



FELIPE DELFINI CAETANO FIDALGO

DIVIDINDO E CONQUISTANDO COM SIMETRIAS EM
GEOMETRIA DE DISTÂNCIAS

CAMPINAS
2015



UNIVERSIDADE ESTADUAL DE CAMPINAS

Instituto de Matemática, Estatística
e Computação Científica

FELIPE DELFINI CAETANO FIDALGO

DIVIDINDO E CONQUISTANDO COM SIMETRIAS EM
GEOMETRIA DE DISTÂNCIAS

Tese apresentada ao Instituto de Matemática, Estatística e Computação Científica da Universidade Estadual de Campinas como parte dos requisitos exigidos para a obtenção do título de Doutor em matemática aplicada.

Orientador: Carlile Campos Lavor

ESTE EXEMPLAR CORRESPONDE À VERSÃO FINAL DA
TESE DEFENDIDA PELO ALUNO FELIPE DELFINI CAETANO FIDALGO, E ORIENTADA PELO PROF. DR. CARLILE CAMPOS LAVOR.

Assinatura do Orientador

A handwritten signature in blue ink, appearing to read "Carlile Lavor", is placed over a horizontal line.

CAMPINAS
2015

Ficha catalográfica
Universidade Estadual de Campinas
Biblioteca do Instituto de Matemática, Estatística e Computação Científica
Ana Regina Machado - CRB 8/5467

Fidalgo, Felipe Delfini Caetano, 1987-
F448d Dividindo e conquistando com simetrias em geometria de distâncias / Felipe Delfini Caetano Fidalgo. – Campinas, SP : [s.n.], 2015.

Orientador: Carlile Campos Lavor.
Tese (doutorado) – Universidade Estadual de Campinas, Instituto de Matemática, Estatística e Computação Científica.

1. Geometria de distâncias. 2. Simetria (Matemática). 3. Estrutura molecular. 4. Algoritmos branch-and-prune. I. Lavor, Carlile Campos, 1968-. II. Universidade Estadual de Campinas. Instituto de Matemática, Estatística e Computação Científica. III. Título.

Informações para Biblioteca Digital

Título em outro idioma: Divinding and conquering with symmetries in distance geometry

Palavras-chave em inglês:

Distance geometry

Symmetry (Mathematics)

Molecular structure

Branch-and-prune algorithms

Área de concentração: Matemática Aplicada

Titulação: Doutor em Matemática Aplicada

Banca examinadora:

Carlile Campos Lavor [Orientador]

Sueli Irene Rodrigues Costa

Eduardo Xavier Silva Miqueles

Giuliano Gadioli La Guardia

Tiberius de Oliveira e Bonates

Data de defesa: 27-02-2015

Programa de Pós-Graduação: Matemática Aplicada

Tese de Doutorado defendida em 27 de fevereiro de 2015 e aprovada

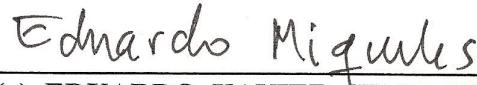
Pela Banca Examinadora composta pelos Profs. Drs.



Prof(a). Dr(a). CARLILE CAMPOS LAVOR



Prof(a). Dr(a). SUEL IRENE RODRIGUES COSTA



Prof(a). Dr(a). EDUARDO XAVIER SILVA MIQUELES



Prof(a). Dr(a). GIULIANO GADIOLI LA GUARDIA



Prof(a). Dr(a). TIBERIUS DE OLIVEIRA E BONATES

Abstract

Motived by studies in 3D structures of proteins, essential biomolecules for Life, arised a problem called Discretizable Molecular Distance Geometry Problem (DMDGP) which proved to be NP-Hard. Aiming to solve it, there is an algorithm in the literature, Branch & Prune (BP), which uses a combinatorial strategy of exploring a binary tree of solutions that is associated to the problem. Moreover, some symmetry relations have been discovered which allows the obtainance of one solution from the other one by means of reflections in the so-called symmetry vertices.

Some researchers started to do this work using parallel computing (ParallelBP), dividing one instance into sub-instances, solving the problem locally with the BP (what can be done in two directions) and joining the sub-solutions with rigid movements, with the objective of determining the solutions in a smaller time.

Our purpose, thus, is to provide a Divide-and-Conquer strategy to solve the DMDGP in order to improve the parallel version. It has three stages. Initially, the instance is divided into sub-instances two-by-two overlapping by means of the symmetry vertices. After, the so-called gaps are used to decide the direction that the BP ought to provide the local solution. Finally, we propose to use Quaternion Rotations to combine sub-solutions into feasible solutions.

Keywords: Distance geometry, Symmetry (Mathematics), Molecular structure, Branch-and-prune algorithms

Resumo

Motivado por estudos em estruturas 3D de proteínas, biomoléculas imprescindíveis no estudo da vida, surgiu um problema chamado Discretizable Molecular Distance Geometry Problem (DMDGP) que provou ser NP-Difícil. Para resolvê-lo, existe um algoritmo da literatura, Branch & Prune (BP), que utiliza uma estratégia combinatória de exploração de uma árvore binária de soluções associada ao problema. Além disso, foram descobertas relações de simetria que permitem obter uma solução, a partir de outra, através de reflexões nos chamados vértices de simetria.

Alguns pesquisadores passaram a realizar este trabalho em paralelo (ParallelBP), dividindo uma instância em sub-instâncias, resolvendo localmente com o BP (o que pode ser feito em duas direções) e unindo as sub-soluções com movimentos rígidos, com o intuito de determinar as soluções em menor tempo.

Nossa proposta é fornecer uma estratégia Dividir-e-Conquistar para resolver o DMDGP, de modo a melhorar a abordagem em paralelo. Ela possui três estágios. Inicialmente, dividimos uma instância em sub-instâncias duas-a-duas sobrepostas através dos vértices de simetria. Depois, utiliza-se os chamados gaps para decidir a direção em que o BP deve fornecer a solução local. Por fim, utilizamos rotações baseadas na Álgebra de Quaternios para combinar as sub-soluções em soluções factíveis.

Palavras-chave: Geometria de distâncias, Simetria (Matemática), Estrutura molecular, Algoritmos branch-and-prune

Sumário

Dedicatória	xiii
Agradecimentos	xv
1 Introdução	1
1.1 Formação das Proteínas e sua estrutura química	2
1.2 Estrutura Tridimensional das Proteínas	4
1.2.1 Estrutura Primária	5
1.2.2 Estrutura Secundária	6
1.2.3 Estrutura Terciária	7
1.2.4 Estrutura Quaternária	8
1.2.5 Comparação das Estruturas	8
1.3 A Cadeia Principal de uma proteína	8
1.3.1 Definição e propriedades intrínsecas	8
1.3.2 Geometria Local	10
1.4 Dados de Distâncias	12
1.4.1 Ressonância Magnética Nuclear	13
1.5 Roteiro da Tese	14
2 Discretizable Molecular Distance Geometry Problem	17
2.1 <i>Distance Geometry Problem</i>	18
2.1.1 Definição e Aplicações	18
2.1.2 Matrizes de Distâncias	24
2.2 <i>Molecular Distance Geometry Problem</i>	26
2.3 <i>Discretizable Molecular Distance Geometry Problem</i>	32
2.3.1 Discretização do MDGP	32
2.4 <i>Branch-and-Prune</i>	39
2.4.1 O Algoritmo	40
2.4.2 Estrutura Algorítmica	47
2.4.3 Representação Eficiente de Soluções	53
3 Simetrias do DMDGP	55
3.1 Simetria do Quarto Nível	56

3.2	Explorando o conjunto de simetrias	59
3.2.1	Contando o número de soluções de um K DMDGP	59
3.2.2	O conjunto dos vértices de simetria	63
3.3	<i>SymBP</i> : uma versão adaptada do BP para o uso de simetrias	65
4	Uma abordagem Dividir & Conquistar para o DMDGP	71
4.1	Motivações Preliminares	72
4.1.1	Uma versão paralela do algoritmo BP	72
4.1.2	Resolvendo o DMDGP por múltiplas árvores de solução	73
4.1.3	Uma versão paralela do <i>SymBP</i> : uma possibilidade	84
4.2	Dividir-e-Conquistar com Quatérnios em Geometria de Distâncias	86
4.2.1	Dividindo com Simetrias	86
4.2.2	Conquistando com <i>gaps</i> no conjunto de distâncias	101
4.2.3	Combinando com Quatérnios	115
4.2.4	Método <i>GapSym</i>	119
4.3	Resultados Computacionais	122
4.4	Resultados Computacionais	122
4.4.1	Testes Realizados com instâncias artificiais	124
4.4.2	Resumo	131
5	Conclusão	133
5.1	Contribuições da Tese	133
5.2	Trabalhos oriundos do Estudo	133
5.3	Trabalhos Futuros	134
Referências		136
A	Ordens em Grafos	143
A.1	Conceitos Preliminares	143
A.2	Ordem em V	143
B	Matrizes e Rotações	146
B.1	Álgebra de Matrizes e Grupos de Matrizes Clássicos	146
B.2	A exponencial de uma matriz	148
B.3	O Grupo de Rotações - $SO(3)$	148
B.3.1	Matriz de Rotações via Fórmula de Rodrigues	149
C	A Álgebra Geométrica dos Quatérnios e Rotações	153
C.1	Quatérnios e sua Álgebra	154
C.1.1	Produto Algébrico de Quatérnios	155
C.1.2	Forma Matricial do Produto Geométrico de Quatérnios	157
C.2	Quatérnios como Rotações	157
C.2.1	Forma Matricial do Operador L_q	160
C.2.2	Um operador de rotação em \mathbb{R}^3	160

C.2.3	Definir um Quatérnio de Rotação com eixo e ângulo	162
C.3	Contagens de Número de Operações com Quatérnios	162
D	Métodos Dividir-e-Conquistar para desenvolver algoritmos eficientes	164
D.1	Estrutura	164
D.2	Vantagens	165
D.3	Exemplos de Aplicações	167
I	Licença	168
I.1	Sobre a licença dessa obra	168

*À minha doce e belíssima esposa,
pela compreensão e companheirismo
nos tempos bons e nos tempos difíceis,
e à minha família,
que me mostra a cada dia
o caminho da Graça de Deus
e da Esperança.*

Agradecimentos

Inicialmente, e acima de todas as coisas, agradeço a Deus por todas as coisas. Tudo se fez nEle, por Ele e para Ele e sem Ele nada do que foi feito se fez. Me permito utilizar estas palavras para descrever a importância dEle para mim e para o meu trabalho.

Agradeço, de forma especial e romântica, à minha linda esposa **Juliéti Fidalgo** por ter partilhado comigo os anos de Doutorado (os quais não foram fáceis) com amor, carinho, compreensão, intrepidez, bravura, muita docura e companheirismo. Os meus dias contigo são mais do que especiais, meu amor, minha herança eterna!

De forma tão especial quanto, agradeço aos meus pais **Luiz Carlos e Margaret Fidalgo** por terem me dado vida. Por me ensinarem a lutar e não desistir diante das dificuldades. Por me dizerem do valor do ensino, do magistério e da carreira acadêmica. Amo vocês e me inspirarei em vocês eternamente! Também, especialmente, agradeço à minha única irmã e companheira “de sangue” **Ana Karina Fidalgo** por todo o amor, apoio e dedicação, e à minha vó **Anacyr Delphini** por dar à vida aquela pitada de aventura que só alguém que já viveu bastante pode ter.

Agradeço de forma mais que especial à minha segunda família: **Maria de Lourdes e Wagner Basaglia** e **Rosa Jurema**, além dos tão queridos **Adão** e **Luzinete Basaglia**, os quais sempre me ensinaram que o caminho da simplicidade é a maior alegria da vida.

Meus sinceros agradecimentos se estendem ao meu orientador **Prof. Dr. Carlile Lavor** que tantas vezes me deu “uma segunda chance”, que quando eu estava “aperreado” trazia sempre suas palavras de sabedoria a tira-colo. Agradeço por me ensinar os valores e a humildade todos os dias. Obrigado, chefe!

Sou grato aos meus pastores **Jetro** e **Léia Paiva** que sempre oraram por mim e também a toda a família da Terceira Igreja Batista de Andradina.

Agradeço aos meus amigos mais chegados do que irmãos: **Daniel e Bianca Barros Gomes**, **José Ricardo e Monique Barros Gomes**, **Marcos e Andréia Caliri**. A vida com vocês é sempre de par-em-par.

Obrigado, meu amigo e irmão **Carlos Renato Medeiros**, o Cabelo, pelos dias compartilhados, pelas madrugadas de risadas e pelos cafés incomparáveis.

Agradeço à minha família na UNICAMP: Eduardo Miqueles, Estevão Esmi, Juca (Heron Félix) e Josiane Gonzaga, Victor Aprikian, Murilo Basseto, Pedro Pitoli, Bruno Aramaki, Valter Camargo, Germano Abud, Jorge Alencar, Douglas Maioli e todos os outros.

Meus agradecimentos se estendem ao **Prof. Dr. Douglas Gonçalves** pela amizade e as preciosas contribuições para esta tese. No mesmo sentido, agradeço profundamente ao **Prof. Dr. Jaime Rodriguez** pela parceria desde os meus tempos de Graduação.

Agradeço aos membros da banca de defesa desta tese pelas preciosas dicas e contribuições com este trabalho e futuros: **Profa. Dra. Sueli Irene Costa, Prof. Dr. Eduardo Xavier Miqueles, Prof. Dr. Giuliano Gadioli La Guardia e Prof. Dr. Tiberius Bonates.**

Agradeço ao **Prof. Dr. Aurélio Ribeiro Leite de Oliveira** e ao **Prof. Dr. Laécio Carvalho de Barros** pelo apoio institucional que sempre me deram. Neste sentido, agradeço a todos os funcionários da Seção de Pós-Graduação do IMECC pelos excelentes serviços prestados com competência, prontidão e um “sorriso no rosto”.

Meus agradecimentos às Faculdades Integradas “Rui Barbosa” de Andradina por darem oportunidade de exercer minha profissão.

Agradeço, por fim, ao Conselho Nacional de Desenvolvimento Científico e Tecnológico (CNPq) pelo suporte financeiro durante todo o Doutorado.

“E conhecerão a **Verdade**
e a **Verdade** os libertará.”
- Jesus Cristo, o Caminho
da Verdade (João 8:32).

Listas de Ilustrações

1.1	Estrutura básica de dezenove dos vinte aminoácidos naturais [87].	3
1.2	Formação de uma proteína a partir da conexão entre uma sequência de aminoácidos. Há a liberação de uma molécula de água para cada uma das uniões [87].	3
1.3	Estrutura química de um aminoácido genérico [87].	4
1.4	Os bioquímicos americanos Linus Pauling e Robert Corey [74].	4
1.5	Figura representando uma alfa-hélice, extraída de Nelson <i>et al.</i> [74].	6
1.6	As duas formas de β -folhas: antiparalela e paralela, extraídas de [74].	7
1.7	Estrutura terciária da Mioglobina, proteína do esperma de baleias, extraída de [74].	7
1.8	Complexo <i>gvp - ssdna</i> , extraída de [87].	8
1.9	Comparação das quatro estruturas, extraída de [74].	8
1.10	A junção de aminoácidos com a formação da cadeia principal em destaque [75].	9
1.11	Ambas figuras representam o cerne de um aminoácido, explicitando, na média, os comprimentos de ligações peptídicas e ângulos entre elas, extraídas de [82].	9
1.12	Segundo Creighton, após observação dos átomos entre dois carbonos-alfa em pequenas cadeias peptídicas, cristalográficamente, obtiveram as seguintes dimensões [15].	10
1.13	Representação de r_{ij} e r_{jk} e de θ_{ik} , para os átomos consecutivos i , j e k	10
1.14	Duas imagens ilustram o ângulo diedral. A primeira a partir de três ligações consecutivas e a segunda a partir do ângulo entre dois planos com um vetor em comum [74, 87].	11
1.15	Representação dos ângulos diedrais em uma cadeia, extraída de [75].	12
1.16	Representação dos ângulos diedrais com os comprimentos de ligação, extraída [74]. .	12
1.17	Exemplos dos espectros de RMN uni e bi-direcional, extraído de [74].	13
1.18	Família de conformações possíveis para a proteína Globina, presente em vermes de sangue marinho, cuja função é transportar átomos de oxigênio, extraído de [74]. . .	14
1.19	Átomos de Hidrogênio pertencentes a aminoácidos distantes na cadeia principal de uma proteína podem ter sua distância calculada pelo dobramento da molécula, extraído de [77].	14
2.1	O austríaco Karl Menger (1902 - 1985), à esquerda, e o britânico Arthur Cayley, à direita, extraído de www.wikipedia.com	19
2.2	Duas soluções são possíveis para esse DGP, as quais são simétricas em relação ao segmento dado pelas posições dos vértices u e v	21

2.3	Infinitas (não-enumerável) soluções são possíveis para esse DGP, as quais estão em um círculo com centro na posição de v e raio igual a 5.	22
2.4	Mínimos locais e globais, figura extraída de [52].	23
2.5	Representação do grafo G , mencionado anteriormente.	26
2.6	Gordon Crippen	28
2.7	Timothy Havel	28
2.8	Representação dos comprimentos de ligação, ângulos de ligação e ângulos de torção, extraído de Lavor et al. [54].	36
2.9	Caso em que instância é não-discretizável, extraída de [56].	37
2.10	A discretização do problema passa pela binaridade fornecida pelo ângulo de torção: cada átomo i pode estar apenas em uma das duas posições i ou i' a fim de satisfazer a distância $d_{i-3,i}$, ao invés de um círculo contínuo de possibilidades como era anteriormente. Figura extraída de [54].	39
2.11	Interseção de três superfícies esféricas em \mathbb{R}^3 , contendo dois pontos [59].	40
2.12	Inicialização do algoritmo: três primeiros nós da árvore.	41
2.13	Branching: a ramificação em posições possíveis.	41
2.14	O primeiro caminho é factível e o segundo é podado.	46
2.15	O primeiro caminho é podado e o segundo é factível.	46
2.16	Os dois caminhos são infactíveis e, portanto, são podados.	47
2.17	Um exemplo da maneira na qual a árvore T é representada	48
2.18	Os três primeiros níveis da árvore T	49
2.19	Os quatro primeiros níveis da árvore T	49
2.20	Os cinco primeiros níveis da árvore T	50
2.21	Os seis primeiros níveis da árvore T até então: a primeira poda.	50
2.22	Seis primeiros níveis da árvore T até então: primeira solução encontrada.	51
2.23	Cinco primeiros níveis da árvore T	51
2.24	Seis primeiros níveis da árvore T : segunda solução encontrada.	51
2.25	Seis primeiros níveis da árvore T : metade dos caminhos explorados.	52
2.26	Recomeça-se a busca.	52
2.27	Escolhendo o primeiro caminho da segunda metade.	52
2.28	Árvore explorada completamente.	53
3.1	Duas realizações associadas pela troca do sinal em $\text{sen}(\omega_{i-3,i})$	57
3.2	Árvore explorada pela metade, encontrando as primeiras duas soluções: x^1 e x^2	58
3.3	A simetria do quarto nível aplicada em x^1 e x^2 , encontrando as soluções x^3 e x^4 . . .	59
3.4	Reflexão pelo hiperplano definido por x_{i-K}, \dots, x_{i-1} . Figura extraída de [55]. . . .	60
3.5	Árvore representando as soluções para G pelo BP clássico.	68
3.6	Árvore representando as soluções para G pelo SymBP.	69
4.1	A comunicação em cascata entre os processadores. Figura extraída de [69].	72
4.2	Árvore explorada completamente.	75
4.3	Primeira transformação Euclideana: uma translação na estrutura deslizante.	80
4.4	Segunda transformação: uma rotação plana em termos de θ na estrutura deslizante.	81

4.5	Terceira transformação: uma rotação espacial em termos do ângulo φ .	82
4.6	União Raiz - Raiz: árvores $T_{1,6}^-$, à esquerda, e $T_{4,8}^+$, à direita.	83
4.7	União Folha - Raiz: árvores $T_{1,6}^+$, à esquerda, e $T_{4,8}^+$, à direita.	83
4.8	União Folha - Folha: árvores $T_{1,6}^+$, à esquerda, e $T_{6,10}^-$, à direita.	83
4.9	A árvore T_1 associada a G_1 .	95
4.10	A árvore T_2 associada a G_2 .	96
4.11	A árvore T_1 associada a G_1 .	96
4.12	A árvore T_2 associada a G_2 .	96
4.13	A árvore T associada a G , com a primeira solução.	97
4.14	A sub-árvore T_1 .	99
4.15	A sub-árvore T_2 .	99
4.16	A sub-árvore T_3 .	99
4.17	A sub-árvore T_1 .	100
4.18	A sub-árvore T_2 .	100
4.19	A sub-árvore T_3 .	100
4.20	Árvore associada à instância G acima, cujas informações de distâncias estão em D_G .	102
4.21	Árvore associada à instância G^* acima, cujas informações de distâncias estão em D_{G^*} .	107
4.22	Árvore associada à instância G acima, cujas informações de distâncias estão em D_G .	110
4.23	Árvore associada à instância G^* acima, cujas informações de distâncias estão em D_{G^*} .	111
4.24	Primeira transformação na estrutura deslizante T_{i+1} : uma translação.	116
4.25	Segunda transformação na realização deslizante T_{i+1} : uma rotação plana em termos do ângulo θ e em torno do eixo gerado pelo vetor \mathbf{n} .	117
4.26	Terceira transformação: uma rotação espacial em termos do ângulo φ , dado entre as projeções de F_i e F_{i+1} no complemento ortogonal \mathbb{P} de $\langle E_i \rangle$, e do eixo \mathbf{m} paralelo a este espaço, a qual é definida na estrutura deslizante.	118
4.27	Matriz de distâncias de INSTANCIATEST100.	124
4.28	<i>spy</i> de D_{G_1}	125
4.29	<i>spy</i> de D_{G_2}	125
4.30	<i>spy</i> de D_{G_3}	125
4.31	<i>spy</i> de D_{G_4}	125
4.32	<i>spy</i> de $D_{G_1^*}$	125
4.33	<i>spy</i> de $D_{G_2^*}$	125
4.34	<i>spy</i> de $D_{G_3^*}$	125
4.35	<i>spy</i> de $D_{G_4^*}$	125
4.36	Solução com o BP clássico.	126
4.37	Solução com Simetrias e <i>Gaps</i> .	126
4.38	Representação gráfica da geometria de ambas as soluções em \mathbb{R}^3 .	126
4.39	Matriz de distâncias de INSTANCIATEST300.	127
4.40	<i>spy</i> de D_{G_7}	127
4.41	<i>spy</i> de $D_{G_7^*}$	127
4.42	<i>Spy</i> das duas maiores sub-instâncias.	127
4.43	Solução com o BP clássico.	129
4.44	Solução com Simetrias e <i>Gaps</i> .	129

4.45	Representação gráfica da geometria de ambas as soluções em \mathbb{R}^3	129
4.46	Matriz de distâncias de INSTANCIAEST500	129
4.47	<i>spy</i> de D_{G_3}	130
4.48	<i>spy</i> de $D_{G_3^*}$	130
4.49	<i>Spy</i> da maior sub-instância e de sua inversa.	130
4.50	Solução com o BP clássico.	131
4.51	Solução com Simetrias e <i>Gaps</i> .	131
4.52	Representação gráfica da geometria de ambas as soluções em \mathbb{R}^3	131
A.1	Exemplos de grafos sob as Ordens PEO, DVOP e Henneberg Tipo I em seus vértices, extraídos de Liberti et al. [57]	145
A.2	Exemplos das Ordens K -Trilateration e DDGP, extraídos de [57].	145
C.1	Ilustração do isomorfismo existente entre \mathbb{R}^3 e \mathbb{H}_0 .	158

Lista de Abreviaturas e Siglas

DGP Distance Geometry Problem

K **DDGP** K-Discretizable Distance Geometry Problem

K **DMDGP** K-Discretizable Molecular Distance Geometry Problem

DMDGP Discretizable Molecular Distance Geometry Problem

MDGP Molecular Distance Geometry Problem

DVOP Discrete Vertex Ordering Problem

BP Branch-And-Prune Algorithm

EDMCP Euclidean Distance Matrix Completion Problem

EDMCP $_K$ Euclidean Distance Matrix Completion Problem in dimension K

Capítulo 1

Introdução

“Molecular Biology is mankind’s attempt to figure out how God engineered His greatest invention - life. (...) I find it a great privilege to live in a time where God allows us to gain some insight into His construction plans, only a short step away from giving us the power to control life processes genetically. I hope it will be to the benefit of mankind and not to its destruction.”

Arnold Neumaier, em (Neumaier, 1997, [75]).

Quando a Ciência investiga as características fisiológicas intrínsecas aos seres vivos, dos mais variados tipos, tamanhos e origens, inevitavelmente se depara com os elementos de uma classe de moléculas chamadas de *Proteínas*. A saber, estas interferem em todas as características de um organismo vivo, como por exemplo, a informação genética contida nos ácidos nucleicos cuja expressão depende inteiramente delas (Creighton, [15]).

Uma lista com algumas das propriedades inerentes a estas moléculas, cuja maior parte foi extraída de (Creighton, [15]), é descrita nos itens abaixo.

- As enzimas, que consistem propriamente em proteínas, são essenciais para otimizar e acelerar as reações químicas no organismo vivo. Nisto consiste o principal objeto de estudo da Bioquímica: descobrir as funções das enzimas em sistemas de seres vivos.
- Estas moléculas transportam e armazenam uma gama grande de partículas, abrangendo de macromoléculas a elétrons. Por exemplo: as proteínas transmitem informações entre células específicas e órgãos, como os hormônios.
- Algumas delas, controlam o fluxo de moléculas que passam pelas membranas.
- São elas que ligam e desligam os genes no sequenciamento genético.
- Na parte físico-química, elas convertem energia química em energia mecânica no interior dos músculos.
- São imprescindíveis para sentidos humanos como visão e audição.
- Algumas, em si mesmo, produzem toxinas, como a proteína da carne bovina que pode desencadear a chamada Doença da Vaca-Louca [6].

1.1 Formação das Proteínas e sua estrutura química

Apesar de tamanha abrangência e variedade em sua existência e atuação, a formação dessas moléculas é feita de modo simples. Comentamos, aqui, um pouco de seu desenvolvimento.

Atribui-se a criação do termo *Proteína* a Jöns Jacob Berzelius (1770 - 1848), um dos químicos mais influentes do século XIX, como uma derivação da palavra *Proteios* (grego, “de primeira importância”), pois o mesmo acreditava que tais substâncias eram “fundamentais” para os seres vivos [87]. Entretanto, o primeiro a publicar este termo (neste sentido) em algum trabalho científico foi o químico holandês Gerardus Mulder (1802 - 1880) em dois artigos de 1838 [71, 72]. Neste, ele se referia às substâncias *albuminóides* que coagulavam em meio ácido e quando submetidas a altas temperaturas e afirmou que tais substâncias eram feitas a partir de um radical comum, a Proteína, e que esta possuía a mesma fórmula empírica, a menos de alguma variação na quantidade de Enxofre e Fósforo [42]. Toda a história dessa origem bem como as suas implicações estão no artigo “*The origin of the word protein*” de H. B. Vickery, publicado em 1950 [92].

Com o passar do tempo, os cientistas continuaram pesquisando as propriedades destes compostos químicos e descobriram, no início do século XX, que a degradação de tais proteínas liberava os chamados **aminoácidos**. A partir de então, foi possível a identificação de em torno de doze aminoácidos diferentes, os quais são liberados desta maneira [87]. Já em 1902, Hofmeister afirmou que elas eram formadas por uma sequencia destes aminoácidos [87], consistindo em uma cadeia linear destas moléculas conectadas umas às outras por ligações *peptídicas*, podendo haver repetições desses pedaços ao longo da estrutura [15]. Por fim, em 1940, os pesquisadores completaram a identificações dos vinte tipos de aminoácidos existentes naturalmente e que compõem os pedaços das proteínas [87]. Por este ponto de vista, pode-se dizer que existe uma determinada homogeneidade na formação destas moléculas. A diferença entre as proteínas está na ordem que esses aminoácidos aparecem na sequênciā que forma a tal cadeia polimérica. Além disso, são os genes que especificam os aminoácidos que participarão da combinação em cada proteína [87].

De acordo com Creighton [15], as proteínas são moléculas mais simples que a maioria dos polímeros, já que possuem sequências exatas de aminoácidos de modo que é possível saber, precisamente, o comprimento de cada cadeia. Entretanto, também, são mais complexas no sentido da vasta variedade de possibilidades de compor esses pedaços dos polímeros com aminoácidos, já que a maioria destes são compostos por pedaços de um ou dois compostos químicos que se repetem ao longo da cadeia. Esta sequência é que identifica a proteína e suas funções nos organismos vivos de forma a eliminar ambiguidades [15].

Segundo [15, 87], dezenove dos vinte aminoácidos existentes naturalmente são formados por um átomo de carbono no centro da estrutura, denominado de carbono-alfa C^α . A partir desse, para cada uma das quatro ligações possíveis que pode fazer, tem-se:

- (i) um átomo de *hidrogênio* H ;
- (ii) um *grupo amina* NH_3^+ ;
- (iii) um *grupo carboxílico* COO^- e
- (iv) uma cadeia lateral R - de *resíduo*.

A Figura 1.1 ilustra a disposição desses átomos em um aminoácido. O vigésimo aminoácido tem estrutura parecida com essa, entretanto é o único cuja cadeia lateral R está ligada ao átomo de nitrogênio pertencente ao grupo amina citado acima [15].

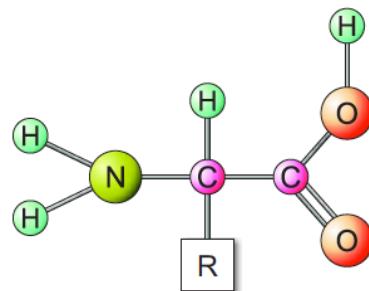


Figura 1.1: Estrutura básica de dezenove dos vinte aminoácidos naturais [87].

A Tabela 1.1 abaixo traz os nomes e as siglas dos aminoácidos presentes nos seres vivos [87].

Ala - Alanina	Arg - Arginina	Asn - Asparagina	Asp - Aspartato	Cys - Cisteína
Gln - Glutamina	Glu - Glutamato	Gly - Glycina	His - Histidina	Ile - Isoleucina
Leu - Leucina	Lys - Lisina	Met - Metionina	Phe - Fenilalanina	Pro - Prolina
Ser - Serina	Thr - Treonina	Trp - Triptofano	Tyr - Tirosina	Val - Valina

Tabela 1.1: Tabela com os nomes e siglas dos vinte aminoácidos existentes nos seres vivos.

Segundo Souza [87], nove dentre estes vinte aminoácidos não podem ser sintetizados pelos seres humanos, os quais são chamados de *Aminoácidos Essenciais* e devem ser incluídos na dieta diária. Esses aminoácidos essenciais são a Histidina, a Leucina, a Metionina, a Treonina, a Isoleucina, a Lisina, a Fenilalanina, o Triptofano e a Valina [87].

A informação contida no RNA induz a união do grupo carboxílico de um aminoácido ao grupo amina do aminoácido seguinte na sequência, formando uma ligação peptídica entre o átomo de carbono do grupo carboxílico com o átomo de nitrogênio do grupo amina, e ainda hidrolizando, ou seja, consumindo uma molécula de água (H_2O) nesta ligação [75, 87].

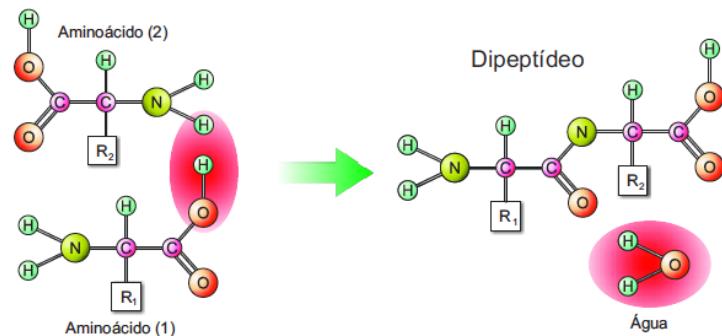


Figura 1.2: Formação de uma proteína a partir da conexão entre uma sequência de aminoácidos. Há a liberação de uma molécula de água para cada uma das uniões [87].

1.2 Estrutura Tridimensional das Proteínas

Ainda que se conheça a sequência de aminoácidos que compõe uma proteína, pode-se não saber ao certo a quais outras proteínas esta pode se associar e qual sua função nos organismos vivos [87].

Os genes assumem formas lineares, mas as proteínas podem se dobrar e assumir formas ainda mais curvilíneas tornando o trabalho de predizer sua estrutura tridimensional ainda mais árduo [87], mesmo que a Figura 1.3 induza a um pensamento sobre a linearidade da estrutura do aminoácido.

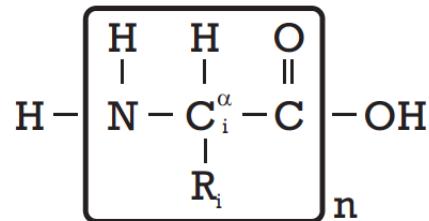


Figura 1.3: Estrutura química de um aminoácido genérico [87].

Arnold Neumaier publicou um artigo pioneiro cujo objetivo é descrever e analisar este problema de predição da estrutura tridimensional de uma proteína a partir de sua dada sequência de aminoácidos, o qual foi chamado de *Protein Folding Problem*. Suas dificuldades são exploradas por Neumaier, bem como as formulações matemáticas, seus métodos, suas deficiências e sua estabilidade numérica [75]. Tal problema é definido, literalmente, como “a tarefa de entender e predizer como a informação codificada na sequência de aminoácidos das proteínas, no momento de sua formação, traduz isto em uma estrutura tridimensional para a proteína ativa biologicamente.”



Figura 1.4: Os bioquímicos americanos Linus Pauling e Robert Corey [74].

Para estudar melhor uma estrutura molecular tridimensional, define-se como uma **Conformação** de uma molécula à disposição de seus átomos no espaço 3D [74]. E, como a estrutura de uma proteína possui centenas de ligações covalentes, podem haver muitas rotações livres ao redor delas. Assim, existem proteínas que podem assumir uma grande gama de conformações distintas [74].

Entretanto, cada proteína possui propriedades e funções estruturais e químicas tão específicas que fica fortemente sugerido a existência de apenas (e tão somente) uma estrutura tridimensional protéica, dentre todas as possibilidades mencionadas acima [74].

Portanto, é possível notar que as ligações covalentes entre átomos de uma proteína são de real importância para o estudo e a determinação de suas conformações quimicamente viáveis.

Os pais do estudo daquilo que entendemos pela estrutura de proteínas hoje-em-dia foram os bioquímicos americanos *Linus Pauling* (1901 - 1994) e *Robert Corey* (1897 - 1971) [74]. Eles procederam um amplo estudo sobre as ligações peptídicas em proteínas e descobriram, principalmente, os dois fatos seguintes:

- dois átomos de C_α referentes a dois aminoácidos consecutivos são separados por três ligações covalentes como



- e por sua ligação peptídica dupla, as ligações $C - N$ são rígidas, sem a possibilidade de rotacionar livremente, o que, portanto, é possível apenas para as duas ligações



No caso de quase todas as proteínas naturais, a cadeia de aminoácidos assume uma conformação tridimensional específica para cada uma, após ser dobrada e torcida [75, 87]. Esta proteína se torce no espaço de modo a atingir uma forma chamada de *Estado de Dobramento* [75]. Segundo Neumaier [75], esta configuração do estado de dobramento e os grupos ativos quimicamente na superfície desta proteína retorcida são quem determinam suas funções biológicas.

Algumas questões aparecem, no entanto, mostrando a urgência de diferenciações entre as proteínas, como: o que faz de uma proteína um hormônio? Ou uma enzima? Ou ainda uma proteína estrutural? Ou, como mais um exemplo, um anticorpo? Estas diferenciações passam, imprescindivelmente pela discussão de suas propriedades estruturais [74].

Além disso, tal estrutura tridimensional possuem alguns padrões. De modo a facilitar o estudo e a determinação das estruturas protéicas, observando tal regularidade, criou-se quatro subdivisões estruturais para elas. Ou seja, de uma certa forma, estabeleceu-se uma hierarquia conceitual a fim de classificá-las [74, 75, 15, 87]. Adiante, comentamos brevemente sobre esta classificação.

1.2.1 Estrutura Primária

A **Estrutura Primária** de uma proteína consiste em uma descrição de todas as ligações covalentes que ligam os aminoácidos dessa molécula (mostrando a relevância das ligações peptídicas).

A parte principal desta primeira estrutura é composta pela sequência ordenada dos resíduos R_i da cadeia de aminoácidos que formam a proteína [75, 87]. De fato, isso basta para diferenciar as proteínas em uma primira instância, já que os aminoácidos possuem quase que uma mesma estrutura atômica, apenas diferindo de um para o outro por seus resíduos.

Esta estrutura elimina ambiguidades e determina suas propriedades químicas e biológicas, codificadas pela sequência de aminoácidos. Além disso, ela aponta indiretamente para as estruturas dos próximos níveis da proteína. Este é o passo mais básico nesta determinação [15].

Como uma aplicação, as técnicas de sequenciamento de proteínas são necessárias para entender as modificações que ocorrem a uma cadeia polipeptídica depois de ser gerada a partir de sua sequência genética [15].

1.2.2 Estrutura Secundária

A **Estrutura Secundária** refere-se a uma secção arbitrária da cadeia polipeptídica a fim de descrever o arranjo espacial dos átomos da chamada Cadeia Principal (a ser definida e melhor estudada *a posteriori*), sem preocupar-se com os outros segmentos ou com as cadeias laterais [74].

Alguns exemplos dessa classe são mencionados como segue.

- (i) Uma **α - hélice** é definida pelos átomos de uma parte da sequência de aminoácidos em forma de hélice, que se retorce em torno de um eixo longitudinal imaginário em seu centro de modo que os resíduos R formam uma saliência em sua parte exterior [74, 75, 87]. Esta nomenclatura foi definida também por Pauling e Corey, motivados pelos resultados experimentais pioneiros de W. Astbury que submeteu proteínas como a queratina a raios-x. Estas hélices se contorcem de acordo com a Regra da Mão-Direita. Além disso, este tipo de conformação é mais comum, pois faz uso otimizado das ligações internas de hidrogênio [74].

A Figura 1.5 representa esta subestrutura de uma proteína por diferentes ângulos de visão.

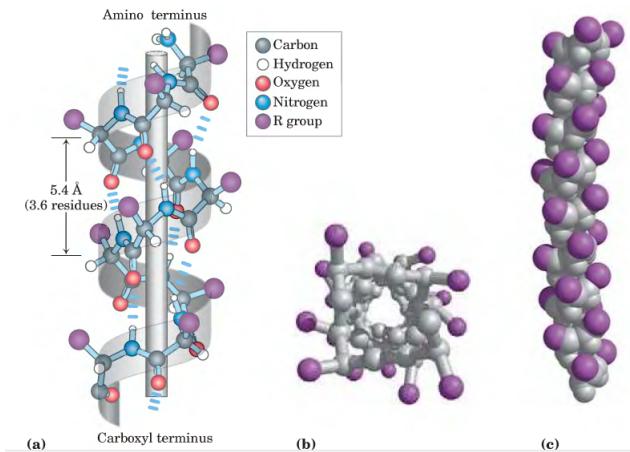


Figura 1.5: Figura representando uma alfa-hélice, extraída de Nelson *et al.* [74].

- (ii) Uma **β -folha** é uma estrutura que também foi identificada por Pauling e Corey, em 1951, classificada como uma β - conformação [74]. É mais longa e extensa do que as hélices e foi confirmada por análises de raios-x. Ao invés de ser organizada como uma hélice, esta conformação realiza um zigue-zague na cadeia polipeptídica e a união dessas cadeias ziguezagueantes se parecem com pregas. Ao invés de apontar na mesma direção, os grupos residuais da cadeia lateral alternam nos aminoácidos adjacentes, apontando em direções opostas [74].

A Figura 1.6 representa esta subestrutura por dois ângulos de visão diferentes e em duas subclasses chamadas de *Antiparalela* e *Paralela* [74].

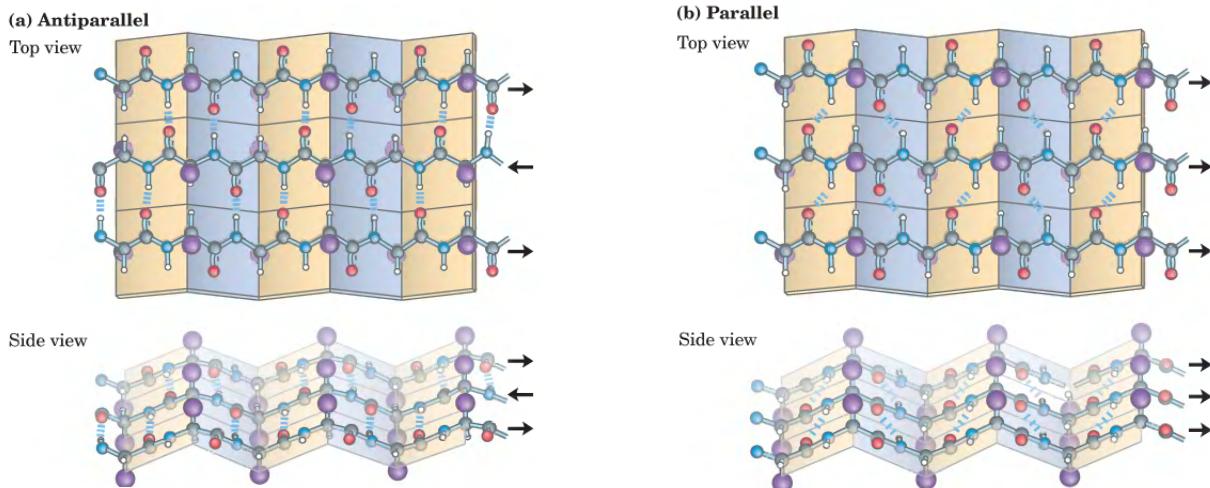


Figura 1.6: As duas formas de β -folhas: antiparalela e paralela, extraídas de [74].

1.2.3 Estrutura Terciária

A **Estrutura Terciária** consiste no arranjo tridimensional de todos os átomos em uma proteína [74]. Em comparação com as outras duas estruturas anteriores, esta se concentra na proteína como um todo, como no modo como a proteína toda se desdobra e retorce no espaço tridimensional, e não em propriedades locais, como a determinação de α -hélices. Assim, ela mapeia o modo como aminoácidos de partes distantes da molécula se aproximam no espaço, mesmo não sendo adjacentes. A Figura 1.7 representa a estrutura terciária da Mioglobina, encontrada no esperma de baleias [74].

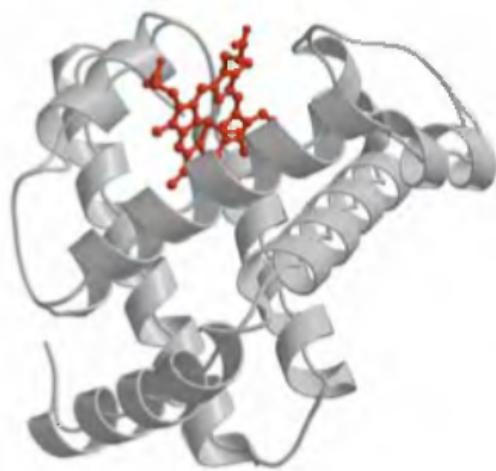


Figura 1.7: Estrutura terciária da Mioglobina, proteína do esperma de baleias, extraída de [74].

1.2.4 Estrutura Quaternária

Por fim, a **Estrutura Quaternária** de uma proteína se refere ao arranjo tridimensional de subestruturas independentes de proteína, ou seja, que não sejam interligadas mas pertençam à mesma molécula [74, 87]. É o padrão no qual estas estruturas cristalizam-se, que não é tão interessante do ponto de vista biológico [75]. Um exemplo é o Complexo *gvp - ssdna*, visto na Figura 1.8.

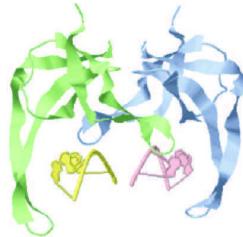


Figura 1.8: Complexo *gvp - ssdna*, extraída de [87].

1.2.5 Comparação das Estruturas

A Figura 1.9 permite a comparação entre essas quatro classes. Neumaier ([75], p.4) diz da avidez em que os bioquímicos desejavam entender como o estado de dobramento de uma proteína (estrutura terciária) se originava a partir da sequência de resíduos dos aminoácidos em um polipeptídeo (estrutura primária), revelando uma relação entre essas seções.

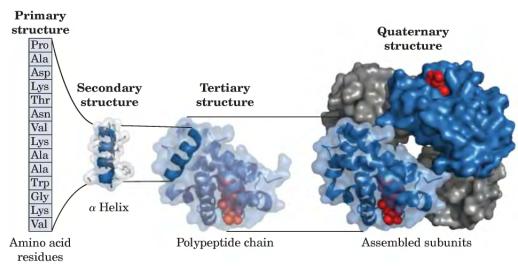


Figura 1.9: Comparação das quatro estruturas, extraída de [74].

1.3 A Cadeia Principal de uma proteína

1.3.1 Definição e propriedades intrínsecas

Neste trabalho, nos interessa, apenas, uma subestrutura da proteína chamada de *Cadeia Principal*, a qual possui uma geometria rica e tem sido estudada recentemente por matemáticos. Nesta seção, estudaremos sua geometria local, motivação para o estudo posterior.

A cadeia principal de uma proteína é composta por uma sequência repetida de três átomos centrais, provenientes de cada um dos aminoácidos, ligados covalentemente:

- um átomo de nitrogênio N , proveniente de uma molécula de amido;
- o chamado carbono-alfa C^α e
- o carbono C , oriundo de uma carbonila [15].

Após a junção dos aminoácidos, como na Figura 1.10, esta cadeia é definida pela sequênciа

$$\dots - N - C^\alpha - C - N - C^\alpha - C - N - \dots$$

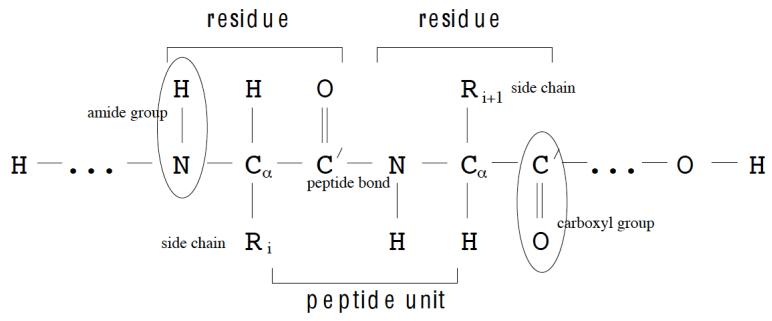


Figura 1.10: A junção de aminoácidos com a formação da cadeia principal em destaque [75].

Ela funciona como a espinha dorsal da proteína (em inglês *Backbone*, uma tradução literal).

Os comprimentos das ligações peptídicas e dos ângulos entre elas são fixos, na média, a menos de erro de medição. Diz-se, então, que estes são *dados a priori*. Ramachandran *et al.* fizeram um estudo sobre a geometria média das proteínas a partir de dados cristalográficos [82]. A Figura 1.11, extraída do artigo original, traz informações sobre esses tipos de dados, que se repetem nos pedaços da cadeia polimérica.



Figura 1.11: Ambas figuras representam o cerne de um aminoácido, explicitando, na média, os comprimentos de ligações peptídicas e ângulos entre elas, extraídas de [82].

Creighton também discorre sobre esses valores médios para as distâncias entre átomos na cadeia principal e os ângulos de ligações, como na Figura 1.12 [15].

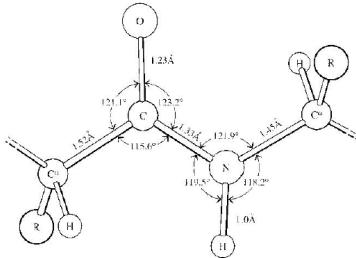


Figura 1.12: Segundo Creighton, após observação dos átomos entre dois carbonos-alfa em pequenas cadeias peptídicas, cristalograficamente, obtiveram as seguintes dimensões [15].

1.3.2 Geometria Local

Neumaier [75] traz definições geométricas dos itens concernentes a conformações de proteínas.

Considerando uma ordenação dos átomos na cadeia principal (que existe e é dada pelos ângulos de torção, como veremos *a posteriori*), define-se matematicamente a posição do i -ésimo átomo da cadeia principal no espaço Euclidiano \mathbb{R}^3 como o **Vetor de Coordenadas**

$$\mathbf{x}_i = \begin{bmatrix} x_{i1} \\ x_{i2} \\ x_{i3} \end{bmatrix}.$$

Esta definição é amplamente usada, como se pode ver em estudos como [75, 24, 54].

Dessa forma, tomando dois átomos i e j , cujas posições respectivas em \mathbb{R}^3 são \mathbf{x}_i e \mathbf{x}_j , unidos por uma ligação, define-se o **Vetor de Ligação** associado a i e j como

$$\mathbf{r}_{ij} = \mathbf{x}_j - \mathbf{x}_i,$$

cujo comprimento é dado pela equação $\|\mathbf{r}_{ij}\| = \sqrt{\langle \mathbf{r}_{ij}, \mathbf{r}_{ij} \rangle}$, utilizando a norma Euclidiana $\|\cdot\|$ [75].

Como já mencionado, esses comprimentos de ligação $r_{i,i+1}$ são dados *a priori*. Além disso, dados três átomos consecutivos i, j e k (nesta ordem da cadeia principal), define-se o **Ângulo de Ligação** $0 \leq \theta_{ik} \leq \pi$ entre os vetores \mathbf{r}_{ij} e \mathbf{r}_{jk} , calculado a partir das expressões

$$\cos(\theta_{ik}) = \frac{\langle \mathbf{r}_{ij}, \mathbf{r}_{jk} \rangle}{\|\mathbf{r}_{ij}\| \|\mathbf{r}_{jk}\|} \quad \text{e} \quad \sin(\theta_{ik}) = \frac{\mathbf{r}_{ij} \times \mathbf{r}_{jk}}{\|\mathbf{r}_{ij}\| \|\mathbf{r}_{jk}\|},$$

sendo que \times é o produto vetorial usual de \mathbb{R}^3 [75, 87].

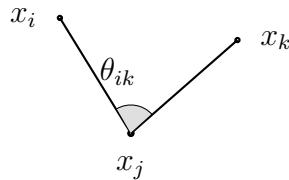


Figura 1.13: Representação de \mathbf{r}_{ij} e \mathbf{r}_{jk} e de θ_{ik} , para os átomos consecutivos i, j e k .

Ângulos de torção

Geometricamente, os trabalhos de *Pauling* e *Corey* mostraram que os átomos da cadeia principal de uma proteína podem ser representados a partir de uma série de planos rígidos consecutivos, encaixados e ordenados, segundo a ordem implícita existente nos aminoácidos, de modo que dois planos consecutivos possuem intersecção apenas em um ponto relativo a um átomo de C^α [74]. Matematicamente, é bem sabido, da Geometria Euclidiana, que três pontos no espaço determinam, unicamente, um subconjunto de pontos referente a um plano. Assim, aplicando à cadeia principal de uma proteína, cada conjunto $\{N, C^\alpha, C\}$ estabelece um plano rígido no espaço e o carbono alfa como o único ponto que dá liberdade de rotação rígida para eles. Esta rigidez das ligações nos planos limitam o número de conformações possíveis para esta proteína [74].

A partir destes resultados, definiu-se três ângulos chamados de **Ângulos Diedrais** (ou Ângulos de torção), representados pelas letras gregas ϕ, ψ e ω . Cada um desses três tipos de ângulos diedrais representam as rotações possíveis ao redor das três ligações peptídicas que fornecem graus de liberdade [74]. Define-se um ângulo diedral como o ângulo dado pela interseção de dois planos em \mathbb{R}^3 . Tais planos são definidos, a partir dos vetores de ligação, da seguinte forma:

- (i) dois vetores de ligação consecutivos, definidos na cadeia principal de um polipeptídeo, (que possuam, desta maneira, um átomo de intersecção) definem um plano em \mathbb{R}^3 e
- (ii) três vetores de ligação consecutivos, definidos na mesma cadeia principal, definem dois planos os quais possuem uma ligação em comum (a ligação central).

Assim, três vetores de ligação consecutivos induzem a existência do ângulo de torção entre os dois planos formados por eles [74]. A Figura 1.14 ilustra a formação geométrica desse ângulo.

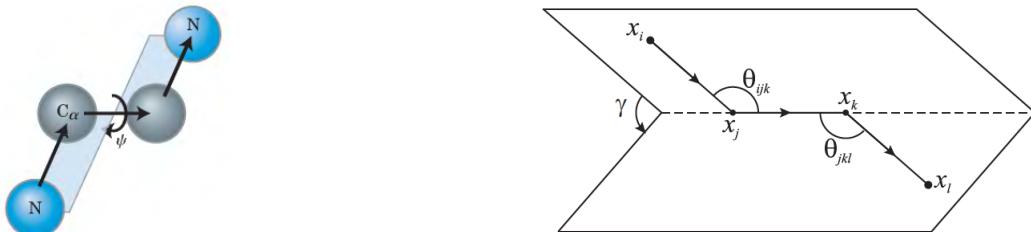


Figura 1.14: Duas imagens ilustram o ângulo diedral. A primeira a partir de três ligações consecutivas e a segunda a partir do ângulo entre dois planos com um vetor em comum [74, 87].

Procedemos, agora, com uma descrição detalhada sobre esses ângulos, a partir de [74, 75, 87]. Os que nos interessam, em um polipeptídeo, são os ângulos diedrais definidos por três vetores de ligação consecutivos, referentes a quatro átomos consecutivos da cadeia principal. Os principais são dados pelas letras gregas ϕ, ψ e ω e definidos como segue.

- (1) ϕ envolve as ligações entre os quatro átomos consecutivos $C - N - C^\alpha - C$, de modo que a rotação se dá em torno da ligação $N - C^\alpha$.
- (2) ψ é definido pelas ligações entre os quatro átomos $N - C^\alpha - C - N$, onde a variação deste ângulo é dada pela rotação em torno de $C^\alpha - C$.

- (3) Por fim, ω envolve as ligações $C^\alpha - C - N - C^\alpha$, cuja ligação central é peptídica, ou seja, a rotação em torno dela é restrita. Ele nem sempre é considerado em razão dessa rigidez.

A Figura 1.15 representa os três ângulos de torção supracitados na cadeia principal.

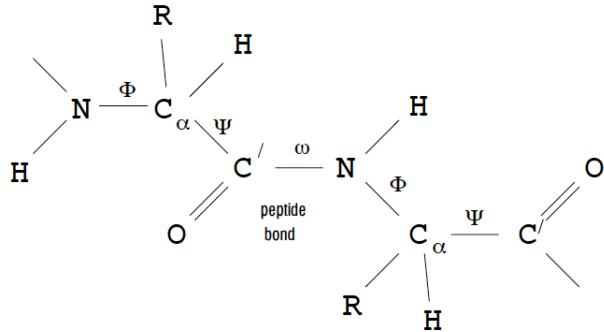


Figura 1.15: Representação dos ângulos diédrais em uma cadeia, extraída de [75].

Nelson *et al.* [74] também traz uma representação dos ângulos diédrais na Figura 1.16.

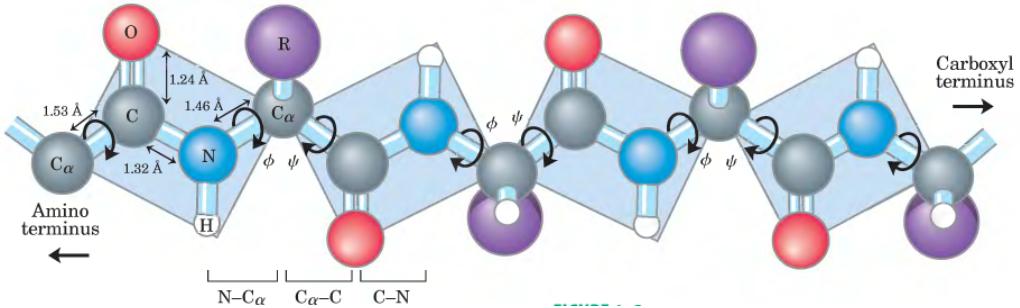


Figura 1.16: Representação dos ângulos diédrais com os comprimentos de ligação, extraída [74].

Matematicamente, esses são dados como o ângulo entre os vetores normais aos planos respectivos [75]. Logo, como em [75], seja $\omega \in [-\pi, \pi]$ o ângulo diédral formado pelos planos determinados, respectivamente, pelos átomos consecutivos i, j, k e j, k, l . Ele pode ser calculado a partir de

$$\cos(\omega) = \frac{\langle r_{ij} \times r_{jk}, r_{jk} \times r_{kl} \rangle}{\|r_{ij} \times r_{jk}\| \|r_{jk} \times r_{kl}\|} \quad \text{e} \quad \sin(\omega) = \frac{\langle r_{kl} \times r_{ij}, r_{jk} \rangle \|r_{jk}\|}{\|r_{ij} \times r_{jk}\| \|r_{jk} \times r_{kl}\|}.$$

O sinal de ω é dado por $\langle r_{kl} \times r_{ij}, r_{jk} \rangle$. Além disso, se temos um conjunto *completo* de comprimentos de ligações, ângulos de ligações e diédrais, então a geometria proteíca é sobre determinada [75, 87].

1.4 Dados de Distâncias

Os dados de distâncias, utilizados para reconstruir estruturas de proteínas, são obtidos através de processos físicos e químicos. Nesta seção, trataremos do mais comum: a **Ressonância Magnética Nuclear (RMN)**.

1.4.1 Ressonância Magnética Nuclear

A RMN é um processo físico ao qual as macromoléculas, como as proteínas, são submetidas com o objetivo de se extrair suas propriedades estruturais. É uma manifestação do momento angular do *spin* do núcleo de alguns isótopos, como Hidrogênio - 1, Carbono - 13, Nitrogênio - 15, Flúor - 19 e o Fósforo - 31 [74]. Em sua maioria, a RMN observa a estrutura dos hidrogênios e, ainda assim, para alguns grupos peptídicos, tais átomos só são detectados quando em meio com pH ligeiramente mais ácido, onde não há tanta troca desses átomos com o solvente aquoso [15].

As seguintes informações estão em [74]. Na RMN *Unidirecional*, um pulso de energia eletromagnética, com rádio-frequência adequada é aplicado por um ângulo correto aos núcleos alinhados no campo magnético. Certa energia é absorvida de modo a alternar os núcleos a um estado de alta-energia, onde o espectro de absorção resultante contém informações sobre suas identidades e o ambiente em que se encontram. As amostras são submetidas a diversas aplicações para o cálculo de uma média, gerando um espectro como na Figura 1.17.

O Hidrogênio (H) é um elemento de especial importância para a RMN, por sua alta sensibilidade e abundância na natureza [74]. Apesar disso, o espectro dele via RMN unidirecional pode ter análise complicada. Esta determinação para proteínas tornou-se de mais fácil manipulação com a criação da RMN bi-direcional, cujo espectro é ilustrado na Figura 1.17. Assim é possível medir o acoplamento de *spins* nucleares dependentemente da distância para átomos próximos espacialmente [74]. Isto é, pode-se estabelecer uma rede de distâncias entre pares de átomos próximos [87]. Isto chama-se **Efeito Overhauser Nuclear (NOE)** [74, 87]. Segundo [87], este é o nome dado à transferência de polarização de *spins* entre populações de átomos. Tais transferências geram picos no espectro de RMN. Os sinais NOE dão informações sobre alguns átomos e, para que façam sentido e sejam úteis, é necessário identificar quais deles geram tais sinais, o que torna a tarefa de estabelecer uma estrutura tridimensional a partir de RMN bi-direcional algo trabalhoso [74]. A sequência de aminoácidos da proteína também é um importante fator da determinação da mesma.

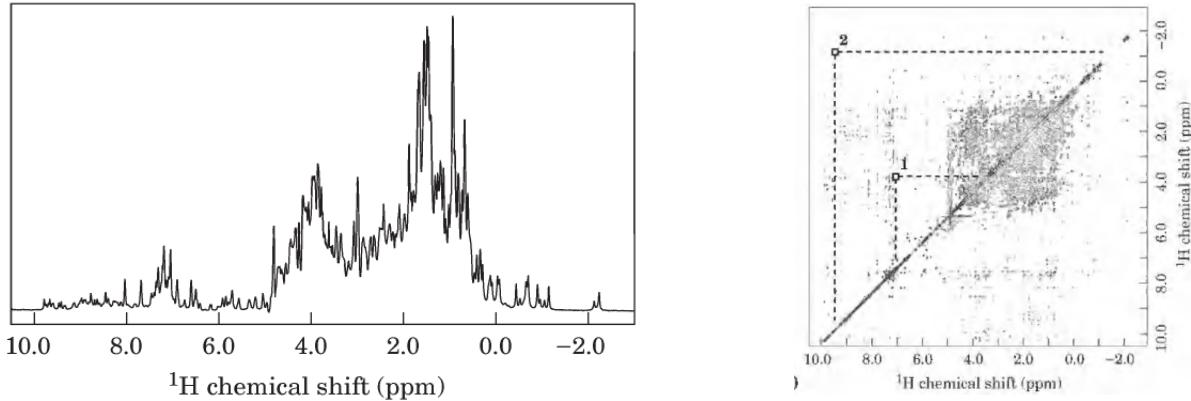


Figura 1.17: Exemplos dos espectros de RMN uni e bi-direcional, extraído de [74].

Para compor a estrutura tridimensional, une-se os dados de distâncias a restrições geométricas, como quiralidade, raios *van der Waals* e comprimentos e ângulos de ligação, que são processados em computadores gerando uma família de conformações possíveis pelos sinais NOE [74].

Considere a proteína Globina, encontrada em vermes de sangue marinho, parente da Mioglobina, proveniente de esperma de baleias, cuja função é transportar oxigênio. Os dados de RMN uni e bi-direcionais, fornecidos na Figura 1.17, referem-se a ela. A diagonal no espectro de um experimento de RMN bi-direcional corresponde ao espectro da RMN uni-direcional. Os dados bi-direcionais permitem encontrar a família de possíveis conformações para a globina, dada na Figura 1.18 [74]. Na prática, os sinais NOE são detectados para átomos de hidrogênio que não estejam mais distantes um do outro do que 5 angstrons, ou para alguns, até 6 angstrons [15, 87].

Como visto anteriormente, os valores de distâncias entre átomos e alguns ângulos de ligação e diedrais são conhecidos, na média, determinando uma aproximação da geometria local de aminoácidos e partes das proteínas, como a cadeia principal. Entretanto, como a proteína possui uma conformação que se dobra no espaço, a RMN é capaz de detectar distâncias entre átomos de hidrogênio que estejam consideravelmente “distantes” na molécula no sentido da ordem dos átomos na cadeia polipeptídica [87, 77]. Isto é representado na Figura 1.19.



Figura 1.18: Família de conformações possíveis para a proteína Globina, presente em vermes de sangue marinho, cuja função é transportar oxigênio, extraído de [74].

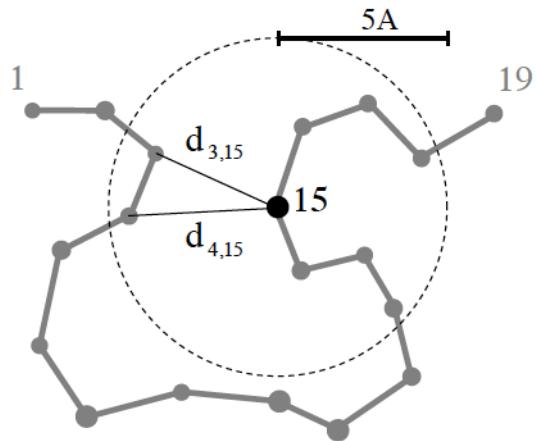


Figura 1.19: Átomos de Hidrogênio pertencentes pelo dobramento da molécula, extraído de [77].

1.5 Roteiro da Tese

Com a motivação do estudo da conformação de proteínas para encontrar sua estrutura no espaço Euclidiano de três dimensões, dada a sua importância para entender as funções biológicas das mesmas, como vimos na Introdução deste trabalho, desenvolveu-se um problema inverso chamado *Discretizable Molecular Distance Geometry Problem* (DMDGP) a fim de modelar a busca por tais conformações. Este é o objeto principal do Capítulo 2. Ele aparece como uma particularização de um problema chamado *Distance Geometry Problem* (DGP) e é estudado sob o olhar da Teoria de Grafos. Ainda neste capítulo, analisamos a estrutura do principal método que resolve este problema, o algoritmo *Branch & Prune* (BP).

Já no Capítulo 3, exploramos os conceitos de simetrias que aparecem na estrutura DMDGP. Esta será uma ferramenta de vital importância das contribuições originais desta tese. Devido a esta relevância, tal assunto passou a compor um capítulo próprio.

Por fim, o Capítulo 4 é a parte principal deste trabalho. Como o problema que deseja-se resolver já está estabelecido e as principais ferramentas já foram apresentadas, ele inicia com as motivações que nos conduziram ao objetivo deste estudo: possibilitar a definição de uma estratégia (ou estratégias), usando o paradigma Dividir-e-Conquistar (D&C), para melhorar o uso do algoritmo BP na resolução do DMDGP. Isto é feito em três frentes, baseado no paradigma D&C:

- (1) **Dividir** uma instância DMDGP usando os vértices de simetria, a fim de englobar as infactibilidades nas sub-instâncias,
- (2) **Conquistar** cada sub-instância, utilizando o que chamamos de *gaps*, e
- (3) **Combinar** as soluções para cada “pedaço” utilizando rotações de Quaternios.

Os resultados computacionais trazem resultados promissores no sentido de indicar uma melhora com o uso dessa estratégia, possibilitando o estabelecimento de trabalhos futuros e colaborações a fim de melhorar ainda mais (Conclusão).

Ainda há quatro apêndices para consulta sobre temas adjacentes. O primeiro traz um breve comentário sobre ordens em grafos, utilizada na definição de instâncias DMDGP, o segundo traz um *survey* sobre a definição de matrizes de rotação arbitrárias usando a Fórmula de Olinde Rodrigues, o terceiro traz um resumo sobre a Álgebra dos Quaternios e Rotações e, por fim, o quarto apêndice traz um breve resumo sobre o paradigma Dividir e Conquistar..

Capítulo 2

Discretizable Molecular Distance Geometry Problem

Neste capítulo, estudaremos uma classe de problemas geométricos baseados na busca pelas conformações factíveis de determinadas estruturas de pontos em \mathbb{R}^3 . Esta determinação deve ser realizada a partir da utilização de dados de distâncias que estiverem disponíveis entre os pontos da estrutura que representem os átomos neste espaço.

O objeto principal desse estudo, na prática, é a determinação de conformações para Proteínas (estruturas), as quais foram tratadas no capítulo introdutório desta tese, onde os pontos considerados devem ser os lugares geométricos factíveis com os “núcleos” dos átomos.

Para isso, devemos analisar a história, a evolução, os fundamentos e os principais resultados de uma série de problemas estritamente geométricos, baseados nos valores de distâncias entre pontos, o que vem sendo ampla e progressivamente estudado ao longo dos últimos anos [57].

Delinearemos um esboço da progressão deste capítulo. Inicialmente, faremos uma introdução com o chamado *Distance Geometry Problem* (DGP) - ou, em uma tradução à Língua Portuguesa, Problema de Geometria de Distâncias - que consiste no problema em sua roupagem mais geral, cujos pontos podem representar quaisquer objetos.

A seguir, considera-se a posição do núcleo de um átomo de uma certa molécula como um ponto no espaço e, assim, torna-se possível aplicar o DGP a estruturas moleculares de forma mais geral [19, 57]. Esta aplicação permitiu a definição de uma sub-classe de problemas do DGP, à qual deu-se o nome de *Molecular Distance Geometry Problem* (MDGP) - ou, traduzido também, o Problema de Geometria de Distâncias Moleculares. Veremos que este é um problema de otimização global contínua cujo espaço de busca por soluções é contínuo (não-enumerável) para o qual existem alguns métodos de solução descritos na literatura. Entretanto, não usaremos esta abordagem do problema, mas o modelo que lança mão da Teoria de Grafos.

Por fim, com o objetivo de modelar estruturas geométricas espaciais de Proteínas, restringiu-se as instâncias do MDGP a um subconjunto especial que goza de certas propriedades adicionais. Este sub-problema deu lugar a um novo problema chamado de *Discretizable Molecular Distance Geometry Problem* (DMDGP) - ou Problema Discretizável de Geometria de Distâncias Moleculares. Este nome origina-se no fato de que o espaço de busca pelas soluções deste problema é discreto e finito, como veremos com mais detalhe mais adiante.

O DMDGP é o problema que ocupa posição central no estudo apresentado neste trabalho. Para resolvê-lo, existe um mecanismo de solução na literatura, chamado de Algoritmo *Branch & Prune* (BP), cujos detalhes serão descritos no final deste capítulo. Em resumo, este é um algoritmo robusto que realiza uma busca (que, de certa maneira, segue uma determinada orientação) pelo conjunto discreto de soluções de um DMDGP. A mais importante distinção entre ele e outros métodos da literatura é que este é capaz de encontrar, de forma razoavelmente rápida, todas as soluções factíveis, enquanto que os outros encontram apenas uma [56].

Além disso, descobriu-se a existência de relações de simetria em vértices dessas estruturas. Dessa forma, prova-se que basta que o *Branch & Prune* encontre apenas uma solução e todas as outras podem ser determinadas através de reflexões parciais. Com isso, a busca que ele realiza por soluções torna-se ainda mais robusta, pois necessita de um espaço menor para a exploração. Utilizaremos este fato também neste trabalho.

Enfim, a missão deste trabalho é desmembrar um DMDGP em sub-problemas DMDGP menores, resolvê-los usando o *Branch & Prune* e, então, unir as soluções dos subproblemas usando uma estratégia também eficiente através de transformações Euclidianas. Com isso, é possível ganhar em tempo de computação, além de criar uma estrutura para que, no futuro, essa determinação seja feita com o uso de computação em paralelo.

2.1 *Distance Geometry Problem*

2.1.1 Definição e Aplicações

A história e os conceitos apresentados nesta seção se baseiam, em sua maioria, em um *survey* elaborado por Leo Liberti e colaboradores [57] e recém publicado pela SIAM Review.

Em 1928, o matemático austríaco *Karl Menger* (1902 - 1985), filho do renomado economista *Carl Menger*, publicou um artigo no periódico *Mathematische Annalen* sob o título *Untersuchungen über allgemeine Metrik* que, traduzido literalmente ao Português, significa “Estudos sobre a métrica geral”. Neste estudo, Menger traz à luz uma caracterização de diversos conceitos geométricos em termos de valores de distâncias entre pontos em espaços Euclidianos, tais como congruência e convexidade de conjuntos [65, 57]. Assim como Menger, o britânico *Arthur Cayley* (1821 - 1895) também investigou problemas neste sentido. E, foi partindo dos trabalhos de Menger e Cayley supracitados que o matemático americano *Leonard Mascot Blumenthal* (1901 - 1984) deu início a uma nova área de investigação da Ciência sob o nome de **Distance Geometry** (DG) [57], com a publicação de seu livro *Theory and Applications of Distance Geometry* em 1953 [8]. Para ele, o problema mais importante nesta área era o chamado “Problema do Subconjunto” (ou *Subset Problem*, originalmente) que consistia em encontrar condições necessárias e suficientes a fim de decidir se uma matriz simétrica era, de fato, uma matriz cujos valores representam distâncias entre pontos [57]. Na direção de encontrar soluções para este problema, uniu-se resultados dos trabalhos de Cayley [11] e Menger [64], culminando em condições para solução que dependem de uma ferramenta chamada *Determinante de Cayley - Menger* [57].

Contemporâneo a Menger, Schoenberg [86], em 1935, descobriu uma relação entre este problema e as formas bilineares, a partir de uma caracterização alternativa [87].

Blumenthal definiu este problema como segue:

“Quando temos como dado um conjunto de distâncias entre pares de pontos, a geometria das distâncias pode dar uma dica para encontrar um conjunto de coordenadas correto para pontos no espaço Euclídeo tridimensional, satisfazendo as restrições de distâncias dadas.”

(Blumenthal, 1953, [8])

Mas, foi em um artigo de Yemini, em 1978, que a primeira referência explícita sobre o conceito contemporâneo do DGP foi dada. Neste, o autor referiu-se a ele como Problema de Posicionamento e a seguinte definição foi fornecida:

“O Problema de Posicionamento surge quando é necessário localizar um conjunto de objetos geograficamente distribuídos usando medidas de distâncias entre alguns pares de objetos.”

(Yechiam Yemini, 1978, [96])

No ano seguinte, Yemini publicou outro artigo flexibilizando esta definição por considerar dados de distâncias esparsas, isto é, levando em conta algumas distâncias entre pares de objetos. Com este *conjunto esparso de distâncias*, introduziu-se o que se chamou de *Problema Posição - Localização*, para o qual deseja-se calcular a localização de todos os objetos no espaço geográfico no qual estão inseridos [95, 57]. Além disso, faz uma análise de complexidade computacional para alguns problemas de rigidez em grafos no mesmo trabalho [57].

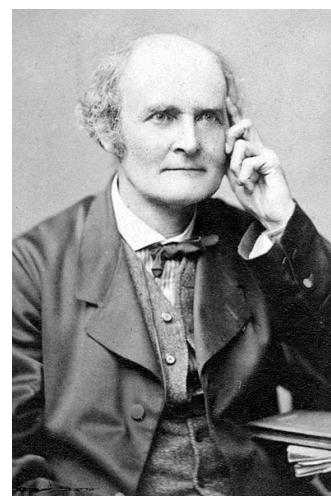


Figura 2.1: O austríaco Karl Menger (1902 - 1985), à esquerda, e o britânico Arthur Cayley, à direita, extraído de www.wikipedia.com.

Nesse interim, J. Saxe publicou um trabalho lidando com os mesmos aspectos [83], também em 1979, onde chama o DGP de “*K - Embeddability Problem*”. Neste, ele mostrou que:

- para $K = 1$, este problema é **NP** - Completo e
- para $K > 1$, em geral, ele é **NP** - Difícil [57].

Como um resumo desta nova área do conhecimento, Havel declara que:

“Geometria de Distâncias é uma metodologia geral que formula problemas de conformação em termos de restrições de distância e quiralidade”.

(Timothy Havel, 1995, [38])

Assim, o problema fundamental de GD pode ser caracterizado através da Teoria de Grafos, uma moderna caracterização elaborada por Liberti *et al* [57].

Distance Geometry Problem (DGP). Dado $K \in \mathbb{Z}_+$ e um grafo simples e não-direcionado $G = (V, E)$ cujas arestas são ponderadas pelos valores da função não-negativa $d : E \rightarrow \mathbb{R}_+$, encontre imersões $x : V \rightarrow \mathbb{R}^K$ tais que

$$\forall \{u, v\} \in E, \quad \|x_u - x_v\| = d_{u,v}. \quad (2.1.1)$$

Para simplificar a notação, tem-se que $x_u = x(u)$, $x_v = x(v)$ e os valores das distâncias são representados por $d_{u,v} = d(\{u, v\})$, os quais são calculados usando a distância proveniente da norma Euclidiana usual $\|\cdot\|$.

Em resumo, resolver um DGP é associar cada vértice de G a um ponto em \mathbb{R}^K de modo que se satisfaçam, simultaneamente, as Equações (2.1.1). Ou seja, deseja-se encontrar imersões de G neste espaço de modo a preservar suas características métricas [57, 52].

Cada imersão x que satisfaz tais condições do DGP acima é chamada de **Realização** da instância G em \mathbb{R}^K . Vamos denotar o conjunto de todas soluções de um DGP como \mathcal{S} .

É fácil ver que qualquer realização de uma instância G do DGP em \mathbb{R}^K pode ser transformada em outra realização distinta em \mathbb{R}^K , a partir de uma combinação de rotações e translações [52]. Com isso, pode-se afirmar que o número de realizações de uma instância é infinito não-enumerável. Se as soluções que puderem ser encontradas dessa maneira forem desconsideradas, procedimento a ser adotado neste trabalho a partir de agora, o conjunto \mathcal{S} pode ser:

- Vazio;
- Finito;
- Infinito Não-Enumerável.

Desconsiderar essas estruturas determinadas por rotações e translações significa fixar unicamente o sistema de coordenadas para o espaço a fim de representar as realizações “diferentes”, que preservem as propriedades métricas.

Agora, considere dois casos sobre a cardinalidade de uma instância.

- (i) Se G possui dois vértices u e v e $E = \{\{u, v\}\}$, então existe uma única realização para G ;
- (ii) Já, se G possui três vértices u , v e r e $E = \{\{u, v\}, \{u, r\}, \{v, r\}\}$, então o número de realizações dependerá de que os pesos das arestas satisfaçam as desigualdades triangulares entre eles, as quais são dadas por

$$\begin{cases} d_{uv} \leq d_{ur} + d_{vr}, \\ d_{ur} \leq d_{uv} + d_{vr} \quad \text{e} \\ d_{vr} \leq d_{uv} + d_{ur}. \end{cases} \quad (2.1.2)$$

Se elas forem satisfeitas, o número de realizações é finito. Caso contrário, não existem realizações para o DGP. Portanto, a desigualdade triangular é uma importante restrição para caracterizar a existência ou não de soluções.

Abaixo, seguem alguns exemplos de instâncias $G = (V, E, d)$ do DGP em \mathbb{R}^2 .

Exemplo 2.1.1 (\mathcal{S} é vazio). Tomando $V = \{u, v, r\}$ e $E = \{\{u, v\}, \{v, r\}, \{u, r\}\}$, onde $d_{uv} = d_{vr} = 1$ e $d_{ur} = 3$, temos que a desigualdade triangular é violada por

$$3 = d_{ur} > d_{uv} + d_{vr} = 2. \quad (2.1.3)$$

Logo, não existem pontos que satisfazem esse triângulo no plano. Por isso, não é possível determinar uma realização $x = (x_u, x_v, x_r)$ para G em \mathbb{R}^2 , ou seja, o conjunto-solução X é vazio.

Exemplo 2.1.2 (\mathcal{S} é finito). Considere $V = \{u, v, r\}$ e $E = \{\{u, v\}, \{v, r\}, \{u, r\}\}$, onde $d_{uv} = 4$, $d_{vr} = 5$ e $d_{ur} = 3$. É fácil ver que é possível tomar ao menos um triângulo que satisfaça essas restrições de distâncias no plano. Na verdade, existem dois triângulos cujos vértices são soluções desse DGP. Basta considerar a aresta $\{u, v\}$ como o eixo de simetria de dois triângulos, fixando uma única posição para os vértices u e v , permitindo que r possa ser posicionado em duas posições simétricas em relação à aresta citada. Tais soluções podem ser observadas como na Figura 2.2.

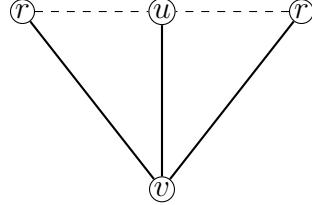


Figura 2.2: Duas soluções são possíveis para esse DGP, as quais são simétricas em relação ao segmento dado pelas posições dos vértices u e v .

Uma mudança simples no conjunto E pode alterar extremamente a cardinalidade do conjunto-solução \mathcal{S} , como vemos na comparação entre o Exemplo 2.1.2 anterior e o Exemplo 2.1.3 seguinte.

Exemplo 2.1.3. Sejam $V = \{u, v, r\}$ e $E = \{\{u, v\}, \{v, r\}\}$, onde as distâncias associadas às arestas são dadas por $d_{uv} = 4$ e $d_{vr} = 5$. Fixando as posições de u e v unicamente (a fim de eliminar as rotações e translações), a quantidade de posições possíveis para o vértice r é infinito não-enumerável, já que qualquer ponto no círculo de centro na posição do vértice v e de raio igual a 5 é uma posição possível. Isso pode ser visualizado na Figura 2.3. Portanto, ao retirar uma aresta de E , o DGP deixa de ter duas soluções e passa a ter infinitas soluções.

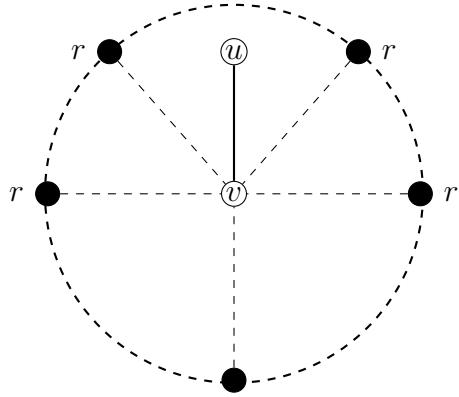


Figura 2.3: Infinitas (não-enumerável) soluções são possíveis para esse DGP, as quais estão em um círculo com centro na posição de v e raio igual a 5.

De acordo com [5], é impossível obter um conjunto-solução para um DGP que seja infinito-enumerável. Não há uma maneira simples de se demonstrar esse fato, mas usando ferramentas de Geometria Algébrica [52].

Ainda que utilizamos a Teoria de Grafos para estudar este problema neste trabalho, a abordagem clássica para um DGP é dada como um problema de Otimização Global contínua [52], como veremos a seguir. Sabe-se que encontrar uma solução para o conjunto de equações não-lineares (2.1.1) que definem o DGP envolve severas dificuldades numéricas. Mesmo para instâncias pequenas com menos do que dez vértices, o sistema não consegue ser resolvido computacionalmente [57].

Uma abordagem alternativa que se encontrou foi lançar mão de Programação Matemática (PM) [57]. Como motivação, considere o seguinte exemplo proposto e analisado em Lavor *et al* [52]. Seja $G = (V, E, d)$ uma instância DGP para $K = 2$ onde $V = \{u, v, s\}$ e $E = \{\{u, v\}, \{v, s\}\}$ cujo sistema não-linear dado por (2.1.1) associado é

$$\begin{cases} \|x_u - x_v\| = d_{uv} \\ \|x_v - x_s\| = d_{vs} \end{cases} . \quad (2.1.4)$$

Elevando essas duas equações ao quadrado, temos

$$\begin{cases} (x_{u_1} - x_{v_1})^2 + (x_{u_2} - x_{v_2})^2 = d_{uv}^2 \\ (x_{v_1} - x_{s_1})^2 + (x_{v_2} - x_{s_2})^2 = d_{vs}^2 \end{cases} . \quad (2.1.5)$$

Logo, é possível reescrever este sistema como

$$\begin{cases} (x_{u_1} - x_{v_1})^2 + (x_{u_2} - x_{v_2})^2 - d_{uv}^2 = 0 \\ (x_{v_1} - x_{s_1})^2 + (x_{v_2} - x_{s_2})^2 - d_{vs}^2 = 0 \end{cases} . \quad (2.1.6)$$

Considerando, então, a função polinomial de grau 4 $f : \mathbb{R}^6 \rightarrow \mathbb{R}$, dada por

$$f(x_{u_1}, x_{u_2}, x_{v_1}, x_{v_2}, x_{s_1}, x_{s_2}) = ((x_{u_1} - x_{v_1})^2 + (x_{u_2} - x_{v_2})^2 - d_{uv}^2)^2 + ((x_{v_1} - x_{s_1})^2 + (x_{v_2} - x_{s_2})^2 - d_{vs}^2)^2 . \quad (2.1.7)$$

Logo, é fácil perceber que uma solução global (minimizador global) $x^* \in \mathbb{R}^6$ do problema de Otimização Não-Linear

$$\min_{x \in \mathbb{R}^6} f(x) \quad (2.1.8)$$

é uma solução para (2.1.6) e, portanto, uma solução para o DGP (2.1.4), desde que sejam compatíveis com o conjunto de distâncias [52].

Portanto, em generalização a este raciocínio, resolver um DGP passa a ser encarado como o problema de encontrar minimizadores globais $x \in \mathbb{R}^{3K}$ para a função

$$f(x) = \sum_{\{u,v\} \in E} \left(\|x_u - x_v\|^2 - d_{uv}^2 \right)^2. \quad (2.1.9)$$

A essência dessa abordagem é entender que o objetivo principal desse programa não-linear é minimizar as infactibilidades quadradas de pontos de \mathbb{R}^K que estejam na variedade \mathcal{M} , definida pelo lugar-geométrico dos pontos que satisfazem as Equações (2.1.1) [57].

Observe, também, que esse é um problema de Otimização Global Contínua Não-Linear, o qual é não-convexo e irrestrito, cuja função-objetivo é um polinômio não-negativo de grau 4 com a propriedade de que

$$x \in \mathcal{S} \iff f(x) = 0. \quad (2.1.10)$$

Mesmo que a abordagem descrita acima seja clássica na teoria e nas aplicações de Geometria de Distâncias, ela também possui um inconveniente: a função f , dada por (2.1.7), que se quer minimizar, possui diversos mínimos locais, dificultando a determinação dos mínimos globais (os menores dos mínimos locais) que se deseja encontrar [52]. A situação com que se depara neste problema é representada na Figura 2.4.

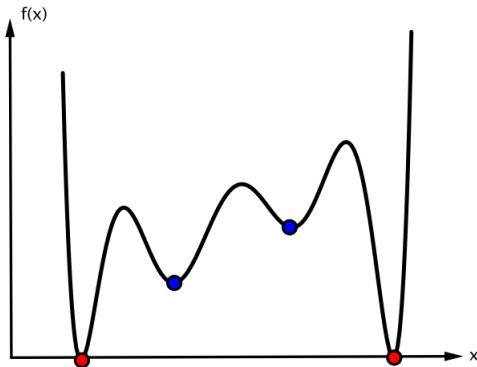


Figura 2.4: Mínimos locais e globais, figura extraída de [52].

Essa situação ainda é complicada pelo fato de que a quantidade de mínimos locais cresce exponencialmente em função do tamanho do problema, que é dado pela quantidade de vértices do grafo associado [52, 57].

Em relação a complexidade do problema, ao associar-se um DGP com o problema de Saxe, mencionado anteriormente, temos o seguinte teorema.

Teorema 2.1.1 (Lavor [52], Saxe [83]). O DGP é um problema **NP**-completo, para $K = 1$, e **NP**-difícil, para $K = 2$.

Portanto, ao que vimos, a essência em resolver um DGP é encontrar um conjunto de pontos em um determinado espaço geométrico que satisfaz todas as restrições de distâncias entre eles que são conhecidas. Assim, a partir da modelagem via Teoria dos Grafos, discutida nesta seção, esse problema inverso de estudar a geometria que envolve distâncias tem se consolidado e ampliado a sua abrangência, demonstrando grande poder interdisciplinar para resolver questões e desafios que residem na intersecção de uma grande gama de Ciências puramente teóricas e aplicadas. Os vértices de uma instância DGP podem representar corpos celestes, pontos de articulações de robôs, sensores ou antenas, átomos em moléculas e até pessoas em uma rede social, dentre outras possíveis aplicações. Abaixo, segue uma lista com algumas das áreas de aplicações desse problema.

- Astronomia;
- Estática;
- Estatística;
- Estruturas Geométricas;
- Estruturas Moleculares;
- Psicologia;
- Reconhecimento de Padrões;
- Redes de Computadores;
- Robótica;
- Teoria de Códigos;
- Teoria de Grafos;
- Visualização de Dados, dentre outras.

2.1.2 Matrizes de Distâncias

Nesta seção, apresentaremos algumas definições e conceitos que serão de vital importância para a visualização de alguns aspectos do estudo proposto neste trabalho. Todas as definições a seguir são baseadas em (Liberti *et al.*, 2014, [57]).

A partir da estrutura matricial para mapear a adjacência em grafos (especialmente, no caso de grafos ponderados), é possível representar o conjunto de distâncias de um DGP através de uma matriz quadrada simétrica.

Definição 2.1.1. Uma **Matriz de Distâncias** é uma matriz quadrada $D_{n \times n}$ que representa um espaço métrico finito (X, d) , onde $X = \{x_1, \dots, x_n\}$, e que é definida através de

$$D_{u,v} = d_{u,v} = d(x_u, x_v), \quad \text{para } 0 \leq u, v \leq n.$$

Neste caso, os pontos do espaço métrico discreto X são conhecidos e todas as distâncias podem ser calculadas, formando uma matriz completa de distâncias. Por outro lado, pode-se estabelecer o procedimento inverso. Antes, segue uma importante definição [1].

Definição 2.1.2. Uma matriz $D = (d_{i,j}) \in \mathbb{R}^{n \times n}$ com elementos não-negativos e diagonal nula é chamada de **Pré-Matriz de Distâncias** (ou Matriz de Dissimilaridade).

Surge, naturalmente, uma questão. Uma pré-matriz de distâncias pode representar um espaço métrico discreto? A definição abaixo tange este questionamento para o caso Euclidiano.

Definição 2.1.3. Uma matriz $D_{n \times n}$ é dita ser uma **Matriz de Distâncias Euclidianas** (MDE) se existe um inteiro $K > 0$ e um conjunto $X = \{x_1, \dots, x_n\} \subset \mathbb{R}^K$ tal que para $i, j \leq n$, temos

$$D_{i,j} = \|x_i - x_j\|.$$

De posse desse conceito, pode-se definir um problema de decisão que funcione a partir de uma extensão e um completamento das lacunas dessa matriz a fim de que esta seja uma MDE quadrada, simétrica, completa e tenha a diagonal nula. Este problema é chamado de *Euclidean Distance Matrix Completion Problem (EDMCP)* - ou Problema de Completamento de Matrizes de Distâncias Euclidianas - e é geralmente tratado como um problema de Otimização.

Há uma vasta literatura sobre este assunto e enumeramos algumas dessas referências a seguir. Os trabalhos de Gower [35], Farebrother [27] e Bakonyi *et al.* [3] fornecem uma perspectiva teórica deste problema, bem como algumas de suas propriedades e resultados interessantes. Já os trabalhos de Johnson [43], Alfakih *et al.* [1] e Huang *et al.* [41] fazem uma conexão entre este tipo de problema e problemas de Programação Semidefinida (SDP), o que provê um ferramental ainda mais poderoso para sua resolução. Por fim, é possível encontrar uma descrição mais completa da teoria e abordagens de resolução em Laurent [50].

Há que salientar, adicionalmente, que em vários dos trabalhos citados acima, uma MDE é definida com seus valores de distâncias ao quadrado.

Como descrito em Lavor [54], pode-se estabelecer uma íntima ligação entre um DGP e um EDMCP, em se considerando que no primeiro interessa-se em encontrar as posições dos pontos e, no segundo, em se encontrar as entradas desconhecidas na matriz de distâncias. Os valores de distâncias de um DGP estão em $d(E)$ e podem ser, respectivamente, armazenados em uma matriz Euclidiana de distâncias $D = (d_{i,j})$, onde $d_{i,j} = d(\{i, j\})$, se $\{i, j\} \in E$ e $d_{i,j} = 0$, caso contrário, ou seja, se não houver informação de distância entre os vértices i e j associados.

Neste trabalho, não estudaremos tais completamentos, mas utilizaremos estas matrizes de distâncias apenas como armazéns dos valores dos comprimentos das arestas de E , utilizando certa regularidade que existirá sob determinadas hipóteses a serem definidas ainda. Portanto, esta seção também tem objetivo introdutório.

Antes de prosseguirmos com as particularizações e aplicações do DGP, considere o seguinte exemplo para o modo como se define tais MDEs a partir de um DGP.

Seja $G = (V, E, d)$ um grafo não-orientado, ponderado e simples com seis vértices tal que

$$E = \{\{1, 2\}, \{1, 4\}, \{1, 6\}, \{2, 5\}, \{2, 6\}, \{3, 4\}, \{3, 5\}, \{4, 5\}, \{5, 6\}\},$$

o qual está representado geometricamente na Figura 2.5.

A matriz de distâncias Euclidianas associada a esta instância do DGP é dada por

$$D_G = \begin{bmatrix} 0 & d_{1,2} & 0 & d_{1,4} & 0 & d_{1,6} \\ d_{1,2} & 0 & 0 & 0 & d_{2,5} & d_{2,6} \\ 0 & 0 & 0 & d_{3,4} & d_{3,5} & 0 \\ d_{1,4} & 0 & d_{3,4} & 0 & d_{4,5} & 0 \\ 0 & d_{2,5} & d_{3,5} & d_{4,5} & 0 & d_{5,6} \\ d_{1,6} & d_{2,6} & 0 & 0 & d_{5,6} & 0 \end{bmatrix}. \quad (2.1.11)$$

Como nosso intuito é apenas o de armazenamento de dados nessas matrizes Euclidianas, vamos usar a estrutura de simetria para armazenar apenas metade dela. A outra metade é considerada como zeros. Assim, de uma matriz quadrada e simétrica com diagonal nula, ela se torna uma matriz triangular superior com diagonal nula

$$D_G = \begin{bmatrix} 0 & d_{1,2} & 0 & d_{1,4} & 0 & d_{1,6} \\ 0 & 0 & 0 & 0 & d_{2,5} & d_{2,6} \\ 0 & 0 & 0 & d_{3,4} & d_{3,5} & 0 \\ 0 & 0 & 0 & 0 & d_{4,5} & 0 \\ 0 & 0 & 0 & 0 & 0 & d_{5,6} \\ 0 & 0 & 0 & 0 & 0 & 0 \end{bmatrix}. \quad (2.1.12)$$

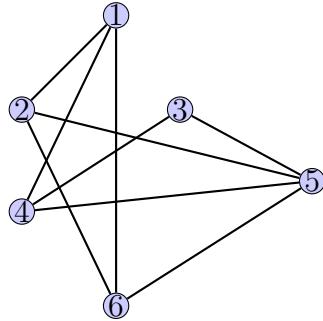


Figura 2.5: Representação do grafo G , mencionado anteriormente.

2.2 Molecular Distance Geometry Problem

No capítulo introdutório desta tese, pode-se encontrar um resumo com algumas características de uma série de moléculas fundamentais presentes nos seres vivos e, ainda, algumas informações sobre experimentos com Ressonância Magnética Nuclear (RMN) aplicada a estas. Tais informações foram introduzidas a fim de motivar o que segue a partir desta seção.

Sabe-se, adicionalmente, que a espectroscopia da RMN, em particular o experimento NOESY, fornece uma grande gama de dados com informações sobre os átomos que prontamente podem ser interpretadas em termos de restrições de distâncias em intervalos, a fim de englobar as imprecisões do próprio método experimental [38].

Além disso, pode-se considerar conceitos relativos à simetria de moléculas como um conjunto de pontos (átomos), como, por exemplo, o conceito de **Quiralidade** [37, 78]. Uma definição histórica deste é a seguinte:

“Eu chamo qualquer figura geométrica ou grupo de pontos de ‘quiral’, e diz-se possuir quiralidade, se sua imagem em um espelho plano, idealmente realizado, não pode ser conduzido à coincidência com ela mesma.”

(Lord Kelvin, 1904, [45])

Dessa maneira, a simples utilização desses dados de distâncias e quiralidade auxiliou na precisa formulação matemática de um problema, bem como de novas idéias e métodos eficientes para resolvê-lo, com o objetivo principal de determinar a estrutura geométrica de moléculas, como as proteínas. Este passo é fundamental pois as suas composições químicas e a sua estrutura tridimensional estão intimamente associadas com o conhecimento de suas funções na natureza.

Antes de prosseguir na definição desse problema e seus métodos para resolução, é preciso introduzir certa nomenclatura de itens pertencentes à Geometria Molecular (GM), as quais foram extraídas de Crippen & Havel [21].

- Uma **molécula** consiste em um grupo de átomos que permanecem intactos espacialmente em uma determinada escala temporal.
- Sua **estrutura** é dada pelas características que permanecem invariantes com o tempo.
- A **conformação** de uma molécula é sua estrutura espacial precisa em um instante de tempo fixo. No caso de moléculas orgânicas (objeto de estudo deste trabalho) a transformação de uma conformação em outra é dada, geralmente, por rotações em ligações simples.
- O **Espaço de Conformações** se refere ao conjunto de todas as conformações geradas pelos movimentos intra-moleculares.

Muitas tentativas têm sido realizadas a fim de se calcular o mais eficientemente possível as conformações de várias moléculas. Em particular, moléculas orgânicas como Proteínas e polipeptídeos [20]. Entretanto, este problema pode ser surpreendentemente difícil de se resolver, especialmente, se estivermos lidando com moléculas de grande porte (grande quantidade de átomos) [16].

A abordagem tradicional efetua esta determinação utilizando o Hamiltoniano do sistema (uma solução contendo a molécula de modo estático) [21]. Ou seja, leva-se em consideração a “energia” associada a cada conjunto de possíveis posições para os átomos [21].

Portanto, considere uma função de energia $f(\mathbf{x})$ aplicada a uma conformação $\mathbf{x} = (x_1, \dots, x_n)$, codificada em um vetor de n ângulos diedrais correspondentes às rotações ao redor de ligações simples [20]. De acordo com Crippen [20], o domínio I dessa função f (que corresponde ao conjunto de conformações da molécula) consiste em um espaço não-Euclidiano dado por uma variedade toroidal, pois cada x_i é uma rotação determinada por um ângulo (em graus, por exemplo) e considerando que $0^\circ = 360^\circ$. Portanto, conclui-se que este conjunto de conformações I é limitado.

Ao longo dos anos, diversos químicos têm desenvolvido funções analíticas que permitem estimar a energia como função das posições atômicas no espaço [21]. Um modo prático de se estimar essa energia do sistema é aproximar tal função por uma função semi-empírica da forma

$$f = \sum_{i \neq j} g_{ij}(d_{ij}), \quad (2.2.1)$$

onde d_{ij} representa o valor da distância entre quaisquer pares de átomos i e j da molécula [20].

De posse dessa função quantificadora da energia em função de conformações da molécula, o procedimento usual é aplicar técnicas de otimização global a fim de encontrar os mínimos globais de f [20, 48]. Um benefício dessa abordagem é a presença, normalmente, de um menor número de parâmetros invariantes com respeito a movimentos rígidos [21]. Por outro lado, dois inconvenientes podem ser facilmente detectados:

- na maioria das vezes estas funções de energia são muito complicadas de se lidar em termos das chamadas coordenadas internas da molécula [20] e
- as superfícies de energia potencial para determinadas moléculas contém alto índice de mínimos locais, dificultando a localização precisa de mínimos globais [48].

Um método de resolução, para este problema de encontrar conformações com energia mínima, muito comum nos idos dos anos 1970 (veja, por exemplo, o artigo de H. Scheraga [84]) era o de se posicionar os átomos em simples posições iniciais, satisfazendo as restrições geométricas dadas *a priori* (ângulos e comprimentos de ligação fixos), e alterar essa conformação inicial utilizando pequenos *shifts* nessas coordenadas atômicas a fim de encontrar outras posições para este conjunto de átomos que satisfaçam as mesmas restrições geométricas. Este processo continua até que alguma função tipo-energia correspondente a todas as distâncias interatômicas seja mínima [16].

Foi nessa conjuntura em que o químico-biofísico americano Gordon M. Crippen (Figura 2.6) foi o pioneiro em utilizar Geometria de Distâncias para calcular conformações de estruturas moleculares, adaptando as idéias desenvolvidas por Leonard Blumenthal, o que foi publicado no artigo “*A Novel Approach to Calculation of Conformation: Distance Geometry*” no ano de 1977 no periódico *Journal of Computational Physics* [16]. Neste artigo, Crippen propõe encontrar essas posições atômicas utilizando os “meios incomuns” (como o próprio autor descreve) de matrizes de distâncias que, sujeitas a certas condições necessárias e suficientes, garantem a determinação de uma conformação “energeticamente favorável” [16]. Nos anos seguintes, várias evoluções foram conseguidas por Crippen e colaboradores, como Irwin D. Kuntz e Timothy F. Havel (Figura 2.7) [18, 48, 39, 49] no sentido de melhorar a utilização de GD nesta descoberta das conformações moleculares.

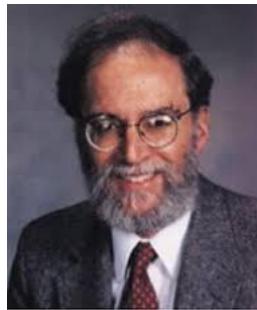


Figura 2.6: Gordon Crippen



Figura 2.7: Timothy Havel

O *turning point* para transformar GD em um método científico padrão neste sentido foi a compilação de todas essas idéias no livro “*Distance Geometry and Molecular Conformation*”, de G. Crippen e T. Havel em 1988. Ele descreve os aspectos teóricos e práticos dessa abordagem e ainda propõe um algoritmo para encontrar conformações usando GD, chamado EMBED [21]. Estes métodos que utilizam coordenadas Cartesianas para descrever estruturas moleculares são favoritos na Química Computacional desde então, pois são abordados por meio de uma álgebra mais simples e ainda evitam a utilização excessiva de funções trigonométricas, que sempre mostraram um elevado custo computacional [21].

Uma grande dificuldade encontrada nesta abordagem deve-se à possibilidade de encontrar valores de distâncias inconsistentes (o conjunto de restrições não satisfaz às desigualdades triangulares possíveis). Entretanto, uma das maiores conveniências deste método é dada pela possibilidade de definição clara e objetiva do problema. Assim, por exemplo, as inconsistências nas distâncias demonstram que existe algum problema na medição e/ou coleta de dados experimentais ou na interpretação geométrica dos dados. Por outro lado se elas são consistentes, então deve existir um conjunto de posições Cartesianas que satisfazem tais restrições [21].

Assim, Crippen e Havel [21] definem o problema molecular fundamental associado à Geometria de Distâncias da seguinte maneira:

Dadas as restrições de distância e quiralidade, que definem uma molécula, encontre uma ou mais conformações que as satisfazem ou, então, demonstre que elas não existem.

Depois dessa descrição histórica sobre o problema de encontrar conformações moleculares, passaremos à parte que mais interessa. O objetivo principal desta seção é analisar a já conhecida associação entre o problema fundamental da Geometria de Distâncias e a determinação de estruturas moleculares, enunciando e referenciando técnicas e métodos de solução.

As informações de distâncias $d_{i,j}$ entre os átomos i e j de uma estrutura molecular, estimadas experimentalmente (e.g., por RMN), estão sujeitas a ruído e incertezas e, por isso, são dadas em termos de limitantes inferior $l_{i,j} \geq 0$ e superior $u_{i,j} \geq 0$ tais que

$$l_{i,j} \leq d_{i,j} \leq u_{i,j}, \quad (2.2.2)$$

como se pode encontrar em [21, 85]. Assim, em termos matemáticos, o problema fundamental descrito nesta seção possui uma redefinição em termos de distâncias, a qual foi descrita por Jorge Moré e Zhijun Wu em [66] e cujo nome foi definido como *interval Molecular Distance Geometry Problem* por Lavor *et al.* em [53].

interval Molecular Distance Geometry Problem (iMDGP) ([53, 59, 57]). Seja S^i um conjunto de pares ordenados de átomos i e j de uma molécula com m átomos associados a valores de distâncias $d_{i,j}$ intervalares, as quais estão estimadas por limitantes inferiores e superiores ($u_{i,j}, l_{i,j}$), respectivamente, devido a imprecisões nos dados. Encontre posições $x_1, \dots, x_m \in \mathbb{R}^3$ para eles tais que satisfaçam as restrições

$$l_{i,j} \leq \|x_i - x_j\| \leq u_{i,j}, (i, j) \in S^i. \quad (2.2.3)$$

Cada molécula, então, é representada em \mathbb{R}^3 como um conjunto de pontos de modo que cada uma dessas posições representa o núcleo de um átomo da estrutura [19].

Mesmo que na prática os valores de distâncias sejam fornecidos da maneira em que se descreve acima, este trabalho não considera as possíveis imprecisões existentes nos valores de distâncias entre pares de átomos. Se levará em conta, portanto, distâncias com valores exatos, como também é feito em outras abordagens, e.g., [24, 23, 94, 56, 28, 54, 32] de modo que

$$l_{i,j} = u_{i,j} = d_{i,j}. \quad \forall (i, j) \in S^i, i \neq j. \quad (2.2.4)$$

Logo, este problema pode ser redefinido em termos exatos, da forma que se segue, cuja apresentação é formulada em Moré *et al* [67]. O nome referido a este problema, neste trabalho, será *Molecular Distance Geometry Problem*, como mencionado em Liberti *et al.* [59].

Molecular Distance Geometry Problem (MDGP) ([21, 67, 59]). Seja S um conjunto de pares de átomos i e j para os quais se conhece o valor exato da distância d_{ij} , para uma molécula com m átomos. Encontre posições $x_1, \dots, x_m \in \mathbb{R}^3$ para eles tais que satisfaçam as restrições

$$\|x_i - x_j\| = d_{ij}, \quad \forall (i, j) \in S . \quad (2.2.5)$$

Na prática, o sistema (2.2.5) não é resolvido por métodos exatos, mas de maneira aproximada considerando $\|\|x_i - x_j\| - d_{ij}\| < \varepsilon$, onde o erro ε é definido de acordo com a aplicação.

Ainda na modelagem do MDGP, a cada conjunto S , associa-se o conjunto $D_S = \{d_{ij} : (i, j) \in S\}$ que, mais a frente, será visto de forma mais fácil como imagem de um determinado mapa.

Na literatura, existe uma divisão para o MDGP em duas classes, dependendo de S .

- (i) Se $S = \{1, 2, \dots, n\}^2$, diz-se que D_S é um conjunto *completo* de distâncias exatas, isto é, para o qual se conhece os valores de distâncias entre *todos* os pares de átomos. Assim, o MDGP associado a ele pode ser resolvido em tempo linear, como em Dong e Wu [24].
- (ii) Caso contrário, se $S \subsetneq \{1, 2, \dots, n\}^2$, então, diz-se que D_S é um conjunto *incompleto* (ou *esparsa*) de distâncias exatas. Neste caso, o MDGP é classificado como **NP** - difícil [83].

Esta definição pode ser tratada como a resolução do sistema de equações não-lineares definidas pela Equação 2.2.5. Segundo Liberti et al. [57], há uma série de dificuldades numéricas ao resolver este sistema e, mesmo usando o algoritmo Branch-and-Bound espacial (sBB), realizar tal tarefa em um tempo computacional razoável ainda é um desafio [59, 57].

É possível definir um modelo para ele usando programação matemática, considerando-o como um problema de Otimização Global. Para isto, considera-se a função de penalidade f , dada em função de uma conformação $\mathbf{x} \in \mathbb{R}^{3n}$ para os átomos da molécula M em \mathbb{R}^3

$$f(\mathbf{x}) = \sum_{(i,j) \in S} \left(\|x_i - x_j\|^2 - d_{ij}^2 \right)^2 . \quad (2.2.6)$$

Dessa forma, o MDGP pode ser definido como um problema de otimização não-convexa global e sem restrições, a partir da minimização de f em relação às conformações. Logo, o espaço de busca por soluções para em \mathbb{R}^3 para o MDGP é contínuo, demandando grande esforço computacional.

A título de observação, no caso inexato, Liberti et al. [59] modela este problema de otimização, de maneira ponderada, como

$$\min_{\mathbf{x}} \sum_{(i,j) \in S} \left[\left(\max \left(\frac{l_{ij}^2 - \|x_i - x_j\|^2}{l_{ij}^2}, 0 \right) \right)^2 + \left(\max \left(\frac{\|x_i - x_j\|^2 - u_{ij}^2}{u_{ij}^2}, 0 \right) \right)^2 \right] .$$

Há a possibilidade de definir muitas funções de penalidade, já que, infelizmente, não há unicidade na formulação em Programação Matemática para este problema. Muitas das formulações,

como a anterior, não são diferenciáveis, inviabilizando métodos eficientes e robustos de Otimização Clássica, além de não possuirem a convexidade desejada [87].

Souza [87] traz uma lista de métodos para a resolução deste problema, formulado a partir de Programação Matemática. Neste trabalho, não vamos defini-los e demonstrar suas propriedades, mas tão somente citar alguns deles com breve descrição. Para mais detalhes, basta olhar as referências indicadas.

- (i) *EMBED* - G. Crippen e T. Havel definem este método dado em três fases, onde lançam mão da desigualdade triangular para completar as cotas inferior e superior das distâncias que faltam [17, 21].
- (ii) *ABBIE* - Hendrickson utiliza aqui a estratégia *divide-and-conquer* de modo que divide o problema exato em pequenos subproblemas [40].
- (iii) *DGSOL* - J. Moré e Z. Wu desenvolveram este método que utiliza técnicas de suavização Gaussiana para diminuir o número de minimizadores globais da função-objetivo do MDGP e, então, aplicar métodos de Otimização Global [66].
- (iv) *Programação Semidefinida* - P. Biswas *et al.* [7] utilizam o método de relaxação da programação semidefinida aliado à busca na direção do gradiente, a fim de resolver o MDGP, utilizando um determinado agrupamento de vértices e arestas em um grafo.
- (v) *Algoritmos Geometric Build-Up* - Q. Dong e Z. Wu transformam o sistema de equações quadráticas que define o método em um sistema de equações lineares, resolvendo agrupadamente. Isto gerou o Algoritmo *Geometric Build-Up* [23]. A fim de evitar o acúmulo de erros numéricos, Z. Wu e D. Wu propuseram o Algoritmo *Updated Geometric Build-Up* [94], que atualiza as coordenadas dos átomos a cada iteração, fazendo com que a determinação da conformação dependa, apenas, das distâncias e não de determinação de átomos anteriores.

Inicialmente, as moléculas eram identificadas por sua *fórmula atômica*, a qual indica a quantidade e a identidade dos átomos presentes nela [57]. Por exemplo, uma molécula de Gás Carbônico tem sua fórmula atômica dada como CO_2 . Entretanto, segundo Liberti *et al.* [57], os químicos se depararam com moléculas que possuíam a mesma fórmula atômica, mas tinham funções e propriedades químicas distintas. Isto os levou a investigar a forma com que estes átomos estavam ligados quimicamente. Depois disto, muitas formas de representar moléculas foram inventadas. A maneira padrão por meio da qual representamos moléculas, hoje-em-dia, como um conjunto de símbolos atômicos conectados por segmentos, foi originalmente descrita em Brown [9].

Este problema pode ser, facilmente, modelado usando Teoria de Grafos. Mais do que isso, Liberti *et al.* [57] também discorre a respeito da gênese da palavra *grafo* que, neste sentido, está intimamente relacionada com *variantes* e *co-variantes* de moléculas e consta, primeiramente, em um artigo de James Joseph Sylvester (1814 - 1897) na revista *Nature*, publicado em 1877 [90].

Lançaremos mão desta ferramenta em todo este texto a fim de descrever os problemas moleculares e seus métodos. O MDGP, dessa forma, pode ser modelado como o DGP para $K = 3$ (daí a motivação do nome) e apresentado como o seguinte problema de decisão.

Molecular Distance Geometry Problem (MDGP) (Teoria de Grafos - [56, 59, 54, 57]). Dada uma molécula M com n átomos, seja $G = (V, E, d)$ um grafo simples, não-orientado e ponderado, representando M , da seguinte maneira:

- (i) V é o conjunto de vértices que representam os átomos de M , indexados por $1 \leq i \leq n$;
- (ii) E é o conjunto de arestas de G , denotadas como um par de vértices $\{i, j\}$, para as quais está disponível o valor de distância $d_{i,j}$ e
- (iii) $d : E \rightarrow \mathbb{R}_+$ é um mapa que define os pesos nas arestas de G , valores de distâncias exatas entre os átomos.

Existe, então, uma realização $x : G \rightarrow \mathbb{R}^3$ de modo que esta seja uma imersão isométrica de G em \mathbb{R}^3 , ou seja, de maneira que valha a relação

$$\|x(i) - x(j)\| = d_{ij}, \quad (2.2.7)$$

para $\{i, j\} \in E$? Em caso afirmativo, encontre as conformações para G que são factíveis em relação às Equações (2.2.7).

2.3 Discretizable Molecular Distance Geometry Problem

Sob algumas hipóteses adicionais é possível considerar um sub-espaco de busca do MDGP que seja discreto. Esta possibilidade se concretizou nos trabalhos de L. Liberti, N. Maculan, C. Lavor e A. Mucherino. Alguns conceitos sobre grafos e ordens em vértices são necessários para o estudo desta seção, os quais estarão no Apêndice deste texto.

2.3.1 Discretização do MDGP

A discretização do MDGP tem sua motivação fundamentada no estudo de estruturas de proteínas. Neste caso, assim como no MDGP, V representa os átomos e o conjunto das arestas E representa distâncias entre átomos. Esta forma discreta de representação do problema se torna possível quando as seguintes hipóteses sobre V e E [56] (as quais são baseadas nas seguintes informações sobre as coordenadas internas da cadeia principal de átomos em proteínas):

- (1) O conjunto de átomos M consiste em uma sequência de n átomos de modo a existir uma ligação covalente entre cada par de átomos consecutivos;
- (2) Os comprimentos das ligações covalentes e dos ângulos entre essas ligações são conhecidos, ou seja, sabe-se os valores de $d_{i-1,i}$, para $i = 2, \dots, n$ e $\theta_{i-2,i}$, para $i = 3, \dots, n$;
- (3) As distâncias entre átomos separados por três ligações covalentes são conhecidas, isto é, tem-se os valores $d_{i-3,i}$, para cada $i = 4, \dots, n$;
- (4) Os átomos do conjunto $\{i-2, i-1, i\}$ são não-colineares, isto é, os ângulos de ligações $\theta_{i-2,i}$ são diferentes de $k\pi$, para $k \in \mathbb{Z}^+$;

Assumindo a validade desse conjunto de hipóteses, é possível mostrar que cada átomo possui um número finito de posições possíveis no espaço tridimensional, respeitando as restrições de distâncias em relação aos outros átomos da cadeia. Será possível considerar, dessa forma, que o espaço de busca por imersões isométricas viáveis dessas estruturas tornar-se-á discreto, como será mencionado mais adiante. A classe mais importante de moléculas que satisfaz estas hipóteses são as proteínas [56], cuja estrutura tridimensional é objeto de estudo de diversos pesquisadores e ilustrará os resultados desse trabalho.

Dessa forma, olhando pelo ponto de vista teórico-matemático, ao unir estas hipóteses a idéias sobre rigidez de grafos e ao problema SNLP (*Sensor Network Localization Problem*, explorado por Eren et al. [26]), Liberti e seus colaboradores definiram um subproblema do MDGP que englobasse apenas as instâncias que possuem a possibilidade de haver em si tal discretização (como é o caso das proteínas). A este problema chamou-se de **Discretizable Molecular Distance Geometry Problem** (DMDGP) (Liberti, 2008,[59]).

As instâncias do DMDGP consistem, portanto, em um subconjunto de instâncias do MDGP, o problema molecular mais geral. Assim, pressupõe-se que a busca por soluções deve ser realizada em tempo menor, já que o espaço de busca por soluções reduz-se a um conjunto discreto de imersões e, não mais, um conjunto contínuo não enumerável, como no caso geral. Esta discretização ainda será detalhada mais à frente.

Portanto, formalmente, estas hipóteses sobre tais instâncias do MDGP definem uma nova relação de *ordem* sobre os vértices do grafo que representa tal estrutura molecular para a qual deseja-se encontrar uma conformação tridimensional. Assim, é possível dizer que existe uma **Ordem DMDGP** nos vértices dessas instâncias. Além disso, é possível observar que tal ordem consiste em um caso particular da Ordem DDGP, descrita anteriormente [59].

Um exemplo do uso de ordens que otimizam este tipo de busca, através de uma discretização, é dada em (Eren et al., 2004,[26]), que define uma ordem nos vértices do grafo de modo que cada cinco átomos consecutivos formam uma clique, para resolver o SNLP. Desse modo, seu problema é resolvido polinomialmente [26]. Assim, a Ordem DMDGP é estabelecida assim:

- (i) para cada vértice $v_i \in V$ com $i = \rho(v_i) > 3$, isto é, v_i representa um vértice a partir do quarto em diante, o conjunto $N(v_i) \cap \gamma(v_i)$ possui, ao menos, três elementos.
- (ii) para cada vértice $v_i \in V$ com $i = \rho(v_i) > 3$: $N(v_i) \cap \gamma(v_i)$ contém um subconjunto U_i com *exatamente* três elementos, os **três vértices antecessores imediatos** de v_i , de modo que:
 - $G(U_i)$ é uma clique,
 - valem as desigualdades triangulares $d_{i,i+2} < d_{i,i+1} + d_{i+1,i+2}$, $\forall v_i, v_{i+1}, v_{i+2} \in V$ sobre o conjunto de distâncias disponíveis d entre os vértices em V , e
- (iii) os três primeiros vértices de G são unicamente determinados.

De posse desta ordem nos vértices do grafo (que representa uma molécula de proteína), Liberti et al. [59] formaliza o DMDGP como o problema de decisão que segue (aplicado a mergulhos de grafos no espaço Euclídeo tridimensional).

Discretizable Molecular Distance Geometry Problem (DMDGP). Dado um grafo ponderado e não-direcionado $G = (V, E, d)$, onde $d : E \rightarrow \mathbb{R}_+$, o subconjunto de vértices $U_0 = \{v_1, v_2, v_3\}$ e uma relação de ordem total em V que satisfaz a seguinte relação de axiomas:

- (1) $G[U_0]$ é um clique em três vértices (iniciando a configuração);
- (2) para todo vértice v_i com posto $i = \rho(v_i) > 3$, $G[U_i]$ é a clique com quatro vértices (ordem de discretização, dada anteriormente) e
- (3) para cada vértice v_i , com posto $i = \rho(v_i) > 3$, juntamente com $\{v_{i-3}, v_{i-2}, v_{i-1}\}$, vale a desigualdade

$$d_{i-3,i-1} < d_{i-3,i-2} + d_{i-2,i-1}, \quad (\text{Desigualdade Triangular Estrita})$$

encontre uma imersão $x : V \rightarrow \mathbb{R}^3$ tal que valha $\|x(v_i) - x(v_j)\| = d_{i,j}$, $\forall \{v_i, v_j\} \in E$.

Este problema resume-se como uma busca em \mathbb{R}^3 por imersões isométricas de estruturas de grafos que sigam uma determinada regra para seus vértices e arestas (ordem). Esta ordem se chama DMDGP, fornecendo o nome para a classe de instâncias que satisfazem tais propriedades.

A partir de aqui, os termos átomos de uma molécula e vértices de um grafo que representa esta molécula se referem à mesma coisa, abusando da nomenclatura. Isso deve-se à nossa principal aplicação em Bioinformática.

Esta relação de ordem $<$, definida como dados do DMDGP, pode ser usada a fim de construir uma árvore binária parcial e direcionada T de profundidade $n = |V|$ de modo que cada nó em um nível $i \leq n$ está associado a uma posição possível para o átomo i e, portanto cada caminho de comprimento n em T representa uma imersão de G em \mathbb{R}^3 [59]. Ou seja, o espaço de busca por soluções é, de fato, discreto para as instâncias do DMDGP.

Dados de distâncias

Considere um subconjunto de átomos S de uma molécula M associado a um grafo não-direcionado e ponderado $G = (V, E, d)$, sendo V o conjunto de vértices com uma ordem DMDGP, E o conjunto de arestas e d uma função real que pondera as arestas. Para G , o conjunto $\mathcal{D} = d(E)$ armazena as distâncias disponíveis entre os seus vértices. Mais do que isso: a função-peso $d : E \rightarrow \mathcal{D} \subset \mathbb{R}$ associa cada aresta $\{v_i, v_j\} \in E$ a um valor real de distância $d_{i,j} \in \mathbb{R}$.

Nesta seção, analisamos o conjunto de distâncias \mathcal{D} entre os átomos desta estrutura e as regularidades que são possíveis de se encontrar nas instâncias discretizáveis. Além disso, a possibilidade de traçar um mapa dos valores de distâncias entre átomos, que podem ser utilizados para encontrar as soluções deste problema, é discutida também.

Observa-se que, a partir do segundo item da definição do DMDGP, cada vértice cujo posto é estritamente maior do que três possui três vértices antecessores que são adjacentes a ele. Mais especificamente, para cada vértice $v_i \in V$, as arestas $\{v_{i-3}, v_i\}$, $\{v_{i-2}, v_i\}$ e $\{v_{i-1}, v_i\}$ estão disponíveis no conjunto de arestas E . Portanto, diz-se que as distâncias $d_{i-3,i}$, $d_{i-2,i}$ e $d_{i-1,i}$ são sempre dadas *a priori* para cada $v_i \in V$ com posto maior ou igual do que quatro. Com base no estudo de conformações de proteínas, é possível que através do conhecimento da estrutura química dessas

moléculas, tais como ângulos e comprimentos de ligações entre átomos, se extraia esses três valores de distâncias em todos os casos.

Por outro lado, é possível também conseguir, de modo esparso, os outros valores de distâncias $d_{i,j}$, com $|i - j| \geq 4$, já que em caso geral elas dependem de dados de RMN. Portanto, o conjunto das arestas associadas a esses valores é esparso.

Com isso, particiona-se de forma disjunta o conjunto de arestas E em dois conjuntos, que são agrupados através das características descritas acima, em $E = E_d \cup E_p$, onde

$$E_d = \{\{v_i, v_{i+1}\} : i = 1 \leq n - 1\} \cup \{\{v_i, v_{i+2}\} : i = \leq n - 2\} \cup \{\{v_i, v_{i+3}\} : i = 1 \leq n - 3\} \quad (2.3.1)$$

e

$$E_p = \{\{v_i, v_j\} : |j - i| \geq 4\}. \quad (2.3.2)$$

Estes subconjuntos são formalizados e têm a seguinte definição:

Definição 2.3.1. As arestas de E_d são chamadas de **Arestas de Discretização** (ou, do original, *discretization edges*). Já as arestas em E_p são denominadas **Arestas de Poda** (no original, chamadas de *pruning edges* [57]).

Com certa manipulação combinatória, é fácil mostrar que, para instâncias DMDGP com n vértices, temos as seguintes restrições de cardinalidade para os conjuntos E_d e E_p

$$|E_d| = 3(n - 2) \quad \text{e} \quad 0 \leq |E_p| \leq \left(\frac{n^2 - 7n + 12}{2} \right). \quad (2.3.3)$$

Motivados por estas definições e a fim de facilitar futuras referências e particularizações neste trabalho, particionamos também o conjunto \mathcal{D} de uma forma associada às Equações (2.3.1) e (2.3.2) como

$$\mathcal{D} = \mathcal{D}_d \cup \mathcal{D}_p, \quad (2.3.4)$$

onde $\mathcal{D}_d = d(E_d)$ e $\mathcal{D}_p = d(E_p)$. A nomenclatura desses conjuntos é análoga e segue.

Definição 2.3.2. Neste trabalho, as distâncias no conjunto \mathcal{D}_d são chamadas de *Distâncias de Discretização* e as distâncias no conjunto \mathcal{D}_p de *Distâncias de Poda*.

Esta configuração particionada dos conjuntos de arestas e de distâncias possibilita a definição de uma formulação matemática para essa discretização das possíveis soluções do problema, como se vê a seguir.

Formulação matemática do DMDGP

A seguinte formulação matemática para o DMDGP é apresentada nos trabalhos de Liberti et al. [56] e Lavor et al. [54]. Seja S um subconjunto composto por uma sequência de n átomos de uma molécula M , cujas coordenadas de seus pontos no espaço Euclídeo \mathbb{R}^3 são representadas por x_1, \dots, x_n . Considere, também, os seguintes entes geométricos, definidos em cada subconjunto com quatro vértices consecutivos, seguindo a ordem DMDGP.

- (i) Suponha que haja uma ligação covalente r_i entre cada par $(i-1, i)$, $i = 2, \dots, n-1$;
- (ii) O **comprimento de ligação** para r_i é a distância Euclideana $d_{i-1,i}$ entre os átomos $i-1$ e i , para $i = 2, \dots, n$;
- (iii) O **ângulo de ligação** $\theta_{i-2,i} \in [0, \pi]$ é o ângulo formado pelos segmentos definidos, respectivamente, pelos átomos $i-2, i-1$ e $i-1, i$, para $i = 3, \dots, n$;
- (iv) O **ângulo de torção** $\omega_{i-3,i} \in [0, 2\pi]$ é o ângulo formado entre os dois planos que são, respectivamente, definidos pelos átomos $i-3, i-2, i-1$ e $i-2, i-1, i$, para $i = 4, \dots, n$;

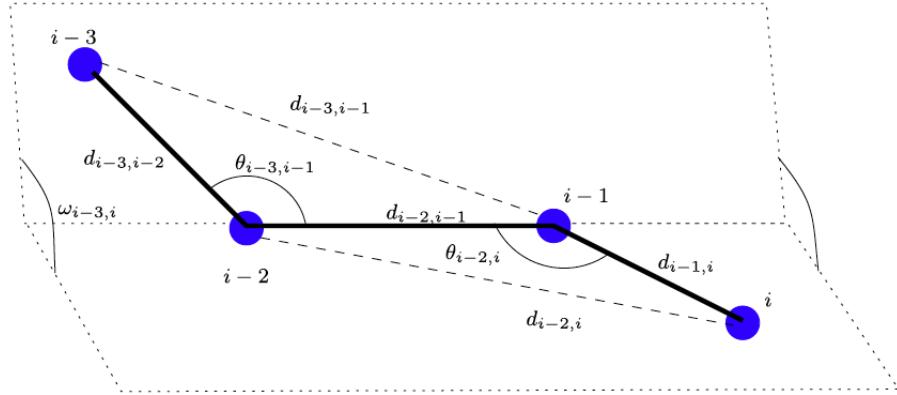


Figura 2.8: Representação dos comprimentos de ligação, ângulos de ligação e ângulos de torção, extraído de Lavor et al. [54].

A Figura 2.8 representa, tridimensionalmente, os itens geométricos pertencentes a uma instância do DMDGP e que foram descritos acima. Todos os comprimentos de ligações covalentes e ângulos de ligações são supostamente conhecidos a priori [79, 56, 54] e, a partir destes dados intrínsecos do problema, se obtém a seguinte proposição que explicita uma fórmula para o cálculo dos cossenos dos ângulos de torção, para cada quarteto de vértices em uma instância do DMDGP. Este resultado é baseado na *Lei dos Cossenos Diedrais* [80].

Proposição 2.3.1. Para um conjunto de quatro átomos consecutivos $\{i, i+1, i+2, i+3\}$, o cosseno do ângulo de torção $\omega_{i,i+3}$ pode ser calculado através de

$$\cos(\omega_{i,i+3}) = \frac{d_{i,i+1}^2 + d_{i+1,i+3}^2 - 2d_{i,i+1}d_{i+1,i+3}\cos(\theta_{i,i+2})\cos(\theta_{i+1,i+3}) - d_{i,i+3}^2}{2d_{i,i+1}d_{i+1,i+3}\sin(\theta_{i,i+2})\sin(\theta_{i+1,i+3})} \quad (2.3.5)$$

em função de distâncias conhecidas entre pontos e ângulos planos de ligação conhecidos.

A Fórmula (2.3.5) está bem definida, já que seu denominador é diferente de zero. De fato, isto se verifica pois os ângulos planos de ligação $\theta_{i,i+2}$ não podem ser iguais a ângulos múltiplos de π . Se assim o fossem, existiria a colinearidade de três vértices consecutivos na cadeia de vértices, o que violaria a hipótese de satisfação da desigualdade triangular estrita. Instâncias que possuem esta colinearidade são chamadas de *não-discretizáveis* e ilustradas na Figura 2.9.

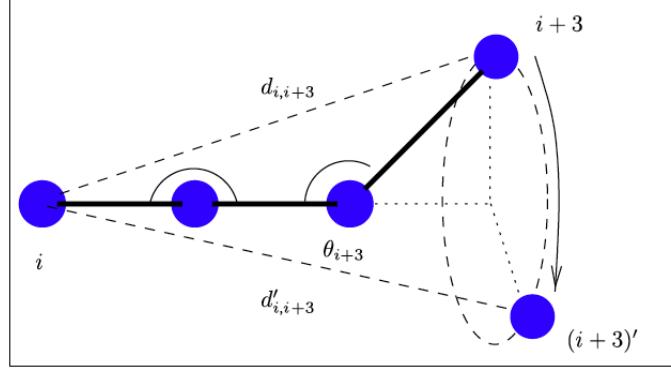


Figura 2.9: Caso em que instância é não-discretizável, extraída de [56].

Como se pode notar, neste caso, há um círculo de possibilidades de posicionar o quarto átomo na sequência, permanecendo com uma quantidade não-enumerável de soluções.

Lavor et al. [54] mostra que por esta fórmula os ângulos de torção são calculados em $\mathcal{O}(1)$.

Assim, de posse dos comprimentos de ligações $d_{1,2}, \dots, d_{n-1,n}$, dos ângulos de ligações $\theta_{1,3}, \dots, \theta_{n-2,n}$ e dos cossenos dos ângulos de torção $\omega_{1,4}, \dots, \omega_{n-3,n}$ de M , as coordenadas Cartesianas $x_i = (x_{i1}, x_{i2}, x_{i3})$ do átomo i , pertencente a M , são determinadas pelo produto

$$\begin{bmatrix} x_{i1} \\ x_{i2} \\ x_{i3} \\ 1 \end{bmatrix} = B_1 B_2 \dots B_i \begin{bmatrix} 0 \\ 0 \\ 0 \\ 1 \end{bmatrix}, \quad i \in \{4, \dots, n\} \quad (2.3.6)$$

onde as chamadas **Matrizes de torção** B_i são dadas por

$$B_i = \begin{bmatrix} -\cos(\theta_{i-2,i}) & -\sin(\theta_{i-2,i}) & 0 & -d_{i-1,i} \cos(\theta_{i-2,i}) \\ \sin(\theta_{i-2,i}) \cos(\omega_{i-3,i}) & -\cos(\theta_{i-2,i}) \cos(\omega_{i-3,i}) & -\sin(\omega_{i-3,i}) & d_{i-1,i} \sin(\theta_{i-2,i}) \cos(\omega_{i-3,i}) \\ \sin(\theta_{i-2,i}) \sin(\omega_{i-3,i}) & -\cos(\theta_{i-2,i}) \sin(\omega_{i-3,i}) & \cos(\omega_{i-3,i}) & d_{i-1,i} \sin(\theta_{i-2,i}) \sin(\omega_{i-3,i}) \\ 0 & 0 & 0 & 1 \end{bmatrix}. \quad (2.3.7)$$

Em particular, as matrizes de torção B_1, B_2 e B_3 , que inicializam o processo iterativo de discretização da estrutura molecular, são definidas por

$$B_1 = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}, \quad B_2 = \begin{bmatrix} -1 & 0 & 0 & -r_2 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & -1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \quad (2.3.8)$$

e

$$B_3 = \begin{bmatrix} -\cos(\theta_{1,3}) & -\sin(\theta_{1,3}) & 0 & -d_{2,3} \cos(\theta_{1,3}) \\ \sin(\theta_{1,3}) & -\cos(\theta_{1,3}) & 0 & d_{2,3} \sin(\theta_{1,3}) \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}. \quad (2.3.9)$$

A matriz dada pelo produto $C_i = B_1 B_2 B_3 \dots B_i$ é chamada de **Matriz de Torção Acumulada** e tem um papel de grande importância na operacionalização do método que resolve o DMDGP, a ser definido a posteriori.

Dois resultados apresentados em Lavor et al. [54] são compilados neste texto como o teorema a seguir, com o objetivo de justificar a quantidade finita de soluções do DMDGP.

Teorema 2.3.1. Dada uma instância DMDGP $G = (V, E, d)$, o número de conformações $x : V \rightarrow \mathbb{R}^3$ tais que $\|x(i) - x(j)\| = d_{i,j}$, para cada aresta $\{v_i, v_j\} \in E$, é finito e, no máximo, igual a 2^{n-3} , a menos de movimentos rígidos (translações, rotações e reflexões).

Demonstração. Para esta demonstração, veja o Teorema 1 e Corolário 1 em Lavor et al. [54]. \square

A prova matemática do Teorema 2.3.1 consiste em um passo-a-passo para a formulação da discretização e é realizada de maneira induktiva. A seguir, o processo de geração da árvore binária T de realizações para G (de modo que cada caminho em profundidade com n nós desta árvore representa uma conformação factível para G) é descrito.

- (1) A partir dos comprimentos $d_{1,2}, d_{2,3} \in \mathcal{D}_d$ e do ângulo $\theta_{1,3}$, constrói-se as matrizes de torção B_1, B_2 e B_3 , com as quais se determina as posições únicas para os três primeiros átomos de M , fazendo

$$x_1 = \begin{bmatrix} 0 \\ 0 \\ 0 \end{bmatrix}, \quad x_2 = \begin{bmatrix} -d_{1,2} \\ 0 \\ 0 \end{bmatrix} \quad \text{e} \quad x_3 = \begin{bmatrix} d_{2,3} \cos(\theta_{1,3}) - d_{1,2} \\ d_{2,3} \sin(\theta_{1,3}) \\ 0 \end{bmatrix}. \quad (2.3.10)$$

- (2) Com o valor da distância $d_{1,4} \in \mathcal{D}_d$, é possível determinar o cosseno do ângulo de torção $\omega_{1,4}$ através da fórmula (2.3.5). Com este valor de cosseno, há duas possibilidades para o cálculo do seno do mesmo ângulo, dadas abaixo pela relação trigonométrica fundamental

$$\operatorname{sen}(\omega_{1,4}) = \pm \sqrt{1 - \cos^2(\omega_{1,4})}. \quad (2.3.11)$$

Para cada um dos valores da equação (2.3.11), obtém-se uma matriz de torção diferente, a saber B_4 , referente ao seno positivo, e B'_4 , referente ao seno negativo. Cada uma dessas é multiplicada pela matriz de torção acumulada C_3 , gerando duas posições x_4 e x'_4 para o quarto átomo da cadeia. Tais posições são, então, dadas por

$$(x_4, 1)^T = C_3 B_4 (0, 0, 0, 1)^T \quad \text{e} \quad (x'_4, 1)^T = C_3 B'_4 (0, 0, 0, 1)^T. \quad (2.3.12)$$

A Figura 2.10 representa essa escolha de duas possibilidades. É a partir do quarto nível da cadeia (ou seja, a partir do vértice de posto igual a 4) que se inicia a estrutura binária da determinação do próximo vértice usando esta formulação matemática.

- (3) Em procedimento análogo, encontra-se quatro posições possíveis para o quinto vértice da instância. Estas posições se referem às combinações dos valores dos senos dos dois ângulos de torção que determinam este vértice

$$\operatorname{sen}(\omega_{1,4}) = \pm \sqrt{1 - \cos^2(\omega_{1,4})} \quad \text{e} \quad \operatorname{sen}(\omega_{2,5}) = \pm \sqrt{1 - \cos^2(\omega_{2,5})}, \quad (2.3.13)$$

calculadas a partir do valor do cosseno do ângulo de torção $\omega_{2,5}$, a partir da Fórmula (2.3.5).

- (4) Dessa maneira, a cada nível $i \geq 4$, existe 2^{i-3} possibilidades para fixar a posição do átomo i no espaço Euclídeo tridimensional, já que os três primeiros átomos da estrutura são fixos pela regra disposta anteriormente. Portanto, ao percorrer toda a estrutura da molécula M , com n átomos, tem-se 2^{n-3} possíveis estruturas para esta molécula em \mathbb{R}^3 .

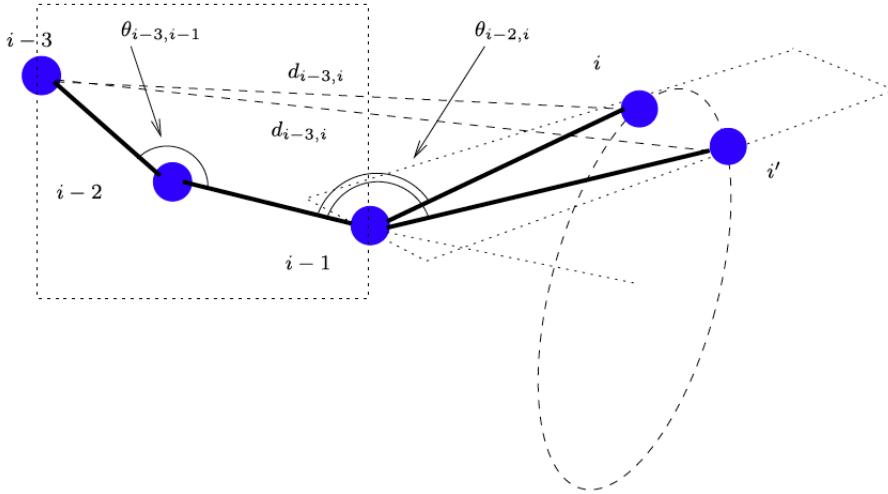


Figura 2.10: A discretização do problema passa pela binaridade fornecida pelo ângulo de torção: cada átomo i pode estar apenas em uma das duas posições i ou i' a fim de satisfazer a distância $d_{i-3,i}$, ao invés de um círculo contínuo de possibilidades como era anteriormente. Figura extraída de [54].

2.4 Branch-and-Prune

Seja $G = (V, E, d)$ uma instância DMDGP com n vértices. Abaixo, temos um panorama do funcionamento do principal método para resolver este problema com base na própria idéia da discretização, usando interseção de três e quatro esferas.

Para cada vértice $v_i \in V$ com posto $\rho(v) = i > 3$, as arestas $\{v_{i-3}, v_i\}, \{v_{i-2}, v_i\}, \{v_{i-1}, v_i\} \in E$ estão sempre disponíveis, devido à definição do DMDGP. Dessa forma, sabe-se *a priori* os valores das distâncias $d_{i-3,i}, d_{i-2,i}, d_{i-1,i}$ e posições factíveis (respeitam os valores de distâncias) $x_{i-3}, x_{i-2}, x_{i-1} \in \gamma(v)$ para os três vértices.

Com isso, define-se três esferas a serem intersectadas:

- a esfera S_1^i centrada no ponto x_{i-3} e com raio $d_{i-3,i}$,
- a esfera S_2^i centrada no ponto x_{i-2} e com raio $d_{i-2,i}$ e
- a esfera S_3^i centrada no ponto x_{i-1} e com raio $d_{i-1,i}$.

Pela definição do DMDGP, as posições para o vértice v_i devem estar concomitantemente nas três esferas. Portanto, o conjunto de posições possíveis para v_i restringe-se bijetivamente ao conjunto $S^i = S_1^i \cap S_2^i \cap S_3^i$. Além disso, a hipótese de que tais instâncias satisfazem a desigualdade triangular estrita implica que $|S^i| \leq 2$ [59].

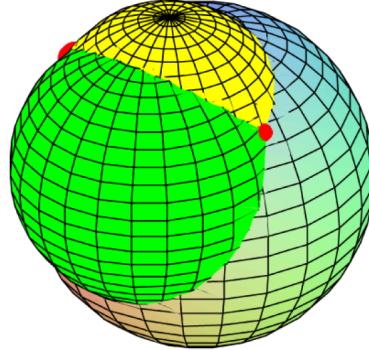


Figura 2.11: Interseção de três superfícies esféricas em \mathbb{R}^3 , contendo dois pontos [59].

Esta é a idéia central do principal método de solução usado na solução do DMDGP [59]. Discorreremos suscintamente sobre os principais passos, estrutura algorítmica e alguns exemplos sobre este método a seguir.

2.4.1 O Algoritmo

Esta seção tem por objetivo analisar o funcionamento do algoritmo *Branch & Prune* (BP), desenvolvido e apresentado por Liberti *et al.* a fim de resolver o DMDGP de modo eficiente, cuja estrutura se confunde com a formulação matemática do problema em si, já que ambas estão intimamente ligadas [56]. A princípio, é necessário observar que o BP não é um método exato, pois encontra apenas soluções aproximadas, definidas por uma tolerância $\varepsilon > 0$ dada como *input* [54]. Tais soluções são fornecidas como dados de saída do BP em um grafo que possui uma estrutura de árvore binária cuja determinação é feita através de buscas em profundidade (*depth-first search*) nesta árvore a fim de encontrar caminhos com n vértices, da raiz até as folhas, de forma que suas posições respeitem as restrições de distâncias existentes no conjunto \mathcal{D} .

Neste trabalho, analisamos este método de forma breve e suscinta, sem apresentar resultados computacionais, já que o interesse é em aplicá-lo a um outro problema específico.

A estrutura algorítmica geral do BP utiliza a interseção de três esferas mencionada anteriormente de forma recursiva, a qual pode ser compreendida através dos três passos a seguir, cujo produto deve ser uma árvore binária T .

Inicialização

O primeiro passo é a inicialização da estrutura. Nela, os primeiros três vértices $v_1, v_2, v_3 \in V$ podem ser fixados unicamente nas posições $x_1, x_2, x_3 \in \mathbb{R}^3$ por meio das Equações (2.3.10), como descrito em [54]. Isto define a orientação do espaço por meio do qual as coordenadas de todas as soluções serão apresentadas.

As posições desses vértices estão binunivocamente associadas aos três primeiros nós da árvore T , como ilustrado pela Figura 2.12 abaixo.



Figura 2.12: Inicialização do algoritmo: três primeiros nós da árvore.

Branching

Este passo, por sua vez, consiste na “ramificação” da árvore binária de possíveis soluções. Assim, suponha que, seguindo a ordem DMDGP de V , tenha-se descoberto as posições para os vértices v_1, \dots, v_{i-1} , com $\rho(v_i) = i \geq 4$. O próximo vértice a ser determinado na sequência é $v_i \in V$.

Como as arestas de discretização $E_d^i = \{\{v_{i-3}, v_i\}, \{v_{i-2}, v_i\}, \{v_{i-1}, v_i\}\} \subset E_d$ estão disponíveis, a intersecção das três esferas, representada pela Figura 2.11, sempre existe e é não-vazia. Algebricamente, esta intersecção reduz-se ao sistema não-linear de equações

$$\begin{cases} \|x_{i-3} - x_i\| = d_{i-3,i} \\ \|x_{i-2} - x_i\| = d_{i-2,i} \\ \|x_{i-1} - x_i\| = d_{i-1,i}, \end{cases} \quad (2.4.1)$$

pelo qual é possível encontrar, no máximo, duas posições possíveis para x_i [54, 68].

Portanto, o BP utiliza a clique de antecessores $V_d^i = \{v_{i-3}, v_{i-2}, v_{i-1}\} \subset V$ para encontrar duas possíveis posições, x_i^1 e x_i^2 , para que o vértice v_i pertença à estrutura DMDGP para a qual se deseja encontrar as conformações. Tal procedimento é chamado de **Branching** (“ramificação”), em analogia ao fato de que essas duas posições estão associadas a um par de nós no nível i da árvore T de soluções (dois novos ramos). Esta ramificação elenca todos os possíveis caminhos até as soluções do DMDGP. Assim, garante-se que, no máximo, existem 2^{n-3} conformações possíveis para G , o que corrobora com a enumerabilidade e finitude do número de soluções.

Considerando, por exemplo, uma instância DMDGP com seis vértices, a árvore com todas as $2^{6-3} = 2^3 = 8$ possíveis soluções é representada pela Figura 2.13.

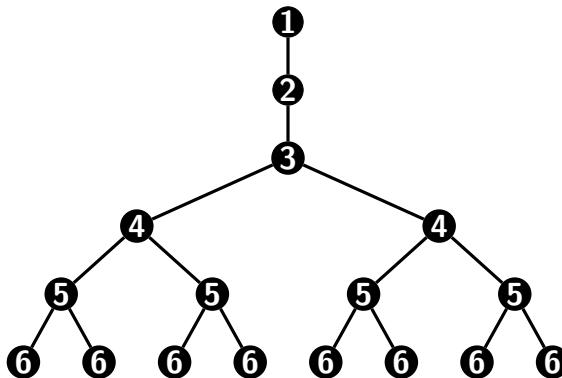


Figura 2.13: Branching: a ramificação em posições possíveis.

Por outro lado, este passo não fornece informações sobre a factibilidade dessas posições em relação às restrições de valores de distâncias dados em \mathcal{D}_d .

Pruning

Por fim, este passo é capaz de escolher os caminhos corretos a se percorrer na árvore T das possíveis soluções, encontrada durante o *Branching*, desprezando os caminhos infactíveis. Dessa maneira, apesar do DMDGP ser um problema **NP** - difícil, o BP encontra rapidamente essas soluções. Este fato é que torna o BP um mecanismo eficiente de resolução.

Como todos os dados do conjunto mais regular E_d são utilizados no passo anterior e o conjunto das arestas de poda é esparsa, então as informações de E_p são usadas adicionalmente a fim de testar se cada uma das posições encontradas é factível ou não para pertencer a uma imersão isométrica de G em \mathbb{R}^3 . Há duas possibilidades a se considerar então.

A primeira é o caso em que E_p é vazio. Neste, tem-se que $E = E_d$ e, portanto, as posições encontradas não apenas são possíveis como também são factíveis. Dessa maneira, todos os 2^{n-3} caminhos encontrados durante o *Branching* representam soluções factíveis para o DMDGP. Este é o pior caso a ser considerado, onde se precisa explorar todos os caminhos existentes em T , demorando mais tempo na execução do algoritmo.

Por outro lado, se $E_p \neq \emptyset$, é possível que uma ou ambas as posições encontradas se inviabilizem, caso não respeitem as restrições de distâncias de poda. Este passo é chamado de **Pruning** (Poda). Para dar sustentação teórica a essa afirmação, é preciso discorrer sobre as condições da unicidade na determinação dos vértices. Assim como Lavor *et al.* [51], considere um vértice indeterminado $v_i \in V$, com $i \geq 4$, e um subconjunto $V_i \subset V$ formado pelos vértices antecessores e adjacentes v_i

$$V_i = \{v_j \in V : \{v_j, v_i\} \in E \text{ e } j < i\}.$$

Este subconjunto induz a formulação de um subproblema do DMDGP, que pode ser definido através do sistema não-linear

$$\|x_j - x_i\| = d_{j,i}, \quad \forall v_j \in V_i. \quad (2.4.2)$$

Proposição 2.4.1. O conjunto V_i possui, ao menos, três vértices.

Demonstração. De fato: para todo vértice $v_i \in V$, cujo posto é maior ou igual a 4, as três arestas em E_d^i estão sempre disponíveis. Portanto, o número de elementos neste conjunto é $|V_i| \geq 3$. \square

Se V_i possui apenas três vértices, então existem somente três valores de distâncias disponíveis entre v_i a seus antecessores. Logo, o sistema formado pelas Equações 2.4.2 redonda na intersecção das três esferas, comentada anteriormente, podendo possuir duas soluções no máximo.

Suponha, outrossim, que $|V_i| \geq 4$. Unindo isto à Proposição 2.4.1, tome um subconjunto ordenado $U \subseteq V_i$ de modo que

$$U = \{v_j, v_{i-3}, v_{i-2}, v_{i-1}\}.$$

Antes de prosseguir, é necessário apresentar a definição de uma ferramenta clássica e de grande utilidade na teoria de Geometria de Distâncias.

Definição 2.4.1. Dado um conjunto $\mathcal{U} = \{p_0, \dots, p_K\}$ de $K + 1$ pontos, o **Determinante de Cayley - Menger** definido por \mathcal{U} é dado por

$$CM(\mathcal{U}) = \begin{vmatrix} 0 & 1 & 1 & \dots & 1 \\ 1 & 0 & d_{01}^2 & \dots & d_{0K}^2 \\ 1 & d_{01}^2 & 0 & \dots & d_{1K}^2 \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ 1 & d_{0K}^2 & d_{1K}^2 & \dots & 0 \end{vmatrix}. \quad (2.4.3)$$

Essa ferramenta é fundamental para se definir um quantificador do volume de invólucros convexos de pontos no espaço [57], objeto principal para o próximo resultado.

Definição 2.4.2. Dado um conjunto $\mathcal{U} = \{p_0, \dots, p_K\}$ de $K + 1$ pontos em \mathbb{R}^K , o **Volume do K - Simplex** envolvendo esses pontos, isto é, o volume do invólucro convexo formado por esses pontos, é definido por meio do determinante de Cayley - Menger como

$$\Delta_K(\mathcal{U}) = \sqrt{\frac{(-1)^{K+1}}{2^K(K!)^2} CM(\mathcal{U})}$$

Lema 2.4.1. A coplanaridade dos pontos de U ocorre com probabilidade zero.

Demonastração. Supondo que tais pontos sejam coplanares, então estes formam um *simplex* em \mathbb{R}^3 de volume nulo, isto é,

$$\Delta_3(U) = 0, \quad (2.4.4)$$

onde Δ_3 define o volume de um invólucro convexo em \mathbb{R}^3 . Segue que, assim como em Liberti *et al.* [57], o conjunto

$$M = \{x \in \mathbb{R}^9 : \Delta_3(U) = 0\}$$

é uma variedade de dimensão estritamente menor em \mathbb{R}^9 . Logo, M tem medida de Lebesgue nula. Como usualmente lança-se mão da medida de Lebesgue a fim de se definir um espaço de probabilidade para uma distribuição de probabilidade uniforme, então a probabilidade deste conjunto conter pontos coplanares é igual a zero, como queríamos demonstrar. \square

O objetivo, neste caso, é de encontrar as posições $x_i \in \mathbb{R}^3$ que satisfaçam as equações de discretização

$$\begin{cases} \|x_{i-3} - x_i\| = d_{i-3,i} \\ \|x_{i-2} - x_i\| = d_{i-2,i} \\ \|x_{i-1} - x_i\| = d_{i-1,i} \end{cases}$$

e a equação referente à distância adicional de v_i ao vértice v_j

$$\|x_j - x_i\| = d_{j,i}. \quad (2.4.5)$$

Elevando tais equações quadráticas ao quadrado, obtemos

$$\begin{cases} \|x_{i-3}\|^2 - 2x_{i-3} \cdot x_i + \|x_i\|^2 = d_{i-3,i}^2 \\ \|x_{i-2}\|^2 - 2x_{i-2} \cdot x_i + \|x_i\|^2 = d_{i-2,i}^2 \\ \|x_{i-1}\|^2 - 2x_{i-1} \cdot x_i + \|x_i\|^2 = d_{i-1,i}^2 \\ \|x_j\|^2 - 2x_j \cdot x_i + \|x_i\|^2 = d_{j,i}^2 \end{cases}.$$

Logo, subtraindo a última equação das outras três, resulta em um sistema linear significativo

$$\begin{cases} 2(x_{i-3} - x_j) \cdot x_i = (\|x_{i-3}\|^2 - d_{i-3,i}^2) - (\|x_j\|^2 - d_{j,i}^2) \\ 2(x_{i-2} - x_j) \cdot x_i = (\|x_{i-2}\|^2 - d_{i-2,i}^2) - (\|x_j\|^2 - d_{j,i}^2) \\ 2(x_{i-1} - x_j) \cdot x_i = (\|x_{i-1}\|^2 - d_{i-1,i}^2) - (\|x_j\|^2 - d_{j,i}^2) \end{cases}.$$

Matricialmente, as possíveis realizações do vértice v_i têm de ser soluções para o sistema

$$2Ax_i = b, \quad (2.4.6)$$

onde

$$A = \begin{bmatrix} (x_{i-3} - x_j)^T \\ (x_{i-2} - x_j)^T \\ (x_{i-1} - x_j)^T \end{bmatrix} \quad \text{e} \quad b = \begin{bmatrix} (\|x_{i-3}\|^2 - d_{i-3,i}^2) - (\|x_j\|^2 - d_{j,i}^2) \\ (\|x_{i-2}\|^2 - d_{i-2,i}^2) - (\|x_j\|^2 - d_{j,i}^2) \\ (\|x_{i-1}\|^2 - d_{i-1,i}^2) - (\|x_j\|^2 - d_{j,i}^2) \end{bmatrix}.$$

Desta formulação matricial, segue o seguinte teorema.

Teorema 2.4.1. O sistema linear $2Ax_i = b$ acima possui única solução, com probabilidade 1.

Demonstração. De fato: temos que $\Delta_3(U_i) > 0$, com probabilidade 1. Segue que os pontos $x_{i-3}, x_{i-2}, x_{i-1}$ e x_j são não - coplanares com probabilidade 1 e, assim, os vetores $x_{i-3} - x_j, x_{i-2} - x_j$ e $x_{i-1} - x_j$ são linearmente independentes. Consequentemente, a matriz quadrada A de dimensão 3 tem posto - completo e, portanto, é inversível.

Dessa discussão, segue que tal sistema tem solução única x_i^* a qual é dada por

$$x_i^* = \frac{1}{2} A^{-1} b.$$

□

Em resumo, é possível aplicar este resultado à existência de um único vértice em uma instância DMDGP.

Corolário 2.4.1. Seja $G = (V, E, d)$ uma instância DMDGP. Se $v_i \in V$ é tal que existem arestas $\{v_j, v_i\}$, com $i \geq 4$ e $j = \rho(v_j) < i - 3$, então existe, no máximo, uma posição factível para este vértice, com probabilidade 1.

Demonstração. Com efeito: caso a solução do sistema linear anterior satisfaça a Equação (2.4.5), então este ponto é a única posição possível para tal vértice. Se não, então não existe posição possível que satisfaça tais restrições no espaço tridimensional. □

Como existem sempre duas posições possíveis, neste caso é garantido a existência de uma poda em uma delas pelo menos, a qual será descrita mais adiante como um aparato numérico.

É possível, ainda, fornecer uma interpretação geométrica para esta unicidade na determinação de um vértice para os qual se possui, ao menos, um valor de distância de poda. Os três valores de distâncias de discretização fornecem uma representação da intersecção de três esferas em \mathbb{R}^3 , como já foi comentado. Adicionando mais a distância $d_{j,i}$, tem-se a intersecção de quatro esferas que possui um único ponto, com probabilidade 1. Este ponto é considerado como posição do vértice v_i , se satisfizer a equação (2.4.5) e pode ser determinado em tempo constante sempre. Caso contrário, se não satisfizer tal equação, este ponto não pertence a nenhum caminho que resolve o DMDGP.

O processo de poda requer um critério numérico adequado a fim de se viabilizar a escolha de uma posição possível ou o seu descarte. A isso, chamaremos de Teste de Poda. Dentre os testes descritos na literatura, adota-se o DDF (*Direct Distance Feasibility*) [54].

Definição 2.4.3. Dada uma aresta de poda $\{j, i\} \in E_p$, com $j < i$, e uma tolerância $\varepsilon > 0$, o **Teste de Factibilidade DDF** consiste em checar se a desigualdade abaixo é válida

$$(\|x_i - x_j\|^2 - d_{ij}^2)^2 < \varepsilon.$$

Esta tolerância é definida de acordo com a natureza do problema ou por conveniência numérica.

Inicialmente, se justificou a existência das podas e adotou-se um procedimento para executá-las. Então, é preciso descrever o modo como este aparato será aplicado na escolha de uma posição factível para v_i . Antes, vamos considerar uma partição disjunta do conjunto V_i na união do conjunto V_d^i , que possui os vértices adjacentes a v_i associados com a discretização, com o conjunto V_p^i , que possui vértices adjacentes a v_i associados com a poda

$$V_i = V_d^i \cup V_p^i.$$

Também, para facilitar a descrição dos próximos passos, particiona-se o conjunto E_p como

$$E_p = E_p^5 \cup E_p^5 \cup \dots \cup E_p^i \cup \dots \cup E_p^n,$$

onde cada $E_p^i = \{\{v_j, v_i\} : v_j \in V_p^i\}$ e $E_p^i \cap E_p^j = \emptyset$ para $i \neq j$. Além disso, o conjunto V_p^i está biunivocamente associado ao conjunto E_p^i .

A partir disso, para que ocorram podas durante o posicionamento de v_i , é preciso que

$$V_p^i = \{v_{i_1}, v_{i_2}, \dots, v_{i_k}\} \neq \emptyset, \quad k = |V_p^i| \leq i - 4.$$

Portanto, após encontrar as duas posições x_i^1 e x_i^2 para o vértice v_i , dadas pela discretização, há que se checar a factibilidade de cada uma delas através do Teste DDF para cada aresta em E_p^i .

Três resultados são possíveis para o teste DDF [56].

(1) Somente x_i^1 é factível

Neste primeiro caso, a posição x_i^1 é factível em relação ao teste DDF, pois todas as arestas $\{v_{i_j}, v_i\} \in E_p^i$ satisfazem a desigualdade

$$(\|x_i^1 - x_{i_j}\|^2 - d_{i,i_j}^2)^2 < \varepsilon. \quad (2.4.7)$$

Por outro lado, existe uma aresta $\{v_{i_k}, v_i\} \in E_p^i$ que não satisfaz a desigualdade DDF para x_i^2 , pois

$$\left(\|x_i^2 - x_{i_k}\|^2 - d_{i,i_k}^2 \right)^2 \geq \varepsilon, \quad (2.4.8)$$

Assim, a primeira posição dá origem a um caminho factível, mas a segunda é podada e não gera nenhum caminho.

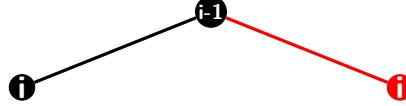


Figura 2.14: O primeiro caminho é factível e o segundo é podado.

(2) Somente x_i^2 é viável

Neste segundo caso, existe uma aresta $\{v_{i_k}, v_i\} \in E_p^i$ que não satisfaz a desigualdade DDF para x_i^1 , pois

$$\left(\|x_i^1 - x_{i_k}\|^2 - d_{i,i_k}^2 \right)^2 \geq \varepsilon. \quad (2.4.9)$$

Logo, os caminhos a partir dessa posição são inviabilizados. Portanto, esse nó é podado e a busca retrocede ao nó-pai $i - 1$. Ocorre, então, um **backtracking**, ou seja, um “retorno” ao nó-pai, em referência a [52].

Após esse retorno, a busca avança para o teste da segunda posição x_i^2 . Esta, por sua vez, é factível, pois todas as arestas $\{v_{i_j}, v_i\} \in E_p^i$ satisfazem a desigualdade

$$\left(\|x_i^2 - x_{i_j}\|^2 - d_{i,i_j}^2 \right)^2 < \varepsilon. \quad (2.4.10)$$

Todos os caminhos factíveis devem, então, passar por essa posição.

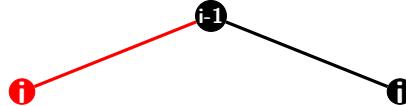


Figura 2.15: O primeiro caminho é podado e o segundo é factível.

(3) Nem x_i^1 nem x_i^2 são viáveis

Por fim, no último caso, existe uma aresta $\{v_{i_k}, v_i\} \in E_p^i$ que não satisfaz a desigualdade DDF para x_i^1 , pois

$$\left(\|x_i^1 - x_{i_k}\|^2 - d_{i,i_k}^2 \right)^2 \geq \varepsilon. \quad (2.4.11)$$

Neste caso, o caminho é podado e o algoritmo retorna a busca para o nó-pai, avançando para a segunda possibilidade. Mas, existe também uma outra aresta $\{v_{i_t}, v_i\} \in E_p^i$ que não satisfaz a desigualdade DDF para esta posição, pois

$$\left(\|x_i^2 - x_{i_t}\|^2 - d_{i,i_t}^2 \right)^2 \geq \varepsilon. \quad (2.4.12)$$

Portanto, nenhuma das duas posições são factíveis. A busca retrocede para o nó-pai anterior que ainda não teve as duas posições avaliadas pelo teste DDF.

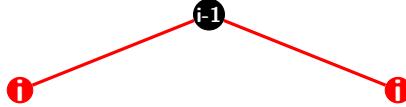


Figura 2.16: Os dois caminhos são infactíveis e, portanto, são podados.

Em resumo, olhando para o modo como o BP explora o espaço de busca por soluções, pode-se afirmar que as arestas de discretização de E_d atuam para moldar este espaço em forma de uma árvore binária e que as arestas de poda de E_p indicam o caminho que deve ser percorrido para encontrar as soluções factíveis [52].

2.4.2 Estrutura Algorítmica

Tendo definido os três passos do BP (inicialização, *branching* e *pruning*), vamos analisar sua estrutura algorítmica e apresentar um exemplo para compreender melhor seu funcionamento.

As informações de entrada são dadas pela instância $G = (V, S, d)$ do DMDGP. No final do processo, o método deve fornecer como saída um grafo em forma de uma árvore binária T para o qual cada nó está associado a uma posição factível em \mathbb{R}^3 para o vértice associado ao seu nível. Como já vimos, esta árvore T representa o espaço de busca por soluções factíveis, que é discreto e finito.

A fim de estabelecer a recursão, define-se o **Nível de um Vértice** v como o seu posto $\rho(v)$. Assim, o índice i do vértice v_i está biunivocamente associado ao nível i de T .

A princípio, T é inicializada nos três primeiros vértices $1 \rightarrow 2 \rightarrow 3$, univocamente nas posições x_1, x_2 e $x_3 \in \mathbb{R}^3$, como na formulação matemática. Até este passo não há ramificações ou podas. Em seguida, com o índice dos vértices i sendo incrementado um-a-um de 4 a $n = |V|$, são armazenados:

- a posição $x_i \in \mathbb{R}^3$ referente ao i -ésimo átomo de M e a matriz de torção acumulada $C_i = B_1 B_2 \dots B_i$, e
- três apontadores em função de i :
 - (i) $P(i)$, que aponta para o nó-pai,
 - (ii) $L(i)$, que aponta para os nós à esquerda, e
 - (iii) $R(i)$, que aponta para os nós à direita.

Esta estrutura de apontadores será imprescindível para explorar a árvore de realizações T em buscas em profundidade, de modo que cada caminho da raiz até uma folha corresponde a uma imersão factível de G em \mathbb{R}^3 . Quando um determinado nó é infactível, ele é inicializado com um valor fictício PRUNED para indicar onde houve uma poda a fim de que este caminho seja inviabilizado [54].

De diferente modo em relação à representação da árvore de soluções apresentada em [54, 56], cuja raiz está em cima da estrutura da árvore, a representamos da direita para a esquerda, como na Figura 2.17. Ou seja, a raiz de T está à esquerda de todo o resto da estrutura, em consonância com [77] cujos benefícios serão apresentados mais à frente neste texto.

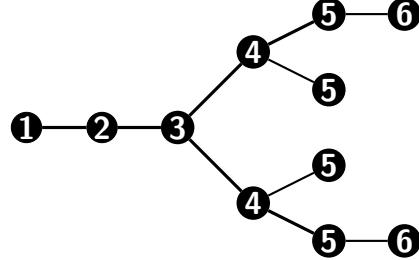


Figura 2.17: Um exemplo da maneira na qual a árvore T é representada

Sejam $y = (0, 0, 0, 1)^T$, a tolerância dada $\varepsilon > 0$ e v um nó de T com posto igual a $i - 1$. A árvore T é codificada por P, L e R . A estrutura recursiva do BP é dada pelo pseudocódigo apresentado no Algoritmo 1, extraído de Lavor et al. [54].

Algoritmo 1 Algoritmo BP

```

1: BranchAndPrune( $T, v, i$ )
2: if  $i \leq n - 1$  then
3:   calcule as matrizes de torção  $B_i$  e  $B'_i$ ;
4:   recupere a matriz de torção acumulada  $C_{i-1}$  referente ao nó-pai  $P(v)$ ;
5:   calcule as próximas matrizes de torção acumuladas  $C_i = C_{i-1}B_i$  e  $C'_i = C_{i-1}B'_i$ ;
6:   utilize-as para calcular as posições  $x_i = C_iy$  e  $x'_i = C'_iy$ ;
7:   if  $x_i$  é factível then
8:     crie um nó  $z$ , armazenando  $C_i$  e  $x_i$ , e fazendo  $P(z) = v$  e  $L(v) = z$ ;
9:     faça  $T \leftarrow T \cup \{z\}$ ;
10:    BranchAndPrune( $T, z, i + 1$ )
11:   else
12:     faça  $L(v) = \text{PRUNED}$ ;
13:   end if
14:   if  $x'_i$  é factível then
15:     crie um nó  $z'$ , armazenando  $C'_i$  e  $x'_i$ , e fazendo  $P(z') = v$  e  $R(v) = z$ ;
16:     faça  $T \leftarrow T \cup \{z'\}$ ;
17:     BranchAndPrune( $T, z', i + 1$ )
18:   else
19:     faça  $R(v) = \text{PRUNED}$ ;
20:   end if
21: else
22:   solução armazenada nos nós-pais de  $n$  a 1, em busca retrocedida.
23: end if

```

O exemplo a seguir foi extraído de Nucci *et al.* [77] e ilustra o funcionamento do algoritmo BP em estruturas DMDGP. Em especial, esta instância permitirá um entendimento da ocorrência de podas e dos retornos (ou *backtrackings*) nas buscas em profundidade pelas soluções. Além disso, ela possui um conjunto de arestas de poda conveniente para exemplificar um caso muito especial para o desenvolvimento deste trabalho.

Considere, portanto, a instância $G = (V, E, d)$ formada por seis vértices, cujo conjunto de arestas é particionado nos dois conjuntos disjuntos $E = E_d \cup E_p$ tais que as arestas de discretização e de poda são dadas, respectivamente, por

- (i) $E_d = \{\{1, 2\}, \{1, 3\}, \{1, 4\}, \{2, 3\}, \{2, 4\}, \{2, 5\}, \{3, 4\}, \{3, 5\}, \{3, 6\}, \{4, 5\}, \{4, 6\}, \{5, 6\}\}$ e
- (ii) $E_p = \{\{2, 6\}\}$.

A partir dos dados de distâncias de discretização em \mathcal{D}_d , é possível estimar os valores dos ângulos de ligação $\theta_{1,3}, \theta_{2,4}, \theta_{3,5}$ e $\theta_{4,6}$, pela simples aplicação da Lei dos Cossenos, e os valores dos cossenos dos ângulos de torção $\omega_{1,4}, \omega_{2,5}$ e $\omega_{3,6}$, usando a Equação (2.3.5) dada anteriormente.

Assim, de posse dessas coordenadas internas, é possível aplicar o BP nesta instância. Todos os passos a seguir serão ilustrados. Os caminhos factíveis serão representados em preto e os factíveis em vermelho.

- (P1) Usando os valores $d_{12}, d_{23} \in \mathcal{D}_d$ juntamente com o ângulo de ligação $\theta_{1,3}$, define-se as matrizes de torção B_1, B_2 e B_3 e, logo, as posições x_1, x_2 e x_3 dos três primeiros vértices, com o primeiro na origem do sistema, o segundo no eixo das abscissas e o terceiro no plano abscissa-ordenada, como descrito pelas Equações (2.3.10). A sub-árvore associada à inicialização com os três primeiros níveis é dada pela Figura 4.9.



Figura 2.18: Os três primeiros níveis da árvore T .

- (P2) Já com o valor de distância $d_{1,4} \in \mathcal{D}_d$, pode-se calcular o valor de $\cos(\omega_{1,4})$. Logo, encontra-se duas posições factíveis para o quarto vértice da sequência, $x_4^{(1)}$ e $x_4^{(2)}$, relativas aos dois valores possíveis para $\sin(\omega_{1,4})$ que são $\pm\sqrt{1 - \cos^2(\omega_{1,4})}$. Essa dupla factibilidade é oriunda da não-existência de arestas de poda relativa a este vértice. Como temos de escolher uma posição, tomamos $x_4^{(1)}$ de início. A sub-árvore com os quatro primeiros níveis de T é dada na Figura 4.10.



Figura 2.19: Os quatro primeiros níveis da árvore T .

- (P3) Analogamente, utilizando o valor de distância $d_{2,5} \in \mathcal{D}_d$, pode-se calcular $\cos(\omega_{2,5})$. Este também gera dois valores possíveis para o seno do ângulo de torção, dados por $\sin(\omega_{2,5}) = \pm\sqrt{1 - \cos^2(\omega_{2,5})}$, os quais geram as respectivas posições $x_5^{(1,1)}$ e $x_5^{(1,2)}$. Também não há distâncias de poda para o quinto vértice. Portanto, escolhemos a posição $x_5^{(1,1)}$ para o primeiro caminho. A sub-árvore com os cinco primeiros níveis de T é dada na Figura 4.11.

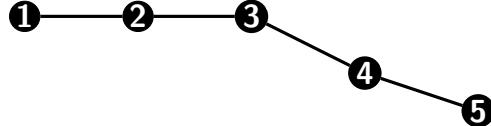


Figura 2.20: Os cinco primeiros níveis da árvore T .

- (P4) Agora, com o valor de $d_{3,6} \in \mathcal{D}_d$, é possível calcular $\cos(\omega_{3,6})$ e, portanto, encontrar duas posições possíveis para o sexto vértice, $x_6^{(1,1)}$ e $x_6^{(1,2)}$, associadas a $\sin(\omega_{3,6}) = \sqrt{1 - \cos^2(\omega_{3,6})}$ e $\sin(\omega_{3,6}) = -\sqrt{1 - \cos^2(\omega_{3,6})}$, respectivamente. Neste caso, há o valor de poda $d_{2,6} \in \mathcal{D}_p$. Escolhemos a posição $x_6^{(1,1)}$ e testamos a sua factibilidade segundo o teste DDF. Como

$$\left(\|x_2 - x_6^{(1,1)}\|^2 - d_{2,6}^2 \right)^2 \geq \varepsilon,$$

esta posição é podada. A sub-árvore associada a este passo é dada pela Figura 4.12.

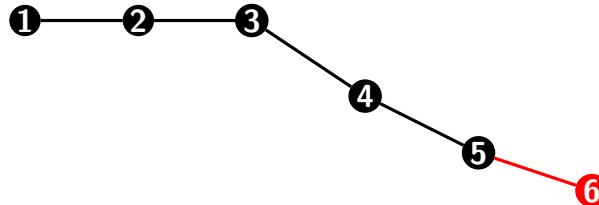


Figura 2.21: Os seis primeiros níveis da árvore T até então: a primeira poda.

- (P5) Retrocedemos a busca fazendo um retorno ao nível anterior e adotamos a posição $x_6^{(1,2)}$, encontrada no passo (P4), para o sexto nível. Novamente, utiliza-se a informação $d_{2,6} \in \mathcal{D}_p$ para checar a factibilidade dessa posição. Aplicando, assim, o teste DDF, encontra-se

$$\left(\|x_2 - x_6^{(1,2)}\|^2 - d_{2,6}^2 \right)^2 < \varepsilon.$$

Portanto, essa posição é factível e, logo, a primeira solução $x^1 \in \mathcal{S}$ é determinada pelo BP. A sub-árvore associada a este passo é dada pela Figura 2.22.

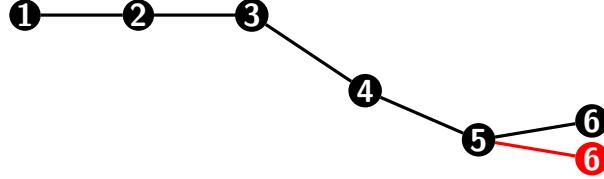


Figura 2.22: Seis primeiros níveis da árvore T até então: primeira solução encontrada.

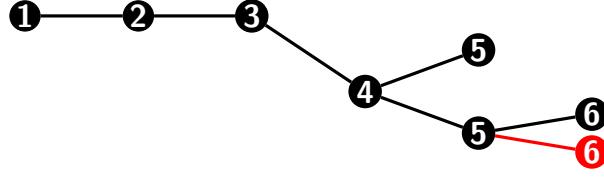


Figura 2.23: Cinco primeiros níveis da árvore T .

- (P6) Realiza-se um retorno até o quarto nível e se avança escolhendo $x_5^{(1,2)}$. A sub-árvore associada a este passo é dada pela Figura 2.23.
- (P7) Analogamente ao passo (P4), mas agora partindo do nó-pai associado à posição $x_5^{(1,2)}$, utilizase a distância $d_{3,6} \in \mathcal{D}_d$ para calcular as posições $x_6^{(2,1)}$ e $x_6^{(2,2)}$. Escolhe-se a primeira e testamos a sua factibilidade em relação à informação $d_{2,6} \in \mathcal{D}_p$, obtendo que

$$\left(\|x_2 - x_6^{(2,1)}\|^2 - d_{2,6}^2 \right)^2 < \varepsilon.$$

Dada a factibilidade dessa posição, encontramos a segunda solução $x^2 \in \mathcal{S}$ para esta instância com seis vértices, cuja sub-árvore é representada na Figura 2.24.

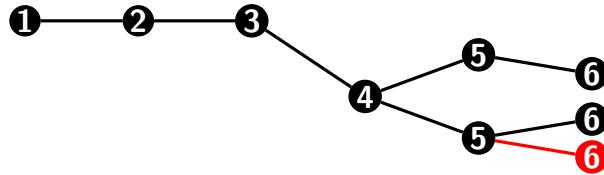


Figura 2.24: Seis primeiros níveis da árvore T : segunda solução encontrada.

- (P8) Ainda falta testar o caminho que termina em $x_6^{(2,2)}$. Usando $d_{2,6}$ para o teste, temos

$$\left(\|x_2 - x_6^{(2,2)}\|^2 - d_{2,6}^2 \right)^2 \geq \varepsilon.$$

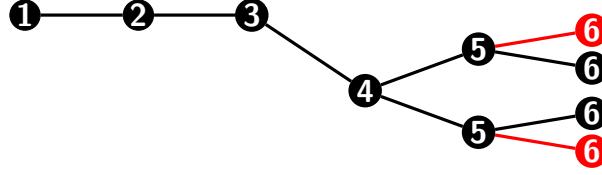


Figura 2.25: Seis primeiros níveis da árvore T : metade dos caminhos explorados.

Portanto, essa posição é infactível. A sub-árvore até então é dada pela Figura 2.25.

- (P9) Em metade dos caminhos possíveis já explorados, encontrou-se duas soluções para o DMDGP em questão. Retorcede-se, então, ao quarto nível e tomamos a segunda posição $x_4^{(2)}$. A sub-árvore para este passo é dada pela Figura 2.26.

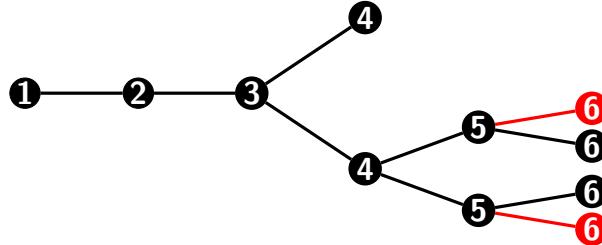


Figura 2.26: Recomeça-se a busca.

- (P10) Também análogo ao passo (P3), utilizamos a informação $d_{2,5} \in \mathcal{D}_d$ para encontrar as duas posições $x_5^{(2,1)}$ e $x_5^{(2,2)}$. Escolhemos, inicialmente, a primeira delas para continuar. Como não há informação de poda, prosseguimos. A sub-árvore associada é dada na Figura 2.27.

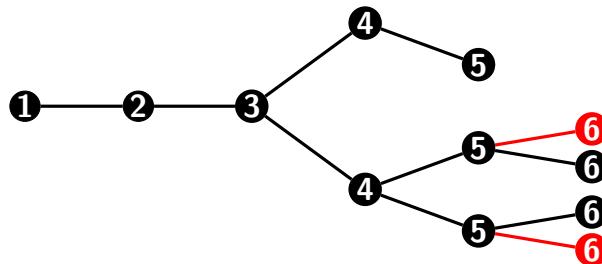


Figura 2.27: Escolhendo o primeiro caminho da segunda metade.

- (P11) Em semelhança, agora, ao passo (P4), utilizamos a distância $d_{3,6} \in \mathcal{D}_d$ para encontrar as duas posições $x_6^{(3,1)}$ e $x_6^{(3,2)}$. Os testes de factibilidade são permitidos, como já vimos, pois existe a

distância $d_{2,6} \in \mathcal{D}_p$. Testando a primeira posição, encontra-se uma infactibilidade. Portanto, essa posição é podada. Retrocedemos até o nó-pai e avançamos para a segunda posição $x_6^{(3,2)}$. Realizando novamente o teste DDF, obtemos uma posição factível, culminando na terceira solução $x^3 \in \mathcal{S}$. Os testes DDF são dados por

$$\left(\|x_2 - x_6^{(3,1)}\|^2 - d_{2,6}^2 \right)^2 \geq \varepsilon \quad \text{e} \quad \left(\|x_2 - x_6^{(3,2)}\|^2 - d_{2,6}^2 \right)^2 < \varepsilon.$$

O caminho até esta posição é factível e, portanto, resolve o DMDGP. A sub-árvore em questão é dada pela Figura 2.28.

- (P12) No último passo, retrocedemos até o quinto nível, novamente, escolhendo a posição $x_5^{(2,2)}$, e utilizamos a informação $d_{3,6} \in \mathcal{D}_d$ para encontrar as posições $x_6^{(4,1)}$ e $x_6^{(4,2)}$. Segundo o teste DDF para a distância de poda, $d_{2,6}$, a primeira é factível (encontrando a quarta solução $x^4 \in \mathcal{S}$) e a segunda é infactível

$$\left(\|x_2 - x_6^{(4,1)}\|^2 - d_{2,6}^2 \right)^2 \geq \varepsilon \quad \text{e} \quad \left(\|x_2 - x_6^{(4,2)}\|^2 - d_{2,6}^2 \right)^2 \geq \varepsilon.$$

Portanto, a posição $x_6^{(4,2)}$ é podada. A árvore T com os quatro caminhos factíveis e os quatro caminhos infactíveis é dada na Figura 2.28.

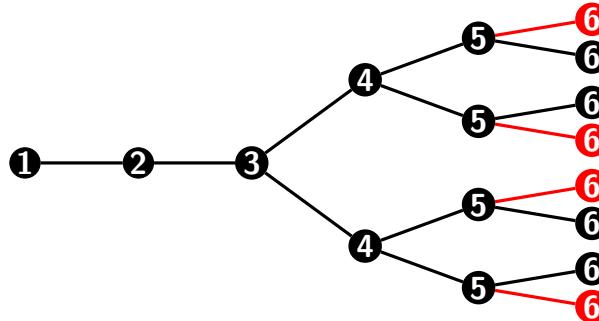


Figura 2.28: Árvore explorada completamente.

2.4.3 Representação Eficiente de Soluções

Existe uma maneira computacionalmente mais eficiente para representar as soluções de um DMDGP descobertas através do algoritmo *Branch & Prune*. Esta estrutura binária das soluções permite identificar biunivocamente cada uma das soluções com uma sequência de zeros e uns. Abaixo, descreveremos um simples algoritmo com o fim de gerar tais sequências.

- As primeiras três posições dessa sequência Booleana são dadas por três zeros, já que são únicas.
- Da quarta posição pra frente:

- (i) a entrada é 0, caso represente a posição à esquerda e
- (ii) a entrada é 1, caso represente a posição à direita.

Voltando ao exemplo, as quatro soluções são representadas, respectivamente, pelas sequências:

- $x^1 = (0, 0, 0, 0, 0, 1)$
- $x^2 = (0, 0, 0, 0, 1, 0)$
- $x^3 = (0, 0, 0, 1, 0, 1)$
- $x^4 = (0, 0, 0, 1, 1, 0).$

O próximo capítulo trata sobre as simetrias que existem no DMDGP e esta representação será de grande valia para expressar as soluções simétricas.

Capítulo 3

Simetrias do DMDGP

O algoritmo BP é um método eficiente para encontrar imersões isométricas para instâncias que possuem uma ordem DMDGP em seu conjunto de vértices.

Notou-se, assim, que esta eficiência, em detrimento de outros métodos, advém do fato de que a largura da árvore binária de soluções é limitada, no que concerne aos resultados obtidos em um conjunto de testes realizados nos trabalhos [60, 68]. Dessa forma, o número de soluções não cresce tanto, mesmo que para instâncias muito grandes.

Como o tempo de computação para o BP depende do número total de soluções, a maior parte dos artigos da literatura do DMDGP divide o seu uso em dois casos de interesse [68]:

- utilizar o BP para encontrar apenas uma solução (BP - One) ou
- utilizar o BP para encontrar todas as soluções factíveis (BP - All), como no exemplo apresentado no final do Capítulo 2.

Além disso, descobriu-se a existência de algumas simetrias inerentes à própria natureza da estrutura DMDGP, como resumido a seguir.

Inicialmente, demonstrou-se a existência de uma simetria básica no quarto nível da árvore de realizações [54], a qual provou ser “universal” no sentido de estar presente em qualquer instância do problema. Os detalhes dessa Simetria de Quarto Nível estão presentes na Seção 3.1.

Depois, após longa pesquisa no tema, descobriu-se uma relação de simetria mais geral, que pode ocorrer em vários níveis de uma estrutura DMDGP [61, 52]. Mais do que isso: descobriu-se que tais simetrias são poderosas aliadas na diminuição do tempo de computação do BP. Veremos, na Seção 3.2, que ao encontrar uma única solução, todas as remanescentes são determinadas através de reflexões parciais desta solução [61]. Portanto, as simetrias nos permitem conhecer quais os caminhos que o BP deve evitar a fim de que a busca por soluções seja eficiente até encontrar a primeira solução e, a partir dela, construir todas as outras [62].

Após discorrer sobre essas simetrias, na Seção 3.3, vamos analisar o chamado Sym BP, que é a adaptação do BP - One para a utilização de simetrias a fim de encontrar todas as soluções factíveis de uma instância.

3.1 Simetria do Quarto Nível

Em busca de estabelecer um padrão na contagem de realizações factíveis para um DMDGP, Lavor *et al.* demonstraram a existência de uma simetria fundamental no quarto nível da árvore binária que representa o espaço de busca por soluções [54].

Em resumo: existe uma identificação biunívoca entre pares das realizações factíveis, o que nos permite considerar as soluções de apenas um dos lados - um ramo - desta árvore. As realizações do outro ramo, portanto, serão construídas a partir de reflexões destas em relação a um plano de simetria, definido pelas únicas posições dos três vértices iniciais da instância, respectivamente.

Dessa maneira, esta relação de simetria aponta para uma significativa redução do esforço computacional para determinar conformações tridimensionais para instâncias do DMDGP [54].

Nesta seção, discorreremos sobre os principais resultados dessa relação, que é chamada formalmente de *Simetria do Quarto Nível* para instâncias DMDGP [54]. As demonstrações, por sua vez, serão omitidas, a fim de garantir a fluência do texto, mas podem ser encontradas nas referências citadas em cada um dos resultados. Além disso, o exemplo anterior é revisitado e as suas soluções são determinadas novamente, agora, permitindo que o BP utilize esta poderosa ferramenta.

De acordo com o Capítulo 2, ambas as posições para o quarto vértice de uma instância G do DMDGP (x_4^1 e x_4^2 , onde x é uma realização de G) são factíveis, já que não existem informações de poda associada a v_4 . Isto independe da instância G . Dessa maneira, qualquer caminho factível da raíz até as folhas da árvore de realizações de G contém a posição x_4^1 ou a posição x_4^2 .

O principal teorema dessa seção tem por objetivo identificá-las de par-em-par. Mas, antes de enunciá-lo, há que se considerar dois lemas. Seja G uma instância arbitrária do DMDGP.

Lema 3.1.1 (Lavor *et al.* [54]). Seja $Q_i = B_4 \dots B_i$, para $i \in \{4, \dots, n\}$, representada por

$$Q_i = \begin{bmatrix} q_{11}^i & q_{12}^i & q_{13}^i & q_{14}^i \\ q_{21}^i & q_{22}^i & q_{23}^i & q_{24}^i \\ q_{31}^i & q_{32}^i & q_{33}^i & q_{34}^i \\ 0 & 0 & 0 & 1 \end{bmatrix}.$$

Se, para cada $i \in \{4, \dots, n\}$, o sinal de $\text{sen}(\omega_{i-3,i})$ for invertido em B_i , originando a matriz B'_i , então a matriz acumulada $Q'_i = B'_4 \dots B'_i$ é representada por

$$Q_i = \begin{bmatrix} q_{11}^i & q_{12}^i & -q_{13}^i & q_{14}^i \\ q_{21}^i & q_{22}^i & -q_{23}^i & q_{24}^i \\ -q_{31}^i & -q_{32}^i & q_{33}^i & -q_{34}^i \\ 0 & 0 & 0 & 1 \end{bmatrix}.$$

Lema 3.1.2 (Lavor *et al.* [54]). Seja $x = (x_1, x_2, x_3, x_4, \dots, x_n) \in \mathbb{R}^{3n}$ uma realização para G , definida pela sequência de ângulos de torção $\omega_{1,4}, \omega_{2,5}, \dots, \omega_{n-3,n}$. Se, para cada $i \in \{4, \dots, n\}$, o sinal de $\text{sen}(\omega_{i-3,i})$ for invertido em B_i , então $x' = (x_1, x_2, x_3, x'_4, \dots, x'_n) \in \mathbb{R}^{3n}$ também é realização factível de G , cujas posições são dadas por

$$x'_i = \begin{bmatrix} x'_{i1} \\ x'_{i2} \\ x'_{i3} \end{bmatrix} = \begin{bmatrix} x_{i1} \\ x_{i2} \\ -x_{i3} \end{bmatrix}, \quad i \in \{4, \dots, n\}.$$

Este segundo lema pode ser considerado mais como um corolário do primeiro. Estabelece-se, portanto, o seguinte resultado para as realizações de uma instância DMDGP.

Teorema 3.1.1 (Lavor *et al.* [54]). Seja x uma realização para uma instância $G = (V, E, d)$ do DMDGP, dada pela sequência de ângulos de torção $\omega_{1,4}, \dots, \omega_{n-3,n}$. Se invertermos os sinais de $\text{sen}(\omega_{i-3,i})$ em B_i , para $i \in \{4, \dots, n\}$, então obtemos outra realização x' para G .

Em suma, quando o sinal de $\text{sen}(\omega_{1,4})$ é invertido, a matriz B_4 é transformada através de uma reflexão em relação ao plano definido pelas posições x_1, x_2 e x_3 .

Portanto, toda realização $x = (x_1, x_2, x_3, x_4^1, \dots)$ pertencente à árvore \mathcal{T} de realizações de G é associada a uma outra realização $x' = (x_1, x_2, x_3, x_4^2, \dots)$ de G .

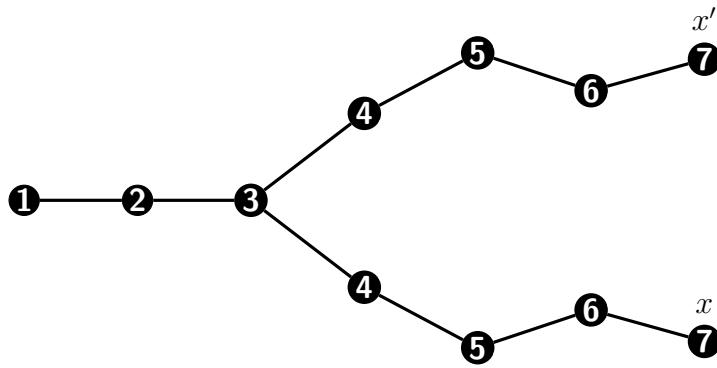


Figura 3.1: Duas realizações associadas pela troca do sinal em $\text{sen}(\omega_{i-3,i})$.

Isto exemplifica com propriedade o que ocorre ao se considerar a simetria no quarto nível: a simples troca de sinal de $\text{sen}(\omega_{i-3,i})$ determina outra solução, evitando que o BP necessite explorar mais do que a metade da árvore \mathcal{T} .

Essa troca de sinal é evidenciada na representação binária de soluções como o intercâmbio dos zeros pelos uns e vice-versa. No exemplo, essa troca faz com que a representação binária de x seja transformada na representação de x'

$$x = (0, 0, 0, 0, 0, 1, 0) \quad \longmapsto \quad x' = (0, 0, 0, 1, 1, 0, 1).$$

Isto mostra, em resumo, que toda realização para G pode ser gerada a partir da reflexão no quarto nível de outra. Por essa razão, define-se uma relação de congruência no conjunto \mathcal{S} de soluções do DMDGP.

Definição 3.1.1 (Realizações Congruentes e Incongruentes). Duas realizações x e x' para G são ditas **Congruentes** se satisfazem as condições e resultados do Teorema 3.1.1. Caso contrário, essas realizações são ditas Incongruentes.

Esta relação de congruência, portanto, define uma partição disjunta no conjunto \mathcal{S} dada por

$$\mathcal{S} = \mathcal{L}_G \cup \mathcal{R}_G, \tag{3.1.1}$$

onde tais componentes são definidos por

- $\mathcal{L}_G = \{x = (0, 0, 0, 0, \dots) \text{ é solução}\}$ é chamado de **Ramo Esquerdo**, pois contém a metade esquerda das realizações pertencentes à árvore \mathcal{T} e
- $\mathcal{R}_G = \{x = (0, 0, 0, 1, \dots) \text{ é solução}\}$ é chamado de **Ramo Direito**, pois contém a metade direita das realizações pertencentes a \mathcal{T} .

Ou seja, a pertinência a \mathcal{L}_G ou a \mathcal{R}_G faz da simetria de quarto nível uma relação de equivalência \sim no conjunto \mathcal{S} , o que justifica tal partição.

Dessa forma, ao invés do BP explorar toda \mathcal{T} para determinar \mathcal{S} , basta que explore uma das metades. Neste trabalho, \mathcal{L}_G é escolhido como a parte determinada pelo BP. Então, tais realizações são transformadas pela reflexão de quarto nível R , obtendo as realizações de \mathcal{R}_G , já que

$$\mathcal{R}_G = \mathcal{T}(\mathcal{L}_G) = \{T(x) : x \in \mathcal{L}_G\}. \quad (3.1.2)$$

Há que se observar uma consequência imediata de se explorar apenas um dos ramos: o número total de soluções de um DMDGP é par. Mais do que isso, o número de soluções é dado em potência de dois. Já havia se conjecturado isso, mas a demonstração matemática veio a posteriori com o uso de Teoria de Grupos [61].

Exemplo

Voltando ao último exemplo dado no capítulo anterior, o BP explorou toda a árvore de soluções possíveis para encontrar as quatro soluções factíveis para a determinada instância que são

$$x^1 = (0, 0, 0, 0, 0, 1), \quad x^2 = (0, 0, 0, 0, 1, 0), \quad x^3 = (0, 0, 0, 1, 0, 1) \quad \text{e} \quad x^4 = (0, 0, 0, 1, 1, 0). \quad (3.1.3)$$

Utilizando os conceitos apresentados nesse capítulo, deve-se explorar apenas a metade esquerda desta árvore, na qual encontramos apenas as soluções x^1 e x^2 , como é mostrado na Figura 3.2.

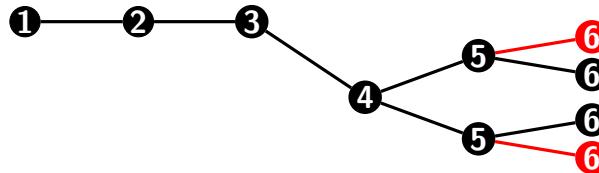


Figura 3.2: Árvore explorada pela metade, encontrando as primeiras duas soluções: x^1 e x^2 .

As outras duas soluções x^3 e x^4 , portanto, são encontradas via reflexões parciais, como segue:

- Considere os pontos a_1, a_2 e a_3 tais que

$$a_1 = x_1^1, \quad a_2 = x_2^1 \quad \text{e} \quad a_3 = x_3^1.$$

- Seja R a reflexão cujo plano de simetria é o definido pelos pontos a_1, a_2 e a_3 .
- Então,

$$x^3 = Rx^2 \quad \text{e} \quad x^4 = Rx^1.$$

A Figura 3.3 representa a combinação do algoritmo BP com a simetria de quarto nível.

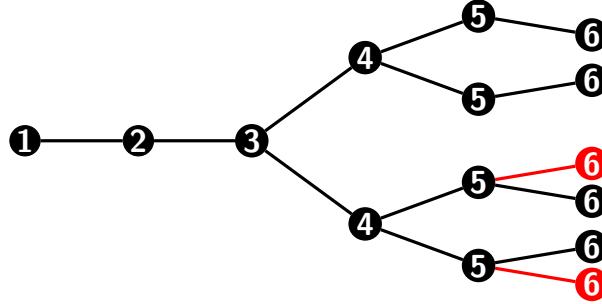


Figura 3.3: A simetria do quarto nível aplicada em x^1 e x^2 , encontrando as soluções x^3 e x^4 .

3.2 Explorando o conjunto de simetrias

Nesta seção, nosso objetivo é estudar os principais resultados já descobertos sobre as relações de simetrias existentes para estruturas DMDGP e dar exemplos que mostram a eficiência de seu uso na descoberta das soluções, em comparação com o BP - All.

Ela será dividida em duas subseções. Na Seção 3.2.1, discorreremos sobre a contagem do número de soluções de um DMDGP usando as arestas de poda e, na Seção 3.2.2, analisaremos a definição de vértice de simetria do DMDGP e suas implicações. Além disso, vamos propor algumas contribuições nesta última seção utilizando ferramentas da anterior para caracterizar tais vértices.

3.2.1 Contando o número de soluções de um K DMDGP

Os desenvolvimentos desta seção serão baseados nos resultados do trabalho de Liberti *et al.* [55] sobre a contagem de soluções de um DMDGP de dimensão $K \in \mathbb{Z}^+$ arbitrária.

Sejam $G = (V, E, d)$ uma estrutura DMDGP, tal que $n = |V|$, e X o conjunto de todas as realizações incongruentes de G em \mathbb{R}^K , as quais provêm da aplicação do BP em G com a utilização da simetria de quarto nível.

Portanto, X pode ser definido como o conjunto-quociente de \mathcal{S} pela relação de equivalência \sim , isto é, $X = \mathcal{S}/\sim$.

Para o estudo das simetrias em geral é preciso definir o seguinte operador.

Definição 3.2.1 (Liberti *et al.* [58, 55]). Para quaisquer $x \in X$ e $v_i \in V$, com $i > K$, seja $R_x^{v_i}$ a reflexão pelo hiperplano definido pelos pontos x_{i-K}, \dots, x_{i-1} , como na Figura 3.4. Assim, para todo $i > K$, define-se o *Operador de Reflexão Parcial* com respeito a v_i de x como

$$g_{v_i}(x) = (x_1, x_2, x_3, \dots, x_{i-1}, R_x^{v_i}(x_i), R_x^{v_i}(x_{i+1}), \dots, R_x^{v_i}(x_n)). \quad (3.2.1)$$

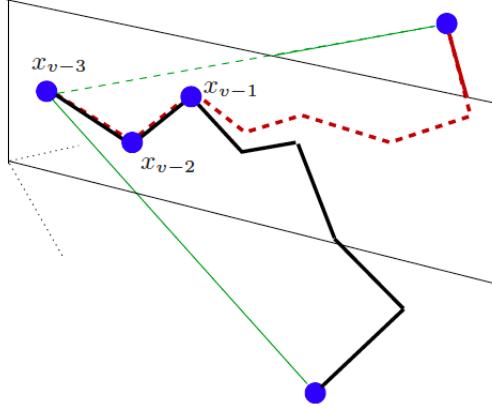


Figura 3.4: Reflexão pelo hiperplano definido por x_{i-K}, \dots, x_{i-1} . Figura extraída de [55].

O operador g_{v_i} reflete as posições relativas aos vértices de v_i até v_n . Além disso, é possível definir uma operação binária entre reflexões parciais também.

Definição 3.2.2 (Liberti *et al.* [55]). Define-se o produto de dois operadores de reflexão g_{v_j} e g_{v_i} pela composição

$$g_{v_j} g_{v_i} = g_{v_j} \circ g_{v_i}. \quad (3.2.2)$$

Sem perda de generalidade, suponha que $j > i > K$. Segue, portanto, que o produto de operadores de reflexão é dado por

$$\begin{aligned} g_{v_j} g_{v_i}(x) &= g_{v_j}(g_{v_i}(x)) = g_{v_j}(x_1, \dots, x_{i-1}, R_x^{v_i}(x_i), R_x^{v_i}(x_{i+1}), \dots, R_x^{v_i}(x_n)) \\ &= (x_1, \dots, x_{i-1}, R_x^{v_i}(x_i), \dots, R_x^{v_i}(x_{i-1}), R_{g_{v_j}(x)}^{v_j}(x_j), \dots, R_{g_{v_j}(x)}^{v_j}(x_n)). \end{aligned}$$

Esta operação binária entre operadores de reflexão parcial tem uma propriedade fundamental para a futura abordagem via Teoria de Grupos.

Lema 3.2.1 (Liberti *et al.* [58]). O operador de reflexão parcial é comutativo em relação ao seu produto, isto é, para uma realização $x \in X$ e dois vértices $v_i, v_j \in V$ tais que $i, j > K$ na ordem K DMDGP, vale

$$g_{v_i} g_{v_j}(x) = g_{v_j} g_{v_i}(x). \quad (3.2.3)$$

O propósito inicial desta seção será estudar um grupo invariante associado às arestas de poda. As arestas de discretização também definem um grupo invariante [58, 55], o qual não será necessário para o resultado principal que será discutido aqui.

Considere, para tanto, que $E_p \neq \emptyset$.

Definição 3.2.3 (Liberti *et al.* [55]). O **Conjunto Gerado** pela aresta de poda $\{v_i, v_j\} \in E_p$ é dado por

$$S^{i,j} = \{i + K + 1, \dots, j\}, \quad (3.2.4)$$

supondo que $i < j$ sem perda de generalidade.

Para o caso onde $K = 3$, o DMDGP clássico, tem-se que

$$S^{i,j} = \{i + 4, \dots, j\}, \quad (3.2.5)$$

o qual sempre é possível de ser definido já que $|i - j| \geq 4$.

Este conjunto armazena, portanto, todos os vértices que estão no caminho entre v_i e v_j , excluindo-se os quatro primeiros, considerando a existência da simetria de quarto nível localmente.

Para enunciar-se, então, o principal resultado dessa seção, é necessário definir o conjunto

$$\Gamma_P = \{g_{v_t} | t > K \text{ e } v_t \notin S^{i,j}, \forall \{u, w\} \in E_p\}. \quad (3.2.6)$$

Observe, primeiramente, que se $g_{v_t} \in \Gamma_P$, então

$$t > K \text{ e } v_t \in V \setminus \left(\bigcup_{\{v_i, v_j\} \in E_p} S^{i,j} \right). \quad (3.2.7)$$

Este conjunto, que armazena os operadores de reflexão parcial associados somente aos vértices que têm posto maior do que K e que não estão no conjunto gerado por nenhuma aresta de poda disponível, gera o grupo mais importante para a contagem de soluções factíveis de um K DMDGP.

Definição 3.2.4 (Liberti *et al.* [58, 55]). O grupo gerado pelos elementos do conjunto Γ_P através do produto de reflexões parciais é chamado de **Grupo de Poda** e denotado por

$$\mathcal{G}_P = \langle \Gamma_P \rangle. \quad (3.2.8)$$

A definição desta estrutura algébrica, portanto, nos permite fazer sucessivas reflexões parciais em níveis distintos em uma estrutura DMDGP. Entretanto, o seguinte resultado depende totalmente do conjunto Γ_P .

Proposição 3.2.1 (Liberti *et al.* [55]). O número de realizações incongruentes de um K DMDGP com probabilidade 1 é

$$|X| = 2^{|\Gamma_P|}. \quad (3.2.9)$$

Tais definições e resultados foram enunciados com o objetivo de se garantir sustentação teórica para a demonstração do seguinte teorema, a ser apresentada neste trabalho. Este resultado tem o propósito de contar as realizações factíveis de um DMDGP ($K = 3$).

Teorema 3.2.1. Sejam $G = (V, E, d)$, uma instância DMDGP, e $\{v_i, v_j\} \in E_p$ uma aresta de poda. Então, com probabilidade 1, existem duas realizações factíveis de v_i até v_j na árvore de soluções \mathcal{T} para cada posição factível de v_i .

Demonstração. Considere o subconjunto de vértices associado à aresta $\{v_i, v_j\} \in E_p$ dado por

$$V^{i,j} = \{i, i + 1, i + 2, \dots, j - 1, j\} \subseteq V$$

e o subconjunto $E^{i,j} \subseteq E$ que contém apenas as arestas que envolvem os vértices de $V^{i,j}$.

Com isso, pode-se tomar uma subinstância de G dada por

$$G^{i,j} = (V^{i,j}, E^{i,j}, d)$$

associada diretamente à aresta de poda $\{v_i, v_j\}$.

Logo, particionando o conjunto de arestas desta subinstância como

$$E^{i,j} = E_d^{i,j} \cup E_p^{i,j},$$

define-se o conjunto Γ_P de reflexões parciais para esta subinstância como

$$\Gamma_P^{i,j} = \{g_{v_t} : t > 3 \text{ e } v_t \notin S^{r,s}, \forall \{v_r, v_s\} \in E_p^{i,j}\}. \quad (3.2.10)$$

Agora, como $\{v_i, v_j\}$ pertence a E_p^{uw} e abrange todo o conjunto V^{uw} , tem-se que

$$\Gamma_P^{i,j} = \emptyset$$

o que ocorre se, e somente se,

$$|\Gamma_P^{i,j}| = 0.$$

Denotamos, adicionalmente, o conjunto de realizações factíveis de $G^{i,j}$ como $X^{i,j}$.

De acordo com a Proposição 3.2.1, portanto, segue que o número de realizações incongruentes da instância $G^{i,j}$, com probabilidade 1, é igual a

$$|X^{i,j}| = 2^{|\Gamma_P^{i,j}|} = 2^0 = 1.$$

Ou seja, desprezando a simetria no quarto nível, existem duas soluções factíveis para G^{uw} com probabilidade 1.

Portanto, para cada posição factível para o vértice $v_i \in V$, existem apenas dois caminhos factíveis entre os vértices v_i e v_j na árvore de soluções \mathcal{T} (com probabilidade 1), gerada pelo algoritmo *Branch & Prune*, os quais são congruentes um ao outro, como queríamos demonstrar. \square

Decorre deste teorema o seguinte corolário.

Corolário 3.2.1. Seja $G = (V, E, d)$ uma instância do DMDGP. Se $\{v_1, v_n\} \in E$, então existe apenas uma solução incongruente para este DMDGP (com probabilidade 1) a menos da simetria de quarto nível.

Demonstração. Pelo Teorema 3.2.1, existem apenas duas soluções para o DMDGP associado a G , as quais são dadas pelo BP, com probabilidade 1, já que $\{v_1, v_n\} \in E$. Além disso, essas duas conformações são congruentes através da simetria do quarto nível. Portanto, $|X| = 1$. \square

3.2.2 O conjunto dos vértices de simetria

Nesta seção, vamos discutir a generalização da idéia de simetrias em estruturas DMDGP. Assim, o objetivo principal aqui será determinar quais vértices são de simetria, bem como suas propriedades e resultados. Com isso, será possível considerar uma partição no conjunto \mathcal{S} em mais do que dois subconjuntos (como na Seção 3.1), já que é possível que se defina mais do que uma relação de congruência (equivalência) que as associe de par-em-par.

Estas simetrias devem ser descritas a partir das reflexões parciais que possam ocorrer, como estudado na última seção. A partir disso, a determinação do número total de soluções de um DMDGP poderá ser feita a partir do número dessas simetrias existentes em cada instância. Ou seja, possuindo a quantidade de simetrias, sabe-se exatamente quantas realizações factíveis existem sem mesmo ter de calculá-las previamente.

Antes desses resultados, é preciso definir o conceito mais fundamental desta seção.

Definição 3.2.5. Seja $G = (V, E, d)$ uma instância DMDGP com n vértices. Um vértice $v_i \in V$ é chamado de **Vértice de Simetria** se toda realização factível $x^k = (x_1^k, \dots, x_{i-1}^k, x_i^k, \dots, x_n^k) \in \mathcal{S}$ é congruente a outra realização $x^j \in \mathcal{S}$, também factível, de modo que

$$x^j = (x_1^k, \dots, x_{i-1}^k, R_{x^k}^{v_i}(x_i^k), \dots, R_{x^k}^{v_i}(x_n^k)), \quad (3.2.11)$$

onde $R_{x^k}^{v_i}$ denota uma reflexão parcial da estrutura no nível i em relação ao plano definido pelos pontos $x_{i-3}^k, x_{i-2}^k, x_{i-1}^k \in \mathbb{R}^3$.

A partir dessa definição, notamos a possibilidade de associar vértices de simetria aos elementos do conjunto gerador do grupo de poda da mesma instância. Como este resultado (vital para este trabalho) não está explicitado nas referências bibliográficas, enunciamos e demonstramos este fato no seguinte teorema.

Teorema 3.2.2. Um vértice $v_i \in V$ é de simetria se, e somente se, $g_{v_i} \in \Gamma_P$.

Demonstração. Seja $g_{v_i} \in \Gamma_P$. Então, $i > 3$ e

$$v_i \notin S^{j,k} = \{j+4, \dots, k\}, \quad \forall \{v_j, v_k\} \in E_p. \quad (3.2.12)$$

Logo, não haverá podas neste nível e, assim, todas as posições possíveis são factíveis para v_i . Dessa forma, dada a estrutura de reflexões do DMDGP, uma realização $y = g_{v_i}(x)$ é sempre factível, para qualquer realização $x \in \mathcal{S}$. Portanto, pela Definição 3.2.5, v_i é um vértice de simetria.

Por outro lado, suponha simultaneamente que v_i seja um vértice de simetria e que $g_{v_i} \notin \Gamma_P$. Assim, $v_i \in S^{j,k}$, para algum $\{v_j, v_k\} \in E_p$, o que é impossível de se acontecer, já que, por definição, os pontos $x_{i-3}^t, x_{i-2}^t, x_{i-1}^t$ não podem definir um plano de simetria, para qualquer realização x^t . Isto viola o fato de que v é um vértice de simetria. Portanto, se v_i é de simetria, então $g_{v_i} \in \Gamma_P$. \square

Na intenção de organizar esses dados de simetria de uma estrutura $G = (V, E, d)$ do DMDGP, Mucherino *et al.* [68] definiu o subconjunto de vértices (originalmente, B [68] ou S [52])

$$S_G = \{v_i \in V : \nexists \{v_j, v_k\} \in E \text{ tal que } j+3 < i \leq k\}. \quad (3.2.13)$$

Com a demonstração do teorema anterior, a definição desse conjunto faz sentido.

Este conjunto, de acordo com o resultado a seguir, caracteriza os vértices de simetria, mostrando que este consiste no conjunto dos índices dos operadores de reflexão parcial de Γ_P .

Teorema 3.2.3. Os vértices do conjunto S_G consistem em todos os vértices de simetria de G .

Demonstração. Seja $v_i \in S_G$. Pela definição em (3.2.13), $\nexists \{v_j, v_k\} \in E$ tal que

$$j + 3 < i \leq k. \quad (3.2.14)$$

Inicialmente, podemos destacar (por (3.2.14)) que

$$k \geq i. \quad (3.2.15)$$

Agora, pela mesma relação, como $j < k$ e $j < i$, segue que $k - j > 0$ e $i - j > 0$. Logo, vale dizer que

$$k - j \geq i - j. \quad (3.2.16)$$

Mas, a partir de (3.2.14), temos que

$$j + 3 < i. \quad (3.2.17)$$

Segue, dessa maneira, que $i - j > 3$ e, logo, que

$$i - j \geq 4. \quad (3.2.18)$$

Portanto, com simples manipulações nos índices, obtém-se que i, j e k devem satisfazer as restrições

$$k - j \geq i - j \geq 4. \quad (3.2.19)$$

Ou seja, $\nexists \{v_j, v_k\} \in E$ tal que $k - j \geq 4$. Em resumo, conclui-se que o fato de $v_i \in S_G$ significa que não existe nenhuma aresta de poda $\{v_j, v_k\} \in E_p$ tal que $i \in \{j + 4, j + 5, \dots, k\}$.

Agora, como definiu-se que $S^{j,k} = \{j + 4, j + 5, \dots, k\}$, então pode-se dizer que

$$v_i \in V \setminus \left(\bigcup_{\{v_j, v_k\} \in E_p} S^{j,k} \right). \quad (3.2.20)$$

Logo, $g_{v_i} \in \Gamma_P$, ou seja, se $x^t \in \mathcal{S}$, então $x^s = g_{v_i}(x^t) \in \mathcal{S}$ também. Portanto, v_i é um vértice de simetria.

Analogamente, se tomarmos um vértice de simetria v_i e seguir-se os passos inversos do feito acima, teremos que $v_i \in S_G$, como queríamos demonstrar. \square

Este teorema considera a existência de mais de um vértice de simetria e os armazena neste conjunto S_G . Agora, vimos, na Seção 3.1, que a simetria de quarto nível sempre está presente em qualquer instância DMDGP. A próxima proposição garante a ocorrência desse fato, utilizando a definição do conjunto S_G , em uma demonstração mais suscinta.

Proposição 3.2.2. O quarto vértice sempre será um vértice de simetria para uma instância $G = (V, E, d)$ do DMDGP.

Demonstração. Se o vértice v_4 não pertencesse a este conjunto S_G , então existiria uma aresta $\{v_j, v_k\} \in E$ tal que

$$j + 3 < 4 \leq k. \quad (3.2.21)$$

Ou seja, os índices dos vértices v_j e v_k deveriam satisfazer a restrição

$$j < 1 < k - 3, \quad (3.2.22)$$

o que é impossível de acontecer já que $j \geq 1$. Portanto, $v_4 \in S_G$, como queria-se provar. \square

Unindo os resultados da Proposição 3.2.1 e do Teorema 3.2.3, obtemos o seguinte corolário a fim de calcular a cardinalidade do conjunto \mathcal{S}_G .

Corolário 3.2.2. O número de soluções de um DMDGP associado a uma instância $G = (V, E, d)$ é dado por

$$|\mathcal{S}| = 2^{|S_G|}. \quad (3.2.23)$$

3.3 *SymBP*: uma versão adaptada do BP para o uso de simetrias

Como se viu nas seções anteriores deste capítulo, as informações sobre quais vértices da estrutura DMDGP possuem a característica de definir simetrias possibilitam não somente a contagem parcial ou total do número de soluções como também a definição de todas as possíveis soluções existentes através de apenas uma delas.

Dessa forma, viabilizou-se o desenvolvimento de um algoritmo que determina todas as soluções de um DMDGP usando esta estrutura de simetrias. Inicialmente, ele utiliza o BP no modo clássico para encontrar a primeira solução. Em seguida, determina-se todas as outras soluções usando reflexões parciais da primeira solução e das que se encontra subsequentemente. A esta modificação, Mucherino *et al.* denominou *Symmetry-Driven BP* (ou simplesmente, *SymBP*) [68].

No que segue, apresenta-se um resumo deste método, total e fielmente baseado em [68], a fim de desenvolvimentos posteriores neste presente trabalho.

Por ser uma adaptação, o *SymBP* tem por base principal o próprio algoritmo BP. Dessa maneira, as entradas do BP fazem parte das entradas do novo algoritmo, as quais são:

- v é o vértice atual a ser posicionado no espaço,
- n é o número de vértices da instância $G = (V, E, d)$ (ou seja, $n = |V|$),
- d é o conjunto de distâncias disponíveis para a instância,
- S é o conjunto de simetrias existentes para a instância G (mais propriamente denotado neste trabalho como S_G),
- $nsols$ é um parâmetro que, dinamicamente, monitora o número de soluções encontradas e
- $prev$ é um vetor de variáveis binárias (0/1), o qual contém em si a última solução encontrada e traduzida para o formato binário de representação, apresentado na Seção 2.4.3.

Como apresentado em [68], segue abaixo um pseudocódigo para este algoritmo, a ser explanado a seguir com mais detalhes de sua mecânica.

Algoritmo 2 O Algoritmo *SymBP*

```

1: symBP( $v, n, d, S, nsols, prev$ )
2: if ( $v > n$ ) then
3:    $nsols \leftarrow nsols + 1$ ;                                 $\triangleright$  Uma solução é encontrada
4:   remova todos os galhos tais que:                          $\triangleright$  Descarte os galhos infactíveis
   • sua raíz é  $u \in S$ ;
   • suas folhas são  $v_f < n$ .
return  $nsols$ ;
5: end if
6: if ( $v \in S$ ) then                                          $\triangleright$  Ramifique em nós possíveis
7:   calcule  $x_v^0$ ;
8:   faça  $prev(v) = 0$ ;
9:   symBP( $v + 1, n, d, S, nsols, prev$ );
10:  calcule  $x_v^1$ ;
11:  faça  $prev(v) = 1$ ;
12:  symBP( $v + 1, n, d, S, nsols, prev$ );
13: end if                                                  $\triangleright$  Sem podas, não há soluções ainda
14: if ( $v \notin S$  e  $nsols = 0$ ) then                       $\triangleright$  Calcule as posições viáveis: faça as podas
15:   calcule  $x_v^0$ ;
16:   if ( $x_v^0$  é factível) then
17:     faça  $prev(v) = 0$ ;
18:     symBP( $v + 1, n, d, S, nsols, prev$ );
19:   end if
20:   if ( $nsols = 0$ ) then
21:     calcule  $x_v^1$ ;
22:     if ( $x_v^1$  é factível) then
23:       faça  $prev(v) = 1$ ;
24:       symBP( $v + 1, n, d, S, nsols, prev$ );
25:     end if
26:   end if
27: end if
28: if ( $v \notin S$  e  $nsols > 0$ ) then                       $\triangleright$  Se a primeira solução já foi calculada:
29:   calcule  $x_v^{-prev(v)}$ ;
30:   faça  $prev(v) = \neg prev(v)$ ;                          $\triangleright$  as outras serão geradas por simetria
31:   symBP( $v + 1, n, d, S, nsols, prev$ );
32: end if return  $nsols$ ;

```

A estrutura desta adaptação do BP também utiliza a recursividade.

Inicialmente, o algoritmo *SymBP* analisa se o último vértice (x_n), na ordem DMDGP, foi

posicionado na última chamada realizada deste método. Neste caso, o parâmetro de contagem de soluções $nsols$ é atualizado, pois uma solução foi encontrada

$$nsols \leftarrow nsols + 1.$$

Se, ao menos, uma das soluções já foi determinada, então as escolhas de 0/1 estão definidas para cada vértice $v_i \notin S_G$. A fim de “refinar” a atual árvore de soluções, os caminhos que não definem soluções são removidos. Os únicos galhos parciais que são mantidos na árvore \mathcal{T} de realizações são os que possuem algum $v_i \in S_G$ como raiz. Portanto, considera-se, apenas, pares de galhos simétricos factíveis, aqueles que possuem suas raízes nos níveis $i - 1$, para cada $v_i \in S_G$.

Depois dessa checagem inicial de existência de solução já determinada, o algoritmo atende ao seu passo básico. Vale lembrar, primeiramente, que se $v_i \in S$, então ambas as posições determinadas pela intersecção das esferas são factíveis e, logo, não são podadas da árvore. Assim, a cada chamada do *SymBP*, a pertinência do vértice atual v_i ao conjunto de simetrias S_G é testada.

Em caso afirmativo, o método calcula as soluções x_i^0 e x_i^1 e anda para o nível seguinte, fazendo $v_i \leftarrow v_{i+1}$ e recursivamente se chama novamente.

Caso contrário, se $v_i \notin S_G$, há que distinguir duas situações excludentes:

(1) Nenhuma solução determinada ainda

- se nenhuma solução foi encontrada até então ($nsols = 0$), então as informações sobre simetrias não podem ser exploradas ainda e a mecânica do BP convencional é aplicada;
- o vetor binário $prev$ deve ser mantido atualizado, a fim de que ele contenha a primeira solução, assim que for encontrada;
- sem perda de generalidade, caso uma solução seja encontrada para o vértice v_i na posição x_i^0 , então a posição x_i^1 deve ser automaticamente descartada;

(2) A primeira solução já determinada

- se a primeira solução foi encontrada, o *SymBP* constrói todas as outras soluções a partir das informações de S_G aplicando as reflexões parciais já mencionadas na Seção 3.2.2;
- Não há mais ramificações a se fazer na árvore (*branchings*), pois $x_i^{\neg prev(v_i)}$ só pode ser factível se $x_i^{prev(v_i)}$ for factível para a solução anterior;
- Também não há mais podas, pois todas as posições a serem geradas são factíveis por simetria, o que implica em uma drástica redução no custo computacional.

A fim de ilustrar o uso prático do SymBP em uma instância DMDGP, segue um exemplo cuja árvore de soluções foi extraída de Fidalgo *et al.* [33].

Exemplo 3.3.1. Considere a instância $G = (V, E, d)$ do DMDGP com onze vértices no total e cujas arestas de poda são

$$E_p = \{\{1, 7\}, \{5, 11\}, \{7, 11\}\} \tag{3.3.1}$$

e X o conjunto de soluções desta instância através do algoritmo Branch & Prune.

Dessa forma, pode-se ver que o conjunto de simetrias associado a esta instância é dado por

$$S_G = \{v_4, v_8\}. \quad (3.3.2)$$

Segue, do Corolário 3.2.2, que

$$|\mathcal{S}| = 2^{|S_G|} = 2^2 = 4, \quad (3.3.3)$$

ou seja, existem quatro soluções factíveis nesse caso.

Além disso, suponha que a árvore T_1 (Fig.3.5) represente a execução do *Branch & Prune* clássico para G . Os caminhos em vermelho representam os que foram descartados devido à infactibilidade dos últimos nós e, consequentemente, mapeam as regiões onde houve retornos.

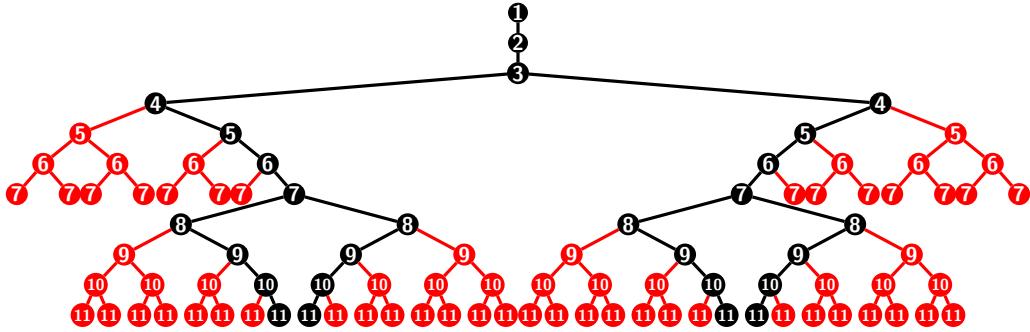


Figura 3.5: Árvore representando as soluções para G pelo BP clássico.

Agora, aplicando o *SymBP* na mesma instância, temos a representação gráfica dada pela árvore T_2 (Fig.3.6).

Pode-se observar que:

- a quantidade de retornos realizada, que de fato é o que gasta tempo no BP, foi menor do que a metade em relação ao BP;
- esses retornos acontecem, apenas, até encontrar-se a primeira solução x_1 , dada por

$$x_1 = (0, 0, 0, 0, 1, 1, 1, 0, 1, 1, 1); \quad (3.3.4)$$

- a segunda solução x_2 é o reflexo da primeira pela simetria de v_8 , ou seja,

$$x_2 = (0, 0, 0, 0, 1, 1, 1, 1, 0, 0, 0); \quad (3.3.5)$$

- finalmente, a terceira x_3 e a quarta x_4 soluções são os reflexos da segunda e primeira soluções, respectivamente, pela simetria de v_4 , isto é,

$$x_3 = (0, 0, 0, 1, 0, 0, 0, 0, 1, 1, 1) \quad \text{e} \quad x_4 = (0, 0, 0, 1, 0, 0, 0, 1, 0, 0, 0). \quad (3.3.6)$$

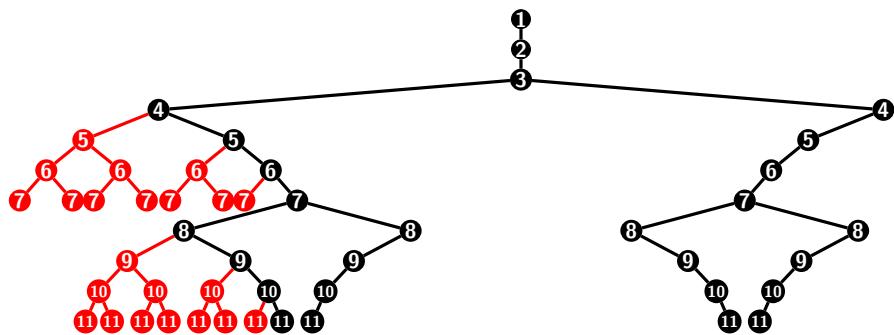


Figura 3.6: Árvore representando as soluções para G pelo SymBP.

Capítulo 4

Uma abordagem Dividir & Conquistar para o DMDGP

Este capítulo ocupa uma posição central neste trabalho, já que contém nossas principais contribuições e resultados no intuito de viabilizar e pavimentar outros caminhos na busca por soluções do DMDGP. Ele será dividido em três seções principais, descritas abaixo.

1) Motivações

Nesta seção, vamos enumerar os métodos e resultados, previamente estudados, que motivaram o presente estudo. Dentre eles, estão as versões para computação em paralelo dos algoritmos BP e SymBP e o estudo das múltiplas árvores de realizações, contendo o primeiro uso bidirecional do BP e a descrição de um método para unir as sub-soluções.

2) Simetrias, Gaps e Quatérnios: dividir-e-conquistar em Geometria de Distâncias

Nesta segunda e principal seção, vamos detalhar toda a base teórica, a mecânica do método *GapSym*, que particiona a instância e computa as realizações, e do método *QStick*, que faz as “colagens” das soluções parciais. É nesta seção que estão nossas contribuições originais.

3) Resultados Computacionais

Por fim, esta seção traz resultados de testes computacionais com os métodos apresentados.

Em breves palavras, nossa contribuição consiste em uma estratégia Dividir & Conquistar para resolver o DMDGP, a qual pode ser naturalmente concebida como uma união de três frentes de trabalho que adotamos:

- (i) **dividir com simetrias** - particionamos o conjunto de vértices da instância utilizando apenas as informações de simetria dadas em sua estrutura,
- (ii) **conquistar com gaps** - utilizamos os chamados *gaps* em cada “pedaço” da partição para decidir em qual direção (positiva ou negativa, como veremos ainda neste capítulo) o BP será aplicado para encontrar as sub-soluções de cada parte e
- (iii) **combinar com Quatérnios** - por fim, empregamos rotações com Quatérnios para unir as sub-soluções factíveis e obter as soluções para o problema original.

4.1 Motivações Preliminares

4.1.1 Uma versão paralela do algoritmo BP

Mucherino *et al.* [69] introduziram uma forma de dividir uma instância DMDGP $G = (V, E, d)$ em sub-instâncias, resolvendo cada uma por meio de processadores independentes (computação em paralelo). O objetivo é obter soluções parciais e uni-las a fim de formarem soluções completas.

Esta abordagem divide uma instância DMDGP como segue. Em primeiro lugar, escolhe-se o número p de sub-instâncias a serem consideradas. Assim, divide-se toda a cadeia de n vértices em uma sequência de blocos com n/p vértices em cada. Caso n não seja divisível por p , então o resto r deles podem formar um novo bloco ou serem acomodados nos p já existentes. Cada um dos blocos de vértices formados neste processo consiste em uma sub-estrutura DMDGP, denotada por $G^i = (V^i, E^i, d)$, pois a nova ordem em cada um ainda é DMDGP.

Todas as arestas de discretização e de poda referentes a vértices de um mesmo bloco estão presentes no mesmo. Assim, o BP utilizará tais informações para computar, em paralelo, as soluções parciais de cada “pedaço” da instância, cada uma em seus respectivos processadores. Isto é, cada sub-instância G^i dará origem a uma árvore T^i que armazena tais soluções parciais.

Em seguida, os p (ou $p + 1$) processadores, dois-a-dois, passam ao processo de comunicação dessas soluções parciais em um sistema de cascata, como se pode ver na Figura 4.1. Nesta figura, temos que os processadores 0 e 1 e os processadores 2 e 3 são combinados respectivamente, gerando dois processadores apenas. Estes, por sua vez, são combinados em um único processador final.

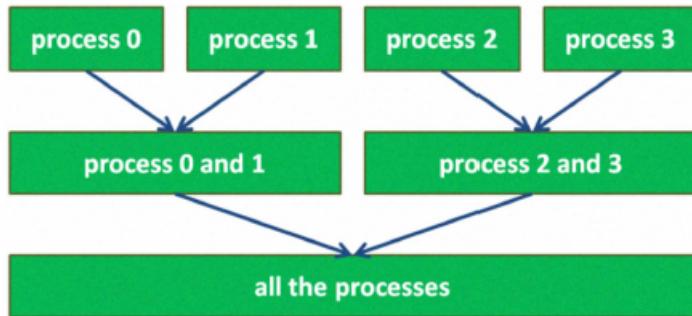


Figura 4.1: A comunicação em cascata entre os processadores. Figura extraída de [69].

É preferível que, durante este processo de cascata, não haja computação alguma, apenas comunicação entre os processadores [69]. Dessa maneira, as junções das soluções parciais são feitas no final de toda a cascata, o que é realizado através de rotações e translações [69].

Além disso, podem existir informações de poda conectando vértices de blocos distintos (e não necessariamente consecutivos). Ou seja, neste caso, ao unirem-se as soluções parciais de cada bloco em soluções completas, a factibilidade dessas também devem ser analisadas em relação a tais informações de poda adicionais, o que é feito posteriormente à comunicação e à junção [69].

4.1.2 Resolvendo o DMDGP por múltiplas árvores de solução

Motivado por esse modelo em paralelo (Seção 4.1.1), Nucci *et al.* [77] propuseram o chamado método de múltiplas árvores de realizações a fim de computar as realizações de instâncias DMDGP.

Este método pode ser resumido como segue: dada uma instância já dividida, utiliza-se o BP para calcular as árvores parciais de realizações (as que armazenam as soluções parciais) e, então, lança mão de uma translação e duas rotações para unir tais árvores duas-a-duas.

Esta seção tem por objetivo dar as principais noções de tal abordagem.

Branch & Prune: uma outra maneira de utilizar

Nesta subseção, vamos analisar, por meio de um exemplo, uma forma diferente de realizar uma instância DMDGP, assunto abordado por Nucci *et al.* em [77].

Para isso, considere a instância $G = (V, E, d)$, com seis vértices, correspondente ao mesmo exemplo dado no Capítulo 2. Vimos que, ao submeter tal instância ao BP, as soluções foram calculadas e armazenadas árvore T , representada pela Figura 2.28. Esta árvore foi construída utilizando o que denominamos de algoritmo “BP clássico”. Neste, a raiz de T corresponde ao vértice v_1 e crescimento da mesma se dá “da esquerda para a direita”, seguindo a ordem usual $v_1 \rightarrow v_6$. Por esta razão, utiliza-se a notação T^+ para representar tal árvore, pois define-se que ela possui crescimento “positivo”.

Os autores, no entanto, instam o leitor a permutar os vértices de G de acordo com a permutação

$$\sigma = (6 \ 5 \ 4 \ 3 \ 2 \ 1). \quad (4.1.1)$$

Logo, obtém-se um novo exemplar $G' = (V', E', d')$ para esta instância cujos vértices são, biunivamente, associados da seguinte maneira

$$\begin{array}{ccc} V & \longrightarrow & v_1 \ v_2 \ v_3 \ v_4 \ v_5 \ v_6 \\ & \uparrow & \uparrow & \uparrow & \uparrow & \uparrow & \uparrow \\ V' & \longrightarrow & v_6 \ v_5 \ v_4 \ v_3 \ v_2 \ v_1 \end{array} . \quad (4.1.2)$$

Dessa maneira, o conjunto de arestas E' é dado por um reordenamento de E , considerando que $\{i, j\} \simeq \{j, i\}$, de modo que o novo subconjunto de arestas de discretização é dado por

$$E = \{\{6, 5\}, \{6, 4\}, \{6, 3\}, \{5, 4\}, \{5, 3\}, \{5, 2\}, \{4, 3\}, \{4, 2\}, \{4, 1\}, \{3, 2\}, \{3, 1\}, \{2, 1\}\}$$

e o conjunto de arestas de poda é dado por

$$F = \{\{6, 2\}\}.$$

Note que G' e G representam o mesmo grafo e, portanto, cada uma das realizações de uma delas é plenamente factível em relação à outra. Inclusive, a quantidade de soluções que o BP encontra para ambas é a mesma.

Desse modo, ao inverter a instância, inverte-se também o sentido da utilização do BP a fim de calcular as realizações. O objetivo principal, portanto, de se inverter uma instância fundamenta-se em buscar uma maneira de se calcular as mesmas realizações, mas com menos retornos (que é o que demanda mais tempo de computação, como já vimos no Capítulo 2).

Além disso, a fim de facilitar a localização das distâncias existentes, considere a seguinte matriz

$$D_{G'} = \begin{pmatrix} 0 & d_{6,5} & d_{6,4} & d_{6,3} & \textcolor{red}{d_{6,2}} & 0 \\ 0 & 0 & d_{5,4} & d_{5,3} & d_{5,2} & 0 \\ 0 & 0 & 0 & d_{4,3} & d_{4,2} & d_{4,1} \\ 0 & 0 & 0 & 0 & d_{3,2} & d_{3,1} \\ 0 & 0 & 0 & 0 & 0 & d_{2,1} \\ 0 & 0 & 0 & 0 & 0 & 0 \end{pmatrix}. \quad (4.1.3)$$

Assim como no Capítulo 2, descrevemos o uso do BP passo-a-passo para a realização de G' e identificamos a nova árvore de soluções como T^- , indicando a troca no sentido de crescimento.

- (P1) Utilizando os valores $d_{6,5}, d_{5,4} \in \mathcal{D}_d$ e o ângulo de ligação plano $\theta_{6,4}$, define-se as matrizes de torção B_1, B_2 e B_3 dadas em (2.3.9). De posse delas, inicializa-se a árvores de realizações T^- , fixando as posições relativas aos três primeiros vértices v_6, v_5 e v_4 em únicas posições x_6, x_5 e x_4 em \mathbb{R}^3 , respectivamente: uma na origem, uma no eixo das abscissas e a outra no plano abscissa-ordenada, de acordo com as Equações (2.3.10).
- (P2) O processo de ramificações se inicia aqui. Com o valor de $d_{6,3} \in \mathcal{D}_d$, calcula-se $\cos(\omega_{6,3})$. Assim, encontra-se duas posições factíveis $x_3^{(1)}$ e $x_3^{(2)}$ para o quarto vértice da sequência v_3 , as quais são associadas aos valores de $\pm\sqrt{1 - \cos^2(\omega_{6,3})}$ respectivamente. Essa dupla factibilidade deve-se à inexistência de informações de poda associadas a v_3 . Como há que se escolher um dos dois caminhos para continuar a busca (até encontrar uma solução), tome-se a posição $x_3^{(1)}$ para T^- , inicialmente, e avança-se para o próximo vértice.
- (P3) Analogamente, $d_{5,2} \in \mathcal{D}_d$ é utilizada a fim de calcular o valor de $\cos(\omega_{5,2})$. A partir dele, calcula-se duas posições possíveis para v_2 , as quais são denotadas por $x_2^{(1,1)}$ e $x_2^{(1,2)}$ em \mathbb{R}^3 . Para este vértice, sim, existe uma informação de poda $d_{6,2} \in \mathcal{D}_p$. Primeiramente, o caminho determinado por $x_2^{(1,1)}$ é escolhido e testado, cujo teste DDF obtido é

$$\left(\|x_6 - x_2^{(1,1)}\|^2 - d_{6,2}^2 \right)^2 \geq \varepsilon. \quad (4.1.4)$$

Ou seja, esta posição é infactível com o conjunto de distâncias e tal caminho é podado. Retrocede-se a busca para o vértice anterior e escolhe-se a segunda posição $x_2^{(1,2)}$ para seguir. Esta posição, por sua vez, é factível, já que

$$\left(\|x_6 - x_2^{(1,2)}\|^2 - d_{6,2}^2 \right)^2 < \varepsilon. \quad (4.1.5)$$

- (P4) Utilizando a distância $d_{4,1} \in \mathcal{D}_d$, calcula-se $\cos(\omega_{4,1})$. A partir dele, duas posições possíveis para v_1 são definidas em \mathbb{R}^3 e denotadas por $x_1^{(1,2,1)}$ e $x_1^{(1,2,2)}$. Como não há informações de poda para este vértice, então ambas são factíveis. Escolhendo a primeira delas, encontra-se a primeira solução $x_1 \in \mathcal{S}$, denotada por

$$x_1 = (0, 1, 0, 0, 0, 0). \quad (4.1.6)$$

Esta solução deve ser lida de trás-para-frente, já que T^- tem crescimento negativo. Retrocede-se, em seguida, a busca para o vértice anterior e a outra posição $x_1^{(1,2,2)}$ é escolhida, compondo a segunda solução $x_2 \in \mathcal{S}$, a qual é denotada por

$$x_2 = (1, 1, 0, 0, 0, 0). \quad (4.1.7)$$

- (P5) A busca retrocede até o vértice v_4 e, então, escolhemos a segunda posição $x_3^{(2)}$ para o vértice v_3 (quarto nível), explorando a segunda metade de T^- .
- (P6) Então, utiliza-se o valor de $\cos(\omega_{5,2})$ novamente. A partir dele, calcula-se duas posições possíveis para v_2 , as quais são denotadas por $x_2^{(2,1)}$ e $x_2^{(2,2)}$. Procedendo o teste de poda DDF, obtém-se

$$\left(\|x_6 - x_2^{(2,1)}\|^2 - d_{6,2}^2 \right)^2 < \varepsilon. \quad (4.1.8)$$

Essa posição é factível e a dá-se prosseguimento à busca das outras duas soluções.

- (P7) Com $\cos(\omega_{4,1})$, calcula-se as duas posições factíveis neste caminho para v_1 . Portanto, as soluções remanescentes x_3 e x_4 em \mathcal{S} são encontradas e determinadas por

$$s_3 = (0, 0, 1, 0, 0, 0) \quad \text{e} \quad s_4 = (1, 0, 1, 0, 0, 0). \quad (4.1.9)$$

- (P8) Retrocede-se a busca para o vértice v_3 e escolhe-se a segunda posição $x_2^{(2,2)}$ para seguir com a busca. Esta posição, por sua vez, é infactível já que

$$\left(\|x_6 - x_2^{(2,2)}\|^2 - d_{6,2}^2 \right)^2 \geq \varepsilon, \quad (4.1.10)$$

completando a exploração da árvore T^- .

Ao final do processo, a árvore T^- de imersões de G em \mathbb{R}^3 é obtida com o total de quatro realizações factíveis, a qual é representada na Figura 4.2.

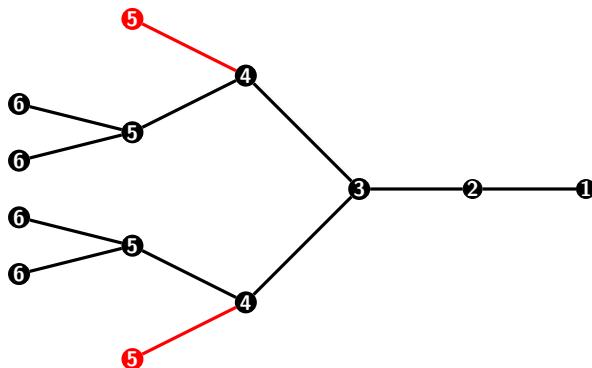


Figura 4.2: Árvore explorada completamente.

Em ambos os usos do BP, duas árvores são geradas T^+ (Fig. 2.28) e T^- (Fig. 4.2) e, em ambas, quatro soluções foram encontradas. Entretanto, a quantidade de retornos em T^- é menor em relação a T^+ , o que deve diminuir o tempo de computação das soluções. A razão pela qual isto ocorre é simplesmente porque em G' a única informação de poda aparece antes (quinto nível) do que em G (sexto nível), diminuindo o número de caminhos possíveis a serem testados.

Com este exemplo, assim, percebe-se que existem outras maneiras de utilizar o BP para explorar o espaço de busca de soluções de uma mesma instância DMDGP além do “BP clássico”. No caso analisado acima, já se obteve sucesso em diminuir os retrocessos apenas trocando o sentido da busca. Apesar do DMDGP ser **NP-difícil**, quanto mais rápido resolvê-lo melhor, mesmo que seu comportamento assintótico não seja alterado [77].

Portanto, é possível utilizar o BP de maneira ainda mais rápida, o que é vantajoso especialmente na resolução de instâncias muito grandes.

Definições

Vamos manter o apelo molecular com o qual essas idéias são tratadas em [77] a partir daqui. As definições, nomenclatura e notações abaixo foram, em sua maioria, amplamente extraídas de [77]. Entretanto, algumas são definidas por este autor para melhor esclarecer alguns pontos.

Seja M uma molécula com n átomos, representada pelo grafo $G = (V, E, d)$ que possui uma ordem DMDGP < em seus n vértices. Seguindo esta ordem, é possível considerar esta instância como uma sequência de números naturais $1, 2, \dots, n$, os índices dos vértices de V .

Definição 4.1.1 (Nucci *et al.* [77]). Um intervalo $I = [a, b]$ de M consiste na subsequência ordenada dos átomos

$$I = \{a, a+1, \dots, b-1, b\}, \quad \text{com } 1 \leq a \leq b \leq n. \quad (4.1.11)$$

Além disso, o comprimento desse intervalo é calculado por $|I| = b - a$.

Até agora, estudamos as realizações de instâncias completas. Mas, com a definição de subestruturas dadas pelos intervalos, podemos restringir esta definição a eles, como segue.

Definição 4.1.2 (Nucci *et al.* [77]). Uma realização do intervalo $I = [a, b]$ de M é uma aplicação

$$\begin{aligned} R_{a,b} : [a, b] &\longrightarrow \mathbb{R}^3 \\ u &\longmapsto R_{a,b}(u) . \end{aligned} \quad (4.1.12)$$

Esta é dita factível se satisfizer completamente as restrições

$$\|R_{a,b}(i) - R_{a,b}(j)\| = d_{ij}, \quad \forall \{i, j\} \subset E \cap I. \quad (4.1.13)$$

Caso ela viole tão somente uma dessas restrições, então é dita ser infactível.

Definição 4.1.3 (Nucci *et al.* [77]). Uma realização $R_{a,b}$ é completa se $[a, b] = [1, n]$. Caso contrário, dizemos que a realização $R_{a,b}$ é parcial.

Essas realizações parciais no espaço Euclidiano serão de grande importância nesta seção, já que são a base para o método das múltiplas árvores de realizações.

Além deste relaxamento no conceito de realização, é possível flexibilizar a maneira como se representa o espaço de busca das soluções. Até então, tais resultados eram armazenados em uma árvore binária na qual se fazia buscas em profundidade a fim de encontrar as soluções. Agora, é possível armazenar soluções parciais nas ditas árvores parciais.

Definição 4.1.4 (Nucci *et al.* [77]). Uma **Árvore de Realizações** $T_{a,b}$, relativa ao intervalo $[a, b]$ de uma instância M do DMDGP, satisfaz as seguintes duas propriedades:

- (i) Cada nível k de $T_{a,b}$ corresponde a um átomo do intervalo $[a, b]$;
- (ii) Cada nó pertencente ao nível k está associado a um vetor de coordenadas em \mathbb{R}^3 , sendo esta possível posição para o átomo k .

Formaliza-se, também, o crescimento bidirecional, apontado anteriormente, de árvores parciais de realizações.

Definição 4.1.5 (Nucci *et al.* [77]). $T_{a,b}^+$ denota a direção de crescimento positivo da árvore (da direita para a esquerda) e $T_{a,b}^-$ denota a direção de negativa (da esquerda para a direita).

Segundo Nucci *et al.* [77], a idéia de trabalhar com realizações parciais foi motivada, primeiramente, por algumas heurísticas exploradas pelo próprio autor e uma conjectura em [76] e, também, pela iniciativa de se paralelizar a atuação do BP, aplicando-o a pedaços menores das instâncias, abordagem tal que foi apresentada por Mucherino *et al.* em [69].

União de duas árvores de realizações parciais

Depois de dividir uma instância DMDGP em intervalos (Mucherino *et al.* [68]) e construir uma árvore de realizações parciais para cada intervalo (Nucci et al. [77]), como descrito acima, é necessário unir as realizações parciais de intervalos subsequentes em pares até que se construa realizações completas. Depois disso, deve-se analisar a factibilidade de cada solução em relação às restrições de distâncias, pois não há nenhuma garantia teórica de que a união das realizações parciais factíveis componham necessariamente realizações completas factíveis.

De fato, se fizermos uma decomposição do conjunto de vértices V em $V_1 \cup \dots \cup V_k$ e definirmos o conjunto de arestas para cada subgrafo como

$$E_i = \{\{v_s, v_t\} : v_s, v_t \in V_i\}, \quad \text{para } i = 1, \dots, k, \quad (4.1.14)$$

então pode ser que existam arestas $\{v_j, v_k\} \in E$ de modo que $v_j \in V_i$ e $v_k \in V_{i+1}$, mas $\{v_j, v_k\} \notin E_i \cup E_{i+1}$.

No restante dessa seção, discutimos a proposta de Nucci *et al.* [77] para unir duas realizações parciais, bem como as justificativas teóricas necessárias para tal, sob as mesmas hipóteses utilizadas até então. Depois disso, enunciaremos os critérios elaborados no intuito de unir duas árvores de realizações parciais.

Nesta abordagem, algumas hipóteses sobre as realizações - a serem unidas - devem ser consideradas a fim de ajustá-las para a aplicação de transformações rígidas Euclidianas.

Sejam $[a, x]$ e $[b, y]$ dois intervalos em M tais que

$$a < b < x < y. \quad (4.1.15)$$

Adicionalmente, suponha que eles possuam, pelo menos, três vértices em comum. Segundo Nucci [77], isso deve implicar que

$$x - b \geq 3, \quad (4.1.16)$$

mas, na verdade, basta que os limites x e b satisfaçam a inequação

$$x - b \geq 2. \quad (4.1.17)$$

Como um exemplo disso, suponha uma molécula com dez vértices $M = [1, 10]$ e vamos dividi-la em dois intervalos com exatamente três vértices em comum, isto é,

$$M = [1, 6] \cup [4, 10]. \quad (4.1.18)$$

De fato, os vértices 4, 5 e 6 estão em ambos os intervalos. Como, neste caso, $x = 6$ e $b = 4$, então

$$x - b = 6 - 4 = 2. \quad (4.1.19)$$

Adicionando um vértice a mais na intersecção, então dividimos esta instância como

$$M = [1, 7] \cup [4, 10]. \quad (4.1.20)$$

Dessa maneira, os quatro vértices 4, 5, 6 e 7 estão na intersecção, ou seja, $x = 7$ e $b = 4$. Logo,

$$x - b = 7 - 4 = 3, \quad (4.1.21)$$

completando o argumento da desigualdade (4.1.17).

Portanto, com objetivo de combinar tais realizações parciais $R_{a,x}$ e $R_{b,y}$ de modo a obter realizações possíveis $R_{a,y}$, cobrindo o conjunto de vértices $[a, y]$, Nucci e colaboradores desenvolveram um esquema lançando mão de uma combinação de três transformações Euclidianas: uma translação plana, uma rotação plana e uma outra rotação em 3D [77].

Para calcular as referidas transformações, considere três vértices i, j e k que pertençam ao conjunto $[a, x] \cap [b, y]$. Além disso, sejam $R_{a,x}$ e $R_{b,y}$ realizações viáveis para os respectivos intervalos.

A proposta, então, tem como base a fixação de uma das realizações acima e a operação com a outra, por meio destas transformações, de modo que os pontos em comum entre elas sejam mantidos em suas posições originais e os vértices excedentes sejam transformados de forma que ambas pertençam ao mesmo sistema de coordenadas referenciais.

Motivados pelas definições em Fidalgo *et al.* [34], neste cenário, fixamos a realização $R_{a,x}$ e transformamos os pontos da realização $R_{b,y}$. Disso, seguem as definições abaixo a fim de facilitar para futuras referências.

Definição 4.1.6 (Fidalgo *et al.* [34]). A estrutura fixa será chamada de *Realização Base* e a estrutura a ser transformada será denominada *Realização Deslizante*, já que esta deve “deslizar” no espaço Euclidiano tridimensional a fim de se unir à realização base.

No teorema abaixo, a palavra “melhor” tem o sentido de minimizar a quantidade de pontos transformados. Como vimos, é suficiente utilizar três pontos na intersecção para realizar a transformação. Assim, como contribuição desta tese, fazemos tal “escolha”.

Proposição 4.1.1. A melhor escolha de três pontos na intersecção entre os dois intervalos consiste nos três últimos pontos da realização base.

Demonstração. De fato, suponha que exista $m > 3$ pontos nesta intersecção. Isto é possível já que, por hipótese, não pode haver menos do que isso. Assim, os dois conjuntos de pontos

$\{R_{a,x}(x-m+1), R_{a,x}(x-m+2), \dots, R_{a,x}(x)\}$ e $\{R_{b,y}(b), R_{b,y}(b+1), \dots, R_{b,y}(b+m-1)\}$ coincidem e consistem em tal intersecção. Agora, estes m pontos da estrutura deslizante já estão fixos na realização base. Então, qualquer ponto da intersecção a ser transformado consistirá, de certa forma, em uma redundância. Logo, a escolha

$$i = x - 2 = b, \quad j = x - 1 = b + 1 \quad \text{e} \quad k = x = b + 2$$

contempla a hipótese de selecionar três pontos neste conjunto e ainda minimiza a quantidade de pontos transformados redundantes. \square

Como $R_{a,x}$ é fixa, $R_{a,y}$ herda as coordenadas dos pontos de forma original [77]

$$R_{a,y}(t) = R_{a,x}(t), \quad t \in [a, x].$$

A fim de cobrir todo este intervalo, basta determinar posições para os átomos em $[x+1, y]$, o que será feito a partir de uma translação e duas rotações, como proposto por Nucci *et al.* Para operar tais transformações, considere a realização auxiliar $R'_{b,y}$, relativa a $[b, y]$. Deseja-se que, ao final do processo, se obtenha

$$R_{a,y}(t) = R'_{b,y}(t), \quad t \in [x+1, y].$$

Inicialmente, considere $R'_{b,y}$ como cópia de $R_{b,y}$. Assim, aplica-se uma sequência de três transformações ortogonais a fim de que as Equações (4.1.22) abaixo sejam satisfeitas

$$R'_{b,y}(i) = R_{a,x}(i), \quad R'_{b,y}(j) = R_{a,x}(j) \quad \text{e} \quad R'_{b,y}(k) = R_{a,x}(k) \quad (4.1.22)$$

Neste trabalho, compilamos estas transformações como a série de proposições que seguem. Alguma nomenclatura e detalhes foram adicionados para melhor elucidação dos resultados.

Proposição 4.1.2 (Translação). A primeira relação de 4.1.22 é satisfeita aplicando uma translação na estrutura $R'_{b,y}$.

Demonstração. Com efeito, considere o vetor

$$\mathbf{v} = R_{a,x}(i) - R'_{b,y}(i). \quad (4.1.23)$$

Aplicando uma translação por este vetor na estrutura auxiliar, ou seja, fazendo

$$R'_{b,y}(t) \leftarrow R'_{b,y}(t) + \mathbf{v}, \quad t \in [b, y],$$

tal relação é satisfeita. De fato, temos que

$$R'_{b,y}(i) + \mathbf{v} = R'_{b,y}(i) + R_{a,x}(i) - R'_{b,y}(i) = R_{a,x}(i).$$

□

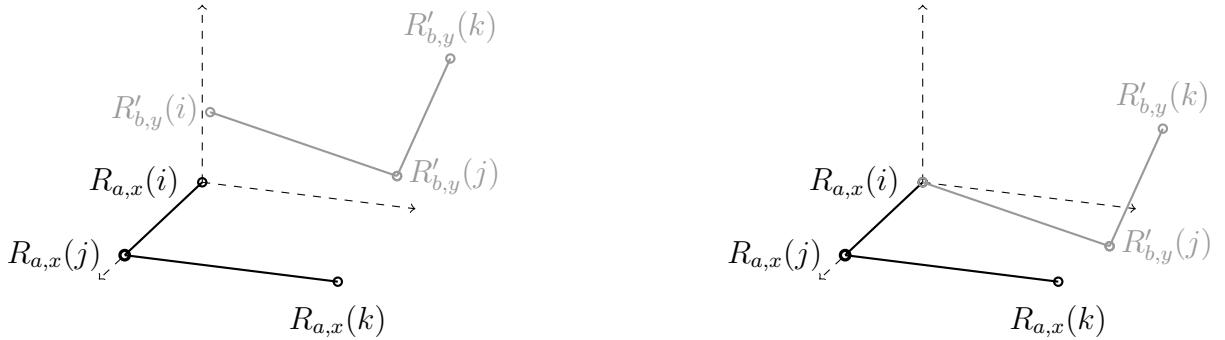


Figura 4.3: Primeira transformação Euclideana: uma translação na estrutura deslizante.

Proposição 4.1.3 (Primeira Rotação). A segunda relação de 4.1.22 é atingida ao aplicar uma rotação na estrutura $R'_{b,y}$, já transladada.

Demonstração. Com a estrutura $R'_{b,y}$, considere os vetores

$$L_{ij} = R_{a,x}(j) - R_{a,x}(i) \quad \text{e} \quad L'_{ij} = R'_{b,y}(j) - R'_{b,y}(i)$$

que possuem um ponto em comum, a saber, a mesma extremidade $O = R_{a,x}(i) = R'_{b,y}(i)$. Considerando O como vértice, existe um ângulo θ entre eles que é dado por

$$\theta = \arccos \left(\frac{\|L_{ij}\|^2 + \|L'_{ij}\|^2 - \|L_{ij} - L'_{ij}\|^2}{2 \|L_{ij}\| \|L'_{ij}\|} \right),$$

usando a Lei dos Cossenos usual no triângulo vetorial formado por L_{ij} , L'_{ij} e $L_{ij} - L'_{ij}$. Assim, para que as duas posições para o vértice j sejam coincidentes, as duas extremidades dos vetores acima devem coincidir. Para tanto, tendo em vista o caráter rotacional da Regra da Mão-Direita e a fim de não perder o que já se ganhou com a translação, basta aplicar uma rotação em $R'_{b,y}$ em termos de θ e cujo eixo é gerado pelo vetor unitário

$$\mathbf{n} = \frac{L'_{ij} \times L_{ij}}{\|L'_{ij} \times L_{ij}\|}.$$

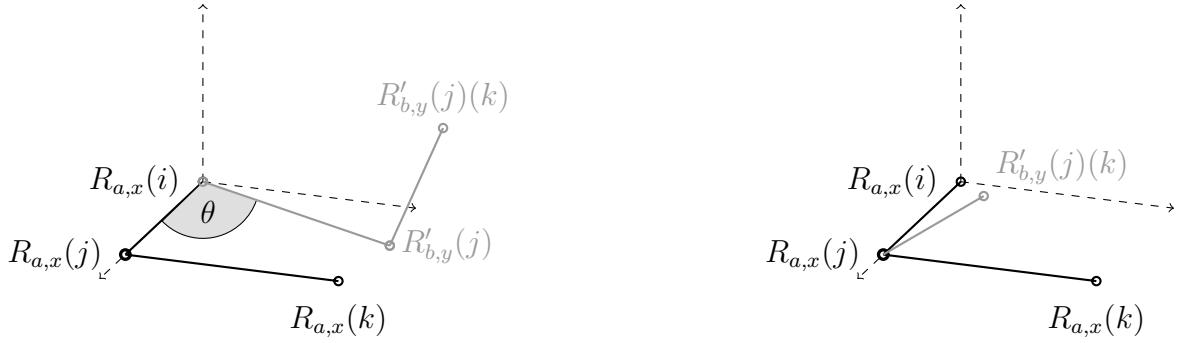


Figura 4.4: Segunda transformação: uma rotação plana em termos de θ na estrutura deslizante.

□

Proposição 4.1.4 (Segunda Rotação). A terceira relação de 4.1.22 é satisfeita ao aplicar uma segunda rotação na estrutura $R'_{b,y}$, já transladada e rotacionada uma vez.

Demonstração. A intenção é aplicar uma rotação que não perca o que foi estabelecido com nenhuma das duas transformações anteriores, já que determinados movimentos de rotação das arestas consideradas, como no item anterior, pode violar as condições que queremos satisfazer em conjunto. Vamos determinar esta rotação construtivamente. Considere a reta que passa pelo vetor L_{ij} como direção para o eixo de rotação, gerado por um determinado vetor \mathbf{m} , e os vetores

$$L_{jk} = R_{a,x}(k) - R_{a,x}(j) \quad \text{e} \quad L'_{jk} = R'_{b,y}(k) - R'_{b,y}(j).$$

Pela estrutura da instância do DMDGP, há a igualdade nos comprimentos $\|L_{jk}\| = \|L'_{jk}\|$, além de que o ângulo formado pelos vetores L_{ij} e L_{jk} é congruente ao ângulo, agora, formado entre L_{ij} e L'_{jk} . Esta simetria garante que as extremidades $R_{a,x}(k)$ e $R'_{b,y}(k)$ estão em um mesmo círculo contido no hiperplano ortogonal à reta que contém L_{ij} .

Sejam \mathbb{P} este plano e P_k e P'_k as projeções de L_{jk} e L'_{jk} , respectivamente, em \mathbb{P} . Logo,

$$P_k = M L_{jk} \quad \text{e} \quad P'_k = M L'_{jk},$$

onde M é a matriz de projeção no subespaço ortogonal a $\langle L_{ij} \rangle$, ou seja, $M = I - L_{ij} L_{ij}^T$.

Assim, para que $R_{a,x}(k) = R'_{b,y}(k)$, a estrutura $R'_{b,y}$ deve ser rotacionada ao redor de \mathbf{m} por um ângulo φ tal que

$$\varphi = \arccos \left(\frac{\|P_k\|^2 + \|P'_k\|^2 - \|P_k - P'_k\|^2}{2 \|P_k\| \|P'_k\|} \right),$$

o qual também é calculado a partir da aplicação da Lei dos Cossenos. □

Observação 4.1.1. Nos cabe salientar uma errata em Nucci et al. [77]. Na página 185, a matriz M é definida como $M = L_j L_j^T$ que, na verdade, não fornece a projeção ortogonal que de fato se deseja. Na Proposição 4.1.4 acima, isto é corrigido. □



Figura 4.5: Terceira transformação: uma rotação espacial em termos do ângulo φ .

Em um sentido prático (computacional), não há menção em Nucci *et al.* [77] sobre qual o método usado nas implementações para se realizar rotações em \mathbb{R}^3 . Por isso, assumimos que tal transformação é codificada em matrizes de rotação. Mais detalhes e características dessas matrizes de rotação são discutidas no Apêndice B.

Como fundir duas árvores de realizações

Em resumo, os seguintes itens já foram abordados nesta seção até agora:

- o crescimento de árvores de realizações de instância do DMDGP pode ser feito em duas direções, por isso, diz-se crescimento bidirecional;
- realizações de intervalos distintos e subsequentes que partilhem, ao menos, três vértices (átomos) podem ser conectadas e formar uma realização para um intervalo maior a partir da utilização de transformações Euclidianas (movimentos rígidos).

Com isso, Nucci *et al.* [77] une estes dois processos em um método para gerar árvores de realizações para instâncias do DMDGP através da “colagem” literal de árvores parciais de par-em-par. É como um prosseguimento de uma estratégia dividir-e-conquistar. Diz-se, dessa forma, resolver o DMDGP através de **múltiplas árvores de realizações**.

Neste método, é possível unir duas árvores parciais, mesmo que cresçam em direções opostas, a partir da junção de cada par possível de realizações. O que se faz é tão somente colocar uma no mesmo sistema referencial de coordenadas da outra.

O propósito do algoritmo é combinar duas árvores $T_{a,b}$ e $T_{c,y}$ de intervalos subsequentes e sobrepostos de modo a gerar uma única árvore de realizações a fim de cobrir $[a, y]$.

Contemplando todas as possibilidades, dada a bidirecionalidade do crescimento das árvores, esta união pode ser feita segundo três tipos de combinações diferentes:

- (C1) **União Raiz - Raiz:** Sejam $T_{a,b}^-$ e $T_{c,y}^+$ duas árvores tais que os três respectivos vértices de inicialização de ambas coincidem (Fig. 4.6), ou seja, vale a associação

$$b - 2 \leftrightarrow c, \quad b - 2 \leftrightarrow c \quad \text{e} \quad b - 2 \leftrightarrow c. \quad (4.1.24)$$



Figura 4.6: União Raiz - Raiz: árvores $T_{1,6}^-$, à esquerda, e $T_{4,8}^+$, à direita.

- (C2) **União Folha - Raiz:** Sejam $T_{a,b}^+$ e $T_{c,y}^+$ duas árvores tais que os três vértices finais da primeira coincidem com os três vértices da inicialização da segunda árvore (Fig. 4.7), também, de acordo com a associação (4.1.24).



Figura 4.7: União Folha - Raiz: árvores $T_{1,6}^+$, à esquerda, e $T_{4,8}^+$, à direita.

- (C3) **União Folha - Folha:** Sejam $T_{a,b}^+$ e $T_{c,y}^-$ duas árvores que crescem em direções opostas de modo que suas folhas se sobrepõem e suas raízes se distanciam (Fig. 4.8). Além disso, sem perder a generalidade, suponha que $b \leftrightarrow c$, ou seja, existe apenas um vértice em comum. Para que satisfaçam a regra 4.1.17, uma dessas árvores deve ainda crescer por mais dois níveis, o que implica nas três possibilidades abaixo:

- a árvore de crescimento positivo sobe mais dois níveis em direção à outra e, desse modo, unem-se as árvores $T_{a,x+2}^+$ e $T_{b,y}^-$;
- a árvore de crescimento negativo sobe mais dois níveis em relação à outra e, assim, unem-se as árvores $T_{a,x}^+$ e $T_{b-2,y}^-$ e, por fim,
- a árvore de crescimento positivo sobe mais um nível em direção à outra árvore e vice-versa e, logo, as árvores $T_{a,x+1}^+$ e $T_{b-1,y}^-$ são unidas pelo processo dado anteriormente.

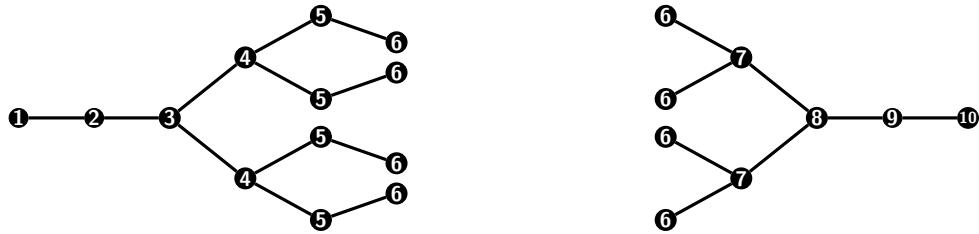


Figura 4.8: União Folha - Folha: árvores $T_{1,6}^+$, à esquerda, e $T_{6,10}^-$, à direita.

Nos dois primeiros casos, descritos acima, fizemos algumas adaptações no intuito de satisfazer as condições de intersecção, não sendo necessário permitir que uma das árvores tenha de crescer mais, após esta análise.

Já o último caso foi preservado como está descrito em [77], onde as árvores precisam subir, no total, mais dois níveis, de acordo com as regras de (i) → (iii). Assim, a árvore que estiver prestes a sofrer mais podas, caso o BP continue a ser executado, deve ter maior prioridade de crescimento, já que terá uma largura menor no final do processo. Dependendo do tamanho das árvores, subir um ou dois níveis aumenta, consideravelmente, a quantidade de realizações e, consequentemente, o número de operações aritméticas. Essa escolha, portanto, possibilita a diminuição do número de realizações factíveis e, logo, haverá um número menor de duplas de estruturas a serem unidas.

O Algoritmo 3, extraído de [77], explora essas idéias e traz um *outline* para a implementação da união de árvores de soluções. A única modificação que fizemos neste algoritmo foi inserir mais um dado de entrada, o qual é o limitante ε para os testes DDF de poda feitos *a posteriori*.

Nele, inicialmente, cada realização da árvore $T_{a,x}$ é combinada com cada uma das realizações de $T_{b,y}$, respectivamente, de acordo com as regras dadas anteriormente, gerando novas estruturas. Note que o número dessas novas estruturas coincide com o produto dos números de estruturas de cada uma das árvores. Depois, é checada a factibilidade direta (DDF) de acordo com o limitante ε , dado como entrada. Por fim, as estruturas que forem factíveis devem compor uma nova árvore $L_{a,y}$, devolvida como saída desse algoritmo.

Observação 4.1.2. Através de um conjunto de experimentos, constatamos que as “colagens” podem demandar um grande esforço computacional. Assim, unir duas estruturas de modo a obter uma configuração infactível é desperdiçar tempo de cálculo. Portanto, um dos nossos objetivos aqui, é encontrar divisões na estrutura que minimizem esse descarte de realizações. □

Heurística para controle do crescimento das árvores

Como visto acima, as uniões dos tipos (C1) ou (C2) só podem ser feitas de um modo único. Já para o caso (C3), há uma multiplicidade de maneiras. Com o objetivo de minimizar o tempo de execução do Algoritmo 3 e, também, o número de nós na árvore resultante, Nucci et al. fornecem uma heurística que controla o crescimento das árvores [77].

Nesta abordagem, a prioridade de crescimento é dada àquela árvore que possui menos folhas ao atingir o nível i . De fato, possivelmente ela crescerá menos do que as outras no nível seguinte, onde ela deve passar por uma nova avaliação como esta em relação à outra, e assim sucessivamente.

Tal método se enquadra na classe de algoritmos gulosos, não considerando a possibilidade de que a outra árvore cresça primeiro. O Algoritmo 5 ([77], p. 187) traz um roteiro para ele.

Como um exemplo ilustrativo: é mais conveniente permitir que uma árvore que passará por uma poda mais vasta em poucos passos tenha prioridade de crescimento para o próximo nível [77].

4.1.3 Uma versão paralela do *SymBP*: uma possibilidade

Com o objetivo de unir eficiência computacional com custo reduzido, Mucherino et al. [68] mencionam a possibilidade de utilizar o *SymBP* em uma versão paralela, semelhantemente ao que

Algoritmo 3 *Merge Trees*

```

1: MergeTrees( $T_{a,x}, T_{b,y}, \varepsilon$ )
2: Crie uma realização vazia  $L_{a,y} = \{\}$ 
3: para (cada realização  $R_{a,x}$  em  $T_{a,x}$ ) faça
4:   para (cada realização  $R_{b,y}$  em  $T_{b,y}$ ) faça
5:     alinhe a estrutura deslizante  $R_{b,y}$  com a estrutura fixa  $R_{a,x}$ 
6:     fail = false
7:     para (cada  $(i,j) \in E_p$  tal que  $a \leq i \leq x$  e  $b \leq j \leq y$ ) faça
8:       se  $(|R_{b,y}(j) - R_{a,x}(i)| - \varepsilon) > d_{i,j}$  então
9:         fail = true
10:        fim se
11:      fim para
12:      se (not fail) então
13:        crie uma realização  $R_{a,y}$  tal que
14:        se ( $i \in [a, x]$ ) então
15:           $R_{a,y}(i) = R_{a,x}(i)$ 
16:        senão
17:           $R_{a,y}(i) = R_{b,y}(i)$ 
18:        fim se
19:      fim se
20:    fim para
21:  fim para
22: return  $L_{a,y}$ 

```

foi feito com o BP (Seção 4.1.1). Isto porque o *SymBP* é intrinsecamente paralelo [68]. Abaixo, segue um esboço desta idéia, a qual deveremos implementar de modo mais abrangente em nossa proposta.

Sejam $G = (V, E, d)$ uma instância DMDGP e p o número de processadores em um computador. Assim:

- Divida a instância G em p sub-instâncias G_1, \dots, G_p de modo que

$$G = G_1 \cup \dots \cup G_p.$$

- Aplique o algoritmo *SymBP* em cada sub-instância G_i : encontre uma solução para cada parte e aplique as regras de simetria para encontrar as outras.
- Faça a comutação entre os processadores das soluções parciais e combine-as a fim de compor as soluções originais para a instância G .

Essa paralelização, portencialmente, melhora ainda mais a velocidade de processamento do *SymBP*, permitindo uma redução do tempo computacional ainda mais drasticamente.

4.2 Dividir-e-Conquistar com Quaternios em Geometria de Distâncias

Ambos os trabalhos, citados acima, lidam com uma estratégia DC para resolver o DMDGP, mas nenhum deles referem-se aos seus três estágios, como abordado no Apêndice D:

- Nucci et. al [77] não formalizaram um método para dividir instâncias do DMDGP, apenas forneceram uma estratégia para unir duas árvores parciais de realizações subsequentes por meio de movimentos rígidos, obtendo-se realizações para as seções maiores de tal instância;
- Mucherino et. al [69] trabalharam para dividir uma instância em sub-instâncias de comprimento fixo, paralelizando os processos de execução do BP nas partes, sem levar em conta que podem haver divisões que ainda sejam mais eficientes e sem mencionar o modo de realizar tais junções dessas partes.

Nossa contribuição, portanto, vem no sentido de fornecer uma abordagem Dividir-e-Conquistar de modo completo, buscando extraír, dinamicamente, uma eficiência característica da própria estrutura de cada instância.

No caminho até estabelecer plenamente os três estágios eficientemente, foi necessário enfrentar alguns desafios, os quais são elencados abaixo e nortearam nossa pesquisa:

- (i) dividir a instância de modo a minimizar (se possível, zerar) a quantidade de informações de poda que fiquem para uma teste de poda posterior - tal desafio é enfrentado na Seção 4.2.1;
- (ii) determinar as realizações dos subgrafos associados através do BP utilizando o menor número de *backtrackings* possível, já que estes são os principais responsáveis por tornar o BP mais lento, como já vimos no Capítulo 2 - a Seção 4.2.2 contempla este problema;
- (iii) fundir as soluções parciais utilizando tempo computacional reduzido - a Seção 4.2.3 traz uma proposta para este desafio.

Em toda essa seção, seja $G = (V, E, d)$ uma instância DMDGP com $n = |V|$ vértices.

4.2.1 Dividindo com Simetrias

Nossa proposta, nesta seção, é de partitionar o grafo G em uma sequência G_1, G_2, \dots, G_k de subgrafos $G_i = (V_i, E_i, d)$, com a mesma ordem DMDGP em cada V_i , de modo que

$$G = G_1 \cup \dots \cup G_k. \quad (4.2.1)$$

Desse modo, não devem existir informações de poda envolvendo os vértices de uma determinada parte para excluir galhos infactíveis em outra parte na aplicação do BP.

Grosso modo, a intenção é que cada parte retenha as infactibilidades em si sem passar adiante em uma abordagem dividir-e-conquistar ou até em um provável método envolvendo computação em paralelo.

Para isso, utilizaremos as simetrias existentes na própria estrutura DMDGP (Capítulo 3). As idéias aqui apresentadas tiveram sua gênese nos estudos que compõem o resumo estendido ***Exploiting symmetries in a divide-and-conquer approach for solving the DMDGP*** [33], e foram apresentadas por este autor no *workshop Many Faces of Distances*, realizado na Universidade Estadual de Campinas (UNICAMP), em meados de Outubro de 2014.

Para uma pesquisa mais aprofundada, um *survey* sobre Partição de Grafos e seus avanços recentes pode ser encontrado em um artigo de Buluç *et al.*, publicado em 2013 no ArXiv [10].

Inicialmente, devemos revisitar a definição de simetrias. Segue, da Equação (3.2.13) e do Teorema 3.2.3, que um vértice $v_i \in V$ é de simetria se, e somente se, $\nexists \{v_j, v_k\} \in E$ tal que

$$j + 3 < v \leq k. \quad (4.2.2)$$

Isso significa que v é um vértice de simetria se, e somente se, não existem arestas da forma $\{v_j, v_k\}$ em E tais que $j \in \{1, \dots, i-4\}$ e $k \in \{i, \dots, n\}$. Essa identificação é mais fácil de se fazer ao olhar para os elementos da matriz de distâncias D_G , associada à instância para a qual $v_i \in S_G$:

$$D_G = \begin{bmatrix} 0 & d_{1,2} & d_{1,3} & d_{1,4} & \dots & d_{1,i-2} & d_{1,i-1} & 0 & 0 & \dots & 0 \\ 0 & d_{2,3} & d_{2,4} & \dots & d_{2,i-2} & d_{2,i-1} & 0 & 0 & \dots & 0 \\ 0 & d_{3,4} & \vdots & 0 \\ 0 & \vdots & d_{i-4,i-2} & d_{i-4,i-1} & 0 & 0 & 0 & 0 & \dots & 0 \\ & \ddots & d_{i-3,i-2} & d_{i-3,i-1} & d_{i-3,i} & d_{i-3,i+1} & \dots & d_{i-3,n} \\ & & 0 & d_{i-2,i-1} & d_{i-2,i} & d_{i-2,i+1} & \dots & d_{i-2,n} \\ & & 0 & d_{i-1,i} & d_{i-1,i+1} & \dots & d_{i-1,n} \\ & & & 0 & d_{i,i+1} & \dots & d_{i,n} \\ & & & & 0 & \dots & d_{i+1,n} \\ & & & & & \ddots & \vdots \\ & & & & & & 0 \end{bmatrix} \quad (4.2.3)$$

Como se pode observar em D_G , a submatriz $D_G(1 : i-4, i : n)$ (contornada pelo retângulo **vermelho**) deve ser identicamente nula, já que suas entradas correspondem, biunivocamente, à arestas que faltam em E associadas às restrições (4.2.2).

Dessa maneira, pode-se partitionar V em dois conjuntos disjuntos, a partir de v_i ,

$$V_1 = \{v_1, \dots, v_{i-1}\} \quad \text{e} \quad V_2 = \{v_i, \dots, v_n\}$$

tais que $V = V_1 \cup V_2$. Entretanto, se os conjuntos de arestas E_1 e E_2 , associados a cada um deles, forem definidos como

$$E_i = \{\{v_k, v_j\} : v_k, v_j \in V_i\}, \quad (4.2.4)$$

não é possível dizer que $G = G_1 \cup G_2$, já que $E_1 \cup E_2 \neq E$. Como um exemplo disso, suponha que a aresta de poda $\{v_{i-3}, v_{i+1}\}$ esteja presente em E . Se essa divisão for feita, então tal aresta não pertence nem a E_1 e nem a E_2 , de forma a não caminhar na direção de que desejamos desde início, mesmo que indique um caminho promissor para a utilização desses dados de simetria.

Logo, a fim de particionarmos G em duas sub-instâncias DMDGP, será preciso considerar uma partição em subgrafos sequencialmente dois-a-dois sobrepostos (para um melhor entendimento, veja o artigo de Chen e Hu [Chen:06] que apresenta um algoritmo lidando com isso). Ou seja, os subgrafos G_1, \dots, G_k devem ser tais que

$$G_i \cap G_{i+1} \neq \emptyset. \quad (4.2.5)$$

Mais que isso: é preciso dar as garantias mínimas de intersecção para as uniões das realizações parciais. Então, vamos considerar, no exemplo, uma partição tal que $|G_1 \cap G_2| = 3$, onde

$$G_1 \cap G_2 = \{v_{i-3}, v_{i-2}, v_{i-1}\}. \quad (4.2.6)$$

Ou seja, define-se os subgrafos $G_1 = (V_1, E_1, d)$ e $G_2 = (V_2, E_2, d)$ tais que os dois conjuntos de vértices são dados por

$$V_1 = \{v_1, \dots, v_{i-1}\} \quad \text{e} \quad V_2 = \{v_{i-3}, \dots, v_n\} \quad (4.2.7)$$

e, os dois conjuntos de arestas, dados por

$$E_1 = \{\{v_k, v_j\} : v_k, v_j \in V_1\} \quad \text{e} \quad E_2 = \{\{v_k, v_j\} : v_k, v_j \in V_2\}. \quad (4.2.8)$$

Dessa forma é possível agrupar todas as arestas que faltavam na partição, além de possuir três vértices em intersecção, garantindo a aplicação das três transformações (uma translação e duas rotações) para unir as realizações (como vimos no trabalho de Nucci *et al.*). Veja que, neste caso, se a aresta $\{v_{i-3}, v_{i+1}\}$ estivesse disponível no conjunto E , então ela estaria tão somente em E_2 , dada a partição de E em (4.2.8).

De uma forma mais geral, seja $S_G = \{v_4, v_{s_1}, v_{s_2}, \dots, v_{s_t}\}$ o conjunto com os $t + 1$ vértices de simetria para uma instância G . Então, nossa proposta é particionar G em subgrafos G_1, \dots, G_k (onde $k \leq t$, como veremos a seguir) sobrepostos de maneira que $|G_i \cap G_{i+1}| = 3$, para cada $i = 1, \dots, k - 1$, utilizando as idéias expostas há pouco nesta seção. O objetivo dessa partição é garantir que cada informação de poda $\{a, b\} \in E_p$ pertença a tão somente um único subgrafo G_i e que não exista nenhuma informação de poda conectando dois subgrafos distintos.

Antes disso, é necessário fazer um pré-processamento no conjunto S_G , como veremos a seguir por meio das Observações 4.2.1 e 4.2.2.

Observação 4.2.1. Note que não devemos considerar o vértice v_4 referente ao quarto nível como gerador da primeira parte, mesmo sabendo que $v_4 \in S_G$ sempre. Se assim o fosse, o primeiro e o segundo subgrafos teriam, respectivamente, os seguintes conjuntos de vértices

$$V_1 = \{v_1, v_2, v_3, v_4\} \text{ e } V_2 = \{v_1, v_2, v_3, v_4, v_5, \dots, v_{s_1-1}\},$$

onde v_{s_2} é o segundo vértice de simetria em S_G . Logo, as realizações (através do BP) de G_1 deveriam ser desprezadas, pois já pertencem - em sua totalidade - às realizações de G_2 .

□

Observação 4.2.2. Note, adicionalmente, que vértices de simetria consecutivos devem pertencer a um mesmo subgrafo. Caso contrário, seria necessário incluir vários subgrafos de quatro vértices que já estão inteiramente contidos na união do subgrafo anterior com o subgrafo, aumentando o número de subgrafos sem efeito prático nenhum.

Por exemplo, sejam $v_i, v_{i+1} \in S_G$ dois vértices de simetria consecutivos. Pela definição que demos, devem existir uma sequência de subgrafos sobrepostos G_r, G_{r+1}, G_{r+2} tais que

$$G_r = \{\dots, v_{i-3}, v_{i-2}, v_{i-1}\}, \quad G_{r+1} = \{v_{i-3}, v_{i-2}, v_{i-1}, v_i\} \quad \text{e} \quad G_{r+2} = \{v_{i-2}, v_{i-1}, v_i, v_{i+1}\}.$$

Neste caso, se considerarmos apenas dois subgrafos da forma

$$G_r = \{\dots, v_{i-3}, v_{i-2}, v_{i-1}\} \quad \text{e} \quad G_{r+1} = \{v_{i-3}, v_{i-2}, v_{i-1}, v_i, v_{i+1}\}$$

continuamos cobrindo esses dois vértices de simetria em subgrafos “desconexos” (não existem arestas de poda conectando os dois) e com três vértices de intersecção. \square

Dessa maneira, propomos que o conjunto S_G passe por um pré-processamento dando origem ao conjunto P_G , gerador da partição que desejamos, da seguinte maneira:

- Descarte o quarto vértice v_4 ;
- Além disso, caso o último vértice v_n pertença a S_G , descarte-o também.
- Tome o próximo vértice v_{s_1} e o inclua como v_{p_1} em P_G .
- Avalie quais dos próximos vértices formam uma sequência de vértices consecutivos. Caso isso ocorra, descarte todos estes vértices e tome o próximo dos vértices de simetria como v_{p_2} e o inclua em P_G . É por essa razão que $k \leq t$, pois há a possibilidade de alguns vértices de simetria não serem geradores de nenhuma das partes, mas serem apenas elementos. Esses vértices descartados são apenas elementos de uma partição, mas não são geradores.
- Caso contrário, tome v_{s_2} como v_{p_2} e o inclua em P_G .
- Agindo dessa maneira, esgote todas as possibilidades de obter os geradores de partes.
- Se o último elemento de S_G ainda não foi descartado, então o inclua como o último elemento v_{p_s} de P_G .

Assim, temos o subconjunto de S_G ao qual chamaremos de *Conjunto Gerador da Partição por Simetrias*

$$P_G = \{v_{p_1}, \dots, v_{p_s}\} \subseteq S_G \subset V. \quad (4.2.9)$$

De posse desse conjunto, vamos propor uma partição de G em um uma sequência de subgrafos G_1, G_2, \dots, G_k , com $k = s + 1$, tal que

$$G = G_1 \cup \dots \cup G_k \quad \text{e} \quad |G_i \cap G_{i+1}| = 3 \quad (4.2.10)$$

de forma que cada vértice gerador (de simetria) em P_G corresponderá ao quarto vértice de um dos processos, exceto do primeiro cujo quarto vértice será v_4 . Isto será feito da seguinte maneira:

- (P1) Tome o primeiro vértice gerador $v_{p_1} \in P_G$. Considere, então, o subconjunto com $p_1 - 1$ vértices

$$V_1 = \{v_1, v_2, v_3, v_4, \dots, v_{p_1-1}\}. \quad (4.2.11)$$

A partir dele, tome o subconjunto de arestas $E_1 = \{\{u, w\} \in E : u, w \in V_1\}$. Estes dois conjuntos formam o subgrafo ponderado $G_1 = (V_1, E_1, d)$.

- (P2) Agora, considere o segundo vértice gerador $v_{p_2} \in P_G$. A partir dele e do anterior, construa o seguinte subconjunto com $p_2 - p_1 + 3$ vértices

$$V_2 = \{v_{p_1-3}, v_{p_1-2}, v_{p_1-1}, v_{p_1}, \dots, v_{p_2-1}\}.$$

De semelhante modo, a partir dele, tome o subconjunto de arestas $E_2 = \{\{u, w\} \in E : u, w \in V_2\}$. Estes dois conjuntos formam o subgrafo ponderado $G_2 = (V_2, E_2, d)$.

- (P3) Como um passo geral, suponha que tenhamos utilizado os vértices $v_{p_1}, v_{p_2}, \dots, v_{p_{i-1}}$ e construído os subgrafos G_1, G_2, \dots, G_{i-1} do modo designado em (P1) e (P2). Assim, o próximo vértice gerador da sequência é v_{p_i} , o qual dá origem ao subconjunto com $p_i - p_{i-1} + 3$ vértices

$$V_i = \{v_{p_{i-1}-3}, v_{p_{i-1}-2}, v_{p_{i-1}-1}, v_{p_{i-1}}, \dots, v_{p_i-1}\}.$$

A partir dele, tome o subconjunto de arestas $E_i = \{\{u, w\} \in E : u, w \in V_i\}$. Os dois conjuntos formam o subgrafo $G_i = (V_i, E_i, d)$.

- (P4) O último vértice v_{p_s} é utilizado pra fixar os dois últimos subgrafos. Os subconjuntos com, respectivamente, $p_s - p_{s-1} + 3$ e $n - p_s + 3$ vértices construídos através deles são dados por

$$V_s = \{v_{p_{s-1}-3}, v_{p_{s-1}-2}, v_{p_{s-1}-1}, v_{p_{s-1}}, \dots, v_{p_s-1}\} \quad \text{e} \quad V_{s+1} = \{v_{p_s-3}, v_{p_s-2}, v_{p_s-1}, v_{p_s}, \dots, v_n\}.$$

Deles, temos os subconjuntos de arestas $E_s = \{\{u, w\} \in E : u, w \in V_s\}$ e $E_{s+1} = \{\{u, w\} \in E : u, w \in V_{s+1}\}$. Com eles, definimos os dois últimos subgrafos da partição os quais são $G_s = (V_s, E_s, d)$ e $G_{s+1} = (V_{s+1}, E_{s+1}, d)$. Por isso, dizemos que nesta partição $k = s + 1$.

Assim, a partir dessa construção, podemos enunciar e demonstrar o teorema mais importante dessa seção.

Teorema 4.2.1. Sejam $G = (V, E, d)$ uma instância DMDGP, P_G o conjunto de gerador da partição por simetrias e $\mathcal{P} = \{G_1, \dots, G_k\}$ o conjunto com os subgrafos definidos através de P_G . Então, \mathcal{P} é uma partição do grafo G de modo que $|V_i \cap V_{i+1}| = 3$, para cada $1 \leq i \leq k - 1$, cada aresta de poda de E_p pertence a exatamente um subgrafo e, por fim, não existem arestas de poda conectando subgrafos distintos, ou seja,

$$E_p \setminus (E_p \cap (E_1 \cup \dots \cup E_k)) = \emptyset. \quad (4.2.12)$$

Demonstração. Inicialmente, vamos demonstrar que $|V_i \cap V_{i+1}| = 3$, para qualquer $1 \leq i \leq k - 1$. Considere, portanto, os conjuntos de vértices referentes aos dois primeiros subgrafos da construção de \mathcal{P} dados por

$$V_1 = \{v_1, v_2, v_3, v_4, \dots, v_{p_1-3}, v_{p_1-2}, v_{p_1-1}\} \quad \text{e} \quad V_2 = \{v_{p_1-3}, v_{p_1-2}, v_{p_1-1}, v_{p_1}, \dots, v_{p_2-1}\}.$$

É fácil ver, pela ordem dos vértices de cada um, que os únicos vértices em comum entre os dois conjuntos são v_{p_1-3}, v_{p_1-2} e v_{p_1-3} . Ou seja, $|V_1 \cap V_2| = 3$. Agora, tome os conjuntos de vértices de dois subgrafos consecutivos arbitrários G_i e G_{i+1} de \mathcal{P} , dados por

$$V_i = \{v_{p_{i-1}-3}, \dots, v_{p_i-3}, v_{p_i-2}, v_{p_i-1}\} \quad \text{e} \quad V_{i+1} = \{v_{p_i-3}, v_{p_i-2}, v_{p_i-1}, v_{p_i}, \dots, v_{p_{i+1}-1}\}.$$

Isto é, temos que $|V_i \cap V_{i+1}| = 3$, para qualquer $1 \leq i \leq k-1$.

Agora, seja $\{u, w\}$ uma aresta de poda arbitrária e, sem perda de generalidade, suponha que $\{u, w\} \in G_i$ e $\{u, w\} \in G_j$, com $i < j$. Isso significa que $u, w \in (V_i \cap V_j)$. Já sabemos que ou $V_i \cap V_j = \emptyset$ (se $j > i+1$) ou $|V_i \cap V_j| = 3$ (se $j = i+1$). Segue, então, que $w - u \leq 3$. Mas, por definição de aresta de poda, temos que $w - u \geq 4$, o que gera um contradição. Portanto, cada aresta de poda está presente a um único subgrafo.

Por fim, temos que vale a identidade de conjuntos (4.2.12). Sabemos da Teoria de Conjuntos que

$$E_p \cap (E_1 \cup \dots \cup E_k) = (E_p \cap E_1) \cup \dots \cup (E_p \cap E_k). \quad (4.2.13)$$

Seja $\{u, w\}$ uma aresta de poda que não pertença a nenhum conjunto do tipo $E_p \cap E_i$, para qualquer i . Isso significa que $u \in V_i$ e $w \in V_j$, para algum $i < j$, sem perda de generalidade. Então, temos que u e w satisfazem às restrições

$$u \leq v_{p_i-4} \quad \text{e} \quad w \geq v_{p_i}.$$

o que também satisfaz a definição $w - u \geq 4$. Mas, isso implica que

$$u + 4 \leq v_{p_i} \leq w.$$

E, como v_{p_i} é um vértice de simetria, então é impossível que exista essa aresta, pela definição de v_{p_i} , como queríamos demonstrar. \square

Este teorema nos sugere um corolário imediato.

Corolário 4.2.1. As realizações parciais de cada subgrafo G_i são globalmente factíveis.

Demonstração. Com efeito, como vimos no teorema anterior que todas as distâncias de poda estão dentro de algum dos subgrafos, então, ao se realizar cada um deles, todas as podas já foram feitas e nenhuma informação adicional de poda associada a tais vértices sobrou. Portanto, cada uma das realizações parciais será factível com respeito a todo o conjunto de distâncias. \square

O resultado deste corolário consiste na maior vantagem de se utilizar tais partições em vértices de simetrias: não haver podas posteriores à realização. Ou seja, ao realizarmos cada pedaço e unirmos com transformações Euclidianas, então a solução final já é factível.

Colocado os passos da forma que propomos de fazer tal partição, bem como os teoremas que nos embasam a realizar tal divisão, considere os dois exemplos a seguir. Nosso objetivo é comparar a partição proposta anteriormente (Nucci e Mucherino) com esta partição de simetrias. O que representará, neste trabalho, como custo computacional será o número de chamadas do BP necessárias para determinar as soluções, o que já contabiliza a quantidade de *backtrackings* (que deixa esses cálculos mais lentos).

Exemplo 4.2.1. Seja $G = (V, E, d)$ uma instância com $n = 15$ cujas arestas de poda são

$$E_p = \{\{1, 5\}, \{1, 10\}, \{2, 6\}, \{4, 9\}, \{4, 10\}, \{6, 10\}, \{8, 12\}, \{8, 15\}, \{11, 15\}\}.$$

Aplicando a definição de vértices de simetria, temos que

$$S_G = \{v_4, v_{11}\}. \quad (4.2.14)$$

Através deste conjunto, portanto, é possível saber que o número de soluções deste DMDGP será

$$|\mathcal{S}| = 2^{|S_G|} = 2^2 = 4, \quad (4.2.15)$$

resultado obtido através do Corolário 3.2.2.

Assim, vamos estudar as duas divisões propostas e compará-las no que segue.

Divisão Anterior

Inicialmente, a instância é dividida em duas sub-instâncias $G_1 = (V_1, E_1, d)$ e $G_2 = (V_2, E_2, d)$, com o mesmo número de vértices em cada (com uma intersecção de três vértices para realizar as junções das soluções por movimentos rígidos) dados por

$$V_1 = \{v_1, v_2, v_3, v_4, v_5, v_6, v_7, v_8, v_9\} \quad \text{e} \quad V_2 = \{v_7, v_8, v_9, v_{10}, v_{11}, v_{12}, v_{13}, v_{14}, v_{15}\},$$

com o objetivo de contemplar os métodos apresentados anteriormente neste capítulo. Ambos os subgrafos, portanto, possuem nove vértices.

Assim, representamos o conjunto de distâncias d , associado biunivocamente a E , através da matriz de distâncias abaixo, dando destaque às submatrizes associadas a G_1 e a G_2 .

$$D_G = \left(\begin{array}{cccccc|cccccc|ccccc} 0 & d_{1,2} & d_{1,3} & d_{1,4} & \color{red}{d_{1,5}} & 0 & 0 & 0 & 0 & \color{red}{d_{1,10}} & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & d_{2,3} & d_{2,4} & d_{2,5} & \color{red}{d_{2,6}} & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & d_{3,4} & d_{3,5} & d_{3,6} & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & d_{4,5} & d_{4,6} & d_{4,7} & 0 & \color{red}{d_{4,9}} & 0 & 0 & 0 & \color{red}{d_{4,10}} & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & d_{5,6} & d_{5,7} & d_{5,8} & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & d_{6,7} & d_{6,8} & d_{6,9} & 0 & \color{red}{d_{6,10}} & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ \hline & & & & & 0 & d_{7,8} & d_{7,9} & d_{7,10} & 0 & 0 & 0 & 0 & 0 & 0 \\ & & & & & 0 & d_{8,9} & 0 & d_{8,10} & d_{8,11} & \color{red}{d_{8,12}} & 0 & 0 & 0 & \color{red}{d_{8,15}} \\ & & & & & 0 & & 0 & d_{9,10} & d_{9,11} & d_{9,12} & 0 & 0 & 0 & 0 \\ & & & & & & & & 0 & d_{10,11} & d_{10,12} & d_{10,13} & 0 & 0 & 0 & 0 \\ & & & & & & & & 0 & d_{11,12} & d_{11,13} & d_{11,14} & \color{red}{d_{11,15}} & 0 & 0 & 0 \\ & & & & & & & & 0 & d_{12,13} & d_{12,14} & d_{12,15} & 0 & d_{13,14} & d_{14,15} & 0 \\ & & & & & & & & 0 & d_{14,15} & 0 & & 0 & & & 0 \end{array} \right)$$

Pode-se ver que tanto a primeira sub-instância G_1 (cujo conjunto de distâncias corresponde à submatriz contornada pelo retângulo azul) quanto a segunda sub-instância G_2 (cujo conjunto de

distâncias corresponde à submatriz contornada pelo retângulo **verde**) possuem, respectivamente, três informações de poda distintas. Entretanto, outras três informações de poda $d_{1,10}$, $d_{4,10}$ e $d_{6,10}$ (retângulo **vermelho**) ficam de fora de ambas.

Pode-se saber o número de soluções de cada uma pelo BP, analisando a estrutura de simetrias existente em cada uma delas. A sub-instância G_1 possui dois vértices de simetria dados por

$$S_{G_1} = \{v_4, v_7\}.$$

Portanto, de acordo com o Corolário 3.2.2, a quantidade de soluções para ela a serem determinadas pelo BP é

$$|\mathcal{S}_{G_1}| = 2^{|S_{G_1}|} = 2^2 = 4. \quad (4.2.16)$$

Já para G_2 , o conjunto de simetrias é dado por

$$S_{G_2} = \{v_{10}, v_{11}\}.$$

Segue, de acordo com o mesmo resultado, que a quantidade de soluções para esta sub-instância (pelo BP) é

$$|\mathcal{S}_{G_2}| = 2^{|S_{G_2}|} = 2^2 = 4. \quad (4.2.17)$$

Logo, em um raciocínio combinatório, é possível dizer que o número de estruturas possíveis obtidas da combinação das soluções de G_1 com as de G_2 é igual a

$$ns_{1,2} = |\mathcal{S}_{G_1}| \cdot |\mathcal{S}_{G_2}| = 4 \cdot 4 = 16. \quad (4.2.18)$$

Portanto, sem dúvidas, muitas dessas estruturas calculadas terão de ser descartadas ao avaliar a factibilidade utilizando as informações de poda que ficaram de fora das duas estruturas, já que

$$4 = |\mathcal{S}| \neq ns_{1,2} = 16. \quad (4.2.19)$$

Ou seja, somente um quarto das estruturas obtidas com a combinação das soluções parciais de G_1 e G_2 serão factíveis, desperdiçando tempo de computação.

Para ilustrar essas soluções de cada sub-estrutura, suponhamos que as árvores T_1 (Fig. 4.9) e T_2 (Fig. 4.10) são o produto do BP clássico, armazenando as realizações parciais associadas aos subgrafos G_1 e G_2 dados pela divisão anterior.

Divisão por simetrias

Pela nossa proposta, vamos utilizar as informações de simetrias para construir uma partição $G = G_1 \cup G_2$, dividindo G em duas subinstâncias $G_1 = (V_1, E_1, d)$ e $G_2 = (V_2, E_2, d)$ tais que

$$V = V_1 \cup V_2, \quad \text{com} \quad |V_1 \cup V_2| = 3 \quad \text{e} \quad E = E_1 \cup E_2. \quad (4.2.20)$$

A mecânica dessa divisão se dará como apresentado há pouco nesta seção.

Novamente, como o conjunto de simetrias que é dado por $S_G = \{v_4, v_{11}\}$, o conjunto \mathcal{S} de realizações de G tem cardinalidade $|\mathcal{S}| = 2^{|S_G|} = 2^2 = 4$.

Vamos definir, portanto, a partição de V em

$$V_1 = \{v_1, v_2, v_3, v_4, v_5, v_6, v_7, v_8, v_9, v_{10}\} \quad \text{e} \quad V_2 = \{v_8, v_9, v_{10}, v_{11}, v_{12}, v_{13}, v_{14}, v_{15}\}. \quad (4.2.21)$$

Assim, os conjuntos de arestas E_1 e E_2 são, respectivamente, dados por

$$E_p^1 = \{\{1, 5\}, \{1, 10\}, \{2, 6\}, \{4, 9\}, \{4, 10\}, \{6, 10\}\} \quad \text{e} \quad E_p^2 = \{\{8, 12\}, \{8, 15\}, \{11, 15\}\}.$$

Vê-se, além disso, que essa divisão cobre o conjunto E completamente, já que as arestas de discretização são tais que

$$E_d = E_d^1 \cup E_d^2, \quad \text{com} \quad |E_d^1 \cap E_d^2| = 6 \quad (4.2.22)$$

e as arestas de poda são tais que

$$E_p = E_p^1 \cup E_p^2, \quad \text{com} \quad E_d^1 \cap E_d^2 = \emptyset. \quad (4.2.23)$$

Também, como na abordagem anterior, d está associado biunivocamente com a matriz D_G , na qual há destaque novamente às partes associadas, respectivamente, a E_1 e a E_2

$$D_G = \left(\begin{array}{ccccccccc} 0 & d_{1,2} & d_{1,3} & d_{1,4} & \color{red}{d_{1,5}} & 0 & 0 & 0 & \color{red}{d_{1,10}} \\ 0 & d_{2,3} & d_{2,4} & d_{2,5} & \color{red}{d_{2,6}} & 0 & 0 & 0 & 0 \\ 0 & d_{3,4} & d_{3,5} & d_{3,6} & 0 & 0 & 0 & 0 & 0 \\ 0 & d_{4,5} & d_{4,6} & d_{4,7} & 0 & \color{red}{d_{4,9}} & \color{red}{d_{4,10}} & 0 & 0 \\ 0 & d_{5,6} & d_{5,7} & d_{5,8} & 0 & 0 & 0 & 0 & 0 \\ 0 & d_{6,7} & d_{6,8} & d_{6,9} & \color{red}{d_{6,10}} & 0 & 0 & 0 & 0 \\ 0 & d_{7,8} & d_{7,9} & d_{7,10} & 0 & 0 & 0 & 0 & 0 \\ \hline & & & & & 0 & d_{8,9} & d_{8,10} & d_{8,11} & \color{red}{d_{8,12}} & 0 & 0 & \color{red}{d_{8,15}} \\ & & & & & 0 & d_{9,10} & 0 & d_{9,11} & d_{9,12} & 0 & 0 & 0 \\ & & & & & 0 & & d_{10,11} & d_{10,12} & d_{10,13} & 0 & 0 & 0 \\ & & & & & & & 0 & d_{11,12} & d_{11,13} & d_{11,14} & \color{red}{d_{11,15}} & 0 \\ & & & & & & & 0 & d_{12,13} & d_{12,14} & d_{12,15} & 0 & d_{13,14} & d_{14,15} \\ & & & & & & & 0 & d_{14,15} & 0 & & & & 0 \end{array} \right).$$

Nenhuma distância de poda fica de fora da partição, como aconteceu no caso anterior. Tal fato pode ser percebido tanto pelas Equações (4.2.22) e (4.2.23), quanto pela divisão em submatrizes em (4.2.1), já que a submatriz com contorno **vermelho** é nula. Portanto, ao unir as realizações de G_1 com as de G_2 , teremos construído todas as realizações factíveis de G .

Dessa forma, os conjuntos de simetrias são dados, respectivamente, por

$$S_{G_1} = \{v_4\} = S_G \quad \text{e} \quad S_{G_2} = \{v_{11}\}.$$

Sendo assim, os conjuntos \mathcal{S}_1 e \mathcal{S}_2 de realizações de G_1 e G_2 , respectivamente, são tais que

$$|\mathcal{S}_1| = 2^{|S_{G_1}|} = 2^1 = 2 \quad \text{e} \quad |\mathcal{S}_2| = 2^{|S_{G_2}|} = 2^1 = 2.$$

Apelando, novamente, para o raciocínio combinatório de contagem de soluções através da união das subsoluções, o número total de estruturas resultantes da união das realizações de \mathcal{S}_1 e \mathcal{S}_2 é

$$ns_{1,2} = |\mathcal{S}_1| \cdot |\mathcal{S}_2| = 2 \cdot 2 = 4. \quad (4.2.24)$$

Portanto, ao unir as estruturas, teremos completado o conjunto \mathcal{S} de realizações de G completamente sem que tenhamos que descartar qualquer estrutura, já que o número de soluções combinadas é exatamente igual ao número de soluções originais

$$ns_{1,2} = 4 = |\mathcal{S}|. \quad (4.2.25)$$

Esta divisão deve não desperdiçar tempo de computação de estruturas “inúteis” na formação das realizações da estrutura original. Note que, de fato, não existem distâncias de poda para checar factibilidades *a posteriori*, pois todas elas estão armazenadas em alguma das duas subestruturas.

As árvores T_1 e T_2 associadas a este caso estão mostradas, respectivamente, nas Figuras 4.11 e 4.12.

Comparações

Como dito no início, o custo computacional será dado pelo número de chamadas que a instância faz do *Branch & Prune*. Através das Figuras 4.9, 4.10, 4.11 e 4.12, é possível fazer essa contagem, que coincide com a quantidade de nós da árvore em cada nível a menos da inicialização.

	$ \mathcal{S}_1 $	$ \mathcal{S}_2 $	$\sum \mathcal{S}_i $
Divisão Anterior	38	70	108
Divisão com Simetrias	46	34	80

Tabela 4.1: Tabela com a comparação do custo computacional das divisões

O número de chamadas com a abordagem de simetrias foi igual a 80 e para a abordagem anterior foi de 108. Além disso, a primeira abordagem produziu um grande número de realizações a serem descartadas (doze delas), ao contrário desta abordagem em que todas as realizações foram aproveitadas por serem factíveis.

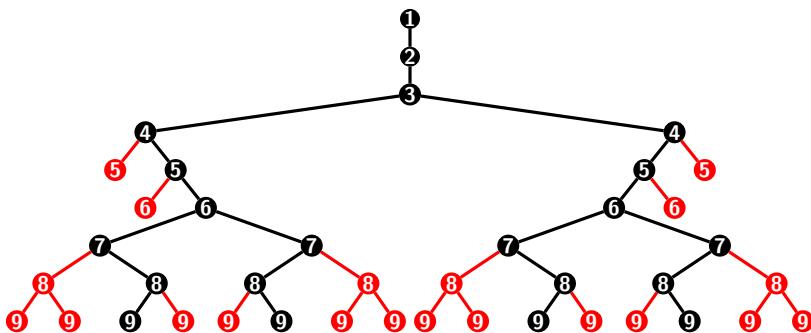


Figura 4.9: A árvore T_1 associada a G_1 .

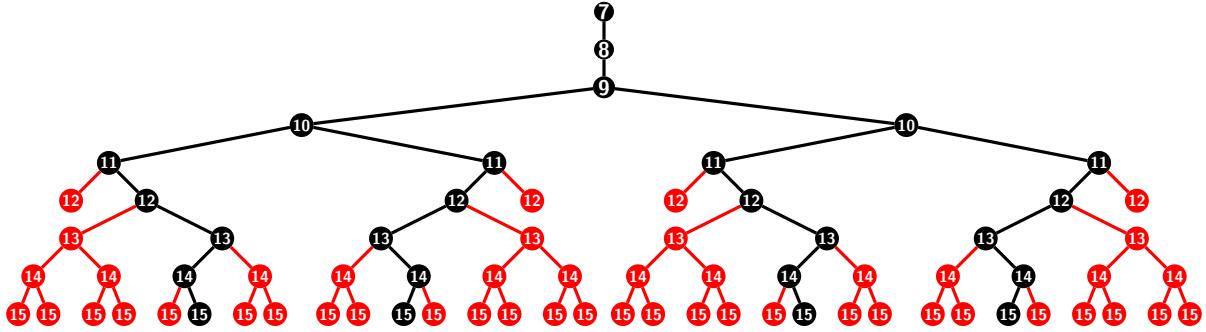


Figura 4.10: A árvore T_2 associada a G_2 .

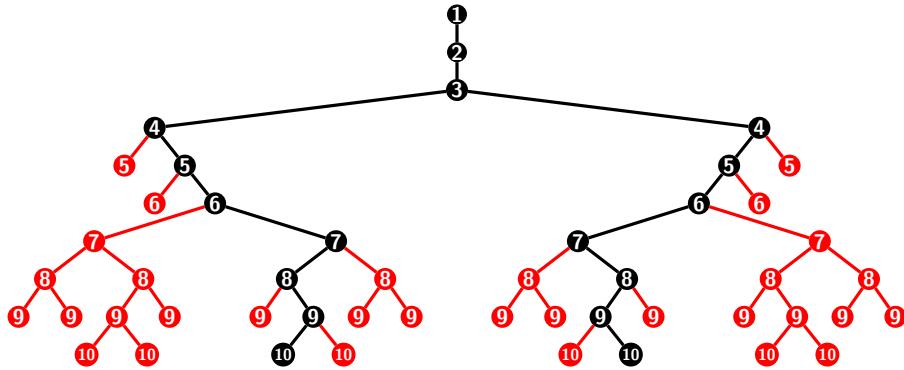


Figura 4.11: A árvore T_1 associada a G_1 .

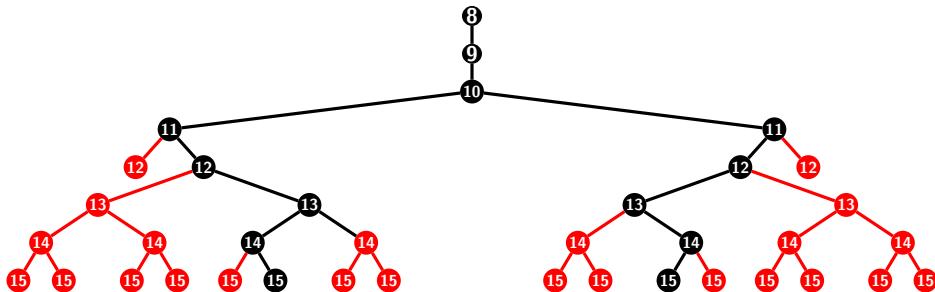


Figura 4.12: A árvore T_2 associada a G_2 .

□

Neste exemplo, utilizamos o algoritmo *Branch & Prune* no modo clássico nas partes. Entretanto, é possível torná-lo ainda mais eficiente ao utilizar o *Sym BP*, como visto na Seção 3.3.

No próximo exemplo, vamos argumentar sobre as razões pelas quais a utilização do *Sym BP* pode ser inviável na estratégia anterior, mas que pode ser muito eficiente na abordagem das simetrias que propomos. Além disso, vamos fazer uma análise de pior caso. Isto é, com sorte, as duas podem funcionar bem, entretanto a segunda será melhor no pior caso.

Exemplo 4.2.2. Seja $G = (V, E, d)$ uma instância com $n = 15$ vértices cujas arestas de poda são dadas por

$$E_p = \{\{1, 8\}, \{5, 12\}, \{10, 15\}\}. \quad (4.2.26)$$

Assim, pelas definições de vértices de simetria, temos que $S_G = \{v_4, v_9, v_{13}\}$. Segue, portanto, que a quantidade de soluções do DMDGP definido por G pelo BP é

$$|\mathcal{S}| = 2^{|S_G|} = 2^3 = 8. \quad (4.2.27)$$

No pior caso, a primeira solução seria descoberta pelo *BP-One* através da exploração da árvore de soluções, que teria a aparência da árvore T da Figura 4.13.

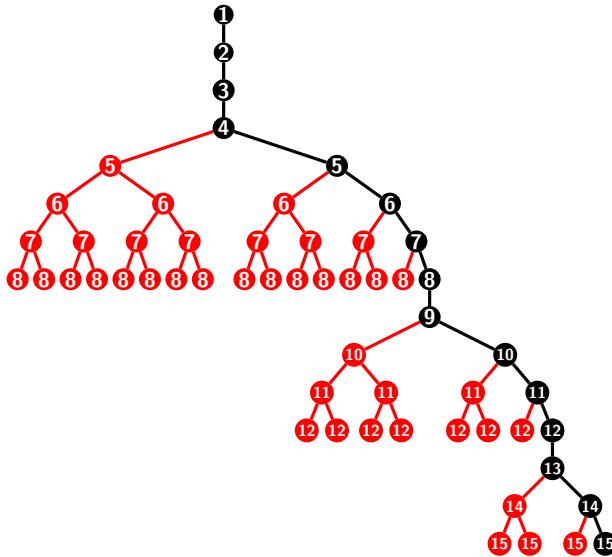


Figura 4.13: A árvore T associada a G , com a primeira solução.

Com isso, de acordo com a mecânica do *Sym BP*, as outras sete soluções devem ser determinadas a partir desta solução e das três simetrias existentes na estrutura.

Divisão Anterior

Pela divisão anterior, gera-se três subgrafos $G_1 = (V_1, E_1, d)$, $G_2 = (V_2, E_2, d)$ e $G_3 = (V_3, E_3, d)$ tais que seus conjuntos de vértices são sobrepostos em três vértices e dados por

$$V_1 = \{v_1, \dots, v_7\}, \quad V_2 = \{v_5, \dots, v_{11}\} \quad \text{e} \quad V_3 = \{v_9, \dots, v_{15}\}. \quad (4.2.28)$$

Assim, suas informações de distâncias são dadas na matriz D_G abaixo, respectivamente, pelas submatrizes associadas às cores azul, verde e amarela. As distâncias que conectam vértices em subgrafos diferentes estão separadas nas matrizes em vermelho. Pode-se ver que as informações de poda $d_{1,8}$ e $d_{6,12}$ estão fora dos três subgrafos.

$$D_G = \left(\begin{array}{cccccc|ccc|c|c|c|c} 0 & d_{1,2} & d_{1,3} & d_{1,4} & 0 & 0 & 0 & d_{1,8} & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ & 0 & d_{2,3} & d_{2,4} & d_{2,5} & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ & & 0 & d_{3,4} & d_{3,5} & d_{3,6} & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ & & & 0 & d_{4,5} & d_{4,6} & d_{4,7} & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ & & & & 0 & d_{5,6} & d_{5,7} & d_{5,8} & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ & & & & & 0 & d_{6,7} & d_{6,8} & d_{6,9} & 0 & 0 & 0 & d_{6,12} & 0 & 0 & 0 \\ & & & & & & 0 & d_{7,8} & d_{7,9} & d_{7,10} & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ & & & & & & & 0 & d_{8,9} & d_{8,10} & d_{8,11} & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ & & & & & & & & 0 & d_{9,10} & d_{9,11} & d_{9,12} & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ & & & & & & & & & 0 & d_{10,11} & d_{10,12} & d_{10,13} & 0 & 0 & d_{10,15} & 0 & 0 & 0 & 0 \\ & & & & & & & & & & 0 & d_{11,12} & d_{11,13} & d_{11,14} & d_{11,15} & 0 & 0 & 0 & 0 & 0 & 0 \\ & & & & & & & & & & & 0 & d_{12,13} & d_{12,14} & d_{12,15} & 0 & d_{13,14} & d_{14,15} & 0 & 0 & 0 & 0 \\ & & & & & & & & & & & & 0 & d_{14,15} & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \end{array} \right)$$

Assim como na árvore T da Figura 4.13, vamos aplicar o algoritmo *BP-One* nas três partes, já que ao unirmos as três partes da primeira solução, basta utilizar as simetrias como no *Sym BP*, abordagem já conhecida como bem eficiente.

Como não existem informações de poda para G_1 , uma primeira realização $x_1^{1,7} \in \mathcal{S}_1$ já se mostra “localmente factível”. Isto é, o algoritmo determina esta primeira solução muito rapidamente, já que não ocorrem *backtrackings*. A mesma coisa acontece para G_2 e uma primeira solução $x_1^{5,11} \in \mathcal{S}_2$, factível para aquele pedaço, aparece rapidamente sem retornos.

É fácil ver que a união dessas soluções será globalmente infactível: basta olhar para a árvore T acima. A falta da informação de poda $d_{1,8}$ dentro de um dos dois subgrafos faz toda a diferença. Assim que se unir as soluções e aplicar o teste das distâncias de poda adicionais, teremos

$$\|x_1^{1,7}(1) - x_1^{5,11}(8)\| \neq d_{1,8}, \quad (4.2.29)$$

o que já inviabiliza esta realização que está sendo construída.

Para encontrar a solução factível para G_1 , na verdade, é necessário explorar oito caminhos (sete deles infactíveis), o que demanda 14 chamadas do algoritmo *BP-One* (subárvore T_1 da Figura 4.14). Já para resolver o DMDGP para G_2 de modo a encontrar uma subsolução globalmente factível, é necessário explorar cinco caminhos (quatro deles infactíveis), necessitando de 9 chamadas do *BP-One* (subárvore T_2 da Figura 4.15). Por fim, para encontrar a primeira solução globalmente factível para o DMDGP associado a G_3 é preciso explorar os cinco caminhos da subárvore T_3 , chamando o *BP-One* 9 vezes também (Figura 4.16).

Lidando com o *BP-One*, nesta abordagem, é provável que essa situação de pior caso ocorra, ou seja, que escolhamos caminhos errados e, portanto, geremos soluções infactíveis. Daí, seria preciso voltar ao processo do início. No pior caso, portanto, seria preciso explorar as $200 = 8 \cdot 5 \cdot 5$ possibilidades de combinações de subsoluções até encontrar, de fato, a solução factível.

Portanto, essa abordagem não parece ter ganhos com a estratégia D & C.

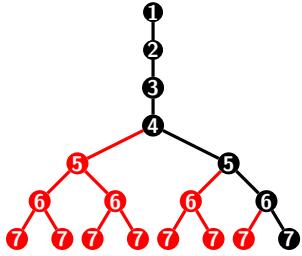


Figura 4.14: A sub-árvore T_1 .

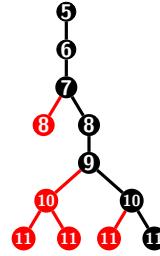


Figura 4.15: A sub-árvore T_2 .

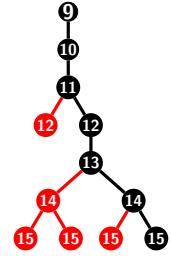


Figura 4.16: A sub-árvore T_3 .

Divisão com Simetrias

Por outro lado, da abordagem do uso de simetrias para dividir a instância DMDGP, gera-se também três subgrafos $G_1 = (V_1, E_1, d)$, $G_2 = (V_2, E_2, d)$ e $G_3 = (V_3, E_3, d)$ tais que

$$V_1 = \{v_1, \dots, v_8\}, \quad V_2 = \{v_6, \dots, v_{12}\} \quad \text{e} \quad V_3 = \{v_{10}, \dots, v_{15}\}. \quad (4.2.30)$$

Assim, suas informações de distâncias são dadas na matriz D_G abaixo, respectivamente, pelas submatrizes associadas às cores azul, verde e amarela. As distâncias que conectam vértices em subgrafos diferentes estão separadas nas matrizes em vermelho.

$$D_G = \left(\begin{array}{cccccc|ccc|ccc} 0 & d_{1,2} & d_{1,3} & d_{1,4} & 0 & 0 & 0 & d_{1,8} & 0 & 0 & 0 & 0 & 0 \\ 0 & d_{2,3} & d_{2,4} & d_{2,5} & 0 & 0 & 0 & & 0 & 0 & 0 & 0 & 0 \\ 0 & d_{3,4} & d_{3,5} & d_{3,6} & 0 & 0 & 0 & & 0 & 0 & 0 & 0 & 0 \\ 0 & d_{4,5} & d_{4,6} & d_{4,7} & 0 & 0 & 0 & & 0 & 0 & 0 & 0 & 0 \\ 0 & d_{5,6} & d_{5,7} & d_{5,8} & & & & & 0 & 0 & 0 & 0 & 0 \\ \hline & & & & 0 & d_{6,7} & d_{6,8} & d_{6,9} & 0 & 0 & d_{6,12} & 0 & 0 \\ & & & & 0 & d_{7,8} & & d_{7,9} & d_{7,10} & 0 & 0 & 0 & 0 \\ & & & & 0 & & d_{8,9} & d_{8,10} & d_{8,11} & 0 & & 0 & 0 \\ & & & & 0 & d_{9,10} & d_{9,11} & d_{9,12} & & & & 0 & 0 \\ \hline & & & & & & 0 & d_{10,11} & d_{10,12} & d_{10,13} & 0 & d_{10,15} & 0 \\ & & & & & & 0 & d_{11,12} & & d_{11,13} & d_{11,14} & 0 \\ & & & & & & 0 & & & d_{12,13} & d_{12,14} & d_{12,15} \\ & & & & & & & & & 0 & d_{13,14} & d_{14,15} \\ & & & & & & & & & 0 & d_{14,15} & 0 \end{array} \right)$$

Esta abordagem necessita de 30 chamadas do *BP-One* para realizar G_1 , explorando dezesseis caminhos até encontrar aquele que é factível (Figura 4.17). Além disso, são necessárias 14 chamadas do algoritmo para realizar G_2 , explorando os oito caminhos possíveis (Figura 4.18). E, por fim, é preciso de 6 chamadas do *BP-One* para determinar a primeira solução para G_3 , explorando quatro caminhos (Figura 4.19).

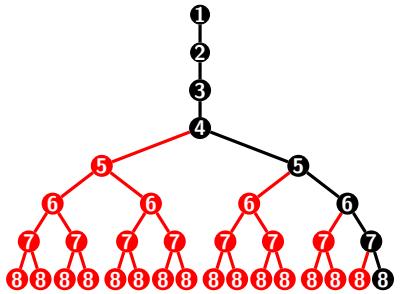


Figura 4.17: A sub-árvore T_1 .

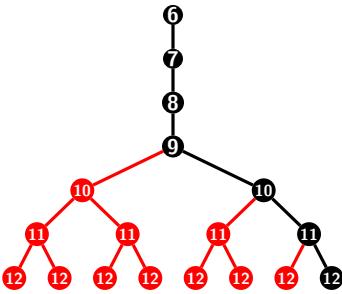


Figura 4.18: A sub-árvore T_2 .

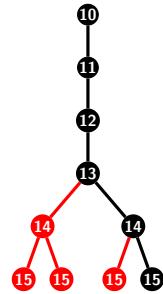


Figura 4.19: A sub-árvore T_3 .

Comparações

Aparentemente, o custo computacional da abordagens de simetria é maior. De fato, se se levar em conta apenas o número de chamadas do BP, que é maior, isto ocorre mesmo. De acordo com a Tabela 4.2, o pior caso da Divisão Anterior requer 32 chamadas do *BP-One*, enquanto que o pior caso da Divisão por Simetrias requer 50 chamadas.

Entretanto, o número de uniões de subsoluções que a abordagem anterior necessita é muito mais alto. Como não há garantia imediata de factibilidade, é preciso testar todas as possibilidade até descobrir qual é, de fato, factível. Este procedimento torna o custo computacional dela dinâmico e, no pior caso descrito acima, inviável. No caso descrito, seria preciso testar as duzentas combinações possíveis em relação a apenas uma combinação da Divisão por Simetrias (Tabela 4.2).

Portanto, para o pior caso acima, a abordagem de simetrias é mais eficiente já que ela produz subsoluções que são factíveis.

	$ \mathcal{S}_1 $	$ \mathcal{S}_2 $	$ \mathcal{S}_3 $	$\sum \mathcal{S}_i $	Combinações
Divisão Anterior	14	9	9	32	200
Divisão com Simetrias	30	14	6	50	1

Tabela 4.2: Tabela com a comparação do custo computacional das divisões

□

Como vimos em ambos os exemplos, a divisão que utiliza as informações de simetria é deveras eficiente, em detrimento da divisão “clássica” de Nucci *et al.* e Mucherino *et al.* que toma uma partição de G em subgrafos com o mesmo número de vértices.

Desse modo, o primeiro passo para se dividir a instância G dada é descobrir quais vértices de V possuem a característica de definirem simetrias ao longo da estrutura. O pseudocódigo do Algoritmo 4, a seguir, realiza esta tarefa. A ele chamaremos de *FindSym*.

Depois de encontrados esses vértices de simetria, é preciso definir uma rotina que de fato partitiona completamente o grafo G em relação a estas simetrias. O Algoritmo 5 a seguir faz essa tarefa utilizando os dados de distâncias e os dados de simetria, provenientes da saída do Alg.4.

Algoritmo 4 *FindSym*

```
1: FindSym( $d$ )
2: Defina uma matriz  $D$  triangular superior a partir de  $d$ , fazendo  $D(i, j) \leftarrow d_{i,j}$ , para  $j > i$ ;
3: Defina um vetor de zeros  $sym$  de comprimento  $n$ ;
4: Como o quarto nível sempre define simetria, faça  $sym(4) = 1$ ;
5: para (cada coluna  $j > 4$  de  $D$ ) faça
6:     se (a submatriz  $D(1 : j - 4, j : n)$  é nula) então
7:          $sym(j) \leftarrow 1$ ;
8:     fim se
9: fim para
10: Defina um vetor vazio de índices de simetria  $S$ ;
11:  $ind \leftarrow 0$ ;
12: para ( $j$  de 1 a  $n$ ) faça
13:     se ( $sym(j) = 1$ ) então
14:          $S(ind) \leftarrow j$ ;
15:          $ind \leftarrow ind + 1$ ;
16:     fim se
17: fim para
18: return  $D$  e  $S$ 
```

Algoritmo 5 *SymSplit*

```
1: SymSplit( $d$ )
2:  $(D, S) = FindSym(d)$ 
3:  $n \leftarrow c(D)$ , onde  $c(D)$  é o número de colunas de  $D$ .
4: Exclua primeiro elemento de  $S$ 
5: se ( $S(|S|) = n$ ) então
6:     Exclua o último elemento de  $S$ 
7: fim se
8: Defina uma matriz  $split$  com duas colunas e  $|S|+1$  linhas identicamente nula
9:  $split(1, 1 : 2) \leftarrow [1 \ S(1)]$ 
10: para ( $i$  de 2 a  $|S|$ ) faça
11:      $split(i, 1 : 2) \leftarrow [split(i - 1, 2) - 3 \ S(i) - 1]$ 
12: fim para
13:  $split(|S|+1, 1 : 2) \leftarrow [split(|S|, 2) - 3 \ n]$ 
14: return  $split$ 
```

4.2.2 Conquistando com *gaps* no conjunto de distâncias

Na intenção de melhorar ainda mais a eficiência dessa abordagem dividir-e-conquistar para resolver o DMDGP, nosso objetivo é explorar a estrutura de distâncias para decidir se o crescimento de cada subárvore será positivo (clássico) ou negativo.

A determinação de uma estrutura DMDGP com o algoritmo BP não é temporalmente homo-

gênea em cada vértice, isto é, ele não determina cada vértice com o mesmo tempo. Pelo contrário, existem setores que são muito “lentos” e setores em que ele funciona muito rápida e eficientemente.

Neste sentido, passamos a estudar o funcionamento do *Branch & Prune* a fim de determinar em quais regiões da instância, de fato, ele deve demorar mais tempo para conseguir realizações factíveis. Através dessa análise mais cuidadosa dos piores casos de instâncias de diversos tamanhos e estruturas que identificamos que tais setores “lentos” estão intimamente relacionados à ausência de informações de distâncias de poda em subconjuntos de vértices sucessivos.

Considere, portanto, uma instância $G = (V, E, d)$ onde $n = |V| = 9$ e

$$E_p = \{\{1, 5\}, \{1, 9\}\}. \quad (4.2.31)$$

Dessa maneira, sua matriz de distâncias D_G terá a aparência abaixo.

$$D_G = \begin{bmatrix} 0 & d_{1,2} & d_{1,3} & d_{1,4} & \textcolor{red}{d_{1,5}} & 0 & 0 & 0 & \textcolor{red}{d_{1,9}} \\ 0 & 0 & d_{2,3} & d_{2,4} & d_{2,5} & 0 & 0 & 0 & 0 \\ 0 & d_{3,4} & 0 & d_{3,5} & d_{3,6} & 0 & 0 & 0 & 0 \\ 0 & d_{4,5} & d_{4,6} & d_{4,7} & 0 & 0 & 0 & 0 & 0 \\ 0 & d_{5,6} & d_{5,7} & d_{5,8} & 0 & 0 & 0 & 0 & 0 \\ 0 & d_{6,7} & d_{6,8} & d_{6,9} & 0 & 0 & 0 & 0 & 0 \\ 0 & d_{7,8} & d_{7,9} & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & d_{8,9} & 0 & 0 & 0 & 0 & 0 & 0 & 0 \end{bmatrix}$$

Fazendo uma análise de pior caso dessa instância, o *BP-One* precisaria explorar a árvore T da Figura 4.20 a fim de encontrar a primeira solução.

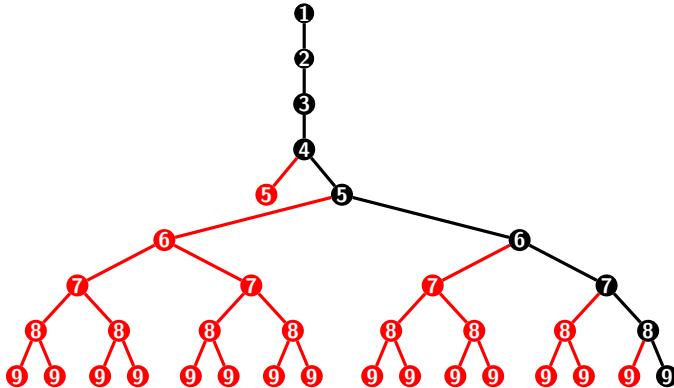


Figura 4.20: Árvore associada à instância G acima, cujas informações de distâncias estão em D_G .

Facilmente, podemos verificar que a parte que comprehende os vértices de v_1 a v_5 tem rápida determinação. De fato, os quatro primeiros vértices são da inicialização e o quinto vértice possui uma informação de poda $d_{1,5}$ associada a ele, ou seja, no máximo será preciso duas chamadas do BP para determiná-lo.

Agora, a parte envolvendo os vértices seguintes é mais demorada. Pela Figura 4.20, é preciso trinta chamadas do BP para determinar uma realização factível para os vértices v_6, v_7, v_8 e v_9 .

Isso deve-se ao fato de que, depois do vértice v_5 , apenas o vértice v_9 possui uma distância de poda associada. Então, o BP abre todos os ramos possíveis até o nono vértice. Em se verificando a infactibilidade, através da informação $d_{1,9}$, ele retrocede a busca novamente para o vértice v_8 e testa a segunda opção. Novamente, encontra uma infactibilidade e retrocede a busca para o vértice v_7 , testando mais um caminho possível até o v_9 . No pior caso ele procede assim até que o último caminho possível se mostre factível.

Em resumo:

- para os vértices v_j que possuem alguma aresta de poda $\{i, j\} \in E_p$ associada a eles, há apenas uma posição factível para cada trio de posições associadas aos vértices v_i, v_{i+1}, v_{i+2} (intersecção de quatro esferas) e, portanto, o BP determina as realizações rapidamente;
- para os vértices v_j para os quais não existe incidência de arestas de poda, o BP é bem mais lento em encontrar o caminho factível.

Foi dessa maneira que identificamos os setores da instância aos quais denominamos *gaps*, cuja teoria será explanada nas definições e resultados que seguem. A gênese deste estudo foi publicada no artigo **Uma nova abordagem para dividir instâncias do Discretizable Molecular Distance Geometry Problem usando gaps**, por Fidalgo et al. [31], apresentado no XXXV Congresso Nacional de Matemática Aplicada e Computacional (CNMAC) de 2014 em Natal/Brasil.

Para cada vértice $v_i \in V$ com posto $\rho(v) > 4$, considere o conjunto de vértices antecessores e adjacentes

$$V_i = N(v_i) \cap \gamma(v_i) = \{v_j \in E : \{j, i\} \in E \text{ e } j < i\} \quad (4.2.32)$$

para o qual sabe-se que $|V_i| \geq 3$, como já visto neste texto, pois seus antecessores imediatos v_{i-3}, v_{i-2} e v_{i-1} estão em V_i sempre em razão da ordem DMDGP.

Como nosso interesse maior está em quantificar os vértices que não possuem informações de poda associados a eles, temos a seguinte definição.

Definição 4.2.1. Um vértice $v_i \in V$ é chamado de **g-vértice** se, e somente se, $|V_i| = 3$.

Ou seja, um g-vértice é aquele que não possui informações de poda associadas a ele. No exemplo anterior, os vértices v_6, v_7 e v_8 são g-vértices.

Definição 4.2.2. Um conjunto de k g-vértices consecutivos, começando em $v_i \in V$, é chamado de *gap* de comprimento k iniciando em v_i e é denotado por

$$\Gamma_i^k = \{v_i, v_{i+1}, \dots, v_{i+k-1}\}. \quad (4.2.33)$$

No exemplo anterior, só existe um *gap*, iniciando em v_6 de comprimento 3, dado por

$$\Gamma_6^3 = \{v_6, v_7, v_8\}. \quad (4.2.34)$$

Pela descrição do início desta seção, o BP se torna mais lento quando tem de encontrar as realizações factíveis para os vértices em algum *gap* e só volta a ser eficiente quando sair dele.

Segue mais uma definição.

Definição 4.2.3. Dado um *gap* Γ_i^k em uma estrutura DMDGP, diremos que (i, k) é o **g-par** associado a ele.

O g-par associado a este *gap* Γ_6^3 é $(6, 3)$.

Assim, se mapearmos os *gaps* da estrutura, poderemos saber, de fato, os setores onde a determinação será mais lenta. Mais do isso: seremos capazes de, no pior caso, saber o número máximo de chamadas que pode-se fazer do BP até encontrar a primeira solução. Esse mapeamento deve ser feito utilizando esses g-pares.

Definição 4.2.4. Seja G uma instância DMDGP. O **Gap Map** dessa instância é o conjunto dos g-pares associados a ela

$$gmap_G = \{(i, k) : \Gamma_i^k \text{ é um gap de } G\}. \quad (4.2.35)$$

A partir disso, vamos denotar um *gap* através do seu g-par.

Além disso, fazem-se necessárias as seguintes definições.

Definição 4.2.5. O número de vértices envolvidos em algum *gap* é dado pelo número

$$vg = \sum_{(i,k) \in gmap_G} k + 1. \quad (4.2.36)$$

E, o número de vértices que não está envolvido com algum *gap* e também não está na inicialização $v_1 \rightarrow v_4$, é dado por

$$vc = n - vg - 4. \quad (4.2.37)$$

No exemplo dado anteriormente, $vg = \sum_{(i,k) \in gmap_G} k + 1 = 3 + 1 = 4$ e, logo,

$$vc = 9 - 4 - 4 = 9 - 8 = 1. \quad (4.2.38)$$

Com essas definições, somos capazes de contar quantas chamadas do BP, dentro de um *gap* (i, k) , uma instância fará para obter a primeira realização factível no pior caso.

Tome, por exemplo, o *gap* $(6, 3)$ dado anteriormente. Para o o primeiro vértice do *gap* v_6 , há a possibilidade de $2 = 2^1$ chamadas no pior caso. Já para o vértice v_7 , há a possibilidade de $4 = 2^2$ chamadas do BP. Para o vértice v_8 , último vértice do *gap*, há a possibilidade de $8 = 2^3$ chamadas do BP no pior caso. Além dos vértices interiores ao *gap*, o vértice seguinte, que no caso é v_9 também importa para esta contagem, já que a árvore ramifica mais uma vez e é, então, podada. Neste nível, no pior caso, há $16 = 2^4$ chamadas do BP. Portanto, do vértice v_6 ao vértice v_9 , o número de vezes que o BP precisa ser acionado até encontrar a solução (pior caso) é

$$2^1 + 2^2 + 2^3 + 2^4 = 2 + 4 + 8 + 16 = 30. \quad (4.2.39)$$

Baseado nesta soma, vamos analisar um caso mais arbitrário. Para um *gap* (i, k) qualquer, há ramificações possíveis para os vértices próprios do *gap* (de v_i a v_{i+k-1}) e também para o primeiro vértice sucessor ao *gap* (v_{i+k} , que é o lugar onde haverá as podas. Assim, temos que o número de chamadas do BP (pior caso) para cada vértice envolvido será:

Vértices	v_i	v_{i+1}	v_{i+2}	\dots	v_{i+k-1}	v_{i+k}
No. de chamadas do BP	2	2^2	2^3	\dots	2^k	2^{k+1}

(4.2.40)

Não é difícil ver que este conjunto forma uma progressão geométrica com o primeiro termo $a_1 = 2$ e a razão $q = 2$ com $k + 1$ termos.

Portanto, o número de chamadas total do pior caso do BP aplicado à seção $v_i \rightarrow v_{i+k}$, com $k + 1$ vértices, é dado pela soma dessa progressão, ou seja, por

$$2 + 2^2 + 2^3 + \dots + 2^k + 2^{k+1} = \frac{2 \cdot (2^{k+1} - 2)}{2 - 1} = 2^{k+2} - 2. \quad (4.2.41)$$

Aplicando, novamente, ao exemplo anterior, temos que esse número é dado por

$$2^{k+2} - 2 = 2^{3+2} - 2 = 2^5 - 2 = 32 - 2 = 30. \quad (4.2.42)$$

Esse número, associado ao *gap* (i, k) , é definido como segue.

Definição 4.2.6. O g - índice do *gap* (i, k) , que compreende à quantidade máxima de chamadas que se pode fazer na realização dos vértices v_i, \dots, v_k é

$$Ind_g(i, k) = 2^{k+2} - 2. \quad (4.2.43)$$

No exemplo, o g-índice associado a este *gap* é $Ind_g(6, 3) = 2^{3+2} - 2 = 2^5 - 2$.

De posse de tal índice, portanto, é possível generalizar esta contagem para um índice “global” da instância, contabilizando tais possibilidades para todos os vértices envolvidos em algum dos *gaps* de $gmap_G$ no pior caso. Este índice é definido abaixo.

Definição 4.2.7. O índice de *gaps* da instância G é dado por

$$I_g(G) = \sum_{(i,k) \in gmap_G} Ind_g(i, k). \quad (4.2.44)$$

Como o exemplo possui apenas um *gap*, então

$$I_g(G) = \sum_{(i,k) \in gmap_G} Ind_g(i, k) = Ind_g(6, 3) = 30. \quad (4.2.45)$$

Além desse índice associado aos *gaps*, é preciso proceder a contagem do número de possibilidades, no pior caso, de chamada do BP para os outros vértices. Como o número de possibilidades é fixo em dois, esse índice complementar é dado como segue.

Definição 4.2.8. O índice complementar desta instância G é dado por

$$I_c(G) = 2 \cdot vc. \quad (4.2.46)$$

Para a instância G do exemplo, o índice complementar é dado por

$$I_c(G) = 2 \cdot vc = 2 \cdot 1 = 2. \quad (4.2.47)$$

Em resumo, estas definições dão a base necessária para definir o Índice de uma instância DMDGP, que compreende no número de chamadas (máximo) que a instância G pode fazer do algoritmo *Branch & Prune* até encontrar a primeira solução, no pior caso. Para as decisões que iremos tomar sobre a direcionalidade do BP em uma instância, esse número será o parâmetro. Os termos “mais rápido” ou “mais lento”, portanto, estarão intimamente associados a ele.

Definição 4.2.9. O Índice de uma instância G é dado por

$$I(G) = I_g(G) + I_c(G). \quad (4.2.48)$$

Portanto, o número de chamadas que a instância G de nove vértices acima pode fazer do BP no pior caso até encontrar a primeira solução é

$$I(G) = I_g(G) + I_c(G) = 30 + 2 = 32. \quad (4.2.49)$$

Para formalizar o método de escolha de direção, é preciso mais uma definição.

Definição 4.2.10. Seja $G = (V, E, d)$ uma instância DMDGP com $n = |V|$ vértices, os quais são v_1, v_2, \dots, v_n . A **Instância Inversa** associada a G é dada por $G^* = (V^*, E^*, d^*)$ tal que $V^* = \{v_n, v_{n-1}, \dots, v_2, v_1\}$, $E^* = E$ e $d^* = d$.

Teorema 4.2.2. Se $G = (V, E, d)$ é uma instância DMDGP, então a instância inversa $G^* = (V^*, E^*, d^*)$ associada a G possui a ordem DMDGP em seus vértices.

Demonstração. Como G possui uma estrutura DMDGP, por hipótese, então as arestas $\{v_{n-2}, v_{n-1}\}$, $\{v_{n-2}, v_n\}$ e $\{v_{n-1}, v_n\}$ estão disponíveis no conjunto E de G . Isso garante, dessa forma, que as arestas $\{v_n, v_{n-1}\}$, $\{v_n, v_{n-2}\}$ e $\{v_{n-1}, v_{n-2}\}$ estão disponíveis no conjunto E^* . Logo, ao considerarmos o subconjunto de vértices $U_0^* = \{v_n, v_{n-1}, v_{n-2}\}$, temos que o subgrafo $G^*(U_0^*)$ forma uma 3 - clique, atendendo à primeira hipótese da definição da ordem DMDGP 2.3.1.

Agora, seja $v_i \in V^*$ um vértice arbitrário com $i < n - 2$. Como $v_i \in V$ também, então as arestas $\{v_i, v_{i+1}\}$, $\{v_i, v_{i+2}\}$, $\{v_i, v_{i+3}\}$, $\{v_{i+1}, v_{i+2}\}$, $\{v_{i+1}, v_{i+3}\}$ e $\{v_{i+2}, v_{i+3}\}$ estão disponíveis em E . Segue que o conjunto $U_{v_i}^*$ induz uma 4 - clique em G^* , satisfazendo à segunda hipótese da definição do DMDGP.

Por fim, sabe-se, também pela ordem DMDGP em G , que

$$d_{i,i+2} < d_{i,i+1} + d_{i+1,i+2},$$

para $i > 3$. Ou seja, reformulando esta desigualdade, temos

$$d_{i+2,i} < d_{i+2,i+1} + d_{i+1,i},$$

satisfazendo a última hipótese da ordem DMDGP, como queríamos demonstrar. \square

Este fato é mais simplesmente enxergado ao olharmos para a matriz de distâncias de G^* . Considere o exemplo G dado acima com nove vértices. Sua instância reversa é tal que a matriz de distâncias D_{G^*} é dada por

$$D_G = \begin{bmatrix} 0 & d_{9,8} & d_{9,7} & d_{9,6} & 0 & 0 & 0 & 0 & \textcolor{red}{d_{9,1}} \\ 0 & d_{8,7} & d_{8,6} & d_{8,5} & 0 & 0 & 0 & 0 & 0 \\ 0 & d_{7,6} & d_{7,5} & d_{7,4} & 0 & 0 & 0 & 0 & 0 \\ 0 & d_{6,5} & d_{6,4} & d_{6,3} & 0 & 0 & 0 & 0 & 0 \\ 0 & d_{5,4} & d_{5,3} & d_{5,2} & \textcolor{red}{d_{5,1}} & & & & \\ 0 & d_{4,3} & d_{4,2} & d_{4,1} & & & & & \\ 0 & d_{3,2} & d_{3,1} & & & & & & \\ 0 & d_{2,1} & & & & & & & \\ 0 & & & & & & & & \end{bmatrix}$$

A matriz D_{G^*} é obtida através de uma permutação σ de linhas e colunas de D_G^T de acordo com

$$\begin{array}{ccccccccc} \sigma : & 1 & 2 & \dots & 8 & 9 \\ & \downarrow & \downarrow & \vdots & \downarrow & \downarrow \\ & 9 & 8 & \dots & 2 & 1 \end{array} \quad (4.2.50)$$

Ao aplicarmos o algoritmo *BP-One* na instância inversa G^* , obtemos as mesmas realizações que obteríamos para a instância G , já que é a mesma instância. Entretanto, elas estão expressas como coordenadas em um outro sistema referencial. Isto é equivalente ao que Nucci et al. propuseram em seu algoritmo de múltiplas árvores de realizações no sentido de fazer com que a árvore binária tenha um crescimento “negativo” (da esquerda para a direita). O que estamos fazendo é generalizando esta idéia, que estava expressa em intervalos, para a linguagem de grafos do DMDGP.

As definições dadas acima, juntamente com o conceito de instância inversa, serão suficientes para fazer uma análise de pior caso na aplicação do *BP-One* em duas direções: positiva e negativa. Como veremos, a seguir, a comparação entre os índices $I(G)$ e $I(G^*)$ será o fator decisivo para escolhermos qual crescimento será adotado para a árvore em questão.

Continuando com o exemplo dado nesta seção, sabemos que a instância G fornece, no pior caso, a árvore T com a primeira realização e que o índice desta instância consiste em $I(G) = 32$. Agora, a sua instância inversa G^* , no pior, caso será explorada pelo *BP-One* e produzirá a árvore T^* da Figura 4.21. Essa árvore tem crescimento negativo, ou seja, cresce de $v_9 \rightarrow v_1$.

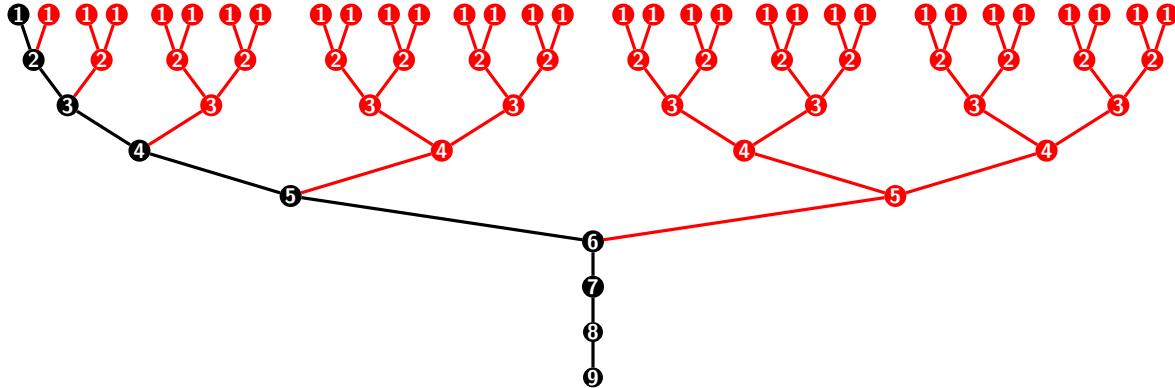


Figura 4.21: Árvore associada à instância G^* acima, cujas informações de distâncias estão em D_{G^*} .

Esta instância possui apenas um *gap*, dado por

$$\Gamma_5^4 = \{v_5, v_4, v_3, v_2\} \quad (4.2.51)$$

e associado ao g-par $(5, 4)$. Com isso, temos que o g-mapa desta instância é

$$gmap_{G^*} = \{(5, 4)\}. \quad (4.2.52)$$

Este conjunto mapeia os *gaps* da instância G^* e, portanto, servirá de base para calcular o índice de G^* . Também, para esse *gap*, temos o seu g-índice dado por

$$Ind_g(5, 4) = 2^{4+2} - 2 = 2^6 - 2 = 64 - 2 = 62. \quad (4.2.53)$$

Além disso, temos os contadores de vértices envolvidos em algum *gap* e vértices complementares, sem a inicialização, dados por

$$vg = \sum_{(i,k) \in gmap_{G^*}} k + 1 = 4 + 1 = 5 \quad \text{e} \quad vc = n - vg - 4 = 9 - 5 - 4 = 0. \quad (4.2.54)$$

Por fim, podemos quantificar os índices de *gaps* e complementar por

$$I_g(G^*) = \sum_{(i,k) \in gmap_{G^*}} Ind_g(i, k) = 62 \quad \text{e} \quad I_c(G^*) = 2 \cdot vc = 2 \cdot 0 = 0. \quad (4.2.55)$$

Portanto, o índice da instância G^* é dado por

$$I(G^*) = I_g(G^*) + I_c(G^*) = 62 + 0 = 62. \quad (4.2.56)$$

O índice $I(G)$, portanto, faz uma análise do pior caso até que o algoritmo *BP* determine a primeira solução da instância. No exemplo de motivação, acima, o índice $I(G) = 32$ é menor do que o índice da instância inversa $I(G^*) = 62$. Portanto, em termos de eficiência, não compensa inverter a instância a fim de realizá-la em menor tempo. Isto também prova que não necessariamente $I(G) = I(G^*)$, mesmo sendo a mesma instância.

Com isso, podemos definir o seguinte teorema que resume esta análise de pior caso para o problema da bidirecionalidade da aplicação do algoritmo BP a uma instância DMDGP. Este é o resultado mais importante desta seção e a sua demonstração segue imediatamente do exposto acima e analisado nos exemplos, ou seja, a prova deste teorema foi construtiva e diluída na teoria.

Teorema 4.2.3. Sejam G e G^* uma instância DMDGP e sua inversa. Se os índices dessas estruturas são tais que

$$I(G^*) < I(G),$$

então o pior caso para G^* é determinado com menos chamadas do BP do que o pior caso de G .

Vamos analisar mais um exemplo e aplicar esse teorema como motivo para decidir a direção.

Exemplo 4.2.3. Considere a instância DMDGP $G = (V, E, d)$ com $n = 14$ vértices tal que seu conjunto de arestas de poda é dado por

$$E_p = \{\{1, 5\}, \{1, 9\}, \{1, 14\}, \{5, 9\}, \{5, 10\}, \{8, 14\}, \{10, 14\}\}. \quad (4.2.57)$$

Analisaremos a determinação de G e G^* com o *BP-One*, no pior caso de cada uma.

Crescimento Positivo

Sua matriz de distâncias D_G , portanto, é

$$D_G = \begin{bmatrix} 0 & d_{1,2} & d_{1,3} & d_{1,4} & \textcolor{red}{d_{1,5}} & 0 & 0 & 0 & \textcolor{red}{d_{1,9}} & 0 & 0 & 0 & 0 & \textcolor{red}{d_{1,14}} \\ 0 & d_{2,3} & d_{2,4} & d_{2,5} & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & d_{3,4} & d_{3,5} & d_{3,6} & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & d_{4,5} & d_{4,6} & d_{4,7} & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & d_{5,6} & d_{5,7} & d_{5,8} & \textcolor{red}{d_{5,9}} & \textcolor{red}{d_{5,10}} & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & d_{6,7} & d_{6,8} & d_{6,9} & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & d_{7,8} & d_{7,9} & d_{7,10} & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & d_{8,9} & d_{8,10} & d_{8,11} & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & \textcolor{red}{d_{8,14}} & 0 & 0 \\ 0 & d_{9,10} & d_{9,11} & d_{9,12} & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & \textcolor{red}{d_{10,14}} & 0 \\ 0 & d_{10,11} & d_{10,12} & d_{10,13} & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & d_{11,12} & d_{11,13} & d_{11,14} & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & d_{12,13} & d_{12,14} & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & d_{13,14} & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \end{bmatrix}$$

Seus *gaps* são descritos no *Gap Map*, que é dado por $gmap_G = \{(6, 3), (11, 3)\}$, e com isso, os seus g-índices correspondem a

$$Ind_g(6, 3) = Ind_g(11, 3) = 2^{3+2} - 2 = 2^5 - 2 = 32 - 2 = 30. \quad (4.2.58)$$

Portanto, o índice de gaps é dado por

$$I_g(G) = Ind_g(6, 3) + Ind_g(11, 3) = 30 + 30 = 60. \quad (4.2.59)$$

Ou seja, são necessários 60 chamadas do *BP-One* para realizar os vértices que são envolvidos em algum dos *gaps* da primeira solução, no pior caso.

Em contrapartida, a quantidade de vértices envolvidos em algum dos *gaps* e os que não estão envolvidos em nenhum são dados, respectivamente, por

$$vg = (3 + 1) + (3 + 1) = 4 + 4 = 8 \quad \text{e} \quad vc = 14 - 8 - 4 = 14 - 12 = 2. \quad (4.2.60)$$

Segue, dessa forma, que o índice complementar é dado por

$$I_c(G) = 2 \cdot vc = 2 \cdot 2 = 4. \quad (4.2.61)$$

Portanto, a quantidade total de chamadas que se faz no pior caso do *BP-One* para a instância G é dado pelo índice de G , o qual corresponde a

$$I(G) = I_g(G) + I_c(G) = 60 + 4 = 64. \quad (4.2.62)$$

Isso também pode ser constatado na Figura 4.22 que mostra a árvore T para o pior caso.

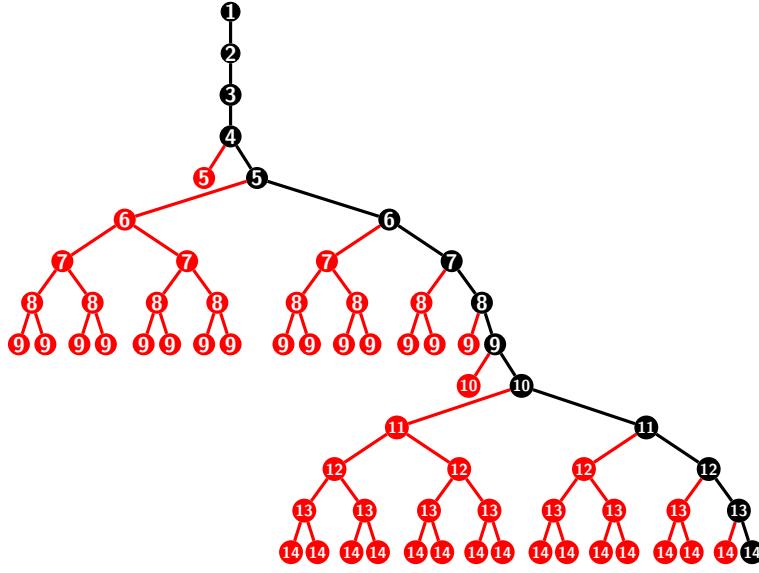


Figura 4.22: Árvore associada à instância G acima, cujas informações de distâncias estão em D_G .

Crescimento Negativo

Agora, tome a instância inversa $G^* = (V^*, E^*, d^*)$ cujas arestas de poda são dadas por

$$E_p^* = \{\{14, 10\}, \{14, 8\}, \{14, 1\}, \{10, 5\}, \{9, 5\}, \{9, 1\}, \{5, 1\}\}. \quad (4.2.63)$$

Dessa maneira, a matriz de distâncias D_{G^*} é dada por

$$D_{G^*} = \begin{bmatrix} 0 & d_{14,13} & d_{14,12} & d_{14,11} & \textcolor{red}{d_{14,10}} & 0 & \textcolor{red}{d_{14,8}} & 0 & 0 & 0 & 0 & 0 & 0 & \textcolor{red}{d_{14,1}} \\ 0 & d_{13,12} & d_{13,11} & d_{13,10} & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & d_{12,11} & d_{12,10} & d_{12,9} & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & d_{11,10} & d_{11,9} & d_{11,8} & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & d_{10,9} & d_{10,8} & d_{10,7} & 0 & \textcolor{red}{d_{10,5}} & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & d_{9,8} & d_{9,7} & d_{9,6} & \textcolor{red}{d_{9,5}} & 0 & 0 & 0 & 0 & 0 & 0 & 0 & \textcolor{red}{d_{9,1}} & 0 \\ 0 & d_{8,7} & d_{8,6} & d_{8,5} & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & d_{7,6} & d_{7,5} & d_{7,4} & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & d_{6,5} & d_{6,4} & d_{6,3} & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & d_{5,4} & d_{5,3} & d_{5,2} & \textcolor{red}{d_{5,1}} & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & d_{4,3} & d_{4,2} & d_{4,1} & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & d_{3,2} & d_{3,1} & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & d_{2,1} & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \end{bmatrix}$$

Como podemos ver, esta estrutura possui três *gaps*:

$$\Gamma_9^1 = \{v_9\}, \quad \Gamma_7^2 = \{v_7, v_6\} \quad \text{e} \quad \Gamma_4^3 = \{v_4, v_3, v_2\}. \quad (4.2.64)$$

Logo, o *Gap Map* desta estrutura é dado por

$$gmap_{G^*} = \{(9, 1), (7, 2), (4, 3)\}. \quad (4.2.65)$$

Para cada um desses *gaps*, há um g-índice, os quais são dados por

$$\begin{aligned} Ind_g(9, 1) &= 2^{1+2} - 2 = 2^3 - 2 = 8 - 2 = 6 \\ Ind_g(7, 2) &= 2^{2+2} - 2 = 2^4 - 2 = 16 - 2 = 14 \\ Ind_g(4, 3) &= 2^{3+2} - 2 = 2^5 - 2 = 32 - 2 = 30 \end{aligned} \quad (4.2.66)$$

De posse desses g-índices, portanto, é possível calcular o índice de *gaps*, que consiste no número máximo possível de chamadas do BP até encontrar a primeira solução para os vértices envolvidos nesses *gaps*, o qual é calculado por

$$I_g(G^*) = Ind_g(9, 1) + Ind_g(7, 2) + Ind_g(4, 3) = 6 + 14 + 30 = 60. \quad (4.2.67)$$

Agora, vamos calcular esse número de chamadas para os vértices que não envolvem os *gaps*. Antes, temos que calcular o número desses vértices. Os vértices envolvidos em algum dos três *gaps* são em um número de

$$vg = (1 + 1) + (2 + 1) + (3 + 1) = 2 + 3 + 4 = 9. \quad (4.2.68)$$

Já os vértices complementares, sem a inicialização, são em número de

$$vc = 14 - 9 - 4 = 1. \quad (4.2.69)$$

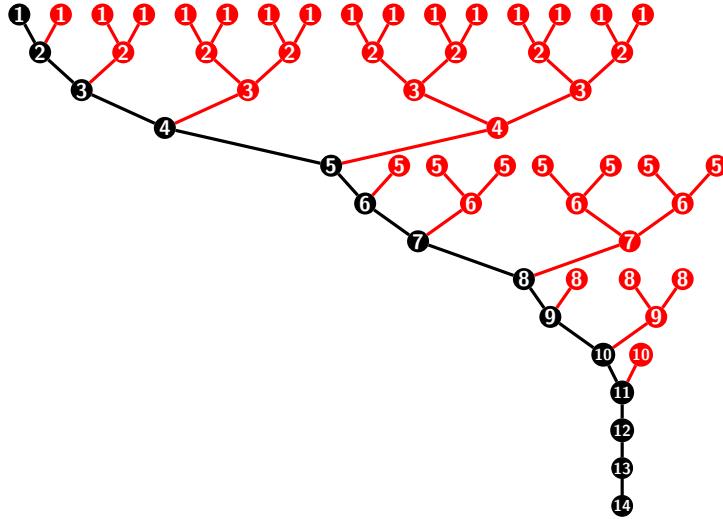


Figura 4.23: Árvore associada à instância G^* acima, cujas informações de distâncias estão em D_{G^*} .

Logo, o índice complementar é dado por $I_c(G^*) = 2 \cdot 1 = 2$. Portanto, o número de chamadas que G pode fazer do *BP-One*, no pior caso, é de

$$I(G^*) = I_g(G^*) + I_c(G^*) = 60 + 2 = 62. \quad (4.2.70)$$

Abaixo, na Figura 4.22, isto pode ser verificado na árvore T^* que retrata o pior caso explorado.

Comparação

Como já explicitado em vários lugares neste texto, o que torna a execução do *BP-One* mais lenta em alguns setores é a falta de informações de poda que, consequentemente, gera *backtrackings* maiores e em maior número, já que a árvore cresce exponencialmente. Também, a quantidade de retornos está intimamente conectada à quantidade de vezes em que o algoritmo é chamado.

Dessa maneira, já demonstramos que o índice $I(G)$ da instância G é quem quantifica esse número de chamadas no pior caso e, portanto, utilizamos este índice para decidir se o crescimento da árvore será positivo ou negativo.

Assim, como o índice da instância G é dado por $I(G) = 64$ e o índice da instância inversa G^* é dado por $I(G^*) = 52$, então o pior caso da instância inversa G^* determina a primeira realização ainda mais rápido do que o pior caso da instância G . Portanto, neste caso vale a pena inverter a instância e permitir o crescimento negativo da árvore.

□

A fim de realizar eficientemente a primeira solução de uma instância DMDGP, portanto, devemos calcular os índices $I(G)$ e $I(G^*)$ e optar pelo menor. O método do Algoritmo 6 faz essa escolha de direção (positiva ou negativa) para a árvore de realizações.

Os dados de entrada são os valores de distâncias entre os vértices armazenados na matriz de distâncias D . Os dados de saída ou é a própria matriz D , caso $I(G)$ seja o menor, ou então é uma permutação igual de linhas e colunas invertendo a instância.

Portanto, o Algoritmo *Gap Grow* abaixo foi designado para decidir se a instância deve ser invertida ou não, analisando o número maximo (pior caso) de chamadas do BP.

Algoritmo 6 *GapGrow*

- 1: $\text{GapGrow}(D)$
 - 2: Determine a matriz $gmap$ cujas linhas são os g-pares associado a D ;
 - 3: Calcule os índices de gap e complementar I_g e I_c ;
 - 4: $I \leftarrow I_g + I_c$;
 - 5: $\sigma \leftarrow (n \ n - 1 \ \dots \ 2 \ 1)$; ▷ Permutação que inverte a instância
 - 6: $D^* = D^T$
 - 7: $D^* \leftarrow D^*(\sigma, \sigma)$; ▷ Aplique a permutação na matriz invertendo linhas e colunas
 - 8: Determine a matriz $gmap^*$ cujas linhas são os g-pares associado a D^* ;
 - 9: Calcule os índices de gap e complementar I_g^* e I_c^* ;
 - 10: $I^* \leftarrow I_g^* + I_c^*$;
 - 11: **se** ($I^* < I$) **faça**
 - 12: $D \leftarrow D^*$;
 - 13: **fim se**
 - 14: **return** D
-

Vamos, por fim, mostrar uma situação em que claramente podemos ver os ganhos de utilizar essa análise do método *Gap Grow* no pior-caso a fim de decidir a direcionalidade do uso do BP.

Exemplo 4.2.4. Considere a instância DMDGP $G = (V, E, d)$ com $n = 8$ vértices tal que o conjunto de arestas de poda é dado por

$$E_p = \{\{1, 8\}, \{2, 8\}, \{3, 8\}, \{4, 8\}\}. \quad (4.2.71)$$

Crescimento Positivo

Sua matriz de distâncias D_G é dada por

$$D_G = \begin{bmatrix} 0 & d_{1,2} & d_{1,3} & d_{1,4} & 0 & 0 & 0 & \textcolor{red}{d_{1,8}} \\ 0 & 0 & d_{2,3} & d_{2,4} & d_{2,5} & 0 & 0 & \textcolor{red}{d_{2,8}} \\ 0 & 0 & d_{3,4} & d_{3,5} & d_{3,6} & 0 & 0 & \textcolor{red}{d_{3,8}} \\ 0 & 0 & d_{4,5} & d_{4,6} & d_{4,7} & \textcolor{red}{d_{4,8}} & 0 & 0 \\ 0 & 0 & d_{5,6} & d_{5,7} & d_{5,8} & 0 & 0 & 0 \\ 0 & 0 & d_{6,7} & d_{6,8} & 0 & 0 & 0 & 0 \\ 0 & 0 & d_{7,8} & 0 & 0 & 0 & 0 & 0 \end{bmatrix} \quad (4.2.72)$$

Só existe um *gap* nesta estrutura, cujo g - par associado a ele é dado por $(5, 3)$. Segue que o *Gap Map* de G é dado por

$$gmap_G = \{(5, 3)\}. \quad (4.2.73)$$

Além disso, o g - índice associado a $(5, 3)$ é

$$Ind_g(5, 3) = 2^{3+2} - 2 = 2^5 - 2 = 32 - 2 = 30. \quad (4.2.74)$$

Agora, o número de vértices associado a este *gap* é

$$vg = 3 + 1 = 4 \quad (4.2.75)$$

e, assim, o número de vértices que não está envolvido neste *gap* é

$$vc = 8 - 4 - 4 = 0. \quad (4.2.76)$$

Dessa maneira, o índice complementar da instância é dado por

$$I_c(G) = 2 \cdot vc = 2 \cdot 0 = 0. \quad (4.2.77)$$

Portanto, o índice de G é dado por

$$I(G) = I_g(G) + I_c(G) = 30 + 0 = 30, \quad (4.2.78)$$

Isto é, no pior caso, o BP deverá fazer 30 chamadas recursivas de si para encontrar a primeira solução do DMDGP associado a G .

Crescimento Negativo

Considere, agora, a instância inversa G^* cuja matriz de distâncias associada D_{G^*} é dada por

$$D_G = \begin{bmatrix} 0 & d_{8,7} & d_{8,6} & d_{8,5} & \textcolor{red}{d_{8,4}} & \textcolor{red}{d_{8,3}} & \textcolor{red}{d_{8,2}} & \textcolor{red}{d_{8,1}} \\ 0 & d_{7,6} & d_{7,5} & d_{7,4} & 0 & 0 & 0 & 0 \\ 0 & d_{6,5} & d_{6,4} & d_{6,3} & 0 & 0 & 0 & 0 \\ 0 & d_{7,4} & d_{7,3} & d_{7,2} & 0 & 0 & 0 & 0 \\ 0 & d_{5,3} & d_{5,3} & d_{5,3} & d_{5,1} & 0 & 0 & 0 \\ 0 & d_{4,2} & d_{4,1} & 0 & d_{3,2} & 0 & 0 & 0 \\ 0 & & & & & & & 0 \end{bmatrix} \quad (4.2.79)$$

É fácil ver que existe uma distância de poda associada a cada um dos vértices com posto maior do que 4, isto é, é necessário duas chamadas do BP para fixar cada uma das posições após a inicialização no pior caso.

Assim, como não há *gaps* em G^* , então $vg = 0$ e $I_g(G) = 0$. Além disso,

$$vc = 8 - 4 - 0 = 4 \quad (4.2.80)$$

e, portanto,

$$I_c(G) = 2 \cdot vc = 2 \cdot 4 = 8. \quad (4.2.81)$$

Segue que

$$I(G) = I_g(G) + I_c(G) = 0 + 8 = 8. \quad (4.2.82)$$

Ou seja, no pior caso, o BP fará oito chamadas recursivas de si mesmo para fixar a estrutura de G^* .

Comparações

O método *Gap Grow* calcula os dois índices de G e G^* e avalia qual o menor deles. Neste caso, o menor está associado ao G^* que necessita de oito chamadas recursivas de si mesmo. O índice de G é trinta, maior, e precisa de trinta chamadas recursivas de si mesmo para fixar uma realização para o mesmo conjunto de vértices.

□

Um trabalho com este algoritmo, intitulado *Deciding directions for the Branch & Prune Algorithm using gaps - a worst-case analysis* [30], foi apresentado no congresso *Mathematica Methods and Modeling of Biophysical Phenomena*, organizado pelo IMPA que teve lugar em Cabo Frio, Brasil, em Março de 2015.

4.2.3 Combinando com Quatérnios

Até agora, nas Seções 4.2.1 e 4.2.2, propusemos métodos que, respectivamente, divide uma instância G do DMDGP em sub-instâncias, tirando proveito da estrutura de simetrias inerente a ela, e resolve cada subinstância com o *BP-One*, lançando mão dos *gaps* para escolher a direção da aplicação (positiva ou negativa) a fim de minimizar a quantidade de *backtrackings*.

Dessa maneira, produziu-se uma sequência de sub-árvores de soluções que armazenam “pedaços” factíveis de soluções, também factíveis, para G , garantia dada pelo corte em vértices de simetria.

A partir disso, portanto, o desafio que nos resta para completar a abordagem Dividir-e-Conquistar é fazer uma combinação eficiente desses “pedaços” de soluções. Nossa estratégia será utilizar a teoria de rotações via Álgebra de Quatérnios (Apêndice C) para essa tarefa.

Nucci et. al [77] trouxe uma abordagem, da qual se comentou em uma das subseções deste capítulo, onde se lança mão de uma translação e duas rotações para unir duas realizações referentes a sub-instâncias consecutivas. Nesta seção, vamos utilizar a mesma idéia, entretanto as duas rotações serão realizadas através da estrutura fornecida pela Álgebra de Quatérnios. Estas idéias tiveram sua gênese em um trabalho Fidalgo *et al.* [34] para o *Workshop on Distance Geometry and Applications*, que teve lugar em Manaus, Brasil em meados de junho de 2013. Neste trabalho, fazíamos todas as rotações com o operador L_q . Nesta tese, vamos lançar mão das matrizes de rotações de quatérnios.

O procedimento para operar as transformações Euclidianas, cujas rotações são dadas por Quaternions, serão descritas em detalhes abaixo.

Sejam G_i e G_{i+1} dois subgrafos consecutivos e arbitrários da partição \mathcal{P} de uma instância G do DMDGP através do *SymSplit* e sejam T_i e T_{i+1} as respectivas realizações dadas pelo *BP-One* combinado com o *GapGrow*.

Temos as três transformações descritas abaixo para unir as duas realizações T_i e T_{i+1} .

(i) Translação

Desejamos transladar a realização deslizante em direção à realização fixa, de modo que o primeiro ponto de T_{i+1} coincida com o antepenúltimo ponto de T_i . Ou seja, queremos que $T_i(a) \leftarrow T_{i+1}(a)$ e, consequentemente, que esta transformação atinja toda a estrutura deslizante.

O vetor de translação, dessa maneira, será dado por

$$\mathbf{v}_T = T_i(a) - T_{i+1}(a).$$

Logo, a transformação no galho deslizante deve ser dada por

$$T_{i+1}(x) \leftarrow T_{i+1}(x) + \mathbf{v}_T, \quad \forall x \in V_i. \tag{4.2.83}$$

Com isso, atingimos o objetivo desejado para o vértice a , já que

$$T_{i+1}(x) + \mathbf{v}_T = T_{i+1}(x) + T_i(a) - T_{i+1}(a) = (T_{i+1}(x) - T_{i+1}(a)) + T_i(a) = T_i(a).$$

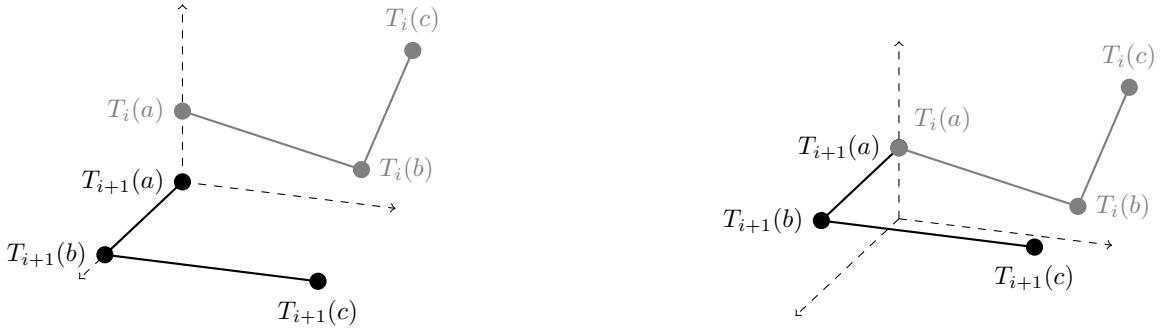


Figura 4.24: Primeira transformação na estrutura deslizante T_{i+1} : uma translação.

A Figura 4.24 representa os efeitos dessa primeira transformação no galho deslizante, apenas, para os três vértices da intersecção.

(ii) Primeira Rotação

O objetivo desta transformação é rotacionar os segmentos que ligam, respectivamente, os pontos $T_i(b)$ e $T_i(a)$ e os pontos $T_{i+1}(b)$ e $T_{i+1}(a)$, no plano que os contém, de modo que $T_i(b) \leftarrow T_{i+1}(b)$. De fato, estes quatro pontos pertencem ao mesmo plano pois se reduzem a três, já que $T_i(a) = T_{i+1}(a)$. Isso justificando tal rotação no plano.

Esses respectivos pares de pontos, em \mathbb{R}^3 , formam os vetores

$$E_i = T_i(b) - T_i(a) \quad \text{e} \quad E_{i+1} = T_{i+1}(b) - T_{i+1}(a),$$

os quais darão origem aos parâmetros da rotação. Desejamos rotacionar $E_i \leftarrow E_{i+1}$.

Seja θ o ângulo formado por estes vetores que compartilham o mesmo vértice. Para definir o quatérnio que estabelece esta transformação rígida, é necessário descobrir qual é esse ângulo. Isso se dará pelo produto interno usual do espaço. É preciso restringir o domínio da função cosseno ao intervalo $[0, \pi]$, para que não se atinja o ângulo $2\pi - \theta$, que possui o mesmo valor de cosseno que θ . O cálculo desse valor de cosseno é dado como na Equação (4.2.84).

Além do valor do cosseno do ângulo de rotação, é preciso calcular o vetor unitário que gere o eixo de rotação. O produto vetorial em \mathbb{R}^3 fornece um ferramental conveniente para esse cálculo, já que está intimamente ligado com sentido de rotação em torno dele, usando a chamada *Regra da Mão-Direita*. Essa regra versa que a ordem dos fatores no produto vetorial é importante, já que este é anticomutativo. Ou seja, se se mudar a ordem dos fatores, muda-se o sentido da rotação. O sinal do produto vetorial codifica o sentido da rotação. Dessa forma, temos que $(E_{i+1} \times E_i)$ é o produto que realiza a rotação que desejamos. Assim, como o vetor \mathbf{n} que gera o eixo deve ser unitário, procedemos seu cálculo, também, através de (4.2.84).

$$\cos(\theta) = \frac{\langle E_{i+1}, E_i \rangle}{\|E_{i+1}\| \|E_i\|} \quad \text{e} \quad \mathbf{n} = \frac{(E_{i+1} \times E_i)}{\|E_{i+1} \times E_i\|}. \quad (4.2.84)$$

O quatérnico que realiza esta transformação, definida através do operador de rotação $L_{\mathbf{n},\theta}$, é dado por $q_{\theta,\mathbf{n}} = q_0 + \mathbf{q}_v = \cos(\frac{\theta}{2}) + \mathbf{n} \sin(\frac{\theta}{2})$.

De posse desse operador $L_{\mathbf{n},\theta}$, transformamos apenas o vetor E_{i+1} , lançando mão de **34 operações**, fazendo

$$T_{i+1}(E_{i+1}) \leftarrow L_{\mathbf{n},\theta} T_{i+1,t}(E_{i+1}), \quad \forall x \in M_{i+1}. \quad (4.2.85)$$

A Figura 4.25 ilustra, geometricamente, esta rotação na estrutura deslizante $T_{i+1,t}$.



Figura 4.25: Segunda transformação na realização deslizante T_{i+1} : uma rotação plana em termos do ângulo θ e em torno do eixo gerado pelo vetor \mathbf{n} .

(iii) Segunda Rotação

Por fim, a terceira transformação consiste em uma segunda rotação, a qual deve ser realizada com alguns cuidados. Esta deve transformar a estrutura deslizante de modo que $T_i \leftarrow T_{i+1}$, sem desfazer quaisquer das modificações realizadas previamente. Considere os vetores em \mathbb{R}^3

$$F_i = T_i(c) - T_i(b) \quad \text{e} \quad F_{i+1} = T_{i+1}(c) - T_{i+1}(b),$$

dos quais se originarão os parâmetros dessa transformação. Desejamos rotacionar F_{i+1} em torno de um eixo a fim de que coincida com F_i . Em semelhança a Nucci et. al [77], escolhemos um eixo que contenha o vetor E_i . Seja \mathbb{P} o plano ortogonal a este vetor. O projetor cuja imagem coincide com \mathbb{P} é dado por $\mathbf{P} = I_3 - E_i E_i^T$, já que este espaço coincide com o complemento ortogonal de $\langle E_i \rangle$. Assim, definimos as projeções de F_i e F_{i+1} por

$$P_i = \mathbf{P} F_i \quad \text{e} \quad P_{i+1} = \mathbf{P} F_{i+1}.$$

Para preservar o sentido da rotação, usando a Regra da Mão-Direita novamente, encontramos o cosseno do ângulo φ de rotação e o diretor do eixo fazendo

$$\cos(\varphi) = \frac{\langle P_{i+1}, P_i \rangle}{\|P_{i+1}\| \|P_i\|} \quad \text{e} \quad \mathbf{m} = \frac{(P_{i+1} \times P_i)}{\|P_{i+1} \times P_i\|}. \quad (4.2.86)$$

Essas projeções são dadas para auxiliar a descobrir o ângulo e o eixo de rotação. Dessa maneira, o quatérnio $q_{\varphi, \mathbf{m}} = q_0 + \mathbf{q}_v = \cos(\frac{\varphi}{2}) + \mathbf{m} \sin(\frac{\varphi}{2})$ gera o operador $L_{\mathbf{m}, \varphi}$ que rotaciona os vetores em \mathbb{R}^3 por um ângulo φ em torno de \mathbf{m} .

Com a definição desse operador de rotação $L_{\mathbf{m}, \varphi}$, define-se a transformação como

$$T_{i+1}(x) \leftarrow L_{\mathbf{m}, \varphi} T_{i+1}(x), \quad . \quad (4.2.87)$$

A Figura 4.26 representa esta última rotação, com a determinação de eixo e ângulo a partir das projeções no complemento ortogonal do espaço gerado por E_i .

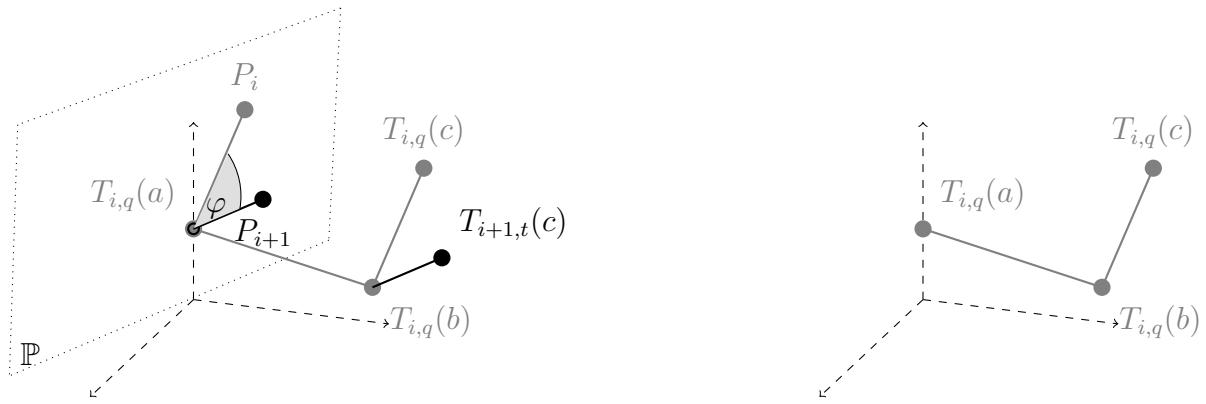


Figura 4.26: Terceira transformação: uma rotação espacial em termos do ângulo φ , dado entre as projeções de F_i e F_{i+1} no complemento ortogonal \mathbb{P} de $\langle E_i \rangle$, e do eixo \mathbf{m} paralelo a este espaço, a qual é definida na estrutura deslizante.

Contagem de Operações

Vamos contar as operações realizadas para aplicar esta série de transformações Euclidianas em um vetor usando matrizes (Apêndice B) e também quatérnios (Apêndice C) a fim de comparar.

No caso das matrizes, são:

- **3 operações** para realizar a translação,
- **25 operações** para calcular a primeira matriz de rotação R_1 ,
- **25 operações** para calcular a segunda matriz de rotação R_2 ,
- **15 operações** para rotacionar o vetor $T_{i+1}(c)$ e
- **45 operações** para compor ambas R_1 e R_2 .

Ou seja, é necessário **113 operações** para realizar esta tarefa. Já para os quatérnios, são:

- **3 operações** para realizar a translação,

- **4 operações** para determinar o primeiro quatérnio de rotação q_1 ,
- **34 operações** para rotacionar o vetor $T_{i+1}(c)$,
- **4 operações** para determinar o segundo quatérnio de rotação q_2 ,
- **28 operações** para compor ambas q_1 e q_2 e
- **29 operações** para extrair a matriz de rotação associada ao produto $q_1 q_2$.

Isto é, são necessárias 102 operações para realizar esta sequência, 11 operações a menos do que no caso das matrizes.

Depois disso, basta aplicar a transformação

$$T_{i+1}(x) \leftarrow Rx, \quad \forall x \in V_{i+1}. \quad (4.2.88)$$

4.2.4 Método *GapSym*

O objetivo deste método é ser como uma junção de dois métodos: *SymSplit* e *GapGrow*.

Para ele, a entrada é o conjunto de distâncias d associado ao DMDGP dado pela instância $G = (V, E, d)$. A saída deste algoritmo é uma matriz *partsols* de três linhas cujas colunas são ordenadamente as coordenadas das realizações parciais encontradas pelo BP em relação às partes da divisão por simetrias. Abaixo, segue o pseudo-código deste algoritmo.

Algoritmo 7 *GapSym*

```

1: GapSym( $d$ )
2: Calcule a matriz de distâncias triangular superior  $D$  associado a este conjunto  $d$ ;
3:  $split = SymSplit(D)$ ;
4:  $\triangleright$  Calcule a matriz de duas colunas cujas linhas são os índices (inferior e superior) da divisão
5: Calcule o número  $dims$  de linhas de split;
6: para ( $i$  de 1 a  $dims$ ) faca
7:    $D_i \leftarrow D(split(i, 1 : 2), split(i, 1 : 2))$ ;            $\triangleright$  Determine cada sub-instância
8:    $D_i = GapGrow(D_i)$ ;            $\triangleright$  Determine a direção de cada sub-instância
9:    $x^i = BranchAndPrune(D_i)$ ;            $\triangleright$  Determine a realização para esta sub-instância
10: fim para
11:  $partsols \leftarrow (x^1, x^2, \dots, x^{dims})$             $\triangleright$  Componha todas as soluções parciais como uma matriz
12: return partsols

```

Considere o seguinte exemplo em que vamos aplicar o algoritmo *GapSym*.

Exemplo 4.2.5. Seja $G = (V, E, d)$ uma instância DMDGP com $n = |V| = 15$ vértices cujas arestas de poda, ordenadamente, são dadas por:

$$E_p = \{\{1, 5\}, \{1, 6\}, \{4, 10\}, \{5, 10\}, \{6, 10\}, \{8, 12\}, \{8, 15\}, \{9, 15\}\}. \quad (4.2.89)$$

Vamos analisar o andamento do algoritmo passo-a-passo. Sua matriz de distâncias associada, dessa maneira, é dada por D_G como segue:

$$D_G = \left[\begin{array}{cccccc|cccccc|ccccc} 0 & d_{1,2} & d_{1,3} & d_{1,4} & \textcolor{red}{d_{1,5}} & \textcolor{red}{d_{1,6}} & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & d_{2,3} & d_{2,4} & d_{2,5} & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & d_{3,4} & d_{3,5} & d_{3,6} & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ \hline & \boxed{0 & d_{4,5} & d_{4,6} & d_{4,7} & 0 & 0 & \textcolor{red}{d_{4,10}} & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0} \\ & & & 0 & d_{5,6} & d_{5,7} & d_{5,8} & 0 & \textcolor{red}{d_{5,10}} & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ & & & 0 & 0 & d_{6,7} & d_{6,8} & d_{6,9} & \textcolor{red}{d_{6,10}} & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ & & & 0 & d_{7,8} & d_{7,9} & d_{7,10} & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ \hline & & & & \boxed{0 & d_{8,9} & d_{8,10} & d_{8,11} & \textcolor{red}{d_{8,12}} & 0 & 0 & 0 & \textcolor{red}{d_{8,15}} & 0 & 0 & 0 & 0 & 0 & 0 & 0} \\ & & & & 0 & d_{9,10} & d_{9,11} & d_{9,12} & 0 & 0 & 0 & 0 & \textcolor{red}{d_{9,15}} & 0 & 0 & 0 & 0 \\ & & & & 0 & d_{10,11} & d_{10,12} & d_{10,13} & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ & & & & 0 & d_{11,12} & d_{11,13} & d_{11,14} & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ & & & & 0 & d_{12,13} & d_{12,14} & d_{12,15} & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ & & & & 0 & d_{13,14} & d_{13,15} & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ & & & & 0 & d_{14,15} & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ & & & & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \end{array} \right]$$

Assim, o algoritmo *FindSym* deve encontrar os vértices de simetria que são v_7 e v_{11} , exceto o quarto vértice que sempre é de simetria, e o algoritmo *SymSplit* deve encontrar a seguinte matriz de índices para as três sub-instâncias

$$split = \begin{bmatrix} 1 & 6 \\ 4 & 10 \\ 8 & 15 \end{bmatrix}. \quad (4.2.90)$$

Dessa forma, computamos as matrizes D_1 , D_2 e D_3 associadas às sub-instâncias G_1 , G_2 e G_3 .

A primeira delas possui seis vértices e é dada por $D_1 = D(1 : 6, 1 : 6)$, explicitada abaixo ao lado da matriz de sua instância inversa D_1^*

$$D_1 = \begin{bmatrix} 0 & d_{1,2} & d_{1,3} & d_{1,4} & \textcolor{red}{d_{1,5}} & \textcolor{red}{d_{1,6}} \\ 0 & d_{2,3} & d_{2,4} & d_{2,5} & 0 \\ 0 & d_{3,4} & d_{3,5} & d_{3,6} & 0 \\ 0 & d_{4,5} & d_{4,6} & 0 & d_{5,6} \\ 0 & 0 & 0 & 0 & 0 \end{bmatrix} \quad \text{e} \quad D_1^* = \begin{bmatrix} 0 & d_{6,5} & d_{6,4} & d_{6,3} & 0 & \textcolor{red}{d_{6,1}} \\ 0 & d_{5,4} & d_{5,3} & d_{5,2} & \textcolor{red}{d_{5,1}} \\ 0 & d_{4,3} & d_{4,2} & d_{4,1} & 0 \\ 0 & d_{3,2} & d_{3,1} & 0 & d_{2,1} \\ 0 & 0 & 0 & 0 & 0 \end{bmatrix}.$$

Temos que os índices de cada uma dessas instâncias são, respectivamente, dados por

$$I_{G_1} = 4 \quad \text{e} \quad I_{G_1^*} = 6. \quad (4.2.91)$$

Como o índice da instância inversa é maior, então a primeira sub-instância não precisa ser invertida já que, no pior caso, ela deve demorar menos para ser determinada.

A segunda delas tem sete vértices e é dada por $D_2 = D(4 : 10, 4 : 10)$, explicitada abaixo ao lado da matriz de sua instância inversa D_2^*

$$D_2 = \begin{bmatrix} 0 & d_{4,5} & d_{4,6} & d_{4,7} & 0 & 0 & \textcolor{red}{d_{4,10}} \\ 0 & d_{5,6} & d_{5,7} & d_{5,8} & 0 & \textcolor{red}{d_{5,10}} & \\ 0 & d_{6,7} & d_{6,8} & d_{6,9} & \textcolor{red}{d_{6,10}} & & \\ 0 & d_{7,8} & d_{7,9} & d_{7,10} & & & \\ 0 & d_{8,9} & d_{8,10} & & & & \\ 0 & d_{9,10} & & & & & \\ 0 & & & & & & \end{bmatrix} \quad \text{e} \quad D_2^* = \begin{bmatrix} 0 & d_{10,9} & d_{10,8} & d_{10,7} & \textcolor{red}{d_{10,6}} & \textcolor{red}{d_{10,5}} & \textcolor{red}{d_{10,4}} \\ 0 & d_{9,8} & d_{9,7} & d_{9,6} & 0 & 0 & \\ 0 & d_{8,7} & d_{8,6} & d_{8,5} & 0 & 0 & \\ 0 & d_{7,6} & d_{7,5} & d_{7,4} & & & \\ 0 & d_{6,5} & d_{6,4} & & & & \\ 0 & d_{5,4} & & & & & \\ 0 & & & & & & \end{bmatrix}.$$

Calculando os índices das duas instâncias, temos que

$$I_{G_2} = 14 \quad \text{e} \quad I_{G_2^*} = 6. \quad (4.2.92)$$

Isto é, o índice da inversa é menos da metade do índice da sub-instância. Assim, vale a pena inverter já que, no pior caso, determinar G_2^* deve ser mais rápido do que determinar G_2 . Então, faz-se $D_2 \leftarrow D_2^*$.

Por fim, a terceira sub-instância possui oito vértices cuja matriz de distâncias associada é dada por $D_3 = D(8 : 15, 8 : 15)$, explicitada abaixo ao lado da matriz de sua instância inversa D_3^*

$$D_3 = \begin{bmatrix} 0 & d_{8,9} & d_{8,10} & d_{8,11} & \textcolor{red}{d_{8,12}} & 0 & 0 & \textcolor{red}{d_{8,15}} \\ 0 & d_{9,10} & d_{9,11} & d_{9,12} & 0 & 0 & 0 & \textcolor{red}{d_{9,15}} \\ 0 & d_{10,11} & d_{10,12} & d_{10,13} & 0 & 0 & 0 & \\ 0 & d_{11,12} & d_{11,13} & d_{11,14} & 0 & & & \\ 0 & d_{12,13} & d_{12,14} & d_{12,15} & & & & \\ 0 & d_{13,14} & d_{13,15} & & & & & \\ 0 & d_{14,15} & & & & & & \\ 0 & & & & & & & \end{bmatrix}$$

e

$$D_3^* = \begin{bmatrix} 0 & d_{15,14} & d_{15,13} & d_{15,12} & 0 & 0 & \textcolor{red}{d_{15,9}} & \textcolor{red}{d_{15,8}} \\ 0 & d_{14,13} & d_{14,12} & d_{14,11} & 0 & 0 & 0 & \\ 0 & d_{13,12} & d_{13,11} & d_{13,10} & 0 & 0 & 0 & \\ 0 & d_{12,11} & d_{12,10} & d_{12,9} & \textcolor{red}{d_{12,8}} & & & \\ 0 & d_{11,10} & d_{11,10} & d_{11,9} & & & & \\ 0 & d_{10,9} & d_{10,8} & & & & & \\ 0 & d_{9,8} & & & & & & \\ 0 & & & & & & & \end{bmatrix}.$$

Assim, calculamos os índices de ambas as estruturas

$$I_{G_3} = 16 \quad \text{e} \quad I_{G_3^*} = 16. \quad (4.2.93)$$

Como os dois índices são iguais, então não é necessário inverter a instância pois os piores casos são semelhantes, já que fazem as mesmas 16 chamadas recursivas do BP.

Aplicando o BP às três instâncias D_1 , D_2 e D_3 , temos as três soluções dadas por

$$x^1 = (x_1^1, \dots, x_6^1), \quad x^2 = (x_4^2, \dots, x_{10}^2) \quad \text{e} \quad x^3 = (x_8^3, \dots, x_{15}^3) \quad (4.2.94)$$

as quais serão unidas no método seguinte e, no final, não será preciso avaliar a factibilidade da solução final já que não há mais informações de poda para utilizar.

Portanto, no final, o método une essas soluções em uma única matriz

$$partsols = (x_1^1, x_2^1, \dots, x_5^1, x_6^1, x_4^2, x_5^2, \dots, x_9^2, x_{10}^2, x_8^3, x_9^3, \dots, x_{14}^3, x_{15}^3). \quad (4.2.95)$$

□

4.3 Resultados Computacionais

4.4 Resultados Computacionais

A fim de ilustrar os algoritmos deste trabalho, realizamos alguns experimentos computacionais com um conjunto de instâncias DMDGP artificialmente geradas, baseado no trabalho de [12], as chamadas *Lavor Instances*.

A versão do algoritmo *Branch & Prune* utilizada nos testes foi uma implementação própria com colaboradores feita em MATLAB, baseada nas linhas do *MD-Jeep* (código estruturado em linguagem C) elaborado por Mucherino *et al.* cujos detalhes estão no trabalho “*MD-Jeep: an implementation of a Branch-and-Prune algorithm for Distance Geometry Problems*” [70].

A qualidade da solução foi testada utilizando a medida *Largest Distance Error* (*LDE*), a qual consiste na função de penalidade mais comum na avaliação de soluções para problemas moleculares de Geometria de Distâncias como um problema de otimização [70]. Para uma conformação $x = (x_1, x_2, \dots, x_n)$ de uma instância com n vértices, cujo conjunto de distâncias disponíveis é dada por d e o número dessas distâncias é dado por $m = |d|$, o *LDE* é definido por

$$LDE(x) = \frac{1}{m} \sum_{i,j} \frac{|\|x_i - x_j\| - d_{i,j}|}{d_{i,j}}. \quad (4.4.1)$$

Todos os algoritmos utilizados neste trabalho foram escritos em ambiente MATLAB e os testes foram realizados em um computador com processador Intel Core i5 1,6 Ghz, com uma memória de 2 Gb 1333 Mhz DDR3 em um ambiente operacional de Mac OS X.

Nestes testes, foram utilizados estruturas DMDGP artificialmente geradas com $n = 100, 300, 500$ e 1000 vértices. Após a geração das instâncias, computou-se a matriz de distâncias (triangular superior) associada a ela e aplicou-se o valor de corte de 4: as distâncias abaixo desse valor são consideradas como disponíveis no problema e as distâncias acima desse valor são consideradas indisponíveis e, portanto, as entradas da matriz associadas a elas são dadas em valor zero. Usualmente, os valores de corte para problemas moleculares de Geometria de Distâncias estão entre 5 e 6, simulando o problema com proteínas cujas distâncias provêm de experimentos de Ressonância Magnética Nuclear que fornece distâncias entre 5 e 6 angstrons. Aplicando estes cortes às instâncias Lavor, notamos que as matrizes se tornaram matrizes quase-banda ainda pouco esparsas para

simular uma estrutura como desejávamos. Então, passamos a estudar com mais afinco o comportamento dessas instâncias para vários cortes, o que gerou o relatório técnico *Lavor Instances for the DMDGP: some features and behavior* [29]. Concluimos, portanto, que 4 seria o corte ideal para essas instâncias a fim de simular uma situação como desejávamos.

Depois de aplicado o corte, o novo conjunto de distâncias se tornou *input*. Assim, cada instância $G = (V, E, d)$, com $n = |V|$ vértices, e cada instância inversa $G^* = (V^*, E^*, d^*)$, também com n vértices, foram respectivamente divididas de acordo com o método *SymSplit* em partes G_1, \dots, G_k e G_1^*, \dots, G_k^* , onde cada parte é uma sub-instância $G_i = (V_i, E_i, d)$ e $G_i^* = (V_i^*, E_i^*, d^*)$.

Os resultados destes testes serão apresentados por meio de duas tabelas para cada instância, como segue detalhado abaixo.

A primeira tabela tem o objetivo de detalhar características de cada uma das partes bem como de suas respectivas inversas, como segue:

- identificação da parte (primeira coluna),
- o número de vértices de cada subestrutura (segunda coluna),
- o índice de cada uma delas (terceira coluna),
- o tempo de computação (CPU) da realização de cada uma em segundos (quarta coluna) e
- o LDE da estrutura determinada para cada uma das partes a fim de checar se resolveu o problema local (quinta coluna).

Já a segunda tabela traz os dados das uniões entre as sub-instâncias com o menor índice para cada parte G_i da divisão. Nas primeiras duas instâncias, computamos os tempos para a junção utilizando quatérnios e matrizes. Mas, como os quatérnios se mostraram mais lentos em tempo de CPU (o que vem em sentido contrário ao que vimos na teoria), adotamos apenas as uniões com matrizes de rotação dada pela Fórmula de Rodrigues.

Como os tempos podem variar na máquina devido a vários fatores, vamos executar o mesmo algoritmo um número M de vezes a fim de encontrar uma tendência média temporal de determinação.

As comparações que desejamos fazer serão relacionadas sempre com a aplicação do chamado “BP Clássico”, ou seja, do *BP-One*. O tempo de determinação da estrutura completa, em M vezes, com este algoritmo será chamado de *TempoBP* e o LDE associado a ele de *LdeBP*.

Como a motivação inicial dada neste trabalho foi de um método que paralelize o BP na determinação de soluções para o DMDGP (Seção 4.1.1), então o tempo de determinação de uma estrutura em M vezes também, utilizando a combinação do método *GapSym* e alguma forma de unir as realizações parciais em forma de cascata, será dada pela soma

$$TempoGapSym = \left(\max_{1 \leq i \leq k} Tempo(G_i) \right) + \left(\sum TempoUniao_I \right), \quad (4.4.2)$$

onde $Tempo(G_i)$ é o tempo de determinação em M vezes da parte G_i e $TempoUniao_I$ é o tempo utilizado para a união em M vezes das partes de índices no conjunto I .

4.4.1 Testes Realizados com instâncias artificiais

Instância Um: $n = 100$ vértices

A primeira instância testada tem $n = 100$ vértices e a chamamos de **INSTANCIATESTE100**, cuja matriz de distâncias está dada pela ferramenta *spy* do MATLAB na Figura 4.27.

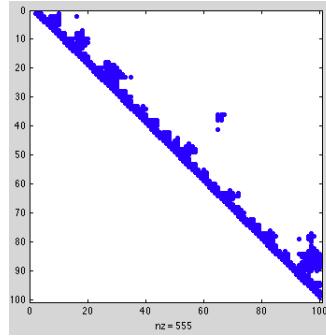


Figura 4.27: Matriz de distâncias de INSTANCIATESTE100.

Esta instância possui 555 valores não-nulos. Como 294 deles compreendem às distâncias de discretização, inerente à estrutura DMDGP, então temos 261 informações de poda nesta estrutura.

A fim de aferir um tempo médio, esta estrutura foi submetida à determinação pelo BP de forma completa com $M = 100.000$ vezes. Isto demandou um tempo de 951,49 segundos, ou aproximadamente, 15,8 minutos. O LDE da solução *solucaoBP* encontrada foi de $9,39 \times 10^{-15}$.

Assim, a divisão proposta pelo método *SymSplit* tem como os índices inferiores e superiores das quatro partes G_1 , G_2 , G_3 e G_4 como as entradas nas linhas da matriz *split100* abaixo:

$$split100 = \begin{bmatrix} 1 & 20 \\ 18 & 35 \\ 33 & 74 \\ 72 & 100 \end{bmatrix}. \quad (4.4.3)$$

Nas Figuras 4.28 - 4.31, temos o *spy* das matrizes associadas a G_1 , G_2 , G_3 e G_4 e, nas Figuras 4.32 - 4.35, temos o *spy* das matrizes associadas a G_1^* , G_2^* , G_3^* e G_4^* .

Na Tabela 4.3, temos os dados da primeira tabela, já referidos anteriormente, com os números de vértices, os índices, os tempos de CPU e o LDE de cada reaalaização parcial para a aplicação do BP em cada uma das quatro partes e suas inversas em $M = 100.000$ vezes.

Temos de escolher uma sequência de quatro sub-instâncias (entre direção direita e esquerda) a fim de cobrir toda a instância, utilizando os critérios do método *GapGrow*, o que é feito tomando as de menor índice $I(G_i)$. Pelos dados da tabela, portanto, temos a sequência G_1^* , G_2^* , G_3^* e G_4^* , ou seja, todas as instâncias inversas. De fato, isto ocorre já que

$$36 = I_{G_1^*} < I_{G_1} = 44, \quad 30 = I_{G_2^*} < I_{G_2} = 36, \quad 96 = I_{G_3^*} < I_{G_3} = 98 \text{ e } 52 = I_{G_4^*} < I_{G_4} = 82.$$

Pode-se ver, pela tabela, que para a primeira, a terceira e a quarta sub-instâncias, de fato, mostram que a direção de menor índice implicou em menor tempo de realização com o *BP-One*,

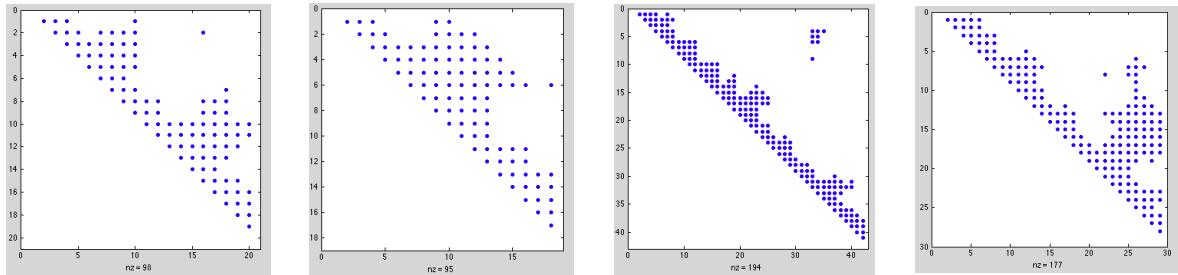


Figura 4.28: *spy* de Figura 4.29: *spy* de Figura 4.30: *spy* de Figura 4.31: *spy* de D_{G_1}

D_{G_2}

D_{G_3}

D_{G_4}

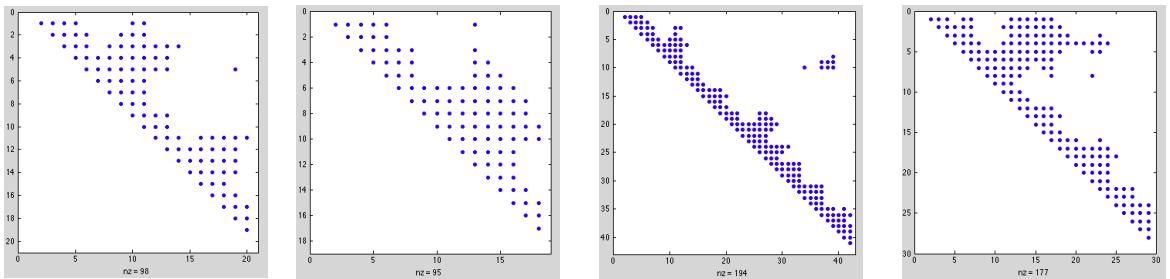


Figura 4.32: *spy* de Figura 4.33: *spy* de Figura 4.34: *spy* de Figura 4.35: *spy* de $D_{G_1^*}$

$D_{G_2^*}$

$D_{G_3^*}$

$D_{G_4^*}$

Parte	$ V_i $	$I(G_i)$	Tempo	LDE	Parte	$ V $	$I(G)$	Tempo	LDE
G_1	20	44	162,46	$2,91 \times 10^{-14}$	G_1^*	20	36	162,41	$4,50 \times 10^{-14}$
G_2	18	36	157,36	$3,44 \times 10^{-15}$	G_2^*	18	30	158,67	$2,83 \times 10^{-15}$
G_3	42	98	349,84	$1,15 \times 10^{-15}$	G_3^*	42	96	343,94	$2,06 \times 10^{-15}$
G_4	29	82	273,45	$5,32 \times 10^{-15}$	G_4^*	29	52	273,41	$6,81 \times 10^{-15}$

Tabela 4.3: Tabela com os dados de índice, tempo e LDE das partes e suas inversas.

mesmo que a diferença fosse pequena. Já para a segunda sub-instância, isso não ocorreu. Isto é possível, já que a análise teórica feita é com os piores casos de cada uma das direções.

Depois disso, utilizamos tanto as Matrizes de Rotação quanto as matrizes de Quaternios para unir estas quatro sub-instâncias em uma instância completa e medimos o tempo de CPU (em segundos) em $M = 100.000$ vezes e o LDE das sub-soluções. Estes detalhes estão mostrados na segunda tabela associada a esta instância, dada pela Tabela 4.4 abaixo.

Vê-se nos dados da última tabela, que mesmo com os ganhos teóricos, a implementação feita não conseguiu superar em tempo de computação a implementação para as matrizes. Dessa maneira, vamos utilizar as rotações com matrizes para fazer a computação do tempo em geral.

Como já comentou-se, o tempo da determinação da estrutura total com a estratégia dividir-e-conquistar é dado pelo tempo de CPU da estrutura com o tempo máximo (já que a análise tem motivação de computação em paralelo, sem computar o tempo de comunicação entre os processadores) juntamente com a soma dos tempos de todas as uniões. Segue, portanto, que o tempo de

União	Intervalo	Tempo	LDE	Tipo
$G_1^* \cup G_2^*$	[1, 35]	16, 15	$2,72 \times 10^{-14}$	Matrizes
$G_3^* \cup G_4^*$	[33, 100]	16, 69	$4,99 \times 10^{-15}$	Matrizes
$G_1^* \cup G_2^* \cup G_3^* \cup G_4^*$	[1, 100]	17, 91	$1,48 \times 10^{-14}$	Matrizes
$G_1^* \cup G_2^*$	[1, 35]	23, 20	$2,72 \times 10^{-14}$	Quatérnios
$G_3^* \cup G_4^*$	[33, 100]	22, 82	$6,71 \times 10^{-15}$	Quatérnios
$G_1^* \cup G_2^* \cup G_3^* \cup G_4^*$	[1, 100]	23, 65	$1,93 \times 10^{-14}$	Quatérnios

Tabela 4.4: Tabela com os dados de uniões com matrizes e quatérnios.

determinação da estrutura com o método deste trabalho é $TempoGapSym = 394,69$ segundos, o que dá aproximadamente, 6,58 minutos. Além disso, temos que $LdeGapSym = 1,48 \times 10^{-14}$.

As representações geométricas, dadas em MATLAB, respectivamente, das soluções com o BP clássico e com o nosso método são dadas pelas Figuras 4.36 e 4.37 abaixo.

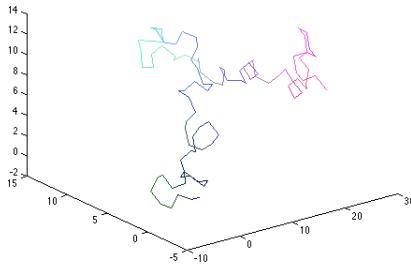


Figura 4.36: Solução com o BP clássico.

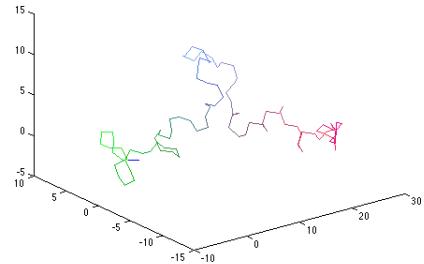


Figura 4.37: Solução com Simetrias e *Gaps*.

Figura 4.38: Representação gráfica da geometria de ambas as soluções em \mathbb{R}^3 .

Instância Dois: $n = 300$ vértices

A segunda instância testada tem $n = 300$ vértices. Ela foi chamada, em consonância com a primeira, de **INSTANCIATESTE300** cuja esparsidade de sua matriz de distâncias pode ser vista em termos da ferramenta *spy* do MATLAB na Figura 4.39.

A fim de aferir um tempo médio, esta estrutura foi submetida à determinação pelo BP de forma completa com $M = 10.000$ vezes. Isto demandou um tempo de 414,44 segundos, ou aproximadamente, 6,91 minutos. O LDE da solução *solucaoBP* encontrada foi de $7,43 \times 10^{-14}$.

Dessa maneira, utilizando o roteiro do algoritmo *SymSplit*, a instância G foi dividida em sete partes $G_1, G_2, G_3, G_4, G_5, G_6$ e G_7 cujos índices dos vértices (limitantes inferior e superior, respectivamente) são dados nas entradas das linhas da matriz *split300* abaixo

$$split300 = \begin{bmatrix} 1 & 53 & 117 & 119 & 127 & 130 & 133 \\ 55 & 119 & 121 & 129 & 132 & 135 & 300 \end{bmatrix}^T.$$

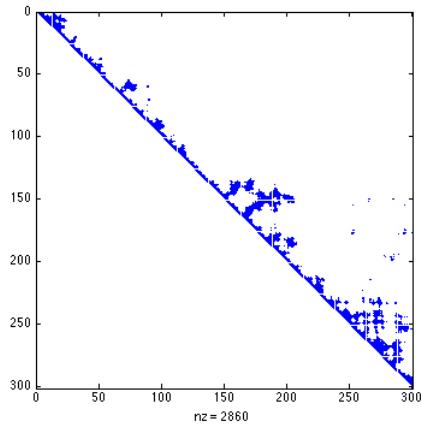


Figura 4.39: Matriz de distâncias de INSTANCETEST300.

Adicionamos os *spy* da matriz de distâncias associada à sub-instância G_7 (Figura 4.40) e da matriz associada às sub-instâncias G_7^* (Figura 4.41) porque são as maiores, ambas com 168 vértices.

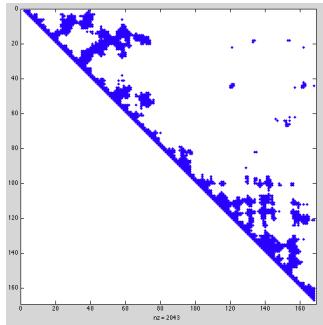


Figura 4.40: *spy* de D_{G_7}

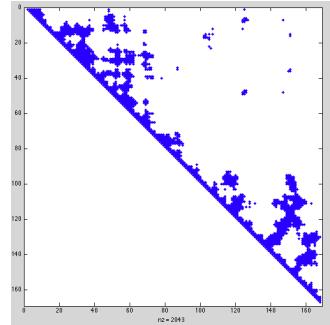


Figura 4.41: *spy* de $D_{G_7^*}$

Figura 4.42: *Spy* das duas maiores sub-instâncias.

Já na Tabela 4.5, mostramos os dados da primeira tabela para a segunda instância com os números de vértices de cada sub-instância, os índices (inclusive das inversas), os tempos de CPU e o LDE de cada realização parcial para a aplicação do BP em cada parte em $M = 10.000$ vezes.

A nossa implementação não conseguiu determinar uma solução para a última instância inversa por uma razão alheia ao nosso conhecimento.

Para encontrar uma solução completa, temos de escolher um conjunto de sete sub-instâncias e o critério utilizado são aqueles de menor índice. Assim, vamos unir, respectivamente, as estruturas G_1^* , G_2^* , G_3 , G_4^* , G_5 , G_6 e G_7 . Em todos os casos, exceto por erros pequenos nos valores dos tempos, as sub-instâncias escolhidas ou foram determinadas pelo BP em tempo menor ou em tempo aproximado (vide Tabela 4.5).

A partir dessa escolha das sub-instâncias com menor índice, portanto, vamos uni-las e medir

Parte	$ V_i $	$I(G_i)$	Tempo	LDE	Parte	$ V $	$I(G)$	Tempo	LDE
G_1	55	126	54,26	$7,81 \times 10^{-14}$	G_1^*	55	124	53,85	$1,15 \times 10^{-13}$
G_2	67	302	63,97	$1,61 \times 10^{-14}$	G_2^*	67	260	63,10	$1,11 \times 10^{-14}$
G_3	5	2	2,09	$2,81 \times 10^{-16}$	G_3^*	5	2	2,09	$2,81 \times 10^{-16}$
G_4	11	18	6,84	$8,33 \times 10^{-16}$	G_4^*	11	16	6,84	$8,95 \times 10^{-16}$
G_5	6	4	3,07	$2,61 \times 10^{-16}$	G_5^*	6	4	3,20	$3,72 \times 10^{-16}$
G_6	6	4	3,05	$4,90 \times 10^{-16}$	G_6^*	6	4	3,06	$5,06 \times 10^{-16}$
G_7	168	408	254,85	$6,93 \times 10^{-14}$	G_7^*	168	—	—	—

Tabela 4.5: Tabela com os dados de índice, tempo e LDE das partes e suas inversas.

o tempo em CPU também para executar as uniões em $M = 10.000$ vezes. A Tabela 4.6 traz os dados de tempo e de LDE. As uniões por Quaternários novamente tiveram tempo maior e, por isso, foram suprimidas para esta instância e para os dois próximos exemplos também.

União	Intervalo	Tempo	LDE	Tipo
$G_1^* \cup G_2^*$	[1, 119]	1,78	$6,04 \times 10^{-14}$	Matrizes
$G_3 \cup G_4^*$	[117, 129]	1,59	$9,24 \times 10^{-16}$	Matrizes
$G_5 \cup G_6$	[127, 135]	1,58	$4,31 \times 10^{-16}$	Matrizes
$G_1^* \cup G_2^* \cup G_3 \cup G_4^*$	[1, 129]	1,6102	$5,59 \times 10^{-14}$	Matrizes
$G_5 \cup G_6 \cup G_7$	[127, 300]	2,10	$7,25 \times 10^{-14}$	Matrizes
$G_1^* \cup G_2^* \cup G_3 \cup G_4^* \cup G_5 \cup G_6 \cup G_7$	[1, 300]	2,14	$6,48 \times 10^{-14}$	Matrizes

Tabela 4.6: Tabela com os dados de uniões com matrizes.

De todas as determinações das sub-instâncias, a que possui o tempo máximo é a sétima (que também possui o maior número de vértices). Assim, temos que o tempo de computar todas elas (em paralelo) e unir as sete estruturas selecionadas seria de $TempoGapSym = 265,66$ segundos ou, aproximadamente, 4,43 minutos. O LDE da solução final é dado por $LdeGapSym = 6,48 \times 10^{-14}$.

As representações geométricas, dadas em MATLAB, respectivamente, das soluções com o BP clássico e com o nosso método são dadas pelas Figuras 4.43 e 4.44 abaixo.

Instância Três: $n = 500$ vértices

A terceira instância que foi testada tem $n = 500$ vértices. O nome dado a ela foi **INSTANCIA-TESTE500** cuja esparsidade de sua matriz de distâncias pode ser vista na Figura 4.46.

Para medir um tempo médio, esta estrutura foi submetida à determinação pelo BP de forma completa com $M = 10.000$ vezes. Isto demandou um tempo de 702,14 segundos, ou aproximadamente, 11,7 minutos. O LDE da solução *solucaoBP* encontrada foi de $7,56 \times 10^{-14}$.

Assim, pelo algoritmo *SymSplit*, G foi dividida em quatorze partes $G_1 - G_{14}$ cujos índices dos

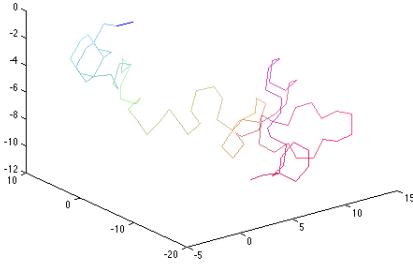


Figura 4.43: Solução com o BP clássico.

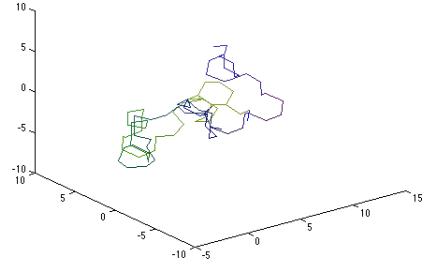


Figura 4.44: Solução com Simetrias e *Gaps*.

Figura 4.45: Representação gráfica da geometria de ambas as soluções em \mathbb{R}^3 .

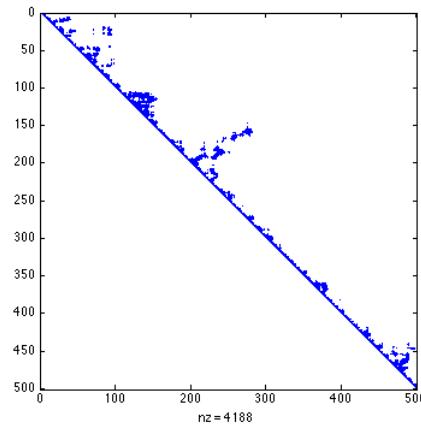


Figura 4.46: Matriz de distâncias de INSTANCIATEST500.

vértices (limitantes inferior e superior, respectivamente) são dados nas entradas das linhas de

$$split500 = \begin{bmatrix} 1 & 5 & 104 & 281 & 307 & 320 & 328 & 334 & 338 & 342 & 352 & 356 & 402 & 406 \\ 7 & 106 & 283 & 309 & 322 & 330 & 336 & 340 & 344 & 354 & 358 & 404 & 408 & 500 \end{bmatrix}^T.$$

Adicionamos os *spy* das matrizes de distâncias associadas à maior sub-instância G_3 (Figura 4.47), que provavelmente deve demorar mais tempo para ser determinada, bem como sua inversa G_3^* (Figura 4.48). Ambas têm 180 vértices.

Já na Tabela 4.7, mostramos os dados da primeira tabela para a segunda instância com os números de vértices de cada sub-instância, os índices (inclusive das inversas), os tempos de CPU e o LDE de cada realização parcial para a aplicação do BP em cada parte em $M = 10.000$ vezes.

É possível ver que a maior instância, de fato, é a que possui o maior tempo de determinação via BP. Portanto, como a sub-instância G_3^* tem um índice menor que a sub-instância G_3 (apesar de o tempo computado na determinação ser maior), então

$$\max(\text{Tempo}(G_i)) = 251, 48. \quad (4.4.4)$$

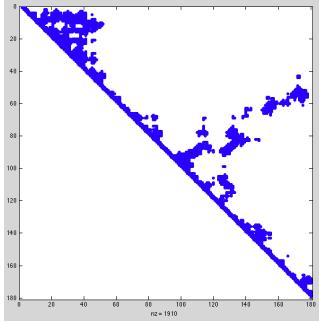


Figura 4.47: *spy* de D_{G_3}

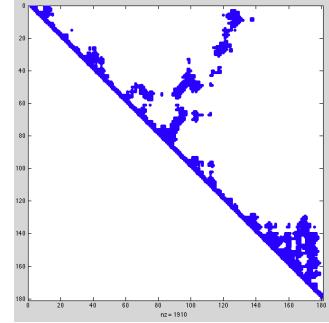


Figura 4.48: *spy* de $D_{G_3^*}$

Figura 4.49: *Spy* da maior sub-instância e de sua inversa.

Parte	$ V_i $	$I(G_i)$	Tempo	LDE	Parte	$ V $	$I(G)$	Tempo	LDE
G_1	7	6	4,46	$3,86 \times 10^{-16}$	G_1^*	7	6	4,29	$3,29 \times 10^{-16}$
G_2	102	282	120,44	$2,20 \times 10^{-14}$	G_2^*	102	252	120,70	$5,33 \times 10^{-14}$
G_3	180	918	251,13	$1,43 \times 10^{-13}$	G_3^*	180	782	251,48	$1,13 \times 10^{-13}$
G_4	29	80	25,10	$2,38 \times 10^{-14}$	G_4^*	29	96	24,57	$1,92 \times 10^{-14}$
G_5	16	34	12,04	$1,44 \times 10^{-15}$	G_5^*	16	26	12,19	$1,46 \times 10^{-15}$
G_6	11	16	6,95	$9,65 \times 10^{-16}$	G_6^*	11	18	6,92	$1,17 \times 10^{-15}$
G_7	9	12	5,82	$1,16 \times 10^{-15}$	G_7^*	9	10	6,06	$1,06 \times 10^{-15}$
G_8	7	6	4,05	$5,15 \times 10^{-16}$	G_8^*	7	6	4,10	$6,91 \times 10^{-16}$
G_9	7	6	4,05	$8,46 \times 10^{-16}$	G_9^*	7	6	4,05	$9,43 \times 10^{-16}$
G_{10}	13	20	9,86	$1,83 \times 10^{-15}$	G_{10}^*	13	18	10,04	$1,68 \times 10^{-15}$
G_{11}	7	8	3,83	$1,91 \times 10^{-15}$	G_{11}^*	7	6	3,95	$1,96 \times 10^{-15}$
G_{12}	49	222	45,27	$5,29 \times 10^{-14}$	G_{12}^*	49	272	45,18	$4,06 \times 10^{-14}$
G_{13}	7	8	3,68	$6,50 \times 10^{-15}$	G_{13}^*	7	6	3,72	$6,10 \times 10^{-16}$
G_{14}	95	246	105,98	$5,05 \times 10^{-14}$	G_{14}^*	95	226	105,57	$3,81 \times 10^{-14}$

Tabela 4.7: Tabela com os dados de índice, tempo e LDE das partes e suas inversas.

Mesmo com várias tentativas, não foi possível computar as uniões $G_5^* \cup G_6$, $G_{75}^* \cup G_8$ e $G_{11}^* \cup G_{12}$, de forma factível, por motivos os quais ainda não descobertos. Assim, as uniões feitas (com Matrizes) estão detalhadas na Tabela 4.8. Foram realizadas cada união um número de $M = 10.000$ vezes.

Portanto, computando todas as estruturas (em paralelo) e unindo as selecionadas com matrizes, o tempo de determinação seria de $TempoGapSym = 281,10$ segundos ou, aproximadamente, 4,68 minutos. Também, o LDE da solução final é dado por $LdeGapSym = 7,92 \times 10^{-14}$.

As representações geométricas, dadas em MATLAB, respectivamente, das soluções com o BP clássico e com o nosso método são dadas pelas Figuras 4.50 e 4.51 abaixo.

União	Intervalo	Tempo	LDE	Tipo
$G_1 \cup G_2^*$	[1, 106]	1, 95	$4,81 \times 10^{-14}$	Matrizes
$G_3^* \cup G_4$	[104, 309]	2, 01	$1,07 \times 10^{-13}$	Matrizes
$G_5 \cup G_6$	[307, 330]	1, 61	$1,61 \times 10^{-15}$	Matrizes
$G_7 \cup G_8$	[328, 340]	2, 16	$1,14 \times 10^{-15}$	Matrizes
$G_9 \cup G_{10}^*$	[338, 354]	2, 22	$3,32 \times 10^{-15}$	Matrizes
$G_{11} \cup G_{12}$	[352, 404]	2, 30	$4,51 \times 10^{-14}$	Matrizes
$G_{13}^* \cup G_{14}^*$	[402, 500]	2, 59	$4,22 \times 10^{-14}$	Matrizes
$G_1 \cup G_2^* \cup G_3^* \cup G_4$	[1, 309]	2, 82	$9,48 \times 10^{-14}$	Matrizes
$G_5 \cup G_6 \cup G_7 \cup G_8$	[307, 340]	2, 18	$1,61 \times 10^{-15}$	Matrizes
$G_9 \cup G_{10}^* \cup G_{11} \cup G_{12}$	[338, 404]	2, 32	$4,45 \times 10^{-14}$	Matrizes
$G_1 \cup G_2^* \cup G_3^* \cup G_4 \cup G_5 \cup G_6 \cup G_7 \cup G_8$	[1, 340]	2, 27	$8,64 \times 10^{-14}$	Matrizes
$G_9 \cup G_{10}^* \cup G_{11} \cup G_{12} \cup G_{13}^* \cup G_{14}^*$	[338, 500]	2, 46	$5,25 \times 10^{-14}$	Matrizes
$G_1 \cup G_2^* \cup G_3^* \cup G_4 \cup \dots \cup G_{11} \cup G_{12} \cup G_{13}^* \cup G_{14}^*$	[1, 500]	2, 70	$7,92 \times 10^{-14}$	Matrizes

Tabela 4.8: Tabela com os dados de uniões com matrizes.

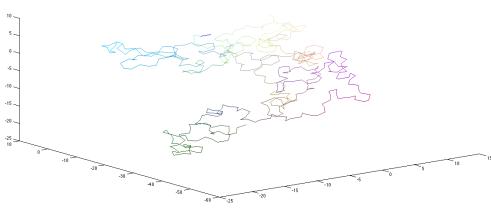


Figura 4.50: Solução com o BP clássico.

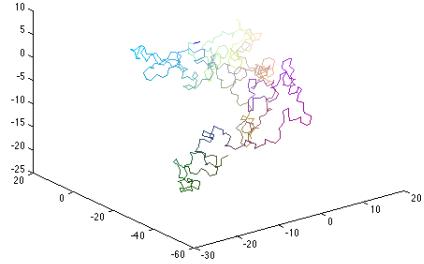


Figura 4.51: Solução com Simetrias e Gaps.

Figura 4.52: Representação gráfica da geometria de ambas as soluções em \mathbb{R}^3 .

4.4.2 Resumo

Para resumir os experimentos a fim de compará-los, vamos apresentar a Tabela abaixo.

Instancia	$ V $	M	Tempo BP	Tempo GS	LDE BP	LDE GS
instanciateste100	100	100.000	951, 49	394, 69	$9,39 \times 10^{-15}$	$1,93 \times 10^{-14}$
instanciateste300	300	10.000	414, 44	265, 66	$7,43 \times 10^{-14}$	$7,05 \times 10^{-14}$
instanciateste500	500	10.000	702, 14	281, 10	$7,56 \times 10^{-14}$	$7,92 \times 10^{-14}$

Pode-se notar que, em todas as instâncias analisadas, o método *GapSym* e a união das soluções parciais por matrizes de rotação determina uma conformação para a instância com a mesma precisão do BP clássico mas em tempo menor.

Esses testes ilustram o que apresentamos na teoria de cada método e indica para resultados

promissores. O próximo passo é testar com instâncias de proteínas reais e, também, realizar testes com computação em paralelo.

Capítulo 5

Conclusão

5.1 Contribuições da Tese

A primeira contribuição desta tese vem no sentido de mostrar a existência de particionamentos de instâncias DMDGP de modo a isolar as infactibilidades e os *backtracings* nas partes. Este resultado traz consigo a possibilidade de dar um tratamento em paralelo para o problema, realizando cada parte com o algoritmo BP em soluções parciais e sem a necessidade de fazer podas posteriores à combinação das soluções parciais. Isto é, ao se realizar uma parte, tal realização parcial não somente é localmente factível como o é de forma global na instância, eliminando a existência de informações de poda que “conectem” pares de sub-instâncias da partição. A principal ferramenta que possibilitou este entendimento foram as Simetrias inerentes à estrutura do próprio problema. Esta é a principal contribuição, a qual é bem exemplificada, está bem estabelecida e pode-se notar ao avaliar os resultados computacionais obtidos com instâncias artificialmente geradas.

A segunda contribuição, um pouco mais “fraca”, mas não menos importante, define formalmente a bidirecionalidade em que o algoritmo BP pode ser aplicado a uma instância DMDGP. Assim, aplica-se uma análise de pior-caso para escolher em qual direção o algoritmo teria mais sucesso de encontrar a solução mais rapidamente. Os experimentos computacionais também mostram que este critério é uma escolha promissora. Dissemos que esta contribuição é mais fraca por não indicar uma escolha otimizada em qualquer caso, mas apenas no pior deles.

A terceira e última contribuição desta tese é a aplicação de rotações com Quatérnios que economizam algumas operações aritméticas. Como o tempo é uma condição de eficiência que deseja-se minimizar o quanto mais, qualquer ganho de operações pode ter um bom impacto na redução temporal. Os resultados computacionais não mostraram sucesso neste ponto, provavelmente, devido a uma implementação que ainda não tirou o proveito exato da estrutura.

5.2 Trabalhos oriundos do Estudo

1. **F. Fidalgo e C. Lavor** : *Algoritmos para Problemas de Geometria Molecular*, 28º Colóquio Brasileiro de Matemática, IMPA, Julho/2011.
2. **F. Fidalgo e C. Lavor** : *Algoritmos para Resolução de Problemas de Geometria Molecular*.

lar, XVI ELAVIO - Escuela Latinoamericana de Verano en Investigación Operativa, Bento Gonçalves/RS, Fevereiro/2012.

3. **F. Fidalgo, D. Maioli, E. Abreu e C. Lavor** : *Uma Formulação Numérica para Resolução de Problemas de Geometria de Distâncias Moleculares*, XVI CLAIO - Congresso Latino-Iberoamericano de Investigación Operativa / XLIV SBPO - Simpósio Brasileiro de Pesquisa Operacional, Rio de Janeiro/RJ, Setembro/2012.
4. **F. Fidalgo e J. Rodriguez** : *Quaternion as a tool for merging multiple realization trees*, DGA13 - Workshop on Distance Geometry and Applications, Manaus/AM, Junho/2013.
5. **F. Fidalgo, D. Maioli e E. Abreu** : *Updated T Algorithm for the resolution of the Molecular Distance Geometry Problem by means of linear systems*, DGA13 - Workshop on Distance Geometry and Applications, Manaus/AM, Junho/2013.
6. **F. Fidalgo, C. Lavor, J. Rodriguez** : *Uma nova abordagem para dividir instâncias do DMDGP usando gaps*, XXXV CNMAC, Natal/RN, 2014.
7. **F. Fidalgo, J. Rodriguez** : *Exploiting symmetries in a divide-and-conquer approach for solving the DMDGP*, Many Faces of Distances Workshop, Campinas/SP, 2014.
8. **F. Fidalgo e C. Lavor** : *Deciding directions for the Branch and Prune algorithm using gaps - a worst-case analysis*, Mathematical Methods of Biophysical Phenomena Workshop, Cabo Frio/RJ, 2015.

5.3 Trabalhos Futuros

Como trabalhos futuros, temos proposto os seguintes:

- (i) Definir, ainda que heuristicamente, um modo de unir as sub-instâncias pequenas com outras a fim de ter sub-instâncias maiores e diminuir o número de processos de computação paralela necessários para a determinação, tendo em vista o tamanho da maior sub-instância.
- (ii) Fazer uma análise de caso-médio e/ou uma análise estatística a fim de encontrar um indicador que melhore ainda mais a escolha da bidirecionalidade usando os *gaps*.
- (iii) Definir (e testar), de fato, um método em paralelo para resolver instâncias do BP, visto que não fizemos isto neste trabalho que apenas traz indicações de uma proposta assim.
- (iv) Estudar a aplicabilidade deste tipo de partição de instâncias, considerando o caso em que algumas (ou todas) as distâncias são dadas em intervalos.
- (v) Buscar a generalização destas propostas para o *Discretizable Distance Geometry Problem* (DDGP) ainda mais amplo de modo a envolver outras aplicações que não moleculares.

- (vi) Em se definindo e se consolidando um “método dual” para o BP (via completamento de matrizes de distâncias), buscar o acoplamento com essas propostas já que há um apelo no sentido da “visão” matricial.
- (vii) Verificar a aplicabilidade destes métodos em aplicações como Robótica, Telecomunicações, bem como outras que seja modeladas via grafos.

Referências

- [1] A. Alfakih, A. Khandani e H. Wolkowicz. “Solving Euclidean Distance Matrix Completion Problems via Semidefinite Programming”. Em: *Computational Optimization and Applications* 12 (1999), pp. 13–30.
- [2] S. Altmann. *Rotations, Quaternions and Double Groups*. Dover Books on Mathematics, Dover Publications Inc., New York, 2005.
- [3] M. Bakonyi e C. Johnson. “The Euclidean distance matrix completion problem”. Em: *SIAM Journal on Matrix Analysis and Applications* 16 (2) (1995), pp. 646–654.
- [4] J. Belinfante e B. Kolman. *A Survey of Lie Groups and Lie Algebras with Applications and Computational Methods*. SIAM, Philadelphia, 1989.
- [5] R. Benedetti e J-J. Risler. *Real algebraic and semi-algebraic sets*. Paris: Hermann, 1990.
- [6] Instituto Biológico. *Encefalopatia Espongiforme Bovina - EEB*. http://www.biologico.sp.gov.br/artigos_ok.php?id_artigo=18.
- [7] P. Biswas, K. Toh e Y. Ye. “A distributed SDP approach for large-scale noisy anchor-free graph realization with applications to molecular conformation”. Em: *SIAM Journal on Scientific Computing* 30 (3) (2008), pp. 1251–1277.
- [8] L. Blumenthal. *Theory and Applications of Distance Geometry*. Oxford University Press, Oxford, 1953.
- [9] A. Brown. “On the theory of isomeric compounds”. Em: *Transactions of the Royal Society of Edinburgh* 23 (1864), pp. 707–719.
- [10] A. Buluc, H. Meyerhenk, I. Safro, P. Senders e C. Schulz. “Recent Advances in Graph Partitioning”. Em: *eprint arXiv:1311.3144* (2013).
- [11] A. Cayley. “A theorem in the Geometry of Position”. Em: *Cambridge Mathematical Journal* II (1841), pp. 267–271.
- [12] C. Lavor. “On generating instances for the molecular distance geometry problem”. Em: *Global Optimization: from implementation to theory*. Ed. por L. Liberti e N. Maculan. Springer, New York, v. 84, 2006, pp. 405–414.
- [13] J.W. Cooley e J.W. Turkey. “An algorithm for the machine calculation of complex Fourier series”. Em: *Mathematics of Computations* 19 (1965), pp. 297–301.
- [14] T. Cormen, C. Leiserson, R. Rivest e C. Stein. *Introduction to Algorithms*. MIT Press, 1990.

- [15] T. E. Creighton. *Proteins: Structures and Molecular Properties*. W. H. Freeman e Company, New York, 1993.
- [16] G. Crippen. “A Novel Approach to Calculation of Conformation: Distance Geometry”. Em: *Journal of Computational Physics* 24 (1977), pp. 96–107.
- [17] G. Crippen. *Distance Geometry and Conformational Calculations*. Research Studies Press, Wiley, New York, 1981.
- [18] G. Crippen. “Distance Geometry Approach for Rationalizing Binding Data”. Em: *Journal of Medicinal Chemistry* 22 (1979), pp. 988–997.
- [19] G. Crippen. “Distance Geometry for Realistic Models”. Em: *Distance Geometry: Theory, Methods and Applications*. Ed. por A. Mucherino, C. Lavor, Liberti. L. e N. Maculan. Springer, New York, U.S.A.
- [20] G. Crippen. “Global Optimization and Polypeptide Conformation”. Em: *Journal of Computational Physics* 18 (1975), pp. 224–231.
- [21] G. Crippen e T. Havel. *Distance Geometry and Molecular Conformation*. Research Studies Press, Wiley, New York, 1988.
- [22] M. Curtis. *Matrix Groups*. Springer-Verlag, New York, 1984.
- [23] Q. Dong e Z. Wu. “A Geometric Build-up Algorithm for Solving the Molecular Distance Geometry Problem with Sparse Distance Data”. Em: *Journal of Global Optimization* 26 (2003), pp. 321–333.
- [24] Q. Dong e Z. Wu. “A linear-time algorithm for solving the Molecular Distance Geometry Problem with exact interatomic distances”. Em: *Journal of Global Optimization* 22 (2002), pp. 365–375.
- [25] L. Dorst, S. Mann e D. Fontijne. *Geometric Algebra for Computer Science*. Morgan Kauffman Publishers Inc., 2007.
- [26] T. Eren, D. K. Goldenberg, W. Whiteley, Y. R. Yang, A. R. Morse, B. D. O. Anderson e P. N. Belhumeur. “Rigidity, computation and randomization in network localization”. Em: *IEEE Infocom 2004 Proceedings*. 2004, pp. 2673–2684.
- [27] R. Farebrother. “Three theorems with applications to Euclidean distance matrices”. Em: *Linear Algebra and Its Applications* 95 (1987), pp. 11–16.
- [28] F. Fidalgo. “Algoritmos para Problemas de Geometria Molecular”. Diss. de mestrado. UNICAMP, 2011.
- [29] F. Fidalgo. *Lavor Instances for the DMDGP: some features and behavior*. Rel. téc. Universidade Estadual de Campinas, UNICAMP, 2015.
- [30] F. Fidalgo e C. Lavor. “Deciding directions for the Branch & Prune Algorithm using gaps - a worst-case analysis”. Em: *Proceedings of the Mathematical Methods of Biophysical Phenomena Workshop, Cabo Frio, Brazil (a ser publicado)*. 2015.

- [31] F. Fidalgo, Carlile Lavor e J. Rodriguez. “Uma nova abordagem para dividir instâncias do Discretizable Molecular Distance Geometry Problem usando gaps”. Em: *Anais do XXXV Congresso Nacional de Matemática Aplicada e Computacional, Natal, Brasil (a ser publicado)*. 2014, pp. 1–7.
- [32] F. Fidalgo, D. Maioli, E. Abreu e C. Lavor. “Uma formulação numérica para resolução de Problemas de Geometria de Distâncias Moleculares”. Em: *Proceedings of XVI Congresso Latino-Iberoamericano de Investigación Operativa / XLIV Simpósio Brasileiro de Pesquisa Operacional, Rio de Janeiro*. 2012, pp. 1–11.
- [33] F. Fidalgo e J. Rodriguez. “Exploiting symmetries in a divide-and-conquer approach for solving DMDGP”. Em: *Proceedings of the Many Faces of Distances Workshop, Campinas Brazil (a ser publicado)*. 2014, pp. 1–3.
- [34] F. Fidalgo e J. Rodriguez. “Quaternions as a tool for merging multiple realization trees”. Em: *Proceedings of the Workshop on Distance Geometry and Applications, Manaus, Brazil*. 2013, pp. 1–5.
- [35] J. Gower. “Properties of Euclidean and non-Euclidean distance matrices”. Em: *Linear Algebra and Its Applications* 67 (1985), pp. 1–97.
- [36] R.L. Graham, D.E Knuth e O. Patashnik. *Concrete Mathematics*. Addison Wesley, 1989.
- [37] I. Hargittai e M. Hargittai. *Symmetry Through the Eyes of a Chemist*. Plenum Press, 2nd Edition, New York, 1995.
- [38] T. Havel. “Distance Geometry”. Em: *Encyclopedia of Magnetic Resonance*. Ed. por D. Grant e R. Harris. Wiley, New York, 1995, pp. 1701–1710.
- [39] T. Havel, I. Kuntz e G. Crippen. “The Theory and Practice of Distance Geometry”. Em: *Bulletin of Mathematical Biology* 45 (1983), pp. 665–720.
- [40] B. Hendrickson. “The molecule problem: Exploiting structure in global optimization”. Em: *SIAM Journal on Optimization* 5 (1995), pp. 835–857.
- [41] H-X. Huang, Z-A. Liang e P. Pardalos. “Some Properties for the Euclidean Distance Matrix and Positive Semidefinite Matrix Completion Problems”. Em: *Journal of Global Optimization* 25 (2003), pp. 3–21.
- [42] A. Ihde. *The development of modern Chemistry*. Harper e Row, 1964.
- [43] C. Johnson e P. Tarazaga. “Connections between the real positive semidefinite and distance matrix completion”. Em: *Linear Algebra and Its Applications* 223/224 (1995), pp. 375–391.
- [44] A. Karatsuba e Y. Ofman. “Multiplication of Many-Digital Numbers by Automatic Computers”. Em: *Proceedings of the USSR Academy of Sciences* 145. 1962, pp. 293–294.
- [45] Lord Kelvin. *Baltimore Lectures on Molecular Dynamics and the Wave Theory of Light*. Cambridge University Press, London, 1904.
- [46] D. Knuth. *The Art of Computer Programming: Sorting and Searching, Volume III, 2nd edition*. Addison Wesley, 1998.

- [47] J. B. Kuipers. *Quaternions and Rotation Sequences*. Princeton University Press, Princeton, New Jersey, 1999.
- [48] I. Kuntz, G. Crippen e P. Kollman. “Application of Distance Geometry to Protein Tertiary Structure Calculations”. Em: *Biopolymers* 18 (1979), pp. 939–957.
- [49] I. Kuntz, J. Thomason e C. Oshiro. “Distance Geometry”. Em: *Methods in Enzymology* 177 (1989), pp. 159–204.
- [50] M. Laurent. “Matrix completion problems”. Em: *Encyclopedia of Optimization*. Ed. por C. Floudas e P. Pardalos. Springer, New York, 2nd edition, 2009, pp. 1967–1975.
- [51] C. Lavor, J. Lee, A. Lee-St.John, L. Liberti, A. Mucherino e N. Maculan. “Discretization orders for distance geometry problems”. Em: *Optimization Letters* 6 (2012), pp. 783–796.
- [52] C. Lavor e L. Liberti. *Um Convite à Geometria de Distâncias*. CNMAC - SBMAC, 2014.
- [53] C Lavor, L. Liberti, N. Maculan e A. Mucherino. “Recent advances on the Discretizable Molecular Distance Geometry Problem”. Em: *European Journal of Operational Research* 219 (2012), pp. 698–706.
- [54] C. Lavor, L. Liberti, N. Maculan e A. Mucherino. “The discretizable molecular distance geometry problem”. Em: *Computational Optimization and Applications* 52 (2012), pp. 115–146.
- [55] L. Liberti, C. Lavor, J. Alencar e G. Abud. “Counting the number of solutions of K DMDGP instances”. Em: *Lecture Notes on Computer Science* 8085 (2013), pp. 224–230.
- [56] L. Liberti, C Lavor e N. Maculan. “A Branch-and-Prune Algorithm for the Molecular Distance Geometry Problem”. Em: *International Transactions in Operational Research* 15 (2008), pp. 1–17.
- [57] L. Liberti, C. Lavor, N. Maculan e A. Mucherino. “Euclidean Distance Geometry and Applications”. Em: *SIAM Review* 56 (2014), pp. 3–69.
- [58] L. Liberti, C. Lavor e A. Mucherino. “The discretizable molecular distance geometry problem seems easier on proteins”. Em: *Distance Geometry: Theory, Methods and Applications*. Ed. por A. Mucherino, C. Lavor, L. Liberti e N. Maculan. Springer, New York, U.S.A., 2013.
- [59] L. Liberti, C Lavor, A. Mucherino e N. Maculan. “Molecular distance geometry methods: from continuous to discrete”. Em: *International Transactions in Operational Research* 18 (2010), pp. 33–51.
- [60] L. Liberti, B. Masson, C. Lavor e A. Mucherino. “A branch-and-prune trees with bounded width”. Em: *Proceedings of the 10th Cologne-Twente Workshop on Graphs and Combinatorial Optimization (CTW11), Rome, Italy*. 2011, pp. 189–193.
- [61] L. Liberti, B. Masson, J. Lee, C. Lavor e A. Mucherino. “On the number of realizations of certain Hennenberg graphs arising in protein conformation”. Em: *Discrete Applied Mathematics, accepted* (2014).

- [62] L. Liberti, B. Masson, J. Lee, C. Lavor e A. Mucherino. “On the number of solutions of the discretizable molecular distance geometry problem”. Em: *Proceedings of the 5th Annual International Conference on Combinatorial Optimization and Applications (COCOA11), volume 6831 of Lecture Notes in Computer Science, Springer*. 2011, pp. 322–342.
- [63] R. S. Lima. “Descrição de Rotações: prós e contras na teoria e na prática”. Diss. de mestrado. UNICAMP, 2007.
- [64] K. Menger. “New foundation of Euclidean Geometry”. Em: *American Journal of Mathematics* 53 (1931), pp. 721–745.
- [65] K. Menger. “Untersuchungen über allgemeine metrik.” Em: *Mathematisch Annalen* 103 (1928), pp. 466–501.
- [66] J. Moré e Z. Wu. “Distance Geometry Optimization for Protein Structures”. Em: *Journal of Global Optimization* 15 (1999), pp. 219–234.
- [67] J. Moré e Z. Wu. “Global continuation for distance geometry problems”. Em: *SIAM Journal of Optimization* 7 (1997), pp. 814–846.
- [68] A. Mucherino, C. Lavor e L. Liberti. “Exploiting symmetry properties of the Discretizable Molecular Distance Geometry Problem”. Em: *Journal of Bioinformatics and Computational Biology* 10 (3) (2012), pp. 1–15.
- [69] A. Mucherino, C. Lavor, L. Liberti e E.-G. Talbi. “A parallel version of the Branch-and-Prune algorithm for the molecular distance geometry problem”. Em: *IEEE Conference Proceedings, ACS/IEEE International Conference on Computer Systems and Applications (AICCSA10), Hammamet, Tunisia*. 2010, pp. 1–6.
- [70] A. Mucherino, L. Liberti e C. Lavor. “MD-Jeep: an implementation of a Branch-and-Prune Algorithm for Distance Geometry Problems”. Em: *Lecture Notes in Computer Science* 6327 (2010), pp. 186–197.
- [71] G. J. Mulder. “Over Proteine en hare Verbindingen en Ontledingsproducten”. Em: *Natuur-en scheikundig Archief* 6 (1838), pp. 87–162.
- [72] G. J. Mulder. “Sur le composition de quelques substances animales”. Em: *Bulletin des sciences physiques et naturelles en Néerlande* 1 (1838), p. 104.
- [73] R. Murray, Z. Li e S. Sastry. *A Mathematical Introduction to Robotic Manipulation*. CRC Press, First Edition, 1994.
- [74] D. Nelson, A. Lehninger e M. Cox. *Lehninger Principles of Biochemistry*. W.H. Freeman, New York, 2008.
- [75] A. Neumaier. “Molecular modeling of proteins and mathematical prediction of protein structure”. Em: *SIAM Review* 39 (1997), pp. 407–460.
- [76] P. Nucci, L. Nogueira e C. Lavor. *Heurísticas para o problema molecular de geometria de distâncias aplicado a proteínas*. Undergraduate Dissertation, Universidade Federal Fluminense.

- [77] P. Nucci, L. Nogueira e C. Lavor. “Solving the Discretizable Molecular Distance Geometry Problem by multiple realization trees”. Em: *Distance Geometry: Theory, Methods and Applications*. Ed. por A. Mucherino, C. Lavor, Liberti. L. e N. Maculan. Springer, New York, U.S.A.
- [78] M. Petitjean. “Chirality and Symmetry Measures: A Transdisciplinary Review”. Em: *Entropy* 5 (2003), pp. 271–312.
- [79] A. T. Philips, J. B. Rosen e V. H. Walke. “Molecular structure determination by convex underestimation of local energy minima”. Em: *DIMACS Series in Discrete Mathematics and Theoretical Computer Science* 23 (1996), pp. 181–198.
- [80] A. Pogorelov. *Geometry*. MIR, Moscow, 1987.
- [81] I. Porteous. *Clifford Algebras and the Classical Groups*. Cambridge University Press, 1995.
- [82] G. Ramachandran, A. Kolaskar, C. Ramakrishnan e V. Sasisekharan. “The mean geometry of the peptide unit from crystal structure data”. Em: *Biochimica et Biophysica Acta* 359 (1974), pp. 298–302.
- [83] J. B. Saxe. “Embeddability of weighted graphs in k-space is strongly NP-hard”. Em: *Proceedings of 17th Allerton Conference in Communications, Control and Computing, Monticello, IL*. 1979, pp. 480–489.
- [84] H. Scheraga. “Theoretical and Experimental Studies of Conformation of Polypeptides”. Em: *Chemical Reviews* 71 (1971), pp. 195–217.
- [85] T. Schlick. *Molecular Modeling and Simulation: An Interdisciplinary Guide*. Springer, 2010.
- [86] I. Schoenberg. “Remarks to Maurice Frechet’s article Sur la définition axiomatique d’une classe d’espaces distanciés vectoriellement applicable sur l’espace de Hilbert”. Em: *Annals of Mathematics* 36 (1935), pp. 724–732.
- [87] M. Souza. “Suavização hiperbólica aplicada à otimização de Geometria Molecular”. Tese de doutorado. COPPE - UFRJ, 2010.
- [88] V. Strassen. “Gaussian Elimination is not optimal”. Em: *Numerische Mathematik* 13 (1969), pp. 354–356.
- [89] J. Suter. *Geometric Algebra Primer*. 2003.
- [90] J.J. Sylvester. “Chemistry and Algebra”. Em: *Nature* 17 (1877), pp. 284–284.
- [91] T. Toffolo e M.A. Carvalho. *Divisão e Conquista*. http://www.decom.ufop.br/toffolo/site_media/uploads/2011-1/bcc402/slides/08._divisao_e_conquista.pdf.
- [92] H.B. Vickery. “The Origin of the word Protein”. Em: *Yale Journal of Biology and Medicine* 22 (5 1950), pp. 387–393.
- [93] Free Encyclopedia Wikipedia. *Divide-and-Conquer Algorithms*. en.wikipedia.org/wiki/Divide_and_conquer_algorithm.
- [94] D. Wu e Z. Wu. “An updated geometric build-up algorithm for solving the molecular distance geometry problems with sparse distance data”. Em: *Journal of Global Optimization* 37 (2007), pp. 661–673.

- [95] Y. Yemini. "Some theoretical aspects of position location problems". Em: *Proceedings of the 20th Annual Symposium on the Foundations of Computer Science, IEEE*. 1979, pp. 1–8.
- [96] Y. Yemini. "The positioning problem - a draft of an intermediate summary". Em: *Proceedings of the Conference on Distributed Sensor Networks, Carnegie Mellon University, Pittsburg*. 1978, pp. 137–145.

Apêndice A

Ordens em Grafos

A discretização do MDGP é realizada através de uma ordem total no conjunto de vértices de suas instâncias. Mas, o que significa "ordenar" os vértices de um grafo?

A.1 Conceitos Preliminares

Antes de tudo, há que se introduzir uma nomenclatura básica de conceitos preliminares, o que é baseado em Liberti et al. [56, 57].

Definição A.1.1. Seja $G = (V, E)$ um grafo e $U \subseteq V$. Define-se o subconjunto de **arestas induzidas** por U como $E[U] = \{\{u, v\} \in E : u, v \in U\}$.

Definição A.1.2. Seja $G = (V, E)$ um grafo e $H = (U, F)$ tal que $F = E[U]$. Diz-se que H é um subgrafo de G induzido por U .

Definição A.1.3. Um grafo $G = (V, E)$ é completo ou um *clique* se $E = \{\{u, v\} : \forall u, v \in E\}$.

Definição A.1.4. Dado um vértice $v \in V$, o conjunto $N(v) = \{u \in V : \{u, v\} \in E\}$ é chamado de **Vizinhança de v** ou *conjunto de vértices adjacentes a v* .

A definição para mergulhos de grafos em espaços Euclidianos deve-se a Hendrickson [40].

Definição A.1.5 (Realização). Dado um grafo $G = (V, E)$, uma realização de G em \mathbb{R}^3 comprehende em um mergulho isométrico $x : G \longrightarrow \mathbb{R}^3$.

A.2 Ordem em V

Se o grafo, de um MDGP, possui propriedades como rigidez, o número de imersões é finito, tornando a busca combinatorial, como deseja-se.

A priori, como rigidez é difícil de determinar, há que se ter outras condições mais fáceis de se verificar e a maioria delas lida com a existência de uma ordem em V com propriedades topológicas especiais [57].

Tais ordens nos vértices estão intimamente ligadas à natureza da aplicação. Com a existência desta, cada vértice é mergulhado no espaço Euclídeo seguindo a ordem existente. Como veremos mais adiante, o que é o caso de nossa aplicação, problemas moleculares (e em especiais protéicos, por ter uma geometria interessante) possuem ordem nos vértices relativos à cadeia principal da molécula.

Seja $G = (V, S, d)$ um grafo simples, ponderado e não-orientado, que representa uma molécula M . Abaixo, necessitamos apresentar um pouco ainda de nomenclatura para definir esta ordem.

Definição A.2.1. Dados uma ordem total $<$ no conjunto de vértices V e um elemento $v \in V$, o conjunto $\gamma(v) = \{u \in V : u < v\}$ é chamado de *Conjunto de Antecessores de v* com relação a $<$.

Definição A.2.2. Dados uma ordem total $<$ no conjunto de vértices V e um elemento $v \in V$, o conjunto $\eta(v) = \{u \in V : v < u\}$ é chamado de *Conjunto de Sucessores de v* com relação a $<$.

Definição A.2.3. Dado um vértice $v \in V$, seu *posto* é definido por $\rho(v) = |\gamma(v)|+1$ e representa a quantidade de vértices até v , inclusive, seguindo a ordem $<$.

Unindo as definições de *vizinhança* e *antecessor*, obtém-se o conceito seguinte.

Definição A.2.4. Denota-se o conjunto dos vértices antecessores adjacentes a v como $N(v) \cap \gamma(v)$.

Algumas ordens em vértices são enumeradas em [57], das quais citaremos algumas.

- (i) **Perfect Elimination Order** (PEO) é uma ordem tal que, para cada $v \in V$, o subgrafo $G[N(v) \cap \eta(v)]$ é uma *clique* em G . A Figura A.1 exemplifica um grafo que possui esta ordem.
- (ii) Uma **Ordem Discretization Vertex Order Problem** (DVOP) em um grafo $G = (V, E)$, com respeito a um inteiro $1 \leq K \leq n$, onde $n = |G|$, é uma ordem no conjunto de vértices V de modo a satisfazer os seguintes axiomas:
 - (a) os primeiros K vértices induzem um clique em G e
 - (b) cada vértice $v \in V$ de posto $\rho(v) > K$ é tal que $|N(v) \cap \gamma(v)| \geq K$.

A Figura A.1 possui um exemplo de um grafo com esta ordem em seus vértices (para $K = 2$).

- (iii) Uma **Ordem Henneberg tipo I** é um caso particular da Ordem DVOP onde, para cada vértice v tal que $\rho(v) > K$, vale a igualdade $|N(v) \cap \gamma(v)| = K$.

A Figura A.1, também, possui uma instância desta ordem (para $K = 2$).

- (iv) Uma **Ordem K -Trilateration** corresponde a uma Ordem DVOP onde
 - (a) os primeiros $K + 1$ vértices induzem um clique em G e
 - (b) para cada vértice $v \in V$ com posto $\rho(v) > K + 1$, vale a desigualdade $|N(v) \cap \gamma(v)| \geq K + 1$.

A Figura A.2 retrata uma instância desta ordem (para $K = 2$).

- (v) Uma **Ordem Discretizable Distance Geometry Problem** (DDGP) é uma ordem DVOP onde, para cada vértice $v \in V$ com posto $\rho(v) > K$, existe um subconjunto de vértices $U_v \subseteq N(v) \cap \gamma(v)$ com $|U_v| = K$ e $G[U_v]$ é um clique em G .

A Figura A.2, também, ilustra um exemplo desta ordem (para $K = 2$).

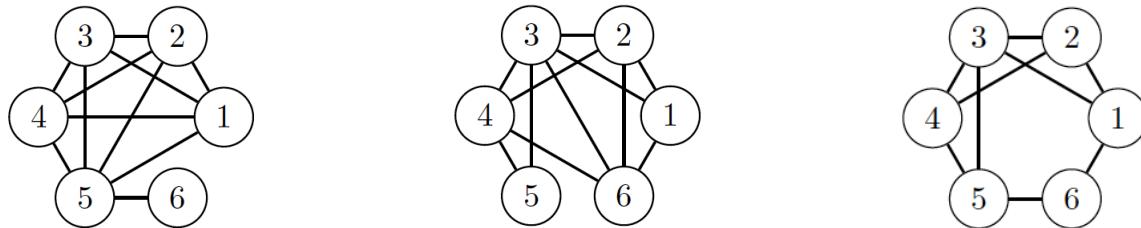


Figura A.1: Exemplos de grafos sob as Ordens PEO, DVOP e Henneberg Tipo I em seus vértices, extraídos de Liberti et al. [57].



Figura A.2: Exemplos das Ordens K -Trilateration e DDGP, extraídos de [57].

Apêndice B

Matrizes e Rotações

Neste capítulo, exploraremos conceitos algébricos envolvendo matrizes, como Grupos e Álgebras de Matrizes, com o objetivo de estudar a definição mais arbitrária de matrizes de rotação (independentemente do referencial). A teoria básica analisada foi apresentada em Curtis [22].

B.1 Álgebra de Matrizes e Grupos de Matrizes Clássicos

Sabe-se, da Álgebra Linear (AL) básica, que as matrizes com entradas no corpo $\mathbb{K}(\mathbb{R}, \mathbb{C}$ ou \mathbb{H}) constituem um espaço vetorial. Mais do que isso, é possível definir uma estrutura de *álgebra* dessas matrizes em relação ao produto matricial usual.

Definição B.1.1. O conjunto $M_n(\mathbb{K})$, das matrizes de dimensão n com entradas em \mathbb{K} , munido de soma, produto por escalar e produto matricial associativo, que tem por identidade à matriz I_n , é chamada de **Álgebra de Matrizes**.

Agora, como a definição de *unidade* de uma álgebra traz consigo o conceito de inversibilidade, então uma matriz $A \in M_n(\mathbb{K})$ é uma unidade se for inversível, isto é, se $\det(A) \neq 0$. Além disso, o conjunto dessas unidades possui uma estrutura algébrica de grupo, dada pela seguinte definição.

Definição B.1.2 (Grupo Linear Geral). O subgrupo formado pelas unidades do grupo $M_n(\mathbb{K})$ é chamado de **Grupo Linear Geral** e é denotado por $GL(n, \mathbb{K})$.

Segundo Curtis [22], $A \in GL(n, \mathbb{K})$ se, e somente se, $\det(A) \neq 0$, e representa um automorfismo de \mathbb{K}^n . Assim, seguem alguns exemplos que caracterizam este grupo para algumas dimensões.

- $GL(1, \mathbb{K}) = \mathbb{K} \setminus \{0\}$, pois cada elemento não-nulo deste corpo representa uma unidade.
- $GL(2, \mathbb{K})$ corresponde ao conjunto de unidades do espaço das matrizes de dimensão 2. Portanto, a caracterização deste grupo é dada por

$$GL(2, \mathbb{K}) = \left\{ \begin{bmatrix} a & b \\ c & d \end{bmatrix} : a, b, c, d \in \mathbb{K}, ad - bc \neq 0 \right\}.$$

A partir deste grupo, então, define-se alguns subgrupos que possuem definições e propriedades interessantes a fim de evoluirmos até uma estrutura algébrica para as matrizes de rotações.

Definição B.1.3 (Belinfante [4]). Ao subgrupo $SL(n, \mathbb{K}) = \{A \in GL(n, \mathbb{K}) : \det A = 1\} \subset GL(n, \mathbb{K})$ chama-se **Grupo Linear Especial**.

Municando, agora, o espaço \mathbb{K}^n de um produto interno, então verifica-se o seguinte resultado.

Proposição B.1.1 (Curtis [22]). Seja \mathbb{K}^n um espaço vetorial munido de um produto interno não-degenerado $\langle \cdot, \cdot \rangle$. Para vetores $x, y \in \mathbb{K}^n$ e uma matriz $A \in M_n(\mathbb{K})$, vale a igualdade $\langle xA, y \rangle = \langle x, yA^* \rangle$, onde A^* é a matriz transposta e conjugada associada a A .

A partir desse resultado, portanto, pode-se definir um novo subgrupo de matrizes que será devidamente importante nesta teoria, formado por matrizes que preservam este produto interno.

Definição B.1.4 (Curtis [22]). O conjunto $\mathcal{O}(n, \mathbb{K}) = \{A \in M_n(\mathbb{K}) : \langle Ax, Ay \rangle = \langle x, y \rangle\}$, definido sobre um espaço vetorial \mathbb{K}^n , munido do respectivo produto interno $\langle \cdot, \cdot \rangle$, forma um grupo com a multiplicação de matrizes. O nome deste grupo depende do corpo \mathbb{K} no qual está definido, como segue.

- $\mathcal{O}(n) = \mathcal{O}(n, \mathbb{R})$ é chamado de **Grupo Ortogonal**, o qual contém as matrizes ortogonais (A tais que $AA^T = A^TA = I$) as quais preservam os comprimentos de vetores,
- $\mathcal{U}(n) = \mathcal{O}(n, \mathbb{C})$ é chamado de **Grupo Unitário**, o qual contém as matrizes Hermitianas (B tais que $BB^* = B^*B = I$) e
- $Sp(n) = \mathcal{O}(n, \mathbb{H})$ é chamado de **Grupo Simplético**.

O próximo resultado fornece uma característica das matrizes dos grupos ortogonal e unitário, essencial para a definição do conceito de um novo grupo de matrizes. Tome $\mathbb{K} = \mathbb{R}$ ou \mathbb{C} .

Proposição B.1.2. Se $A \in \mathcal{O}(n, \mathbb{K})$, então $(\det A)\overline{(\det A)} = 1$.

A partir disso, define-se dois grupos especiais que modelam as rotações.

Definição B.1.5. O **Grupo Especial Ortogonal** e o **Grupo Especial Unitário** são definidos, respectivamente, por $SO(n) = \{A \in \mathcal{O}(n) : \det A = 1\}$ e $SU(n) = \{A \in \mathcal{U}(n) : \det A = 1\}$.

Com isso, afirma-se que $SO(n) = O(n) \cap SL(n, \mathbb{C})$ e $SU(n) = U(n) \cap SL(n, \mathbb{C})$ [4].

Mais três subespaços de $M_n(\mathbb{K})$ são estudados, para complementarem este estudo: $so(n)$ é o subespaço de $M_n(\mathbb{R})$ que contém as matrizes anti-simétricas, $su(n) \subset M_n(\mathbb{C})$ contém as matrizes anti-hermitianas e $sp(n) \subset M_n(\mathbb{H})$ contém as matrizes anti-simpléticas.

B.2 A exponencial de uma matriz

Nesta seção, lidamos com matrizes sobre \mathbb{R} . Matrizes em \mathbb{C} e \mathbb{H} têm estrutura análoga [22].

Definição B.2.1. Seja $A \in M_n(\mathbb{R})$. Define-se a **Exponencial da Matriz** A como a série

$$e^A = \sum_{k=0}^{\infty} \frac{A^k}{k!} = I + A + \frac{A^2}{2} + \frac{A^3}{3!} + \dots, \quad (\text{B.2.1})$$

onde A^k é o produto em k vezes da matriz A , segundo o produto de matrizes da álgebra $M_n(\mathbb{R})$.

Para [22], a série (B.2.1) converge se todas as n^2 séries que compoem as entradas de e^A convergem. As demonstrações dos resultados seguintes se encontram em [22].

Proposição B.2.1. Para qualquer $A \in M_n(\mathbb{R})$, a série e^A é convergente.

Proposição B.2.2. Se $A \in M_n(\mathbb{R})$, então a matriz e^A é não-singular.

Essa exponencial induz um *Mapa Exponencial* $\exp : M_n(\mathbb{R}) \longrightarrow M_n(\mathbb{R})$, que está bem definido pela Proposição B.2.1. Para ele, vale o seguinte resultado, que decorre da Proposição B.2.2.

Corolário B.2.1. A imagem da álgebra $M_n(\mathbb{R})$ pelo mapa exponencial \exp também tem uma estrutura algébrica de grupo, já que satisfaz a igualdade $\exp(M_n(\mathbb{R})) = GL(n, \mathbb{R})$.

O próximo resultado caracteriza a aplicação deste mapa ao subespaço $so(n)$, das matrizes anti-simétricas. Sua fácil prova está em Curtis [22].

Proposição B.2.3. Se $A \in so(n)$, então $e^A \in \mathcal{O}(n)$.

Logo, este mapa transforma as matrizes anti-simétricas em ortogonais $\exp(so(n)) \subseteq \mathcal{O}(n)$.

Iremos enunciar o próximo lema, já que seu uso pode ser de grande utilidade ao realizar cálculos com exponenciais de matrizes, argumento usado também por Curtis [22].

Lema B.2.1. Se $A, B \in M_n(\mathbb{K})$ de modo que B é não-singular, então vale a igualdade

$$e^{BAB^{-1}} = Be^A B^{-1}.$$

B.3 O Grupo de Rotações - $SO(3)$

Nosso objetivo, ao estudar tais grupos, é obter representações para matrizes de rotação em seu espaço devido. No caso de \mathbb{R}^3 , estas estão intimamente ligadas a $SO(3)$, como veremos a seguir.

Toda rotação é uma transformação de simetria e deve preservar normas de vetores em \mathbb{R}^3 e ângulos entre eles [2]. Logo, dados $u, v \in \mathbb{R}^3$ e uma matriz A que faz esta transformação, então

$$\langle u, v \rangle = \langle Au, Av \rangle \implies v^T u = (Av)^T Au = v^T A^T Au = v^T (A^T A) u,$$

o que implica necessariamente que $A^T A = I$. Portanto, A é uma matriz ortogonal.

Nosso interesse é estudar essas matrizes cujo espectro $\lambda(A) \subset \mathbb{R}^+$, isto é, matrizes ortogonais com determinante positivo de $SO(3)$. As matrizes ortogonais especiais, de forma geral, compoem este grupo e são chamadas de **Matrizes de Rotação**. Por consequência, $SO(n)$ é chamado de **Grupo de Rotação**, como se vê nas referências [4, 22, 81]. Mais ainda, Porteous [81] chama tais transformações de *automorfismos ortogonais especiais*, e afirma que estes preservam a orientação do espaço, explicitando o caráter das rotações próprias [2] que queremos para \mathbb{R}^3 .

B.3.1 Matriz de Rotações via Fórmula de Rodrigues

Toda rotação R em $SO(3)$ é uma transformação ortogonal que necessita de um vetor $w \in \mathbb{R}^3$, a servir de eixo para rotacionar os vetores em \mathbb{R}^3 , e um ângulo θ . Pode-se escrever R em função de w e θ , feito através da forma matricial da chamada **Fórmula de Rodrigues**, em referência ao matemático francês *B. Olinde Rodrigues (1795 - 1851)*. Esta fórmula pode ser derivada de uma série de resultados, utilizando a exponencial de uma matriz, como feito em Murray et. al [73].

A fim de amparar o seu uso em rotações gerais, vamos comentar as idéias principais.

Considere a velocidade angular relativa a um ponto q que esteja em uma partícula que se quer rotacionar. Ao rotacionarmos este corpo em uma velocidade constante ao redor do eixo gerado pelo vetor w , esta velocidade pode ser descrita como a equação diferencial linear tempo-invariante

$$\frac{dq(t)}{dt} = w \times q(t), \quad (\text{B.3.1})$$

sendo t o tempo [73], ou o ângulo de rotação. A Equação (B.3.1) pode ser escrita como

$$w \times q(t) = \begin{bmatrix} w_2 q_3(t) - w_3 q_2(t) \\ w_3 q_1(t) - w_1 q_3(t) \\ w_1 q_2(t) - w_2 q_1(t) \end{bmatrix} = \begin{bmatrix} 0 & -w_3 & w_2 \\ w_3 & 0 & -w_1 \\ -w_2 & w_1 & 0 \end{bmatrix} \begin{bmatrix} q_1(t) \\ q_2(t) \\ q_3(t) \end{bmatrix}.$$

Denotando esta matriz de coeficientes do produto vetorial como \hat{w} , já que possui zeros na diagonal principal e combinações dos elementos de w , então a Equação (B.3.1) pode ser escrita como

$$\frac{dq(t)}{dt} = \hat{w}q(t). \quad (\text{B.3.2})$$

Integrando (B.3.2), em relação ao tempo, temos que q pode escrita como $q(t) = e^{\hat{w}t}q_0$, onde $q(0)$ é a posição inicial em relação a $t = 0$ [73] e $e^{\hat{w}t}$ é a exponencial da matriz $\hat{w}t$. Ou seja, após t unidades de tempo, $q(t)$ é a posição do objeto que inicialmente estava posicionado em q_0 . A matriz $R = e^{\hat{w}t}$ foi quem realizou esta transformação. Agora, segundo a definição de exponencial de uma matriz, dada pela Equação (B.2.1), temos que

$$e^{\hat{w}t} = I + \hat{w}t + \frac{(\hat{w}t)^2}{2!} + \frac{(\hat{w}t)^3}{3!} + \dots \quad (\text{B.3.3})$$

Seguindo este desenvolvimento, se quer realizar a rotação de um vetor em torno de um eixo w em um ângulo θ , a matriz que realiza tal transformação é dada por $R_{w,\theta} = e^{\hat{w}\theta}$, onde \hat{w} é dada por

$$\hat{w} = \begin{bmatrix} 0 & -w_3 & w_2 \\ w_3 & 0 & -w_1 \\ -w_2 & w_1 & 0 \end{bmatrix}.$$

Esta matriz é, claramente, anti-simétrica, já que $\hat{w}^T = \hat{w}$. Logo, $\hat{w} \in so(3)$.

Há um resultado preliminar, em [73], que caracteriza potências de matrizes anti-simétricas.

Lema B.3.1. Dada a matriz $\hat{a} \in so(3)$, valem as relações

$$(i) \hat{a}^2 = aa^T - \|a\|^2 I \quad (ii) \hat{a}^3 = -\|a\|^2 \hat{a}.$$

Demonstração. A matriz \hat{a} é gerada a partir de um vetor $a = (a_1, a_2, a_3)^T$

$$\hat{a} = \begin{bmatrix} 0 & -a_3 & a_2 \\ a_3 & 0 & -a_1 \\ -a_2 & a_1 & 0 \end{bmatrix}.$$

(i) A partir da definição, tem-se que o quadrado dessa matriz resulta em

$$\hat{a}^2 = \begin{bmatrix} 0 & -a_3 & a_2 \\ a_3 & 0 & -a_1 \\ -a_2 & a_1 & 0 \end{bmatrix} \begin{bmatrix} 0 & -a_3 & a_2 \\ a_3 & 0 & -a_1 \\ -a_2 & a_1 & 0 \end{bmatrix} = \begin{bmatrix} -(a_2^2 + a_3^2) & a_1 a_2 & a_1 a_3 \\ a_1 a_2 & -(a_1^2 + a_3^2) & a_2 a_3 \\ a_1 a_3 & a_2 a_3 & -(a_1^2 + a_2^2) \end{bmatrix}.$$

Agora, como $\|a\|^2 = a_1^2 + a_2^2 + a_3^2$, então a equação acima pode ser reescrita como

$$\hat{a}^2 = \begin{bmatrix} a_1^2 - \|a\|^2 & a_1 a_2 & a_1 a_3 \\ a_1 a_2 & a_2^2 - \|a\|^2 & a_2 a_3 \\ a_1 a_3 & a_2 a_3 & a_3^2 - \|a\|^2 \end{bmatrix} = \begin{bmatrix} a_1^2 & a_1 a_2 & a_1 a_3 \\ a_1 a_2 & a_2^2 & a_2 a_3 \\ a_1 a_3 & a_2 a_3 & a_3^2 \end{bmatrix} - \begin{bmatrix} \|a\|^2 & 0 & 0 \\ 0 & \|a\|^2 & 0 \\ 0 & 0 & \|a\|^2 \end{bmatrix}$$

isto é, $\hat{a}^2 = aa^T - \|a\|^2 I$.

(ii) Como $a^T \hat{a} = 0$, segue que

$$\hat{a}^3 = \hat{a}^2 \hat{a} = (aa^T - \|a\|^2 I) \hat{a} = a(a^T \hat{a}) - \|a\|^2 \hat{a} = -\|a\|^2 \hat{a}.$$

□

Este lema garante que é possível calcular as potências de uma matriz anti-simétrica, recursivamente, ferramenta que será de grande utilidade no que segue.

Agora, temos que garantir que todas as matrizes de rotação podem ser representadas como uma exponencial de uma matriz anti-simétrica, isto é, que o mapa

$$\exp : so(3) \longrightarrow SO(3)$$

é sobrejetor [73]. Desse modo, estabelece-se o próximo resultado cuja demonstração pode ser encontrada, também, em Murray et. al [73].

Teorema B.3.1. Dada uma matriz $R \in SO(3)$, existem $w \in \mathbb{R}^3$, unitário, e $\theta \in \mathbb{R}$ tais que

$$R = \exp(\hat{w}\theta).$$

Demonstração. Omitimos a prova, pois é demasiada extensa e construtiva. Esta pode ser encontrada em Murray et. al [73]. □

Agora, vimos que a matriz que rotaciona vetores ao redor da direção gerada por um vetor unitário w em um ângulo θ é dada pela série

$$R_{w,\theta} = \sum_{k=0}^{\infty} \frac{\hat{w}\theta^k}{k!} = I + \theta\hat{w} + \frac{\theta^2}{2!}\hat{w}^2 + \frac{\theta^3}{3!}\hat{w}^3 + \dots$$

Computacionalmente, o cálculo de uma série não seria tão eficiente quanto o uso de uma forma fechada para a definição da matriz para esta rotação. Isto é possível com a recursividade fornecida pelo Lema B.3.1. De fato, utilizando este resultado, temos que

$$R_{w,\theta} = I + \left(\theta - \frac{\theta^3}{3!} + \frac{\theta^5}{5!} + \dots \right) \hat{w} + \left(\frac{\theta^2}{2!} - \frac{\theta^4}{4!} + \frac{\theta^6}{6!} + \dots \right) \hat{w}^2. \quad (\text{B.3.4})$$

Mas, as séries que são coeficientes dos termos linear e quadrático, respectivamente, coincidem com as funções seno e cosseno para o ângulo θ . Portanto, eis o resultado desta seção.

Teorema B.3.2 (Forma matricial da Fórmula de Rodrigues). Dados um vetor unitário w como eixo e um ângulo θ para rotação, a matriz de rotação do grupo $SO(3)$ que realiza esta transformação é dada por

$$R_{w,\theta} = e^{\hat{w}\theta} = I + \hat{w} \sin(\theta) + \hat{w}^2(1 - \cos(\theta)). \quad (\text{B.3.5})$$

Em resumo, este resultado fornece uma forma fechada para o cálculo de uma matriz de rotação com eixo gerado pelo vetor unitário w e ângulo θ , caracterizando uma rotação de $SO(3)$ de forma geral, sem necessitar fixar explicitamente eixos coordenados para o espaço e rotacionar cada componente separadamente. Esta forma será fundamental na união de árvores de realizações para resolver um DMDGP.

Expandindo a Fórmula de Rodrigues e contando operações

Sejam $\mathbf{u} = [u_1 \ u_2 \ u_3]^T$ e θ , respectivamente, o vetor unitário que gera o eixo e o ângulo referente à rotação desejada. A matriz de rotação que realiza esta transformação é dada pela Fórmula de Rodrigues como $R_{\mathbf{u},\theta} = I + \hat{\mathbf{u}} \sin(\theta) + \hat{\mathbf{u}}^2(1 - \cos(\theta))$. Expandindo, temos

$$\begin{aligned} & I + \begin{bmatrix} 0 & -u_3 \sin(\theta) & u_2 \sin(\theta) \\ u_3 \sin(\theta) & 0 & -u_1 \sin(\theta) \\ -u_2 \sin(\theta) & u_1 \sin(\theta) & 0 \end{bmatrix} + \\ & \begin{bmatrix} -(u_2^2 + u_3^2)(1 - \cos(\theta)) & u_1 u_2 (1 - \cos(\theta)) & u_1 u_3 (1 - \cos(\theta)) \\ u_1 u_2 (1 - \cos(\theta)) & -(u_1^2 + u_3^2)(1 - \cos(\theta)) & u_2 u_3 (1 - \cos(\theta)) \\ u_1 u_3 (1 - \cos(\theta)) & u_2 u_3 (1 - \cos(\theta)) & -(u_1^2 + u_2^2)(1 - \cos(\theta)) \end{bmatrix} \end{aligned}$$

ou seja, somando as três matrizes, obtemos a seguinte forma para $R_{\mathbf{u},\theta}$

$$\begin{bmatrix} 1 - (u_2^2 + u_3^2)(1 - \cos(\theta)) & -u_3 \sin(\theta) + u_1 u_2 (1 - \cos(\theta)) & u_2 \sin(\theta) + u_1 u_3 (1 - \cos(\theta)) \\ u_3 \sin(\theta) + u_1 u_2 (1 - \cos(\theta)) & 1 - (u_1^2 + u_3^2)(1 - \cos(\theta)) & -u_1 \sin(\theta) + u_2 u_3 (1 - \cos(\theta)) \\ -u_2 \sin(\theta) + u_1 u_3 (1 - \cos(\theta)) & u_1 \sin(\theta) + u_2 u_3 (1 - \cos(\theta)) & 1 - (u_1^2 + u_2^2)(1 - \cos(\theta)) \end{bmatrix}.$$

Como $\|\mathbf{u}\| = 1$, então $u_1^2 + u_2^2 + u_3^2 = 1$ e, logo, temos que $-(u_2^2 + u_3^2) = u_1^2 - 1$, $-(u_1^2 + u_3^2) = u_2^2 - 1$ e $-(u_1^2 + u_2^2) = u_3^2 - 1$. Portanto, para $i = 1, 2$ e 3 , os elementos da diagonal principal de $R_{\mathbf{u},\theta}$ tornam-se $1 + (u_i^2 - 1)(1 - \cos(\theta)) = 1 - 1 + \cos(\theta) + u_i^2(1 - \cos(\theta)) = \cos(\theta) + u_i^2(1 - \cos(\theta))$. Portanto, a forma matricial expandida de uma matriz de rotação por θ ao redor de \mathbf{u} é

$$R_{\mathbf{u},\theta} = \begin{bmatrix} \cos(\theta) + u_1^2(1 - \cos(\theta)) & u_1u_2(1 - \cos(\theta)) - u_3\sin(\theta) & u_1u_3(1 - \cos(\theta)) + u_2\sin(\theta) \\ u_1u_2(1 - \cos(\theta)) + u_3\sin(\theta) & \cos(\theta) + u_2^2(1 - \cos(\theta)) & u_2u_3(1 - \cos(\theta)) - u_1\sin(\theta) \\ u_1u_3(1 - \cos(\theta)) - u_2\sin(\theta) & u_2u_3(1 - \cos(\theta)) + u_1\sin(\theta) & \cos(\theta) + u_3^2(1 - \cos(\theta)) \end{bmatrix}$$

A partir disso, pode-se contar o número de operações envolvidas na determinação dessa matriz. Considera-se adição (+) e multiplicação (\times).

Agora, sejam $\mathbf{v} \in \mathbb{R}^3$ um vetor que se deseja rotacionar, $R_{\mathbf{u},\theta}$ e $R_{\mathbf{m},\varphi}$ duas matrizes de rotação. É preciso realizar 45 operações para multiplicar as duas matrizes de rotação. Além disso, são necessárias mais 15 operações aritméticas para calcular o produto $\mathbf{w} = R_{\mathbf{u},\theta}\mathbf{v}$.

A Tabela B.1 traz essas contabilização dividindo entre as operações necessárias para encontrar a matriz de rotação e as necessárias para rotacionar de fato.

Tarefa	No. Operações
Calcular $R_{\mathbf{u},\theta}$	25
Multiplicar $R_{\mathbf{u},\theta}R_{\mathbf{m},\varphi}$	45
Calcular $R_{\mathbf{u},\theta}\mathbf{v}$	15

Tabela B.1: Tabela com os números de operações totais para rotacionar vetores em \mathbb{R}^3 , usando a Fórmula de Rodrigues matricial.

Além de contar o número de operações, é necessário analisar o espaço necessário de armazenamento. No caso dessa matriz de rotações, é preciso utilizar nove posições de memória para armazená-la.

É possível que realizemos uma sequência de rotações, o que demanda muito espaço de memória para armazenar todas as matrizes.

No apêndice seguinte, vamos estudar rotações com quatérnios, cuja estrutura utiliza menos espaço de memória para armazenamento e realize menos operações aritméticas.

Apêndice C

A Álgebra Geométrica dos Quaternios e Rotações

A Álgebra Geométrica (AG) vem no intuito de dar sentido prático ao estudo de espaços vetoriais. A nomenclatura da introdução é baseada em Dorst *et al.* [25] e Suter [89].

A AG se relaciona de modo mais eficiente com a Geometria do que a Álgebra Linear (AL).

- **Espaços vetoriais são boas ferramentas para modelagem:** vetores representam elementos em modelos do espaço. A AG oferece três bons modelos para o espaço Euclidiano: *Espaço Vetorial*, *Espaço Homogêneo* e *Espaço Conforme*;
- **Subespaços são vistos como elementos:** a partir de vetores, novos elementos maiores aparecem como combinação. Representam subespaços orientados e possuem ricas contribuições geométricas.
- **Transformações ortogonais:** no caso do Espaço Euclidiano, as transformações ortogonais são amplamente estudadas, já que preservam características como ângulos e comprimentos. Em AL, elas são representadas como matrizes, como visto no capítulo anterior. Já em AG, é possível representá-las de modo universalmente aplicável em qualquer geometria e em dimensões arbitrárias.

Em AL, vetores representam *direções*, como retas homogêneas. Já em AG, estes definem *pontos* no espaço. O nosso interesse maior tange as álgebras que modelam \mathbb{R}^3 . Deseja-se, então, utilizar um modelo, baseado em espaços vetoriais a fim de representar movimentos em *Geometria Euclidiana* (GE), os quais devem preservar certas medidas (como comprimentos e ângulos).

Há três poderosos modelos para GE:

- (1) O primeiro é o **Modelo Espaço Vetorial**, para o qual usa-se a métrica Euclidiana para descrever a álgebra de direções de \mathbb{R}^3 . Rotações podem ser representadas de duas maneiras distintas: *Matrizes Ortogonais*, conectada com AL, e a *Álgebra dos Quaternios*.
- (2) O segundo é o **Modelo Homogêneo**. Lança-se mão de direções em quatro dimensões para representar pontos em três e, dessa forma, as translações passam a ser transformações lineares

e podem ser combinadas com matrizes de rotação. Além disso, há liberdade para se escolher a métrica, como no caso da Geometria Projetiva. A dimensão adicional pode ser interpretada como *um ponto na origem*.

- (3) Por fim, o **Modelo Conforme**: melhora ainda mais as condições do modelo anterior: além das translações serem lineares, elas também são ortogonais. Logo, as rotações, reflexões e translações podem ser combinadas em uma única transformação linear e ortogonal. A *Métrica de Minkowski* é usual neste modelo. Este espaço tem quatro dimensões positivas e uma dimensão negativa e é denotado por $\mathbb{R}^{4,1}$ e as duas dimensões adicionais referem-se ao *ponto na origem* e ao *ponto no infinito*.

Este capítulo vai se concentrar no primeiro, descrevendo rotações em \mathbb{R}^3 com os Quatérnios. O intuito é fornecer a base teórica para entender esta ferramenta que foi utilizada no Capítulo 4.

C.1 Quatérnios e sua Álgebra

Nesta seção, vamos definir o espaço vetorial dos quatérnios bom como as suas operações fundamentais, além do produto dos quatérnios e sua utilização para descrever as rotações em \mathbb{R}^3 usando esta estrutura. A maior parte dos conceitos e resultados foram baseados em Kuipers [47].

Seja $B_{\mathbb{R}^3} = \{\mathbf{i}, \mathbf{j}, \mathbf{k}\}$ a base canônica de \mathbb{R}^3 . Um *quatérnio* é definido como um elemento da forma $q = q_0 + \mathbf{q}_v$, onde $q_0 \in \mathbb{R}$ é um escalar e $\mathbf{q}_v = q_1\mathbf{i} + q_2\mathbf{j} + q_3\mathbf{k}$ é um vetor de \mathbb{R}^3 . A coleção obtida na variação completa desses escalares e vetores em todo o espaço é chamada de **Conjunto dos Quatérnios**, denotado por \mathbb{H} . Além dessa representação, um quatérnio $q = q_0 + \mathbf{q}_v$ pode ser visto como o ponto $q = (q_0, q_1, q_2, q_3) \in \mathbb{R}^4$. Mais do que isso, há uma relação biunívoca entre \mathbb{H} e \mathbb{R}^4 que nos será importante mais à frente neste texto. Define-se duas operações aritméticas em \mathbb{H} :

- (1) **Adição:** dados $p = p_0 + \mathbf{p}_v$ e $q = q_0 + \mathbf{q}_v$ em \mathbb{H} , define-se a adição de p a q como $p + q = (p_0 + q_0) + (\mathbf{p}_v + \mathbf{q}_v)$. É fácil ver que \mathbb{H} é fechado para a adição. *De fato:* $r = p + q$, pode ser escrito como $r = r_0 + \mathbf{r}_v$, onde $r_0 = p_0 + q_0$ e $\mathbf{r}_v = \mathbf{p}_v + \mathbf{q}_v$. Além disso, dados $p, q, r \in \mathbb{H}$, esta operação satisfaz as propriedades seguintes, fazendo de $(\mathbb{H}, +)$ um grupo abeliano.
 - (i) **Associatividade:** $(p + q) + r = p + (q + r)$.
 - (ii) **Comutatividade:** $p + q = q + p$.
 - (iii) **Existência de Elemento Neutro:** existe um elemento $0 = 0 + \mathbf{0}_v \in \mathbb{H}$, de modo que $p + 0 = 0 + p = p$.
 - (iv) **Existência de Elemento Oposto:** para cada $p = p_0 + \mathbf{p}_v \in \mathbb{H}$, existe o seu oposto $-p = -p_0 - \mathbf{p}_v$, de modo que $p + (-p) = (-p) + p = 0$.
- (2) **Multiplicação por Escalar:** dados o quatérnio $q = q_0 + \mathbf{q}_v \in \mathbb{H}$ e o escalar $\alpha \in \mathbb{R}$, define-se a multiplicação de q por α como $\alpha q = (\alpha q_0) + (\alpha \mathbf{q}_v)$. Pode-se ver, também, que \mathbb{H} é fechado para a multiplicação de escalares reais. *De fato:* $\alpha q_0 \in \mathbb{R}$ e $\alpha \mathbf{q}_v \in \mathbb{R}^3$ e, portanto, $\alpha q \in \mathbb{H}$. Dados $\alpha, \beta \in \mathbb{R}$ e $p, q \in \mathbb{R}^3$, temos:

- (i) **Associatividade:** $(\alpha\beta)q = \alpha(\beta q)$.
- (ii) **Multiplicação pela unidade:** $1q = q$.
- (iii) **Distributividades em Relação à Soma:** adição e a multiplicação por escalar são mescladas por $(\alpha + \beta)q = \alpha q + \beta q$ e $\alpha(p + q) = \alpha p + \alpha q$.

Essas operações, com as propriedades supracitadas, induzem uma estrutura algébrica para \mathbb{H} .

Proposição C.1.1. O conjunto de quatérnios \mathbb{H} , munido das operações de adição e multiplicação por escalar real, forma uma estrutura de um 4–espaço vetorial real.

Assim, é possível identificar fortemente \mathbb{H} e \mathbb{R}^4 por isomorfismo de espaços vetoriais.

Proposição C.1.2. O espaço vetorial dos quatérnios \mathbb{H} é isomorfo ao espaço vetorial Euclideano de dimensão 4, isto é, $\mathbb{H} \cong \mathbb{R}^4$.

Mais uma definição essencial para a teoria.

Definição C.1.1. Dado $q = q_0 + \mathbf{q}_v \in \mathbb{H}$, define-se seu **conjugado** por $q^* = q_0 - \mathbf{q}_v$.

Algumas propriedades algébricas para este conjugado seguem.

Proposição C.1.3. Dados $p, q \in \mathbb{H}$, o conjugado da soma de p e q é dado por $(p + q)^* = p^* + q^*$.

Proposição C.1.4. Dado $q \in \mathbb{H}$, o conjugado do conjugado de q é igual a q , ou seja, $(q^*)^* = q$.

Proposição C.1.5. A soma de um quatérnio e seu conjugado é um número real igual a duas vezes a parte real do quatérnio, ou seja, $q + q^* = 2q_0$.

C.1.1 Produto Algébrico de Quatérnios

Escolhe-se a base $B_{\mathbb{H}} = \{1, \mathbf{i}, \mathbf{j}, \mathbf{k}\}$ para o espaço vetorial \mathbb{H} , em função da base $B_{\mathbb{R}^3}$, de modo que seus componentes são tais que satisfazem as seguintes igualdades

$$\mathbf{i}^2 = \mathbf{j}^2 = \mathbf{k}^2 = \mathbf{ijk} = -1. \quad (\text{C.1.1})$$

O produto de quatérnios será apresentado como em Kuipers [47].

Considere dois quatérnios $p = p_0 + \mathbf{p}_v = p_0 + (\mathbf{i}p_1 + \mathbf{j}p_2 + \mathbf{k}p_3)$, $q = q_0 + \mathbf{q}_v = q_0 + (\mathbf{i}q_1 + \mathbf{j}q_2 + \mathbf{k}q_3) \in \mathbb{H} - \{0\}$ escritos nessa base. O produto termo-a-termo de p por q , usando a propriedade distributiva, é dado por

$$\begin{aligned} pq &= (p_0 + \mathbf{i}p_1 + \mathbf{j}p_2 + \mathbf{k}p_3)(q_0 + \mathbf{i}q_1 + \mathbf{j}q_2 + \mathbf{k}q_3) \\ &= p_0q_0 + \mathbf{i}(p_0q_1 + p_1q_0) + \mathbf{j}(p_0q_2 + p_2q_0) + \mathbf{k}(p_0q_3 + p_3q_0) + \mathbf{i}^2p_1q_1 \\ &\quad + \mathbf{j}^2p_2q_2 + \mathbf{k}^2p_3q_3 + \mathbf{ij}p_1q_2 + \mathbf{ji}p_2q_1 + \mathbf{ik}p_1q_3 + \mathbf{ki}p_3q_1 + \mathbf{jk}p_2q_3 + \mathbf{kj}p_3q_2. \end{aligned} \quad (\text{C.1.2})$$

Para simplificar (C.1.2) e agrupar em produtos notáveis, definiu-se os produtos:

$$\begin{array}{rcl}
\hline \hline
\mathbf{i}\mathbf{j} & = & \mathbf{k} & = & -\mathbf{j}\mathbf{i} \\
\mathbf{j}\mathbf{k} & = & \mathbf{i} & = & -\mathbf{k}\mathbf{j} \\
\hline \hline
\mathbf{k}\mathbf{i} & = & \mathbf{j} & = & -\mathbf{i}\mathbf{k}
\end{array}$$

Tabela C.1: Tabela de produto entre itens da base.

Logo, a Equação (C.1.2) pode ser reescrita pelo agrupamento de termos como

$$\begin{aligned}
pq &= p_0q_0 - (p_1q_1 + p_2q_2 + p_3q_3) + p_0(\mathbf{i}q_1 + \mathbf{j}q_2 + \mathbf{k}q_3) + q_0(\mathbf{i}p_1 + \mathbf{j}p_2 + \mathbf{k}p_3) \\
&\quad + \mathbf{i}(p_2q_3 - p_3q_2) + \mathbf{j}(p_3q_1 - p_1q_3) + \mathbf{k}(p_1q_2 - p_2q_1).
\end{aligned}$$

Usando o produto interno usual de \mathbb{R}^3 , tem-se que

$$\begin{aligned}
pq &= p_0q_0 - \langle \mathbf{p}_v, \mathbf{q}_v \rangle + p_0\mathbf{q}_v + q_0\mathbf{p}_v + \mathbf{i}(p_2q_3 - p_3q_2) + \mathbf{j}(p_3q_1 - p_1q_3) \\
&\quad + \mathbf{k}(p_1q_2 - p_2q_1).
\end{aligned} \tag{C.1.3}$$

Agora, temos $\mathbf{p}_v \times \mathbf{q}_v = \mathbf{i}(p_2q_3 - p_3q_2) + \mathbf{j}(p_3q_1 - p_1q_3) + \mathbf{k}(p_1q_2 - p_2q_1)$ pelo produto vetorial de \mathbb{R}^3 . Aplicando isso a (C.1.3), consegue-se o agrupamento final para este produto.

Definição C.1.2 (Produto Algébrico de Quatérnios). Dados $p = p_0 + \mathbf{p}_v, q = q_0 + \mathbf{q}_v \in \mathbb{H} - 0$, o produto algébrico entre p e q é dado por

$$pq = p_0q_0 - \langle \mathbf{p}_v, \mathbf{q}_v \rangle + p_0\mathbf{q}_v + q_0\mathbf{p}_v + \mathbf{p}_v \times \mathbf{q}_v. \tag{C.1.4}$$

Teorema C.1.1. O conjunto \mathbb{H} munido de adição, multiplicação por escalar e do produto de quatérnios forma uma álgebra associativa chamada de **Álgebra dos Quatérnios**.

Vistos como álgebras, temos que $\mathbb{H} \cong \mathbb{R}^4$, mantendo o isomorfismo da Proposição C.1.2.

Já vimos que \mathbb{H} é um grupo abeliano. Entretanto, a álgebra dos quatérnios \mathbb{H} não é comutativa. Basta ver que $\mathbf{i}\mathbf{j} = \mathbf{k} = -\mathbf{j}\mathbf{i}$.

Além disso, a álgebra do quatérnios possui um elemento identidade dado por $1_{\mathbb{H}} = 1 + \mathbf{0}_v$. De fato: dado $q = q_0 + \mathbf{q}_v \in \mathbb{H}$, temos que $q1_{\mathbb{H}} = q$, pois fazendo este produto temos

$$q1_{\mathbb{H}} = (q_0 + \mathbf{q}_v)(1 + \mathbf{0}_v) = q_01 - \langle \mathbf{q}_v, \mathbf{0}_v \rangle + q_0\mathbf{0}_v + 1\mathbf{q}_v + (\mathbf{q}_v \times \mathbf{0}_v) = q_0 + \mathbf{q}_v = q. \tag{C.1.5}$$

Uma propriedade que relaciona este produto com o conjugado de um quatérnio segue.

Proposição C.1.6. Dados $p, q \in \mathbb{H}$, o conjugado do produto de p por q é dado por $(pq)^* = q^*p^*$.

De posse do conjugado e do produto, é possível definir um conceito de norma em \mathbb{H} .

Definição C.1.3. Dado $q \in \mathbb{H}$, sua *norma* é dada por $N(q) = \sqrt{q^*q}$.

Duas propriedades para normas de quatérnios seguem.

Proposição C.1.7. Dado $p \in \mathbb{H}$, a norma do conjugado é igual à própria norma $N(p^*) = N(q)$.

Proposição C.1.8. Dados $p, q \in \mathbb{H}$, a norma do produto de p e q é dada por $N(pq) = N(p)N(q)$.

A partir de agora, sempre consideraremos \mathbb{H} como uma álgebra normada.

Definição C.1.4. Para cada $q \in \mathbb{H} - \{0\}$, existe um elemento $q^{-1} \in \mathbb{H}$ tal que $qq^{-1} = q^{-1}q = 1_{\mathbb{H}}$ e $q^{-1} = \frac{1}{N^2(q)}q^*$ chamado de *inverso do quatérnico* q .

Teorema C.1.2 (Inverso de Quatérnico Unitário). Se $q \in \mathbb{H}$ é unitário, isto é, $N(q) = 1$, então o inverso de q coincide com seu conjugado $q^{-1} = q^*$.

Duas definições se fazem necessárias.

Definição C.1.5. $q \in \mathbb{H}$ é dito **quatérnico real** se for da forma $q = q_0 + \mathbf{0}_v$, isto é, se $\mathbf{q}_v \equiv 0$. O conjunto dos quatérnicos reais é denotado como \mathbb{R} , por uma relação biunívoca entre os dois conjuntos.

Definição C.1.6. $q \in \mathbb{H}$ é dito **quatérnico puro** se é dado por $q = 0 + \mathbf{q}_v$, ou seja, se $q_0 = 0$. O conjunto dos quatérnicos puros é denotado por \mathbb{H}_0 .

Há mais um teorema de isomorfismo entre álgebras, o qual permite que tratemos de vetores como quatérnicos puros.

Teorema C.1.3. A subálgebra de quatérnicos puros é isomorfa ao espaço euclidiano de três dimensões, ou seja, $\mathbb{R}^3 \cong \mathbb{H}_0$.

C.1.2 Forma Matricial do Produto Geométrico de Quatérnios

Pode-se expressar o produto algébrico por meio de produto de matrizes, facilitando a sua implementação computacional.

Teorema C.1.4. Sejam dois quatérnios $p = p_0 + \mathbf{p}_v = p_0 + \mathbf{i}p_1 + \mathbf{j}p_2 + \mathbf{k}p_3, q = q_0 + \mathbf{q}_v = q_0 + \mathbf{i}q_1 + \mathbf{j}q_2 + \mathbf{k}q_3 \in \mathbb{H}$. Então, o quatérnico $r = r_0 + \mathbf{i}r_1 + \mathbf{j}r_2 + \mathbf{k}r_3$, resultante do produto geométrico de p e q , pode ter suas coordenadas dadas a partir do produto

$$\begin{bmatrix} r_0 \\ r_1 \\ r_2 \\ r_3 \end{bmatrix} = \begin{bmatrix} p_0 & -p_1 & -p_2 & -p_3 \\ p_1 & p_0 & -p_3 & p_2 \\ p_2 & p_3 & p_0 & -p_1 \\ p_3 & -p_2 & p_1 & p_0 \end{bmatrix} \begin{bmatrix} q_0 \\ q_1 \\ q_2 \\ q_3 \end{bmatrix}. \quad (\text{C.1.6})$$

C.2 Quatérnios como Rotações

Temos todo o ferramental algébrico da estrutura \mathbb{H} que nos permitirá discutir um modo eficiente de realizar rotações de vetores em \mathbb{R}^3 usando seus elementos. Dessa maneira, temos de estabelecer uma identificação entre $SO(3)$ e \mathbb{H} . Há que se lembrar que existe um isomorfismo $\varphi : \mathbb{R}^3 \rightarrow \mathbb{H}_0$. Esta relação um-a-um, ilustrada na Figura C.1 que foi extraída de [63], será fundamental para definir esta transformação. Por isso, abusaremos da linguagem a partir de então.

Considere cada vetor $\mathbf{v} \in \mathbb{R}^3$ como seu equivalente em \mathbb{H}_0 , pela aplicação φ , por $\mathbf{v} \longleftrightarrow v = 0 + \mathbf{v}$. Agora, seja $v \in \mathbb{H}_0$. Nossa desejo é encontrar $p, q \in \mathbb{H}$ tais que

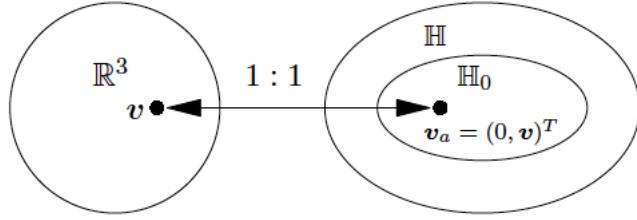


Figura C.1: Ilustração do isomorfismo existente entre \mathbb{R}^3 e \mathbb{H}_0 .

$$w = qvp \in \mathbb{H}_0.$$

Usando as regras do produto algébrico de quatérnios (C.1.4), temos

$$\begin{aligned} w = qvp &= (qv)p \\ &= [(q_0 + \mathbf{q}_v)(0 + \mathbf{v})](p_0 + \mathbf{p}_v) \\ &= [(-\mathbf{q}_v \cdot \mathbf{v}) + (q_0 \mathbf{v} + (\mathbf{q}_v \times \mathbf{v}))][(p_0 + \mathbf{p}_v)] \\ &= [(-\mathbf{q}_v \cdot \mathbf{v})p_0 - q_0(\mathbf{v} \cdot \mathbf{p}_v) - ((\mathbf{q}_v \times \mathbf{v}) \cdot \mathbf{p}_v)] \\ &\quad + [(-\mathbf{q}_v \cdot \mathbf{v})\mathbf{p}_v + (p_0 q_0)\mathbf{v} + p_0(\mathbf{q}_v \times \mathbf{v}) + q_0(\mathbf{v} \times \mathbf{p}_v) + (\mathbf{p}_v \times \mathbf{v} \times \mathbf{q}_v)]. \end{aligned} \tag{C.2.1}$$

A parte real do quatérnio w é dada na forma $w_0 = \text{re}(qvp) = (-\mathbf{q}_v \cdot \mathbf{v})p_0 - q_0(\mathbf{v} \cdot \mathbf{p}_v) - ((\mathbf{q}_v \times \mathbf{v}) \cdot \mathbf{p}_v)$ e estima-se que $w_0 = 0$, para que w seja um quatérnio puro.

As propriedades da AL para \mathbb{R}^3 valem para \mathbb{H}_0 , através do isomorfismo φ . Adicionalmente, pelo produto misto, temos que $(\mathbf{q}_v \times \mathbf{v}) \cdot \mathbf{p}_v = -(\mathbf{q}_v \times \mathbf{p}_v) \cdot \mathbf{v}$ e, pelo produto escalar em \mathbb{R}^3 e por φ , que $(\mathbf{q}_v \cdot \mathbf{v})p_0 = p_0(\mathbf{q}_v \cdot \mathbf{v})$ e $q_0(\mathbf{v} \cdot \mathbf{p}_v) = q_0(\mathbf{p}_v \cdot \mathbf{v})$. Logo,

$$-p_0(\mathbf{q}_v \cdot \mathbf{v}) - q_0(\mathbf{p}_v \cdot \mathbf{v}) + ((\mathbf{q}_v \times \mathbf{p}_v) \cdot \mathbf{v}) = 0.$$

Suponha, agora, que $p_0 = q_0$ e que $\mathbf{p}_v = -\mathbf{q}_v$, hipóteses fundamentais para mostrar a operação de rotação e, logo, atinge-se o desejado: que a parte real do produto triplo (C.2.1) seja anulada

$$-q_0((\mathbf{q}_v - \mathbf{q}_v) \cdot \mathbf{v}) - ((\mathbf{q}_v \times \mathbf{q}_v) \cdot \mathbf{v}) = 0.$$

Tomando $w = qvp$, onde $q = q_0 + \mathbf{q}_v$, $p = q_0 - \mathbf{q}_v \in \mathbb{H}$ e $v = 0 + \mathbf{v} \in \mathbb{H}_0$, tem-se que w também é um quatérnio puro. Dizemos, então, que o quatérnio q e seu conjugado operam no vetor \mathbf{v} de \mathbb{R}^3 transformando-o em outro vetor w de \mathbb{R}^3 , o que nos permite definir o seguinte operador.

Definição C.2.1. Dado $q \in \mathbb{H}$, define-se o operador L_q em \mathbb{H}_0 dado por

$$\begin{array}{rcl} L_q : & \mathbb{H}_0 & \longrightarrow \mathbb{H}_0 \\ & \mathbf{v} & \longmapsto L_q(\mathbf{v}) = q\mathbf{v}q^* \end{array}.$$

Algumas questões sobre o funcionamento deste operador aparecem naturalmente. Ele é linear? Preserva normas? Resultados como esses seguem das definições e propriedades da álgebra \mathbb{H} .

Teorema C.2.1. L_q é um operador linear.

Doravante, considere $q \in \mathbb{H}$ unitário, isto é, $N(q) = 1$.

Teorema C.2.2. L_q preserva normas.

Seja $\|\cdot\|_2$ a norma Euclideana para \mathbb{R}^3 . Se $q = q_0 + \mathbf{q}_v \in \mathbb{H}$ tem norma 1, então

$$1 = N^2(q) = q_0^2 + \|\mathbf{q}_v\|_2^2. \quad (\text{C.2.2})$$

Desse modo, há uma identificação trigonométrica entre a relação $\cos^2(\theta) + (\operatorname{sen})^2(\theta) = 1$ e a Equação (C.2.2): existe um único ângulo $\theta \in [0, \pi]$, que está associado ao quatérnio q de modo que

$$q_0 = \cos(\theta) \quad \text{e} \quad \|\mathbf{q}_v\|_2 = \operatorname{sen}(\theta). \quad (\text{C.2.3})$$

É possível escrever o quatérnio unitário q em termos de θ , a fim de mostrar que L_q , de fato, realiza uma rotação. Primeiramente, define-se o quatérnio puro \mathbf{u} , representando o vetor $\mathbf{u} \in \mathbb{R}^3$ que fornece a direção do vetor \mathbf{q}_v , como o vetor unitário

$$\mathbf{u} = \frac{\mathbf{q}_v}{\|\mathbf{q}_v\|_2} = \frac{\mathbf{q}_v}{\operatorname{sen}(\theta)}$$

e então toma-se $q = \cos(\theta) + \mathbf{u}\operatorname{sen}(\theta)$. Além disso, pela paridade das funções trigonométricas, temos que $\cos(-\theta) + \mathbf{u}\operatorname{sen}(-\theta) = \cos(\theta) - \mathbf{u}\operatorname{sen}(\theta)$ e, logo, seu conjugado pode ser escrito também nesses termos como $\cos(\theta) + \mathbf{u}\operatorname{sen}(\theta) = q_0 - \mathbf{q}_v = q^*$.

Considere $q \in \mathbb{H}$ e $v \in \mathbb{H}_0$. Desenvolvendo o produto $\mathbf{w} = q\mathbf{v}q^*$, temos

$$\begin{aligned} \mathbf{w} &= q\mathbf{v}q^* = (q_0 + \mathbf{q}_v)(0 + \mathbf{v})(q_0 - \mathbf{q}_v) \\ &= [(q_0 + \mathbf{q}_v)(0 + \mathbf{v})](q_0 - \mathbf{q}_v) = [(-\mathbf{q}_v \cdot \mathbf{v}) + (q_0\mathbf{v} + (\mathbf{q}_v \times \mathbf{v}))](q_0 - \mathbf{q}_v) \\ &= -q_0(\mathbf{q}_v \cdot \mathbf{v}) + q_0(\mathbf{q}_v \cdot \mathbf{v}) + (\mathbf{q}_v \times \mathbf{v}) \cdot \mathbf{q}_v + (\mathbf{q}_v \cdot \mathbf{v})\mathbf{q}_v + q_0^2\mathbf{v} + q_0(\mathbf{q}_v \times \mathbf{v}) \\ &\quad - q_0(\mathbf{v} \times \mathbf{q}_v) - (\mathbf{q}_v \times \mathbf{v} \times \mathbf{q}_v) \\ &= (\mathbf{q}_v \times \mathbf{v}) \cdot \mathbf{q}_v + (\mathbf{q}_v \cdot \mathbf{v})\mathbf{q}_v + q_0^2\mathbf{v} + 2q_0(\mathbf{q}_v \times \mathbf{v}) - (\mathbf{q}_v \times \mathbf{v} \times \mathbf{q}_v) \end{aligned} \quad (\text{C.2.4})$$

Pelo produto misto em \mathbb{R}^3 , temos que $(\mathbf{q}_v \times \mathbf{v}) \cdot \mathbf{q}_v = \mathbf{q}_v \cdot (\mathbf{q}_v \times \mathbf{v}) = \mathbf{v} \cdot (\mathbf{q}_v \times \mathbf{q}_v) = 0$ o que, sendo aplicado na Equação (C.2.4), gera o quatérnio

$$\mathbf{w} = (\mathbf{q}_v \cdot \mathbf{v})\mathbf{q}_v + q_0^2\mathbf{v} + 2q_0(\mathbf{q}_v \times \mathbf{v}) - (\mathbf{q}_v \times \mathbf{v} \times \mathbf{q}_v). \quad (\text{C.2.5})$$

Para continuar este desenvolvimento, usamos o chamado *Produto Vetorial Triplo*.

Lema C.2.1 (Produto Vetorial Triplo). Dados os vetores $\mathbf{a}, \mathbf{b}, \mathbf{c} \in \mathbb{R}^3$, vale a seguinte relação para o produto vetorial triplo $\mathbf{a} \times (\mathbf{b} \times \mathbf{c}) = (\mathbf{a} \cdot \mathbf{c})\mathbf{b} - (\mathbf{a} \cdot \mathbf{b})\mathbf{c}$.

A partir do lema, temos $(\mathbf{q}_v \times \mathbf{v} \times \mathbf{q}_v) = \mathbf{q}_v \times (\mathbf{v} \times \mathbf{q}_v) = (\mathbf{q}_v \cdot \mathbf{q}_v)\mathbf{v} - (\mathbf{q}_v \cdot \mathbf{v})\mathbf{q}_v$ e, logo,

$$\begin{aligned} \mathbf{w} &= q\mathbf{v}q^* = (q_0 + \mathbf{q}_v)(0 + \mathbf{v})(q_0 - \mathbf{q}_v) \\ &= (\mathbf{q}_v \cdot \mathbf{v})\mathbf{q}_v + q_0^2\mathbf{v} + 2q_0(\mathbf{q}_v \times \mathbf{v}) - \|\mathbf{q}_v\|_2^2\mathbf{v} + (\mathbf{q}_v \cdot \mathbf{v})\mathbf{q}_v \\ &= q_0^2\mathbf{v} - \|\mathbf{q}_v\|_2^2\mathbf{v} + 2q_0(\mathbf{q}_v \times \mathbf{v}) + 2(\mathbf{q}_v \cdot \mathbf{v})\mathbf{q}_v. \end{aligned} \quad (\text{C.2.6})$$

Agora, por hipótese, q é unitário. Logo,

$$1 = N^2(q) = q_0^2 + \|\mathbf{q}_v\|_2^2 \text{ do que segue que } -\|\mathbf{q}_v\|_2^2 = q_0^2 - 1.$$

Aplicando esta relação na Equação (C.2.6), consegue-se o seguinte desenvolvimento

$$\begin{aligned}
\mathbf{w} &= q\mathbf{v}q^* = (q_0 + \mathbf{q}_v)(0 + \mathbf{v})(q_0 - \mathbf{q}_v) \\
&= q_0^2\mathbf{v} - \|q_v\|_2^2\mathbf{v} + 2q_0(\mathbf{q}_v \times \mathbf{v}) + 2(\mathbf{q}_v \cdot \mathbf{v})\mathbf{q}_v \\
&= q_0^2\mathbf{v} + (q_0^2 - 1)\mathbf{v} + 2q_0(\mathbf{q}_v \times \mathbf{v}) + 2(\mathbf{q}_v \cdot \mathbf{v})\mathbf{q}_v \\
&= 2q_0^2\mathbf{v} - \mathbf{v} + 2q_0(\mathbf{q}_v \times \mathbf{v}) + 2(\mathbf{q}_v \cdot \mathbf{v})\mathbf{q}_v \\
&= (2q_0^2 - 1)\mathbf{v} + 2(\mathbf{q}_v \cdot \mathbf{v})\mathbf{q}_v + 2q_0(\mathbf{q}_v \times \mathbf{v}).
\end{aligned} \tag{C.2.7}$$

Portanto, temos o seguinte resultado.

Teorema C.2.3 (Forma Explícita do Operador L_q). Dados o quatérnio $q = q_0 + \mathbf{q}_v \in \mathbb{H}$ e o quatérnio puro $\mathbf{v} = 0 + \mathbf{v} \in \mathbb{H}_0$, temos que o operador L_q pode ser definido explicitamente como

$$\mathbf{w} = L_q(\mathbf{v}) = (2q_0^2 - 1)\mathbf{v} + 2(\mathbf{q}_v \cdot \mathbf{v})\mathbf{q}_v + 2q_0(\mathbf{q}_v \times \mathbf{v}). \tag{C.2.8}$$

Retrocedendo alguns passos do raciocínio anterior, consegue-se outra fórmula para definir L_q explicitamente que pode ser conveniente para uso mais adiante.

Teorema C.2.4 (Forma Explícita Alternativa do Operador L_q). Dados $q = q_0 + \mathbf{q}_v \in \mathbb{H}$ e $\mathbf{v} = 0 + \mathbf{v} \in \mathbb{H}_0$, temos que o operador L_q pode ser definido explicitamente por

$$\mathbf{w} = L_q(\mathbf{v}) = (q_0^2 - \|q_v\|_2^2)\mathbf{v} + 2(\mathbf{q}_v \cdot \mathbf{v})\mathbf{q}_v + 2q_0(\mathbf{q}_v \times \mathbf{v}). \tag{C.2.9}$$

C.2.1 Forma Matricial do Operador L_q

Vamos descrever, explicitamente, a ação do operador L_q em um vetor usando linguagem matricial. Esta nova roupagem fornece uma maneira eficiente e conveniente de demonstrar a ação de L_q em \mathbb{H}_0 , baseado em [47].

Teorema C.2.5 (Forma Matricial do Operador L_q). Dados $q \in \mathbb{H}$, representado por um vetor de \mathbb{R}^4 como $q = (q_0 \ q_1 \ q_2 \ q_3)$, e $\mathbf{v} \in \mathbb{H}_0$, representado como um vetor de \mathbb{R}^3 por $\mathbf{v} = (v_1 \ v_2 \ v_3)$, podemos representar o quatérnio puro $\mathbf{w} = L_q(\mathbf{v})$ em forma do produto matricial

$$\begin{bmatrix} w_1 \\ w_2 \\ w_3 \end{bmatrix} = \begin{bmatrix} (2q_0^2 - 1 + 2q_1^2) & (2q_1q_2 - 2q_0q_3) & (2q_1q_3 - 2q_0q_2) \\ (2q_1q_2 + 2q_0q_3) & (2q_0^2 - 1 + 2q_2^2) & (2q_2q_3 - 2q_0q_1) \\ (2q_1q_3 - 2q_0q_2) & (2q_2q_3 + 2q_0q_1) & (2q_0^2 - 1 + 2q_3^2) \end{bmatrix} \begin{bmatrix} v_1 \\ v_2 \\ v_3 \end{bmatrix} \tag{C.2.10}$$

C.2.2 Um operador de rotação em \mathbb{R}^3

Já existem ferramentas suficientes para a demonstração do principal resultado desta seção, que une quatérnios e rotações em \mathbb{R}^3 . Antes, é necessário enunciar outro lema.

Lema C.2.2. Sejam $q = q_0 + \mathbf{q}_v \in \mathbb{H}$ um quatérnio unitário fixo e $S = \langle \mathbf{q}_v \rangle$ o subespaço vetorial de \mathbb{R}^3 gerado pelo vetor \mathbf{q}_v . Então, S é L_q -invariante.

Com esse ferramental, é possível demonstrar os principais teoremas desta seção, os quais serão usados para realizar rotações de realizações de grafos em \mathbb{R}^3 .

Teorema C.2.6. Sejam $q = q_0 + \mathbf{q}_v \in \mathbb{H} \setminus \{0\}$ um quatérnico unitário fixo, $\mathbf{v} \in \mathbb{R}^3$ um vetor arbitrário e \mathbf{w} tal que $\mathbf{w} = q\mathbf{v}q^*$. Então, \mathbf{w} pode ser obtido a partir de uma rotação do vetor \mathbf{v} pelo ângulo 2θ ao redor do subespaço gerado por \mathbf{q}_v , onde $\theta = \text{arc cos}(q_0)$.

Demonstração. Sejam $S = \langle \mathbf{q}_v \rangle$, o subespaço gerado pelo vetor \mathbf{q}_v e S^\perp seu complemento ortogonal. Logo, $\mathbb{R}^3 = S \oplus S^\perp$, que segue da AL. Desse modo, \mathbf{v} pode ser escrito como $\mathbf{v} = \mathbf{v}_p + \mathbf{v}_o$, onde $\mathbf{v}_p \in S$ e $\mathbf{v}_o \in S^\perp$. Agora, da definição de S , segue que $\mathbf{v}_p = \alpha \mathbf{q}_v$, para algum $\alpha \in \mathbb{R}$. Ou seja, $\mathbf{v} = \alpha \mathbf{q}_v + \mathbf{v}_o$. Usando a linearidade do operador L_q , do Teorema C.2.1, e o resultado do Lema C.2.2, temos que $L_q(\mathbf{v}) = L_q(\alpha \mathbf{q}_v + \mathbf{v}_o) = \alpha L_q(\mathbf{q}_v) + L_q(\mathbf{v}_o)$ e, logo, $L_q(\mathbf{v}) = \alpha \mathbf{q}_v + L_q(\mathbf{v}_o)$. Mas, pela definição explícita alternativa do operador L_q , dada pela Equação (C.2.9), temos a relação $L_q(\mathbf{v}_o) = q\mathbf{v}_o q^* = (q_0^2 - \|\mathbf{q}_v\|^2)\mathbf{v}_o + 2(\mathbf{q}_v \cdot \mathbf{v}_o)\mathbf{q}_v + 2q_0(\mathbf{q}_v \times \mathbf{v}_o)$. Como $\mathbf{v}_o \in S^\perp$, então $(\mathbf{q}_v \cdot \mathbf{v}_o) = 0$, pois \mathbf{q}_v é gerador de S . Segue que $L_q(\mathbf{v}_o) = (q_0^2 - \|\mathbf{q}_v\|^2)\mathbf{v}_o + 2q_0(\mathbf{q}_v \times \mathbf{v}_o)$. Tomamos, então, \mathbf{u} como o vetor unitário na direção \mathbf{q}_v , isto é,

$$\mathbf{u} = \frac{\mathbf{q}_v}{\|\mathbf{q}_v\|} \text{ e, logo, } \mathbf{q}_v = \|\mathbf{q}_v\| \mathbf{u}. \quad (\text{C.2.11})$$

Isso produz a equação $L_q(\mathbf{v}_o) = (q_0^2 - \|\mathbf{q}_v\|^2)\mathbf{v}_o + 2q_0(\|\mathbf{q}_v\| \mathbf{u} \times \mathbf{v}_o)$, que por linearidade, se escreve $L_q(\mathbf{v}_o) = (q_0^2 - \|\mathbf{q}_v\|^2)\mathbf{v}_o + 2q_0 \|\mathbf{q}_v\| (\mathbf{u} \times \mathbf{v}_o)$. Por hipótese, o quatérnico q é unitário e, assim, $q_0^2 + \|\mathbf{q}_v\|^2 = 1$. Desse modo, é possível encontrar um ângulo $\theta \in [0, \pi]$, único, de maneira que $q_0 = \cos(\theta)$ e $\|\mathbf{q}_v\| = \sin(\theta)$. Deriva-se, então

$$L_q(\mathbf{v}_o) = [\cos^2(\theta) - \sin^2(\theta)]\mathbf{v}_o + [2\cos(\theta)\sin(\theta)](\mathbf{u} \times \mathbf{v}_o). \quad (\text{C.2.12})$$

Define-se \mathbf{n} como o vetor normal ao plano que contém os vetores \mathbf{u} e \mathbf{v}_o da forma $\mathbf{n} = (\mathbf{u} \times \mathbf{v}_o)$.

Afirmamos que $\|\mathbf{n}\| = \|\mathbf{v}_o\|$. De fato: como $\|\mathbf{u}\| = 1$, por construção, e, além disso, o ângulo entre \mathbf{u} e \mathbf{v}_o é $\frac{\pi}{2}$, já que $\mathbf{u} \in S$ e $\mathbf{v}_o \in S^\perp$, então, pela definição do produto vetorial, temos

$$\|\mathbf{n}\| = \|\mathbf{u} \times \mathbf{v}_o\| = \|\mathbf{u}\| \|\mathbf{v}_o\| \sin\left(\frac{\pi}{2}\right) = \|\mathbf{v}_o\|, \quad (\text{C.2.13})$$

ou seja, temos que $\|\mathbf{n}\| = \|\mathbf{v}_o\|$. Segue, então, que $L_q(\mathbf{v}_o) = [\cos^2(\theta) - \sin^2(\theta)]\mathbf{v}_o + [2\cos(\theta)\sin(\theta)]\mathbf{n}$. Mas, como $\cos(2\theta) = \cos^2(\theta) - \sin^2(\theta)$ e $\sin(2\theta) = 2\sin(\theta)\cos(\theta)$, então podemos reescrever nestes novos termos o vetor $\mathbf{w}_o = L_q(\mathbf{v}_o) = \cos(2\theta)\mathbf{v}_o + \sin(2\theta)\mathbf{n}$.

Afirmamos que $\|\mathbf{w}_o\| = \|\mathbf{v}_o\|$. Com efeito: temos que

$$\begin{aligned} \|\mathbf{w}_o\|^2 &= \|\cos(2\theta)\mathbf{v}_o + \sin(2\theta)\mathbf{n}\|^2 = \|\cos(2\theta)\mathbf{v}_o\|^2 + \|\sin(2\theta)\mathbf{n}\|^2 \\ &= \cos^2(2\theta) \|\mathbf{v}_o\|^2 + \sin^2(2\theta) \|\mathbf{n}\|^2 = \cos^2(2\theta) \|\mathbf{v}_o\|^2 + \sin^2(2\theta) \|\mathbf{v}_o\|^2 \\ &= [\cos^2(2\theta) + \sin^2(2\theta)] \|\mathbf{v}_o\|^2 = \|\mathbf{v}_o\|^2 \end{aligned} \quad (\text{C.2.14})$$

e, logo, $\|\mathbf{w}_o\| = \|\mathbf{v}_o\|$. Ou seja, conclui-se que \mathbf{w}_o coincide com o vetor \mathbf{v}_o , rotacionado pelo ângulo 2θ em torno do subespaço S como eixo. Mas, $L_q(\mathbf{v}) = \mathbf{v}_p + L_q(\mathbf{v}_o)$, ou seja, $\mathbf{w} = L_q(\mathbf{v}) = \mathbf{v}_p + \mathbf{w}_o$. Então, \mathbf{w} é fornecido a partir de rotação de \mathbf{v} pelo ângulo 2θ , o que é realizado por L_q . Portanto, dado $q = q_0 + \mathbf{q}_v$ fixo, L_q é um operador de rotação em \mathbb{R}^3 , onde o ângulo de rotação é 2θ , sendo $\theta = \text{arc cos}(q_0)$, e o eixo dessa rotação é o subespaço vetorial $S = \langle \mathbf{q}_v \rangle$, como queríamos. \square

Teorema C.2.7 ($Q(q)$ é matriz de rotação). Dado $q = q_0 + \mathbf{q}_v \in \mathbb{H}$, $Q(q)$ representa uma matriz de rotação em \mathbb{R}^3 em torno de $S = \langle \mathbf{q}_v \rangle$ pelo ângulo 2θ , onde $\theta = \arccos(q_0)$.

Pode-se identificar $q \in \mathbb{H}$ com a matriz $Q(q) \in SO(3)$ como feito em [63], a partir do mapa

$$\begin{aligned} \Phi : S^3(\mathbb{H}) &\longrightarrow SO(3) \\ q &\longmapsto Q(q) \end{aligned} . \quad (\text{C.2.15})$$

onde $S^3(\mathbb{H})$ é a esfera unitária no conjunto \mathbb{H} (usamos uma relação biunívoca entre esses dois conjuntos). O ângulo $\varphi = 2\theta$ da rotação feita em torno de \mathbf{q}_v em questão está no intervalo $[0, 2\pi]$, já que $\theta \in (0, \pi)$. Mas, rotacionar por φ em torno de \mathbf{q}_v é semelhante a rotacionar por $2\pi - \varphi$ em torno de $-\mathbf{q}_v$. Ou seja, cada matriz $Q \in SO(3)$ é imagem por Φ de dois elementos (e exatamente dois, inclusive para o elemento identidade) na esfera S^3 . É fácil verificar que $Q(q) = Q(-q)$ e $L_q(\mathbf{v}) = L_{-q}(\mathbf{v})$, para qualquer $\mathbf{v} \in \mathbb{H}_0$. Portanto, como conclui Lima [63], Φ é sobrejetiva, mas não injetiva. A fim de ter injetividade, particionamos $S^3(\mathbb{H})$ a partir da relação de equivalência \sim

$$p, q \in S^3(\mathbb{H}) \subset \mathbb{H} : p \sim q \iff p = -q. \quad (\text{C.2.16})$$

A partir de \sim , consideramos o novo conjunto quociente $SQ^3 = S^3/\sim$. Com ele, é possível definir um mapa bijetivo. De fato, tal associação é feita pela aplicação

$$\Omega : SQ^3 \longrightarrow SO(3) \begin{aligned} q &\longmapsto Q(q) \end{aligned} . \quad (\text{C.2.17})$$

C.2.3 Definir um Quatérnio de Rotação com eixo e ângulo

Dado um quatérnio unitário, sabemos calcular o eixo $\langle \mathbf{u} \rangle$ em que a rotação se dará e o ângulo θ . Agora, desejamos saber o processo contrário.

Seja \mathbf{u} um vetor unitário gerador do eixo de rotação e θ o ângulo no qual se deseja rotacionar os vetores. Então, baseado no Teorema C.2.6, temos que tal quatérnio de rotação pode ser definido como

$$q_{\theta, \mathbf{u}} = \cos\left(\frac{\theta}{2}\right) + \sin\left(\frac{\theta}{2}\right)\mathbf{u}. \quad (\text{C.2.18})$$

É de simples manipulação algébrica mostrar que este quatérnio é unitário.

C.3 Contagens de Número de Operações com Quatérnios

Por fim, vamos contar a quantidade de operações aritméticas a serem realizadas em algumas das operações com quatérnios. Vamos nos concentrar naquelas que serão utilizadas para fazer as transformações Euclidianas que desejamos com as realizações de grafos do Capítulo 4.

Sejam q e p dois quatérnios unitários (de rotação) e \mathbf{v} um vetor. São necessárias:

- **28 operações** para compor os dois quatérnios pelo produto algébrico qp ,

- **34 operações** para rotacionar um vetor \mathbf{v} utilizando o operador de rotação associado ao quatérnio q L_q e
- **29 operações** para extrair uma matriz de rotação R_q , associada ao quatérnio q e ao operador L_q .

Apêndice D

Métodos Dividir-e-Conquistar para desenvolver algoritmos eficientes

Existem muitas maneiras de se elaborar algoritmos computacionais para resolver problemas diversos na Matemática e na Computação. Uma delas, em especial, se mostrou uma estratégia deveras útil no desenvolvimento de padrões de algoritmos eficientes, a qual é denominada Recursão ou Recursividade. A sua estrutura consiste no seguinte: para resolver um problema dado, estes algoritmos recorrem a si mesmos uma, ou mais de uma, vez a fim de lidar com subproblemas próprios os quais são estreitamente relacionados [14].

Para muitos problemas, a estrutura algorítmica que determina uma ou outras soluções (mesmo que aproximadas) ganhou muita eficiência ao se adotar tal estrutura em sua implementação. Um bom exemplo é o algoritmo com o qual estamos lidando neste trabalho *Branch & Prune*. Apesar de sua complexidade computacional ser exponencial, a estrutura recursiva evita erros de arredondamento e saturamento da memória, procurando caminhos factíveis na árvore binária de soluções.

A idéia de recursividade, portanto, origina-se na existência de subproblemas que possuem estrutura análoga à do problema original. É comum, dessa maneira, seguir a chamada abordagem (ou paradigma) Dividir & Conquistar (D&C) [14] - do inglês, *Divide & Conquer*, em referência a uma estratégia de governo do imperador romano César.

D.1 Estrutura

De acordo com Cormen et al. [14], há três passos básicos para o paradigma D&C:

- (P1) **DIVIDIR:** o problema é dividido em um determinado número de subproblemas que devem ser estruturalmente análogos ao problema original, mas em tamanho menor;
- (P2) **CONQUISTAR:** tais subproblemas são solucionados através da estratégia recursiva de que se dispõe;
- (P3) **COMBINAR:** as soluções dos subproblemas são combinadas a fim de obter uma solução do problema original.

Uma estratégia que, geralmente, é utilizada para implementar este paradigma é dividir uma instância em a partes de comprimento $\frac{n}{b}$, ou seja, escolhe-se a quantidade de subproblemas e divide-se em tamanhos iguais, com excessão de um pedaço que pode complementar com a parte relacionada ao resto da divisão de n por b .

Há três condições básicas, pelo menos, que indicam a utilização de uma estratégia D&C com sucesso, como se pode ver em Toffolo et al [91]:

- (1) Deve ser possível decompor uma instância em subinstâncias do problema;
- (2) A combinação dos resultados deve ser eficiente;
- (3) As subinstâncias devem ser razoavelmente do mesmo tamanho.

Além disso, é possível identificar, pelo menos, duas situações mais genéricas em que tal paradigma D&C torna-se adequado:

- Problemas nos quais um conjunto de operações são correlacionados ou repetidos como, por exemplo, a multiplicação de matrizes;
- Problemas de tomada de decisão que, com isso, particiona o problema em várias peças, geralmente disjuntas. Tal abordagem é muito interessante principalmente quando algumas dessas partes tornam-se irrelevantes.

Com isso, Toffolo et al. [91] sugerem um pseudocódigo genérico de um algoritmo D&C, como segue.

Algoritmo 8 divisao-e-conquista(X)

```

1: if  $X$  é pequeno ou simples then
2:   Resolva  $X$  e fornece a solução  $Y$ ;
3: else
4:   Decomponha  $X$  em  $n$  conjuntos menores  $X_0, \dots, X_{n-1}$ ;
5:   for  $i$  do  $0 \dots n - 1$ 
6:      $Y_i = \text{divisao-e-conquista}(X_i)$ ;
7:   end for
8:   Combine  $Y_0, \dots, Y_{n-1}$  em  $Y$ ;
9: end if return  $Y$ 

```

D.2 Vantagens

Quando estuda-se algoritmos para resolver algum problema, naturalmente, surgem perguntas do tipo: quais vantagens pode-se apontar sobre esse método em detrimento de outros já existentes? Usualmente, as vantagens estão relacionadas com baixo custo temporal de cálculo [14].

É possível, portanto, apontar algumas vantagens para o uso deste paradigma, apontando para melhor eficiência computacional na resolução de problemas, as quais estão descritas em [93].

(i) Resolver problemas difíceis

A primeira vantagem do paradigma D&C está em seu poder de ser uma ferramenta com grande poder de resolver problemas conceitualmente difíceis. Para isso, é preciso:

- encontrar apenas uma maneira eficiente de se dividir o problema original em subproblemas;
- determinar um modo recursivo (ou até mesmo iterativo) de se resolver os subproblemas;
- fornecer uma forma de se combinar as diferentes soluções parciais em uma solução para o problema original.

Como um exemplo: a Torre de Hanói [36]. É um problema difícil que se torna mais simples de resolver usando esta estratégia.

Um outro exemplo é o DMDGP, que também é um problema difícil de se resolver. Neste trabalho, encontramos uma maneira determinística de dividir a instância, a fim de tirar proveito da estrutura de distâncias, com o objetivo de poupar esforço computacional. Isso é feito através de uma partição no grafo da instância cujas partes estão adequadamente sobrepostas. Depois, resolvemos cada subproblema utilizando o algoritmo *Branch & Prune*. Por fim, unimos as partes já realizadas com o BP através de translações e rotações com Quaternios.

(ii) Eficiência Algorítmica

A segunda vantagem de uma abordagem D&C está relacionada com a eficiência computacional que, neste caso, é tempo. Por isso, este paradigma usualmente proporciona a descoberta de algoritmos eficientes, com forte tendência de possuir complexidade logarítmica [91].

Como um exemplo, considere o Algoritmo de Strassen para multiplicação de matrizes, publicado por Volker Strassen em [88]. Este algoritmo é útil para matrizes grandes, pois quebra as matrizes em blocos menores de maneira eficiente. Além disso, sua complexidade assintótica é $\mathcal{O}(n^{2.8074})$, enquanto que a complexidade assintótica da multiplicação clássica de matrizes é dada por $\mathcal{O}(n^3)$, evidentemente mais rápido [88].

(iii) Controle de Erros de arredondamento

A terceira vantagem mencionada aqui também está associada a eficiência computacional. Mas, ao invés de tempo de computação, esta fala sobre a precisão das soluções. Os cálculos nesta abordagem devem ser mais precisos na maioria das vezes, já que os erros de arredondamento não são acumulados em toda a estrutura da instância, como em uma abordagem iterativa, ficando reclusos aos subproblemas determinados pela divisão.

(iv) Paralelismo

Outra vantagem mencionada aqui é que algoritmos com esta roupagem são natural e facilmente adaptáveis para execução em máquinas com multiprocessadores. Em especial, isto

ocorre para sistemas com memória compartilhada, onde a comunicação de dados entre processadores não precisa ser planejada com antecedência, facilitando a combinação das sub-soluções [93]. Podemos dizer, portanto, que esses algoritmos são altamente paralelizáveis [91].

(v) Bom uso da Memória Cache

Por fim, a última vantagem dessa lista é que os algoritmos D&C, naturalmente, requerem um menor número de acessos à memória principal da máquina, pois estes fazem um uso eficiente da memória cache. Isto deve-se ao fato de que, uma vez que um subproblema é muito pequeno, ele e todos os seus subproblemas podem, em princípio, serem resolvidos dentro dessa memória cache, a qual fica mais próximo ao processador, tornando os cálculos mais rápidos [91, 93].

D.3 Exemplos de Aplicações

A técnica D&C serve de base para vários algoritmos eficientes que resolvem uma série de problemas. Segue, portanto, abaixo uma lista com alguns exemplos de problemas que são resolvidos com algum algoritmo deste tipo [91, 93].

(E1) Ordenação

Um exemplo de um algoritmo de ordenação eficiente que utiliza a estratégia D&C é o chamado *Mergesort*, desenvolvido por John Von Neumann em 1945 [46], o qual é baseado em comparações entre os números a serem ordenados.

(E2) Multiplicação de Grandes Números

O russo Anatolli Alexeevich Karatsuba, em colaboração com o também russo Yuri Petrovich Ofman, desenvolveram e publicaram, em 1962 [44], um algoritmo para a multiplicação de grandes números com a estratégia D&C, particionando em grupos menores de algarismos.

(E3) Cálculo da Transformada Rápida de Fourier

Uma estratégia D&C também foi utilizada, em 1965, por James Cooley, da IBM, e John Turkey, de Princeton, para reinventar um algoritmo desenvolvido por Gauss de modo a tornar eficientemente implementável em um computador. O objetivo deste algoritmo é calcular a chamada Transformada Rápida de Fourier (FFT) [13].

Anexo I

Licença

Copyright (c) 2015 de Felipe Delfini Caetano Fidalgo.

Exceto quando indicado o contrário, esta obra está licenciada sob a licença Creative Commons Atribuição-CompartilhaIgual 3.0 Não Adaptada. Para ver uma cópia desta licença, visite <http://creativecommons.org/licenses/by-sa/3.0/>.



A marca e o logotipo da UNICAMP são propriedade da Universidade Estadual de Campinas. Maiores informações sobre encontram-se disponíveis em <http://www.unicamp.br/unicamp/a-unicamp/logotipo/normas%20oficiais-para-uso-do-logotipo>.

I.1 Sobre a licença dessa obra

A licença Creative Commons Atribuição-CompartilhaIgual 3.0 Não Adaptada utilizada nessa obra diz que:

1. Você tem a liberdade de:

- Compartilhar — copiar, distribuir e transmitir a obra;
- Remixar — criar obras derivadas;
- fazer uso comercial da obra.

2. Sob as seguintes condições:

- Atribuição — Você deve creditar a obra da forma especificada pelo autor ou licenciatente (mas não de maneira que sugira que estes concedem qualquer aval a você ou ao seu uso da obra).
- Compartilhamento pela mesma licença — Se você alterar, transformar ou criar em cima desta obra, você poderá distribuir a obra resultante apenas sob a mesma licença, ou sob uma licença similar à presente.