PROGRAMAÇÃO DE HORÁRIOS ESCOLARES ATRAVÉS DE SAT E METAHEURÍSTICAS

George H. G. Fonseca, Rodrigo G. Ribeiro, Flávio V. Cruzeiro Martins

Universidade Federal de Ouro Preto
george@decea.ufop.br, rodrigogribeiro@decea.ufop.br, flavio@decea.ufop.br

Resumo – O Problema da Programação de Horários Escolares (PPHE) é classificado como NP-Difícil e heurísticas para sua solução são alvo de diversas pesquisas na área de computação, matemática e pesquisa operacional. Normalmente, o problema é resolvido através de metaheurísticas como Algoritmos Genéticos, *Simulated Annealing* e GRASP. O presente trabalho propõe uma abordagem alternativa. Pretende-se reduzir do problema ao problema da Satisfazibilidade Proposicional (SAT) e resolvê-lo usando um resolvedor SAT para geração de uma solução inicial. Como não é viável tratar todos os requisitos PPHE através de satisfazibilidade, posteriormente, aplicar-se-á as metaheurísticas Busca Tabu e *Simulated Annealing* para otimização da solução obtida. Por fim serão apresentados os resultados de cada algoritmos e uma comparação com outro trabalho do gênero.

Palavras Chave – Programação de Horários Escolares, Satisfazibilidade Proposicional, Busca Tabu, *Simulated Annealing*.

1 Introdução

O problema da programação de horários escolares (PPHE) foi proposto por [9] e consiste em programar um horário escolar de modo que nenhuma turma ou professor participe de mais de uma aula no mesmo instante de tempo; além disso, o horário deve atender a outras restrições especificadas a priori.

A programação de horários escolares tem sido alvo de diversas pesquisas nas áreas de Teoria da Computação e Pesquisa Operacional. Em [15] são apresentadas algumas das razões para este interesse:

- Dificuldade de Resolução: encontrar um quadro de horários que satisfaça todos os interesses envolvidos é uma tarefa difícil, ademais, frequentemente, a simples construção de um quadro de horários válido já é uma tarefa complicada;
- Importância Prática: a confecção de um bom quadro de horários pode melhorar a satisfação do corpo docente e permitir que a instituição de ensino seja mais eficiente na gestão de seus recursos, além do mais, a programação adequada das atividades letivas permite um melhor desempenho dos alunos;
- Importância Teórica: o problema apresentado neste trabalho é classificado como NP-Completo ou NP-Difícil [8] e, progressos na solução de problemas desse tipo são um dos grandes objetivos das pesquisas correntes em computação, matemática e pesquisa operacional.

Normalmente, o problema é resolvido através de heurísticas, como Busca Tabu [14], *Simulated Annealing*, GRASP [13] e Algoritmos Genéticos [5] [6]. O presente trabalho pretende realizar a redução do problema ao problema da satisfazibilidade de uma fórmula da lógica proposicional (SAT), submeter a fórmula para resolução usando um resolvedor SAT e otimizar a solução encontrada pelo resolvedor SAT através das metaheurísticas Busca Tabu e *Simulated Annealing*. Até recentemente, a redução do problema ao SAT não era utilizada devido à ineficiência dos algoritmos para resolução de problemas desse tipo. Porém, na última década, houve um grande avanço no projeto e implementação de resolvedores SAT, hoje capazes de resolver fórmulas com centenas de milhares de variáveis de maneira eficiente [4].

O restante do trabalho é organizado da seguinte maneira: na Seção 2 será descrito o problema da programação de horários escolares bem como sua definição formal, na Seção 3 será descrito o problema da Satisfazibilidade Proposicional bem como as metaheurísticas Busca Tabu e *Simulated Annealing*. Na Seção 4 será explicitada a solução proposta, a codificação da fórmula da lógica proposicional e a variante das metaheurísticas Busca Tabu e *Simulated Annealing* implementadas. Na Seção 5 serão apresentados os experimentos computacionais e na Seção 6, as considerações finais.

2 Problema da Programação de Horários Escolares

Segundo [15], o Problema de Programação de Horários Escolares (PPHE), também conhecido na literatura como Problema de Programação de Horários Professor × Turma (PPT) é um problema clássico de otimização combinatória sendo provavelmente o problema mais conhecido dos problemas de programação de horários. Nesse problema, cada professor deve lecionar um determinado número de aulas para cada turma em um conjunto de períodos. As alocações devem ser feitas considerando-se a não existência de conflitos, ou seja, cada professor e turma deve se envolver com, no máximo, uma atividade letiva por período. Há ainda várias outras restrições que o horário deve atender.

2.1. Restrições

No PPHE temos duas categorias de restrições: restrições fortes e restrições fracas que serão descritas a seguir.

2.1.1. Restrições Fortes

Restrições desse tipo devem ser satisfeitas a qualquer custo, visto que não é possível a implementação de um quadro de horários que não as satisfaça. As restrições fortes determinam o espaço de busca que será considerado: somente soluções que respeitam todas as restrições desse tipo são consideradas factíveis; se o problema considerado consiste em obter uma solução factível qualquer, então esse denomina-se problema de busca [15]. As restrições fortes do PPHE consideradas são apresentadas a seguir [18]:

a) Não permitir a alocação de mais de um professor em uma mesma classe no mesmo horário;

b) Não permitir a alocação de mais de uma classe a um mesmo professor no mesmo horário;

c) A carga horária de cada classe deve ser atendida;

d) Atender às restrições de horário de professores.

2.1.2. Restrições Fracas

Restrições desse tipo são aquelas cuja satisfação é desejável, mas caso não seja possível respeitá-las, pode-se, ainda assim, implementar o quadro de horários. O atendimento das restrições fracas é a medida de qualidade de um quadro de horários; se o problema considerado consiste em encontrar uma solução que minimize a violação das restrições fracas, tem-se então um problema de otimização [15]. As restrições fracas do PPHE consideradas são apresentadas a seguir [18]:

a) Atender ao número requerido de aulas geminadas de cada classe;

b) Respeitar o limite diário de aulas para cada classe;

c) Evitar "janelas" nos horários dos professores;

d) Minimizar o número de dias que cada professor deverá vir à escola.

2.2. Definição Formal

Tomaremos como base a definição formal do problema de construção de horários escolares apresentada por [10] e descrita a seguir. Dados:

a) Um conjunto de professores P = $\{p_i\}$, $1 \leq i \leq \alpha$;

b) Um conjunto de classes C = $\{c_j\}$, $1 \leq j \leq \beta$;

c) Um conjunto de horários semanais H = h_k, $1 \leq k \leq \sigma$, $\sigma = \lambda . \mu$, onde λ é o número de dias semanais letivos e μ é o número de horários por dia;

d) Uma matriz $\alpha \times \beta$ de requisitos R = $[r_{ij}]$, onde $r_{ij} \geq 0$ representa o número de encontros semanais do professor p_i com a classe c_j;

e) Uma matriz $\alpha \times \sigma$ de restrições de indisponibilidade de professores D = $[d_{ik}]$, onde $d_{ik} = 1$ se o professor p_i está indisponível no horário h_k; caso contrário, $d_{ik} = 0$.

Uma solução do problema de programação de horários escolares é uma matriz $\alpha \times \beta \times \sigma$, F = $[f_{ijk}]$, onde $f_{ijk} = 1$ se o professor p_i leciona para a classe c_j no horário h_k, que atenda as restrições fortes.

3. Revisão da Literatura

3.1. Satisfazibilidade Proposicional

O problema da Satisfazibilidade Proposicional (SAT) consiste em determinar uma atribuição de valores às variáveis de uma fórmula da lógica proposicional de maneira a tornar esta fórmula verdadeira. Usualmente instâncias deste problema são representadas por fórmulas na forma normal conjuntiva, que possui a seguinte definição indutiva:

- F e T são fórmulas na forma normal conjuntiva
- Os literais α e $\neg\alpha$ são fórmulas na forma normal conjuntiva, onde α é uma variável da lógica proposicional.
- Uma disjunção $l_1 \vee l_2 \vee ... \vee l_n$, onde cada l_i é um literal, é uma fórmula na forma normal conjuntiva. Dá-se o nome de cláusula a uma disjunção de literais.

- Se α e σ são duas fórmulas na forma normal conjuntiva, então $\alpha \wedge \sigma$ é uma fórmula na forma normal conjuntiva. Uma fórmula na forma normal conjuntiva consiste de uma conjunção de cláusulas.

3.2. Busca Tabu

Conforme definição de [16], a Busca Tabu é um método de busca local que consiste em explorar o espaço de soluções movendo-se de uma solução para outra que seja seu melhor vizinho. Esta estratégia, juntamente com uma estrutura de memória para armazenar as soluções geradas (ou características destas) permite que a busca não fique presa em um ótimo local. A implementação desenvolvida da Busca Tabu será descrita na Seção 4.4.

3.3. *Simulated Annealing*

Segundo [16], *Simulated Annealing* trata de uma busca local probabilística que se fundamenta em uma analogia à termodinâmica ao simular o resfriamento de um conjunto de átomos aquecidos.

O procedimento principal consiste de um loop que gera aleatoriamente, em cada iteração, um único vizinho s' de uma solução s. A temperatura T assume, inicialmente, um valor elevado. Após um número fixo de iterações, a temperatura é gradativamente diminuída. Com esse procedimento, dá-se, no início uma chance maior para escapar de mínimos locais e, à medida que T aproxima-se de zero, o algoritmo comporta-se como o método de descida, uma vez que diminui a probabilidade de se aceitar movimentos de piora [16]. A implementação desenvolvida do *Simulated Annealing* será descrita na Seção 4.5.

4. Solução Proposta

Uma das principais noções da teoria da computabilidade é a redução de problemas. Dizemos que um problema p_1 é redutível a um problema p_2 se existe um algoritmo M que transforma instâncias do problema p_1 em instâncias do problema p_2 de maneira que o resultado de p_2 é igual ou complementar ao resultado de p_1 para a instância em questão [11]. No presente trabalho realizar-se-á a redução do problema de programar um horário escolar ao problema SAT através das codificações descritas nas Seções 4.2 e 4.3.

Tratar as restrições fracas *c* e *d* descritas na Seção 2.1.2 através da redução ao problema SAT leva a um número muito grande de cláusulas, inviabilizando o uso da abordagem para tal; então, propõe-se utilizar o resolvedor SAT para gerar uma solução inicial e aplicar as metaheurísticas Busca Tabu e *Simulated Annealing* para otimizar a solução obtida pelo mesmo.

4.1. Representação de Solução

No presente trabalho, uma aula $x_{d,h,e,p,c}$, é composta de:

- Um dia d;
- Um horário h;
- Um encontro e entre a classe c e o professor p;
- Um professor $p \in$ P;
- Uma classe $c \in$ C.

Assim, uma solução pode intuitivamente ser representada por uma lista de aulas.

4.2. Codificação das Restrições Fortes

A codificação de cada restrição descrita na Seção 2.1.1 é dada a seguir.

4.2.1. Não Permitir a Alocação de um Professor a Mais de uma Classe num dado Horário

Para modelar essa restrição, adiciona-se predicados do tipo se o professor p é alocado ao horário h do dia d, então nenhuma das outras aulas ministradas pelo professor p pode ser alocada no horário h do dia d.

$$x_{d,h,e,p,c} \rightarrow \neg x'_{d,h,e,p,c} \mid p = p' \quad (1)$$

4.2.2. Não Permitir a Alocação de Classes Pertencentes ao Mesmo Curso no Mesmo Horário

Para atender a esse requisito, deve-se adicionar predicados do tipo se a classe c for alocada ao horário h do dia d, então nenhuma das outras classes pertencentes ao mesmo curso pode ser alocada no horário h do dia d. Formalmente, temos:

$$x_{d,h,e,p,c} \rightarrow \neg x'_{d,h,e,p,c} \mid c \text{ e } c' \text{ pertencem ao mesmo curso} \quad (2)$$

4.2.3. A Carga Horária de Cada Classe deve ser Atendida

Essa restrição é codificada adicionando-se para cada encontro, de cada classe, uma disjunção de todos os horários possíveis, de modo a garantir que ao menos um deles deva ser assinado com T. A codificação abaixo ilustra a situação:

$$x_{1,1,e,p,c} \lor x_{1,2,e,p,c} \lor \dots \lor x_{d,h,e,p,c} \tag{3}$$

4.2.4. Atender às Restrições de Horário de Professores

Eventualmente, professores têm outros compromissos, como cursos de aperfeiçoamento ou exercício da docência em outra instituição para complementar a renda, sendo impossível que compareçam à escola em determinados horários. Essas restrições são codificadas adicionando-se a negação dos literais relacionados ao professor com os horários de indisponibilidade.

$$\neg x_{d,h,e,p,c} \mid p \text{ está indisponível no horário } h \text{ do dia } d \tag{4}$$

4.3. Codificação das Restrições Fracas

A codificação das restrições *a* e *b* descritas na Seção 2.1.2. é dada a seguir.

4.3.1. Atender ao Número Requerido de Aulas Geminadas de Cada Classe

Muitas classes requerem um determinado número de aulas geminadas para possibilitar um aprendizado melhor do conteúdo lecionado. Para atender ao número requerido de aulas geminadas de cada classe, adiciona-se à fórmula cláusulas do tipo se um encontro da classe foi alocado no horário *h*, então, outro encontro da classe será alocado no horário $h + 1$. O inverso também é válido, conforme a codificação abaixo:

$$\neg x_{d,h,e*2+1,p,c} \rightarrow \neg x_{d,h+1,\ e*2+2,p,c} \land x_{d,h,e*2+1,p,c} \rightarrow x_{d,h+1,\ e*2+2,p,c} \tag{5}$$

4.3.2. Respeitar o Limite Diário de Aulas

Para algumas classes é importante não concentrar muitas aulas num dado dia uma vez que seu conteúdo é cansativo ou exige muito esforço dos estudantes, comprometendo o aprendizado. Para respeitar o limite diário de aulas de cada classe, dado que o limite diário de aulas da classe *c* é *l*, adiciona-se cláusulas do tipo se *l* encontros da classe *c* foram alocados no dia *d*, então o encontro $l + 1$ da classe c não pode ser alocado no dia *d*.

$$x_{d,h1,e1,p,c} \lor x_{d,h2,e2,p,c} \lor \dots \lor \neg x_{d,hn,em,p,c} \tag{6}$$

4.4. Implmentação da Busca Tabu

Ao utilizar lógica proposicional para gerar a solução inicial, garante-se que caso haja uma solução que atenda a todos os requisitos tratados com custo 0, ela será encontrada. A partir dessa solução inicial gerada pelo resolvedor SAT, a metaheurística Busca Tabu pretende explorar a vizinhança da solução de modo a encontrar vizinhos de custo menor no que tange as restrições fracas *a*, *c* e *d*. A restrição fraca *b*, assim como em [12], é tratada como uma restrição forte. É importante também que os outros requisitos já atendidos continuem sendo atendidos. Considerar-se-á vizinho de uma solução *s* uma solução *s'* obtida pela troca do horário entre duas aulas pertencentes ao mesmo curso (ou grade horária). A variação da Busca Tabu desenvolvida é apresentada no pseudocódigo da Figura 1.

No algoritmo da Figura 1, *s* representa a solução encontrada pelo resolvedor SAT, $f(.)$, a função objetivo que será descrita detalhadamente na Seção a seguir, $BT_{Max} = 200$, o limite de iterações sem melhora aceitado pela Busca Tabu, $|T| = 5$, o comprimento da lista tabu e $A(.)$, a função de aspiração. $A(.)$ aceita um movimento tabu apenas se ele conduzir a um vizinho melhor que s^* ($A(f(s)) = f(s^*)$). Todos os parâmetros foram ajustados empiricamente.

4.4. Implementação do *Simulated Annealing*

Similarmente, o *Simulated Annealing* foi implementado como uma técnica alternativa para explorar a vizinhança da solução gerada pelo resolvedor SAT. O mesmo conceito de vizinhança do utilizado na Busca Tabu é seguido. A Figura 2 apresenta o pseudocódigo do *Simulated Annealing* desenvolvido.

No algoritmo da Figura 2, assim como no da Figura 1, *s* representa a solução encontrada pelo resolvedor SAT e $f(.)$, a função objetivo que será descrita detalhadamente a seguir. $SA_{Max} = 100$ representa o de iterações que o algoritmo deve processar em cada temperatura, , $T_0 = 20$, a temperatura inicial e $\alpha = 0{,}5$, a taxa de resfriamento. Mais uma vez, todos os parâmetros foram ajustados empiricamente.

```
procedimento BT(s, f(.), BTmax, |T|, A(.))
    s* ← s;                      {Melhor solução obtida até então}
    Iter ← 0;                    {Contador do número de iterações}
    MelhorIter ← 0;              {Iteração mais recente que forneceu s*}
    T ← ∅;      {Lista Tabu}
    enquanto (Iter - MelhorIter ≤ BT_max) faça
        Iter ← Iter + 1;
        Seja s' ← s ⊕ m o melhor elemento de V ⊆ N(s) tal que
        m não infrinja as restrições fortes a, b, c e d e
        m não infrinja a restrição fraca b e
        m não seja tabu (m ∉ T) ou
        s' atenda a condição de aspiração (f(s') < A(f(s)));
        Atualize a lista tabu T;
        s ← s';
        se (f(s) < f(s*)) então
            s* ← s;
            MelhorIter ← Iter;
        fim-se;
        Atualize a função de aspiração A;
    fim-enquanto;
    s ← s*;
    Retorne s;
fim BT;
```

Figura 1 – Implementação da Busca Tabu.

```
procedimento SA(s, f(.), SAmax, T_0, α)
    s* ← s;              {Melhor solução obtida até então}
    IterT ← 0;           {Iterações na temperatura T}
    T ← T_0;             {Temperatura Corrente}
    enquanto (T > 0) faça
        enquanto (IterT < SA_max) faça
            IterT ← IterT + 1;
            Gere um vizinho s' ∈ N(s) tal que
            s' não infrinja as restrições fortes a, b, c e d e
            s' não infrinja a restrição fraca b
            Δ = f(s') – f(s)
            se (Δ < 0) então
                s ← s';
                se (f(s') < f(s*)) então
                    s* ← s';
                fim-se;
            senão
                tome x ∈ [0, 1];
                se (x < e^(−Δ / T)) então
                    s ← s';
                fim-se;
            fim-se;
        fim-enquanto;
        T ← α × T;
        IterT ← 0;
    fim-enquanto;
    s ← s*;
    Retorne s;
fim SA;
```

Figura 2 – Implementação do *Simulated Annealing*.

5. Resultados Experimentais

Para avaliar os resultados obtidos foram consideradas duas variáveis: tempo levado para alcance da solução e qualidade da solução encontrada. A qualidade da solução é definida pela função objetivo a seguir.

5.1. Função Objetivo

A programação de horários escolares é um problema de decisão multicritério porque, para determinar a qualidade de uma programação, faz-se necessário considerar diferentes objetivos [17]. Para avaliar uma programação de horários escolares, os requisitos do problema são classicamente organizados em requisitos essenciais e requisitos não-essenciais, equivalentes, respectivamente a restrições fortes e fracas já definidas anteriormente.

Desse modo, uma programação (ou solução) s pode ser medida com base em duas componentes, uma de inviabilidade ($g(s)$), a qual mede o não atendimento aos requisitos essenciais, e outra de qualidade ($h(s)$), a qual mede o não atendimento aos requisitos considerados não essenciais. A função de custo de uma solução s, a qual deve ser minimizada, pode ser calculada, portanto, pela Equação 7:

$$\boldsymbol{f(s) = g(s) + h(s)} \tag{7}$$

A parcela $g(s)$, que mensura o nível de inviabilidade de uma solução s, é avaliada com base na Equação 8:

$$\boldsymbol{g(s)} = \sum_{\boldsymbol{k=1}}^{\boldsymbol{K}} \propto_{\boldsymbol{k}} \boldsymbol{I_k} \tag{8}$$

onde K é número de medidas de inviabilidade, I_k o valor da k-ésima medida de inviabilidade e α_k o peso associado à essa k-ésima medida.

A parcela $h(s)$, que mensura a qualidade de uma solução s, é avaliada com base na Equação 9:

$$\boldsymbol{h(s)} = \sum_{\boldsymbol{l=1}}^{\boldsymbol{L}} \boldsymbol{\beta_l\, Q_l} \tag{9}$$

onde L representa o número de medidas de qualidade, Q_l o valor da l-ésima medida de qualidade e β_l o peso associado a essa l-ésima medida. Deve ser observado que uma solução s é viável se e somente se $g(s) = 0$. Nas componentes da função $f(s)$ os pesos dados às diversas medidas refletem a importância relativa de cada uma delas [17].

Assim como em [12], assumir-se-á, o peso associado a todas as medidas de inviabilidade como 50 e o peso associado às restrições fracas, *a*, *b*, *c* e *d*, respectivamente 1, 50, 3 e 9. Os critérios para a contagem de violação de cada restrição também foram baseados no trabalho de [12].

5.2. Testes Realizados

Para efeito de teste do protótipo desenvolvido, foram utilizadas instâncias de problemas de construção de horários escolares disponíveis no sítio do laboratório de inteligência computacional da Universidade Federal Fluminense (http://labic.ic.uff.br/).

Todos os testes foram realizados em um computador AMD Athlon(tm) Dual Core 2.2 Ghz com 1GB de Memória RAM sobre sistema operacional Windows XP Professional. A linguagem de programação utilizada para desenvolvimento do software foi C#. O SAT4J [3], foi o resolvedor SAT escolhido para compor o projeto.

N Tabela 1, são apresentados os resultados obtidos pela metaheurística Busca Tabu (BT). Foi considerado a melhor solução encontrada dentre cinco execuções do algoritmo. Omitiu-se o tempo demandado pelo resolvedor SAT por ser insignificante.

Tabela 1 – Resultados da Obtidos pela Busca Tabu. ($h(s)$ é a solução obtida por BT. +SA é o resultado obtido aplicando-se o SA à melhor solução obtida pela BT.)

Instância	**Tempo (s)**	***g(s)***	***h(s)***	**+SA**
1	24	0	203	203
2	60	0	362	365
3	105	0	487	487
4	323	0	698	698
5	466	0	848	844
6	790	0	845	840
7	1154	0	1166	1127

A Tabela 2 apresenta os resultados obtidos pela metaheurística *Simulated Annealing*. Mais uma vez o tempo demandado pelo resolvedor SAT foi omitido.

Tabela 2 – Resultados da Obtidos pelo *Simulated Annealing*. (h(s) é a solução obtida por SA. +BT é o resultado obtido aplicando-se a BT à melhor solução obtida pelo SA.)

Instância	Tempo (s)	*g*(*s*)	*h*(*s*)	+BT
1	15	0	209	209
2	50	0	361	361
3	62	0	475	475
4	134	0	690	690
5	320	0	844	844
6	326	0	827	827
7	610	0	1140	1140

Pode-se observar que, para ambos os algoritmos, em todas as instâncias, as soluções obtidas têm custo 0 de inviabilidade, ou seja, satisfazem todos os requisitos essenciais (restrições fortes). As soluções apresentaram também um custo baixo de qualidade, proporcional à complexidade de cada instância a um tempo de processamento aceitável, que cresce proporcionalmente em função da complexidade das instâncias.

O *Simulated Annealing* foi mais rápido que a Busca Tabu em todas as instâncias e os resultados obtidos foram no geral melhores que os da Busca Tabu. Apenas na instância 1 a Busca Tabu teve um desempenho superior. Conforme a Tabela 1, A aplicação do *Simulated Annealing* ao resultado da Busca Tabu pode reduzir o custo da solução em algumas unidades; entretanto, segundo a Tabela 2, aplicar a Busca Tabu ao resultado do *Simulated Annealing* introduz um aumento no custo computacional sem melhora das soluções.

5.3. Trabalhos Relacionados

Em [7] utilizou-se resolvedores SAT para a solução de um modelo do problema de programação de horários escolares semelhante ao aqui apresentado, entretanto alguns requisitos importantes não foram tratados, como minimização do número de dias que cada professor deve vir à escola e minimização de janelas no horário dos professores. Do ponto de vista computacional os resultados obtidos não foram muito animadores: não raro, o tempo de processamento passava de 2 horas para instâncias não muito grandes.

Outro relato relevante da aplicação de Satisfizibilidade Proposicional na programação de horários escolares pode ser encontrado em [1], onde, reduzindo o PPHE ao problema Weighted Partial Max SAT [2] obteve-se um resultado expressivo: a melhor solução conhecida para 10 das 32 instâncias abordadas.

Em [12] é apresentada uma abordagem de Programação Linear Inteira Mista para resolver o problema apresentado. Na referida abordagem obteve-se resultados significativos, principalmente para as instâncias com um número reduzido de classes. Para três das instâncias abordadas foram encontradas as soluções ótimas. A tabela abaixo compara os resultados obtidos aos de [12]. A coluna Diferença apresenta o percentual de diferença entre a melhor solução dentre a obtida pela Busca Tabu e pelo *Simulated Annealing* (em negrito) em relação à obtida por [12]:

Tabela 3 – Comparação de Resultados com [12].

Instância	Resultado Obtido (*f*(*s*))			
	[12]	**Busca Tabu**	***Simulated Annealing***	**Diferença (%)**
1	202	**203**	209	0,5
2	333	362	**361**	8,4
3	423	487	**475**	12,3
4	653	698	**690**	5,7
5	766	848	**844**	10,2
6	760	845	**827**	8,8
7	1029	1166	**1140**	10,8

O trabalho de [12] foi implementado em C++ utilizando o pacote CPLEX 10 sobre um computador Dell Optiplex G620 com processador Pentium D 3.0Ghz e 2GB de memória RAM, rodando o sistema operacional Linux.

Apesar das soluções obtidas por [12] possuírem um custo menor que as observadas nesse trabalho, um aspecto relevante é o fato de o trabalho de [12] utilizar-se do CPLEX, uma ferramenta proprietária de otimização em detrimento do presente trabalho que se utiliza apenas de componentes gratuitos de software. Uma outra diferença está na capacidade das heurísticas apresentadas juntamente com o SAT serem capazes de resolver instâncias de grande porte, pois seu tempo computacional não cresce exponencialmente como o apresentado por [12].

6. Considerações Finais

Neste trabalho descreveu-se como resolver o problema da programação de horários escolares utilizando uma abordagem híbrida de SAT e metaheurísticas Busca Tabu e *Simulated Annealing* ainda não explorada na literatura. Além de apresentar um benchmarking entre as metaheurísticas.

A principal contribuição deste trabalho para a comunidade científica é a introdução da Satisfazibilidade Proposicional em detrimento de abordagens gulosas para a geração de soluções iniciais [14] [17] [18]. O uso de Satisfazibilidade Proposicional para geração de soluções iniciais se mostrou muito eficaz, uma vez que leva apenas uma fração de segundo para obter as soluções e as soluções geradas atendem a praticamente todos os requisitos do problema aqui apresentado.

Entretanto, para que as soluções obtidas sejam interessantes para a aplicação em situações práticas de programação de horários escolares, é indispensável maximizar o atendimento das restrições fracas *c* e *d*. Para tal, foram aplicadas as metaheurísticas Busca Tabu e *Simulated Annealing* que, apesar de não superarem resultados conhecidos da literatura, se aproximaram consideravelmente dos mesmos. O tempo demandado para obtenção das soluções foi aceitável, o que indica que a utilização de resolvedores SAT para geração de soluções iniciais em conjunto com heurísticas de otimização para refinamento das soluções é viável para resolver o problema da programação de horários escolares.

Como possível trabalho futuro, pretende-se desenvolver uma codificação mais poderosa do problema usando lógica proposicional, que seja capaz de modelar todas as restrições do Problema da Programação de Horários Escolares. Pretende-se adicionalmente, submeter a mesma abordagem a outros problemas variantes do PPHE, além de outros problemas de agendamento em geral.

7 Referências

[1] Achá, R. A. e Nieuwenhuis, R., Curriculum-based Course Timetabling with SAT and MaxSAT. In: **Practice and Theory of Automated Timetabling** (PATAT), 2010, Belfast. Proceedings of the 8th International Conference on the Practice and Theory of Automated Timetabling. Belfast, UK, 2010. v. 1. p. 42-57.

[2] Anstotegui, C., Bonet, M. L. e Levy, J., A New Algorithm for Weighted Partial MaxSAT. In: **Proceedings of the Twenty-Fourth AAAI Conference on Artificial Intelligence**, 2010, Atlanta, USA.

[3] Berre, D. L., Parrain, A., The Sat4j library, release 2.2. In: **Journal on Satisfiability, Boolean Modeling and Computation**, 2010, p. 59-64.

[4] Biere, A., Heule, M., Maaren, H. V. e Walsh, T., A history of Satisfiability. In: **Handbook of Satisfiability**, 2009. IOS Press, Amsterdan, Nethelands. p. 3-55.

[5] Burke, E. K. e Petrovik, S., Recent directions in automated timetabling. **European Journal of Operational Research**, Volume 140, Issue 2. 2002.

[6] Colorni, A., Dorigo, M., Maniezzo, V., Metaheuristics for high school timetabling. **Computational Optimization and Applications**, Volume 3, Issue 3, 1998.

[7] Fat K. C. A., School Timetabling using Satisfiability Solvers. 2004. **Dissertação de Mestrado**. Delft University of Technology, Delft, Nethelands.

[8] Garey, M. R. e Jonhson, D. S., Computers and Intractability: A Guide to the Theory of NP-Completeness. **Freeman**, San Francisco, CA, USA, 1979.

[9] Gotlieb, C. C., The construction of class-teacher time-tables. **Proc. IFIP Congress**, Munich, North Holland Pub. Co., Amsterdam, 1963, pp. 73-77.

[10] Neufeld, G. A. e Tartar, J., Graph coloring conditions for the existence of solutions to the timetabling problem. **Communications of the ACM**, Volume 17, Issue 8, 1974.

[11] Papadimitriou, C. H., Computational Complexity, 1994. **Addison-Wesley Publishing Company**, San Diego, USA.

[12] Santos, H. G., Formulações e Algoritmos para o Problema da Programação de Horários em Escolas. 2006. **Tese (Doutorado em Computação)**. Universidade Federal Fluminense, Niterói, Rio de Janeiro.

[13] Santos, H. G., Ochi, L. S., Drummond, L. M. A. e Souza, M. J. F., GRASP with Data Mining for The School Timetabling Problem. In: **Two Days on Combinatorial Optimization: A Bridge Between Rio and Niterói**, 2003, Rio de Janeiro. Two Days on Combinatorial Optimization: A Bridge Between Rio and Niterói, 2003.

[14] Santos, H. G., Ochi, L. S. e Souza, M. J. F., A Tabu search heuristic with efficient diversification strategies for the class/teacher timetabling problem. **ACM Journal of Experimental Algorithmics**, v. 10, p. 2.9, 2005.

[15] Santos, H. G. e Souza M. J. F., Programação de horários em instituições educacionais: formulações e algoritmos. **XXXIX Simpósio Brasileiro de Pesquisa Operacional**, 2007, Fortaleza. Anais do XXXIX SBPO. Rio de Janeiro : SOBRAPO, 2007. v. 1. p. 2827-2882.

[16] Souza, M. J. F., Inteligência Computacional para Otimização. 2008. **(Desenvolvimento de material didático)**.

[17] Souza, M. J. F., Guimarães, I. F. e Costa, F. P., Um algoritmo evolutivo híbrido para o problema de programação de horários em escolas. In: **XXII Encontro Nacional de Engenharia de Produção**, 2002, Curitiba. Anais do XXII ENEGEP. Santa Bárbara D´Oeste : ABEPRO, 2002. v. 1. p. 1-8.

[18] Souza, M. J. F., Maculan, N. e Ochi, L. S., A GRASP-Tabu Search Algorithm for Solving School Timetabling Problems. In: Maurício G. C. Resende; Jorge Pinho de Sousa. (Org.). **METAHEURISTICS: Computer Decision-Making**. Dordrech: Kluwer Academic Publishers, 2004, v. 86, p. 659-672.