LVIII SBPO
Belo Horizonte - MG | 2026

Otimização de Grades Horárias Acadêmicas com GRASP e *Simulated Annealing*: Um Estudo de Caso em um Instituto Federal

João Vitor de Melo
Carlos Alexandre Silva
Bruno Nonato Gomes

Instituto Federal de Minas Gerais – Campus Sabará (IFMG-Sabará)
Sabará – MG, Brasil

joaovitor502013@gmail.com
{carlos.silva,bruno.nonato}@ifmg.edu.br

RESUMO

Este artigo apresenta um *solver* híbrido para o problema de *timetabling* acadêmico em um Instituto Federal. A abordagem combina uma heurística construtiva inspirada no GRASP com *Simulated Annealing*, operadores de vizinhança e calibração automática de hiperparâmetros. O problema foi modelado por uma função objetivo baseada em penalidades para restrições fortes e fracas, contemplando conflitos de professores e turmas, indisponibilidades, limites pedagógicos e critérios de qualidade da grade. Foram avaliadas três instâncias derivadas de dados reais, com 127, 255 e 382 blocos semanais de aula, em 30 repetições computacionais com sementes pseudoaleatórias distintas. Os resultados indicaram geração de soluções viáveis, sem violações de restrições fortes, com baixa variação dos custos finais e tempos compatíveis com o uso institucional. O *software* comercial utilizado como referência operacional não produziu, nas configurações testadas, grades completas sem intervenção manual substancial, reforçando a necessidade de soluções customizadas.

PALAVRAS CHAVE. ***Timetabling*** **Educacional; *Simulated Annealing*; GRASP; Otimização Combinatória; Meta-heurísticas.**

Tópicos: OA – Outras Aplicações em PO; MH – Meta-heurísticas.

ABSTRACT

This paper presents a hybrid solver for the academic timetabling problem in a Federal Institute. The proposed approach combines a GRASP-inspired constructive heuristic with *Simulated Annealing*, neighborhood operators, and automatic hyperparameter calibration. The problem is modeled through a penalty-based objective function that accounts for hard and soft constraints, including teacher and class conflicts, unavailability, pedagogical limits, and timetable quality criteria. Three real-data-based instances with 127, 255, and 382 weekly class blocks were evaluated through 30 computational repetitions using distinct pseudorandom seeds. The results indicate that the solver generated feasible timetables without hard-constraint violations, with low variation in final costs and execution times compatible with institutional use. The commercial software adopted as an operational reference did not produce, under the tested configurations, complete timetables without substantial manual intervention, highlighting the relevance of customized optimization tools.

KEYWORDS. Educational Timetabling; Simulated Annealing; GRASP; Combinatorial Optimization; Metaheuristic.

Paper topics: OR – Other Applications in OR; MH – Metaheuristics.

1. Introdução

A alocação de horários de aulas, tratada na literatura como problema de *timetabling*, é um dos desafios clássicos da otimização combinatória. A resolução do problema envolve a organização de eventos (aulas), recursos (professores, turmas e salas) e períodos de tempo de forma a atender simultaneamente a um conjunto de restrições. Em contextos reais, conflitos de professores e turmas, indisponibilidades, limites de carga horária, turnos de oferta e critérios pedagógicos tornam a elaboração manual de grades uma tarefa complexa, demorada e sujeita a ajustes sucessivos.

No *campus* analisado, a construção de horários é apoiada pelo *software* comercial *aScTimetables*, mas ainda depende de intervenção humana significativa para adequação às regras institucionais. Embora ferramentas comerciais ofereçam interface gráfica e mecanismos automáticos de verificação, seu uso pode ser limitado quando há muitas restrições locais, regras pedagógicas específicas e necessidade de transparência sobre a função de qualidade adotada. Esse cenário motiva o desenvolvimento de abordagens customizadas, capazes de incorporar diretamente as regras do *campus* e produzir soluções reprodutíveis em tempo computacional compatível com a rotina de gestão acadêmica.

Diante desse contexto, este artigo apresenta um *solver* híbrido para o problema de *timetabling* acadêmico. A abordagem combina uma fase construtiva inspirada no *Greedy Randomized Adaptive Search Procedure* (GRASP) com uma etapa de refinamento baseada em *Simulated Annealing* (SA) e operadores de busca local. O *software aScTimetables* é utilizado como referência operacional do processo atualmente empregado, enquanto a avaliação experimental concentra-se na viabilidade, estabilidade operacional e convergência do *solver* proposto em instâncias derivadas de dados reais. A contribuição central do trabalho está na integração, em um mesmo *solver*, de uma função de penalidades hierarquizada para regras institucionais, uma seleção probabilística de operadores exploratórios e intensivos guiada pela temperatura e uma calibração automática de hiperparâmetros via Optuna/TPE, avaliadas em instâncias reais de complexidade crescente.

O restante do artigo está organizado da seguinte forma: a Seção 2 apresenta os trabalhos relacionados; a Seção 3 descreve a metodologia e a modelagem do problema; a Seção 4 detalha a meta-heurística proposta; a Seção 5 apresenta os resultados experimentais; e a Seção 6 traz as conclusões.

2. Trabalhos Relacionados

A literatura sobre *timetabling* é ampla e diferencia variantes como *school timetabling*, *university course timetabling*, *curriculum-based course timetabling* e *examination timetabling*, que compartilham uma estrutura geral semelhante, mas diferem quanto às restrições fortes, restrições fracas e critérios de qualidade adotados [Schaerf, 1999; Ceschia et al., 2023]. A literatura clássica mostra que versões do problema de *timetabling* são NP-completas [Even et al., 1976], o que justifica o uso de heurísticas e meta-heurísticas em instâncias reais. Modelos de programação inteira oferecem formulações rigorosas, mas podem se tornar custosos em cenários com muitas regras específicas, indisponibilidades, limites de carga horária, agrupamento de aulas e janelas ociosas [Fonseca et al., 2017].

Por isso, heurísticas e meta-heurísticas são amplamente utilizadas em problemas educacionais de *timetabling*, incluindo algoritmos genéticos, busca tabu, busca local, *Simulated Annealing*, GRASP e métodos híbridos [Abdipoor et al., 2023]. O *Simulated Annealing*, introduzido por Kirkpatrick et al. [1983], aceita movimentos de piora de forma controlada, reduzindo o risco de aprisionamento prematuro em ótimos locais. O GRASP combina construção gulosa-randomizada e busca local, sendo adequado para gerar soluções iniciais diversificadas e de boa qualidade [Feo e Resende, 1995]. No contexto brasileiro, há trabalhos que exploram estratégias híbridas para programação de horários em escolas, incluindo algoritmo evolutivo híbrido e uma abordagem GRASP

com Busca Tabu [Souza et al., 2002, 2003]. No SBPO, estudos recentes trataram o problema em Institutos Federais e em revisões sobre busca local aplicada à programação de horários escolares [Queiroz et al., 2020; Oliveira et al., 2024; Santos et al., 2024].

Ferramentas comerciais, como o *aScTimetables*, são usadas em instituições de ensino por oferecerem interface gráfica, verificação de conflitos e geração automática de horários, conforme descrito no manual do *software* [Applied Software Consultants, 2026]. Entretanto, a literatura que utiliza esse *software* como objeto de comparação experimental ainda é limitada, sendo mais comum seu uso como ferramenta operacional [Pupeikiene e Mockus, 2010]. Neste trabalho, o *aScTimetables* é tratado como referência operacional, pois corresponde à ferramenta já utilizada no contexto analisado. A solução proposta, implementada em Julia e baseada em GRASP e *Simulated Annealing*, utiliza uma função objetivo que contempla as mesmas restrições do *software* comercial, apresentando, contudo, maior complexidade ao incorporar regras institucionais adicionais não utilizadas pela ferramenta de referência.

3. Metodologia e Modelagem

3.1. Objeto de estudo

Esta pesquisa, de natureza aplicada e quantitativa, avalia um *solver* customizado para a otimização da grade de um *campus* de um Instituto Federal. A instância real de referência engloba 5 cursos, sendo 4 superiores e 1 técnico integrado, com 27 turmas, 62 docentes e 3 turnos. Os dados de entrada e as regras de negócio foram extraídos de registros oficiais e refinados por meio de levantamento junto aos gestores responsáveis pela elaboração dos horários, de modo a representar o nível de complexidade e as restrições pedagógicas do cenário institucional analisado. Nesta versão, as salas são assumidas como previamente definidas e não entram como variável de decisão.

3.2. Modelagem Matemática

O problema de *timetabling* foi formulado como otimização combinatória baseada em penalidades. Sejam A o conjunto de todas as aulas a serem alocadas, D o conjunto de dias letivos da semana e S o conjunto de *slots* diários. A alocação é regida pela variável de decisão binária $x_{a,d,s}$, definida como:

$$x_{a,d,s} = \begin{cases} 1, & \text{se a aula } a \text{ é alocada no dia } d \text{ e horário } s, \\ 0, & \text{caso contrário.} \end{cases}$$

Para reduzir o custo de avaliação, os metadados textuais das aulas (professores, turmas e cursos) são previamente sanitizados e convertidos em identificadores inteiros. Essa representação substitui comparações recorrentes de *strings* por consultas a vetores e dicionários pré-computados. A base original contempla 764 aulas atômicas de 50 minutos, distribuídas em $|D| = 5$ dias letivos e 16 *slots* diários, totalizando $764 \times 5 \times 16 = 61.120$ variáveis binárias. Como 96,20% dessas aulas (735 de 764) são lecionadas em blocos duplos de 1h40, as aulas foram consolidadas em $|A| = 382$ blocos e a matriz temporal foi reduzida para $|S| = 8$ *slots* operacionais, resultando em $382 \times 5 \times 8 = 15.280$ variáveis binárias. O mapeamento adotado é apresentado na Tabela 1; os *slots* 5 e 6 representam janelas alternativas parcialmente sobrepostas e são tratados como mutuamente exclusivos para a mesma turma ou professor.

Tabela 1: Mapeamento dos *slots* para o horário físico da instituição

Índice	Turno Letivo	Horário Físico	Duração
Slot 1	Manhã	07:00 – 08:40	1h40
Slot 2	Manhã	08:50 – 10:30	1h40
Slot 3	Manhã	10:40 – 12:20	1h40
Slot 4	Tarde	14:00 – 15:40	1h40
Slot 5	Tarde	15:50 – 17:30	1h40
Slot 6	Tarde / Noite	17:00 – 18:40	1h40
Slot 7	Noite	18:50 – 20:30	1h40
Slot 8	Noite	20:40 – 22:20	1h40

3.3. Restrições Fortes (*Hard Constraints*)

As restrições fortes representam as condições físicas e lógicas intransponíveis da instituição. Qualquer grade que viole essas regras é considerada inviável para operação real. Na abordagem computacional proposta, a modelagem permite que o algoritmo navegue temporariamente por zonas inviáveis para evitar ótimos locais, aplicando uma penalidade máxima restritiva ($w = 10000$) por restrição forte violada na função objetivo.

As restrições centrais de unicidade impedem que professores e turmas recebam mais de uma aula no mesmo período. Sejam $\mathcal{P}$ o conjunto de professores, $\mathcal{C}$ o conjunto de turmas, A_p o conjunto de aulas associadas ao professor p e A_c o conjunto de aulas associadas à turma c. Essas condições podem ser expressas por:

$$\sum_{a \in A_p} x_{a,d,s} \leq 1, \quad \forall p \in \mathcal{P},\ \forall d \in D,\ \forall s \in S, \qquad \sum_{a \in A_c} x_{a,d,s} \leq 1, \quad \forall c \in \mathcal{C},\ \forall d \in D,\ \forall s \in S.$$

- **Unicidade:** impede choques de professores e turmas no mesmo horário.
- **Legislação e Descanso:** garante limites de carga horária diária, intervalos legais, respeito ao descanso entre turnos (interjornada), limites de aulas consecutivas (intrajornada) e pelo menos um dia livre sem ministração de aula na semana para o docente.
- **Disponibilidade Docente:** respeito estrito às indisponibilidades oficiais previamente declaradas pelos professores, como, por exemplo, restrições médicas ou de pós-graduação.
- **Diretrizes Pedagógicas:** respeito ao turno de oferta, coesão de disciplinas extensas e obediência ao limite máximo de aulas diárias para o ensino médio técnico.

3.4. Restrições Fracas (*Soft Constraints*)

Enquanto as restrições fortes garantem a viabilidade operacional da grade, as restrições fracas referem-se à qualidade pedagógica e operacional da solução. Sua violação não invalida o quadro de horários, mas reduz a qualidade da solução. Tais restrições foram modeladas a partir da coleta de conhecimento junto à coordenação pedagógica, traduzindo diretrizes institucionais em métricas de otimização. O conjunto de regras implementadas no motor de busca compreende:

- **Prevenção de Exaustão:** penaliza a concentração da carga de aulas de um docente em um único dia e aulas geminadas excessivas, evitando esgotamento discente e docente.
- **Sincronismo de Subturmas:** pareia a alocação de divisões práticas, garantindo que a turma original chegue e saia do *campus* unificada.
- **Empregabilidade Discente:** penaliza aulas no último horário matutino para turmas concluintes de cursos superiores, visando liberar os discentes para estágio.
- **Compactação Fracionada:** força disciplinas com frações irregulares (50 min ou 2h30) para o primeiro ou último horário do turno.

- **Ociosidade Discente:** aplica penalidade quadrática (x^2) para buracos na grade e penaliza deslocamentos para uma única aula no dia.
- **Minimização de Presença:** penaliza a quantidade total de dias em que professores e turmas precisam comparecer fisicamente à instituição.

A calibração das penalidades estabelece uma hierarquia entre os níveis de exigência. A atribuição de pesos prioriza a eliminação de violações de restrições fortes antes do refinamento das restrições fracas, sem impedir que diferentes combinações de penalidades influenciem a função objetivo. Os pesos adotados são apresentados na Tabela 2.

Tabela 2: Pesos das Penalidades para a Função Objetivo

Variável (Código)	**Tipo/Severidade**	**Foco da Restrição**	**Peso (w_k)**
*Hard Constraints**	Forte (Crítica)	Conflitos físicos e diretrizes legais	10000
`prof_um_dia`, `subturma`	Fraca (Grave)	Carga concentrada ou descompasso A/B	2000
`frag_fora_ponta`	Fraca (Grave)	Frações (50/150 min) no meio do turno	2000
`veteranos_bsi`	Fraca (Média)	Veda último horário para formandos BSI	1000
`aula_unica`	Fraca (Leve)	Alocação isolada no dia letivo	50
`buraco_turma`	Fraca (Leve)	Ociosidade discente (Custo Quadrático)	20
`geminada_exces`	Fraca (Ajuste)	Blocos excessivos ininterruptos	10
`dia_turma`	Fraca (Ajuste)	Minimização de deslocamento semanal	3
`dia_prof`	Fraca (Ajuste)	Minimização de deslocamento semanal	2

***Hard Constraints:** indisponibilidade, conflito_professor, conflito_turma, fora_janela, interjornada, intrajornada, aulas_tecnico, aula_tripla_separada e professor_sem_folga.

3.5. Função Objetivo

O objetivo central do modelo é encontrar uma matriz de alocação X que minimize o custo total das violações. A avaliação de qualidade da solução é calculada por uma função objetivo $f(X)$, que realiza o somatório ponderado de todas as falhas detectadas:

$$\min f(X) = \sum_{k=1}^{K} w_k \cdot V_k(X)$$

Em que K representa o número total de regras avaliadas (fortes e fracas), w_k é o peso (nível de gravidade) atribuído institucionalmente à regra k, e $V_k(X)$ é o número de vezes que a referida regra foi violada na configuração X. Por meio dessa função de custo escalar, o algoritmo compara diferentes soluções vizinhas e direciona a busca para configurações de menor penalidade.

4. Meta-heurística

Como discutido na literatura, variantes de *timetabling* educacional apresentam elevada complexidade combinatória, o que inviabiliza a busca exata em tempo polinomial para instâncias reais. Diante dessa complexidade, o algoritmo *Simulated Annealing* (SA) destaca-se como uma meta-heurística eficaz, inspirada no recozimento de metais. O SA opera sob o princípio de aceitação estocástica: além de aceitar movimentos de melhoria, o algoritmo possui uma probabilidade controlada de aceitar movimentos de piora para escapar de mínimos locais.

Essa transição entre estados é governada pelo cálculo da variação de energia $\Delta E = f(S') - f(S)$. Enquanto movimentos que reduzem o custo ($\Delta E \leq 0$) são sempre aceitos, movimentos de piora ($\Delta E > 0$) são submetidos a uma probabilidade de aceitação estocástica, sendo aceitos se um valor aleatório $U(0, 1)$ for menor que o limiar definido por:

$$P = e^{-\frac{\Delta E}{T}}$$

Essa capacidade de aceitar soluções temporariamente inferiores é a principal vantagem do SA frente a técnicas de busca local puras. O algoritmo inicia com uma temperatura (T) alta, permitindo ampla exploração do espaço de busca. À medida que o sistema esfria, a probabilidade de aceitar movimentos ruins decresce, forçando o refinamento da melhor solução encontrada até a estabilização da grade. Para garantir que o SA inicie em uma zona promissora, implementou-se uma fase construtiva prévia inspirada na heurística GRASP.

4.1. Geração da Solução Inicial: Heurística Construtiva (GRASP)

O desempenho da meta-heurística tende a melhorar quando a solução inicial (S_0) já mitiga falhas triviais. Para evitar a convergência precoce para ótimos locais, típica de inicializações puramente gulosas, a geração da grade primária foi modelada sob o paradigma GRASP.

Durante a construção, o algoritmo avalia as restrições físicas para cada disciplina e calcula o custo parcial das inserções válidas. Em vez de escolher a alocação de menor custo, constrói-se uma Lista Restrita de Candidatos (RCL) ordenada. A heurística aplica um fator de aleatoriedade ($\alpha_{construtivo} = 0,3$) para selecionar o horário mediante sorteio restrito aos 30% melhores candidatos. Essa margem exploratória favorece a geração de soluções iniciais diversificadas e estruturalmente promissoras, transferindo ao *Simulated Annealing* o refinamento das soluções.

4.2. Operadores de Vizinhança

A busca local utiliza cinco operadores de perturbação para gerar a solução vizinha S'. A seleção dos operadores é probabilística, utilizando uma roleta com probabilidades estáticas com pesos conforme a temperatura:

1. **Relocação (*Shift*):** Move uma aula para um *slot* ocioso, visando preencher janelas.
2. **Troca (*Swap*):** Permuta os horários de duas aulas distintas para resolver conflitos de alocação e otimizar a qualidade geral da grade.
3. **Reorganização (*Shuffle*):** Reordena a grade diária de uma única turma para fechar lacunas internas.
4. **Realocação Intelig. (*Shift-HC*):** atua como *Hill Climbing* de vizinhança amostrada. Avalia dez realocações aleatórias e aplica estritamente a de menor custo. Aborta caso nenhuma reduza o custo atual.
5. **Troca Intelig. (*Swap-HC*):** sob a mesma lógica gulosa de amostragem, testa dez trocas aleatórias e efetiva apenas a de menor custo absoluto, abortando se não houver melhoria.

A roleta ajusta-se dinamicamente (Tabela 3), ela prioriza movimentos globais de diversificação (1, 2 e 3) no início da busca e passa a privilegiar operadores locais intensivos (4 e 5) ao cruzar o limiar de refinamento.

Tabela 3: Probabilidades de seleção dos operadores por fase

ID	**Operador de Vizinhança**	**Fase Explor.** ($T > Lim$)	**Fase Refin.** ($T \leq Lim$)
1	Mover Simples (*Shift*)	35%	22%
2	Trocar Simples (*Swap*)	35%	22%
3	Reorganizar Turma (*Shuffle*)	20%	6%
4	Realocação Intelig. (*Shift-HC*)	5%	35%
5	Troca Intelig. (*Swap-HC*)	5%	15%
Busca Local Intensiva (4 e 5)		**10%**	**50%**

Para reduzir o custo computacional, o *solver* manipula diretamente a estrutura da solução (*in-place*), evitando cópias integrais da matriz de horários a cada iteração. Quando um movimento é rejeitado pela política de aceitação, a alteração é revertida por meio do registro do movimento aplicado, preservando a eficiência da avaliação sucessiva de vizinhos.

4.3. Dinâmica de Resfriamento e Hiperparâmetros do *Simulated Annealing*

A eficácia do *Simulated Annealing* depende do esquema de resfriamento. Um decaimento de temperatura muito abrupto pode estabilizar a busca em ótimos locais, enquanto um resfriamento excessivamente lento aumenta o tempo de processamento. Para adequar o método à densidade do problema analisado, o algoritmo adota resfriamento geométrico dinâmico e calibração da temperatura inicial inspirada em procedimentos usuais da literatura, nos quais a temperatura é ajustada até atingir uma taxa mínima de aceitação de movimentos de piora [Gomes Júnior et al., 2005; Souza e Penna, 2021]. Os principais hiperparâmetros são:

- **Aquecimento e temperatura inicial (T_0):** o algoritmo executa uma rotina prévia de calibração, avaliando movimentos aleatórios e ajustando a temperatura pelo fator β até que a taxa de aceitação de movimentos de piora alcance o alvo γ [Souza e Penna, 2021].

- **Decaimento geométrico (α):** após a definição de T_0, a temperatura decai a cada ciclo segundo $T_{i+1} = \alpha \cdot T_i$, controlando a transição entre exploração e refinamento.

- **Constante (K):** o número de iterações por temperatura é dado por $SA_{max} = \lceil K \times |Aulas| \rceil$, permitindo avaliar um conjunto proporcional de vizinhos antes da próxima redução.

- **Critérios de parada:** o ciclo é interrompido quando a temperatura mínima (T_f) é atingida ou quando o limite de estagnação é alcançado após ciclos sucessivos sem melhoria no melhor custo encontrado [Souza e Penna, 2021].

4.4. Calibração de Hiperparâmetros

A convergência do método exige a sintonia dos parâmetros de temperatura, resfriamento, platô e estagnação. Devido à interdependência entre essas variáveis, uma calibração apenas manual pode levar a configurações pouco estáveis para instâncias com muitas restrições.

Para apoiar essa etapa, utilizou-se o *framework* Optuna com o algoritmo TPE (*Tree-structured Parzen Estimator*). O procedimento explorou diferentes combinações de parâmetros e selecionou a configuração apresentada na Tabela 4. O tempo de calibração não foi incluído nos tempos de execução reportados, que correspondem apenas às execuções finais do *solver* com os parâmetros fixados.

Tabela 4: Hiperparâmetros selecionados para o *Simulated Annealing*

Parâmetro	**Símbolo**	**Valor Calibrado**
Taxa de Aceite Inicial	γ	0,8635
Fator de Incremento	β	1,1785
Fator de Resfriamento	α	0,9982
Limiar de Refinamento	Lim	0,35
Constante k	K	4,1749
Máximo de Estagnação	`max_sem_melhoria`	11.500
Temperatura Final	T_{final}	0,01

4.5. Pseudocódigo do algoritmo

O pseudocódigo do Algoritmo 1 resume o *Simulated Annealing* proposto, integrando a inicialização heurística, a avaliação *in-place* e a seleção probabilística de operadores. Ele ilustra o laço principal da meta-heurística e as condições de atualização da temperatura discutidas nas seções anteriores.

Algoritmo 1 Meta-heurística Híbrida: SA com Seleção Probabilística de Operadores

Entrada: Instância I, hiperparâmetros (Tabela 4)
Saída: Melhor grade S_{best}

$S \leftarrow$ Gerar_Inicial_GRASP(I); $T \leftarrow$ Calibrar_Aquecimento(S); $T_0 \leftarrow T$; $S_{best} \leftarrow S$
$estagnacao \leftarrow 0$; $T_{\text{final}} \leftarrow 0{,}01$; $L \leftarrow \lceil K \times |Aulas| \rceil$
while $(T > T_{\text{final}})$ **e** $(estagnacao < max\_sem\_melhoria)$ **do**
  $melhorou \leftarrow$ **falso**
  for $i \leftarrow 1$ **até** L **do**
    if $(T/T_0) \leq Lim$ **then**
      $op \leftarrow$ Selecionar_Roleta_Refinamento()
    else
      $op \leftarrow$ Selecionar_Roleta_Exploratoria()
    end if
    $S' \leftarrow$ Aplicar_Movimento_In_Place(S, op)
    $\Delta E \leftarrow f(S') - f(S)$
    if $(\Delta E \leq 0)$ **ou** $(aleatorio(0,1) < e^{-\Delta E/T})$ **then**
      $S \leftarrow S'$
      if $f(S) < f(S_{best})$ **then**
        $S_{best} \leftarrow S$; $melhorou \leftarrow$ **verdadeiro**
      end if
    else
      Reverter_Movimento_O1(S)
    end if
  end for
  if $melhorou$ **then** $estagnacao \leftarrow 0$
  else $estagnacao \leftarrow estagnacao + 1$
  end if
  $T \leftarrow T \times \alpha$
end while
return S_{best}

5. Resultados

Os experimentos computacionais foram conduzidos em um ambiente equipado com processador Intel Core i5-13400F, com frequência base de 2,50 GHz, e 16 GB de memória RAM DDR4. O modelo foi implementado em Julia (versão 1.10.x), sob o sistema operacional Windows 10 Pro. As instâncias utilizadas foram derivadas de dados reais do ano letivo de 2026, contemplando cursos técnicos e superiores. A base integral considerada possui 5 cursos, sendo 4 superiores (manhã e noite) e 1 técnico integrado (manhã e tarde), com 62 docentes e 27 turmas no período 2026/01, com 382 blocos semanais de aula alocados em *slots* de 1h40.

Visando avaliar a estabilidade e convergência do *solver*, elaboraram-se três instâncias a partir de 2026/01, detalhadas na Tabela 5. A pequena possui 127 blocos semanais de aula (12 turmas, 31 professores); a média, 255 blocos (15 turmas, 52 docentes); e a completa abrange o cenário integral com 382 blocos (27 turmas, 62 professores).

Tabela 5: Escalonamento das instâncias para avaliação experimental

Instância	Cursos	Profs.	Turmas	Disciplinas	Total de Blocos*
Pequena	3	31	12	69	127
Média	2	52	15	115	255
Completa	5	62	27	175	382

* Total de Blocos refere-se ao número de *slots* semanais de 1h40 alocados.

5.1. Limitações do *Software* Comercial e Justificativa da Análise

O *aScTimetables* foi considerado como referência operacional por ser a ferramenta comercial utilizada no processo institucional. Nas tentativas realizadas, o *software* não gerou grades completas e diretamente utilizáveis para as instâncias propostas sem intervenção manual substancial. Como não houve uma solução final completa produzida exclusivamente pela ferramenta, não foi possível calcular a mesma função objetivo para uma grade comparável. A Tabela 6 sintetiza as evidências operacionais observadas.

Tabela 6: Tentativas com o *aScTimetables*

Instância	Tempo	Horários verif.	Cartões não alocados
Pequena	00:54:29	426.443.104	35
Média	02:49:16	466.075.581	114
Completa	04:28:00	965.213.571	149

Constata-se que o *aScTimetables* encerrou as três tentativas sem obter uma solução completa para as instâncias avaliadas. No *software*, os cartões podem representar aulas de 50 min, 1h40 ou 2h30, sendo informada apenas a quantidade não alocada. Além dessa falha de cobertura, foram observadas inconsistências em restrições institucionais relevantes, especialmente associadas a descanso e sequenciamento de aulas docentes. Como essas inconsistências não foram sistematicamente quantificadas, o *software* foi mantido apenas como referência operacional, e não como *baseline* quantitativo direto.

5.2. Análise dos Resultados

Cada instância foi executada 30 vezes com sementes pseudoaleatórias distintas, permitindo observar a variabilidade do método sob diferentes trajetórias de busca. Em todos os testes, as soluções finais foram viáveis, sem violações de restrições fortes. A Tabela 7 resume custos finais e tempos de execução, enquanto a Figura 1 apresenta a curva de convergência da instância completa.

Tabela 7: Resumo experimental das 30 repetições com sementes pseudoaleatórias distintas

Instância	Custo médio	DP custo	Custo mín.–máx.	Tempo médio (s)	Tempo mín.–máx. (s)
Pequena	324,93	2,70	319–330	11,80	11,61–12,22
Média	427,53	3,48	421–436	61,22	59,89–62,59
Completa	712,67	7,23	701–739	304,59	287,68–321,72

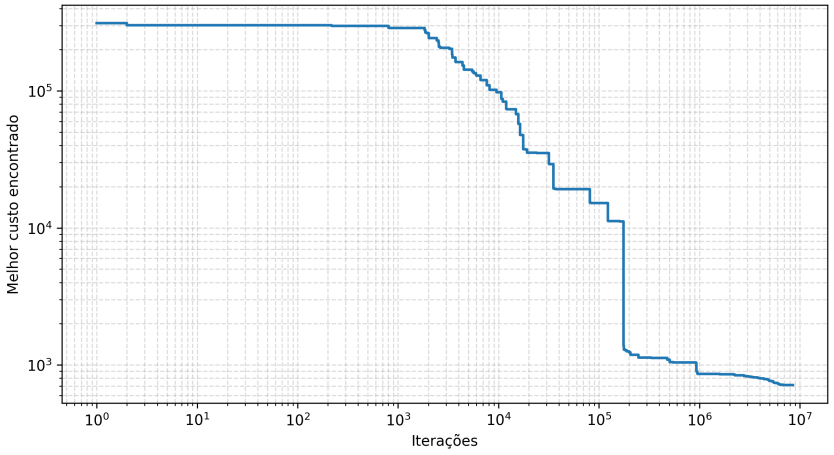

Figura 1: Curva de convergência do melhor custo encontrado na instância completa.

Na execução representativa da instância completa usada na Figura 1, a solução inicial apresentou custo 312 149, enquanto a solução final atingiu custo 712, valor compatível com o intervalo observado nas 30 execuções (701–726). A redução aproximada de 99,77% evidencia o papel do SA na redução das penalidades residuais.

Como não houve violações fortes nas soluções finais, o custo remanescente corresponde integralmente a penalidades de restrições fracas. Na configuração analisada, essa parcela está associada principalmente à minimização de presença semanal, representada por `dia_turma` e `dia_prof`. As três instâncias apresentaram baixa variação de custo nas 30 repetições, sugerindo estabilidade da configuração calibrada no ambiente experimental.

Em relação ao tempo de processamento, a instância completa foi resolvida em aproximadamente 12,5 minutos em média, com tempos entre 729,81 s e 768,99 s. Como a Figura 1 representa o melhor custo encontrado ao longo da execução, a visualização em degraus é esperada.

Discussão e ameaças à validade

A interpretação dos resultados deve considerar que a função objetivo é específica do contexto institucional analisado. Os valores absolutos de custo não devem ser lidos como indicadores universais de qualidade, mas como medidas comparáveis dentro do mesmo conjunto de pesos, restrições e instâncias. Como as soluções finais não apresentam violações fortes, os custos remanescentes indicam penalizações associadas a critérios de qualidade. A decomposição detalhada por tipo de penalidade não foi registrada nesta etapa, o que limita uma análise mais granular da composição do custo final.

A baixa variação observada nas 30 repetições com sementes pseudoaleatórias distintas sugere estabilidade da configuração calibrada no ambiente experimental, mas não constitui, por si só, uma avaliação plena da robustez do método. Além disso, a comparação direta com o *aScTimetables* depende da obtenção de uma grade completa e mensurável pela mesma função objetivo; nas condições avaliadas, isso não foi possível porque permaneceram cartões não alocados nas três instâncias.

6. Conclusão

Este artigo apresentou um *solver* híbrido baseado em GRASP e *Simulated Annealing* para o problema de *timetabling* acadêmico em um Instituto Federal. A abordagem modela restrições fortes e fracas por meio de uma função objetivo baseada em penalidades e incorpora operadores de vizinhança, seleção probabilística de operadores e calibração automática de hiperparâmetros.

Os experimentos com três instâncias derivadas de dados reais indicaram que o método foi capaz de gerar grades viáveis, sem violações de restrições fortes, com baixa variação dos custos finais nas 30 repetições com sementes pseudoaleatórias distintas e tempos compatíveis com o uso institucional. A instância completa, composta por 382 blocos semanais de aula, foi processada em aproximadamente 12,5 minutos em média. Esses resultados sugerem que o *solver* pode apoiar a construção de horários em cenários com muitas regras institucionais e necessidade de ajuste rápido.

Como limitação da pesquisa, ressalta-se que a comparação quantitativa com o *software* *aScTimetables* não pôde ser consolidada, visto que a ferramenta comercial falhou em produzir grades completas e legalmente viáveis nas condições avaliadas, mantendo cartões não alocados nas três instâncias. Trabalhos futuros devem focar no desenvolvimento de métricas para mensurar a diversidade geométrica das soluções finais, verificando se custos semelhantes produzem grades estruturalmente diferentes. Além disso, recomenda-se avaliar o impacto isolado da etapa GRASP e dos operadores de vizinhança, testar diferentes configurações de pesos para conforto docente e comparar a abordagem meta-heurística adotada com outros modelos.

Referências

Abdipoor, S., Yaakob, R., Goh, S. L., e Abdullah, S. (2023). Meta-heuristic approaches for the university course timetabling problem. *Intelligent Systems with Applications*, 19:200253. DOI: 10.1016/j.iswa.2023.200253.

Applied Software Consultants (2026). asc timetables manual. https://www.automatictimetable.com/aScTimeTables_Manual_English.pdf. Acessado em: 25 abr. 2026.

Ceschia, S., Di Gaspero, L., e Schaerf, A. (2023). Educational timetabling: Problems, benchmarks, and state-of-the-art results. *European Journal of Operational Research*, 308(1):1–18. DOI: 10.1016/j.ejor.2022.07.011.

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

Feo, T. A. e Resende, M. G. C. (1995). Greedy randomized adaptive search procedures. *Journal of Global Optimization*, 6(2):109–133. DOI: 10.1007/BF01096763.

Fonseca, G. H. G., Santos, H. G., Carrano, E. G., e Stidsen, T. J. R. (2017). Integer programming techniques for educational timetabling. *European Journal of Operational Research*, 262(1):28–39. DOI: 10.1016/j.ejor.2017.03.020.

Gomes Júnior, A. d. C., Souza, M. J. F., e Martins, A. X. (2005). Simulated Annealing aplicado à resolução do problema de roteamento de veículos com janela de tempo. *Transportes*, 13(2):5–20. URL https://transportes.anpet.org.br/anpet/article/view/98.

Kirkpatrick, S., Gelatt, C. D., e Vecchi, M. P. (1983). Optimization by simulated annealing. *Science*, 220(4598):671–680. DOI: 10.1126/science.220.4598.671.

Oliveira, T. F., Real, L. B., e Mapa, S. M. S. (2024). Otimização da grade de horários: um estudo de caso no Instituto Federal de Minas Gerais. In *Anais do LVI Simpósio Brasileiro de Pesquisa Operacional*, Fortaleza. Galoá. DOI: 10.59254/sbpo-2024-193391.

Pupeikiene, L. e Mockus, J. (2010). High school time table problem solving and comparison with automatic optimized timetable. *Proceedings of Information Technology*, p. 164–172.

Queiroz, R. S., Mesquita, L. M. O., Lima, V. G. d., Marques, C. A. N., e Frutuoso, R. L. (2020). Proposta de um modelo para o School Timetabling Problem em um Instituto Federal de educação, ciência e tecnologia. In *Anais do LII Simpósio Brasileiro de Pesquisa Operacional*, João Pessoa. Galoá. DOI: 10.59254/sbpo-2020-122645.

Santos, T. E. d., Vilela, S. G., Santana, A. R., e Amaral, H. F. (2024). Resolução do problema de programação de horários escolares usando busca local: Uma revisão de literatura. In *Anais do LVI Simpósio Brasileiro de Pesquisa Operacional*, Fortaleza. Galoá. DOI: 10.59254/sbpo-2024-193614.

Schaerf, A. (1999). A survey of automated timetabling. *Artificial Intelligence Review*, 13(2):87–127. DOI: 10.1023/A:1006576209967.

Souza, M. J. F. e Penna, P. H. V. (2021). Simulated annealing. Notas de aula de Técnicas Metaheurísticas para Otimização Combinatória. URL http://www.decom.ufop.br/prof/marcone/Disciplinas/InteligenciaComputacional/SimulatedAnnealing.pptx. Acessado em: 3 de maio de 2026.

Souza, M. J. F., Guimarães, I. F. G., e Costa, F. P. (2002). Um algoritmo evolutivo híbrido para o problema de programação de horários em escolas. In *Anais do XXII Encontro Nacional de Engenharia de Produção*, p. 1–8, Curitiba. ABEPRO. Disponível em: http://www.decom.ufop.br/marcone/Publicacoes/ENEGEP-2002-PHE.pdf.

Souza, M. J. F., Maculan, N., e Ochi, L. S. (2003). A grasp-tabu search algorithm for solving school timetabling problems. In Resende, M. G. C. e Pinho de Sousa, J., editors, *Metaheuristics: Computer Decision-Making*, p. 659–672. Kluwer Academic Publishers, Boston. DOI: 10.1007/978-1-4757-4137-7_31.