RAFAEL LOUBACK FERRAZ

Orientador: Marco Antonio Moreira de Carvalho

UM ESTUDO SOBRE ALGORITMOS GENÉTICOS DE CHAVES ALEATÓRIAS VICIADAS

Ouro Preto

Fevereiro de 2016

Universidade Federal de Ouro Preto
Instituto de Ciências Exatas
Bacharelado em Ciência da Computação

UM ESTUDO SOBRE ALGORITMOS GENÉTICOS DE CHAVES ALEATÓRIAS VICIADAS

Monografia apresentada ao Curso de Bacharelado em Ciência da Computação da Universidade Federal de Ouro Preto como requisito parcial para a obtenção do grau de Bacharel em Ciência da Computação.

RAFAEL LOUBACK FERRAZ

Ouro Preto
Fevereiro de 2016

UNIVERSIDADE FEDERAL DE OURO PRETO

FOLHA DE APROVAÇÃO

Um Estudo Sobre Algoritmos Genéticos de Chaves Aleatórias Viciadas

RAFAEL LOUBACK FERRAZ

Monografia defendida e aprovada pela banca examinadora constituída por:

Dr. Marco Antonio Moreira de Carvalho – Orientador
Universidade Federal de Ouro Preto

Msc. Marcelo Luiz Silva
Universidade Federal de Ouro Preto

Msc. Reinaldo Silva Fortes
Universidade Federal de Ouro Preto

Ouro Preto, Fevereiro de 2016

Resumo

Na ciência da computação existem uma série de problemas de grande interesse prático no dia a dia da sociedade moderna, classificados como problemas de otimização combinatória. Grande parte destes problemas possuem difícil solução em um tempo aceitável. Para alguns, sequer se conhece um método eficiente de solução. Daí surge um campo fértil para aplicação de métodos heurísticos e metaheurísticos, que têm se mostrado cada vez mais importantes para a resolução destes problemas. Neste trabalho será estudado o método metaheurístico *Algoritmo Genético com Chaves Aleatórias Viciadas*, um método recente que demonstrou ser eficiente em diversos problemas de complexa solução, principalmente quando o tamanho das instâncias do problema aumenta consideravelmente. Este método utiliza técnicas inspiradas na biologia evolutiva e pertencente a classe dos algoritmos genéticos, evoluindo soluções de acordo com restrições durante um conjunto finito de gerações, sendo diferenciado principalmente por ter chaves viciadas e ter maior chance de passar as características genéticas das melhores soluções dentre as gerações. Estes conceitos são apresentados em detalhes, bem como uma interface de programação de aplicações relacionada ao método.

Abstract

In computer science there is a series of problems that modern society has great pratical interests to solve daily problems, classified as combinatorial optimization problems. Great part of these problems has difficult solution in an acceptable time frame. For some, is not known if there is, or could exist, some efficient method to solve the problem. Because of that, emerges an entire new field of heuristics and meta-heristics to solve these problems, which had showed to be a very efficient finding good solutions for some of these problems. In this work it will be studied the *Biased Random-Key Genetic Algorithm* (BRKGA) metaheuristic, an recent method that has demonstrated to be efficient in a variety of difficult solution problems, mainly when instance size increases. This method make use of evolutionary biology inpired techniques, and is classified as a genetic algorithm, evolving solutions following some problem restrictions until an finite number of generations are developed, been differentiated mainly for the use of biased random keys and provide more chance of an offspring solution inherit an genetical material from the best solution father between the generations. These concepts will be presented in details, as well the application programming interface related to the method.

Dedico este trabalho a Deus, meus pais Cilas e Leila, minha irmã Ester e minha querida companheira Marise. Pessoas que são de suma importância em minha vida.

Agradecimentos

Quero agradecer ao meu orientador Marco e a todo corpo docente da UFOP, que me instruiu durante esses anos, no meu desenvolvimento acadêmico.

Sumário

Lista de Figuras

Lista de Tabelas

Lista de Acrônimos

API *Application Programming Interface*

GRASP *Greedy Randomized Adaptive Search Procedure*

RKGA *Random-Key Genetic Algorithm*

BRKGA *Biased Random-Key Genetic Algorithm*

Capítulo 1

Introdução

Na sua definição original, métodos *Metaheurísticos* são métodos que coordenam métodos para a busca de soluções ótimas locais e estratégias de alto nível para fugir desses ótimos locais. Dessa maneira, é maior a chance de se encontrar uma solução ótima global ou mais próxima dela, sem ter a necessidade de explorar todo o espaço de soluções de um problema (Gendreau e Potvin, 2010). Alguns exemplos de metaheurísticas mais utilizadas são: *Greedy Randomized Adaptive Search Procedure* (GRASP), Busca Tabu, Busca Local Iterada e Busca de Vizinhança Variável.

Algoritmos Evolutivos são uma categoria de metaheurísticas inspiradas no conceito da Teoria da Seleção Natural proposta por Darwin (1859). Nesses algoritmos, existe uma série de indivíduos pertencentes a uma população que, devido a uma pressão causada pelo ambiente, somente os mais adaptados participarão das próximas gerações desta população, gerando uma melhor adaptação da população em geral ao ambiente específico (Eiben e Smith, 2003). Esses algoritmos, geralmente, são organizados tal que cada indivíduo da população é uma possível solução válida, e a pressão que o ambiente causaria para estimular a escolha dos mais aptos, usualmente, é uma função objetivo que mede a qualidade dos indivíduos, indicando se ele é mais apto ou não para maximizar o valor da solução de um problema. Comumente também são utilizados mecanismos para criar diversidade nas características dos indivíduos, como, por exemplo, mutação das características de um indivíduo e seleção de alguns indivíduos que não são os melhores de uma geração para reprodução. Com isso, facilita-se escapar de soluções ótimas locais e gera-se maior chance de se aproximar de soluções ótimas globais.

Dentre os algoritmos evolutivos, alguns dos mais utilizados são os *Algoritmos Genéticos*. O método foi proposto originalmente por Holland (1975), e teve como diferencial ser uma técnica que pretendia ser mais genérica ao invés de focar em resolver somente um problema específico, (Mitchell, 1998). O método original proposto consiste em estabelecer uma população inicial de "cromossomos" ou indivíduos (e.g., cadeias de valores binários), tal que cada cromossomo é composto por "genes" (e.g., 0 ou 1). A cada geração, um grupo dos cromossomos é escolhido para se reproduzir e frequentemente os mais aptos (os de melhor função objetivo)

produzem mais descendentes do que os demais. A recombinação (*crossover*) dos genes desses cromossomos é feita imitando o conceito biológico, copiando subpartes (genes) de cada um dos dois cromossomos envolvidos na geração de um descendente. Mutação também é utilizada para gerar diversidade dentre as novas gerações, aleatoriamente mudando valores dos genes pertencentes a um cromossomo.

Uma variante dos algoritmos genéticos é o *Random-Key Genetic Algorithm* (RKGA) ou Algoritmo Genético com Chaves Aleatórias, proposto por Bean (1994). Seu maior diferencial é utilizar um sistema de chaves aleatórias para codificar as soluções (indivíduos ou cromossomos), onde cada cromossomo é composto por uma sequência de chaves aleatórias, tipicamente no intervalo $[0, 1)$. Essas soluções com chaves aleatórias podem ser decodificadas para o espaço de soluções do problema específico a ser resolvido, possibilitando que o RKGA trate diretamente sobre as chaves e não soluções específicas do problema, criando maior desacoplamento do algoritimo em relação ao problema específico a ser resolvido. Este método também evita o problema de geração de descendentes inviáveis, já que ele representa codificações dos cromossomos de um modo indireto. Essas codificações são avaliadas para sempre gerar novos cromossomos que satisfazem todas as restrições do problema.

Além do uso de chaves aleatórias, o RKGA também utiliza o conceito de *elitismo* para a seleção de quais cromossomos são os "melhores" de uma geração. Por exemplo, são separados em torno de 20% dos cromossomos da atual geração, para serem copiados para a próxima geração, assegurando que as melhores soluções sejam levadas adiante nas novas gerações. Outros 79% são gerados escolhendo-se pares de indivíduos aleatoriamente na geração atual recombinando os seus genes para produzir as novas gerações. A recombinação utiliza *Parametrized Uniform Crossover* (Spears e Jong, 1991), tal que para cada gene é escolhido aleatoriamente de cada cromossomo pai para ser passado ao novo indivíduo. O 1% restante é gerado adicionando-se cromossomos criados de modo aleatório, *mutantes*, à nova geração, criando diversidade e evitando convergência prematura dos indivíduos.

Inspirado no RKGA, foi proposto também o BRKGA ou Algoritmo Genético com Chaves Aleatórias Viciadas. Proposto por Gonçalves e Resende (2010), ele tem operação semelhante ao predecessor, porém ambos diferem em como os pares são escolhidos para reprodução e como a recombinação é aplicada. No BRKGA, a escolha dos cromossomos para a recombinação é aleatória como no RKGA, mas ela precisa que um dos escolhidos seja pertencente à elite das soluções, já o outro cromossomo escolhido pode ser de qualquer parte da população. Outro detalhe é que na fase de recombinação do par de indivíduos, o gene provindo do indivíduo da elite deve ter sempre maior chance de ser escolhido sobre o gene do segundo indivíduo.

Para facilitar a utilização do BRKGA, foi desenvolvida por Toso e Resende (2014), uma *Application Programming Interface* (API) ou biblioteca para a linguagem C++, que implementa a maior parte das rotinas necessárias para a utilização do BRKGA. Deixando a cargo do utilizador da API somente implementar a parte específica do problema tratado e ajustar

os parâmetros para que ela execute.

1.1 Justificativa

O BRKGA é uma metaheurística recente, mas que vem demonstrando ser eficiente em determinar boas soluções para problemas de otimização em diversas áreas, incluindo:

- Telecomunicações;
- Transportes;
- Leilões combinatórios;
- Problemas de empacotamento;
- Escalonamento de tarefas;
- Engenharia de Produção.

Comparações feitas por Gonçalves et al. (2012) demonstraram que, no geral, o BRKGA converge mais rapidamente em direção à solução ótima desejada do que o RKGA. Evidenciando ser uma metaheurística mais eficiente, o BRKGA pode ser empregado em problemas que precisem de soluções rápidas ou em tempo real.

Adicionalmente, há pouco, uma API do BRKGA foi proposta por Toso e Resende (2014) para uso geral, porém, ainda é pouco disseminada na comunidade acadêmica apesar de seu grande potencial.

1.2 Objetivos

De uma maneira geral, este trabalho consiste em realizar pesquisa para geração de embasamento teórico para compreensão dos conceitos relacionados a algoritmos genéticos, algoritmos genéticos de chaves aleatórias e algoritmos genéticos de chaves aleatórias viciadas, bem como o funcionamento da API brkgaAPI.

Além dos objetivos principais, temos os seguintes objetivos específicos:

- Compreender os conceitos relacionados a algoritmos genéticos, algoritmos genéticos de chaves aleatórias e algoritmos genéticos de chaves aleatórias viciadas;
- Compreender o funcionamento da API brkgaAPI;
- Aplicar a brkgaAPI a um problema combinatório.

Outros produtos deste projeto de pesquisa serão trabalhos publicados em eventos nacionais, os quais contribuem para a promoção da Universidade Federal de Ouro Preto e também do tema tratado.

1.3 Organização do Trabalho

O restante deste texto encontra-se organizado da seguinte forma:

- No Capítulo 2 apresenta a revisão bibliográfica acerca das aplicações do BRKGA;
- No Capítulo 3 encontra-se a fundamentação teórica sobre os Algoritmos Genéticos e suas variações com foco no BRKGA e sua API;
- No Capítulo 4 é descrito o plano de atividades restantes, apresentando o planejamento do que deve ser feito futuramente;
- No Capítulo 5 apresenta-se as conclusões preliminares.

Capítulo 2

Revisão da Literatura

A metaheurística BRKGA tem sido aplicada a inúmeros problemas de otimização. Neste capítulo, os trabalhos publicados anteriormente são descritos brevemente agrupados por área de aplicação. Para maiores informações sobre todas as aplicações do método, pode-se consultar o exame da literatura de autoria de Prasetyo et al. (2015). Extensões do conceitos original do BRKGA podem ser encontradas no trabalho de Lucena et al. (2014).

2.1 Telecomunicações

Os autores Goulart et al. (2011a) utilizaram o BRKGA para minimizar os custos de instalação de uma rede de fibra óptica, descobrindo a configuração das rotas que minimizem a quantidade de dispositivos ópticos utilizados na rede.

Noronha et al. (2011) propuseram o uso do BRKGA para resolver o problema de roteamento e atribuição de comprimento de onda para redes de fibra óptica. Eles mesclaram o BRKGA juntamente com métodos conhecidos na literatura e obtiveram resultados superiores aos que até então eram considerados estado da arte.

Como demonstrado por Reis et al. (2011), o uso do BRKGA juntamente com os protocolos *open shortest path first* e *distributed exponentially-weighted flow splitting*, para resolver o problema de atribuição de pesos para roteamento em redes, tem resultados favoráveis. Tal que o *distributed exponentially-weighted flow splitting* gera menos congestionamento na rede que o *open shortest path first*, mas acarreta maior atraso no tempo de entrega dos pacotes na rede.

Os autores Ruiz et al. (2011) comparam os métodos de roteamento de redes *internet protocol/multi-protocol label switching* e *wavelength Switched Optical Network* de modo a minimizar o investimento em bens de capital. Um algoritmo, baseado no BRKGA, também é proposto visando minimizar os custos e os dois métodos são extensivamente testados.

Pedrola et al. (2013b) confrontam os métodos *internet protocol/multi-protocol label switching* e *wavelength Switched Optical Network* em uma rede multicamada, visando maximizar

a disponibilidade de serviços na rede enquanto matem o menor custo possível. Primeiramente eles propuseram um método que usa programação inteira linear para prover uma melhor noção da complexidade do problema. Depois foi proposto um método GRASP com *path relinking* e também um outro utilizando BRKGA para ajudar a resolver o problema. Experimentos numéricos executados pelos autores demonstraram que em redes altamente complexas o método GRASP tem eficiência superior ao BRKGA. Além de averiguar que o uso do GRASP com *path relinking* melhora consideravelmente a eficiência do algoritmo comparado com o GRASP original.

Foi desenvolvido por Goulart et al. (2011b) um algoritmo baseado no BRKGA que busca minimizar os custos dos equipamentos necessários para operar uma rede de fibra óptica, achando o melhor roteamento para essa rede. O algoritmo consiste em utilizar o algoritmo estado da arte conhecido na literatura em conjunto com o BRKGA. Resultados computacionais demonstraram que este novo método tem resultados melhores que o uso do algoritmo que até então era conhecido como estado da arte.

Morán-Mirabal et al. (2013c) contrastam o uso de metaheurísticas, como GRASP com *path relinking*, GRASP com *evolutionary path relinking* e o BRKGA para minimizar o *handover problem* em redes móveis. Este problema consiste em minimizar as quedas de conexão que um dispositivo móvel possa sofrer quando for transferido de uma torre de transmissão para outra (*handover*). Os autores demonstraram que o método estado da arte encontrado na literatura, até então, só tinha capacidade de resolver instâncias pequenas do problema. Por isso compararam estes três métodos e notaram que todos eles conseguiram encontrar soluções ótimas para instâncias maiores do problema, mas que o GRASP com *evolutionary path relinking* se demonstrou superior aos demais na maioria dos casos.

Os autores Pedrola et al. (2013a) paragonaram o uso do GRASP e BRKGA, com variações que hibridavam com técnicas de *path relinking* e *variable neighborhood descent* para intensificar os resultados. Dessa forma, visando minimizar os custos de implantação e de operação de retransmissores elétricos em redes de fibra óptica. Resultados demonstram que o uso do BRKGA juntamente com as técnicas de intensificação é viável e mais eficiente do que outras técnicas já conhecidas na literatura.

Duarte et al. (2014) propõem implementações mais eficientes de heurísticas estabelecidas e o uso dessas heurísticas em conjunto com GRASP e BRKGA, para tratar o problema de minimizar os custos de implantação de retransmissores em redes de fibra óptica. Os resultados demonstraram que o uso do GRASP *multi-start* frequentemente encontra soluções de qualidade superior ao uso do GRASP *single-start*. GRASP demonstrou gerar soluções de melhor qualidade que o BRKGA. O BRKGA só gera soluções de melhor qualidade se o tempo de execução do algoritmo for superior ao GRASP (maior número de gerações). Ambos os métodos se mostraram mais eficientes que o estado da arte até então.

O BRKGA é utilizado por Andrade et al. (2013), para resolver o problema do k-

interconnected multi-depot multi-traveling salesmen, problema similar ao do caixeiro viajante, mas que trata vários vendedores e inclui múltiplos depósitos que adicionam uma dificuldade extra, pois o número de depósitos não é fixo. Os autores compararam o BRKGA com o algoritmo *multi-start heuristic*, que também utiliza busca local. O BRKGA frequentemente apresentou-se levemente mais eficiente que o outro comparado.

Existe também uma pesquisa de autoria de Resende (2011), sobre as aplicações do BRKGA em telecomunicações. Primeiramente ele introduz os conceitos básicos do algoritmo, seguindo de uma descrição de metaheurísticas baseadas no BRKGA para roteamento em redes IP, projeto de redes IP com tolerância a falhas, localização de servidores redundantes na distribuição de conteúdo, localização de retransmissores em redes de fibra óptica, e roteamento e alocação de comprimento de ondas em redes de fibra óptica. O autor afirma que para cada problema tratado o BRKGA demonstrou ser uma metaheurística eficiente, frequentemente encontrando soluções de melhor qualidade que de outros métodos conhecidos. Além de que, em alguns casos, ele encontrou soluções de qualidade semelhante, mas em menor tempo.

2.2 Transportes

Para o problema de congestionamento de trânsito em rodovias com cobrança de tarifa em pedágios, Buriol et al. (2010) e Stefanello et al. (2015) utilizaram métodos inspirados no BRKGA para encontrar soluções para grandes instâncias do problema, tal que sejam definidos o local de cobranças na rodovia e também qual valor deve ser atribuído para cobrança com finalidade de reduzir o congestionamento. Os resultados foram favoráveis, encontrando soluções boas, mas nem sempre ótimas para instâncias grandes do problema que não eram possíveis de serem resolvidas com outros métodos.

Grasas et al. (2014) trataram o problema de coleta de amostras de sangue para o transporte até o laboratório de análise utilizando o BRKGA. O método se mostrou eficaz nos casos estudados pelos autores, reduzindo o número de viagens necessárias para as coletas.

2.3 Escalonamento

No problema de escalonamento de projetos com restrições de recursos, os autores Gonçalves et al. (2010) utilizaram uma variante do BRKGA que utiliza o conhecido *forward-backward improvement* para melhorar os resultados obtidos inicialmente pela metaheurística. O algoritmo foi extensivamente testado e os resultados foram no geral superiores ou similares a outros algoritmos conhecidos.

Tangpattanakul et al. (2012) utilizaram o BRKGA no problema de otimização multiobjetivo de seleção e escalonamento de observação por satélites ágeis de observação terrestre. Eles focaram em maximizar o lucro total da operação dos satélites, mas ao mesmo tempo mantendo

a disponibilidade para todos os usuários da rede, não deixando os usuários que geram menor lucro sem disponibilidade do serviço.

O BRKGA foi empregado no *job-shop scheduling problem* ou problema do escalonamento de processos por Gonçalves e Resende (2011). No geral o método proposto se demonstrou eficaz quando submetido a testes extensivos.

2.4 Empacotamento

Gonçalves e Resende (2012) aplicaram o BRKGA com múltiplas populações no problema de carregamento de um contêiner, usando pacotes heterogêneos e homogêneos. Os resultados validaram que o método proposto gera soluções de alta qualidade em um tempo aceitável.

No problema de empacotamento 2D e 3D, Gonçalves e Resende (2013) utilizam um algoritmo baseado no BRKGA para encontrar a melhor posição que cada pacote deve tomar dentro da caixa para melhor aproveitar o espaço dela. Os resultados demonstraram que frequentemente o método proposto demora menos tempo para chegar a uma solução, além que no geral, as soluções são de melhor qualidade que de outros métodos.

2.5 Recobrimento

Resende et al. (2011) utilizaram um método baseado no BRKGA paralelo com múltiplas populações para resolver o *Steiner triple covering problem*, um problema de cobertura em árvores de alta dificuldade computacional. Para as instâncias dos problemas testados, o algoritmo proposto encontrou as soluções ótimas em todos os casos onde se conhecia a solução ótima. Em inúmeras outras, conseguiu melhorar a solução conhecida até então. O paralelismo utilizado no algoritmo demonstrou amplo aumento na velocidade de execução.

2.6 Otimização em Redes

Os autores Coco et al. (2012) trataram o problema do caminho mais curto robusto utilizando o BRKGA. Os resultados demonstraram que o algoritmo proposto consegue resolver problemas com até mil vértices e demonstrou ser eficiente.

Novamente, uma variante do BRKGA com múltiplas populações foi utilizada por Fontes e Gonçalves (2012) para resolver o *hop-constrained minimum cost flow spanning tree problem*. Os resultados apresentados pelos autores demonstram que o algoritmo proposto é com frequência capaz de encontrar soluções próximas do ótimo global, além de ser executado em pouco tempo, mesmo que para grandes instâncias do problema.

2.7 Engenharia de Produção

Moreira et al. (2012) utilizaram o BRKGA para resolver o *assembly line worker assignment and balancing problem* ou problema de alocação e balanceamento de trabalhadores para linha de montagem. O método demonstrou ser tão eficaz quanto os métodos propostos na literatura.

No problema do *family traveling salesperson problem*, que é uma variante do conhecido problema do caixeiro viajante, onde o caixeiro precisa visitar um subconjunto dos vértices pertencentes ao grafo para cada um processar tipos diferentes de itens, Morán-Mirabal et al. (2013b) utilizaram o BRKGA e o GRASP com *evolutionary path-relinking* para resolver o problema. A análise feita pelos autores demonstrou que o BRKGA tende a levar menos tempo para instâncias menores, o GRASP se demonstrou mais eficiente para instâncias maiores, e ambos são enormemente mais eficientes que o método estado da arte.

2.8 Ajuste Automático de Parâmetros em Metaheurísticas

Os autores Festa et al. (2010) utilizaram o BRKGA para definir automaticamente os parâmetros utilizados para executar o GRASP com *path-relinking* para resolver o problema de atribuição quadrática generalizada. Em uma primeira fase, o BRKGA é executado para definir quais os parâmetros geram melhores resultados para a execução do GRASP e, em uma segunda fase, o GRASP é realmente executado. Resultados avaliados pelos autores demonstraram que a solução é viável e robusta.

Morán-Mirabal et al. (2013a) também utilizaram o BRKGA para definir automaticamente os parâmetros para o GRASP com *evolutionary path-relinking* para resolver três problemas: cobertura de vértices, corte máximo e partição de vértices em grafos capacitados. O resultados demonstrados pelos autores mostram que o GRASP com definição automática de parâmetros pelo BRKGA, geralmente, são executados com menos tempo e geram soluções de qualidade superior que o GRASP com definição de parâmetros manuais.

2.9 Leilões Combinatórios

Variações do BRKGA foram utilizadas por de Andrade et al. (2015) para resolver o problema *winner determination problem* ou problema de determinação do vencedor em leilões. Os autores propuseram seis variações do BRKGA e um algoritmo relaxado de inicialização da população para o BRKGA baseado em programação linear inteira. Os resultados publicados demonstram que o BRKGA nem sempre encontra a melhor solução, mas algo bem próximo dela. Porém, este método leva muito menos tempo para ser executado que o estado da arte comparado pelos autores, principalmente, quando comparado em instâncias maiores.

2.10 Otimização Global Contínua

Os autores Silva et al. (2012) utilizaram o BRKGA para resolver problemas de otimização contínua global com restrições ou *global optimization problems subject to box constraints*. Foi tratado especificamente o problema de cinemática robótica. O algoritmo foi executado múltiplas vezes, tal que, em cada execução, ele evitava o espaço de solução que foi encontrado como melhor solução nas execuções anteriores. Silva et al. (2014) também demonstra este processo de procurar diversas áreas diferentes no espaço de soluções de sistemas não lineares. Os resultados ilustram que o BRKGA é um opção viável para resolver problemas de otimização global.

Silva et al. (2015) desenvolveram uma API distinta para as linguagens Python e C++ que agiliza o desenvolvimento de soluções para problemas de otimização global com restrições utilizando o BRKGA. Os autores descrevem como o método funciona e como se pode compilar, instalar e usar sua API.

Capítulo 3

Fundamentação Teórica

Neste capitulo será apresentada a fundamentação teórica necessária para o entendimento dos conceitos básicos relacionados a otimização e os conceitos relacionados aos algoritmos genéticos e o BRKGA.

3.1 Conceitos Introdutórios

Problemas de Otimização consistem basicamente em se determinar a "melhor" solução entre todas as possíveis soluções válidas com o intuito de alcançar um objetivo, como descrito por Papadimitriou e Steiglitz (1982). Uma *solução* para estes problemas é uma configuração de valores para as variáveis de decisão, dentre as existentes no espaço de soluções específico à um determinado problema. As *variáveis de decisão* são as incógnitas a serem determinadas para solução de um problema, por exemplo, podem indicar quantidades, pertinência ou ordenação de elementos. O *espaço de soluções* é o conjunto de todos os valores ou configurações que as variáveis de decisão do problema possam assumir, sendo composto de um número finito (pequeno ou grande) ou infinito de elementos, dependendo do problema.

A melhor solução de um problema de otimização, ou *solução ótima global*, é a solução que possui o melhor valor, de acordo com uma função objetivo, dentre todas as possíveis no espaço de soluções. A *função objetivo*, por sua vez, é a função matemática que relaciona os valores atribuídos às variáveis de decisão de uma determinada solução indicando a sua qualidade, maior valor no caso de uma função maximização e menor valor no caso de uma função de minimização. Essa função objetivo deve ser otimizada, respeitando as devidas *restrições* que o problema possa impor, em outras palavras, as condições que uma solução deve obedecer para ser *viável*.

Em contraposição à solução ótima global, tem-se as *soluções ótimas locais* ou *subótimas*. Em uma região específica do espaço de soluções, pode haver uma solução dentre toda uma vizinhança de soluções tal que não se pode achar uma outra melhor do que esta. Porém, nem sempre esta solução será a melhor existente em todo espaço de soluções e, sim, somente

a melhor em sua vizinhança. Uma *vizinhança*, por sua vez, pode ser definida como um conjunto de soluções para um problema, tal qual é possível relacioná-las dada uma função de proximidade e, com isso, encontrar soluções similares e definir regiões onde estas vizinhanças se encontram no espaço de soluções.

Como exemplo, pode-se ilustrar os conceitos definidos anteriormente com uma instância do *Problema da Mochila Binária*, no qual tem-se uma mochila com uma capacidade máxima de peso suportado e um conjunto de itens com um valor e um peso específicos associados. Deseja-se descobrir a melhor seleção destes itens tal que o maior valor total é atingido, respeitando a capacidade máxima da mochila. Na versão binária deste problema, tem-se a característica de que somente uma unidade de cada item pode ser selecionada para ser colocada na mochila e os itens devem ser escolhidos por inteiro, ou seja, deve ser colocado ou não na mochila.

Neste problema, tem-se as quatro seguintes características:

1. Dados do Problema

   n = Número de itens disponíveis a seleção;

   i = índice para os itens disponíveis, $i = 1, 2, ..., n$;

   v_i = Valor de um item i;

   w_i = Peso de um item i;

   W = Peso máximo suportado pela mochila.

2. Variáveis de Decisão

   $$x_i = \begin{cases} 1, & \text{Se o item } i \text{ foi selecionado para ser colocado na mochila;} \\ 0, & \text{Caso contrário.} \end{cases}$$

3. Função Objetivo

   $$\text{Maximize} \sum_{i=1}^{n} x_i v_i$$

   Na função objetivo descrita acima, busca-se o maior valor somado dos itens colocados na mochila, onde se tem um somatório das variáveis x_i multiplicadas pelas variáveis v_i.

4. Restrições

   $$\sum_{i=1}^{n} x_i w_i \leq W$$

   A função objetivo do problema deve obedecer às restrições impostas pelo problema, caso contrário, a maximização da função objetivo seria simplesmente colocar todos os itens na mochila. No problema da mochila binária, tem-se como restrição que a soma dos

pesos dos itens colocados na mochila não deve ultrapassar o peso máximo suportado por ela.

No exemplo a seguir, uma instância do problema da mochila binária é composta por quatro itens e uma mochila com capacidade máxima de 10 unidades de peso. A Tabela 3.1 apresenta os itens e seus respectivos valores e pesos.

Tabela 3.1: Itens disponíveis para seleção na mochila

Item	Peso	Valor
1	6	30
2	3	14
3	4	16
4	2	9

A Figura 3.1 apresenta um gráfico que demonstra a evolução do valor que a mochila pode ter para cada solução possível do problema. As soluções no gráfico são representadas como um vetor de valores binários, onde a primeira posição no vetor representa o valor de x_1 e assim por diante até x_n, tal que $n = 4$. Neste gráfico, é possível verificar a relação das possíveis soluções e seus respectivos valores para a função objetivo. Existem dois pontos de ótimo local (amarelo) e um ponto de ótimo global (vermelho), e em cada uma destas soluções apresentadas nota-se que as soluções vizinhas possuem valores piores para a função objetivo.

A existência de diferentes ótimos locais motiva a utilização de métodos que não se prendam a ótimos locais e possam diversificar sua busca em outras vizinhanças se for necessário.

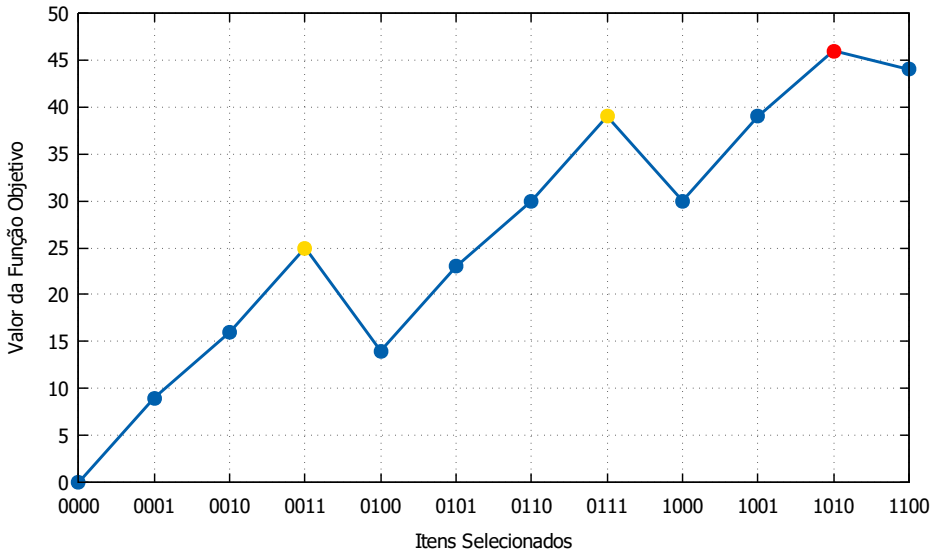

Figura 3.1: Gráfico da mochila binária: os pontos de ótimo local estão marcados de amarelo e o global de vermelho.

3.2 Algoritmos Genéticos de Chaves Aleatórias Viciadas

Como citado anteriormente, o BRKGA ou algoritmos genéticos de chaves aleatórias viciadas é uma metaheurística evolutiva para problemas de otimização combinatória que se baseia na teoria da evolução de Darwin (1859) para aprimorar uma população de indivíduos até encontrar uma solução ótima ou subótima. Características específicas do BRKGA são a representação de seus indivíduos como vetores de chaves aleatórias e maior probabilidade de passar as características genéticas dos indivíduos pertencentes à elite para as novas gerações.

Este método é composto por cinco etapas. São elas:

1. Geração de chaves aleatórias: inicialização da população, etapa de criação dos vetores de chaves aleatórias;
2. Decodificação de vetores de chaves aleatórias: etapa que codifica os vetores do espaço de chaves aleatórias para o espaço de soluções reais do problema;
3. Classificação da população em elite e não elite: após a geração de uma nova população, ela é ordenada e dividida em dois conjuntos de indivíduos (elite e não elite), de acordo com sua adaptação. A elite é então selecionada para ser copiada para a nova geração sem alterações;
4. Mutação ou geração de mutantes: uma parte da nova geração é formada por indivíduos gerados aleatoriamente. Nesta etapa alguns mutantes são inseridos na nova geração;
5. Cruzamento ou recombinação de indivíduos: o restante dos indivíduos é gerado selecionando aleatoriamente um indivíduo da elite e outro não pertencente à elite, para recombinar os genes dos dois indivíduos, porém, sempre tendendo a escolher um gene do indivíduo pertencente à elite.

Estas etapas são executadas repetidamente até que uma determinada condição de parada seja satisfeita, tal como número máximo de gerações ou convergência da solução para um valor limite. O diagrama apresentado pela Figura 3.2 apresenta o fluxo de execução do BRKGA. Nas seções, a seguir, cada uma destas etapas serão detalhadas e exemplificadas.

3.2.1 Geração de Chaves Aleatórias

Nesta etapa é inicializada a população a ser evoluída pela metaheurística. Esta população é formada por p vetores de n chaves aleatórias, tal que cada vetor representa um indivíduo ou cromossomo. Cada um dos n genes pertencente ao cromossomo é gerado aleatoriamente por uma chave que pode assumir valores no intervalo de $[0, 1] \in \mathbb{R}$.

Na Figura 3.3, pode-se ver um exemplo de um cromossomo representado pelo vetor x.

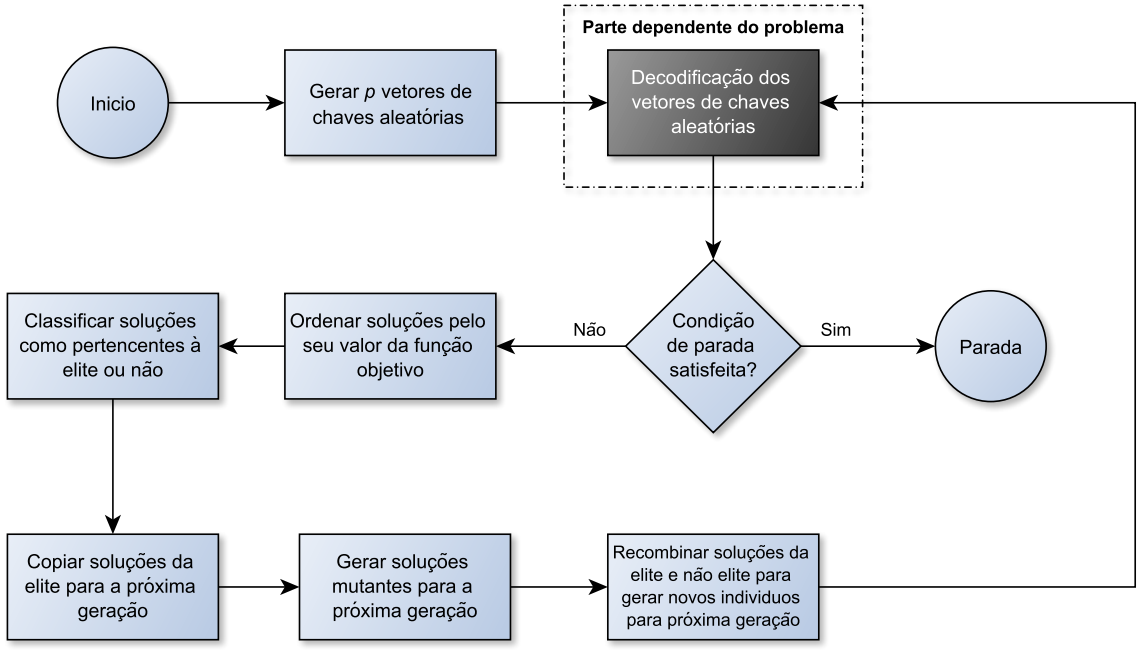

Figura 3.2: Esquema de funcionamento do BRKGA. Adaptado de Gonçalves e Resende (2010).

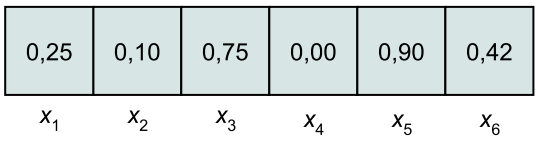

Figura 3.3: Exemplo de cromossomo com seis chaves aleatórias.

A codificação do cromossomo como um vetor de chaves aleatórias, é importante, pois cria desacoplamento da metaheurística em relação ao problema, tal que o algoritmo pode lidar com chaves aleatórias diretamente e somente uma pequena porção do algoritmo trata a parte especifica do problema, aumentando reuso de código e até a utilização de APIs para tal, somente obrigando o usuário da API a implementar esta pequena parte da metaheurística referente à decodificação, vide Figura 3.2.

3.2.2 Decodificação de Indivíduos

Após a inicialização da população, a decodificação dos cromossomos é executada. Esta etapa é a parte com maior dependência do problema a ser tratado, devendo mapear o cromossomo do espaço de solução de chaves aleatórias para o espaço de solução do problema.

Como demonstrado na Figura 3.4, o papel do decodificador é mapear o vetor x (cromossomo) de chaves aleatórias para o espaço de soluções especifico ao problema e deve ser

implementado de modo distinto em cada problema. Somente depois da fase de decodificação é possível calcular o valor da função objetivo de cada cromossomo.

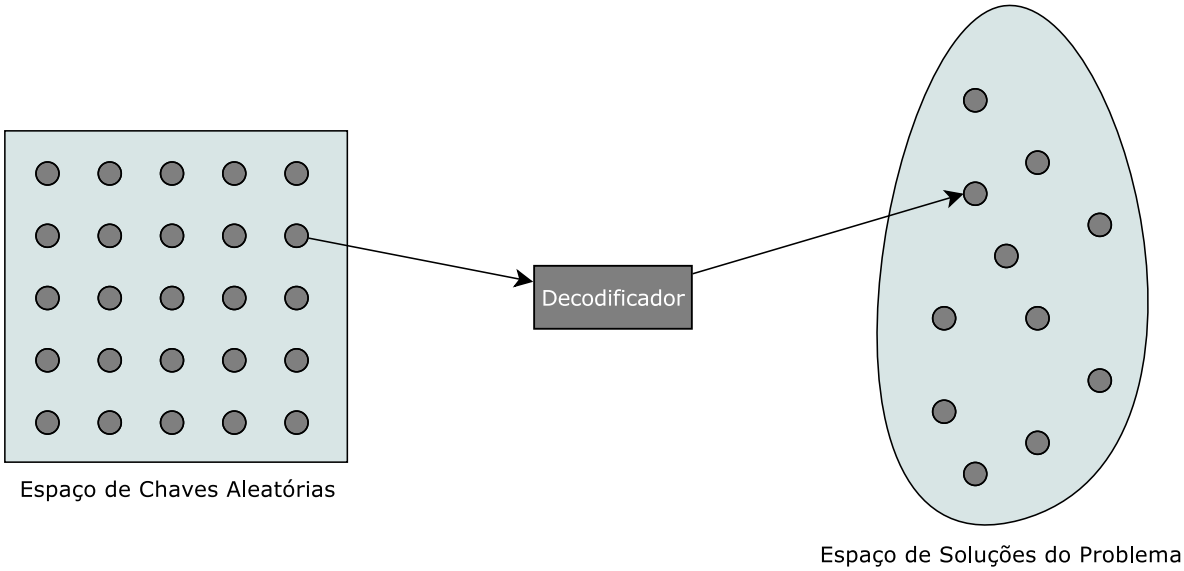

Figura 3.4: Esquema de funcionamento do decodificador, que mapeia as soluções do espaço de chaves aleatórias para o espaço de soluções do problema. Adaptado de Gonçalves e Resende (2010).

3.2.3 Elitismo

Depois dos cromossomos terem seus valores da função objetivo calculados na etapa de decodificação, eles são ordenados e posteriormente classificados em uma parte p_e pertencente à elite que comporta os melhores cromossomos e o restante não pertencente a elite. Essa parcela da população com as melhores soluções é copiada intacta para a próxima geração da população. Com isso o conceito de elitismo força parcialmente que a metaheurística busque por soluções similares as que estão demonstrando serem as melhores até então. Gonçalves e Resende (2010) afirmam que a proporção recomendada de cromossomos elite na população seja tal que $0,10p \leq p_e \leq 0,25p$ para se obter bons resultados.

Na Figura 3.5, é ilustrada a classificação da população da geração atual, em cromossomos que fazem parte da elite, ou que não fazem.

3.2.4 Mutação

Uma pequena parte da nova geração é produzida aleatoriamente para introduzir diversidade ao processo de otimização e evitar a convergência prematura em ótimos locais, estes são os cromossomos mutantes inseridos na próxima população. Na Figura 3.5, é ilustrada que uma pequena porção na próxima geração composta por estes cromossomos mutantes. Os autores

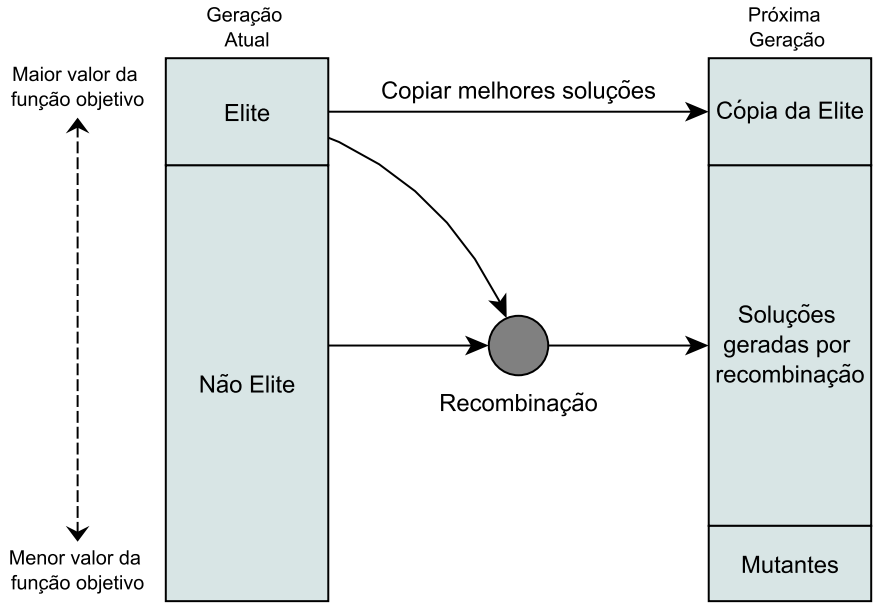

Figura 3.5: Esquema produção de uma nova geração. Adaptado de Gonçalves e Resende (2010).

Gonçalves e Resende (2010) recomendam que a proporção de mutantes na próxima geração seja tal que $0,10p \leq p_m \leq 0,30p$ para que sejam obtidos bons resultados.

3.2.5 Recombinação

Os indivíduos restantes da próxima geração são produzidos na etapa de recombinação. No caso do BRKGA, é utilizado o método *Parametrized Uniform Crossover* (Spears e Jong, 1991), que consiste em sortear para cada gene de qual cromossomos pai deve ser herdado a característica genética. Para que a metaheurística tenha a característica *biased*, ou viciada, um dos cromossomos pai deve ser escolhido aleatoriamente mas sempre pertencendo à elite e o outro deve ter sua escolha aleatória mas provindo do grupo dos que não pertencem a elite. Além disso, no processo de recombinação dos genes, a probabilidade ρ_e de um gene ser herdado do cromossomo da elite deve ser sempre maior ou igual do que a chance de um gene ser herdado do cromossomo não pertencente à elite, ou seja $\rho_e \geq 50\%$.

Na Figura 3.6, é ilustrado o processo de recombinação no BRKGA, tal que dois cromossomos são escolhidos para serem recombinados, seguindo as regras anteriormente citadas e, para cada gene, um número aleatório é sorteado entre 0 e 100. Como tem-se $\rho_e = 70\%$, se o número for menor ou igual a 70, então, é herdado o gene do cromossomo da elite, se for maior que 70, então, o gene é herdado do outro cromossomo.

O valores recomendados pelos autores Gonçalves e Resende (2010) para a probabilidade

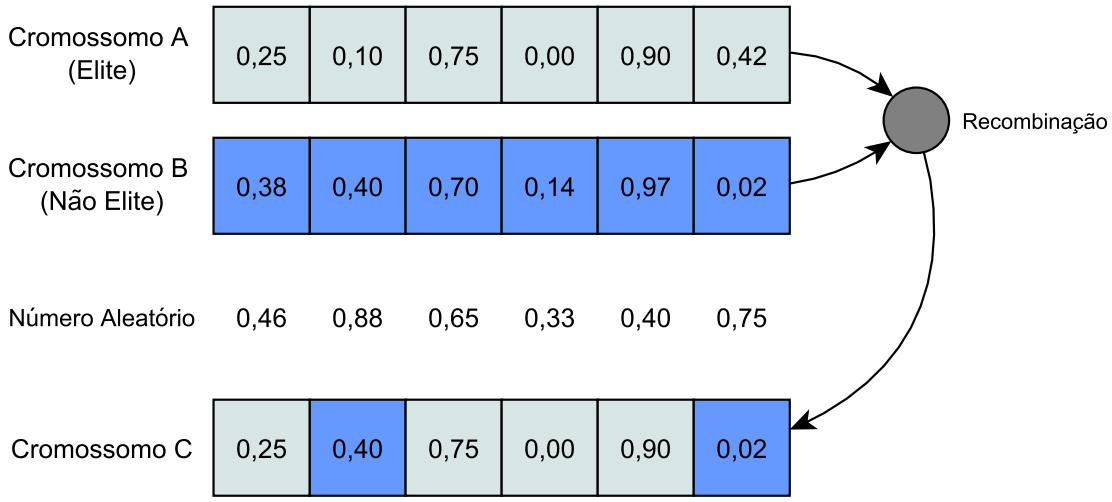

Figura 3.6: Exemplo de recombinação, com $\rho_e = 70\%$. Adaptado de Gonçalves e Resende (2010).

de um gene ser escolhido do progenitor pertencente à elite é de $50\% \leq \rho_e \leq 80\%$.

3.2.6 Condição de Parada

Para que o algoritmo não execute indefinidamente, devemos dizer a ele qual é a condição que deve ser satisfeita para que ele termine sua execução. Existem vários tipos de condição de parada, dentre elas:

- Número fixo de gerações desde o início da execução do algoritmo;
- Número fixo de gerações desde que se houve melhora na qualidade da melhor soluções;
- Tempo máximo de execução atingido;
- Encontrar uma solução com valor da função objetivo melhor ou igual a um valor estabelecido.

Esta parte do algoritmo é dependente ao problema e deve ser adaptada de acordo as necessidades intrínsecas a ele.

Como a escolha da condição de parada geralmente envolve em determinar parâmetros que façam o algoritmo dar a melhor solução dentro de um prazo de tempo ou número máximo de gerações, pode-se utilizar também de paralelização de alguns componentes do algoritmo para acelerar o tempo de execução ou aumentar a chance de se encontrar mais soluções de qualidade.

3.2.6.1 Paralelização

De acordo com Gonçalves e Resende (2010), o BRKGA tem a capacidade de paralelizar alguns elementos para prover *speedup*[1] na sua execução. Os elementos que são possíveis de serem paralelizados são:

- Geração de chaves aleatórias;
- Geração de cromossomos mutantes para a próxima geração;
- Recombinação;
- Decodificação de vetores de chaves aleatórias;

Os três primeiros não geram tanto impacto na velocidade de execução do algoritmo, mas o quarto item pode se beneficiar imensamente da paralelização, pois a etapa de decodificação é frequentemente a que mais consome processamento durante a execução da metaheurística.

Outro tipo de paralelização aceita pelo método estudado é o uso de múltiplas populações, sendo cada uma delas evoluídas em paralelo e periodicamente mesclando seus cromossomos (Gonçalves e Resende, 2010).

[1] Relação que indica melhora no tempo de execução dentre duas tarefas ou métodos

Capítulo 4

Plano de Atividades Restantes

As atividades restantes planejadas para o trabalho de monografia II no próximo período, no qual será cursada a disciplina, estão descritas na Tabela 4.1.

Tabela 4.1: Descrição das atividades e fases de desenvolvimento durante o período letivo da disciplina de monografia II

Atividades	1º Mês	2º Mês	3º Mês	4º Mês
Estudo da API	✓	✓		
Implementação utilizando a API		✓	✓	
Escrita da descrição da API	✓			
Escrita do capítulo de metodologia	✓			
Projeto de experimentos			✓	
Aplicação e análise dos experimentos				✓
Escrita do capítulo de experimentos				✓
Escrita do capítulo de conclusão				✓

Estas tarefas tem as seguintes finalidades:

- Estudo da API: aprofundar o conhecimento referente a API brkgaAPI (Toso e Resende, 2014);
- Implementação utilizando a API: colocar em prática o que foi aprendido na estudo da API, desenvolvendo um algoritmo utilizando o BRKGA para resolver o problema que será futuramente definido;
- Escrita da descrição da API: desenvolver a escrita do capítulo que apresenta a API brkgaAPI e seu funcionamento detalhado;
- Escrita do capítulo de metodologia: desenvolver a escrita do capítulo que elucida qual foi a metodologia utilizada no trabalho;
- Projeto de experimentos: estabelecer e projetar quais experimentos serão realizados para aferir a qualidade e eficiência do algoritmo proposto;

- Aplicação e análise dos experimentos: aplicar os experimentos previamente projetados e é feita a análise dos resultados;
- Escrita do capítulo de experimentos: desenvolver a escrita e formato que os resultados provenientes dos experimentos serão apresentados;
- Escrita do capítulo de conclusão: atualizar as novas conclusões, após a execução de todas as tarefas previstas para o trabalho.

Capítulo 5

Conclusões

Neste trabalho foram introduzidos os conceitos referentes a algoritmos genéticos, RKGA e BRKGA. Também foi revisada a literatura referente ao BRKGA, assim como suas aplicações e eficiência nas diversas áreas, além de efetuado um estudo detalhado do funcionamento desta metaheurística. Devido a este levantamento, é possível concluir que o BRKGA demonstra ser eficiente em vários tipos de problemas e se mostra uma ferramenta interessante para a aplicação em outros problemas diversos. Foi deixado em aberto a escolha de um problema para ser tratado na continuidade deste trabalho, utilizando a API do BRKGA brkgaAPI e posterior analise de sua eficiência.

Referências Bibliográficas

Andrade, C. E.; Miyazawa, F. K. e Resende, M. G. (2013). Evolutionary algorithm for the k-interconnected multi-depot multi-traveling salesmen problem. In *Proceedings of the 15th Annual Conference on Genetic and Evolutionary Computation*, GECCO '13, pp. 463–470, New York, NY, USA. ACM.

Bean, J. C. (1994). Genetic algorithms and random keys for sequencing and optimization. *INFORMS Journal on Computing*, 6(2):154–160.

Buriol, L. S.; Hirsch, M. J.; Pardalos, P. M.; Querido, T.; Resende, M. G. C. e Ritt, M. (2010). A biased random-key genetic algorithm for road congestion minimization. *Optimization Letters*, 4(4):619–633.

Coco, A. A.; Noronha, T. F. e Santos, A. C. (2012). A biased random-key genetic algorithm for the robust shortest path problem. In *Proceedings of Global Optimization Workshop (GO2012)*, pp. 53–56.

Darwin, C. (1859). *On the origin of species*. New York :D. Appleton and Co.,.

de Andrade, C. E.; Toso, R. F.; Resende, M. G. C. e Miyazawa, F. K. (2015). Biased random-key genetic algorithms for the winner determination problem in combinatorial auctions. *Evolutionary computation*, 23(2):279–307.

Duarte, A.; Martí, R.; Resende, M. e Silva, R. (2014). Improved heuristics for the regenerator location problem. *International Transactions in Operational Research*, 21(4):541–558.

Eiben, A. E. e Smith, J. E. (2003). *Introduction to Evolutionary Computing*. SpringerVerlag.

Festa, P.; Gonçalves, J. F.; Resende, M. G. C. e Silva, R. M. A. (2010). *Experimental Algorithms: 9th International Symposium, SEA 2010, Ischia Island, Naples, Italy, May 20-22, 2010. Proceedings*, chapter Automatic Tuning of GRASP with Path-Relinking Heuristics with a Biased Random-Key Genetic Algorithm, pp. 338–349. Springer Berlin Heidelberg, Berlin, Heidelberg.

Fontes, D. B. M. M. e Gonçalves, J. F. (2012). A multi-population hybrid biased random key genetic algorithm for hop-constrained trees in nonlinear cost flow networks. *Optimization Letters*, 7(6):1303–1324.

Gendreau, M. e Potvin, J.-Y. (2010). *Handbook of Metaheuristics*. Springer Publishing Company, Incorporated, 2ª edição.

Gonçalves, J. F. e Resende, M. G. C. (2010). Biased random-key genetic algorithms for combinatorial optimization. *Journal of Heuristics*, 17(5):487–525.

Gonçalves, J. F. e Resende, M. G. C. (2011). A biased random-key genetic algorithm for job-shop scheduling. *AT&T Labs Research Technical Report*, 46:253–271.

Gonçalves, J. F.; Resende, M. G. C. e Mendes, J. J. M. (2010). A biased random-key genetic algorithm with forward-backward improvement for the resource constrained project scheduling problem. *Journal of Heuristics*, 17(5):467–486.

Gonçalves, J. F. e Resende, M. G. (2012). A parallel multi-population biased random-key genetic algorithm for a container loading problem. *Computers & Operations Research*, 39(2):179 – 190.

Gonçalves, J. F. e Resende, M. G. (2013). A biased random key genetic algorithm for 2d and 3d bin packing problems. *International Journal of Production Economics*, 145(2):500 – 510.

Gonçalves, J. F.; Resende, M. G. C. e Toso, R. F. (2012). Biased and unbiased random-key genetic algorithms: An experimental analysis. Technical report, AT&T Labs Research, Florham Park.

Goulart, N.; de Souza, S.; Dias, L. e Noronha, T. (2011a). Biased random-key genetic algorithm for fiber installation in optical network optimization. In *Evolutionary Computation (CEC), 2011 IEEE Congress on*, pp. 2267–2271.

Goulart, N.; de Souza, S.; Dias, L. e Noronha, T. (2011b). Biased random-key genetic algorithm for fiber installation in optical network optimization. In *Evolutionary Computation (CEC), 2011 IEEE Congress on*, pp. 2267–2271.

Grasas, A.; Ramalhinho, H.; Pessoa, L. S.; Resende, M. G.; Caballé, I. e Barba, N. (2014). On the improvement of blood sample collection at clinical laboratories. *BMC Health Services Research*, 14(1):1–9.

Holland, J. H. (1975). *Adaptation in Natural and Artificial Systems*. The University of Michigan Press.

Lucena, M. L.; Andrade, C. E.; Resende, M. G. e Miyazawa, F. K. (2014). Some extensions of biased random-key genetic algorithms.

Mitchell, M. (1998). *An Introduction to Genetic Algorithms*. MIT Press, Cambridge, MA, USA.

Morán-Mirabal, L. F.; González-Velarde, J. L. e Resende, M. G. C. (2013a). *Hybrid Metaheuristics: 8th International Workshop, HM 2013, Ischia, Italy, May 23-25, 2013. Proceedings*, chapter Automatic Tuning of GRASP with Evolutionary Path-Relinking, pp. 62–77. Springer Berlin Heidelberg, Berlin, Heidelberg.

Morán-Mirabal, L. F.; González-Velarde, J. L. e Resende, M. G. C. (2013b). Randomized heuristics for the family traveling salesperson problem. In *International Transactions in Operational Research*.

Morán-Mirabal, L. F.; González-Velarde, J. L.; Resende, M. G. C. e Silva, R. M. A. (2013c). Randomized heuristics for handover minimization in mobility networks. *Journal of Heuristics*, 19(6):845–880.

Moreira, M. C. O.; Ritt, M.; Costa, A. M. e Chaves, A. A. (2012). Simple heuristics for the assembly line worker assignment and balancing problem. *Journal of Heuristics*, 18(3):505–524.

Noronha, T. F.; Resende, M. G. e Ribeiro, C. C. (2011). A biased random-key genetic algorithm for routing and wavelength assignment. *J. of Global Optimization*, 50(3).

Papadimitriou, C. H. e Steiglitz, K. (1982). *Combinatorial Optimization: Algorithms and Complexity*. Prentice-Hall, Inc., Upper Saddle River, NJ, USA.

Pedrola, O.; Careglio, D.; Klinkowski, M.; Velasco, L.; Bergman, K. e Solé-Pareta, J. (2013a). Metaheuristic hybridizations for the regenerator placement and dimensioning problem in sub-wavelength switching optical networks. *European Journal of Operational Research*, 224(3):614 – 624.

Pedrola, O.; Ruiz, M.; Velasco, L.; Careglio, D.; de Dios, O. G. e Comellas, J. (2013b). A grasp with path-relinking heuristic for the survivable ip/mpls-over-wson multi-layer network optimization problem. *Computers & Operations Research*, 40(12):3174 – 3187.

Prasetyo, H.; Fauza, G.; Amer, Y. e Lee, S. (2015). Survey on applications of biased-random key genetic algorithms for solving optimization problems. In *Industrial Engineering and Engineering Management (IEEM), 2015 IEEE International Conference on*, pp. 863–870. IEEE.

Reis, R.; Ritt, M.; Buriol, L. S. e Resende, M. G. C. (2011). A biased random-key genetic algorithm for ospf and deft routing to minimize network congestion. *International Transactions in Operational Research*, 18(3):401–423.

Resende, M. G. C. (2011). Biased random-key genetic algorithms with applications in telecommunications. *TOP*, 20(1):130–153.

Resende, M. G. C.; Toso, R. F.; Gonçalves, J. F. e Silva, R. M. A. (2011). A biased random-key genetic algorithm for the steiner triple covering problem. *Optimization Letters*, 6(4):605–619.

Ruiz, M.; Pedrola, O.; Velasco, L.; Careglio, D.; Fernández-Palacios, J. e Junyent, G. (2011). Survivable ip/mpls-over-wson multilayer network optimization. *Optical Communications and Networking, IEEE/OSA Journal of*, 3(8):629–640.

Silva, R.; Resende, M. e Pardalos, P. (2015). A python/c++ library for bound-constrained global optimization using a biased random-key genetic algorithm. *Journal of Combinatorial Optimization*, 30(3):710–728.

Silva, R.; Resende, M.; Pardalos, P. e Gonçalves, J. (2012). Biased random-key genetic algorithm for bound-constrained global optimization. In *Proceedings of the global optimization workshop*, pp. 133–136.

Silva, R. M.; Resende, M. G. e Pardalos, P. M. (2014). Finding multiple roots of a box-constrained system of nonlinear equations with a biased random-key genetic algorithm. *Journal of Global Optimization*, 60(2):289–306.

Spears, V. M. e Jong, K. A. D. (1991). On the virtues of parameterized uniform crossover. In *In Proceedings of the Fourth International Conference on Genetic Algorithms*, pp. 230–236.

Stefanello, F.; Buriol, L. S.; Hirsch, M. J.; Pardalos, P. M.; Querido, T.; Resende, M. G. C. e Ritt, M. (2015). On the minimization of traffic congestion in road networks with tolls. *Annals of Operations Research*, pp. 1–21.

Tangpattanakul, P.; Jozefowiez, N. e Lopez, P. (2012). *Parallel Problem Solving from Nature - PPSN XII: 12th International Conference, Taormina, Italy, September 1-5, 2012, Proceedings, Part II*, chapter Multi-objective Optimization for Selecting and Scheduling Observations by Agile Earth Observing Satellites, pp. 112–121. Springer Berlin Heidelberg, Berlin, Heidelberg.

Toso, R. F. e Resende, M. G. C. (2014). A c++ application programming interface for biased random-key genetic algorithms. *Optimization Methods and Software*.