UTILIZAÇÃO DOS MÉTODOS DE OTIMIZAÇÃO EM PROBLEMAS DE TIMETABLING

Ivone Piedade Terra
Professora do Curso de Computação - Sistemas de Informação - Unileste-MG.
Mestre em Ciência da Computação – UFMG.

Joyce Lopes Radaelli
Graduanda do Curso de Computação - Sistemas de Informação – Unileste-MG.

RESUMO

Este artigo aborda o problema de construção de horários de aulas em universidades. O problema de alocação de horários é largamente conhecido por sua complexidade devido a necessidade de conciliar diversos recursos, tais como professores, aulas, salas, entre outros. Sendo assim, busca-se neste trabalho abordar e relatar algumas técnicas, tais como Teoria dos Grafos, Programação Matemática, Sistemas Interativos, Programação Lógica com Restrições, Algoritmos Genéticos ou Evolucionários, utilizadas na literatura para resolução de problemas de alocação de horários (*timetabling*).

Palavras-chave: Timetabling, Otimização, Horário Acadêmico.

ABSTRACT

This article deals with the problem of construction of schedules of lessons in university. The problem of allocation of schedules wide is known by its complexity due the necessity to conciliate diverse resources, such as professors, lessons, rooms, among others. Being thus, one searchs in this work to approach and to tell to some techniques, such as Theory of the Graphs, Mathematical Programming, Interactive Systems, Logical Programming with Restrictions, Genetic or Evolutionary Algorithms, used in literature for resolution of problems of allocation of schedules (timetabling).

Key-words: *Timetabling, Optimize, Academic Schedule*

INTRODUÇÃO

Ao longo dos últimos 40 anos, aproximadamente, a comunidade científica esforça-se para buscar uma solução computacional que auxilie em um antigo, longo e duro processo: a alocação de recursos sob restrições. Um dos problemas típicos desta natureza é o problema de alocação de horários escolares (*timetabling*). O casamento de interesses entre disponibilidades de salas, recursos audiovisuais, professores e alunos ao longo de determinados períodos da semana é tarefa árdua e consumidora de tempo. Contudo, é essencial e periódica em instituições de ensino, seja semestral ou anualmente, já que todas as atividades são pautadas sobre o período letivo.

A construção de um horário de aulas satisfatório tem sido sempre de grande importância na maioria das instituições acadêmicas, e neste sentido, inúmeras soluções individuais são propostas na literatura, nas últimas três décadas.

Os problemas de *timetabling* tiveram as primeiras soluções propostas baseadas na teoria dos grafos, em soluções utilizando técnicas de otimização, e em algoritmos e técnicas usadas em inteligência artificial.

Quanto aos caminhos propostos para a solução, pode-se classificar os principais tratamentos encontrados na literatura nos seguintes conceitos básicos: Teoria dos Grafos, Programação Matemática, Sistemas Interativos, Programação Lógica com Restrições, Algoritmos Genéticos ou Evolucionários, entre outros.

Foi realizado um estudo dos métodos de otimização utilizados no processo de solução dos problemas de confecção de horários acadêmicos. Este estudo dos métodos computacionais de solução de problemas clássicos é valioso, pois é um ótimo caminho para o desenvolvimento de novas técnicas de solução e para um excelente entendimento dos processos e pacotes computacionais disponíveis no mercado.

OS MÉTODOS DE OTIMIZAÇÃO

São descritos a seguir os métodos de otimização utilizados em problemas de timetabling.

Os Algoritmos Genéticos são algoritmos que baseiam-se na Teoria da Evolução através de seleção natural proposta por Charles Darwin no século XIX. Exploram duas idéias: a de que os indivíduos lutam pela sobrevivência tentando adaptar-se ao ambiente onde vivem e somente os indivíduos mais adaptados resistem e a do cruzamento de espécies, acumulando pequenas variações no material genético, gerando uma nova espécie com melhor capacidade de sobrevivência. A idéia fundamental dos Algoritmos Genéticos é adquirir e tratar possíveis soluções de problemas como "indivíduos" de uma "população" que "evoluirá" a cada iteração ou "geração". O funcionamento básico de um Algoritmo Genético pode ser descrito da seguinte forma: inicialmente é gerada uma população, aleatoriamente, com indivíduos que podem ser vistos como possíveis soluções do problema. Durante o processo evolutivo, é calculado o grau de adaptação de cada indivíduo, através de uma função de custo, sendo que os mais adaptados, ou seja, os que possuem baixa penalização, são selecionados enquanto os outros são eliminados através do mecanismo de seleção natural. É gerada uma nova população que pode ter sofrido modificações em suas características fundamentais através de mutações e cruzamentos ou recombinação genética de acordo com as probabilidades definidas

no algoritmo. Após a reprodução, temos uma nova população que deverá ser avaliada, repetindo o processo descrito anteriormente. Como critérios de parada fixamos uma meta (solução ótima) e uma tolerância (uma solução viável). Deve-se também especificar um número máximo de gerações. Após esse processo, não havendo violação das condições descritas no algoritmo, este retorna uma solução viável e não necessariamente ótima.

Em (Costa & Bruna, 2003) um problema foi resolvido utilizando algoritmos genéticos apresentando resultados satisfatórios em relação ao tempo de processamento e execução.

Assim como os Algoritmos Genéticos os Algoritmos Meméticos se baseiam em processos naturais, tais como recombinação, seleção, mutação, entre outros. Algoritmos Meméticos utilizam ainda o conceito de "evolução cultural", onde a adaptabilidade de um indivíduo pode ser modificada no decorrer de sua existência dentro da população. No caso da geração de horários essa evolução cultural poderia ser representada por mudanças em características como ausência de aulas aos sábados ou em mais de um turno. Um indivíduo pode ser geneticamente pouco favorecido ao nascer, mas devido às condições em que vive, por trocas de informação com outros indivíduos, experiências pessoais entre outros aspectos, pode se tornar mais adaptado, e mais do que isso, transmitir essa experiência aos seus descendentes (evolução cultural). As etapas básicas de um Algoritmo Memético podem ser descritas da seguinte maneira: inicialmente gera-se uma população constituída de um número fixo de soluções possíveis dentro do espaço de soluções do problema. Em seguida, avalia-se a população através de uma função de avaliação que mede a qualidade da solução representada por cada agente, ou seja, quanto melhor for a solução representada maior será o valor da sua função de avaliação. A população é modificada através de recombinação, troca de memes (uma unidade análoga ao gene dos Algoritmos Genéticos tradicionais) entre dois agentes, ou de mutação, mudança aleatória no valor dos memes para garantir a diversidade de soluções e que é intensificada nas crises de diversidade quando ocorre uma estagnação da população.

Simulated Annealing é um método de otimização baseado em conceitos da física, utilizados na termodinâmica, sendo tratado inicialmente em Kirkpatrick, Gellat &Vecchi (1983). O nome recozimento (annealing) é dado ao processo de aquecimento de um sólido até seu ponto de fusão, seguido de um resfriamento lento e gradual, até que se alcance novamente seu estado sólido. É uma técnica de busca local, que obtém boas soluções sobre problemas com muitas restrições. O processo se inicia com um membro qualquer do espaço de soluções, normalmente gerado aleatoriamente, e seleciona um de seus vizinhos (indivíduos) randomicamente. Se este vizinho for melhor que o anterior ele é aceito e substitui a solução corrente. Se o novo vizinho for pior, ele pode ser aceito de acordo com uma probabilidade

relacionada a um parâmetro de controle chamado “Temperatura”, que é assim chamado por analogia ao fenômeno físico. A temperatura assume, inicialmente, um valor elevado. Após um número de iterações, a temperatura é gradativamente diminuída por uma razão de resfriamento. Quando a temperatura está elevada, praticamente todos os movimentos são autorizados como no caso da matéria em fusão onde a energia do sistema é muito elevada. Quanto mais a temperatura abaixa, mais as degradações vão sendo penalizadas em função de lucro dos movimentos melhores, fazendo o método convergir para o ótimo local mais próximo. Esse processo é repetido até que a temperatura seja tão pequena que mais nenhum movimento seja aceito. A melhor solução encontrada durante a busca é tomada como uma boa aproximação para a solução ótima.

Johnson (1990) utilizou-se de heurísticas num problema de escalonamento de exames que atingiu 96050 variáveis e 287240 restrições num modelo linear inteiro.

A Busca Tabu é uma estratégia de busca local desenvolvida para encontrar soluções quase ótimas para problemas de otimização. A técnica utiliza uma estrutura de memória para guiar a busca e continuar a exploração do espaço de soluções mesmo que não haja movimento de melhora, evitando a formação de ciclos, isto é, o retorno a um ótimo local previamente visitado. A palavra tabu sugere algo proibido ou pelo menos inibido. A Busca Tabu utiliza restrições tabu para inibir certos movimentos e alguns procedimentos denominados critérios de aspiração são utilizados para decidir quando movimentos classificados como tabu podem ser executados. As restrições tabu são controladas por uma lista tabu, que define todos os movimentos que têm um certo atributo como sendo tabu por um determinado número de iterações, que memoriza os últimos movimentos executados. Tais movimentos são proibidos, a menos que a solução satisfaça a um certo critério de aspiração, que é uma técnica que permite aceitar certos movimentos tabu julgados interessantes para o prosseguimento da pesquisa. Em geral, espera-se que essa solução seja melhor que a melhor solução encontrada até então. A lista tabu usualmente é de tamanho fixo e quando um novo movimento é adicionado à lista, ao mesmo tempo, o mais antigo é removido. Quando um movimento atinge um nível de aspiração, pode ser removido da lista tabu, sendo aceito como solução viável ou não.

Na teoria de grafos, um grafo é uma estrutura formada por vértices e arestas. Pode ser interpretado como uma tripla (N,A,G) onde: N é um conjunto não vazio de nós ou vértices, A é um conjunto de arcos ou arestas e G é a função que associa a cada arco um par não ordenado de nós, denominados terminações de A (Almeida, 2001). O processo de coloração de grafos consiste na atribuição de cores aos vértices que satisfazem as seguintes

propriedades: vértices adjacentes recebem cores diferentes e a menor quantidade possível de cores deve ser utilizada. Em outras palavras, uma coloração de vértices é uma função G: $V \rightarrow C$ que associa a cada vértice $v \in V$ do grafo uma cor e de modo que $G(v) \neq G(w)$ sempre que v é adjacente a $w \in V$. Uma k-coloração de um grafo é uma coloração que utiliza um total de k cores. Denomina-se número cromático de um grafo o menor número de cores k, para o qual existe uma k-coloração. Em analogia com o problema de confecção de horário, cada disciplina é representada por um vértice de um grafo. Um arco conecta dois vértices se as disciplinas correspondentes serão escolhidas pelo mesmo estudante ou serão ministradas pelo mesmo professor. O algoritmo de solução procura por uma coloração dos nodos de forma que quaisquer dois vértices unidos por um arco não possuem a mesma cor. Além disso, o algoritmo minimiza o número de cores utilizadas.

O método "branch-and-bound" é utilizado para otimizar a busca por soluções inteiras para um problema linear, de forma iterativa intercalado com o método Simplex para problemas lineares. O algoritmo Simplex, proposto por George Dantzig (Dantzig, 1963) nos anos 40, constituiu um grande avanço científico-tecnológico e deu grande impulso ao campo da pesquisa operacional que estava dando os primeiros passos. O método percorre os vértices do conjunto compacto de soluções do problema. Quando o problema só admite soluções inteiras, o conjunto é formado por um número não necessariamente finito de pontos do $\Re^n$. Cada dimensão corresponde a uma variável a ser otimizada. O algoritmo Simplex pode ser visto como um processo combinatório que procurar encontrar as colunas da matriz de restrições que induzem uma base e, portanto, uma solução básica ótima. A dificuldade advém do fato que tipicamente existe um número exponencial de possíveis combinações de colunas, gerando portando um desempenho de pior caso de ordem exponencial. Apesar deste aspecto desfavorável, o algoritmo Simplex é eficaz para muitas instâncias e continua sendo o algoritmo mais rápido, mesmo quando comparado com a algoritmos de pontos-interiores que têm desempenho polinomial no pior caso. A complexidade computacional do método Simplex depende do número total de iterações e do número de operações elementares necessárias em cada iteração. O algoritmo "branch-and-bound" (Algoritmo de Bifurcação e Limite) pode ser entendido como uma extensão da estratégia de divisão e conquista para problemas de natureza inteira mista, ou seja, divida um problema P em um conjunto de subproblemas $\{SP_k\}$ de forma que a solução de P possa ser obtida através da solução dos subproblemas, resolva os subproblemas e, no final, obtenha a solução do problema de interesse. (Camponogara, 2003).

No algoritmo branch-and-bound, as divisões são feitas iterativamente, sempre observando que os subproblemas devem ser mais fáceis de serem resolvidos que o problema original, além de se procurar descartar subproblemas por meio de enumeração implícita; isto equivale a dizer que, de alguma forma, podemos garantir que a solução ótima não é solução de um certo subproblema e, por conseguinte, podemos descartá-lo.

Em Terra (2001) um problema de atribuição de disciplinas e professores com 22000 variáveis foi resolvido usando a técnica de branch-and-bound, obtendo solução ótima, embora no problema de tamanho máximo a solução obtida tenha sido apenas viável.

Um Sistema Interativo é um conjunto de programas computacionais criados para serem manipulados por usuários com o intuito de resolver um problema específico (Furtado, 1997). Ele é constituído por duas partes: uma parte não interativa, também chamada de aplicação, onde os dados serão manipulados pelo sistema durante a resolução de um problema; e de outra interativa, também chamada de apresentação, que corresponde aos módulos utilizados na exposição dos dados ao usuário e na recepção de estímulos e informações do usuário, possibilitando a troca de informações entre eles. Um sistema Interativo é um sistema onde o usuário pode inserir e modificar dados durante a execução do programa. No caso do problema de timetabling essa interação facilita muito a parte árdua da tarefa de montar uma grade horária, pois o usuário informa para o sistema as possíveis alocações que devem ser feitas e este checa se há restrições sendo violadas por aquela alocação.

A programação lógica é fundamentada na convicção de que ao invés dos seres humanos terem que adaptar seus pensamentos e raciocínios de forma que se possam expressá-los em termos de operações diretamente codificáveis num sistema computacional físico, o contrário é que deve ocorrer. Os sistemas computacionais reais é que devem evoluir e se tornarem capazes de manipular diretamente o domínio de conhecimento relacionado a um problema e, dessa forma, se tornarem aptos a resolvê-lo. Nesta abordagem cabe ao ser humano apenas estabelecer, as leis que regem o problema e os mecanismos necessários para a obtenção de soluções (Alencar, 2001). A Programação Lógica com Restrições (*Constraint Logic Programming, CLP*) é a tentativa de superar as limitações apresentadas pela programação lógica por meio da adição de mecanismos responsáveis pela resolução de restrições.

Em nosso cotidiano estamos submetidos a um enorme número de limitações vindas das mais diversas fontes externas. Essas limitações se apresentam, por exemplo, na insuficiência de recursos financeiros para adquirirmos algo que precisamos ou desejamos. Lidamos diariamente com tais condicionantes e somos obrigados a criar soluções de privação diante de tal realidade. A idéia de que, durante a resolução de um problema, se estabeleça a quais

restrições as possíveis soluções do problema devem estar subjugadas, é propícia para a obtenção das soluções adequadas. Uma proposta de associação de valores para cada uma das variáveis é apenas considerada solução se satisfizer a todas as restrições definidas. Em aplicações reais é habitual que haja preferência por certas associações em relação a outras possíveis e que satisfaçam, igualmente, as restrições impostas.

Karmarkar (1984) citado por Caixeta-Filho (2001), apresentou um método de pontos interiores para Programação Linear bastante notório ao campo da pesquisa, alcançando, em alguns casos, resposta final até 50 vezes mais rápido que o método Simplex. A resolução de um problema de Programação Linear através do Método Simplex se inicia em um extremo ao longo da fronteira da região viável, saltando para um ponto extremo vizinho melhor ao longo da fronteira, e finalmente parando no ponto extremo ótimo. Já o algoritmo de Pontos Interiores raramente visita pontos extremos antes de encontrar a solução ótima. Este método tem por objetivo caminhar pelo interior da região viável, até encontrar o ponto ótimo. Se por um lado a abordagem de Pontos Interiores (PI) requer maior tempo computacional encontrando uma direção de busca, por outro, o fato de se alcançar uma melhor direção de busca resulta em um número inferior de iterações.

Segundo Souza (2002), GRASP - Greedy Randomized Adaptive Search Procedure, ou procedimento de busca adaptativa gulosa e aleatória, é uma técnica iterativa proposta em Feo & Resende (1995) que consiste de duas fases: uma fase de construção, na qual uma solução é gerada, elemento a elemento, e outra fase de busca local, na qual um ótimo local na vizinhança da solução construída é pesquisado. A melhor solução encontrada ao longo de todas as iterações GRASP realizadas é retornada como resultado do algoritmo de otimização GRASP. Na Fase de Construção, uma solução é iterativamente construída, elemento por elemento. A cada iteração dessa fase, os próximos elementos candidatos a serem incluídos na solução são colocados em uma lista de candidatos, seguindo um critério de ordenação pré-determinado. O processo de seleção é baseado em uma heurística adaptativa gulosa que estima o benefício da seleção de cada um dos elementos. A fase de busca local começa de uma solução obtida pela fase de construção GRASP, navega pelo espaço de pesquisa passando de uma solução para outra, que seja sua vizinha, em busca de um ótimo. A eficiência da busca local depende da qualidade da solução construída na fase de construção. O algoritmo GRASP procura, portanto, conjugar bons aspectos dos algoritmos puramente gulosos, com aqueles dos algoritmos aleatórios de construção de soluções.

O Método de Pesquisa em Vizinhança Variável (Variable Neighborhood Search, VNS) proposto por Mladenovic & Hansen (1997, 1999) é um método de busca local que consiste

em explorar o espaço de soluções através de trocas sistemáticas de estruturas de vizinhança, explorando vizinhanças gradativamente mais "distantes" da solução corrente e focalizando a busca em torno de uma nova solução se e somente se um movimento de melhora é realizado. O método inclui, também, um procedimento de busca local a ser aplicado sobre a solução corrente. Esta rotina de busca local também pode usar diferentes estruturas de vizinhança. O funcionamento básico de um algoritmo VNS pode ser descrito como a seguir: parte-se de uma solução inicial "s" qualquer e a cada iteração seleciona-se aleatoriamente um vizinho "s" dentro da vizinhança da solução corrente. Esse vizinho é então submetido a um procedimento de busca local. Se a solução ótima local "s'' for melhor que a solução s corrente, a busca continua de "s'' recomeçando da primeira estrutura de vizinhança. Caso contrário, continua-se a busca a partir da próxima estrutura de vizinhança. Este procedimento é encerrado quando uma condição de parada for atingida, tal como o tempo máximo permitido de CPU, o número máximo de iterações ou número máximo de iterações consecutivas entre dois melhoramentos for usada.

CONCLUSÃO

O problema de confecção do horário acadêmico deve ser tratado segundo o tipo de escola e seu sistema educacional. Este problema tem sido extensivamente estudado ao longo das últimas décadas e muitas soluções já foram propostas com o intuito de solucioná-lo.

O estudo dos métodos apresentados por este trabalho nos permite concluir que, não há uma única técnica de otimização adequada para todos os problema de timetabling. Cada um dos métodos que já foram estudados apresentam resultados satisfatórios na maioria dos casos.

Os algoritmos genéticos têm apresentado bons resultados para variados tamanhos do problema. O algoritmo Branch and bound alcança bons resultados embora para problemas muito grandes os resultados obtidos sejam apenas viáveis.

O método GRASP precisa de boas soluções iniciais enquanto a busca tabu atinge boas soluções mesmo se a solução inicial é ruim, pois esta possui mecanismos de diversificação. O método GRASP só confia em randomização de uma iteração para outra e cada iteração se beneficia da qualidade da solução inicial.

O método VNS aplica randomização durante a busca local, enquanto as o método GRASP aplicam randomização durante a fase de construção.

REFERÊNCIAS BIBLIOGRÁFICAS

ALENCAR, W.S. *HORUS:* sistema de elaboração automática de horários de aulas combinando programação lógica com restrições hierárquicas. 2001. 231 f. Dissertação (Mestrado em Engenharia Elétrica e de Computação) - Universidade Federal de Goiás, 2001.

ALMEIDA, M.A.F. *Teoria dos autômatos e da computabilidade.* Lages, RS: UNIPLAC, 2001. (Notas de aula.)

CAIXETA-FILHO, J.V. *Pesquisa operacional:* técnicas de otimização aplicadas a sistemas agroindustriais. São Paulo: Atlas, 2001. 171 p.

CAMPONOGARA, E. *Métodos de otimização*: teoria e prática. Florianópolis, SC: UFSC, 2003. (Notas de aula)

COSTA, E.O.; BRUNA, M.D. *Resolução de timetabling utilizando evolução cooperativa.* Curitiba: UFPR, Departamento de Informática, 2003. 20p. Disponível em: <http://www.sbc.org.br/reic/edicoes/2003e1/cientificos/ResolucaoDeTimetablingUtilizandoEvolucaoCooperativa.pdf>

DANTZIG, G.B. *Linear programming and extensions*. Princeton: Princeton University, 1963. 632 p.

FEO, T.A.; RESENDE M.G.C. Greedy randomized adaptive search procedures. *Journal of Global Optimization*, v. 6, n. 2, p.109-133, Mar.1995.

FURTADO, Elizabeth. *Mise en oeuvre d'une méthode de conception d'interfaces adaptatives pour des systèmes de supervision à partir des spécification conceptuelles.* 1997. Thèse (doctorat) - Université d'Aix Marseille III, France, 1997.

JOHNSON, D. Timetabling university examinations. *Journal of the Operational Research Society,* v. 41, n. 1, p. 39-47, 1990.

KARMARKAR, N. A new polynomial-time algorithm for linear-programming. *Combinatorica,* v. 4, n. 4, p. 373-395, 1984. apud CAIXETA-FILHO, J.V. *Pesquisa operacional:* técnicas de otimização aplicadas a sistemas agroindustriais. São Paulo: Atlas, 2001. 171 p.

KIRKPATRICK, S.; GELLAT, D.C.; VECCHI, M. P. *Optimization by simulated annealing. Science*, n. 220, p. 671–680, 1983.

MLADENOVIC, N.; HANSEN, P. A variable neighborhood search. *Computers and Operations Research,* v.24, n. 1, p. 1097-1100, 1997.

SOUZA, M.J.F. *Inteligência computacional para otimização.* Ouro Preto: UFOP/DECOM/ICEB, 2002. (Notas de aula da disciplina Inteligência Computacional para Otimização)

TERRA, I.P. *Uma solução para a confecção do horário acadêmico*. 2001. 59 f. Dissertação (Mestrado) – Departamento de Ciência da Computação, Universidade Federal de Minas Gerais, 2001.