UNIVERSIDADE FEDERAL DE GOIÁS
INSTITUTO DE INFORMÁTICA
CIÊNCIA DA COMPUTAÇÃO

JOELMA DE MOURA FERREIRA

Problemas de Otimização Combinatória para União Explícita de Arestas

Goiânia
2018

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: **[ ] Dissertação** **[x] Tese**

2. Identificação da Tese ou Dissertação:

Nome completo do autor: Joelma de Moura Ferreira

Título do trabalho: Problemas de Otimização Combinatória para União Explícita de Arestas

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.

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.

JOELMA DE MOURA FERREIRA

Problemas de Otimização Combinatória para União Explícita de Arestas

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

Área de concentração: Ciência da Computação.

Orientador: Prof. Dr. Hugo A. D. do Nascimento

Co-Orientador: Prof. Dr. Les Foulds

Goiânia
2018

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

de Moura Ferreira, Joelma
Problemas de Otimização Combinatória para União Explícita de Arestas [manuscrito] / Joelma de Moura Ferreira. - 2018.
CXCI, 191 f.: il.

Orientador: Prof. Dr. Hugo Alexandre Dantas do Nascimento; co orientador Dr. Les Foulds.
Tese (Doutorado) - Universidade Federal de Goiás, Instituto de Informática (INF), Programa de Pós-Graduação em Ciência da Computação em rede (UFG/UFMS), Goiânia, 2018.
Bibliografia. Apêndice.
Inclui algoritmos, lista de figuras, lista de tabelas.

1. Desenho de grafos. 2. União explícita de arestas. 3. Feixes centralizados. 4. Computação evolucionária. 5. Otimização combinatória. I. Alexandre Dantas do Nascimento, Hugo, orient. II. Título.

CDU 004

UNIVERSIDADE FEDERAL DE MATO GROSSO DO SUL
UNIVERSIDADE FEDERAL DE GOIÁS
INSTITUTO DE INFORMÁTICA
DOUTORADO EM CIÊNCIA DA COMPUTAÇÃO

Ata de Defesa de Tese de Doutorado

Aos vinte e um dias do mês de março de dois mil e dezoito, no horário das nove horas, foi realizada, nas dependências do Instituto de Informática da UFG, a defesa pública da Tese de Doutorado da aluna Joelma de Moura Ferreira, matrícula no. 2013 0348, intitulada **"Problemas de Otimização Combinatória para União Explícita de Arestas"**.

A Banca Examinadora, constituída pelos professores:

Prof. Dr. Hugo Alexandre Dantas do Nascimento – INF/UFG – orientador

Prof. Dr. Leslie Richard Foulds – INF/UFG - coorientador

Profa. Dra. Carla Maria Dal Sasso Freitas – UFRGS

Prof. Dr. Fernando Vieira Paulovich – DALHOUSIE UNIVERSITY

Prof. Dr. Humberto José Longo– INF/UFG

Profa. Dra. Telma Woerle de Lima Soares – INF/UFG

emitiu o resultado:

(X) Aprovado

( ) Aprovado com revisão

(A Banca Examinadora deve definir as exigências a serem cumpridas pelo aluno na revisão, ficando o orientador responsável pela verificação do cumprimento das mesmas.)

( ) Reprovado

com o seguinte parecer: Tendo Joelma de Moura Ferreira demonstrado domínio do conhecimento, maturidade e contribuições importantes desenvolvidas no seu doutorado, bem como respondido adequadamente os questionamentos da banca, consideramos a aluna aprovada na sua defesa de doutorado.

Hugo A. D. do Nascimento
Prof. Dr. Hugo Alexandre Dantas do Nascimento

Prof. Dr. Leslie Richard Foulds

Profa. Dra. Carla Maria Dal Sasso Freitas

FVP
Prof. Dr. Fernando Vieira Paulovich

Prof. Dr. Humberto José Longo

Profa. Dra. Telma Woerle de Lima Soares

À minha tia Dedé que partiu como um anjo.

Agradecimentos

Agradeço primeiramente a Deus que é minha fortaleza e a Nossa Senhora Aparecida que me protege e guarda. Quero agradecer a minha família por entender e respeitar os momentos de ausência. Agradecer a meus pais, na figura da minha mãe, que se esforçou tanto para garantir que todos os seus filhos estudassem vendendo cocada nas ruas de Goiânia. Essa é uma vitória dela e não minha. Agradeço aos meus sobrinhos, João Victor e Theo, que são tão pequenos, e que não concebem o quanto as tardes de sábado assistindo *Discovery Kids* me ajudaram a relaxar e conseguir forças para continuar. Agradeço a meus professores, todos, do primário ao doutorado. Como professora eu sei o quanto esta profissão é linda e difícil. Não é possível mensurar em palavras o respeito que sinto por cada um deles. Começando pela tia Irani, que me ensinou a ler, e terminando com o professor Hugo que me acompanhou nesta jornada, a quem agradeço muito pela mentoria, dedicação, ajuda e paciência. Sem nossas discussões, embates, concordâncias e discordâncias eu jamais teria evoluído a este nível. Agradeço ao professor Les, que me deu uma contribuição inimaginável. Sua simplicidade e comprometimento me permitiram visualizar o que quero ser no futuro. Quero agradecer a Fundação de Amparo a Pesquisa do Estado de Goiás, que financiou a apresentação do trabalho em um evento internacional. Por fim, agradeço a todos os amigos e amigas que direta ou indiretamente contribuíram para a conclusão desta tese. Foi uma jornada longa com altos e baixos, mas satisfatória como tinha de ser.

The eye... the window of the soul, is the principal means by which the central sense can most completely and abundantly appreciate the infinite works of nature.

Leonardo Da Vinci,
Treatise on Painting.

Resumo

Ferreira, Joelma de Moura. **Problemas de Otimização Combinatória para União Explícita de Arestas**. Goiânia, 2018. 191p. Tese de Doutorado. Ciência da Computação, Instituto de Informática, Universidade Federal de Goiás.

A união de arestas em feixes é uma técnica para agrupar, alinhar, coordenar e posicionar a representação de arestas em um desenho de grafo, de modo que os conjuntos de arestas pareçam ser reunidos em estruturas visuais compartilhadas, ou seja, feixes. O objetivo final é reduzir a poluição visual do desenho melhorando a forma como ele transmite informações. Esta tese apresenta uma formulação geral para problemas de união explícita de arestas, como um problema formal de otimização. Essa formulação pode ser usada para definir e comparar problemas de união de arestas. Ainda, são definidos quatro problemas de otimização de união explícita de arestas, que têm por objetivo minimizar o número total de feixes, em conjunto com outros aspectos. Um algoritmo evolucionário é descrito. O algoritmo foi testado com sucesso em três dos problemas relacionados aplicados a instâncias do mundo real. Os resultados experimentais demonstram a eficácia e a aplicabilidade do algoritmo evolutivo proposto para ajudar a resolver problemas de união de arestas em feixes formalmente definidos como um modelo de otimização.

Palavras–chave

Desenho de grafo, união explícita de arestas, feixes centralizados, computação evolucionária, otimização combinatória.

Abstract

Ferreira, Joelma de Moura. **Combinatorial Optimization Problems for Explicit Edge Bundling**. Goiânia, 2018. 191p. PhD. Thesis. Ciência da Computação, Instituto de Informática, Universidade Federal de Goiás.

Edge bundling is a technique to group, align, coordinate and position the depiction of edges in a graph drawing, so that sets of edges appear to be brought together into shared visual structures, i.e. bundles. The ultimate goal is to reduce clutter to improve how it conveys information. This thesis provides a general formulation for the explicity edge bundling problems, as a formal combinatorial optimization problem. This allows for the definition and comparison of edge bundling problems. In addition, we present four explicity edge bundling optimization problems that address minimizing the total number of bundles, in conjunction with other aspects, as the main goal. An evolutionary edge bundling algorithm is described. The algorithm was successfully tested by solving three related problems applied to real-world instances. The reported experimental results demonstrate the effectiveness and the applicability of the proposed evolutionary algorithm to help resolve edge bundling problems formally defined as optimization models.

Keywords

Graph drawing, explicity edge bundling, centralized bundle, evolucionary computation, combinatorial optimization.

Sumário

Lista de Figuras **13**

Lista de Tabelas **17**

Lista de Algoritmos **18**

1 Introdução **19**
- 1.1 Motivação 19
- 1.2 Definição do Problema 22
- 1.3 Objetivos 24
- 1.4 Contribuições 24
- 1.5 Estrutura da Tese 25

2 Fundamentação Teórica **26**
- 2.1 União de Arestas em Feixes 26
- 2.2 Padrões de Compatibilidade 28
- 2.3 Métodos para União de Arestas em Feixes 30
- 2.4 Problemas de Otimização Relacionados à União de Feixes de Arestas 34
- 2.5 Considerações Finais 40

3 União de Arestas em Feixes como um Problema de Otimização Combinatória **42**
- 3.1 Considerações Iniciais 42
- 3.2 Caracterização de Feixes de Arestas 43
- 3.3 Metodologias para União de Arestas em Feixes 47
- 3.4 Classificação das Abordagens 51
- 3.5 Otimização da União Explícita de Arestas em Feixes (EB) 53
- 3.6 Problema de União Explícita de Arestas em Feixes Centralizados (EB-star) 55
  - 3.6.1 Descrição do Problema 55
  - 3.6.2 Modelagem Matemática 55
  - 3.6.3 NP-completude do Problema EB-star 57
- 3.7 Considerações Finais 58

4 União Explícita de Arestas em Feixes Centralizados com Restrição Angular **60**
- 4.1 Descrição do Problema 60
- 4.2 Modelagem Matemática 60
- 4.3 Considerações sobre o Ângulo Máximo do Feixe 63
- 4.4 Modelo de Programação Linear Inteira 64
- 4.5 Modelo Evolucionário 66
  - 4.5.1 Fluxo de Execução 67

4.5.2 Codificação do Indivíduo 69
4.5.3 Função de Aptidão 70
4.5.4 Função de Teste de Igualdade 71
4.5.5 Inicialização da População 71
4.5.6 Seleção 73
4.5.7 Operador de Cruzamento 76
4.5.8 Operadores de Mutação 76
4.5.9 Modelos de Visualização 78
Visualização Primária 79
Visualização Parcial de Arestas 79
4.6 Experimentos e Discussões 82
4.6.1 Calibração dos Parâmetros 84
4.6.2 Resultados 84
4.6.3 Análise da Convergência do EEB 89
4.7 Considerações Finais 92

5 União Explícita de Arestas em Feixes Centralizados com Multicritérios 94
5.1 Descrição do Problema 94
5.2 Modelagem Matemática 96
5.3 Função de Compatibilidade 97
5.4 Método Evolucionário EEB Aplicado ao Problema CBEB 99
5.5 Experimentos e Discussões 100
5.5.1 Calibração dos Parâmetros 101
5.5.2 Resultados 101
5.5.3 Análise de Convergência 112
5.6 Comparação do Problema CBEB com as Abordagens Clássicas 114
5.7 Considerações Finais 115

6 União Explícita de Arestas em Feixes Descentralizados 117
6.1 Descrição do Problema 117
6.2 Modelagem Matemática 117
6.3 Função de Compatibilidade 118
6.4 Método Evolucionário EEB Aplicado ao Problema GBEB 119
6.4.1 Cálculo da População Inicial 119
6.4.2 Operadores de Mutação 120
6.4.3 Cooperação entre Populações 121
6.5 Experimentos e Discussões 121
6.5.1 Resultados 122
6.6 Comparação dos Resultados dos Problemas Propostos 123
6.7 Considerações Finais 130

7 Conclusões 132
7.1 Publicações Decorrentes da Tese 135

Referências Bibliográficas 136

A Definições e Terminologias Relacionados ao Tema **147**
A.1 Teoria de Grafos 147
A.2 Desenho de Grafos 150
A.3 Otimização Combinatória 153
A.4 Computação Evolucionária 155

B Visualização das Melhores Soluções **162**

C Gráficos de Evolução da Qualidade das Soluções **173**
C.1 Gráficos de Evolução da Qualidade das Soluções para o Problema ABEB 173
C.2 Gráficos de Evolução da Qualidade das Soluções para o Problema CBEB 176

D Trabalhos Relacionados **180**
D.1 Trabalhos Relacionados à União Explícita de Arestas 180
D.2 Trabalhos Relacionados ao Uso de Feixes Centralizados 187

Lista de Figuras

1.1 Exemplo de união de arestas em feixes 21
1.2 Desenhos criados por abordagens diferentes de união de arestas 22

2.1 Exemplo de união de arestas em feixes em um grafo de pequena dimensão 27
2.2 Exemplo de união de arestas em feixes em um grafo de grande dimensão 27
2.3 Princípios da compatibilidade geométrica de arestas 28
2.4 Exemplo da abordagem geométrica de união de arestas 31
2.5 Exemplo da abordagem hierárquica de união de arestas 32
2.6 Exemplo da abordagem de energia de união de arestas 32
2.7 Exemplo da abordagem baseada em imagem de união de arestas 32
2.8 Diferentes desenhos com feixes de arestas para o grafo *USAirline* 35
2.9 Diferentes desenhos com feixes de arestas para o grafo *USMigration* 36
2.10 Concentração de arestas em um grafo bipartido 37
2.11 Solução para os problemas CA e MC 38
2.12 Resultado do problema de roteamento de arestas 39

3.1 Exemplo de feixe de arestas 45
3.2 Tipos de feixes 46
3.3 Ambiguidade de arestas 46
3.4 Feixes com arestas roteadas ao longo do plano 47
3.5 Exemplo de feixes e subfeixes 48
3.6 Diagrama da união explícita de arestas 49
3.7 Diagrama da união implícita de arestas 49
3.8 Resultados diferentes para feixes implícitos e explícitos 50
3.9 Arcabouço geral de otimização da união explícita de arestas 54
3.10 Solução do problema de união explícita de arestas em feixes centralizados 56
3.11 União explícita de arestas em feixes centralizados via cobertura de vértices 58

4.1 Comparação de soluções para os problemas $EB-star$ e $ABEB$ 62
4.2 Exemplos de casos relacionados à restrição angular 63
4.3 Arcabouço da abordagem evolucionária para união de arestas em feixes 68
4.4 Esquema de representação de um indivíduo no *framework* EEB 70
4.5 Esquema de teste de igualdade entre soluções 72
4.6 Exemplo de cruzamento de um ponto 76
4.7 Operador de mutação de união 77
4.8 Operador de mutação de fusão 77
4.9 Operador de mutação de divisão 77
4.10 Operador de mutação de movimentação 78
4.11 Operador de mutação de remoção 78

4.12 Exemplo de visualização primária para resultados do modelo EEB 79
4.13 Destaque de um desenho parcial de arestas 80
4.14 Visualização parcial de arestas 81
4.15 *Interface* da aplicação que implementa o modelo EEB 83
4.16 Soluções diferentes com mesmo valor de aptidão para o grafo 88
4.17 União de arestas de tamanhos diferentes no grafo *PlanarGD2015* 88
4.18 Resultados para o problema $ABEB$ com diferentes restrições angulares 90
4.19 Quantidade de gerações da melhor execução do EEB para o problema $ABEB$ 91
4.20 Convergência do grafo *Les Miserables* para o problema $ABEB$ com EEB 92
4.21 Convergência do grafo *Flare* para o problema $ABEB$ com EEB 92

5.1 Soluções diferentes para uma mesma instância do problema $ABEB$ 95
5.2 Soluções diferentes com mesmo valor de compatibilidade 98
5.3 Comparação dos resultados do problema $CBEB$ com ângulo como objetivo 104
5.4 Comparação dos resultados do problema $CBEB$ com ângulo como objetivo 105
5.5 Comparação dos resultados do problema $CBEB$ com ângulo como objetivo 105
5.6 Comparação dos resultados dos problemas $ABEB$ e $CBEB$ 106
5.7 Resultados para diferentes penalidades para o grafo *MovieLens* 108
5.8 Resultados para diferentes penalidades para o grafo *PlanarGD2015* 108
5.9 Resultados para diferentes penalidades para o grafo *LesMiserables* 109
5.10 Resultados do grafo *Sintético* para o problema $CBEB$ 109
5.11 Resultados do grafo *Zachary Club* para o problema $CBEB$ 110
5.12 Resultados do grafo *PlanarGD2015* para o problema $CBEB$ 110
5.13 Resultados do grafo *Dolphin* para o problema $CBEB$ 110
5.14 Resultados do grafo *MovieLens* para o problema $CBEB$ 110
5.15 Resultados do grafo *Les Miserables* para o problema $CBEB$ 111
5.16 Resultados do grafo *Book USPolitics* para o problema $CBEB$ 111
5.17 Resultados do grafo *Word Adjacencies* para o problema $CBEB$ 111
5.18 Resultados do grafo *Flare* para o problema $CBEB$ 111
5.19 Quantidade de gerações da melhor execução do EEB para o problema $CBEB$. 113
5.20 Convergência do grafo *MovieLens* para o problemas $CBEB$ 114
5.21 Convergência do grafo *USAirline* para o problemas $CBEB$ 114
5.22 Comparação de soluções para o grafo *USAirline* 116

6.1 Cooperação entre populações 121
6.2 Solução do problema $GBEB$ para o grafo *Planar* 124
6.3 Comparação do número de feixes da melhor solução 125
6.4 Comparação da média do número de feixes da melhor solução 125
6.5 Comparação da média de gerações 126
6.6 Comparação da média de tempo 126
6.7 Comparação de resultados para o grafo *Sintético* 127
6.8 Comparação de resultados para o grafo *Zachary Club* 128
6.9 Comparação de resultados para o grafo *Planar* 129
6.10 Comparação de resultados para o grafo *Dolphin* 130

A.1 Exemplo de um grafo bipartido completo $K_{3,3}$ 148

A.2 Exemplo de subgrafo induzido por vértices 149
A.3 Exemplo de uma cobertura de vértice 150
A.4 Desenhos diferentes de um mesmo grafo 150
A.5 Estilos de um desenho de grafo 151
A.6 Efeito da aplicação dos critérios estéticos em um desenho de grafo 153

B.1 Visualizações dos resultados para o grafo *Sintético* 163
B.2 Visualizações dos resultados para o grafo *Zachary Club* 164
B.3 Visualizações dos resultados para o grafo *PlanarGD2015* 165
B.4 Visualizações dos resultados para o grafo *Dolphin* 166
B.5 Visualizações dos resultados para o grafo *MovieLens* 167
B.6 Visualizações dos resultados para o grafo *LesMiserables* 168
B.7 Visualizações dos resultados para o grafo *Book USPolitics* 169
B.8 Visualizações dos resultados para o grafo *Word Adjacences* 170
B.9 Visualizações dos resultados para o grafo *Flare* 171
B.10 Visualizações dos resultados para o grafo *USAirline* 172

C.1 Convergência do grafo *Zachary Club* para o problema $ABEB$ com EEB 173
C.2 Convergência do grafo *PlanarGD2015* para o problema $ABEB$ com EEB 173
C.3 Convergência do grafo *Dolphin* para o problema $ABEB$ com EEB 174
C.4 Convergência do grafo *MovieLens* para o problema $ABEB$ com EEB 174
C.5 Convergência do grafo *Les Miserables* para o problema $ABEB$ com EEB 174
C.6 Convergência do grafo *Book USPolitics* para o problema $ABEB$ com EEB 175
C.7 Convergência do grafo *Word Adjacences* para o problema $ABEB$ com EEB 175
C.8 Convergência do grafo *Flare* para o problema $ABEB$ com EEB 175
C.9 Convergência do grafo *USAirline* para o problema $ABEB$ com EEB 176
C.10 Convergência do grafo *Sintético* para o problemas $CBEB$ 176
C.11 Convergência do grafo *Zachary Club* para o problemas $CBEB$ 176
C.12 Convergência do grafo *Planar GD2015* para o problemas $CBEB$ 177
C.13 Convergência do grafo *Dolphin* para o problemas $CBEB$ 177
C.14 Convergência do grafo *MovieLens* para o problemas $CBEB$ 177
C.15 Convergência do grafo *Les Miserables* para o problemas $CBEB$ 178
C.16 Convergência do grafo *Book USPolitics* para o problemas $CBEB$ 178
C.17 Convergência do grafo *Word Adjacences* para o problemas $CBEB$ 178
C.18 Convergência do grafo *Flare* para o problemas $CBEB$ 179
C.19 Convergência do grafo *USAirline* para o problemas $CBEB$ 179

D.1 Resultado da técnica de Gansner e Koren 181
D.2 Resultado da técnica de Pupyrev, Nachmanson e Kaufmann 182
D.3 Resultado da técnica IBEB 183
D.4 Resultado da técnica SBEB 183
D.5 Resultado da técnica MINGLE 184
D.6 Resultado da técnica de Bouts e Speckmann 185
D.7 Resultado da técnica 3D FDEB 186
D.8 Resultado da técnica TBEB 188
D.9 Resultado da técnica de Čermák, Dokulil e Katreniaková 189
D.10 Resultado da técnica AFEB 189
D.11 Resultado da técnica NBB 190

Lista de Tabelas

3.1 Classificação das abordagens para união de arestas em feixes 51
3.2 Funções e restrições relacionados a problemas e restrições 59

4.1 Testes preliminares com a taxa de mutação para o problema $ABEB$ 85
4.2 Solução do problema $ABEB$ para o grafo *Sintético* usando o modelo PLI 86
4.3 Solução do problema $ABEB$ para o grafo *Zachary Club* usando o modelo PLI 86
4.4 Solução do problema $ABEB$ para o grafo *Planar* usando o modelo PLI 86
4.5 Resultados para o problema $ABEB$ usando o modelo EEB 87
4.6 Quantidade de gerações da melhor execução do EEB para o problema $ABEB$ 91

5.1 Testes preliminares com a taxa de mutação para o problema $CBEB$ 102
5.2 Resultados do experimento 1 para o problema $CBEB$ 103
5.3 Resultados do experimento 2 para o problema $CBEB$ 106
5.4 Resultados do experimento 3 para o problema $CBEB$ 107
5.5 Quantidade de gerações da melhor execução do EEB para o problema $CBEB$. 113

6.1 Resultados do experimento para o problema $GBEB$ 123

Lista de Algoritmos

4.1 Algoritmo evolucionário para união explícita de arestas em feixes 69
4.2 Algoritmo para gerar a população inicial via cobertura 74
4.3 Algoritmo de seleção de indivíduos para cruzamento 75

A.1 *Algoritmo evolucionário básico* 156

CAPÍTULO 1

Introdução

1.1 Motivação

Visualização de informações é uma área de pesquisa que investiga a criação de representações visuais de dados abstratos, com o objetivo de facilitar a percepção da informação e diminuir a carga cognitiva. Para que a informação extraída de uma base seja útil, é necessário que sejam criadas visualizações efetivas que permitam identificar a estrutura global dos dados, seus padrões, tendências ou discrepâncias [77, 110, 107, 115].

Para dados que possuem algum tipo de relacionamento, um dos métodos de visualização mais utilizados é o desenho de grafos. Grafos são estruturas matemáticas utilizadas para expressar o relacionamento entre dados (identificados por vértices e arestas). A forma comumente empregada para representar visualmente essa estrutura é o *diagrama de pontos e linhas*, no qual os vértices são exibidos como pontos e as arestas como linhas que conectam esses pontos.

O desenho de um grafo, geralmente, é guiado por critérios estéticos, que são conjuntos de propriedades gráficas desejáveis que podem torná-lo mais "agradável" esteticamente para a maioria dos usuários. Os algoritmos usados para desenhar grafos de forma automática são caracterizados por satisfazer a critérios estéticos como: mostrar simetria, minimizar cruzamentos, apresentar uniformidade do tamanho das arestas e, na distribuição da posição dos vértices, maximizar a distância entre vértices e arestas, maximizar a resolução angular, etc.

Entretanto, o objetivo de um algoritmo de desenho de grafos não é apenas encontrar uma visualização que seja esteticamente "agradável", mas também que produza o melhor desenho possível, tendo como padrão um conjunto de objetivos e restrições. Para classes particulares de grafos (por exemplo, grafos planares, acíclicos e árvores), existem algoritmos que costumam gerar bons desenhos.

No entanto, grafos do mundo real, geralmente, são muito grandes e complexos, e para esses os algoritmos clássicos de desenho de grafos podem ser ineficientes. Os grafos oriundos de uma grande quantidade de dados são chamados de *grafos de grande dimensão* (do inglês *large graphs*). Provavelmente, a fonte mais conhecida de grafos de

grande dimensão seja a Internet. Na área da biologia, têm-se redes de neurônios com centenas de bilhões de vértices. Um outro exemplo são os estudos na área da física que utilizam informações com um grande número de partículas discretas que também podem ser modeladas como grafos [69].

Encontrar algoritmos que produzam bons resultados tendo como entrada grafos arbitrários de grande dimensão não é tão trivial. Um dos problemas em se lidar com essas estruturas é a perda de legibilidade causada pela sobreposição de vértices e arestas. Essa complexidade visual é chamada de *poluição visual*.

A poluição visual em um desenho de grafo prejudica tarefas como encontrar um vértice ligado a uma dada aresta ou visualizar o fluxo das informações. É meta da visualização de grafos de grande dimensão definir formas que aumentem o nível de compreensão da informação representada pela estrutura, reduzindo a complexidade e ampliando a legibilidade do desenho.

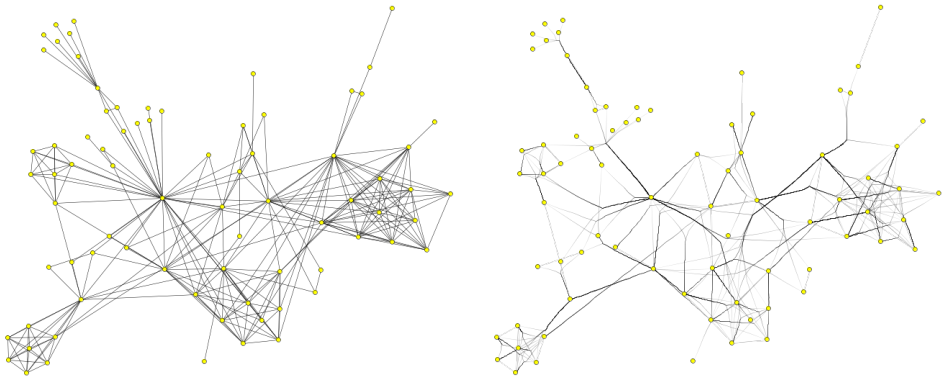
Ao longo dos anos, várias técnicas foram usadas para reduzir o problema da poluição visual de um desenho de grafo [6, 108]. As técnicas podem ser centradas nos vértices, nas arestas ou em ambos. A mais comum é a simples movimentação de vértices e arestas, mas, dependendo do tamanho do grafo, essa não é uma solução efetiva. Além de que, em alguns domínios de aplicação, os vértices não podem ser movidos, o que, em grande escala, aumenta o número de cruzamentos de arestas.

A simplificação de grafos de grande dimensão também pode ser feita por meio da eliminação de objetos do desenho, ocultação de informação ou a clusterização de vértices e/ou arestas. Tais técnicas podem ser aplicadas de forma global, restrita a uma região mais densa ou até sob ação do usuário.

Uma técnica recente de redução da poluição visual de um grafo de grande dimensão que tem ganhado muita atenção é a "união de arestas em feixes" (do inglês *edge bundling*). Ela se baseia em unir as arestas ditas compatíveis, segundo algum critério pré-estabelecido, e mostrá-las como curvas interpoladas que compartilham um mesmo caminho. Essa é uma abordagem efetiva para diminuir a complexidade visual de desenhos de grafos e descobrir padrões em alto nível. A Figura 1.1 (b) mostra um grafo com feixes de arestas gerado a partir do grafo original da Figura 1.1 (a).

Na literatura estão documentadas várias abordagens heurísticas para união de arestas em feixes. Dentre as técnicas criadas, destacam-se: *Hierarchical-Based Edge Bundling* (HBEB) [42], *Geometry-Based Edge Bundling* (GBEB) [20], *Energy-Based Edge Bundling* (EBEB) [43], *Image-Based Edge Bundling* (IBEB) [26, 27] e *Skeleton-Based Edge Bundling* (SBEB) [25].

Essas técnicas clássicas se diferenciam tanto na forma utilizada para produzir os feixes de arestas, como no resultado gerado. São abordagens que, na sua essência, visam diminuir o nível de poluição visual de forma rápida e sempre lidando com grafos cada



(a) (b)

Figura 1.1: *Exemplo de união de arestas em feixes.*

vez maiores. Um dos problemas desse tipo de investigação baseada em algoritmo é que a comparação dos resultados é restrita. Ainda não existe um consenso entre a comunidade da área com relação a quais medidas matemáticas devem ser usadas para comparar os resultados de diferentes abordagens. Geralmente, o que se emprega é avaliar a velocidade de processamento do método ou determinar informalmente o nível de redução da poluição visual.

A Figura 1.2 mostra a união de arestas de um grafo de grande dimensão (*USMigration*) usando diferentes abordagens clássicas. Percebe-se pelas figuras o quanto os resultados podem ser diferentes visualmente.

Uma possível solução para a comparação dos resultados de diferentes abordagens, seria a delimitação do problema investigado pelas abordagens. Segundo Franklin [33], citado por [68], no campo da ciência, a descrição formal do problema investigado é importante, pois é por meio dos sistemas formais que é possível caracterizar o objeto em estudo, o que pode influenciar o desenvolvimento, o reuso e a melhoria das soluções encontradas. Não existe registro (conhecido até o momento) de abordagens para união de arestas em feixes que tenham discutido formalmente o problema de otimização que é subjacente ao processo de união de arestas implementada por elas. Por vezes, é possível implicitamente inferir o objeto investigado de alguns trabalhos, como a tentativa de minimizar a quantidade de tinta gasta para desenhar as arestas e/ou a quantidade de cruzamentos, mas a discussão dos métodos algorítmicos sobrepõe a essa investigação.

Nesta tese, são propostos e discutidos alguns problemas de otimização combinatória para união de arestas em feixes, cujo objetivo principal é a minimização da quantidade de feixes. Infere-se que esses problemas sejam NP-difíceis. Tem-se como objetivo investigar o impacto da variação das restrições na dimensão dos problemas e, por consequência, nos feixes atribuídos à melhor solução.

Essa investigação foi motivada pela pesquisa de Newbery [82], a qual apresentou um dos primeiros trabalhos relacionados ao problema de diminuição da poluição visual de

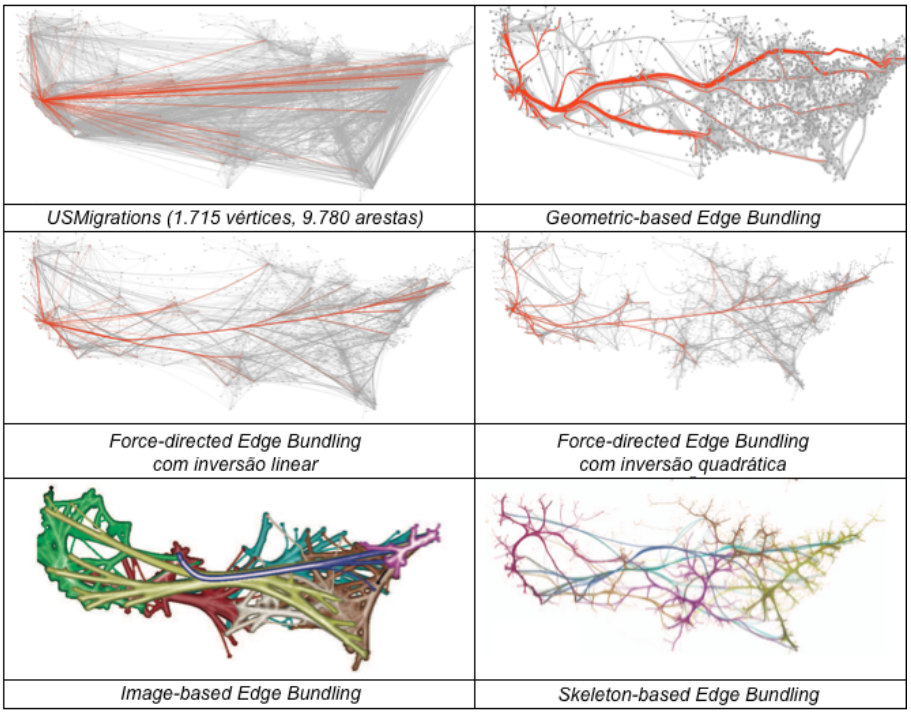


Figura 1.2: *Desenhos criados por abordagens diferentes de união de arestas.*

um desenho de grafos por meio da união de arestas. Na ocasião, a técnica foi denominada *concentração de arestas*, sendo aplicada a grafos direcionados bipartidos. O processo principal de Newbery consistiu em agrupar arestas de forma a diminuir a densidade de um grafo, mas mostrando a mesma informação do grafo original, também no modificado. Isso foi feito por meio da identificação de um conjunto de subgrafos bipartidos completos. Cada subgrafo bipartido era, então, associado a um vértice de concentração de arestas e suas arestas conectadas a esse vértice. Tal técnica diminuiu a quantidade de arestas presentes no grafo sem modificar a informação de conectividade representada na sua estrutura.

Newbery supôs que o problema de encontrar um conjunto adequado de subgrafos bipartidos era NP-completo e propôs uma solução com complexidade $O(n^4)$, sendo n o número de vértices do grafo. Lin [66] provou que esse é um problema NP-difícil. Ele o definiu assim: encontrar uma cobertura bipartida de um grafo tal que o número de arestas no grafo seja minimizado. Esse problema é muito semelhante à minimização da quantidade de feixes, discutido na presente tese.

1.2 Definição do Problema

Esta tese investiga a união de arestas em feixes em grafos estáticos como um problema de otimização combinatória. O principal problema abordado é, dado um

desenho de um grafo simples não direcionado, com vértices fixos, e objetivos e restrições associados ao desenho, decidir qual é o melhor agrupamento de arestas que otimiza a configuração desejada para os feixes.

É certo que existem diversas abordagens que produzem, de maneira muito rápida, bons desenhos que diminuem consideravelmente o nível de poluição visual. No entanto, esses métodos não são orientados a problemas, mas, sim, orientados ao algoritmo empregado. Eles se preocupam com a técnica usada para criar os feixes e não com o problema a ser resolvido. No geral, a formalização do problema investigado não é feita, o que se tem é uma inferência sobre o problema desenvolvido. Essas abordagens não se preocupam, por exemplo, em encontrar o "melhor" conjunto de feixes que respeita critérios matematicamente determinados. Essa característica dificulta a identificação do objeto investigado, o que, por consequência, leva a incertezas sobre se o que os métodos estão produzindo é a melhor solução de um problema formal.

Ao desenvolver esta pesquisa, propõe-se investigar e resolver problemas de união de arestas em feixes por meio de uma formulação matemática para uma categoria de problemas da área, e, consequentemente, desenvolver estratégias de otimização. As principais questões de pesquisa a serem respondidas, nesta tese, estão listadas abaixo:

- ***A minimização da quantidade de feixes de arestas, inserida como um critério estético, pode produzir desenhos esteticamente melhores?*** - Até o momento, não se tem conhecimento de qualquer abordagem para união de arestas em feixes que incorpore, de forma explícita, a busca por criar desenhos com a menor quantidade de feixes possível; esse geralmente é um resultado implícito do método. As técnicas propostas, em sua essência, possuem o objetivo de diminuir a poluição visual e criam os feixes com base em aspectos geométricos ou semânticos do desenho de grafo. Nenhuma abordagem usou a técnica como forma de simplificação da informação em grafos gerais, em que a minimização da quantidade de feixes[1] poderia ser uma opção.
- ***A minimização da quantidade de feixes é NP-difícil?*** - Como nenhum dos trabalhos anteriores fez uma discussão sobre esse problema, investigar tal aspecto foi parte das metas do presente trabalho de doutorado.
- ***Como determinar que um desenho com união de arestas é melhor que outro, quando os dois foram construídos com abordagens diferenciadas e se aplicam ao mesmo fim?*** - Não existem medidas ou experimentos controlados que atestem qual a melhor solução dentre as soluções criadas pelas diversas abordagens já

[1] Minimização de tinta e minimização de feixes de arestas são conceitos diferentes. A primeira procura, geralmente, minimizar o comprimento total das arestas e, assim, assegurar que a quantidade de tinta gasta para desenhar os feixes também é minimizada. A segunda busca diminuir a quantidade de feixes criados.

existentes. A definição precisa do problema objeto de estudo pode suprir esta lacuna, delimitando parâmetros que indiquem a qualidade dos desenhos.

A seguir são detalhados os objetivos do estudo apresentado nesta tese.

1.3 Objetivos

O objetivo geral deste trabalho é investigar aspectos de otimização combinatória da união de arestas em classes de grafos simples não direcionados e estáticos, por meio da proposição de problemas, implementação e avaliação de estratégias de resolução desses problemas. A concretização deste objetivo geral se desdobra nos seguintes objetivos específicos:

- teorizar a área de práticas experimentais de união de arestas em feixes, determinando conceitos, parâmetros e tipos de metodologia, que permitam refletir sobre o processo, possibilitem categorizar técnicas e problemas, e limitar os parâmetros de comparação;
- investigar e propor uma formulação matemática genérica para problemas de otimização de união de arestas que permita a ampliação de pesquisas, o uso de diferentes ferramentas e técnicas que possam ajudar a entender a essência de problemas, efetuar a análise de complexidade, e, também, a comparação da efetividade de diferentes abordagens algorítmicas;
- propor novos problemas específicos de união de arestas em feixes, tendo como base a teorização e restrições pré-estabelecidas, se possível analisando a complexidade desses problemas; e
- prover mecanismos algorítmicos para resolução dos problemas sugeridos e avaliar a sua efetividade e desempenho.

1.4 Contribuições

Esta tese propõe uma formulação geral de otimização para união explícita de arestas em feixes que pode ser usada para definir novos problemas e, consequentemente, desenvolver estratégias de otimização. Também são descritos quatro problemas de otimização combinatória cujo objetivo principal é minimizar o número total de feixes.

O primeiro problema consiste em unir em feixes disjuntos somente arestas adjacentes. Este problema foi provado ser NP-completo. O segundo problema também tem como objetivo minimizar o conjunto de feixes disjuntos do grafo, unindo somente arestas adjacentes, só que utiliza o ângulo máximo entre arestas como restrição. O terceiro problema se assemelha ao segundo, entretanto, o ângulo máximo não é mais

uma restrição, ele é inserido no problema como função objetivo que busca maximizar a compatibilidade entre as arestas. E, por fim, o quarto problema que relaxa a restrição de adjacência, sendo este um problema mais geral e próximo dos resultados produzidos por algumas técnicas clássicas.

Uma outra contribuição é a proposição de um algoritmo evolucionário para problemas de otimização combinatória para união de arestas. O uso dessa ferramenta se destaca por ser uma das primeiras abordagens para montagem de feixes usando computação evolucionária. Esta solução se mostrou efetiva na resolução dos problemas propostos.

Para o segundo problema, também foi proposta uma modelagem usando programação linear inteira, o método exato foi implementado usando o *solver* Gurobi. Os resultados foram comparados com a abordagem evolucionária.

1.5 Estrutura da Tese

Esta tese está organizada da seguinte forma. O **Capítulo 2** descreve alguns conceitos e terminologias relativos à união de arestas em feixes. Além disso, são descritos os principais trabalhos relacionados ao tema do trabalho. No **Capítulo 3** é apresentada uma caracterização teórica sobre feixes de arestas importante para posicionar o tema dessa pesquisa. É descrita também uma formulação geral de otimização e proposto um problema genérico de união de arestas em feixes. Uma prova da NP-completude desse problema é apresentada. O **Capítulo 4** descreve o segundo problema proposto para união de arestas, são feitas algumas considerações sobre a implicação da restrição angular, apresentados um modelo de programação linear inteira e uma heurística evolucionária de resolução desse problema. O **Capítulo 5** apresenta a formulação do terceiro problema, são discutidas as implicações da inserção de um outro objetivo ao modelo de otimização e apresentadas as modificações da abordagem evolucionária. O **Capítulo 6** discute o último problema apresentando experimentos com diferentes estratégias populacionais. Por fim, o **Capítulo 7** conclui o trabalho apresentado nesta tese e discute perspectivas futuras. Este trabalho também possui quatro apêndices, o **Apêndice** A, que apresenta uma breve teoria sobre grafos, otimização combinatória e computação evolucionário; **Apêndice** B, que contém as visualizações geradas para grafos testados durante os experimentos, e que não foram incluídas no corpo do trabalho; **Apêndice** C que apresenta os gráficos de evolução da qualidade ao longo do ciclo evolutivo; e o **Apêndice** D que traz uma lista de trabalhos que, de alguma forma, estão relacionados ao tema da tese.

CAPÍTULO 2

Fundamentação Teórica

Este capítulo apresenta a fundamentação teórica desta tese. Inicialmente, são definidas união de arestas e quais as características dessa técnica. São descritas as principais abordagens da área e são detalhadas algumas medidas de compatibilidade e critérios que podem ser usados para guiar um algoritmo. No final do capítulo, são relacionados trabalhos que descrevem problemas de otimização ligados a união de arestas em feixes.

2.1 União de Arestas em Feixes

Grafos do mundo real tendem a ser grandes, densos e não planares (ver Seção A.1 para elucidação desses termos). Essas características são potencializadas quando o grafo precisa ser desenhado em um espaço de visualização pequeno, tal como uma tela de computador, o que gera, na maioria do casos, o congestionamento de arestas e vértices. Essa alta densidade de elementos visuais em algumas posições do desenho de grafo é conhecida como poluição visual.

A saturação do espaço visual do desenho gerada pela poluição visual dificulta perceber as relações de adjacência entre vértices e arestas, pois estes elementos ficam sobrepostos uns sobre os outros. Como consequência, o entendimento da informação codificada pelo grafo fica prejudicado, o que pode causar a ambiguidade na interpretação dessas relações. Além disso, geralmente, essa alta densidade torna difícil entender a estrutura geral do grafo, ou seja, os caminhos formados entre os seus vértices [43].

Em 2006, Holten [42] apresentou uma técnica para simplificação de desenho de grafos que reduz tal problema, que foi denominada união de arestas em feixes (do inglês *edge bundling*). Essa técnica consiste em agrupar visualmente as arestas consideradas compatíveis, segundo algum critério, de maneira que aquelas que pertençam a um mesmo grupo sigam uma rota comum no plano e sejam desenhadas como curvas. Os grupos são identificados como feixes de arestas.

A união de arestas em feixes permite organizar o desenho visualmente, aumentar a legibilidade, minimizar cruzamentos e, ainda, revelar possíveis padrões estruturais do

grafo. A Figura 2.1 mostra um grafo pequeno e não complexo, que teve suas arestas unidas em forma de feixe. A figura da esquerda apresenta o grafo original e a da direita o mesmo grafo com feixes de arestas montados. Já a Figura 2.2 ilustra o resultado da criação de feixes de arestas em um grafo de grande dimensão. Há que se notar que o nível de poluição visual nesse grafo foi reduzido consideravelmente.

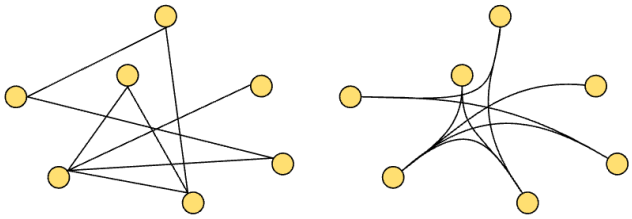


Figura 2.1: *Exemplo de união de arestas em feixes em um grafo de pequena dimensão.*

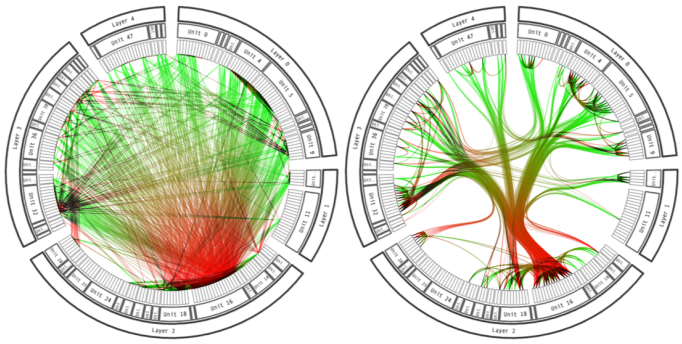


Figura 2.2: *Exemplo de união de arestas em feixes em um grafo de grande dimensão. Retirado de [19].*

Um dos objetivos dos métodos comuns de desenho de grafo é produzir desenhos que sejam considerados visualmente agradáveis, de acordo com critérios estéticos específicos (ver Apêndice A.2), como minimizar o número de cruzamentos de arestas ou maximizar a simetria, etc. No entanto, a maioria das heurísticas conhecidas para união de arestas em feixes nem sempre exploram a qualidade de um desenho por meio da avaliação de critérios clássicos.

São poucas as tentativas de incorporar e otimizar critérios estéticos comuns em desenhos com feixes de arestas. Recentemente, Angelini e outros [2] restringiram a união de aresta permitindo, no máximo, um cruzamento por feixe. Alam, Fink e Pupyrev [1] propuseram uma formulação baseada em minimizar o número de cruzamentos de feixes em grafos circulares, mas nem a prova da NP-completude do problema, ou uma heurística foram apresentadas. Saga [101] propôs duas medidas estéticas para avaliar quantitativamente a união de arestas: comprimento de arestas e área de ocupação. A eficácia dessas medidas não foi discutida.

Entre os principais critérios usados por abordagens clássicas para união de arestas em feixes destacam-se:

- cruzamentos de arestas [2, 35];
- tinta gasta para desenhar as arestas [96, 97, 120];
- comprimento de arestas [96];
- nível de poluição visual (áreas em branco) [119];
- área de ocupação (densidade) [35, 101];
- ambiguidade ao rastrear as arestas [70] (mais detalhes sobre ambiguidade será dado na Seção 3.2).

Neste trabalho, são propostas critérios adicionais que estão descritas no Capítulo 3.

2.2 Padrões de Compatibilidade

Uma outra característica importante a ser levada em consideração durante a construção de feixes é o critério usado para dimensionar o quanto duas arestas são compatíveis. A determinação de quais arestas podem pertencer a um mesmo feixe é, certamente, um dos pilares dos métodos de união de arestas.

Os diversos algoritmos podem ter uma forma própria de determinar a similaridade de arestas, mas, a partir do trabalho de Holten e Wijk [43], o uso de medidas de compatibilidade se tornou popular. Existem diversas medidas que geralmente são adotadas, em destaque: geométricas, semânticas, importância, topologia, vizinhança, temporal e conectividade.

A **compatibilidade geométrica**, também chamada compatibilidade espacial, foi definida por Holten e Wijk [43], sendo baseada em quatro medidas básicas: ângulo, escala, posição e visibilidade das arestas. O princípio de cada uma delas esta ilustrado na Figura 2.3, enquanto as fórmulas que as representam são descritas a seguir:

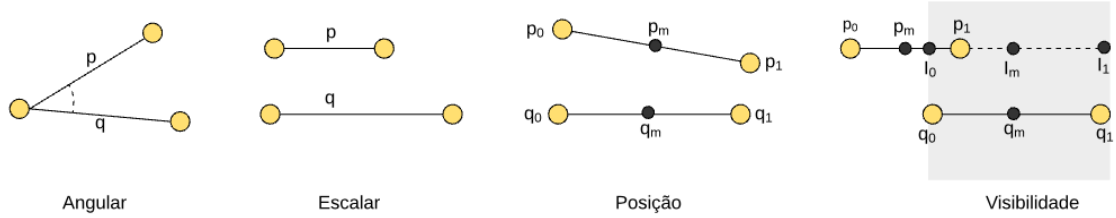


Figura 2.3: *Princípios da compatibilidade geométrica de arestas. Adaptado de [43].*

Dadas arestas $p, q \in E$, a *medida de compatibilidade angular*, $C_a(p,q) \in [0,1]$, permite prevenir a união de arestas perpendiculares (ou quase). Segundo Zhonghua e

Lingda [123], essa medida possibilita agrupar arestas que tenham uma mesma direção, sendo que, quanto menor o ângulo, mais compatíveis serão as arestas:

$$C_a(p,q) = \| \, cos(\alpha) \, \| \tag{2-1}$$

com $\alpha = arccos(\frac{p \cdot q}{|p||q|})$. O valor de $C_a(p,q)$ será 0 se p e q forem ortogonais e será 1 se forem paralelas.

A *medida de compatibilidade escalar*, $C_s(p,q) \in [0,1]$, permite prevenir que arestas com uma grande diferença de tamanho sejam unidas em um mesmo feixe. Sua formulação, conforme Holten e Wijk [43], é:

$$C_s(p,q) = \frac{2}{\ell_{avg} \cdot \min(| \, p \, |, | \, q \, |) + \max(|p|,|q|)/\ell_{avg}}, \tag{2-2}$$

com $\ell_{avg} = \frac{|p|+|q|}{2}$. O valor de $C_s(p,q)$ será 1 se p e q tiverem tamanho iguais e 0 se a razão entre a aresta maior e a menor tender ao infinito.

A *medida de compatibilidade de posição*, $C_p(p,q) \in [0,1]$, permite prevenir que arestas distantes sejam unidas, tendo como referência a distância entre o centro de cada uma (ver Figura 2.3). Sua definição formal é:

$$C_p(p,q) = \frac{\ell_{avg}}{(\ell_{avg} + \| \, p_m - q_m \, \|)}, \tag{2-3}$$

com p_m e q_m pontos médios das arestas p e q. O valor de $C_p(p,q)$ será 1 se p_m e q_m coincidirem e será 0 se $\| \, p_m - q_m \, \|$ tender ao infinito.

A *medida de compatibilidade de visibilidade*, $C_v(p,q) \in [0,1]$, permite prevenir que arestas opostas sejam unidas, e é calculada por meio da intersecção da extensão das arestas (ver Figura 2.3), sendo definida como:

$$C_v(p,q) = \min(V(p,q), V(q,p)), \tag{2-4}$$

com $V(p,q) = \max(1 - \frac{2 \cdot \|p_m - I_m\|}{\|I_0 - I_1\|}, 0)$, I_0 e I_1 os pontos de interceção da área estendida pelos extremos da aresta q em p (ver Figura 2.3) e I_m o ponto médio entre I_0 e I_1. O valor de $C_v(p,q)$ será 1 se p_m coincide com o ponto I_m, sendo esta a configuração ideal, e será 0 se p é movido para fora da faixa de visão ao longo de sua linha estendida.

A *compatibilidade geométrica total*, $C(p,q) \in [0,1]$, foi definida por Holten e Wijk [43] como:

$$C(p,q) = C_a(p,q) \cdot C_s(p,q) \cdot C_p(p,q) \cdot C_v(p,q). \tag{2-5}$$

Além da compatibilidade geométrica, existem outras medidas clássicas para computar a similaridade entre arestas. Kienreich e Seifert [55] introduziram a **compa-**

tibilidade semântica, também chamada **compatibilidade orientada a dados**, que mede o grau de relação semântica entre duas arestas. A função de compatibilidade semântica é dependente do domínio da aplicação e dos dados representados pelo grafo e deve refletir a relação do vetor de dados definido nas arestas.

Nguyen, Hong e Eades [87] apresentaram duas medidas de compatibilidade que são independentes de aspectos geométricos do desenho do grafo: importância e topologia. A **compatibilidade de importância** diz respeito ao nível de importância das arestas e também definida no domínio da aplicação. A segunda medida, **compatibilidade topológica**, leva em consideração a estrutura topológica do grafo, sendo dependente da estrutura do grafo, não sendo influenciada pela geometria do desenho.

Em 2012, Nguyen, Eades e Hong [85] apresentaram duas novas medidas de compatibilidade. A primeira é a **compatibilidade temporal**, a qual é usada para grafos em fluxo (do inglês *streaming graphs*), cujas arestas e vértices podem variar no tempo. Ela é uma medida formada pelas informações temporais das arestas, como momento de criação, duração, tempo de finalização, etc. A outra medida é a **compatibilidade de vizinhança**, a qual determina a relação causal entre uma aresta e seus vizinhos.

Por fim, Selassie [104] definiu a **compatibilidade de conectividade**, a qual também se baseia na topologia do grafo. Essa medida avalia o quão próximas as arestas estão. Para isso, usa um grafo de distância que faz com que não seja privilegiada apenas a proximidade das arestas no espaço, mas também a sua proximidade na estrutura do grafo (distância teórica). A compatibilidade de conectividade, $C_c(p,q) \in [0,1]$, foi definida como:

$$C_c(p,q) = \frac{1}{1 + D_{min}(p,q)}, \tag{2-6}$$

sendo $D_{min}(p,q)$ o número de arestas no caminho mais curto conectando os vértices de p e q.

2.3 Métodos para União de Arestas em Feixes

Existem diversos métodos para união de arestas que se diferenciam principalmente na forma utilizada para agrupar e rotear as arestas. Segundo Zhou e outros [124], as técnicas clássicas para união de arestas em feixes podem ser categorizadas como[1]:

- *Geometry-based Edge Bundling*,
- *Hierarchical-based Edge Bundling*,

[1] Sob pena de perder o sentido, o nome de cada técnica clássica para união de arestas em feixes não será traduzido neste trabalho.

- *Energy-based Edge Bundling*,
- *Image-based Edge Bundling*.

A abordagem ***Geometry-Based Edge Bundling*** [20, 61, 62, 99] é baseada na criação de uma malha que controla a formação dos feixes. Para gerar essa malha, é usado algum tipo de triangulação, como: *quadtree*, *octree*, triangulação de Delaunay ou diagrama de Voronoi. Pontos de controle definidos sobre a malha são responsáveis por controlar o roteamento das arestas. A Figura 2.4 ilustra um exemplo fictício dessa estratégia. No grafo à direita, as linhas tracejadas representam a malha, os círculos em vermelho são os pontos de controle. Uma função paramétrica usa tais pontos para definir a nova curva representante da aresta. Os feixes são formados à medida que as curvas compartilham uma mesma rota.

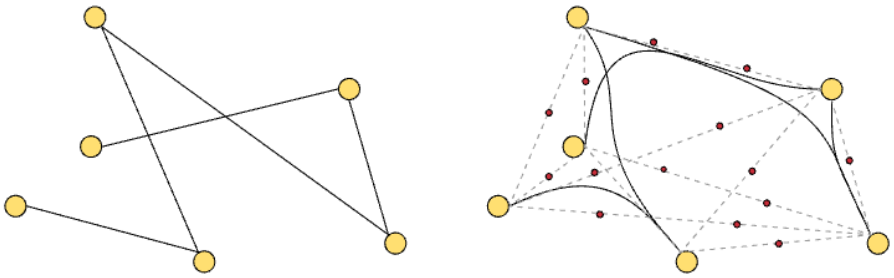


Figura 2.4: *Exemplo da abordagem geométrica de união de arestas.*

A técnica ***Hierarchical-Based Edge Bundling*** [42, 44, 45, 52, 56], geralmente, é aplicada em grafos compostos, que possuem dois tipos de arestas: aquelas que pertencem à estrutura hierárquica dos vértices e as que expressam uma relação de adjacência. Em grafos compostos, a estrutura hierárquica já é parte inerente dos dados. As arestas de adjacência são unidas em feixes por meio da sua curvatura ao longo do caminho definido pela hierarquia, tendo os vértices da árvore hierárquica como pontos de controle que guiam a função paramétrica que gera as novas curvas. A Figura 2.5 ilustra esse processo por meio de um exemplo fictício. Os vértices em preto e as linhas tracejadas representam a árvore hierárquica do grafo.

As abordagens classificadas como ***Energy-Based Edge Bundling*** [43, 87, 102, 104] representam técnicas que utilizam um modelo, cujo objetivo é mover as arestas em direção a uma força, ou baseado em uma função, até que o sistema esteja equilibrado. A Figura 2.6 ilustra o esquema de uma técnica baseada em energia denominada *Force-Directed Edge Bundling* [43]. No esquema de montagem dos feixes, primeiramente, as arestas são segmentadas em pontos. Após isso, computam-se duas forças de atração, uma força de molas F_s entre cada par consecutivo de pontos de uma aresta, e uma força de atração eletrostática F_e entre pontos correspondentes de arestas diferentes. A força exercida em um ponto é a composição destas duas e ajuda a definir a sua nova posição.

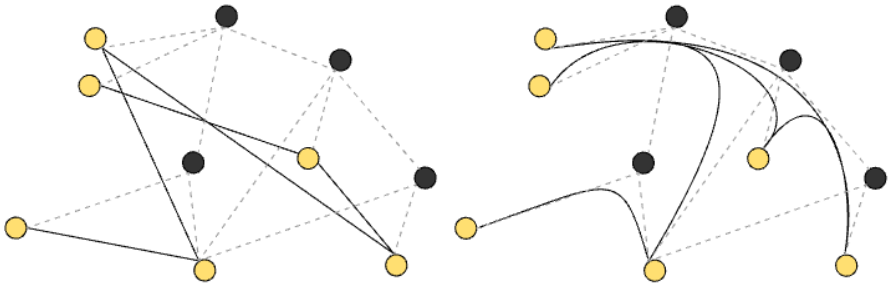


Figura 2.5: *Exemplo da abordagem hierárquica de união de arestas.*

Os vértices de origem e destino das arestas originais se mantêm fixos, apenas os pontos inseridos durante a segmentação das arestas sofrem mudança de posição. Os pontos são colocados em uma posição inicial e o sistema de força é simulado até entrar em equilíbrio quando, então, tem-se uma solução como um desenho com união de arestas. Para formar a curvatura das linhas a cada iteração, o número de segmentos por aresta aumenta.

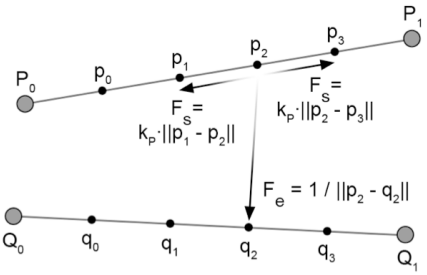


Figura 2.6: *Exemplo da abordagem energia de união de arestas. Retirado de [43].*

A técnica ***Image-Based Edge Bundling*** [25, 47, 65, 111, 113] utiliza operações de computação gráfica para montagem dos feixes de arestas. A proposta é usar como base o método de estimação de *kernel* para envolver as arestas e construir um mapa de densidade. As arestas são, então, movidas ao longo do gradiente desse mapa. O mapa de densidade reflete o grau de aglomeração local das arestas. A Figura 2.7 mostra a relação entre o mapa de densidade e o desenho de grafo com feixes.

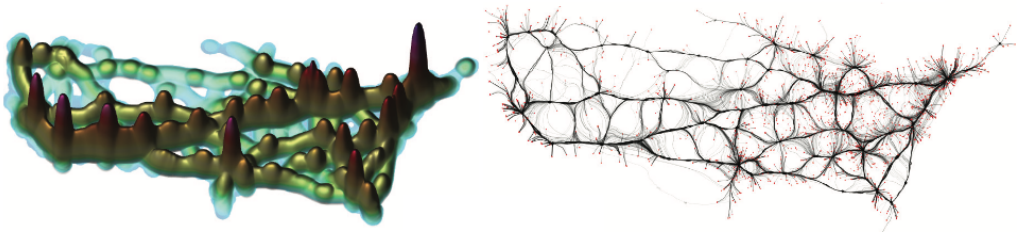


Figura 2.7: *Exemplo da abordagem baseada em imagem de união de arestas. Adaptado de [47].*

Claro que estas não são as únicas estratégias existentes. No entanto, são as que concentram um número maior de trabalhos relacionados. Mais informações sobre outras técnicas de união de arestas em feixes podem ser encontradas em *surveys* [65, 124], em livros da área [50, 108], e no Apêndice D, que descreve em mais detalhes alguns trabalhos relacionados ao tema desta tese.

Um dos problemas da união de arestas em feixes é que as diversas técnicas costumam gerar desenhos diferentes. Isso é, em parte, uma consequência das estratégias algorítmicas e medidas de compatibilidade usadas. As arestas podem ser consideradas mais ou menos compatíveis com base na estrutura do grafo, na sua posição, nos atributos dos dados que elas representam, na densidade da área do desenho onde estão localizadas ou em uma combinação desses aspectos, entre outras abordagens [25, 47].

Devido à diversidade de técnicas e resultados produzidos, não há um consenso sobre qual delas produz o melhor desenho. É notório que os diversos algoritmos produzem desenhos que diminuem a poluição visual em diferentes níveis. No entanto, ainda não foi encontrada uma medida definitiva de classificação da qualidade de cada desenho.

Existe um esforço da comunidade em estabelecer padrões que avaliem o nível de legibilidade e fidelidade que um desenho com feixes de arestas possui, e o que a literatura costuma comparar é o nível de simplificação visual do desenho. Isso é percebido pela proposta de medida de comparação de desenhos feita em [64, 119].

Recentemente, Lhuillier e outros [64] sugeriram uma medida para computar a qualidade Q de um desenho, baseado na razão entre o nível de redução da poluição C e a quantidade de distorção T:

$$Q = \frac{C}{T}. \tag{2-7}$$

A quantidade de distorção T sofrida pelo grafo com feixes é calculada a partir da medição do nível de distorção de cada aresta do grafo. Dados $D(e_i)$, o desenho de uma aresta e_i em um grafo sem feixes, $D^r(e_i)$, o desenho da mesma aresta em um grafo com feixes, e a função $\delta(D(e_i), D^r(e_i))$ que computa a distância entre os dois desenhos, a quantidade de distorção T sofrida pelo grafo com feixes é calculada como:

$$T = \sum_{i=1}^{n} \delta(D(e_i), D^r(e_i)), \tag{2-8}$$

sendo n a quantidade de arestas do grafo.

A medição do nível da redução da poluição visual C não foi definida em [64], mas Wu e outros [119] propuseram calcular esta medida baseada na redução do número de *pixels* usado no grafo:

$$C = P - P^r, \tag{2-9}$$

sendo P o número de pixels no desenho original e P^r o número de pixels no desenho com feixes. Wu e outros [119] utilizaram esta medida em um método de união de arestas baseado na aproximação dos mínimos quadrados móveis.

No entanto, segundo Nguyen, Hong e Eades [86], a simplificação visual excessiva do desenho diminui a fidelidade ao grafo original, tornando difícil a recuperação de informação local. É necessário estabelecer parâmetros que permitam criar feixes com uma ligação maior com a semântica dos dados originais, e medir o nível de agregação. Poucos foram os trabalhos que se preocuparam em dar sentido ao feixe criado. Em geral, esses casos aplicam a técnica de união de aresta a problemas específicos [96, 105].

As Figuras 2.8 e 2.9 mostram desenhos dos grafos *USAirline* e *USMigration*, respectivamente, gerados por diversos algoritmos para união de arestas. Pode ser verificado pelos exemplos o quanto os desenhos gerados são diferentes.

2.4 Problemas de Otimização Relacionados à União de Feixes de Arestas

Esta seção apresenta problemas relacionados e que envolvem ideias ligadas ao conceito de união de arestas. O primeiro é a concentração de arestas, o segundo é a minimização da cardinalidade, o terceiro trata aspectos do roteamento de arestas, e por fim, o quarto trabalho apresenta um estudo de caso que avalia parâmetros usados por algoritmos de união de arestas por meio de uma abordagem evolucionária.

Em 1989, Newbery [82] apresentou um dos primeiros trabalhos relacionados ao problema de diminuição da poluição visual de um desenho de grafos por meio da união de arestas. Usando a premissa de que grafos com poucas arestas possuem poucos cruzamentos, Newbery propôs a eliminação de algumas arestas por meio da troca de conjuntos com mesma origem e destino por um vértice falso. Na ocasião, a técnica foi identificada como *concentração de arestas*, sendo aplicada a grafos direcionados bipartidos.

O método consiste em buscar um conjunto de subgrafos bipartidos completos, que são então associados a um vértice falso cada. As arestas dos subgrafos são roteadas através desse vértice. O grafo original assume uma forma hierárquica com três níveis, com uma quantidade menor de arestas. A Figura 2.10 exemplifica tal operação. O vértice falso é denominado *vértice de concentração de arestas*.

Para garantir que o novo grafo diminua a poluição visual, a seleção dos subgrafos bipartidos completos deve assegurar o **menor número de arestas no grafo concentrado**. Esse problema é conhecido como problema de concentração de arestas (*CA*). A sua versão de otimização pode ser vista como minimizar o número de arestas de um grafo

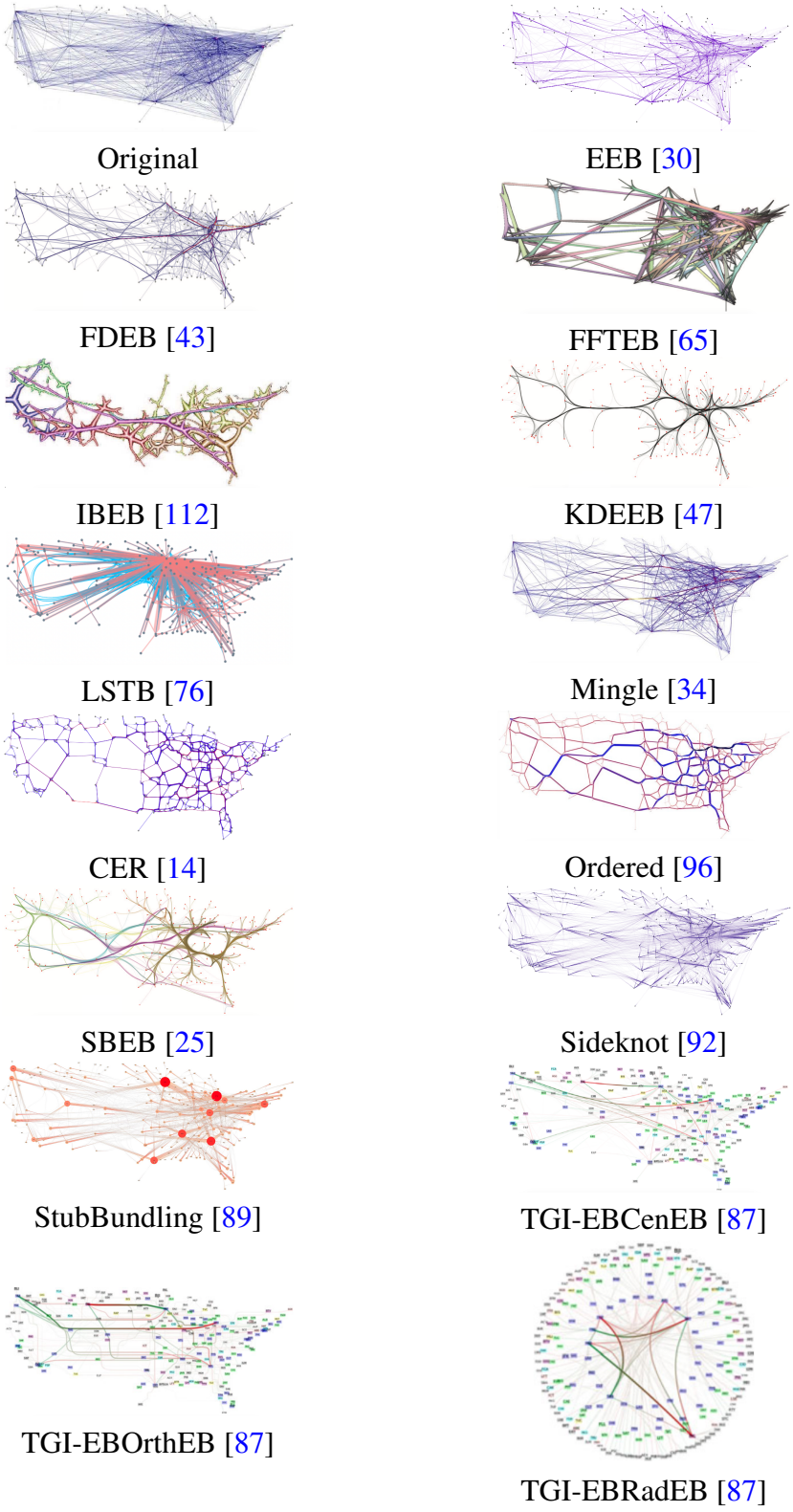


Figura 2.8: *Diferentes desenhos com feixes de arestas para o grafo USAirline. A primeira imagem é o desenho sem união de arestas.*

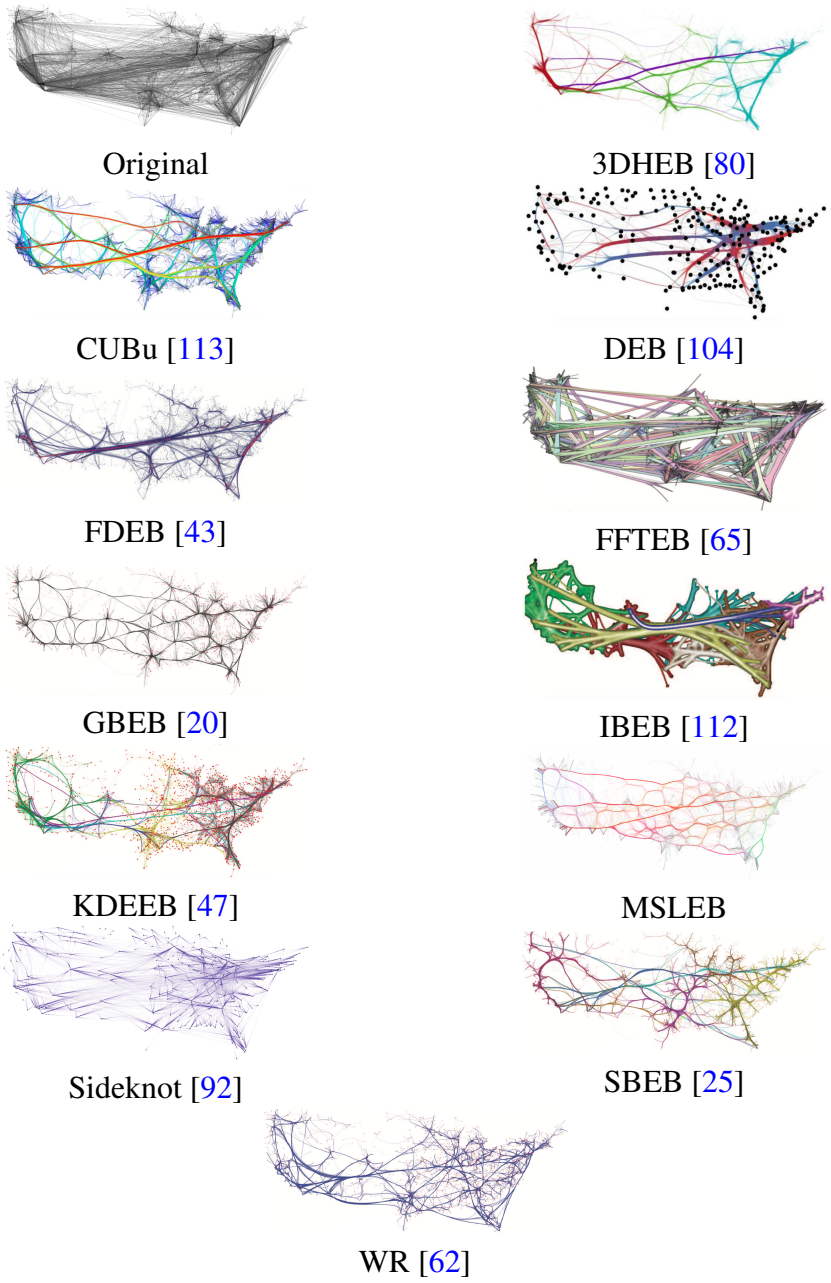


Figura 2.9: *Diferentes desenhos com feixes de arestas para o grafo USMigration. A primeira imagem é o desenho sem união de arestas.*

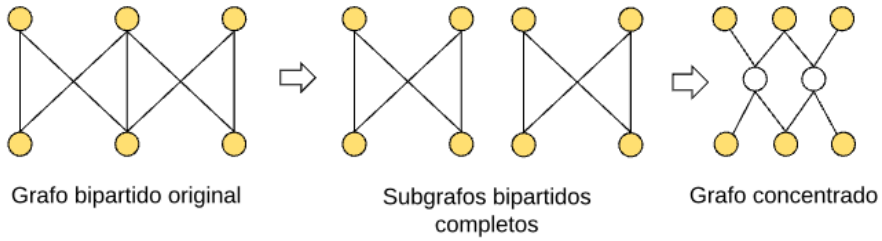


Figura 2.10: *Concentração de arestas em um grafo bipartido.*

bipartido, comprimindo aquelas que pertencem a um mesmo subgrafo bipartido completo. Newbery [82] conjeturou que o problema *CA* é NP-completo, e Lin [66] forneceu sua prova. O problema *CA* pode ser formalmente definido como:

Definição 2.1 (Concentração de arestas) *Seja um grafo bipartido $G = (V, E)$, um inteiro positivo k e a função $f(G_i)$ que retorna o número total de vértices de origem e destino no subgrafo bipartido completo G_i, o problema de concentração de arestas é decidir se existe um conjunto de subgrafos bipartidos $G_1, G_2, \ldots, G_s$ de G que cobre todas as arestas em G com $\sum_{i=1}^{s} f(G_i) \leq k$.*

Um outro problema é o da *minimização da cardinalidade* (*MC*), que também é aplicado a grafos bipartidos. Ele busca o menor número de subgrafos bipartidos completos que cobre o grafo original. Garey e Johnson [36] afirmam que este é um problema NP-completo. O problema de decisão pode ser definido como:

Definição 2.2 (Minimização da cardinalidade) *Seja um grafo bipartido direcionado $G = (V, E)$ e um inteiro positivo k, o problema de minimização da cardinalidade (MC) é decidir se existe um conjunto de subgrafos bipartidos completos $G_1, G_2, \ldots, G_s$ de G que cobre todas as arestas em G com $s \leq k$.*

Em termos de simplificação do grafo e representação visual, no *MC*, os subgrafos bipartidos completos identificados são também substituídos no grafo original por um vértice falso conectado aos vértices reais, assim como é feito na abordagem de concentração de arestas.

Apesar de parecidos, esses problemas, na verdade, são diferentes. No problema *CA*, a definição de componentes da cobertura bipartida é uma forma de chegar ao objetivo principal, que é a *diminuição do número de arestas*. Diferentemente do problema *MC*, em que a *minimização dos componentes da cobertura* bipartida é a meta. A Figura 2.11 exemplifica essa afirmação, apresentando as soluções dos dois problemas para um mesmo grafo. A Figura 2.11 (b) ilustra a solução do problema *CA* com 15 arestas e três vértices concentradores (componentes bipartidos). Apesar de ter uma quantidade maior

de concentradores de arestas, é uma solução melhor para *CA*, pois contém uma quantidade menor de arestas.

$$\sum_{i=1}^{3} f(G_i) = (2+2)+(3+4)+(2+2) = 15$$

Já o grafo da Figura 2.11 (c) é solução do problema *MC* com 16 arestas e dois vértices concentradores de arestas.

$$\sum_{i=1}^{2} f(G_i) = (2+6)+(2+6) = 16$$

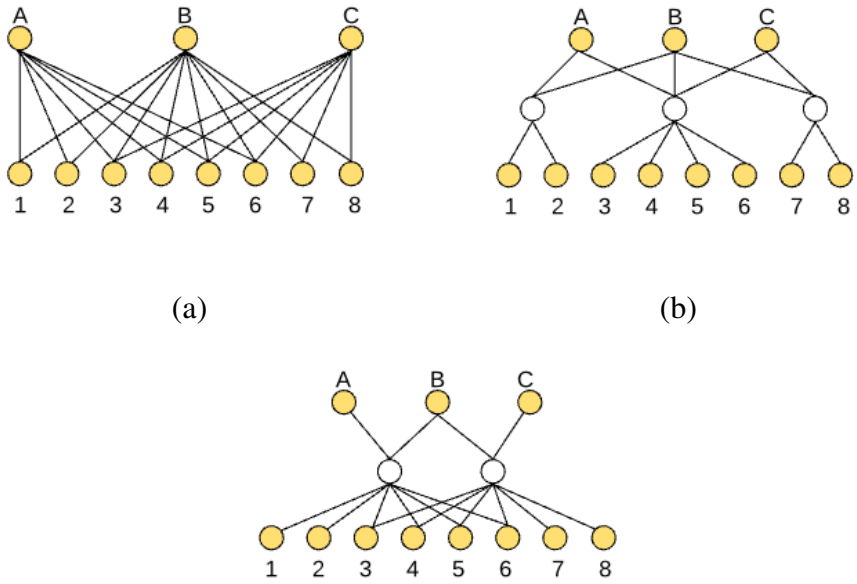


Figura 2.11: *Solução para os problemas CA e MC: (a) um grafo bipartido; (b) a solução ótima para o problema de concentração de arestas; (c) solução ótima para a minimização da cardinalidade. Adaptado de [66].*

O terceiro problema não esta relacionado à definição dos componentes, mas trata do roteamento das arestas que formam os feixes. Pupyrev e outros [96] criaram uma abordagem que não desenha as arestas dos feixes umas sobre as outras, mas executa o roteamento posicionando-as em paralelo. O elemento principal dessa estratégia é efetuar um roteamento eficiente, evitando que arestas sobreponham vértices e assegurando que os caminhos sejam relativamente curtos. A Figura 2.12 mostra o resultado da técnica de Pupyrev e outros [96]. À esquerda, tem-se o grafo original, e à direita, a sua versão com as arestas roteadas.

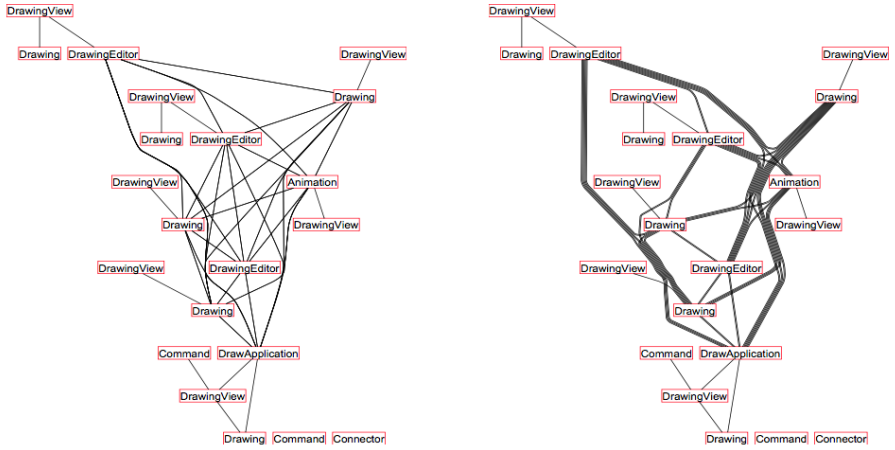


Figura 2.12: *Resultado do problema de roteamento de arestas. Retirado de Pupyrev e outros [96].*

Trata-se de uma abordagem de otimização, cujo objetivo é definir o melhor roteamento que minimiza a soma total do tamanho dos caminhos percorridos pelas arestas, em conjunto com a minimização de tinta. Eles usaram a seguinte função de custo:

$$f = \alpha \, \mathrm{INK} + \beta \sum_{p,q \in E} l_{pq}, \tag{2-10}$$

sendo l_{pq} o tamanho do caminho entre as arestas p e q no roteamento, INK é o tamanho total da união de todos os caminhos definidos no roteamento, e α e β são constantes não negativas que ponderam os componentes da função.

O problema para rotear as arestas com custo mínimo foi enunciado por Pupyrev e outros [96] como:

Definição 2.3 (Roteamento de arestas) *Seja um grafo $G = (V, E)$ com posições fixas para os vértices, rotear as arestas do conjunto E no plano de forma que o custo de roteamento seja minimizado.*

Segundo Pupyrev e outros [96], esse problema, com $\alpha = 1$ e $\beta = 0$, é equivalente ao *Geometric Steiner Tree* (*GST*). Garey e Johnson [36] provam que problema *GST* é NP-difícil.

Para resolver o problema, eles usaram um grafo de roteamento G^r, construído a partir de um grafo de visibilidade ou da triangulação de Delaunay. A estratégia foi rotear os caminhos das arestas usando esse grafo. Baseado nesse objetivo, um novo problema foi enunciado por Pupyrev e outros:

Definição 2.4 (Roteamento de caminhos) *Seja um grafo G^r e um conjunto de pares de vértices $(u_i, v_i) \in W^2$, sendo W o conjunto de vértices de G^r. Encontrar caminhos entre u_i e v_i para todo i, tal que o custo de roteamento seja minimizado.*

Esse também é um problema NP-completo, equivalente ao *Steiner Forest Problem* [36], com $\beta = 0$.

Recentemente, Polisciuc e outros [94] apresentaram um trabalho que investiga a importância de critérios estéticos em visualização de desenho de grafos e a influência de algoritmos genéticos na redução da poluição visual. Eles fizeram dois estudos de caso, um envolve desenhos de grafos circulares e o outro tratou de parâmetros para união de arestas.

No primeiro estudo ,Polisciuc e outros aplicaram algoritmos genéticos para reduzir a quantidade de cruzamentos presentes em um desenho de grafo circular. Foram testados três grafos de tamanho médio com no máximo 200 arestas cada.

No segundo estudo de caso, mais próximo do objeto de investigação desta tese, Polisciuc e outros utilizaram um *framework* evolucionáro, que envolvia parâmetros de um algoritmo para união de arestas, para encontrar uma solução próxima da ótima que otimizasse essas medidas. O método evolucionário automático avaliou soluções dos algoritmos *Swarm-based Edge Bundling* [95] e *Force-based Edge Bundling* [43] baseado na maximização do espaço em branco W no desenho com feixes e na minimização da curvatura das arestas C. A função objetivo a ser maximizada foi definida como:

$$f = \frac{W}{(1+C)}. \tag{2-11}$$

Na análise do resultados Polisciuc e outros [94] chegaram à conclusão que a medida de espaço em branco é o componente que mais contribuiu para a evolução, sendo que a minimização da curvatura foi o objetivo mais difícil de ser alcançado.

Na segunda fase do experimento dois, o usuário foi inserido no processo de evolução. Logo após o processo automático avaliar soluções usando a Equação 2-11, 12 soluções de boa, média e baixa qualidade, foram apresentadas ao usuário que deveria então pontuar cada uma delas. Esta avaliação foi normalizada e combinada em uma nova função objetivo que ponderava a avaliação do algoritmo evolucionário e do usuário. Um número de 20 soluções foram selecionadas pelo algoritmo para compor a geração corrente. O usuário poderia controlar o processo de gerar uma nova geração via interface da aplicação e decidir quando terminar o processo.

2.5 Considerações Finais

Este capítulo teve por objetivo apresentar uma fundamentação teórica, por meio da discussão, de maneira introdutória, dos principais conceitos utilizados neste trabalho. Inicialmente, a estratégia de união de aresta em feixe é definida e são detalhados os principais critérios e medidas de compatibilidade que podem ser usados para guiar o

processo de união de arestas. Algumas abordagens clássicas da área são descritas. Por fim, são apresentados problemas de otimização que envolvem as ideias precursoras do conceito de união de arestas e que serviram de motivação para esta tese.

CAPÍTULO 3

União de Arestas em Feixes como um Problema de Otimização Combinatória

Este capítulo está dividido em duas partes. A primeira parte descreve aspectos teóricos relativos à união de arestas em feixes. São descritos alguns conceitos e terminologias relativos à união de arestas em feixes, identificados e resumidos nesta tese a partir da revisão de diversas abordagens encontradas na literatura. Diferentes conceitos relacionados à definição de feixes de aresta são discutidos, bem como é apresentada uma caracterização dos tipos de feixes existentes. São definidos dois modelos de união de arestas que categorizam essas técnicas. A parte dois apresenta uma formulação geral de otimização que pode ser utilizada para definir problemas a serem resolvidos por meio da união explícita de arestas em feixes. Para exemplificar a utilidade da formulação, na parte final do capítulo, é proposto um problema genérico de união de arestas em feixes e uma prova da NP-completude desse problema é apresentada.

3.1 Considerações Iniciais

Como dito antes, o conceito comumente usado para representar a abordagem união de aresta é que esta é o agrupamento de arestas compatíveis, visando diminuir a poluição visual. No entanto, este é um conceito muito simples que não expressa todas as variantes que essa técnica pode assumir. Não foi feito, até o momento, a teorização da área que apresente de forma completa conceitos, características, modelos, problemas ou categorias de algoritmos que ajudem a investigar, explicar, comparar e até propor novos trabalhos na área, que estejam delimitados por objetos de estudo específicos. Alguns *surveys* foram propostos [65, 124], mas, utilizando categorias diferentes para explicar um mesmo trabalho. Isso demonstra o quão confusa esta área ainda se encontra no momento.

A consequência natural da falta de teoria é que, boa parte dos trabalhos dessa área não segue um processo sistemático, identificando de forma clara e direta quais as características do ambiente a ser estudado e o objeto a ser investigado, baseado em especificações formais da área. Desta maneira, a proposição e resolução de problemas de

otimização por meio da definição de variáveis e funções objetivo não é o foco, geralmente, seguido por tais trabalhos.

As abordagens tradicionais para união de arestas em feixes, no geral, são orientadas ao método empregado. Esses métodos efetuam a busca de uma solução de forma heurística, apresentado apenas uma definição informal e/ou implícita de suas metas de otimização. São conhecidos apenas três trabalhos que exploram formalmente a complexidade de problemas associados, de alguma forma, à união de arestas em feixes [82, 66, 96]. Esses trabalhos são aqueles discutidos na Seção 2.4.

No processo de união de arestas a escolha da melhor configuração dos feixes depende de uma variedade de aspectos, como o desenho corrente do grafo, a similaridade entre arestas, decisões de roteamento, ambiguidade introduzida, nível de poluição visual, etc. Para analisar a efetividade de um método de união de arestas em feixes, todos os aspectos de interesse deveriam ser considerados.

Nesse cenário, o presente trabalho segue uma linha de pesquisa diferenciada da maioria das abordagens clássicas, ao reforçar o estudo da união de arestas em feixes como um problema de otimização combinatória.

No presente capítulo é apresentada uma formulação matemática para a categoria de união explícita de arestas. O objetivo é que esta nova formulação diminua a lacuna de ferramentas teóricas sobre união de arestas em feixes, ampliando as possibilidades de pesquisas na área e permitindo o uso de diferentes técnicas que possam ajudar a entender a essência de seus problemas.

Em especial, por meio da formulação proposta, será possível definir formalmente alguns problemas de união de arestas em feixes, facilitando a análise de sua complexidade e também a comparação da efetividade de diferentes abordagens algorítmicas.

Antes de apresentar a formulação matemática e os problemas propostos, a próxima seção descreve outros conceitos e definições a respeito de união de arestas em feixes que são importantes para entender e posicionar os elementos teóricos tratados neste trabalho. O objetivo não é propor uma caracterização completa de união de arestas, sendo que o assunto não se esgota nos elementos discutidos aqui.

3.2 Caracterização de Feixes de Arestas

De forma geral, o conceito de união de arestas em feixes aceito pela comunidade é a de que essa técnica agrupa arestas compatíveis, determinando um caminho no plano de desenho para as curvas (arestas) componentes de cada feixe. No entanto, esse conceito é muito simplista para unificar e explicar todos os métodos conhecidos, uma vez que é possível encontrar na literatura diversas técnicas que montam feixes de arestas de formas diferentes, gerando resultados distintos, como visto na Seção 2.3.

Exemplos de tentativas de teorizar união de arestas em *feixes* são as de McKnight [76] e Lhuillier, Hurter e Telea [64] que, recentemente, discutiram aspectos complementares apresentando formulações matemáticas sobre união de arestas em feixes um pouco mais complexas. McKnight define um *feixe* como um conjunto de dois ou mais "segmentos de arestas" e *união de arestas* como a decisão da forma de segmentação empregada. Lhuillier, Hurter e Telea definem *feixe* como um conjunto de caminhos que compartilham similaridades e *união de arestas em feixes* como um método que cria feixes e trilhas.

Percebe-se que essas duas visões tratam de forma diferente o processo de criação dos feixes. Enquanto McKnight segmenta as arestas e busca unir tais segmentos, sem se preocupar num primeiro momento com a rota (apesar deste elemento ser intrínseco ao problema de união de arestas), Lhuillier, Hurter e Telea trabalham com a ideia de determinar, principalmente, o caminho compartilhado por onde as arestas passariam. Essas duas maneiras diferentes de tratar o que é um feixe de arestas e o processo de criá-lo, naturalmente, gera desenhos diferenciados, o que não é ruim, mas também limita o esforço de comparação, uma vez que, nem sempre é possível identificar e medir de forma direta os mesmos elementos nas soluções geradas por cada técnica ou durante o processo de união das arestas. Por exemplo, se o parâmetro de comparação fosse eficiência de segmentação e união dos segmentos gerados, isso limitaria a comparação com os resultados produzidos segundo a definição de Lhuillier. Já se a comparação fosse baseada na eficiência de roteamento, a definição de McKnight estaria em desvantagem.

De acordo com Lhuillier, Hurter e Telea [64], feixes de arestas não costumam ser definidos explicitamente pelos métodos de união de arestas. Eles podem ser identificados como um conjunto de segmentos, um conjunto de trilhas ou um conjunto de arestas, por exemplo, dependendo da abordagem. No entanto, nesta tese, um **feixe de arestas** é definido explicitamente e de forma simples como um conjunto de arestas que compartilham segmentos. A formalização dessa definição é dada a seguir.

Seja um grafo $G = (V, E)$ e um conjunto finito P de segmentos de reta no plano. Cada aresta $e_i \in E$, $i = 1, 2, \ldots, |E|$, pode ser definida como uma tupla $e_i = \{u, v, p_i | u, v \in V, p_i \in P\}$ que representa a conexão entre os vértices u e v e correspondendo ao segmento p_i, cujo tamanho é designado por $|p_i|$, sendo $|p_i| \neq 0$. Cada segmento p_i pode ser decomposto em sub-segmentos conectados, não necessariamente do mesmo tamanho, por intermédio da inserção de vértices falsos entre os extremos u e v, de forma que, $p_i = \langle p_i^1, p_i^2, p_i^3, ..., p_i^k \rangle$ com $k \in \mathbb{Z}^+$. A partir dessa representação, um feixe de aresta pode ser definido como:

Definição 3.1 (Feixe de arestas) *Um feixe de arestas é composto pelo conjunto de arestas $e_1, e_2, \ldots, e_n \in E$, se existe $n \in \mathbb{Z}$ sub-segmentos $p_1^r, p_2^s, ..., p_n^t$ coincidentes e de mesmo tamanho, pertencentes a cada aresta, respectivamente.*

Informalmente, isso significa dizer que um feixe representa um conjunto de arestas que possuem sub-segmentos do plano coincidentes associados por algum critério de similaridade pré definido. A Figura 3.1 exemplifica essa definição. Na Figura 3.1 (a), vê-se um exemplo de dois segmentos disjuntos p_1 e p_2, representantes de arestas, compostos por três sub-segmentos cada $p_1 = \langle p_1^1, p_1^2, p_1^3 \rangle$ e $p_2 = \langle p_2^1, p_2^2, p_2^3 \rangle$. Já a Figura 3.1 (b) mostra os dois segmentos unidos no feixe definido pelos sub-segmentos coincidentes $\langle p_1^2 \rangle$ e $\langle p_2^2 \rangle$.

(a) (b)

Figura 3.1: *Exemplo de feixe de arestas.*

Um outra definição importante diz respeito ao procedimento usado para construir os feixes. De acordo com Lhuillier, Hurter e Telea [64], o **processo de união de arestas** pode ser visto como a clusterização ou a segmentação da imagem de um desenho de grafo com o objetivo de formar conjuntos de trilhas ao longo do desenho. Seguindo a Definição 3.1, nesta tese será utilizado um conceito mais geral e menos baseado em métodos. Assim, a união de arestas em feixes é considerada como o processo de criar um desenho de grafo cujas arestas compartilham segmentos ao longo do plano, formando feixes de arestas. A maneira como isso é feito, se por meio de clusterização, segmentação do desenho ou outra técnica, é considerada como uma especificidade do método usado para gerar o desenho final. Formalmente, a união de feixes de arestas poderia ser definida como:

Definição 3.2 (União de arestas em feixes) *A união de feixes de arestas é a criação de um desenho de grafo em que as arestas são agrupadas por meio do compartilhamento de segmentos no plano, melhorando a legibilidade e preservando a conectividade e as informações do grafo original.*

Um método para união de arestas em feixes pode agrupar tanto arestas não adjacentes quanto adjacentes. No primeiro caso, são gerados feixes cujas arestas compartilham um ou mais segmentos comuns, e estes feixes podem possuir múltiplos vértices de origem e destino [89]. Quando são unidas apenas arestas adjacentes, estas compartilham um único vértice (origem ou destino). Uma versão híbrida também é possível, com união de arestas adjacentes e não adjacentes em um mesmo feixe. Neste caso, tem-se também feixes com múltiplos vértices de origem e destino. A grande maioria dos métodos gera fei-

xes híbridos. A Figura 3.2 (a) ilustra um feixe com múltiplos vértices de origem e destino. A Figura 3.2 (b) mostra um feixe com um vértice comum como origem ou destino.

(a) (b)

Figura 3.2: *Tipos de feixes: (a) múltiplos vértices de origem e destino; (b) um vértice comum.*

A partir deste ponto da tese, os feixes com múltiplos vértices de origem e destino são identificados como **descentralizados**. Já aqueles cujas arestas possuem todas um único vértice comum (formado por arestas adjacentes), são chamados de **centralizados**.

Segundo Peng e outros [92], o uso de feixes descentralizados ajuda a revelar a estrutura geral do grafo, mas esconde padrões relacionados às extremidades, como a adjacência, uma vez que a identificação dos vértices de origem e de destino pode ser obscurecida. No exemplo da Figura 3.3 (a), é difícil estabelecer a relação correta de adjacência dos vértices A, B, C e D. A Figura 3.3 (b) mostra, mais claramente, as duas configurações possíveis (AC, BD) ou (AD, BC). Este problema é conhecido como ambiguidade de arestas [70].

(a) (b)

Figura 3.3: *Ambiguidade de arestas: (a) feixe com ambiguidade; (b) possíveis configurações de adjacência.*

A ambiguidade de arestas é um problema para aplicações na qual é mais importante mostrar as tendências de conexões de um vértice ou grupo de vértices do que revelar a estrutura do grafo [16]. Para resolver esse problema, e com o objetivo de descobrir possíveis padrões relacionados aos vértices, alguns pesquisadores têm investigado abordagens que unem em feixes somente arestas adjacentes (feixes centralizados). Segundo Beck e outros [7], os desenhos de grafos formados apenas por feixes centralizados tendem a possuir um nível de redução da poluição visual menor do que os desenhos compostos por feixes descentralizados.

Essa estratégia já foi empregada com mapas de fluxo e redes geográficas para produzir desenhos que geram subgrafos estrela [89]. Entretanto, até o momento, são poucos os trabalhos que investigam a união em feixes de arestas adjacentes, a saber: *Traceability-based Edge Bundling*[1] [8], *Routing-based Edge Bundling*[2] [17], *Node-based Edge Bundling* [92], *Ambiguity-free Edge Bundling* [70] e *Stub Bundling* [89]. Devido à sua importância na caracterização do problema a ser estudado nesta tese, essas abordagens são discutidas brevemente no Apêndice D.2.

Uma outra característica associada aos tipos de feixe é a ideia de **subfeixes**, ou seja, segmentos de um feixe. Para os feixes descentralizados, a identificação de seus subfeixes é mais complexa pois a própria individualização do feixe é difícil de ser feita devido à ambiguidade gerada. Além do que, as arestas podem entrar e sair do feixe ao longo da sua rota (ver Figura 3.4).

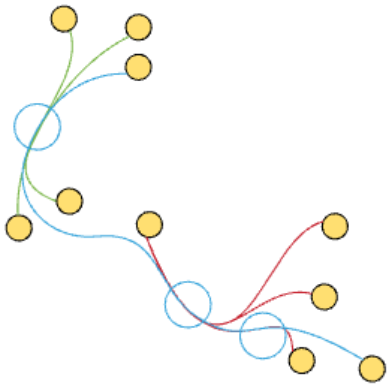


Figura 3.4: *Feixes com arestas roteadas ao longo do plano.*

Pode-se dizer que, neste caso, o conceito de feixes e subfeixes é dependente do problema. Já para feixes centralizados a definição de um padrão é mais perceptível. Este trabalho utiliza a noção de subfeixes "centralizados" definida por Čermák, Dokulil e Katreniaková [17] que estabelece que um subfeixe é formado a partir da posição onde os subfeixes se separam do feixe maior, sendo que um subfeixe "centralizado" nunca volta a se juntar a seu feixe original. Na Figura 3.5 podem ser vistos o feixe F e seus subfeixes F_1 e F_2

3.3 Metodologias para União de Arestas em Feixes

Um primeiro passo para amenizar o problema de comparação dos diferentes desenhos gerados por técnicas de união de arestas é a categorização dos métodos em torno de definições e metodologias. Esta seção apresenta a definição de duas metodologias

[1] Nomeclatura usada somente nesta tese.
[2] Nomeclatura usada somente nesta tese.

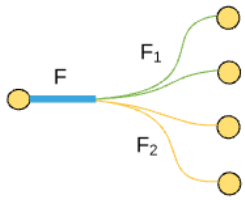


Figura 3.5: *Exemplo de feixes e subfeixes. Adaptado de [8].*

definidas a partir do estudo das abordagens clássicas de união de arestas. Elas são identificadas como implícita e explícita.

Conforme afirmam Hurter e outros [49], a construção de feixes de arestas pode ser feita de forma explícita ou implícita[3]. Nos algoritmos que utilizam a **metodologia explícita**, primeiramente, as arestas são decompostas em grupos de similaridades $E_1, E_2, \ldots, E_n$, não necessariamente disjuntos. Esses grupos são identificados como feixes e servem de entrada, por exemplo, para um algoritmo de roteamento e renderização que define a rota que cada um deve seguir no plano. A Figura 3.6 mostra um diagrama exemplificando a decomposição explícita de arestas em forma de feixes.

Observe que diversos parâmetros poderiam ser usados para definir a decomposição, como o tipo de feixe a ser criado, os critérios estéticos (ver Apêndice A) priorizados a semântica das arestas e/ou vértices, bem como medidas de compatibilidade extras baseadas no domínio da aplicação. O método de roteamento renderização é responsável tanto por formar os feixes, quanto por aplicar técnicas para renderizar os elementos do grafo (arestas, vértices e feixes). São poucos os trabalhos encontrados na literatura que poderiam ser classificados na categoria de união explícita de arestas e alguns deles são discutidos no Apêndice D.

Por outro lado, conforme afirma McKnight [76], existem diversas abordagens que não criam os feixes diretamente. Na forma **implícita** de união de arestas não existe uma pré-computação do agrupamento de arestas, sendo a saída apenas um desenho de grafo. O principal artifício utilizado é a definição de rotas[4] por onde as arestas ditas compatíveis são traçadas. É como se as arestas se "*auto organizassem*" em busca do trajeto que melhor assegure os objetivos da técnica usada.

Neste caso, por meio de um ciclo iterativo, os caminhos são definidos utilizando os parâmetros gerais do modelo e a advecção (movimentação) das arestas é feita. Ao final

[3] O termo explícito e implícito pode ser usado de duas formas nesta tese, primeiro para caracterizar a forma como as metodologias criam os feixes de arestas, ou então, para identificar se uma determinada técnica identifica o problema resolvido pela sua "metodologia" de forma implícita ou explícita.

[4] Define o caminho preciso que cada feixe, e possivelmente arestas não alocadas em nenhum feixe, segue ao longo do plano.

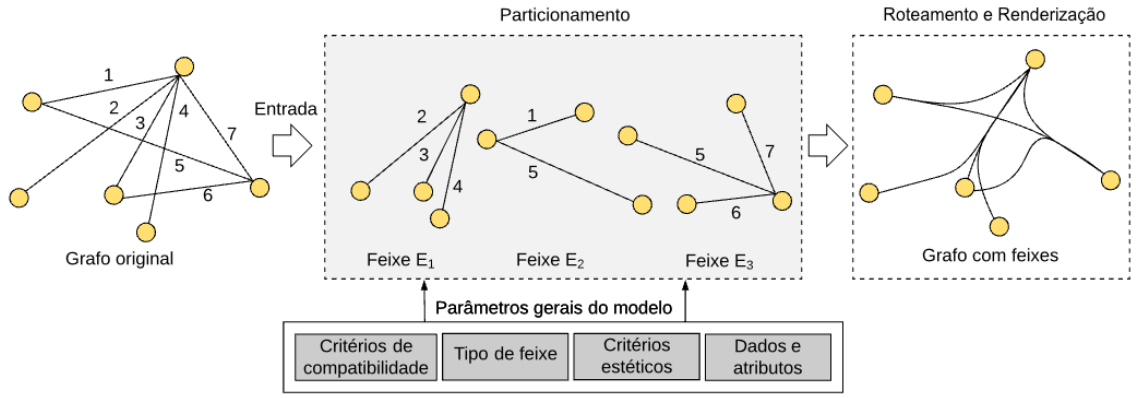


Figura 3.6: *Diagrama da união explícita de arestas.*

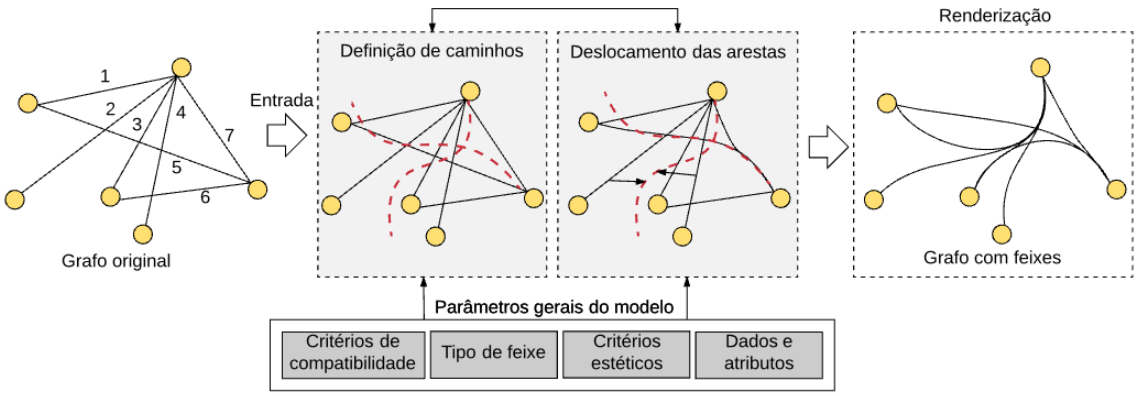


Figura 3.7: *Diagrama da união implícita de arestas.*

do processo, o desenho está pronto e o método de renderização tem a responsabilidade apenas de garantir os aspectos visuais dos elementos componentes do desenho.

Essa maneira de gerar feixes foi formalizada recentemente por Lhuillie, Hurter e Telea [64]. A maioria dos métodos encontrados na literatura foram desenvolvidos usando essa noção de feixe, São exemplos clássicos dessa metodologia, os algoritmos *Kernel Density Estimation Edge Bundling* [47] e *Force-directed Edge Bundling* [43]. A Figura 3.7 mostra um diagrama exemplificando o modelo implícito de união de arestas em feixes.

Obviamente, uma das desvantagens dos algoritmos que montam feixes de forma implícita, em comparação com o modelo explícito, é que no primeiro é mais difícil determinar os feixes individualmente. Böttger e outros [13] trataram esse problema utilizando padrões de distância entre as arestas para determinar a individualização dos feixes. Esse cálculo é feito em uma etapa de pós processamento, depois do roteamento de todas as arestas.

A Figura 3.8 ilustra a diferença entre unir arestas de forma implícita, priorizando as rotas, e de forma explícita, priorizando os agrupamentos. A Figura 3.8 (a) representa um grafo com cinco arestas. Nitidamente as arestas 1 e 2 e as arestas 3 e 4, em pares, são compatíveis entre si com relação ao tamanho. Pode-se considerar que as quatro arestas são

compatíveis com a aresta 0 se considerado somente um ângulo limite baixo entre elas, por exemplo, menor que 45°. Entretanto, as arestas 3 e 4 possuem posições opostas às arestas 1 e 2, sendo incompatíveis em posição.

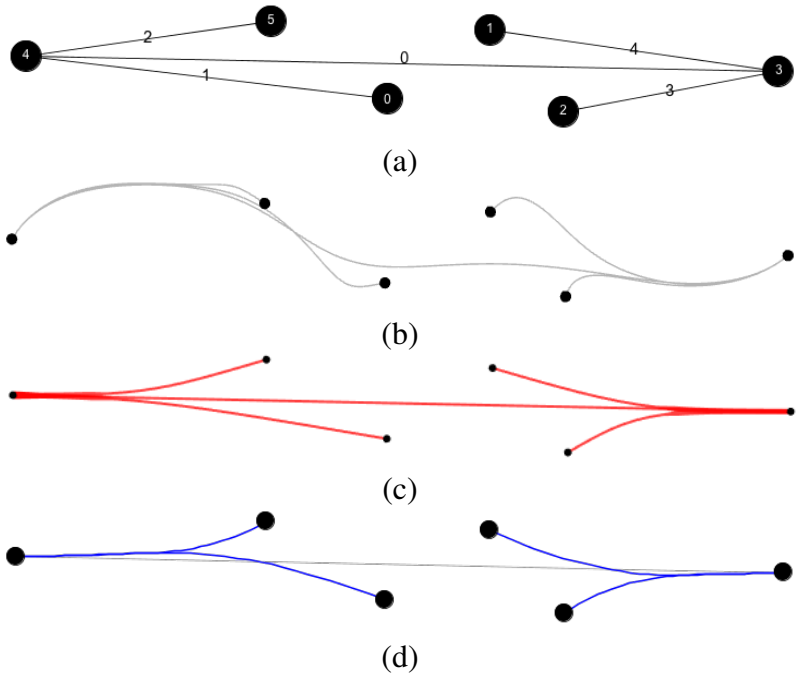


Figura 3.8: *Resultados diferentes para feixes implícitos e explícitos: (a) desenho de grafo original; (b) desenho gerado pela abordagem FDEB; (c) desenho gerado pela abordagem DEB; (d) desenho gerado pela abordagem EEB para o problema modelado CBEB.*

As Figuras 3.8 (b) e (c) foram geradas por duas abordagens de união **implícita** de arestas: a versão do FDEB (*Force-Direct Edge Bundling*) [43] implementada no software Tulip©[5] e o método DEB (*Divided Edge Bundling*) [104][6], respectivamente. Nestes dois exemplos não houve uma divisão explícita de feixes compatíveis. Os feixes são definidos ao longo de uma rota que, neste caso, poderia ser considerado o próprio trajeto principal da aresta 0. Nos trechos onde as arestas foram identificadas como compatíveis, elas foram colapsadas, compartilhando segmentos. Por exemplo, não houve a movimentação das arestas 1 e 2 no sentido de se unirem às arestas 3 e 4.

Já a união **explicita** busca priorizar os grupos de arestas que podem ser formados com base em algum critério. A Figura 3.8 (d) foi criada usando a modelagem *CBEB* e o métodos *EEB* introduzidos neste trabalho e discutidos nos Capítulos 4 e Capítulos 5. Foram formados os grupos de compatibilidade $E_1 = \{1,2\}$ e $E_2 = \{3,4\}$. A aresta 0, que

[5]Disponível em: http://tulip.labri.fr/TulipDrupal
[6]Disponível em: https://github.com/selassid/DividedEdgeBundling

se diferencia em tamanho em relação as demais, não foi alocada a nenhum grupo, ou, também é possível interpretar que ela ficou em um terceiro grupo. Os parâmetros que delimitam a compatibilidade poderiam ser alterados e, eventualmente, a aresta 0 poderia entrar no feixe E_1 ou E_2, ou até fazer uma composição de feixe único com essas quatro arestas, como neste último caso, a forma de roteamento seria mais complicada.

3.4 Classificação das Abordagens

Nas Seções 3.2 e 3.3 foram discutidas, repectivamente, os tipos de feixes existentes (descentralizado ou centralizado) e as metodologias utilizadas para criar feixes de arestas (implícita ou explícita). Usando estes parâmetros é possível classificar em categorias as principais abordagens para união de arestas em feixes.

Para tal, foi feito um levantamento dessas características nos artigos da área publicados entre 2006 e 2017, tendo sido analisados 20 artigos pertencentes à abordagem *energy-based edge bundling*, 24 da técnica *image-based edge bundling*, 44 da classe *hierarchical-based edge bundling*, 6 da *geometric-based edge bundling* e 12 artigos que não pertencem a nenhuma dessas categorias. Esses artigos foram acessados via bases de dados, indexadores e ferramentas de busca como: *ACM, IEEE, Springer, Scielo, Science Direct*, Portal Períodicos Capes, *Google Scholar e CiteSeerx*.

A Tabela 3.1 resume a classificação, enumerando algumas abordagens para união de arestas segundo tais categorias. Foram referenciados apenas os artigos que realmente definem uma nova abordagem de união de arestas ou que possuem uma importância significativa para a área. Foram suprimidos aqueles que apenas utilizam uma técnica específica ou que correspondem a alterações não relacionadas à classificação proposta nesta seção.

Observa-se na tabela que a maioria dos métodos enumerados podem ser identificados como pertencentes ao modelo Implícito/Descentralizado, ou seja, a montagem dos feixes é feita simultaneamente à convergência das arestas compatíveis para uma rota comum, sem pré-computação de um conjunto de feixes. Diferentemente, existem poucos trabalhos associados à abordagem explícita.

Tabela 3.1: *Classificação das abordagens para união de arestas em feixes.*

Técnica	**Explícito**	**Implícito**
Centralizado		[8],[17],[70],[89],[92]
Descentralizado	[14],[25],[34],[35],[74], [97],[112],[120],[129],[128]	[42],[125],[126],[43],[96],[104], [87],[127], [23],[47],[48],[46], [13], [80],[93],[118],[75],[102], [88],[129],[113], [65]

Neste trabalho, o objetivo foi além da simples redução da poluição visual, a meta foi construir desenhos de grafos compostos por feixes com algum significado no grafo ou para o usuário. Um dos aspectos importantes nesta investigação foi determinar de forma direta elementos de configuração dos feixes que nortearam variáveis de decisão e restrições dos problemas propostos e que não estejam fortemente associados a elementos visuais como a rota do feixe no plano.

Acredita-se, que, além de reduzir a desordem, uma técnica de união de arestas poderia potencialmente melhorar a legibilidade de um desenho de grafo se fosse implementada com o objetivo de otimizar um ou mais critérios estéticos relacionados à estrutura dos feixes. Como resultado, a união de arestas poderia ser formalizada como um problema de otimização multiobjetivo, onde o desenho é gerado de acordo com um determinado conjunto de critérios estéticos. São propostos nesta tese alguns critérios estéticos para união de arestas, a saber:

- minimizar o número total de feixes;
- maximizar a compatibilidade das arestas;
- maximizar o número de arestas por feixe;
- minimizar a ambiguidade;
- maximizar a simetria axial dos feixes; e
- minimizar o número total de cruzamentos entre arestas e/ou feixes.

Para este fim, a categoria de feixes explícitos (com arestas centralizadas ou descentralizadas) pode ser considerada como fortemente relacionada aos objetivos desta pesquisa, uma vez que permite a individualização dos feixes de forma direta e, consequentemente, a possibilidade natural de estabelecimento de parâmetros de configuração, como, dados relacionados às arestas, critérios estéticos, quantificação dos feixes e outros a serem configurados pelo usuário. Esses parâmetros podem ser facilmente mapeados em funções a serem otimizadas. Além do que, analisando-se o que está representado na Tabela 3.1, percebe-se ser esta uma linha pouco estudada.

Entretanto, para a ampliação organizada desse campo de estudo de forma, é necessário antes prover uma definição geral, que funcione como um arcabouço para criar e comparar problema reais nessa área. Na próxima seção é apresentada uma formulação geral para união explícita de arestas como um modelo do qual é possível derivar formulações específicas nessa categoria de problemas.

3.5 Otimização da União Explícita de Arestas em Feixes (EB)

Uma definição geral do problema de otimização de união explícita de arestas em feixes deve conter os elementos envolvidos tanto no problema de composição dos feixes pela união de arestas quanto aqueles relacionados ao problema de roteamento de feixes. Ela deve ser também genérica o suficiente para permitir que vários problemas específicos de união explícita de arestas possam ser formulados a partir dela e comparados. A seguir é dada a definição da **formulação geral de otimização para união explícita de arestas em feixes** identificada apenas como **formulação EB**:

Problema 3.3 *Seja um grafo $G = (V,E)$, D um desenho (ponto-linha e sem feixes) de G no plano e $S = (E_1, E_2, ..., E_n)$, $n \in \mathbb{N}^+$, uma decomposição de E (com cada E_i não necessariamente disjuntos). Seja também $R()$ uma função de roteamento e renderização que recebe G, D e S e produz D^r, uma versão com feixes do desenho de grafo D. O problema geral de união explícita de arestas em feixes (EB) é determinar a decomposição S (sendo cada E_i um feixe), com $E = \cup_{i=1}^{n} E_i$, tal que:*

- *um conjunto F de funções objetivo são otimizadas (minimizadas ou maximizadas), podendo representar medidas de aspectos estéticos dos feixes e de poluição visual, e*
- *um conjunto P de restrições são satisfeitas (definindo principalmente quais arestas poderiam ser unidas).*

As funções em F e as restrições em P podem ser definidas sobre G, D e D^r, bem como sobre os subconjuntos dinâmicos $E_i \in S$ a serem computados. A geração de um desenho com feixes D^r é feita por meio de $R()$. Em D^r, as arestas de G que são originalmente desenhadas como linhas em D, são agora roteadas e desenhadas como curvas e unidas em feixes se elas pertencem a um mesmo subconjunto E_i. As variáveis de decisão do problema de otimização são restritas à escolha de como decompor E em S. Isto é, o problema de união explícita de arestas em feixes é primariamente encontrar os subconjuntos $S = E_1, E_2, \ldots E_n$, de forma que o desenho correspondente D^r criado por R seja ótimo em relação a F e P, respectivamente. A Figura 3.9 mostra um arcabouço para resolver problemas de união explícita de arestas em feixes usando a formulação EB.

Apesar de focar na união explícita de arestas, o Problema 3.3 permite a inclusão do problema de roteamento como uma questão a ser tratada pelo problema de otimização. Isso já é feito quando se tem que R embute o roteamento dos feixes definidos pelos conjuntos de arestas em S. De certo modo, no Problema 3.3, a união de arestas é tratada

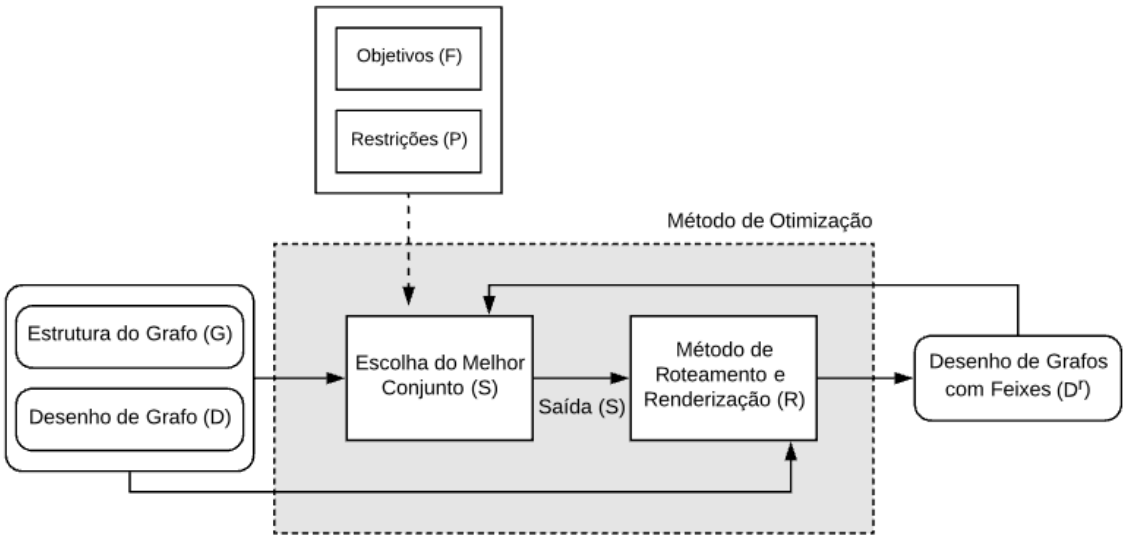


Figura 3.9: *Arcabouço geral de otimização da união explícita de arestas.*

em dois níveis, em que, no nível mais alto, agrupam-se as arestas e, no nível mais baixo, como um subproblema, faz-se o roteamento e a produção de um desenho em feixes. A escolha do conjunto S afeta o resultado de R e vice-versa.

Uma associação mais intrínseca entre os processos de união das arestas e de roteamento é possível, retirando de R a escolha da rota a ser seguida por cada feixe e trazendo esses elementos para um nível mais alto como variáveis a serem determinadas concomitantemente com a escolha de S. Neste caso, R se transformaria em uma mera função de renderização do desenho.

No presente trabalho, contudo, o roteamento não é considerado crítico para o processo de otimização. Isso foi feito porque a simples escolha de S com base em algumas funções objetivo e restrições já caracteriza um problema de otimização complexo e interessante o suficiente para estudo no âmbito desta tese.

Assim podemos entender que a formulação EB não é um problema concreto de união explícita de arestas em feixes, mas serve como um modelo para definir problemas mais específicos. A fim de definir um problema real, EB precisa ser reescrito estabelecendo formalmente e de modo claro os elementos G, D, R, F e P. Por exemplo, G pode ser um grafo direcionado, D pode ser um desenho com arestas sendo representadas como linhas retas ou ortogonais, e R pode ser um método de renderização que utiliza a posição dos vértices extremos das arestas em D e o conjunto S para montar feixes como *splines* passando por dois pontos de controle perto da posição baricêntrica dessas arestas. O conjunto P pode incluir uma ou mais restrições de compatibilidade, tais como similaridade e/ou ambiguidade, prevenindo feixes de serem formados caso infrinjam certas condições (as quais poderiam também ser restrições locais dentro de R). Um exemplo seria incluir a restrição de que a união de arestas não deve acontecer quando um determinado ângulo

limite entre elas for ultrapassado. Ainda, o conjunto F pode estar associado à qualquer aspecto visual do grafo resultante. As funções em F também podem ter mais de um objetivo. Por exemplo, a quantidade de cruzamentos ou a quantidade de tinta no desenho em feixes D^r, critérios esses a serem minimizados.

Na próxima seção é apresentado um problema que é uma especialização simples do problema EB, para o qual já é possível chegar a uma classe de complexidade computacional elevada. Além de que, esse problema é a base para a formulação de problemas mais complexos descritos nos demais capítulos desta tese.

3.6 Problema de União Explícita de Arestas em Feixes Centralizados (EB-star)

3.6.1 Descrição do Problema

Uma especialização do problema EB que representa um caso simples e potencialmente útil de união explícita de arestas, mas que tem recebido pouca atenção na literatura consiste em unir em feixes somente arestas adjacentes (como justificado na Tabela 3.1). Em uma abordagem que adota essa restrição, as arestas que formam um feixe compõem um subgrafo estrela, tendo como vértice central da estrela o extremo comum entre elas. Adicionalmente, é considerado como objetivo do problema, a minimização do tamanho de S, já que, intuitivamente quanto menos feixes o desenho tiver (considerando que cada $E_i \in S$ é um feixe, mesmo que $|E_i| = 1$), mais agrupadas estão as arestas e menos poluído será o desenho. Por este motivo, tal problema será identificado aqui como *problema de união explícita de arestas em feixes centralizados* ($EB-star$) e os feixes de uma solução como feixes centralizados de arestas.

3.6.2 Modelagem Matemática

A definição do problema de decisão para o $EB-star$ segue abaixo:

Problema 3.4 *Seja um grafo simples $G = (V,E)$, um desenho D de G com arestas representadas por linhas retas e vértices fixos e um número positivo $k \leq |E|$. A versão de decisão do problema de união explícita de arestas em feixes centralizados é determinar se existe uma decomposição S de E em subconjuntos disjuntos $E_1, E_2, \ldots, E_n$, com $E = \cup_{i=1}^{n} E_i$ e $E_i \cap E_j \neq \emptyset$, para $1 \leq i, j \leq |E|$, $i \neq j$, $n \leq k$, tal que cada E_i induz um subgrafo estrela G_i (i.e. todas as arestas em E_i compartilham o mesmo vértice), para $i, j = 1, 2, \ldots, n$.*

É claro que, quando $k = |E|$, encontrar a decomposição das arestas é trivial, representando o próprio conjunto E de arestas.

Identificando-se os elementos da formulação EB neste problema tem-se que:

- G é um grafo simples;
- D é um desenho de pontos e linhas com vértices fixos;
- o objetivo em F é encontrar um conjunto S que seja o menor possível ($|S| = n \leq k$);
- restrições em P são:
  - P_1: os subconjuntos E_i devem ser disjuntos; e
  - P_2: as arestas em E_i devem ser todas adjacentes entre si, para $i = 1, \ldots, n$.

A Figura 3.10 mostra o desenho de um grafo e a sua versão com feixes de arestas, obtida pela solução do problema de otimização $EB-star$.

(a) (b)

Figura 3.10: *Solução do problema de união explícita de arestas em feixes centralizados: (a) um desenho de grafo; (b) uma representação com feixes de arestas.*

No problema $EB-star$, aspectos visuais da união de arestas que podem ser percebidos após serem desenhados os feixes não são tratados, já que o desenho D^r não é utilizado em sua definição. Assim, assume-se que, neste caso, a função R seja um simples pós processamento que desenha cada feixe $E_i \in S$, renderizando-os usando alguma regra de roteamento, mas que não contribui para a resolução do problema de otimização.

Uma limitação do problema de otimização $EB-star$ é a possibilidade de geração de soluções com uma relação angular alta entre as arestas de um feixe. Por exemplo, uma solução poderia resultar em feixes de arestas que teriam 180° entre elas. Também a simples união de arestas de um grafo pode ser uma solução não realística pois, em particular, elas podem estar distantes umas das outras ou ter uma diferença significativa de tamanho, além de não satisfazer outros critérios de similaridade.

Uma forma de resolver este problema é usar medidas de compatibilidade que ajudam a guiar a maneira como os grupos de arestas são formados. Observando-se apenas os aspectos geométricos (ver Seção 2.2) de similaridade entre arestas, os padrões de compatibilidade poderiam ser inseridos como restrições do problema e, dessa forma, evitar-se-ia que pares de arestas adjacentes com ângulos elevados fossem unidos. A maioria das abordagens convencionais para união de arestas em feixes já faz isso por meio

do uso dessas medidas na composição de seus algoritmos e essa ideia será explorada nos próximos capítulos.

Apesar da simplicidade da definição do problema $EB-star$ e do resultado visual modesto quando aplicado a grafos de tamanho pequeno a mediano, verificou-se que este problema pertence à classe de problemas NP-completo.

3.6.3 NP-completude do Problema EB-star

A seguir será apresentada uma prova formal da NP-completude do problema de decisão de união explícita de arestas em feixes centralizados. A demonstração é feita via redução do problema de cobertura mínima de vértices (VC[7]):

Teorema 3.5 *O problema de decisão $EB-star$ é NP-completo.*

Prova. Para provar que o problema $EB-star$ é NP-completo, é necessário estabelecer que ele pertence as classes NP e NP-difícil. A primeira parte da prova é direta: seja um grafo $G = (V,E)$, um inteiro $k \leq |E|$ e um certificado $S = \{E_1, E_2, ..., E_l\}$ que representa uma decomposição de E em subconjuntos. É possível verificar em tempo polinomial se S tem no máximo k subconjuntos $(l \leq k)$, se $E_1, E_2, ..., E_l$ são disjuntos, se $E = \cup_{i=1}^{l}(E_i)$ e se cada E_i induz um subgrafo estrela. Isso mostra que $EB-star$ está em NP.

A prova de que $EB-star$ pertence à classe de problemas NP-difícil pode ser demonstrada pela redução do problema da cobertura mínima de vértices (VC). Seja um grafo $G = (V,E)$; $A \subseteq V$ é uma *cobertura de vértices* se, para toda aresta $\{u,v\} \in E, u \in A$ ou $v \in A$. A prova de que $EB-star$ é NP-difícil é baseada na observação de que uma cobertura de vértice válida para VC pode ser mapeada diretamente para uma decomposição correta de $EB-star$ de tamanho maximo k e vice-versa. Assumindo que exista uma cobertura de vértice $A = \{v_1, v_2, \ldots v_p\}$ com tamanho $p \leq k$, considere D_i o conjunto de arestas adjacentes a v_i, para todo v_i em A, $i = 1, 2, \ldots p$. Note-se que todo v_i "cobre" as arestas em D_i e que $E = \cup_{i=1}^{p} D_i$. Note-se ainda que D_i induz um subgrafo estrela de G com vértice central v_i. Os conjuntos D_i não são necessariamente disjuntos. Assim, constrói-se $E_i = \{e \in D_i\}$, de tal modo que, se também $e \in D_j, i \neq j$, a aresta e é alocada somente à decomposição E_i de menor índice e omitida das demais. Os $q \leq p$ conjuntos E_i resultantes são agora disjuntos e induzem um subgrafo estrela cada, formando uma decomposição válida para $EB-star$ com $q \leq p \leq k$ subconjuntos. Por outro lado, uma solução para $EB-star$ gera uma solução para VC. Primeiro, assuma que exista uma decomposição $S = \{E_1, E_2, ..., E_l\}$ de E com $l \leq k$, isto é, uma solução para $EB-star$.

[7] O problema de decisão da cobertura mínima de vértices é definido como: dado um grafo $G = (V,E)$ e um inteiro positivo k, decidir se existe uma cobertura de vértice $A \subseteq V$ com tamanho $|A| \leq k$ [54, 106] (ver Apêndice A para elucidação do termo cobertura de vértices).

Cada E_i, $1 \leq i \leq l$, induz um subgrafo estrela de G com vértice central v_i. Uma vez que todo vértice central v_i "cobre" as arestas em E_i, e S contêm todas as arestas em E, existe uma cobertura de vértice $A = \cup_{i=1}^{l} v_i$ com tamanho no máximo k. □

Note-se que os subgrafos induzidos por E_i, $i = 1, \ldots, l$, compartilham um mesmo vértice central. Neste caso, $A = |\cup_{i=1}^{l} (v_i)| < k$. E A ainda é uma cobertura de vértices válida.

Pela prova acima, é possível concluir também que os problemas VC e $EB-star$ são polinomialmente equivalentes. Assim, qualquer algoritmo para resolver VC pode ser utilizado para prover uma solução para $EB-star$. A Figura 3.11 mostra um exemplo de escolha de conjuntos de arestas para união explícita em feixes centralizados com base em uma solução para cobertura de vértices. A cobertura de vértices $A = \{v_1, v_2, v_3\}$ foi usada para determinar o vértice central de cada feixe.

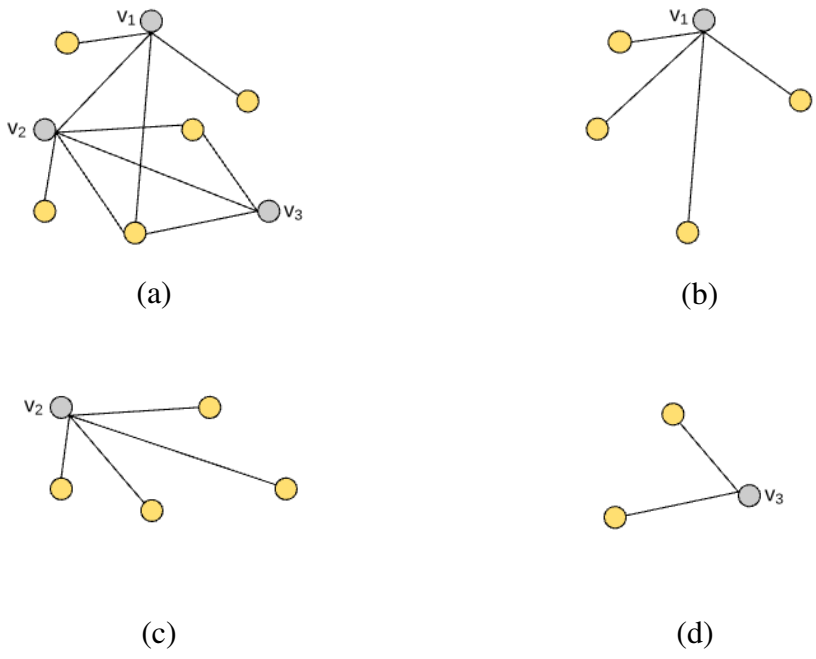


Figura 3.11: *União explícita de arestas em feixes centralizados via cobertura de vértices: (a) grafo original com cobertura $A = \{v_1, v_2, v_3\}$; (b), (c) e (d) subgrafos estrela induzidos pela cobertura de vértices sem arestas repetidas.*

3.7 Considerações Finais

Em resumo, a união de arestas em feixes lida com dois tipos de metodologias que classificam a forma como o desenho é construído e não o problema computacional

subjacente em si: o método explícito e o implícito. O primeiro, inicialmente, decide quais arestas juntar para depois estabelecer a forma de roteamento. A segunda abordagem, em sua essência, usa medidas que expressem algum sentido de compatibilidade de arestas par-a-par com o foco de resolver o roteamento.

Para ambos os casos várias funções objetivo e restrições podem ser identificadas. Essas funções e restrições podem estar relacionadas às seguintes metas detalhadas na Tabela 3.2:

Tabela 3.2: *Funções e restrições relacionados a problemas e restrições.*

Meta geral	**Meta específica: forma de atingir a meta geral**
Redução da poluição visual	- Maximizar a compatibilidade de arestas que seguem por um mesmo caminho - Minimizar a quantidade de tinta gasta - Minimizar a quantidade de cruzamentos entre arestas - Minimizar o número de feixes
Reduzir/evitar ambiguidade na compreensão da relação de adjacência entre vértices	- Não unir arestas que não sejam adjacentes - Minimizar cruzamento entre arestas

Baseado nesta constatação, esta tese trata união de aresta em feixes não como apenas uma metodologia algorítmica, mas sob o ponto de vista de problemas. Assim este capítulo delimitou o seu escopo e apresentou uma formulação geral de otimização para problemas de união explícita de arestas em feixes, identificada como EB.

O objetivo da formulação proposta é inicialmente decompor as arestas, para na sequência efetuar o seu roteamento, permitindo a configuração do conjunto de feixes conforme a necessidade do usuário, e garantindo que uma solução ótima ou próxima da ótima seja encontrada. A formulação EB foi utilizada para definir o problema de otimização de união explícita de arestas em feixes centralizados chamado $EB{-}star$, o qual visa decompor o conjunto de arestas, no menor número de conjuntos possível, construindo feixes centralizados e disjuntos.

O problema $EB{-}star$ foi provado ser NP-completo. No entanto, ele ainda é um problema não realístico de união de arestas, já que não emprega qualquer critério de compatibilidade. Nos demais capítulos, são investigados novos problemas de otimização quer estendem a definição do $EB{-}star$. Métodos para a resolução desses problemas também são apresentados.

CAPÍTULO 4

União Explícita de Arestas em Feixes Centralizados com Restrição Angular

Este capítulo apresenta um problema para união de arestas em feixes que é uma extensão do $EB-star$ discutido no Capítulo 3. Uma nova restrição foi acrescentada, a delimitação do ângulo máximo entre arestas a serem unidas. O novo problema é denominado nesta tese de *união explícita de arestas em feixes centralizados com restrição angular* ou apenas *ABEB* (sigla derivada do termo em inglês *angular-based edge bundling*). Ao longo deste capítulo, são apresentadas a formulação matemática do problema e algumas considerações sobre a implicação da restrição angular na solução gerada. Um modelo de programação linear inteira e uma heurística evolucionária de resolução desse problema são discutidos no final do capítulo.

4.1 Descrição do Problema

Conforme apresentado na Seção 3.6, o problema $EB-star$ não é realístico do ponto de vista prático, pois pode gerar feixes com arestas para as quais visualmente não faz sentido uní-las. Esta sessão apresenta um novo problema para união explícita de arestas, *ABEB*, o qual utiliza o ângulo máximo entre arestas como restrição do problema. O objetivo da inserção desta restrição é investigar como as variações angulares interferem na formação final de um desenho de grafo com feixes de arestas.

Seguindo a formulação do problema $EB-star$, o critério de compatibilidade ângulo máximo entre arestas é incluído no problema *ABEB* como uma nova restrição do conjunto P.

A definição da nova versão de decisão do novo problema é apresentada a seguir.

4.2 Modelagem Matemática

Problema 4.1 *Seja um grafo simples* $G = (V,E)$*, um desenho* D *de* G *com arestas representadas por linhas retas e vértices em posições fixas e um ângulo* $0 \leq \alpha \leq 180°$.

Seja $\gamma_{e_i e_j}$ o menor ângulo (no sentido horário e anti-horário) entre duas arestas $e_i, e_j \in E$. O problema de união explícita de arestas em feixes centralizados com restrição angular (ABEB) é determinar uma decomposição S de E em subconjuntos disjuntos $E_1, E_2, ..., E_n$, com $E = \cup_{i=1}^{n} E_i$ e $E_i \cap E_j \neq \emptyset$, para $1 \leq i, j \leq |E|$, $i \neq j$, tal que, cada E_i induz um subgrafo estrela G_i para todo par de arestas e_{ij}, $e_{ik} \in E_i$, $\gamma_{e_i e_j} \leq \alpha$, para $i, j = 1, 2, ..., n$, que minimize n.

Elementos da formulação *EB*:

- *G* é um grafo simples;
- *D* é um desenho de pontos e linhas com vértices fixos;
- o objetivo em *F* é encontrar um conjunto *S* de *E* que seja o menor possível ($n \leq k$);
- as restrições são:
  - P_1: os conjuntos E_i devem ser disjuntos;
  - P_2: as arestas de E_i devem ser adjacentes entre si; e
  - P_3: ângulo entre duas arestas de um feixe deve ser menor ou igual ao ângulo α.

Uma consequência lógica dessa definição é que o número mínimo de subconjuntos em qualquer decomposição de E vai, em geral, variar com o valor de α. Para ilustrar esta afirmação, considere o grafo original representado na Figura 4.1 (a). A Figura 4.1 (b) mostra a solução para o problema $EB-star$. No entanto, quando aplicada à restrição angular, alguns feixes da solução $EB-star$ foram separados em feixes menores e mais legíveis na solução do problema *ABEB* como mostram as Figuras 4.1 (c) e (d).

Para poder investigar a complexidade computacional do problema *ABEB*, foram definidos os valores min_γ e max_γ como sendo, respectivamente, o menor e o maior ângulo em *D* entre todos os pares de arestas adjacentes. Analisando-se então como a natureza do problema varia com a mudança da ângulo α, identificam-se três situações, as quais são descritas a seguir como Propriedades 4.2, 4.3 e 4.4.

Propriedade 4.2 *Quando $0 \leq \alpha < min_\gamma$, o tamanho da solução ótima S para ABEB é $|E|$.*

Prova. O estabelecimento de $0 \leq \alpha < min_\gamma$ torna todos os pares de arestas incompatíveis, o que resulta em uma solução trivial para *ABEB*, onde *E* é decomposto em subconjuntos disjuntos $E_1, E_2, \ldots E_{|E|}$, todos com tamanho 1. □

Propriedade 4.3 *Quando $max_\gamma \leq \alpha \leq 180°$, ABEB tem a mesma solução do $EB-star$.*

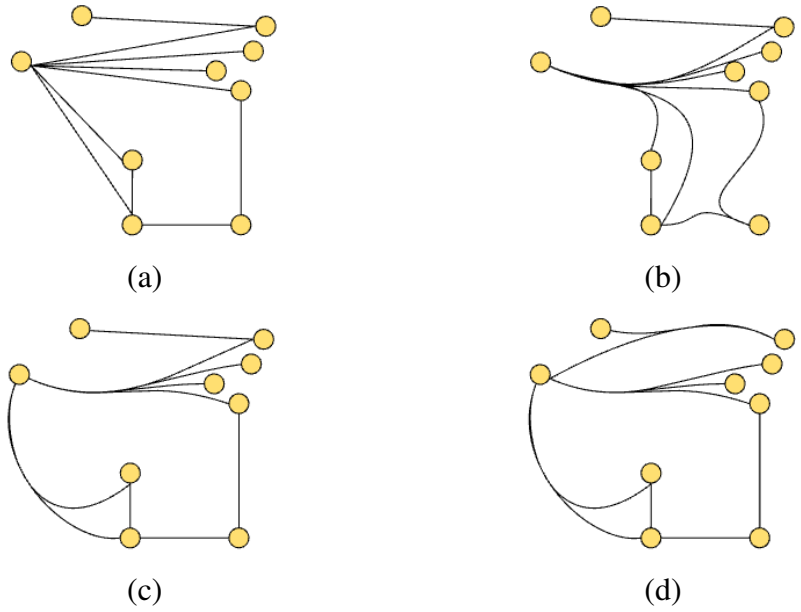


Figura 4.1: *Comparação de soluções para os problemas EB−star e ABEB: (a) desenho de grafo original; (b) uma solução para o EB−star com quatro feixes; (c) e (d) duas soluções diferentes para o ABEB com ângulo máximo* $\alpha \leq 30°$ *e cinco feixes.*

Prova. Se $max_\gamma \leq \alpha \leq 180°$, então todas as arestas adjacentes em G podem ser unidas, o que faz com que $ABEB$ seja equivalente ao $EB-star$. Assim, para esta faixa de valores α, $ABEB$ se torna NP-completo (NP-difícil para o problema de otimização). □

Propriedade 4.4 *Para um grafo G e seu desenho D, seja* S' *a decomposição das arestas de G que é uma solução para* $EB-star$ *e S que é uma solução para ABEB. Então,* $|S'| \leq |S|$.

Prova. Considerando S' a solução ótima para $EB-star$. Sendo $|S'| = p$ a quantidade de elementos de S', não existe decomposição de E, sob as condições definidas para $EB-star$, com um valor menor do que p. Por outro lado, considerando S a solução ótima de $ABEB$ e $|S|$ a quantidade de elementos de S. Como $ABEB$ esta condicionado ao mesmo objetivo e restrições do $EB-star$, a inserção do ângulo máximo α faz com $|S'| \leq |S|$, uma vez que uma solução do problema $ABEB$ pode ter um número menor de arestas compatíveis por feixe, ampliando a cardinalidade de S. □

O desafio, porém é estabelecer a complexidade de $ABEB$ quando $min_\gamma \leq \alpha < max_\gamma$. Conjectura-se que, para certos tipos de grafos, a complexidade desse problema diminui e que o número de feixes aumenta na medida em que α diminui. Entretanto, esta

investigação requer mais detalhes. Uma análise destes aspectos é um tópico interessante para futuras pesquisas.

4.3 Considerações sobre o Ângulo Máximo do Feixe

A formulação do problema *ABEB* não considera explicitamente a soma dos ângulos entre cada duas arestas consecutivas na sua estrutura seguindo uma orientação fixa (por exemplo, anti-horário). Ao contrário, a definição de *ABEB* usa somente o conceito de menor ângulo (no sentido horário ou anti-horário) entre pares de arestas adjacentes, como forma de indiretamente impor também algum limite no ângulo máximo dos feixes. Como consequência, para alguns valores altos do ângulo α, arestas adjacentes podem ainda ser unidas em um feixe de forma a deixar maior do que α o ângulo total do feixe.

Este problema ocorre quando o ângulo entre algumas arestas do feixe é medido no sentido horário, enquanto outros são medidos no sentido anti-horário. O efeito é ilustrado na Figura 4.2 (a). Neste exemplo, o grafo tem somente três arestas adjacentes (A, B, C) e o ângulo entre cada duas arestas é o mesmo para todas as arestas ($\gamma_{AB} = 120°$, $\gamma_{BC} = 120°$ e $\gamma_{CA} = 120°$). Considerando um ângulo máximo $\alpha = 135°$, a solução ótima do *ABEB* para esta instância é unir todas as arestas, o que produz um feixe, cujo ângulo total é 240°, portanto superior ao próprio α.

(a) (b)

Figura 4.2: *Exemplos de casos relacionados à restrição angular: (a) caso em que todas as arestas são unidas em um único feixe, se* $\alpha = 135°$*; (b) feixes pequenos se* $\alpha = 90°$.

Este problema não ocorre para valores pequenos de α. Considere, como exemplo a Figura 4.2 (b) com dezesseis arestas igualmente espaçadas e conectadas a um vértice central. Suponha $\alpha = 90°$, e seja E_i uma decomposição (feixe) de E qualquer, com todas as arestas em E_i adjacentes entre si e com o ângulo entre elas no máximo α. Seja A e B as arestas de E_i com maior valor de γ_{AB}. Por causa da restrição de ângulo máximo e sendo $\gamma_{AB} \leq \alpha$, é fácil mostrar para esse caso específico que γ_{AB} é também o ângulo do

feixe E_i. Isso pode ser feito observando-se que qualquer outra aresta C em E_i com uma posição distinta, tem que aparecer necessariamente entre A e B na orientação definida por γ_{AB}. Se C aparecer antes de A, então, $\gamma_{CB} = \gamma_{CA} + \gamma_{AB}$, o que contradiz a suposição de que γ_{AB} é o maior ângulo entre arestas no feixe. Uma conclusão similar pode ser feita quando se considera que C pode aparecer depois de B na orientação de A para B. Tal argumento não é válido, no entanto, para o caso de $\alpha = 135°$, porque C poderia aparecer antes de A e ainda ter um ângulo $\gamma_{CB} \leq \alpha$ se for considerado uma orientação oposta (indo de C diretamente para B sem passar por A).

De fato, é possível conceber uma regra geral para quando a restrição de ângulo máximo α entre arestas impõe também (indiretamente) uma restrição de ângulo máximo para o feixe inteiro. A ideia é ter α suficientemente pequeno de forma que, se $\gamma_{AB} = \alpha$, qualquer aresta C posicionada fora do intervalo definido pelo ângulo máximo α_{AB} terá $\gamma_{CA} > \alpha$, mesmo que avaliado em orientação oposta. Assim, uma aresta externa C (antes de A ou depois de B) nunca seria incluída no mesmo feixe de A e B. Tal condição é válida quando o ângulo máximo permitido (α) entre duas arestas A e B, mais o mesmo ângulo antes de A, adicionado a este o ângulo depois B, é menor do que 360° (ou seja, não permite C completar uma volta completa, sendo tanto compatível com A em um sentido quanto compatível com B no sentido oposto). Essa regra pode ser formulada como $3\alpha < 360°$, que implica em $\alpha < 120°$. A sua percepção é útil, porque mostra que é possível limitar o ângulo máximo da montagem de feixe somente restringindo os ângulos entre as arestas unidas (impondo $\alpha < 120°$).

Ângulos entre arestas também são mais fáceis de computar e comparar a um limite do ângulo entre feixes. Ainda, dado um grafo G, um desenho D e o valor de α, um estágio de pré-processamento pode ser executado para calcular os ângulos entre todos os pares de arestas e determinar quais pares de arestas adjacentes podem ser unidas, economizando tempo computacional nos métodos de otimização.

Os problemas e algoritmos propostos neste trabalho levam em consideração essa regra, a qual foi descoberta como parte do estudo de doutorado. Assim, foi usado apenas α sempre menor que 120°, que garante que todos os feixes terão também ângulo máximo (da soma de suas arestas em D) limitados a α.

4.4 Modelo de Programação Linear Inteira

Programação Linear Inteira (*PLI*) se refere a problemas de otimização consistindo de três partes: variáveis de decisão, uma função objetivo a ser minimizada ou maximizada e restrições. Além disso, tanto a função objetivo quanto as restrições são expressões lineares com variáveis inteiras. *PLI* é um *framework* exato de propósito geral

que pode ser utilizado para resolver diversos problemas, em particular para problemas NP-difíceis [63, 91].

Nesta seção, é introduzido um modelo de programação linear inteira com variáveis binárias (0, 1) para resolver o problema *ABEB*. A motivação para esse modelo é ter uma representação linear do *ABEB* que possa ser utilizada para gerar soluções ótimas para tal problema, pelo menos para instâncias de pequeno porte. Antes de apresentar o modelo *PLI*, é necessário introduzir primeiramente uma notação matemática importante ao seu entendimento.

Como considerado anteriormente, sejam $G = (V, E)$ um grafo e D um desenho de G no plano com arestas representadas por linhas retas. Para todo par de arestas $i, j \in E$, seja $\gamma_{i,j}$ o ângulo entre essas arestas no desenho D se i e j forem adjacentes no grafo, e $\gamma_{i,j} = \infty$ caso contrário. Para um dado ângulo α, a *compatibilidade* α*-angular* entre quaisquer duas arestas $i, j \in E$ conforme D, referenciada aqui de *compatibilidade* apenas, por simplicidade, pode ser definida como:

$$\delta_{i,j} = \begin{cases} 1, & \text{se } \gamma_{i,j} \leq \alpha, \\ 0, & \text{caso contrário.} \end{cases} \tag{4-1}$$

O modelo *PLI* visa computar uma decomposição $S = \{E_1, \ldots, E_p\}$ de E com menor cardinalidade (p deve ser o menor possível) e com cada conjunto E_k (representado um feixe), $k = 1, \ldots, p$, composto apenas por arestas compatíveis entre si. Obviamente, a maior cardinalidade possível de S é $p = |E|$, o que, caso aconteça, significa que cada aresta ficou sozinha em um subconjunto E_k. Para formular então o modelo de programação linear, supõe-se que há $m = |E|$ subconjuntos (feixes) disponíveis e define-se uma variável lógica x_{ik} que assume o valor um (1) caso a aresta $i \in E$ seja alocada ao subconjunto E_k e valor zero (0), caso contrário. O modelo completo é definido como segue:

Minimizar:

$$Z_{ABEB} = \sum_{i \in E} \sum_{k=1}^{m} k x_{ik} \tag{4-2}$$

sujeito a

$$\sum_{k=1}^{m} x_{ik} = 1, \; i \in E, \tag{4-3}$$

$$x_{ik} + x_{jk} \leq 1, \; i, j \in E, \delta_{i,j} = 0, k = 1, \ldots, m, \tag{4-4}$$

$$x_{ik} \in \{0, 1\}, \; i \in E, k = 1, \ldots, m, \tag{4-5}$$

onde $x_{ik} = 1, \text{ se } i \in E_k, x_{ik} = 0, \text{ caso contrário}, i \in E, k = 1, \ldots, m.$

A função objetivo no modelo adiciona um custo k para cada aresta alocada ao subconjunto E_k. Como o problema envolve minimizar essa função, a solução ótima deve alocar as arestas à menor quantidade possível de subconjuntos e utilizar apenas os subconjuntos de menor índice k. As restrições 4-3 asseguram que cada aresta é alocada a exatamente um subconjunto da decomposição. Isso implica também que os subconjuntos E_k serão disjuntos. As restrições 4-4 impedem que arestas incompatíveis sejam alocadas a um mesmo subconjunto, garantindo assim a condição de compatibilidade angular entre as arestas que irão compor um mesmo feixe. As restrições 4-5 são as condições binárias usuais que refletem o fato de cada aresta estar associada a um subconjunto particular ou não.

Apesar de ter uma formulação curta, o tamanho do modelo *PLI* supracitado em termos quantidade de variáveis e de restrições cresce significativamente com o aumento da quantidade de arestas do grafo e com a escolha do valor de α. Quanto menor for o valor de α, provavelmente maior será a quantidade de pares de arestas consideradas incompatíveis. Seja L_α essa quantidade para um dado α (nota-se que $0 \leq L_\alpha \leq (|E|^2 - |E|)/2$, assumindo pares distintos não-ordenados de arestas). A quantidade total de variáveis x_{ik} no modelo é $|E|^2$, já que há $|E|$ arestas e $|E|$ possíveis subconjuntos. A função objetivo é, portanto, um somatório de $|E|^2$ termos. O modelo tem $|E|$ restrições do tipo 4-3, uma para cada aresta $i \in E$, e todas contendo um somatório com $|E|$ termos. A restrições do tipo 4-4 são curtas, mas há $L_\alpha \cdot |E|$ delas. Por fim, há $|E|^2$ condições de integralidade (como 0 ou 1) das variáveis, definidas pelas restrições do tipo 4-5.

Como ilustração do tamanho do modelo, uma instância com $|E| = 200$ e $L = 9950$ (isto é, metade dos pares de arestas sendo incompatíveis) implica em 2.030.200 restrições e 40.000 variáveis. Isso já caracteriza um problema de tamanho elevado para vários métodos de solução de *PLI*, o que justifica o emprego de abordagens heurísticas e de meta-heurísticas para resolver o *ABEB*. Deste modo, a próxima seção apresenta uma abordagem para o *ABEB* baseada em um método evolutivo. Enquanto a resolução do *ABEB* usando *PLI* deve ser impossível para instâncias grandes (com o recursos de *hardware* atual), é esperado que a abordagem evolutiva (ver Seção A.4) ainda consiga produzir uma boa solução em um tempo computacional aceitável.

4.5 Modelo Evolucionário

Um dos desafios de lidar com problemas de otimização combinatória complexos é desenvolver algoritmos que garantam encontrar uma solução razoavelmente boa em um tempo computacional aceitável. O modelo baseado em programação linear inteira proposto, apesar de resolver o problema devolvendo o ótimo global é impraticável em termos de tempo de execução para grandes instâncias (ver Seção 4.6).

Uma alternativa a este tipo de abordagem é a chamada computação evolutiva, uma metaheurística de otimização baseada em população, advinda do campo da inteligência artificial. Um algoritmo evolutivo é um subconjunto da computação evolucionária, que usa mecanismos inspirados na evolução biológica, tais como reprodução, mutação, recombinação e seleção [4].

Neste capítulo, é descrito um algoritmo evolucionário para o problema de união explícita de arestas em feixes, sendo aplicado ao problema *ABEB* com o intuito de produzir uma solução local ótima, boa o suficiente e em um tempo aceitável. O método foi identificado como **algoritmo evolucionário para união de arestas em feixes**, ou simplesmete *EEB* (sigla do termo em inglês *Evolutionary Edge Bundling*). A computação evolucionária é uma técnica que foi pouco testada na construção de feixes de arestas, pelo que se sabe apenas o trabalho de [94] abordou tal estratégia (ver Seção 2.4). A descrição do fluxo de execução e o detalhamento dos componentes da abordagem *EEB* são apresentados a seguir.

4.5.1 Fluxo de Execução

O algoritmo *EEB* adota o ciclo evolucionário básico envolvendo os passos de inicialização, avaliação, seleção, cruzamento e mutação, conforme pode ser visto na Figura 4.3. O processo começa recebendo um grafo G e um desenho D desse grafo que são, então, repassados ao módulo de criação da população inicial ($P_{inicial}$). São usados dois algoritmos distintos, um que segue uma heurística baseada no cálculo da cobertura de vértices e outro completamente aleatório. Cada um cria metade da população. É feito, ainda, o controle da duplicidade de soluções para evitar que um indivíduo esteja duplicado na população.

Na sequência, são selecionados, para cruzamento, dois indivíduos I_{p1} e I_{p2} e gerados dois indivíduos filhos I_{f1} e I_{f2}. Esses indivíduos recém-criados sofrem uma mutação para, em seguida, competirem pela sobrevivência com os indivíduos da população corrente. Esse processo se repete até que t indivíduos sejam criados e uma nova geração (P_{nova}) seja gerada. Neste ponto, o melhor indivíduo da população anterior é adicionado a esta nova geração, seguindo critérios pré estabelecidos (para o caso da geração um, é inserida a melhor solução da população inicial), e a melhor solução da população corrente é salva.

Todo esse processo se repete até que a condição de parada seja atingida. O melhor indivíduo, o qual contém uma decomposição $I = (E_1, E_2, \ldots, E_n)$ codificada e explicada mais adiante é, então, repassado ao módulo de renderização, que gera um novo desenho com feixes D^R.

Observe-se que os critérios de compatibilidade, função objetivo e restrições

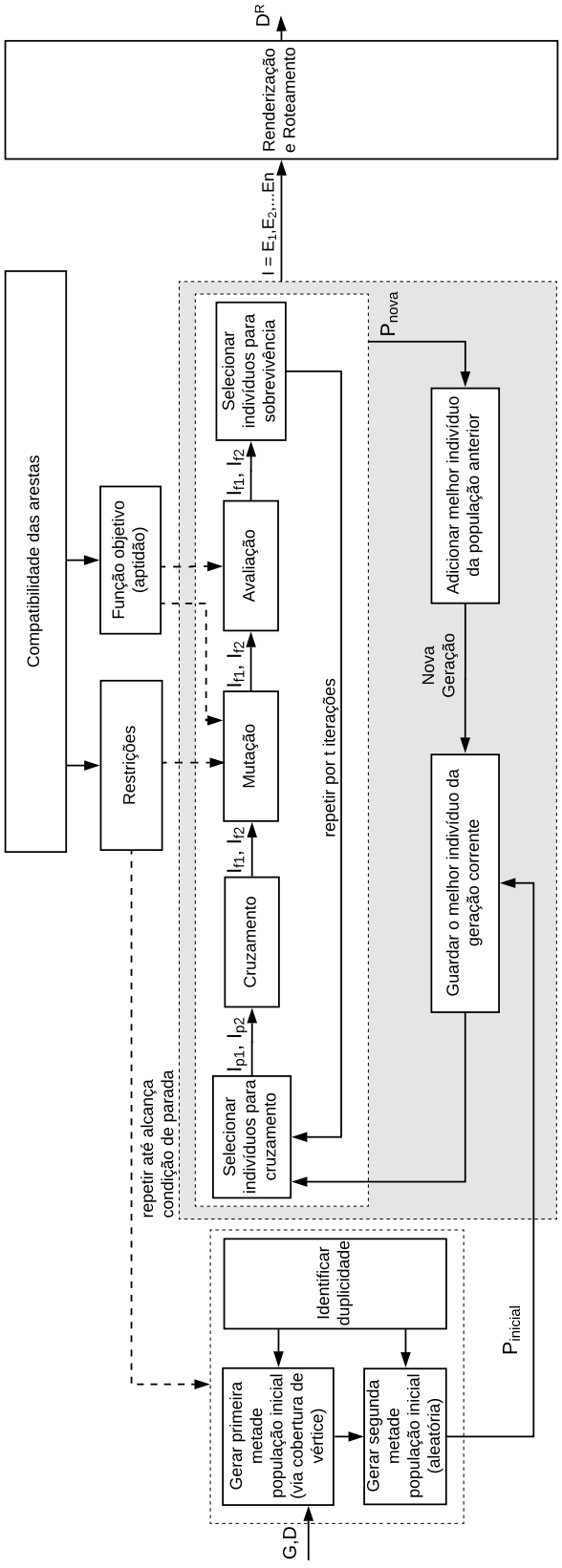


Figura 4.3: *Arcabouço da abordagem evolucionária para união de arestas em feixes.*

são acessados ao longo do processo, bem como no módulo de avaliação que calcula a aptidão dos indivíduos gerados. A estrutura básica do método evolucionário é mostrada no Algoritmo 4.1. As próximas subseções descrevem a representação da solução e os detalhes das etapas desse algoritmo.

Algoritmo 4.1: Algoritmo evolucionário para união explícita de arestas

Entrada: grafo, desenho de grafo.
Saída: desenho de grafo com feixes de aresta.

```
1  P ← {}
2  g ← 0
3  Inicializar metade da população usando a cobertura de vértices (P)
4  Inicializar metade da população de forma aleatória (P)
5  tamMaxPop ← tamanho(P)
6  enquanto NÃO atingiu condição de parada faça
7      I_best ← Melhor indivíduo (P)
8      t ← 0
9      P' ← {}
10     enquanto t < tamMaxPop faça
11         t ← t + 1
12         I_1, I_2 ← Selecionar indivíduos para cruzamento(P)
13         F_1, F_2 ← Recombinar os pais(I_1, I_2)
14         Efetuar a mutação dos filhos(F_1, F_2)
15         Avaliar Aptidão(F_1, F_2)
16         P' ← Selecionar sobreviventes (I_1, I_2, F_1, F_2)
17     fim
18     g ← g + 1
19     Adicionar melhor indivíduo anterior na população(P', I_best)
20     P ← P'
21 fim
22 Retornar a melhor solução de P
```

4.5.2 Codificação do Indivíduo

A proposta de codificação de uma solução (identificado aqui como um indivíduo) não segue o padrão de sequência de números binários ou real como geralmente algumas classes de algoritmos evolucionários o fazem, ao invés disso, na abordagem EEB, o fenótipo e genótipo possuem a mesma representação, não existindo, assim, a necessidade de mapeamento entre eles.

Um indivíduo é uma decomposição S de E, sendo que cada feixe é composto por uma lista de valores representando o identificador das arestas que o formam. Formalmente, a representação de uma solução com feixes de arestas para o problema *ABEB* é:

Definição 4.5 *Dado um grafo $G = (V, E)$, a representação de um indivíduo da população é simplesmente $I = \langle E_1, E_2, \ldots, E_n \rangle$, isto é, um vetor de subconjuntos não-vazios de arestas $E_i \subseteq E$, $i = 1, \ldots, n$. A cardinalidade do indivíduo I é variável, mas não pode exceder um valor máximo, $n \leq |E|$. A satisfação de restrições é parte integral do conceito de operador genético. Desta forma, assume-se que os conjuntos E_i são disjuntos e que $E = \cup_{i=1}^{n} E_i$, assim como especificado na definição de ABEB.*

A Figura 4.4 ilustra a representação de um grafo com quatro feixes E_1, E_2, E_3 e E_4. O feixe E_1, por exemplo, é composto pelas arestas $1, 2$ e 3.

A escolha dessa representação exigiu que os operados genéticos fossem adaptados e implementados de forma específica para esse esquema. Vale destacar que, o roteamento das arestas e outros aspectos visuais da união de arestas não estão incluídos na representação da solução. Esses são elementos tratados automaticamente pela função de renderização e roteamento no estágio de pós-processamento de união das arestas.

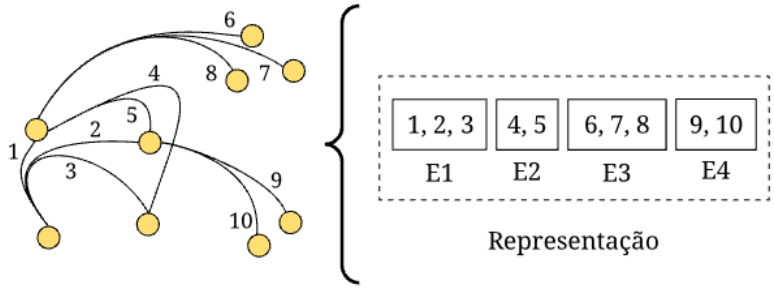


Figura 4.4: *Esquema de representação de um indivíduo no framework EEB.*

4.5.3 Função de Aptidão

A qualidade de cada indivíduo é avaliada de acordo com uma função de aptidão. Essa função é uma versão de **minimização** da função objetivo do problema *ABEB*. Como o objetivo é encontrar a menor quantidade de feixes possível, satisfazendo as restrições do problema, a função de aptidão é o inverso da contagem do número de feixes (n):

$$f(I) = \frac{1}{n}. \tag{4-6}$$

A **função Avaliar** do Algoritmo 4.1 (Linha 15) efetua a contagem dos feixes do indivíduo e calcula o seu valor de aptidão conforme a Equação 4-6.

4.5.4 Função de Teste de Igualdade

Para evitar que uma população de indivíduos tenha duas ou mais soluções iguais, o que reduziria a diversidade do processo evolutivo, uma função de teste de igualdade foi especificada. A função adota uma abordagem heurística para testar se dois indivíduos são iguais e é chamada para comparar par-a-par todos indivíduos a serem colocados na população a cada iteração do ciclo evolutivo. Para testar a igualdade de dois indivíduos, são comparados em sequência os seguintes valores deles:

- número n de feixes;
- valor de aptidão; e
- valor da soma de verificação Ω_I.

O valor de verificação Ω_{E_i} de um feixe E_i é calculado multiplicando-se o identificador de cada aresta presente neste feixe. Já a soma de verificação Ω_I do indivíduo I é a soma dos valores de verificação de todos os feixes que o compõem.

$$\Omega_I = \sum_{i=1}^{n} \Omega_{E_i}, \tag{4-7}$$

$$\Omega_{E_i} = \prod_{e_{ij} \in E_i} (j+1). \tag{4-8}$$

sendo e_{ij} a j-ésima aresta pertencente i-ésimo feixe.

As Figura 4.5 (a) e (b) ilustram o cálculo do teste de igualdades de dois indivíduos. Note-se que a quantidade de feixes, e por consequência, o valor de aptidão deles, é o mesmo. No entanto, com a troca da aresta e_7 entre os feixes E_3 e E_4, o valor de verificação é alterado de 597 na Figura 4.5 (a) para 933 no indivíduo representado na Figura 4.5 (b), o que mostra que as soluções são diferentes. O valor de verificação não varia em relação à posição dos feixes na representação do indivíduo.

4.5.5 Inicialização da População

Para explorar o espaço de busca da forma mais ampla possível, foram elaborados dois métodos para gerar a população inicial, como mencionado anteriormente. O primeiro é um método heurístico baseado em uma solução para o problema de **cobertura mínima de vértice** [106], enquanto o segundo algoritmo cria **aleatoriamente** parte da população inicial. Cada método gera metade da população.

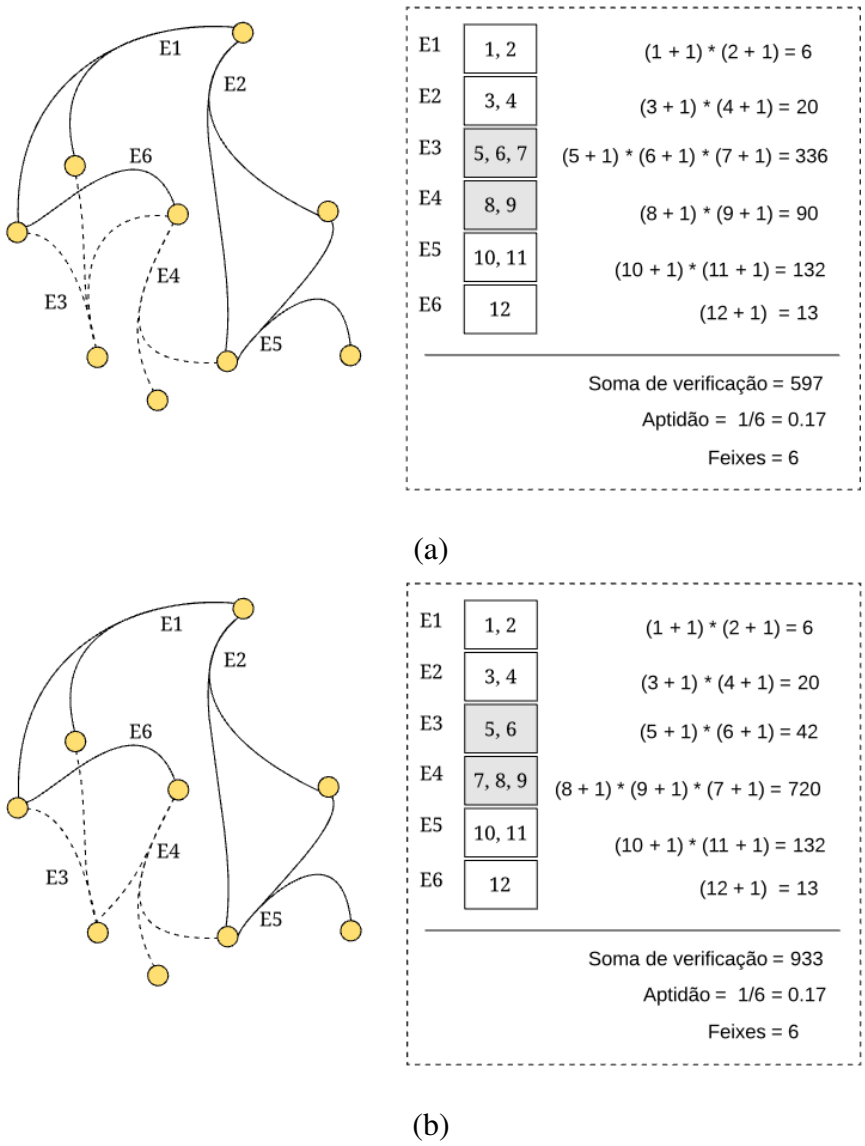


Figura 4.5: *Esquema de teste de igualdade entre soluções.*

Na primeira estratégia, uma cobertura mínima de vértice A de V é gerada por um processo heurístico. Em seguida, cada vértice $v \in A$ é considerado o centro de um subgrafo estrela induzido em G. Conjuntos de feixes são então criados, cada um contendo as arestas compatíveis de um subgrafo estrela escolhido aleatoriamente. Para um vértice $v \in A$, pode ser gerado mais de um feixe, pois a relação de compatibilidade entre suas arestas adjacentes pode ser diferente para subgrupos distintos de arestas. No caso do problema *ABEB*, a compatibilidade é o limite angular máximo pré-estabelecido pelo usuário.

A escolha de qual vértice v e quais de suas arestas adjacentes serão selecionadas primeiro é aleatória. O algoritmo, por vezes, decide se usará ou não o vértice ou aresta selecionados. Isso faz com que, ao terminar o processo, algumas arestas, apesar de visitadas, não tenham sido usadas na composição de nenhum feixe. Para garantir que o

máximo de arestas seja alocada a algum feixe antes de o ciclo terminar, a parte específica de criação de feixes se repete um número limite de vezes (linha 6 do Algoritmo 4.2). Nos experimentos, foi usado o limite do tamanho das arestas. Assegura-se por meio destes procedimentos o aspecto estocástico do algoritmo. A estrutura básica da criação de indivíduos usando a cobertura de vértices é descrita no Algoritmo 4.2.

Em experimentos preliminares a abordagem via cobertura foi considerada eficaz, gerando, boa parte das vezes, indivíduos próximos da melhor solução (relacionada ao número de feixes) para grandes populações. No entanto, esses experimentos preliminares também mostraram que uma população criada usando apenas tal método poderia levar à convergência prematura.

Foi com o intuito de resolver o problema da convergência prematura, que a estratégia que inicializa a segunda metade da população aleatoriamente foi definida. Ela cria indivíduos ao escolher de forma estocástica arestas adjacentes para gerar feixes, enquanto ainda assegura a viabilidade e a satisfação do limite angular. Esse método pode produzir indivíduos não tão interessantes, mas permite aumentar a diversidade populacional. A estrutura básica desse método é a mesma do algoritmo via cobertura. A diferença é que os vértices são escolhidos aleatoriamente entre todos os vértices do grafo.

Ao final do ciclo de criação de um indivíduo, é necessário verificar se ele não é inviável (que viola a restrição do conjunto disjunto ou não possui todas as arestas alocadas a um feixe), e, neste caso, conserta-se a solução eliminando-se as duplicações e criando-se feixes individuais para as arestas não usadas.

Depois que um indivíduo é gerado, ambas as estratégias permitem que os feixes sejam embaralhados em suas posições. Essa etapa é necessária para garantir que o final da lista de feixes não contenha uma quantidade grande de feixes com apenas uma aresta. Isso melhora a eficiência do operador de cruzamento, criando indivíduos aleatórios mais interessantes e também impedindo a convergência prematura.

Por fim, se qualquer indivíduo for encontrado duplicado (ver Seção 4.5.4), uma função de remoção é usada para substituí-lo ou descartá-lo. Foi utilizada a abordagem de Saroj e Devraj [103] em que qualquer indivíduo duplicado é substituído probabilisticamente por uma versão mutada do melhor indivíduo presente na população. Depois de aplicar um teste, se o novo indivíduo ainda estiver duplicado, ele é descartado.

4.5.6 Seleção

O operador de seleção tem duas funções. A primeira é selecionar indivíduos para cruzamento, e a segunda é decidir sobre a sobrevivência de indivíduos nas novas gerações.

Algoritmo 4.2: Algoritmo para gerar a população inicial via cobertura

Entrada: grafo, desenho de grafo, quantidade de indivíduos.

Saída: população de indivíduos.

```
1  A ← CalcularCobertura(G)
2  tamAtualPopulacao ← 0
3  limitePopulacao ← 0
4  enquanto (tamAtualPopulacao < quantidadeIndividuos) E
     (limitePopulacao < quantidadeIndividuos * 10) faça
5      limitePopulacao ← limitePopulacao + 1
6      I ←<> //vetor vazio
7      enquanto limite de repetição não é atingido faça
8          enquanto não visitou todos os vértices da cobertura faça
9              v ← recuperarVérticeCobertura(A)
10             enquanto não visitou todas as arestas adjacentes a v faça
11                 feixe ← montarFeixeArestasCompativeis(v)
12                 AdicionarFeixe(I, feixe)
13             fim
14         fim
15     fim
16     Montar feixes com arestas não usadas(I)
17     Checar se solução é viável(I)
18     Avaliar Aptidão(I)
19     se I não está duplicado em P então
20         AdicionarNaPopulacao(P, I)
21         tamAtualPopulacao ← tamAtualPopulacao + 1
22     senão
23         Mutar(I_Best)
24         Avaliar Aptidão(I_Best)
25         se I_Best não esta duplicado em P então
26             AdicionarNaPopulacao(P, I_Best)
27             tamAtualPopulacao ← tamAtualPopulacao + 1
28         fim
29     fim
30 fim
31 Retornar a melhor solução de P
```

A estratégia utilizada para **seleção de indivíduos para recombinação** foi a seleção por torneio com substituição. São sorteados dois indivíduos, cada um deles em uma competição individual com cinco outros competidores da população, o indivíduo com melhor aptidão dos cinco presentes em cada torneio é o escolhido. Nos dois torneios executados, participam todos os indivíduos da população, mesmo que já tenham sido escolhidos antes. O Algoritmo 4.3 mostra os passos dessa seleção.

Algoritmo 4.3: Algoritmo de seleção de indivíduos para cruzamento

Entrada: população de indivíduos P.
Saída: dois indivíduos pais.

```
1  numIndividuos ← 2
2  numCompetidores ← 5
3  enquanto i < numIndividuos faça
4      listaCompetidores ← ∅
5      enquanto j < numCompetidores faça
6          listaCompetidores ← Selecionar aleatoriamente um competidor(P)
7          j ← j + 1
8      fim
9      I_i ← SelecionarMelhor(listaCompetidores)
10     i ← i + 1
11 fim
12 Retornar os indivíduos I_1, I_2
```

O segundo tipo é a **seleção de indivíduos para sobrevivência**. Após a execução dos operadores de cruzamento e mutação, o método *steady state* ou incremental é aplicado. Segundo Jong [53], nessa abordagem de gerenciamento populacional, os indivíduos gerados (filhos) competem pela sobrevivência contra membros da população atual. Isso foi feito com uma ligeira variação, pois ao invés de selecionar aleatoriamente membros da população corrente para competir com os indivíduos recém criados, os dois recém-nascidos na população são inseridos na população corrente e, na sequência, são removidos os dois piores em termos de qualidade. Se os filhos recém-criados tiverem uma qualidade inferior a todos os outros da população, serão os a serem retirados.

Depois de repetir este processo t vezes, onde t é o número máximo de indivíduos da população, o melhor indivíduo da geração anterior é inserido na nova população, se ele tiver maior aptidão do que o melhor indivíduo da população corrente. Se isso acontecer, esse melhor indivíduo substitui aleatoriamente uma solução na população atual.

4.5.7 Operador de Cruzamento

O cruzamento é realizado trocando-se parte dos feixes entre os dois pais, criando-se novos conjuntos aleatórios, representando dois novos indivíduos. Foram utilizados os métodos clássicos de cruzamentos de um e dois pontos. A escolha de qual método usar é feita aleatoriamente a cada execução. No cruzamento de um ponto, o ponto selecionado é o centro do indivíduo pai de menor tamanho. O lado direito dos dois pais é, então, trocado entre eles (veja a Figura 4.6). No cruzamento de dois pontos, os pontos também são definidos pelo tamanho do menor indivíduo, dividindo-o em três partes de tamanho iguais. A parte do meio de cada um é, então, trocada entre eles.

Assim como na inicialização, durante o processo de cruzamento, indivíduos inviáveis podem ser gerados e estes precisam ser reparados antes de serem inseridos na população. Dessa forma, se as arestas são identificadas como duplicadas, o algoritmo de reparo busca pela última ocorrência dessa aresta que é, então, removida. Já no caso de arestas que não foram alocadas a nenhum feixe, estas são inseridas em novos feixes individuais e adicionadas ao final da representação da solução. O procedimento de reparo está ilustrado na Figura 4.6. Por exemplo, após o cruzamento, a aresta 2 ficou duplicada nos feixes E_1 e E_4 no Filho 1. No procedimento de reparo, essa aresta foi removida do feixe E_4. No caso do Filho 2, a mesma aresta não foi alocada a nenhum feixe. Assim, durante o processo de reparo, foi criado um novo feixe para ela.

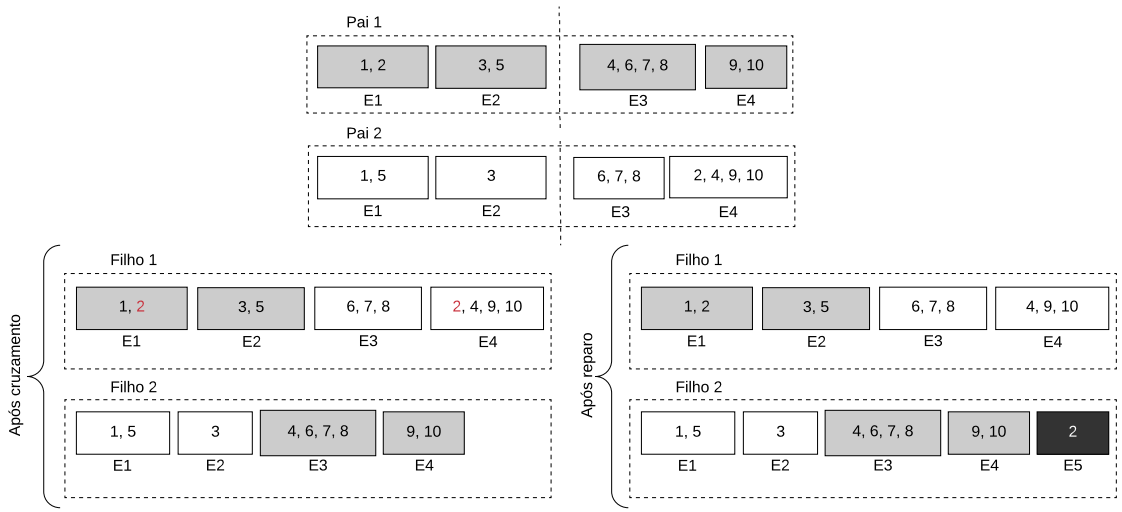


Figura 4.6: *Exemplo de cruzamento de um ponto.*

4.5.8 Operadores de Mutação

Foram implementados cinco operadores de mutação. O método a ser usado é escolhido aleatoriamente a cada execução. A tentativa de aplicação da mutação é feita com todos os indivíduos filhos a uma taxa t_m.

O primeiro operador é o de **união** (do inglês *join*). Este operador seleciona os dois primeiros feixes (considerando a sequência de feixes do indivíduo) que possuam *somente* uma aresta cada, e gera um novo feixe com duas arestas, caso elas sejam adjacentes e compatíveis (ver Figura 4.7). Caso não o sejam, a operação é abordada.

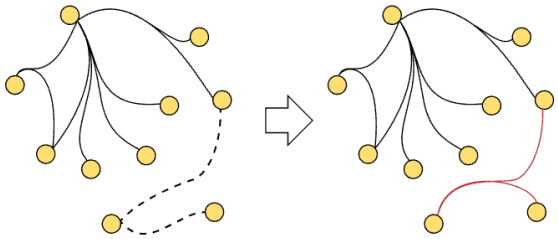


Figura 4.7: *Operador de mutação de união.*

O operador de **fusão** (do inglês *merge*) é similar ao de união. A diferença é que a fusão tenta agrupar feixes de *qualquer* tamanho. O algoritmo seleciona aleatoriamente um dos feixes do indivíduo, e, depois, busca um segundo que possua o mesmo vértice central, pegando a primeira ocorrência. Todas as arestas compatíveis são transferidas do segundo feixe para o primeiro feixe. Caso o segundo feixe tenha ficado vazio, ele é removido do indivíduo. Caso contrário, ele é mantido com as arestas restantes (ver Figura 4.8).

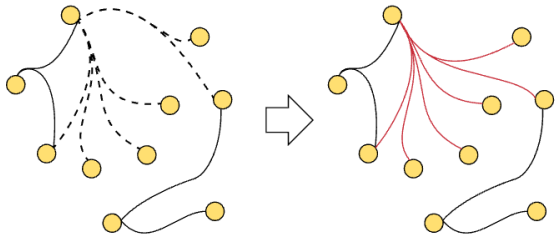


Figura 4.8: *Operador de mutação de fusão.*

O operador de **divisão** (do inglês *split*) seleciona aleatoriamente um feixe de tamanho maior do que um e o divide em dois novos feixes. O ponto de divisão é escolhido aleatoriamente, conforme a quantidade de arestas. As arestas cujo identificador varia de 0 até o ponto limite vão para o feixe 1, e o restante vai para o feixe 2 (ver Figura 4.9).

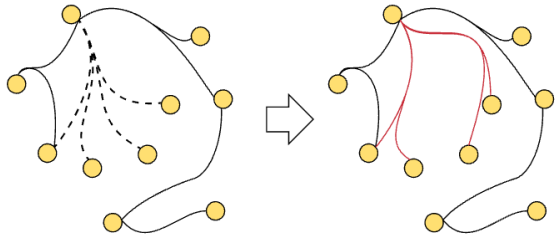


Figura 4.9: *Operador de mutação de divisão.*

O quarto operador é o de **movimentação** (do inglês *move*). Ele seleciona aleatoriamente um feixe de tamanho maior que um. Uma vez selecionado, uma aresta aleatória

é retirada dele. O algoritmo busca, então, um outro feixe que tenha como vértice central o outro nó da aresta e a adiciona a ele, se for compatível com as arestas presentes neste segundo feixe. Caso não tenha encontrado nenhum novo feixe hospedeiro, ou as arestas não sejam compatíveis, o algoritmo cria um novo feixe para a aresta retirada (ver Figura 4.10).

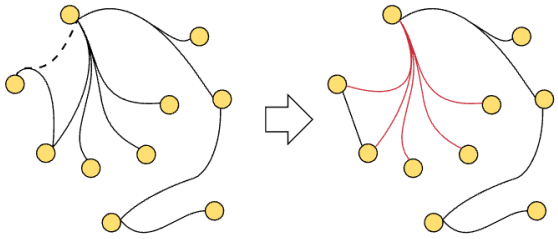


Figura 4.10: *Operador de mutação de movimentação.*

O último operador de mutação é o operador de **remoção** (do inglês *remove*). Esse operador seleciona aleatoriamente um feixe de tamanho maior que um, e uma aresta deste feixe. A aresta selecionada é removida e um novo feixe é criado apenas para ela (ver Figura 4.11).

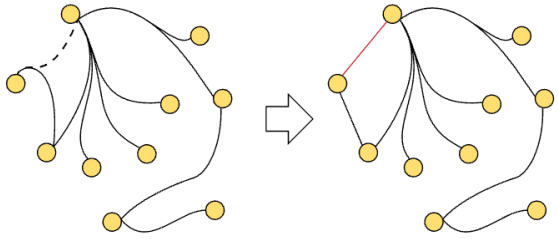


Figura 4.11: *Operador de mutação de remoção.*

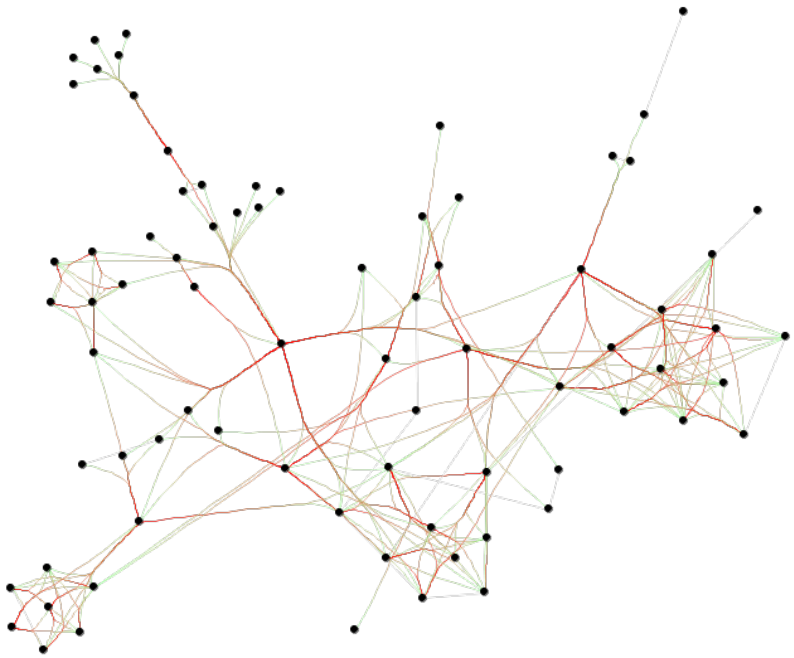
4.5.9 Modelos de Visualização

A maioria dos métodos existentes para união de arestas desenham arestas individualmente, roteando-as próximas umas das outras. Em contraste, a abordagem *EEB* aplicada ao problem *ABEB* produz conjuntos de arestas que representam os feixes individualmente. Esses grupos de arestas formam subgrafos estrela que, após a conclusão do algoritmo evolutivo, são submetidos a um módulo de roteamento e renderização para produzir um desenho com arestas em feixes. A seguir são abordados os aspectos de renderização (desenho) das arestas.

Para permitir uma interpretação visual adequada dos resultados produzidos pelo *framework EEB*, principalmente a compreensão dos padrões de conexão entre os vértices, no presente trabalho são propostas duas visualizações: a visualização primária e a visualização de arestas parciais.

Visualização Primária

A **visualização primária** mostra as arestas como curvas Bézier cúbicas coloridas, sendo o esquema de cor padrão usado o degradê do vermelho para verde. A cor vermelha representa a fonte (o vértice central do feixe) e o destino é colorido em verde. Em feixes com apenas uma aresta, ela é desenhada como uma linha reta colorida em cinza claro. A Figura 4.12 mostra um grafo com feixes representados usando este esquema. Ainda, os vértices podem ser coloridos em preto e vermelho. Vermelho significa que o vértice pertence ao conjunto de cobertura de vértices do grafo. Esse esquema de cor é configurável e pode ser modificado pelo usuário.



(b)

Figura 4.12: *Exemplo de visualização primária para resultados do modelo EEB.*

Visualização Parcial de Arestas

Para soluções do problema *ABEB*, a visualização primária é efetiva somente para grafos de tamanho pequeno e médio. Apesar de ser possível identificar padrões de relação em nível de vértices, para grafos com alto grau de poluição visual, os cruzamentos de arestas impedem a sua legibilidade mesmo após a união das arestas. Para minimizar este problema, reduzindo os cruzamentos de forma visual e destacando os padrões de relação presentes na estrutura, foi utilizada a estratégia descrita em [15, 92] que usa a ideia de **desenho parcial de arestas**. Neste modelo, o desenho de uma curva é dividido em três partes, e a parte do meio é desenhada usando-se uma transparência parcial.

A Figura 4.13 mostra o detalhe do resultado do desenho parcial de arestas. Nota-se que, entre a extremidade de origem das arestas em vermelho e a extremidade de destino em verde, as curvas foram desenhadas com um nível de transparência maior.

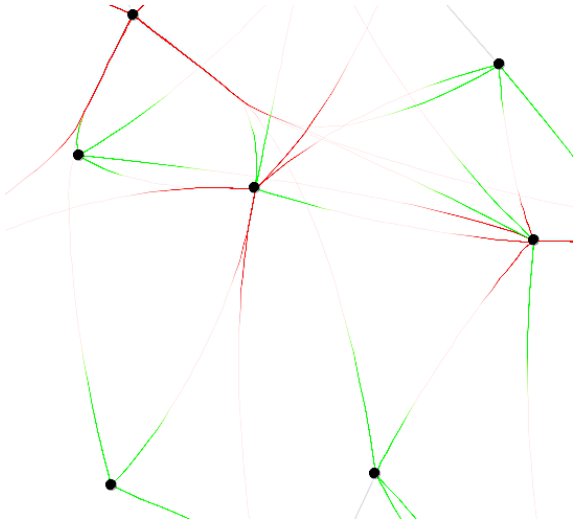


Figura 4.13: *Destaque de um desenho parcial de arestas.*

Este modelo de visualização mais "suave" pode ser usado quando não é necessário rastrear todo o trajeto da aresta, mas, sim, entender os seus pontos de conexão. Neste caso, o desenho parcial de arestas oculta as informações que não são pertinentes para o problema, simplificando o desenho gerado. Essa visualização é semelhante ao *layout* usado pela abordagem SideKnot [92] (descrita no Apêndice D).

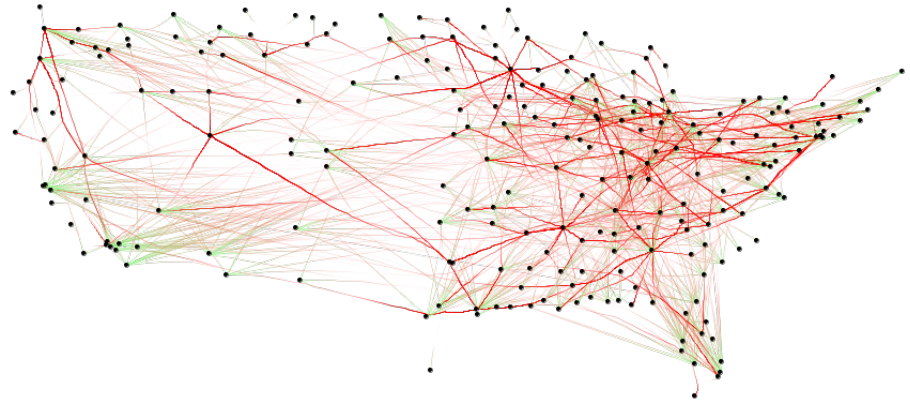
Foram definidas ainda três formas de visualização de desenho parcial de arestas:

- **Modelo padrão**: que concentra a transparência no centro da aresta,
- **Modelo de destaque na origem** (vértice central do feixe): que deixa transparente o meio e na extremidade de destino,
- **Modelo de destaque no destino**: que eleva o nível de transparência da região central da aresta e da extremidade de origem.

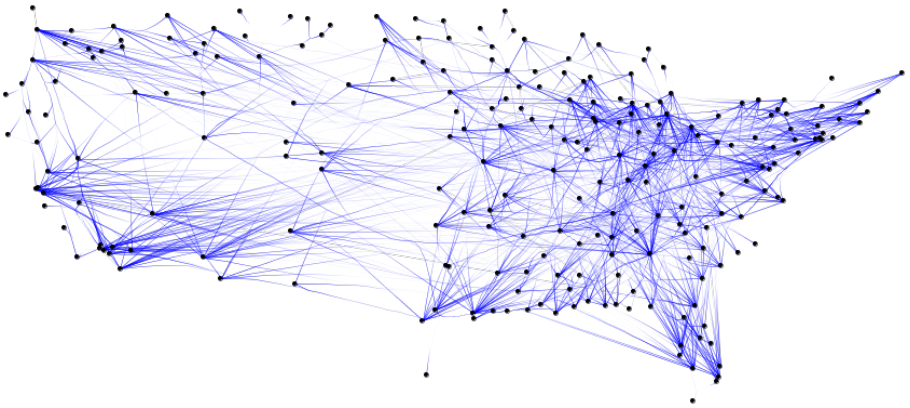
Estas três visões permitem entender as conexões em nível de vértices de grafos de formas diferentes. Além disso, as partes de uma aresta incidente em seus extremos ainda podem ser coloridas usando-se cores diferentes. Todos esses aspectos, como cores, nível de transparência e opções de destaque, também podem ser escolhidos pelo usuário.

A Figura 4.14 ilustra as três maneiras de configurar a visualização parcial das arestas em um grafo de grande dimensão. Na Figura 4.14 (a), é exemplificado o modelo padrão, no qual a transparência é aplicada no centro das arestas. São usadas duas cores conforme o esquema vermelho/verde para dar mais destaque no contrates entre vértices de destino e de origem das arestas. A Figura 4.14 (b) mostra o modelo de destaque no destino com transparência central e na origem. Foi usada uma única cor para as duas

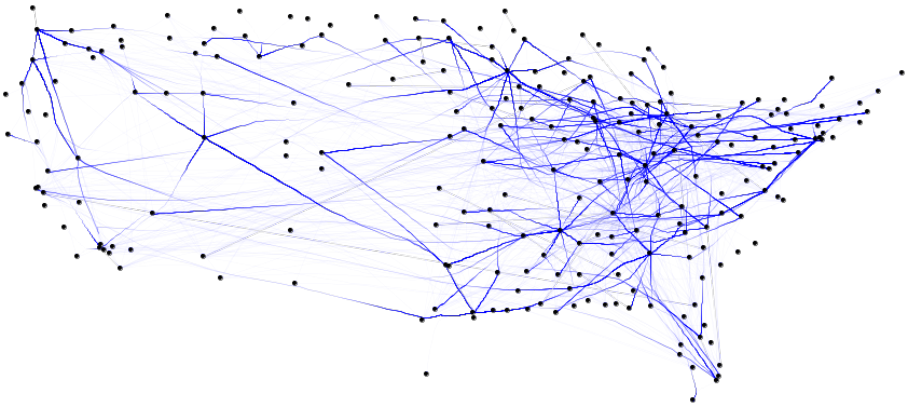
extremidades. Essa visualização destaca os vértices que recebem conexões (arestas), e não são centro dos feixes aos quais essas arestas pertencem. Tais vértices seriam os "sumidouros" dos feixes. Por fim, na Figura 4.14 (c) também foi usada uma única cor para as duas extremidades, sendo que a transparência agora foi definida no centro e na extremidade de destino. Percebe-se que o destaque é dado aos vértices que são origem de conexões (vértices centrais dos feixes).



(a)



(b)



(c)

Figura 4.14: *Visualização parcial de arestas: (a) modelo padrão; (b) modelo de destaque no destino; (c) modelo de destaque na origem.*

É importante destacar que o esquema de duas cores é mais apropriado para grafos direcionados. No entanto, este não é o caso do tipo de grafo tratado nesta tese. Assim, a cor não infere aspectos de direcionamento não existente no grafo, sendo apenas um artifício que pode ser usado para a identificação visual das conexões no vértice central do feixe.

Para o desenho dos feixes, é preciso construir um roteamento no plano para cada conjunto $E_i, i = 1, \ldots, n$, e então, usando-se algum dos modelos de visualização apresentados acima, desenhar tanto a parte da linha em que as arestas estão unidas quanto a parte em que separam do seu feixe e seguem sozinhas para o outro vértice extremo.

Como o roteamento não é o foco da otimização no presente trabalho, sugere-se o uso de uma implementação especializada da abordagem *force-based edge bundling* de Holten e Wijk [43]. O algoritmo foi modificado para receber os feixes como entrada e os desenhar individualmente. O algoritmo baseado em força garante que a simetria axial individual de cada feixe pareça plausível. No entanto, a simetria geral do grafo não é tratada, uma vez que nem o algoritmo *EEB* e nem o módulo de roteamento e renderização procuram maximizar a simetria do grafo.

4.6 Experimentos e Discussões

Com o objetivo de verificar a efetividade das abordagens *PLI* e *EEB* na resolução do problema *ABEB*, esta seção discute os experimentos realizados com tais métodos.

A abordagem *PLI* foi implementada usando o *solver* Gurobi [90], interfaceado com o GNU Octave [24], e o código rodou em um servidor DELL M630 com 128 GB de RAM e 2 processadores com 10 *cores* (40 visíveis) de 3-3.6Ghz cada. Já a abordagem *EEB* foi implementada usando a linguagem Java 8. Um sistema interativo foi construído para explorar diversos parâmetros do método evolutivo, bem como os modelos de visualização. A Figura 4.15 mostra a *interface* da aplicação que implementa o modelo. No painel A é possível configurar todos os parâmetros do algoritmo evolucionário e, desta forma, ajustá-lo ao problema a ser resolvido; já os painéis B, C, D configuram os parâmetros relativos à visualização gerada pelo módulo de renderização externo. Todos os testes do método *EEB* foram executados em um MacBook Pro com um processador Intel Core i7 de 2,9 GHz e 8 GB de RAM 1600MHz-DDR3.

Para a abordagem *EEB*, foram executados testes controlados independentes com diversos grafos de tamanho crescente, e com valores diferentes para o ângulo α. Os experimentos da abordagem foram conduzidos para um grafo sintético e nove grafos do mundo real, a saber:

- G1 - Sintético (20 vértices, 28 arestas) [31];
- G2 - *Zachary Karate Club* (34 vértices, 78 arestas) [122];

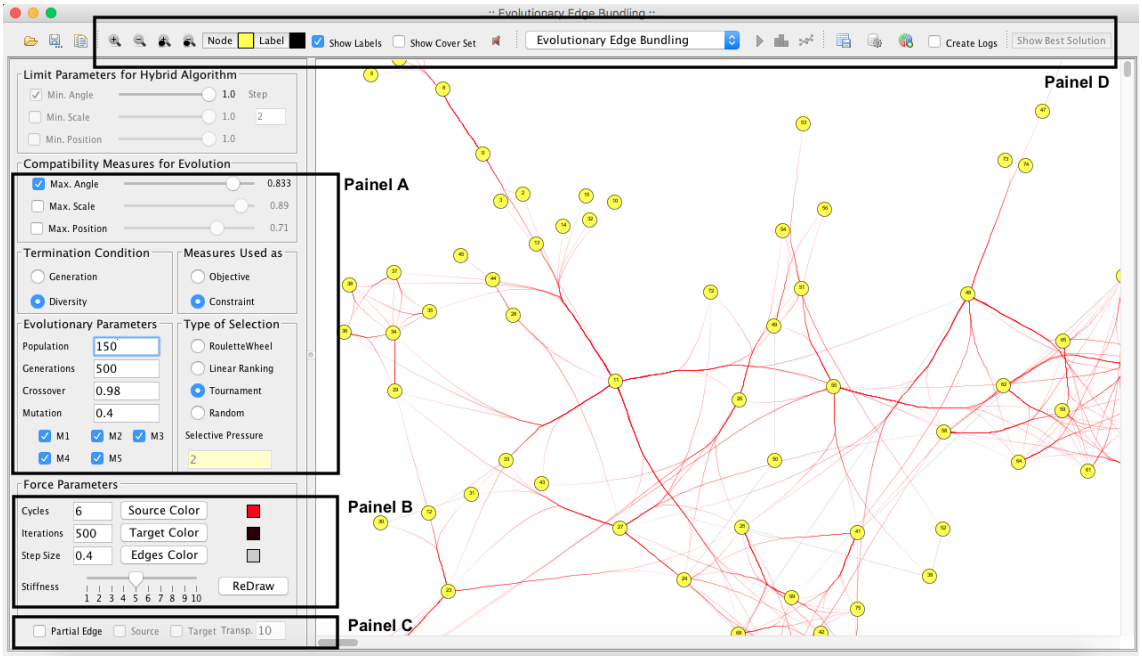


Figura 4.15: *Interface da aplicação que implementa o modelo EEB.*

- G3 - *Planar Graph GD2015* (66 vértices, 101 arestas) [51];
- G4 - *Dolphin Social Network* (62 vértices, 160 arestas) [37];
- G5 - *MovieLens* (160 vértices, 161 arestas) [81];
- G6 - *Les Miserables* (77 vértices, 254 arestas) [57];
- G7 - *Books About US Politics* (105 vértices, 401 arestas) [83];
- G8 - *Word Adjacencies* (112 vértices, 425 arestas) [83];
- G9 - *Flare Software Class* (220 vértices, 709 arestas) [42]; e
- G10 - *USAirline* (235 vértices, 1297 arestas) de fonte desconhecida.

O desenho inicial de cada grafo foi pre-definido em um arquivo de entrada, sendo que o posicionamento dos vértices não sofreu alteração durante a execução do algoritmo. Para garantir maior confiabilidade, os experimentos controlados da abordagem *EEB* consistiram em 100 testes independentes do algoritmo evolutivo, para cada grafo e ângulo. Para o problema *ABEB*, o parâmetro do ângulo foi progressivamente fixado como $\alpha = 30°, 45°, 70°$. Como condição de parada, o ciclo evolutivo foi repetido até que não houvesse mais melhora na população por um número de 500 iterações consecutivas ou o número máximo de gerações (fixado em 16.500) fosse alcançado. O melhor indivíduo, em termos de número de feixes, foi coletado para análise, bem como o tempo total de execução do método até atingir a condição de parada.

4.6.1 Calibração dos Parâmetros

A maioria dos parâmetros usados para configurar os experimentos controlados da abordagem *EEB* foram definidos de forma empírica depois de testes preliminares: o **tamanho da população** ($u = 150$), a **taxa de cruzamento** ($p_c = 0,98$) e **taxa de mutação** ($p_m = 0,4$). No entanto, para a taxa de mutação foi feita uma validação do seu valor por meio de testes com cinco grafos (G1, G2, G3, G4 e G5). Para cada grafo foram feitos 100 testes preliminares para cada uma das taxas de mutação, $p_m = 0,1$, $p_m = 0,2$, $p_m = 0,3$, $p_m = 0,4$, e $p_m = 0,5$. A Tabela 4.1 sumariza os valores médios nos testes com as taxas de mutação diversas e registra o número de feixes da melhor solução.

Analisando o tempo de execução e os resultados obtidos para o número de feixes, percebe-se que a quantidade de feixes utilizados tende a diminuir (melhor qualidade) e o tempo de execução do *EEB* tende a aumentar com a elevação da taxa de mutação p_m. Interessantemente, para $p_m = 0,4$, o tempo de execução médio foi, em geral, menor do que o das duas taxas de mutação anteriores. Assim, $p_m = 0,4$ foi escolhido como um bom valor de compromisso entre a qualidade das solução e o tempo demandado para gerá-las.

4.6.2 Resultados

Inicialmente foram feitos testes de performance do modelo *PLI*. Foram usados três grafos, G1, G2 e G3. Como pode ser visto nas Tabelas 4.2, 4.3 e 4.4 que sumarizam os resultados, a execução do modelo *PLI* gerou soluções ótimas para instâncias menores do problema, mas levou uma quantidade excessiva de tempo para resolver instâncias maiores do problema, falhando na resolução que instâncias com ângulos muito altos. Com destaque para o grafo G2, que mesmo para um ângulo pequeno de $\alpha = 30°$, o tempo de resposta excedeu todos os outros testes. Isso salienta a complexidade do problema *ABEB*.

Nota-se que, com o aumento do tamanho do grafo, os problemas crescem significativamente de tamanho e torna-se impossível de resolvê-lo de modo eficiente por meio de um "solver" *PLI*.

A Tabela 4.5 apresenta os resultados numéricos do *EEB* para os vários grafos testados. As três primeiras colunas consistem em informações gerais. A quarta coluna mostra o número de feixes da melhor solução entre todas as soluções geradas nos 100 testes. A quantidade de testes em que essa solução apareceu é apresentada entre parênteses. A quinta, a sexta e a sétima colunas apresentam os valores médios do número de feixes, o desvio padrão e o erro padrão, respectivamente, das melhores soluções. A oitava, a nona e a décima colunas destacam, nessa ordem, a média de aptidão, o desvio padrão e o erro padrão dos valores de aptidão. Por fim, a última coluna mostra a média do tempo de execução total.

Tabela 4.1: *Testes preliminares com a taxa de mutação para o problema ABEB.*

Grafo	Ângulo	Núm. feixes (melhor)	Média de feixes	Média da aptidão	Desvio padrão da aptidão	Média do tempo (sec.)
Taxa de mutação $p_m = 0,1$						
G1	30	17	17,000	0,059	0,000	1
	45	16	16,000	0,063	0,000	1
	70	13	13,050	0,077	0,001	1
G2	30	47	50,950	0,020	0,001	6
	45	38	40,570	0,025	0,001	7
	70	30	33,920	0,030	0,001	5
G3	30	74	77,470	0,013	0,000	10
	45	72	73,990	0,014	0,000	11
	70	60	63,670	0,016	0,000	9
G4	30	108	114,630	0,009	0,000	16
	45	93	99,760	0,010	0,000	19
	70	77	82,170	0,012	0,000	19
G5	30	67	69,840	0,014	0,000	10
	45	51	55,270	0,018	0,000	10
	70	39	42,260	0,024	0,001	10
Taxa de mutação $p_m = 0,2$						
G1	30	17	17,000	0,059	0,000	1
	45	16	16,000	0,063	0,000	1
	70	13	13,050	0,077	0,001	1
G2	30	47	50,220	0,020	0,001	7
	45	37	40,070	0,025	0,001	7
	70	31	33,400	0,030	0,001	5
G3	30	75	77,110	0,013	0,000	10
	45	71	73,700	0,014	0,000	9
	70	60	63,220	0,016	0,000	9
G4	30	108	113,500	0,009	0,000	19
	45	94	98,190	0,010	0,000	19
	70	75	79,990	0,013	0,000	18
G5	30	67	69,660	0,014	0,000	12
	45	49	54,720	0,018	0,001	11
	70	39	41,690	0,024	0,001	14
Taxa de mutação $p_m = 0,3$						
G1	30	17	17,000	0,059	0,000	2
	45	16	16,000	0,063	0,000	2
	70	13	13,110	0,076	0,002	2
G2	30	46	50,130	0,020	0,001	8
	45	37	39,780	0,025	0,001	6
	70	31	33,470	0,030	0,001	7
G3	30	74	76,820	0,013	0,000	12
	45	71	73,320	0,014	0,000	11
	70	60	62,850	0,016	0,000	10
G4	30	106	111,950	0,009	0,000	22
	45	91	97,100	0,010	0,000	23
	70	74	79,670	0,013	0,000	18
G5	30	66	69,490	0,014	0,000	11
	45	50	54,210	0,018	0,001	13
	70	38	41,210	0,024	0,001	11
Taxa de mutação $p_m = 0,4$						
G1	30	17	17,000	0,059	0,000	1
	45	16	16,000	0,063	0,000	1
	70	13	13,070	0,077	0,001	1
G2	30	47	50,020	0,020	0,001	6
	45	37	39,580	0,025	0,001	5
	70	30	33,200	0,030	0,001	4
G3	30	75	76,760	0,013	0,000	8
	45	71	73,250	0,014	0,000	8
	70	60	62,870	0,016	0,000	8
G4	30	107	111,290	0,009	0,000	18
	45	92	96,600	0,010	0,000	17
	70	73	79,250	0,013	0,000	15
G5	30	65	69,150	0,014	0,000	10
	45	51	54,110	0,018	0,000	10
	70	37	40,980	0,024	0,001	10
Taxa de mutação $p_m = 0,5$						
G1	30	17	17,000	0,059	0,000	2
	45	16	16,000	0,063	0,000	2
	70	13	13,090	0,076	0,002	2
G2	30	45	49,350	0,020	0,001	7
	45	37	39,450	0,025	0,001	6
	70	30	33,270	0,030	0,001	5
G3	30	74	76,660	0,013	0,000	9
	45	70	73,010	0,014	0,000	10
	70	58	62,570	0,016	0,000	10
G4	30	105	110,910	0,009	0,000	22
	45	91	96,360	0,010	0,000	21
	70	74	78,880	0,013	0,000	17
G5	30	66	69,110	0,014	0,000	12
	45	51	53,870	0,019	0,000	13
	70	36	40,540	0,025	0,001	12

Analisando-se a Tabela 4.5, percebe-se que a média do número de feixes das melhores soluções (média de 100 execuções) está próxima da quantidade de feixes da melhor solução encontrada entre todos os testes independentes. Conforme esperado, a

Tabela 4.2: *Solução ótima para o problema ABEB para o grafo Sintético (G1), com 28 arestas, usando o modelo PLI.*

Ângulo máximo	Variáveis	Restrições	Tempo (s)	Feixes
30	784	9996	0,305924	17
45	784	9856	0,419298	16
70	784	9548	0,354541	13
90	784	9268	0,173727	11
110	784	9044	0,168548	10

Tabela 4.3: *Solução ótima para o problema ABEB para o grafo Zachary Club (G2), com 78 arestas, usando o modelo PLI.*

Ângulo máximo	Variáveis	Restrições	Tempo (s)	Feixes
30	6084	228852	12070,3	45
45	6084	224874	1103,77	36
70	6084	218790	135,931	29
90	6084	213798	445,188	25
110	6084	211692	801,322	25

Tabela 4.4: *Solução ótima para o problema ABEB para o grafo PlanarGD2015 (G3), com 101 arestas, usando o modelo PLI.*

Ângulo máximo	Variáveis	Restrições	Tempo (s)	Feixes
30	10201	504899	125,654	74
45	10201	502475	656,139	69
70	10201	497021	1210,29	58
90	10201	492880	1275,27	50
110	10201	487123	-	-

diferença aumenta à medida que o número de arestas cresce, uma vez que o espaço de busca passa a ser maior.

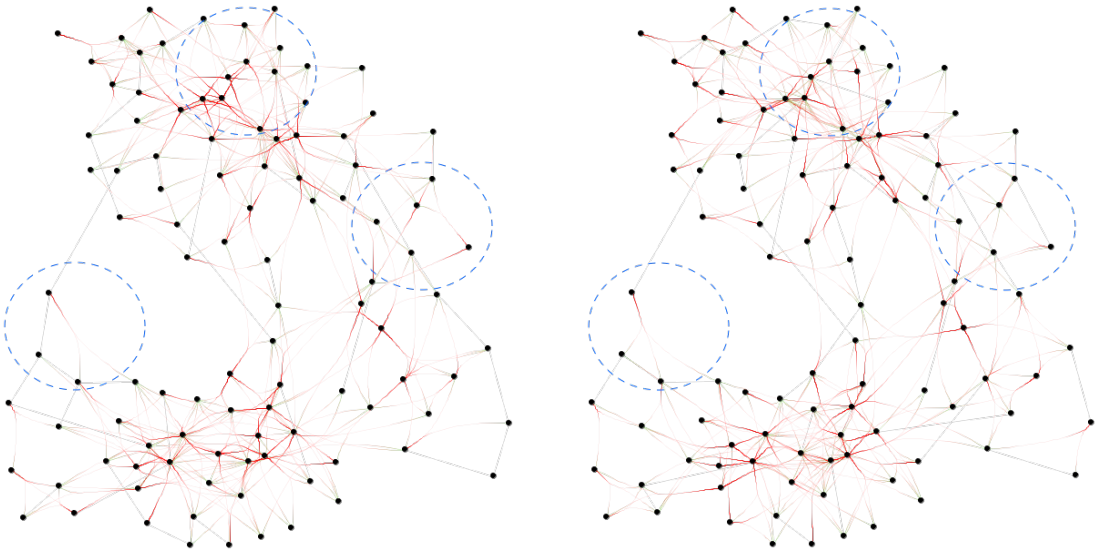
No algoritmo, o valor de aptidão $1/n$ é o dado que, efetivamente, foi utilizado no processo evolutivo para medir a qualidade das soluções. Analisando-se os valores obtidos, verifica-se que o desvio-padrão da aptidão foi baixo, o que evidencia que a solução gerada em cada uma das 100 execuções está perto da média de qualidade.

Comparado o resultado ótimo do *PLI* em temos de número de feixes e tempo de execução, nota-se a abordagem do EEB foi bem sucedida encontrando soluções próximas da solução ótima (para os grafo G1, G2 e G3). Além disso, o tempo de execução do EEB é significativamente menor para uma execução independente do que o do método *PLI* para os grafos maiores, mesmo tendo sido usado um computador menos poderoso. Lembrando que, para definir a melhor solução, usando-se o método EEB, foram feitas 100 execuções. Assim, foram gastos em torno de 900 segundos para a definição da solução do grafo G3. Isso implica que o método evolucionário só começa a se tornar efetivo a partir de instâncias maiores quando o tempo do método exato seria inviável.

Tabela 4.5: *Resultados para o problema ABEB usando o modelo EEB. Média de 100 execuções independentes,* $u = 150$*,* $p_c = 0,98$*,* $p_m = 0,4$*.*

Grafo	Arestas	α(°)	Feixes da melhor solução	Média de feixes	Desvio padrão de feixes	Erro padrão de feixes	Média da aptidão	Desvio padrão da aptidão	Erro padrão da aptidão	Média tempo (s)
G1	28	30	17 (100)	17,00	0,0000	0,0000	0,0588	0,00000	0,000000	2
		45	16 (100)	16,00	0,0000	0,0000	0,0625	0,00000	0,000000	1
		70	13 (89)	13,11	0,3145	0,0314	0,0763	0,00173	0,000173	1
G2	78	30	45 (1)	49,74	1,7844	0,1784	0,0201	0,00073	0,000073	6
		45	37 (3)	39,75	1,2978	0,1298	0,0252	0,00081	0,000081	6
		70	31 (6)	33,30	1,1237	0,1124	0,0301	0,00101	0,000101	5
G3	101	30	75 (4)	76,84	0,8495	0,0849	0,0130	0,00015	0,000015	9
		45	69 (1)	73,01	1,1677	0,1168	0,0137	0,00023	0,000023	9
		70	60 (4)	62,63	1,0978	0,1098	0,0160	0,00028	0,000028	9
G4	160	30	107 (4)	111,08	2,1494	0,2149	0,0090	0,00019	0,000019	20
		45	92 (6)	96,57	2,3192	0,2319	0,0104	0,00026	0,000026	19
		70	75 (2)	79,48	1,9461	0,1946	0,0126	0,00032	0,000032	18
G5	161	30	65 (2)	69,28	1,3339	0,1334	0,0144	0,00028	0,000028	11
		45	51 (1)	54,17	1,4775	0,1477	0,0185	0,00050	0,000050	11
		70	37 (1)	40,84	1,4193	0,1419	0,0245	0,00085	0,000085	10
G6	254	30	121 (4)	128,23	3,6621	0,3662	0,0078	0,00022	0,000022	156
		45	99 (1)	107,13	3,7515	0,3752	0,0093	0,00032	0,000032	40
		70	82 (4)	88,01	3,0501	0,3050	0,0114	0,00039	0,000039	29
G7	401	30	238 (1)	251,82	5,5965	0,5596	0,0040	0,00009	0,000009	101
		45	207 (1)	217,45	4,5202	0,4520	0,0046	0,00010	0,000010	88
		70	169 (2)	178,71	4,5267	0,4527	0,0056	0,00014	0,000014	66
G8	425	30	217(1)	230,19	5,1280	0,5128	0,0043	0,00010	0,000010	103
		45	189(1)	199,94	4,6446	0,4645	0,0050	0,00012	0,000012	85
		70	150(1)	164,54	4,4776	0,4478	0,0061	0,00017	0,000017	71
G9	709	30	339 (1)	354,39	6,5473	0,6547	0,0028	0,00006	0,000006	198
		45	280 (3)	293,35	6,6809	0,6681	0,0034	0,00008	0,000008	167
		70	235 (2)	247,62	5,7169	0,5717	0,0040	0,00009	0,000009	167
G10	1297	30	338 (1)	355,85	8,9005	0,8901	0,0028	0,00008	0,000008	524
		45	281 (1)	295,32	7,8905	0,7891	0,0034	0,00009	0,000009	451
		70	221 (1)	236,72	6,4980	0,6498	0,0040	0,00012	0,000012	420

Para os grafos menores, a abordagem *EEB* gerou uma mesma solução diversas vezes. Isso acontece pois, por ter um conjunto mais limitado de arestas, as opções de composição dos feixes são menores, fazendo com que o algoritmo gere a mesma solução durante iterações diversas. Essa afirmação pode ser verificada na terceira coluna da Tabela 4.5. Por exemplo, para o grafo G1 obteve-se a mesma solução para os ângulos $\alpha = 30°$ e $\alpha = 45°$, nas 100 execuções. No entanto, comparando-se essas soluções, verifica-se que, em boa parte dos casos, se tratavam de soluções que possuíam conjuntos diferentes de feixes, ou seja, eram iguais em relação à aptidão, mas não em sua estrutura. A Figura 4.16 mostra duas soluções com mesmo valor de aptidão para o grafo *Book USPolitics* (G7) e $\alpha = 70°$. As duas soluções possuem 169 feixes. Mas, como pode ser visto em algumas áreas destacadas na figura, os conjuntos de feixes são diferentes.

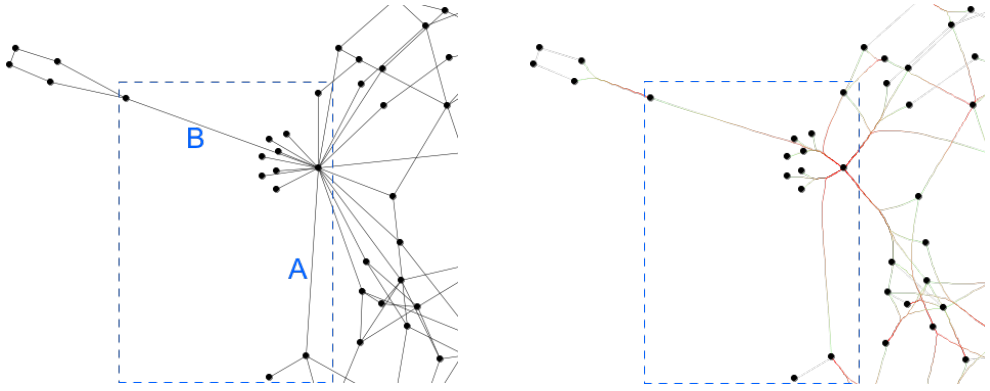


(a) (b)

Figura 4.16: *Soluções diferentes com mesmo valor de aptidão.*

Em geral, o método *EEB* produziu bons resultados em termos de número de feixes e ângulo máximo α. Por meio da alteração do parâmetro angular foi possível controlar o nível de união que se desejava variar. Por outro lado, a manipulação somente do ângulo produz alguns feixes que podem ter arestas que diferem significativamente em comprimento. Portanto, como esperado, o problema *ABEB* geralmente cria feixes com uma compatibilidade geométrica baixa.

Esse efeito está destacado na Figura 4.17, a qual detalha a solução obtida para o grafo *PlanarGD2015* (G3) com restrição angular $\alpha = 90°$. As arestas intituladas *A* e *B*, para esta versão de ângulo, foram unidas a outras arestas muito menores do que elas em tamanho.



(a) (b)

Figura 4.17: *União de arestas de tamanhos diferentes no grafo PlanarGD2015: (a) grafo original; (b) grafo com feixes.*

A variação da restrição angular provocou mudanças na composição dos feixes, sendo que mais arestas foram unidas em um número menor de feixes à medida que o ângulo aumentou. Um outro valor que variou com o ângulo foi o tempo médio de execução. No geral, o *EEB* levou menos tempo para os ângulos maiores. Isso se justifica pois ângulos menores no *EEB* significam que mais feixes são produzidos e, por consequência, as iterações nos operadores genéticos crescem, o que faz com que o algoritmo demore mais tempo avaliando as opções.

A Figura 4.18 ilustra as melhores soluções obtidas para os grafos G3, G5 e G6 com restrições angular $\alpha = 30°$ e $\alpha = 70°$. Comparando-se as figuras, é possível verificar o efeito do ângulo α na montagem dos feixes. Isso é mais evidenciado no grafo G5. Por exemplo, alguns vértices da solução para $\alpha = 70°$ possuem uma concentração menor de feixes, com mais arestas cada um. Os desenhos da melhor solução dos demais grafos encontram-se no Apêndice B.

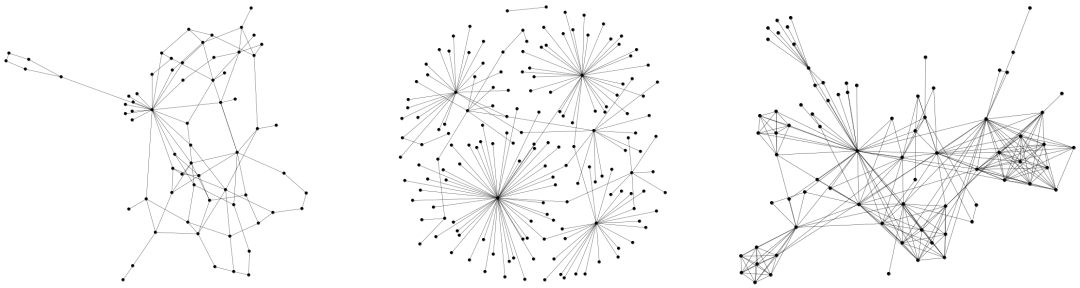
4.6.3 Análise da Convergência do EEB

Nesta seção, será analisado o comportamento de convergência do experimento (dentre os 100 testes controlados) com *EEB* que gerou a melhor solução de cada grafo testado, para os ângulos $\alpha = 30°, 45°$ e $70°$. É importante lembrar que o ciclo evolucionário termina quando não existe nenhuma alteração na qualidade da melhor solução durante 500 gerações consecutivas, sendo que são geradas no máximo de 16500 gerações.

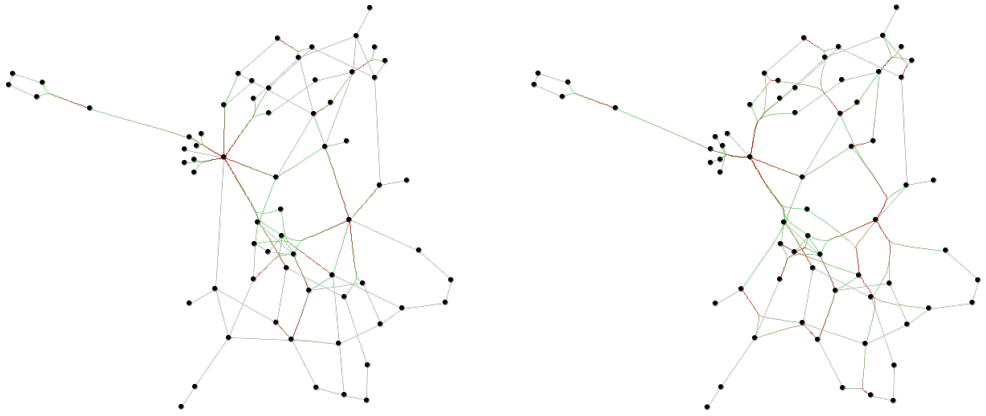
A Tabela 4.6 resume o número de gerações necessárias para produzir a melhor solução para cada grafo (esse é o número da geração em que a solução foi produzida). Entre parênteses, é indicada a média da quantidade de gerações da melhor execução. A Figura 4.19 ilustra os mesmos dados da tabela, apresentando a geração de convergência da melhor solução de cada grafo à esquerda, e a média de gerações de convergência (de todos os testes) à direita. É importante destacar que o número de gerações não é a mesma coisa de tempo de execução, já que uma geração pode levar mais tempo do que outra para ser computada devido ao tamanho dos indivíduos.

De forma geral, quanto maior o número de arestas, mais gerações foram necessárias até a convergência. A exceção mais significativa é para o grafo *MovieLens* (G5), o qual tem a característica de conter poucos ciclos. Sua convergência foi bem mais rápida do que, por exemplo, a do grafo *Dolphin* (G4) que se assemelha em número de arestas.

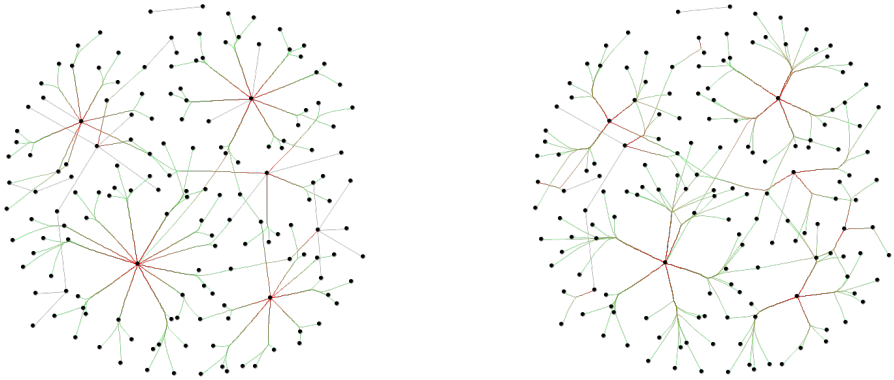
Ao analisar os ângulos, a quantidade de gerações para convergência para os grafos maiores (G6, G7, G8 e G9) foi, em geral, maior para a ordem inversa de tamanho dos ângulos. Por exemplo, para gerar a melhor solução do grafo *Book USPolitics* (G7) para $\alpha = 30°$ foram necessárias 5017 gerações, tendo sido gerada uma solução com 238 feixes, enquanto, para $\alpha = 70°$ levou apenas 1540 gerações para produzir uma solução



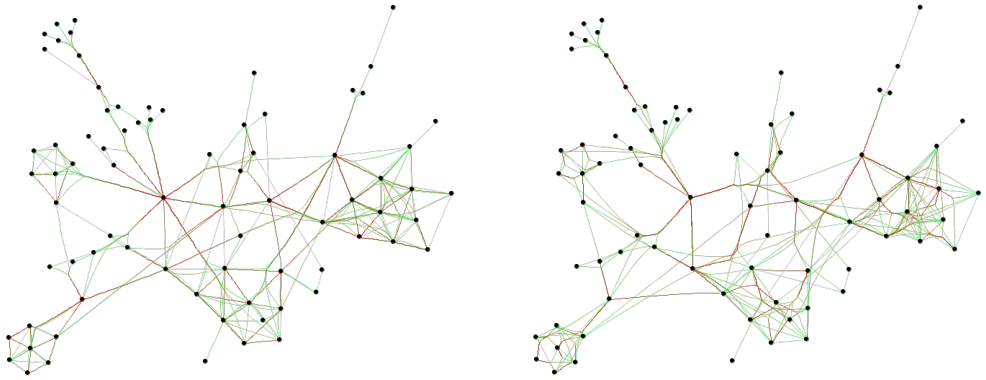
(a) grafo original G3 (b) grafo original G5 (c) grafo original G6



(d) grafo G3, $\alpha = 30°$ e 75 feixes (e) grafo G3, $\alpha = 70°$ e 60 feixes



(f) grafo G5, $\alpha = 30°$ e 65 feixes (g) grafo G5, $\alpha = 70°$ e 37 feixes



(h) grafo G6, $\alpha = 30°$ e 121 feixes (i) grafo G6, $\alpha = 70°$ e 82 feixes

Figura 4.18: *Resultados para o problema ABEB com diferentes restrições angulares.*

Tabela 4.6: *Comparação da quantidade de gerações da melhor execução do EEB para o problema ABEB.*

Grafo	30°	45°	70°
G1	2 (2,95)	1 (6,24)	16 (17,38)
G2	1094 (370,52)	421 (362,52)	441 (190,67)
G3	629 (263,21)	769 (263,21)	600 (290,88)
G4	960 (661,00)	1469 (640,27)	1253 (593,00)
G5	1039 (189,63)	245 (291,41)	176 (296,96)
G6	1356 (1416,21)	2125 (1155,95)	709 (872,47)
G7	5017 (1967,00)	2056 (1813,37)	1540 (1435,30)
G8	3403 (1823,52)	2741 (1533,54)	3076 (1365,77)
G9	2935 (2386,10)	4117 (2287,24)	3456 (2032,08)
G10	4013 (3529,78)	4752 (2977,76)	2841 (2407,45)

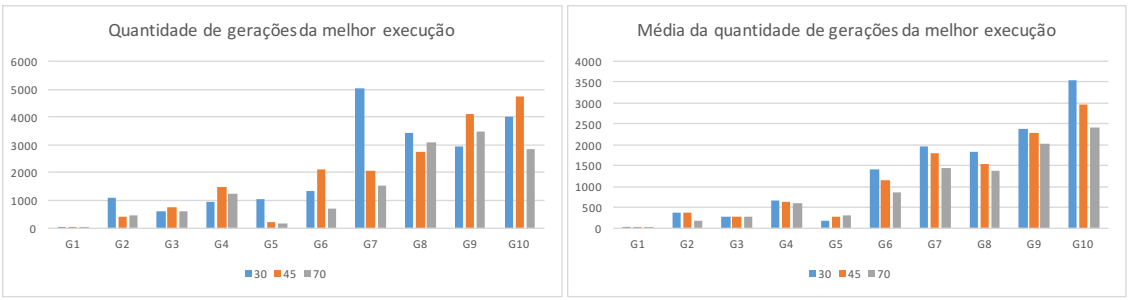


Figura 4.19: *Comparação da quantidade de gerações da melhor execução do EEB para o problema ABEB.*

do 169 feixes. No entanto, ao contrário do que se observou para o aspecto "tempo de processamento", essa não é uma característica geral, apenas uma tendência para os grafos maiores e para certos ângulos.

No caso do teste que gerou a melhor solução (gráfico a esquerda da Figura 4.19), o comportamento é semelhante ao da média com picos de variação. Um fator que influencia esse comportamento, provavelmente, seja a característica estocástica da técnica de computação evolutiva implementada.

As Figuras 4.20 e 4.21 correspondem aos gráficos de evolução da qualidade ao longo das gerações do teste que gerou a melhor solução para os grafos Les Miserables e Flare, respectivamente. Os gráficos dos demais grafos encontram-se no Apêndice C. Analisando-se esses grafos, percebe-se que a evolução da qualidade da melhor solução foi condizente com os padrões de um algoritmo evolucionário.

Esses resultados, juntamente com os demais discutidos nesse capítulo, indicam a boa performance do método da abordagem *EEB* na resolução do problema *ABEB* para grafos de pequeno, médio e grande porte. Os resultados se mostraram adequados às restrições impostas. Contudo, o tempo de execução é uma de suas desvantagens à medida que a instância do problema cresce.

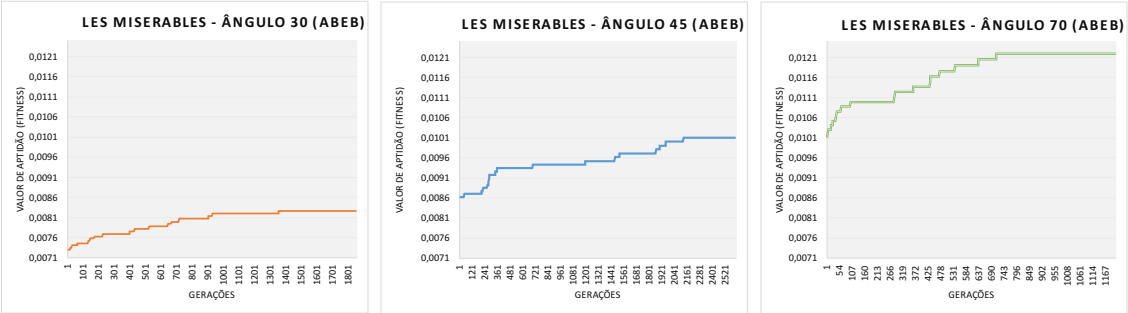


Figura 4.20: *Convergência do grafo Les Miserables (G6 - 254 arestas) para o problema ABEB com EEB. Soluções com 121 ($\alpha = 30°$), 99 ($\alpha = 45°$) e 82 ($\alpha = 70°$) feixes.*

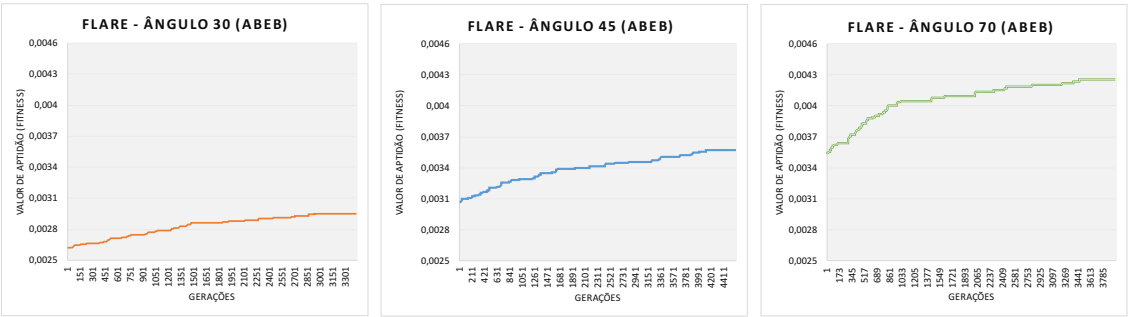


Figura 4.21: *Convergência do grafo Flare (G9 - 709 arestas) para o problema ABEB com EEB. Soluções com 339 ($\alpha = 30°$), 280 ($\alpha = 45°$) e 235 ($\alpha = 70°$) feixes.*

4.7 Considerações Finais

Neste capítulo, foi proposto um novo problema de união de arestas identificado como *ABEB*. Ele é uma extensão do problema genérico $EB-star$ incluindo uma restrição angular máxima. O problema foi definido e algumas propriedades e considerações a respeito da interferência do limite angular nos resultados foram discutidas.

Um modelo de programação linear inteira foi proposto para o problema *ABEB*. A análise da sua própria implementação demonstrou o quanto o problema *ABEB* se torna complexo conforme o tamanho do grafo e o valor do ângulo máximo aumentam. A implementação falhou ao tentar resolver o problema para instâncias medianas. Por outro lado, uma abordagem heurística baseada em computação evolucionária, o *EEB*, foi desenvolvida e conseguiu resolver de forma satisfatória instâncias de problemas para dez grafos de tamanho crescente. Verificou-se também que o *EEB* tende a demorar menos tempo para valores maiores do ângulo máximo

Os resultados das técnicas foram satisfatórios. Entretanto, do ponto de vista de problemas para união de arestas, para um dado desenho de grafo como entrada, podem

existir diversas soluções como o mesmo número de feixes satisfazendo uma restrição angular específica, mas com uma composição diferente de feixes. Para distinguir entre estas soluções, um objetivo adicional que maximize o valor de compatibilidade (geométrica, semântica, topológica, etc.) entre as arestas deveria ser adicionado ao problema, como normalmente é identificado em algoritmos encontrados na literatura.

No próximo capítulo, esse aspecto é investigado por meio da proposição de um novo problema, cuja meta é produzir soluções em que as arestas de um feixe são o mais compatíveis possível, gerando o menor conjunto de feixes de arestas.

CAPÍTULO 5

União Explícita de Arestas em Feixes Centralizados com Multicritérios

Este capítulo apresenta um outro problema para união de arestas que também foi modelado usando-se a formulação *EB* (ver Capítulo 3). Esse problema se assemelha ao *ABEB* discutido no Capítulo 4, sendo que o ângulo máximo não é mais uma restrição. Ele é empregado agora, junto com outro critério de compatibilidade, como parte de um componente da função objetivo visando maximizar a compatibilidade geométrica entre as arestas. Ao longo deste capítulo, são apresentadas a formulação matemática do novo problema e algumas considerações sobre as implicações da inserção de um outro critério de compatibilidade no modelo de otimização. Na implementação do algoritmo para resolução do problema proposto, a abordagem *EEB* foi adaptada. Essa adaptação é explicada, bem como os resultados de experimentos.

5.1 Descrição do Problema

Como discutido no Capítulo 4, analisando-se os resultados do problema *ABEB*, é possível notar que, para um dado desenho de grafo de entrada, podem existir diversas soluções com o mesmo número de feixes que satisfazem a restrição angular, mas que possuem uma composição distinta de arestas.

Para exemplificar essa afirmação, além do caso já citado para a Figura 4.16, considere a Figura 5.1. Ela mostra três desenhos de grafos que são soluções do problema *ABEB* para o grafo *MovieLens*. Todos os desenhos possuem 38 feixes restritos a um ângulo $\alpha = 70°$. Considerando-se a definição do problema *ABEB*, esses desenhos possuem a mesma qualidade, apesar dos agrupamentos diferentes de arestas.

Na Figura 5.1, a lista de feixes está posicionada abaixo de cada grafo. Cada linha da lista representa um feixe seguindo o esquema:

$$\langle IDFeixe1\rangle : \langle IDAresta1\rangle - \langle IDAresta2\rangle - \ldots - \langle IDArestak\rangle$$

Alguns conjuntos de arestas são diferentes (eles estão destacados em verde). Isso também pode ser verificado na imagem dos três grafos, na Figura 5.1. Nela, estão circulados em azul alguns conjuntos de feixes associados a um mesmo vértice central, mas que diferem nas três soluções.

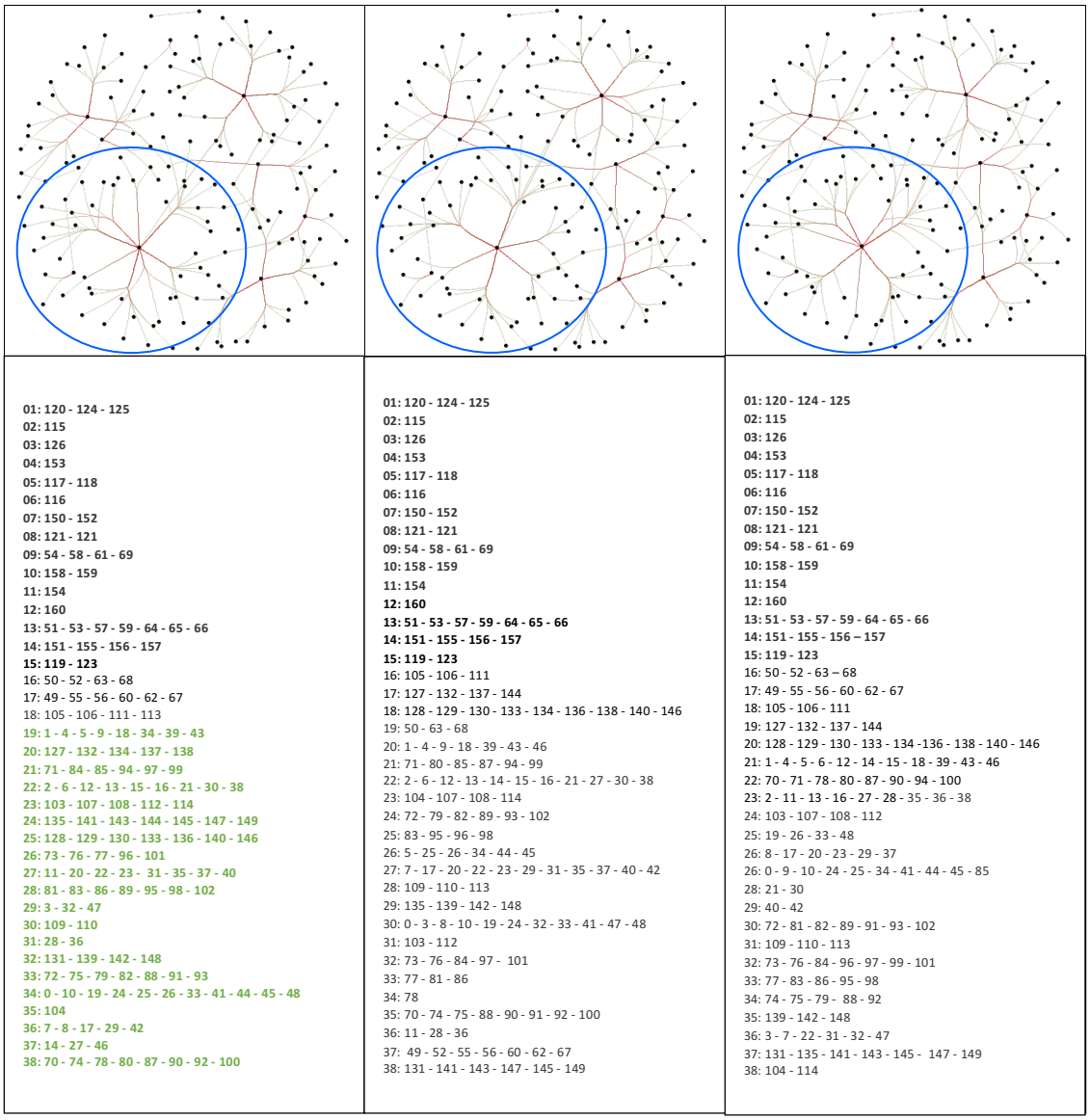


Figura 5.1: *Soluções diferentes para uma mesma instância do problema ABEB.*

Dessa forma, a otimização apenas da quantidade de feixes não é interessante, quando o nível de qualidade requerida também está ligado a outros critérios de similaridade, além do ângulo. Uma métrica natural que pode ser usada para distinguir a qualidade dos conjuntos de arestas é a medida de compatibilidade de tamanho (ver Seção 2.2). Além disso, é possível ponderar em uma escala mais gradual o atendimento de uma medida de ângulo máximo.

Neste capítulo, é proposto um problema para união de arestas que inclui uma nova função objetivo de maximização da compatibilidade entre arestas. A meta é produzir o menor número de feixes cujas arestas sejam as mais compatíveis possível, segundo esses novos critérios. Este problema é denominado como **união explícita de arestas em feixes centralizados com multicritérios**, ou, simplesmente, *CBEB* (sigla em inglês de *Compatibility-Based Edge Bundling*). A definição do *CBEB* é dada a seguir.

5.2 Modelagem Matemática

Problema 5.1 *Seja um grafo simples $G = (V,E)$, um desenho D de G com arestas representadas por linhas retas e vértices em posições fixas. Seja $C(p,q)$ a medida de compatibilidade entre um par de arestas $p,q \in E$ com base em D. O problema de união explícita de arestas em feixes centralizados com multicritério (CBEB) é determinar uma decomposição S de E em subconjuntos disjuntos $E_1, E_2, \ldots, E_n$, com $E = \cup_{i=1}^{n} E_i$ e $E_i \cap E_j \neq \emptyset$, para $1 \leq i, j \leq |E|$, $i \neq j$, tal que, cada E_i induz um subgrafo estrela G_i para todo par de arestas e_{ij}, $e_{ik} \in E_i$, que maximize a compatibilidade total do grafo $C_G = \sum_{i=1}^{n} C_{E_i}$, onde $C_{E_i} = \sum_{p,q \in E_i} C(p,q)$ é a compatibilidade de um feixe E_i, e minimize n*

Identificando-se os elementos da formulação *EB* neste problema, tem-se que:

- G é um grafo simples;
- D é um desenho de pontos e linhas com vértices fixos;
- os objetivos são:
  - F_1: encontrar um conjunto S de E que seja o menor possível ($n \leq k$); e
  - F_2: encontrar um conjunto S de E que maximize a compatibilidade geral C_G do grafo.
- As restrições são:
  - P_1: que assegure que os conjuntos E_i sejam disjuntos; e
  - P_2: que assegure que as arestas de E_i sejam adjacentes entre si.

Para fins de simplificação, o problema *CBEB* foi convertido em um problema mono objetivo de maximização da função de soma ponderada:

$$f = w_1 \cdot C_G + w_2 \cdot \frac{1}{n}, \text{ sendo } 0 \leq w_i \leq 1, i = 1,2. \tag{5-1}$$

Além disso, o ângulo limite α não é mais uma restrição do problema, mas entra, como discutido adiante, como um *threshold* para penalizar o cálculo de compatibilidade angular entre arestas. Em termos de compatibilidade C_{E_i}, não existe um rigor sobre

qual medida utilizar. É possível empregar qualquer medida que expresse o nível de compatibilidade desejado pelo usuário, como: a compatibilidade geométrica, semântica, de topologia, etc. Na resolução do problema *CBEB* foram usadas duas das medidas geométricas definidas em [43]. A escolha dessas medidas se deve ao seu uso consolidado em diversos trabalhos da área.

Na próxima seção, são discutidas as características dessa nova função introduzida no modelo *CBEB*, identificada simplesmente como função de compatibilidade. Além de ser responsável por calcular o valor de compatibilidade do feixe e do grafo, a mesma define também a penalidade aplicada a soluções de baixa compatibilidade.

5.3 Função de Compatibilidade

Como visto na Seção 2.2, Holten e Wijk [43] definiram que a compatibilidade geométrica é baseada em quatro medidas elementares: angular (C_a), escalar (C_s), posição (C_p) e visibilidade (C_v) das arestas, todas em uma escala entre $[0,1]$, sendo o valor 1 considerado muito compatível e 0 pouco compatível. A **compatibilidade total** entre duas arestas $C(p,q) \in [0,1]$ foi definida conforme a Equação 2-5.

Para o problema *CBEB*, foi implantada uma parte desse esquema de cálculo da compatibilidade geométrica. Apenas os valores da compatibilidade angular e da compatibilidade escalar foram usados no cálculo da compatibilidade entre duas arestas. Para o cálculo da compatibilidade escalar, foi empregada a Equação 2-2 sem modificação. Entretanto, o cálculo da compatibilidade angular foi ligeiramente alterado para refletir o ângulo real entre duas arestas e não o seu cosseno. O objetivo foi limitar a união de arestas com ângulo acima de um máximo pré-definido, ao invés de somente evitar arestas perpendiculares.

No problema *ABEB*, o ângulo foi tratado como restrição. Já no problema *CBEB*, ele faz parte do objetivo e está associado à compatibilidade de escala. Após a determinação de qual é o ângulo, em graus, entre as arestas analisadas, esse valor é normalizado entre 0 e 1, conforme a equação abaixo.

$$C_a^x(p,q) = 1 - (\alpha(p,q) \cdot 0,5)/90 \text{ , com } \alpha = arccos(\frac{p \cdot q}{\mid p \mid\mid q \mid}) \tag{5-2}$$

As medida de compatibilidade de posição e visibilidade são dispensáveis, pois já são tratadas por meio da restrição de adjacência.

Com base nas equações 2-2 e 5-2, a nova fórmula da compatibilidade total entre duas arestas para o problema *CBEB* é:

$$C(p,q) = C_a^x(p,q) \cdot C_s(p,q). \tag{5-3}$$

Nos experimentos relatados na Seção 5.5 foram feitos dois testes. Um com $C(p,q)$ incluindo C_a^x e $C_s(p,q)$ em sua composição e outro tratando somente o ângulo como componente da compatibilidade, conforme a fórmula:

$$C(p,q) = C_a^x(p,q). \tag{5-4}$$

Nesse último caso, o problema se assemelha ao *ABEB*, mas tendo o ângulo como objetivo e não restrição.

A **compatibilidade total de um feixe** E_i, denotada por C_{E_i}, é definida nesta tese como o somatório da compatibilidade das arestas de E_i duas a duas. No entanto, se entre as arestas pertencentes ao feixe existirem pares de arestas consideradas incompatíveis, esse somatório talvez não reflita a realidade.

Por exemplo, para a situação fictícia representada na Figura 5.2, têm-se dois grupos de arestas que poderiam ser unidas em feixes e que possuem o mesmo valor de compatibilidade total ($C_E = 1.6$). Se a compatibilidade das arestas $C(e_1,e_3) = 0.2$ para o primeiro grafo for considerada baixa para os parâmetros de configuração do problema, nota-se, claramente que a qualidade do grupo da figura superior será pior do que a do grafo inferior, apesar de o valor numérico ser igual.

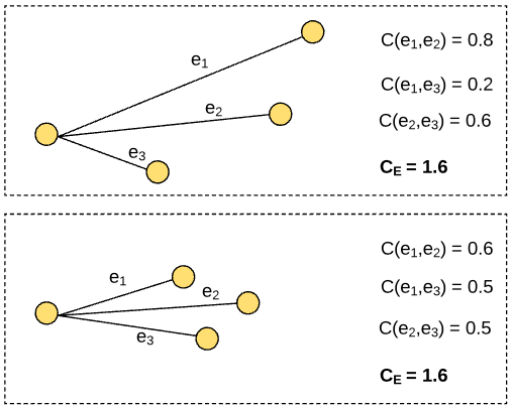


Figura 5.2: *Soluções diferentes com mesmo valor de compatibilidade.*

Para resolver esse problema, foi proposto um esquema de penalização de soluções com baixa compatibilidade por meio da definição de *thresholds* para os valores de C_a^x e C_s. O *threshold* T_a (relacionado à C_a^x) é o ângulo máximo aceitável α, e o *threshold* T_s (relacionado a C_s) é a diferença mínima permitida entre o tamanho das arestas. O *threshold* total é $T = T_a \cdot T_s$.

Durante o ciclo evolucionário, foi permitida a criação de indivíduos que possuíssem um valor de compatibilidade baixo, por exemplo, com arestas que não respeitassem o ângulo máximo, ou possuíssem tamanhos muito diferentes. No entanto, essas soluções foram penalizadas, obtendo-se um valor de aptidão menor.

Para avaliar e penalizar a compatibilidade de um feixe E_i, primeiramente é encontrado o par de arestas $p,q \in E_i$ que possui o menor valor de compatibilidade $C(p,q)$ entre todos os pares de arestas do feixe. Se $\min C(p,q) < T$, então C_{E_i} é penalizado com uma constante $p_e < 0$. Para o exemplo da Figura 5.2, caso $T = 0.3$, o valor de C_E para o primeiro grafo não seria 1,6, mas, sim, um valor negativo, pois o feixe seria penalizado por ter a compatibilidade de duas arestas abaixo do *threshold*.

Com base nestas regras, tem-se a Equação 5-5 que expressa o cálculo da compatibilidade C_{E_i} de um feixe E_i para o problema *CBEB*.

$$C_{E_i} = \begin{cases} \sum C(p,q), \forall p,q \in E_i, & \text{se } \min(C(p,q)) \geq T; \\ p_e, & \text{se } \min(C(p,q)) < T; \\ 0, & \text{se } |E_i| = 1. \end{cases} \tag{5-5}$$

Observa-se que, pela equação 5-5, os feixes com apenas uma aresta tem sua compatibilidade definida como 0. Desta forma, eles não influenciam a função objetivo via compatibilidade, mas apenas por meio da contagem final do número de feixes.

Os valores T_a, T_s e p_e são configuráveis e definidos pelo usuário, que pode por meio deles controlar o resultado final do desenho. A possibilidade de controle das características do problema, e, portanto, das soluções, por parte do usuário, é uma das grandes vantagens de uma versão de otimização de problemas de união de arestas.

Por fim, a Equação 5-6 mostra o cálculo da **compatibilidade total do grafo** C_G, que é o somatório da compatibilidade de cada feixe. Quanto maior for o valor de C_G mais apto é um indivíduo, em termo de similaridade das arestas pertencentes a cada feixe.

$$C_G = \sum_{i=1}^{n} C_{E_i}. \tag{5-6}$$

5.4 Método Evolucionário EEB Aplicado ao Problema CBEB

A abordagem *EEB* discutida no Capítulo 4 precisou ser adaptada para resolver o problema *CBEB*, contudo, as mudanças foram poucas. A representação de um indivíduo não foi alterada e o fluxo de execução é o mesmo descrito na Seção 4.5.1. A principal modificação se deu na construção da função de aptidão que combina os dois valores a serem otimizados, como descrito na Seção 5.2.

Foram utilizados os mesmos procedimentos de inicialização discutidos na Seção 4.5.5. A única alteração é com relação ao teste de satisfação das restrições. Na geração aleatória da primeira metade da população, não é mais verificado se está sendo satisfeita a

restrição angular. São feitos apenas os testes de asseguração da adjacência e dos conjuntos disjuntos.

Já para a segunda metade da população, guiada pela cobertura de vértices, além dessas duas restrições, os indivíduos que possuem uma compatibilidade de arestas inferior ao *threshold* T também são rejeitados. Apesar de a compatibilidade não ser uma restrição do problema *CBEB*, o seu uso na inicialização visou criar diversidade na população inicial.

Nenhuma alteração foi feita nas funções de seleção (ver Seção 4.5.6) e cruzamento (ver Seção 4.5.7). Os operadores de mutação (ver Seção 4.5.8) sofreram poucas mudanças. Para o problema *ABEB*, os operadores de união, fusão e movimentação verificam a restrição angular antes de as arestas serem adicionadas a seus novos feixes. Para o *CBEB* isso não é realizado. Assim como na inicialização aleatória, essa característica foi relaxada, pois é tratada em nível de função objetivo. Os testes de asseguração da adjacência e dos conjuntos disjuntos se mantêm.

O módulo de roteamento e renderização dos feixes também não foi alterado e continua-se usando uma versão adaptada do algoritmo *force-based edge bundling*.

5.5 Experimentos e Discussões

Os experimentos foram conduzidos para o mesmo grupo de grafos descritos na Seção 4.6. O desenho inicial de cada grafo foi pre-definido em um arquivo de entrada. Foram executados três experimentos controlados e similares. No Capítulo 4 o ângulo máximo entre duas arestas foi usado como restrição do problema.

Como dito anteriormente, no primeiro experimento de resolução do problema *CBEB*, o ângulo entre duas arestas foi inserido como medida de compatibilidade, conforme a Fórmula 5-4. O objetivo desse teste foi verificar qual o impacto de tratar o ângulo, não mais como uma restrição, mas sim como um objetivo do problema. Nesse experimento a penalidade foi definida como $p_e = -1$.

O segundo experimento utilizou o ângulo e a diferença de escala entre duas arestas para definir o nível de compatibilidade entre elas, conforme a Fórmula 5-3. Para esse teste a penalidade também foi configurada como $p_e = -1$.

O terceiro experimento usou a mesma compatibilidade do teste dois, mas nesse caso com $p_e = -3$. Os testes com penalidades diferentes visaram identificar como esse parâmetro influência o tempo, e em que nível ele restringe as soluções não condizentes com valores de compatibilidade desejáveis na solução.

Para garantir maior confiabilidade, os experimentos controlados consistiram em executar o algoritmo evolutivo em 100 testes independentes para cada grafo e ângulo,

cujos parâmetros de controle definidos foram: o **tamanho da população** ($u = 150$), **taxa de cruzamento** ($p_c = 0,98$) e **taxa de mutação** ($p_m = 0,4$).

Para o problema *CBEB*, os ângulos $\alpha \in \{30°, 45°, 70°\}$ também foram considerados nos três experimentos, mas eles foram empregados para cálculo do *thresold* T_a usando-se a fórmula $T_a = 1 - (\alpha \cdot 0,5)/90$. Já o *thresold* T_b foi definido como $0,890$. Os pesos dos componentes da função objetivo foram escolhidos empiricamente como $w_1 = 0,2$ e $w_2 = 0,8$ (ver Seção 5.2).

Durante os testes o ciclo evolucionário foi executado até não ser computada nenhuma melhoria da melhor solução por 500 gerações consecutivas ou um limite total de 16500 gerações. Todos os testes foram executados em um MacBook Pro com um processador Intel Core i7 de 2,9 GHz e 8 GB de RAM 1600MHz-DDR3.

5.5.1 Calibração dos Parâmetros

Assim, como para o problema *ABEB*, a taxa de mutação foi definida depois de testes preliminares com cinco grafos (G1, G2, G3, G4 e G5). Para cada, grafo foram feitas 100 execuções para cada uma das taxas de mutação, $p_m = 0,1$, $p_m = 0,2$, $p_m = 0,3$, $p_m = 0,4$, e $p_m = 0,5$.

A Tabela 5.1 sumariza os valores médios dos testes e registra o número de feixes da melhor solução. As cinco taxas tiveram resultados semelhantes, mas houve uma pequena variação em qualidade e tempo de execução para pior, no caso de taxas inferiores a 0,4. A taxa de 0,5 foi a que obteve melhor performance em termos de qualidade, com pequena variação no tempo de execução.

Apesar do bom desempenho da taxa de 0,5, levou-se em consideração que efetuar a mutação em 50% da população seria um valor muito alto; desta forma, a taxa de mutação $p_m = 0,4$ foi escolhida.

5.5.2 Resultados

A Tabela 5.2 apresenta o resultado do experimento 1 para o problema *CBEB*. Ela segue a mesma estrutura da Tabela 4.5, com a inclusão de duas colunas, uma que mostra o número de feixes dos resultados do problema *ABEB*, e outra que mostra a média da compatibilidade dos resultados do problema *CBEB*.

De uma forma geral, os desenhos das soluções do problema *ABEB* e *CBEB* são muito semelhantes para os grafos pequenos, quando o número de feixes é igual, mas possuem diferenças significativas em grafos maiores. A Figura 5.3 compara os resultados para o grafo Les Miserables (G6). As figuras dos demais grafos podem ser encontradas no Apêndice B.

Tabela 5.1: *Testes preliminares com a taxa de mutação para o problema CBEB.*

Grafo	Ângulo	Número de feixes	Média de feixes	Média da aptidão	Desvio padrão da aptidão	Média do tempo (sec.)
Taxa de mutação $p_m = 0,1$						
G1	30	20	20,060	1,924	0,054	2
	45	17	18,150	2,709	0,272	2
	70	16	16,740	4,145	0,293	3
G2	30	57	60,410	4,093	0,327	13
	45	49	53,180	6,440	0,511	15
	70	39	43,620	12,585	1,170	18
G3	30	83	84,950	4,327	0,288	13
	45	80	82,430	6,386	0,773	14
	70	68	71,450	16,090	0,628	19
G4	30	131	136,740	5,010	0,533	36
	45	118	125,220	8,512	0,895	50
	70	92	101,100	19,129	1,645	65
G5	30	82	91,570	22,965	1,791	25
	45	70	77,840	36,657	4,047	36
	70	63	61,490	65,970	8,609	44
Taxa de mutação $p_m = 0,2$						
G1	30	20	20,020	1,934	0,051	2
	45	16	18,140	2,704	0,299	2
	70	16	16,740	4,122	0,297	3
G2	30	57	59,200	4,384	0,310	17
	45	49	51,910	6,839	0,496	17
	70	37	41,000	13,349	1,011	20
G3	30	82	84,440	4,481	0,265	14
	45	78	81,620	6,784	0,786	18
	70	65	70,400	16,461	0,660	20
G4	30	130	134,400	5,534	0,429	37
	45	116	122,410	9,305	0,845	49
	70	93	97,530	20,720	1,454	62
G5	30	80	87,410	25,186	2,067	36
	45	68	73,010	40,459	4,578	41
	70	47	54,690	75,467	7,885	47
Taxa de mutação $p_m = 0,3$						
G1	30	20	20,010	1,936	0,025	2
	45	16	17,960	2,786	0,320	2
	70	16	16,750	4,100	0,251	2
G2	30	56	58,860	4,472	0,245	11
	45	47	50,920	7,091	0,519	18
	70	36	39,680	13,754	0,892	18
G3	30	81	84,230	4,530	0,275	15
	45	77	81,330	6,790	0,801	15
	70	65	70,090	16,581	0,706	15
G4	30	130	133,620	5,705	0,420	37
	45	114	120,430	9,831	0,758	47
	70	90	96,060	21,126	1,358	59
G5	30	78	85,600	26,164	2,455	39
	45	60	70,080	42,977	4,283	37
	70	49	53,570	76,987	6,223	36
Taxa de mutação $p_m = 0,4$						
G1	30	20	20,020	1,934	0,033	2
	45	16	18,100	2,722	0,289	2
	70	16	16,790	4,097	0,273	2
G2	30	56	58,520	4,544	0,238	13
	45	48	50,720	7,093	0,491	14
	70	37	39,600	13,833	0,862	15
G3	30	81	84,000	4,604	0,267	13
	45	77	80,630	7,025	0,759	19
	70	65	69,400	16,795	0,726	18
G4	30	130	132,740	5,914	0,374	34
	45	116	119,860	9,982	0,606	46
	70	88	94,880	21,737	1,427	51
G5	30	78	83,450	27,921	2,334	36
	45	66	68,980	44,263	3,933	38
	70	47	53,090	76,923	5,876	35
Taxa de mutação $p_m = 0,5$						
G1	30	20	20,000	1,938	0,010	1
	45	16	18,160	2,704	0,340	1
	70	16	16,810	4,120	0,294	1
G2	30	57	58,340	4,578	0,250	11
	45	46	50,400	7,216	0,446	14
	70	37	39,400	13,941	0,921	15
G3	30	79	83,670	4,680	0,332	14
	45	77	80,630	7,041	0,784	17
	70	65	68,980	16,850	0,754	19
G4	30	129	132,440	5,983	0,332	34
	45	112	119,400	10,044	0,721	40
	70	88	94,080	22,102	1,296	51
G5	30	76	83,260	27,852	2,572	43
	45	64	68,100	45,617	4,019	40
	70	50	52,170	78,450	6,303	37

No entanto, em sua maioria, as soluções do problema *ABEB* (ver Tabela 4.5) obtiveram uma quantidade maior de feixes do que os resultados obtidos no experimento 1 para o problema *CBEB*.

Tabela 5.2: *Resultados do experimento 1 para o problema CBEB, média de 100 execuções independentes,* $u = 150$, $p_c = 0,98$, $p_m = 0,4$, $w_1 = 0,2$, $w_2 = 0,8$ *e* $p_e = -1$.

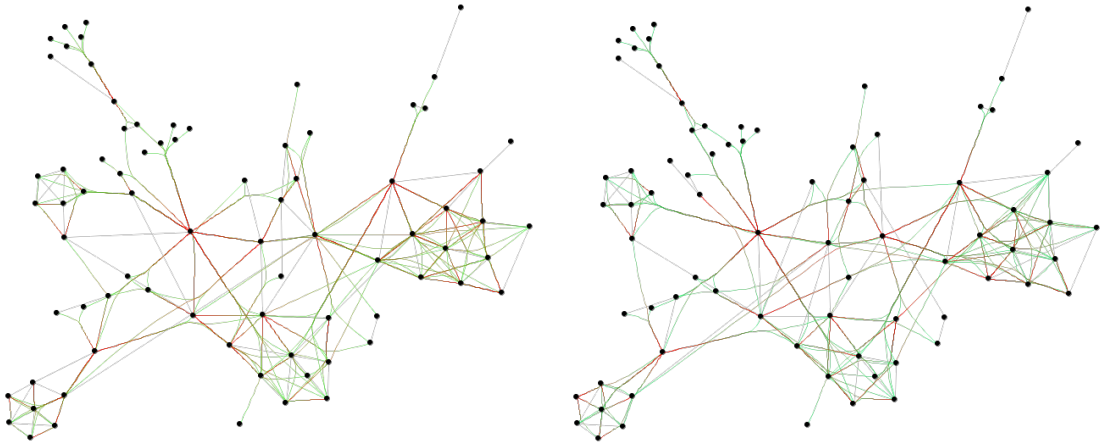
Grafo	Arestas	α(°)	Feixes ABEB	Feixes da melhor solução	Média da compatibilidade	Média de feixes	Desvio padrão de feixes	Erro padrão de feixes	Média da aptidão	Desvio padrão da aptidão	Erro padrão da aptidão	Média tempo (sec.)
G1	28	30	17 (100)	17 (3)	15,47	17,00	0,0000	0,0000	3,1417	0,00046	0,000467	2
		45	16 (100)	16 (1)	18,16	16,00	0,0000	0,0000	3,6826	0,00187	0,001878	2
		70	13 (89)	13 (13)	28,83	13,11	0,3145	0,0314	5,8265	0,19680	0,019680	1
G2	78	30	45 (1)	45 (1)	39,06	47,61	1,3096	0,1310	7,8290	0,31660	0,031660	12
		45	37 (3)	37 (1)	59,91	38,85	1,2503	0,1250	12,0029	0,47361	0,047361	9
		70	31 (6)	30 (1)	100,38	32,51	1,2752	0,1275	20,1003	0,90779	0,090779	10
G3	101	30	75 (4)	75 (3)	35,53	76,50	0,9266	0,0927	7,1157	0,22087	0,022087	13
		45	69 (1)	71 (1)	54,54	73,64	1,2514	0,1251	10,9182	0,30489	0,030489	15
		70	60 (4)	60 (1)	93,19	62,81	1,5222	0,1522	18,6504	0,49209	0,049209	16
G4	160	30	107 (4)	106 (1)	61,08	109,74	1,9208	0,1921	12,225	0,56580	0,056580	33
		45	92 (6)	91 (1)	94,37	94,35	1,9142	0,1914	18,8823	0,79239	0,079239	31
		70	75 (2)	72 (1)	169,83	76,26	2,3034	0,2303	33,9759	1,59542	0,159542	34
G5	161	30	65 (2)	68 (1)	198,88	69,33	1,7000	0,1700	39,7883	1,19975	0,119975	19
		45	51 (1)	51 (1)	320,55	54,23	1,6869	0,1687	64,1253	1,95537	0,195537	18
		70	37 (1)	39 (1)	575,29	41,32	1,6810	0,1681	115,0769	3,79203	0,379203	15
G6	254	30	121 (4)	120 (1)	259,52	122,83	3,0848	0,3085	51,9103	2,28275	0,228275	92
		45	99 (1)	94 (1)	401,32	99,61	2,5933	0,2593	80,2716	3,55609	0,355609	84
		70	82 (4)	76 (1)	668,18	79,71	2,4997	0,2500	133,6469	5,64992	0,564992	77
G7	401	30	238 (1)	227 (1)	332,89	236,80	5,0010	0,5001	66,5815	2,57686	0,257686	244
		45	207 (1)	192 (1)	476,02	201,99	4,5137	0,4514	95,2086	3,52921	0,352921	263
		70	169 (1)	156 (1)	817,91	161,85	4,3632	0,4363	163,5866	5,87810	0,587810	270
G8	425	30	217 (1)	210 (1)	366,04	220,11	5,0430	0,5043	73,2115	2,96272	0,296272	305
		45	189 (1)	178 (1)	551,06	186,58	4,8246	0,4825	110,2163	4,49887	0,449887	249
		70	150 (1)	145 (1)	936,94	150,45	4,4571	0,4457	187,3943	7,14529	0,714529	245
G9	709	30	339 (1)	332 (1)	773,42	339,33	6,0654	0,6065	154,6858	4,74599	0,474599	529
		45	280 (3)	264 (1)	1172,92	276,64	6,0795	0,6079	234,5875	7,91199	0,791199	546
		70	235 (2)	228 (1)	1870,71	230,66	5,0055	0,5005	374,1446	11,69390	1,169390	548
G10*	1297	30	338 (1)	316 (1)	5534,76	316,73	4,6823	1,2090	1106,9536	29,73234	7,676856	2173
		45	281 (1)	256 (1)	8235,99	261,20	5,5703	1,4383	1647,2007	52,88496	13,654838	2118
		70	221 (1)	207 (1)	13111,60	211,67	5,2735	1,3616	2622,3237	90,99913	23,495874	2149

**Para o grafo G10 foram feitas apenas 15 execuções independentes para cada ângulo.*

Analisando-se individualmente os feixes, observa-se que os desenhos gerados para o problema *CBEB* tendem a ter uma compatibilidade maior entre as arestas. As Figuras 5.4 e 5.5 exemplificam esta afirmação mostrando soluções similares para os dois problemas. As regiões marcadas representam alguns feixes diferentes. Percebe-se que a solução do problema *CBEB* possui uma relação angular menor, o que equivale a dizer que as arestas são mais compatíveis. No geral, a inserção da compatibilidade como objetivo no primeiro experimento obteve bons resultados.

No segundo teste avaliou-se a inserção da escala como métrica de similaridade. A Tabela 5.3 sumariza os resultados. Ela também segue a mesma estrutura da Tabela 4.5, com a inclusão de apenas uma coluna para a média da compatibilidade.

A natureza estocástica do algoritmo, associada ao tamanho do espaço de busca, produz soluções significativamente diferentes em termos de média e desvio padrão da aptidão. Entretanto, o erro padrão reflete a baixa flutuação da amostra. É importante observar que resultado similar não acontece com o grafo *USAirline* (G10); primeiro, porque foram executados somente 15 testes independentes para esse grafo e, segundo devido ao seu tamanho. Esses resultados evidenciam como o número de arestas pode aumentar consideravelmente o conjunto de soluções candidatas para o problema *CBEB*.



(a) (b)

Figura 5.3: *Comparação dos resultados do problema CBEB com ângulo como objetivo para o grafo LesMiserables (G6) com* $\alpha = 30°$*: (a) solução ABEB com 121 feixes e (b) solução CBEB com 120 feixes.*

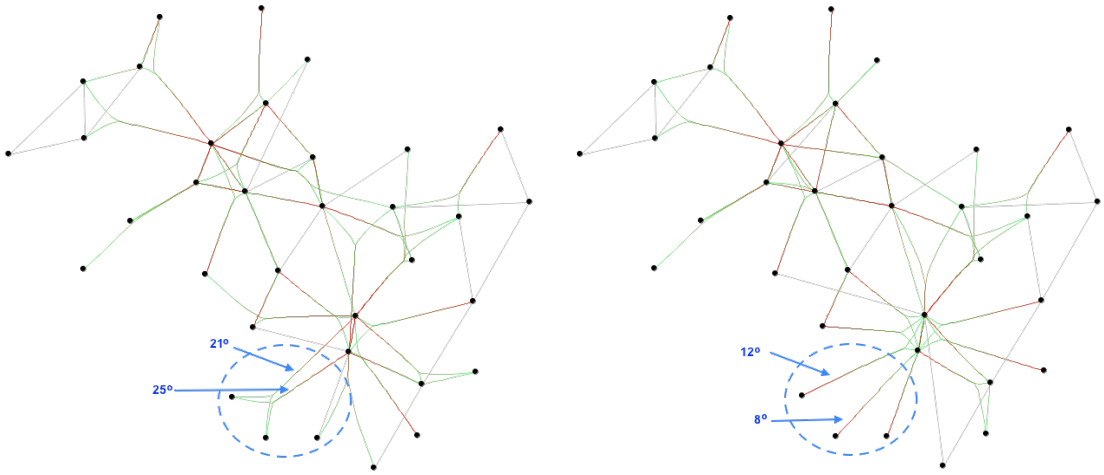
Obter soluções ótimas para grafos com muitas arestas demanda um alto custo de processamento, pois o espaço de busca é grande. Qualquer modificação com a inserção ou retirada de uma aresta de um feixe causa impacto no valor de aptidão, pois o tamanho da aresta também influencia a função objetivo.

Com relação às características dos feixes, como esperado, as soluções geradas possuem feixes mais homogêneos com relação ao ângulo e tamanho das arestas. Arestas com uma diferença grande de tamanho, e que, ainda assim, foram unidas na resolução do problema *ABEB*, no geral, não aparecem unidas na melhor solução do problema *CBEB*. Como a compatibilidade escalar também faz parte dos objetivos do problema, o algoritmo busca formar feixes com arestas o mais compatíveis segundo esse critério.

Comparando-se as soluções do problema *ABEB* (com 82 feixes) e *CBEB* (com 97 feixes) para o grafo *LesMiserables* (G6), apesar de o número de feixes ter aumentado de um problema para outro, a solução do problema *CBEB* possui feixes com arestas mais similares em tamanho do que a solução do primeiro problema. A Figura 5.6 mostra o desenho das duas soluções. No destaque da figura, verifica-se que alguns feixes presentes na solução para *ABEB* foram divididos na solução do problema *CBEB*, e novos foram criados. Por exemplo, a aresta *A* passou a formar um feixe sozinha, e a aresta *B* foi inserida em um feixe com arestas mais compatíveis.

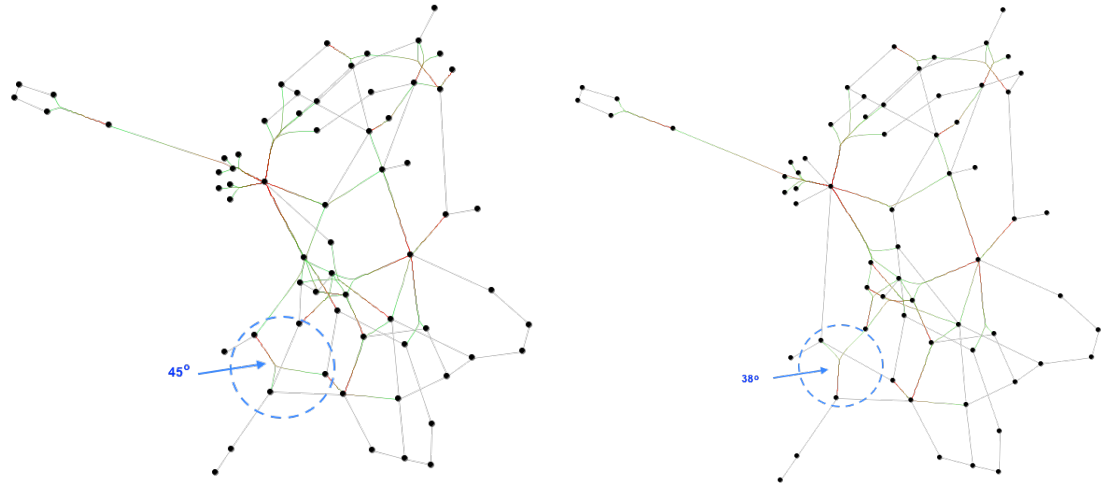
Um outro ponto interessante é a relação ângulo $\times$ tempo. Ao contrário do que aconteceu para o problema *ABEB*, o ângulo maior não simplificou o problema. Para a maioria dos grafos, gastou-se mais tempo para resolver o problema para os ângulos $\alpha = 45°$ e $\alpha = 70°$.

Com relação a dispersão da aptidão em torno da média, os valores do desvio padrão são altos, o que indica que as ocorrências não foram tão uniformes como no teste



(a) (b)

Figura 5.4: *Comparação dos resultados do problema CBEB com ângulo como objetivo para o grafo Zachary Club (G2) com* $\alpha = 30°$*: (a) solução ABEB com 37 feixes e (b) solução CBEB com 37 feixes.*



(a) (b)

Figura 5.5: *Comparação dos resultados do problema CBEB com ângulo como objetivo para o grafo PlanarGD2015 (G3) com* $\alpha = 45°$*: (a) solução ABEB com 69 feixes e (b) solução CBEB com 71 feixes.*

do problema *ABEB*. Esse comportamento é evidenciado pelo fato de que, em quase todos os testes, não foi encontrada a mesma solução com a mesma qualidade mais de uma vez (coluna quatro da Tabela 5.3).

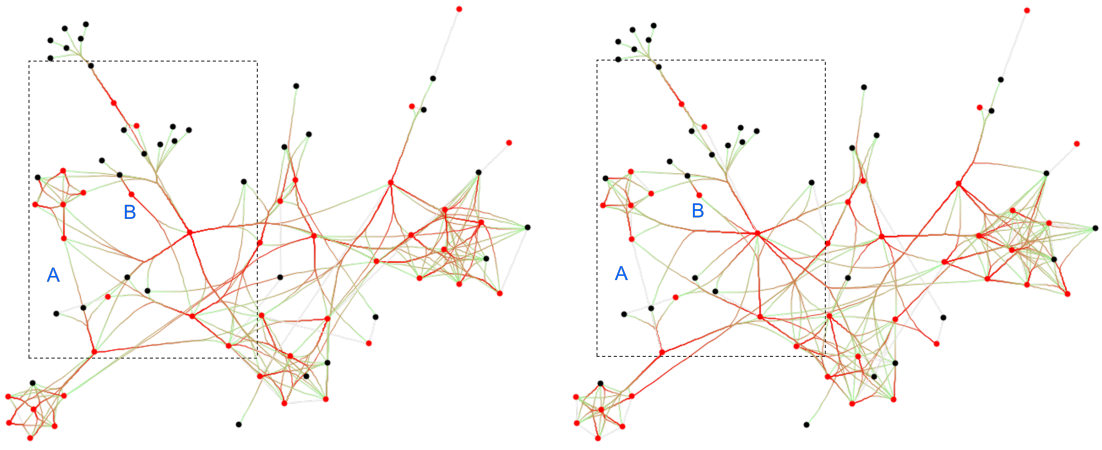
A Tabela 5.4 mostra os resultados do terceiro experimento, com penalidade $p_e = -3$. Os símbolos $\uparrow$, $\downarrow$ e $=$ significam que o valor foi superior, inferior ou igual, respectivamente, ao encontrado no segundo experimento e registrado na Tabela 5.3.

O cenário ideal é que o número de feixes diminua e o nível de compatibilidade aumente. No entanto, analisando-se os resultados da Tabela 5.4, verifica-se que para o

Tabela 5.3: *Resultados do experimento 2 para o problema CBEB, média de 100 execuções independentes, $u = 150$, $p_c = 0,98$, $p_m = 0,4$, $w_1 = 0,2$, $w_2 = 0,8$ e $T_s = 0,890$, $p_e = -1$.*

Grafo	Arestas	α(°)	Feixes da melhor solução	Média da compatibilidade	Média de feixes	Desvio padrão de feixes	Erro padrão de feixes	Média da aptidão	Desvio padrão da aptidão	Erro padrão da aptidão	Média tempo (sec.)
G1	28	30	20 (3)	9,48	20,01	0,001	0,0110	1,9354	0,02551	0,002551	2
		45	16 (2)	13,27	18,17	0,7661	0,0766	2,6990	0,29839	0,029839	2
		70	16 (2)	20,22	16,82	0,3861	0,0386	4,0925	0,24830	0,024830	2
G2	78	30	56 (2)	22,69	58,49	1,0777	0,1078	4,5518	0,24592	0,024592	11
		45	48 (1)	35,87	50,72	1,5510	0,1551	7,1901	0,49003	0,049003	13
		70	36 (1)	69,97	39,10	1,6787	0,1679	14,0145	0,86055	0,086055	15
G3	101	30	81 (1)	23,23	83,86	1,2872	0,1287	4,6546	0,33595	0,033595	12
		45	76 (1)	34,94	81,10	1,5407	0,1541	6,9985	0,80816	0,080816	13
		70	65 (1)	84,31	69,05	1,6598	0,1660	16,8739	0,61844	0,061844	18
G4	160	30	129 (1)	29,49	132,78	1,7672	0,1767	5,9044	0,40687	0,040687	31
		45	112 (1)	50,48	119,24	2,5470	0,2547	10,1020	0,66873	0,066873	39
		70	89 (1)	108,68	94,89	3,1811	0,3181	21,7440	1,45149	0,145149	47
G5	161	30	76 (1)	136,71	83,90	2,9525	0,2952	27,305	2,18483	0,218483	33
		45	60 (1)	218,55	69,79	3,3160	0,3316	43,7209	4,24025	0,424025	36
		70	50 (1)	387,42	52,88	2,3324	0,2332	77,5001	5,80255	0,580255	35
G6	254	30	174 (1)	121,04	180,17	4,6451	0,4645	24,2121	2,08592	0,208592	103
		45	137 (1)	206,14	147,32	5,1402	0,5140	41,2327	3,00882	0,300882	111
		70	97 (1)	426,76	101,97	3,8283	0,3828	85,3606	4,74743	0,474743	101
G7	401	30	305 (1)	170,56	312,62	4,7028	0,4703	34,1143	1,68656	0,168656	300
		45	260 (1)	266,62	270,62	5,4952	0,5495	53,3276	2,48894	0,248894	315
		70	209 (1)	501,12	213,89	5,4604	0,5460	100,2274	4,68943	0,468943	363
G8	425	30	303 (1)	148,28	314,20	5,0911	0,5091	29,6593	1,8706	0,187060	209
		45	267 (1)	241,85	274,03	5,8972	0,5897	48,3727	3,1208	0,312085	240
		70	208 (1)	472,18	213,04	5,9236	0,5924	94,4395	6,0949	0,609494	273
G9	709	30	452 (1)	410,16	464,50	8,9561	0,8956	82,0344	4,58910	0,458910	684
		45	379 (1)	663,31	395,09	6,3343	0,6334	132,6634	4,62701	0,462701	834
		70	312 (1)	1108,79	318,49	7,9500	0,7950	221,7608	9,51946	0,951946	828
G10*	1297	30	530 (1)	2152,49	536,33	6,7788	1,7503	430,4998	15,50054	4,002223	19184
		45	428 (1)	3249,19	437,40	6,3110	1,6295	649,8400	22,67482	5,854614	11110
		70	318 (1)	5774,74	327,73	6,3298	1,6344	1154,9496	57,56356	14,862847	6859

**Para o grafo G10 foram feitas apenas 15 execuções independentes para cada ângulo.*



(a) (b)

Figura 5.6: *Comparação dos resultados dos problemas ABEB e CBEB para o grafo LesMiserables (G6) com $\alpha = 70°$ e penalidade $p_e = -1$: (a) solução ABEB com 82 feixes e (b) solução CBEB com 97 feixes.*

Tabela 5.4: *Resultados do experimento 3 para o problema CBEB, média de 100 execuções independentes,* $u = 150$*,* $p_c = 0,98$*,* $p_m = 0,4$*,* $w_1 = 0,2$*,* $w_2 = 0,8$ *e* $T_s = 0,890$*,* $p_e = -3$*.*

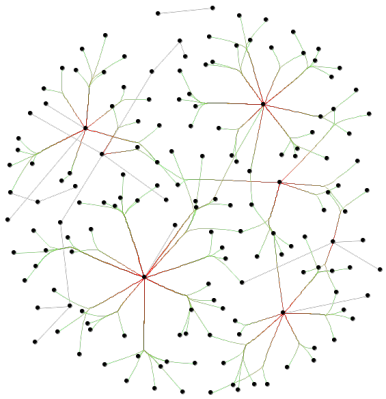
Grafo	Arestas	α(°)	Feixes da melhor solução	Média da compatibilidade	Média de feixes	Desvio padrão de feixes	Erro padrão de feixes	Média da aptidão	Desvio padrão da aptidão	Erro padrão da aptidão	Média tempo (sec.)
G1	28	30	= 20 (4)	↑ 9,49	20,00	0,0000	0,0000	↑ 1,9373	0,01391	0,001391	↓ 1
		45	↑ 17 (1)	↓ 12,27	18,68	0,5840	0,0584	↓ 2,4959	0,22399	0,022399	↓ 1
		70	= 16 (4)	↓ 19,97	16,89	0,3145	0,0314	↓ 4,0422	0,18282	0,018282	= 2
G2	78	30	↑ 57 (1)	↓ 22,68	58,51	0,9898	0,0990	↓ 4,5500	0,24224	0,024224	↓ 10
		45	= 48 (1)	↓ 35,61	50,77	1,7047	0,1705	↓ 7,1381	0,50292	0,050292	↓ 12
		70	↑ 37 (1)	↓ 69,32	39,28	1,6148	0,1615	↓ 13,8847	0,87500	0,087500	= 15
G3	101	30	= 81 (1)	↑ 23,25	83,78	1,1941	0,1194	↓ 4,6599	0,31141	0,031141	↑13
		45	↑ 77 (1)	↓ 34,69	80,82	1,6104	0,1610	↓ 6,9487	0,74041	0,074041	↑ 14
		70	= 65 (1)	↑ 83,69	69,48	1,8006	0,1801	↓ 16,7490	0,74374	0,074374	↓ 16
G4	160	30	= 129 (1)	↓ 29,29	132,97	1,8393	0,1839	↓ 5,8642	0,41292	0,041292	↑ 32
		45	↑ 116 (1)	↓ 50,14	119,63	2,0432	0,2043	↓ 10,0346	0,61277	0,061277	↑ 40
		70	= 89 (1)	↑ 109,23	94,80	3,0715	0,3072	↓ 21,8537	1,28648	0,128648	↓ 42
G5	161	30	↑ 81 (1)	↓ 136,60	84,17	3,0585	0,3059	↓ 27,3295	2,34040	0,234040	↓ 30
		45	↑ 64 (1)	↑ 220,81	69,43	3,0460	0,3046	↓ 44,1737	3,84387	0,384387	↓ 29
		70	↓ 46 (1)	↑ 394,00	52,65	2,7902	0,2790	↑ 78,8155	7,32994	0,732994	↓ 29
G6	254	30	↑ 177 (1)	↑ 123,45	179,82	4,0586	0,4059	↑ 24,6949	2,10018	0,210018	↓ 87
		45	↑ 140 (1)	↑ 209,01	146,98	4,6034	0,4603	↑ 41,8069	2,83851	0,283851	↓ 99
		70	↑ 99 (1)	↑ 427,81	101,65	4,5380	0,4538	↑ 85,5705	5,86285	0,586285	↓ 101
G7	401	30	↓ 301 (1)	↑ 171,23	312,44	4,7019	0,4702	↑ 34,2731	1,67959	0,167959	↓ 279
		45	↓ 258 (1)	↑ 267,53	270,75	4,7148	0,4715	↑ 53,5093	2,09169	0,209169	↑ 622
		70	↓ 205 (1)	↓ 498,09	214,47	5,4707	0,5471	↓ 99,6217	4,35717	0,435717	↓ 330
G8	425	30	↓ 302 (1)	↑ 151,22	312,71	5,6627	0,5663	↑ 30,2469	2,11784	0,211784	↑ 227
		45	↓ 255 (1)	↑ 242,67	273,18	5,6718	0,5672	↑ 48,5360	2,92520	0,292520	↑ 255
		70	↓ 193 (1)	↓ 471,57	212,85	6,3887	0,6389	↑ 94,3176	6,42094	0,642094	↑ 297
G9	709	30	↓ 451 (1)	↓ 408,47	465,11	8,1015	0,8101	↓ 81,6960	3,86121	0,386121	↓ 627
		45	↑ 381 (1)	↓ 656,77	397,44	7,3599	0,7360	↓ 131,3566	5,25592	0,525592	↓ 650
		70	↓ 310 (1)	↓ 1105,19	318,88	8,3319	0,8332	↓ 221,0402	9,02951	0,902951	↓ 690

problema *CBEB* e com penalidade $p_e = -3$, a média do número de feixes e o número de feixes da melhor solução aumentaram para a maioria dos grafos pequenos, quando comparado com a configuração de penalidade $p_e = -1$ (seta para cima), enquanto a média de compatibilidade diminuiu (seta para baixo). A performance da penalidade $p_e = -3$ foi melhor para os grafos maiores, pois a tendência foi diminuir a média do número de feixes e o número de feixes da melhor solução (seta para baixo), sendo que a média de compatibilidade da maioria aumentou (seta para cima).

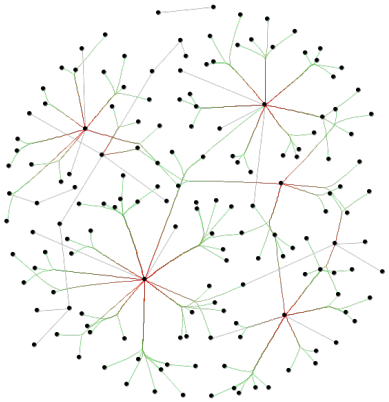
Mas, em alguns grafos, como o G6, apesar de as soluções do experimento dois terem obtido um número de feixes maior, o valor de compatibilidade também cresceu. Como o problema *CBEB* é multiobjetivo, nem sempre a solução com menor número de feixes será a ótima. Isso depende dos pesos escolhidos na função objetivo e dos valores de compatibilidade.

Comparando-se a média da aptidão (que é o valor usado no processo de otimização) no experimento dois, percebe-se que o seu valor cresceu nos grafos com uma quantidade maior de arestas. No entanto, o desvio padrão é alto para alguns grafos, como o G8, G9 e o G10, o que demonstra que as soluções ficaram distantes da média da aptidão.

No geral, o terceiro experimento foi executado em um tempo ligeiramente menor em comparação com o primeiro, provavelmente devido ao uso de uma penalidade maior.

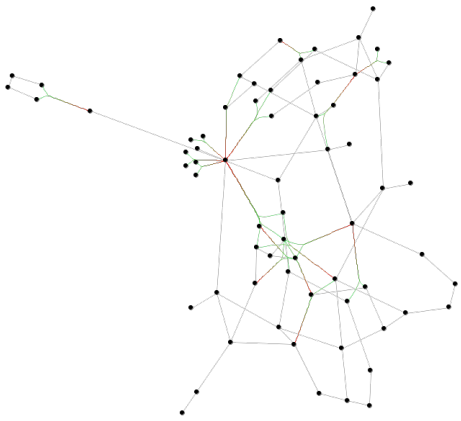


$p_e = -1$, 60 feixes, compat. 218,55

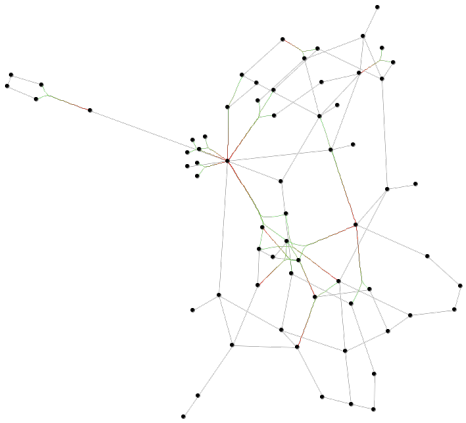


G5, $p_e = -3$, 64 feixes, compat. 220,81

Figura 5.7: *Comparação de resultados para diferentes valores de penalidade para o grafo MovieLens (G5),* $\alpha = 45°$.



$p_e = -1$, 81 feixes, compat. 23,23



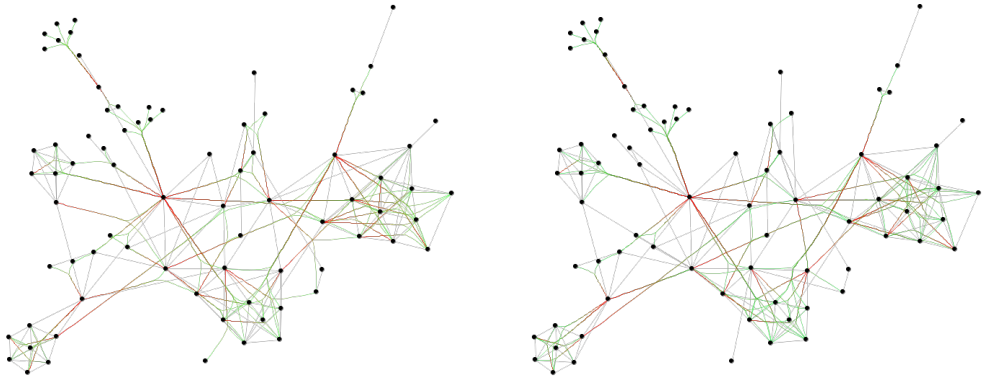
$p_e = -3$, 81 feixes, compat. 23,25

Figura 5.8: *Comparação de resultados para diferentes valores de penalidade para o grafo Planar (G3),* $\alpha = 30°$.

Em um algoritmo evolucionário essa característica faz com que o método não perca tempo avaliando soluções indesejadas. Quando soluções inviáveis são penalizadas em sua aptidão, isso faz com que a busca pela melhor solução seja levada de volta à uma região factível, fazendo com que soluções de baixa qualidade sejam descartadas mais rapidamente, o que melhora o tempo de execução, principalmente para as instâncias maiores. Como consequência, a ampliação da penalidade gerou resultados em um tempo menor, com uma qualidade superior para instâncias maiores, sendo o efeito contrário em instâncias menores.

As Figuras 5.7, 5.8 e 5.9 comparam algumas soluções usando penalidades diferentes, que obtiveram melhor compatibilidade no experimento com $p_e = -3$.

Verifica-se que as soluções não são muito diferentes visualmente, mas os grafos para os quais $p_e = -3$, tendem a ser um pouco mais simétricos com relação ao tamanho das arestas internas do feixe, e com relação ao ângulo total de cada feixe, mas isso não foi uma regra.



$p_e = -1$, 174 feixes, compat. 121,04 $p_e = -3$, 177 feixes, compat. 123,45

Figura 5.9: *Comparação de resultados para diferentes valores de penalidade para o grafo LesMiserables (G6),* $\alpha = 30°$.

Apesar do efeito positivo do aumento da penalidade no tempo de execução, o valor da média desse tempo ainda foi muito elevado para os grafos de maior dimensão. Por exemplo, o grafo *USAirline* (G10) não foi testado, devido ao tempo de processamento excessivo. Assim como no primeiro experimento, o sistema gasta em torno de cinco horas para gerar a resposta de cada teste individual.

Comportamento semelhante ocorre na relação do tempo de execução entre os problemas *ABEB* e *CBEB*. A abordagem *EEB* levou mais tempo para convergir para a melhor solução no segundo problema. Como na maioria dos algoritmos de busca baseada em população, o tempo de execução do *EEB* foi influenciado pelo tamanho da população, pelo número de gerações, pelo tamanho do grafo e outros parâmetros. Os parâmetros evolucionários para o problema *CBEB* foram ajustados na tentativa de encontrar uma solução próxima da ótima, o que, de certa forma, elevou o tempo de processamento nos grafos maiores.

As Figuras 5.10 até 5.18 ilustram, por meio de gráficos de barra, a relação número de feixes × compatibilidade × tempo de execução dos resultados dos experimentos dois e três para cada grafo testado.

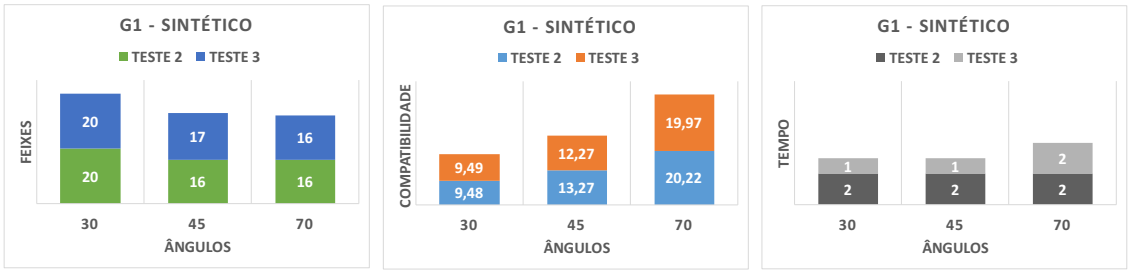


Figura 5.10: *Relação feixe × compatibilidade × tempo dos resultados do grafo Sintético (G1) para o problema CBEB.*

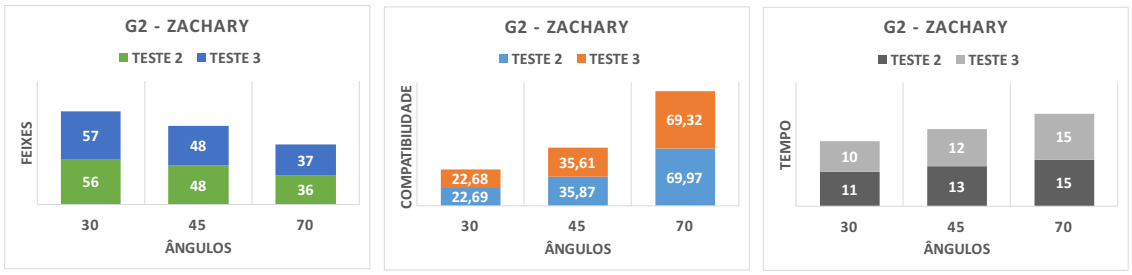


Figura 5.11: *Relação feixe × compatibilidade × tempo dos resultados do grafo Zachary Club (G2) para o problema CBEB.*

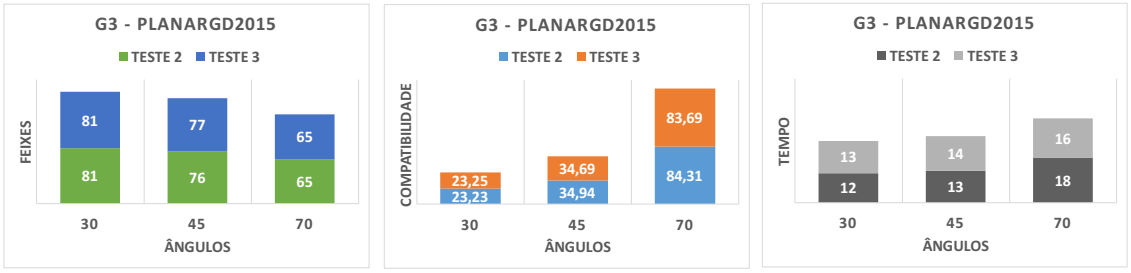


Figura 5.12: *Relação feixe × compatibilidade × tempo dos resultados do grafo PlanarGD2015 (G3) para o problema CBEB.*

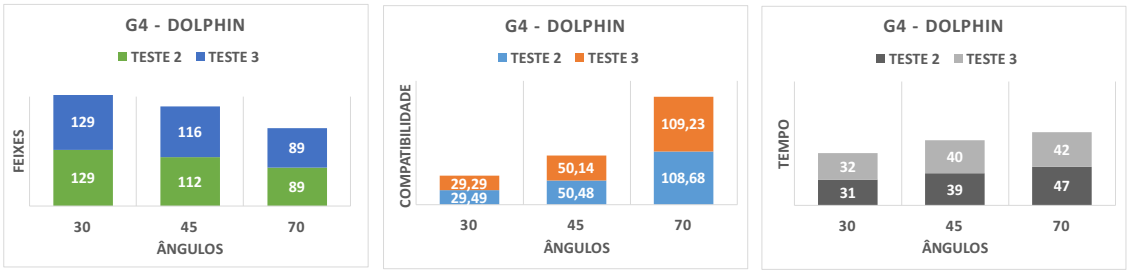


Figura 5.13: *Relação feixe × compatibilidade × tempo dos resultados do grafo Dolphin (G4) para o problema CBEB.*

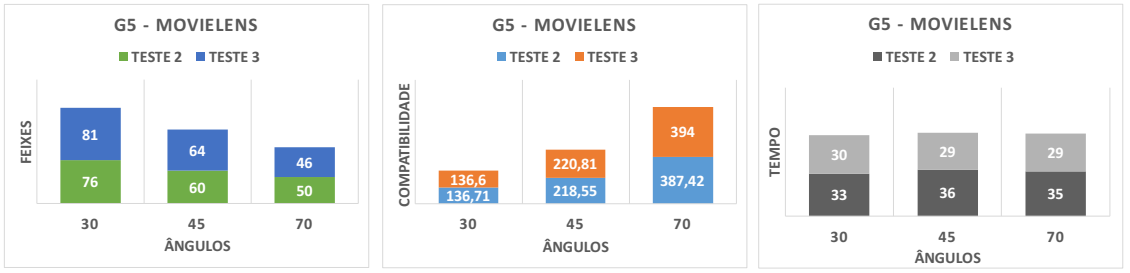


Figura 5.14: *Relação feixe × compatibilidade × tempo dos resultados do grafo MovieLens (G5) para o problema CBEB.*

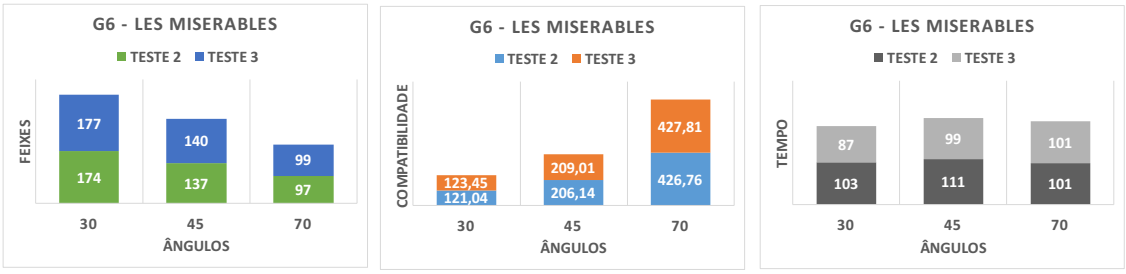


Figura 5.15: *Relação feixe × compatibilidade × tempo dos resultados do grafo Les Miserables (G6) para o problema CBEB.*

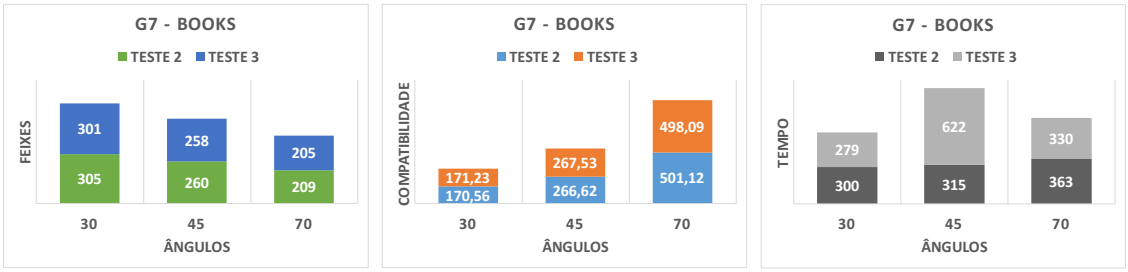


Figura 5.16: *Relação feixe × compatibilidade × tempo dos resultados do grafo Book USPolitics (G7) para o problema CBEB.*

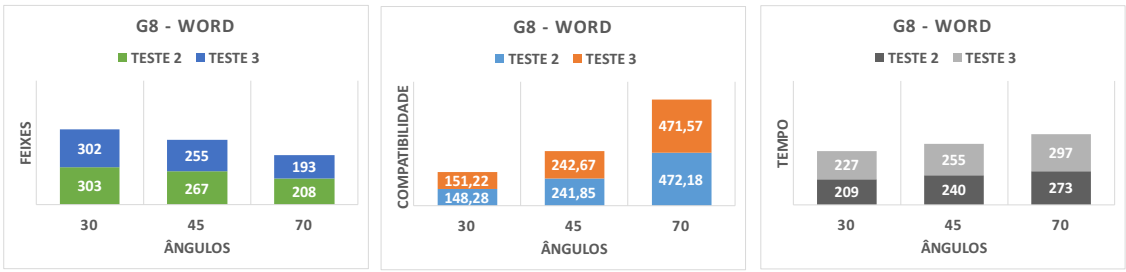


Figura 5.17: *Relação feixe × compatibilidade × tempo dos resultados do grafo Word Adjacences (G8) para o problema CBEB.*

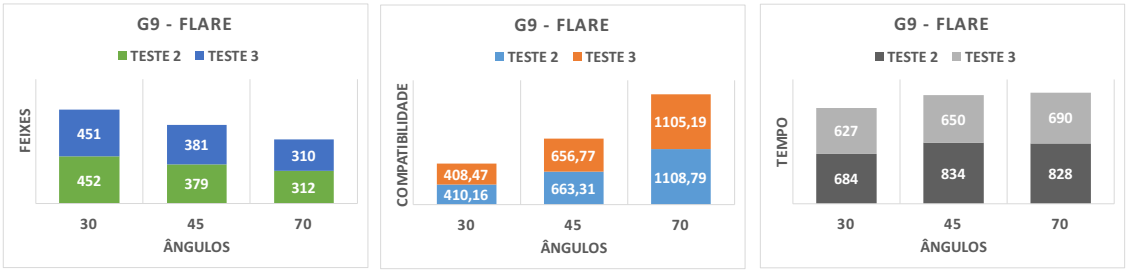


Figura 5.18: *Relação feixe × compatibilidade × tempo dos resultados do grafo Flare (G9) para o problema CBEB.*

Os resultados numéricos não permitem obter outras conclusões a respeito de todos os critérios estéticos propostos na Seção 3.4. No entanto, a abordagem *EEB* tentou satisfazer, de forma implícita ou explícita, tais critérios. Por exemplo, a minimização do número total de feixes, a maximização da compatibilidade entre as arestas e a maximização do número de arestas por feixes são abordados, explicitamente, pela função objetivo do problema *CBEB* (e parcialmente por A*BEB*). Além disso, de forma implícita, a união de arestas adjacentes tende a minimizar a ambiguidade no rastreamento das arestas. A simetria do feixe foi alcançada pelo uso de um algoritmo de renderização guiado por força, e a minimização do número total de cruzamentos entre arestas e feixes foi conduzido usando-se desenho parcial de arestas.

5.5.3 Análise de Convergência

Nesta seção, será analisado o comportamento de convergência do experimento que gerou a melhor solução (dentre os 100 testes controlados) de cada grafo testado, para os ângulos $\alpha = 30°, 45°, 70°$ e penalidades $p_e = -1$ e $p_e = -3$ (experimentos dois e três, respectivamente). A Tabela 5.5 resume o número de gerações necessárias para produzir a melhor solução para cada grafo, entre parênteses é indicada a média de gerações de todas as execuções.

A Figura 5.19 ilustra os dados da Tabela 5.5. Os dois gráficos superiores dizem respeito ao teste com $p_e = -1$, e os grafos inferiores, ao experimento com $p_e = -3$. A geração de convergência da melhor solução de cada grafo está à esquerda e a média de gerações de convergência (de todos os testes), à direita.

O padrão de convergência é diferente do registrado para o problema *ABEB*. Em média, os testes do problema *CBEB* foram ligeiramente mais rápidos para ângulos menores nos dois testes, sendo que os grafos mais complexos levaram um número grande de gerações até a convergência, principalmente para $p_e = -3$.

Uma outra diferença fica por conta das gerações dos grafos individualmente. O número de gerações para convergência é quase o dobro do registrado para o problema *ABEB*, na maioria dos grafos. Até os grafos pequenos, como o *Sintético* (G1), que para o problema *ABEB* obteve resposta quase instantânea, para o problema *CBEB* levou mais tempo para convergir. Isso demonstra como o problema *CBEB* é mais complexo do que o anterior. A inserção de um novo objetivo tornou o problema mais difícil de ser resolvido, com um espaço de busca maior para ser analisado.

No entanto, a evolução da média das gerações mostra uma tendência de crescimento de gerações proporcional ao tamanho do grafo, apesar de algumas variações, principalmente para os grafos G5 e G8. Essa variação também é registrada no gráfico de evolução da convergência da melhor solução. O grafo G5 contêm poucos ciclos, enquanto

Tabela 5.5: *Comparação da quantidade de gerações da melhor execução do EEB para o problema CBEB.*

Grafo	Penalidade $p_e = -1$			Penalidade $p_e = -3$		
	30°	**45°**	**70°**	**30°**	**45°**	**70°**
G1	48 (69,60)	57 (88,65)	21 (132,16)	31 (78,64)	246 (78,80)	57 (122,88)
G2	1485 (759,70)	2130 (1091,90)	1693 (1403,08)	1562 (759,76)	1165 (1046,49)	1477 (1425,50)
G3	1666 (561,83)	803 (643,06)	2033 (1097,40)	609 (536,77)	939 (710,63)	2361 (956,59)
G4	1815 (1244,82)	3161 (1755,47)	2686 (2314,24)	1894 (1274,67)	1759 (1815,96)	4004 (2427,21)
G5	2967 (1581,09)	2122 (1798,44)	2733 (1817,83)	2987 (1712,28)	1637 (1760,24)	3034(1987,40)
G6	4331 (2816,48)	6831 (3665,91)	4680 (3718,29)	4266 (2981,42)	5114 (3765,14)	6201 (3989,29)
G7	6921 (4928,58)	6881 (5823,50)	6790 (6694,77)	8119 (5205,95)	9058 (6102,01)	7868 (6739,52)
G8	5156 (4070,89)	6802 (5100,85)	6494 (6250,95)	6079 (4432,47)	7635 (4993,88)	9583 (6378,84)
G9	11854 (7096,92)	8058 (8458,49)	8572(9056,92)	11740 (7250,37)	11698 (7956,10)	10834(9420,66)
G10	16475 (15537,86)	16450 (15896,00)	16415 (16340,67)	-	-	-

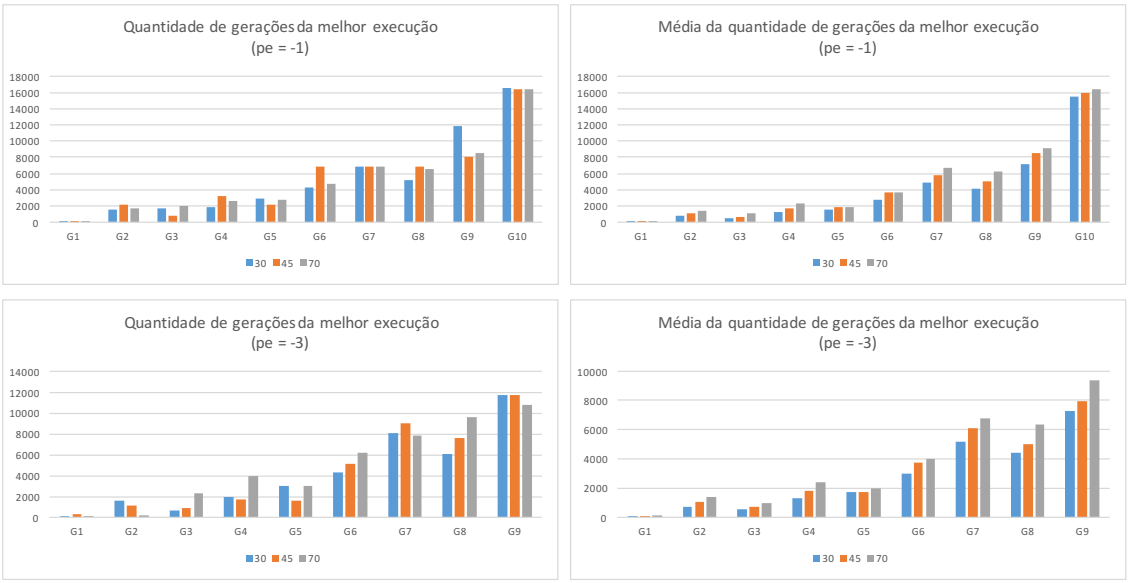


Figura 5.19: *Comparação da quantidade de gerações da melhor execução do EEB para o problema CBEB.*

o grafo G8 possui um desenho de grafo muito compacto, com várias conexões por vértices com ângulos próximos.

Comparando-se o tempo médio de resposta, a geração de convergência e os resultados em termos de número de feixes e valor de compatibilidade, acredita-se que talvez ainda exista espaço para melhorar a qualidade das soluções nos grafos maiores. Essa melhoria talvez possa ser obtida elevando-se o tempo de espera por uma alteração na melhor solução (nos testes foi usado um limite de 500 gerações). No entanto, essa alteração não foi testada, pois o tempo de execução seria alto.

As Figuras 5.20 e 5.21 correspondem aos gráficos de evolução da qualidade ao longo das gerações do teste que gerou a melhor solução para os grafos MovieLens e USAirline, respectivamente. Os gráficos dos demais grafos encontram-se no Apêndice C. Analisando-se esses grafos, percebe-se que a evolução da qualidade da melhor solução foi condizente com os padrões de um algoritmo evolucionário.

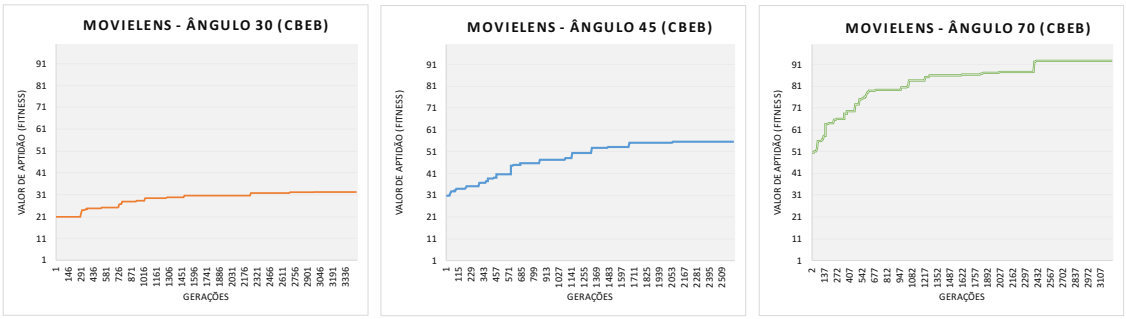


Figura 5.20: *Convergência do grafo MovieLens (G5 - 161 arestas) para o problema CBEB com penalidade $p_e = -1$. Soluções com 76 ($\alpha = 30°$), 60 ($\alpha = 45°$) e 50 ($\alpha = 70°$) feixes.*

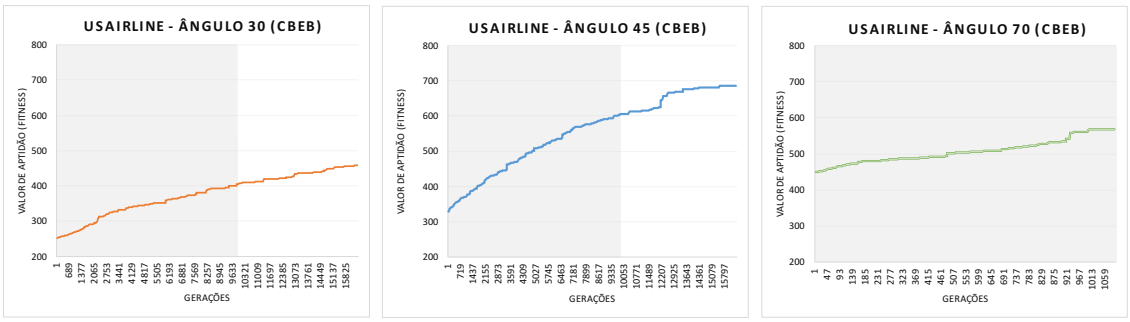


Figura 5.21: *Convergência do grafo USAirline (G10 - 1297 arestas) para o problema CBEB com penalidade $p_e = -1$. Soluções com 530 ($\alpha = 30°$), 428 ($\alpha = 45°$) e 318 ($\alpha = 70°$) feixes.*

5.6 Comparação do Problema CBEB com as Abordagens Clássicas

A comparação dos resultados dos problemas *CBEB* (*ABEB*) com as abordagens clássicas para união de arestas não é direta, sem antes entender o problema de que essas técnicas tratam. Como já exposto, boa parte dos algoritmos clássicos não são baseados em um problema de otimização combinatória explícito e bem definido. Um exemplo dessa afirmação pode ser observado na Figura 5.22 (a), o método clássico *Divide Edge Bundling* (DEB) produz um desenho significativamente diferente do criado pelo algoritmo *EEB* para o problema *CBEB*. Uma vez que o DEB não monta feixes com somente arestas adjacentes, e não possui o objetivo de mostrar conexão em nível de vértice, é difícil estabelecer, de forma geral, quais vértices estão conectados diretamente a outros.

Por outro lado, abordagens como *stub bundling* [89] e *sideknot* [92] (Figuras 5.22 (b) e (c)) são mais similares aos resultados obtidos pelo *framework EEB* aplicado ao problema *CBEB*. Essas técnicas agrupam as arestas adjacentes com a finalidade de

destacar a indicação de direção nos extremos das arestas, as conexões em nível de vértices e facilitar a rastreabilidade das arestas de ponta a ponta. Entretanto, esses algoritmos não foram desenvolvidos como métodos baseado em problema, uma vez que não existe uma definição formal matemática indicando qual o problema que foi investigado, quais suas restrições e objetivos.

Na tentativa de formalizar qual o problema tratado pela técnica *stub bundling*, poder-se-ia inferir que a abordagem tenta encontrar a decomposição contendo somente arestas adjacentes e respeitando as seguintes restrições, citando diretamente Nocaj e Brandes [89]: "*o ângulo entre duas arestas-metade (do inglês half-edges) em um feixe deve ser no máximo* α*, e o ângulo entre as arestas consecutivas no feixe tem no máximo* γ". Este é um problema similar ao *ABEB*, exceto pelo fato de que as arestas são do tipo *half-edges*, isto é, elas podem ser compartilhadas por dois feixes, e, ainda, não está claro se a minimização do número de feixes de arestas foi objetivo deste problema.

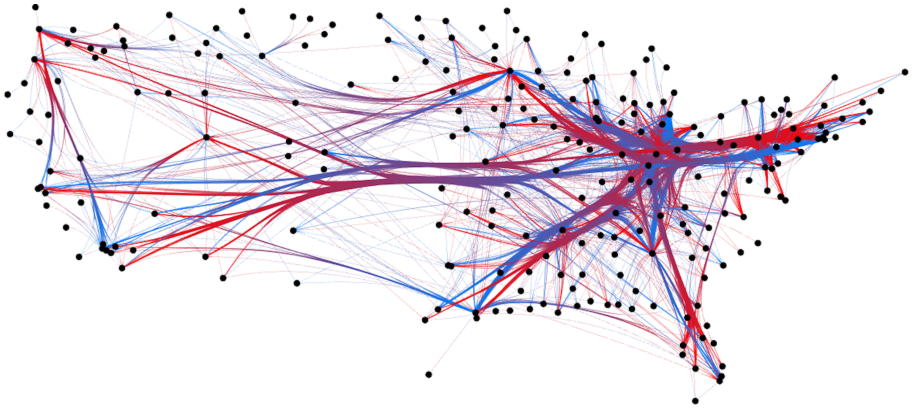
Como resultado, a comparação quantitativa efetiva entre *EEB*, *stub bundling* e *sideknot* é difícil de ser feita por não existirem informações sobre a quantidade de feixes que as últimas duas geram em suas soluções e o nível de compatibilidade pré estabelecido. A comparação visual das Figuras 5.22 (b - d) revela a eficiência de *EEB* para o problema *CBEB*, com *threshold* $\alpha = 70^{\circ}$. O algoritmo gerou 241 feixes com mais de uma arestas e 77 feixes contendo apenas uma única aresta em sua composição (318 feixes no total). Analisando-se a Figura 5.22, fica claro que o agrupamento das arestas feito pela abordagem *EEB* difere bastante das soluções geradas por *stub bundling* e *sideknot*.

5.7 Considerações Finais

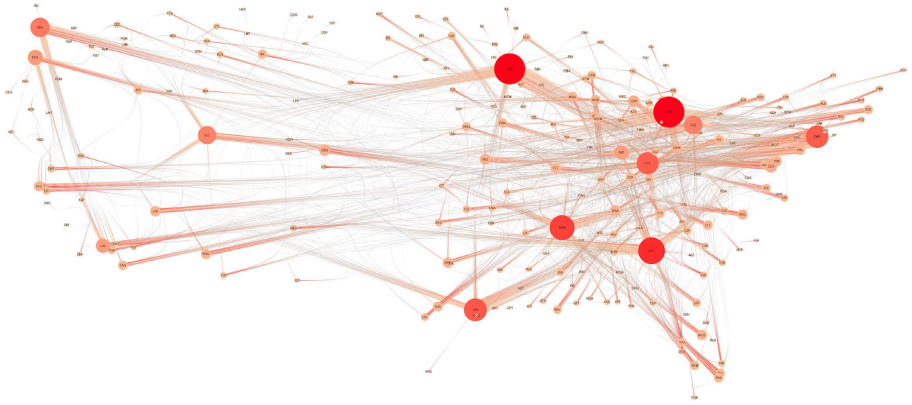
Neste capítulo, foi apresentado um problema para união de arestas, cujo objetivo é minimizar o número de feixes e maximizar a compatibilidade de arestas. Comparando-se com o problema *ABEB*, as restrições união de arestas adjacentes e criação de feixes disjuntos foram mantidas, entretanto, a restrição angular se tornou objetivo do problema, sendo inserido no cálculo da função de compatibilidade. Esse problema foi identificado como união explícita de aresta em feixes centralizados com multicritérios (*CBEB*).

Para resolução do problema *CBEB*, o *framework EEB* foi adaptado, quase toda a sua estrutura foi mantida, com modificações significativas apenas no cálculo da compatibilidade geral do grafo. Foi implementada uma política de penalização das soluções que possuíssem feixes que violassem o limite desejável para a compatibilidade.

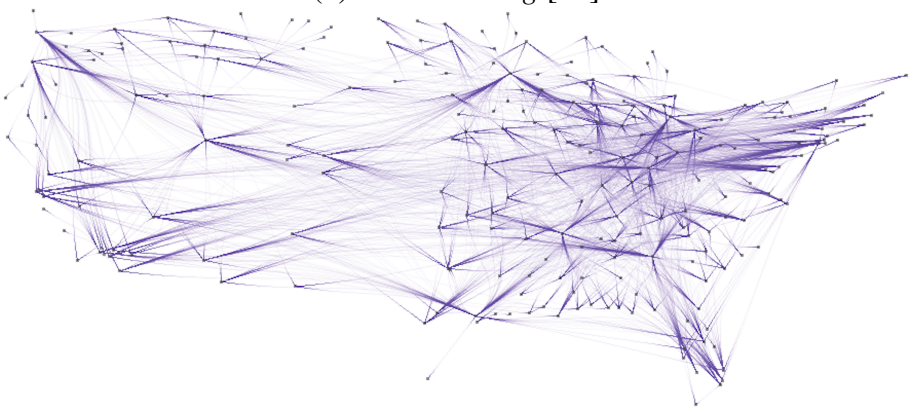
Foram executados, ainda, dois experimentos com valores diferentes de penalização. O algoritmo se mostrou eficiente, gerando soluções condizentes com os parâmetros de controle pré-configurados. De uma forma geral, os experimentos sugerem que a principal desvantagem do *EEB* é seu tempo de execução para grafos de grande dimensão.



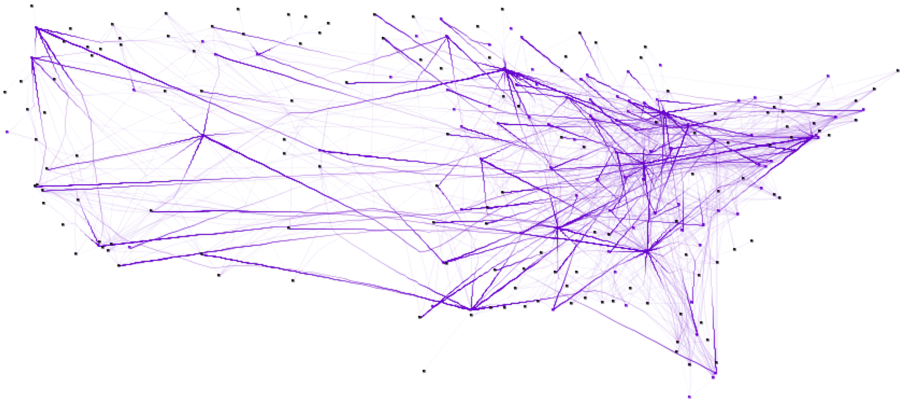
(a) *Divided Edge Bundling*



(b) *Stub Bundling* [89]



(c) *SideKnot* [89]



(d) *EEB* com arestas parciais, $\alpha = 70°$

Figura 5.22: *Comparação de soluções para o grafo USAirline (G10).*

CAPÍTULO 6

União Explícita de Arestas em Feixes Descentralizados

6.1 Descrição do Problema

A maioria das abordagens para união de arestas produzem feixes que são do tipo descentralizado. Os métodos costumam unir as arestas pelo seu meio e não nos extremos. No entanto, os problemas tratados nos capítulos anteriores foram definidos para situações em que o resultado é um desenho de grafo com feixes centralizados. Este capítulo apresenta um problema mais geral que relaxa a restrição de adjacência, permitindo a geração de feixes com arestas múltiplas. O problema foi denominado **união explícita de arestas em feixes descentralizados**, ou simplesmente, *GBEB* (sigla em inglês do termo *General-Based Edge Bundling*). A definição deste problema é apresentada a seguir.

6.2 Modelagem Matemática

Problema 6.1 *Seja um grafo simples $G = (V,E)$, um desenho D de G com arestas representadas por linhas retas e vértices em posições fixas. Seja $C(p,q)$ a medida de compatibilidade entre um par de arestas $p,q \in E$ com base em D. O problema de união explicita de arestas em feixes descentralizados (GBEB) é determinar uma decomposição S de E em subconjuntos disjuntos $E_1,E_2,\ldots,E_n$, com $E = \cup_{i=1}^{n} E_i$ e $E_i \cap E_j \neq \emptyset$, para $1 \leq i,j \leq |E|$, $i \neq j$, que maximize a compatibilidade total do grafo $C_G = \sum_{i=1}^{n} C_{E_i}$, onde $C_{E_i} = \sum_{p,q \in E_i} C(p,q)$ é a compatibilidade de um feixe E_i, e minimize n*

Observe-se que no problema *GBEB*, a restrição de feixes disjuntos ainda foi mantida, sendo os objetivos a maximização do valor de compatibilidade total do grafo e a minimização da quantidade de feixes. Identificando-se os elementos da formulação *EB* neste problema, tem-se que

- G é um grafo simples;
- D é um desenho de pontos e linhas com vértices fixos;

- os objetivos são:
  - F_1: encontrar um conjunto S de E que seja o menor possível ($n \leq k$); e
  - F_2: encontrar um conjunto S de E que maximize a compatibilidade geral C_G do grafo.
- a restrição P assegure que os conjuntos E_i são disjuntos.

6.3 Função de Compatibilidade

Para os problemas *ABEB* e *CBEB*, como as arestas eram adjacentes, foi usado o ângulo em graus normalizado entre um limite de 0 (para 180°) e 1 (para 0°), e, dessa forma, não foi necessário o uso das medidas de visibilidade e posição. Na resolução do problema *GBEB*, este tipo de cálculo não se mostrou efetivo para determinar um nível de compatibilidade aceitável, pois, como podem ser unidas quaisquer arestas do conjunto E, e a complexidade de seleção de arestas compatíveis aumentou.

Dessa forma, optou-se por usar todas as medidas geométricas propostas por Holten e Wijk [43] no cálculo da compatibilidade total: **angular** $C_a(p,q)$, **escalar** $C_s(p,q)$, de **posição** $C_p(p,q)$ e **visibilidade** $C_v(p,q)$, todas como valores entre $[0,1]$. As fórmulas estão descritas na Seção 2.2. A compatibilidade geral entre duas arestas é definida conforme a Equação 2-5.

Durante os testes preliminares, verificou-se que a abordagem *EEB* aplicada ao problema *GBEB* estava unindo arestas que visualmente poderiam ser consideradas distantes. Como o roteamento das arestas não faz parte do problema, o resultado esteticamente não foi "agradável". Dessa forma, foi proposta uma nova medida de compatibilidade $C_d \in [0,1]$, identificada como **compatibilidade de distância**. Essa medida foi incorporada ao cálculo da compatibilidade total, com o objetivo de assegurar que as arestas que estão além de um certo limite de distância não sejam unidas. A nova medida associa a distância entre duas arestas p,q, a partir do ponto médio de cada uma, com um parâmetro limiar:

$$C_d(p,q) = 1 - \frac{||p_m - q_m||}{k}. \tag{6-1}$$

sendo k uma constante de comparação da distância. Nesta tese foi usado um valor para k relativo à distância euclidiana entre os vértices mais extremos do grafo, à esquerda e à direita. Os valores p_m e q_m são os pontos médios das arestas p e q, respectivamente.

Apesar de o problema *GBEB* não possuir restrições geométricas ou semânticas com respeito à forma de montar os feixes de arestas, o cálculo da população inicial foi restringido por *thresholds* pré-definidos. Foram estabelecidas cinco medidas T_a, T_s, T_p, T_v e T_d, representando medidas restritivas de ângulo, tamanho, posição, visibilidade e

distância, respectivamente, todos valorados entre 0 e 1. O *threshold* total é calculado como:

$$T = T_a \cdot T_s \cdot T_p \cdot T_v \cdot T_d \tag{6-2}$$

O modelo de penalização adotado é o mesmo da abordagem *CBEB*:

$$C_{E_i} = \begin{cases} \sum C(p,q), \forall p,q \in E_i, & \text{se } \min(C(p,q)) \geq T; \\ p_e, & \text{se } \min(C(p,q)) < T; \\ 0, & \text{se } |E_i| = 1; \end{cases} \tag{6-3}$$

sendo que

$$p_e = \begin{cases} \sum C(p,q) \cdot (-3), & \text{se } \sum C(p,q) > 0; \\ -3, & \text{se } \sum C(p,q) = 0. \end{cases} \tag{6-4}$$

Os valores de p_e foram definidos empiricamente.

A **compatibilidade total do grafo** C_G é o somatório da compatibilidade de cada feixe:

$$C_G = \sum_{i=1}^{n} C_{E_i}. \tag{6-5}$$

6.4 Método Evolucionário EEB Aplicado ao Problema GBEB

A abordagem *EEB* foi alterada para resolver o problema *GBEB*. A nova concepção é semelhante à adotada no Capítulo 5, sendo elementos comuns: a compatibilidade como parte da função objetivo, a penalização de indivíduos menos aptos e a forma de execução dos operadores de cruzamento.

No entanto, com a relaxação da restrição de adjacência, o espaço de busca aumentou consideravelmente, e como consequência, algumas adaptações foram feitas no algoritmo, em destaque, foi alterada a forma de cálculo da compatibilidade, a maneira como é gerada a população inicial, os operadores de mutação, e, principalmente, foram utilizadas múltiplas populações durante o processo evolutivo. Na sequência, são descritas essas e outras modificações efetuadas no modelo *EEB*.

6.4.1 Cálculo da População Inicial

Durante a montagem da população inicial, os *thresholds* (T_a, T_s, T_p, T_v e T_d) são usados para garantir que serão gerados indivíduos que respeitem os limites pré-

estabelecidos. Foram definidos três algoritmos, cada algoritmo gera $1/3$ da população inicial total. O primeiro método monta feixes com base na cobertura de vértices, apenas com arestas adjacentes e respeitando o limite de compatibilidade estabelecido pelos *thresholds*. Esta é a mesma versão do algoritmo de inicialização, usado para resolver os problemas *ABEB* e *CBEB*, a única diferença está no cálculo das compatibilidades adicionais.

No segundo algoritmo, as arestas do grafo são escolhidas aleatoriamente para serem ou não inseridas em um feixe. No entanto, enquanto os outros métodos de inicialização geram um indivíduo a cada ciclo, o segundo algoritmo cria cinco indivíduos de cada vez. Os *thresholds* são usados para limitar cada um destes cinco indivíduos, ou seja, o primeiro indivíduo deve respeitar a compatibilidade T_a; o segundo, a compatibilidade T_s; o terceiro T_p; o quarto T_v; e, por fim, o quinto indivíduo a compatibilidade T_d. Dessa forma, assegura-se que representantes de diferentes características farão parte da população inicial.

O terceiro algoritmo gera indivíduos escolhendo de forma aleatória as arestas que serão agrupadas em feixes comuns. A escolha das arestas não sofre restrições de adjacência, cobertura ou de compatibilidade. Assim, como para os problemas tratados nos capítulos anteriores, esse procedimento assegura um certo nível de diversidade à população inicial.

Os mesmos artifícios discutidos nos capítulos anteriores e usados para garantir que os indivíduos são viáveis, que todas as arestas estão alocadas a um único feixe, e que os indivíduos não estarão duplicados na população são utilizados também na abordagem *GBEB*.

6.4.2 Operadores de Mutação

Foram mantidos os mesmos operadores de mutação descritos na Seção 4.5.8: **união**, **fusão**, **divisão**, **movimentação** e **remoção**. As modificações nos operadores de mutação foram pequenas, mas necessárias para remover a restrição de adjacência.

O operador de união tinha como procedimento unir duas arestas que estavam sozinhas e que fossem adjacentes. O teste de adjacência foi retirado, não havendo no novo operador qualquer referência à essa condição, ou seja, duas arestas quaisquer podem ser agrupadas em um mesmo feixe. A aleatoriedade da escolha das arestas se manteve. Seguindo a mesma estratégia, os operadores de fusão e movimentação também foram alterados para relaxar a adjacência. Os operadores de divisão e remoção não precisaram ser modificados.

6.4.3 Cooperação entre Populações

Devido à dimensão do espaço de solução e com o objetivo de evitar mínimos locais, foi usada uma abordagem de múltiplas populações que cooperam entre si, baseada no modelo descrito em [22]. Segundo Albuquerque [22], o uso de múltiplas populações além de evitar mínimos locais, permite explorar um intervalo maior do espaço de busca e a reintrodução de material genético nas populações.

No processo de inicialização, são criadas duas populações iniciais diferentes. Essas populações possuem comportamentos evolutivos distintos, no entanto, ao final de cada ciclo as duas populações se comunicam e trocam informações sobre o melhor indivíduo local gerado por cada uma delas. O melhor indivíduo é, então, repassado para a outra população. A Figura 6.1 mostra um esquema geral do processo de cooperação entre as populações.

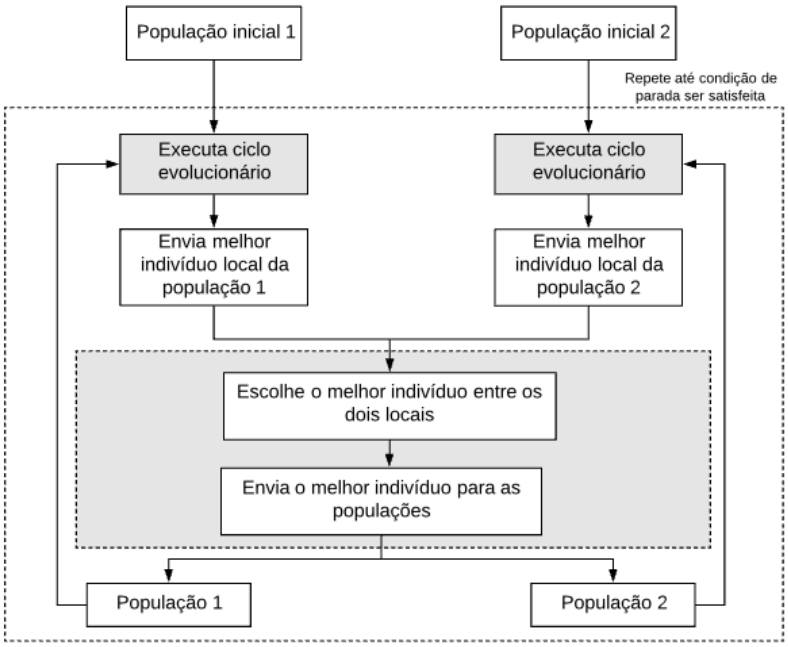


Figura 6.1: *Cooperação entre populações.*

6.5 Experimentos e Discussões

O objetivo de estudo do experimento com a abordagem *EEB* para o problema *GBEB* é verificar, além da efetividade da modelagem, o quanto o algoritmo evolutivo proposto foi efetivo na resolução do problema, e, principalmente, comparar os resultados dos três problemas propostos nesta tese. O problema *GBEB* refere-se a encontrar um conjunto mínimo de feixes de arestas entre as quais haja a máxima compatibilidade, gerando feixes descentralizados e disjuntos. Para tal, foram executados testes independentes com

grafos de tamanho pequeno. Os experimentos foram conduzidos com um grafo sintético e cinco grafos do mundo real:

- G1 - Sintético (20 vértices, 28 arestas) [31];
- G2 - *Zachary Karate Club* (34 vértices, 78 arestas) [122];
- G3 - *Planar Graph GD2015* (66 vértices, 101 arestas) [51];
- G4 - *Dolphin Social Network* (62 vértices, 160 arestas) [37];
- G5 - *MovieLens* (160 vértices, 161 arestas) [81]; e
- G6 - *Les Miserables* (77 vértices, 254 arestas) [57].

O desenho inicial de cada grafo foi predefinido em um arquivo de entrada. Para manter a consistência da forma como os experimentos foram conduzidos, os testes consistiram em executar o algoritmo evolutivo em 100 testes independentes para cada grafo e ângulo. Para o problema *GBEB*, um ângulo $\alpha \in \{30^\circ, 45^\circ, 70^\circ\}$ foi considerado, mas, como a compatibilidade angular calculada de forma diferente da feita nos capítulos anteriores. Os valores que representam esses ângulos são $T_a(30^\circ) = 0,154$, $T_a(45^\circ) = 0,525$ e $T_a(70^\circ) = 0,633$. Para os demais *thresholds*, foram usados $T_s = 0,70$, $T_p = 0,96$, $T_v = 0,72$ e $T_d = 0,96$. Esses valores foram definidos empiricamente em testes preliminares.

Os testes preliminares também mostraram que os mesmos valores para os pesos dos componentes da função objetivo usados no problema *CBEB* não geraram bons resultados. Assim, estes parâmetros foram redefinidos para $w_1 = 0,4$ e $w_2 = 0,6$. Outros parâmetros definidos foram: **tamanho da população** ($u = 300$), a **taxa de cruzamento** ($p_c = 0,98$), e a **taxa de mutação** ($p_m = 0,4$).

Todos os testes foram executados em um MacBook Pro com um processador Intel Core i7 de 2,9 GHz e 8 GB de RAM 1600MHz-DDR3.

6.5.1 Resultados

Os resultados dos testes estão descritos na Tabela 6.1. A média tempo de execução para o problema foi muito alta em comparação com os outros problemas tratados neste trabalho. Analisando-se esses dados, infere-se o quanto a complexidade do problema aumentou com a retirada da restrição de adjacência das arestas, o que, aliada à estratégia de usar a cooperação de populações, tornou o tempo de execução para os grafos maiores impeditivo. Como esta explicado mais a frente, estes aspectos podem ter influenciado a quantidade de gerações que o algoritmo levou para convergir para a melhor solução, o que acarretou o aumento do tempo de execução.

Os valores médios e o desvio padrão para o número de feixes mostram que a variação das soluções em relação à média do número de feixes foi baixa. Entretanto, o

Tabela 6.1: *Resultados do experimento para o problema GBEB, média de 100 execuções independentes,* $u = 300$*,* $p_c = 0,98$*,* $p_m = 0,4$*,* $w_1 = 0,6$*,* $w_2 = 0,4$*,* $T_s = 0,70$*,* $T_p = 0,96$*,* $T_v = 0,72$ *e* $T_d = 0,96$*.*

Grafo	Arestas	α(°)	Feixes da melhor solução	Média da compatibilidade	Média de feixes	Desvio padrão de feixes	Erro padrão de feixes	Média da aptidão	Desvio padrão da aptidão	Erro padrão da aptidão	Média tempo (sec.)
G1	28	30	12 (1)	216,86	12,96	0,7095	0,0710	86,7912	7,51227	0,751227	33
		45	14 (1)	49,14	15,88	0,7004	0,0700	19,6958	0,91644	0,091644	30
		70	14 (1)	40,51	16,00	0,7247	0,0725	16,2408	0,77918	0,077918	28
G2	78	30	25 (1)	624,11	26,86	1,7467	0,1747	249,6655	25,56416	2,556516	262
		45	33 (1)	143,96	35,98	1,7921	0,1792	57,6017	5,94585	0,594585	300
		70	37 (1)	109,97	38,49	1,6847	0,1685	44,0051	5,35199	0,535199	274
G3	101	30	44 (1)	566,65	46,50	1,8023	0,1806	226,6720	17,05982	1,705982	412
		45	58 (1)	133,32	60,33	2,2068	0,2207	53,3396	4,17013	0,417013	452
		70	61 (1)	95,11	64,32	2,0733	0,2073	38,0542	3,78500	0,378500	466
G4	160	30	60 (1)	912,02	60,68	2,7630	0,2763	364,8194	37,40251	3,740251	1366
		45	79 (1)	199,65	82,11	3,1427	0,3143	79,8684	8,35245	0,835245	1446
		70	84 (1)	145,98	90,42	3,6131	0,3613	58,3976	6,21716	0,621816	1447
G5	161	30	44 (1)	2352,17	46,28	3,3123	0,3312	940,8794	102,08115	10,208115	1225
		45	53 (1)	562,70	61,04	3,1267	0,3127	225,0904	24,79434	2,479434	474
		70	60 (1)	445,56	64,27	3,0313	0,3031	178,2331	17,99502	1,799502	1381
G6	254	30	74 (1)	2272,36	83,13	4,2868	0,4287	908,9531	93,80185	9,380185	3159
		45	102 (1)	521,86	110,70	4,2295	0,4230	208,7476	20,71637	2,071637	3898
		70*	112 (1)	387,32	120,03	6,1615	0,6644	154,9317	20,19618	2,177810	3966

*Neste caso foram realizadas apenas 86 execuções.

mesmo padrão não se repete para o valor de aptidão. O alto valor do desvio padrão da aptidão, principalmente para os ângulos menores, demonstra que, apesar de terem sido executados 100 testes independentes, ainda assim, as soluções ficaram esparsas e longe da média de todas as execuções.

Analisando-se a qualidade da melhor solução (medida por meio do número de feixes e valor de compatibilidade), pode-se notar um comportamento semelhante ao do problema *CBEB*. A melhor solução apareceu apenas uma vez dentre os 100 testes independentes executados, mais uma vez evidenciando a complexidade do problema.

As melhores soluções geradas apresentam um padrão visual condizente com os *thresholds* pré-determinados. A Figura 6.2 mostra a solução gerada para o grafo *PlanarGD2015*. Em azul está destacado o feixe de maior tamanho em número de arestas. É possível perceber por essa figura que o modelo respeitou a restrição de feixes descentralizados, mas também gerou feixes centralizados quando isso significou uma qualidade maior do desenho final.

6.6 Comparação dos Resultados dos Problemas Propostos

As Figuras 6.3, 6.4, 6.5 e 6.6 comparam os dados de todos os experimentos executados nesta tese. Os gráficos contêm dados dos problemas *ABEB*, *CBEB* com o ângulo como objetivo, representado como T1 no gráfico, *CBEB* com ângulo e escala

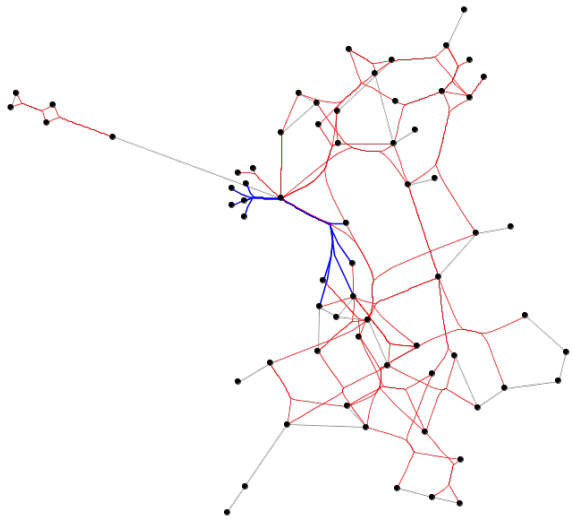


Figura 6.2: *Solução do problema GBEB,* $\alpha = 45°$*, para o grafo Planar.*

como objetivos e com penalidade (-1), representado como T2, *CBEB* com ângulo e escala como objetivos e com penalidade (-3), representado como T3, e por fim do problema *GBEB*. Lembrando que, para esse último, apenas os seis primeiros grafos foram testados, e também que não foi gerado o resultado do problema *CBEB* com penalidade (-3) para o grafo G10 .

As Figuras 6.3 e 6.4 representam os gráficos que comparam a quantidade de feixes da melhor solução e a média de feixes, respectivamente, para todos os experimentos feitos neste trabalho. No geral, os resultados do problema *CBEB*, tendo ângulo e escala como objetivos, possuem mais feixes de arestas, sendo que, como esperado, os resultados do problema *GBEB* foram melhores nesse quesito para os grafos menores, com algumas variações para o ângulo $\alpha = 70°$. Essa característica aparece tanto para a melhor solução de cada bateria de testes, quanto para a média. No entanto, como para o problema *GBEB* os testes foram feitos apenas nos seis primeiros grafos, esses resultados não permitem obter conclusões se essa tendência se mantém nos grafos de médio e grande porte.

Uma outra característica do comportamento dos testes que pode ser analisada é a semelhança de número de feixes para os testes T2 e T3 do problema *CBEB*, bem como o padrão de número de feixes dos problemas *ABEB* e T1 do *CBEB*.

A Figura 6.5 ilustra os gráficos da média de gerações que os cinco testes levaram para gerar a melhor solução. O problema *ABEB* foi mais rápido do que os demais, sendo os testes para o *GBEB* consideravelmente mais lentos, para os seis grafos testados. Como já dito antes, esse comportamento é reflexo tanto do método de evolução implementado, que difere dos demais por usar múltiplas populações, bem como pela complexidade do problema.

Os gráficos da Figura 6.6 ilustram a média do tempo de processamento em

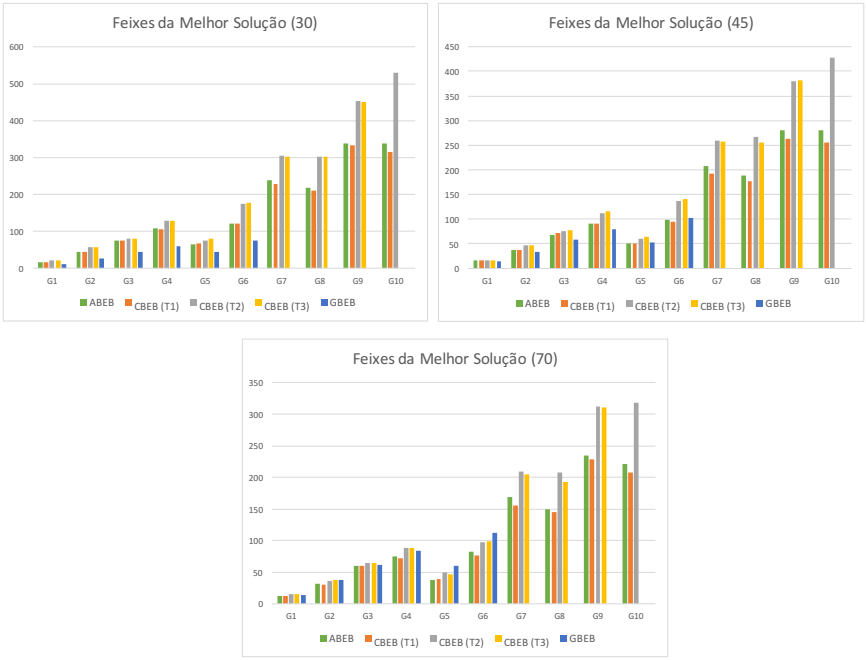


Figura 6.3: *Comparação do número de feixes da melhor solução.*

Figura 6.4: *Comparação da média do número de feixes da melhor solução.*

segundos. O destaque desses gráficos fica por conta do comportamento do grafo G10, que obteve um valor alto para o tempo de processamento. Este valor deve ser desconsiderado, pois para o problema *CBEB* T1 e T2 foram executados apenas 15 testes independentes com esse grafo, sendo que para o problema T3 ele sequer foi avaliado. Para os outros testes, o comportamento é similar ao da média de gerações, sendo o problema *ABEB* mais rápido e o *GBEB* mais lento.

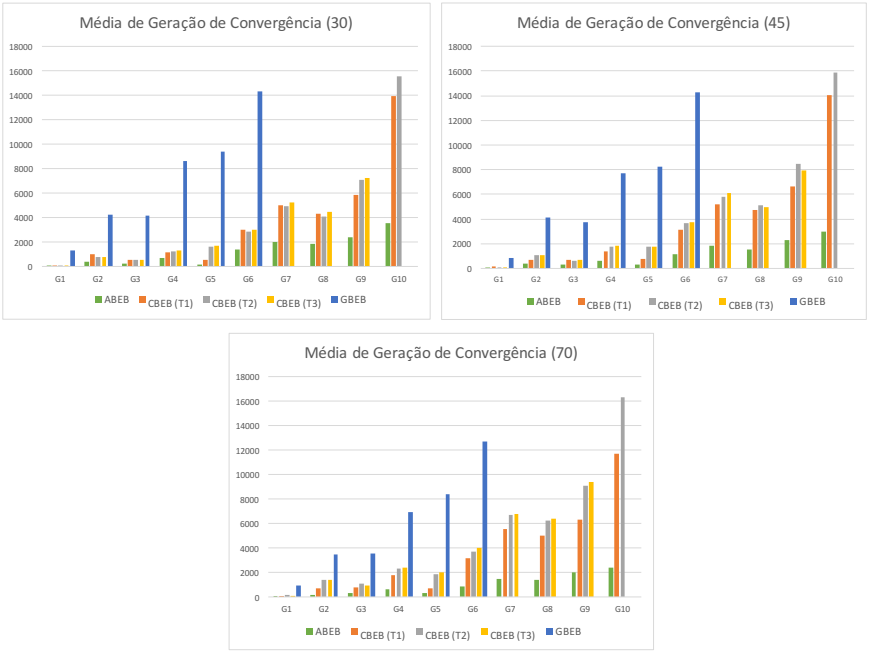


Figura 6.5: *Comparação da média de gerações.*

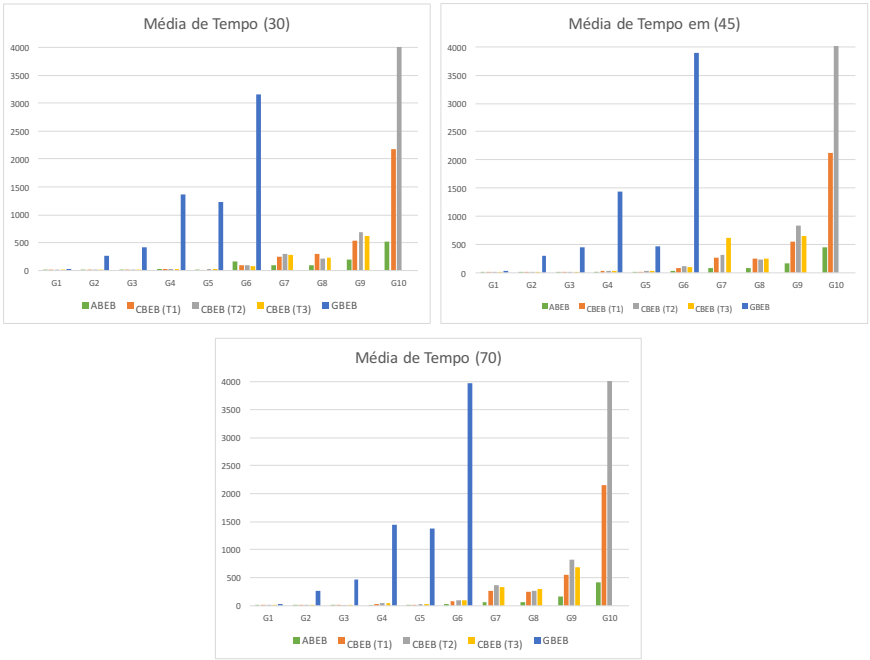


Figura 6.6: *Comparação da média de tempo.*

As Figuras 6.7, 6.8, 6.9 e 6.10 comparam os desenhos da melhor solução dos problemas *ABEB*, *CBEB*, *GBEB*, para os grafos G1, G2, G3 e G4, respectivamente, com uma versão gerada pelo algoritmo *Force-directed edge bundling* (FDEB) [43], implementada no software Tulip©. O algoritmo FDEB faz a união implícita de arestas em feixes descentralizados, sendo que a restrição de feixes disjuntos não é aplicada.

O resultado em todos os grafos gerados pelo Tulip possui feixes com um alto índice de ambiguidade. Apesar de que, em alguns casos, visualmente ser possível

identificar um nível maior de redução da poluição visual (por exemplo, Figura 6.10) comparado aos resultados dos problemas resolvidos pela abordagem *EEB*.

No entanto, as soluções dos problemas *ABEB*, *CBEB*, *GBEB* se apresentam condizentes com o parâmetro angular de $\alpha = 70°$, respeitando a compatibilidade no problema *CBEB*, que foi ampliada no problema *GBEB* pela possibilidade de gerar feixes descentralizados.

Isso demonstra o potencial de tratar união de arestas como um problema de otimização. Os resultados podem ser guiados por parametrização de objetivos e restrições, e, dessa forma, gerar desenhos, cuja qualidade final será mais condizente com a natureza dos dados modelados pelo grafo.

Uma das características não tratadas nesta tese, e que influenciou o aspecto final do desenho, foi o roteamento dos feixes. Como o problema modelado não incluiu nenhuma especificação sobre rotas, nem como restrição ou objetivo, apesar do algoritmo ter gerado boas soluções numéricas, o módulo de renderização usado, por vezes, para os grafos mais complexos, gerou desenhos com um índice alto de poluição visual.

Essa é uma característica dos problemas que trabalham apenas com arestas adjacentes, mas que poderia ser minimizado com um roteamento dos feixes mais eficiente, sendo tratado em nível de problema, ou externamente pela renderização.

Uma outra desvantagem foi o tempo de processamento da abordagem *EEB*. Apesar da eficiência na qualidade da resposta, nos grafos testados, o tempo de execução para os problemas mais complexos, como foi o caso do *GBEB*, demonstra a dificuldade de aplicar uma abordagem baseada em população como estratégia de resolução de problemas de otimização de união de arestas.

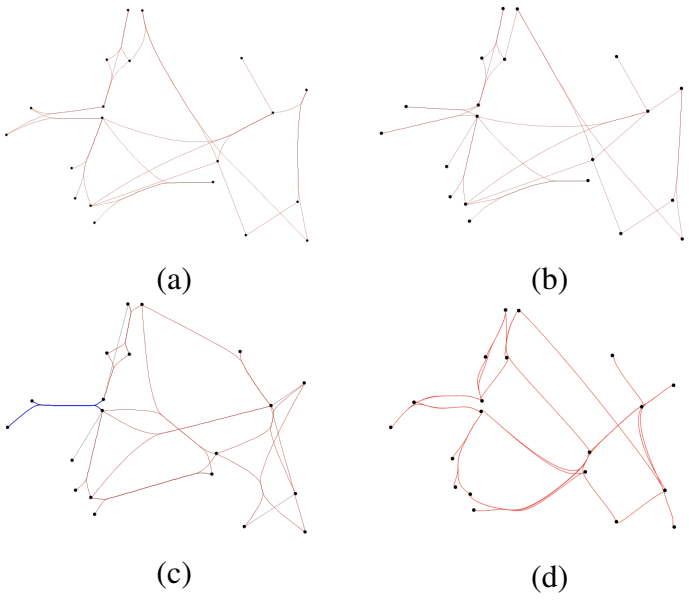


Figura 6.7: *Comparação de resultados do grafo Sintético (G1),* $\alpha = 70°$*: (a) ABEB; (b) CBEB; (c) GBEB; (d) Tulip.*

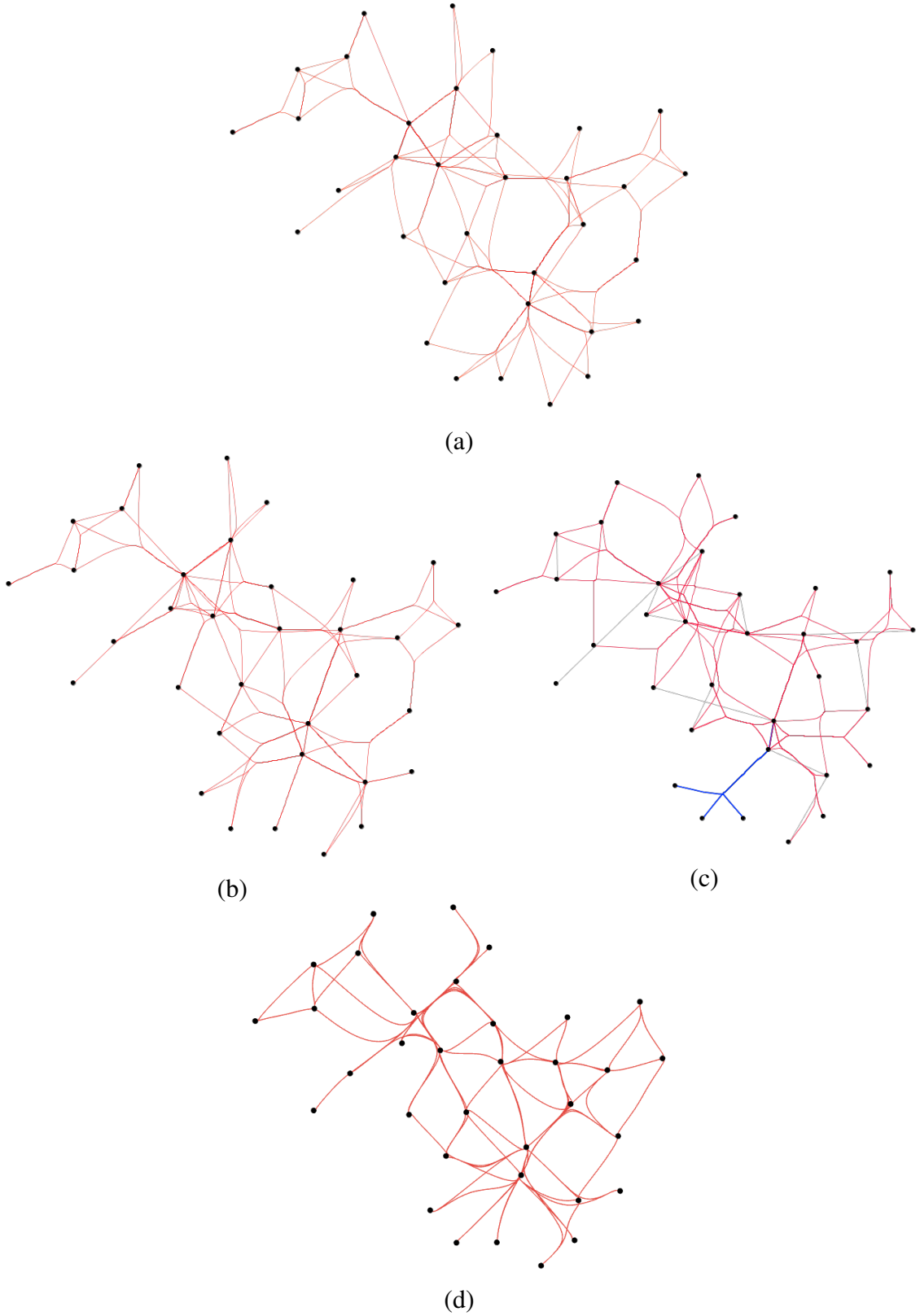


Figura 6.8: *Comparação de resultados para o grafo Zachary Club (G2),* $\alpha = 70°$*: (a) ABEB; (b) CBEB; (c) GBEB; (d) Tulip.*

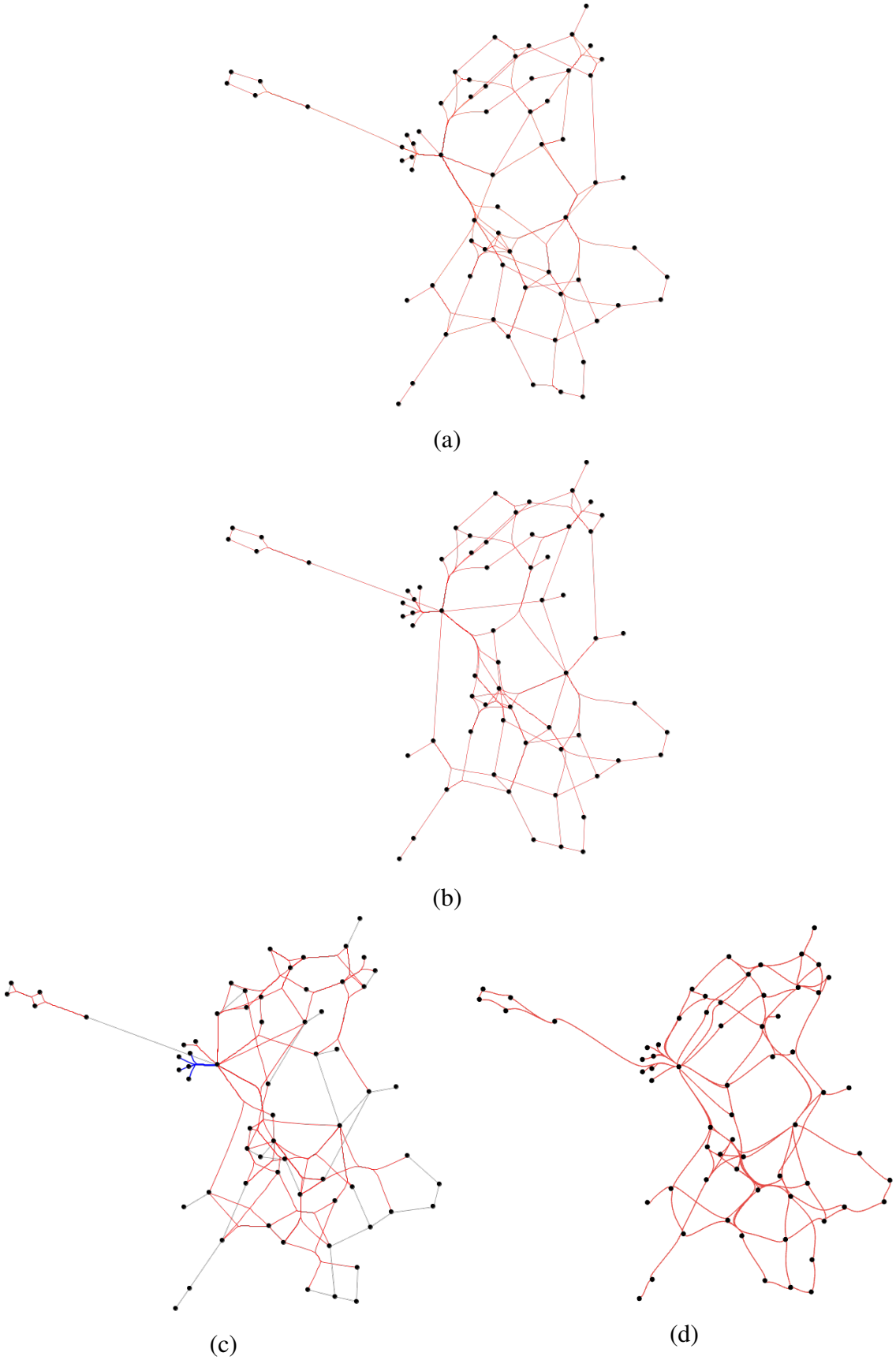


Figura 6.9: *Comparação de resultados para o grafo Planar (G3),* $\alpha = 70°$*: (a) ABEB; (b) CBEB; (c) GBEB; (d) Tulip.*

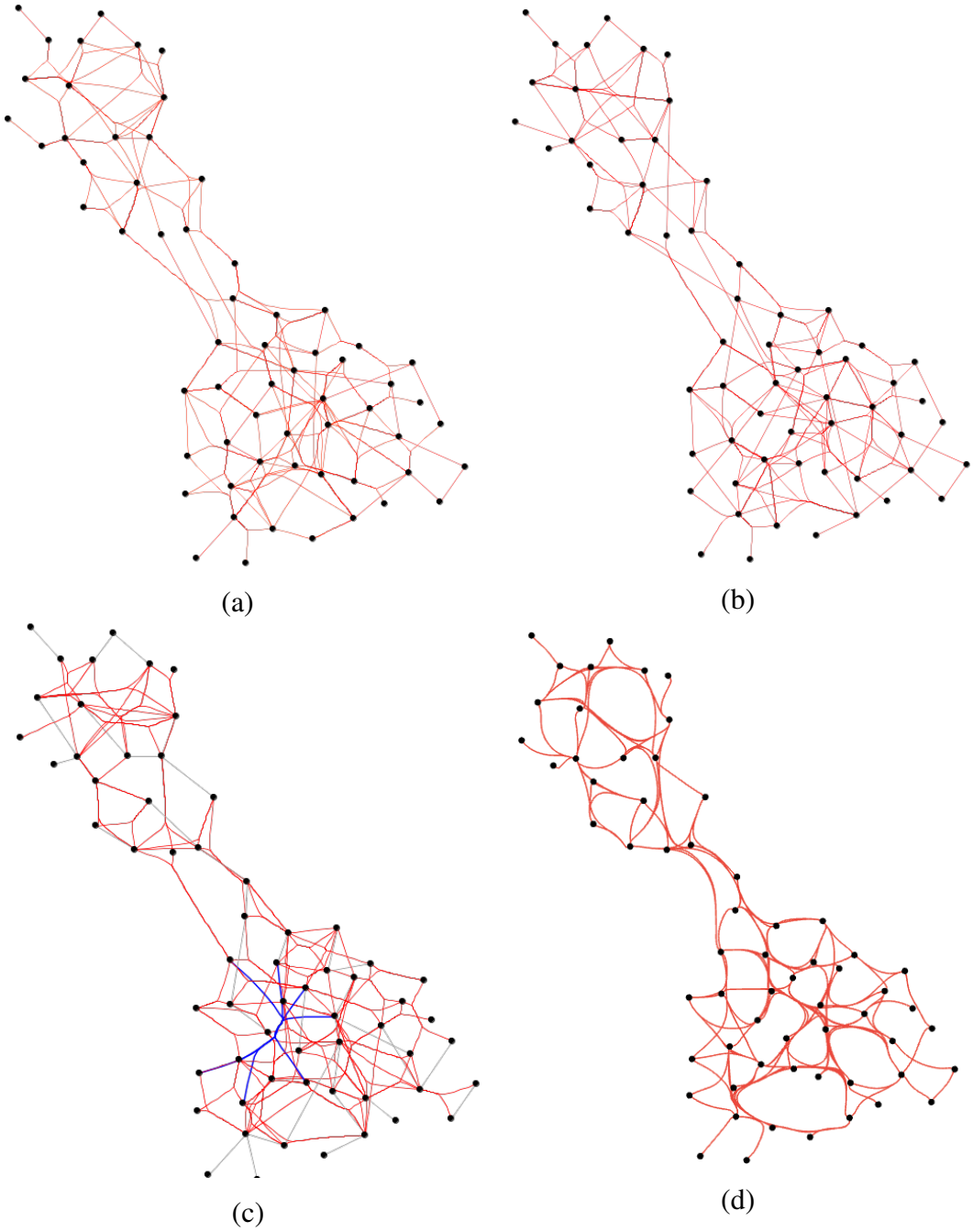


Figura 6.10: *Comparação de resultados para o grafo Dolphin (G4),* $\alpha = 70°$*: (a) ABEB; (b) CBEB; (c) GBEB; (d) Tulip.*

6.7 Considerações Finais

Neste capítulo, foi apresentado o problema *GBEB* para união de arestas descentralizadas, cujo objetivo é minimizar a quantidade de feixes e maximizar um valor de compatibilidade. O problema foi resolvido empregando-se a versão completa das compatibilidades geométricas. Foi proposta, ainda, uma nova medida de compatibilidade que

representa a distância relativa das arestas comparada à dimensão do desenho de grafo.

Uma nova abordagem evolucionária foi implementada, usando cooperação entre múltiplas populações. Novas estratégias de geração de população inicial foram testadas.

O problema *GBEB* foi resolvido para seis grafos de pequeno porte. Os resultados foram satisfatórios em termo de qualidade. No entanto, o tempo de processamento foi muito alto, inviabilizando testes com grafos maiores. Possíveis soluções seriam implementar a recombinação sem a necessidade de reparo, o que implicaria em alterar a representação do indivíduo, ou implementar uma abordagem utilizando apenas o operador genético de mutação. A atualização da função de aptidão apenas do que foi alterado também pode ser uma solução para diminuir o tempo de execução.

Os resultados do problema *GBEB* foram comparados com os dados dos problemas *ABEB* e *CBEB*, bem como com uma versão gerada por uma abordagem implícita de geração de feixes descentralizados. Dentre as conclusões obtidas com os experimentos, está o fato de que, por meio da modelagem de união de arestas como problema de otimização, foi possível controlar o nível de junção das arestas com base nas restrições impostas e nos objetivos estabelecidos. As diferenças entre as soluções de cada problema são reflexos dos parâmetros empregados na resolução.

CAPÍTULO 7

Conclusões

A união de arestas em feixes é uma técnica que busca reduzir a poluição visual do desenho de grafos, por meio da união de arestas identificadas como compatíveis. Apesar de existirem diversos métodos que produzem bons desenhos de maneira muito rápida, muitas questões relacionadas a qualidade das soluções geradas ainda não foram bem definidas. De forma geral, essas abordagens não se preocupam com a formalização do problema investigado e em encontrar o "melhor" conjunto de feixes que respeite critérios matematicamente determinados, as quais vão além da redução da poluição visual.

Esta tese investigou a união de arestas como um problema de otimização combinatória, e, em particular, foram tratados pontos como a formalização de problemas da área, o estudo da complexidade de um deles e a implementação de uma heurística evolutiva para resolver tais problemas. Os temas abordados nesta tese são complementos importantes pouco explorados em outras técnicas para desenho de grafos com feixes que, geralmente, possuem um foco maior no *layout* final, e não na complexidade computacional do processo de redução da poluição visual.

Com esse enfoque, primeiramente, foi apresentada uma teorização baseada no levantamento e análise dos principais artigos da área. O objetivo foi delimitar o escopo das técnicas para união de arestas, não no método de resolução implementado, mas sim nas características do problema tratado por eles. Como contribuição foram discutidas duas classes de problemas de união de arestas: implícita e explícita. Bem como definido dois tipos de feixes: centralizados e descentralizados. Essa caracterização auxilia o estabelecimento de parâmetros que podem ser facilmente mapeados em funções a serem otimizadas, como, por exemplo, critérios estéticos, quantificação dos feixes e outros a serem configurados pelo usuário.

Foi proposta uma formulação matemática genérica para problemas de otimização de união explícita de arestas que permite a ampliação de pesquisas, o uso de diferentes ferramentas e técnicas que possam ajudar a entender a essência de problemas de união de arestas, efetuar a análise de complexidade, e também a comparação da efetividade de diferentes abordagens algorítmicas.

Baseado nesta formulação geral foram definidos e analisados quatro problemas

de otimização combinatória para união explícita de arestas. O primeiro problema tem como objetivo principal minimizar a quantidade de feixes de um grafo, unindo em conjuntos disjuntos somente as arestas adjacentes e gerando feixes centralizados. O segundo problema possui o mesmo objetivo e as mesmas restrições do anterior, mas acrescenta uma restrição que estabelece que as arestas de um mesmo feixe devem respeitar um ângulo máximo, quando comparadas duas a duas. Já o terceiro problema, que também é uma evolução do primeiro, insere um valor de compatibilidade como um novo objetivo do problema. A meta é minimizar o número de feixes e maximizar a compatibilidade geral do grafo. Por fim, o quarto problema é uma extensão do terceiro, no qual a restrição de adjacência é relaxada, podendo ser gerados tanto feixes centralizados como descentralizados. Foi feita a análise de complexidade do primeiro problema, que foi provado pertencer a classe de problemas NP-completos.

Com o objetivo de investigar o quão complexos são os problemas definidos nesta tese, foi proposto um modelo de programação linear para o primeiro problema, o mais simples deles. O modelo se mostrou efetivo na resolução exata do problema, no entanto, a sua dimensão, em termos de complexidade de resolução, para vários métodos de solução do modelo de programação linear justificou o emprego de uma abordagem heurística.

Um algoritmo evolucionário foi implementado e aplicado em três dos problemas definidos. O método segue a estrutura básica desse tipo de técnica, sendo que os operadores de seleção, cruzamento e mutação foram adequados à representação do indivíduo e implementados para respeitar as restrições dos problemas.

O método se mostrou efetivo na resolução dos problemas propostos para instâncias de pequeno, médio e grande porte. A principal desvantagem é o seu tempo de execução em comparação com abordagens clássicas. O método evolucionário leva mais tempo para gerar um desenho com feixes, mas isso porque seu foco é resolver um problema de otimização da melhor forma possível em busca de uma solução próxima da ótima e não em desempenho, enquanto os os métodos clássicos para união de arestas possuem o foco em ser uma heurística rápida.

Para melhorar o tempo de execução da abordagem evolucionária podem ser implementadas melhorias, como por exemplo, evitar recalcular os valores de compatibilidade ou outros parâmetros para feixes que não melhoram entre uma interação e outra, é possível ainda, explorar paralelismo na codificação da abordagem. Além disso, pode-se montar uma solução híbrida usando-se soluções advindas de algoritmos clássicos na montagem da população inicial, e assim possivelmente ajudar na convergência para a melhor solução.

Foram executados experimentos controlados para instância de 10 grafos de tamanhos diferentes. Os resultados conseguidos atendiam as expectativas buscadas durante o doutorado. Identificou-se que a minimização da quantidade de feixes é uma medida

efetiva a ser otimizada em um desenho de grafo com feixes e que auxilia a redução da poluição visual. No entanto, para assegurar que os feixes gerados estivessem ajustados aos dados, foi necessário associá-la a medidas de compatibilidade que possibilitaram guiar o processo de otimização para o padrão de desenho final desejado.

Verificou-se também que à medida que a complexidade dos problemas aumenta, com a inserção de novos objetivos e/ou restrições, e mais difícil foi para a abordagem evolucionária encontrar a melhor solução. Esse fato, juntamente com a demonstração da NP-completude do problema de número um, mostra o quão a união de arestas em feixes pode ser complexa.

A comparação de desenhos com arestas unidas em feixes é hoje um dos principais pontos que alguns pesquisadores tentam resolver. Nesta tese foi apresentada uma alternativa, que permite orientar os esforços na definição de medidas para determinação da qualidade, baseada no problema resolvido por cada técnica. Mostrou-se que por meio da definição precisa do objeto de estudo é possível construir soluções diferentes que exploram aspectos diversos do desenho resultante ou dos dados representados pelo grafo. Assim, as medidas de qualidade podem ser centradas tanto na visualização final (redução da poluição visual, curvatura das arestas, etc.), quanto nos dados (qualidade do agrupamento, semântica, etc.). A escolha está relacionada ao problema tratado pelo algoritmo.

Formalizando as contribuições desta teste destacam-se:

- a definição de uma teoria sobre abordagens implícitas e explícitas para união de arestas;
- a criação de uma formulação geral para problemas de otimização para união explícita de arestas;
- a formalização de quatro problemas de otimização para união explícita;
- a prova da NP-completude do problema de união explícita de arestas em feixes centralizados;
- a identificação de propriedades, em relação ao ângulo máximo, do problema da união explícita de arestas em feixes centralizados com restrição angular;
- a implementação de um modelo de modelo de programação linear inteira para o problema da união explícita de arestas em feixes centralizados com restrição angular;
- a criação de uma abordagem evolucionária para união explícita de arestas; e
- o fornecimento de valores quantitativos, advindos dos experimentos, que podem servir como *benchmarking* para futuras pesquisas.

Como trabalhos futuros pretende-se ampliar a abordagem evolucionária para que ela cumpra todo o ciclo proposto na formulação matemática genérica, e possa ser aplicada a problemas cujo desenho com feixes seja também um elemento de entrada do problema.

Dessa forma, medidas como poluição visual, curvatura de arestas e outros elementos mais relacionados ao desenho poderão fazer parte dos objetivos a serem otimizados. Além disso, pretende-se examinar a possibilidade da inclusão do roteamento dos feixes no processo de otimização, possibilitando assim condicionar a escolha dos conjuntos de arestas, também a elementos de roteamento como obstáculos, cruzamentos, etc.

Por fim, também como trabalho futuro, pretende-se explorar problemas retirados da literatura. Assim será possível comparar efetivamente a qualidade desses métodos a partir de medidas de otimização, que atestem qual a melhor solução dentre as soluções criadas pelas diversas abordagens já existentes.

7.1 Publicações Decorrentes da Tese

Ao longo do desenvolvimento da tese foram gerados dois artigos. O primeiro, publicado na conferência IVAPP, apresenta os dois problemas *ABEB* e *CBEB*, bem como detalha a abordagem EEB e os experimentos realizados.

- Ferreira J., Nascimento H. and Foulds L. (2018). *An Evolutionary Algorithm for an Optimization Model of Edge Bundling*. In Proceedings of the 13th International Joint Conference on Computer Vision, Imaging and Computer Graphics Theory and Applications (***Best Student Paper Award***).

Uma versão mais completa desse artigo esta sendo finalizada e será submetida para uma edição especial (*Selected Papers from IVAPP 2018*) da revista *Information* publicada pela MDPI.

O segundo foi submetido à revista *COMPUTER GRAPHICS Forum*, estando em fase de *major review*. Nesse artigo são discutidos a formulação geral de otimização combinatória para problemas de união de arestas, a NP-completude do problema $EB-star$ e o modelo de programação linear implementado.

- Ferreira, J.M., Nascimento, H,A.D., Quigley, A.J., Foulds, L.R.. *Computational Complexity of Edge Bundling Problems*. Submitted to COMPUTER GRAPHICS Forum (6/2017) (***under Major Review***).

Referências Bibliográficas

[1] ALAM, J.; FINK, M.; PUPYREV, S. **The bundled crossing number**. In: *Proceedings of the 24th International Symposium Graph Drawing and Network Visualization*, p. 399–412, 2016.

[2] ANGELINI, P.; BEKOS, M.; KAUFMANN, M.; KINDERMANN, P.; SCHNECK, T. **1-fan-bundle-planar drawings of graphs**. In: *24th Internat. Symp. Graph Drawing – Poster*, p. 634–636, 2016.

[3] ASHLOCK, D. **Evolucionary Computation for Modeling and Optimization**. Springer, 2006.

[4] BÄCK, T. **Evolutionary Algorithms in Theory and Practice: Evolution Strategies, Evolutionary Programming, Genetic Algorithms**. Oxford Univ. Press., 1996.

[5] BANZHAF, W.; NORDIN, P.; KELLER, R. E.; FRANCONE, F. D. **Genetic Programming: An Introduction: on the Automatic Evolution of Computer Programs and Its Applications**. Morgan-Kaufmann, 1998.

[6] BATTISTA, G.; EADES, P.; TAMASSIA, R.; TOLLIS, I. G. **Graph Drawing: Algorithms for the Visualization of Graphs**. Prentice Hall PTRr, 1st edition, 1998.

[7] BECK, F.; BURCH, M.; DIEHL, S.; WEISKOPF, D. **The state of the art in visualizing dynamic graphs**. In: *Proceedings of the EuroVis - STARs*, p. 83–103. Eurographics Association, 2014.

[8] BECK, F.; PUPPE, M.; BRAUN, P.; BURCH, M.; DIEHL, S. **Edge bundling without reducing the source to target traceability**. In: *Proceedings of the InfoVis Poster*, p. 298–305, 2011.

[9] BENNETT, C.; RYALL, J.; SPALTEHOLZ, L.; GOOCH, A. **The aesthetics of graph visualization**. In: *Proceedings of the Computational Aesthetics*, p. 57–64. Eurographics Association, 2007.

[10] BERTSEKAS, D. M. **Network Optimization: Continuous and Discrete Models**. Athena Scientic, 1998.

[11] BEZERRA, S. N. **Algoritmos Evolutivos Paralelos Aplicados ao Problema das pMedianas**. dissertation, Centro Federal de Educação Tecnológica de Minas Gerais, 2008.

[12] BLUM, C.; ROLI, A. **Metaheuristics in combinatorial optimization: Overview and conceptual comparison**. *ACM Computing Surveys*, 35(3):268–308, 2003.

[13] BÖTTGER, J.; SCHÄFER, A.; LOHMANN, G.; VILLRINGER, A.; MARGULIES, D. S. **Three-dimensional mean-shift edge bundling for the visualization of functional connectivity in the brain**. *IEEE Transactions on Visualization and Computer Graphics*, 20(3):471–480, 2014.

[14] BOUTS, Q. W.; SPECKMANN, B. **Clustered edge routing**. In: *IEEE Pacific Visualization Symposium (PacificVis)*, p. 55–62, 2015.

[15] BRUCKDORFER, T.; CORNELSEN, S.; GUTWENGER, C.; KAUFMANN, M.; MONTECCHIANI, F.; NÖLLENBURG, M.; WOLFF, A. **Progress on partial edge drawings**. *CoRR*, abs/1209.0830, 2012.

[16] BURCH, M.; SCHMAUDER, H.; WEISKOPF, D. **Edge bundling by rapidly-exploring random trees**. In: *Proceedings of the 17th International Conference on Information Visualisation*, p. 28–35, 2013.

[17] ČERMÁK, M.; DOKULIL, J.; KATRENIAKOVÁ, J. **Edge routing and bundling for graphs with fixed node positions**. In: *Proceedings of the 15th International Conference on Information Visualisation, IV*, p. 475–481, 2011.

[18] COELLO, C.; LECHUNGA, M. **A proposal for multiple objective particle swarm optimization**. In: *Proceedings of the IEEE Congress on Evolutionary Computation*, p. 1051–1056, 2002.

[19] CORNELISSEN, B.; HOLTEN, D.; ZAIDMAN, A.; MOONEN, L.; VAN WIJK, J. J.; VAN DEURSEN, A. **Understanding execution traces using massive sequence and circular bundle views**. In: *Proceedings of the 15th International Conference on Program Comprehension, ICPC*, p. 49–58, 2007.

[20] CUI, W.; ZHOU, H.; QU, H.; WONG, P. C.; LI, X. **Geometry-based edge clustering for graph visualization**. *IEEE Transactions on Visualization Computer Graphics*, 14(6):1277–1284, 2008.

[21] DANTAS, M. J. P. **Metodologia de Síntese de Filtros de Microondas de Topologias Arbitrárias Utilizando Algoritmo Evolucionário Híbrido (Memético) associado a Conhecimento Especialista**. PhD thesis, Universidade de Brasília. Faculdade de Tecnologia. Brasília, 2008.

[22] DE ALBUQUERQUE, A. C. M. L. **Algoritmos Genéticos e Processamento Paralelos Aplicados à Definição e Treinamento de Redes Neurais Perceptron de Múltiplas Camadas**. PhD thesis, Universidade Federal do Rio Grande do Norte, 2005.

[23] DEBIASI, A.; SIMOES, B.; AMICIS, R. D. **Force directed flow map layout**. In: *Proceedings of the International Conference on Information Visualization Theory and Applications*, p. 170–177, 2014.

[24] EATON, J. W. **Gnu octave**. https://www.gnu.org/software/octave/doc/interpreter/, 2017. [Online; accessed 14-May-2017].

[25] ERSOY, O.; HURTER, C.; PAULOVICH, F. V.; CANTAREIRO, G.; TELEA, A. **Skeleton-based edge bundling for graph visualization**. *IEEE Transactions on Visualization Computer Graphics*, 12(17):2364–2373, 2011.

[26] ERSOY, O.; HURTER, C.; TELEA, A. **Mean shift for graph bundling**. In: *Proceedings of the ICT.OPEN*, 2012.

[27] ERSOY, O. **Image-based graph visualization: simplified visualization of large graphs**. dissertation, University of Groninge, 2013.

[28] ERSOY, O.; TELEA, A. **Graph edge bundling by medial axes**. In: *Proceedings of the Sixteenth Annual Conference of the Advanced School for Computing and Imaging, ASCI/IPA/SIKS*, 2011.

[29] FERREIRA, J. M. **Uma Investigação em Otimização Interativa Multiusuário para Desenho de Grafos**. dissertation, Universidade Federal de Goiás, 2006.

[30] FERREIRA, J. M.; NASCIMENTO, H. A. D.; FOULDS, L. R. **An evolutionary algorithm for an optimization model of edge bundling**. In: *Proceedings of the 13th International Joint Conference on Computer Vision, Imaging and Computer Graphics Theory and Applications*, p. 132–143, 2018.

[31] FERREIRA, J. M.; NASCIMENTO, H. A. D.; QUIGLEY, A. J.; FOULDS, L. R. **Computational complexity of edge bundling problemss**. http://inf.ufg.br/biblioteca-digital, 2017. Technical report, Federal University of Goiás.

[32] FOGEL, D. B. **An introduction to simulated evolutionary optimization**. *IEEE Trans. Neural Networks*, 5(1):3–14, 1994.

[33] FRANKLIN, J. **The formal sciences discover the philosophers stone**. *Studies in History and Philosophy of Science PartA*, 25(4):513–533, 1994.

[34] GANSNER, E. R.; HU, Y.; NORTH, S.; SCHEIDEGGER, C. **Multilevel agglomerative edge bundling for visualizing large graphs**. In: *Proceedings of the IEEE Pacific Visualization Symposium*, p. 187–194, 2011.

[35] GANSNER, E. R.; KOREN, Y. **Improved circular layouts**. In: *Proceedings of the 14th International Symposium on Graph Drawing*, p. 386–398, 2006.

[36] GAREY, M. R.; JOHNSON, D. S. **Computers and Intractability. A Guide to the Theory of NP-Completeness**. W. H. Freeman, 1979.

[37] GIRVAN, M.; NEWMAN, M. **Community structure in social and biological networks**. *Natl. Acad. Sci. USA*, 99(12):7821–7826, 2002.

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

[39] GREFENSTETTE, J. **Optimization of control parameters for genetic algorithms**. *IEEE Transactions on System, Man and Cybernetics,*, 16(1):122–128, 1986.

[40] HARARY, F. **Graph Theory**. Addison-Wesley, 1969.

[41] HERMAN, I.; MELANCON, G.; MARSHALL, M. S. **Graph visualisation and navigation in information visualization: A survey**. *IEEE Transactions on Visualization and Computer Graphics*, 6(10):24–43, 2000.

[42] HOLTEN, D. **Hierarchical edge bundles: Visualization of adjacency relations in hierarchical data**. *IEEE Transactions on Visualization Computer Graphics*, 12(5):741–748, 2006.

[43] HOLTEN, D.; VAN WIJK, J. J. **Force-directed edge bundling for graph visualization**. *Comput. Graph. Forum*, 28(3):983–990, 2009.

[44] HOLTEN, D.; CORNELISSEN, B.; VAN WIJK, J. J. **Trace visualization using hierarchical edge bundles and massive sequence views**. In: *Proceedings of the 4th IEEE International Workshop on Visualizing Software for Understanding and Analysis, VISSOFT*, p. 47–54, 2007.

[45] HOP, W.; DE RIDDER, S.; FRASINCAR, F.; HOGENBOOM, F. **Using hierarchical edge bundles to visualize complex ontologies in glow**. In: Ossowski, S.; Lecca, P., editors, *Proceedings of the 27th Symposium on Applied Computing*, p. 304–311. ACM, 2012.

[46] HURTER, C.; ERSOY, O.; FABRIKANT, S. I.; KLEIN, T. R.; TELEA, A. C. **Bundled visualization of dynamic graph and trail data**. *IEEE Transactions on Visualization and Computer Graphics*, 20(8):1141–1157, 2014.

[47] HURTER, C.; ERSOY, O.; TELEA, A. **Graph bundling by kernel density estimation**. *Comput. Graph. Forum*, 31(3):865–874, 2012.

[48] HURTER, C.; ERSOY, O.; TELEA, A. **Smooth bundling of large streaming and sequence graphs**. In: *Proceedings of the Pacific Visualization Symposium*, p. 41–48, 2013.

[49] HURTER, C.; PUECHMOREL, S.; NICOL, F.; TELEA, A. **Functional decomposition for bundled simplification of trail sets**. *IEEE Transactions on Visualization and Computer Graphics*, p. 1, 2017.

[50] HURTER, C. **Image-Based Visualization: Interactive Multidimensional Data Exploration**. Morgan and Claypool Publishers, 2016.

[51] ISGCI. **Information system on graph classes and their inclusions**. http://www.csun.edu/gd2015/topics.htm, 2015. [Online; accessed 02-November-2017].

[52] JIA, Y.; GARLAND, M.; HART, J. C. **Social network clustering and visualization using hierarchical edge bundles**. *Comput. Graph. Forum*, 30(8):2314–2327, 2011.

[53] JONG, K. A. D. **Evolucionary Computation: A Unified Approach**. The MIT Press, 2006.

[54] KARP, R. M.; HOPCROFT, J. E. **A n5/2 algorithm for maximum matchings in bipartite graphs**. *SIAM Journal of Computing*, 1973.

[55] KIENREICH, W.; SEIFERT, C. **An application of edge bundling techniques to the visualization of media analysis results**. In: *Proceedings of the International Conference on Information Visualisation, IV*, p. 375–380. IEEE Computer Society, 2010.

[56] KIENREICH, W.; WOZELKA, R.; SABOL, V.; SEIFERT, C. **Graph visualization using hierarchical edge routing and bundling**. In: Matkovic, K.; Santucci, G., editors, *Proceedings of the International Workshop on Visual Analytics*. The Eurographics Association, 2012.

[57] KNUTH, D. **The Stanford GraphBase: A Platform for Combinatorial Computing**. Addison-Wesley, 1993.

[58] KOBOUROV, S. G.; PUPYREV, S.; SAKET, B. **Are crossings important for drawing large graphs?** In: *Graph Drawing*, volume 8871 de **Lecture Notes in Computer Science**, p. 234–245. Springer, 2014.

[59] KOROSEC, P. **New Achievements in Evolutionary Computation**. INTECH, 2010.

[60] KRUSE, R.; BORGELT, C.; KLAWONN, F.; MOEWES, C.; STEINBRECHER, M.; HELD, P. **Computational Intelligence: A Methodological Introduction**. Springer, 2013.

[61] LAMBERT, A.; BOURQUI, R.; AUBER, D. **3d edge bundling for geographical data visualization**. In: *Proceedings of the IV*, p. 329–335. IEEE Computer Society, 2010.

[62] LAMBERT, A.; BOURQUI, R.; AUBER, D. **Winding roads: Routing edges into bundles**. *Comput. Graph. Forum*, 29(3):853–862, 2010.

[63] LAWLER, E. **Combinatorial Optimization – Networks and Matroids**. Holt, Rinehart and Wiston, 1976.

[64] LHUILLIER, A.; HURTER, C.; TELEA, A. **State of the art in edge and trail bundling techniques**. *Computer Graphics Forum*, 36(3):619–645, 2017.

[65] LHUILLIER, A.; HURTER, C.; TELEA, A. **FFTEB: Edge bundling of huge graphs by the fast fourier transform**. *IEEE Pacific Visualization Symposium*, p. 190–199, 2017.

[66] LIN, X. **On the computational complexity of edge concentration**. *DAMATH: Discrete Applied Mathem. and Combin. Operat. Research*, 101:1–3, 2000.

[67] LINDEN, R. **Algoritmos Genéticos**. Ciencia Moderna, 2012.

[68] LOBATO, F. M. F. **Estratégias Evolucionárias para Otimização no Tratamento de Dados Ausentes por Imputação Múltipla de Dados**. dissertation, Universidade Federal do Pará, 2016.

[69] LOVASZ, L. **Large Networks and Graph Limits**. American Mathematical Society, 2012.

[70] LUO, S. J.; LIU, C. L.; CHEN, B. Y.; MA, K. L. **Ambiguity-free edge-bundling for interactive graph visualization**. *IEEE Transactions on Visualization Computer Graphics*, 18(5):810–821, 2012.

[71] MANN, Z. A. **Optimization in Computer Engineering–Theory and Applications**. Scientific Research Publishing, 2011.

[72] MARSHALL, M. S. **Methods and Tools for the Visualization and Navigation of Graphs**. PhD thesis, University Bordeaux, 2001.

[73] MARTINO, J. A. D.; CELANI, G. **O algoritmo evolutivo como método projetual**. In: *Proceedings of the Congreso de la Sociedad Iberoamericana de Gráfica Digital, SIGRADI*, p. 570–574, 2012.

[74] MCDONNELL, K. T.; MUELLER, K. **Illustrative parallel coordinates**. In: *Proceedings of the 10th Joint Eurographics / IEEE - VGTC Conference on Visualization*, p. 1031–1038, 2008.

[75] MCGRAW, T. **Graph-based visualization of neuronal connectivity using matrix block partitioning and edge bundling**. In: *Proceedings of the 11th International Symposium Advances in Visual Computing, ISVC*, p. 3–13. Springer International Publishing, 2015.

[76] MCKNIGHT, R. L. **Low-stretch trees for network visualization**. dissertation, University of British Columbia, 2015.

[77] MICHAILIDIS, G.; DE LEEUW, J. **Data visualization through graph drawing**. *Comput. Statist*, 16:435–450, 2001.

[78] MICHALEWICZ, Z.; JANIKOW, J. **Handling constraints in genetic algorithms**. In: *Proceedings of the 4th International Conference on Genetic Algorithms*, p. 151–157, 1991.

[79] MICHALEWICZ, Z. **Genetic Algorithms + Data Structures = Evolution Programs**. Springer, Charllote, NC, USA, 1996.

[80] MOURA, D. C. **3D density histograms for criteria-driven edge bundling**. *CoRR*, abs/1504.02687, 2015.

[81] MOVIELENS. **Movielens**. http://www.eecs.wsu.edu/~yyao/, 2017. [Online; accessed 02-November-2017].

[82] NEWBERY, F. J. **Edge concentration: A method for clustering directed graphs**. *SACM SIGSOFT Software Engineering Notes*, 14(7):76–85, 1989.

[83] NEWMAN, M. **Modularity and community structure in networks**. *National Academy of Sciences*, 103(23):8577–8582, 2006.

[84] NEWMAN, M.; GIVAN, M. **Finding and evaluating community structure in networks**. *Phys. Rev.*, E 69(2), 2004.

[85] NGUYEN, Q.; EDGES, P.; HONG, S.-H. **Streameb: Stream edge bundling**. In: Didimo, W.; Patrignani, M., editors, *Proceedings of the 20th International Symposium on Graph Drawing*, volume 7704 de **Lecture Notes in Computer Science**, p. 400–413. Springer, 2012.

[86] NGUYEN, Q.; EDGES, P.; HONG, S.-H. **On the faithfulness of graph visualizations**. In: *Proceedings of the IEEE Pacific Visualization Symposium*, p. 209–216, 2013.

[87] NGUYEN, Q.; HONG, S.-H.; EDGES, P. **Tgi-eb: A new framework for edge bundling integrating topology, geometry and importance**. In: *Proceedings of the 19th International Symposium on Graph Drawing*, volume 7034 de **Lecture Notes in Computer Science**, p. 123–135. Springer, 2011.

[88] NGUYEN, Q.; HONG, S.-H.; EDGES, P. **2.5d edge bundling**. In: *Proceedings of the IEEE InfoVis 2016 (poster)*, 2016.

[89] NOCAJ, A.; BRANDES, U. **Stub bundling and confluent spirals for geographic networks**. In: *Proceedings of the 21st International Symposium on Graph Drawing*, p. 388–399, 2013.

[90] OPTIMIZATION, G. **Gurobi optimizer quick start guide**. http://www.gurobi.com/documentation/, 2017. [Online; accessed 14-May-2017].

[91] PAPADIMITRIOU, C. H.; STEIGLITZ, K. **Combinatorial Optimization: Algoritms and Complexity**. Prentice Hall, 1998.

[92] PENG, D.; N. LU, W. C.; PENG, Q. **Sideknot: Revealing relation patterns for graph visualization**. In: *Proceedings of the IEEE Pacific Visualization Symposium*, p. 65–72, 2012.

[93] PEYSAKHOVICH, V.; HURTER, C.; TELEA, A. **Attribute-driven edge bundling for general graphs with applications in trail analysis**. In: *Proceedings of the IEEE Pacific Visualization Symposium*, p. 39–46, 2015.

[94] POLISCIUC, E.; CRUZ, A.; MACHADO, P.; ARRAIS, J. P. **On the role of aesthetics in genetic algorithms applied to graph drawing**. In: *Proceedings of the Genetic and Evolutionary Computation Conference Companion*, p. 1713–1720, 2017.

[95] POLISCIUC, E.; CRUZ, P.; AMARO, H.; AS, C. M.; MACHADO, P. **Flow map of products transported among warehouses and supermarkets**. In: *Proceedings of the 11th Joint Conference on Computer Vision, Imaging and Computer Graphics Theory and Applications*, p. 177–186, 2016.

[96] PUPYREV, S.; NACHMANSON, L.; BEREG, S.; HOLROYD, A. L. **Edge routing with ordered bundles**. In: *Proceedings of the 19th International Conference on Graph Drawing*, p. 136–147, 2011.

[97] PUPYREV, S.; NACHMANSON, L.; KAUFMANN, M. **Improving layered graph layouts with edge bundling**. In: *Proceedings of the Conference on Graph Drawing*, p. 329–340, 2010.

[98] PURCHASE, H. C.; COHEN, R. F.; JAMES, M. **Validating graph drawing aesthetics**. *Lecture Notes in Computer Science*, 1027:435–446, 1995.

[99] QU, H.; ZHOU, H.; WU, Y. **Controllable and progressive edge clustering for large networks**. In: *Proceedings of the 14th International Conference on Graph Drawing*, p. 399–404, 2006.

[100] ROEVA, O.; FIDANOVA, S.; PAPRZYCKI, M. **Influence of the population size on the genetic algorithm performance in case of cultivation process modelling.** In: *Proceedings of the Federated Conference on Computer Science and Information Systems*, p. 371–376, 2013.

[101] SAGA, R. **Quantitative evaluation for edge bundling by difference of edge lengths and area occupation**. In: *HCI International Posters Extended Abstracts*, p. 287–290, 2016.

[102] SAGA, R.; YAMASHITA, T. **Multi-type edge bundling in force-directed layout and evaluation**. In: *Proceedings of the 19th International Conference in Knowledge Based and Intelligent Information and Engineering Systems*, p. 1763–1771, 2015.

[103] SAROJ.; DEVRAJ. **A non-revisiting genetic algorithm for optimizing numeric mult-dimensional functions**. *International Journal on Computational Sciences and Applications (IJCSA)*, 2(1):83–93, 2012.

[104] SELASSIE, D.; HELLER, B.; HEER, J. **Divided edge bundling for directional network data**. *IEEE Transactions on Visualization Computer Graphics*, 17(12):2354–2363, 2011.

[105] SIKANSI, F. H. G. **A Similarity-based Approach to Generate Edge Bundles**. PhD thesis, Universidade de SÃ£o Paulo, 2017.

[106] SKIENA, S. **Minimum Vertex Cover. § 5.6.2 in Implementing Discrete Mathematics: Combinatorics and Graph Theory with Mathematica**. Addison-Wesley, 1990.

[107] SPENCE, R. **Information Visualization: An Introduction**. Springer, 2014.

[108] TAMASSIA, R. **Handbook of Graph Drawing and Visualization**. CRC Press, 2014.

[109] TANG, W. H.; WU, Q. H. **Condition Monitoring and Assessment of Power Transformers Using Computational Intelligence, Power Systems**. Springer-Verlag, 2011.

[110] TARAWANEH, R. M.; KELLER, P.; EBERT, A. **A general introduction to graph visualization techniques**. In: *Proceedings of the IRTG 1131 Workshop 2011*, p. 151–164, 2011.

[111] TELEA, A.; ERSOY, O.; HOOGENDORP, H.; RENIERS, D. **Comparison of node-link and hierarchical edge bundling layouts: A user study**. In: *Proceedings of the Visualization and Monitoring of Network Traffic*, número 09211 em Dagstuhl Seminar Proceedings. Schloss Dagstuhl - Leibniz-Zentrum fur Informatik, Germany, 2009.

[112] TELEA, A. C.; ERSOY, O. **Image-based edge bundles: Simplified visualization of large graphs**. *Comput. Graph. Forum*, 29(3):843–852, 2010.

[113] VAN DER ZWAN, M.; CODREANU, V.; TELEA, A. **CUBu: Universal real-time bundling for large graphs**. *IEEE Transactions on Visualization and Computer Graphics*, 22(12):2550–2563, 2016.

[114] VAN LIERE, R. **Studies in Interactive Visualization**. dissertation, University of Amsterdam, 2001.

[115] WARE, C. **Information Visualization: Perception for Design**. Morgan Kaufmann, 2012.

[116] WEBBER, R. J. **Finding the Best Viewpoint for Three-Dimensional Graph Drawings**. PhD thesis, Department of Computer Science and Software Engineering, The University of Newcastle, Australia, 1998.

[117] WEST, D. B. **Introduction to Graph Theory**. Prentice Hall, 1996.

[118] WU, J.; YU, L.; YU, H. **Texture-based edge bundling: A web-based approach for interactively visualizing large graphs**. In: *Proceedings of the IEEE International Conference on Big Data*, p. 2501–2508, 2015.

[119] WU, J.; ZENG, J.; ZHU, F.; YU, H. **MIseb: Edge bundling using moving least squares approximation**. In: *arXiv:1709.01221v2*, 2017.

[120] YAMASHITA, T.; SAGA, R. **Cluster-based edge bundling based on a line graph**. In: *Proceedings of the 12th International Joint Conference on Computer Vision, Imaging and Computer Graphics Theory and Applications*, p. 311–316. SCITEPRESS, 2017.

[121] YU, X.; GEN, M. **Introduction to Evolutionary Algorithms**. Springer-Verlag, 2010.

[122] ZACHARY, W. **An information flow model for conflict and fission in small groups**. *Journal of Anthropological Research*, 33(4):452–473, 1977.

[123] ZHONGHUA, Y.; LINGDA, W. **Research on network simplification by edge bundling**. In: *IEEE First International Conference on Data Science in Cyberspace*, p. 466–472, 2016.

[124] ZHOU, H.; XU, P.; YUAN, X.; QU, H. **Edge bundling in information visualization**. *Tsinghua Science and Technology*, 18(2):145–156, 2013.

[125] ZHOU, H.; YUAN, X.; CUI, W.; QU, H.; CHEN, B. **Energy-based hierarchical edge clustering of graphs**. In: *Proceedings of the Pacific Visualization Symposium*, p. 55–61, 2008.

[126] ZHOU, H.; YUAN, X.; QU, H.; CUI, W.; CHEN, B. **Visual clustering in parallel coordinates**. *Comput. Graph. Forum*, 27(3):1047–1054, 2008.

[127] ZHU, D.; WU, K.; GUO, D.; CHEN, Y. **Parallelized force-directed edge bundling on the gpu**. In: *Proceedings of the 11th International Symposium on Distributed Computing and Applications to Business, Engineering Science, DCABES*, p. 52–56, 2012.

[128] ZIELASKO, D.; WEYERS, B.; HENTSCHEL, B.; KUHLEN, T. W. **Interactive 3d force-directed edge bundling on clustered edges**. In: *Proceedings of the IEEE Symposium on Information Visualization - Poster*, 2014.

[129] ZIELASKO, D.; WEYERS, B.; HENTSCHEL, B.; KUHLEN, T. W. **Interactive 3d force-directed edge bundling**. *Comput. Graph. Forum*, 35(3):51–60, 2016.

APÊNDICE A

Definições e Terminologias Relacionados ao Tema

Este apêndice apresenta definições, conceitos e terminologias relacionados ao tema em estudo. Em especial serão apresentados elementos teóricos sobre teoria de grafos, otimização combinatória e modelos bioinspirados.

A.1 Teoria de Grafos

A seguir serão apresentados alguns conceitos relacionados à teoria de grafos retirados de [117] e descritos em [29].

Grafos são modelos matemáticos que representam relacionamentos entre objetos. Um grafo é definido como $G = (V, E)$, onde $V = \{v_1, v_2, ..., v_n\}$ é um conjunto finito de **vértices**, $E = \{e_1, e_2, ..., e_m\}$ um conjunto finito de **arestas**, sendo $E \subseteq V \times V$. Uma aresta é representada por um par $e = (u, v)$ de vértices não necessariamente ordenados, sendo u e v identificados como **vértices extremos** da aresta. Um grafo com um único vértice e que não possui arestas é denominado grafo **trivial**.

A **densidade** de um grafo é a medida gerada pelo número de arestas dividido pelo número de vértices. A categorização de um grafo em função do seu **tamanho** diz respeito à quantidade de elementos (vértices + arestas) que ele possui, podendo um grafo ser classificados como [72]:

- **Pequeno**: grafo com até 100 elementos;
- **Médio**: grafo com até 1000 elementos;
- **Grande**: grafo com até 10.000 elementos;
- **Esparso**: grafo com mais de 10.000 elementos.

Seja $(u, v) \in E$ uma aresta, os seus extremos u, $v \in V$ são **vértices adjacentes** e a aresta (u, v) é **incidente** à u e a v. Duas arestas são identificadas como adjacentes se possuem um vértice extremo em comum. Uma aresta (u, v) é um **laço** se $u = v$. Um **grafo direcionado** é aquele onde cada aresta é um par ordenado de vértices, ou seja, cada

aresta possui uma direção. Em um grafo direcionado, uma aresta (u,v) possui o sentido do vértice u para v, já na aresta (v,u) o sentido é de v para u.

O **grau** $gr(v)$ de um vértice v é o número de arestas incidentes à v (laços são contados duas vezes). Nos grafos direcionados distingue-se o grau de saída como o número de arestas saindo de um vértice, e o grau de entrada como sendo o número de arestas entrando em um vértice.

Duas arestas são **paralelas** se possuem o mesmo par de vértices extremos. Um grafo é classificado como **simples** caso não possua laços nem arestas paralelas. Um grafo é dito **completo** se for simples, e para todo par de vértices existir uma aresta entre eles. Um grafo completo com n vértices é denominado K_n.

Um grafo simples é **bipartido** se for possível dividir o conjunto V dos vértices em dois subconjuntos V_1 e V_2, tal que, nenhuma aresta do grafo tenha ambas extremidades em V_1 nem ambas extremidades em V_2, ou seja, o conjunto das arestas E é definido em $V_1 \times V_2$. Um grafo bipartido é completo se para todo par de vértices $v_1 \in V_1$ e $v_2 \in V_2$ existir a aresta (v_1,v_2). Um grafo bipartido completo é chamado $K_{p,q}$, sendo $|V_1| = p$ e $|V_2| = q$. A Figura A.1 mostra um exemplo de um grafo bipartido completo identificado como $K_{3,3}$.

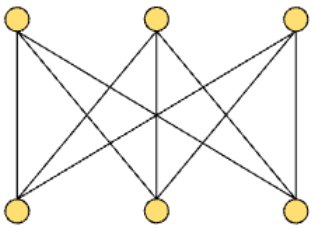


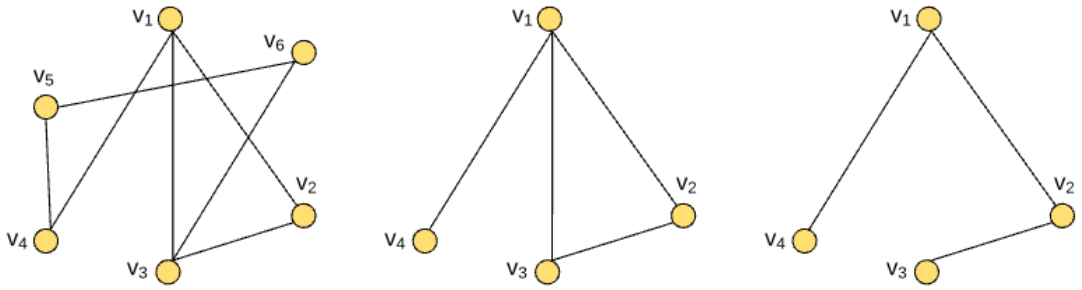
Figura A.1: *Exemplo de um grafo bipartido completo* $K_{3,3}$.

Um **passeio** é uma sequência finita de vértices e arestas alternados entre si $v_0, e_1, v_1, e_2, \ldots, v_{k-1}, e_k, v_k$, tal que $e_i = (v_{i-1}, v_i)$ para todo $1 \leq i \leq k$. O vértice v_0 é a origem do passeio e v_k o seu término. Um passeio pode ou não terminar no mesmo vértice que começou, e os vértices e arestas podem se repetir ao longo do passeio. Uma **trilha** é passeio que não passa por uma mesma aresta duas vezes, mas pode repetir vértices. Um **caminho** é um passeio cujos vértices e arestas são distintos, não se repetem. Um caminho denominado $u,v-path$ tem o primeiro vértice em u e o último em v. O primeiro e o último vértice de um caminho podem ser coincidentes. Um caminho com n vértices é denominado P_n.

Um **ciclo** é um caminho fechado em que os únicos vértices do caminho que se repetem são o primeiro e o último. Um grafo que não contém ciclos é chamado **acíclico**. Um grafo é dito **conexo** se existe um caminho de qualquer vértice para qualquer outro

vértice do grafo. Um grafo simples, conexo e acíclico é chamado de **árvore**. Um grafo **estrela** S_n de ordem n é uma árvore com n vértices, tendo um único vértice de grau $n-1$, sendo o grau de todo o restante dos vértices igual a um.

Um **subgrafo** de um grafo G é um grafo H tal que $V(H) \subseteq V(G)$ e $E(H) \subseteq E(G)$, neste caso, G é o **supergrafo** de H. Um subgrafo H de G é dito **induzido** por vértices de G quando todas as arestas $E(G)$ que possuem ambos os extremos em $V(H)$ estão em $E(H)$. Considerando o grafo original G ilustrado na Figura A.2 (a), a Figura A.2 (b) é um exemplo de um subgrafo induzido H_1 de G pois as arestas $E(H_1) = \{(v_1,v_2),(v_1,v_3),(v_1,v_4),(v_2,v_3)\}$ são as únicas arestas de G que possuem ambos os extremos em $V(H_1)$. Já o subgrafo H_2 (Figura A.2 (c)) não é induzido de G pois a arestas (v_1,v_3) que possui ambos os extremos em $V(H_2)$, não esta em $E(H_2)$.



(a) (b) (c)

Figura A.2: *Exemplo de subgrafo induzido por vértices: (a) grafo G; (b) um subgrafo induzido H_1 de G; (c) subgrafo H_2 não induzido de G.*

Uma **clique** de um grafo é um subgrafo que também é um grafo completo. Um subgrafo de G é **próprio** se for diferente de G, ou seja, $V(H) \neq V(G)$ ou $E(H) \neq E(G)$. Um subgrafo de G é **gerador** (*spanning*) se contêm todos os vértices do grafo G.

Uma **cobertura de vértices** de um grafo $G = (V,E)$ é um conjunto de vértices $T \subseteq V$ onde toda aresta do grafo tem pelo menos uma das extremidades em T. A Figura A.3 mostra uma cobertura de vértices de um grafo formado pelos vértices em coloração cinza. O tamanho da cobertura de vértices é o número de vértices que ela contém.

Em um grafo representado visualmente por um diagrama de pontos e linhas, os vértices são pontos e as arestas são segmentos de linha que conectam esses pontos (as figuras desta seção seguem este padrão). Esse estilo de desenho, eventualmente, pode conter cruzamentos de arestas. Um grafo é chamado **planar** se ele pode ser desenhado sem cruzamentos de arestas. Alguns grafos não possuem uma representação visual planar, via diagrama de pontos e linhas.

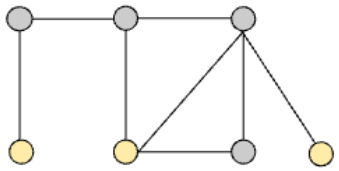


Figura A.3: *Exemplo de uma cobertura de vértices.*

A.2 Desenho de Grafos

Um **desenho de grafo**[1] é uma representação visual de um grafo, através da atribuição de posições no espaço (bi ou tri-dimensional) para os vértices e arestas. Um grafo pode ser representado visualmente de formas diferentes, e a utilidade de um desenho de grafo depende da capacidade que se tem de rapidamente e de maneira clara perceber o seu significado.

Geralmente os grafos são desenhados como diagramas de pontos e linhas. Existem diversos algoritmos para criar desenhos de um grafo através de diagramas deste tipo, e cada um deles fornece percepções diferentes de um mesmo conjunto de dados. A Figura A.4 apresenta desenhos distintos de um mesmo grafo.

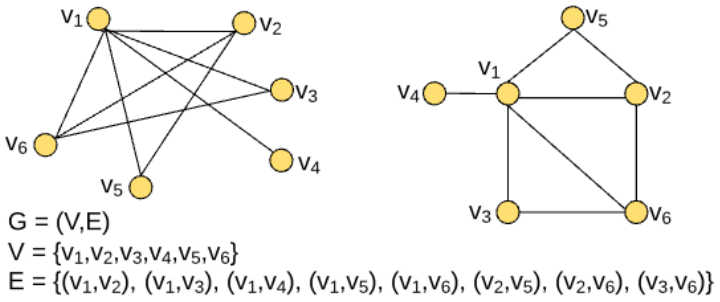


Figura A.4: *Desenhos diferentes de um mesmo grafo.*

Outras técnicas como *treemaps*, *space-filling layout* e *space-nested layout* podem ser usadas na representação visual de grafos, no entanto, estas e outras formas de visualização não serão abordadas neste trabalho, mais informações podem ser encontradas em [110]. Desta maneira, no restante deste trabalho sempre que for referenciado desenho de grafos, trata-se de um diagrama de pontos e linhas.

[1] Existe uma diferença entre a definição de visualização de grafos e desenho de grafos. O primeiro tem o objetivo de definir técnicas que ampliem a cognição facilitando a navegação e a interação com a representação visual. Já a área de desenho de grafos esta preocupada em definir métodos automáticos que estabeleçam a melhor posição dos vértices e arestas em um plano 2D ou 3D, com base em um conjunto de critérios estéticos [41]. Este trabalho tem foco em desenho de grafos.

O **estilo** usado para desenhar um grafo é um dos principais pontos que determina qual algoritmo usar para executar esta função. Abaixo pode ser visto uma lista retirada de [29], com a descrição dos principais estilos de desenho de grafos.

- **Linha reta**: cada aresta é representada como único segmento de linha reta;
- **Poligonal**: cada aresta é mapeada para um ou mais segmentos de reta;
- **Curva**: cada aresta é representada por uma curva;
- **Grade**: é um desenho onde cada vértice e curva de arestas estão em uma coordenada inteira do espaço de visualização do desenho;
- **Ortogonal**: é um desenho com linhas retas e poligonais onde cada segmento de reta é paralelo a um dos eixos do espaço de visualização do desenho; e
- **Visibilidade**: os vértices são representados por barras horizontais, se existir uma aresta entre dois vértices existirá uma linha vertical que ligará as duas barras horizontais, sem tocar qualquer outra;
- **Superfície**: os grafos são representados por superfícies 3D, como esferas, cones, cilindros, poliedros e torus.

Esses estilos podem ainda ser combinados. A Figura A.5 mostra um mesmo grafo desenhado em estilos diferentes. Além do estilo, a representação dos vértices também pode variar, eles podem ser representados como pontos, círculos, caixas, barras, etc.

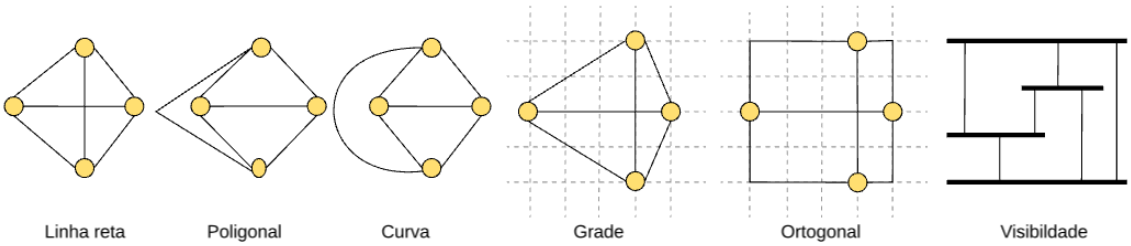


Figura A.5: *Estilos de um desenho de grafo. Adaptado de [29].*

Os **critérios estéticos** de um desenho de grafo são padrões que se seguidos produzem desenhos agradáveis ao olho humano. Esses critérios, em sua essência, são usados para determinar o quão "bom" é um desenho. Eles dependem inteiramente ou parcialmente do estilo do desenho, e os métodos para desenho de grafos tentam otimizar um ou mais destes critérios. Battista e outros [6] enumeram os principais critérios estéticos que definem um bom desenho de grafo e que se observados podem aumentar a legibilidade do desenho:

- Ausência ou redução de cruzamentos entre arestas;
- Ausência ou redução de curvas feitas por arestas;
- Redução da área total do desenho;
- Maximização do ângulo entre arestas;

- Simetria do desenho;
- Minimização do tamanho máximo de uma aresta;
- Vértices e arestas uniformemente distribuídos;
- Tamanho de arestas uniformes;
- Minimização da soma do tamanho das arestas; e
- Uniformidade na direção das arestas (para grafos direcionados).

Experimentos mostram que um dos critérios estéticos mais importantes é a quantidade de cruzamentos. Além de diminuir a poluição visual, um número reduzido de cruzamentos de arestas torna mais fácil a identificação de quais são os vértices conectados à uma determinada aresta. Segundo Kobourov, Pupyrev e Saket [58], a quantidade excessiva de cruzamentos pode provocar um impacto negativo na performance da tarefa executada pelo usuário.

Analizando os outros critérios estéticos enumerados, a redução da curvatura é importante pois o olho humano, normalmente consegue seguir de formal mais fácil uma aresta com *poucas ou nenhuma curva*. O ideal é que o desenho esteja em uma *área reduzida*, mas que mantenha uma densidade homogênea por área, com vértices e arestas uniformemente distribuídos. Ainda, as arestas, preferencialmente, devem estar distantes umas das outras, para isso é preciso *maximizar o ângulo* entre elas e assim obter a máxima resolução para um determinado tamanho de tela. A *simetria* é outro critério importante, os grafos costumam esconder algum tipo de simetria entre os seus dados, um bom algoritmo de desenho de grafos deve ser capaz de produzir desenhos que explorem e apresentem esta simetria. Os critérios descritos acima não são os únicos, existem outros, que podem ser encontrados em [6].

Entretanto, alguns destes critérios são conflitantes entre si, sendo difícil combiná-los na prática. Por exemplo, manter a uniformidade na direção das arestas em um grafo direcionado pode significar aumentar a quantidade de cruzamentos. Segundo Tarawaneh, Keller e Ebert [110], para que o processo de desenho de grafos seja satisfatório a priorização dos critérios estéticos é uma pré condição importante.

Através de experimentos, Purchase, Cohen e James [98], citado por Webber [116], definiram uma ordem de importância para alguns critérios estéticos: minimização de cruzamentos, minimização de curvas e maximização de simetria. A Figura A.6 mostra um grafo aleatório do lado esquerdo, e a direita a sua representação aplicados estes critérios.

O principal objetivo de um **algoritmo para desenho de grafos** é produzir um desenho que seja inteligível para o ser humano, ou seja, que permita a ele facilmente entender a informação codificada pelo grafo e identificar padrões escondidos. Liere [114] aponta que os algoritmos para desenho de grafos possuem dois passos: criação do *layout* e renderização. Produzir *layout* significa determinar o posicionamento dos vértices e arestas

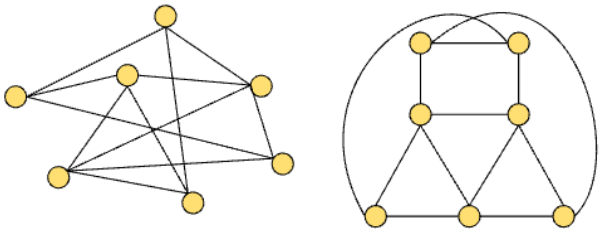


Figura A.6: *Efeito da aplicação dos critérios estéticos em um desenho de grafo. Adaptado de [9].*

no espaço, já a renderização mapeia estes vértices e arestas em objetos gráficos a serem posicionados no plano conforme informações do *layout*.

Os algoritmos para desenho de grafos devem levar em consideração os aspectos: complexidade do estilo do desenho, critérios estéticos, preservação do mapa mental, restrições impostas e eficiência computacional. Para classes particulares de grafos, como árvores, grafos planares ou grafos acíclicos direcionados existem algoritmos que produzem bons resultados, não só com relação à complexidade de tempo e espaço, mas também à aspectos relacionados aos critérios estéticos. Entretanto, grafos do domínio de aplicação do mundo real geralmente não pertencem à essas classes, o que dificulta a utilização prática de algoritmos clássicos [62].

A.3 Otimização Combinatória

Problemas de otimização são aqueles que possuem um conjunto finito (embora extremamente grande) de soluções possíveis, sendo a meta encontrar a "melhor" solução dentre todas. Esses problemas são modelados para maximizar ou minimizar uma ou mais **funções objetivo**, com base em um conjunto de requisitos, ambos relacionados às **variáveis de decisão** que obedecem às **restrições** do problema [63, 91].

Um problema de otimização, para o caso de minimização, pode ser enunciado formalmente como:

encontrar x que minimize $f(x)$

sujeito a

$g_i(x) \geq 0$, sendo $i = 1, \ldots, m$

$h_j(x) = 0$, sendo $j = 1, \ldots, p$

onde $x \in \mathbb{R}^n$ são variáveis do problema, f é a função objetivo, g_i e h_j são restrições.

A **instância** de um problema de otimização é um par (S, f), onde S é o conjunto de todas as soluções possíveis (**espaço de busca**) e f é a função objetivo. Considerando o mapeamento $f : S \longleftrightarrow \mathbb{R}$, o objetivo é encontrar um $x \in S$ tal que $f(x) \leq f(y)$ para todo $y \in S$. O ponto x é chamado **ótimo global**, ou simplesmente, solução ótima, e será o menor valor possível para a função objetivo [91].

A magnitude do espaço de busca é um dos principais obstáculos para os algoritmos de otimização. Geralmente, quando o problema é muito complexo o espaço de busca é tão grande que, computacionalmente, é impraticável enumerar todas as soluções possíveis e escolher a melhor entre elas. Neste caso, encontrar uma solução local ótima pode ser aceitável. Dependendo da sua qualidade a solução local ótima pode ser considerada como uma solução "boa" o suficiente para o problema investigado.

O **ótimo local** de um problema é definido como uma solução $x^* \in S$ tal que $f(x^*) \leq f(x)$ (para o caso de minimização) para todo $x \in N(x^*)$, sendo $N(x^*)$ a vizinhança de x^*. Uma **vizinhança** é um mapeamento $N : S \rightarrow 2^S$ que leva as soluções de S em um subconjunto deste mesmo conjunto de soluções, assim $N(x) = x_1, x_2, \ldots, x_k$ soluções vizinhas de x.

Os problemas de otimização são divididos em três categorias: aqueles que lidam com variáveis contínuas, aqueles que lidam com variáveis discretas e os que possuem variáveis inteiras e contínuas. Os problemas cujas variáveis assumem valores contínuos buscam como solução números reais ou funções, e são identificados como problemas de otimização contínua. Já os problemas cujas variáveis assumem valores discretos procuram por um inteiro, um conjunto, uma permutação ou até um grafo, estes problemas são chamados problemas de otimização combinatória. O último caso é definido como otimização mista. Os problemas alvo de estudo nesta tese foram modelados e resolvidos via técnicas de otimização combinatória.

O passo seguinte à formulação matemática de um problema de otimização é o uso de um algoritmo que resolva este modelo. Os métodos para resolução de problemas de otimização podem ser classificados em: exatos, aproximativos e heurísticas [10, 12, 71].

Os métodos **exatos** possuem a propriedade de terminar depois de um número finito de passos e garantir encontrar a solução ótima. No entanto, para problemas complexos, mesmo quando é conhecido um algoritmo exato que o resolva, ainda assim, o seu uso pode ser impraticável, pois, em geral, eles levam um tempo exponencial de iterações. Por este motivo, algoritmos exatos são normalmente aplicados à problemas de baixa complexidade.

Os algoritmos **aproximativos** também executam em número finito de passos, mas entregam uma solução sub ótima que possui um certo fator de aproximação $c > 1$ da solução ótima. Por exemplo, seja $OPT(i)$ o valor da solução ótima para uma entrada i, para um problema de minimização, um método aproximativo entrega uma solução $x \in S$

com:

$$OPT(i) \leq f(x) \leq c \times OPT(i) \tag{A-1}$$

Os métodos **heurísticos** são usados para resolver problemas específicos. Em geral, são eficientes e rápidos, e também entregam uma solução sub ótima, mas não dão garantia da sua qualidade. As heurísticas encontram uma solução aceitável que, em comparação com a solução produzida pelos algoritmos aproximativos, ainda pode estar longe da solução ótima. São algoritmos dependentes da instância de entrada, e por isso podem não entregar uma solução válida para todas as instâncias.

Os problemas investigados nesta tese são considerados de complexidade elevada, devido a esta característica utilizou-se uma técnica heurística na resolução destes problemas. Em especial foi usada a técnica de computação evolucionária, uma heurística baseada na teoria evolucionária, que até o momento não tinha sido testada como abordagem de união de arestas em feixes.

A.4 Computação Evolucionária

A computação evolucionária é muito usada na resolução de problemas complexos de otimização. Os algoritmos evolutivos são inspirados na teoria da evolução de Darwin através da simulação de processos naturais de sobrevivência e reprodução de indivíduos. São métodos de busca estocástica que procuram por uma "boa solução" ao invés da solução global ótima, fazendo com que o tempo de execução seja menor comparado às técnicas exatas.

Uma solução candidata é identificada como um indivíduo e representa um ponto no espaço de busca do problema. Em sua essência, esses algoritmos geram novos indivíduos através de operadores genéticos que simulam o processo de reprodução. Durante o processo de evolução os indivíduos considerados inaptos são descartados e os melhores são mantidos. Desta forma, os bons indivíduos tem uma probabilidade maior de se reproduzir e transmitir suas características genéticas às novas gerações. O sucesso dos algoritmos evolucionários esta inerentemente associado à sua característica aleatória [11, 60].

Existem diversas técnicas de computação evolucionária, conhecidas como metaheurísticas evolucionárias, entre elas se destacam: algoritmos genéticos [38], programação genética [5], estratégias evolutivas [121] e programação evolutiva [121]. Todas as abordagens evolucionárias compartilham similaridade na forma de execução da busca e possuem a mesma estrutura básica composta dos seguintes elementos:

- **Indivíduo**: uma possível solução do problema;

- **População**: um conjunto de soluções;
- **Aptidão ou *fitness***: quantificação da qualidade do indivíduo;
- **Operadores genéticos**: operações a serem aplicadas à indivíduos para promover a diversificação; e
- **Parâmetros de controle**: valores que controlam a forma como os operadores são executados.

O **processo evolutivo** é cíclico e começa criando uma população inicial de indivíduos. A cada iteração, identificada como uma **geração**, os indivíduos são avaliados e associados à um valor que determina a sua qualidade (aptidão). Baseado neste valor os melhores indivíduos são selecionados para serem recombinados e gerar os indivíduos da próxima geração. Novos indivíduos substituem antigos baseado no valor de aptidão de cada um. O ciclo continua até que a melhor solução surja ou um critério de parada seja alcançado [3, 21, 79]. O Algoritmo A.1 mostra a estrutura básica de um algoritmo evolucionário.

A terminologia da computação evolucionária é compartilhada pelas diversas técnicas evolucionárias. A seguir serão apresentados os principais conceitos dessa área.

Os objetos/valores que são soluções do problema são identificados como **fenótipos**. A forma de codificação de uma solução do problema no algoritmo é identificado como **genótipo**. O mapeamento do fenótipo em genótipo é a chamada **representação do indivíduo**. Fenótipo e genótipo podem ser muito diferentes, por exemplo, enquanto o primeiro poderia ser um número real, o segundo pode ser uma composição de bits.

Algoritmo A.1: Algoritmo evolucionário básico

$t = 0$;
inicializar $P(t)$;
avaliar $P(t)$;
enquanto *NÃO condição de parada* **faça**
    $t = t + 1$;
    selecionar $P(t)$ de $P(t-1)$;
    recombinar $P(t)$;
    avaliar $P(t)$;

Neste trabalho será usada a expressão **indivíduo** para representar uma solução, independente de estar sendo referenciado um genótipo ou fenótipo. Os indivíduos são compostos por elementos identificados pelos sinônimos: variável, locus, **posição** ou gene. Um elemento de uma posição é identificado como **valor** ou alelo.

Dentre as possibilidades de representação de um indivíduo são mais usadas a representação binária, em árvores, valores, por permutação ou máquinas de estado finito. Essas não são as únicas formas de representação, cada problema pode envolver uma forma própria de codificação do indivíduo, que melhor se adapta aos requisitos do problema. A escolha da representação de um indivíduo é um dos pontos centrais na construção de algoritmos evolucionários. A escolha inadequada de um padrão de representação pode acarretar resultados insatisfatórios no processo de busca da solução [73].

A **população** é o conjunto de soluções candidatas (indivíduos), sendo que o **espaço de busca** é onde estão todas as possíveis soluções do problema. A **população inicial** geralmente é criada de forma aleatória, baseada em uma solução já existente ou através de uma heurística relacionada ao domínio do problema, e deve garantir a diversidade dos indivíduos. A **diversidade** de uma população mede a quantidade de soluções diferentes presentes na população corrente. Quando apenas os melhores indivíduos se reproduzem a população tende a perder diversidade e ser composta apenas por indivíduos muito semelhantes, este efeito é denominado **convergência genética**. A diversidade da população tem um papel importante, pois é ela que ajuda a evitar a **convergência prematura** para um ótimo local. A convergência prematura ocorre quando o algoritmo não produz mais indivíduos que possuam uma qualidade superior a da melhor solução. Neste caso é dito que o algoritmo convergiu para um ótimo local [32]. Se a população inicial é bem escolhida, o algoritmo evolutivo tem uma probabilidade maior de gerar uma boa solução no final do ciclo evolucionário, dependendo, da capacidade do processo de recombinação de introduzir e/ou manter a diversidade populacional.

Cada indivíduo gerado pelo algoritmo evolutivo é avaliado, e a sua qualidade é quantificada usando uma **função de aptidão** que é específica do problema. Essa qualidade recebe o nome de aptidão (do inglês *fitness*). Geralmente o valor de aptidão de um indivíduo é atribuído através de uma função matemática bem definida ou de um método que transforma a função objetivo do problema de otimização em um valor numérico [100].

O **mecanismo de seleção** é responsável por direcionar a busca para regiões do espaço de busca que possuam soluções viáveis. Durante a seleção os melhores indivíduos (com valor de aptidão alta) são identificados, classificados e selecionados. A seleção ocorre a nível de população, onde os indivíduos selecionados são usados para cruzamento ou como sobreviventes para a próxima geração. Os métodos de seleção podem ser determinísticos ou probabilísticos [73]. Existem diversas abordagens de seleção como, por exemplo: roleta, *ranking*, ponto de corte e torneio.

O método de **seleção proporcional** aplica um esquema probabilístico, onde os indivíduos são selecionados de acordo com a probabilidade de seleção que é diretamente proporcional à função objetivo. A probabilidade $p(s_i)$ de selecionar um indivíduo s_i é calculado como:

$$p(s_i) = \frac{f(s_i)}{\sum_{j=1}^{k} f(s_j)}, \text{onde } f(s_i) \text{ é a função de aptidão do indivíduo } s_i \qquad \text{(A-2)}$$

Um exemplo de seleção proporcional é o método de **seleção por roleta** que faz uma analogia ao jogo da roleta. Cada indivíduo ocupa uma fatia da roleta baseado no valor da sua aptidão: $f(s_i)$ é o i-ésimo elemento da aptidão total $\sum_{i=1}^{n} f_i$, onde n é o tamanho da população. A probabilidade do indivíduo s_i ser selecionado é $p(s_i)$. A escolha aleatória de um valor $r \in [0,1]$, representando um pedaço da roleta, indicará a porcentagem correspondente a um indivíduo. Se r estiver entre a probabilidade acumulada do i-ésimo e $(i+1)$-indivíduo, então o indivíduo i é selecionado. O indivíduo com a melhor aptidão tem a probabilidade de ser selecionado com mais frequência. Este método requer valores positivos de aptidão, por isso, normalmente, é utilizada a versão escalonada do valor de aptidão para comparar indivíduos. Evita-se dessa forma, que uma super solução ocupe a maior parte da roleta e que em populações com soluções com aptidão semelhantes a probabilidade de escolha seja igual para todas as soluções.

Uma outra alternativa de seleção é a **seleção por torneio** onde a população é dividida em pequenos grupos aleatórios. O processo é simples, sorteia-se aleatoriamente um número k de indivíduos (dimensão do torneio), sendo $k \geq 2$, e dentre os indivíduos sorteados, é selecionado aquele que possui maior valor de aptidão. O torneio é justamente a comparação da aptidão entre os indivíduos do grupo. O número de torneios é o número de indivíduos a serem selecionados. Por exemplo, para a seleção por torneio de dois indivíduos, pode-se fazer dois torneios com grupos de k indivíduos cada. Este método possui a vantagem de não gerar super indivíduos, uma vez que aquele com maior aptidão tem a mesma probabilidade de ser selecionado do que qualquer outro.

Após os indivíduos serem selecionados são aplicados os **operadores genéticos** que efetuam a recombinação de "material genético". São os operadores genéticos que simulam a sobrevivência das características dos indivíduos que formarão uma nova geração. As diversas variações de uma solução evolucionária podem criar novos operadores que auxiliem o processo de convergência para a solução ótima, mas a característica principal desses operadores, independente da sua forma básica de funcionamento, é a presença de elementos estocásticos em sua operação. Existem dois operadores genéticos básicos: cruzamento e mutação.

Através do operador de **cruzamento**, partes da estrutura dos indivíduos são combinadas gerando novas soluções. Geralmente são cruzados apenas dois indivíduos por vez produzindo dois novos que podem ou não ser inseridos na nova população. Os indivíduos selecionados são identificados como **pais** e os gerados como **filhos**. Algoritmos evolucionários diferentes podem implementar este operador de forma diferente, ou até

nem chegar a usá-lo, por exemplo, a programação evolucionária possui apenas mutação. De acordo com Grefenstette [39], o operador de cruzamento tem duas funções básicas, a primeira é inserir novos pontos de teste dentro de regiões do espaço de busca, e segundo inserir novos indivíduos representantes de sub espaços ainda não avaliados. Dentre as técnicas de cruzamento destacam-se: cruzamento de um ponto e cruzamento de dois pontos.

No **cruzamento de um ponto de corte** são gerados dois indivíduos filhos a partir de dois pais. Um ponto de corte é selecionado na estrutura dos pais. Um novo indivíduo é gerado pela troca de informação do código genético que aparece antes do ponto de corte no primeiro pai com a parte que esta após o ponto de conte no segundo indivíduo pai. O outro filho é gerado unindo a segunda parte do primeiro pai com a primeira parte do segundo pai. No **cruzamento de dois pontos de corte** o processo é similar, exceto pelo fato de haver dois pontos de corte em cada indivíduo.

Para evitar uma convergência prematura pode ser usado o operador de **mutação**. Esse operador possui a função de possibilitar uma varredura maior do espaço de busca e prevenir que a busca fique estagnada em sub-regiões do espaço. A mutação é a transformação de um único indivíduo através da alteração de parte dos valores da sua "estrutura genética". Essa mutação genética faz com que uma área diferente do espaço de solução do problema seja avaliada. Na maioria das vezes, é aplicada nos filhos gerados pelo operador de cruzamento.

A mutação pode ser guiada por uma heurística ou feita complemente "as cegas". São exemplos de mudanças que poderiam ser feitas pelo operador de mutação no valor das posições do indivíduo: a mudança aleatória de valores, inversão, troca de posição entre valores, duplicação de valores ou grupo de valores ou deleção. Estes são apenas alguns exemplos de possíveis rotinas de mutação, e é óbvio que as alternativas não se esgotam nessas, e dependendo do domínio do problema soluções mais adaptadas ao problema são necessárias.

Após um número de iterações uma nova geração surge através da **seleção de indivíduos sobreviventes**, sendo que a cada geração a população se altera, de forma geral ou gradual. A população pode ser alterada completamente somente após todo ciclo iterativo terminar, ou um conjunto de novos indivíduos é inserido gradualmente na população ao longo do ciclo. Um esquema eficiênte de **sobrevivência** (substituição) de indivíduos entre as gerações deve ser utilizado para garantir a convergência para a melhor solução. A decisão de quais indivíduos passarão para a próxima geração é feita geralmente com base na função de aptidão selecionando de preferência indivíduos com valores altos. Por exemplo, a **seleção determinística** seleciona os melhores indivíduos dentre os novos gerados para substituir os indivíduos de pior aptidão, já na **seleção elitista**, os n melhores indivíduos da população são sempre inseridos na próxima geração. No entanto, outras

abordagens que não privilegiem os melhores indivíduos poderiam ser adotadas como sempre inserir os filhos na nova geração em substituição dos pais, sem comparação de qualidade.

Em algum momento o processo evolucionário deve parar. O **critério de parada** mais comum é o número máximo de gerações, o algoritmo executa até que este limite seja alcançado. O critério de convergência também pode ser usado como critério de parada, neste caso, todos os indivíduos da população convergem para uma única solução. Outro critério de parada é o cálculo da soma do desvio entre os indivíduos, quando esta soma se tornar menor do que um limite o processo evolutivo para. Pode ser usado como critério de parada também o fato de nenhuma melhora ser obtida na solução ótima produzida ao longo das gerações ou ainda, ser usado um limite, uma medida de avaliação arbitrária [109]. As diversas técnicas não precisam ser usadas de forma isolada, mas é possível usar vários critérios em conjunto como forma de terminar o algoritmo.

A performance de um algoritmo evolutivo é também dependente da escolha correta dos chamados **parâmetros de controle**. Esta atividade não é tão simples e requer a execução de muitos testes. Estes parâmetros geralmente são selecionados antes do processo evolutivo começar. Alguns parâmetros considerados importantes para a maioria dos algoritmos evolutivos, sem perda de generalidade, são:

- O **tamanho da população** é a quantidade de elementos que a população terá em cada geração. Geralmente esse valor é fixo, mas existem propostas de variação desse parâmetro a cada ciclo do algoritmo. O tamanho da população é um parâmetro importante que influencia a qualidade da solução e o tempo de busca. Uma população pequena poderá guiar o algoritmo para uma solução com qualidade ruim pois apenas uma parte do espaço de busca será explorado. Enquanto, uma população muito grande exigirá um tempo maior de execução [100];
- A **taxa de cruzamento** controla a frequência com que o operador de cruzamento é aplicado a um par de indivíduos. Segundo Grefenstette [39], a taxa de cruzamento p_c alta introduz rapidamente novos indivíduos na população, no entanto, se esta taxa for alta demais, indivíduos com uma qualidade alta podem ser descartados muito cedo. E se for muito baixa, o algoritmo pode estagnar em uma região do espaço de busca fazendo com que os filhos sejam simples cópias dos pais;
- A **taxa de mutação** controla a probabilidade de mudança aleatória em um indivíduo. Segundo Linden [67], uma taxa de mutação p_m alta leva a uma busca essencialmente aleatória, permitindo que boas soluções sejam descartadas ao longo do processo, enquanto, uma taxa muito baixa faz com que o algoritmo evolutivo convirja muito rápido para um ótimo local; e
- A **estratégia de seleção** é a definição de qual técnica usar na seleção de indivíduos. Por exemplo, pode ser usado uma abordagem pura ou elitista.

Estes não são os únicos parâmetros. Grefenstette[39] avalia seis parâmetros (os citados neste trabalho também fazem parte da lista) considerados fundamentais para a performance e eficiência do processo evolutivo. Nesta tese apenas os parâmetros da lista anterior são observados durante o processo evolutivo.

Nos algoritmos evolucionários a manipulação das restrições do problema não é feita de forma direta. Existem algumas abordagens que buscam garantir a **satisfação das restrições** impostas pela definição do problema, entre elas é possível destacar [18, 59, 78, 79]:

- Utilizar uma função de penalidade;
- Preservar soluções viáveis;
- Recuperar soluções inviáveis;
- Separar objetivos e restrições; e
- Utilizar método híbridos.

O uso de **funções de penalidade** é a forma mais comum de assegurar a satisfação das restrições do problema. Valores de penalidade altos desencorajam o algoritmo a fazer buscas em regiões inviáveis, enquanto penalidades baixas fazem com que o método gaste muito tempo avaliando soluções inviáveis. A função de penalidade geralmente mede a distância entre a solução avaliada e a região de soluções viáveis, ou o esforço para recuperar a solução. As penalidades podem ser estáticas (não mudam ao longo do processo), dinâmicas (mudam ao longo do processo), adaptativas (se adaptam ao retorno da busca), baseadas em funções "*annealing*", co-evolucionário (baseado em mais de uma população) ou baseadas na morte das soluções inviáveis.

A segunda abordagem tem como objetivo a **preservação das soluções viáveis** que surgem ao longo do ciclo evolutivo. Isso pode ser feito de forma direta mantendo estas soluções na população, através do descarte de soluções inviáveis, a utilização de chaves aleatórias, a busca por soluções somente em áreas próximas a soluções viáveis ou até através do uso de indicações diretas de como construir uma solução viável.

A **recuperação de soluções inviáveis** estabelece métodos para modificar o indivíduo considerado como inviável para que ele não mais viole uma restrição do problema. O novo indivíduo gerado pode ou não ser inserido na população.

Na **separação dos objetivos e restrições** estes dois elementos são avaliados de forma independente durante o ciclo evolutivo. Pode-se, por exemplo, considerar o problema como sendo uma otimização multi objetivo ou usar mais de uma população, uma para avaliar restrições e outra para avaliar objetivos.

Por fim, usar uma **abordagem híbrida**, neste caso os algoritmos evolucionários não são puros e outras técnicas são utilizadas para ajudar a avaliação das restrições do problema.

APÊNDICE B

Visualização das Melhores Soluções

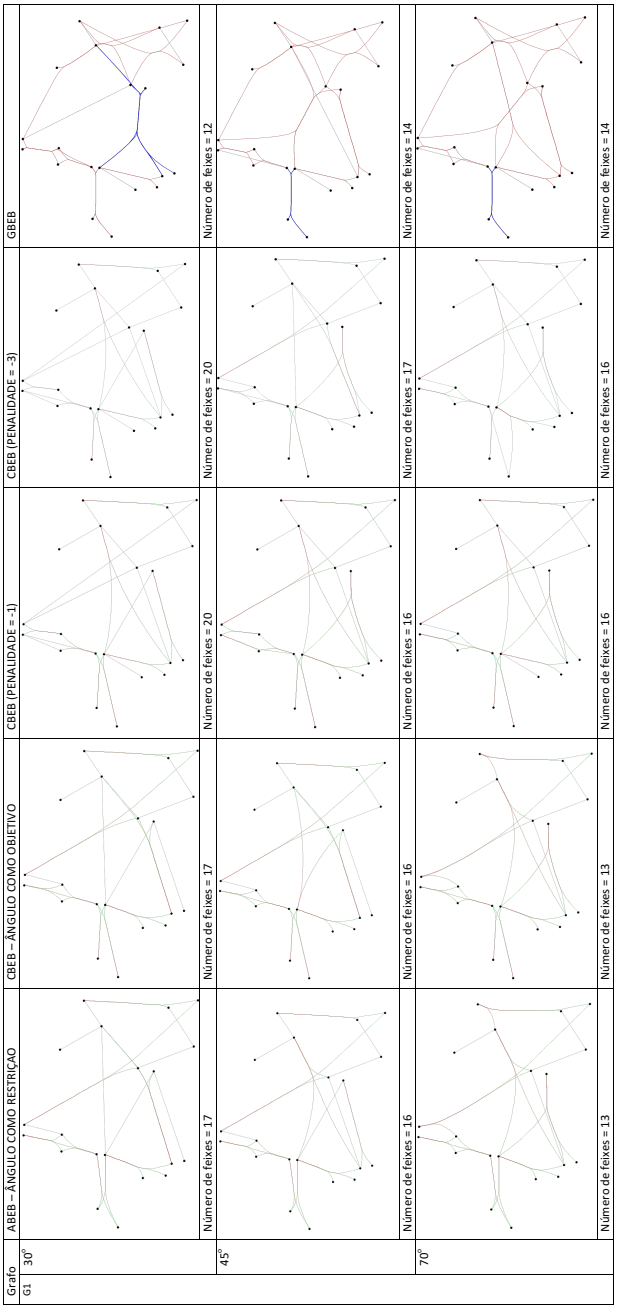


Figura B.1: *Visualizações dos resultados para o grafo Sintético (G1).*

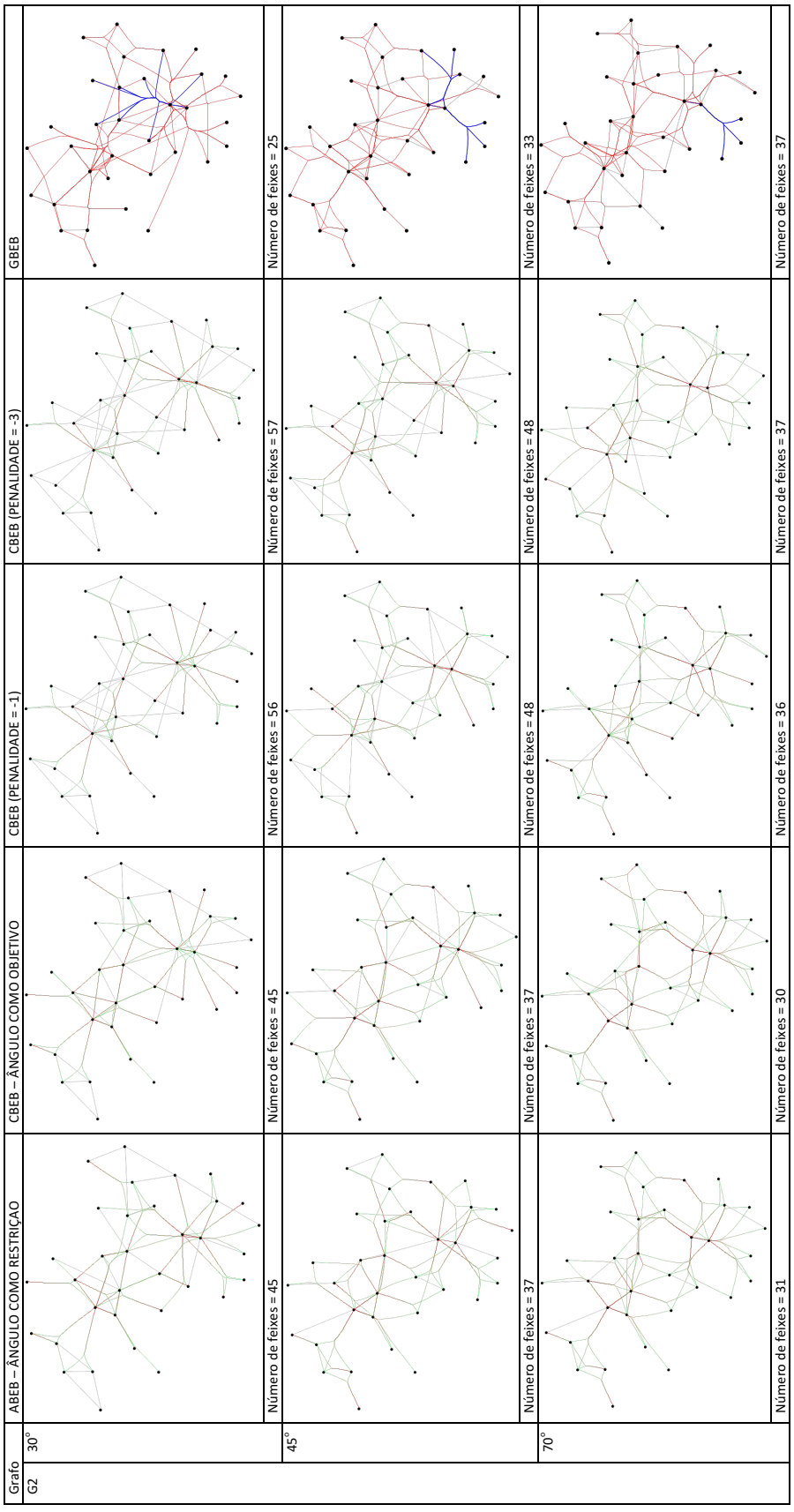


Figura B.2: *Visualizações dos resultados para o grafo Zachary Club (G2).*

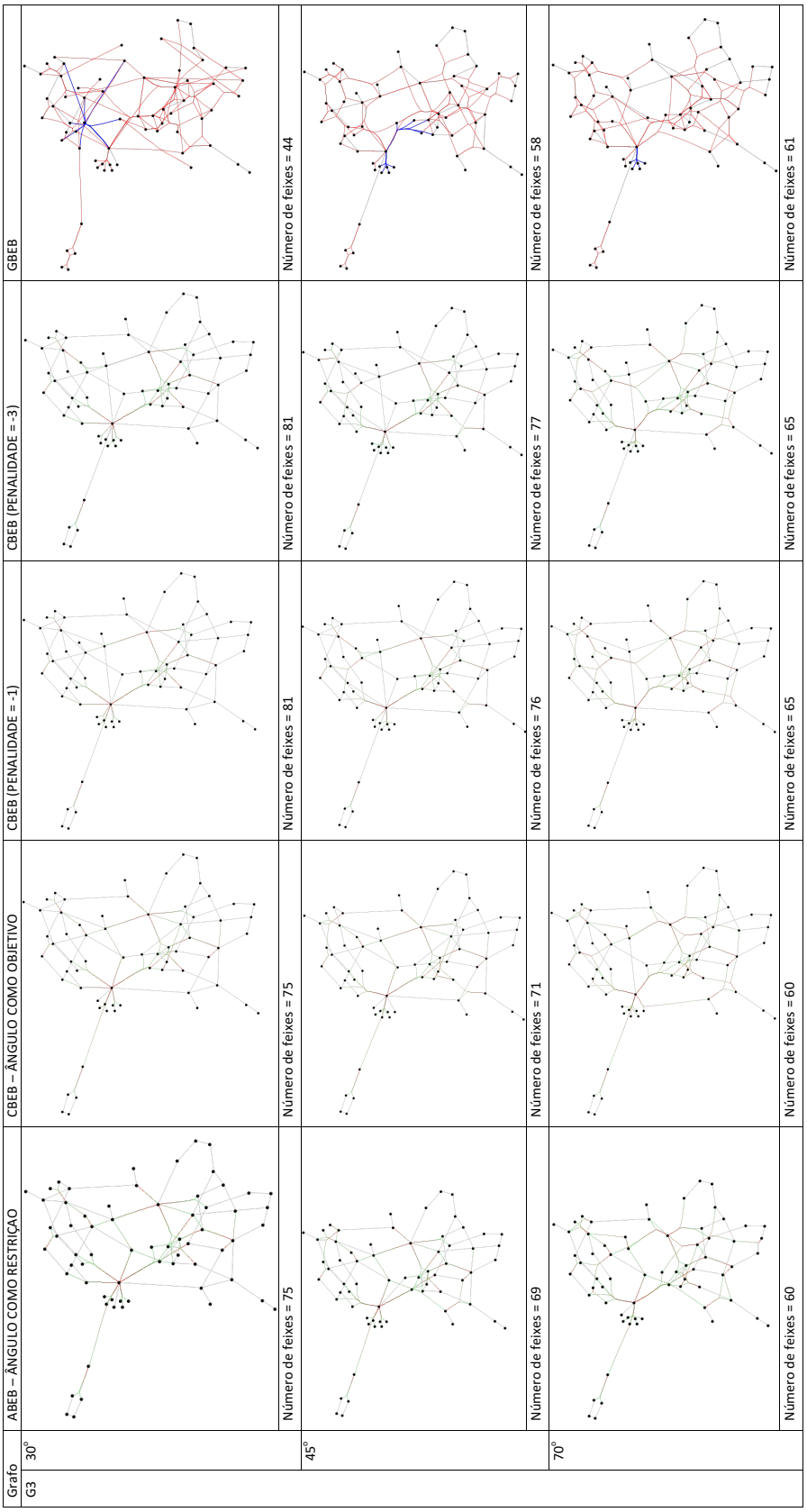


Figura B.3: *Visualizações dos resultados para o grafo Planar GD2015 (G3).*

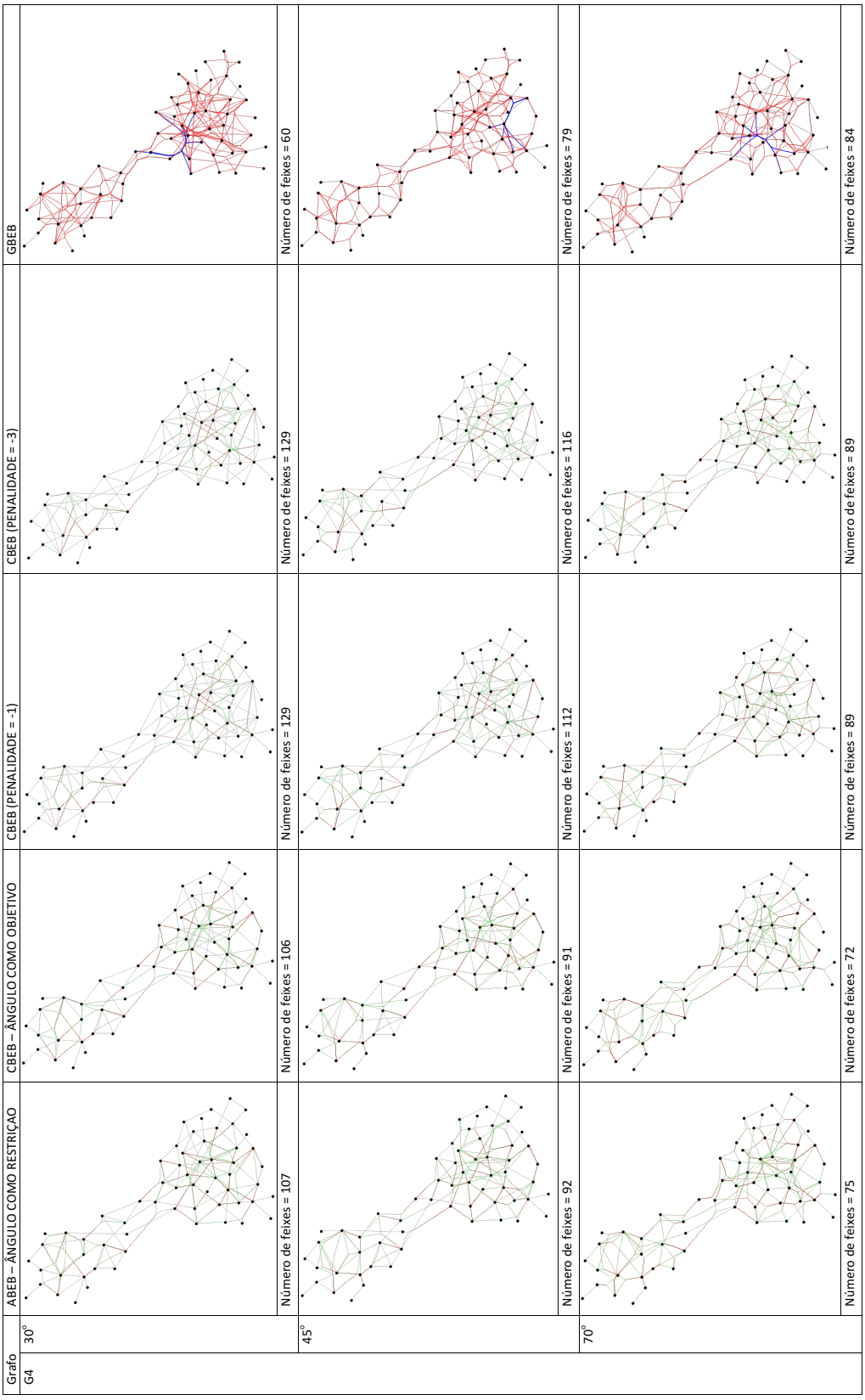


Figura B.4: *Visualizações dos resultados para o grafo Dolphin (G4).*

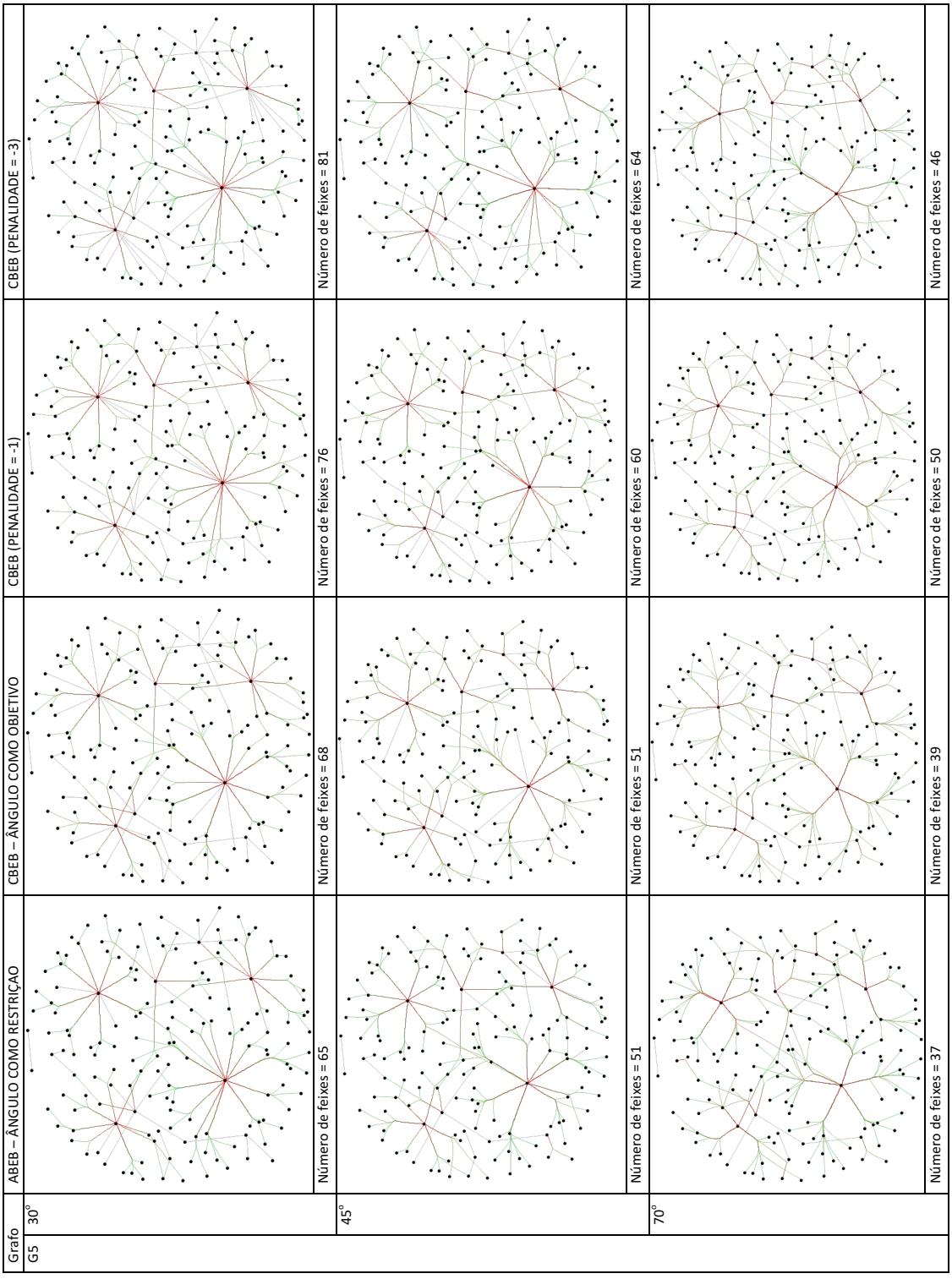


Figura B.5: *Visualizações dos resultados para o grafo MovieLens (G5).*

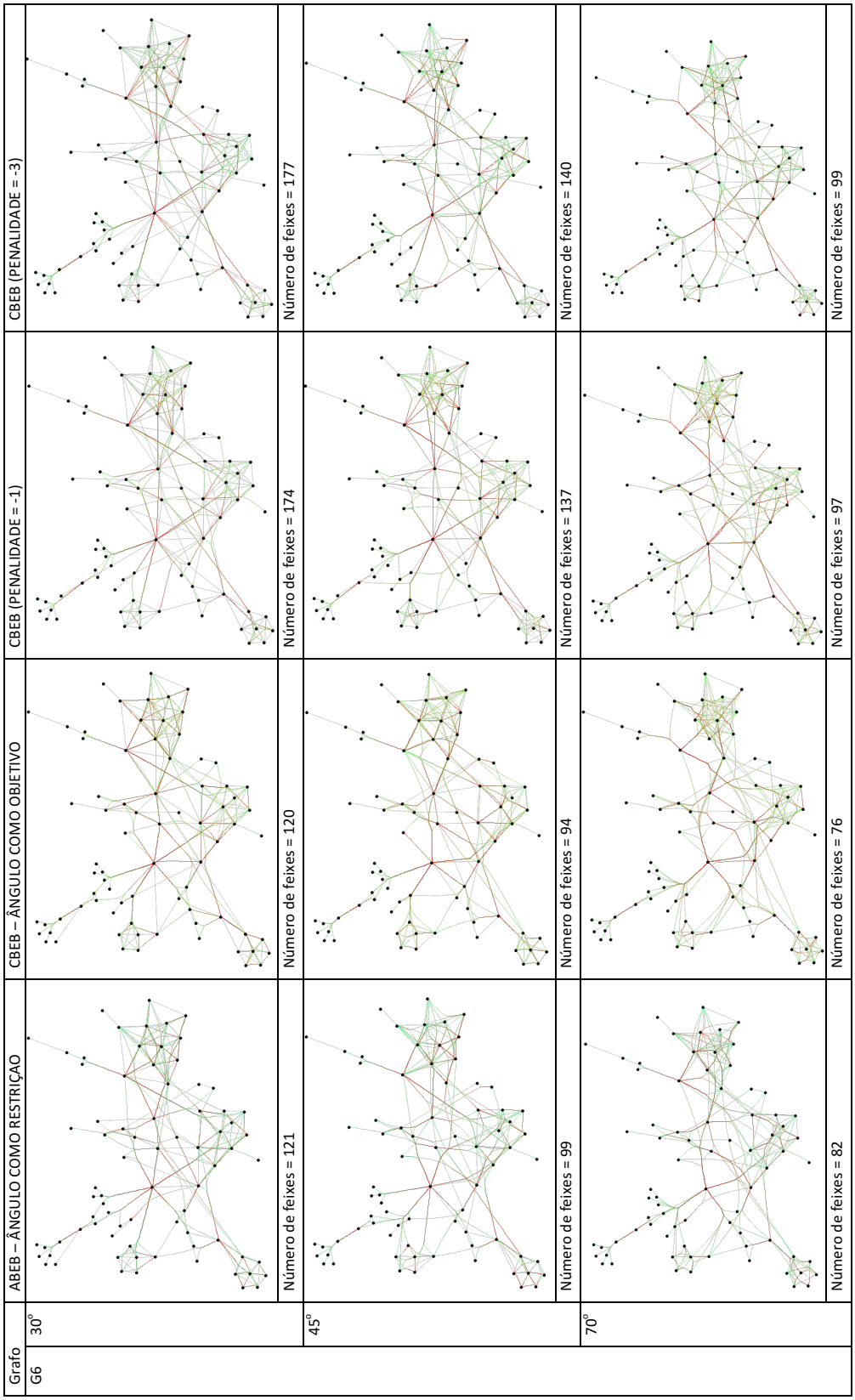


Figura B.6: *Visualizações dos resultados para o grafo LesMiserables (G6).*

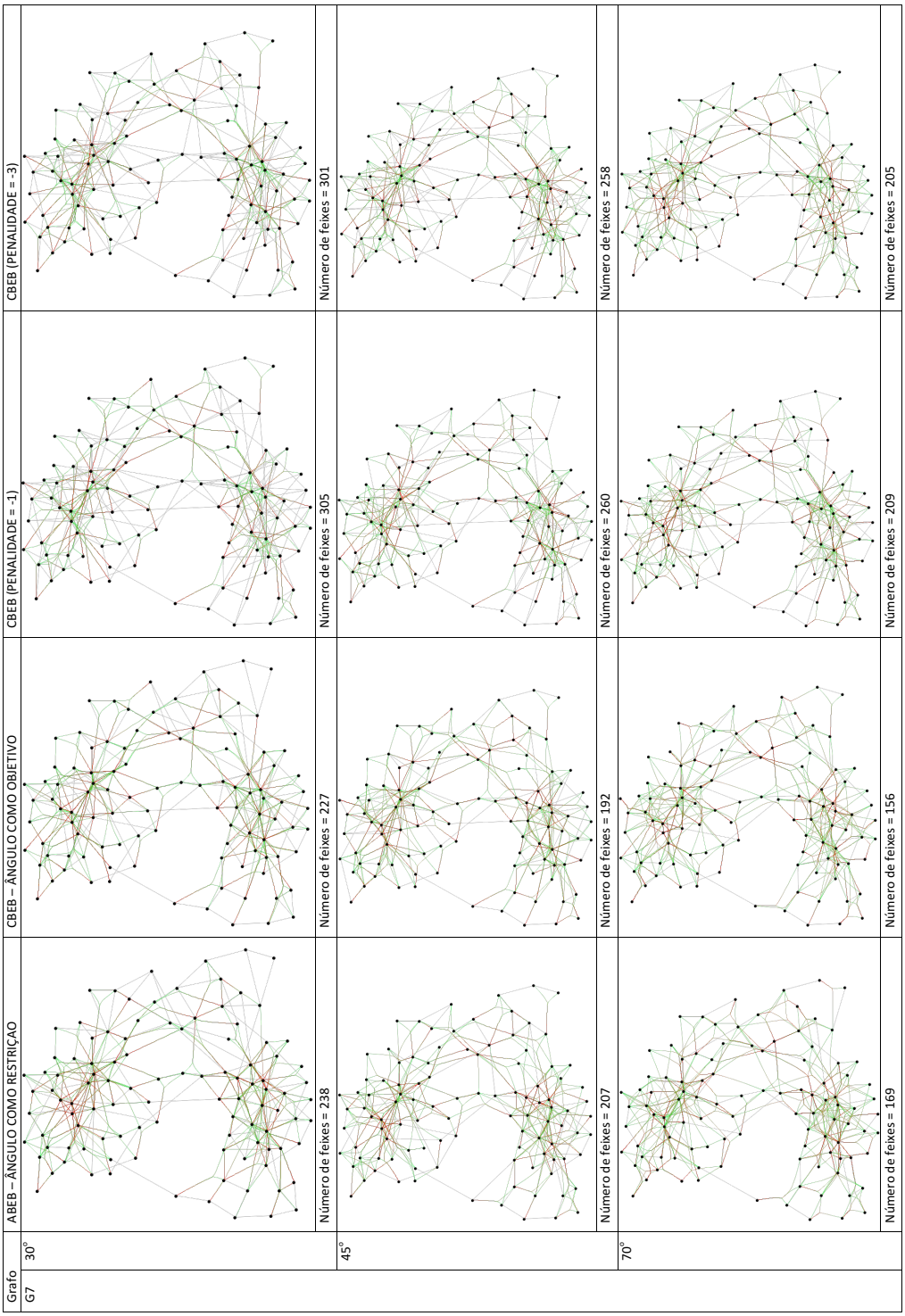


Figura B.7: *Visualizações dos resultados para o grafo Book USPolitics (G7).*

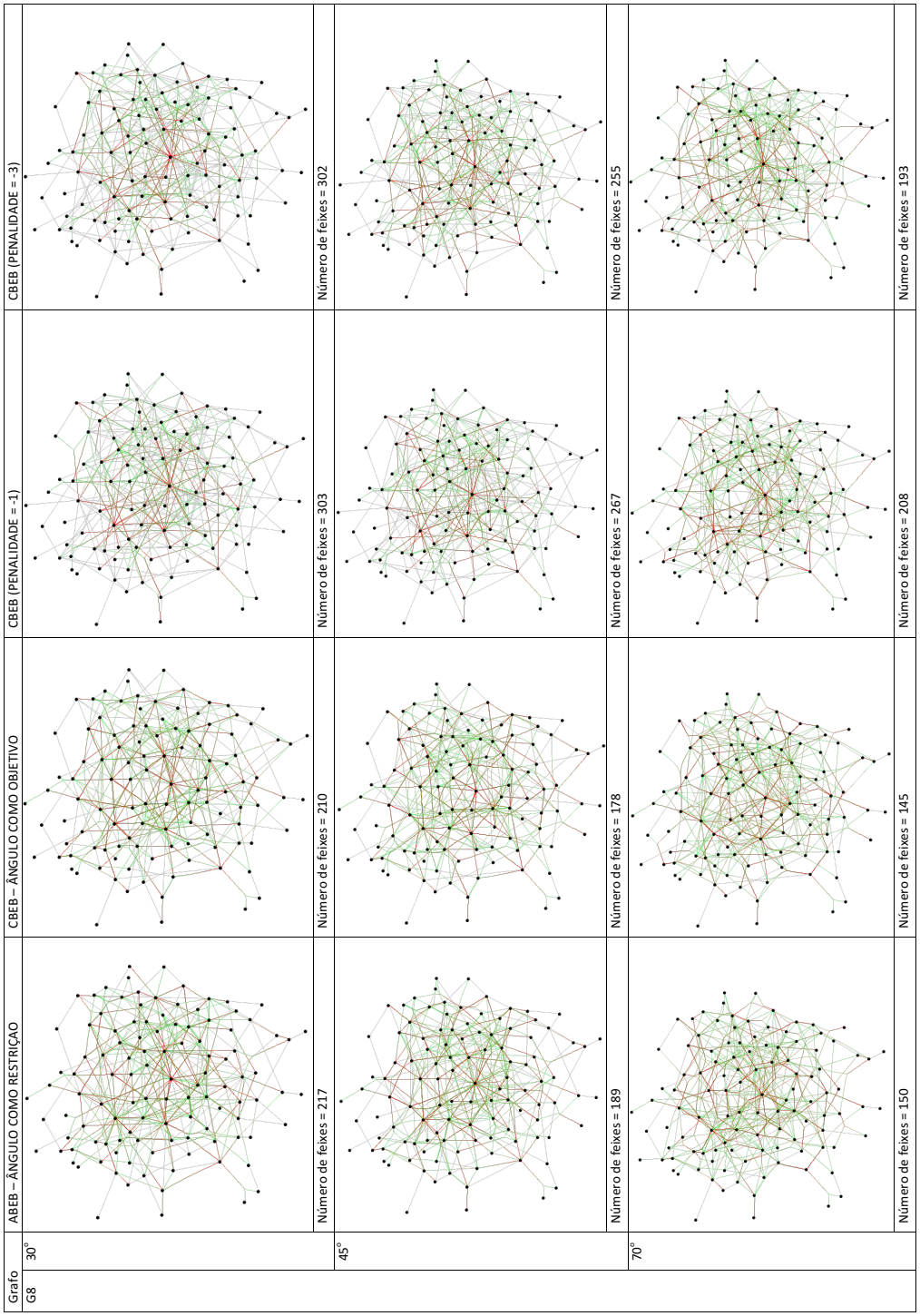


Figura B.8: *Visualizações dos resultados para o grafo Word Adjacences (G8).*

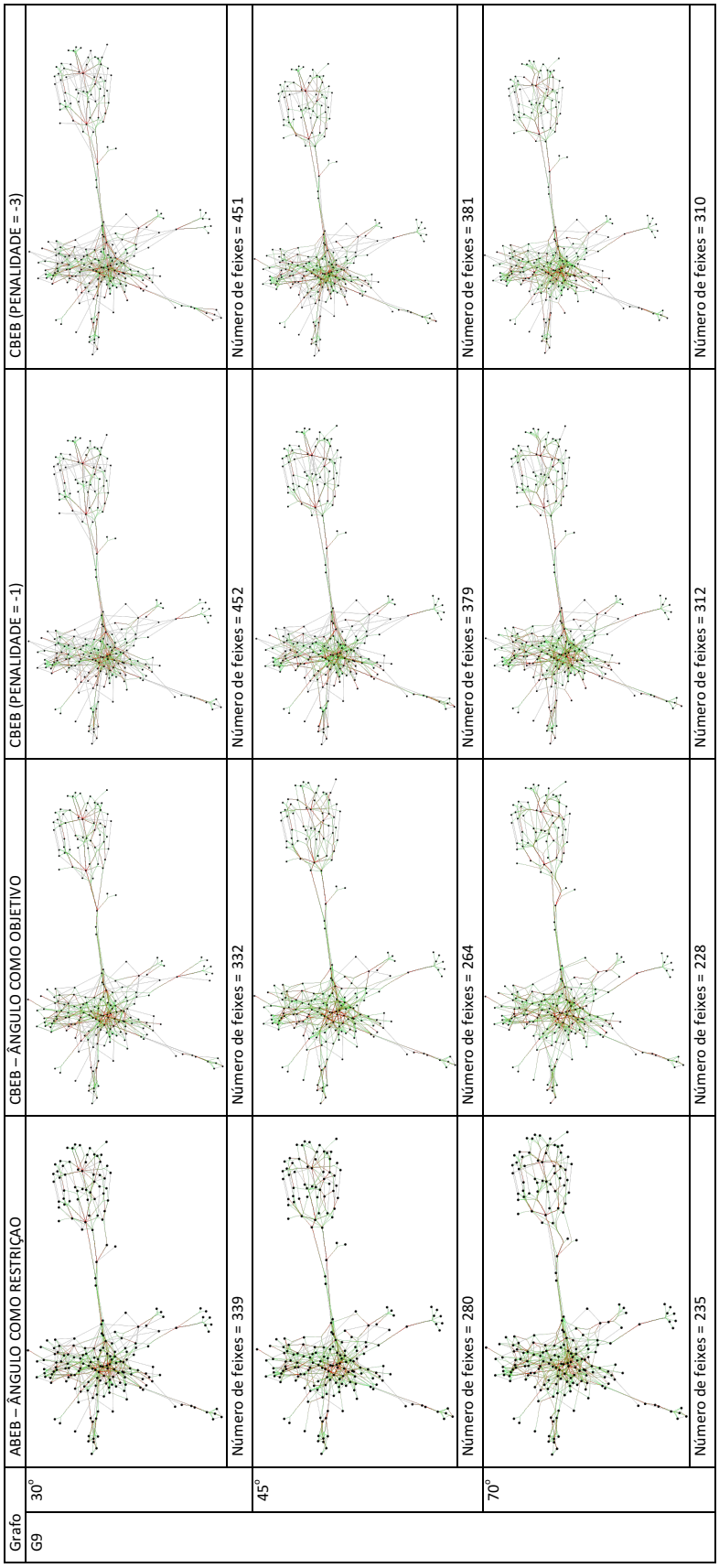


Figura B.9: *Visualizações dos resultados para o grafo Flate (G9).*

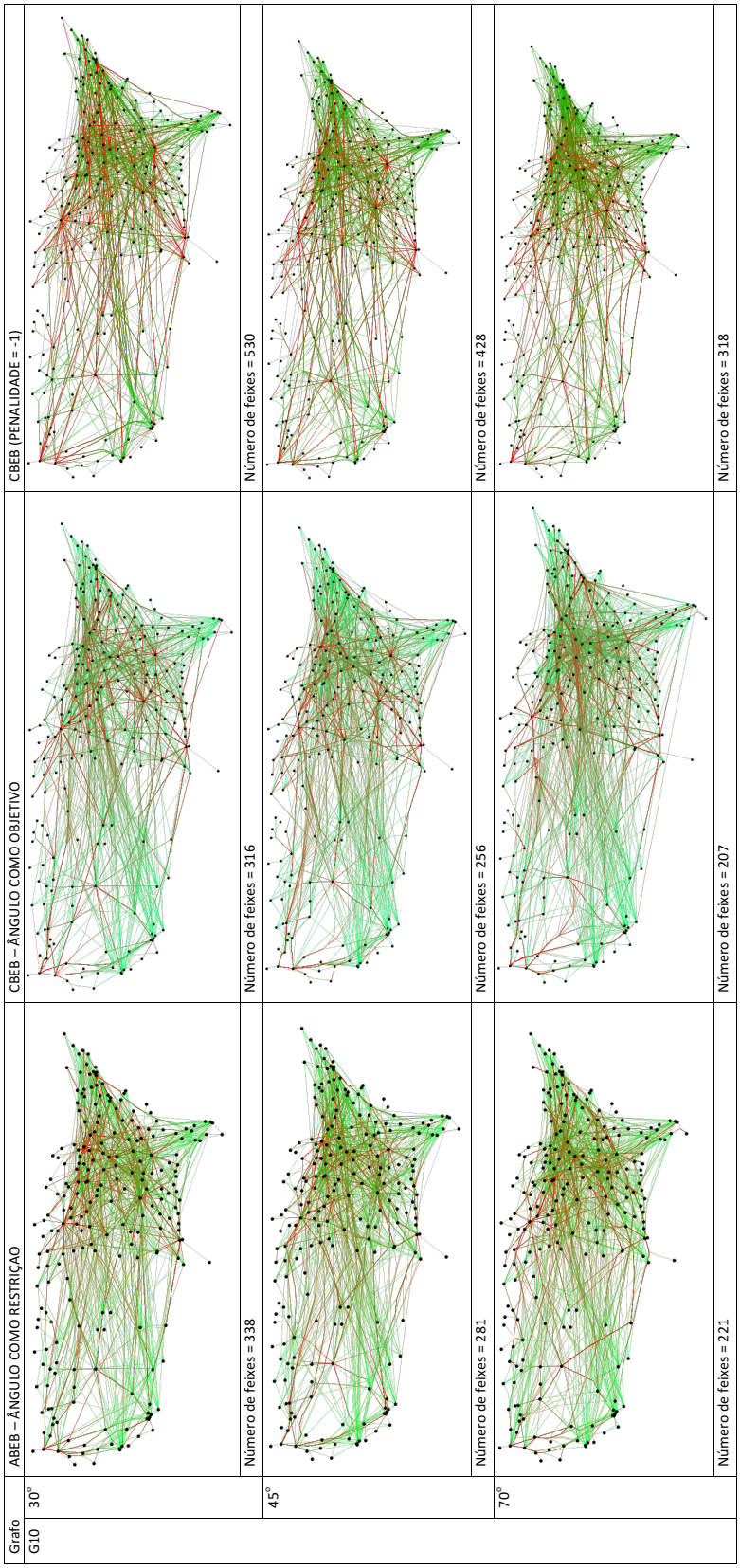


Figura B.10: *Visualizações dos resultados para o grafo USAirline (G10).*

APÊNDICE C

Gráficos de Evolução da Qualidade das Soluções

Este apêndice apresenta os gráficos de evolução da qualidade ao longo das gerações do teste que gerou a melhor solução para os problemas *ABEB* e *CBEB*.

C.1 Gráficos de Evolução da Qualidade das Soluções para o Problema ABEB

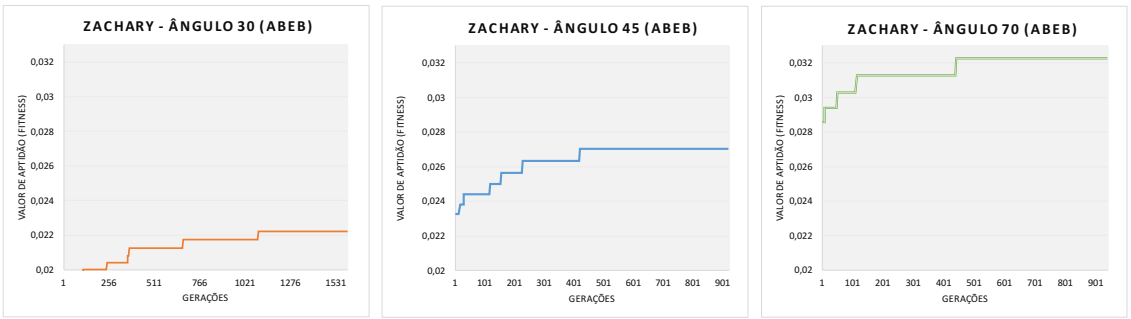


Figura C.1: *Convergência do grafo Zachary Club (G2 - 78 arestas) para o problema ABEB com EEB. Soluções com 45 ($\alpha = 30°$), 37 ($\alpha = 45°$) e 31 ($\alpha = 70°$) feixes.*

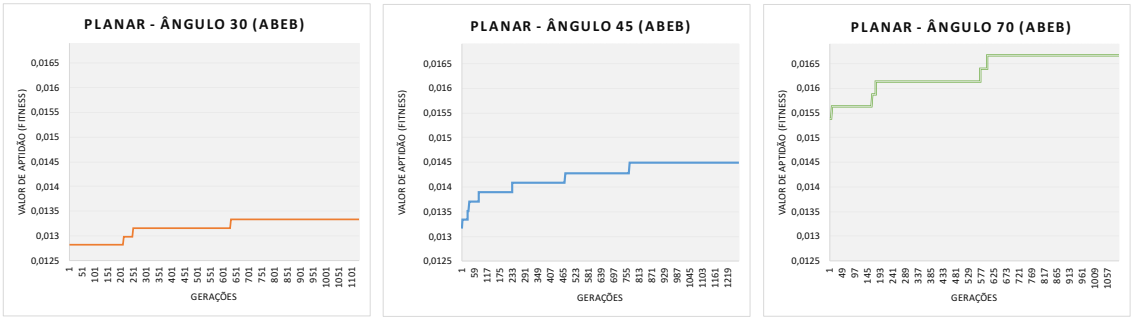


Figura C.2: *Convergência do grafo PlanarGD2015 (G3 - 101 arestas) para o problema ABEB com EEB.Soluções com 75 ($\alpha = 30°$), 69 ($\alpha = 45°$) e 60 ($\alpha = 70°$) feixes.*

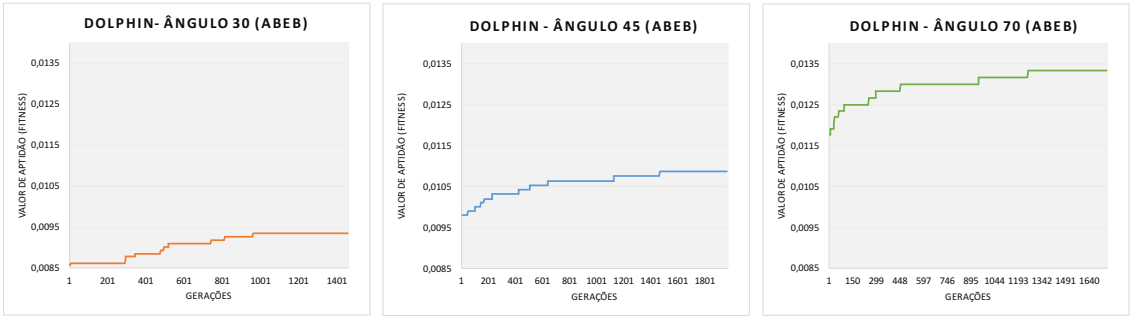


Figura C.3: *Convergência do grafo Dolphin (G4 - 160 arestas) para o problema ABEB com EEB. Soluções com 107 ($\alpha = 30°$), 92 ($\alpha = 45°$) e 75 ($\alpha = 70°$) feixes.*

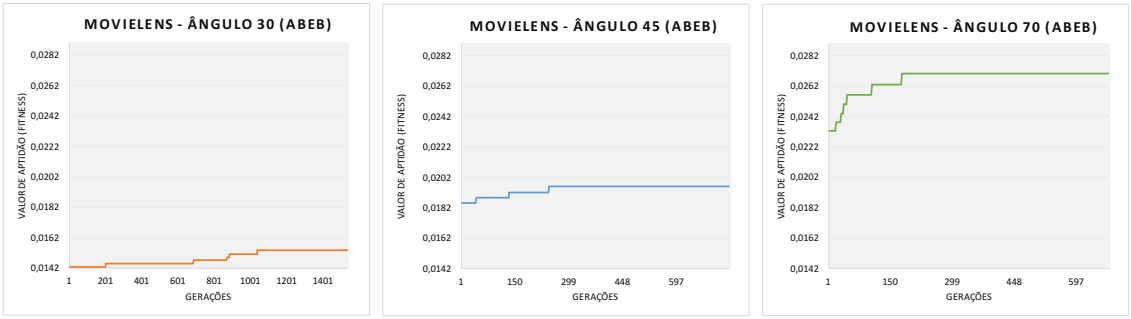


Figura C.4: *Convergência do grafo MovieLens (G5 - 161 arestas) para o problema ABEB com EEB. Soluções com 65 ($\alpha = 30°$), 51 ($\alpha = 45°$) e 37 ($\alpha = 70°$) feixes.*

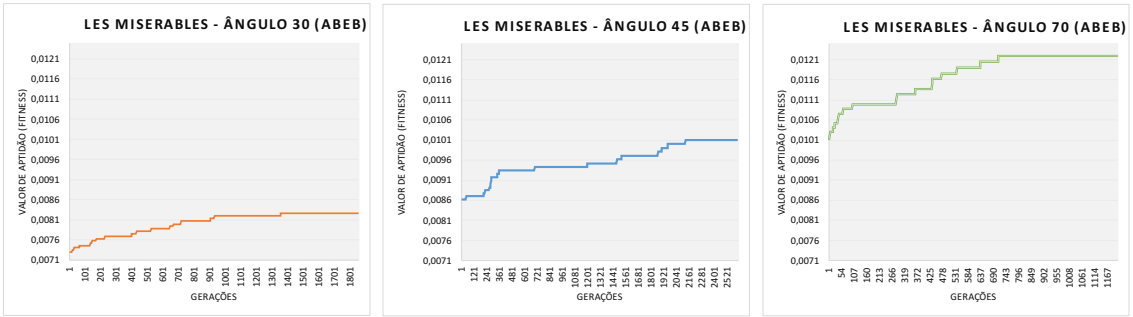


Figura C.5: *Convergência do grafo Les Miserables (G6 - 254 arestas) para o problema ABEB com EEB. Soluções com 121 ($\alpha = 30°$), 99 ($\alpha = 45°$) e 82 ($\alpha = 70°$) feixes.*

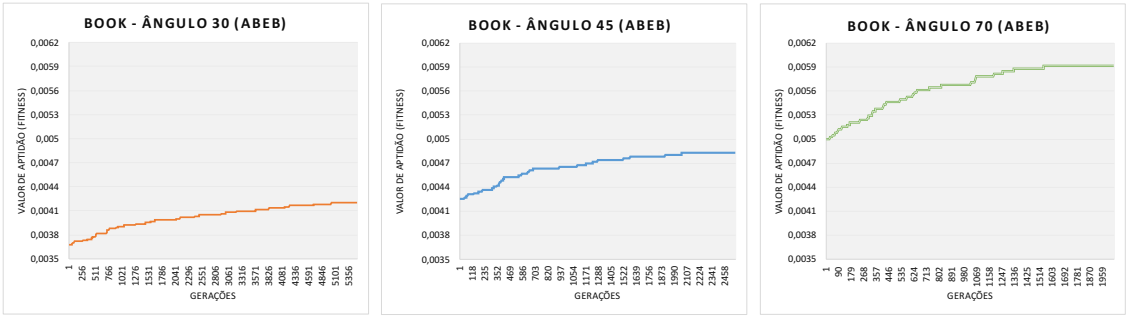


Figura C.6: *Convergência do grafo Book USPolitics (G7 - 401 arestas) para o problema ABEB com EEB. Soluções com 238 ($\alpha = 30°$), 207 ($\alpha = 45°$) e 169 ($\alpha = 70°$) feixes.*

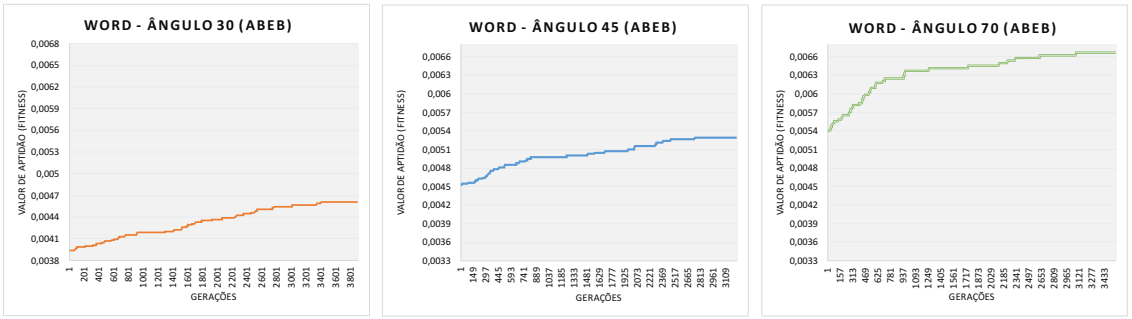


Figura C.7: *Convergência do grafo Word Adjacences (G8 - 425 arestas) para o problema ABEB com EEB. Soluções com 217 ($\alpha = 30°$), 189 ($\alpha = 45°$) e 150 ($\alpha = 70°$) feixes.*

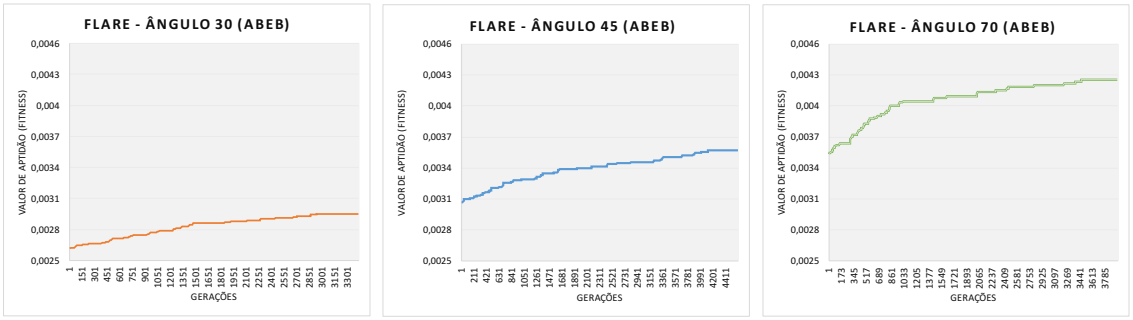


Figura C.8: *Convergência do grafo Flare (G9 - 709 arestas) para o problema ABEB com EEB. Soluções com 339 ($\alpha = 30°$), 280 ($\alpha = 45°$) e 235 ($\alpha = 70°$) feixes.*

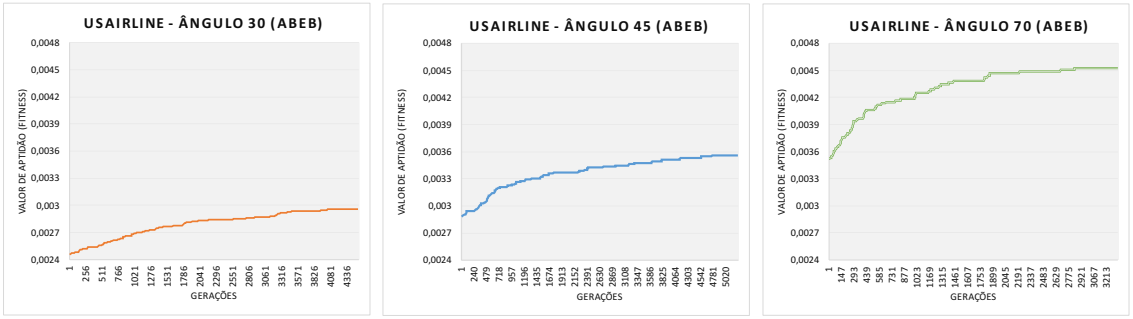


Figura C.9: *Convergência do grafo USAirline (G10 - 1297 arestas) para o problema ABEB com EEB. Soluções com 338 ($\alpha = 30°$), 281 ($\alpha = 45°$) e 221 ($\alpha = 70°$) feixes.*

C.2 Gráficos de Evolução da Qualidade das Soluções para o Problema CBEB

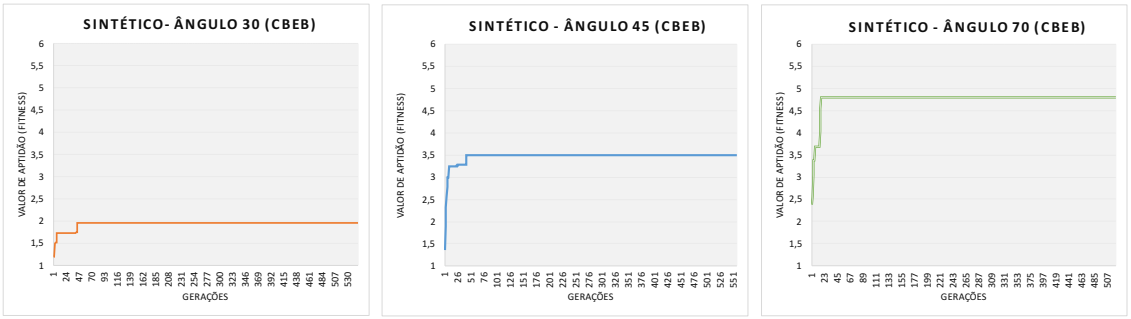


Figura C.10: *Convergência do grafo Sintético (G1 - 28 arestas) para o problema CBEB com penalidade $p_e = -1$. Soluções com 20 ($\alpha = 30°$), 16 ($\alpha = 45°$) e 16 ($\alpha = 70°$) feixes.*

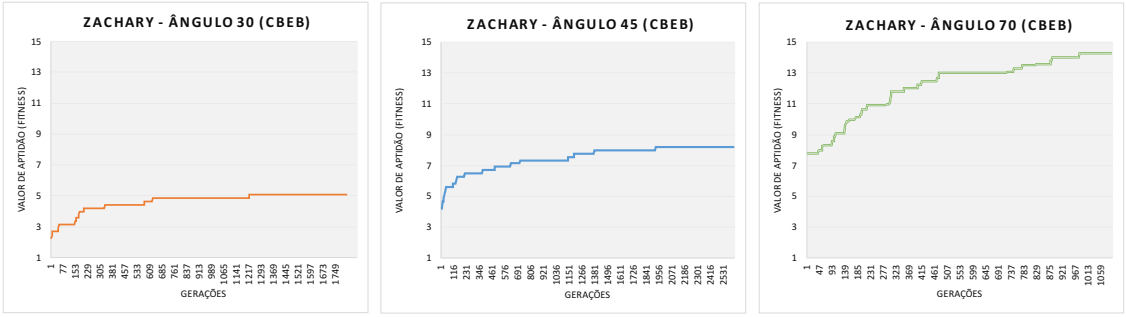


Figura C.11: *Convergência do grafo Zachcary Club (G2 - 78 arestas) para o problema CBEB com penalidade $p_e = -1$. Soluções com 56 ($\alpha = 30°$), 48 ($\alpha = 45°$) e 36 ($\alpha = 70°$) feixes.*

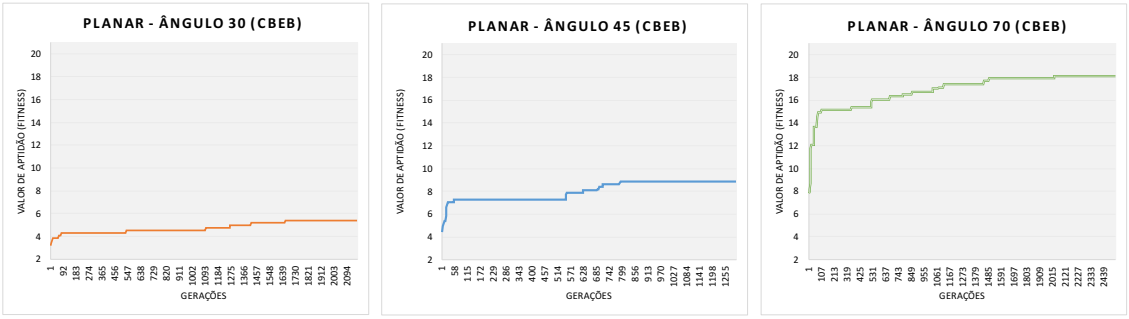


Figura C.12: *Convergência do grafo Planar GD2015 (G3 - 101 arestas) para o problema CBEB com penalidade* $p_e = -1$. *Soluções com 81* ($\alpha = 30°$), *76* ($\alpha = 45°$) *e 65* ($\alpha = 70°$) *feixes.*

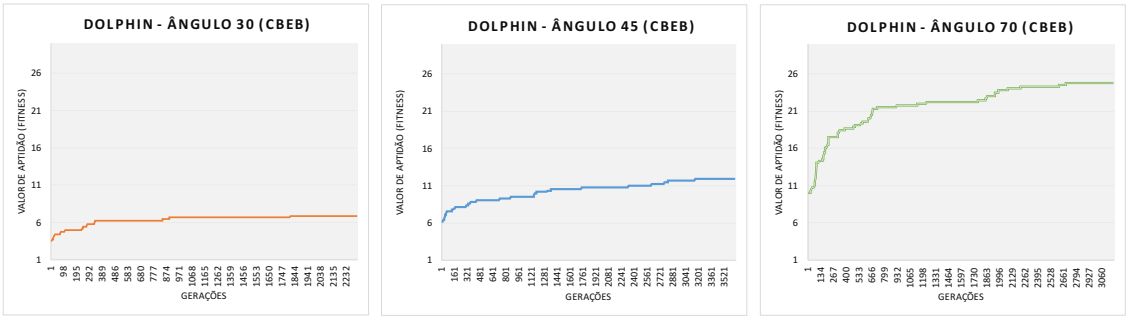


Figura C.13: *Convergência do grafo Dolphin (G4 - 160 arestas) para o problema CBEB com penalidade* $p_e = -1$. *Soluções com 129* ($\alpha = 30°$), *112* ($\alpha = 45°$) *e 89* ($\alpha = 70°$) *feixes.*

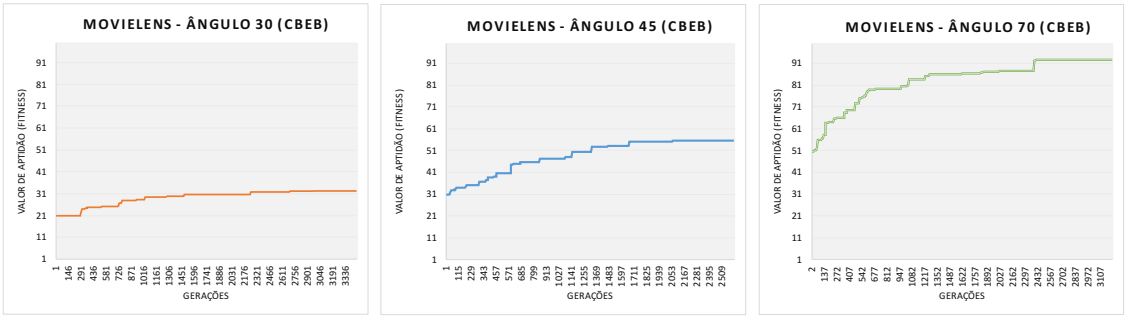


Figura C.14: *Convergência do grafo MovieLens (G5 - 161 arestas) para o problema CBEB com penalidade* $p_e = -1$. *Soluções com 76* ($\alpha = 30°$), *60* ($\alpha = 45°$) *e 50* ($\alpha = 70°$) *feixes.*

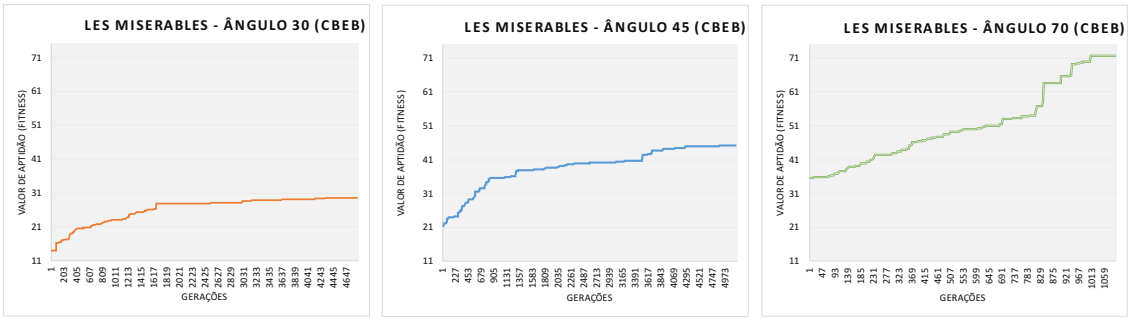


Figura C.15: *Convergência do grafo Les Miserables (G6 - 254 arestas) para o problema CBEB com penalidade* $p_e = -1$. *Soluções com 174* ($\alpha = 30°$), *137* ($\alpha = 45°$) *e 97* ($\alpha = 70°$) *feixes.*

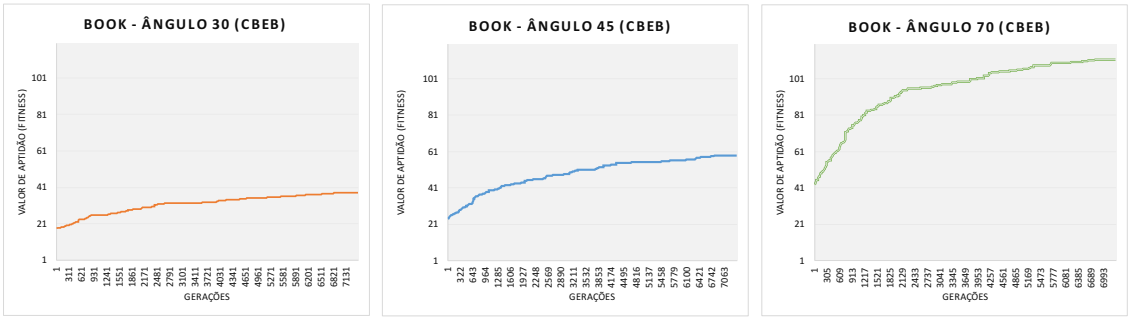


Figura C.16: *Convergência do grafo Book USPolitics (G7 - 401 arestas) para o problema CBEB com penalidade* $p_e = -1$. *Soluções com 305* ($\alpha = 30°$), *260* ($\alpha = 45°$) *e 209* ($\alpha = 70°$) *feixes.*

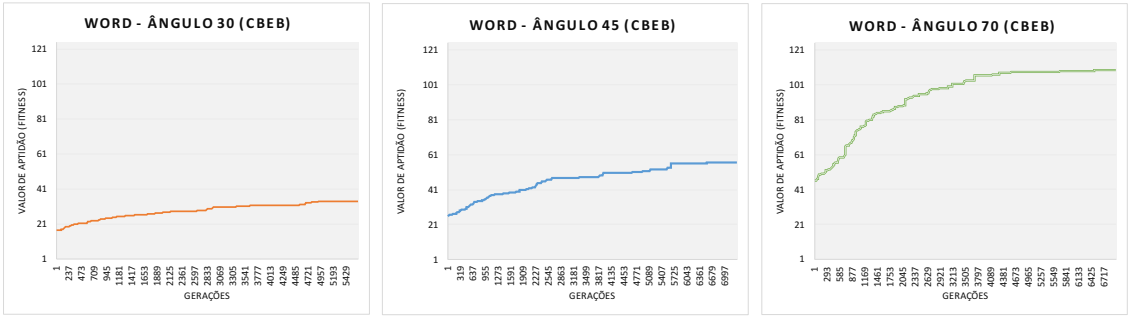


Figura C.17: *Convergência do grafo Word Adjacences (G8 - 425 arestas) para o problema CBEB com penalidade* $p_e = -1$. *Soluções com 303* ($\alpha = 30°$), *267* ($\alpha = 45°$) *e 208* ($\alpha = 70°$) *feixes.*

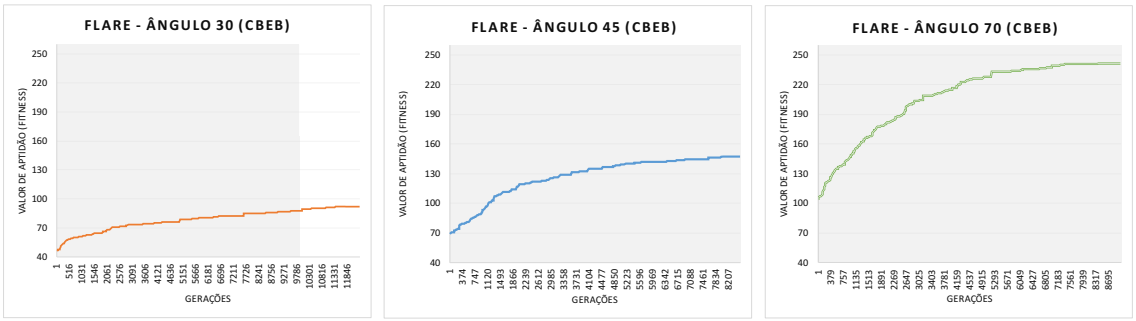


Figura C.18: *Convergência do grafo Flare (G9 - 709 arestas) para o problema CBEB com penalidade $p_e = -1$. Soluções com 452 ($\alpha = 30°$), 379 ($\alpha = 45°$) e 312 ($\alpha = 70°$) feixes.*

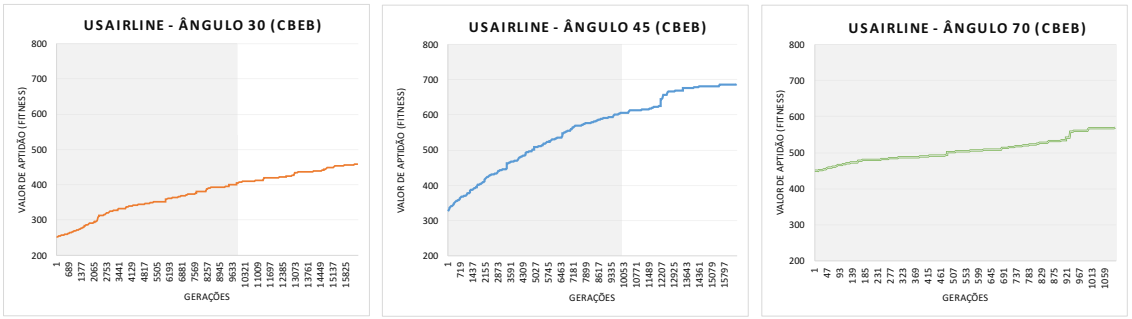


Figura C.19: *Convergência do grafo USAirline (G10 - 1297 arestas) para o problema CBEB com penalidade $p_e = -1$. Soluções com 530 ($\alpha = 30°$), 428 ($\alpha = 45°$) e 318 ($\alpha = 70°$) feixes.*

APÊNDICE D

Trabalhos Relacionados

Este apêndice apresenta uma relação de trabalhos relacionados à união explícita de arestas ou utilização de feixes centralizados, e que não foram discutidos no corpo do trabalho.

D.1 Trabalhos Relacionados à União Explícita de Arestas

Apesar do roteamento das arestas ser uma parte importante da união de aresta em feixes, o presente trabalho não tem o objetivo de tratar tal assunto mas, sim investigar problemas de criação explícita de feixes. São poucas as abordagens para união explícita de arestas. Esta seção discute alguns desses trabalhos.

O primeiro trabalho de união explícita de arestas foi o de Gansner e Koren [35]. Eles apresentaram um método para desenhos de grafos circulares cuja meta é reduzir cruzamentos e a densidade das arestas. O algoritmo é composto de três passos: posicionamento dos vértices em círculo, posicionamento externo das arestas e união das arestas em feixes.

Inicialmente, o algoritmo posiciona os vértices no círculo de forma a garantir a diminuição do tamanho das arestas e reduzir cruzamentos entre elas. Para isso, Gansner e Koren usaram um método baseado em energia que assegura a posição ótima dos vértices. Após o posicionamento dos vértices, o algoritmo roteia um conjunto de arestas na parte externa do círculo. Os objetivos são assegurar que o conjunto de arestas escolhido seja o maior possível, que elas não se cruzem na parte externa, e que quando roteadas externamente minimizem o número total de cruzamentos no grafo. O passo final é a união das arestas que restaram na parte interna do círculo, formando os feixes. É utilizado, neste passo, um algoritmo de clusterização hierárquico que define os grupos de arestas baseado na minimização da quantidade de tinta gasta para desenhá-las.

Segundo Gansner e Koren [35], a complexidade do algoritmo de união das arestas é $O(|n|^2)$, sendo n a quantidade de arestas. A Figura D.1 mostra o resultado dessa técnica aplicada a um grafo aleatório.

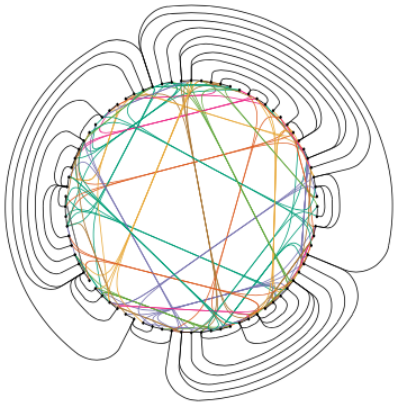


Figura D.1: *Resultado da técnica de Gansner e Koren. Adaptado de [35].*

Um outro trabalho relacionado é o de Pupyrev, Nachmanson e Kaufmann [97]. Eles desenvolveram um método que cria feixes de arestas em desenhos de grafos hierárquicos já pré definidos e apresentaram uma abordagem gulosa que efetua o roteamento das arestas entre as camadas visando a minimização de tinta.

Os vértices do grafo não são alterados ao longo do processo, mas o algoritmo usa como princípio a mudança do posicionamento e a ordem de vértices falsos[1] presentes nas arestas para permitir o seu roteamento. O procedimento de união do algoritmo consiste em gerar feixes para grupos mutuamente disjuntos de arestas, conectando vértices de camadas adjacentes quando as extremidades de origem e destino das arestas coincidirem em camadas, e a união representar a redução da tinta. Para evitar curvas excessivas, o método de Pupyrev, Nachmanson e Kaufmann restringe a união de arestas que geram um ângulo superior a $\pi/4$ no desenho com feixes. Depois que um desenho intermediário é criado, para diminuir a quantidade de cruzamentos, é utilizada uma abordagem de minimização de cruzamentos que gera um desenho final no estilo *metro-line*.

Em [97] são usados três métodos para identificar quais arestas são compatíveis para serem unidas em feixe, definidos por Pupyrev, Nachmanson e Kaufmann como: *naive lookup*, *efficient lookup* e *fast restricted lookup*. O primeiro é baseado na metodologia de força bruta, onde todos os pares de arestas são enumerados e analisados. O método tem complexidade de pior caso igual a $O(m^3t)$, sendo t o tempo necessário para calcular o ganho de tinta quando unindo dois feixes e m o tamanho do grupo de arestas analisado. O segundo método, com complexidade $O(m^2)$, evita o cálculo do ganho de tinta de pares muito distantes usando o conceito de compatibilidade. Dois feixes são ditos compatíveis para união se não existem outros feixes entre eles. O terceiro método evita enumerar todos os pares de arestas. A compatibilidade é restrita àquelas que possuem uma extremidade

[1] Em desenhos de grafo hierárquicos, durante o processo de organização dos vértices, se uma aresta cruza um ou mais níveis, são posicionados vértices falson em cada ponto de interseção com os níveis [6].

em comum. A complexidade é $O(tmlogm)$.

A Figura D.2 mostra o resultado do método aplicado ao grafo referente ao modelo do diagrama de estado do software *Notepad.exe*.

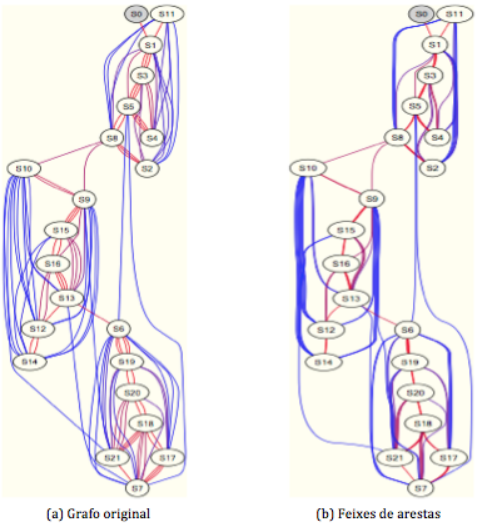


Figura D.2: *Resultado da técnica união de arestas de Pupyrev, Nachmanson e Kaufmann. Adaptado de [97].*

Em 2010, Telea e Ersoy [112] apresentaram um trabalho que usa técnicas de processamento de imagens na união de arestas. Eles não criaram propriamente uma abordagem para gerar os feixes de arestas, mas sim uma forma de destacar os feixes criados por um outro algoritmo. Foi implementada uma representação visual para desenhos de grafos com feixes de arestas que permite sobressair os feixes, e dessa forma reconhecer a individualidade de cada um. O método foi denominado por Telea e Ersoy como *image-based edge bundling* (IBEB).

O algoritmo de Telea e Ersoy recebe como entrada um desenho com os feixes de arestas já preestabelecidos. Utilizando uma técnica de clusterização hierárquica, as arestas similares são agrupadas do ponto de vista de posição no desenho (forma, posição e topologia) e, opcionalmente, densidade espacial e atributos de dados. Na sequência, os clusteres são renderizados usando técnicas de computação gráfica. Para destacar a densidade, o tipo e a similaridade das arestas, Ersoy e Telea usaram artifícios visuais como luminância, saturação, tonalidade e sombreamento. A visualização criada evidencia os clusteres de arestas dando aos feixes uma aparência orgânica. Há que se observar que a clusterização é usada nesse método apenas para montar as formas visuais e executar o sombreamento das arestas e, dessa forma, possibilitar destaque à partição das arestas já existentes. A Figura D.3 apresenta o resultado final dessa aplicação da abordagem ao grafo *USMigration*.

Em 2011, Ersoy e outros [25] construíram uma abordagem que gera feixes a partir da clusterização de arestas, do cálculo do mapa de distância e do eixo medial 2D

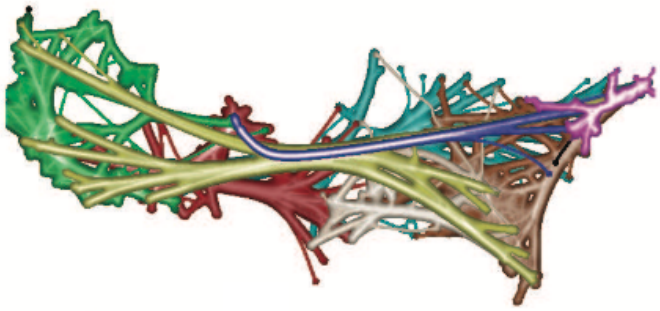


Figura D.3: *Resultado da técnica IBEB. Adaptado de [112].*

(esqueleto). A abordagem foi chamada de *skeleton-based edge bundling* (SBEB) e foi implementada em CUDA. Na abordagem, primeiramente, os feixes são definidos a partir do agrupamento de arestas com forte similaridade geométrica. Também podem ser usados atributos de dados como parâmetro de compatibilidade. Os clusteres de arestas são criados usado o algoritmo de clusterização descrito em [112]. É definida uma forma (do inglês *shape*) que circunda o desenho das arestas de cada cluster e, o seu eixo medial (esqueleto) é calculado. O esqueleto é o conjunto de todos os pontos no espaço que são equidistantes no mínimo a dois pontos das bordas da superfície. As arestas são discretizadas em pontos, sendo estes pontos movimentados em direção ao esqueleto da sua respectiva forma.

Esse processo é iterativo. A clusterização é repetida em cada ciclo de montagem de feixes porque, na primeira clusterização, é usado um *threshold* de similaridade muito alto com o objetivo de gerar feixes longos e finos. Nas iterações seguintes o valor do *threshold* é reduzido evitando que se produza clusteres cada vez maiores. Seguindo esta ideia, clusteres menores se anexam a clusteres maiores.

Segundo Ersoy e outros [25], a complexidade do método SBEB é $O(|C|)$ onde C é o tamanho médio do cluster. A Figura D.4 mostra o resultado final da aplicação da abordagem SBEB no grafo *USMigration*.

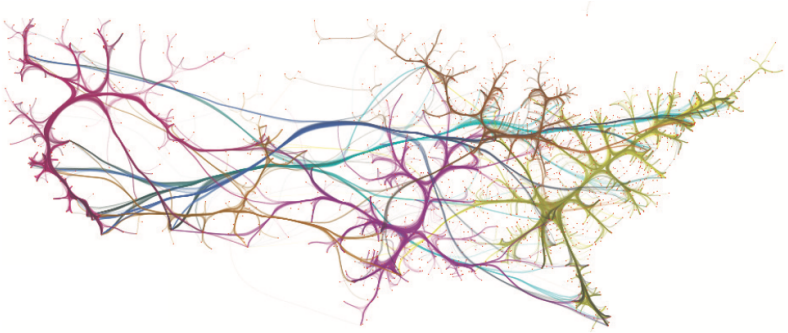


Figura D.4: *Resultado da técnica SBEB. Adaptado de [28].*

Também em 2011, Gansner e outros [34] apresentaram uma técnica para união de arestas de grafos não direcionados gerais baseada em vizinhança que possui como objetivo a minimização da quantidade de tinta. A técnica foi inspirada na abordagem aglomerativa

multinível, a qual é uma técnica recursiva que divide o problema em problemas menores que servem como entrada para os demais níveis. O método foi denominado como *multilevel agglomerative edge bundling* (MINGLE).

No método MINGLE, uma aresta é tratada como um ponto em um espaço de 4-dimensões. Uma *kd-tree* (com dimensão quatro) decompõe o espaço $4D$ para ajudar a encontrar os k vizinhos mais próximos de cada aresta. Usando a estrutura *kd-tree* como entrada, é construído então um grafo de proximidade de arestas que modela as suas similaridades. Esse grafo serve para identificar arestas vizinhas e, desta forma, permitir ao algoritmo descobrir a informação se a união de uma aresta com seus vizinhos minimiza ou não a quantidade de tinta. Duas arestas estão próximas se a distância Euclidiana no espaço $4D$ é pequena. Esse algoritmo recursivo permite unir arestas que inicialmente poderiam não ser consideradas similares o suficiente para serem unidas. O algoritmo termina quando nenhuma tinta é mais reduzida.

Gansner e outros [34] afirmam que, como o grafo de proximidade está limitado à análise dos k vizinhos mais próximos, e também não são utiizadas as medidas de compatibilidade de Holten e Wijk [43] envolvendo ângulo, escala, posição e visibilidade, o MINGLE não garante unir todas as arestas compatíveis.

O algoritmo que implementa o MINGLE executa em $O(n^2)$, com $n = |E|$ sendo a quantidade de arestas. Na Figura D.5 pode ser visto o resultado quando aplicado o método ao grafo *USAirline*.

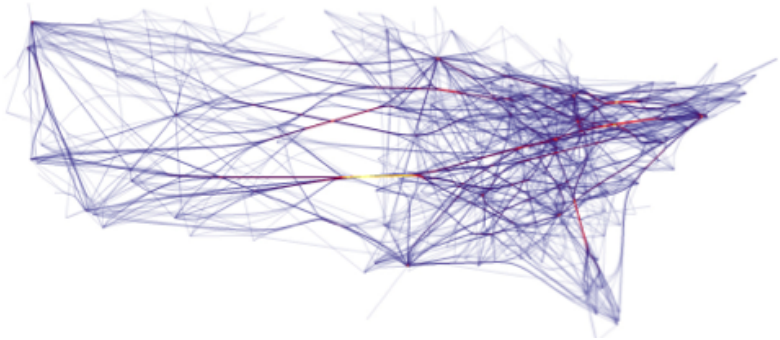


Figura D.5: *Resultado da técnica MINGLE. Adaptado de [34]*

Bouts e Speckmann [14] apresentaram um algoritmo para união de arestas baseado nos princípios de clusterização e roteamento. Esses dois passos são bem definidos e independentes dentro da abordagem. Para clusterização das arestas, foi usada a técnica *well-separated pair decomposition* (*WSPD*), as medida de compatibilidade de Holten e Wijk [43], além de um parâmetro que permite ao usuário controlar o nível de clusterização desejada. Para evitar que as arestas sejam desenhadas próximas de vértices e evitar obstáculos, é usado um algoritmo de roteamento baseado em uma abordagem gulosa da esparcificação de um grafo de visibilidade. Os clusteres formados são disjuntos, sendo que o algoritmo de roteamento toma o cuidado de evitar que arestas de grupos diferentes

sigam uma rota comum. Para minimizar ambiguidade [2], as arestas são roteadas no estilo *metro-line*.

Na Figura D.6, pode ser visto o resultado quando aplicado o método ao grafo TLR4 com 124 arestas.

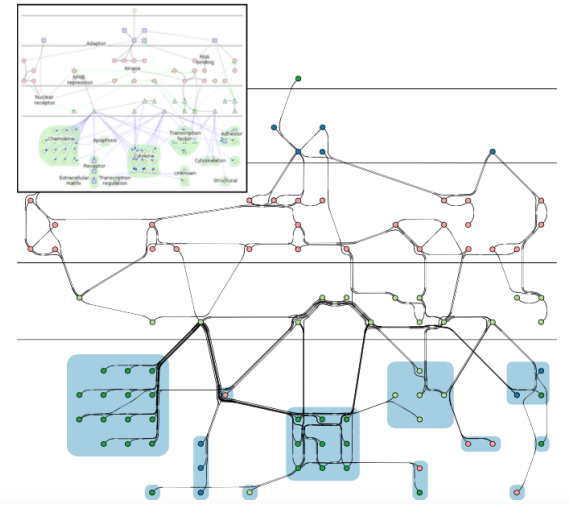


Figura D.6: *Resultado da técnica de Bouts e Speckmann. Adaptado de [14].*

Zielasko e outros [129] combinam a clusterização de arestas com o algoritmo de força para montagem dos feixes de arestas em um método denominado *3D force-directed edge bundling* (3D FDEB). Primeiramente o algoritmo agrupa as arestas com alta compatibilidade em clusteres que são subgrafos não necessariamente conectados. As medidas de compatibilidade usadas são tamanho, gradiente e posição. Elas são inspiradas nas medidas originais de Holten e Wijk [43].

É calculada então a força em cada cluster. Nesta nova etapa a compatibilidade entre as arestas é recalculada usando agora as funções para escala, angulo e posição definidas em [43]. Ao invés de usar um número de iterações fixas o método de Zielasko e outros itera até a força total não diminuir mais. A força foi modificada de um padrão linear para quadrática o que gera resultados mais estáveis.

O algoritmo proposto foi paralelizado para GeForce GTX480 e usando OpenGL para renderização. Na Figura D.7 pode ser visto o resultado quando aplicado o método ao grafo que representa a simulação NEST do cérebro de um macaco.

Recentemente, Yamashita e Saga [120] apresentaram um algoritmo que usa a metáfora de detecção de comunidades usando gafo linha[3] para formação dos clusteres

[2] A ambiguidade acontece quando as arestas são sobrepostas, de tal forma que o rastreamento delas entre os seus vértices extremos é prejudicado.

[3] Em um grafo linha $L(G)$ de um grafo G cada aresta de G é representado como um vértice de $L(G)$, quando duas arestas e_i e e_j são adjacentes em G, os vértices representantes de e_i e e_j também são adjacentes em $L(G)$ [40], citado por [120].

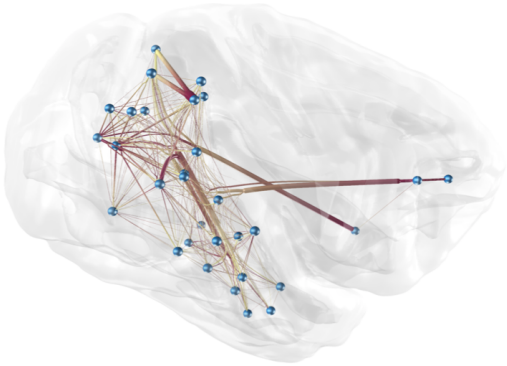


Figura D.7: *Resultado da técnica 3D FDEB. Adaptado de [129].*

de arestas. As comunidades definidas no grafo linha são na verdade grupos de arestas no grafo original. Segundo Yamashita e Saga [120], usar o grafo linha ao invés do grafo original para gerar clusteres permite que as arestas sejam unidas topologicamente com base em seus relacionamentos, e não mais geograficamente.

Na detecção de comunidades Yamashita e Saga usaram um método de clusterização baseado em modularidade [84]. Depois que os clusteres são montados, as arestas são unidas usando uma versão do algoritmo FDEB [43], só que neste caso uma nova medida de compatibilidade é utilizada a compatibilidade de cluster (C_c).

$$C_c(P,Q) = \begin{cases} N & \text{se c(P) = c(Q)} \\ N^{-1} & \text{caso contrário} \end{cases}$$

sendo $c(P)$ e $c(Q)$ é o cluster ao qual as arestas P e Q pertencem, e N é o número de clusteres.

Esta nova medida assegura que as arestas que pertencem a um mesmo cluster sejam desenhadas bem próximas, e as que se encontram em grupos diferentes sejam repelidas ou desenhadas menos próximas.

Em resumo, analisando todos esses trabalhos para união explícita de arestas verifica-se que as abordagens clássicas utilizaram estratégias diversas para definir os feixes (clusteres) como: clusterização hierárquica, enumeração, relação de vizinhança, modularidade, etc. No geral, eles implementaram a medida geométrica de compatibilidade, com alguns casos definindo outros parâmetros como a topologia dos dados. Como objetivos recorrentes estão a minimização de tinta e cruzamento de arestas. Algumas abordagens impuseram, por vezes, restrições às arestas como união de arestas somente em feixes disjuntos e limite angular.

No entanto, novamente, essas observações sobre as características dos problemas tratados são implícitas, sendo preocupações principais dessas técnicas, muitas vezes, a estratégia de roteamento dos feixes que assegure a redução da complexidade do desenho e a velocidade e escalabilidade do método. A garantia de que a clusterização e/ou

roteamento definem o melhor grupo de arestas com base nas restrições estabelecidas e de que o desenho resultante mantém a fidelidade dos dados, nem sempre, foi o foco desses trabalhos.

D.2 Trabalhos Relacionados ao Uso de Feixes Centralizados

Na seção anterior discutiu-se os trabalhos relacionados à união explícita de arestas. Esta seção destaca outras técnicas, também relacionadas ao tema desta tese, que são as abordagens que formam feixes somente com arestas adjacentes. Este tópico é importante para este trabalho, pois alguns dos problemas propostos pertencem a essa categoria.

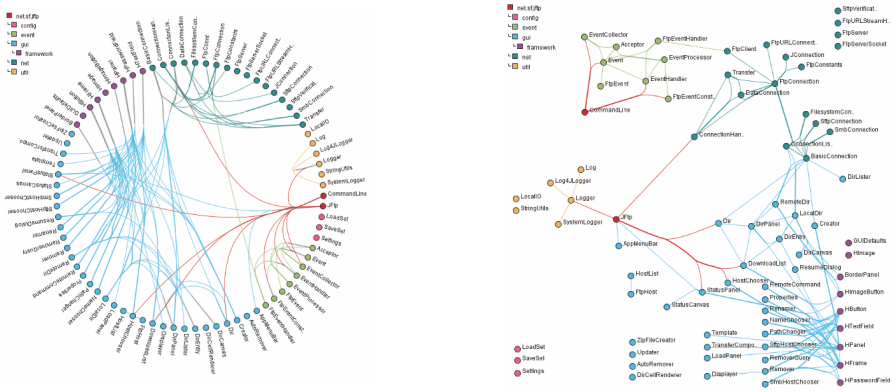
Beck e outros [8] propuseram a abordagem para união de arestas adjacentes chamada *traceability-based edge bundling* (TBEB) que preserva a rastreabilidade de arestas. Essa abordagem usa a hierarquia dos vértices implícita ou pré computada para definir as arestas que serão unidas em feixe. Foram implementados dois *layouts*: um radial e outro arbitrário.

A abordagem radial trabalha com a hierarquia global do grafo. Primeiramente os vértices são posicionados em formato circular. Para cada vértice e cada grupo de vértices adjacentes a este é computado um feixe distinto, sendo que a direção dos feixes é calculada com base no baricentro dos vértices de destino. Esse processo é aplicado tanto para arestas de saída quanto para as de entrada. Dessa forma cada vértice tem duas árvores de feixes. Esta característica permite que uma aresta pertença a mais de um feixe.

A outra abordagem gera um *layout* geral, que foi identificado como arbitrário. Neste modelo, o posicionamento geográfico dos vértices já está definido, e o algoritmo utiliza a hierarquia local para computar os feixes. Inicialmente para cada vértice, todas as suas arestas adjacentes compõem um feixe único. Uma restrição angular máxima entre dois vértices de destino é usada no refinamento dos feixes iniciais. Sempre que a restrição angular é ultrapassada, o feixe é subdividido em subfeixes, este processo e recursivo.

A Figura D.8 mostra desenhos de grafos em *layout* radial e arbitrário criados usando a abordagem TBEB.

Čermák, Dokulil e Katreniaková [17] propuseram quatro algoritmos para a união de arestas em feixes centralizados, que previnem que as arestas ou vértices sejam sobrepostos, e usam um algoritmo de roteamento de arestas para adicionar curvas a elas. A escolha das arestas que devem ser parte de um mesmo feixe segue duas estratégias. A primeira divide as arestas em grupos, considerando um ângulo crítico, e a segunda



(a) (b)

Figura D.8: *Resultado da técnica TBEB: (a) layout radial; (b) layout arbitrário. Adaptado de [8].*

abordagem usa um algoritmo de clusterização de vértices que escolhe as arestas que serão unidas, considerando a adjacência de grupos de vértices.

A técnica de Čermák, Dokulil e Katreniaková executa a montagem de feixes através de passos sucessivos, começando com o vértice de maior grau. A cada feixe montado as arestas e vértices envolvidos são excluídos do próximo passo. Para determinar quais arestas são unidas em feixes a partir de um vértice escolhido foram desenvolvidos quatro algoritmos. O primeiro algoritmo, identificado como *fixed-angle*, divide a área ao redor do vértice em regiões que determinam quais arestas serão unidade em feixe. O segundo método chamado *variable-angle*, tem como base o primeiro, mas utiliza um valor de ângulo crítico para determinar quais arestas de uma região serão agrupadas. O terceiro método também é uma variação do primeiro, chamado de *first-bend* e, usa como critério para unir as arestas de uma região apenas aquelas cuja primeira curva esta na mesma posição. E por fim, o último algoritmo que é baseado na clusterização dos vértices de destino como elemento de divisão dos feixes, identificado como *clustering*.

Nessa abordagem os vértices possuem posições fixas, são usadas linhas retas ao invés de curvas, e uma aresta nunca aparece como componente de mais de um feixe. A Figura D.9 mostra o resultado da técnica de Čermák, Dokulil e Katreniaková usando o algoritmo de clusterização.

Luo e outros [70] propuseram um método chamado *ambiguity-free edge-bundling* (AFEB) que une arestas que são adjacentes e também geometricamente próximas, aplicando a estrutura *quadtree* para escolher as arestas de cada feixe. O diferencial dessa técnica é a forma como ela agrupa as arestas e o ajuste automático da curva para garantir que não existirá ambiguidade no desenho gerado.

O algoritmo começa tendo como entrada um grafo com os vértices já posicionados no plano. É feito então o particionamento do espaço usando a estrutura de dados

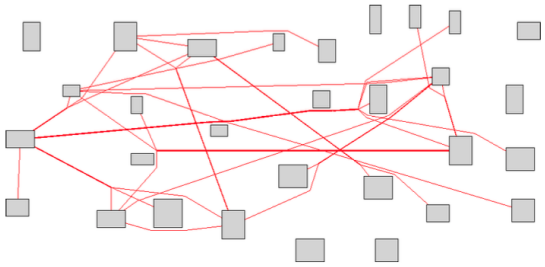


Figura D.9: *Resultado da técnica de Čermák, Dokulil e Katreniaková. Adaptado de [17].*

quadtree, de forma que cada célula tenha unicamente um vértice, gerando células ocupadas e vazias do ponto de vista dos vértices. Essas células possuem um tamanho máximo definido e calculado em função do espaço e da quantidade de vértices. O passo seguinte é a detecção de quais dessas células estão ocupadas por arestas, se uma aresta passa apenas por células vazias a probabilidade de ambiguidade é baixa. Esta informação é usada no passo seguinte responsável pela eliminação da ambiguidade de arestas. Se for verificado que uma ou mais arestas passam por células ocupadas, pontos de controle são calculados para cada aresta por onde estas serão roteadas. Esse processo e guiado por critérios estéticos como: redução de cruzamentos, redução da *curvas* (*bends*) das arestas e manutenção da continuidade da curvatura da aresta. O passo seguinte é a criação do feixe propriamente dito através da união das arestas que passam pela mesma célula vazia e possuem um vértice em comum. A Figura D.10 mostra um grafo com feixes e sua *quadtree* gerada.

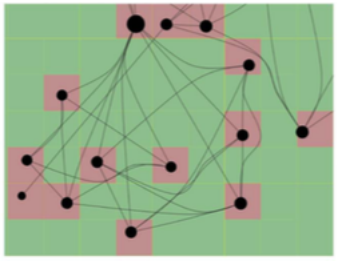


Figura D.10: *Resultado da técnica AFEB. Adaptado de [70].*

Peng e outros [92] definiram um método que clusteriza e "amarra" (do inglês *knot*) arestas. Eles notaram que a união de arestas em feixes é semelhante a dar um nó nas arestas, e que as técnicas tradicionais costumam unir as arestas pelo centro. A abordagem proposta por eles move o "nó" para uma posição mais próxima do vértice comum entre as arestas do feixe, baseado na sua orientação, isto é, o "nó" é posicionado próximo ao vértice central e não no centro. Eles chamaram esta abordagem de *node-based bundling* (NBB) e criaram um algoritmo que foi identificado como SideKnot.

Primeiramente as arestas adjacentes que respeitam um determinado ângulo limite para cada vértice são agrupadas. O processo começa escolhendo duas arestas que possuam

um ângulo mínimo para formar o primeiro feixe. Novas arestas vizinhas são adicionadas a este feixe caso o ângulo entre as arestas não exceda um valor predefinido e o ângulo total do feixe também não exceda um limite. Uma vez definidos o agrupamento das arestas a próxima etapa calcula a direção e os pontos de controle da curva *spline* de cada aresta. A direção de cada feixe é calculada com base na direção média das arestas que o compõe. Peng e outros afirmam que podem ser usadas a média ou mediana do ângulo entre as arestas para traçar uma linha de base. Pontos de controle são então traçados nesta linha base para cada aresta do feixe, estes pontos são proporcionais ao tamanho da aresta. Esta abordagem permite que uma aresta esteja em mais de um feixe. O último passo é a renderização onde a aplicação de transparência busca enfatizar as pontas das arestas, próximo aos vértices, mantendo o seu interior opaco, isso é importante para destacar os padrões de relação nos vértices.

Segundo Peng e outros [92], o algoritmo executa em $O(|E|)\log(|E|)$ sendo $|E|$ a quantidade de arestas. A Figura D.11 mostra o resultado da aplicação do SideKnot ao grafo *USAirline*.

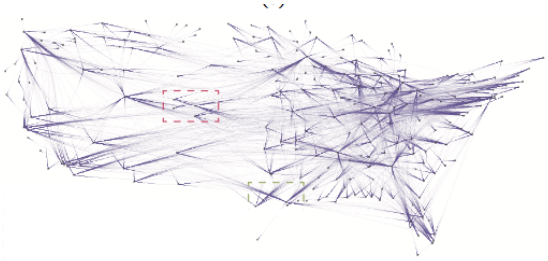


Figura D.11: *Resultado da técnica NBB. Adaptado de [92].*

Nocaj e Brandes [89] propuseram um método chamado *stub bundling* que é usado para visualizar redes geométricas não ambíguas baseadas em arestas adjacentes. Eles assumem que unir arestas em seus vértices extremos assegura que elas não se desviam de uma linha conectando esses vértices, induzindo assim uma direção geral de destino. Nocaj e Brandes refinaram a abordagem de Peng e outros [92] e usaram o ângulo entre arestas consecutivas como o critério de escolha daquelas que seriam unidas.

Para assegurar que as arestas de um feixe mantenham a direção geral da sua tangente, são usados ângulos entre arestas como critério para determinar a partição das arestas do grafo em grupos de feixes. Considerando ângulos α, γ, foram estabelecidos os seguintes critérios para determinar uma (α,γ)-união: o ângulo entre duas arestas em um feixe é no máximo α, e o ângulo entre duas arestas consecutivas em um feixe é no máximo γ. Observa-se que a maioria das técnicas para união de arestas em feixes centralizados utilizou o ângulo como elemento limitador de quais arestas seriam agrupadas.

O procedimento é muito simples em sua essência primeiro encontra-se partições de arestas ao redor de vértices comuns representando os feixes, quebrando o feixe em

todas as ocorrências de ângulo máximo γ. Para os casos onde os ângulos máximos são iguais quebrar em feixes menores. Esses passos são repetidos iterativamente até alcançar um (α,γ)-união. Após esse processo os pontos de controle para guiar o roteamento das arestas de cada feixe são calculados e as arestas roteadas.

Segundo Nocaj e Brandes [89], o algoritmo executa no pior caso em $O(|E|\log\Delta)$ onde $|E|$ é a quantidade de aresta e Δ é o ângulo máximo de um vértice. A Figura D.12 mostra o resultado da abordagem *stub bundling* aplicada ao grafo *USAirline*.

Figura D.12: *Resultado da técnica de união de arestas denominada stub bundling. Adaptada de [89].*