UNIVERSIDADE FEDERAL DE ITAJUBÁ

SMAC03 - GRAFOS

PROGRAMAÇÃO DE HORÁRIOS DE AULAS

2024006649 - IVAN MATHEUS RIBEIRO SILVERIO
2024003315 - JOAO VITOR PINHEIRO FORTUNATO
2024008830 - PEDRO LUIZ DE MORAES FERREIRA
2024006729 - THEO HENRIQUE AZEVEDO DE CARVALHO PEREIRA

SMAC03 - GRAFOS

Prof. Rafael Frinhani

INSTITUTO DE MATEMÁTICA E COMPUTAÇÃO
UNIFEI - Itajubá

2025

UNIVERSIDADE FEDERAL DE ITAJUBÁ
INSTITUTO DE MATEMÁTICA E COMPUTAÇÃO
SISTEMAS DE INFORMAÇÃO
SMAC03 - GRAFOS

Programação de Horários de Aulas

1 Introdução

A elaboração de grades horárias em instituições de ensino superior é uma atividade essencial para o funcionamento acadêmico, porém complexa e altamente suscetível a erros quando realizada manualmente. A necessidade de alocar disciplinas, professores, salas e períodos de tempo de forma simultânea e consistente gera um problema de grande escala e elevado grau de interdependência entre os elementos. Este desafio é amplamente estudado na literatura como o *University Course Timetabling Problem* (UCTP), classificado como um problema NP-difícil, o que implica que soluções exatas tornam-se impraticáveis à medida que o número de disciplinas cresce Vieira & Macedo (2011).

Diversas abordagens têm sido propostas para solucionar o UCTP, como coloração de grafos, heurísticas, metaheurísticas e modelos híbridos Bello et al. (2008). Em particular, abordagens baseadas em restrições e decomposição são amplamente utilizadas, pois permitem separar o problema em partes menores e tratáveis. Burke e Petrovic Burke & Petrovic (2010) apresentam uma referência consolidada para o domínio, introduzindo a ideia de tratar diferentes tipos de conflitos como estruturas independentes — conceito equivalente ao uso de grafos multicamadas.

O presente trabalho aplica essa perspectiva de multicamadas para gerar automaticamente a grade horária dos cursos CCO e SIN da instituição, integrando restrições de professores, laboratórios, períodos e conflitos curriculares. A abordagem busca fornecer uma solução automatizada, eficiente e escalável em comparação ao processo manual tradicional.

1.1 Cenário de Estudo (Contextualização e Conceitos)

A instituição em estudo oferece cursos com diversas especificidades, envolvendo disciplinas teóricas, aulas práticas, laboratórios e múltiplos turnos. O processo de montagem da grade horária deve atender simultaneamente a um conjunto de restrições que afetam professores, recursos físicos e currículos.

Segundo Bello et al. Bello et al. (2008), problemas de grade horária podem ser modelados como um *grafo de conflitos*, no qual cada aula é representada por um vértice, e cada restrição é representada por uma aresta. A coloração desse grafo corresponde à alocação de horários distintos.

Entretanto, a literatura mais recente indica a vantagem de organizar essas restrições em **múltiplas camadas independentes**, cada uma representando um tipo específico de conflito — uma estratégia discutida no trabalho de Burke e Petrovic Burke & Petrovic (2010). Inspirado por esse paradigma, o presente trabalho modela o problema utilizando três camadas principais:

- **Camada Curricular**: conflitos entre disciplinas de um mesmo curso/período;
- **Camada Professor**: impede que um professor ministre duas aulas simultâneas;
- **Camada de Recursos**: controla disputas por laboratórios específicos;

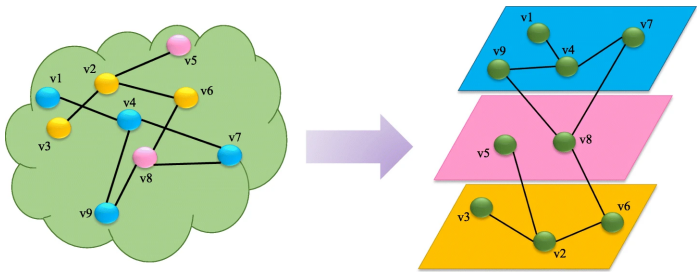

Figura 1: Representação do grafo multicamadas utilizado neste trabalho, ilustrando a separação de conflitos curriculares, de professor e de recursos.

Fonte: Autoria própria.

1.2 Objetivos do Projeto (Detalhamento do Problema)

O objetivo geral deste projeto é desenvolver um sistema capaz de gerar automaticamente a grade horária completa dos cursos CCO e SIN, respeitando as restrições institucionais, curriculares e de recursos.

Os objetivos específicos incluem:

1. **Construir o grafo multicamadas** que integra restrições curriculares, de professor e de laboratório, conforme discutido por Burke e Petrovic Burke & Petrovic (2010);
2. **Gerar o grafo complementar** para representar compatibilidades entre aulas;
3. **Aplicar heurísticas baseadas em cliques maximais**, inspiradas em abordagens de coloração de grafos utilizadas em timetabling Müller (2002);
4. **Equilibrar a distribuição semanal das aulas**, evitando dias vazios por período;
5. **Produzir como saída um arquivo CSV e uma visualização HTML** da grade horária final.

Assim, este projeto combina conceitos de otimização combinatória, teoria dos grafos e heurísticas computacionais para propor uma solução automatizada, flexível e aderente ao cenário real da instituição.

2 Referencial Teórico

O problema de agendamento de horários em instituições de ensino, conhecido internacionalmente como *University Course Timetabling Problem* (UCTP), é um dos problemas clássicos mais desafiadores na área de Pesquisa Operacional e Inteligência Artificial. Cooper & Kingston (1996) demonstraram que o problema de construir uma grade horária válida é NP-Difícil (*NP-Hard*), o que significa que não existe, até o momento, um algoritmo capaz de encontrar a solução ótima em tempo polinomial para todas as instâncias do problema.

A literatura classifica as restrições do UCTP em duas categorias distintas, conforme detalhado por Vieira & Macedo (2011) e Burke & Petrovic (2010):

- **Restrições Rígidas (*Hard Constraints*):** São condições *sine qua non* para a viabilidade da grade. A violação de qualquer uma dessas regras torna a solução inválida. No contexto deste projeto, consideramos como restrições rígidas: (i) unicidade de alocação docente (um professor não pode estar em dois lugares ao mesmo tempo); (ii) unicidade de alocação discente (uma turma não pode ter choques de horário); (iii) capacidade e disponibilidade de recursos físicos (laboratórios).
- **Restrições Suaves (*Soft Constraints*):** Referem-se a preferências e qualidade da solução. Uma grade que viola estas restrições ainda é executável, porém considerada de baixa qualidade. Exemplos incluem a minimização de janelas ociosas para alunos e a compactação da carga horária dos professores para evitar deslocamentos desnecessários.

A Teoria dos Grafos é amplamente usada no UCTP, pois permite representar conflitos por meio de arestas. Ao reduzir o problema à Coloração de Grafos, busca-se o número cromático mínimo que atenda às restrições de adjacência entre eventos.

3 Solução Proposta

A solução desenvolvida consiste num sistema automatizado que modela o problema através de grafos multicamadas e aplica um algoritmo de busca exata com heurísticas de ordenação para gerar uma grade horária válida de forma eficiente.

3.1 Evolução da Abordagem

O desenvolvimento da solução seguiu um processo iterativo e incremental. Inicialmente, buscou-se uma abordagem direta de satisfação de restrições (coloração gulosa). Embora eficaz para eliminar conflitos rígidos, essa abordagem gerava grades fragmentadas.

3.2 Modelagem do Problema

O problema foi modelado utilizando a abordagem de Grafos Multicamadas, na qual cada vértice representa uma matéria (contendo professor, respectivo curso, código, período, laboratório requerido, carga horária e nome) e as arestas representam conflitos gerados pela união de três categorias de restrições rígidas:

1. **Camada de Professores:** Conecta aulas ministradas pelo mesmo docente, impedindo que este esteja em dois lugares simultaneamente.
2. **Camada Curricular / Turmas (Conflito Discente):** Conecta aulas obrigatórias de um mesmo período, garantindo que os alunos daquela coorte não tenham choques de horário. É nesta camada que se aplica a estratégia de **Trilhas Optativas**:
   - Foi desenvolvida uma estratégia de agrupamento de disciplinas optativas em "Trilhas" (como Trilha 1 e Trilha 2).
   - Disciplinas que pertencem a **Trilhas diferentes** dentro do mesmo período são tratadas como **compatíveis**, permitindo que o algoritmo de clique as agende em horários paralelos.
3. **Camada de Recursos Físicos:** Conecta aulas que disputam o mesmo laboratório ou sala especializada.

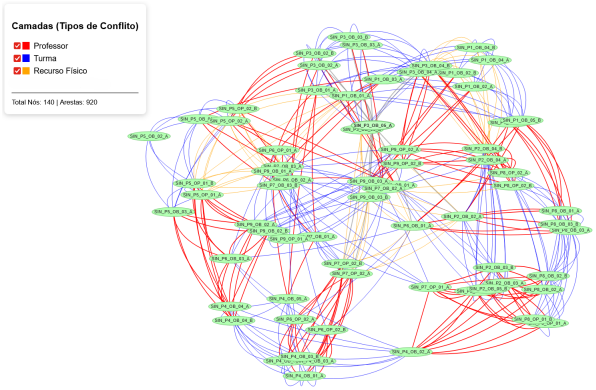

Figura 2: Representação do grafo multicamadas utilizado neste trabalho, ilustrando a separação de conflitos de Professor, Recursos e a Camada Curricular/Turma, que inclui a lógica de Trilhas Optativas.

Fonte: Representação criada por IA com base no grafo de conflito do dataset.

3.2.1 Caracterização e Estrutura do Dataset

Para validar a eficácia da solução proposta, foi utilizado um conjunto de dados fictícios referente à oferta de disciplinas dos cursos de Ciência da Computação (CCO) e Sistemas de Informação (SIN). Os dados foram pré-processados e consolidados num arquivo CSV (*Comma-Separated Values*), que serve como entrada para o algoritmo.

A instância do problema é composta por **140 aulas (vértices)** distribuídas entre disciplinas obrigatórias e optativas do 1º ao 9º período. A estrutura do arquivo de entrada é detalhada na Tabela 1, onde cada linha representa uma aula a ser alocada.

Tabela 1: Dicionário de dados do arquivo de entrada (`dataset_processado.csv`).

Coluna	Descrição
`ID_Aula`	Identificador único da aula (ex: `CCO_P1_OB_01_A`). Usado como chave do vértice no grafo.
`ID_Disciplina`	Código da disciplina. Aulas da mesma disciplina (turmas A e B) compartilham este ID.
`Curso`	Curso ao qual a disciplina pertence (CCO ou SIN), determinante para a preferência de turno.
`Periodo`	Semestre letivo sugerido (1 a 9). Fundamental para criar conflitos de fluxo regular.
`Professor`	Identificador do docente responsável (ex: `Prof_CCO_4`). Base para a camada de restrição de professores.
`Lab_Requerido`	Indica se a aula exige um laboratório específico (ex: `Lab_Redes`) ou se é em sala comum (nulo).
`CH_Aula`	Duração da aula em horas (2 ou 3), definindo quantos slots consecutivos ela ocupa.

Fonte: Autoria própria (Com base no CSV utilizado no projeto).

A leitura e manipulação destes dados são realizadas através da biblioteca `pandas`, que converte a estrutura tabular em objetos manipuláveis pelo algoritmo de construção do grafo.

3.3 Algoritmo: Busca Limitada no Tempo

O núcleo do sistema é um algoritmo de **Busca em Profundidade (DFS) com Backtracking**. O sistema opera como um algoritmo *Anytime*, configurado nesse caso para executar dentro de uma janela de **15 segundos**, esse tempo pordendo ser definido pelo usuário caso ele queira obter avaliações mais profundas ou não.

O fluxo de execução do algoritmo, desde a seleção da disciplina até a validação e retrocesso (*backtracking*), está ilustrado na Figura 3.

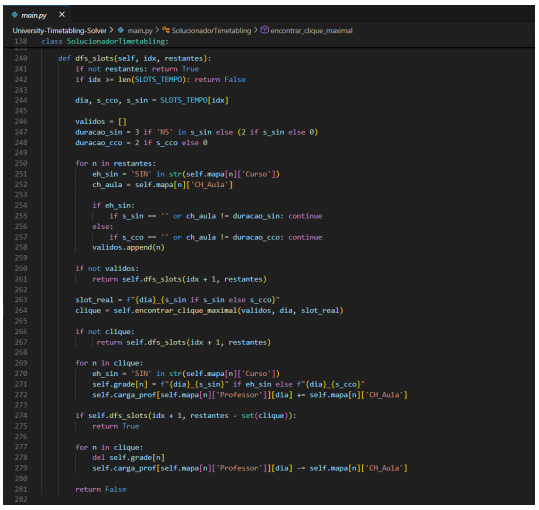

Figura 3: Fluxograma do algoritmo de alocação, detalhando as etapas de seleção, validação de conflitos e backtracking.

Fonte: Autoria própria (Parte do código original).

3.4 Heurística de Ordenação

Para mitigar a complexidade computacional e evitar a exploração de ramos inférteis na árvore de busca, o algoritmo utiliza uma estratégia de ordenação dos candidatos inspirada na *Degree Heuristic*.

Na implementação atual, a prioridade de alocação não é definida estritamente pelo grau do vértice no grafo, mas sim por uma função de pontuação composta (ρ). Para cada aula candidata v, a pontuação é calculada considerando a carga horária total do professor responsável (C_{prof}) e suas preferências de horário (P_{slot}):

$$\rho(v) = P_{slot} + (C_{prof} \times 0.5) \tag{1}$$

Onde P_{slot} recebe um valor alto (e.g., 100) se o horário for preferencial para o docente, e baixo (e.g., 0) se for um horário a ser evitado. Essa abordagem prioriza a alocação de disciplinas de professores com maior carga horária — que naturalmente possuem maior potencial de conflito ("grau"implícito) — e tenta satisfazer suas preferências pessoais logo no início da busca.

3.5 Avaliação de Qualidade (Scoring)

Para diferenciar soluções válidas e buscar a otimização global da grade, foi implementado um módulo avaliador (método `calcular_pontuacao_global`). A métrica de qualidade S de uma grade completa é definida pela aderência às Restrições Suaves (*Soft Constraints*), especificamente as preferências declaradas pelos docentes.

A pontuação é calculada iterando sobre todas as aulas alocadas:

$$S = \sum_{i=1}^{n} \delta(aula_i) \tag{2}$$

Onde $\delta(aula_i)$ atribui bonificações (+10 pontos) caso a aula esteja alocada em um horário marcado como "Preferencial"pelo professor, e penalizações (-10 pontos) caso esteja em um horário "A evitar". Dessa forma, o algoritmo busca maximizar não apenas a viabilidade técnica (conflitos rígidos), mas também a satisfação do corpo docente.

4 Resultados e Discussão

Os experimentos foram realizados utilizando o dataset fictício que visa representar da maneira mais fiel possível a carga horária dos cursos de Ciência da Computação (CCO) e Sistemas de Informação (SIN). Após o pré-processamento dos dados, a instância final do problema totalizou **140 aulas (vértices)** a serem alocadas em uma grade semanal composta por **20 slots de horário** possíveis (considerando os turnos matutino, vespertino e noturno ao longo de 5 dias úteis).

4.1 Desempenho e Convergência

O algoritmo demonstrou alta eficiência na convergência. Graças à aplicação da heurística de grau, o sistema foi capaz de encontrar uma solução válida completa em poucas iterações, demonstrando que a priorização das disciplinas mais conflitantes é uma estratégia eficaz para este tipo de problema.

4.2 Grade Horária Final

A inspeção da solução final confirmou o cumprimento de 100% das restrições rígidas (*Hard Constraints*). Devido à extensão da grade completa (140 disciplinas), apresenta-se na Figura 4 um recorte detalhado focado no 7º período do curso de Ciência da Computação (CCO).

A visualização evidencia a eficácia da estratégia de **Trilhas de Optativas**: a disciplina eletiva "Gestão e Governança de TI" (SADG01) foi alocada na terça-feira à tarde, encaixando-se perfeitamente com as disciplinas obrigatórias do período (Computação e Sociedade, Metodologia Científica), sem gerar choques para os alunos.

A saída final completa é exportada em formato HTML interativo, permitindo que a coordenação filtre a visualização por qualquer outro período ou professor.

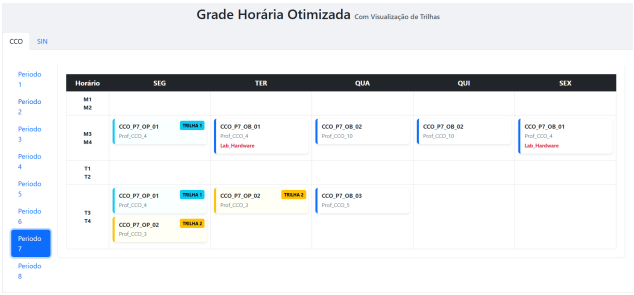

Figura 4: Detalhamento da grade gerada para o 7º período de CCO. Note a alocação sem conflitos entre disciplinas obrigatórias (azul) e optativas da trilha sugerida.

Fonte: Autoria própria.

5 Conclusão

O presente trabalho atingiu seu objetivo ao desenvolver e validar um protótipo capaz de automatizar a alocação de horários acadêmicos.

5.1 Validação das Hipóteses

A hipótese de que a modelagem por Grafos Multicamadas seria suficiente para representar o problema foi confirmada. A adição da lógica de **Trilhas** permitiu capturar a compatibilidade entre optativas e obrigatórias. Além disso, validou-se que o uso de DFS com heurísticas de ordenação é capaz de convergir rapidamente para soluções viáveis, eliminando a necessidade de processos manuais.

5.2 Contribuições

As principais contribuições deste projeto incluem:

1. A implementação da lógica de **Trilhas Acadêmicas** no grafo de conflitos.
2. A criação de um *dataset* digital estruturado e de uma ferramenta de visualização interativa.
3. O desenvolvimento de um algoritmo robusto que garante o cumprimento de todas as restrições rígidas de professores e salas.

6 Trabalhos Futuros

Para aprimorar a qualidade das soluções e estender o escopo de aplicação da ferramenta, propomos as seguintes direções de trabalho futuro:

1. **Avanço em Metaheurísticas:** Embora a solução atual empregue o método de Busca em Profundidade com Reinício Aleatório (*Random Restart*) — configurando uma heurística *Anytime* —, a implementação de metaheurísticas mais robustas, como **Algoritmos Genéticos** ou **Simulated Annealing**, permitiria explorar o espaço de busca de forma mais eficiente, escapando de ótimos locais e potencialmente encontrando soluções com *score* global superior.
2. **Refinamento das Trilhas Optativas:** Expandir o conjunto de regras para a criação e teste de combinações de Trilhas Optativas, passando de um modelo estático para uma geração de agrupamentos dinâmica e otimizada. Isso permitiria testar mais cenários de compatibilidade para os alunos e, consequentemente, reduzir o número total de slots de tempo necessários para o agendamento.
3. **Restrição de Separação Intradia de Aulas:** Implementar uma nova **Restrição Suave** (*Soft Constraint*) que penalize a alocação de partes da mesma disciplina (que possuem o mesmo `ID_Disciplina`) em diferentes slots de tempo **no mesmo dia** (ex: alocar 2 horas de uma matéria de manhã e 2 horas à noite). Esta restrição visa melhorar o conforto e a distribuição da carga de estudo do discente.
4. **Visualização Interativa:** Aprimorar o módulo de visualização (`grade_visual.html`) para incluir filtros, estatísticas de *scoring* por período/professor e a possibilidade de interagir com a grade, facilitando a análise e validação dos resultados por um usuário final.

Contribuições dos Autores

A realização deste projeto foi possível graças à colaboração de todos os membros da equipe:

- **Pedro:** Responsável pela camada de Entrada de Dados. Implementou o pré-processamento do *dataset* utilizando a biblioteca Pandas, realizou a ingestão do arquivo CSV e definiu as estruturas iniciais de *slots* de horário, separando disciplinas obrigatórias de optativas.
- **João:** Atuou na Modelagem Matemática do problema. Utilizando a biblioteca NetworkX, implementou a função de construção do grafo multicamadas, definindo as regras de criação de arestas para conflitos de professores e choques de período.
- **Ivan:** Desenvolveu o núcleo do Algoritmo (Solver). Foi responsável pela implementação da lógica de Busca em Profundidade (DFS) com *Backtracking*, aplicação das heurísticas de ordenação (*Degree Heuristic*) e configuração dos limites de recursão do sistema.
- **Theo:** Implementou o Ciclo de Execução e Visualização. Desenvolveu a lógica de otimização por tempo (*Time-Bounded Loop*) para garantir a melhor solução dentro do

prazo e criou os scripts de exportação e geração da interface visual em HTML.

Link do GitHub:

https://github.com/IvanSilverio/University-Timetabling-Solver

Link do vídeo:

https://youtu.be/Nst9dYsIEVo

Referências

Bello, G. S., Rangel, M. C., & Boeres, M. C. S. (2008). Uma abordagem do problema de programação de grade horária sujeito a restrições utilizando coloração de grafos. Em *XL Simpósio Brasileiro de Pesquisa Operacional* João Pessoa.

Burke, E. K. & Petrovic, S. (2010). University timetabling. Em *Handbook of Applied Optimization*. Springer.

Cooper, T. B. & Kingston, J. H. (1996). The complexity of timetable construction problems. *Practice and theory of automated timetabling*, (pp. 283–295).

Müller, T. (2002). Some novel approaches to lecture timetabling. Em *CPDC*.

Vieira, F. & Macedo, H. (2011). Sistema de alocação de horários de cursos universitários. *Scientia Plena*, 7(3).