



DEFINIÇÃO DE PARÂMETROS NA UTILIZAÇÃO DE METAHEURÍSTICAS PARA A PROGRAMAÇÃO DE HORÁRIOS ESCOLARES

José Carlos dos Santos Ferreira ¹ e Jean Eduardo Glazar ²

¹ Aluno do Curso de Ciência da Computação

² Professor do Curso de Ciência da Computação

Departamento de Ciência da Computação

Faculdade de Aracruz (UNIARACRUZ)

jcarlos@jcsoft.com.br, jean@fsjb.edu.br

RESUMO

O objetivo deste trabalho é tratar do problema de programação de horários em escolas. Para isso, utilizou-se o procedimento GRASP para gerar um horário inicial, priorizando a não ocorrência de conflitos e, em seguida, utilizou-se a metaheurística conhecida como Busca Tabu para melhorar o horário gerado inicialmente. Foram utilizados coeficientes e penalidades para compor a avaliação do horário. Foram realizados testes para definir os parâmetros a serem utilizados nas metaheurísticas, como forma de melhorar ainda mais as soluções geradas.

Palavras-chave: Otimização combinatória. Metaheurísticas. Busca Tabu

ABSTRACT

The goal of this paper is to deal with the problem of programming schedules in schools. The GRASP algorithm was used to generate an initial schedule, prioritizing the occurrence of no conflicts. After that, a metaheuristic known as Search Tabu was used to improve the schedule generated initially. Coefficients and penalties were used to compose the evaluation of the schedule. Tests were carried out to define the parameters to be used in the metaheuristics, as a form to get an extra improvement in the generated solutions.

Keywords: Combinatorial optimization. Metaheuristics. Tabu Search

INTRODUÇÃO

O propósito deste trabalho é encontrar uma solução mais próxima possível da ótima para se resolver o problema de alocação de professores de uma determinada instituição. A solução consiste em promover o encontro entre professores e turmas durante a semana, sem conflitos entre as partes envolvidas, obedecendo a restrições que garantirão um melhor arranjo das aulas, gerando assim um quadro de horário viável e de boa qualidade. Para isso, utilizou-se o procedimento GRASP para gerar um horário inicial, priorizando a não ocorrência de conflitos e, em seguida, utilizou-se a metaheurística conhecida como Busca Tabu para melhorar o horário gerado inicialmente, satisfazendo diversas restrições.

O PROBLEMA DE ALOCAÇÃO DE HORÁRIOS

O problema de alocação de horários há tempos é amplamente discutido e estudado, principalmente para instituições de ensino, onde conciliar, manualmente, disponibilidade entre professores e alunos é tarefa difícil que demanda tempo e nem sempre se consegue resolver o problema sem conflitos de disponibilidades das partes envolvidas.

Também conhecido como *timetabling*, o problema de alocação de horários escolar é assunto abordado por muitos autores que propõem uma variedade de soluções e de restrições. Schaefer (1999) afirma que esse problema consiste na alocação das aulas de uma escola a um conjunto restrito de horários, satisfazendo diversas restrições.

Schaefer (1999) dividiu o problema em busca e otimização. A primeira objetiva encontrar qualquer quadro de horários que satisfaça todas as restrições do problema. Já um problema de otimização é visto como aquele que produz o melhor quadro de horários possível, satisfazendo às restrições mais críticas.

Classificado por Even *et al.* (1976) como problema NP-difícil, sua automação é de elevada complexidade. Comumente ela é abordada por meio de técnicas heurísticas, nas quais se destacam as chamadas metaheurísticas, que vêm sendo aplicadas com relativo sucesso em problemas de programação de horários. As

metaheurísticas que mais se destacam são: Busca Tabu, GRASP, *Simulated Annealing* e Algoritmo Genético.

GRASP

Segundo Feo (1995), o método de Procedimento de Busca Adaptativa Gulosa e Randomizada (GRASP) é um processo iterativo, no qual cada iteração consiste de duas fases: construção e busca local. Na fase de construção, uma solução viável é gerada iterativamente, elemento a elemento, até que a solução esteja completa. Já a fase de busca local consiste na busca de um ótimo local na vizinhança da solução construída. A melhor solução encontrada ao longo das iterações GRASP é retornada como resultado.

Na fase de construção, os elementos candidatos que compõem a solução são ordenados em uma lista, chamada de lista de candidatos, a qual contém todos os candidatos. Essa lista é ordenada por uma função gulosa, que mede o benefício que o mais recente elemento escolhido concede à parte da solução já construída. Um subconjunto, denominado lista restrita de candidatos (LRC), é formado pelos melhores elementos que compõem a lista de candidatos. O tamanho da lista restrita de candidatos é controlado por um parâmetro NLRC compreendido entre 1 e o tamanho da lista, onde para $NLRC = 1$, tem-se um comportamento puramente guloso do algoritmo e para $NLRC = \text{tamanho da lista}$, um comportamento aleatório. O componente probabilístico do método é devido à escolha aleatória de um elemento da lista restrita de candidatos. Esse procedimento permite que diferentes soluções de boa qualidade sejam geradas.

A METAHEURÍSTICA BUSCA TABU

A Busca Tabu ("*Tabu Search*") é um procedimento heurístico proposto por Glover (1989 e 1990), para resolver problemas de otimização combinatória. A idéia básica é evitar que a busca por soluções ótimas termine ao encontrar um mínimo local (LAGUNA, 1994; GLOVER *et al.*, 1993).

Assim, a partir da solução inicial gerada pelo procedimento construtivo, a metaheurística Busca Tabu faz uma busca agressiva no espaço de soluções, procurando por soluções melhores que a corrente e utilizando uma estrutura de memória que armazena os últimos movimentos realizados. A essa estrutura é dado o nome de Lista Tabu, porque os movimentos que se encontram nessa lista são proibidos por um certo número de iterações. A Lista Tabu será útil quando, em algum momento, não for possível melhorar a solução corrente, o que caracteriza um ótimo local. Nesse momento, o algoritmo terá que escolher a solução menos pior, o que poderia ocasionar um retorno à solução anteriormente visitada, gerando, assim, um ciclo. Como os últimos movimentos são proibidos pela lista tabu, o algoritmo deve prosseguir por um outro caminho que ainda não foi visitado, aumentando o seu universo de busca.

Segundo Glazar (2000), a geração da vizinhança $N(x, k)$, em que x é a solução atual e k é o número de iterações em que o movimento permanece tabu, passa a ser um aspecto crucial na BT. Num ponto inicial, a forma da memória pode ser personificada em uma lista T , chamada *lista tabu*, que contém as $|T|$ soluções mais recentemente visitadas, produzindo $N(x, k) = N(x) - T$. Tal técnica de memória, baseada na solução mais recente, irá prevenir ciclos menores ou iguais a $|T|$. Mas essa formulação consome bastante espaço de memória.

DESCRIÇÃO DO PROBLEMA

O problema escolhido para análise é o da programação de horários da Faculdade de Aracruz – ES (UNIARACRUZ). Foram incluídos os departamentos de: Ciências Contábeis, Administração, Secretariado Executivo, Letras, Engenharia Química, Ciência da Computação, Engenharia Mecânica, Arquitetura e Urbanismo e Secretariado Trilíngüe, atendendo, assim, 28 turmas e 62 professores, no turno noturno.

O objetivo é gerar um horário escolar, relacionando professores, turmas e horários, sem conflitos. Em um horário sem conflitos, uma turma não pode ter aulas em um horário com mais de um professor. A divisão de turmas entre dois professores não foi abordada neste trabalho.

De posse da carga horária de cada professor em cada turma e das restrições dos professores (não pode trabalhar em determinados horários), o sistema gerará um horário sem conflitos. Porém, um horário sem conflitos não pode ser considerado um bom horário para uso na prática. Alguns requisitos também deverão ser atendidos.

REQUISITOS

Visando a um melhor aproveitamento no processo de ensino/aprendizagem, vários requisitos são impostos antes de se iniciar a programação de horários dessa instituição:

- não permitir a alocação de mais de um professor em uma mesma turma no mesmo horário (conflito);
- evitar o excesso de aulas geminadas;
- evitar aulas únicas;
- eliminar, sempre que possível, os “buracos” nos horários dos professores;
- minimizar a quantidade de dias que cada professor necessita ir à escola;
- evitar, sempre que possível, aulas geminadas quebradas pelo recreio;
- evitar quebras de aulas, isto é, aulas não consecutivas de um professor para uma turma em um mesmo dia;
- eliminar, sempre que possível, os “buracos” nos horários das turmas.

Foi atribuído um peso para cada um desses requisitos. Logo, a solução é encontrar um horário sem conflito e com o menor peso possível.

REPRESENTAÇÃO DO PROBLEMA

Para a representação do problema, foram utilizadas duas estruturas de matrizes. A Matriz Principal é uma matriz de 62 linhas representando os professores, as trinta colunas representando os seis dias da semana (segunda a sábado) e cinco aulas

diárias (veja exemplo na Tabela 1). O conteúdo das células indica a turma para a qual o professor estará ministrando aula naquele horário. Com essa estrutura, elimina-se a possibilidade de um professor estar ministrando aulas para mais de uma turma (conflitos de professor). Caso um professor não possa ministrar aulas em um determinado horário, aparecerá o símbolo “-” na matriz.

Tabela 1 - Exemplo de matriz relacionando professores, horários e turmas

	SEGUNDA					TERÇA					QUARTA					...
	1	2	3	4	5	1	2	3	4	5	1	2	3	4	5	
P01	T2	T1		T1	T1	T1	T1	T1	T1		T2	T2	T2	T1		
P02	-	-	T0	T0	T0	-	-	T0	T0		-	-				
P03	T2	T2	T1	T1		T2	T1		T2			T2			T1	
P04	T2	T2			T2	T2	T2	T2	T2		-	-	-	-	-	
P05	T0				T1	T0		T1			T0	T0				
P06	-	-	-	-	-	-	-	-	-	-	T1	T1	T1	T1	T1	
P07	T2		T2	T2								T2	T2			
P08	T0	T0	T0	T0	T0	T0		T0	T0		T1	T0	T0	T0	T0	
P09	-	-	-	-	-	-	-	-	-	-	T2	T2	T2		T2	
P10	T2	T2	T2	T2	T2	-	-	-	-	-	T2	T2	T2	T2	T2	
...																

A Matriz de Conflitos é uma matriz auxiliar que é utilizada para manter atualizados os conflitos de turma que ocorrem durante a execução do programa (veja exemplo na Tabela 2). Essa matriz de 28 linhas, representando as turmas, trinta colunas representando os seis dias da semana e cinco aulas diárias, armazena o número de professores que estarão ministrando aulas para a turma naquele dia. Com isso, tem-se a quantidade de conflitos de cada turma (maior que um é conflito), sem necessidade de algoritmo extra que vasculhe a Matriz Principal.

Tabela 2. Exemplo de matriz de conflitos das aulas

	SEGUNDA					TERÇA					QUARTA					...
	1	2	3	4	5	1	2	3	4	5	1	2	3	4	5	
T01	1	1	1	0	0	1	1	1	1	1	1	1	1	1	1	
T02	1	1	1	1	0	1	1	1	1	1	1	1	1	1	1	
T03	1	1	0	0	1	1	1	1	1	1	1	1	1	1	1	
T04	1	1	0	0	0	1	1	1	0	0	1	1	1	1	1	
T05	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	
T06	1	2	1	1	1	1	0	1	1	1	2	1	1	1	1	
T07	2	1	1	1	1	1	1	1	1	1	1	1	1	1	1	
T08	1	1	1	0	0	1	1	1	1	1	1	1	1	1	1	
T09	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	
T10	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	
...																

LISTA TABU

A estrutura de memória utilizada para armazenar os movimentos tabus foi definida como uma lista encadeada dinâmica de tamanho $|T|$, na qual cada elemento é uma estrutura de memória denominada movimento, contendo informações do professor, horário de origem e horário de destino. Como essa é uma lista ordenada e tem como atributo o seu número de elementos, quando do momento da inserção de um novo movimento, e a lista já estiver no seu tamanho máximo, ocorre a eliminação do último elemento.

O MOVIMENTO

O movimento utilizado para gerar a vizinhança do horário é o movimento horizontal, em que ocorre a troca de valores distintos e diferentes de “–”, onde o símbolo “–” indica a indisponibilidade do professor naquele horário. Na Figura 1, tem-se um fragmento de horário com um exemplo de movimento, onde P01 indica o Professor 1 e T20, T12 e T16 são respectivamente as Turmas 20, 12 e 16.

Antes do movimento							Depois do movimento					
1 2 3 4 5							1 2 3 4 5					
P01	T20	T16	–	T12	T16	⇒	P01	T20	T12	–	T16	T16

Figura 1 - Exemplo de um movimento tabu, envolvendo T12 e T16

RESULTADOS COMPUTACIONAIS

Os resultados foram obtidos utilizando uma base de testes onde foram incluídos os departamentos de Ciências Contábeis, Administração, Letras, Engenharia Química, Ciência da Computação, Engenharia Mecânica, Arquitetura e Urbanismo e Secretariado Trilíngüe, atendendo, assim, 28 turmas e 62 professores, no turno noturno.

As restrições de horários dos professores também foi considerada. Essas informações foram obtidas com os coordenadores de cada curso.

Ao longo de todo o trabalho, foram obtidos resultados razoáveis com o programa elaborado. Contudo, percebeu-se que, alterando alguns parâmetros, seria possível melhorar ainda mais os resultados obtidos. Com isso, foram feitos alguns testes alterando parâmetros e verificando os resultados obtidos. Para dar início aos testes, fixaram-se valores para $BTMax = 1000$, $NumHorAleat = 500$ e $NLRC = 4$. À medida que se obtinham melhores valores para os valores fixados, estes iam sendo substituídos. As comparações foram feitas analisando a solução gerada para cada parâmetro. Quanto menor o valor da solução, melhor será o horário gerado.

O primeiro parâmetro testado foi o tamanho da lista tabu ($TamListaTabu$), conforme Figura 2. Observou-se que, para o tamanho 500 e o tamanho 1000, as soluções são muito próximas. Portanto, foi escolhido o tamanho 500, por utilizar menos memória.

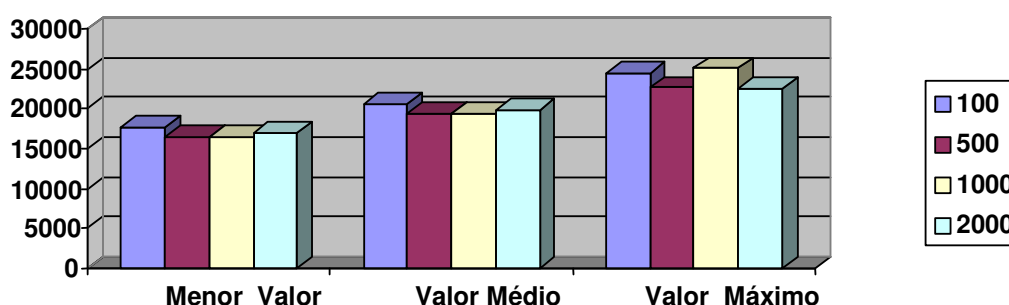


Figura 2 - Comparativos do parâmetro $TamListaTabu$ (Tamanho da Lista Tabu)

A partir dessa escolha, foram feitos testes para a variável $NLRC$ que indica o tamanho da lista restrita utilizada no método GRASP (Figura 3). Observa-se claramente que o valor 3 é melhor nos três cálculos estatísticos.

Fixando o valor 3 para $NLRC$, testou-se a variável $NumHorIni$, que indica o número de horários gerados inicialmente pelo procedimento GRASP (Figura 4). Para essa variável, verificou-se que o menor valor encontrado foi para 100. Apesar de se obter um valor médio com 500, preferiu-se optar pelo valor 100 que demanda 1/5 do tempo que levaria com o valor 500.

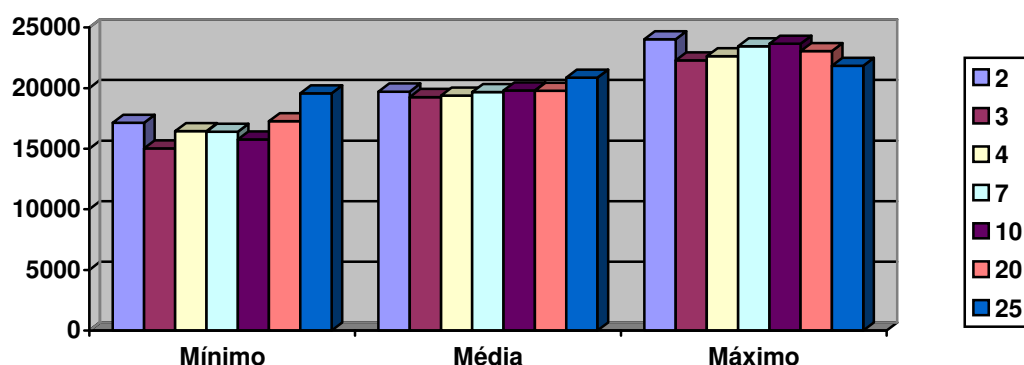


Figura 3 - Comparativos do parâmetro NLRC (Tamanho da Lista Restrita de Candidados)

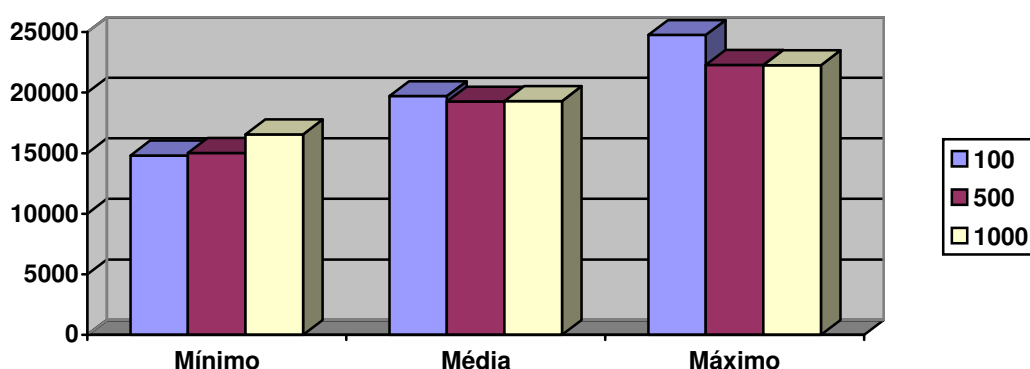


Figura 4 - Comparativos do parâmetro NumHorIni (Número de Horários Gerados Inicialmente)

Por fim, fica claro que, com o tamanho 1000, obteve-se o melhor resultado para a variável BTMax, que indica quantas iterações sem melhora o algoritmo Busca Tabu deve executar antes de parar (Figura 5). Com os testes realizados, apesar de não serem conclusivos, em virtude de se tratar de algoritmos semi-aleatórios, pode-se direcionar em determinado sentido. De qualquer forma, utilizando os parâmetros iniciais, foram obtidos valores próximos a 17.000 e, após os testes e utilização dos parâmetros que obtiveram melhor resultado, os valores chegaram a 14.000.

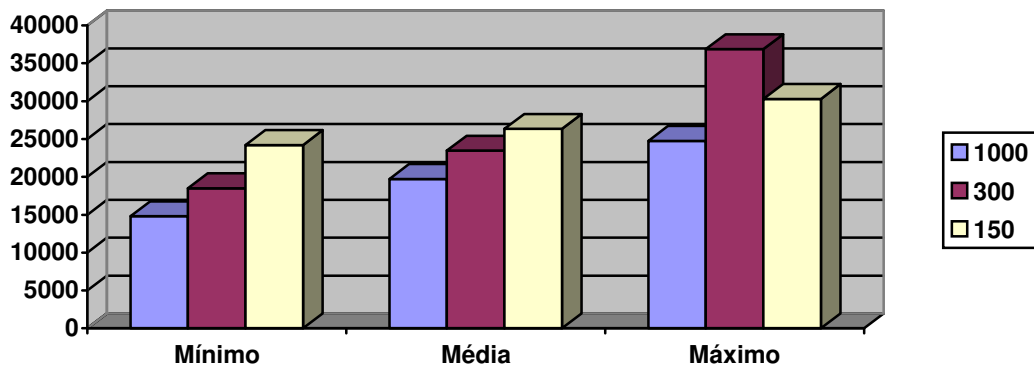


Figura 5 - Comparativos do parâmetro BTMax

CONCLUSÕES

Com este trabalho, conclui-se que é possível, por meio de metaheurísticas, gerar boas soluções para o problema de alocação de horários. Fica claro que é possível melhorar ainda mais o programa desenvolvido, seja pela inclusão de outras penalidades, seja até mesmo utilizando Sistemas Distribuídos para aproveitar uma rede de computadores e obter um resultado melhor em menor tempo.

É possível ainda melhorar o resultado obtido nos procedimentos como os descritos em Souza (2000), nos quais, por meio de ciclos mínimos, consegue-se eliminar conflitos e melhorar a qualidade do horário.

A próxima etapa é realizar testes comparativos entre os horários gerados pelo sistema e os gerados manualmente.

Futuramente, mais restrições presentes no problema real, que não foram abordadas, poderão ser agregadas às heurísticas, como a divisão de uma turma entre dois professores.

REFERÊNCIAS

EVEN, S.; ITAI, A.; SHAMIR, A. On the complexity of timetabling and multicommodity flow problems. **SIAM Journal of Computation**, n. 5, p. 691-703, 1976.

FEO, T. A.; RESENDE, M. G. C. Greedy randomized adaptive search procedures. **Journal of Global Optimization**, v. 6, p. 109-133, 1995.

GLAZAR, J. E. **Análise de diferentes estratégias na aplicação da Busca Tabu ao problema de roteamento de veículos**. 2000. Dissertação (Mestrado em Engenharia de Produção / Pesquisa Operacional) – Universidade Federal do Rio de Janeiro, Rio de Janeiro/RJ, 2000.

GLOVER, F. Tabu Search – Part I. **ORSA J. on Computing**, v. 1, p. 190-206, 1989.

GLOVER, F. Tabu Search – Part II. **ORSA Journal on Computing**, v. 2, p. 4-32, 1990.

GLOVER, F.; TAILLARD, E; WERRA, D. A user's guide to Tabu Search. **Annals of Operations Research**, v. 41, p. 3-28, 1993.

LAGUNA, M. A guide to implementing Tabu Search. **Investigación Operativa**, v. 4, p. 5-25, 1994.

SCHAEFER, A. A survey of automated timetabling. **Artificial Intelligence Review**, v. 13, p. 87-127, 1999.

SOUZA, M. J. F. **Programação de horários em escolas: uma aproximação por metaheurísticas**. 2000. Tese (Doutorado em Engenharia de Produção / Pesquisa Operacional) – Universidade Federal do Rio de Janeiro, Rio de Janeiro/RJ, 2000.