TÉCNICAS METAHEURÍSTICAS APLICADAS À CONSTRUÇÃO DE GRADES HORÁRIAS ESCOLARES

Arnaldo Moura[1], Rafael Scaraficci[2], Rafael Silveira[3], Volnei dos Santos[4]
Instituto de Computação – UNICAMP
Caixa Postal 6176, 13084-971 Campinas, SP
{[1]arnaldo, [2]rafael.scaraficci, [3]rafael.silveira, [4]volnei.santos}@ic.unicamp.br

Resumo
Este trabalho lida com um problema de otimização conhecido como agendamento acadêmico (school *timetabling*). A abordagem é testada sobre uma instância do problema, derivada da realidade brasileira, típica das escolas de ensino fundamental e médio, públicas ou particulares. O trabalho está embasado no uso de metaheurísticas evolutivas (Algoritmos Genéticos) e de busca local (Busca Tabu, GRASP), com a aplicação de técnicas alternativas específicas para obter melhores resultados sobre a instância do problema tratado (*Path Relinking*). Tais técnicas foram alvo de vários estudos e podem servir como referência para novas pesquisas com problemas semelhantes. O trabalho visa também, através da criação de uma interface amigável para usuários leigos, prover uma ferramenta útil para instituições de ensino brasileiras, que atualmente dispõem de poucas ferramentas para lidar com o problema e despendem grande quantidade de tempo para a montagem manual de seus horários.
Palavras-Chave: Agendamento Acadêmico, Algoritmos Genéticos, Busca Local, Hibridização.

Abstract
This work treats the optimization problem known as the school timetabling problem. The approach was tested over real data, typical of a Brazilian high school establishment. The development was based on evolutive programs (using genetic algorithms) and local search (tabu search and GRASP) together with alternative heuristics (*Path Relinking*) when appropriate. By using the same problem instances it was possible to compare the results. In addition to the heuristics solvers, a friendly user interface was also implemented. The interface allows for non-technical staff to use the solvers, thereby greatly limiting the many hours of manual labor that are used today to produce adequate solutions for the problem.
Keywords: Timetabling, Genetic Algorithms, Local Search, Hybridization.

1. Introdução

No início de cada período letivo, surge um problema sutil, porém de grande relevância para toda instituição de ensino: a alocação dos horários dos professores e turmas de alunos. Na ausência de uma ferramenta computacional para auxílio, o agendamento das atividades acadêmicas é uma tarefa laboriosa e que costuma consumir vários dias, às vezes semanas, de esforço de um ou mais funcionários que ficam encarregados exclusivamente desta função.

A resolução manual deste problema baseia-se num processo de tentativa e erro que consome muito tempo e que, na maioria das vezes, produz soluções de baixa qualidade. Uma agenda de horários ruim prejudica a instituição de ensino em vários sentidos: além do transtorno recorrente a cada período letivo, alunos e professores são prejudicados, insatisfeitos com uma distribuição ruim de suas cargas horárias, tendo excessivas aulas duplas, janelas de horários, poucos dias-livres, entre outras distorções.

A principal dificuldade apresentada por este problema, conhecido na literatura como *school timetabling*, encontra-se no fato de sua natureza combinatória. Analisando sob a ótica de um problema de otimização, o agendamento, apesar de já ter sido bastante explorado, continua sendo um desafio. Isto se deve a vários motivos, tais como a natureza combinatória do mesmo, a ampla gama de variações que pode assumir devido à peculiaridade das diferentes instituições de ensino, além das variações dos próprios sistemas de ensino em diferentes países. Dadas estas características, o emprego de metaheurísticas na busca de soluções para este problema é uma abordagem viável e que tem-se revelado bastante eficaz [CL98, SCH96], superando, inclusive, métodos tradicionais de otimização que envolvem processos mais determinísticos, como o uso de programação linear e inteira e programação por restrições.

Neste trabalho, abordamos o problema de agendamento através do uso de três heurísticas distintas: uma evolutiva usando algoritmos genéticos; outras duas de busca local usando busca tabu e GRASP. Estas duas últimas foram implementadas em duas versões distintas: uma implementação pura e outra hibridizada com o *path relinking*. No caso do algoritmo genético propõem-se uma modificação na estrutura do algoritmo tradicional o que ocasionou uma melhora substancial na qualidade da solução. Aplicando diversas técnicas na solução de um mesmo problema, permite comparar a eficácia e a eficiência das técnicas aplicadas a este problema específico, além de comparar as diferentes abordagens de uma mesma técnica, verificando os possíveis ganhos obtidos com o uso da hibridização. Um aspecto original desse trabalho está relacionado ao fato de que os horários de início de aulas no ensino fundamental e médio não coincidem. Isso força o tratamento do problema usando-se mais de uma grade horária, para evitar colisões nos horários de professores que lecionam nos dois níveis de ensino. É interessante salientar, também, que na literatura consultada para a realização deste projeto não foram encontrados indícios de que um problema de agendamento análogo a este tivesse sido modelado.

Além dos fatores relativos à aplicabilidade, o trabalho encontra também um viés na pesquisa e comparação de técnicas diversas aplicadas na resolução do problema. O enfoque dado às técnicas heurísticas (Algoritmos Genéticos, Busca Tabu e GRASP) deveu-se aos poucos resultados satisfatórios obtidos pelo grupo na utilização de técnicas determinísticas, no caso Programação por Restrições.

Além disso, para que uma ferramenta aplicada na resolução do problema de agendamento possa ser amplamente utilizada por leigos nas instituições de ensino, é necessário oferecer uma interface gráfica amigável para os usuários. Com a união desses dois fatores, um resolvedor heurístico mais uma interface amigável, é possível dispor de um sistema completo que pode ser usado na resolução de um problema que toma muitas horas de trabalho manual, na grande maioria das instituições de ensino brasileiras [OP03].

Este artigo encontra-se organizado da seguinte forma: na seção 2 é detalhada a instância do problema de agendamento tratado, com as peculiaridades inerentes ao modelo de ensino brasileiro. Na seção 3 faz-se a descrição da abordagem evolutiva, usando algoritmos genéticos, para o problema. A seção seguinte trata das técnicas de busca local (Busca Tabu, GRASP e *Path Relinking*) usadas para resolver o problema. Na quinta seção detalhamos os resultados obtidos nas implementações e apresentamos uma comparação de desempenho e qualidade dos métodos utilizados. A seção 6 ilustra uma visão geral do sistema de agendamento compreendendo os resolvedores e a interface gráfica do mesmo.

2. O Problema de Agendamento Escolar

O problema abordado neste artigo, em sua forma geral conhecido por *timetabling*, é de larga aplicabilidade nas mais variadas áreas, que vão desde a indústria, onde é realizado o agendamento dos horários dos funcionários e recursos, até as empresas prestadoras de serviços, como escolas e universidades. Até hoje não se conseguiu construir um sistema geral para busca de soluções do problema de *timetabling* com eficiência comprovadamente polinomial o que não causa surpresa, uma vez que o mesmo está incluso na classe dos problemas NP-Difíceis.

Esta dificuldade é típica dos problemas combinatórios, e o principal motivo disto é a enorme diversidade e complexidade das restrições que devem ser satisfeitas em cada caso. No caso da variante do problema conhecida como agendamento escolar (citado na literatura como *school timetabling*), as características variam de país para país, dadas as peculiaridades dos sistemas educacionais de cada um. Além disso, ocorrem também variações dentro de um mesmo país, como é facilmente verificado no Brasil. A instância do problema de agendamento usada neste trabalho é a de uma típica escola de ensino fundamental e médio do estado de São Paulo. Em particular, serão usadas as especificações do Colégio "Stella Maris", localizado na cidade de Santos, e de colégios públicos da região de Barão Geraldo, na cidade de Campinas. Mas o modelo é facilmente expansível para escolas de outras localidades e até mesmo para algumas instituições de ensino superior.

Em resumo, o problema consiste em distribuir os períodos das aulas de tal modo que nenhum professor ou classe esteja envolvido em mais de uma aula num dado período e de forma que cada professor ministre um número especificado de aulas para cada classe. Estes são exemplos de restrições fortes do problema, ou seja, restrições que não podem ser negligenciadas. Outra restrição forte é a que diz respeito ao preenchimento dos horários de uma turma, que garante a inexistência de espaços vazios, com exceção dos intervalos, entre uma aula e outra. Existem ainda outras restrições como, por exemplo, atender às

preferências de horários dos professores, evitar muitas aulas consecutivas de uma mesma matéria ou ministradas por um mesmo professor em uma dada classe, evitar períodos ociosos nos horários dos professores, entre outras. Estas últimas são classificadas como restrições fracas e podem ser violadas, desde que com parcimônia e quando houver necessidade. No sistema desenvolvido, presume-se que a carga horária de professores e turmas já vem alocada. Essa é a situação real dos colégios tomados como exemplo e também de grande parte das escolas de ensino fundamental e médio do Brasil.

Uma descrição formal, embora restrita, do problema de agendamento escolar é apresentada em seguida. Temos um conjunto com m turmas $\{c_1, ..., c_m\}$, um conjunto com n professores $\{p_1, ..., p_n\}$ e são considerados os períodos 1, ..., t. É dada também uma matriz R_{mxn} de inteiros não-negativos onde a célula r_{ij} é o número de aulas semanais que o professor p_j deve ministrar para a turma c_i. As disponibilidades de professores e turmas é dada por duas matrizes binárias auxiliares P_{nxt} e C_{mxt}, respectivamente, onde $p_{jk} = 1$ (e o correspondente $c_{ik} = 1$) se o professor p_j (ou a respectiva turma c_i) está disponível no período k, e $p_{jk} = 0$ (ou $c_{ik} = 0$) caso contrário. Ainda, D_{mxt} é uma matriz binária tal que se $d_{ik} = 1$ a turma c_i deve estar necessariamente em aula no horário k, e $d_{ik} = 0$ caso contrário [SCH96]. A formulação visa buscar x_{ijk} (i = 1, ..., m; j = 1, ..., n; k = 1, ..., t) tais que:

$$\sum_{k=1}^{t} x_{ijk} = r_{ij}, \text{ onde } (i = 1, ..., m; j = 1, ..., n) \quad (1)$$

$$\sum_{i=1}^{m} x_{ijk} \le p_{jk}, \text{ onde } (j = 1, ..., n; k = 1, ..., t) \quad (2)$$

$$\sum_{j=1}^{n} x_{ijk} \le c_{ik}, \text{ onde } (i = 1, ..., m; k = 1, ..., t) \quad (3)$$

$$\sum_{j=1}^{n} x_{ijk} = d_{ik}, \text{ onde } (i = 1, ..., m; k = 1, ..., t) \quad (4)$$

A variável binária x_{ijk} recebe o valor 1 quando o professor j ministra aula para a turma i no período k. A restrição (1) refere-se ao cumprimento da carga horária de cada professor tal como está especificado na matriz R_{mxn}. A restrição (3) garante que cada turma só tem aulas em períodos disponíveis. A restrição (4) força que haja aulas nos períodos para tal designadas. No trabalho, estas três restrições fortes são evitadas pela própria modelagem adotada. As demais restrições fortes do problema são:

- Aulas Simultâneas: é a restrição representada em (2), que indica que um dado professor pode ministrar no máximo uma aula num dado horário disponível.
- Indisponibilidades: também representada por (2), a qual significa que um professor só pode ministrar aulas caso esteja disponível no horário especificado.
- Aulas N-Uplas: são proibidas aulas n-uplas (n aulas no mesmo dia, seguidas ou não) para $n > 2$, se o número de aulas da matéria na semana for maior que 2. Caso o número de aulas seja igual a 2 são proibidas aulas duplas num dado dia.

Podemos listar as seguintes restrições fracas, que foram salientadas como mais significativas pelas escolas tomadas como exemplo:

- Janelas: minimizar o número de janelas no horário dos professores.
- Preferências: satisfazer horários à preferência dos professores, tanto horários fixos como preferências por dias livres na semana.
- Aulas Geminadas: agrupar as aulas duplas existentes.
- Espalhamento: distribuir as aulas de maneira equilibrada durante a semana.
- Aulas Coordenadas: é de interesse da escola que algumas aulas sejam coordenadas entre duas ou mais turmas distintas, mantendo-as num mesmo horário.

No caso do Colégio Stella Maris, devemos destacar dois pontos importantes a serem observados e que tornam este problema de agendamento um tanto mais complicado: (1) as janelas de professores são consideradas indesejáveis já que a escola deverá pagar pelos horários ociosos dos seus professores. Deste modo, deve-se dar maior prioridade à eliminação deste tipo de violação tal qual é feito para as restrições fortes; (2) os horários de aulas das turmas de ensino fundamental e médio não estão alinhados, isto é, os horários de início e fim das aulas não são iguais para turmas de níveis diferentes. Assim, existe mais de uma grade de horário neste colégio. Por este motivo, deve-se tomar um cuidado especial ao se fazer o agendamento, já que existem professores que lecionam disciplinas em turmas com grades distintas, evitando sobreposições de períodos de aula na alocação dos horários destes professores. Por este motivo, o sistema resolvedor desenvolvido propõe uma modelagem utilizando o que denominamos *modelo*

multigrades. Isto significa que a cada turma é associada uma grade de horários, permitindo que o resolvedor avalie as possíveis sobreposições nos horários dos professores.

Por fim, uma solução do problema de agendamento é dada por uma matriz X_{mxnxt} de valores binários atribuídos às variáveis x_{ijk}, onde $1 \leq i \leq m$, $1 \leq j \leq n$, $1 \leq k \leq t$. Denotamos uma solução por $\{x_{ijk}\}$ e o conjunto de todas as soluções por $\aleph$. A cada solução $\{x_{ijk}\}$ atribuímos penalidades positivas quando violam alguma das restrições apresentadas anteriormente. A forma como as penalidades são atribuídas é um dos importantes fatores de estudo, e que determinam o sucesso da heurística empregada. A *função objetivo*, F, mapeia cada solução $\{x_{ijk}\}$ na soma das penalidades atribuídas a essa solução. A heurística usada, então, procura uma solução $\{x_{ijk}\}$ onde $F(\{x_{ijk}\})$ seja mínimo, entre todas as soluções possíveis.

Diversas abordagens para o problema de agendamento escolar utilizando técnicas determinísticas e heurísticas podem ser encontradas na literatura [CL98], várias dessas obtendo relativo sucesso quanto aos resultados, principalmente em respeito às técnicas heurísticas. Alguns fatores motivaram o desenvolvimento de uma nova abordagem ao problema, proposta neste trabalho. Dentre esses fatores, podemos citar:

- Instâncias específicas: as aplicações desenvolvidas para resolução dos problemas de agendamento tratam de instâncias específicas, geralmente focadas nas diretrizes e restrições típicas do local da pesquisa. A maioria das pesquisas neste campo encontra-se nos Estados Unidos e Europa, ficando o Brasil com relativamente pouca diversidade de trabalhos na área.
- Restrições reduzidas: mesmo os projetos aplicados a instâncias brasileiras do problema apresentam diversas limitações quanto às restrições tratadas. Um exemplo típico é a abordagem *multigrade* dada ao projeto, que não foi encontrada na literatura pesquisada pelo grupo. Tal abordagem fornece uma grande flexibilidade às instituições de ensino na definição de seus horários, e uma ferramenta que leve em consideração esse fato é de grande valia em tais instituições.

3. Programas Evolutivos e Algoritmos Genéticos

Os Programas Evolutivos, em especial os Algoritmos Genéticos (ou *AGs*), são uma classe de heurísticas que utilizam métodos probabilísticos, inspirados em conceitos da teoria da Evolução Natural para resolução de problemas de otimização combinatória [CRR97, FAN94, MIC96]. Basicamente, para encontrar uma solução s de um problema de otimização, é considerado um conjunto P de n soluções $\{s_1, s_2, \ldots, s_n\}$, sendo cada solução s_i chamada de *indivíduo* e P chamado de *população*. Partindo-se de uma população inicial P_0, cujos indivíduos são gerados por métodos determinísticos ou aleatórios, o algoritmo simula ciclos de evolução, no qual novas populações P_i são criadas através do cruzamento (*crossover*) de informações de diferentes indivíduos da população P_{i-1}. Para isso também são realizados mutações e processos de seleção para, respectivamente, garantir a diversidade da população e tentar privilegiar os indivíduos melhor qualificados por uma *função de avaliação*, que destaca as melhores soluções.

As motivações para a aplicação de AGs no projeto são as mais diversas. Além dos fatores já citados na seção 2, a implementação do algoritmo genético deste projeto foi desenvolvida com diversas experimentações sobre o algoritmo tradicional [MIC96], a fim de se obter uma melhor qualidade das soluções e um melhor desempenho computacional. Algumas dessas modificações podem servir como ponto de pesquisa para outros trabalhos que utilizem AGs aplicados a problemas da mesma natureza.

3.1. Representação e Operadores

Para a construção de um algoritmo genético eficiente, vários pontos devem ser levados em consideração: a estrutura de dados que representa os indivíduos (*cromossomo*), os *operadores genéticos* definidos (*crossover* e *mutação*), como os indivíduos serão avaliados e qual o processo de seleção a ser aplicado. Esses fatores estão intimamente relacionados à natureza do problema, principalmente no que diz respeito aos cromossomos e operadores genéticos. Isso implica num esforço de pesquisa em representações e operações sobre as mesmas que sejam eficientes e preservem a idéia original do algoritmo: novas e melhores soluções do problema podem ser obtidas mesclando-se informações de soluções já encontradas.

Os cromossomos devem ser concebidos de forma a representar todas as soluções factíveis do problema com um mínimo de violações de restrições. No problema de agendamento, matrizes de horários são comumente utilizadas para representar as soluções, por serem uma forma intuitiva e de fácil manipulação. No caso do algoritmo desenvolvido nesse trabalho, o cromossomo reflete o modelo multigrade discutido na

seção 2 e é constituído de matrizes paralelas M_{cxt}, cada qual representando uma grade envolvendo c turmas em t horários de aula, onde são alocadas as matérias. Todas as grades são interligadas entre si, já que um mesmo professor pode lecionar matérias alocadas em grades diferentes.

O modelo multigrade acrescenta um grau de complexidade elevado ao algoritmo, uma vez que dificilmente pequenas alterações realizadas nos cromossomos, como mutação e crossover, não gerem infactibilidades nas soluções. Para evitar tal problema, dois modelos são comumente propostos na literatura [MS03]: a eliminação de indivíduos infactíveis na população e a correção das soluções infactíveis geradas. Ambas as soluções restringem o espaço de busca do algoritmo, o que fez que com que neste projeto fosse proposta uma forma híbrida de se tratar o caso: os indivíduos infactíveis continuam existindo na população, recebendo penalidades mais altas pelas violações de restrições fortes e sofrendo aplicação de operadores de *mutação corretiva,* para tentar eliminar gradativamente da população os genes que criam tais violações.

Os *operadores genéticos*, que são parte importante de um AG, receberam uma abordagem especial neste trabalho. Para o operador de *crossover*, responsável pela miscigenação de informações para a geração de novos indivíduos, foi adotada uma abordagem via o chamado *crossover uniforme*, que apresentou melhores resultados do que os operadores tradicionais de *crossover*. Este operador, no caso deste projeto, gera apenas um descendente a partir de dois indivíduos progenitores. Analisando cada turma da escola, é alocada para o novo indivíduo a turma correspondente com menor carga de violação de penalidades dentre os progenitores. Com isso, as boas alocações de horários de turmas são mantidas na descendência. Mas devido à alta carga de restrições fortes definidas para o problema e ao fato de que indivíduos infactíveis podem ser criados a partir desse processo, foi elaborada a proposta da *mutação corretiva*. Este operador de mutação age como um operador de mutação tradicional, que é aplicado com determinada probabilidade sobre o indivíduo e ocasiona uma modificação nas informações contidas no mesmo. A diferença na abordagem deste projeto está no fato da *correção*: ao fazer a modificação, o operador procura escolher locais onde ocorre uma violação forte do problema, como um professor lecionando aulas simultaneamente. Através de um método heurístico, o operador realoca uma das aulas problemáticas em outro horário, fazendo com que o indivíduo não possua mais aquela violação.

A diferença na abordagem via mutação corretiva daquela que usa procedimento de correções de indivíduos está no caráter probabilístico da mesma: ela só é aplicada em uma parcela dos indivíduos, e sobre uma única violação de cada vez, ao contrário dos algoritmos de correção tradicionais. O que se percebe na prática é que a quantidade de violações existentes nos indivíduos da população diminui consideravelmente com o passar das gerações, sem que isso impossibilitasse a investigação de determinados locais no espaço de busca.

3.2. Ciclo de Evolução

Definidos os cromossomos, os operadores genéticos e o processo de seleção, o algoritmo repete um ciclo onde uma nova população P_{i+1} é gerada a partir de uma população atual P_i, através de processos de *crossover* entre indivíduos selecionados da população. A seleção de tais indivíduos é feita através da chamada *Seleção por Torneio* [MIC96], que apresentou melhores resultados neste caso específico do que a tradicional *Seleção por Roleta*, encontrada na literatura. A Seleção por Torneio escolhe aleatoriamente 2 indivíduos da população, fazendo com que aquele que possui menor valor de penalidades seja selecionado para reprodução.

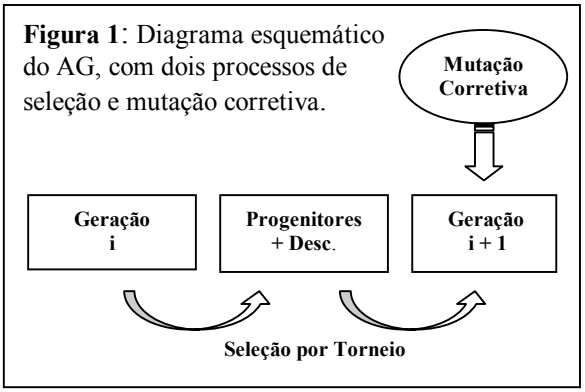


Figura 1: Diagrama esquemático do AG, com dois processos de seleção e mutação corretiva.

No algoritmo proposto, é realizado um outro processo de seleção de indivíduos *após* a geração dos descendentes, garantindo a sobrevivência apenas das proles mais adaptadas ao meio. Dessa forma, os indivíduos progenitores disputam com os descendentes para poderem fazer parte da nova geração. Esse procedimento dificulta a permanência na população de indivíduos com muitas violações de restrições do problema.

Diversos outros aspectos do algoritmo são discutidos de forma detalhada em [SAN03]. Mesmo com a grande quantidade de restrições fortes do problema, a robustez e flexibilidade do algoritmo foram fatores motivadores para a implementação do mesmo no projeto.

4. Heurísticas de Busca Local

De modo geral, as heurísticas de busca local são embasadas em processos que fazem o algoritmo inspecionar o espaço de soluções executando movimentos sobre a solução corrente. O que diferencia entre as diferentes metaheurísticas desta classe é a metodologia aplicada para fazer esta inspeção, isto é, as estratégias de intensificação e diversificação aplicadas, dentre outras técnicas específicas de cada abordagem. Nesta seção iremos apresentar três destas técnicas de busca local, as quais foram aplicadas ao problema de agendamento descrito: Busca Tabu, GRASP e *Path Relinking* [GL97, RR02, GLM00].

É importante citar que, no nível de implementação deste projeto, estas três técnicas utilizam estruturas de dados comuns, diferindo apenas no algoritmo de busca de cada uma. A estrutura que modela uma solução para o problema emprega um conjunto de matrizes que permitem diferentes visões da mesma solução, além do uso de estruturas para guardar os valores intermediários das penalidades recebidas por cada professor e turma. Estas especializações nas estruturas de dados propiciaram um alto nível de performance do algoritmo, especialmente no que se refere ao cálculo da função objetivo.

Outro aspecto fundamental na construção de um algoritmo de busca local é definir a estrutura de sua *vizinhança*. Isto é feito descrevendo-se os tipos de movimentos possíveis que poderão ser executados sobre uma solução de modo a gerar um vizinho da mesma. Foram definidos dois tipos diferentes de movimentos. O movimento simples de troca (*swap*), que é representado por uma quádrupla de inteiros *<t, g, slot1, slot2>*. Esta quádrupla representa um movimento que troca o par professor/matéria alocado no *slot1* da turma *t* na grade *g* pelo par professor/matéria alocado no *slot2* desta mesma turma. Observe que este tipo de movimento não permite que sejam violadas as restrições de carga horária de turmas e professores, evitando entrar em certas regiões infactíveis do espaço de busca. Todavia, este tipo de movimento pode levar a um vizinho pior do que a solução atual. Uma forma encontrada para remediar esta limitação, porém sem garantia de sucesso, foi a criação do movimento duplo, que consiste de uma seqüência de dois movimentos simples, sendo que o segundo movimento tem um caráter corretivo. A idéia baseia-se em gerar um movimento simples e verificar se este movimento causa a violação de alguma restrição não presente na solução corrente. Em caso afirmativo, gera-se um novo movimento simples procurando corrigir esta violação. A principal vantagem desta abordagem é a sua capacidade de escapar de mínimos locais. Pois, uma vez estagnado o processo de busca em um mínimo local, é permitido que o primeiro movimento piore a solução, ao passo que o segundo movimento corrija a penalidade, melhorando a solução.

O fato de que todos os resolvedores detalhados neste trabalho terem sido implementados usando programação orientada a objetos (em linguagem C++), permitiu uma fácil estruturação do projeto e a possibilidade de reutilização dos componentes desenvolvidos no projeto, a exemplo do que foi feito com os algoritmos de busca local, que compartilham as principais estruturas de dados entre si.

4.1. Busca Tabu

A Busca Tabu (BT) é uma metaheurística proposta por Fred Glover para resolver problemas de otimização combinatória. Trata-se de um método de busca local inteligente que procura evitar que a busca estacione em um ótimo local. Para conseguir este efeito, a heurística faz uso de estruturas de memória bastante flexíveis que são estrategicamente guiadas por um agressivo processo de busca no espaço de soluções. O algoritmo realiza busca no espaço de soluções com o intuito de, a cada iteração, encontrar o melhor vizinho da solução corrente e que não seja *tabu*, ou seja, proibido. Dada a flexibilidade desta metaheurística, é possível relaxar esta condição de tabu-ativo usando-se um *critério de aspiração*. Além disso, o processo de busca não se dá de forma estática, pois ocorrem alternâncias entre estratégias de *intensificação* e *diversificação* ao longo das iterações [GL97].

A *lista tabu* utilizada constitui-se de uma lista simples que armazena os *L* últimos movimentos realizados, os quais tornam-se tabu-ativos por *L* iterações. Esta lista é implementada usando-se uma tabela de espalhamento que permite atribuir e verificar o *estado tabu* do movimento em tempo de execução constante. Outro aspecto importante é que o algoritmo pode variar o comprimento desta lista de forma

dinâmica, moldando-a de acordo com a região do espaço de busca que está sendo analisada. Devido a isto, a heurística emprega um mecanismo de *relaxação adaptativa*, reduzindo o comprimento da lista em regiões promissoras, tornando a busca menos restritiva e, em contrapartida, aumentando o tamanho da lista quando a região que está sendo explorada mostra-se pouco promissora. O critério de aspiração empregado para relaxar a condição de tabu-ativo permite a execução de um movimento tabu caso ele promova um ganho quantitativo no valor da função objetivo.

O algoritmo usado pelo núcleo heurístico do resolvedor opera em ciclos iterativos compreendendo duas fases: uma fase que aplica uma estratégia de diversificação e outra que faz uso de um mecanismo de intensificação. Estas duas fases são implementadas em forma de um laço iterativo cujo critério de parada é um dado número de soluções sem melhorar a agenda atual:

- Diversificação: aplica um método de busca tabu simples aliado a uma forte estratégia para a criação de uma lista de movimentos candidatos. Nesta fase é possível que um dado movimento que seja executado degenere a qualidade da solução atual, já que a execução depende apenas da lista tabu e do critério de aspiração. Isto possibilita o algoritmo escapar de vizinhanças pouco promissoras.
- Intensificação: aplica uma busca local não-ascendente aliada aos efeitos da lista tabu e ignorando o critério de aspiração. Esta fase não permite movimentos que aumentem o valor da função objetivo.

A aplicação da alternância entre estas duas metodologias operando em ciclos obteve ótimos resultados, porém ela é dependente do balanceamento entre essas duas fases [SCH96]. Três parâmetros do algoritmo são associados a este balanceamento: número de iterações sem melhora da fase de diversificação e da fase de intensificação e o comprimento da lista de movimentos candidatos. Vários testes foram realizados para encontrar valores adequados para estes parâmetros. Maiores detalhes podem ser encontrados em [SM03].

4.2. GRASP (*Greedy Randomized Adaptive Search Procedure*)

GRASP é um método iterativo, proposto por Resende e Ribeiro [RR02], onde cada iteração consiste basicamente de duas fases: uma *fase de construção*, onde uma solução é gerada de maneira incremental via um procedimento guloso-aleatório, e de uma *fase de busca local*, na qual um mínimo local é pesquisado na vizinhança da solução construída. O método retorna a melhor solução encontrada ao longo das iterações.

Para a construção de uma solução inicial procede-se de maneira análoga a um construtor manual de quadros de horários, buscando agendar as aulas mais difíceis aos períodos mais críticos de maneira que viole o menor número de restrições. A dificuldade em se agendar uma aula é determinada por uma função que considera as indisponibilidades e o número de aulas ministradas pelo professor, além do fato de a aula ser coordenada e do professor atuar em mais de uma grade. Com relação aos períodos de aula, a criticidade é determinada pelo número de professores disponíveis para aquele horário. Quanto menos professores disponíveis, mais crítico é o período.

O caráter probabilístico é incorporado através de uma seleção aleatória da próxima aula a ser adicionada a solução, a partir de uma lista restrita de candidatos (LCR), a qual é formada pelas aulas de mais alta criticidade, sendo o seu tamanho regulado por um parâmetro α conforme descrito em [RR02]. Uma vez selecionada uma aula, procura-se agendá-la ao período mais crítico de forma que viole o menor número de restrições. Em seguida, atualiza-se a LCR e a lista de períodos críticos, visando incorporar as mudanças oriundas do agendamento da última aula. Procede-se desta maneira, até que todas as aulas tenham sido agendadas.

Para refinar as soluções construídas utiliza-se um método randômico não ascendente (RNA) que utiliza movimentos híbridos (simples e duplos). A idéia consiste, basicamente, em gerar um movimento duplo *md*, composto por dois movimentos simples *m1* e *m2*, e escolher o melhor movimento dentre *m1* e *md*. Esta técnica aceita tanto vizinhos de qualidade superior como de qualidade igual à solução corrente, permitindo encontrar caminhos de descida que passem por "regiões planas". No entanto, a principal vantagem deste método é a sua capacidade de ultrapassar mínimos locais, em virtude de poder executar movimentos duplos. A busca é interrompida após um dado número de iterações sem melhora no valor da função objetivo.

4.3. *Path Relinking*

O *path relinking* (PR), em sua forma clássica, é uma busca determinística que combina soluções de elite obtidas de uma população. Esta busca gera um caminho, i.e. uma seqüência de soluções, entre as

soluções de elite. O que se espera é que neste caminho que leva de uma solução a outra, possamos encontrar soluções de melhor qualidade que as soluções iniciais [GLM00]. Porém, é importante ressaltar que o PR não é uma heurística que pode ser usada independentemente pois o mesmo precisa, em sua inicialização, de soluções de boa qualidade que devem ser geradas por outros métodos. Por isso, neste projeto, as implementações que envolvem esta heurística são hibridizações com busca tabu ou GRASP.

No *path relinking*, os caminhos são gerados entre o par de soluções (s_1, s_2) do conjunto E de soluções de elite. O ponto inicial do caminho percorrido pela busca é a solução s_1 (denominada solução *inicial*) e o ponto final é a solução s_2 (dita solução *guia*). Na nossa abordagem do *path relinking*, os atributos comuns entre s_1 e s_2 nunca são modificados, enquanto que os atributos que fazem estas soluções diferir são modificados durante o caminho, minimizando as diferenças à medida que nos aproximamos da solução guia. Mais precisamente, o caminho que liga as soluções s_1 e s_2 é a seqüência de soluções (factíveis ou não) a seguir:

$$s_1 = m^0, m^1, m^2, ..., m^p, m^{p+1} = s_2$$

onde $m^{k+1} \in N^{PR}(m^k)$, sendo N^{PR} a vizinhança selecionada para o *path relinking*, de modo que a cada iteração do PR, as soluções geradas durante o caminho vão ficando cada vez mais próximas da solução final, dado algum critério para o cálculo de uma distância d entre duas soluções, baseando-se em um ou mais atributos da solução. De fato, ao fim do caminho de ligação devemos ter $d = 0$.

Na abordagem implementada para o problema de agendamento tratado, o atributo selecionado para o cálculo da distância entre duas soluções é a alocação de uma aula, representada pela quádrupla <*t*, *g*, *s*, *m*>, que indica que a matéria *m* está alocada no período *s* para a turma *t* da grade *g*. Assim, cada movimento de troca simples procura minimizar a diferença entre as duas soluções nas extremidades do caminho. Dessa forma, a heurística modifica progressivamente as matrizes da solução inicial fazendo-as ficarem cada vez mais similares com as matrizes da solução guia. Dada a dificuldade de manipular matrizes diretamente, estas foram separadas em *n* vetores e, deste modo, pode-se ligar as soluções fazendo-se a ligação individual de cada um dos vetores. Foram projetadas duas abordagens distintas do PR, diferindo apenas na forma como as matrizes são divididas para gerar os vetores:

- PR-Linha**:** as matrizes são divididas em linhas (que representam as turmas de alunos) e o PR é aplicado a estas linhas que são selecionadas aleatoriamente, uma de cada vez;
- PR-Coluna**:** similar à técnica anterior, porém usa como vetores as colunas das matrizes, que representam os horários de aulas dos alunos.

O PR clássico vêm obtendo bons resultados no tratamento de problemas combinatórios clássicos, tais como *job shop scheduling* e roteamento de veículos [GLM00]. Porém, o agendamento possui uma vizinhança muito extensa fazendo-se necessário modificar a abordagem tradicional de modo a adequá-la a este problema. Primeiramente, é computacionalmente inviável fazer uma busca que retorne o melhor movimento possível para se executar o próximo passo do PR. Assim o que se faz é encontrar um movimento de qualidade razoável e executá-lo. Mesmo assim, a adoção deste método faz com que as soluções intermediárias geradas possuam infactibilidades, inviabilizando a construção de uma nova solução de elite durante o caminho. Assim, a principal modificação adotada no PR aplicado neste projeto é o emprego de buscas locais durante o caminho da busca. Como já explicitamos, o PR implementado possui várias fases, uma para aproximar cada vetor proveniente das matrizes que contêm a solução. Ao final, de cada uma dessas fases é realizada uma busca local com um dado número *I* de iterações. A solução que foi gerada pelas *I* iterações de busca local é comparada com a atual solução elite, tomando seu lugar caso seja de melhor qualidade.

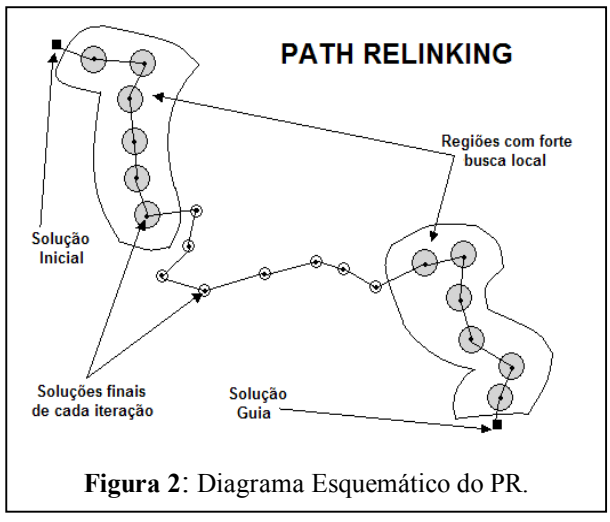


Figura 2: Diagrama Esquemático do PR.

Testes preliminares demonstraram que as melhores soluções obtidas usando o PR são, quase sempre, encontradas nas proximidades dos pontos extremos do caminho, isto é, nas primeiras ou últimas fases do PR. Baseando-se neste fato, julgou-se desnecessário o emprego de uma forte busca local nas soluções

intermediárias mais afastadas das extremidades. Esta observação economiza bastante tempo de processamento evitando buscas desnecessárias em regiões pouco promissoras. Assim, o algoritmo usa um parâmetro que indica a máxima distância dos extremos até a qual se podem realizar fortes processos de busca local. Um resumo esquemático do PR desenvolvido neste trabalho pode ser visto na figura 2.

O uso do *path relinking,* em geral, está atrelado a uma outra técnica de otimização já que o mesmo exige que soluções de boa qualidade lhe sejam passadas como parâmetro para a realização da busca. No caso deste trabalho, o PR foi hibridizado com as outras duas metaheurísticas de busca local, a Busca Tabu e o GRASP. Esta hibridização deu-se de duas formas distintas: (1) o uso do PR como uma técnica de *pós-otimização* das soluções obtidas pela BT e pelo GRASP; neste caso são passadas duas soluções geradas por estas técnicas e é realizado um PR *bidirecional* (fazendo do caminho da solução s_1 até s_2 e depois, inversamente, da solução s_2 à s_1) sobre as mesmas; (2) a *hibridização interna* com as duas técnicas de busca, isto é, o PR é aplicado sobre soluções obtidas em iterações intermediárias da busca tabu e do GRASP.

Na busca tabu, a hibridização interna deu-se da seguinte maneira: relembrando que durante a BT é mantida apenas uma solução de elite, e que quando é encontrada uma nova solução de melhor qualidade esta substitui a solução de elite atual, a modificação que foi feita para a inclusão do PR é que, ao invés de apenas substituirmos a solução de elite anterior pela atual, faz-se, antes disso, um PR bidirecional entre as duas. Ao final, se foi obtida uma solução ainda melhor, essa que vai substituir a solução de elite corrente.

Para o GRASP, utilizou-se uma idéia análoga à empregada para a busca tabu, sendo que a cada iteração GRASP, aplica-se um PR bidirecional entre a melhor solução obtida até a iteração anterior e a solução produzida pela iteração corrente. Caso seja encontrada uma solução de qualidade superior em relação às demais soluções, esta nova solução passa a ser a solução de elite.

5. Resultados

Ao longo de todo o trabalho de desenvolvimento do sistema foram realizadas várias séries de testes para possibilitar a definição dos melhores parâmetros a serem adotados pelos respectivos resolvedores. Porém, existe uma vasta quantidade de parâmetros aos quais as diferentes técnicas estão sujeitas. Devido a isto julgou-se desnecessário fazer aqui uma análise individual destes parâmetros. Assim, será delineado um estudo de mais alto nível, mostrando apenas os resultados finais obtidos por cada heurística (empregando a sua melhor combinação de parâmetros) e será apresentada uma comparação entre estes resultados. Para maiores detalhes sobre os resultados individuais dos testes das heurísticas consulte [SM03, MS03, RAS04].

Sendo assim, para possibilitar uma análise objetiva, deve-se definir alguns pontos de comparação que independam das técnicas avaliadas e que não favoreçam nenhuma delas, garantindo uma análise adequada. Sabendo que os valores de penalidades aplicados pelas diferentes técnicas em suas funções objetivos são um tanto díspares, deve-se, ao invés de realizar uma análise quantitativa sobre estes valores, definir um padrão que qualifique objetivamente as soluções que são dadas como saída pelo resolvedor heurístico de cada técnica. Assim, convencionou-se qualificar as soluções em cinco classes diferentes que se baseiam na quantidade de violações das restrições dos diferentes tipos. Estas classes de soluções são definidas abaixo:

- **(A)** Solução contendo apenas violações de restrições fracas, sem janelas nos horário dos professores;
- **(B)** Solução violando apenas restrições fracas com, no máximo, três janelas e três aulas coordenadas não satisfeitas;
- **(C)** Solução que viola, no máximo, quatro janelas, quatro aulas coordenadas e duas aulas n-uplas;
- **(D)** Solução que viola, no máximo, cinco janelas, cinco aulas coordenadas e três aulas n-uplas;
- **(E)** Solução infactível, ou seja, aquela que viola alguma restrição fisicamente forte (violações de indisponibilidade de horário, aulas simultâneas ou de carga-horária).

Além de comparar a qualidade média da melhor solução final gerada, devemos também avaliar os tempos de processamento demandados para encontrar tal solução. O tempo será medido em segundos, desta forma também tornamos esta medição independente do resolvedor avaliado, já que, por exemplo, o AG tem seu tempo dado pelo número de gerações geradas, enquanto que na BT ele é dado pelo número de ciclos e no GRASP pelo número de iterações.

A tabela 1, dada a seguir, reúne todos os dados relevantes ao nosso estudo comparativo, destacando os tempos de processamento e a distribuição média das melhores soluções de cada resolvedor heurístico pelas diversas classes de qualidade definidas. Executou-se 100 vezes cada um dos resolvedores e coletou-se os

tempos de execução e a melhor solução gerada nesse tempo. A tabela 1 mostra o tempo médio de execução e quantidade de soluções, dentre as 100 de cada resolvedor, em cada classe de qualidade.

Metaheurística[1,2]	Tempo (s)[1]	Classe A	Classe B	Classe C	Classe D	Classe E
BT	240	93	7	0	0	0
BT + PR[3]	420	85	15	0	0	0
GRASP	564	94	6	0	0	0
GRASP + PR[3]	334	96	4	0	0	0
AG	600	0	60	25	15	0

Tabela 1: Tempo de execução e distribuição qualitativa das soluções geradas pelas diferentes técnicas.
[1] BT = Busca Tabu; BT + PR = Busca Tabu com *Path Relinking*; GRASP + PR = GRASP + *Path Relinking*.
[2] Testes realizados em computadores Pentium IV de 2.4GHz.
[3] Utilizando a abordagem PR-Coluna Bidirecional (vide seção 4.3).

Um primeiro fato a se reparar na tabela 1 é a qualidade das soluções obtidas com o Algoritmo Genético. O GA apresentou grandes dificuldades de convergência para o modelo, principalmente com relação às aulas coordenadas (aulas de turmas distintas que precisam ser mantidas no mesmo horário), o que é caracterizado pelo tempo elevado de convergência e pela ausência de soluções de excelência. Relaxada essa restrição, o nível das soluções se aproxima dos demais algoritmos. As técnicas específicas citadas na seção 3, embora tenham ocasionado uma grande melhora no algoritmo [SAN03], não foram suficientes para tratar todas as restrições modeladas. Diversas experimentações podem ser realizadas para tentar sanar esse problema. Dentre elas podemos citar técnicas híbridas com buscas locais, que apresentaram maior facilidade em tratar tais restrições. Na seção 7 são dados mais detalhes sobre possíveis trabalhos futuros.

No caso da heurística GRASP, obteve-se sucesso na solução de todo o conjunto de restrições, a única diferença entre as soluções obtidas deve-se ao número de restrições fracas minimizadas. A incorporação do *path-relinking* à heurística trouxe uma melhora significativa tanto para a qualidade das soluções (o número de restrições fracas não minimizadas é menor) como para o tempo do processamento. Para o problema tratado o GRASP-PR mostrou-se uma técnica computacionalmente viável.

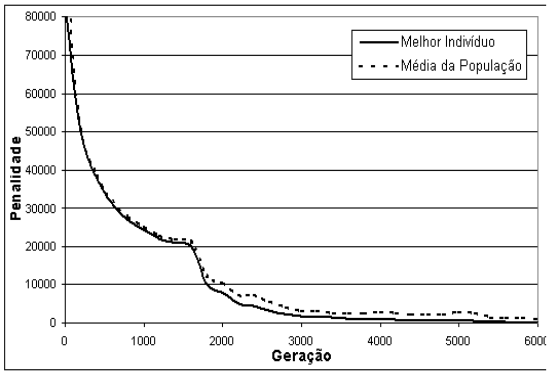
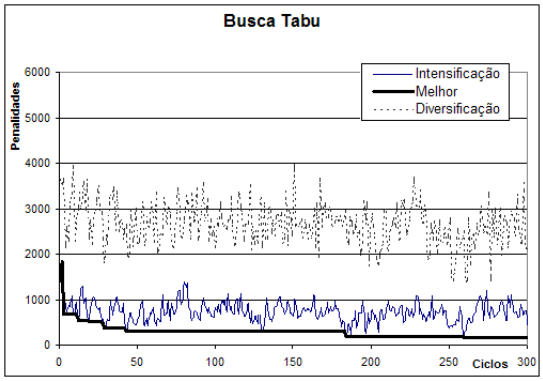


Fig. 3 (a) Algoritmos Genéticos



(b) Busca Tabu

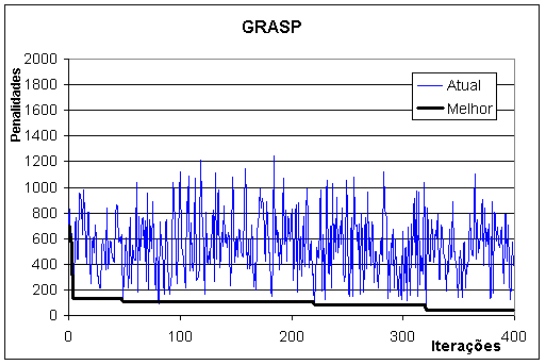
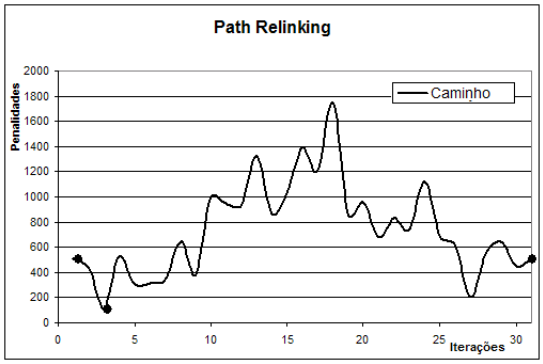


Fig. 3 (c) GRASP



(d) Path Relinking

Figura 3: Evolução temporal dos algoritmos.

A busca tabu obteve um resultado satisfatório, obtendo soluções de alta qualidade em mais de 90% dos casos em um baixo tempo. Comparando a versão pura da BT com a sua versão híbrida com PR vemos que não obtivemos ganhos no uso da hibridização, ao contrário do que aconteceu com o GRASP. Serão necessários mais testes para encontrar parâmetros mais refinados para o algoritmo de BT+PR. Outra possível melhoria na BT é a inclusão de movimentos duplos que também obteve bastante sucesso quando usado pelo GRASP.

De modo geral, as heurísticas de busca local conseguiram maior êxito na resolução do problema de agendamento. Estas heurísticas geraram soluções de boa qualidade em um tempo satisfatório. Dois aspectos foram fundamentais para este êxito: a escolha acertada da vizinhança (com movimentos simples e duplos) e a construção de algoritmos eficientes para o cálculo da função objetivo.

Por fim, é interessante também avaliar o comportamento das heurísticas durante a sua execução, isto é, acompanhar a evolução temporal do valor da função objetivo durante o processamento feito pelo resolvedor. Este comportamento está apresentado na figura 3.

Acompanhando a evolução temporal do AG (figura 3a), percebemos uma mudança de comportamento da convergência em torno da geração 1500, com a melhora da qualidade das soluções. Nesse momento, o algoritmo inicia a aplicação mais intensiva de mutações corretivas, o que evidencia os benefícios ocasionados pelas mesmas na qualidade dos indivíduos. A busca tabu, que opera de maneira cíclica, alternando entre fases de intensificação e diversificação, tal fato explica o comportamento oscilatório do seu resolvedor. Vemos três curvas na figura 3b, que representam o nível de qualidade (inverso da soma das penalidades) da solução de elite e das soluções atuais da fase de intensificação (curva oscilatória com menores valores de penalidade) e da fase de diversificação (curva com maiores valores de penalidade).

No GRASP, observamos um comportamento cíclico típico desta heurística (vide figura 3c), onde cada iteração é independente sendo que a melhor solução pode ser obtida em qualquer iteração. A figura 3d apresenta uma evolução típica de um PR unidirecional entre duas soluções (inicial e final, representadas nos pontos extremos do gráfico). Por este gráfico pode-se ver claramente que as soluções mais próximas dos pontos extremos possuem um menor valor de penalidade, mostrando inclusive que foi encontrada uma nova solução de elite, isto é, com melhor qualidade do que as soluções nas extremidades (marcada no ponto intermediário do gráfico).

6. Interface

Para um sistema que se coloca como uma ferramenta de trabalho para usuários finais, é imprescindível a construção de uma interface de uso que permita aos usuários leigos manipular as informações do problema e obter as soluções desejadas. No caso deste projeto, a interface deve também, de maneira transparente, lidar com as diferentes técnicas utilizadas, sem que o usuário saiba necessariamente o que cada uma representa.

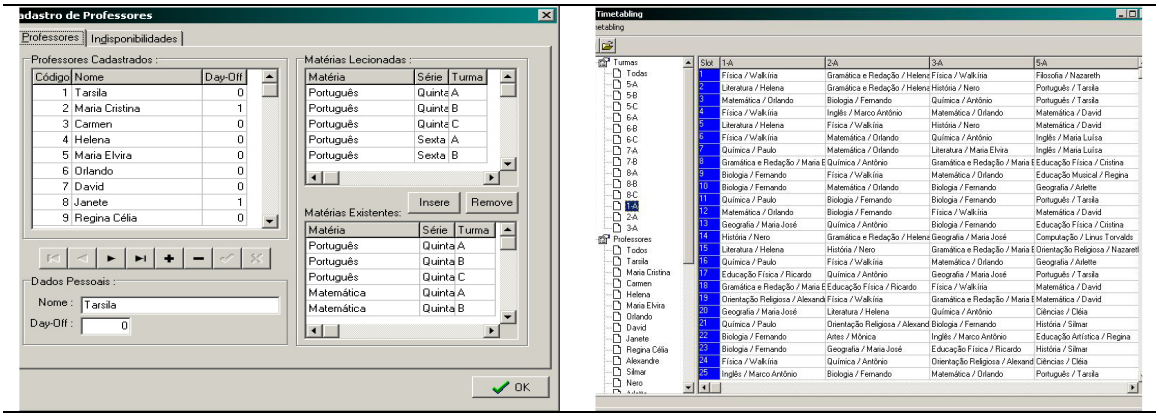


Fig. 4: (a) Exemplo de tela de entrada de dados. (b) Exemplo de tela de saída de dados.

Figura 4: Interface gráfica de entrada e saída do sistema.

A interface desenvolvida no projeto é constituída de conjuntos de telas para entrada de dados e para exibição dos resultados. As primeiras permitem que os usuários das escolas informem ao sistema quais são as características do problema, tais como carga horária das classes, competências e indisponibilidades dos

professores, dentre outras. As demais telas são específicas para mostrar ao usuário, de uma forma familiar, as agendas geradas pelos núcleos resolvedores a partir dos dados informados anteriormente. A interface apresenta ao usuário a agenda gerada sob várias visões distintas, permitindo que ele escolha se deseja focar apenas o horário das turmas ou dos professores. Pode-se também escolher a visualização de uma única turma ou de um único professor por vez. As telas de exibição permitem também que o usuário realize facilmente pequenos ajustes nas agendas geradas, da maneira que melhor lhe convir. Com isso, o sistema pode se tornar uma ferramenta de uso efetivo para as instituições de ensino.

Nas figuras 4a e 4b são mostrados, respectivamente, exemplos de telas de entrada e saída implementadas no projeto, que incluem as funcionalidades essenciais da interface usual.

7. Conclusões

Analisando-se os resultados exibidos na seção 5, confirma-se o sucesso das modelagens e implementações desenvolvidas para tratar o problema. O objetivo inicial de se construir uma ferramenta para agendamento automático através da utilização de metaheurísticas foi alcançado em sua plenitude: as soluções obtidas, além de perfeitamente viáveis do ponto de vista qualitativo, são encontradas em tempos de processamento relativamente baixos, mesmo usando-se computadores pessoais comuns. Por fim, a interface atende aos requisitos mínimos exigidos para se manipular o sistema, permitido a sua utilização por leigos.

Os trabalhos continuarão no sentido de aperfeiçoar o sistema e estudar o comportamento das heurísticas mediante abordagens paralelas e híbridas. Diversas experimentações podem ser sugeridas no Algoritmo Genético para contornar o problema de convergência com aulas coordenadas, citado na seção 5. Dentre elas podemos citar a hibridização do sistema com as técnicas de busca local, a paralelização do algoritmo através de diversos modelos existentes na literatura [MIC96] e mesmo a aplicação de algoritmos de correção mais agressivos.

Outras possibilidades para trabalhos futuros seriam a possível implantação do sistema em escolas públicas e a abordagem do problema de agendamento universitário, utilizando-se os requisitos da própria Unicamp.

Referências Bibliográficas

[CL98] Michael CARTER e Gilbert LAPORTE. *Recent Development in Practical Course Timetabling*. Relatório técnico, University of Toronto e École des Hautes Études Commerciales de Montréal, 1998.

[CRR97] Colin R. REEVES. *Genetic Algorithm for the Operations Researcher*. INFORMS Journal on Computing, 1997.

[FAN94] Hsiao-Lan FANG. *Genetic Algorithms in Timetabling and Scheduling*. University of Edinburgh - Department of Artificial Intelligence, 1994.

[GL97] Fred GLOVER e Manuel LAGUNA. *Tabu Search*. Kluwer Academic Publishers, 1ª Edição, 1997.

[GLM00] Fred GLOVER, Manuel LAGUNA, and Rafael MARTÍ. Fundamentals of Scatter Search and Path Relinking. Control and Cybernetics, 2000.

[KSM03] Eliana G. KOTSKO, Maria T. A. STEINER e Artur L. F. MACHADO. Otimização na Construção da Grade Horária Escolar – Uma Aplicação. Anais do XXXV Congresso da Sociedade Brasileira de Pesquisa Operacional, Natal-RN, 2003.

[MS03] Arnaldo V. MOURA e Volnei dos SANTOS. Algoritmos genéticos aplicados ao problema de agendamento de atividades acadêmicas. Instituto de Computação - Unicamp, *IC-03-27*, 2003.

[MIC96] Zbigniew MICHALEWICZ. *Genetic Algorithms + Data Structures = Evolution Programs*. Springer, 3ª edição, 1996.

[OP03] José A. OLIVEIRA e Rogério P. PINHEIRO. Ambiente de Otimização na Web: Uma Aplicação em Timetabling. Anais do XXXV Congresso da Sociedade Brasileira de Pesquisa Operacional, 2003.

[RAS04] Rafael A. SCARAFICCI. Aplicação da Metaheurística GRASP ao Problema de Agendamento Escolar. Relatório de Atividades – Processo FAPESP 03/09459-4, 2004.

[RR02] M. G. C. RESENDE and C. C. RIBEIRO. Greedy randomized adaptive search procedures. *Handbook of Metaheuristhics*, pages 219-249. Kluwer Academic Publishers, 2002.

[SCH96] Andrea SCHAERF. *Tabu Search Techniques for Large High-school Timetablig Problems*. Relatório Técnico, Centrum voor Wiskunde en Informatica, 1996.

[SM03] Rafael L. SILVEIRA e Arnaldo V. MOURA. Aplicação de heurísticas de busca local ao problema de agendamento escolar. Relatório Técnico IC-03-28, Instituto de Computação – Unicamp, 2003.