ALGORITMO HÍBRIDO ITERATED LOCAL SEARCH E SIMULATED ANNEALING PARA O PROBLEMA DE TABELA-HORÁRIO DE UNIVERSIDADES

Renan Costalonga Monteiro
Universidade Federal do Espírito Santo
Alto Universitário, s/nº - Cx. Postal 16, Guararema – CEP: 29500-000 – Alegre-ES
renanmonteiro@msn.com

Edmar Hell Kampke
Universidade Federal do Espírito Santo
Alto Universitário, s/nº - Cx. Postal 16, Guararema – CEP: 29500-000 – Alegre-ES
edmar.kampke@ufes.br

Dayan de Castro Bissoli, Geraldo Regis Mauri
Universidade Federal do Espírito Santo
Alto Universitário, s/nº - Cx. Postal 16, Guararema – CEP: 29500-000 – Alegre-ES
{dayan.bissoli, geraldo.mauri}@ufes.br

RESUMO

No contexto acadêmico encontra-se o Problema de Tabela-Horário em Universidades, caracterizado como um processo rotineiro e complexo. O problema consiste em alocar um conjunto de aulas em salas disponíveis e períodos de tempo pré-determinados, considerando alunos e professores, e satisfazendo algumas restrições. Neste trabalho, é adotado o modelo baseado em currículo de cursos, proposto no segundo campeonato internacional de tabela-horário (ITC-2007). Para a solução do problema, é utilizada a meta-heurística *Iterated Local Search* com a meta-heurística *Simulated Annealing* como busca local, e Cadeia de Kempe como movimento de perturbação. Os resultados são comparados com as melhores soluções obtidas no ITC-2007 e com resultados da literatura.

PALAVRAS CHAVE. Problema de Tabela-Horário em Universidades, ILS, Simulated Annealing.

Tópicos: MH - Meta-heurísticas; OC - Otimização Combinatória.

ABSTRACT

In the academic context is the University Timetabling Problem, characterized as a routine and complex process. The educational timetabling problem consists of allocating a set of classes to available classrooms and predetermined times, taking into consideration students and teachers, and complying with the existing constraints. In this work, the university-based model, which was proposed on the second international timetabling competition (ITC-2007), was chosen. To solve the problem, in this work, the approach used is the metaheuristic Iterated Local Search with Simulated Annealing as local search and Kempe Chain as perturbation phase. The results were compared with the best solutions obtained on ITC-2007 and with results in literature.

KEYWORDS. University Timetabling Problem. ILS. Simulated Annealing.

Paper topics: MH - Metaheuristics; OC - Combinatorial Optimization.

1. Introdução

Problemas de escalonamento (*scheduling*) baseiam-se em alocar recursos, respeitando um conjunto de restrições, de modo a maximizar (ou minimizar) uma função matemática. Essa característica define vários problemas clássicos na área de otimização combinatória, dentre eles os Problemas de Tabela-Horário (PTH).

Considerando a complexidade dos PTH, vários estudos relacionados ao PTH de escolas e universidades já foram realizados [Lewis, 2007], tendo em vista que a elaboração manual destas tabelas-horário não é uma tarefa trivial. Além disso, as instituições de ensino precisam fazê-la frequentemente.

Como descrito por Rocha [2013] e Souza [2000], os PTH da área educacional podem ser especificados em três categorias: PTH de Escolas de Ensino Médio, PTH de Universidades e PTH de Exames Finais. No Problema de Tabela de Horários de Universidades (PTHU), o problema consiste em alocar uma sequência de encontros entre professores e alunos em um período prefixado de tempo, satisfazendo um conjunto de restrições [Souza, 2000].

O PTHU possui uma das mais altas complexidades da área de otimização combinatória e esta aumenta à medida em que são adicionadas novas restrições. Segundo Schaerf [1995] o problema é classificado como NP-Completo para a maioria das formulações, e assim, uma solução exata só pode ser garantida para instâncias pequenas, o que não corresponde à realidade da maioria das instituições de ensino.

Com objetivo de incentivar o estudo de diferentes abordagens sobre o PTH, o PATAT - *The International Series of Conferences on the Practice and Theory of Automated Timetabling* promoveu três competições (2002, 2007 e 2011) de tabela-horário educacional, chamadas de *International Timetabling Competition* - ITC, sendo que a edição de 2007 foi a última a abordar o PTHU.

O ITC realizado em 2007 (ITC-2007) foi dividido em três diferentes formulações e cada formulação possui seu próprio conjunto de instâncias para testes. No desenvolvimento desse trabalho, foi escolhida a terceira formulação estabelecida pelo ITC-2007, conforme descrito em PATAT [2007]. A principal razão dessa escolha é facilitar a comparação com os resultados obtidos por outros algoritmos propostos na literatura e por essa formulação se aproximar da realidade de grande parte das universidades brasileiras.

Nessas instâncias do ITC-2007, e no PTHU de forma geral, para encontrar a melhor solução possível, é desejável alcançar a satisfação de todas as restrições. Entretanto, nem sempre é possível atendê-las totalmente. Assim, as restrições estabelecidas são classificadas de acordo com sua importância. Conforme Santos e Souza [2007], frequentemente são utilizados dois grupos para classificar as restrições:

- Restrições Fortes: Devem ser obedecidas a qualquer custo. O não atendimento de alguma dessas restrições inviabiliza a solução.
- Restrições Fracas: A satisfação dessas restrições é desejável, porém, o não atendimento não inviabiliza a solução.

Conforme descrito em PATAT [2007], na terceira formulação do ITC-2007, as restrições são descritas como segue:

- Restrições Fortes (*RFt*):
  - Aulas: Todas as aulas das disciplinas devem ser alocadas em períodos diferentes. Uma violação ocorre se uma aula não é alocada (RFt_1).
  - Conflitos: Aulas de disciplinas do mesmo currículo ou lecionadas pelo mesmo professor devem ser alocadas em períodos diferentes (RFt_2).
  - Ocupação de Sala: Duas aulas não podem ocupar uma sala no mesmo horário (RFt_3).

  - Disponibilidade: Uma aula não pode ser alocada num horário em que a disciplina é indisponível (RFt_4).

- Restrições Fracas (*RFc*):
  - Dias Mínimos de Trabalho: As aulas de cada disciplina devem ser espalhadas por uma quantidade mínima de dias. Cada dia abaixo do mínimo é contado como uma violação (RFc_1).
  - Aulas Isoladas: Aulas do mesmo currículo devem ser alocadas em períodos adjacentes. Cada aula isolada é contada como uma violação (RFc_2).
  - Capacidade da Sala: O número de alunos da disciplina deve ser menor ou igual ao número de assentos da sala em que a aula for alocada. Cada aluno excedente contabiliza uma violação (RFc_3).
  - Estabilidade de Sala: Todas as aulas de uma disciplina devem ser alocadas na mesma sala. Cada sala distinta é contada como uma violação (RFc_4).

Devido à importância do problema e a dificuldade de se resolvê-lo, existe a necessidade de propor algoritmos com diferentes abordagens, que produzam soluções satisfatórias para o problema em tempo viável, independente do tamanho da instância. A partir disso, o propósito deste trabalho é avaliar o desempenho da meta-heurística híbrida *Iterated Local Search* (ILS) com *Simulated Annealing* (SA) para a resolução do PTHU.

O restante deste trabalho apresenta uma breve revisão da literatura na Seção 2, seguida pela metodologia na Seção 3. Os experimentos computacionais são detalhados na Seção 4 e, por fim, as conclusões são apresentadas na Seção 5.

2. Revisão da Literatura

Segundo Schaerf [1995], o problema de tabela-horário de instituições de ensino foi estudado inicialmente na década de 60, quando Gotlieb [1962] iniciou estudos nesta área com o objetivo de apresentar resoluções por meio de métodos automáticos de geração da tabela horário. Desde então, o tema ganhou relevância [Schaerf, 1995; Lewis, 2007].

Müller [2009] resolve o PTHU do ITC-2007 utilizando *Conflict-based Statistics* para gerar a solução inicial e *Hill Climbing* combinado com *Great Deluge* e *Simulated Annealing* para refinamento da solução.

Rocha [2013] trata especificamente de tabela-horário de universidades adotando a terceira formulação do ITC-2007. O problema é resolvido com a meta-heurística *Greedy Randomized Adaptive Search Procedure* (GRASP), sendo testados os métodos *Hill Climbing* e *Simulated Annealing* como métodos de busca local. O método *Path-Relinking* também é utilizado para intensificar a busca por soluções de boa qualidade.

Segatto et al. [2015] também aplicaram GRASP na terceita formulação do ITC-2007, sendo que Cadeia de Kempe foi usada como um dos movimentos para a geração de vizinhos na busca local.

Kiefer et al. [2016] utilizam a meta-heurística *Adaptive Large Neighborhood Search* para a resolução do PTHU formulado pelo ITC-2007. O método tem como base a destruição de uma solução, seguida da reconstrução de uma parte da solução, para assim gerar novos vizinhos. Com esse método foram obtidos os melhores resultados conhecidos até então em 5 das 21 instâncias da competição.

3. Metodologia

Esta seção apresenta a metodologia abordada para o desenvolvimento deste trabalho. Inicialmente, é detalhada a modelagem do problema, seguida pela descrição da função objetivo utilizada na terceira formulação do ITC-2007. Por fim, são apresentados os detalhes do algoritmo híbrido ILS com *Simulated Annealing* proposto e implementado neste trabalho.

3.1. Modelagem do Problema

A modelagem do problema foi baseada nas instâncias disponibilizadas pelo ITC-2007 [PATAT, 2007]. A descrição das instâncias é apresentada a seguir:

- **Dias, Horários e Períodos:** É dado o número de dias na semana em que existirão aulas e é estabelecido o número fixo de períodos de aulas em cada dia. Um horário é composto por um dia e um período.
- **Disciplinas e Professores:** Cada disciplina possui uma quantidade de aulas semanais que devem ser alocadas em horários diferentes, dados do professor que a leciona e o total de alunos que a assiste. Um número mínimo de dias é determinado para a distribuição das aulas em uma semana, e é permitido que um professor lecione mais de uma disciplina.
- **Salas:** Cada sala possui sua capacidade de assentos.
- **Currículo:** Um currículo é composto por um grupo de disciplinas que possuem alunos em comum.
- **Indisponibilidades:** Períodos que são indisponíveis para determinadas disciplinas.

3.2. Função Objetivo

Cada solução possui um valor associado que representa sua qualidade, esse valor é chamado de função objetivo (f). Cada restrição infringida aumenta o valor da função objetivo de acordo com seu peso. A melhor tabela-horário (solução S) para o problema é aquela que minimiza o valor da função objetivo, dada pela formula:

$$f(S) = RFt + RFc \tag{1}$$

Em que $RFt = |RFt_1| + |RFt_2| + |RFt_3| + |RFt_4|$ e $RFc = 5 \times |RFc_1| + 2 \times |RFc_2| + |RFc_3| + |RFc_4|$. O símbolo $|.|$ representa a quantidade de violações em S de cada restrição. Para uma solução ser considerada válida, a mesma não deve violar nenhuma das restrições fortes, o que significa: $RFt = 0$. Vale ressaltar que a função objetivo f e os pesos das restrições foram definidos com base na formulação utilizada no ITC-2007.

3.3. Iterated Local Search

A meta-heurística *Iterated Local Search* (ILS) foi introduzida por Lourenço et al. [2003] partindo da ideia que um procedimento de busca local pode ser melhorado gerando-se novas soluções iniciais, as quais são obtidas por meio de perturbações na solução ótima local. O Algoritmo 1 apresenta o pseudocódigo genérico da meta-heurística ILS.

Algoritmo 1: Pseudocódigo genérico do ILS.

Entrada: ILS_{max}
Saída: Solução S

```
1 S ← Solução Inicial Qualquer;
2 S ← BuscaLocal(S);
3 para i ← 1 até ILS_max faça
4 |   S_i ←Pertubação(S);
5 |   S_i ←BuscaLocal(S_i);
6 |   S ← Aceitação(S, S_i);
7 fim
```

O ILS é um procedimento que inicia com a construção de uma solução inicial qualquer, que em seguida é submetida a uma busca local, retornando possivelmente uma solução de melhor qualidade, que corresponde a uma solução ótima local. Em seguida, o método inicia a fase iterativa, sendo executados três componentes em cada iteração: perturbação, busca local e aceitação. A seguir, cada elemento do método ILS é descrito com maiores detalhes.

3.3.1. Construção da Solução Inicial

A construção parte de uma tabela-horário (S) inicialmente vazia. Em seguida, são alocadas as aulas, uma a uma, até que ao final, todas estejam alocadas em S. Assim, cada iteração é composta por duas operações distintas: primeiro seleciona-se uma aula, entre as aulas ainda não alocadas, e logo após é escolhido um horário e sala disponíveis em S para alocar a aula selecionada anteriormente. As escolhas são feitas, conforme apresentado por Lü e Hao [2010], levando em consideração as seguintes definições sobre a tabela-horário S:

- $apd_i(S)$: Total de horários disponíveis para uma disciplina i em S.
- $aps_i(S)$: Total de combinações de horário e sala disponíveis para uma disciplina i em S.
- $nl_i(S)$: Total de aulas não alocadas da disciplina i em S.
- $uac_{i,j}(S)$: Total de aulas das disciplinas ainda não alocadas completamente que se tornam indisponíveis para alocação no horário j após a inserção de uma aula da disciplina i no horário j.

Sobre qualquer tabela-horário parcial S, a escolha da aula de uma disciplina, é realizada de acordo com os seguintes critérios:

1. Escolhe-se a disciplina com o menor valor de $\frac{apd_i(S)}{\sqrt{nl_i(S)}}$.
2. Se existir múltiplas disciplinas com o mesmo valor, é escolhida a disciplina com o menor valor de $aps_i(S)\sqrt{nl_i(S)}$.
3. Se ainda houver a ocorrência de mais de uma disciplina com o menor valor, é escolhida a disciplina com maior valor de $conf_i$, sendo $conf_i$ o número de disciplinas que compartilham estudantes ou professores com a disciplina i. Se ainda existir empates isso é resolvido seguindo a ordem alfabética do nome da disciplina.

Na segunda fase da heurística, é realizada a escolha do horário e a sala que a aula selecionada a_i será alocada. É escolhido o horário j e sala k com o menor valor da função $g(j,k)$, dada pela fórmula:

$$g(j,k) = k_1 \times uac_{i,j}(S) + k_2 \times \Delta f_S(i,j,k) \quad (2)$$

Sendo $\Delta f_S(i,j,k)$ o valor da penalidade das restrições fracas sobre a possível alocação da aula a_i no horário j e sala k. Os pesos k_1 e k_2 são relativos às restrições fortes e fracas, respectivamente, e $uac_{i,j}(S)$ representa o número de aulas das disciplinas ainda não alocadas completamente, que se tornam indisponíveis para alocação no horário j após a possível inserção da aula a_i no horário j e sala k.

3.3.2. Busca Local

Neste trabalho, como algoritmo de busca local, é utilizada a meta-heurística *Simulated Annealing* (SA), considerando que a mesma mostrou-se satisfatória nos trabalhos de Rocha [2013] e Segatto et al. [2015].

O SA possui cinco parâmetros principais: a solução inicial do problema (S), a temperatura inicial (T_0), a temperatura final (T_f), a taxa de resfriamento (β) e o número de soluções vizinhas (Nv) que deverão ser geradas a cada iteração. A estratégia faz analogia ao processo de metalurgia, que aquece um metal sólido e realiza o resfriamento lentamente até que o material se solidifique, sendo acompanhado e controlado até que a estrutura se mantenha uniforme [Kirkpatrick, 1984].

O algoritmo parte de uma temperatura inicial (T_0), e de acordo com β, é resfriada até obter a temperatura final (T_f). Em cada temperatura, são gerados Nv vizinhos. Se o vizinho gerado é

melhor que a solução atual, esta é atualizada. Se o vizinho possuir valor de f maior do que a solução atual, ele pode ser aceito com uma probabilidade igual a $e^{-\Delta f/T}$, sendo Δf a diferença entre o valor de f do vizinho e da solução atual, e T a temperatura atual. Quanto maior for o valor de Δf e menor a temperatura, menores serão as chances de uma solução vizinha ser aceita. O comportamento inicial do algoritmo, ou seja, quando a temperatura está alta, é aceitar grande amplitude de soluções. Com o decréscimo da temperatura, uma menor quantidade de soluções piores são aceitas, e como consequência, uma determinada região de busca é intensificada. O Algoritmo 2 apresenta o pseudocódigo do SA para a fase de busca local.

Algoritmo 2: SA apresentado por Rocha [2013]

Entrada: Solução S , T_0, T_f, β, Nv
Saída: Solução S_{Melhor}
$T \leftarrow T_0$; $S_{Atual} \leftarrow S$; $S_{Melhor} \leftarrow S$;
enquanto $T > T_f$ **faça**
  para $i \leftarrow 1$ **até** Nv **faça**
    $S_{Vizinho} \leftarrow GeraVizinho(S_{Atual})$;
    $\Delta f \leftarrow f(S_{Vizinho}) - f(S_{Atual})$;
    se $\Delta f < 0$ **então**
      $S_{Atual} \leftarrow S_{Vizinho}$;
      se $f(S_{Vizinho}) < f(S_{Melhor})$ **então**
        $S_{Melhor} \leftarrow S_{Vizinho}$;
      fim
    fim
    senão
      Gere um número aleatório $p \in (0, 1]$;
      se $p < e^{-\Delta f/T}$ **então**
        $S_{Atual} \leftarrow S_{Vizinho}$;
      fim
    fim
  fim
  $T \leftarrow T * \beta$
fim

Para escapar de ótimos locais, utiliza-se os seguintes movimentos na geração de soluções vizinhas:

- *Move*: Uma aula é realocada para uma posição livre na tabela-horário. A Figura 1 apresenta um exemplo da execução de um movimento do tipo *Move*.

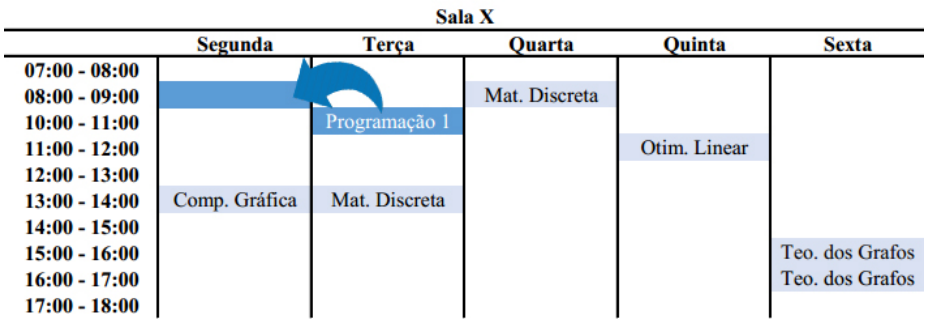

Figura 1: Exemplo de movimento do tipo *Move*.

- *Swap*: Duas aulas trocam de posição na tabela-horário, como demonstrado na Figura 2.

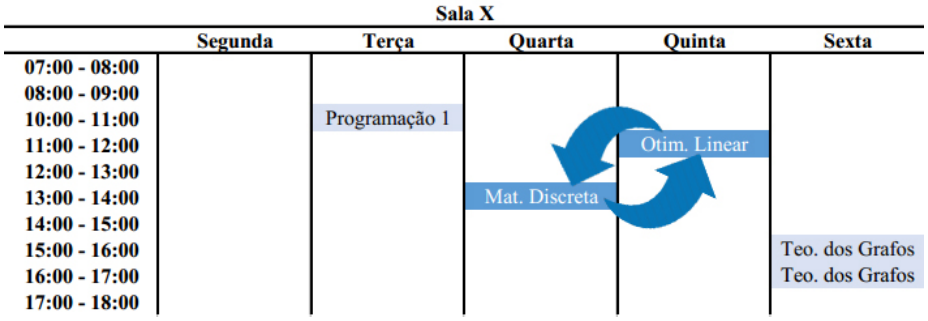

Figura 2: Exemplo de movimento do tipo *Swap*.

- *Time-Move*: Uma aula é movida para outro dia na tabela-horário, sem alterar o horário e sala em que está alocada, desde que o horário esteja disponível. Um exemplo deste movimento é apresentado na Figura 3.

Figura 3: Exemplo de movimento do tipo *Time-Move*.

- *Room-Move*: Uma aula é movida para uma sala disponível na tabela-horário, no mesmo dia e horário em que está alocada, conforme demonstrado na Figura 4.

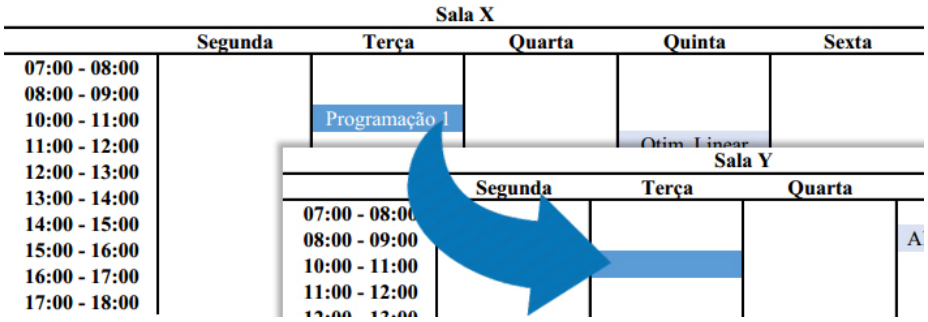

Figura 4: Exemplo de movimento do tipo *Room-Move*.

Na etapa de busca-local, o algoritmo SA é executado primeiramente com os movimentos *Move* e *Swap* e, se após a execução houve melhora na solução, o algoritmo é executado novamente utilizando os movimentos *Time-Move* e *Room-Move*. O ciclo se repete até que não ocorra melhora na solução, conforme apresentado na Figura 5. Além disso, em cada iteração do SA o movimento para geração de uma solução vizinha é escolhido aleatoriamente.

3.3.3. Perturbação

Como estratégia de perturbação, é utilizado o movimento Cadeia de Kempe conforme definido por Segatto et al. [2015]. Essa estratégia foi adotada pois em testes preliminares, mostrou-

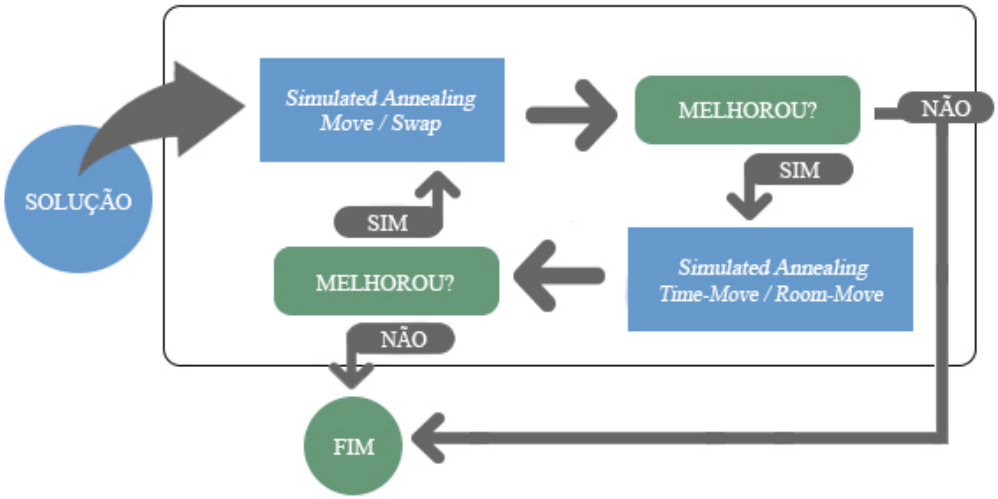

Figura 5: Execução do *Simulated Annealing* para fase de busca local.

se forte o bastante na modificação de uma solução e, ao mesmo tempo, fraca o bastante para que a solução não seja desconstruída totalmente.

No PTHU, a Cadeia de Kempe é executada da seguinte forma: a partir de uma tabela-horário válida, são escolhidos aleatoriamente dois horários, denominados t_1 e t_2. A aula a_{11} está alocada no horário t_1 na sala 1, a aula a_{12} esta alocada no mesmo horário na sala 2, e assim por diante, conforme pode ser visualizado na Figura 6. Em seguida, é gerado um grafo de conflitos, sendo que cada aula representa um vértice. Se duas aulas conflitam por causa de uma restrição forte, são ligadas por uma aresta. Cada subgrafo conexo formado representa uma Cadeia de Kempe. A seguir, é escolhida uma Cadeia de Kempe para que seja realizada a troca entre as aulas pertencentes a essa cadeia.

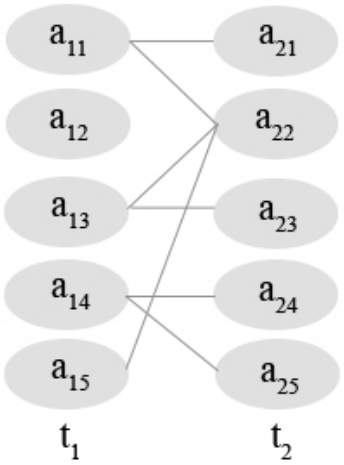

Figura 6: Exemplo de Cadeias de Kempe nos horários t_1 e t_2.
Fonte: Segatto et al. [2015]

No exemplo ilustado na Figura 6, existem três Cadeias de Kempe: a primeira é a cadeia que contém as aulas: a_{11}, a_{21}, a_{22}, a_{13}, a_{23} e a_{15}, a segunda é a cadeia formada pelas aulas a_{14}, a_{24} e a_{25}, e a terceira é a cadeia formada apenas pela aula a_{12}. A tabela horário obtida após a realização das trocas continuará obrigatoriamente sendo válida. Essa garantia ocorre pois todas as

aulas dos horários t_1 e t_2 que conflitam através de uma restrição forte serão trocadas de horário. Um movimento da Cadeia de Kempe válido seria trocar as aulas da primeira Cadeia de Kempe que pertencem ao horário t_1 com as aulas que pertencem a mesma cadeia no horário t_2, obtendo uma nova alocação das aulas nesses dois horários t_1 e t_2, como demonstrado na Figura 7.

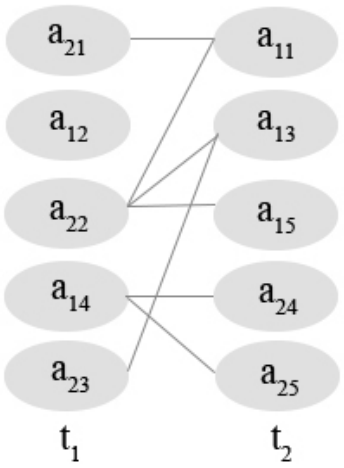

Figura 7: Exemplo de um movimento de Cadeia de Kempe nos horários t_1 e t_2.
Fonte: Segatto et al. [2015]

Na fase de perturbação, é executado um movimento de Cadeia de Kempe qualquer. Esse movimento é importante para diversificar a busca, possibilitando a exploração de novas regiões no espaço de soluções.

3.3.4. Critério de Aceitação

No algoritmo implementado neste trabalho, dada a solução atual S, uma solução intermediária S_i é aceita se $f(S_i) < f(S)$. Nesse caso, S é descartada e as próximas fases do algoritmo ILS são executadas com S_i. Caso contrário, S_i é descartada e a exploração continua com S.

4. Resultados Computacionais

Nesta seção são apresentados os resultados gerados a partir da realização dos experimentos computacionais. Além disso, aponta-se alguns detalhes de implementação, a escolha dos parâmetros do ILS e os parâmetros específicos do SA (busca local). Os resultados computacionais obtidos neste trabalho são comparados com os trabalhos de Segatto et al. [2015] e Kiefer et al. [2016], além dos resultados oficiais dos cinco primeiros colocados no ITC-2007 [PATAT, 2007].

4.1. Detalhes de Implementação

O algoritmo proposto neste trabalho foi implementado utilizando a linguagem de programação C++, compilado com o GCC versão 4.8.4 64 bits e executado em um computador com processador Intel I5-3570 3.4 GHz, 8 Gb de memória RAM e sistema operacional Ubuntu 14.04 LTS 64bits. O código fonte implementado está disponível em: `https://github.com/renancmonteiro/ils_simulated_annealing_timetabling`.

4.2. Escolha dos parâmetros

A meta-heurística ILS possui apenas um parâmetro: o número máximo de iterações (ILS_{max}). Neste trabalho, o número máximo de iterações foi substituído por unidade de tempo. Para obter o tempo equivalente a execução do algoritmo em uma máquina utilizada no ITC-2007 [PATAT, 2007], os organizadores do campeonato ofereceram uma ferramenta executável para realizar o *benchmark* na máquina de testes dos competidores. Esta ferramenta está disponível para *download* em `http://www.cs.qub.ac.uk/itc2007`. Utilizando essa ferramenta na máquina em que foram realizados os testes, foi estipulado o tempo de execução de 192 segundos. Os parâmetros de execução do algoritmo SA foram baseados no trabalho de Rocha [2013]. A Tabela 1 apresenta todos os parâmetros utilizados neste trabalho.

Tabela 1: Valores dos parâmetros do método proposto.

Parâmetro	Descrição	Valor
ILS_{max}	Tempo Máximo de Execução (ILS)	192 segundos
T_0	Temperatura Inicial (SA)	1,5
T_f	Temperatura Final (SA)	0,005
β	Taxa de Resfriamento (SA)	0,999
Nv	Número de Iterações (SA)	500

4.3. Análise dos Resultados

Assim como feito no ITC-2007 [PATAT, 2007], o algoritmo foi executado 10 vezes com diferentes *seeds* para a geração de números aleatórios. Os valores representam apenas as violações das restrições fracas, dado que ao final da execução todas as soluções obtidas são viáveis. Na Tabela 2 são apresentados os resultados obtidos pelo algoritmo ILS+SA. As quatro últimas colunas exibem o melhor valor encontrado, a média, o desvio padrão e o coeficiente de variação (CV), respectivamente, das 10 execuções para cada instância.

Tabela 2: Resultados do algoritmo ILS+SA para as instâncias do ITC-2007.

Instância	ILS+SA			
	Melhor	Média	Desvio Padrão	CV (%)
comp01	5	5,00	0,00	0,00
comp02	68	89,35	10,03	11,23
comp03	97	117,00	11,56	9,88
comp04	44	52,75	3,54	6,71
comp05	430	536,75	46,78	8,72
comp06	68	89,70	8,60	9,59
comp07	43	52,60	6,04	11,48
comp08	55	59,45	3,09	5,20
comp09	114	125,55	6,66	5,30
comp10	44	57,20	8,17	14,28
comp11	0	0,00	0,00	0,00
comp12	428	480,00	28,36	5,91
comp13	83	89,20	4,46	5,00
comp14	66	75,40	3,69	4,89
comp15	90	116,55	11,83	10,15
comp16	59	70,05	7,47	10,66
comp17	102	113,45	7,08	6,24
comp18	93	98,10	3,28	3,34
comp19	94	111,45	10,88	9,76
comp20	73	92,85	11,64	12,54
comp21	136	115,30	9,14	7,93

Conforme pode se observar na Tabela 2, o comportamento do algoritmo mostrou ser estável, uma vez que o maior coeficiente de variação foi de 14,28% para a instância *comp10*. Além disso, em duas instâncias (*comp01* e *comp11*) o resultado obtido pelo algoritmo ILS+SA não mostrou nenhuma variação do valor obtido nas 10 execuções.

Na Tabela 3, podem ser vistos os resultados obtidos pelos cinco melhores colocados no ITC-2007. As colunas estão ordenadas de acordo com o resultado geral da competição. Foram adicionadas três colunas ao final da tabela: uma com os resultados obtidos por Segatto et al. [2015],

outra com os resultados obtidos por Kiefer et al. [2016] e por fim uma com os resultados encontrados pelo algoritmo implementado neste trabalho. A melhor resposta para cada instância está destacada em negrito.

Tabela 3: Resultados do ITC-2007 e de outros trabalhos comparados com os resultados do ILS+SA.

Instância	Muller	Lu	Atzuna	Clark	Geiger	Segatto	Kiefer	ILS+SA
comp01	**5**	**5**	**5**	10	**5**	**5**	**5**	**5**
comp02	43	**34**	55	83	108	117	**34**	68
comp03	72	70	91	106	115	129	**68**	97
comp04	**35**	38	38	59	67	65	**35**	44
comp05	298	298	325	362	408	389	**294**	430
comp06	**41**	47	69	113	94	123	**41**	68
comp07	14	19	45	95	56	77	**10**	43
comp08	**39**	43	42	73	75	81	**39**	55
comp09	103	102	109	130	153	145	**100**	114
comp10	9	16	32	67	66	61	**7**	44
comp11	**0**	**0**	**0**	1	**0**	**0**	**0**	**0**
comp12	331	320	344	383	430	460	**306**	428
comp13	66	65	75	105	101	118	**59**	83
comp14	53	52	61	82	88	99	**51**	66
comp15	84	71	82	119	128	127	**66**	90
comp16	34	39	40	84	81	99	**26**	59
comp17	83	91	102	152	124	139	**67**	102
comp18	83	69	68	110	116	99	**64**	93
comp19	62	65	75	111	107	99	**59**	94
comp20	27	47	61	144	88	125	**19**	73
comp21	103	106	123	169	174	149	**93**	136
Média	75,48	76,05	87,71	121,81	123,05	128,86	**68,71**	104,38

Considerando a média de soluções alcançadas no ITC-2007, o ILS+SA implementado neste trabalho obteve a média de resultados 38% inferior ao primeiro colocado da competição e 14% superior ao quarto colocado. O ILS+SA obteve um resultado 19% superior quando comparado à média dos resultados obtidos por Segatto et al. [2015]. Quando comparados com os melhores resultados conhecidos para estas instâncias, obtidos por Kiefer et al. [2016], o desempenho do ILS+SA foi em média 52% inferior.

5. Conclusões

Este trabalho tratou o Problema de Tabela-Horário de Universidades (PTHU) utilizando a terceira formulação do *International Timetabling Competition* [PATAT, 2007]. Para uma comparação fiel, ou seja, mais próxima ao ambiente computacional no qual os algoritmos submetidos no campeonato foram testados, utilizou-se as mesmas instâncias e tempo equivalente a execução em uma máquina utilizada no ITC-2007. Com os resultados obtidos, pode-se observar que a utilização da meta-heurística ILS combinada com *Simulated Annealing* e Cadeia de Kempe para a resolução do PTHU se mostrou eficiente, apresentando soluções de qualidade. Caso o algoritmo tivesse participado do ITC-2007, ocuparia a quarta colocação na classificação final da competição. A maior diferença pode ser observada para a instância *comp05*, em que foi obtido o sétimo lugar entre os valores obtidos pelos 17 competidores, para esta mesma instância.

Quando comparado com os resultados obtidos pelo método GRASP implementado por Segatto et al. [2015], o ILS+SA obteve resultados superiores em 18 das 21 instâncias executadas. Este resultado evidencia que o ILS+SA, através do processo de perturbação (Cadeia de Kempe),

foi mais eficiente na exploração do espaço de soluções. Além disso, como o tempo computacional gasto para realizar a Cadeia de Kempe é maior quando comparado com o tempo para realizar outros movimentos na geração de soluções vizinhas, o ILS+SA foi mais eficiente que o GRASP, pois realiza esse procedimento de perturbação menos vezes que a busca local de Segatto et al. [2015], o que consequentemente permite o ILS+SA realizar mais iterações e assim diversificar o espaço de busca.

Agradecimentos: Os autores agradecem ao Conselho Nacional de Desenvolvimento Científico e Tecnológico - CNPq (processos 454569/2014-9 e 301725/2016-0) e à Fundação de Amparo à Pesquisa e Inovação do Espírito Santo - FAPES (processos 67656021/2014, 67627153/2014 e 73290475/2016) pelo apoio financeiro.

Referências

Gotlieb, C. C. (1962). The construction of class-teacher time-tables. In *IFIP Congress*, p. 73–77.

Kiefer, A., Hartl, R. F., e Schnell, A. (2016). Adaptive large neighborhood search for the curriculum-based course timetabling problem. *Annals of Operations Research*, p. 1–28. ISSN 1572-9338. URL http://dx.doi.org/10.1007/s10479-016-2151-2.

Kirkpatrick, S. (1984). Optimization by simulated annealing: quantitative studies. *Journal of Statistical Physics*, 34(5-6):975–986.

Lewis, R. (2007). A survey of metaheuristic-based techniques for university timetabling problems. *OR Spectrum*, 30(1):167–190.

Lourenço, H. R., Martin, O. C., e Stützle, T. (2003). Iterated local search. In *Handbook of metaheuristics*, p. 320–353. Springer.

Lü, Z. e Hao, J.-K. (2010). Adaptive tabu search for course timetabling. *European Journal of Operational Research*, 200(1):235–244.

Müller, T. (2009). Itc2007 solver description: a hybrid approach. *Annals of Operations Research*, 172(1):429. ISSN 1572-9338. URL http://dx.doi.org/10.1007/s10479-009-0644-y.

PATAT (2007). International timetabling competition. URL: http://www.cs.qub.ac.uk/itc2007.

Rocha, W. S. (2013). Algoritmo grasp para o problema de tabela-horário de universidades. Master's thesis, Universidade Federal do Espírito Santo.

Santos, H. G. e Souza, M. J. F. (2007). Programação de horários em instituições educacionais: formulações e algoritmos. In *Anais do XXXIX SBPO-Simpósio Brasileiro de Pesquisa Operacional*, number 1, p. 2827–2882.

Schaerf, A. (1995). A survey of automated timetabling. *Artificial Intelligence Review*, 13:87–127.

Segatto, E. A., Boeres, M. C. S., Rangel, M. C., e Kampke, E. H. (2015). Um algoritmo grasp com cadeia de kempe aplicado ao problema de tabela-horário para universidades. In *Anais do XLVII SBPO-Simpósio Brasileiro de Pesquisa Operacional*, number 1, p. 2643–2654.

Souza, M. J. F. (2000). Programação de horários em escolas: uma aproximação por metaheurísticas. Master's thesis, Universidade Federal do Rio de Janeiro.