UNIVERSIDADE FEDERAL DE GOIÁS
INSTITUTO DE INFORMÁTICA

RONEY LOPES LIMA

Operadores de Recombinação Baseados em Permutação para Representações de Grafos

Goiânia
2017

TERMO DE CIÊNCIA E DE AUTORIZAÇÃO PARA DISPONIBILIZAR VERSÕES ELETRÔNICAS DE TESES E DISSERTAÇÕES NA BIBLIOTECA DIGITAL DA UFG

Na qualidade de titular dos direitos de autor, autorizo a Universidade Federal de Goiás (UFG) a disponibilizar, gratuitamente, por meio da Biblioteca Digital de Teses e Dissertações (BDTD/UFG), regulamentada pela Resolução CEPEC nº 832/2007, sem ressarcimento dos direitos autorais, de acordo com a Lei nº 9610/98, o documento conforme permissões assinaladas abaixo, para fins de leitura, impressão e/ou *download*, a título de divulgação da produção científica brasileira, a partir desta data.

1. Identificação do material bibliográfico: **[X] Dissertação** **[ ] Tese**

2. Identificação da Tese ou Dissertação:

Nome completo do autor: Roney Lopes Lima

Título do trabalho: Operadores de Recombinação Baseados em Permutação para Representações de Grafos

3. Informações de acesso ao documento:

Concorda com a liberação total do documento [X] SIM [ ] NÃO[1]

Havendo concordância com a disponibilização eletrônica, torna-se imprescindível o envio do(s) arquivo(s) em formato digital PDF da tese ou dissertação.

Roney Lopes Lima

Assinatura do(a) autor(a)[2]

Ciente e de acordo:

Assinatura do(a) orientador(a)[2]

[1] Neste caso o documento será embargado por até um ano a partir da data de defesa. A extensão deste prazo suscita justificativa junto à coordenação do curso. Os dados do documento não serão disponibilizados durante o período de embargo.

Casos de embargo:
- Solicitação de registro de patente
- Submissão de artigo em revista científica
- Publicação como capítulo de livro
- Publicação da dissertação/tese em livro

[2] A assinatura deve ser escaneada.

RONEY LOPES LIMA

Operadores de Recombinação Baseados em Permutação para Representações de Grafos

Dissertação apresentada ao Programa de Pós–Graduação do Instituto de Informática da Universidade Federal de Goiás, como requisito parcial para obtenção do título de Mestre em Ciência da Computação.

Área de concentração: Ciência da Computação.

Orientadora: Profa. Dra. Telma Woerle de Lima Soares

Goiânia
2017

Ficha de identificação da obra elaborada pelo autor, através do
Programa de Geração Automática do Sistema de Bibliotecas da UFG.

Lopes Lima, Roney
Operadores de Recombinação Baseados em Permutação para Representações de Grafos [manuscrito] / Roney Lopes Lima. - 2017.
81 f.

Orientador: Profa. Dra. Telma Woerle de Lima Soares.
Dissertação (Mestrado) - Universidade Federal de Goiás, Instituto de Informática (INF), Programa de Pós-Graduação em Ciência da Computação, Goiânia, 2017.
Bibliografia.
Inclui siglas, abreviaturas, tabelas, algoritmos, lista de figuras, lista de tabelas.

1. Nó-Profundidade. 2. operadores de recombinação. 3. permutação. 4. projeto de rede. 5. Árvore Geradora Mínima com Restrição de Diâmetro. I. Woerle de Lima Soares, Telma, orient. II. Título.

CDU 004

MINISTÉRIO DA EDUCAÇÃO
UNIVERSIDADE FEDERAL DE GOIÁS
INSTITUTO DE INFORMÁTICA
PROGRAMA DE PÓS-GRADUAÇÃO EM CIÊNCIA DA COMPUTAÇÃO

UFG

ATA Nº 20/2017

ATA DA SESSÃO DE JULGAMENTO DA DISSERTAÇÃO DE MESTRADO DE RONEY LOPES LIMA

Aos vinte e três dias do mês de agosto de dois mil e dezessete, às dezessete horas, na sala 151 do Instituto de Informática da Universidade Federal de Goiás, Campus Samambaia, reuniu-se a banca examinadora designada na forma regimental pela Coordenação do Curso para julgar a dissertação de mestrado intitulada **"Operadores de Recombinação Baseados em Permutação para Representações de Grafos"**, apresentada pelo aluno Roney Lopes Lima como parte dos requisitos necessários à obtenção do grau de Mestre em Ciência da Computação, área de concentração Ciência da Computação. A banca examinadora foi presidida pela orientadora do trabalho de dissertação, Professora Doutora Telma Woerle de Lima Soares (INF/UFG), tendo como membros os Professores Doutores Danilo Sipoli Sanches (UTFPR) e Celso Gonçalves Camilo Júnior (INF/UFG). Aberta a sessão, o candidato expôs seu trabalho. Em seguida, o aluno foi arguido pelos membros da banca e:

(X) tendo demonstrado suficiência de conhecimento e capacidade de sistematização do tema de sua dissertação, a banca concluiu pela **aprovação** do candidato, sem restrições.

( ) tendo demonstrado suficiência de conhecimento e capacidade de sistematização do tema de sua dissertação, a banca concluiu pela **aprovação** do candidato, condicionado a satisfazer as exigências listadas na Folha de Modificação de Dissertação de Mestrado anexa à presente ata, no prazo máximo de 60 dias, a contar da presente data, ficando o professor-orientador responsável por atestar o cumprimento dessas exigências.

( ) não tendo demonstrado suficiência de conhecimento e capacidade de sistematização do tema de sua dissertação, a banca concluiu pela **reprovação** do candidato.

Os trabalhos foram encerrados às 19 horas. Nos termos do Regulamento Geral dos Cursos de Pós-Graduação desta Universidade, lavrou-se a presente ata que, lida e julgada conforme, segue assinada pelos membros da banca examinadora.

Profa. Dra. Telma Woerle de Lima Soares _____

Prof. Dr. Danilo Sipoli Sanches _____

Prof. Dr. Celso Gonçalves Camilo Júnior _____

RONEY LOPES LIMA

Operadores de Recombinação Baseados em Permutação para Representações de Grafos

Dissertação defendida no Programa de Pós–Graduação do Instituto de Informática da Universidade Federal de Goiás como requisito parcial para obtenção do título de Mestre em Ciência da Computação, aprovada em 23 de Agosto de 2017, pela Banca Examinadora constituída pelos professores:

Profa. Dra. Telma Woerle de Lima Soares
Instituto de Informática – UFG
Presidente da Banca

Prof. Dr. Celso Gonçalves Camilo Junior
Instituto de Informática – UFG

Prof. Dr. Danilo Sipoli Sanches
UTFPR

Roney Lopes Lima

Graduou-se em Sistemas de Informação pelo IFG. Especializou-se em Segurança e Integração de Redes de Computadores Corporativos pela Faculdade Senac. Atualmente é professor no Departamento de Áreas Acadêmicas do IFG Câmpus Jataí e ministra as disciplinas de Linguagem de Programação, Engenharia de Software I, Engenharia de Software II e Linguagem de Programação para Web.

Agradecimentos

Aos meus pais, Roberto e Neiva e ao meu irmão Reiner por alicerçarem toda minha trajetória de vida.

À minha namorada Joice pelo amor, companheirismo, paciência e acolhimento desprendidos nessa fase da minha vida.

À minha orientadora Dra. Telma Woerle de Lima Soares por todo suporte acadêmico dedicado e por me inspirar com seu profissionalismo, dedicação e humanidade que caracterizaram nossa relação.

Aos professores Dr. Anderson Soares, Dr. Humberto Longo, Dr. Vinicius Borges por toda contribuição acadêmica de elevadíssima qualidade que dedicaram nas disciplinas que cursei.

Aos amigos que fiz durante o mestrado, especialmente Gilmar, Ana Clara, Alexandre, Beatriz, Áurea e Welder.

Aos meus colegas de coordenação do IFG que me deram todo o suporte para conseguir fazer o mestrado mesmo sem licença, perdoando minhas ausências e priorizando minha situação na elaboração dos horários de aula.

À Fundação de Amparo à Pesquisa do Estado de Goiás (FAPEG) pela concessão de bolsa de formação.

Resumo

Lima, Roney L.. **Operadores de Recombinação Baseados em Permutação para Representações de Grafos**. Goiânia, 2017. 81p. Dissertação de Mestrado. Instituto de Informática, Universidade Federal de Goiás.

A aplicação de Algoritmos Evolutivos na resolução de problemas caracterizados pela inviabilidade de solução através de métodos determinísticos, fez dessa técnica um objeto vastamente investigado. Sua aplicação para Problemas de Projeto de Redes (PPRs), tem sido especialmente estudada. PPRs são caracterizados por modelar problemas reais relacionados a design de redes aplicados a distribuição de recursos, logística, telecomunicações, roteamento e até mesmo redes sociais. A solução desses problemas envolve a busca de um grafo como uma árvore por exemplo que atenda a critérios de minimização de custos, disponibilidade, escala entre outras restrições que os tornam complexos. A aplicação de Algoritmos Evolutivos a PPRs demanda a utilização de uma Representação que codifique adequadamente soluções para esses problemas. A Representação Nó-Profundidade (RNP) tem sido estudada e apresentado resultados que despertaram a atenção dos pesquisadores nesse tema. Neste trabalho, propõe-se o desenvolvimento de um novo operador de recombinação para a RNP chamado NCX, com base no operador CX de recombinação em permutações. Além disso, é proposto um método para correção de soluções infactíveis devido a profundidade inválida para a posição no *array*. O método de correção é aplicado tanto para NCX, quanto para outros dois operadores de recombinação já desenvolvidos para a RNP, o NOX cujo funcioamento é inspirado no operador OX, e NPBX cujo funcionamento é inspirado no operador PBX. Os operadores com os seus devidos métodos de correção são validados para as propriedades tendência e hereditariedade e por fim são aplicados ao Problema da Árvore Geradora Mínima com Restrição de Diâmetro (BDMSTP) através de Algoritmos Evolutivos desenvolvidos para esse PPR. Os resultados mostram que os operadores possuem tendência para árvores estrela e boa hereditariedade das arestas e das profundidades dos vértices. Os operadores desenvolvidos também mostraram competitividade ao serem aplicados ao BDMSTP, chegando a superar outras representações em qualidade das soluções.

Palavras–chave

Nó-Profundidade, operadores de recombinação, permutação, projeto de rede, Árvore Geradora Mínima com Restrição de Diâmetro.

Abstract

Lima, Roney L.. **Permutation-based Recombination Operators for Graph Representations**. Goiânia, 2017. 81p. MSc. Dissertation. Instituto de Informática, Universidade Federal de Goiás.

The application of Evolutionary Algorithms in the solution of problems characterized by the unviability through deterministic methods, has made this technique a vast object investigated. Its application to Network Design Problems (NDPs), has been specially studied. NDPs are characterized by modeling real world problems related to network design applied to resource distribution, logistics, telecommunications, routing and even social networks. The solution to these problems involves searching for a graph such as trees that meets criteria for cost minimization, availability, scaling among other constraints that make them complex. The application of Evolutionary Agorithms to NDPs requires a Representation that codes solutions properly towards to these problems. The Node-Depth Encoding (NDE) has been studied and presented results that have aroused the attention of researchers in this topic. In this work, we propose the development of a new recombination operator for NDE called NCX, based on the permutation recombination operator CX. In addition, a method is proposed for correction of infeasible solutions due to an invalid depth for a position in the array. The correction method is applied to both NCX, NOX and NPBX. The operators with their methods of correction are validated for the bias and heritability properties and finally are applied to the Bounded Diameter Minimmum Spanning Tree (BDMSTP) through Evolutionary Algorithms developed for this NDP. The results show that the operators have bias towards to star like trees and good heritability of the edges and depths of the vertices. The developed operators also showed competitiveness when applied to the BDMSTP, even surpassing other representations in the quality of the solutions.

Keywords

Node-Depth, recombination operators, permutation, network design, Bounded Diameter Minimmum Spanning Tree.

Sumário

Lista de Figuras

Lista de Tabelas

Lista de Algoritmos

Lista de Abreviaturas e Siglas

AE Algoritmo Evolutivo

AG Algoritmo Genético

BDMSTP *Bounded Diameter Minimmum Spanning Tree*

CAO *Change Ancestor Operator*

CV *Characteristic Vector*

CX *Cycle Crossover*

dc-MST *Degree Constraint Minimmum Spanning Tree*

DFS *Depth-First-Search*

EANR *Evolutionary Algorithm Node-Depth Representation*

EE Estratégias Evolutivas

EP *Evolutionary Programming*

ES *Evolutionary Strategies*

GA *Genetic Algorithms*

GP *Genetic Programming*

JR-ESEA *Julstrom and Raidl-Edge-Sets Evolutionary Algorithm*

NCX *NDE Cycle Crossover*

NCX-SDF *NDE Subtree Depth Fix Cycle Crossover*

NDE Representação Nó-Profundidade (*Node-Depth Encoding*)

NetKeys *Network Random Keys*

NOX *NDE Order Crossover*

NOX-SDF *NDE Subtree Depth Fix Order Crossover*

NPBX *NDE Position Based Crossover*

NPBX-SDF *NDE Subtree Depth Fix Position Based Crossover*

OCST *Optimal Communication Spanning Tree*

PAO *Preserve Ancestor Operator*

PPR Problema de Projeto de Redes

RNP Representação Nó-Profundidade

SDF *Subtree Depth Fix*

CAPÍTULO 1

Introdução

No mundo real, vários problemas complexos demandam o uso do poder computacional para serem resolvidos. Dentre eles, há uma classe de problemas relacionados a projetos de redes como sistemas de distribuição de energia elétrica, roteamento de veículos e sistemas de telecomunicação. Esses problemas tornam-se complexos à medida em que suas dimensões limitam ou até mesmo inviabilizam o uso de sistemas computacionais convencionais, haja exigência de resposta rápida e soluções que respeitem determinadas restrições. Essa classe de problemas conhecida como Problemas de Projeto de Redes (PPRs) pode ser modelada através do uso de grafos e tem sido submetida às técnicas de Inteligência Computacional como os Algoritmos Evolutivos (AEs).

AEs simulam os processos da Teoria da Evolução das Espécies de Darwin em que os indivíduos de uma população competem pelos recursos locais para sobreviver e reproduzem entre si trocando informações genéticas que são passadas para gerações posteriores [27, 6]. Nesse paradigma cada indivíduo de uma população representa uma solução para um determinado problema e deve ser codificado e estruturado de forma que possa ser avaliado conforme sua capacidade de solução. A troca de informações entre indivíduos gera novas soluções que também são alternativas para o problema. Em síntese, os melhores indivíduos reproduzem mais enquanto os menos aptos deixam de existir e o algoritmo caminha para solucionar o problema através da busca pelo melhor indivíduo por gerações de populações. Várias pesquisas evidenciam o bom desempenho dos AEs aplicados aos PPRs [49, 9, 40, 35].

Contudo, o uso de AEs convencionais para PPRs é muitas vezes inviável devido ao caráter muito específico da estrutura da solução de um PPR [38]. A geração de soluções inválidas que não condizem com a estrutura de uma rede e a complexidade de espaço e de tempo, levaram os pesquisadores a buscar novas formas de codificar as soluções nos AEs aplicados aos PPRs. Novas propostas de representações para AEs surgiram dessas pesquisas e obtiveram relativo sucesso na resolução desses problemas [15, 41, 35, 48, 13, 46, 32, 10]. O desenvolvimento ou a escolha de uma representação para AEs deve considerar a relação dependente entre os operadores de busca e a forma como as soluções são codificadas [39].

Dentre as representações conhecidas, a Representação Nó-Profundidade (RNP ou NDE *Node-Depth Encoding*) [8, 7, 10], mostrou-se bastante competitiva por sua baixa complexidade de tempo e qualidade das soluções obtidas. A RNP tem sido alvo de várias pesquisas que procuram delinear suas propriedades com a finalidade de contextualizar o desempenho da representação e a que classe de problemas essa pode ser aplicada. A RNP utiliza dois operadores de mutação chamados de PAO (*Preserve Ancertor Operator*) e CAO (*Change Ancestor Operator*) , e dois operadores de recombinação conhecidos como NOX (NDE *Order Crossover*) e NPBX (NDE *Position Based Crossover*). A forma como os indivíduos são codificados pela RNP (ver Seção 3.2.6), utiliza o par (nó, profundidade) para a construção da solução na ordem de uma busca em profundidade. Por esse motivo, durante a aplicação dos operadores de recombinação pode ser necessária a utilização de um mecanismo de correção de profundidade de forma a preservar a ordem de uma busca em profundidade. Essa correção é necessária para que a solução sendo gerada seja sempre válida conforme a codificação da RNP. A proposta de correção definida por Lima et. al [8] altera a profundidade do gene infactível para ser a profundidade do gene anterior acrescida de 1.

Em [9], onde os operadores de recombinação NOX e NPBX para a RNP foram desenvolvidos e analisados para as propriedades tendência e hereditariedade, é relatado que o operador NOX produz soluções em que cerca de 20% das arestas não são oriundas de nenhum dos pais envolvidos na recombinação. É relatado também que o operador NPBX apresenta piora da hereditariedade das arestas a medida em que o tamanho da instância testada aumenta. Além disso, os operadores de recombinação desenvolvidos apresentaram tendência para árvores estrela.

O objetivo deste trabalho é o desenvolvimento de um novo operador de recombinação para a RNP com base nos operadores de permutação e de um novo método de correção de soluções infactíveis com profundidade inválida. O novo operador de recombinação baseado em permutação nomeado como NCX (ver Capítulo 4), é desenvolvido tendo como referência o conhecido operador CX (*Cycle Crossover*) para permutações [31, 2, 1, 33]. O operador de recombinação CX realiza a transferência através da identificação de ciclos de elementos entre os indivíduos e é caracterizado por preservar as posições absolutas e a ordem dos elementos das permutações. O NCX para a RNP possui funcionamento muito semelhante ao CX uma vez que os ciclos são sempre iniciados a partir da primeira posição das soluções, preservando dessa forma os vértices raízes dos pais nos filhos. A hipótese é que da a característica da RNP de codificar através de vértices ordenados, o novo operador possa preservar mais arestas melhorando sua hereditariedade para esse aspecto.

O novo método de correção de soluções infactíveis por profundidade é chamado SDF (*Subtree Depth Fix*) (ver Capítulo 4). SDF é aplicado para corrigir a posição de

um par (nó, profundidade) cuja inserção em determinada posição do cromossomo torna a solução inválida devido ao valor da profundidade. Para tal, busca-se por uma subárvore, posterior ao vértice atual, cuja raiz possua profundidade válida. Dessa forma, antecipa-se a inserção dessa subárvore válida na posição cuja invalidade da profundidade foi constatada, postergando a inserção do gene inválido ao invés de alterar sua profundidade. Como o método SDF prioriza a busca de uma subárvore válida e evita ou posterga a modificação da profundidade do vértice inválido, espera-se que este método possa melhorar a preservação das estruturas das soluções, diminuindo dessa forma a tendência para árvores estrela. O novo método é validado tanto nos dois operadores de recombinação NOX e NPBX da RNP, quanto no novo operador proposto NCX. Os operadores que dispuserem do novo método de correção serão identificados como NOX-SDF, NPBX-SDF e NCX-SDF, com a finalidade de possibilitar a distinção com os operadores com o método de correção proposto anteriormente.

Para validar NCX e SDF, é realizada uma análise das propriedades tendência e hereditariedade. A propriedade tendência é medida para árvores geradoras mínimas e árvores estrela, enquanto a propriedade hereditariedade é medida para arestas e profundidade dos vértices. Nessa análise, considera-se o uso da métrica Distância de Hamming [16] para calcular o quão distantes ou diferente são as soluções entre si.

Além da análise de propriedades, foram executados testes complementares nos operadores com o propósito de obter indícios que ajudassem a compreender os resultados apresentados na análise das propriedades. Dois testes complementares foram definidos. O primeiro realiza uma contagem da aplicação de cada tipo de correção de profundidade que os operadores implementam. O segundo avalia a diferença de profundidade dos vértices ao aplicar o método original de correção, inclusive nos operadores que dispõem do método SDF.

Por fim, um AE com os operadores implementados neste trabalho é aplicado ao Problema da Árvore Geradora Mínima com Restrição de Diâmetro (BDMSTP, *Bounded Diameter Minimum Spanning Tree Problem*) (ver Seção 4.3). Nesse PPR, busca-se uma árvore geradora mínima submetida a uma determinada restrição de diâmetro $D < n - 1$. O diâmetro de uma árvore é a distância máxima entre quaisquer dois vértices da árvore. O BDMSTP é considerado um problema da classe NP-Difícil quando $D \geq 4$ [30].

Os resultados mostram que o operador desenvolvido neste trabalho e o método de correção proposto não possuem tendência para árvores geradoras mínimas. Por outro lado, todos os operadores apresentaram alguma tendência para árvores estrela, independente do método de correção, com ênfase para os operadores NPBX-SDF, NCX e NCX-SDF. Esses operadores apresentam a tendência mais forte para esse tipo de solução em comparação com os outros. Com a exceção do operador NOX-SDF, que nos testes apresentou piora da hereditariedade das arestas, todos os outros operadores mostraram bons resultados

para essa propriedade, com ênfase para os operadores NPBX, NPBX-SDF e NCX. Apesar de implementarem o método de correção SDF para correção de inviabilidades relacionadas às profundidades dos vértices, os testes mostraram que não houve melhora da hereditariedade para essa característica nos operadores.

Os testes complementares revelam que o número de correções de profundidade utilizando o método SDF é ligeiramente maior do que o método original nas primeiras gerações dos operadores NPBX e NCX. Ambos os métodos ficam próximos entre si tendendo a zero ao longo das gerações para esses operadores. Para o operador NOX-SDF, a aplicação do método original de correção de profundidade é maior do que na versão NOX que implementa exclusivamente esse método e não tende a zero ao longo das gerações, o que significa que a inclusão do método SDF no operador NOX, causou modificações mais agudas nas estruturas o que explica sua piora para a hereditariedade das arestas e um indício de dificuldade de convergência da população quando esse operador é aplicado.

Os AEs desenvolvidos com os operadores de recombinação propostos nesse trabalho e aplicados ao BDMSTP, mostraram competitividade para solucionar esse PPR em termos de qualidade da melhor solução. O operador NOX-SDF obteve os melhores resultados, superando em qualidade de solução para instâncias de menor tamanho alguns resultados relatados por outras representações com as quais os operadores desenvolvidos foram comparados, embora tenha exigido um número maior de gerações para tal.

Este trabalho está organizado da seguinte forma:

- No Capítulo 2, os principais conceitos e elementos dos Algoritmos Evolutivos são apresentados.
- No Capítulo 3 as principais representações encontradas na literatura são apresentadas, inclusive a representação objeto deste trabalho, a RNP. Os principais problemas de projeto de redes são apresentados e formulados.
- No Capítulo 4, os materiais e métodos utilizados neste trabalho são apresentados.
- No Capítulo 5, os resultados alcançados são descritos.
- No Capítulo 6 estão as conclusões.

CAPÍTULO 2

Algoritmos Evolutivos

Em diversas áreas tais como engenharias, biológicas e computação, a solução de problemas que demandam grande complexidade computacional, como problemas de otimização por exemplo, têm inspirado a busca por estratégias que possam responder com alguma eficiência a essas demandas. Tendo como referência a Biologia, mais especificamente a Teoria da Evolução das Espécies de Darwin, a Computação Evolutiva despertou a atenção do meio científico ao obter êxito quando aplicada à vários desses problemas complexos.

Na Teoria da Evolução de Darwin, todos os indivíduos de um ecossistema disputam entre si os recursos limitados do ambiente. Os indivíduos cuja capacidade de se adaptar ao ambiente é limitada, tendem a não sobreviver e consequentemente reproduzir menos do que os mais aptos, fazendo com que seus genes não sejam propagados para futuras gerações num processo chamado seleção natural.

Por outro lado, indivíduos mais aptos e justamente por serem mais aptos, acabam reproduzindo mais do que os indivíduos menos aptos, cruzando suas informações genéticas com outros indivíduos que igualmente conservam boa capacidade adaptativa. Essa troca de informações genéticas entre elementos de uma população propicia a possibilidade de geração de uma prole com maior capacidade adaptativa que seus ancestrais, além de proporcionar diversidade à população, o que faz com que a cada geração a população seja cada vez mais apta ao ambiente.

Entretanto esse processo não é dirigido e não existe nenhuma garantia que descendentes de pais muito bem adaptados também o sejam. A evolução é um processo no qual os seres vivos passam por um conjunto de modificações através dos tempos, modificações proporcionadas por alguns fatores como mutação gênica, recombinação gênica, seleção natural e isolamentos [27].

Os AEs podem ser classificados em quatro linhas principais: Programação Evolutiva, Estratégias Evolutivas, Programação Genética e Algoritmos Genéticos [2]. As Seções 2.2, 2.3, 2.4 e 2.5 abordam cada uma dessas linhas.

Algoritmos Evolutivos (AEs) compreendem a utilização de algoritmos computacionais de busca probabilística de soluções baseados no processo de evolução biológica

[9]. De Jong em [6] descreve um modelo padrão para a dinâmica dos AEs da seguinte forma:

- Uma população de tamanho constante m é envolvida durante todo o processo.
- A população corrente é usada como uma fonte de pais para produzir n filhos.
- A população expandida é reduzida de $m+n$ para m indivíduos.

Seguindo esse modelo padrão, pode-se descrever em mais detalhes os princípios de funcionamento de um AE análogos à Teoria da Evolução, os quais são apresentados na Seção 2.1.

2.1 Princípios de funcionamento dos Algoritmos Evolutivos

Com base nos mecanismos da seleção natural e reprodução das espécies, os AEs caracterizam-se por codificar uma possível solução para um problema através de uma estrutura chamada indivíduo. Na Computação Evolutiva um indivíduo é análogo a um cromossomo na Biologia, ou seja, possui a função de manter informações (conjunto de genes) que no caso dos AEs, serão utilizadas para solucionar o problema. Um conjunto de indivíduos é chamado de população. Na teoria da evolução, os indivíduos de uma população se reproduzem trocando suas informações genéticas. Os filhos cuja codificação genética proporciona uma melhor adaptação ao ambiente conseguem transmitir algumas de suas características de geração para geração. Analogamente, nos AEs os indivíduos de uma população trocam suas informações com o objetivo de gerar filhos mais aptos para resolver um dado problema. A cada iteração (geração), indivíduos pais são selecionados na população para trocarem suas informações e gerarem novos filhos. Cada indivíduo é avaliado segundo sua capacidade de solucionar o problema (adaptar-se), mantendo os melhores indivíduos através das gerações e eliminando os piores. O funcionamento de um AE baseia-se nos seguintes princípios: representação, seleção, avaliação, recombinação e mutação.

2.1.1 Representação

Na aplicação de algoritmos de busca, cada solução é uma instância para o problema a ser resolvido e deve carregar consigo algum significado inerente ao problema. A estrutura de uma solução varia de acordo com o problema e pode significar desde a seleção de determinados parâmetros até mesmo a ordem em que tarefas devem ser executadas [2]. Sendo assim, a codificação de soluções pode ser feita de diversas maneiras para o mesmo problema, contudo, os operadores de busca podem ser eficientes para

uma determinada representação e ineficientes para outras o que torna a definição da representação uma tarefa fundamental para a eficiência do algoritmo.

Ao codificar uma solução, deve-se considerar além da estrutura do problema a ser resolvido, os operadores de busca que serão utilizados [38]. Algoritmos Genéticos (AG) usualmente utilizam um vetor de binários para representar uma solução. Já o método Estratégias Evolutivas (EE) utiliza um vetor de valores reais. Considerando os problemas caracterizados por permutação ou ordem, a adoção de uma representação binária e seus operadores tradicionais causaria dificuldades para a busca, uma vez que o alfabeto genético da codificação binária (0 e 1) inviabilizaria representar permutações com mais de dois objetos.

A escolha da representação deve considerar também a complexidade para decodificar a solução no espaço de busca real do problema, visto que nem sempre é possível codificar computacionalmente soluções diretamente no espaço de busca real.

2.1.2 Seleção

Independentemente do número de pais e filhos gerados, a seleção consiste em escolher um subconjunto de indivíduos do conjunto total de indivíduos da população. É o mecanismo utilizado para determinar quais indivíduos serão indicados para reproduzir, assim como determinar quais indivíduos permanecerão na população de uma geração para outra [27].

A seleção pode ser determinística, caso em que para a reprodução, será atribuído um número fixo a cada indivíduo que determinará quantas vezes este será selecionado para reproduzir. Para a sobrevivência, a seleção determinística ordena os indivíduos de uma população pelo seu valor objetivo, ou seja, um valor atribuído ao indivíduo relacionado a sua proposta de solução do problema; então um número fixo dos primeiros indivíduos sobrevivem de uma geração para outra, enquanto o restante é descartado da população num processo chamado de seleção truncada [6].

Outra possibilidade de seleção é a estocástica, situação na qual cada indivíduo da população terá uma probabilidade de ser selecionado para reprodução. Essa probabilidade é uma distribuição proporcional ao *fitness* (medida baseada no valor objetivo que estima a capacidade do indivíduo solucionar o problema) de cada indivíduo [6].

Dentre os métodos de seleção mais conhecidos, destacam-se: torneio, *ranking*, proporcional ao *fitness* e elitista:

- Seleção por torneio: nesse método, um subconjunto de indivíduos (dois ou mais) são selecionados para competirem pelo direito de reproduzir. A disputa é baseada no valor de *fitness* de cada indivíduo. O número de indivíduos selecionados para a disputa é definido por uma constante k.

- Seleção por *ranking*: os indivíduos de uma população são ordenados de forma decrescente usando o valor de *fitness* como critério. Um número x, ($0 < x \leq$ tamanho da população) das melhores posições do *ranking* é selecionado para a próxima geração.
- Seleção proporcional ao *fitness*: também conhecida como seleção por roleta viciada, esse método estocástico de seleção atribui a cada indivíduo uma probabilidade proporcional ao seu *fitness* em relação à soma dos *fitness* de todos os indivíduos da população. Ou seja, quanto melhor o *fitness* do indivíduo, maior a sua fatia na roleta e consequentemente maior a possibilidade desse indivíduo ser selecionado.
- Seleção elitista: nesse método de seleção, um subconjunto dos melhores indivíduos de uma geração não podem ser eliminados junto com a população da sua geração, passando dessa forma à próxima geração e preservando seu genoma. Esse mecanismo garante que o melhor indivíduo da próxima geração seja no mínimo igual ao melhor indivíduo da geração anterior.

2.1.3 Avaliação

O requisito básico para o funcionamento de qualquer método de seleção é a utilização de um mecanismo que possa determinar a qualidade das soluções. É a partir da avaliação da qualidade da solução que indivíduos serão selecionados para reprodução e/ou para sobrevivência. Essa avaliação pode ser realizada submetendo a solução a uma função objetivo que caracteriza-se por ser uma formulação ou modelo do problema para o qual se busca uma solução.

A função de avaliação constitui o elo mais forte entre o algoritmo e o problema real. Esse mecanismo é determinante para a capacidade de um AE encontrar soluções assim como para sua eficiência, ela deve ser capaz de guiar a busca e possibilitar a comparação entre as soluções [39]. Muito cuidado é exigido no design desse elemento dos AEs [27].

2.1.4 Recombinação

A recombinação é uma analogia ao processo de cruzamento em que indivíduos são gerados a partir da troca de informações genéticas de outros dois indivíduos denominados pais. A disponibilidade para o uso de operadores de recombinação exige que uma população seja constituída por mais de um indivíduo, do contrário apenas operadores de busca local (como a mutação por exemplo) seriam possíveis [39].

Em AEs a grande vantagem do uso de operadores de recombinação, reside na possibilidade de duas soluções serem parcialmente constituídas por informações (genes) importantes para a solução do problema, ocasião na qual a geração de um novo indivíduo a

partir da combinação das melhores contribuições que cada solução pai pode dar, signifique a geração de uma solução cuja constituição (conjunto de genes) esteja mais próximo da integralidade de informações importantes que resultem na melhor solução para o problema, ou no pior caso, proporcione um melhor direcionamento para a busca.

Para o bom design dos operadores de recombinação, uma característica desejável é que o problema seja decomponível, ou seja, o problema principal possa ser decomposto em subproblemas que ao serem individualmente resolvidos, suas soluções combinadas contribuam para a solução total do problema [39].

O operador de recombinação mais conhecido e convencionalmente utilizado é o crossover de 1-ponto para representações em que os cromossomos possuam tamanho fixo. Nesse tipo de cruzamento, um ponto de corte é definido aleatoriamente e as informações constantes na *substring* antes do ponto de corte do primeiro pai são combinadas com as informações constantes na *substring* após o ponto de corte no segundo pai para gerar o primeiro filho. Esse processo é repetido para gerar o segundo filho, contudo, utilizando as *substring* não utilizadas em cada pai (*substring* após o ponto de corte do primeiro pai e *substring* antes do ponto de corte do segundo pai). Outros tipos de crossover de ponto para cromossomos de tamanho fixo são:

- Crossover de dois pontos no qual dois pontos de corte são definidos aleatoriamente e os trechos entre eles são trocados para compor outros indivíduos.
- Crossover multi-ponto ou uniforme no qual uma máscara de bits é utilizada para definir de qual pai será obtido cada gente para compor o novo indivíduo.

Existem alguns problemas para os quais a codificação binária não é a mais adequada, por exemplo, quando o problema requer variáveis contínuas com alta precisão ou quando o problema é caracterizado por encontrar o melhor ordenamento de objetos. Para esses casos os operadores convencionais gerariam soluções inválidas que demandariam mecanismos de correção.

Quando o problema é baseado em permutação, operadores especiais devem ser utilizados a fim de que soluções válidas sejam geradas. Para esses problemas alguns dos mais conhecidos operadores de cruzamento são Crossover de Ordem (OX, *Order Crossover*), o Crossover Baseado em Posição (PBX, *Position Based Crossover*) e o Crossover de Ciclo (CX), para os quais é dada especial atenção neste trabalho por tratar-se dos operadores de referência para a RNP (ver Capítulo 4). Eles são descritos nas subseções a seguir.

Crossover de Ordem

No funcionamento deste operador dois pontos de corte são definidos aleatoriamente. Os trechos entre os pontos de corte são copiados dos pais para os filhos e as demais

posições são preenchidas utilizando-se os genes do outro pai a partir do segundo ponto de corte, ignorando os genes já utilizados do primeiro pai. O mecanismo para obter dois filhos (O_1 e O_2) é descrito pelo pseudocódigo do Algoritmo 2.1.

Algoritmo 2.1: OPERADOR DE RECOMBINAÇÃO OX

Entrada: P_1, P_2
Saída: O_1, O_2
1 **início**
2 Define-se aleatoriamente dois pontos de corte. Os genes constantes entre esses dois pontos constituem uma região de corte.
3 Copia-se a região de corte de $P_1(P_2)$ para $O_1(O_2)$ exatamente nas mesmas posições do pai
4 Marque em $P_2(P_1)$ os genes da região de corte de $O_1(O_2)$
5 Preencha os genes de O_1 a partir do segundo ponto de corte com os genes não marcados de P_2 (também a partir do segundo ponto de corte). Considere os cromossomos como *arrays* circulares
6 **fim**

Marcar os genes da região de corte para ignorá-los é um passo que garante que apenas permutações válidas sejam geradas uma vez que evitam a repetição de genes nos cromossomos.

Crossover Baseado em Posição

Consiste em selecionar genes aleatoriamente e considerar suas posições para construir um novo indivíduo. O pseudocódigo do Algoritmo 2.2 descreve o funcionamento do operador PBX.

Algoritmo 2.2: OPERADOR DE RECOMBINAÇÃO PBX

Entrada: P_1, P_2
Saída: O_1, O_2
1 **início**
2 Determine um conjunto de posições aleatórias
3 Copiar os genes dessas posições de $P_1(P_2)$ para as mesmas posições do filho $O_1(O_2)$
4 Marque em $P_2(P_1)$ os genes que foram copiados em $O_1(O_2)$
5 Complete os genes do filho usando os genes de P_2 da esquerda para a direita ignorando aqueles que foram marcados
6 **fim**

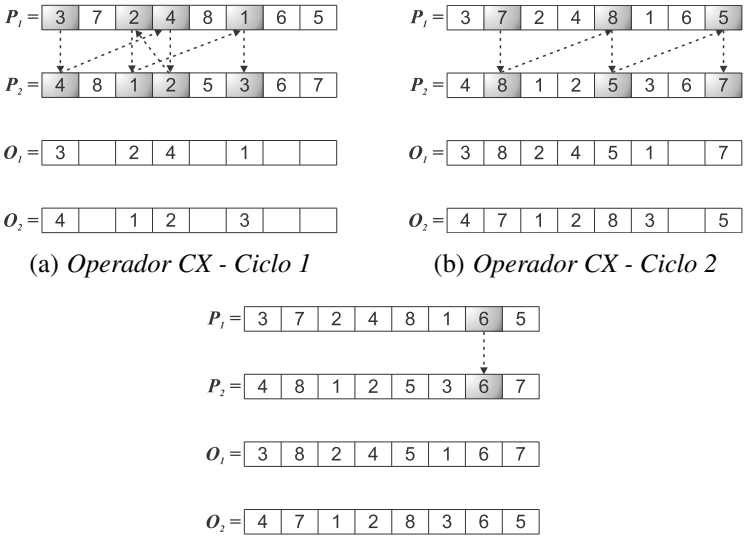
Crossover de Ciclo

O funcionamento do operador CX consiste na identificação de ciclos de genes entre os pais. A construção de uma solução considera a alternância entre os ciclos de genes identificados em cada pai. O pseudocódigo do Algoritmo 2.3 descreve seu funcionamento.

Algoritmo 2.3: OPERADOR DE RECOMBINAÇÃO CX

Entrada: P_1, P_2
Saída: O_1, O_2
1 **início**
2 Identifique o primeiro gene não utilizado de P_1 e copie na mesma posição em O_1
3 Copie para $O_1(O_2)$ o gene correspondente à posição corrente de $P_2(P_1)$ mas na posição em que o mesmo gene se encontra em $P_1(P_2)$
4 Identifique o gene correspondente a posição corrente de $P_2(P_1)$
5 Repita os passos 3 e 4 até o gene da posição corrente de P_2 for igual ao primeiro gene de P_1, concluindo um ciclo
6 Repita todos os passos invertendo a origem dos genes de $O_1(O_2)$
7 Repita todos os passos até concluir a construção das soluções filhas
8 **fim**

A Figura 2.1 ilustra o funcionamento do operador de cruzamento CX. Na Figura 2.1(a) forma-se o primeiro ciclo iniciando-se na primeira posição de P_1 com o gene 3. Na mesma posição em P_2 encontra-se o gene 4. Esses valores são copiados para os respectivos filhos nas mesmas posições. Em seguida, localiza-se em P_1 a posição do último gene de P_2, no caso o gene 4 que em P_1 está na posição 4. Na mesma posição em P_2 encontra-se o gene 2. Ambos são copiados para seus respectivos filhos nas mesmas posições. Em seguida localiza-se em P_1 a posição do último gene de P_2, no caso o gene 2. O gene 2 está na posição 3 em P_1, na mesma posição em P_2 encontra-se o gene 1. Ambos são copiados para seus respectivos filhos nas mesmas posições. Por fim, mais uma vez localiza-se em P_1 a posição do último gene de P_2, no caso o gene 1. Em P_1 o gene 1 encontra-se na sexta posição. Na mesma posição em P_2 encontra-se o gene 3, que é o mesmo gene que iniciou o ciclo. Os genes são copiados em seus respectivos filhos nas mesmas posições e o ciclo é encerrado. Na Figura 2.1(b) inicia-se o segundo ciclo a partir do primeiro gene de P_1 ainda não utilizado, no caso o gene 7. Toda a dinâmica do primeiro ciclo se repete, exceto pelo fato de que dessa vez, os genes são copiados invertendo-se a origem, ou seja, o filho O_1 passa a receber os genes do ciclo de P_2, enquanto o filho O_2 passa a receber os genes do ciclo de P_1. A cada novo ciclo, alterna-se a origem dos genes copiados na construção dos filhos. A Figura 2.1(c), encerra a construção dos filhos copiando os últimos genes restantes.



(a) *Operador CX - Ciclo 1* (b) *Operador CX - Ciclo 2*

(c) *Operador CX - Ciclo 3*

Figura 2.1: *Exemplo de funcionamento do Operador de Cruzamento CX*

2.1.5 Mutação

O mecanismo da mutação nos AEs consiste em provocar perturbações na população com a finalidade de possibilitar a exploração de outros pontos do espaço de busca ou reinserir informações importantes que podem se perder durante o processo de recombinação e seleção. Essas perturbações são provocadas mediante alterações mínimas na constituição dos cromossomos e são realizadas de maneira estocástica, ou seja, a cada cromossomo da população é dada uma probabilidade de submetê-lo a esse mecanismo.

No processo de recombinação, a constituição do cromossomo gerado é fortemente baseado na constituição genética de seus pais. Esse operador juntamente com o operador de seleção faz com que o processo de busca no AE se comporte como uma busca guiada, que pode conduzir a população para um ótimo local. Como a mutação é uma modificação aleatória no cromossomo sem nenhuma referência anterior a não ser um valor válido do alfabeto genético da representação, sua influência no processo de busca é imprevisível tornando-o mais próximo de um processo de busca aleatória. Por esse motivo, convencionalmente a taxa de mutação do AE deve ser baixa o suficiente para causar diversidade na população, mas sem prejudicar o processo de busca guiada.

A mutação funciona de acordo com a representação utilizada. Na representação binária por exemplo, os cromossomos possuem como valor de seus genes um de dois possíveis valores do alfabeto genético da representação (1 ou 0). Nesse caso a mutação consiste em inverter o valor de um ou mais genes escolhidos aleatoriamente [2]. Nas representações de permutações, a mutação deve garantir que o novo indivíduo gerado seja uma permutação válida [9]. Nesse caso, uma forma simples de mutação é a inversão das

posições de dois genes selecionados aleatoriamente.

2.2 Programação Evolutiva

Por volta de 1960, Lawrence J. Fogel desenvolveu a Programação Evolutiva (EP, *Evolutionary Programming*) [2, 12]. Na época, as abordagens em pesquisas na área de Inteligência Artificial eram baseadas geralmente na simulação de redes neurais primitivas e heurísticas. Fogel acreditava que ambas as abordagens eram limitadas pois eram inspiradas no ser humano e não no processo essencial que produz seres de intelecto crescente: a evolução.

Na programação evolutiva encontram-se os principais elementos dos AEs como inicialização, seleção e avaliação. A diferença reside no fato de os indivíduos se reproduzirem apenas através da mutação. Na EP, cada indivíduo da população gera descendentes que são modificados pelo operador de mutação.

2.3 Estratégias Evolutivas

As Estratégias Evolutivas (ES, *Evolutionary Strategies*) [36, 44] são caracterizadas por codificar seus indivíduos através de dois vetores com representação real. Um armazena o valor de cada alelo e o outro armazena o desvio padrão do alelo correspondente [2]. Um operador de mutação com base em uma distribuição Gaussiana com média zero e o desvio padrão correspondente ao desvio padrão de cada gene, é usado para alterar o valor do gene do indivíduo alvo gerando assim um novo indivíduo.

Nas ESs, dois modelos de seleção podem ser aplicados: na primeira representada por $(\alpha+\beta)$ - ES, a população é composta por α genitores e β descendentes com competem entre si para sobreviver à próxima geração composta por α indivíduos. Na segunda representada por (α,β) - ES, os α genitores são substituídos pelos β descendentes de uma geração para outra.

2.4 Programação Genética

Programação Genética (GP, *Genetic Programming*) consiste na implementação de um AE no qual as estruturas de dados que são submetidas à adaptação e evolução, são programas de computador [2, 29]. Nesse paradigma, cada solução é um programa para resolver algum problema. Através da aplicação dos operadores de reprodução, as soluções evoluem e são avaliadas ao serem executadas e terem suas saídas comparadas com as saídas esperadas para o problema. O espaço de busca é formado por programas que são possíveis soluções para o problema.

2.5 Algoritmos Genéticos

Propostos por Holland [17] em meados de 1960, os Algoritmos Genéticos (GA, *Genetic Algorithms*) são caracterizados por possuir todos os mecanismos dos outros paradigmas dos AE tais como seleção, avaliação e mutação. Além desses mecanismos, o que distingue os GAs dos outros AEs é a adoção de operadores de recombinação. Além disso, seus indivíduos codificados através da representação binária no genótipo.

Segundo Goldberg [14], ainda que os parâmetros utilizados em um GA não sejam os melhores, ele continua tendo potencial para encontrar soluções satisfatórias.

2.6 Considerações Finais

Neste capítulo, definiu-se os Algoritmos Evolutivos, que são um paradigma da Inteligência Computacional usados na solução de problemas complexos, cujo funcionamento é baseado na Teoria da Evolução das Espécies.

Os principais mecanismos de funcionamento dos AEs foram apresentados os quais são: representação, seleção, avaliação, recombinação e mutação. Cada um desses mecanismos possui variações que podem relacionar-se ao tipo de problema a ser tratado bem como com a representação utilizada.

No próximo capítulo, será apresentado o conceito de Representação para os AEs, que diz respeito ao design de um AE considerando o problema a ser tratado, os principais critérios a se considerar na elaboração de uma Representação e as principais Representações utilizadas na literatura que são o estado da arte para PPRs. Serão apresentados também os principais PPRs utilizados na literatura para os quais são aplicados os AEs baseados em permutação.

CAPÍTULO 3

Representações de Algoritmos Evolutivos para Problemas de Projeto de Rede

Para uma utilização eficiente e bem sucedida dos AEs, não é suficiente simplesmente aplicar um AE padrão. É necessário encontrar uma representação adequada para o problema e desenvolver operadores de busca apropriados que se ajustem bem às propriedades da representação[38].

Os PPRs são caracterizados por modelar problemas do mundo real relacionados a transporte, distribuição e disponibilidade de recursos. O design de redes é um tópico que captura muitas das questões mais salientes para o planejamento e estratégia que auxiliem na tomada de decisões que signifiquem eficiência e redução de custos[28].

A solução de PPRs é caracterizada pela busca de árvores construídas a partir de grafos, com características singulares o que demanda a utilização de estruturas de dados específicas[9]. A utilização de estruturas convencionais tais como um *string* binário pode não ser eficiente por não considerar as restrições características desse tipo de problema, gerando soluções infactíveis ou com complexidade de tempo que a torna impeditiva.

Nesse capítulo os PPRs, são definidos (Seção 3.1), as propriedades geralmente analisadas em uma representação para PPRs e as principais representações para PPRs são apresentadas (Seção 3.2).

3.1 Problemas de Projeto de Rede

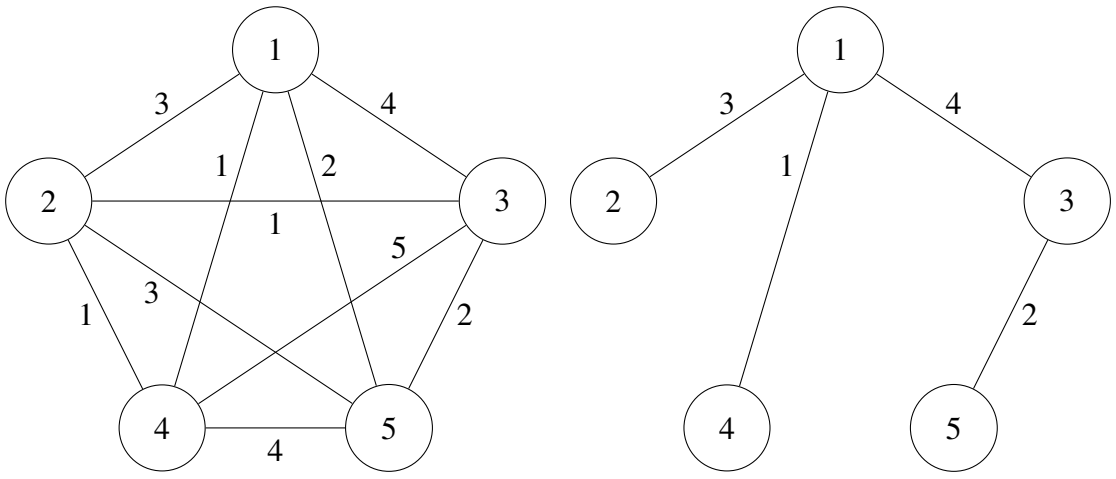
A importância dos PPRs não reside apenas no seu significado prático como ferramenta de design, mas principalmente no grande número de problemas do mundo real que eles modelam. Grande parte dos PPRs alvos das pesquisas em otimização, surgem como versões especiais relativamente próximas dos problemas de projeto de redes do mundo real [28].

A solução de PPRs assume papel fundamental no auxílio a decisões operacionais, estratégicas e táticas que surgem em áreas como transporte, redes de computadores, sistemas de distribuição de energia elétrica, telecomunicações, dentre outros [28]. Para

solucionar esses problemas deve-se considerar restrições como escala, tempo de resposta além de múltiplos critérios como por exemplo limite de orçamento e demandas de capacidade variadas.

Na computação, esses problemas são modelados através de grafos cuja estrutura consiste em um conjunto de vértices (que representam os pontos das redes) e conjunto de arestas que representam as conexões entre os pontos. Dado um grafo não orientado $G = (V, E)$ onde V é o conjunto de vértices e E o conjunto de arestas, há um conjunto de pesos W associados às arestas. Johnson et. al. [21] define um PPR como: dado um grafo $G = (V, E)$, uma função de pesos $L : E \to \mathbb{N}$, um orçamento B e um limitante C para o valor critério $(B, C \in \mathbb{N})$, a solução do PPR consiste em encontrar o subgrafo $H = (V, E_H)$ de G com pesos tais que $\sum_{(i,j)\in E_H} L(i, j) \leq B$ e o valor de critério $f(H) \leq C$, em que $f(H)$ representa a soma dos comprimentos de caminhos mínimos entre todos os vértices em H.

O objetivo ao solucionar um PPR é encontrar um subgrafo otimizado em relação aos critérios de avaliação considerados. Como exemplo de critério considere a minimização dos custos de conexão entre todos os vértices. A otimização pode estar sujeita a restrições como por exemplo um número máximo de arestas incidentes em cada vértice. Em geral, o subgrafo otimizado consiste em uma árvore geradora. Na Figura 3.1, observa-se um grafo que representa um PPR e sua possível solução por meio de uma árvore geradora. Os valores associados as arestas representam o peso de cada uma.



(a) *Exemplo de um Grafo* (b) *Árvore Geradora baseada no Grafo*

Figura 3.1: *Exemplo de solução de um PPR através de uma árvore geradora baseada em um grafo.*

A função de pesos pode significar os custos para interligar dois pontos da rede ou qualquer outro aspecto que possa ser medido nessa conexão. Considerando os pesos das arestas da Figura 3.1, a solução T encontrada teria custo total $C(T) = 10$. Ao considerar aspectos como larga escala e restrições, esses problemas tornam-se computacionalmente complexos e são geralmente classificados na categoria NP-Difícil [28].

Dentre os PPRs que modelam problemas do mundo real destaca-se: árvore geradora mínima com restrição de grau (dc-MST, *degree-constrained Minimum Spanning Tree*) [38], árvore geradora de comunicação ótima (OCST, *Optimal Communication Spanning Tree*) [40] e Árvore Geradora Mínima com Restrição de Diâmetro (BDMST, *Bounded Diameter Minimum Spanning Tree*) [30].

3.2 Representações para PPRs

A definição da representação pode impactar o tempo e a qualidade das soluções de um AE. Uma escolha ruim pode fazer com que o AE se comporte como um algoritmo de busca aleatória [38], inviabilizando uma das principais características dos AEs que é a sua capacidade de realizar uma busca guiada pelas áreas mais promissoras do espaço de busca.

As representações para PPRs podem ser classificadas como diretas, indiretas ou mistas. As diretas são caracterizadas por codificar árvores por meio de seu conjunto de arestas e aplicar operadores de reprodução diretamente ao seu conjunto de arestas[9]. Nessa categoria a representação Conjunto de Arestas (Seção 3.2.5) é uma das mais referenciadas na literatura. As indiretas são caracterizadas por codificar árvores (fenótipo) como uma lista de *strings* (genótipo) e aplicar operadores de reprodução clássicos. Esse é o caso das representações Vetor de Características (Seção 3.2.2), Número de Prüfer (Seção 3.2.3) e Chaves Aleatórias (Seção 3.2.4). As representações mistas possuem características tanto das representações diretas quanto das indiretas utilizando uma função que codifique o fenótipo em genótipo, mas aplicando operadores de reprodução especialmente desenvolvidos para elas. Esse é o caso da Representação Nó-Profundidade (Seção 3.2.6).

Nas próximas Seções as propriedades e as principais representações são apresentadas.

3.2.1 Propriedades das Representações para PPRs

Raidl e Julstrom[35] argumentam que a estrutura de um AE e a configuração de seus parâmetros afetam sua performance, mas o fator determinante para o sucesso ou falha de um AE é sua codificação. Pesquisadores concordam sobre a relevância de algumas características das representações e seus operadores, as quais são apresentadas a seguir:

- Espaço: cromossomos não devem requerer quantidades extravagantes de memória.
- Tempo: a complexidade de tempo para avaliar, recombinar e realizar a mutação deve ser baixa. Quando se trata de árvores geradoras, a avaliação inclui o tempo

necessário para decodificar o cromossomo para identificar a árvore geradora que ele representa.

- Factibilidade: todos os cromossomos, particularmente os resultantes de cruzamento e mutação devem representar soluções factíveis.
- Cobertura: a codificação deve representar todas as soluções possíveis ou ao menos uma solução ótima deve ser alcançável pelo AE e seus operadores.
- Tendência: todos os tipos de soluções devem ser igualmente prováveis de serem alcançados pela representação. A tendência é desejável quando sabe-se previamente informações sobre a solução ótima.
- Localidade: pequenas alterações no cromossomo devem em geral representar soluções similares ao cromossomo original. No caso de árvores geradoras, uma mutação em um cromossomo deve representar uma árvore majoritariamente composta por arestas presentes na árvore original.
- Hereditariedade: soluções oriundas de cruzamento devem representar soluções que combinem as subestruturas das soluções pais. No caso de árvores geradoras, uma prole deve representar uma árvore majoritariamente composta por arestas presentes em seus pais.
- Restrições: a decodificação de cromossomos e a aplicação de operadores de cruzamento e mutação devem ser capazes de respeitar restrições específicas dos problemas como por exemplo restrição de grau em árvores geradoras.
- Hibridismo: os operadores devem ser capazes de incorporar heurísticas relativas ao problema, como por exemplo, favorecer o uso de arestas com menor custo.
- Grafos esparsos: algumas representações podem codificar árvores geradoras de grafos completos. É desejável que a representação possa ser aplicada a subgrafos ou grafos que não contenham todas as arestas possíveis.

Rothlauf[38] afirma que a representação deve ao menos possibilitar a codificação de todas as soluções possíveis de um problema de otimização, e operadores genéticos tais como cruzamento e mutação devem ser aplicáveis. Segundo o autor, outras características de uma representação devem ter seus impactos investigados, tais como: redundância, escalabilidade e localidade.

As Seções que vão de 3.2.2 até 3.2.6 descrevem as representações para árvores PPRs mais conhecidas.

3.2.2 Vetor de Características

Essa representação utiliza um vetor binário como estrutura para armazenar uma árvore geradora. Cada posição do vetor representa uma aresta do grafo e pode armazenar o valor 1 para indicar que a aresta dessa posição é utilizada na construção da árvore,

ou 0 para indiciar o contrário [9]. Dessa forma, o vetor deve ter tamanho suficiente para suportar todas as arestas possíveis de um grafo. A Figura 3.3 exibe o Vetor de Características da árvore geradora da Figura 3.2.

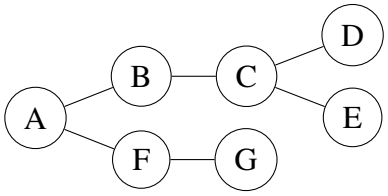


Figura 3.2: *Árvore com 7 vértices.*

1	0	0	0	1	0	1	0	0	0	0
A-B	A-C	A-D	A-E	A-F	A-G	B-C	B-D	B-E	B-F	B-G

1	1	0	0	0	0	0	0	0	1
C-D	C-E	C-F	C-G	D-E	D-F	D-G	E-F	E-G	F-G

Figura 3.3: *Representação CV da árvore da Figura 1.1*

Com essa estrutura, o CV (*Characteristic Vector*) possui características positivas tais como: representar todas as árvores possíveis e alta localidade já que a alteração do valor de uma posição do vetor implica na inclusão ou remoção de uma aresta diretamente na árvore. Por outro lado, o CV requer espaço proporcional ao número de arestas. Em um grafo completo o número de arestas m é dado por $m = n(n-1)/2$ onde n é o número de vértices, e o tamanho do espaço de busca é dado por $2^{n(n-1)/2}$. No entanto, somente uma pequena fração desses cromossomos representam soluções factíveis, uma vez que um grafo completo possui somente n^{n-2} árvores geradoras distintas[35].

Os CVs construídos aleatoriamente podem ser inválidos pela ocorrência de ciclos no grafo representado ou pela geração de grafos não conexos. Esse problema pode ser resolvido aplicando correções como adicionar arestas em grafos não conexos ou remover arestas que formam ciclo no grafo[9]. Outros problemas relacionados ao CV são a alta complexidade de tempo e espaço.

3.2.3 Número de Prüfer

Uma das representações mais conhecidas, é caracterizada por possibilitar a representação de árvores de forma elegante através de uma *string* de tamanho $n-2$ de um alfabeto de n símbolos com correspondência única entre cada *string* e sua árvore, sendo essa *string* denotada como Número de Prüfer [38].

O processo de codificação de uma árvore T começa pela identificação do vértice folha n com menor rótulo entre todas as folhas da árvore. Esse vértice é removido e seu

predecessor é adicionado à primeira posição da *string P*. Esse processo é repetido até que restem apenas dois vértices na árvore. A Figura 3.4 mostra uma árvore e a Figura 3.5 seu respectivo Número Prüfer.

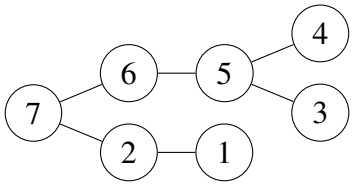


Figura 3.4: *Árvore com 7 vértices.*

2	7	5	5	6

Figura 3.5: *Representação por Número de Prüfer da Árvore da Figura 3.4*

O processo de decodificação de um Número de Prüfer inicia-se pela identificação dos vértices que complementam a *string P*, definindo a *string* $\overline{P}$. Identifica-se em $\overline{P}$ o vértice v_i com o menor rótulo, e em P o vértice v_j mais à esquerda, formando a primeira aresta a ser inserida na árvore T, removendo ambos os vértices de suas *string*. Verifica-se a ocorrência de v_j em P, em caso negativo v_j é inserido em $\overline{P}$. Esse processo é repetido até que não haja mais nenhum vértice em P e restem 2 vértices em $\overline{P}$.

Além de ser capaz de representar todas as árvores possíveis, o Número de Prüfer não possui tendência. Outra vantagem dessa representação é sua baixa complexidade de tempo $O(nlogn)$ [49].

Dentre os fatores negativos dessa representação está a baixa localidade, uma vez que uma pequena alteração na *string* pode gerar indivíduos muito diferente dos pais. Outro ponto negativo é a sua inviabilidade para grafos esparsos já que seu processo de codificação e decodificação considera grafos completos.

3.2.4 Chaves Aleatórias

A representação Chaves Aleatórias (NetKeys, *Network Random Keys*) é similarmente e uniformemente redundante, e pertence à classe das representações ponderadas que diferentemente do CV (ver Seção 3.2.2) (onde cada aresta somente pode ser indicada se está presente ou não - 1 ou 0), utiliza valores reais entre 0 e 1 para indicar a importância de cada aresta [38]. Cada índice do vetor representa uma aresta e cada valor real associado ao índice indica a importância da aresta.

A representação NetKeys não gera soluções infactíveis e possui alta localidade. Contudo sua complexidade de tempo e espaço são superiores se comparadas a outras representações.

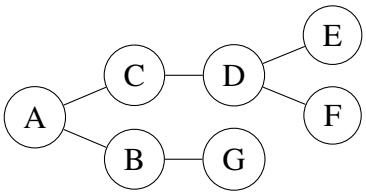
3.2.5 Conjunto de Arestas

Raidl e Julstrom [35] propuseram representar árvores geradoras diretamente pelo seu conjunto de arestas. Essa representação utiliza um vetor ou tabela de dispersão (*hash*) cujas entradas são os pares de vértices que definem cada aresta.

A geração da população inicial de cromossomos se dá através da extensão dos conhecidos algoritmos de Prim [34] e Kruskal [26] para construção de árvores geradoras, nessa representação chamados de PrimRST e KruskalRST. As operações de recombinação também são baseadas nesses algoritmos.

A implementação da representação Conjunto de Arestas (*Edge-Sets*) através de tabela de dispersão possibilita operações de inserção, remoção e busca em tempo constante. Tanto a implementação por tabela de dispersão quanto por vetor possui complexidade $O(n)$, onde n é o número de vértices do grafo [9].

A Figura 3.6 mostra uma árvore representada através do Conjunto de Arestas.



{(A,B), (B,G), (A,C), (C,D), (D,E), (D,F)}

Figura 3.6: *Árvore e sua representação por Conjunto de Arestas.*

3.2.6 Nó-Profundidade

A RNP [9] consiste em um vetor que armazena a dupla de valores (de_i, v_i), onde de_i representa a profundidade do vértice $v_i \in \{1, 2...,n\}$ onde n é a quantidade de vértices da árvore. O processo para codificar uma árvore usando a RNP consiste percorrê-la usando algum método de busca, como por exemplo a busca em profundidade (DFS, *Depth-First-Search*), armazenando para cada vértice seu rótulo e profundidade. A Figura 3.7 mostra uma árvore e a Figura 3.8 sua correspondente representação RNP.

Dois operadores de mutação desenvolvidos especificamente para a representação podem ser aplicados gerando novas soluções sempre factíveis. Eles operam podando uma árvore (chamada de T_{origem}) e transferindo a parte podada para outra árvore (chamada de $T_{destino}$). Os operadores são conhecidos como PAO e CAO e a diferença entre eles é que o operador PAO preserva o vértice da subárvore podada em T_{origem} como raiz, enquanto o CAO altera o vértice raiz escolhendo outro vértice da parte podada antes de conectá-lo à $T_{destino}$. As Seções 3.2.6 e 8 descrevem o funcionamento dos operadores PAO e CAO respectivamente.

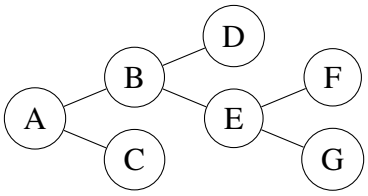


Figura 3.7: *Árvore com sete vértices.*

i	1	2	3	4	5	6	7
de	0	1	2	2	3	3	1
v	A	B	D	E	F	G	C
deg	2	3	1	3	1	1	1

Figura 3.8: *Representação NDE da árvore da Figura 3.7.*

Operador PAO

Como preserva o vértice raiz da árvore podada, o operador PAO causa poucas alterações na árvore. O PAO pode ser aplicado a florestas e também em uma árvore única. Para a aplicação em uma única árvore o pseudocódigo do Algoritmo 3.1 descreve o funcionamento do operador.

Algoritmo 3.1: OPERADOR DE MUTAÇÃO PAO

Entrada: T_{origem}
Saída: $T_{destino}$
1 **início**
2 Um vértice p é escolhido aleatoriamente na árvore T_{origem}
3 Um vértice a aleatoriamente em T_{origem} desde que não esteja dentro da subárvore enraizada no vértice p
4 Determina-se o intervalo (v_p - v_l) que representa a subárvore podada v_p em T_{origem}. O vértice v_l é o último vértice dessa subárvore. Os vértices v_i da subárvore podada são maiores que v_p (v_i ¿ v_p) e a profundidade de v_i é maior que a profundidade de v_p (de_i ¿ de_p)
5 Copie o intervalo compreendido entre o primeiro vértice de T_{origem} até o vértice a para $T_{destino}$
6 Copie as posições de v_p até v_l (a subárvore enraizada no vértice p) de T_{origem} para $T_{destino}$ a partir da posição $i_a + 1$ e atualize os valores de profundidade $de_i = de_i - de_{i_p} + de_{i_a} + 1$
7 Complemente $T_{destino}$ com as posições de T_{origem} a partir da posição $i_a + (i_l - i_p) + 2$
8 **fim**

Operador CAO

Diferentemente de PAO, o operador CAO não preserva o vértice v_p como raiz da árvore podada, escolhendo aleatoriamente outro vértice diferente de v_p no intervalo (v_p - v_l) para ser a nova raiz, chamado de v_r. Dessa forma, a ordem dos vértices da subárvore podada é alterada. Primeiramente armazena-se em um vetor temporário a subárvore enraizada no vértice v_r. Em seguida, armazena-se a subárvore enraizada no primeiro vértice anterior a v_r (v_r - 1), percorrendo e armazenando no vetor temporário em sentido inverso até atingir o vértice v_p. Por fim, transfere-se a subárvore do vetor temporário para $T_{destino}$ na posição (v_a + 1) e atualiza-se os valores de profundidade dos vértices.

Operador de Recombinação NOX

Baseado no operador OX (ver Seção 2.1.4, o NOX pode ser formulado de acordo com o pseudocódigo do Algoritmo 3.2. As principais diferenças entre os operadores NOX e OX são as seguintes:

- No NOX, as duas primeiras posições da RNP possuem valor fixo para a profundidade. A primeira posição deve ser sempre 0 e a segunda posição sempre 1.
- No NOX, o filho é preenchido a partir da primeira posição considerando as posições externas à região de corte. No OX o filho é preenchido a partir da primeira posição após o segundo ponto de corte. Com isso, evitam-se RNPs que correspondam árvores com raízes diferentes, garantindo também as profundidades 0 e 1 para as duas primeiras posições do cromossomo.

Considere P_1 e P_2 as RNPs dos pais, O_1 e O_2 as RNPs dos filhos, A_1 e A_2 *arrays* auxiliares.

Algoritmo 3.2: OPERADOR DE RECOMBINAÇÃO NOX

Entrada: P_1, P_2

Saída: O_1, O_2

1 **início**

2 Determine de forma aleatória dois pontos de corte

3 Marque em $A_2(A_1)$ os vértices da região entre os pontos de corte de $P_1(P_2)$

4 Preencha os genes de $O_1(O_2)$ com os genes de $P_2(P_1)$ ignorando os genes marcados em $A_1(A_2)$ até o primeiro ponto de corte em O_1. Para o gene da primeira posição atribua profundidade 0, e para o gene da segunda posição profundidade 1. Aplique o processo de correção de profundidade se necessário.

5 Copie os genes da região entre os pontos de corte de $P_1(P_2)$ para a região de corte de $O_1(O_2)$. Aplique o processo de correção de profundidade se necessário.

6 Preencha dos genes de $O_1(O_2)$ a partir do segundo ponto de corte com os genes de $P_2(P_1)$ a partir da última posição utilizada no Passo 4, ignorando os vértices marcados em $A_1(A_2)$. Aplique o método de correção de profundidade se necessário.

7 **fim**

Para evitar a geração de soluções inválidas, durante a recombinação com NOX pode ser necessário aplicar a correção de profundidade (ver Passos 4, 5 e 6 do Algoritmo). Exceto as duas primeiras posições cujas profundidades são fixas, os demais genes ao serem copiados dos pais devem preservar suas profundidades. No entanto, quando verifica-se que a profundidade do gene copiado viola a estrutura da árvore, ou seja, quando a diferença entre a profundidade do vértice copiado e a profundidade do vértice anterior é superior a 1 ($de_i - de_{i-1} > 1$), aplica-se a correção que consiste em atribuir ao vértice i profundidade igual à profundidade do vértice anterior + 1 ($de_i = de_{i-1} + 1$).

Par ilustrar o funcionamento do NOX, considere os exemplos de RNPs P_1 e P_2 das Figuras 3.9. As árvores representadas por esses cromossomos são ilustradas nas Figuras 3.10. O primeiro ponto de corte foi definido entre as posições 2 e 3 e o segundo ponto de corte entre as posições 6 e 7 (linha 2 do Algoritmo 3.2). Dessa forma os genes da região de troca são os das posições 3, 4, 5 e 6.

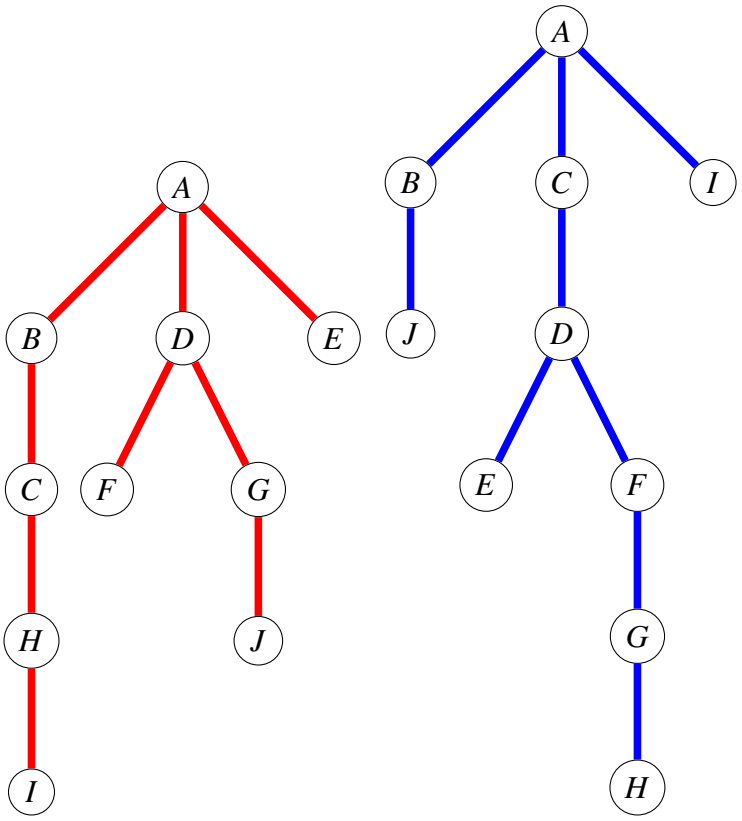
i	1	2	3	4	5	6	7	8	9	10
de	0	1	2	3	4	1	2	2	3	1
v	A	B	C	H	I	D	F	G	J	E

(a) *RNP de* P_1

i	1	2	3	4	5	6	7	8	9	10
de	0	1	2	1	2	3	3	4	5	1
v	A	B	J	C	D	E	F	G	H	I

(b) *RNP de* P_2

Figura 3.9: *RNPs dos Pais*



(a) *Árvore de* P_1 (b) *Árvore de* P_2

Figura 3.10: *Árvores codificadas pelas RNPs da Figura 3.9.*

Os vértices das regiões de corte são então marcados nos *arrays* auxiliares A_1 e A_2 para que sejam ignorados na construção dos filhos O_1 e O_2 (linha 3 do Algoritmo 3.2). No *array* auxiliar A_1 os vértices C, D, E, J referentes à região de corte de P_2 são marcados enquanto no *array* auxiliar A_2 os vértices C, D, H, I referentes à região de corte de P_1 são marcados. A Figura 3.11 mostra como ficariam os *array* auxiliares.

A	B	C	D	E	F	G	H	I	J
		X	X	X					X

(a) *Array auxiliar* A_1 *do NOX*

A	B	C	D	E	F	G	H	I	J
		X	X				X	X	

(b) *Array auxiliar* A_2 *do NOX*

Figura 3.11: *Arrays auxiliares do NOX*

Para obter O_1 copia-se as primeiras posições de P_2 até o primeiro ponto de corte, ou seja, as posições 1 e 2 atribuindo-se as profundidades 0 e 1 respectivamente (linha 4 do Algoritmo 3.2). O próximo passo é a cópia dos genes da região de corte em P_1, ou seja, os genes das posições 3 a 6 (linha 5 do Algoritmo 3.2). Para finalizar, as posições 3, 6, 7 e 8 de P_2 que correspondem aos genes não utilizados de P_2 ignorando os marcados genes marcados em A_2 são copiadas para O_1. O mesmo mecanismo é aplicado para se obter O_2.

As RNPs da Figura 3.12 ilustra O_1 e O_2 como resultado da aplicação de NOX a P_1 e P_2. A Figura 3.13 representa as árvores codificadas por O_1 e O_2.

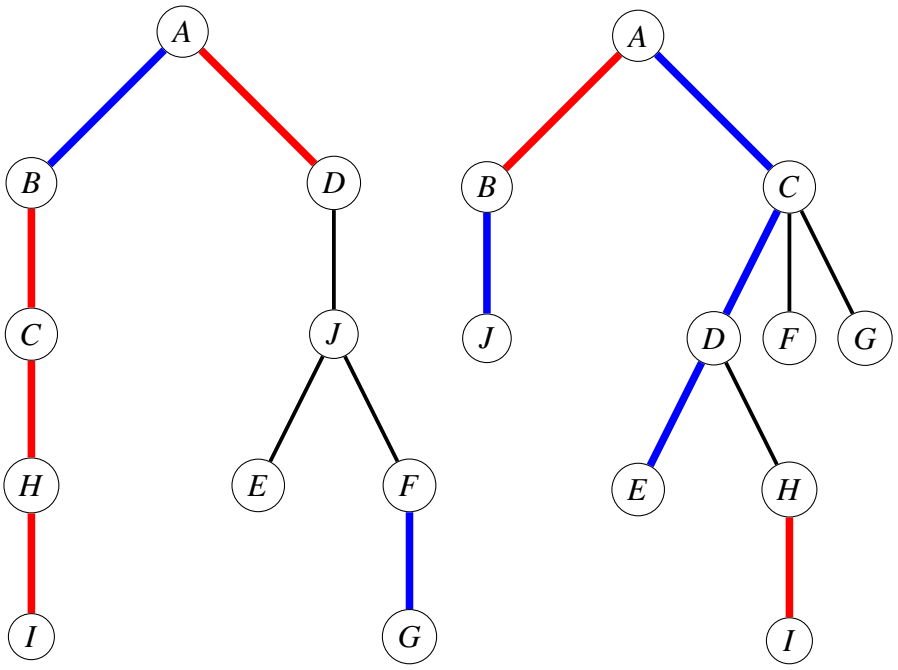
i	1	2	3	4	5	6	7	8	9	10
de	0	1	2	3	4	1	2	3	3	4
v	A	B	C	H	I	D	J	E	F	G

(a) *RNP de* O_1

i	1	2	3	4	5	6	7	8	9	10
de	0	1	2	1	2	3	3	4	2	2
v	A	B	J	C	D	E	H	I	F	G

(b) *RNP de* O_2

Figura 3.12: *RNPs dos Filhos*



(a) *Árvore de* O_1 (b) *Árvore de* O_2

Figura 3.13: *Árvores codificadas pelos filhos* O_1 *e* O_2 *resultantes da aplicação de NOX.*

A arestas azuis e vermelhas representam arestas oriundas de um dos pais e constituem a maior parte da formação dos filhos. As arestas pretas são arestas que não estavam presentes em nenhum dos pais.

Operador de Recombinação NPBX

Tendo como base o operador PBX (ver Seção 6), o NPBX seleciona um conjunto de posições aleatórias e copia os genes dessas posições do pai $P_1(P_2)$ para o filho $O_1(O_2)$. Em seguida, o NPBX remove os genes das posições de um pai $P_1(P_2)$ no outro pai $P_2(P_1)$ e complementa as posições não preenchidas do filho $O_1(O_2)$ com os genes restantes do outro pai $P_2(P_1)$. Assim como em NOX, a diferença entre o NPBX e o PBX é que as profundidades das duas primeiras posições são fixas (0 e 1 respectivamente) e faz-se

necessário verificar a necessidade de aplicar o mecanismo de correção de profundidade. O pseudocódigo do Algoritmo 3.3 descreve o funcionamento desse operador.

Algoritmo 3.3: OPERADOR DE RECOMBINAÇÃO NPBX

Entrada: P_1, P_2
Saída: O_1, O_2
1 **início**
2 Determine de forma aleatória as posições a serem copiadas do pai $P_1(P_2)$
3 Marque em $A_2(A_1)$ os vértices de $P_1(P_2)$ que correspondem aos genes das posições selecionadas na linha 2 em $P_1(P_2)$
4 Preencha os genes de $O_1(O_2)$ com os genes das posições selecionadas de $P_1(P_2)$ e as demais posições preenchidas com os genes de $P_2(P_1)$ ignorando os genes marcados em $A_1(A_2)$. Para o gene da primeira posição atribua profundidade 0, e para o gene da segunda posição profundidade 1. Aplique o processo de correção de profundidade se necessário.
5 **fim**

Para ilustrar o funcionamento do NPBX, considere os mesmos cromossomos pais do exemplo de aplicação do NOX reproduzidos na Figura 3.14 para facilitar o acompanhamento. As posições escolhidas são 2, 4, 6, 9 e 10.

i	1	2	3	4	5	6	7	8	9	10
de	0	1	2	3	4	1	2	2	3	1
v	A	B	C	H	I	D	F	G	J	E

(a) *RNP de* P_1

i	1	2	3	4	5	6	7	8	9	10
de	0	1	2	1	2	3	3	4	5	1
v	A	B	J	C	D	E	F	G	H	I

(b) *RNP de* P_2

Figura 3.14: *RNPs dos Pais*

Na linha 3 do Algoritmo 3.3, as posições selecionadas são marcadas nos *array* auxiliares. Em A_1 são marcadas as posições selecionadas em P_2 que são: B, C, E, H e I. Em A_2 as posições selecionadas em P_1 (B, H, D, J e E) são marcadas. A Figura 3.15 ilustra como ficaram os *array* auxiliares.

A	B	C	D	E	F	G	H	I	J
	X	X		X			X	X	

(a) *Array auxiliar* A_1 *do NPBX*

A	B	C	D	E	F	G	H	I	J
	X		X	X			X		X

(b) Array *auxiliar* A_2 *do NPBX*

Figura 3.15: Arrays *auxiliares do NPBX*

O primeiro filho O_1 começa a ser construído inserindo os genes selecionados de P_1 nas mesmas posições em O_1 (linha 4 do Algoritmo 3.3). Dessa forma os seguintes

genes serão inseridos: B, H, D, J, E nas posições 2, 4, 6, 9 e 10 respectivamente em O_1. Em seguida, as demais posições de O_1 serão preenchidas da esquerda pra direita com os genes de P_2 ignorando as posições marcadas em A_2. Sendo assim, os seguintes genes A, C, F, G, e I de P_2 serão copiados para as posições 1, 3, 5, 7 e 8 em O_1. O mesmo processo é utilizado para a construção de O_2. Ao preencher a posição 4 de O_1 com a posição 4 de P_1, 7 de O_1 com a posição 8 de P_2 e a posição 9 de O_1 com a posição 9 de P_1, verifica-se a necessidade de aplicação de correção das profundidades que passam respectivamente a $de_4 = 3$ para $de_4 = 2$; $de_8 = 4$ para $de_7 = 2$ e $de_9 = 3$ para $de_9 = 2$. Em O_2 o mecanismo de correção de profundidade foi aplicado na posição 9 ao copiar a posição 9 de P_2. A profundidade foi alterada de $de_9 = 5$ para $de_9 = 4$. A Figura 3.16 mostra as RNPs de O_1 e O_2 após a aplicação de NPBX. As posições destacadas em negrito indicam onde foi necessário aplicar a correção de profundidade. A Figura 3.17 mostra as árvores codificadas pelas RNPs de O_1 e O_2.

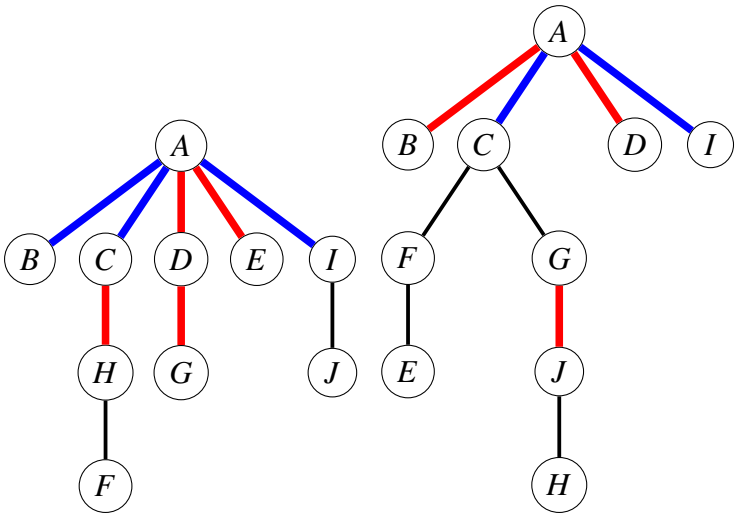
i	1	2	3	4	5	6	7	8	9	10
de	0	1	1	**2**	3	1	**2**	1	**1**	2
v	A	B	C	**H**	F	D	**G**	I	**J**	E

(a) *RNP de* O_1

i	1	2	3	4	5	6	7	8	9	10
de	0	1	1	1	2	3	2	3	**4**	1
v	A	B	D	C	F	E	G	J	**H**	I

(b) *RNP de* O_2

Figura 3.16: *RNPs dos Filhos resultantes da aplicação de NPBX*



(a) *Árvore de* O_1 (b) *Árvore de* O_2

Figura 3.17: *Árvores codificadas pelos filhos* O_1 *e* O_2 *resultantes da aplicação de NPBX.*

Apesar de serem topologicamente diferente dos pais, observa-se que o os filhos são majoritariamente compostos pelas arestas dos pais (arestas em azul e vermelho). Em

testes empíricos realizados em [9] sobre a hereditariedade dos operadores NOX e NPBX tendo como critério o número de arestas dos filhos que não são encontradas nos pais, os resultados mostraram que para grafos com 10 vértices, a hereditariedade do NOX é muito alta indicando número de arestas dos filhos não encontradas nos pais próximo de zero. Conforme o número de vértices é aumentado, a hereditariedade de NOX tende a estabilizar próximo de 20%. A hereditariedade do NBPX é levemente inferior ao NOX e o mesmo comportamento é observado quando se aumenta o número de vértices.

A complexidade média de tempo da RNP é $O(n)$ e foi demonstrada teoricamente [11].

3.3 Considerações Finais

Neste capítulo, foram definidos os problemas de projeto de redes e as principais representações para esses no contexto de algoritmos evolutivos. Além disso, foram destacadas as propriedades geralmente analisadas em uma representação. No próximo capítulo, a proposta desse trabalho, bem como os materiais e métodos utilizados serão definidos.

CAPÍTULO 4

Materiais e Métodos

A combinação entre os operadores de mutação e recombinação é determinante para o desempenho de Algoritmos Evolutivos (AEs) para alguns problemas complexos. A recombinação é o processo responsável por combinar características de soluções existentes para gerar novas soluções, sendo assim esse operador não insere novas informações na população desempenhando o papel de explorar as melhores características já existentes na população. Seu uso exclusivo como operador de busca em AEs reduz a diversidade das soluções e pode levar a convergência prematura [39].

Por outro lado, a mutação desempenha o papel de inserir novas informações nos indivíduos proporcionando diversidade na população. O uso moderado da mutação contribui para a exploração local do espaço de busca. Os operadores de mutação e recombinação são portanto complementares. O uso combinado e controlado dessas operações é estratégico para o sucesso a busca [39].

O conceito de hereditariedade explicita que operações de cruzamento devem criar novas soluções que possuam propriedades similares às soluções utilizadas nessa operação [38]. A propriedade hereditariedade desempenha papel importante em uma representação ao possibilitar que a busca seja guiada pela combinação das melhores características de cada solução. Como essa propriedade diz respeito às informações presentes nos pais que são passadas para os filhos, ela é medida durante a aplicação dos operadores de cruzamento. A hereditariedade não é considerada para a mutação, pois nela as alterações são realizadas aleatoriamente considerando somente o alfabeto genético da representação. A utilização de métricas apropriadas pode indicar o quanto uma representação é aderente à propriedade hereditariedade e o quanto as soluções filhas adquirem características importantes das soluções pais [38].

Em [38], Rothlauf descreve os princípios e recomendações de Radcliffe para o *design* apropriado de operadores de busca para uma representação. Radcliffe introduz o princípio *formae* como um subconjunto do espaço de busca definido como classes de equivalência induzidos por um conjunto de relações equivalentes. Por exemplo, em um espaço de busca composto por faces, uma relação de equivalência poderia ser ”mesma cor de cabelo”ou ”mesma cor dos olhos”. Essas relações de equivalência induzem aos *formaes*

simples ”cabelo preto”, ”cabelo vermelho”, ”olhos azuis”. *Formaes* de mais alta ordem poderiam ser então construídos como ”cabelo preto e olhos azuis”através da composição de *formaes* simples. O espaço de busca composto por todas as faces poderia ser construído então através de *strings* de genes que representam os diferentes *formaes* e operadores de busca poderiam ser construídos baseados nos *formaes* definidos.

A RNP é uma representação para AEs que se aplica a PPRs (ver Seção 3.2.6). A RNP utiliza além dos operadores de mutação PAO e CAO, operadores de recombinação baseados em permutação chamados de NOX e NPBX [9]. A RNP tem se mostrado eficiente para vários PPRs tanto em tempo computacional quanto em qualidade de soluções.

No entanto, durante a aplicação dos operadores de recombinação NOX e NPBX, pode haver a necessidade de realização de correções das profundidades dos genes nos filhos sendo gerados, com o objetivo de garantir a sua factibilidade. Em testes apresentados em [9] e em [8], a hereditariedade dos operadores de recombinação NOX e NPBX diminui à medida em que são submetidos a instâncias maiores. Um outro comportamento apresentado pelos operadores NOX e NPBX é a tendência para árvores tipo estrela. Essa tendência e a queda da hereditariedade para problemas maiores, pode ser causada pela aplicação do método de correção utilizado nos operadores. A tendência não pode ser considerada um fator indesejável nos operadores, uma vez que aplicados a problemas cuja solução é tendenciosa a uma determinada estrutura, AEs cujos operadores possuem tendência similar podem tirar proveito dessa característica reduzindo o tempo de busca e/ou aumentando a qualidade da solução. Contudo, para o caso de problemas em que não se tem conhecimento da estrutura das melhores soluções, é desejável a utilização de operadores que possam gerar todos os tipos de estruturas com a mesma probabilidade, ou seja, sem tendência.

A primeira proposta deste trabalho é o desenvolvimento de um novo operador de recombinação baseado em permutação. Para isso, uma pesquisa foi realizada pelos operadores baseados em permutação mais conhecidos. O operador de recombinação baseado em ciclos (CX) foi selecionado como referência no desenvolvimento desse novo operador e duas versões foram desenvolvidas: uma utiliza o mecanismo correção de profundidade proposto em NOX e NPBX e outra versão com o método de correção de profundidade proposto neste trabalho.

Outra proposta deste trabalho é o desenvolvimento de um novo método de correção de profundidade que implique na melhoria da propriedade hereditariedade (ver Seção 3.2) em relação a profundidade dos vértices. O novo método de correção chamado SDF (ver Seção 4.2) é aplicado aos operadores NOX, NPBX e NCX.

Todos os operadores foram avaliados para verificação de tendência e hereditariedade considerando os dois métodos de correção da profundidades. No caso dos ope-

radores NOX e NPBX, os testes para as mesmas propriedades foram refeitos para fins de comparação com as propostas deste trabalho. A definição de cada propriedade e das métricas utilizadas é apresentada na Seção 4.4.

Testes complementares foram executados com o objetivo de determinar comportamentos dos operadores que ajudassem a compreender os resultados obtidos na análise de propriedades. A definição de cada teste complementar é apresentada na Seção 4.4.3.

Além da análise das propriedades dos operadores, foram desenvolvidos AEs com os operadores NOX-SDF, NPBX-SDF, NCX e NCX-SDF para problema de árvore geradora mínima com restrição de diâmetro (BDMSTP). A definição do BDMSTP é apresentada na Seção 4.3.

4.1 Operador de Recombinação NCX

A codificação de soluções através de permutação de inteiros é uma das formas mais populares e eficientes de representação para AEs aplicados a problemas de otimização combinatória [1]. Problemas de agenda de tarefas, otimização de rotas tais como o Problema do Caixeiro Viajante utilizam em geral a permutação de inteiros para representar suas soluções [1]. Nesse tipo de codificação, a ordem em que os genes são dispostos no cromossomo é importante para o cálculo de seu valor objetivo, razão pela qual esses operadores também são conhecidos como operadores baseados em ordem.

Durante o processo de recombinação de soluções codificadas através de permutação, a troca de informações entre os indivíduos envolvidos pode se dar de várias maneiras. O funcionamento dos operadores de cruzamento baseados em permutação, tem por princípio básico a geração de novas soluções válidas, ou seja, novas permutações baseadas na troca de informações entre indivíduos que gerem novas permutações factíveis e cuja constituição possa ser mapeada através dos pais. Permutações válidas podem ser verificadas por exemplo, ao garantir que nenhum gene é encontrado mais de uma vez no mesmo cromossomo.

Na RNP, a codificação dos indivíduos é realizada através de um percurso em árvore como uma busca em pré-ordem ou em profundidade. Portanto, a ordem do genes é fator determinante para a decodificação. Para cada vértice é armazenada também a sua profundidade (ver 3.2.6). Essas características impõem à RNP que seus operadores considerem as restrições de uma codificação baseada em permutação, como a geração de novas permutações válidas sem a repetição de genes.

Neste trabalho, um novo operador de recombinação para a RNP baseado em permutação foi desenvolvido tendo como referência o operador CX [31] (ver Seção 6) nomeado como NCX (NDE-CX). Dentre as características do CX que inspiraram o desenvolvimento de um novo operador de recombinação para a RNP, está a sua capacidade

de gerar permutações sempre válidas, e também o fato de preservar a posição absoluta dos genes dos pais nos filhos, o que pode ser um fator importante para a propriedade hereditariedade das arestas, considerando que na RNP a ordem dos genes é determinante na decodificação da solução.

Duas versões baseadas no operador CX foram desenvolvidas: na primeira versão (NCX), o método de correção de profundidade desenvolvido para NOX e NPBX é aplicado. Na segunda versão (NCX-SDF), o método de correção baseado em subárvore descrito na Seção 4.2 é aplicado.

O funcionamento do operador NCX é similar ao operador CX. Uma vez que a construção dos ciclos é iniciada sempre a partir das primeiras posições dos vetores, as árvores geradas a partir da aplicação do NCX possuirão sempre os mesmos vértices raízes de seus pais, com profundidade igual a zero. Essa característica do operador NCX o distingue dos operadores NOX e NPBX, já que nestes a atribuição do vértice raiz com profundidade zero das árvores geradas é feita explicitamente utilizando os vértices raízes das árvores pais, sobrepondo a dinâmica de construção original dos operadores OX e PBX nos quais foram inspirados. A partir da segunda posição do vetor, aplica-se a correção de profundidade caso necessário de acordo com as versões desenvolvidas.

Para ilustrar o funcionamento do NCX, considere os exemplos de RNPs P_1 e P_2 da Figura 4.1. As árvores representadas por esses cromossomos são ilustradas na Figura 4.2.

i	1	2	3	4	5	6	7	8	9	10
de	0	1	2	3	4	1	2	2	3	1
v	A	B	C	H	I	D	F	G	J	E

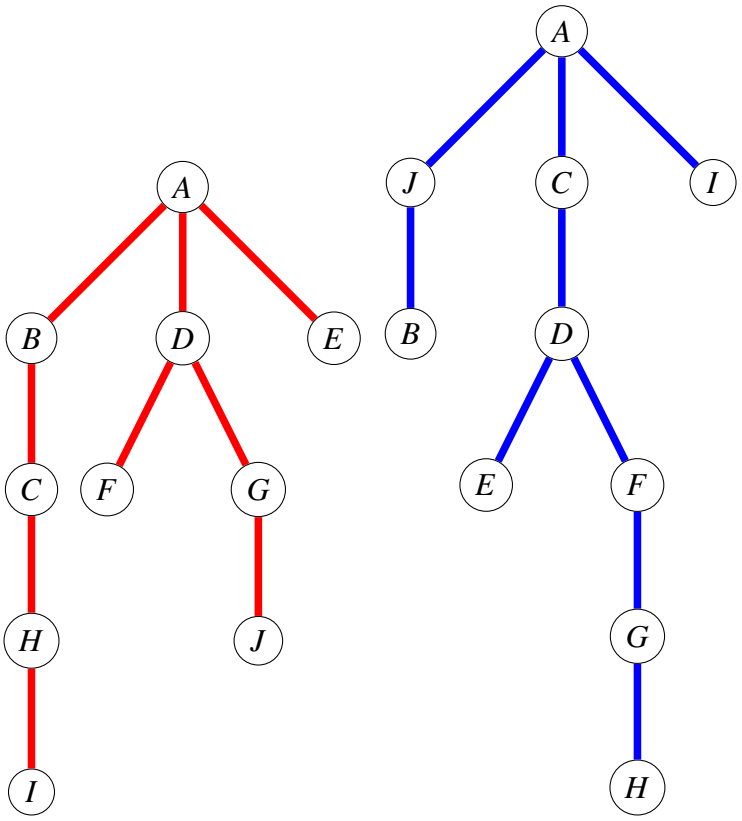
(a) *RNP de* P_1

i	1	2	3	4	5	6	7	8	9	10
de	0	1	2	1	2	3	3	4	5	1
v	A	J	B	C	D	E	F	G	H	I

(b) *RNP de* P_2

Figura 4.1: *RNPs dos Pais*

Conforme o funcionamento do operador CX, a recombinação é realizada através da identificação de ciclos de genes entre os dois pais. O primeiro ciclo inicia-se sempre nas primeiras posições das soluções, no caso do exemplo da Figura 4.1, a primeira posição de P_1 é o gene A. Identifica-se então o gene de P_2 da mesma posição atual de P_1, nesse caso também é o gene A. Copia-se os genes em seus respectivos filhos nas mesmas posições. Como o gene de P_2 é o mesmo que iniciou o ciclo em P_1, o primeiro ciclo é encerrado. O próximo ciclo será identificado e copiado nas soluções filhas porém de maneira invertida, ou seja, O_1 recebe os genes de P_2 enquanto O_2 recebe os genes de P_1 nas mesmas posições. Identifica-se em P_1, a próxima posição ainda não utilizada que é a posição 2 cujo gene é o B. Após identificar a posição do próximo gene em P_1, identifica-se o gene da mesma posição em P_2, no caso é o gene J. Ambos genes B de P_1 e J de P_2 são copiados para as



(a) *Árvore de* P_1 (b) *Árvore de* P_2

Figura 4.2: *Árvores codificadas pelas RNPs da Figura*

soluções filhas invertendo-se a origem. Localiza-se em P_1 a posição do último gene de P_2 que no caso é o gene J. O gene J em P_1 está na posição 9. Na mesma posição em P_2 encontra-se o gene H e ambos são copiados para as soluções filhas preservando a posição dos pais. O processo segue, até que o primeiro gene que iniciou o ciclo (gene B de P_1), seja localizado em P_2, o que fecha o segundo ciclo. Nesse exemplo, o ciclo e as soluções filhas continuariam sendo construídas com os genes H de P_1 e C de P_2, e por fim C de P_1 e B de P_2. O segundo ciclo estaria concluído com a formação parcial das soluções filhas mostradas na Figura 4.3.

i	1	2	3	4	5	6	7	8	9	10
de	0	1	2	1					5	
v	A	J	B	C					H	

(a) *RNP de* O_1

i	1	2	3	4	5	6	7	8	9	10
de	0	1	2	3					3	
v	A	B	C	H					J	

(b) *RNP de* O_2

Figura 4.3: *RNPs dos Filhos após o segundo ciclo*

Concluído o segundo ciclo, o próximo é iniciado na posição do primeiro gene ainda não utilizado em P_1, no caso o gene I da posição 5. O processo é mesmo da formação do segundo ciclo, com a diferença que alterna-se a origem dos genes nos filhos, ou seja,

O_1 passa a receber os genes do ciclo de P_1 novamente, enquanto O_2 recebe os genes de P_2. Essa alternância na origem dos genes permanece a cada conclusão de ciclo até que as soluções estejam completas. Seguindo o exemplo, o segundo ciclo seria formado pelos genes I, D e E de P_1 que serão copiados para O_1 nas mesmas posições de P_1, e os genes D, E, I que serão copiados para O_2 nas mesmas posições de P_2. A Figura 4.4 mostra a nova formação parcial dos filhos O_1 e O_2 após a conclusão do terceiro ciclo.

i	1	2	3	4	5	6	7	8	9	10
de	0	1	2	1	4	1			5	1
v	A	J	B	C	I	D			H	E

(a) *RNP de* O_1

i	1	2	3	4	5	6	7	8	9	10
de	0	1	2	3	2	3			3	1
v	A	B	C	H	D	E			J	I

(b) *RNP de* O_2

Figura 4.4: *RNPs dos Filhos após o terceiro ciclo*

i	1	2	3	4	5	6	7	8	9	10
de	0	1	2	1	4	1	3	2	5	1
v	A	J	B	C	I	D	F	G	H	E

(a) *RNP de* O_1

i	1	2	3	4	5	6	7	8	9	10
de	0	1	2	3	2	3	2	4	3	1
v	A	B	C	H	D	E	F	G	J	I

(b) *RNP de* O_2

Figura 4.5: *RNPs dos Filhos após o último ciclo*

A construção das soluções filhas é concluída através da identificação de mais dois ciclos. O primeiro formado pelos genes F de P_1 e P_2 que como são os mesmos genes nas mesmas posições, iniciam e encerram um ciclo. O último formado pelos genes G de P_1 e P_2 em que a coincidência se repete. A Figura 4.5 mostra as RNPs das soluções concluídas na aplicação do operador NCX.

Após a construção das soluções com a aplicação do operador NCX, as profundidades que as tornam inviáveis devem ser corrigidas, caso existam. Caso a versão utilizada seja a NCX-SDF, a dinâmica descrita para o método SDF é realizada, ou seja, busca-se por uma subárvore com raiz válida a partir do gene inválido identificado. Na Figura 4.5 é possível identificar os genes I, F e H de O_1 e o gene G de O_2 cujas profundidades inviabilizam as soluções. A Figura 4.6 mostra as RNPs das soluções após a aplicação da correção original da RNP.

i	1	2	3	4	5	6	7	8	9	10
de	0	1	2	1	2	1	2	2	3	1
v	A	J	B	C	I	D	F	G	H	E

(a) *RNP de* O_1

i	1	2	3	4	5	6	7	8	9	10
de	0	1	2	3	2	3	2	3	3	1
v	A	B	C	H	D	E	F	G	J	I

(b) *RNP de* O_2

Figura 4.6: *RNPs dos Filhos após aplicação de correção de profundidade*

4.2 Método de Correção de Profundidade - SDF

A RNP possui dois operadores de recombinação. O operador NOX cujo funcionamento é inspirado no conhecido operador OX, e o operador NPBX inspirado no também conhecido operador PBX.

Como mostrou o exemplo de funcionamento do operador NBPX (Ver Seção 3.2.6), durante a construção dos cromossomos dos filhos, houve a necessidade de correção de profundidade com a finalidade de garantir a factibilidade das soluções sendo geradas. A correção aplicada considera a profundidade do vértice anterior de_{i-1} para o cálculo da profundidade do vértice i sendo copiado. Ao realizar a operação $de_i = de_{i-1} + 1$ para determinar a nova profundidade do vértice i, o método atual não considera que possa haver no conjunto de vértices disponíveis ainda não utilizados na construção da solução, algum vértice cuja profundidade mantenha a factibilidade da solução sem a necessidade de sua modificação.

O método de correção de profundidade proposto neste trabalho chamado de SDF (Subtree Depth Fix), diferentemente do método adotado na representação originalmente, é baseado em uma estratégia de ao detectar a infactibilidade da profundidade do gene a ser copiado, buscar no conjunto de genes ainda não inseridos no cromossomo, um gene cuja profundidade não viole a factibilidade. Uma vez que esse gene seja identificado, ele e a sua subárvore tem a sua inserção na solução antecipada para a posição do gene com profundidade inválida. O gene inválido poderá ser então inserido após a antecipação da subárvore válida, com alguma chance de não ter sua profundidade modificada. Dois casos de infactibilidade são tratados pelo método SDF:

- Diferença de profundidade igual a 2: quando a diferença de profundidade entre o gene atual e o anterior é igual a 2, realiza-se a busca por uma subárvore válida e antecipa a sua inserção. Em seguida, realiza-se a inserção do gene inicialmente inválido.
- Diferença de profundidade maior do que 2: quando a diferença de profundidade entre o gene atual e o anterior é maior do que 2, a busca por uma única subárvore válida pode não ser suficiente. A antecipação de uma subárvore válida, pode continuar inviabilizando a inserção do gene detectado como inválido. Nesse caso, verifica-se se o último gene da subárvore válida antecipada viabiliza a inserção do gene inicialmente inválido, caso contrário realiza-se novas buscas e antecipações de subárvores válidas até que o gene inicialmente inválido possa ser inserido.

A hipótese é que esse método possa garantir que a correção seja mais conservadora em relação ao método original por preservar características presentes nos ancestrais, nesse caso a profundidade dos genes.

Para ilustrar o funcionamento de SDF, considere as RNPs P_1 e P_2 da Figura 4.7 e a aplicação do NPBX para geração de O_1 e O_2. Considere como posições sorteadas 2, 4, 6, 9 e 10.

i	1	2	3	4	5	6	7	8	9	10
de	0	1	2	3	4	3	2	2	3	3
v	A	B	C	H	I	D	F	G	J	E

(a) *RNP de* P_1

i	1	2	3	4	5	6	7	8	9	10
de	0	1	2	1	2	3	3	4	5	2
v	A	B	J	C	D	E	F	G	H	I

(b) *RNP de* P_2

Figura 4.7: *RNPs dos Pais*

A Figura 4.8 mostra o indivíduo O_1 infactível após a aplicação NPBX, ou seja sem correção de profundidade. Essa é a ordem em que os genes serão considerados para aplicação de SDF na construção de O_1 factível.

i	1	2	3	4	5	6	7	8	9	10
de	0	1	1	3	3	3	4	2	3	3
v	A	B	C	H	F	D	G	I	J	E

Figura 4.8: *Conjunto de genes elegíveis para* O_1

Na aplicação do NPBX a P_1 e P_2, em O_1 o gene da posição 4 preenchido com o vértice H de P_1, parcialmente construído conforme a Figura 4.9, possui profundidade infactível. A profundidade, factível do vértice H deveria ter o valor 2. Assim, aplica-se SDF para corrigir a posição do par $(H,3)$ de formar a não alterar a sua profundidade.

i	1	2	3	4	5	6	7	8	9	10
de	0	1	1	3						
v	A	B	C	H						

Figura 4.9: *RNPs parcial de* O_1

O método SDF busca por uma subárvore no conjunto de vértices elegíveis para compor O_1 cuja profundidade da raiz permita a inserção do gene do vértice H sem alterar a profundidade. Conforme é possível verificar na Figura 4.8, no conjunto de vértices elegíveis a próxima subárvore cuja profundidade da raiz não invalida a solução, é a subárvore enraizada no vértice I na posição 8 cuja profundidade é 2. Todos os vértices da subárvore enraizada em I são então identificados (I-J-E). A seguir, subárvore enraizada em I é então inserida no lugar do vértice H e todos os genes entre a posição de H e a I são deslocados. Em seguida, tenta-se inserir novamente o vértice H cuja profundidade passa a ser válida sem necessidade de outra correção. A Figura 4.10(a) mostra como ficaria O_1 com o método de correção baseado em subárvore. A Figura 4.10(b) mostra como

ficaria O_1 com o método original. As células preenchidas em vermelho indicam genes cuja profundidade não corresponde à profundidade do mesmo gene em nenhum dos pais.

Ainda que o SDF busque uma subárvore com raiz válida para evitar a correção da profundidade, há a possibilidade de que essa busca não retorne resultados. Nesse caso, o SDF aplica a correção de profundidade original dos operadores da RNP.

i	1	2	3	4	5	6	7	8	9	10
de	0	1	1	2	3	3	3	3	3	4
v	A	B	C	I	J	E	H	F	D	G

(a) O_1 *gerado com método de correção baseado em subárvore*

i	1	2	3	4	5	6	7	8	9	10
de	0	1	1	2	3	3	4	2	3	3
v	A	B	C	H	F	D	G	I	J	E

(b) O_1 *gerado com método de correção original*

Figura 4.10: *RNPs de O_1 com e sem o método de correção baseado em subárvore*

Os operadores NOX e NPBX com o novo método de correção foram nomeados respectivamente como NOX-SDF e NPBX-SDF. O operador NCX desenvolvido neste trabalho, na versão que utiliza o método SDF para correção de profundidade foi nomeado como NCX-SDF.

4.3 Árvore Geradora Mínima com Restrição de Diâmetro

O BDMSTP é um problema clássico de árvore geradora mínima com restrição. Sua restrição está na determinação de um limite para o diâmetro da árvore e pode ser aplicado a projetos de redes de telecomunicação, algoritmos de exclusão mútua distribuída e recuperação de informações [7] [30].

Em um grafo $G = (V, E)$ a distância de um vértice $v \in V$ para um vértice $t \in V$ é determinada pelo número de arestas no menor caminho entre v e t. A maior distância entre um vértice v e qualquer outro vértice é denominada excentricidade do vértice v. O diâmetro de uma árvore é determinado pela excentricidade máxima de todos os seus vértices [7]. Considere o vértice A da Figura 4.11. A distância entre o vértice A e todos os outros vértices folhas do grafo é dado por:

- $A - B = 1$
- $A - I = 3$
- $A - F = 2$
- $A - H = 3$

Logo, a excentricidade do no A é igual a 3. O diâmetro da árvore da Figura 4.11 é determinado pela maior excentricidade entre seus vértices, o que pode ser identificado

através da excentricidade dos vértices I ou H igual a 6, identificado pelo caminho formado pelas arestas em vermelho.

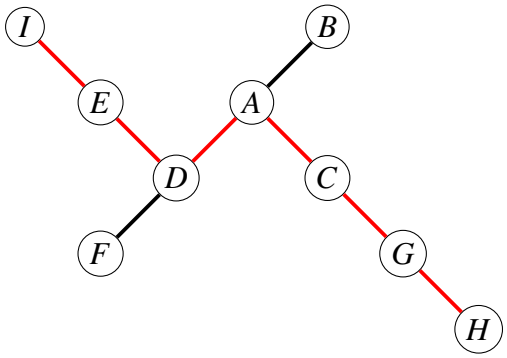


Figura 4.11: *Exemplo de diâmetro de um grafo.*

O BDMSTP estabelece a busca por uma árvore geradora $T \subseteq G$ que minimize

$$min_T \sum_{i,j \in T} w_{v_i,v_j}$$

onde w_{v_i,v_j} é o peso da aresta que conecta os vértices v_i e v_j, submetida a uma restrição de diâmetro $diam(T) \leq k$. Para $diam(T) = 2$, $diam(T) = 3$ ou para árvores cujos pesos das arestas sejam iguais, o BDMSTP pode ser resolvido em tempo polinomial. Para os casos em que $4 \leq diam(T) \leq n-1$, onde n é o número de vértices do grafo, o BDMSTP torna-se NP-Difícil [30].

Neste trabalho, um AE foi desenvolvido para aplicação em instâncias do BDMSTP com cada um dos operadores de recombinação desenvolvidos: NOX-DFS, NPBX-DFS, NCX e NCX-DFS. As instâncias são as mesmas utilizadas em [22, 24, 23, 7] e foram originalmente definidas para o Problema da Árvore de Steiner. As instâncias estão disponíveis em Beasley's OR-Library[1]. Cada operador de recombinação foi submetido a cada uma das seguintes instâncias com a respectiva restrição de diâmetro: $n = 50$ com $k = 5$, $n = 100$ com $k = 10$, $n = 250$ com $k = 15$ e $n = 500$ com $k = 20$.

4.4 Propriedades Analisadas

A análise de propriedades das representações e seus operadores pode identificar o comportamento de um AE e sua capacidade para solucionar um problema. No caso de PPRs, há uma variedade de problemas e as soluções dadas por árvores podem variar em topologia, diâmetro, profundidade dos vértices, grau dos vértices, dentre outras características intrínsecas desse tipo de estrutura.

[1] http://people.brunel.ac.uk/~mastjjb/jeb/info.html

Em [7], uma análise das propriedades da RNP é realizada sobre a inicialização e os operadores de mutação. Em [8], é realizada a análise de propriedades do operador de recombinação NOX da RNP. Neste trabalho, a análise de propriedades foi refeita nos operadores originais NOX e NPBX e aplicada aos novos operadores propostos: NOX-SDF, NPBX-SDF, NCX e NCX-SDF.

A validação dos novos operadores desenvolvidos nesse trabalho é realizada através da análise de duas propriedades: Tendência e Hereditariedade que são apresentadas nas Seções 4.4.1 e 4.4.2.

Os testes para análise de propriedades foram realizados utilizando instâncias de grafos Euclidianos para o problema de Árvores de Steiner disponíveis na biblioteca OR-Library[2]. Nessa biblioteca, as instâncias possuem tamanhos em número de vértices (n) que vão de 10 a 1000. Para cada arquivo $(.txt)$ relacionado ao número de vértices, estão disponíveis 15 instâncias. Há um arquivo para cada tamanho (n) caracterizados por identificar o número de instâncias na primeira linha. Após a primeira linha, o arquivo apresenta as instâncias indicando o número de vértices em uma linha, seguido de n linhas, cada uma contendo um par ordenado (x,y) indicando a posição do vértice no plano cartesiano com dimensões 1.0 x 1.0cm.

Os testes foram realizados para as primeiras 5 instâncias com número de vértices $n = 10, 20, 40, 80, 100$ e 250. Para cada operador de recombinação combinado com método de correção, cada instância foi executada 10 vezes. Com 10 execuções para cada uma das 5 instâncias de cada um dos 6 números de vértices para cada combinação de operador e método de correção em um total de 6, assim, ao todo foram realizadas 1800 execuções.

4.4.1 Análise de Tendência

O conhecimento do comportamento de uma representação e dos seus operadores, constitui uma vantagem na decisão de escolha de um AE para aplicar a um problema para o qual existe uma noção de que tipo de solução deve ser buscada. Quando se tem esse conhecimento, a aplicação de um AE cuja representação e operadores possuam tendência para o tipo de solução a ser buscada, pode causar impacto tanto na qualidade da solução encontrada quanto no tempo de busca. Representações e operadores que não possuam tendência para nenhum tipo de solução, podem ser utilizadas no caso de problemas para os quais não se tem conhecimento de que tipo de solução deve ser buscada.

No caso de problemas de projeto de redes, em geral, as soluções possuem a estrutura de árvores. No contexto de árvores em grafos dois tipos de solução se destacam,

[2] http://people.brunel.ac.uk/~mastjjb/jeb/info.html

similaridade com a árvore geradora mínima e topologia estrela. Assim, a tendência será avaliada para esses dois tipos de estrutura para a solução por meio das seguintes métricas, distância para árvore geradora mínima (d_{mst}) e distância mínima para árvores estrela (d_{star}).

A distância entre duas árvores geradoras T_i e T_j é dada pela distância de Hamming [16], ou seja, o número de arestas diferentes entre as duas árvores. Essa distância é dada por:

$$d_{i,j} = \frac{1}{2} \sum_{u,v \in V, u<v} | \, l^i_{u,v} - l^j_{u,v} \, |,$$

onde $l^i_{u,v}$ é 1 se uma aresta de u para v existe em T_i, e 0 se não existe. A distância mínima de uma solução tipo estrela é calculada por

$$d_{i,STAR} = min(d_{i,star_j})$$

onde $j = 1 \ldots n$ e $star_j$ é a árvore de topologia estrela com centro no vértice j.

Para a análise de tendência, a cada geração foi calculada a média da distância para a árvore geradora mínima da população $d_{mst-pop} = \frac{1}{N} \sum_{i=1}^{N} d_{i,MST}$ onde $d_{i,MST}$ é a distância entre a árvore T_i e a árvore geradora mínima do grafo, e N é o número de indivíduos da população, e a média da distância árvores estrela da população $d_{star-pop} = \frac{1}{N} \sum_{i=1}^{N} d_{i,STAR}$ onde $d_{i,STAR}$ é a distância mínima entre a árvore T_i e todas as árvores estrelas do grafo.

4.4.2 Análise de Hereditariedade

Durante o processo de recombinação de soluções, informações são trocadas entre os indivíduos envolvidos para a geração de novas soluções. Espera-se portanto, que as informações que constituem as novas soluções possam ser mapeadas nos indivíduos originais do processo. Contudo, não se pode garantir que a simples troca de informações entre indivíduos não gere informação nova. No caso das representações para PPRs, a recombinação soluções pode significar a geração de uma árvore não totalmente conectada o que demanda a adoção de mecanismos de correção, que por consequência pode significar o surgimento de uma aresta não mapeável em nenhum dos pais envolvidos[38]. O mesmo pode ocorrer com representações baseadas em permutações, como no caso da RNP. A geração de informações novas durante a recombinação é tolerável desde que não torne essa operação semelhante a uma busca aleatória.

A hereditariedade está relacionada à capacidade de transmissão de informações presentes em soluções pais, para soluções filhas durante a recombinação. Nesse trabalho,

dois aspectos relacionados à hereditariedade dos operadores são analisados: a hereditariedade das arestas (d_{edges}) e a hereditariedade da profundidade dos vértices (d_{depth})

A hereditariedade das arestas é dada pelo número de arestas comuns entre as soluções pais e filhas. Esse cálculo é realizado considerando $E_P = E_{P_1} \cup E_{P_2}$ a união do conjunto das arestas dos pais P_1 e P_2. $E_O = E_{O_1} \cup E_{O_2}$ a união do conjunto de arestas dos filhos O_1 e O_2. Por fim, calcula-se o conjunto de arestas não compartilhadas entre os pais e os filhos por $\mid E_{O-P} \mid = \mid E_O - E_P \mid$. Quanto maior o número de arestas não comuns, menor é a hereditariedade das arestas.

A hereditariedade da profundidade dos vértices é dada pelo número de vértices das soluções filhas cuja profundidade é igual nos mesmos vértices das soluções pais. Essa propriedade calculada em termos de distância de Hamming, é dada por

$$d_{depth} = \frac{1}{2} \sum_{v \in V} d_{v_{O_1}}^{v_{P_1}, v_{P_2}} + d_{v_{O_2}}^{v_{P_1}, v_{P_2}},$$

onde $d_{v_{O_1}}^{v_{P_1}, v_{P_2}}$ é 1 se a profundidade do vértice v do filho O_1 é diferente da profundidade nos pais P_1 e P_2, e 0 se a profundidade do vértice v de O_1 é igual à profundidade em P_1 ou P_2. A mesma definição é aplicada para $d_{v_{O_2}}^{v_{P_1}, v_{P_2}}$, substituindo O_1 por O_2.

A cada geração foi calculada a média da hereditariedade das arestas da população dada por $d_{edges-pop} = \frac{1}{N} \sum_{i=1}^{N} \mid E_{O_i} - E_{P_i} \mid$ onde E_{O_i} é o conjunto de arestas resultantes da união dos filhos gerados na recombinação i, E_{P_i} é o conjunto de arestas resultantes da união dos pais envolvidos na recombinação i e N é o tamanho da população. Também foi calculada a média da hereditariedade da profundidade dos vértices dada por $d_{depth-pop} = \frac{1}{N} \sum_{i=1}^{N} d_{depth}$, onde N é o tamanho da população.

4.4.3 Análises Complementares

Além dos testes de propriedades, outras duas análises complementares foram realizadas com a finalidade de analisar o comportamento dos operadores de recombinação. Essas análises complementares dão indícios que podem ajudar a compreender os resultados obtidos na análise das propriedades. Foi realizada uma contagem do número de correções de profundidade aplicadas pelos operadores originais da RNP (NOX e NPBX) e da versão do operador de recombinação NCX que aplica exclusivamente o mesmo tipo de correção dos operadores originais. Os outros operadores que aplicam a correção de profundidade baseado em subárvore (NOX-SDF, NPBX-SDF e NCX-SDF), também aplicam a correção original quando a busca por uma subárvore válida não retorna resultado. Para esses, foi realizada uma contagem do número de correções aplicadas por cada método. A média de correções aplicadas na população para cada método é dada por $fixes - pop = \frac{1}{N}$

$\sum_{i=1}^{N} fixes(i)$, onde $fixes(i)$ é o número de correções aplicadas pelo método alvo do teste no indivíduo i.

O método de correção de profundidade SDF, não promove alterações nas profundidades dos vértices, pois apenas desloca subárvores de suas posições originais para garantir factibilidade. Todos os operadores preveem o uso do método de correção de profundidade original da RNP, que altera a profundidade do vértice inválido. Os operadores originais (NOX e NPBX) e o operador NCX aplica exclusivamente esse método de correção. Os operadores NOX-SDF, NPBX-SDF e NCX-SDF, aplica o método original como alternativa caso não seja possível aplicar o método SDF.

O último teste determina o impacto em termos de profundidade dos vértices quando o método de correção original é aplicado em todos os operadores, inclusive os que dispõem do método SDF. Para isso, na ocorrência da aplicação do método original, o cálculo dado por $depth^{i}_{change} = depth^{b}_{i} - depth^{a}_{i}$ onde $depth^{b}_{i}$ é a profundidade do vértice i antes da aplicação da correção e $depth^{a}_{i}$ é a profundidade do vértice i após a correção. A média de diferença de profundidade da população é dada por

$$depth^{pop}_{change} = \frac{\sum_{i=1}^{N} \sum_{j=1}^{n} depth_{change}(i,j)}{\sum_{i=1}^{N} fixes(i)}$$

onde $depth_{change}(i,j)$ é a diferença de profundidade antes e depois da aplicação da correção no vértice i do indivíduo j, n é o número de vértices e $fixes(i)$ é o número de vértices corrigidos do indivíduo i.

CAPÍTULO 5

Resultados

Neste capítulo, são apresentados os resultados obtidos pelos testes executados nos operadores de recombinação para a RNP, para análise das propriedades tendência e hereditariedade. Os operadores de recombinação originais da RNP (NOX e NPBX) foram submetidos às mesmas instâncias dos operadores com método de correção SDF desenvolvidos neste trabalho (NOX-SDF, NPBX-SDF), e o novo operador de recombinação com e sem o método de correção SDF (NCX-SDF e NCX).

Além da análise de propriedades, outros testes foram realizados com a finalidade de analisar o comportamento dos operadores. Esses testes não têm relação direta com as propriedades avaliadas, mas podem dar indícios que expliquem os resultados obtidos.

Também neste capítulo são apresentados os resultados de AEs desenvolvidos com alguns dos operadores analisados neste trabalho. Os AEs foram aplicados ao BDMSTP.

5.1 Análise de Propriedades

Ainda que a resolução de problemas complexos através de métodos de busca como os AEs tenha mostrado sua eficiência, a aplicação desse método não é trivial. Um dos fatores mais importantes diretamente relacionados com a qualidade dos resultados e o tempo de busca é a escolha da representação adequada de acordo com o problema a ser resolvido [38]. O processo de decisão da escolha da representação é crítico e o conhecimento das propriedades de uma representação apoia esse processo decisório.

Dentre as diversas propriedades passíveis de análise em representações (ver Seção 3.2.1), a tendência e a hereditariedade estão relacionadas com o potencial do AE para explorar regiões promissoras do espaço de busca e a qualidade das soluções geradas através da recombinação, podendo responder com algum grau de confiança às questões relacionadas à escolha da representação.

A propriedade tendência analisa se, durante o processo de busca, os operadores geram todas as soluções com a mesma probabilidade. Quando uma representação ou operador é tendencioso, significa que existe um probabilidade maior de obter soluções

com determinadas características. Para os operadores de recombinação da RNP, que são utilizados para explorar o espaço de busca de árvores, a análise de tendência terá como objeto soluções com características de árvores com topologia estrela ou da árvore geradora mínima (ver Seção 4.4.1).

A propriedade hereditariedade diz respeito à capacidade dos operadores de recombinação de preservar informações das soluções pais nos filhos. A análise da propriedade hereditariedade dos operadores de recombinação da RNP, é realizada para as arestas e a profundidade dos vértices, ou seja, considerou-se se os filhos preservam as arestas presentes nos pais ou a profundidade que o vértice possuía em um dos pais.

5.1.1 Instâncias e Configurações dos Testes de Propriedades

Para a análise de propriedades, é desejável que a as soluções iniciais possuam uma distribuição uniforme pelo espaço de busca, ou seja sem tendência. Por esse motivo, adotou-se a inicialização por meio do Número de Prüfer (ver 3.2.3), visto que essa representação não possui tendência. Para que a exploração do espaço de busca não seja guiada, ou seja, não haja indicação de qual tipo de solução é melhor, a pressão seletiva não é utilizada. Assim, os indivíduos gerados a cada recombinação substituem os pais e cada pai participa de um único cruzamento.

Os testes para análise de tendência foram executados em instâncias de grafos Euclidianos com número de vértices entre $n = 10$ a $n = 250$ vértices. Cada operador foi executado 10 vezes para cada instância e foram utilizadas 5 instâncias para cada tamanho de grafo.

5.1.2 Resultados dos Testes de Propriedades

Nesta Seção, são apresentados os resultados dos testes das propriedades tendência e hereditariedade aplicados às instâncias descritas na Seção 5.1.1. Com a finalidade de facilitar a comparação entre os diferentes grafos das instâncias, os resultados apresentam as medidas $d_{mst-pop}$, $d_{star-pop}$, $d_{edges-pop}$ e $d_{depth-pop}$ de forma normalizada, ou seja divididos por $n-1$, que é o número arestas presentes nas soluções.

O espaço de busca aumenta conforme aumenta o número de vértices do grafo. Assim, para cada número de vértices n do grafo foi utilizado um tamanho de população diferente. Para os testes, foram utilizados AEs *steady-state* com substituição dos pais pelos filhos. Além disso, optou-se também por variar o número de gerações, de forma a propiciar uma exploração do espaço de busca adequada ao número de vértices do grafo. A Tabela 5.1 sumariza as configurações das execuções para cada tamanho de grafo analisado.

Número de Vértices	Instância	População	Gerações	% Cruzamento	% Mutação
10	1 a 5	50	5000	100	0
20	1 a 5	100	5000	100	0
40	1 a 5	200	5000	100	0
80	1 a 5	400	5000	100	0
100	1 a 5	500	10000	100	0
250	1 a 5	1250	10000	100	0

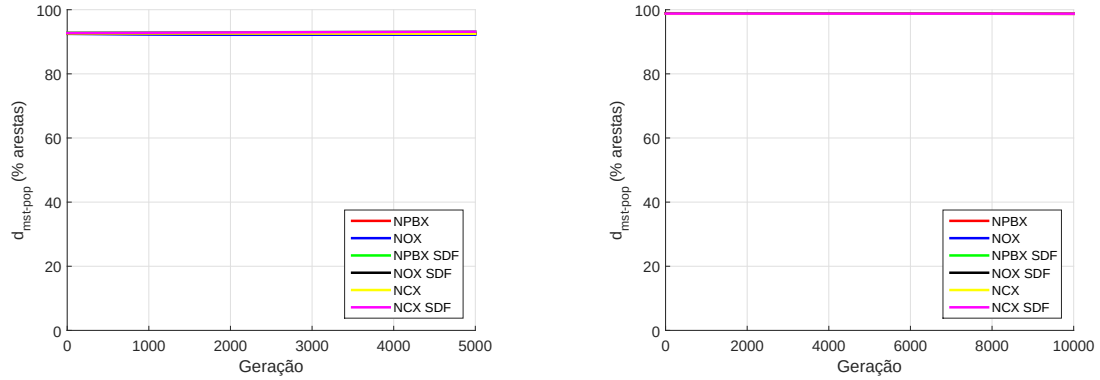
Tabela 5.1: *Configurações dos Testes de Propriedades*

Tendência

Esta Seção apresenta os resultados obtidos na avaliação da tendência dos operadores da RNP para soluções com topologia de árvore estrela ou similares a árvore geradora mínima. Para todas os tamanhos de grafos analisados os resultados foram similares e por isso serão discutidos dois casos.

Nos testes da propriedade tendência, se $d_{mst-pop}$ diminui, o operador possui tendência para MST. Se $d_{star-pop}$ diminui, o operador possui tendência para árvores estrela. Se as medidas permanecem constantes, o operador não possui tendência.

Na Figura 5.1 os gráficos apresentam os resultados para $d_{mst-pop}$ para todos os operadores nas instâncias de 40 e 250 vértices. Os resultados mostram que $d_{mst-pop}$ se mantém constante e com valores similares para todos os operadores. Esse comportamento indica ausência de tendência para esse tipo de estrutura. Para as instâncias de grafos de outros número de vértices testados, o comportamento dos operadores para $d_{mst-pop}$ é similar, ou seja sem tendência para MST.

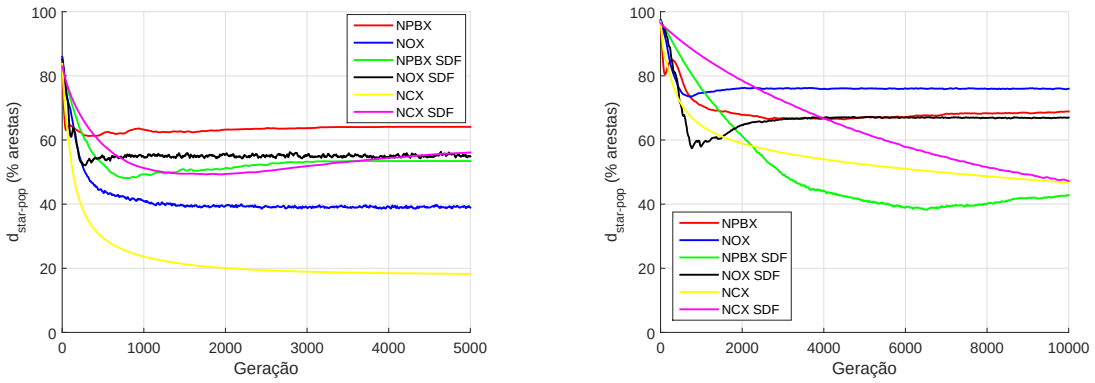


(a) *Grafo com 40 vértices* (b) *Grafo com 250 vértices*

Figura 5.1: *Distância para a MST*

Na Figura 5.2 são apresentados os resultados para $d_{star-pop}$ para todos os operadores nas instâncias de 40 e 250 vértices. Os resultados mostram que a distância para árvores estrela diminui para todos os operadores, com maior intensidade para o operador NCX, que apresentou as menores distâncias em média para árvores de topologia estrela.

Na medida em que o tamanho das instâncias aumenta, os operadores NCX-SDF e NPBX-SDF possuem tendência similar ao operador NCX conforme pode ser visto na Figura 5.2(b).



(a) *Grafo com 40 vértices* (b) *Grafo com 250 vértices*

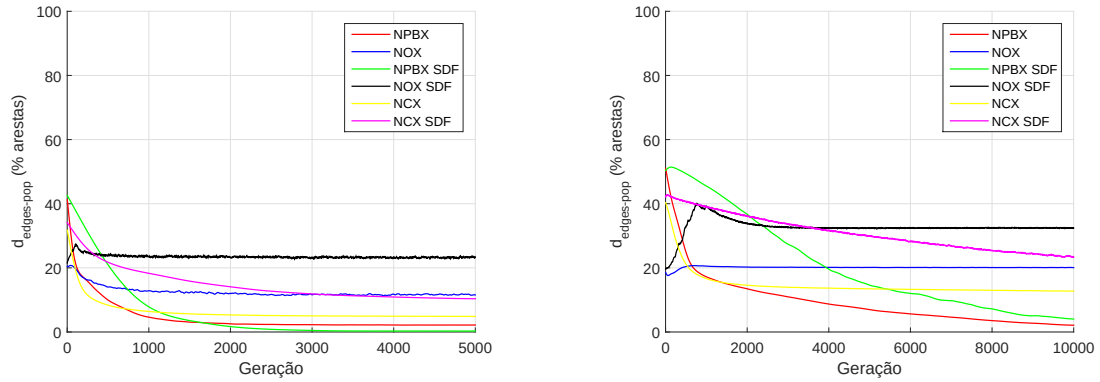
Figura 5.2: *Distância mínima para árvores Estrela*

Hereditariedade

Os resultados dos testes para a propriedade hereditariedade são mostrados nas Figuras 5.3 ($d_{edges-pop}$, hereditariedade das arestas) e 5.4 ($d_{depth-pop}$, hereditariedade da profundidade dos vértices). O cálculo da hereditariedade das arestas considera o conjunto de arestas não compartilhadas entre pais e filhos, portanto se $d_{edges-pop}$ diminui, isso significa aumento da hereditariedade do operador de recombinação. A hereditariedade da profundidade dos vértices é determinada pelo número de vértices dos filhos cuja profundidade é diferente nos pais, portanto se $d_{depth-pop}$ diminui, isso significa aumento da hereditariedade para essa característica. Como ocorreu para a propriedade de tendência, os resultados são similares para todos os casos analisados e assim são apresentados somente dois casos.

Na Figura 5.3(a), a hereditariedade das arestas de todos os operadores melhora no decorrer das gerações para grafos de 40 vértices, com exceção do operador NOX-SDF que apresentou comportamento constante. Destaca-se os resultados alcançados pelos operadores NPBX, NPBX-SDF e NCX como os melhores entre todos os operadores, sendo que o operador NPBX-SDF obtém distância das arestas quase zero.

Para as instâncias de grafos de 250 vértices, nos resultados apresentados na Figura 5.3(b) observa-se novamente o potencial de preservar arestas dos operadores NPBX, NPBX-SDF e NCX. O operador NOX apresenta comportamento constante no decorrer das gerações e o operador NOX-SDF apresentou piora da hereditariedade das arestas nas gerações inicias, estabilizando próximo da geração de número 2000. Para as



(a) *Grafo com 40 vértices* (b) *Grafo com 250 vértices*

Figura 5.3: *Hereditariedade das Arestas*

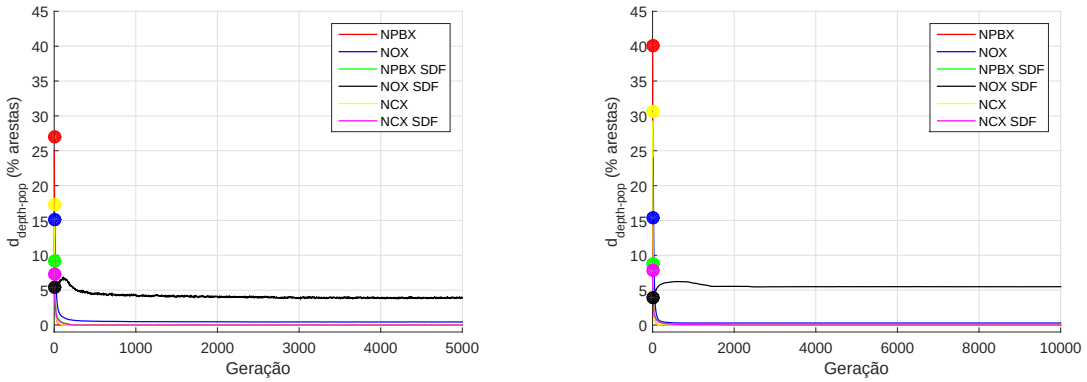
instâncias de grafos de outros tamanhos testados, o comportamento dos operadores para $d_{edges-pop}$ é similar.

Os resultados para a propriedade $d_{depth-pop}$ são exibidos na Figura 5.4. Todos os operadores com a exceção do operador NOX-SDF possuem hereditariedade alta para a profundidade dos vértices desde as primeiras gerações. Ou seja, a profundidade dos vértices é preservada nas soluções geradas pelos operadores. Na Figura 5.4(a) é possível observar uma queda da diferença de profundidade dos vértices entre pais e filhos inicialmente em 25% para muito próximo de zero já nas primeiras gerações para as instâncias de grafos de 40 vértices. Nas instâncias de grafos de 250 vértices, os resultados mostrados na Figura 5.4(b) indicam uma queda da diferença partindo de cerca de 40% do número de arestas da árvore nas primeiras gerações para também muito próximo de zero. O operador NOX-SDF foi o operador que demonstrou o comportamento mais distante dos outros operadores, mantendo-se estável em cerca de 5% do número de vértices da árvore cuja profundidade dos filhos difere da profundidade dos vértices dos pais, para todos os tamanhos de grafos. Dos operadores que adotam o método de correção SDF, os operadores NPBX-SDF e NCX-SDF apresentaram os melhores resultados para a hereditariedade da profundidade dos vértices.

Comparando as duas medidas de hereditariedade, com exceção do operador NOX-SDF todos os outros operadores apresentaram maior potencial para preservar a profundidade dos vértices do que arestas. Para as instâncias de grafos de outros tamanhos testados, o comportamento dos operadores para $d_{depth-pop}$ é similar e por esse motivo são omitidos.

É possível deduzir com base nos resultados, que a adoção do método de correção de profundidade SDF não significou melhora da hereditariedade das arestas. Supõe-se que o fato do método SDF alterar a ordem dos genes no cromossomo transferindo regiões inteiras de uma subárvore, possa ter singificado alterações significativas na estrutura da

solução que não refletiram o ganho por preservar a profundidade dos vértices.



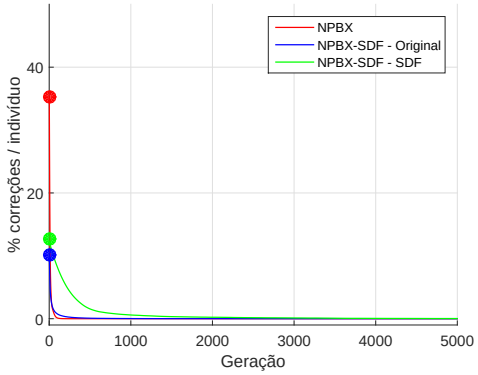
(a) *Grafo com 40 vértices* (b) *Grafo com 250 vértices*

Figura 5.4: *Hereditariedade da Profundidade dos Vértices*

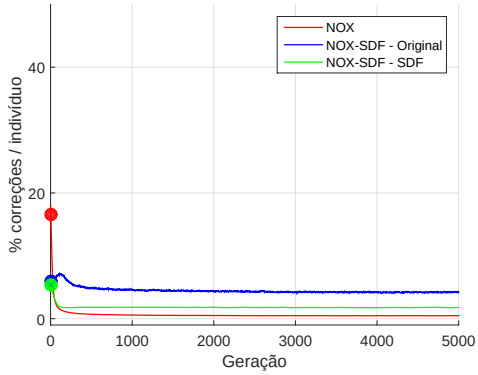
5.1.3 Análises Complementares

Os testes complementares $fixes_{pop}$ e $depth-change_{pop}$ fornecem indícios que podem explicar alguns comportamentos apresentados pelos operadores de recombinação. Na Figura 5.5, são apresentados os gráficos para $fixes_{pop}$ que é a média do tipo de correção aplicada por cada operador por indivíduo. Para facilitar a comparação, os resultados foram normalizados de forma que $fixes_{pop} = fixes_{pop}/(n-1)$. Na Figura 5.5, são apresentados os resultados para as duas versões de cada operador. Foi adotada a nomenclatura básica do operador para indicar a versão que aplica exclusivamente a correção original, como o NPBX, NOX e NCX. As versões dos operadores que aplicam ambos os métodos de correção são identificadas pelo seu nome básico, seguido da sigla SDF e do tipo de correção aplicada, 'Original' quando não foi possível aplicar o método SDF, e 'SDF' quando foi possível. Para $fixes_{pop}$, são apresentados os resultados dos testes aplicados somente para um número de vértices diferente para cada operador, uma vez que os operadores apresentaram comportamento similar independente do número de vértices do grafo.

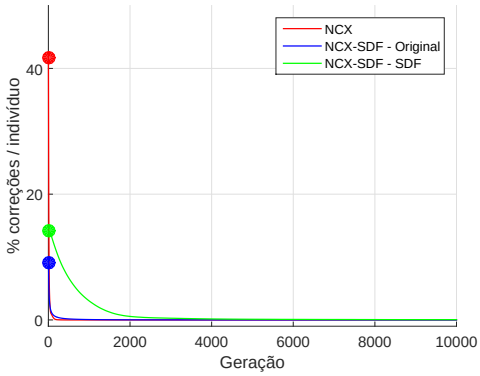
Os resultados para $fixes_{pop}$, indicam que a disposição do método de correção SDF, resultou em redução significativa da aplicação de correções através do método original para os operadores NPBX e NCX nas gerações inicias. Muito rapidamente, as duas versões de cada operador passam aplicar o método original de correção de maneira muito semelhante. A aplicação do método SDF é ligeiramente maior nas primeiras gerações, mas logo se aproxima do método original. No entanto, o operador NOX-SDF aplica o método original de correção inclusive com maior frequência do que a versão NOX que dispõe exclusivamente do método original. Além disso, quando NOX-SDF é utilizado



(a) *Grafo com 80 vértices*



(b) *Grafo com 40 vértices*

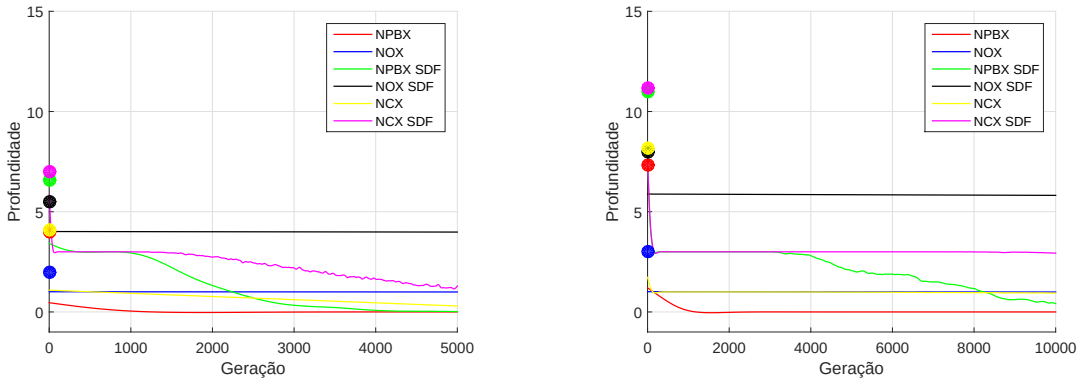


(c) *Grafo com 250 vértices*

Figura 5.5: *Contagem de Tipo de Correção*

o número de correções da profundidade não tende a zero, como nos demais operadores. Diferentemente dos operadores NPBX e NCX que aplicam o cruzamento por posições, o operador NOX considera regiões contíguas dos cromossomos demandando em tese menos correções, o que pode explicar o comportamento observado na Figura 5.5(b) na qual o número de correções da versão do operador que possui apenas o método original, é menor do que a versão que possui o método original e o SDF. A aplicação do método SDF que significa transferir uma região de um ponto a outro do cromossomo no operador NOX, pode ter significado transformações significativas na estrutura da solução e demandado assim um número maior de correções se comparado aos ouros operadores.

Na Figura 5.6, são apresentados os resultados que indicam a média da diferença de profundidade por vértice, quando o método de correção original é aplicado tanto nas versões dos operadores que o aplicam exclusivamente, quanto nas versões que dispõem do método SDF. Os resultados não foram normalizados e indicam diretamente a diferença de profundidade média por vértice corrigido. Assim como nos demais testes, são mostrados os resultados somente para dois números de vértices diferentes de grafos para $depth - change_{pop}$.



(a) *Grafo com 80 vértices* (b) *Grafo com 250 vértices*

Figura 5.6: *Diferença de profundidade média por vértice corrigido*

Os resultados para $d_{depth-pop}$, indicam que a aplicação do método original de correção de profundidade nos operadores NCX-SDF, NPBX-SDF, NCX e NOX-SDF causam impacto mais forte em termos de diferença de profundidade do que os outros operadores nas primeiras gerações. Isso pode significar que esses operadores conduzem a população a uma convergência rápida para soluções mais próximas a árvores estrela, como indicam os resultados da propriedade tendência para árvores estrela da Figura 5.2 nesses operadores.

Apesar do operador NOX-SDF apresentar comportamento maior para $d_{depth-pop}$ durante as gerações, o comportamento estável apresentado para $fixes_{pop}$ indica que correções de profundidade permanecem sendo aplicadas durante as gerações, o que indica dificuldade de convergência da população. Outro indício desse comportamento do operador NOX-SDF pode ser observado nos resultados da propriedade hereditariedade medida sob a profundidade dos vértices na Figura 5.4, em que o operador NOX-SDF demonstra menor potencial para conservar a profundidade dos vértices comparado aos outros operadores.

5.2 AE aplicado ao Problema da Árvore Geradora Mínima com Restrição de Diâmetro

O BDMSTP consiste na busca de uma árvore geradora submetida à uma restrição de diâmetro, cujos custos sejam mínimos. Os custos são baseados nos pesos das arestas e a restrição de diâmetro é definida como o caminho mais longo encontrado entre dois vértices quaisquer da árvore geradora (ver Seção 4.3).

Um AE para cada um dos operadores de recombinação NOX-SDF, NPBX-SDF, NCX e NCX-SDF foi desenvolvido para ser aplicado ao BDMSTP. As configurações e as instâncias dos testes são descritos na Seção 5.2.1. Os resultados obtidos por esses

operadores serão comparados com os resultados relatados em [7] para a RNP (EANR) utilizando apenas seus operadores de mutação e com os resultados de Edge-Sets (JR-ESEA) [45].

5.2.1 Instâncias e Configurações dos Testes

Foram realizados testes nos AEs com os operadores de recombinação da RNP para definição da melhor configuração dos parâmetros para o BDMSTP. Essa configuração possui os seguintes valores que serão utilizados para todos os testes: população de tamanho 400, com taxa de recombinação de 100%, taxa de mutação de 20%, critério de parada definido em 100.000 iterações sem melhora do melhor indivíduo encontrado. Os AEs seguem o modelo *steady-state*. O tamanho da população e o critério de parada foram definidos considerando as configurações dos AEs aplicados ao BDMSTP relatados em [45] e [7] com os quais os resultados alcançados neste trabalho são comparados. Foram realizadas 30 execuções para cada combinação de instância/operador. A RNP dispõe de dois operadores de mutação denominados PAO e CAO (ver Seção 3.2.6) que foram aplicados nos testes de acordo com a taxa de mutação definida. A escolha de qual operador de mutação a ser aplicado foi aleatória com probabilidade de 50% para cada um. A Tabela 5.2 sumariza a configuração utilizada para os testes.

População	400
% Mutação	20
% Recombinação	100
Execuções	30

Tabela 5.2: *Configurações dos Testes aplicados ao BDMSTP*

As instâncias nas quais os testes foram aplicados são grafos Euclidianos para o problema de Árvores de Steiner Euclidiano disponíveis na biblioteca OR-Library[1]. Assim como em [45] os testes foram aplicados para as 5 primeiras instâncias de grafos de número de vértices $n = 50$, $n = 100$, $n = 250$ e $n = 500$. Para cada tamanho de grafo, em [45], foi definida uma restrição de diâmetro k.

5.2.2 Resultados dos Testes do BDMSTP

Os AEs aplicados ao BDMSTP foram desenvolvidos em linguagem C e executados no sistema operacional Linux. Os resultados apresentam o valor da melhor solução encontrada nas 30 execuções em termos de custo das arestas e a média de iterações para cada AE com seu respectivo operador de recombinação para cada instância. Os experimentos foram executados em máquinas de diferentes configurações que vão desde um

[1] http://people.brunel.ac.uk/~mastjjb/jeb/info.html

processador Intel Core i5 com 4GB de memória, a uma máquina com processador Intel Core i7 com 32GB.

Para fins de comparação, os resultados reportados em [7] para a RNP (EANR) e [45] para o Edge-Sets (RJ-ESEA) para AEs aplicados ao BDMSTP são reproduzidos na Tabela 5.3 onde n é o número de vértices do grafo, i é o número da instância e k é a restrição de diâmetro. A RNP foi aplicada ao BDMSTP utilizando apenas seus operadores de mutação e uma população $N = 20$ enquanto o Edge-Sets foi executado com operadores de recombinação e mutação e população $N = 400$.

Instâncias			EANR		JR-ESEA	
n	i	k	Melhor	Iterações	Melhor	Iterações
50	1	5	8,81	108.678	7,60	33.947
50	2	5	8,62	110.208	7,68	36.403
50	3	5	7,98	110.125	7,24	27.919
50	4	5	8,52	107.717	6,59	31.382
50	5	5	8,72	107.790	7,32	34.924
100	1	10	8,66	215.816	8,00	189.026
100	2	10	9,50	198.888	8,10	205.891
100	3	10	9,43	195.152	8,22	176.043
100	4	10	9,36	209.813	8,27	163.142
100	5	10	9,41	202.294	8,48	164.651
250	1	15	13,74	947.338	12,93	471.803
250	2	15	13,72	943.624	12,86	466.047
250	3	15	13,40	986.246	12,69	464.618
250	4	15	13,96	941.691	13,22	442.446
250	5	15	13,74	1.002.782	13,02	497.450
500	1	20	18,58	3.278.443	18,33	527.659
500	2	20	18,83	3.065.966	18,17	652.009
500	3	20	18,93	3.145.114	18,33	504.315
500	4	20	18,57	3.221.440	18,32	654.871
500	5	20	18,70	3.086.630	17,80	648.148

Tabela 5.3: *Resultados dos Testes aplicados ao BDMSTP da RNP (EANR) e Edge-Sets (JR-ESEA)*

Os resultados dos AEs com os operadores de recombinação desenvolvidos nesse trabalho para o BDMSTP são mostrados nas Tabelas 5.4, onde são mostradas as melhores soluções encontradas e a média da qualidade das soluções, e 5.5 onde são mostrados o desvio padrão e o número de iterações necessárias para encontrar a melhor solução. Nas Tabelas 5.4 e 5.5, n é o número de vértices do grafo testado, i é o número da instância e k é a restrição de diâmetro. De acordo com os resultados, o operador NOX-SDF tem a melhor performance entre todos os operadores analisados da proposta deste trabalho, tanto em qualidade das soluções encontradas quanto no número de iterações na maioria das instâncias testadas. O AE com o operador NCX obteve melhores soluções do que o

AE com operador NOX-SDF nas instâncias 4 e 5 do grafo de 100 vértices, embora ainda com pior média de iterações.

Instâncias			NOX-SDF		NPBX-SDF		NCX		NCX-SDF	
n	i	k	Melhor	Média	Melhor	Média	Melhor	Média	Melhor	Média
50	1	5	7,67	8,30	7,96	8,65	7,78	8,49	7,78	8,48
50	2	5	7,69	8,06	7,97	8,35	7,74	8,32	7,78	8,26
50	3	5	7,17	7,78	7,31	7,93	7,51	7,96	7,54	8,20
50	4	5	6,58	7,01	7,15	7,62	6,71	7,29	6,80	7,50
50	5	5	7,37	7,92	7,40	8,13	7,50	8,02	7,37	8,02
100	1	10	8,16	8,45	8,51	9,14	8,35	8,67	8,29	8,78
100	2	10	8,16	8,66	8,70	9,29	8,31	8,93	8,26	8,97
100	3	10	8,30	8,78	8,56	9,27	8,49	9,03	8,53	9,08
100	4	10	8,38	8,93	8,56	9,23	8,37	8,93	8,66	9,14
100	5	10	8,50	8,88	8,94	9,71	8,41	9,06	8,77	9,22
250	1	15	13,53	14,06	15,33	17,65	13,86	15,17	14,50	15,10
250	2	15	13,65	14,29	15,68	17,81	14,04	15,30	14,19	15,66
250	3	15	13,00	13,76	15,52	17,28	13,63	14,72	14,00	15,40
250	4	15	13,95	14,61	16,47	18,53	14,37	15,53	14,51	16,12
250	5	15	13,50	14,20	15,84	17,50	14,03	15,06	14,02	15,50
500	1	20	19,64	20,82	25,98	31,66	20,75	23,30	20,68	24,37
500	2	20	19,74	20,80	26,04	30,91	21,67	21,19	21,86	24,41
500	3	20	19,43	20,70	24,34	32,17	21,16	23,46	21,22	24,03
500	4	20	19,63	20,88	25,15	31,69	21,74	23,96	22,00	24,27
500	5	20	19,43	20,38	25,76	31,72	21,00	23,27	21,01	23,39

Tabela 5.4: *Média e melhor resultado dos testes aplicados ao BDMSTP*

Se comparado aos resultados dos AEs da Tabela 5.3, o operador NOX-SDF apresenta melhores soluções que a RNP (EANR) para todas as instâncias de número de vértices 50, 100 e 250 embora necessite de mais iterações. Comparando com os resultados do Edge-Sets (JR-ESEA), as soluções do operador NOX-SDF são melhores nas instâncias 3 e 4 para o grafo de 50 vértices.

Apesar de ter apresentado resultados piores para a propriedade hereditariedade das arestas e das profundidades dos vértices (Ver Seção 5.1.2), o operador NOX-SDF obteve os melhores resultados em qualidade da solução e número de iterações dentre os operadores desenvolvidos nesse trabalho. Supõe-se que seu comportamento resistente a uma convergência prematura, significou potencial para explorar o espaço de busca e escapar de ótimos locais, pelo menos quando aplicado ao BDMSTP.

Os AEs com os operadores NPBX-SDF, NCX e NCX-SDF superam o AE com RNP na qualidade das soluções para todas as instâncias de 50 e 100 vértices, embora sejam sempre piores no número médio de iterações. Comparadas com o Edge-Sets, os resultados são levemente piores.

Instâncias			NOX-SDF		NPBX-SDF		NCX		NCX-SDF	
n	*i*	*k*	D.P.	Iter.	D.P.	Iter.	D.P.	Iter.	D.P.	Iter.
50	1	5	0,31	347.037	0,14	458.064	0,30	341.128	0,80	409.772
50	2	5	0,29	344.974	0,31	416.000	0,45	379.154	0,98	380.392
50	3	5	0,33	401.166	0,29	476.792	0,34	416.535	0,81	398.006
50	4	5	0,32	379.683	0,36	462.441	0,35	425.931	0,64	427.629
50	5	5	0,33	369.174	0,19	467.189	0,43	406.431	0,61	394.629
100	1	10	0,41	590.739	0,40	960.266	1,43	759.688	3,46	805.327
100	2	10	0,30	606.934	0,39	1.017.444	1,33	788.154	4,05	740.992
100	3	10	0,41	583.602	0,46	1.134.443	1,32	825.291	4,79	814.453
100	4	10	0,32	546.853	0,37	1.084.156	1,70	761.328	5,14	736.245
100	5	10	0,30	545.700	0,54	972.924	1,07	796.963	3,69	804.468
250	1	15	0,42	1.364.288	0,20	3.135.315	0,91	2.121.849	1,43	2.288.825
250	2	15	0,27	1.339.930	0,48	3.452.084	0,80	2.202.913	1,29	2.203.473
250	3	15	0,28	1.518.578	0,38	3.397.480	0,66	2.228.117	1,35	2.245.727
250	4	15	0,32	1.518.578	0,37	3.311.124	0,86	2.240.766	1,70	2.070.025
250	5	15	0,28	1.469.707	0,35	3.476.253	0,54	2.285.492	1,54	2.245.800
500	1	20	0,40	2.916.108	0,30	7.777.363	0,47	4.733.668	2,30	4.918.968
500	2	20	0,26	2.843.276	0,34	7.615.188	1,07	4.474.508	1,90	4.580.941
500	3	20	0,41	2.880.708	0,38	7.788.471	1,06	4.583.107	1,96	4.882.488
500	4	20	0,28	2.851.537	0,37	7.884.682	0,86	4.548.848	2,14	4.780.801
500	5	20	0,37	2.788.063	0,31	7.277.843	0,98	4.468.425	1,57	4.885.995

Tabela 5.5: *Desvio padrão e iterações dos resultados dos Testes aplicados ao BDMSTP*

5.3 Considerações Finais

Neste capítulo, foram apresentados os resultados dos testes das propriedades tendência e hereditariedade dos operadores de recombinação para a RNP. Os testes foram aplicados utilizando grafos Euclidianos e mediram a tendência dos operadores para MST e árvores estrela. Foram medidos também a hereditariedade para arestas e profundidade dos vértices. Além dos testes de propriedades, nesse capítulo foram apresentados os resultados de AEs com os operadores desenvolvidos nesse trabalho aplicados ao BDMSTP.

Os testes mostraram que os operadores não possuem tendência para MST. Todos os operadores mostraram tendência para árvore estrela, com destaque para os operadores NPBX-SDF, NOX-SDF, NCX e NCX-SDF que apresentaram a maior tendência para esse tipo de solução.

Os operadores NPBX, NPBX-SDF e NCX apresentaram o maior potencial de hereditariedade das arestas entre os operadores testados, enquanto o operador NOX-SDF apresentou o pior índice para essa característica para instâncias de grafos maiores. Todos os operadores mostraram forte potencial para preservar a profundidade dos vértices durante a recombinação. Novamente, o operador NOX-SDF apresentou o pior resultado para essa característica também.

Os resultados dos AEs aplicados ao BDMSTP foram comparados com os resultados de outros AEs com representações para PPRs também aplicados ao BDMSTP. O operador NOX-SDF alcançou os melhores resultados para todas as instâncias se comparado aos outros operadores desenvolvidos nesse trabalho. Todos os operadores desenvolvidos neste trabalho mostraram melhora em relação aos resultados relatados de outras representações para algumas instâncias.

CAPÍTULO 6

Conclusão

Neste trabalho, abordou-se a aplicação de Algoritmos Evolutivos na resolução de Problemas de Projeto de Redes. Os principais conceitos e mecanismos sobre Algoritmos Evolutivos foram apresentados, dentre eles a Representação, conceito que estabelece a forma são codificadas computacionalmente as soluções para um problema determinado e o funcionamento dos operadores de busca e seleção aplicados às soluções. Em especial, tratou-se das principais representações conhecidas na literatura aplicadas a problemas baseados em permutação, e as propriedades a serem consideradas em um processo decisório de escolha de representação.

A proposta desse trabalho foi o desenvolvimento de novos operadores de recombinação para a representação Nó-Profundidade. O desenvolvimento dos novos operadores envolveu a adaptação dos operadores de recombinação originais NOX e NPBX da representação com um novo mecanismo de correção proposto neste trabalho, baseado na busca de subárvores como alternativa à correção de profundidades de vértices que causariam inviabilidade das soluções. Dessa proposta, surgiram os operadores NOX-SDF e NPBX-SDF. Além da adaptação dos operadores originais, um outro operador de recombinação foi desenvolvido cujo funcionamento foi inspirado no conhecido operador de recombinação baseado em permutação CX. O operador NCX surgiu dessa proposta em duas versões, uma aplicando o método de correção de profundidade original da representação, e a segunda referenciada como NCX-SDF que aplica o método baseado em subárvore proposto neste trabalho.

Todos os operadores desenvolvidos, além dos operadores originais foram validados para as propriedades tendência e hereditariedade ao serem aplicados a instâncias de grafos Euclidianos de número de vértices $n = 10$ a $n = 250$. Para tendência, mediu-se para cada operador o potencial para privilegiar a geração de soluções do tipo árvore geradora mínima e árvore estrela. Sobre a hereditariedade, foram medidos o potencial para preservar arestas e para preservar a profundidade dos vértices em operações de recombinação para cada operador. Os resultados mostraram que os operadores não possuem tendência para árvores geradoras mínimas. Por outro lado, houve tendência para árvores estrela em todos os operadores, com ênfase para os operadores NPBX-SDF, NCX e NCX-SDF. Os

resultados mostraram também que os operadores possuem potencial para a hereditariedade das arestas, com exceção do operador NOX-SDF que apresentou piora para essa característica ao ser aplicado a instâncias de grafos maiores. A hereditariedade da profundidade dos vértices é alta, contudo não houve melhora dessa característica com o novo método de correção proposto.

Além da validação, os operadores desenvolvidos nesse trabalho foram aplicados ao Problema da Árvore Geradora Mínima com Restrição de Diâmetro. Os testes foram realizados em instâncias de grafos Euclidianos de números de vértices entre $n = 50$ a $n = 500$ com uma restrição k diferente para cada tamanho. O resultados foram comparados aos encontrados por outras representações relatadas e mostraram que os operadores desenvolvidos neste trabalho são eficientes, obtendo soluções de melhor qualidade para algumas instâncias, com ênfase para o operador NOX-SDF que mostrou os melhores resultados.

6.1 Trabalhos Futuros

A proposta deste trabalho para o mecanismo de correção de profundidade dos vértices, considerou a busca de uma subárvore válida no conjunto de genes oriundos do processo de recombinação a partir do vértice com profundidade inválida. Nesse sentido, o mecanismo nem sempre foi bem sucedido na busca de uma subárvore válida, passando a aplicar o método original de correção como alternativa. Propõe-se como sugestões para trabalhos futuros:

- o desenvolvimento de um mecanismo de correção de profundidade que, não só busque uma subárvore a partir do gene inválido, mas que possa realizar a busca inclusive nos genes válidos já inseridos na solução
- exploração de outros operadores de recombinação para permutações para a RNP
- como mostraram os resultados (Ver Capítulo 5), os operadores NCX, NPBX-SDF, NPBX mostraram bons resultados para a propriedade hereditariedade em constraste com o operador NOX-SDF. Propõe-se o uso combinado dos operadores de recombinação aplicados a outros PPRs de forma auto-adaptativa explorando o potencial de conservação dos operadores NCX, NPBX-SDF e NPBX combinados com o potencial explorador do operador NOX-SDF.

Referências Bibliográficas

[1] ANDREICA, A.; CHIRA, C. **Best-order crossover for permutation-based evolutionary algorithms**. *Applied Intelligence*, 42(4):751–776, 2015.

[2] Back, T.; Fogel, D. B.; Michalewicz, Z., editors. **Evolutionary Computation 1: Basic Algorithms and Operators**. IOP Publishing Ltd., Bristol, UK, UK, 1st edition, 1999.

[3] BERTSIMAS, D. **The probabilistic minimum spanning tree problem**. *Networks*, 20(3):245–275, 1990.

[4] CORDONE, R.; PASSERI, G. **Solving the quadratic minimum spanning tree problem**. *Applied Mathematics and Computation*, 218(23):11597 – 11612, 2012.

[5] DAVIS, L. **Applying adaptive algorithms to epistatic domains**. In: *Proceedings of the 9th International Joint Conference on Artificial Intelligence - Volume 1*, IJCAI'85, p. 162–164, San Francisco, CA, USA, 1985. Morgan Kaufmann Publishers Inc.

[6] DE JONG, K. A. **Evolutionary Computation: A Unified Approach**. MIT press, 2006.

[7] DE LIMA, T. W.; ROTHLAUF, F.; DELBEM, A. C. **The node-depth encoding: Analysis and application to the bounded-diameter minimum spanning tree problem**. In: *Proceedings of the 10th Annual Conference on Genetic and Evolutionary Computation*, GECCO '08, p. 969–976, New York, NY, USA, 2008. ACM.

[8] DE LIMA, T. W.; DELBEM, A. C. B.; LIMA, R. L.; SABIN, G. P.; DE OLIVEIRA, M. A. A. **Permutation-based recombination operator to node-depth encoding**. *Procedia Computer Science*, 80:279 – 288, 2016. International Conference on Computational Science 2016, {ICCS} 2016, 6-8 June 2016, San Diego, California, {USA}.

[9] DE LIMA SOARES, T. W. **Estruturas de dados eficientes para algoritmos evolutivos aplicados a projeto de redes**, 2009.

[10] DELBEM, A. C. B.; DE CARVALHO, A. C. P. L. F.; POLICASTRO, C. A.; PINTO, A. K. O.; HONDA, K.; GARCIA, A. C. **Node-depth encoding for evolutionary algorithms applied to network design**. In: Deb, K.; Poli, R.; Banzhaf, W.; Beyer, H.-G.;

Burke, E. K.; Darwen, P. J.; Dasgupta, D.; Floreano, D.; Foster, J. A.; Harman, M.; Holland, O.; Lanzi, P. L.; Spector, L.; Tettamanzi, A.; Thierens, D.; Tyrrell, A. M., editors, *Genetic and Evolutionary Computation - GECCO 2004, Genetic and Evolutionary Computation Conference, Seattle, WA, USA, June 26-30, 2004, Proceedings, Part I*, volume 3102 de **Lecture Notes in Computer Science**, p. 678–687. Springer, 2004.

[11] DELBEM, A. C. B.; DE LIMA, T. W.; TELLES, G. P. **Efficient forest data structure for evolutionary algorithms applied to network design**. *IEEE Trans. Evolutionary Computation*, 16(6):829–846, 2012.

[12] FOGEL, D. B. **Evolutionary Computation: Toward a New Philosophy of Machine Intelligence**. IEEE Press, NY, 1995.

[13] GAUBE, T.; ROTHLAUF, F. **The link and node biased encoding revisited: Bias and adjustment of parameters**. *Lecture Notes in Computer Science*, 2037:1–??, 2001.

[14] GOLDBERG, D. E. **Genetic Algorithms in Search, Optimization, and Machine Learning**. Addison-Wesley, Reading, Mass., 1989.

[15] HAGHIGHAT, A. T.; FAEZ, K.; DEHGHAN, M.; MOWLAEI, A.; GHAHREMANI, Y. **A genetic algorithm for steiner tree optimization with multiple constraints using prüfer number**. In: Shafazand, H.; Tjoa, A. M., editors, *EurAsia-ICT 2002: Information and Communication Technology, First EurAsian Conference, Shiraz, Iran, October 29-31, 2002, Proceedings*, volume 2510 de **Lecture Notes in Computer Science**, p. 272–280. Springer, 2002.

[16] HAMMING, R. W. **Coding and information theory**. Englewood Cliffs, N.J. Prentice-Hall, 1980. Includes index.

[17] HOLLAND, J. **Adaptation in Natural and Artificial Systems.** University of Michigan Press, 1975.

[18] HUYNEN, M. **Exploring phenotype space through neutral evolution**, 1996.

[19] HUYNEN, M. A.; STADLER, P. F. **Smoothness within ruggedness: The role of neutrality in adaptation**, 1996.

[20] ISHII, H.; SHIODE, S.; NISHIDA, T.; NAMASUYA, Y. **Stochastic spanning tree problem**. *Discrete Applied Mathematics*, 3(4):263 – 273, 1981.

[21] JOHNSON, D. S.; LENSTRA, J. K.; KAN, A. H. G. R. **The complexity of the network design problem**. *Networks*, 8(4):279–285, 1978.

[22] JULSTROM, B. A. **Encoding Bounded-Diameter Spanning Trees with Permutations and with Random Keys**, p. 1272–1281. Springer Berlin Heidelberg, Berlin, Heidelberg, 2004.

[23] JULSTROM, B. A. **Greedy heuristics for the bounded diameter minimum spanning tree problem**. *ACM Journal of Experimental Algorithmics*, 14, 2009.

[24] JULSTROM, B. A.; RAIDL, G. R. **A permutation-coded evolutionary algorithm for the bounded-diameter minimum spanning tree problem**. In: Barry, A. M., editor, *GECCO 2003: Proceedings of the Bird of a Feather Workshops, Genetic and Evolutionary Computation Conference*, p. 2–7, Chigaco, 11 July 2003. AAAI.

[25] KIMURA, M. **The neutral theory of molecular evolution**. *Scientific American*, 241(5):98–126, Nov. 1979.

[26] KRUSKAL, J. B. **On the shortest spanning subtree of a graph and the travelling salesman problem**. In: *Proc. Am. Math. Soc.*, p. 48–50, Feb. 1956. Published as Proc. Am. Math. Soc., volume 7, number 1.

[27] LINDEN, R. **Algoritmos Genéticos (2a edição)**. BRASPORT, 2008.

[28] MAGNANTI, T. L.; WONG, R. T. **Network design and transportation planning: Models and algorithms**. *Transportation Science*, 18:1–56, 1984.

[29] MICHALEWICZ, Z. **Genetic Algorithms + Data Structures = Evolution Programs**. Springer-Verlag, 1996. Contains introductory chapter on LCS.

[30] NADIRADZE, G. **Bounded Diameter Minimum Spanning Tree**. PhD thesis, Central European University, 2013.

[31] OLIVER, I. M.; SMITH, D. J.; HOLLAND, J. R. C. **A study of permutation crossover operators on the traveling salesman problem**. In: *Proceedings of the Second International Conference on Genetic Algorithms on Genetic Algorithms and Their Application*, p. 224–230, Hillsdale, NJ, USA, 1987. L. Erlbaum Associates Inc.

[32] PALMER, C. C.; KERSHENBAUM, A. **An approach to a problem in network design using genetic algorithms**. *Networks*, 26(3):151–163, 1995.

[33] POON, P. W.; CARTER, J. N. **Genetic algorithm crossover operators for ordering applications**. *Computers & OR*, 22(1):135–147, 1995.

[34] PRIM, R. C. **Shortest connection networks and some generalizations**. *Bell Systems Technical J.*, p. 1389–1401, Nov. 1957.

[35] RAIDL.; JULSTROM. **Edge sets: An effective evolutionary coding of spanning trees**. *EVCOMP: Evolutionary Computation*, 2003.

[36] RECHENBERG, I. **Cybernetic solution path of an experimental problem**. Technical report, Royal Air Force Establishment, 1965.

[37] REIDYS, C. M.; STADLER, P. F. **Neutrality in fitness landscapes**. *Applied Mathematics and Computation*, 117(2–3):321–350, Jan. 2001.

[38] ROTHLAUF, F. **Representations for genetic and evolutionary algorithms**. Springer, pub-SV:adr, second edition, 2006.

[39] ROTHLAUF, F. **Design of Modern Heuristics: Principles and Application**. Springer Publishing Company, Incorporated, 1st edition, 2011.

[40] ROTHLAUF, F.; GERSTACKER, J.; HEINZL, A. **On the optimal communication spanning tree problem**. *Working Papers in Business Administration and Information Systems*, 10, 2004.

[41] ROTHLAUF, F.; GOLDBERG, D. E.; HEINZL, A. **Network random keys-A tree representation scheme for genetic and evolutionary algorithms**. *Evolutionary Computation*, 10(1):75–97, 2002.

[42] RUDNICK, W. M. **Genetic algorithms and fitness variance with an application to the automatic design of artificial neural networks**. Oregon Graduate Institute of Science and Technology, 1992.

[43] SCHUSTER, P. **Genotypes with phenotypes: Adventures in an RNA toy world**, 1997.

[44] SCHWEFEL, H. P. **Evolutionsstrategie und numerische Optimierung**. PhD thesis, Technische Universitat Berlin, Berlin, May 1975.

[45] SINGH, A.; GUPTA, A. K. **Improved heuristics for the bounded-diameter minimum spanning tree problem**. *Soft Comput*, 11(10):911–921, 2007.

[46] SOAK, S.-M.; CORNE, D.; AHN, B.-H. **A new encoding for the degree constrained minimum spanning tree problem**. In: Negoita, M. G.; Howlett, R. J.; Jain, L. C., editors, *Knowledge-Based Intelligent Information and Engineering Systems, 8th International Conference, KES 2004, Wellington, New Zealand, September 20-25, 2004. Proceedings. Part I*, volume 3213 de **Lecture Notes in Computer Science**, p. 952–958. Springer, 2004.

[47] SYSWERDA, G. **Schedule optimization using genetic algorithms**. In: Davis, L., editor, *Handbook of Genetic Algorithms*. Van Nostrand Reinhold, New York, NY, 1991.

[48] THOMPSON, E.; PAULDEN, T.; SMITH, D. K. **The dandelion code: A new coding of spanning trees for genetic algorithms**. *IEEE Trans. Evolutionary Computation*, 11(1):91–100, 2007.

[49] VINHAL, G. S. **Implementação de um algoritmo evolutivo utilizando a representação nó-profundidade-grau no processador nios ii do fpga**, 2013.