambiente gráfico GNOME 3.32.1.

O programa proposto foi executado com dados reais do Instituto de Computação da Unicamp. Os dados são entregues ao programa sem nenhuma modificação, isto é, como uma planilha (formato *.csv*). O pré-processamento realiza o *parse* das informações necessárias e chama o resolvedor. Essa instância apresenta 32 docentes e 34 disciplinas. Para esses dados, o pré-processamento e a chamada para o *Gurobi* levaram menos de 0,5 segundo. Após a solução ser encontrada, leva cerca de 10 segundos para a exibição da interface gráfica.

Na interface exibida é possível explorar o resultado, observando-se o horário no qual cada disciplina foi alocada e a qual professor. Além disso, o modelo exibe algumas estatísticas da alocação proposta, como, por exemplo, o número de docentes que estão ministrando a matéria de sua maior preferência. Esses dados foram utilizados para comparar a qualidade da solução construída pelo modelo com a feita manualmente pelo Instituto.

Pode-se ver que o modelo mostra uma maneira amigável de se navegar e exportar o resultado, que teve alta qualidade, atendendo de maneira satisfatória as preferências, como

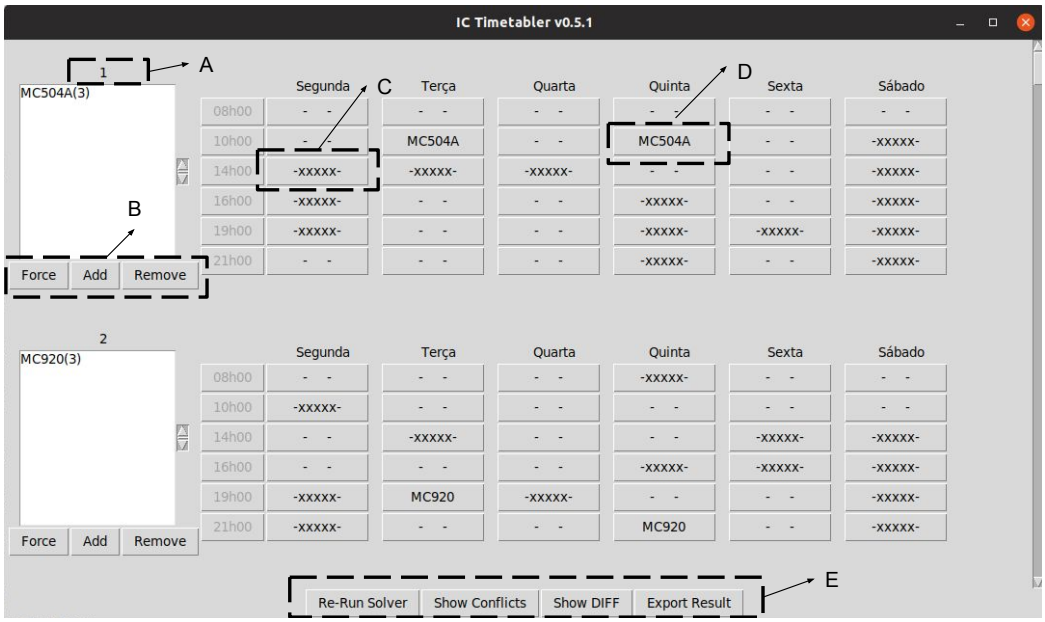

Figura 6: Interface gráfica do programa proposta para exibição do resultado. A região A contém o nome do docente. Na região B encontram-se os botões para adicionar ou remover disciplinas da lista de preferência do docente, além de um botão que força a atribuição de uma disciplina ao professor. As regiões C e D exemplificam um horário restrito (não pode ser utilizado para ministrar aulas) e um *slot* no qual uma matéria é lecionada pelo professor correspondente, respectivamente. Por fim, na localização indicada por E, encontram-se os botões para, respectivamente, encontrar uma solução com as modificações feitas, exibir quais restrições não puderam ser atendidas (em caso de não factibilidade), mostrar as diferenças entre os dados carregados e o modelo atual, após modificações e para exportar o resultado no formato de planilha.

mostra a Tabela 3.

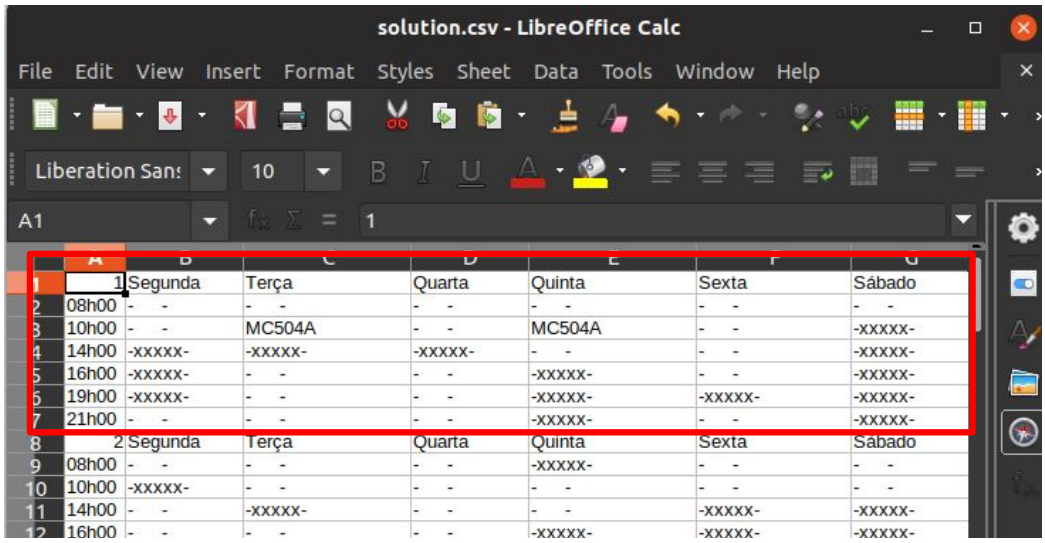

Figura 7: Exemplo de planilha exportada, com informações sobre o quadro de horário de cada docente. A planilha está sendo exibida no *software* LibreOffice Calc.

Além disso, para testar a capacidade do modelo de alterar o quadro de horário de cada disciplina de maneira eficiente, foi adicionada uma restrição artificial de horário proibido de cada docente. Isso foi feito sorteando-se entre 8 e 10 posições, entre as 32 disponíveis na semana (ver Figura 3), aleatórias para cada docente, e tornando-as inválidas para se ministrar aulas. Adicionalmente, foi permitido que cada aula pudesse ser ministrada em qualquer *timeframe* que tivesse a mesma quantidade de horas que o *timeframe* original da disciplina. Mesmo com esse grau adicional de liberdade, o modelo ainda é resolvido no mesmo tempo e a solução tem a mesma qualidade que a reportada no experimento inicial. É claro que, nessa versão, as disciplinas são acomodadas em horários diferentes.

No caso de infactibilidade do modelo é utilizado novamente o *Gurobi* para se determinar

Tabela 3: Número de disciplinas alocadas em cada categoria de preferência para diferentes soluções e entradas de dado. As restrições artificiais reportadas na última linha correspondem a 8 horários da semana aleatórios marcados como proibidos para cada docente.

	Nível de preferência			
	Máximo	Médio	Mínimo	Nenhum
Solução manual	23	4	0	5
Solução do modelo proposto	23	7	2	0
Solução do modelo proposto com restrições aleatórias de horário adicionadas aos docentes	23	7	2	0

o subsistema inconsistente irredutível (*IIS*) e assim indicar para o usuário quais restrições não podem ser atendidas com os parâmetros dados de entrada. Nos testes realizados, a informação se mostrou adequadamente clara, indicando caso não fosse possível alocar um curso, ou respeitar um horário proibido de um professor ou alocar um quadro de horário para uma certa disciplina. Ainda assim, um modelo infactível pode conter múltiplos *IIS*, de forma que todos os conflitos têm de ser removidos, modificando-se a entrada, para que o resolvedor possa encontrar uma solução viável.

4 Conclusão

A proposta de uma solução automática é de fato importante para se permitir explorar de forma mais completa o espaço de busca do problema de tabelamento de horário, já que o resolvedor pode encontrar quadros de alocação que usam melhor os recursos da instituição, aumentando a satisfação geral.

Além disso, apesar de ser um problema computacionalmente difícil de ser resolvido, diversas formulações são tratáveis e úteis. Neste trabalho, por exemplo, os resultados ótimos foram encontrados em menos de meio segundo, o que indica que é viável adicionar ainda mais *soft constraints*.

Adicionalmente, se mostrou prático a implementação de um *parser* que permitisse o uso direto dos dados, sem nenhuma alteração manual. Dessa forma, foi possível testar o modelo com os dados reais e comparar com a solução confeccionada pelo Instituto. Da mesma forma, a interface gráfica e as métricas e comparações feitas para se averiguar a qualidade do modelo também são significativas.

4.1 Trabalhos futuros

Como a formulação proposta foi resolvida muito rapidamente, mesmo com a adição artificial de grau de liberdade (permitindo variar o quadro de horário de cada matéria, por exemplo), ficou evidente que é possível levar em conta mais considerações no modelo. Isso pode ser feito, por exemplo, oferecendo aos docentes a possibilidade de escolher se desejam horários mais compactos ou mais espalhados, se têm preferência por um dia da semana ou horário, entre outras opções que devem ser investigadas de acordo com as características do Instituto.

Também é interessante acrescentar mais rotinas interativas para modificar o modelo em tempo de execução, já que atualmente o programa permite apenas adicionar e remover matérias do arco de preferência de um professor. Com isso, o processo de tornar factível a entrada ou explorar soluções alternativas de forma direcionada ficaria mais prático.

Adicionalmente, como já foi proposto um *parser* que trata os dados em seu formato original, uma extensão prática seria a possibilidade de exibir, para o usuário, sugestões baseadas em alocações anteriores. Essa ferramenta de auxílio, atuando supletivamente com o resolvedor automático, tem o potencial de facilitar e aprimorar o processo de raciocínio sobre as alterações. Esse tipo de proposta visa integrar a capacidade computacional de buscar otimalidade com os aspectos práticos de decisão que não entraram no modelo.

Por fim, a utilidade da solução proposta depende consideravelmente da sua integração no processo de tabelamento do Instituto. Assim sendo, uma métrica de qualidade que pode

ser extraída para a construção futura de programas mais robustos para esse problema é a coleta de *feedback* sobre este projeto.

Referências

[1] D. de Werra, "An introduction to timetabling," *European Journal of Operational Research*, vol. 19, pp. 151–162, feb 1985.

[2] F. Aloul, B. Al-Rawi, A. Al-Farra, and B. Al-Roh, "Solving Employee Timetabling Problems using Boolean Satisfiability," *2006 Innovations in Information Technology, IIT*, pp. 1–5, 2006.

[3] C. Liebchen and L. Peeters, "Some Practical Aspects of Periodic Timetabling," in *Operations Research Proceedings 2001*, pp. 25–32, Berlin, Heidelberg: Springer Berlin Heidelberg, 2002.

[4] O. J. Ibarra-Rojas and Y. A. Rios-Solis, "Synchronization of bus timetabling," *Transportation Research Part B: Methodological*, vol. 46, pp. 599–614, jun 2012.

[5] E. K. Burke and S. Petrovic, "Recent research directions in automated timetabling," *European Journal of Operational Research*, vol. 140, pp. 266–280, jul 2002.

[6] A. Schaerf, "Survey of automated timetabling," *Artificial Intelligence Review*, vol. 13, no. 2, pp. 87–127, 1999.

[7] D. L. de Queiroz and N. V. Nepomuceno, "Um Modelo Em Programação Linear Inteira Para Alocação De Disciplinas: Um Estudo De Caso No Curso De Ciência Da Computação Da Universidade De Fortaleza," *XLIX Simpósio Brasileiro de Pesquisa Operacional, Blumenau. Anais do XLIX SBPO*, vol. 49, 2017.

[8] E. Burke, D. Elliman, P. Ford, and R. Weare, "Examination timetabling in British Universities: A survey," in *International Conference on the Practice and Theory of Automated Timetabling*, pp. 76–90, Springer, Berlin, Heidelberg, 1996.

[9] R. M. L. KRIPKA, N. T. ORO, and M. KRIPKA, "Distribuição de cargas horárias em Instituições de Ensino Superior: uma formulação para a maximização do aproveitamento dos recursos humanos," *Ciência & Engenharia*, vol. 14, no. 1, pp. 65–72, 2005.

[10] L. A. Ciscon, A. C. Oliveira, T. R. Hipólito, G. B. Alvarenga, and A. C. Roullier, "O Problema de Geração de Horários: Um Foco na Eliminação de Janelas e Aulas Isoladas," *XXXVII Brazilian Symposium of Operational Research*, 2005.

[11] R. D. G. Pacífico and A. M. Coelho, "Análise geral sobre Problemas de Horários Educacionais tratados no Brasil," in *Proceeding Series of the Brazilian Society of Computational and Applied Mathematics*, vol. 3, aug 2015.

[12] V. A. Bardadym, "Computer-aided school and university timetabling: The new wave," in *International conference on the practice and theory of automated timetabling*, pp. 22–45, Springer, Berlin, Heidelberg, 1996.

[13] M. W. Carter, "OR Practice—A Survey of Practical Applications of Examination Timetabling Algorithms," *Operations Research*, vol. 34, pp. 193–202, apr 1986.

[14] E. Burke, K. Jackson, J. H. Kingston, and R. Weare, "Automated University Timetabling: The State of the Art," *The Computer Journal*, vol. 40, pp. 565–571, sep 1997.

[15] C. Freitas and P. Guimarães, "Uma ferramenta baseada em algoritmos genéticos para a geração de tabela de horário escolar," *Sétima Escola ...*, no. SEPTEMBER 2014, 2007.

[16] S. Ghaemi, M. Taghi Vakili, and A. Aghagolzadeh, "Using a genetic algorithm optimizer tool to solve University timetable scheduling problem," in *2007 9th International Symposium on Signal Processing and Its Applications*, pp. 1–4, IEEE, feb 2007.

[17] R. Lewis, "A survey of metaheuristic-based techniques for University Timetabling problems," *OR Spectrum*, vol. 30, pp. 167–190, nov 2007.

[18] G. B. Bucco, C. J. Bornia-Poulsen, D. L. Bandeira, G. B. Bucco, C. J. Bornia-Poulsen, and D. L. Bandeira, "Desenvolvimento de um modelo de programação linear para o Problema da Construção de Grades Horárias em Universidades," *Gestão & Produção*, vol. 24, pp. 40–49, feb 2017.

[19] H. Babaei, J. Karimpour, and A. Hadidi, "A survey of approaches for university course timetabling problem," *Computers & Industrial Engineering*, vol. 86, pp. 43–59, aug 2015.

[20] R. S. Xavier and R. R. Saldanha, "Heurísticas baseadas no algoritmo de coloração de grafos para o problema de alocação de salas em uma instituição de ensino superior," *XLII SBPO*, pp. 2123–2134, 2009.

[21] R. H. D. F. SILVA, *Uma aplicação da Programação Inteira no School Timetabling Problem*. PhD thesis, Universidade Federal Rural de Pernambuco, 2015.

[22] H. M. Ten Eikelder and R. J. Willemen, "Some complexity aspects of secondary school timetabling problems," in *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*, vol. 2079 LNCS, pp. 18–27, Springer, Berlin, Heidelberg, 2001.

[23] L. I. B. de Freitas Filho, Francisco Sergio Freitas and C. P. de Souza, "Modelo PLI para Alocação de Professores da UFC-Quixadá," *XLIX Simpósio Brasileiro de Pesquisa Operacional*, 2017.

[24] R. C. Monteiro and E. H. Kampke, "Heurísticas de Construção no Problema de Tabela-Hórario de Universidades," 2017.

[25] M. Chiarandini, M. Birattari, K. Socha, and O. Rossi-Doria, "An effective hybrid algorithm for university course timetabling," *Journal of Scheduling*, vol. 9, pp. 403–432, oct 2006.

[26] V. O. Oladokun and S. O. Badmus, "An Integer Linear Programming Model of a University Course Timetabling Problem.," *Pacific journal of science and technology*, 2008.

[27] T. Birbas, S. Daskalaki, and E. Housos, "Timetabling for Greek high schools," *Journal of the Operational Research Society*, vol. 48, pp. 1191–1200, dec 1997.

[28] H. Rudová, T. Müller, and K. Murray, "Complex university course timetabling," *Journal of Scheduling*, vol. 14, pp. 187–207, apr 2011.

[29] B. A. S. LARA, *Quadro de Horários Acadêmico: Uma Abordagem com Foco na Avaliação Institucional e na Gestão de Custos de Instituições de Ensino Superior Privadas Brasileiras*. PhD thesis, aug 2006.

[30] V. Cacchiani, A. Caprara, R. Roberti, and P. Toth, "A new lower bound for curriculum-based course timetabling," *Computers & Operations Research*, vol. 40, pp. 2466–2477, oct 2013.

[31] O. Rossi-Doria, M. Sampels, M. Birattari, M. Chiarandini, M. Dorigo, L. M. Gambardella, J. Knowles, M. Manfrin, M. Mastrolilli, B. Paechter, L. Paquete, and T. Stützle, "A Comparison of the Performance of Different Metaheuristics on the Timetabling Problem," in *International Conference on the Practice and Theory of Automated Timetabling*, pp. 329–351, Springer, Berlin, Heidelberg, 2003.

[32] L. T. G. Merlot, N. Boland, B. D. Hughes, and P. J. Stuckey, "A Hybrid Algorithm for the Examination Timetabling Problem," in *International Conference on the Practice and Theory of Automated Timetabling*, pp. 207–231, Springer, Berlin, Heidelberg, 2003.

[33] S. Daskalaki and T. Birbas, "Efficient solutions for a university timetabling problem through integer programming," *European Journal of Operational Research*, vol. 160, no. 1, pp. 106–120, 2005.

[34] I. Méndez-Díaz, P. Zabala, and J. J. Miranda-Bront, "An ILP based heuristic for a generalization of the post-enrollment course timetabling problem," *Computers & Operations Research*, vol. 76, pp. 195–207, dec 2016.

[35] B. McCollum, A. Schaerf, B. Paechter, P. McMullan, R. Lewis, A. J. Parkes, L. D. Gaspero, R. Qu, and E. K. Burke, "Setting the Research Agenda in Automated Timetabling: The Second International Timetabling Competition," *INFORMS Journal on Computing*, vol. 22, pp. 120–130, feb 2010.

[36] A. Wren, "Scheduling, timetabling and rostering — A special relationship?," in *International conference on the practice and theory of automated timetabling*, pp. 46–75, Springer, Berlin, Heidelberg, 1996.

[37] S. Daskalaki, T. Birbas, and E. Housos, "An integer programming formulation for a case study in university timetabling," *European Journal of Operational Research*, vol. 153, pp. 117–135, feb 2004.

[38] R. M. Cordeiro, *Sistema de apoio à decisão para a elaboração de mapas de exames no ensino superior*. PhD thesis, Universidade do Porto, aug 2017.

[39] P. Boizumault, Y. Delon, and L. Peridy, "Constraint logic programming for examination timetabling," *The Journal of Logic Programming*, vol. 26, pp. 217–233, feb 1996.

[40] E. K. Burke, S. Petrovic, and R. Qu, "Case-based heuristic selection for timetabling problems," *Journal of Scheduling*, vol. 9, pp. 115–132, apr 2006.

[41] K. Socha, J. Knowles, and M. Sampels, "A MAX-MIN Ant System for the University Course Timetabling Problem," in *International Workshop on Ant Algorithms*, pp. 1–13, Springer, Berlin, Heidelberg, 2002.

[42] W. Erben and J. Keppler, "A genetic algorithm solving a weekly course-timetabling problem," in *International Conference on the Practice and Theory of Automated Timetabling*, pp. 198–211, Springer, Berlin, Heidelberg, 1996.

[43] F. Vieira and H. Macedo, *Sistema de Alocação de Horários de Cursos Universitários: Um Estudo de Caso no Departamento de Computação da Universidade Federal de Sergipe*, vol. 7. apr 2011.

[44] M. R. Poltosi and M. Regina, *Elaboração de escalas de trabalho de técnicos de enfermagem com busca tabu e algoritmos genéticos*. PhD thesis, Universidade do Vale do Rio dos Sinos, mar 2007.

[45] H. Rudová and K. Murray, "University Course Timetabling with Soft Constraints," in *International Conference on the Practice and Theory of Automated Timetabling*, pp. 310–328, Springer, Berlin, Heidelberg, 2002.

[46] C. L. da Silva Nunes, Rosângela Guimaraes, Norton Coelho Carvalho and C. Morrinhos, "Planejamento de Grade de Horário em uma Universidade Brasileira usando Algoritmos Genéticos," in *X Encontro Nacional de Inteligência Artificial e Computacional (ENIAC)*, 2013.

[47] B. Alves and S. Lara, "UM MODELO DE PROGRAMAÇÃO LINEAR PARA SOLUÇÃO DO COMPLEMENTARES EM INSTITUIÇÕES DE ENSINO SUPERIOR," *XXXVI SBPO*, pp. 381–387, 2004.

[48] J. Pistori, H. Pistori, C. Butera, and A. Mira Filho, "Um Ambiente Colaborativo para Confecção de Horários de Aulas no Ensino Superior," in *6 Workshop Software Livre*, 6 Workshop Software Livre, 2005.

[49] R. M. Permanhane and L. D. Secchin, "O problema da elaboração de grade de horários escolares: uma aplicação á Universidade Federal do Espírito Santo," in *Proceeding*

Series of the Brazilian Society of Computational and Applied Mathematics, vol. 2, dec 2014.

[50] G. S. Vasconcelos and C. S. Sakuraba, "Metaheurísticas para a elaboração de grades horárias universitárias," in *VI Simpósio de Engenharia de Produção*, Departamento de Engenharia de Produção - Universidade Federal de Sergipe, 2014.

[51] D. de Werra, "The combinatorics of timetabling," *European Journal of Operational Research*, vol. 96, pp. 504–513, feb 1997.

[52] J. W. Shipman, "Tkinter 8.4 reference: a gui for python," *New Mexico Tech Computer Center*, 2013.