3
Metodologia

Neste capítulo apresentamos a base teórica para entendimento/contextualização das heurísticas propostas. O capítulo se inicia abordando as heurísticas de busca local, onde podem ser encaixadas tanto a busca tabu reativa quanto a HMGC. Uma atenção especial é dada para um tipo bastante utilizado de busca local: as chamadas metaeurísticas, sendo a busca tabu reativa, que é abordada no capítulo 4, um tipo de metaeurística. Em seguida descrevemos a técnica de resolução de problemas de otimização com um grande número de variáveis de decisão (colunas) denominada geração atrasada de colunas ou simplesmente geração de colunas. Esta técnica é um dos principais componentes da nova heurística apresentada no capítulo 5 denominada HMGC, sendo fundamental para o entendimento da mesma. A HMGC é uma combinação de buscas locais com uma estratégia de escape de mínimos locais baseada na geração de colunas.

3.1
Heurísticas de Busca Local

Heurísticas podem ser entendidas como algoritmos que buscam encontrar soluções o mais próximas possível da otimalidade em um tempo computacional razoável. A literatura costuma dividir as heurísticas em duas classes principais: as heurísticas construtivas e as de busca local. As primeiras, como o nome sugere, trabalham construindo iterativamente uma solução a partir de informações da solução parcial e do problema. Já as buscas locais, trabalham melhorando soluções construídas previamente, através da exploração iterativa de soluções chamadas vizinhos. As buscas locais são tipicamente instanciações de vários esquemas gerais de busca, porém com a característica comum de utilização de uma função de vizinhança, [2].

O uso de buscas locais em otimização combinatória data dos anos 50, onde os primeiros algoritmos de trocas de arestas foram utilizados no problema

clássico do caixeiro viajante, como pode ser visto nos trabalhos [11] e [20]. Ao longo do tempo as buscas locais foram aperfeiçoadas e paulatinamente atingindo bons resultados em uma vasta gama de problemas de otimização como design de layout, caixeiro viajante, roteamento de veículos, scheduling, entre outros. O estudo em métodos de busca local é um tema bastante explorado atualmente e sua importância tanto prática quanto teórica já é reconhecida na comunidade acadêmica, [2].

De forma simplificada, o algoritmo de busca local inicia a partir de uma ou um conjunto de soluções iniciais e então tenta continuamente encontrar soluções melhores através da exploração das chamadas soluções vizinhas. Um solução vizinha é obtida a partir de uma solução atual onde é realizado um movimento a partir de um critério vizinhança, que é definido pela função de vizinhança.

Uma versão básica de busca local é a chamada melhoria iterativa. A melhoria iterativa inicia a partir de uma solução inicial e explora todas as soluções vizinhas na busca por uma solução melhor. Se uma solução melhor é encontrada, a solução atual é então substituída por esta, e assim a busca continua. Do contrário, o algoritmo retorna a solução atual, que é denominada mínimo local.

Uma classe especial e mais sofisticada de buscas locais são as chamadas metaeurísticas. Por sua grande importância atualmente na otimização combinatória como ferramenta de resolução de problemas das mais diversas áreas, é feita uma descrição detalhada das mesmas a seguir.

3.1.1

Metaeurísticas

Nos últimos 20 anos, aconteceu o surgimento de um novo tipo de algoritmo aproximado para resolução de problemas de otimização que basicamente tenta combinar métodos heurísticos simples em um "framework" de alto nível na tentativa de explorar de forma eficiente e efetiva o espaço de busca dos problemas. Estes métodos são chamados hoje em dia de metaeurísticas, [38].

Esta classe de algoritmos inclui, mas não se restringe a apenas estes, a Otimização por Colônia de Formigas (Ant Colony Optimization - ACO), a Computação Evolucionária (Evolutionary Computation - EC) da qual fazem parte os Algoritmos Genéticos (Genetic Algorithms - GA), a Busca Local

Iterada (Iterated Local Search - ILS), o Recozimento Simulado (Simulated Annealing - SA) e a Busca Tabu (Tabu Search - TS).

Diversos autores definiram o que é uma metaeurística, entre algumas das definições temos:

"Uma metaeurística é formalmente definida como um processo iterativo que guia uma heurística subordinada combinando de forma inteligente diferentes conceitos para exploração do espaço de busca, utilizando estratégias de aprendizado para estruturar a informação de forma a encontrar eficientemente soluções próximas da otimalidade.", Osman e Laporte, [30].

"Uma metaeurística é um processo iterativo mestre que guia e modifica operações de heurísticas subordinadas para produzir de forma eficiente soluções de alta qualidade. Ela pode manipular uma única solução completa (ou incompleta) ou uma coleção delas a cada iteração. As heurísticas subordinadas podem ser procedimentos de alto (ou baixo) nível, ou uma simples busca local, ou apenas um método construtivo.", Fink e Voss, [12].

Podemos então resumir as principais características das metaeurísticas nos seguintes pontos:

- Metaeurísticas são estratégias que guiam o processo de busca por soluções de problemas de otimização.
- O seu objetivo é explorar de forma eficiente o espaço de busca de forma a encontrar soluções o mais próximo possível da otimalidade.
- Técnicas que constituem as metaeurísticas vão desde simples procedimentos de busca local a complexas estratégias de aprendizado.
- Metaeurísticas são algoritmos aproximados e usualmente não determinísticos.
- Elas podem incorporar mecanismos para evitar que fiquem presas em regiões de confinamento do espaço de busca, os chamados mínimos locais.
- Metaeurísticas não são específicas por problema.
- Metaeurísticas podem fazer o uso de conhecimento específico do domínio do problema na forma de heurísticas que são controladas por uma estratégia de maior nível.
- Metaeurísticas mais avançadas utilizam a experiência da exploração do espaço de busca (embutida de alguma forma na memória) para guiar a busca para regiões mais promissoras.

Em resumo, nós podemos afirmar que metaeurísticas são estratégias de alto nível para explorar espaços de busca utilizando diferentes métodos. Um dos fatores de maior importância para o sucesso de uma metaeurística é o balanceamento entre diversificação e intensificação. O termo diversificação geralmente se refere a exploração diversa do espaço de busca, enquanto o termo intensificação se refere a melhoria contínua, em termos de função objetivo, das soluções acumuladas durante a busca.

O balanceamento mencionado entre diversificação e intensificação é importante, por um lado para identificar rapidamente regiões do espaço de busca com soluções de alta qualidade e por outro para evitar gastar muito tempo em regiões do espaço de busca já exploradas anteriormente ou que não possuam soluções de alta qualidade, [38].

Devido as características mencionadas acima, o uso de metaeurísticas na resolução de problemas de otimização de alta complexidade, particularmente os problemas NP-Difíceis, tem apresentado ótimos resultados. Algoritmos Genéticos, Busca Tabu, Recozimento Simulado entre outras metaeurísticas são frequentemente utilizadas na resolução de problemas de otimização das mais variadas áreas de pesquisa, em [38] é apresentada uma visão geral da área de pesquisa em metaeurísticas. Assim, as metaeurísticas se tornaram uma importante alternativa para resolução de problemas de otimização, motivando dessa forma seu uso no presente trabalho.

3.2 Geração de Colunas

A solução de problemas de otimização de forma exata, ou seja, onde existe a garantia de encontrar uma solução que é ótima global, geralmente faz uso de formulações matemáticas de programação linear ou de programação linear inteira para o problema em questão, estas formulações servem de base para outras técnicas como o branch-and-bound, relaxação lagrangeana, otimização de subgradiente, entre outras. Um componente essencial destas técnicas é a resolução de formulações de programação linear de forma ótima. O algoritmo mais utilizado para resolução de programas lineares é o algoritmo Simplex e suas variações. Uma descrição detalhada do Simplex pode ser encontrada em [4].

Em muitos casos na prática os programas lineares possuem muitas variáveis e restrições, o que exige muito tempo computacional para resolução

dos mesmos utilizando técnicas convencionais como o Simplex. Um detalhe que pode ser percebido nestes casos é que a vasta maioria da matriz de restrições, que é a matriz composta pelos coeficientes das variáveis em cada restrição, é irrelevante, [39]. Para resolução destes problemas precisamos então apenas das variáveis e restrições que são relevantes, estas podem ser caracterizadas da seguinte forma (levando em consideração sua resolução pelo Simplex):

- Restrições que estão ativas na otimalidade, ou seja, aquelas que podem ter variáveis duais positivas, [4].
- Variáveis que são básicas na otimalidade, ou seja, aquelas que podem assumir valores positivos, [4].

O problema é que não sabemos com antecedência quais são estas variáveis e restrições. Uma forma de contornar este problema é utilizar a técnica de geração atrasada de colunas (ou simplesmente geração de colunas), para o caso do programa linear possuir muitas variáveis, e a geração de cortes, para o caso do programa linear possuir muitas restrições. A seguir descrevemos de forma simplificada a técnica de geração de colunas, que é a técnica utilizada na HMGC.

Em problemas com um grande número de variáveis (colunas) existem dois problemas principais na resolução dos mesmos utilizando o Simplex:

- O tempo para gerar toda a matriz de restrições, [39].
- O tempo para o cálculo dos custos reduzidos das variáveis que não fazem parte da base, [39].

Como mencionado anteriormente, uma solução para estes problemas é a utilização da técnica de geração atrasada (ou implícita) de colunas. A técnica consiste em trabalhar apenas com um subconjunto relevante de colunas e iterativamente resolver o programa linear com este subconjunto, adicionar eventuais colunas que possam vir a melhorar a solução atualmente obtida (as colunas que possuam custo reduzido negativo), até que em uma determinada iteração não existam mais colunas com custo reduzido negativo. A técnica, portanto, consiste dos seguintes passos:

1. Inicia-se o programa linear apenas com algumas colunas promissoras.
2. Resolve-se o programa linear apenas com estas colunas.

3. Calcula-se o custo reduzido das colunas restantes e inclui aquelas de custo reduzido negativo no programa linear.

4. Itera se alguma coluna foi incluída no programa linear.

Ao final da geração de colunas, temos a garantia que a solução encontrada é ótima, já que ao final não existirá nenhuma coluna de custo reduzido negativo. Uma observação importante é que precisamos apenas gerar a coluna de menor custo reduzido a cada iteração, já que ao final do método teremos a mesma garantia de que nenhuma coluna terá custo reduzido negativo. Achar a coluna de menor custo reduzido equivale a resolver outro problema de otimização. Este problema é chamado de problema de *pricing*, o mesmo é descrito a seguir, [39].

Considere o problema restrito que considera apenas um subconjunto I de colunas, onde c_i representa os custos das variáveis de decisão x_i e A_i representa a matriz de restrições:

$$\min \sum_{i \in I} c_i x_i \tag{3-1}$$

sujeito a

$$\sum_{i \in I} A_i x_i = b \tag{3-2}$$

$$x \geq 0 \tag{3-3}$$

Após resolver o problema restrito em uma iteração da geração de colunas calculamos então o valor das variáveis duais ($c_B^T B^{-1}$, [4]). Em seguida devemos gerar uma nova coluna que possua custo reduzido negativo. Isso pode ser feito resolvendo o seguinte subproblema da geração de colunas (problema de *pricing*), onde a representa uma coluna válida pertencente ao conjunto de todas as colunas possíveis C, c_a representa o custo de a, c_B^T representa o vetor transposto dos custos das variáveis da base e B^{-1} representa a inversa da matriz básica, matriz formada pelos coeficientes das variáveis básicas nas restrições, [39]:

$$\min_{a \in C} c_a - c_B^T B^{-1} a \tag{3-4}$$

O sucesso da geração de colunas depende da resolução de forma eficiente do problema de *pricing*. A geração de colunas é principalmente aplicada quando

o problema de *pricing* se configura um problema que pode ser resolvido de forma ótima por algoritmos polinomiais ou pseudo-polinomiais. Em alguns casos o próprio problema de *pricing* se encaixa na classe de problemas NP-Difíceis. Neste caso a resolução do mesmo de forma ótima em tempo reduzido pode ser difícil. Uma alternativa neste caso é resolvê-lo utilizando heurísticas, onde não há a garantia de otimalidade para resolução do mesmo. Neste caso a solução a ser gerada pela geração de colunas também não terá a garantia de otimalidade, já que não se pode garantir a inexistência de colunas de custo reduzido negativo, portanto a solução final gerada pela técnica será uma solução heurística.