CAPÍTULO 1

INTRODUÇÃO

Problemas de otimização de natureza numérica, sujeitos a múltiplas restrições, envolvem basicamente uma função-objetivo a ser otimizada e um conjunto de restrições que devem ser simultaneamente atendidas, na forma:

$$\begin{cases} \min\limits_{x} < \text{função - objetivo dependente de } x > \\ \text{s.a.} \quad < \text{conjunto de restrições impostas a } x > \end{cases}$$

Duas etapas estão envolvidas no processo de solução (SHERALI *et al.*, 1993):

1. formulação matemática do problema, procurando descrever todos os aspectos relevantes, seja em termos de objetivos a serem otimizados ou restrições a serem atendidas;
2. obtenção da solução pela aplicação de ferramentas de otimização, desenvolvidas a partir da formulação matemática do problema.

Dependendo das características do problema, a sua formulação leva a um processo de solução que pode ser expresso algebricamente, de forma fechada. Como exemplo, considere a seguinte instância de um problema de otimização multivariável, com restrições lineares de igualdade:

$$\begin{cases} \min\limits_{\mathbf{u}} \dfrac{1}{2}\left\|\mathbf{u}-\mathbf{z}\right\|_2^2, \text{ com } \mathbf{u}, \mathbf{z} \in \Re^n, \ A \in \Re^{m\times n} \ (m < n) \text{ e } \mathbf{b} \in \Re^m, \\ \text{s.a.} \quad A\mathbf{u} = \mathbf{b} \end{cases}$$

sendo:

- $\left\|\cdot\right\|_2$ a norma euclidiana, tal que $\left\|\mathbf{x}\right\|_2^2 = \sum_{i=1}^{n} x_i^2$ para $\mathbf{x} \in \Re^n$;
- $\mathbf{z}$ uma constante e A uma matriz de posto completo (indicando que AA^T é inversível);

Aplicando as condições necessárias de otimalidade, resulta a solução ótima:

$$\mathbf{u}^* = \mathbf{z} - A^T\left(AA^T\right)^{-1}\left(A\mathbf{z}-\mathbf{b}\right).$$

No entanto, grande parte dos problemas de otimização apresentam características (por exemplo, não-linearidades) que impedem a obtenção de uma solução analítica, requerendo a aplicação de processos iterativos de busca da solução, a partir de uma condição inicial. A busca iterativa é geralmente implementada em computador, devido ao elevado custo de processamento associado à sua execução.

A presença de restrições de diversas naturezas introduz uma complexidade adicional por dividir o espaço de candidatos à solução em duas classes: soluções factíveis (que atendem a todas as restrições simultaneamente) e soluções infactíveis (que violam uma ou mais restrições).

Além disso, outra questão a ser considerada é o comportamento de convergência do processo iterativo. Três condições são desejáveis:

- garantia de convergência;
- tempo de convergência compatível com as necessidades de cada aplicação;
- convergência para uma solução de boa qualidade.

Basicamente, quanto mais complexo o problema de otimização e quanto maior seu tamanho (número de variáveis envolvidas), menor a chance de ter atendidas estas três condições simultaneamente, principalmente quando o método de otimização empregado não é suficientemente poderoso na execução do processo de busca da solução.

Neste trabalho, serão considerados problemas com elevado grau de complexidade, apresentando múltiplos objetivos devidamente ponderados, múltiplas restrições e grande número de variáveis. Estas propriedades levam a uma explosão combinatória de candidatos à solução, de modo que uma busca exaustiva pela solução ótima, dentre as soluções candidatas, representa um procedimento computacionalmente intratável (AHUJA *et al.*, 1993).

A tratabilidade de problemas de otimização com este grau de complexidade é conseguida caso se abra mão da solução ótima, em prol da obtenção de uma solução de boa qualidade. Como muitas vezes não se têm condições de saber o quão distante da solução ótima se encontra a solução obtida, a qualidade desta é medida em relação às soluções candidatas anteriormente produzidas pelo processo iterativo, a partir de uma condição inicial.

Dentre outras técnicas, a computação evolutiva tem se destacado no tratamento deste tipo de problema (BÄCK *et al.*, 1997b), pelas seguintes razões (GOLDBERG, 1989):

- por trabalhar com múltiplos candidatos à solução ao mesmo tempo em uma dada geração;
- por privilegiar melhores soluções candidatas na composição das próximas gerações;
- por ser capaz de detectar regiões promissoras no espaço de busca e atingir soluções de boa qualidade sem a necessidade de avaliar um grande número de candidatos.

Segundo BEASLEY (1997), o termo computação evolutiva é relativamente novo, tendo surgido em 1991, e representa um esforço para unir os pesquisadores que trabalhavam com simulação computacional nas áreas ligadas à evolução. As técnicas de algoritmos genéticos, estratégias e programação evolutiva têm um ponto fundamental em comum: envolvem aplicação de operadores genéticos, avaliação e seleção de indivíduos da população. Essas características formam a essência do processo de evolução, seja ele em computador ou no mundo real.

Conforme FOGEL (1997), as aplicações de computação evolutiva podem ser distribuídas em várias áreas, sendo que cada uma delas possuirá uma grande quantidade de ramificações. Em geral, seriam as seguintes: planejamento, modelagem, identificação, controle e classificação. Esta técnica está sendo aplicada na solução de problemas cada vez mais complexos e de natureza diversificada.

A grande dificuldade na aplicação de computação evolutiva a problemas multi-objetivo e com múltiplas restrições é como chegar a uma representação genotípica das soluções candidatas e como obter novas soluções candidatas que sejam factíveis, a partir da aplicação de operadores genéticos (MICHALEWICZ, 1997, C5.5). Uma das contribuições deste trabalho está na proposição de uma representação genética compacta, associada a um algoritmo de expansão de código, representando uma iniciativa de se explorar conjuntamente a computação evolutiva e métodos de busca baseados no atendimento de restrições. Um gerador de números pseudo-aleatórios com a propriedade de repetitividade é utilizado para garantir que a expansão de código sempre produza o mesmo código expandido para cada código compacto possível.

A estratégia de solução que será utilizada envolverá heurísticas para a geração de populações factíveis para o processo evolutivo, pela aplicação de um algoritmo de reparação. Além disso, procedimentos de busca local são empregados para aumentar a velocidade de convergência para ótimos locais.

1.1 Estudo de Casos

Caso 1: Geração Automática de Turnos Completos em Torneios

Em torneios envolvendo disputas dois-a-dois entre os participantes, sendo cada disputa denominada um jogo do torneio, usualmente devem ser atendidas as seguintes restrições básicas:

- cada participante deve jogar contra todos os outros uma única vez;
- cada participante deve obrigatoriamente jogar uma única vez na mesma rodada, a qual representa o menor conjunto de jogos que inclua todos os participantes;
- assumindo que existe distinção de mando de jogo e considerando todas as rodadas, a diferença entre o número de jogos no domínio de cada participante e fora dele deve ser no máximo um.

Se n é o número de participantes, então um turno completo será composto por $n-1$ rodadas, sendo que existe uma explosão combinatória de candidatos factíveis à solução (turnos completos). É assumido aqui que n é par. Na eventualidade de n ser ímpar, basta introduzir um participante fictício, indicando que seu adversário naquela rodada irá folgar. Se o torneio envolver dois turnos, o segundo turno vai representar os jogos de volta do primeiro turno, já que existe uma distinção entre jogos dentro e fora do domínio de cada participante.

Em face da existência de um grande número de candidatos factíveis à solução, é necessário estabelecer os objetivos a serem otimizados, de modo que existam soluções factíveis de melhor qualidade, ou seja, que maximizam o atendimento dos objetivos, as quais devem ser encontradas por um processo de busca iterativa. Neste trabalho, os objetivos a serem atendidos são os seguintes:

- minimizar o número de vezes em que qualquer participante do torneio jogue r ou mais vezes seguidas dentro (ou fora) de seus domínios, sendo que o valor de r depende de n;

- minimizar a diferença entre o maior e o menor percurso a ser realizado pelos participantes do torneio para completar seus jogos. Uma medida de variância entre os percursos poderia também ser minimizada neste caso.

Caso 2: Definição de Grade Horária em Instituições de Ensino

O problema a ser abordado envolve a alocação de carga didática na definição de uma grade horária semanal em instituições de ensino. Os candidatos à solução serão evoluídos, geração a geração, pela aplicação de operadores genéticos devidamente adaptados ao contexto. Este tipo de problema mostra-se compatível com a estratégia de solução proposta, uma vez que é fácil perceber a explosão combinatória existente, além da necessidade de se atender a múltiplas restrições e objetivos. Em outras palavras, existirá uma grande quantidade de combinações para a formação da grade, sendo que este valor será função do número de professores e do número de aulas concomitantes para a definição do horário. Veja o exemplo simplificado a seguir e a Tabela 1.1, mostrando a explosão combinatória existente.

Exemplo: Considere uma grade horária que tenha um único período de aulas durante todos os cinco dias da semana (de segunda à sexta-feira). Adota-se, para esse exemplo, que 25 professores (*A*, *B*, *C*, ..., *X*, *Y*) devem ser alocados uma única vez durante toda a semana e que o número de aulas em paralelo em um mesmo período seja cinco. A Figura 1.1 mostra uma das soluções possíveis para este exemplo.

Esta é uma versão demasiadamente simples para um problema de alocação de carga didática, pois as únicas restrições a serem atendidas são:

1. disponibilidade de salas de aula;
2. ausência de repetição de professores durante o processo de alocação.

Na prática, como também nos casos a serem considerados neste trabalho, existem outras especificações que devem ser incorporadas ao problema:

- restrições adicionais a serem atendidas:

3. professores com cargas horárias múltiplas e distintas;
4. múltiplos períodos no mesmo dia;
5. atendimento das impossibilidades dos professores em certos períodos;

6. atendimento das demandas dos cursos, no sentido de ser necessário ministrar esta ou aquela disciplina neste ou naquele período, o que implica na necessidade de escolha de um professor vinculado àquela disciplina e não qualquer professor disponível;
7. atendimento da demanda com um número mínimo de professores.

- objetivo a ser otimizado:

1. atendimento das preferências dos professores por certos períodos e também por certas seqüências de períodos (por exemplo, professores que preferem dar aula apenas nos primeiros períodos do dia).

Neste trabalho, somente não serão abordados diretamente os itens 6 e 7 do elenco de restrições, uma vez que a solução elaborada está baseada no atendimento das preferências dos professores e não na demanda dos cursos. Os demais serão todos considerados explicitamente na formulação do problema, direcionando o processo de busca e caracterizando o problema como tendo múltiplos objetivos e múltiplas restrições. Dessa maneira, a estratégia de solução proposta realizará uma busca entre os possíveis candidatos à solução ótima (ou aproximadamente ótima), sabendo que a otimização dos objetivos pode levar à violação de restrições.

Segunda-feira	*W*	*P*	*F*	*R*	*A*
Terça-feira	*E*	*S*	*N*	*K*	*L*
Quarta-feira	*D*	*I*	*G*	*X*	*U*
Quinta-feira	*B*	*V*	*T*	*J*	*H*
Sexta-feira	*Y*	*O*	*C*	*M*	*Q*

Figura 1.1: Representação de uma solução possível.

Tabela 1.1: Número de combinações possíveis, conforme o número de professores, para o exemplo dado.

Número de professores	**Número de combinações**
10	$3{,}6288\times10^{6}$
15	$1{,}3077\times10^{12}$
20	$2{,}4329\times10^{18}$
25	$1{,}5511\times10^{25}$
30	$2{,}6525\times10^{32}$

1.2 Aspectos Gerais da Estratégia de Solução

Para viabilizar a aplicação da estratégia de solução proposta neste trabalho, tanto para o caso 1 como para o caso 2, serão desenvolvidos um código genético compacto e um algoritmo de expansão de código, ambos específicos para cada caso.

Além disso, o desempenho da estratégia de solução baseada na aplicação conjunta de computação evolutiva, busca local e otimização baseada em restrições, será comparada à aplicação da computação evolutiva diretamente ao código genético expandido, a qual também vai apresentar etapas de reparação e procedimentos de busca local.

É fundamental que se garanta a produção de um único e mesmo fenótipo para cada representação compacta, justificando a existência de uma única semente para cada indivíduo e da propriedade de repetitividade do gerador de números pseudo-aleatórios (veja Anexo A), a ser empregado no processo de expansão de código. Uma vez que o código genético compacto é sempre um indivíduo factível e que os operadores genéticos sejam aplicados somente a eles, os descendentes gerados também serão factíveis.

A solução empregada caracteriza-se, ainda, por ser salvacionista e parcialmente elitista. Isso significa que o melhor indivíduo de uma dada geração será preservado para a próxima (sobrevivência do elemento mais adaptado), de modo que não ocorra uma degradação de qualidade do melhor indivíduo ao longo das gerações. Acrescenta-se, que uma porcentagem dos melhores elementos de uma geração estará presente na seguinte, fazendo com que haja uma manutenção das boas características da população, sem comprometer de todo o seu nível de diversidade.

O processo de geração de soluções factíveis, pela aplicação do algoritmo de expansão de código, se dá pela implementação de sofisticados mecanismos de reparação, com convergência garantida. É importante ressaltar que o problema estudado tem características próprias que indicam o uso da computação evolutiva juntamente com processos de busca local, conduzindo aos algoritmos meméticos (MOSCATO, 1989). Sendo assim, além da expansão responsável por produzir um código factível, uma busca local é empregada para refinar o código na direção de ótimos locais. Uma vez que as soluções factíveis da geração corrente tenham sido obtidas e refinadas localmente, uma medida da sua qualidade poderá ser obtida e utilizada nas etapas subseqüentes do processo evolutivo.

1.3 Descrição do Conteúdo dos Demais Capítulos

O Capítulo 2 deste trabalho apresenta os aspectos principais relacionados à computação evolutiva, dando ênfase aos algoritmos genéticos. O Capítulo 3 fornece os conceitos básicos associados aos algoritmos meméticos. No Capítulo 4, é apresentada uma revisão teórica de problemas com restrições sob o enfoque da computação evolutiva, de modo a posicionar o trabalho desenvolvido nesta dissertação. Já o Capítulo 5 enfoca o início dos estudos de caso desenvolvidos, mais especificamente a geração automática de turnos completos em torneios. O tratamento apresentado no Capítulo 5 pode ser considerado como sendo uma etapa preliminar à abordagem do problema de alocação de carga didática em instituições de ensino, a ser tratado no Capítulo 6. O Capítulo 7 apresenta comentários conclusivos e aponta as perspectivas futuras da pesquisa. A seguir, o Anexo A apresenta alguns comentários a respeito de geradores de números aleatórios. O Anexo B mostra o questionário que foi elaborado para o obtenção das preferências dos professores no caso de alocação de carga didática. Por último, o Anexo C mostra o cálculo realizado para a definição do número de candidatos factíveis no caso de turnos completos em torneios.