Universidade Federal de Pernambuco
CCEN

Pós-graduação em Matemática Computacional

Formigas resolvem o problema do vetor mais curto

Marcelo Gama da Silva

Tese de Doutorado

Recife
Fevereiro de 2012

Universidade Federal de Pernambuco
CCEN

Marcelo Gama da Silva

Formigas resolvem o problema do vetor mais curto

Trabalho apresentado ao Programa de Pós-graduação em Matemática Computacional do CCEN da Universidade Federal de Pernambuco como requisito parcial para obtenção do grau de Doutor em Matemática Computacional.

Orientador: *Prof. PhD. Ruy José Guerra B. de Queiroz*

Recife
Fevereiro de 2012

Catalogação na fonte
Bibliotecária Monick Raquel Silvestre da S. Portes, CRB4-1217

S586f Silva, Marcelo Gama da
Formigas resolvem o problema do vetor mais curto / Marcelo Gama da Silva. – 2012.
87 f.: il., fig., tab.

Orientador: Ruy José Guerra B. de Queiroz.
Tese (Doutorado) – Universidade Federal de Pernambuco. CCEN. Matemática Computacional, Recife, 2012.
Inclui referências.

1. Otimização matemática. 2. Algoritmos. I. Queiroz, Ruy José Guerra B. de (orientador). II. Título.

519.6 CDD (23. ed.) UFPE- MEI 2016-149

MARCELO GAMA DA SILVA

FORMIGAS RESOLVEM O PROBLEMA DO VETOR MAIS CURTO

Tese apresentada ao Programa de Pós-graduação em Matemática Computacional do CCEN da Universidade Federal de Pernambuco como requisito parcial para obtenção do grau de Doutor em Matemática Computacional.

Aprovada em 20/03/2012

BANCA EXAMINADORA

_____

Prof. Dr. Ruy José Guerra Barreto de Queiroz (Orientador)

Centro de Informática – UFPE

_____

Profa. Dra. Anjolina Grisi de Oliveira (Examinador Interno)

Centro de Informática – UFPE

_____

Profa. Dra. Marcília Andrade Campos (Examinador Interno)

Centro de Informática – UFPE

_____

Prof. Dr. Fernando Menezes Campello de Souza (Examinador Externo)

Departamento de Eletrônica e Sistemas – UFPE

_____

Prof. Dr. Borko Stosic (Examinador Externo)

Departamento de Estatística e Informática – UFRPE

A meus filhos: Patricia, Carol e Arthur.

Agradecimentos

Ao Professor Ruy pela paciência, incentivo e orientação prestados.

Aos amigos Igor e Murilo pelo apoio durante o mestrado e a Francisco Valadares (Chicão) pelo incentivo e pelas discussões sobre algoritmos e complexidade.

Ao Centro de Informática da Universidade Federal de Pernambuco pela oportunidade que me foi dada para realizar este trabalho.

Finalmente, um agradecimento especial à minha esposa (Solange Santos) por toda paciência e dedicação durante minhas quase infindáveis horas de pesquisa e elaboração deste trabalho.

As pessoas passam suas vidas esperando que aconteça algo que vá mudar tudo. Elas buscam poder ou amor, e as respostas às suas maiores perguntas. Acredito que o que elas querem realmente é outra oportunidade. Algum jeito de levar outra vida, aonde todos os erros que cometeram possam ser apagados e elas possam começar tudo novamente, como se nada de ruim tivesse acontecido e todas as suas oportunidades estejam à frente delas.

—ALLIE CLARKE (Personagem da mini-série Taken)

Resumo

Uma característica da Matemáticas é servir de suporte, fornecendo ferramentas para diversas outras ciências. Entre essas ferramentas estão os algoritmos que resolvem, pelo menos de maneira aproximada o chamado Problema do vetor mais curto. Um algoritmo clássico desenvolvido para resolver esse problema é conhecido como Algoritmo LLL. Ele fornece, até agora, as melhores soluções aproximadas possíveis em uma região que chamamos de Região fundamental. O presente trabalho traz uma forma alternativa para a abordagem do problema do valor mais curto de modo que as soluções encontradas na região fundamental são melhores do que as produzidas pelo algoritmo LLL, no sentido de serem melhores aproximações das soluções exatas para o problema.

Palavras-chave: problema do vetor mais curto; algoritmo LLL; otimização em colônias de formigas.

Abstract

A feature of Mathematics is to serve as a support, providing tools for several other sciences. Among these tools are algorithms that solve, at least approximately, the so-called Shortest Vector Problem. A classic algorithm developed to solve this problem is known as the LLL Algorithm. It provides, so far, the best possible approximate solutions in a region we call the fundamental domain. The present work presents an alternative way to approach the shortest value problem so that the solutions found in the fundamental domain are better than those produced by the LLL algorithm, in the sense of being better approximations of the exact solutions to the problem.

Keywords: shortest vector problem; LLL algorithm; ant colony optimization.

Lista de Figuras

Lista de Tabelas

SUMÁRIO

CAPÍTULO 1

INTRODUÇÃO

Não deixaremos de explorar e
ao término da nossa exploração
deveremos chegar ao ponto de partida
e conhecer esse lugar pela primeira vez.
– T. S. ELIOT

—

O uso da criptografia teve início com os egípcios, ainda que de forma rudimentar, há cerca de 4000 anos. Desde então esse ramo da matemática aplicada não pára de crescer. Entretanto, foi a partir da década de 1960, com a proliferação dos computadores, que se tornaram cada vez mais comuns a comunicação e o comércio digital e com eles a procura por serviços seguros. Isto deu um novo impulso à criptografia, culminando em 1977 no RSA (desenvolvido por R. **R**ivest, A. **S**hamir e L. **A**dleman), que é um dos esquemas criptográficos mais utilizados atualmente.

Os processos de cifrar e decifrar mensagens são, em um certo sentido, inversos um do outro. Essa necessidade de inversão torna a estrutura algébrica de grupo um possível ambiente para o desenvolvimento da criptografia moderna. Atualmente existem inúmeros algoritmos criptográficos e que têm como suporte os mais variados exemplos de grupos.

Um dos principais criptossistemas utilizados atualmente é o RSA, definido sobre os grupos $\mathbb{Z}_N^*$, onde $N = pq$ é produto de dois primos. Trabalhos recentes indicam que tipos de primos são mais apropriados para garantir uma relativa segurança a esses criptossistemas. Questões sobre a segurança do RSA serão nosso principal objetivo nesse trabalho.

1.1 ORGANIZAÇÃO DO TRABALHO

O presente trabalho está organizado do seguinte modo:

No capítulo 2, *Uma breve viagem pelo mundo da criptografia*, como o próprio nome diz, faremos uma breve discussão sobre a evolução dos esquemas criptográficos. É nesse capítulo que será introduzida a terminologia utilizada atualmente em criptografia. Daremos também uma visão geral dos aspectos mais importantes da criptografia moderna.

O capítulo 3, *Fundamentos teóricos*, tem como objetivo apresentar os principais conceitos teóricos e resultados relacionados necessários para o desenvolvimento desse trabalho.

O capítulo 4 será dedicado ao estudo dos chamados *problemas computacionalmente difíceis*. Nesse capítulo estudaremos a relação entre esses problemas e a segurança de alguns dos criptossistemas associados a eles.

O capítulo 5, por sua vez, trata da noção de *grupos pseudo-livres*. Embora recente, esse conceito mostra-se promissor por permitir uma generalização de muitas das hipóteses criptográficas assumidas atualmente. O ponto principal desse capítulo é a prova de que o grupo $\mathbb{Z}_N^*$, sob certas condições, é pseudo-livre.

No capítulo 6, *Primos seguros*, faremos um estudo de algumas das principais propriedades desses números primos especiais. Os únicos exemplos que se conhece de grupos pseudo-livres são os grupos dos elementos inversíveis de $\mathbb{Z}_N^*$, onde N é produto de dois primos seguros. Isto evidencia a importância desses números para a criptografia RSA. Ainda nesse capítulo serão apresentados algoritmos para a geração de primos seguros, um dos quais é inédito.

O capítulo 7 será dedicado a algumas *Conclusões* relativas ao estudo de criptossistemas e previsões de *Trabalhos futuros*, destacando possíveis desdobramentos desta dissertação.

1.2 CONTRIBUIÇÕES DESTE TRABALHO

Devido à importância já mencionada dos primos seguros para a criptografia RSA, existe atualmente uma grande necessidade de algoritmos rápidos para a geração de tais números. Para isso diversas classes de algoritmos são utilizadas. Entretanto, praticamente todos eles têm nos testes de primalidade seu principal problema. Embora tenha sido provado em 2001 que é possível decidir se um inteiro é primo em tempo polinomial, os algoritmos para isso ainda são bastante ineficientes. A principal contribuição desse trabalho é o desenvolvimento de um novo algoritmo para a geração de primos seguros que evita grande parte dos testes de primalidade utilizados pelos algoritmos tradicionais.

CAPÍTULO 2

RETICULADOS

The Grid. A digital frontier.
I tried to picture clusters of information as they moved through the computer.
What did they look like? Ships? Motorcycles? Were the circuits like freeways?
I kept dreaming of a world I thought I'd never see. And then, one day, I got in.
– Kevin Flynn (Tron legacy)

—

2.1 INTRODUÇÃO

A utilização de estruturas matemáticas para modelar problemas reais há muito tornou-se algo bastante comum. Compreender e explorar características dessas estruturas constitui parte importante na solução de diversos problemas. Os reticulados estão entre esses objetos matemáticos e, apesar de sua aparente simplicidade, possuem uma estrutura bastante rica que permite uma grande variedade de aplicações. A seguir destacamos algumas dessas aplicações.

- **Otimização combinatória:** Os algoritmos para resolver problemas de programação linear inteira assintoticamente mais rápidos conhecidos até hoje são baseados em reticulados.
- **Criptografia:** Inúmeras primitivas criptográficas, tais como criptografia de chave pública, assinaturas digitais, cifragem resistente a ataques de vazamento de chaves, criptografia baseada em identidade, cifragem totalmente homomórfica, estão entre as aplicações de reticulados nessa área.
- **Complexidade:** O conhecimento da conexão entre caso-médio e pior-caso da complexidade de problemas definidos sobre reticulados inspira muitas de suas aplicações criptográficas.
- **Teoria dos números algébricos:** Esta constitui uma ferramenta importante na criação de reticulados com propriedades especiais o que os tornam úteis em diversas aplicações. Por outro lado, reticulados são utilizados no design de algoritmos para resolver problemas sobre números algébricos.
- **Comunicação wireless:** Espera-se uma substancial intensificação em pesquisas que melhorem a comunicação em redes sem fio para que se possa adequadamente acomodar

o crescente número de usuários e satisfazer o apetite de seus smart phones. Reticulados tem-se mostrado uma ferramenta muito útil no processo de codificação e decodificação de sinais [1, 2, 3].

Informamente, podemos descrever um reticulado como sendo o conjunto dos pontos que formam os vértices de uma grade no espaço n-dimensional.

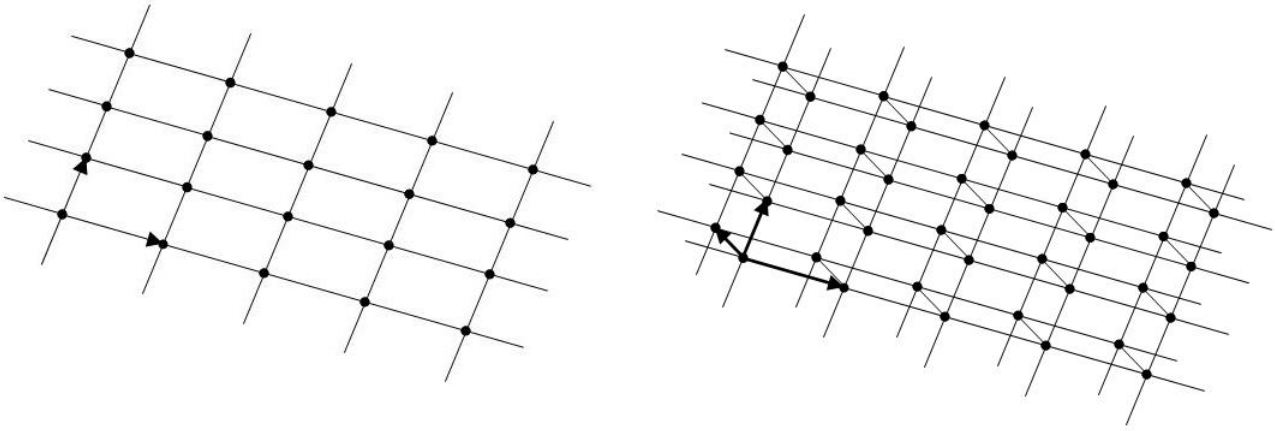

Figura 2.1 Reticulados: No plano (esq.) e no espaço (dir.)

Reticulados são objetos que já eram estudados no inicio do século XVIII por matemáticos como Gauss e Lagrange, entre outros. Recentemente tornaram-se objetos de interesse em outras áreas, principalmente em ciência da computação. Podem ser usados como ferramenta algorítmica para resolver diversos problemas, conforme o exposto anteriormente. Entretanto, a grande complexidade de alguns problemas definidos sobre reticulados os tornaram de grande interesse para aplicações em criptografia.

2.2 DEFINIÇÃO

Um reticulado Λ é gerado a partir de um conjunto linearmente independente de vetores. Outra maneira de imaginar reticulados é como sendo o conjuntos das soluções de um sistema de equações modulares $x \equiv a_i \pmod{n_i}$. A estrurura do grupo aditivo $\mathbb{Z}^n/\Lambda$ possibilita uma classificação para reticulados.

Dados n vetores $b_1, \cdots, b_n$, linearmente independentes no espaço $\mathbb{R}^m$, o conjunto de todas as conbinações lineares de coeficientes inteiros desses vetores será chamado de reticulado de dimensão m e posto n:

$$\Lambda = \Lambda(b_1, \cdots, b_n) = \{\sum_{i=1}^{n} \lambda_i b_i : \lambda_i \in \mathbb{Z}\}$$

O conjunto $\beta = \{b_1, \cdots, b_n\}$ será chamado de base do reticulado Λ.

Existe uma forma alternativa, mais conveniente em algumas situações, de se representar um reticulado. Considere a matriz $B = [b_1, \cdots, b_n] \in \mathbb{R}^{m \times n}$, onde os vetores da base são escritos como vetores colunas. Em notação matricial temos

$$\Lambda = \Lambda(B) = \{Bx : x \in \mathbb{Z}^n\}$$

onde Bx representa o produto da matriz B pela matriz coluna x. Nesse caso, diremos que B é a base do reticulado e que o reticulado é gerado por B.

Exemplo 2.1. *O conjunto $\mathbb{Z}^n = \{(x_1, \cdots, x_n) : x_i \in \mathbb{Z}\}$ é um reticulado que tem como base $I = [e_1, \cdots, e_n]$, onde*

$$e_i = (\underbrace{\overbrace{0, \cdots, 0, 1}^{n}, 0, \cdots, 0}_{i}).$$

Assim, $\mathbb{Z}^n = \Lambda(I)$, onde I é a matriz identidade $n \times n$.

Dizemos que duas bases, B_1 e B_2 são equivalentes se elas geram o mesmo retuculado. O resultado a seguir caracteriza bases equivalentes de um reticulado [6]:

Teorema (Bases equivalentes)
Duas bases, B_1 e B_2 são equivalentes se, e somente se, existe uma matriz unimodular inteira U (ou seja, com entradas inteiras e determinantte ± 1) tal que $B_2 = B_1 U$.

2.3 NORMAS

Muitos dos conceitos que estudaremos, como os determinantes na próxima seção, além de problemas definidos sobre reticulados, como o problema do vetor mais curto, estarão relacionados a algum tipo de medida. Precisamos então, definir a maneira pela qual medidas serão calculadas em um reticulado; isto será feito através do conceito de normas nos espaços $\mathbb{R}^m$.

Uma norma no espaço $\mathbb{R}^m$ é uma função $||\cdot|| : \mathbb{R}^m \longrightarrow \mathbb{R}$ que satisfaz às seguintes condições [7]:

❶ **(Positiva definida)** $||x|| \geq 0$, ocorrendo a igualdade apenas quando $x = 0$.

❷ **(Homogênea)** $||\alpha x|| = |\alpha| \cdot ||x||$

❸ **(Desigualdade triangular)** $||x+y|| \leq ||x|| + ||y||$

Dentre as normas de $\mathbb{R}^m$ a mais conhecida é, provavelmente, a norma euclideana que será denotada por ℓ_2. A norma euclideana do vetor $x = (x_1, \cdots, x_m) \in \mathbb{R}^m$ é dada por

$$||x||_2 = \sqrt{x_1^2 + \cdots + x_m^2}$$

Esta norma faz parte de uma familia importante de normas definidas sobre $\mathbb{R}^m$, conhecidas como normas ℓ_p ($p \geq 1$ ou $p = \infty$):

$$||x||_p = \left(\sum_{i=1}^{m} |x_i|^p \right)^{1/p}$$

Dois outros elementos bastante comuns da familia ℓ_p são as normas ℓ_1 e ℓ_∞, assim definidos

$$||x||_1 = \sum_{i=1}^{m} |x_i| \qquad \text{e} \qquad ||x||_\infty = \lim_{p \to \infty} ||x||_p = \max_{i=1}^{p} |x_i|$$

Evidentemente, diversas outras normas podem ser definidas em $\mathbb{R}^m$; algumas um tanto exóticas, como a do exemplo a seguir [43]:

Exemplo 2.2. *A expressão a seguir define uma norma em* $\mathbb{R}^4$*:*

$$||x|| = 2|x_1| + \sqrt{3|x_2|^2 + (\max(|x_3|, 2|x_4|))^2}, \qquad \text{onde} \quad x = (x_1, x_2, x_3, x_4).$$

Do ponto de vista geométrico, diferentes normas dão origem a diferentes objetos, mesmo que suas definições sejam similares. Tomemos como exemplo o círculo unitário em $\mathbb{R}^m$, que é definido por $C_\alpha = \{x \in \mathbb{R}^m : ||x||_\alpha \leq 1\}$. A figura a seguir mostra a região C_α para três diferentes normas $|| \cdot ||_\alpha$ em $\mathbb{R}^2$.

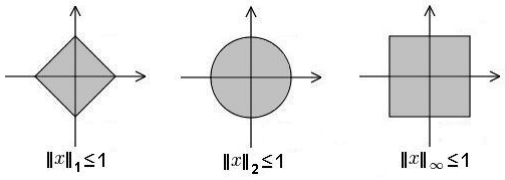

Figura 2.2 Círculos unitários relativos a diferentes normas

Em questões de caráter topológico, como continudade e convergência, diferentes normas podem comportar-se de modo semelhante. Isso acontece, por exemplo, quando temos normas equivalentes:

Definição 2.1. *(Normas equivalentes)*

Duas normas $||\cdot||_\alpha$ e $||\cdot||_\beta$ são ditas equivalentes quando existem constantes $A, B > 0$ tais que, para todo $x \in \mathbb{R}^m$ tem-se

$$A||x||_\alpha \leq ||x||_\beta \leq B||x||_\alpha$$

Exemplo 2.3. *As normas $||\cdot||_1$, $||\cdot||_2$ e $||\cdot||_\infty$ são equivalentes. Isto ocorre devido às seguintes desigualdades [7]:*

- Equivalência entre ℓ_1 e ℓ_2: $||x||_2 \leq ||x||_1 \leq \sqrt{n}\,||x||_2$
- Equivalência entre ℓ_1 e ℓ_∞: $||x||_\infty \leq ||x||_1 \leq n\,||x||_\infty$
- Equivalência entre ℓ_2 e ℓ_∞: $||x||_\infty \leq ||x||_2 \leq \sqrt{n}\,||x||_\infty$

De fato, em espaços vetoriais de dimensão finita (como é o caso de $\mathbb{R}^m$), é conhecido que todas as normas são equivalentes. Elon Lages Lima trata diversas questões importantes sobre normas e espaços métricos em [7].

A partir do conceito de norma pode-se definir a noção de distância entre dois vetores em $\mathbb{R}^m$, que será bastante utilizada nesse trabalho.

Definição 2.2. *(Distância entre vetores)*

Fixada uma norma $||\cdot||_p$ em $\mathbb{R}^m$ e dados dois vetores $x, y \in \mathbb{R}^m$, sua distância será dada por

$$d(x,y) = ||x-y||_p$$

A função $d : \mathbb{R}^m \times \mathbb{R}^m \longrightarrow \mathbb{R}$ é chamada de *métrica* ou *distância* em $\mathbb{R}^m$ e herda algumas propriedades da respectiva norma:

❶ **(Positiva definida)** $d(x,y) \geq 0$, ocorrendo a igualdade apenas quando $x = y$

❷ **(Simetria)** $d(x,y) = d(y,x)$

❸ **(Desigualdade triangular)** $d(x,z) \leq d(x,y) + d(y,z)$

2.4 DETERMINANTES

Estudaremos a seguir dois conceitos de fundamental importância para a teoria dos reticulados. Muitos dos resultados sobre os problemas abordados aqui terão uma relação direta com o conceito de determinante. Por sua vez, o determinante é uma medida associada a um elemento geométrico, a chamada região fundamental de um reticulado. Este é outro conceito fundamental no nosso estudo.

Considere o reticulado $\Lambda = \Lambda(B)$. Seja $\mathscr{P}(B)$ região do $\mathbb{R}^n$ delimitada pelos vetores da base B, isto é,

$$\mathscr{P}(B) = \{Bx : x = (x_1, \cdots, x_n), 0 \leq x < 1\}.$$

Chamamos $\mathscr{P}(B)$de região fundamental (ou paralelepípedo fundamental) do reticulado Λ.

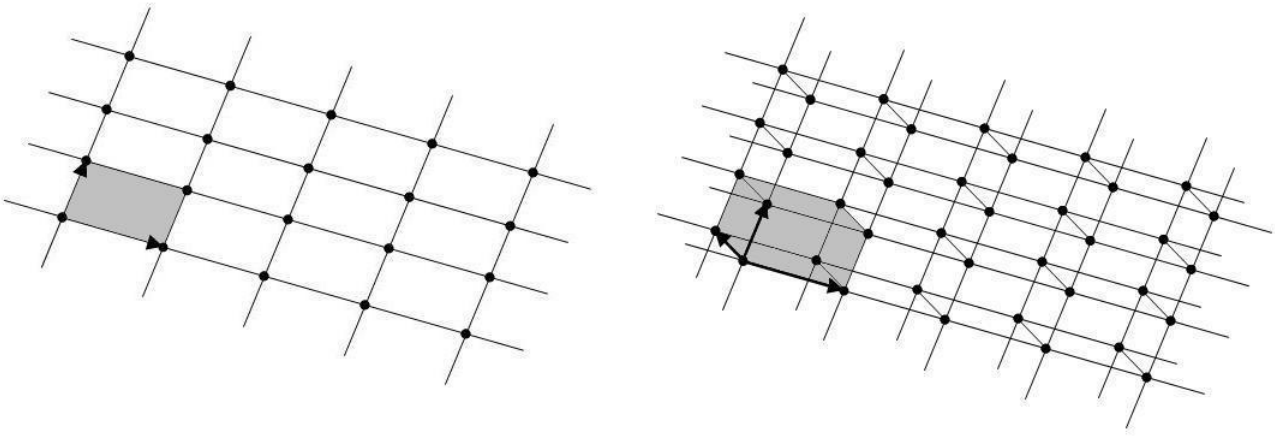

Figura 2.3 Paralelepípedo fundamental de um reticulado

Utilizando o conceito de paralelepípedo fundamental podemos estabeler condições para que B' represente uma base para Λ [6]:

Teorema

B' é uma base para $\Lambda(B)$ se, e somente se, $\mathscr{P}(B')$ não contém qualquer ponto desse reticulado, exceto a origem.

O determinante, $\det(\Lambda)$, de um reticulado $\Lambda = \Lambda(B)$ é definido como sendo o volume n-dimensional do paralelepípedo fundamental $\mathscr{P}(B)$. Pode-se provar que o determinante de um reticulado independe da base escolhida, ou seja, $\det(\Lambda)$ é um invariante do reticulado Λ. Isto segue do Teorema 1.1.

Dado um reticulado $\Lambda(B)$, onde $B = [b_1, \cdots, b_n]$, e considerando a norma ℓ_2, uma das maneiras de se calcular seu determinate utiliza o chamado processo de ortogonalização de Gram-Schmidt, o qual será descrito a seguir.

Dado um vetor $u \in \mathbb{R}^m$ podemos projetá-lo ortogonalmente sobre outro vetor $v \in \mathbb{R}^m$. Obtemos o vetor projeção

$$Proj_v^u = \alpha_{u,v} \cdot v,$$

onde $\alpha_{u,v} = \frac{\langle u,\, v \rangle}{\langle v,\, v \rangle}$ e $\langle x,\, y \rangle = \sum x_i y_i$ é o produto escalar, em $\mathbb{R}^m$, dos vetores $x = (x_1, \cdots, x_m)$ e $y = (y_1, \cdots, y_m)$ associado à norma $|| \cdot ||_2$. Nesse caso, vale a relação (ver [8] para uma prova)

$$\langle u,\, v \rangle = \frac{1}{4}\left(||u+v||^2 - ||u-v||^2\right).$$

Partindo de uma base $B = [b_1, \cdots, b_n]$, o processo de Gram-Schmidt consiste em construir indutivamente, a partir desses vetores, uma sequência $b_1^*, \cdots, b_n^*$ de de n vetores mutuamente ortogonais, ou seja, $\langle b_i^*,\, b_j^* \rangle = 0$ se $i \neq j$, da seguinte forma:

$$\begin{aligned}
b_1^* &= b_1 \\
b_2^* &= b_2 - Proj_{b_1^*}^{b_2} \\
b_3^* &= b_3 - Proj_{b_1^*}^{b_3} - Proj_{b_2^*}^{b_3} \\
&\cdots\cdots\cdots\cdots\cdots\cdots\cdots\cdots \\
b_k^* &= b_k - Proj_{b_1^*}^{b_k} - \cdots - Proj_{b_{k-1}^*}^{b_k}
\end{aligned}$$

Note que, para cada j, o vetor b_j^* é a componente de b_j ortogonal a $b_1, \cdots, b_{j-1}$. Dessa forma, $\langle b_i,\, b_j^* \rangle = 0$ quando $i < j$.

A figura a seguir ilustra o primeiro passo do processo de ortogonalização de Gram-Schmidt

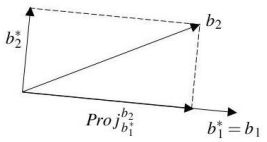

Figura 2.4 Primeiro passo do processo de Gram-Schmidt

Não é necessariamentre verdade que $B^* = [b_1^*, \cdots, b_n^*]$ representa uma base para o reticulado Λ, nem há garantias de que os vetores b_i^* estejam nesse reticulado. De fato, nem todo reticulado possui uma base onde os vetores são mutuamente ortogonais. Entretanto, a região

$$\mathscr{P}^* = \{B^* x : x = (x_1, \cdots, x_n), 0 \leq x < 1\}$$

tem o mesmo volume da região fundamental $\mathscr{P}(B)$. Isto segue do fato que cada b_i^* é combinação linear dos b_j's e das propriedades dos determinantes de matrizes, uma vez que volumes podem ser computados através desses determinantes.

Como os vetores b_i^* são mutuamente ortogonais, o determinante de Λ pode ser calculado através da expressão

$$\det(\Lambda(B)) = \prod_{i=1}^{n} ||b_i^*||$$

O determinante de um reticulado pode ser calculado sem a utilização do processo de Gram-Schmidt. Entretanto, nesse caso, o cálculo envolve o determinante de uma matrriz $n \times n$, o que pode ter um elevado custo computacional. Sendo $B = [b_1, \cdots, b_n]$ a base de $\Lambda(B)$, tem-se

$$\det(\Lambda(B)) = \sqrt{\det(B^T B)}$$

onde B^T é a matriz transposta da matriz B (ver [6]).

2.5 PROBLEMAS DEFINIDOS SOBRE RETICULADOS

A estrutura dos reticulados permite o estudo de uma grande variedade de problemas, conforme discutido no início do capítulo. Os problemas definidos sobre reticulados são de interesse tanto da matemática pura, quanto da matemática aplicada, enquanto que a dificuldade de resolver tais problemas é objeto de crescente interesse em criptografia. Os criptosistemas NTRU [9] e o de Ajtai-Dwork [10] são exemplos de aplicações da chamada criptografia baseada em

reticulados. A seguir daremos uma lista com alguns dos principais problemas definidos sobre reticulados. Inicialmente, fixaremos uma norma $||\cdot||$ em $\mathbb{R}^m$. Todas as medidas a seguir serão relativas a essa norma.

❶ **Problema do vetor mais curto (SVP: Shortest vector problem)**

Dada uma base B para um reticulado $\Lambda = \Lambda(B)$, encontrar um vetor não nulo mais curto possível em Λ.

❷ **GapSVP**

O SVP pode ser formulado como um problema de busca (versão descrita no ítem anterior), como um problema de otimização ou como um problema de decisão. O GapSVP é uma versão aproximada do problema decisional SVP e também é chamado de "problema da distância mínima".

Fixado um valor para o parâmetro γ, dados uma base B para o reticulado Λ e um número real d, o problema GapSVP_γ consiste em decidir entre

- SIM: $\min\{||v|| : v \in \Lambda - \{\vec{0}\}\} \leq d$
- NÃO: $\min\{||v|| : v \in \Lambda - \{\vec{0}\}\} > \gamma d$

Caso esse mínimo esteja entre d e γd, pode-se tomar qualquer decisão.

❸ **Problema do vetor mais próximo (CVP: Closest vector problem)**

Dada uma base B para um reticulado $\Lambda = \Lambda(B) \subset \mathbb{R}^m$ e um vetor $v_0 \notin \Lambda$, encontrar um vetor em Λ mais próximo possível de v_0.

❹ **GapCVP**

Sua formulação é análoga ao GapSVP. Dados uma base B para o reticulado Λ, um vetor $v_0 \in \mathbb{R}^m$ e números reais γ e d, o problema GapCVP_γ consiste em decidir entre

- SIM: $\min\{||v - v_0|| : v \in \Lambda\} \leq d$
- NÃO: $\min\{||v - v_0|| : v \in \Lambda\} > \gamma d$

Caso $\min\{||v - v_0|| : v \in \Lambda\}$ esteja entre d e γd, pode-se tomar qualquer decisão.

❺ **Problema dos vetores independentes mais curtos (SIVP: Shortest independent vector problem)**

Dada uma base $B = [b_1, \cdots, b_n]$ do reticulado Λ, encontrar n vetores linearmente independentes $v_1, \cdots, v_n \in \Lambda$ tais que $\max_i ||v_i|| < \max_i ||b_i||$.

❻ **Decodificação de distância limitada (BDD: Bounded distance decoding)**

Dado um vetor $v_0 \in \mathbb{R}^m$ cuja distância de Λ é no máximo $\frac{1}{2}\min\{||v|| : v \in \Lambda - \{\vec{0}\}\}$, encontrar $v \in \Lambda$ mais próximo de v_0.

❼ **Problema do raio de coberura (CRP: Covering radius problem)**

Dada uma base B de um reticulado Λ, encontrar a maior distância possível de um vetor $v \in \mathbb{R}^m$ ao reticulado.

❽ **Problema da base mais curta (SBP: Shortest basis problem)**

Dada uma base B do reticulado $\Lambda = \Lambda(B)$, encontrar uma base equivalente B' com vetores os mais curtos possíveis.

Dentre os problemas acima, abordaremos nesse trabalho o problema do vetor mais curto. No próximo capítulo estudaremos em detalhes diversos algoritmos que atacam esse problema. Alguns desses resolvem exatamente o SVP, entretanto são ineficientes devido a sua complexidade. Algoritmos mais eficientes foram propostos, mas são capazes de resolvê-lo apenas de forma aproximada.

Mais adiante atacaremos o problema do vetor mais curto utilizando uma técnica conhecida como "otimização em colônias de formigas", conhecida pela sigla ACO (ant colony optimization), onde o algoritmo simula o comportamento de indivíduos em uma colônia de formigas interagindo de modo a produzir uma solução para o problema.

CAPÍTULO 3

O PROBLEMA DO VETOR MAIS CURTO

Se o problema tem solução, não esquente a cabeça, porque tem solução.
Se o problema não tem solução, não esquente a cabeça, porque não tem solução.
– PROVÉRBIO CHINÊS

—

Estudaremos neste capítulo um dos mais importantes problemas computacionais associados a reticulados. Em ciência da computação sua importância está ligada, principalmente, às suas aplicações em criptografia. De fato, esse problema e sua complexidade computacional compõem o núcleo de alguns esquemas criptográficos modernos.

Usaremos o termo SVP (shortest vector problem) para designar o problema do vetor mais curto. Neste capítulo estudaremos diversos resultados importantes sobre o SVP, bem como os principais algoritmos utilizados para resolvê-lo, tanto de forma exata quanto aproximada.

O problema do vetor mais curto pode ser enunciado de três formas, dependendo do que se quer determinar: O vetor mais curto, seu comprimento ou se esse comprimento é menor que um dado número. Enunciaremos, portanto, esse problema em suas três variantes.

Definição 3.1. *(Problema do vetor mais curto)*

São dadas uma base B de um reticulado $\Lambda = \Lambda(B)$ e uma norma $||\cdot||$ em $\mathbb{R}^m$. As três variantes do problema do vetor mais curto são

- **Busca:** Encontrar um vetor $v_0 \in \Lambda - \{\mathbf{0}\}$ tal que $||v_0|| \leq ||v||$ para qualquer $v \in \Lambda - \{\mathbf{0}\}$.
- **Otimização:** Encontrar $\min\limits_{\mathbf{0} \neq v \in \Lambda} ||v||$
- **Decisão:** Dado $r \in \mathbb{Q}$, determinar se $\min\limits_{\mathbf{0} \neq v \in \Lambda} ||v|| \leq r$.

Do ponto de vista computacional, espera-se que os vetores possam ser representados por um número finito de bits. Dessa forma, devemos nos restringir a vetores cujas coordenadas sejam números racionais. Por outro lado, utilizando um fator conveniente, podemos reescalar todos os vetores da base do reticulado para que esses tenham coordenadas inteiras. Assumimos, então, que B tem todas as entradas em $\mathbb{Z}$.

Veremos que os algoritmos que resolvem o SVP de forma exata são, em geral, extremamente ineficientes, de um modo que discutiremos mais adiante. É comum então a busca por soluções aproximadas para problemas com essa característica. Nesses casos, são definidas versões do problemas com um parâmetro de aproximação $\gamma \geq 1$.

Definição 3.2. *(Problema do vetor mais curto aproximado $-$ SVP_γ)*

São dadas uma base B de um reticulado $\Lambda = \Lambda(B)$ e uma norma $||\cdot||$ em $\mathbb{R}^m$. As três variantes do problema de aproximação do SVP são

- **Busca:** Encontrar um vetor $v \in \Lambda - \{\mathbf{0}\}$, tal que $||v|| \leq \gamma$.
- **Otimização:** Encontrar d tal que $d \leq \min_{\mathbf{0} \neq v \in \Lambda} ||v|| \leq \gamma d$
- **Decisão (ou compromisso):** Dado $r \in \mathbb{Q}$
  - retornar SIM se $\min_{\mathbf{0} \neq v \in \Lambda} ||v|| \leq r$
  - retornar NÃO se $\min_{\mathbf{0} \neq v \in \Lambda} ||v|| > \gamma r$.

Essa última versão aproximada é conhecida como GapSVP_γ. Problemas chamados de "compromisso" são definidos sobre dois conjuntos disjuntos de entradas, que chamaremos de "instâncias SIM" e "instâncias NÃO". O algoritmo deve decidir corretamente de qual conjunto foi selecionada uma dada instância. Entretanto, para algums problemas esses dois conjuntos não cobrem todas as entradas possíveis. Portanto, nesses casos existirão entradas ilegais onde o algoritmo terá comportamento indefinido.

Inicialmente estudaremos o SVP a partir de um contexto mais geral, utilizando o conceito de mínimos sucessivos em um reticulado. Isto permitirá a obtenção de cotas, ou limitantes, para o comprimento do vetor mais curto.

3.1 MÍNIMOS SUCESSIVOS

Fixaremos uma norma $||\cdot||$ em $\mathbb{R}^m$. Sejam $P \in \mathbb{R}^m$ e $0 < r \in \mathbb{R}$. Uma bola de centro P e raio r em $\mathbb{R}^m$ é o análogo de um círculo em $\mathbb{R}^2$, ou seja, é o conjunto dos pontos de $\mathbb{R}^m$ cuja distância até P é no máximo r:

- Bola aberta em $\mathbb{R}^m$: $B(P,r) = \{x \in \mathbb{R}^m : ||x - P|| < r\}$
- Bola fechada em $\mathbb{R}^m$: $B[P,r] = \{x \in \mathbb{R}^m : ||x - P|| \leq r\}$

Figura 3.1 Bola aberta no $\mathbb{R}^2$ (esq.) e bola fechada no $\mathbb{R}^3$(dir.)

O comprimento de um vetor mais curto em um reticulado é, usualmente, denotado por λ_1. Podemos redefinir λ_1 como sendo o menor número real r tal que os pontos do reticulado que estão dentro da bola de centro na origem e raio r geram um espaço de dimensão no mínimo 1. Podemos generalizar o mínimo λ_1 da seguinte forma:

Definição 3.3. *(Mínimos sucessivos)*
Seja Λ um reticulado de posto n. Os mínimos sucessivos $\lambda_1, \cdots, \lambda_n$, desse reticulado, são definidos por

$$\lambda_i = \lambda_i(\Lambda) = \inf\{r : \dim(\text{spam}(\Lambda \cap \overline{B}(\mathbf{0}, r))) \geq i\}$$

O valor λ_i é o raio de uma bola no $\mathbb{R}^m$, centrada na origem, que contém, no mínimo, i vetores do reticulado Λ que são linearmente independentes. Nesse caso, o comprimento de um vetor mais curto em Λ é λ_1.

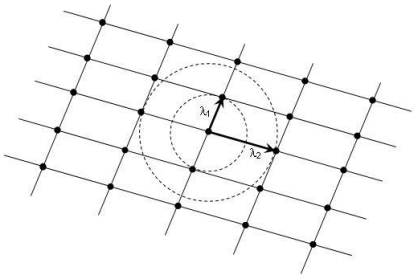

Figura 3.2 Mínimos sucessivos λ_1 e λ_2 no $\mathbb{R}^2$

Veremos a seguir dois resultados importantes sobre mínimos sucessivos de um reticulado. O primeiro provê uma cota infereior para λ_1 enquanto que o segundo mostra que λ_1 é a menor distância entre dois vetores no reticulado.

Teorema

Seja $B = \{b_1 \ \cdots \ b_n\}$ uma base para o reticulado $\Lambda = \Lambda(B)$ e sejam $b_1^*, \cdots, b_n^*$ obtidos de B através do processo de ortogonalização de Gram-Schmidt. Então

$$\lambda_1(\Lambda) \geq \min_{1 \leq i \leq n} ||b_i^*|| > 0,$$

onde $||\cdot|| = ||\cdot||_2$ é a norma euclideana em $\mathbb{R}^m$.

Prova. Seja $x = (x_1, \cdots, x_n) \in \mathbb{Z}^n$. Então $Bx = \sum_{i=1}^{j} x_i b_i$ é um elemento genérico do reticulado Λ. Tome j como sendo o maior índice tal que $x_j \neq 0$. Temos

$$\begin{aligned}
\langle\, Bx,\, b_j^* \,\rangle &= \langle \sum_{i=1}^{j} x_i b_i,\, b_j^* \,\rangle && \text{(i)}\\
&= x_j \langle\, b_j,\, b_j^* \,\rangle && \text{(ii)}\\
&= x_j \langle\, b_j^* + \sum_{i=1}^{j-1} \mu_{ij} b_i^*,\, b_j^* \,\rangle && \text{(iii)}\\
&= x_j \langle\, b_j^*,\, b_j^* \,\rangle + x_j \sum_{i=1}^{j-1} \mu_{ij} \langle\, b_i^*,\, b_j^* \,\rangle && \text{(iv)}\\
&= x_j ||\, b_j^* ||^2 && \text{(v)}
\end{aligned}$$

onde, em (ii) usamos o fato que $\langle\, b_i,\, b_j^* \,\rangle = 0$ para $i < j$; em (iii), $\mu_{ij} = \frac{\langle\, b_j, b_i^* \,\rangle}{\langle\, b_i^*, b_i^* \,\rangle}$; enquanto que (v) é consequência da ortogonalidade mútua dos vetores b_i^* em (iv) e da propriedade da norma euclideana $\langle\, v,\, v \,\rangle = ||v||_2^2$.

Pela desigualdade de Cauchy-Schwarz, tem-se $|\langle\, Bx,\, b_j^* \,\rangle| \leq ||Bx|| \cdot ||b_j^*||$. Como b_j^* é componente de uma base de Λ, $b_j^* \neq 0$. Pela escolha de j e sendo x_j inteiro, temos $|x_j| \geq 1$. Assim,

$$||Bx|| \geq \frac{|\langle\, Bx,\, b_j^* \,\rangle|}{||b_j^*||} = \frac{|x_j| \cdot ||b_j^*||^2}{||b_j^*||} \geq ||b_j^*|| \geq \min_{1 \leq i \leq n} ||b_i^*||$$

Sendo Bx um elemento genérico do reticulado, concluimos que $\lambda_1(\Lambda) \geq \min_{1 \leq i \leq n} ||b_i^*|| > 0$.

♢

Teorema
$\lambda_1(\Lambda)$ é a menor distância entre dois elementos do reticulado Λ.

Prova. Na prova do teorema anterior vimos que qualquer vetor $Bx \in \Lambda$ satisfaz $||Bx|| \geq ||b_j^*||$, onde j foi convenientemente escolhido, porém dependendo de x. Sendo $v, w \in \Lambda$, tem-se $v - w \in \Lambda$. Concluímos que, para algum k, $||v - w|| \geq ||b_k^*|| \geq \lambda_1$.

♢

3.2 LIMITES SUPERIORES PARA OS MÍNIMOS SUCESSIVOS

Nesta seção estamos particularmente interessados em obter cotas superiores para os mínimos sucessivos λ_i. Os teoremas de Minkowski são importantes pois fornecem um limite superior para o primeiro mínimo λ_1 bem como para a média geométrica dos mínimos sucessivos de um reticulado.

Iniciamos com dois resultados que serão utilizados na prova dos teoremas de Minkowski.

No primeiro, utilizaremos a noção de conjunto mensurável. Embora haja uma definição precisa desse conceito, podemos pensar num conjunto mensurável como sendo aquele ao qual é associada algum tipo de medida. Aqui, a medida que nos interessa será o volume n-dimensional da região compreendida por este.

Teorema (Blichfeldt)

Seja B base do reticulado Λ e seja $\text{ger}(\Lambda) = \{Bx : x \in \mathbb{R}^n\}$ o espaço vetorial sobre $\mathbb{R}$ que tem B como base. Dado um conjunto mensurável $S \subseteq \text{ger}(\Lambda)$, se $\text{vol}(S) > \det(\Lambda)$, então existem pontos distintos $z_1, z_2 \in S$ tais que $z_2 - z_1 \in \Lambda$.

Uma representação geométrica desse teorema no plano pode ser vista na figura abaixo. Note que as áreas das quatro partes que compões a região S cobrem, com sobras, todo o parelepípedo fundamental $\mathscr{P}$ do reticulado; portanto, $\text{vol}(S) > \text{vol}(\mathscr{P}) = \det(\Lambda)$.

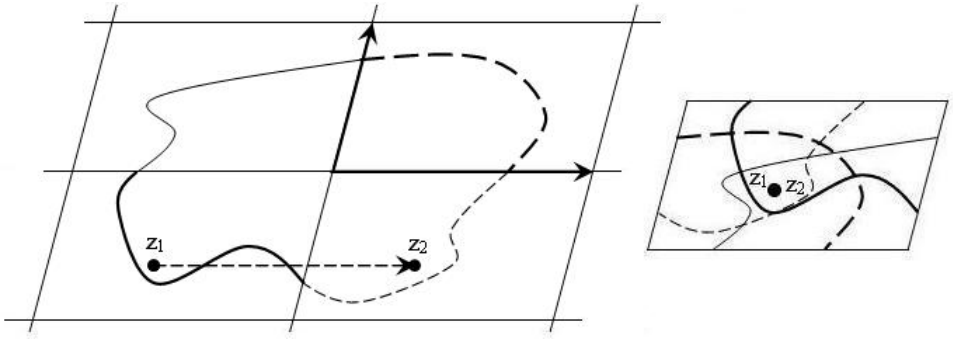

Figura 3.3 Teorema de Blichfeldt no plano: $z_2 - z_1 \in \Lambda$

Prova. Seja $\mathscr{P} = \{Bx : 0 \leq x_i < 1\}$ o paralelepípedo fundamental de Λ e, para cada $v \in \Lambda$, considere a translação $\mathscr{P} + v = \{p + v : p \in \mathscr{P}\}$. Dado $v \in \Lambda$, defina $S_v = S \cap (\mathscr{P} + v)$. Como $\{\mathscr{P} + v : v \in \Lambda\}$ cobre todo o espaço gerado por B, temos $S = \cup_{v \in \Lambda} S_v$. Mostraremos que os conjuntos S_v são disjuntos e, nesse caso, $\{S_v : v \in \Lambda\}$ será uma partição de S.

Suponha que existem $u, v \in \Lambda$ com $S_u \cap S_v \neq \varnothing$. Tomando $w \in S_u \cap S_v$, podemos escrever

$$w = p_1 + u = p_2 + v, \text{ com } p_1, p_2 \in \mathscr{P}.$$

Então $p_1 - p_2 = v - u \in \Lambda \cap \mathscr{P}$. Como o único elemento do reticulado que está em $\mathscr{P}$ é o vetor nulo, obtemos $p_1 - p_2 = \mathbf{0}$. Assim, $v - u = \mathbf{0}$, ou seja, $v = u$. Portanto, $S_u = S_v$.

Como Λ é um conjunto contável, concluímos que

$$\text{vol}(S) = \sum_{v \in \Lambda} \text{vol}(S_v).$$

Defina agora os conjuntos transladados

$$T_v = S_v - v = \{w - v : w \in S_v\} = (S - v) \cap \mathscr{P}$$

Da definição de S_v, concluímos que se $v \in \Lambda$, então $T_v \subseteq \mathscr{P}$ e, de $T_v = S_v - v = \{w - v : w \in S_v\}$, tiramos $\text{vol}(T_v) = \text{vol}(S_v)$.

Diferentemente dos S_v, os conjuntos T_v não são dois a dois disjuntos. Como $T_v \subseteq \mathscr{P}$, temos $\cup_{v\in\Lambda} T_v \subseteq \mathscr{P}$ e assim, se os T_v fossem dois a dois disjuntos, teríamos

$$\sum_{v\in\Lambda} \text{vol}(T_v) = \text{vol}\left(\cup_{v\in\Lambda} T_v\right) \leq \text{vol}(\mathscr{P})$$

Por outro lado,

$$\sum_{v\in\Lambda} \text{vol}(T_v) = \sum_{v\in\Lambda} \text{vol}(S_v) = \text{vol}(S) > \det(\Lambda),$$

pela hipótese do teorema. Combinando essas duas expressões obtemos

$$\text{vol}(\mathscr{P}) \geq \sum_{v\in\Lambda} \text{vol}(T_v) > \det(\Lambda),$$

o que contraria a definição $\det(\Lambda) = \text{vol}(\mathscr{P})$. Portanto, os conjuntos T_v não são dois a dois disjuntos. Resta encontrar $z_1, z_2 \in S$ com $z_2 - z_1 \in \Lambda$.

Pelo exposto acima, existem $u, v \in \Lambda$, distintos, tais que $T_u \cap T_v \neq \varnothing$. Tome $w \in T_u \cap T_v$ e faça $w_1 = z + u$ e $w_2 = z + v$. Sendo $u \neq v$, temos $w_1 \neq w_2$. Além disso, $w_1 \in S_u$, $w_2 \in S_v$ e assim $w_1, w_2 \in S$. Para concluir, observamos que $w_2 - w_1 = v - u \in \Lambda$.

$\diamondsuit$

O resutado a seguir é uma consequência do teorema de Blichfeldt e será utilzado na obtenção de uma cota superior para o comprimento do vetor mais curto em um reticulado.

Teorema (Teorema do corpo convexo − Minkowski)

Seja Λ um reticulado de posto n. Seja $S \subseteq \text{ger}(\Lambda)$ um conjunto convexo e simétrico com relação à origem. Se $\text{vol}(S) > 2^n \det(\Lambda)$, com relação à norma ℓ_2, então S contém algum ponto não nulo do reticulado.

Prova. Faça $T = \{v : 2v \in S\}$. Temos

$$\text{vol}(T) = 2^{-n}\text{vol}(S) > \det(\Lambda).$$

Pelo teorema de Blichfeldt, existem $z_1, z_2 \in T$, distintos, com $z_2 - z_1 \in \Lambda$. Temos ainda $2z_1, 2z_2 \in S$. Sendo S simétrico com relação à origem, temos $-2z_1 \in S$. Pela convexidade de S, o segmento que liga $2z_2$ e $-2z_1$ está inteiramente contido em S. Portanto, seu ponto médio está em S, ou seja,

$$\frac{2z_2 + (-2z_1)}{2} = z_2 - z_1 \in S.$$

Como z_1 e z_2 são distintos, o vetor $v = z_2 - z_1 \in \Lambda$ é um vetor não nulo em S.

♢

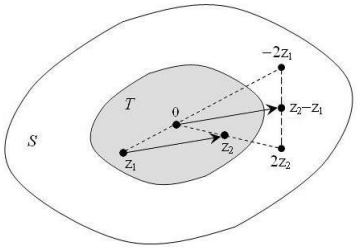

Figura 3.4 Teorema do corpo convexo

Agora dispomos das ferramentas necessárias para estabelecer limites para os mínimos sucessivos de um reticulado. Isto será feito através dos teoremas de Minkowski. O primeiro desses teoremas fornece uma cota superior para o primeiro mínimo λ_1, ou seja, para o comprimento do menor vetor não nulo. O segundo teorema de Minkowski estabelece uma cota superior para a média geométrica dos mínimos sucessivos.

Teorema (Primeiro teorema de Minkowski)

Seja $\Lambda = \Lambda(B)$ um reticulado de posto n. O comprimento λ_1 do menor vetor não nulo, medido com relação à norma euclideana ℓ_2, satisfaz $\lambda_1 \leq \sqrt{n}\det(\Lambda)^{1/n}$.

Prova. Considere a bola aberta $S = B(\mathbf{0}, r)$ de centro na origem e raio r em $\text{ger}(\Lambda)$, o espaço vetorial sobre $\mathbb{R}$ gerador por B. Então o hipercubo $[-\frac{r}{\sqrt{n}}, \frac{r}{\sqrt{n}}]^n$, de arestas $\frac{2r}{\sqrt{n}}$ está condido em S. Assim, $\text{vol}(S) > \left(\frac{2r}{\sqrt{n}}\right)^n$. Tomando $r = \sqrt{n}\det(\Lambda)^{1/n}$, teremos $\text{vol}(S) > 2^n \det(\Lambda)$. Pelo teorema do corpo convexo, S contém um ponto v do reticulado diferente da origem. Segue que

$$\lambda_1 \leq ||v|| \leq r = \sqrt{n}\det(\Lambda)^{1/n}.$$

♢

Teorema (Segundo teorema de Minkowski)

Sejam $\lambda_1, \cdots, \lambda_n$ os mínimos sucessivos, com relação à norma ℓ_2, de um reticulado Λ de posto n. Então

$$\sqrt[n]{\prod_{i=1}^{n} \lambda_i} < \sqrt{n}\det(\Lambda)^{1/n}$$

Prova. Sejam $b_1, \cdots, b_n \in \Lambda$ vetores linearmente independentes com $||b_i|| = \lambda_i$. Sejam $b_1^*, \cdots, b_n^*$ obtidos dos b_i's pelo processo de ortogonalização de Gram-Schmidt e tome esses vetores como um sistema ortogonal de eixos. Para um dado vetor y, sua componente na direção do eixo b_i^* é dada por $\frac{\langle y, b_i^* \rangle}{||b_i^*||}$. Definiremos, nesse sistema de eixos, um elipsóide E cujo eixo na direção b_i^* mede λ_i:

$$E = \left\{ y \in \mathbb{R}^n : \sum_{i=1}^{n} \left(\frac{\langle y,\ b_i^* \rangle}{||b_i^*||\ \lambda_i} \right)^2 < 1 \right\}$$

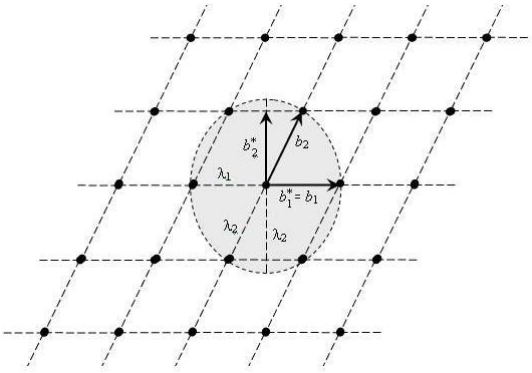

Figura 3.5 Construção do elipsóide E

Agora observamos que E não contém outro ponto do reticulado além do vetor nulo. Para ver isso, tome $\mathbf{0} \neq y \in \Lambda$ e seja k, o maior inteiro com $1 \leq k \leq n$ e $||y|| \geq \lambda_k$. Então os vetores $b_1, \cdots, b_k, y$ não podem ser linearmente independentes, pois teríamos $k+1$ vetores linearmente independentes menores que λ_{k+1}; portanto,

$$y \in \text{ger}(b_1, \cdots, b_k,) = \text{ger}(b_1^*, \cdots, b_k^*,)$$

Sendo assim, se $y = (y_1, \cdots, y_n)$, então $y_{k+1} = \cdots = y_n = 0$ e

$$\sum_{i=1}^{n} \left(\frac{\langle y,\ b_i^* \rangle}{||b_i^*||\ \lambda_i} \right)^2 = \sum_{i=1}^{k} \left(\frac{\langle y,\ b_i^* \rangle}{||b_i^*||\ \lambda_i} \right)^2 \geq \frac{1}{\lambda_k^2} \sum_{i=1}^{k} \left(\frac{\langle y,\ b_i^* \rangle}{||b_i^*||} \right)^2 = \frac{||y||^2}{\lambda_k^2} \geq 1$$

ou seja, $y \notin E$. Segundo o teorema do corpo convexo, devemos ter $\text{vol}(E) \leq 2^n \det(\Lambda)$. Entretanto, como as medidas dos eixos de E são os λ_i, temos

$$\text{vol}(E) = \left(\prod_{i=1}^{n} \lambda_i \right) \text{vol}(B(\mathbf{0}, 1)) \geq \left(\prod_{i=1}^{n} \lambda_i \right) \left(\frac{2}{\sqrt{n}} \right)^n$$

Finalmente,

$$\left(\prod_{i=1}^{n} \lambda_i \right) \left(\frac{2}{\sqrt{n}} \right)^n \leq \text{vol}(E) \leq 2^n \det(\Lambda) \Rightarrow \left(\prod_{i=1}^{n} \lambda_i \right)^{1/n} \leq \sqrt{n} \det(\Lambda)^{1/n}$$

♢

O restante deste capítulo será dedicado a um breve estudo sobre os principais algoritmos criados para resolver o SVP. Discutiremos também a eficiência desses algoritmos em termos de sua complexidade e, em alguns casos, a qualidade das soluções produzidas.

3.3 COMPLEXIDADE DE ALGORITMOS

A complexidade de um algoritmo é, essencialmente, uma medida da quantidade de operações fundamentais que serão feitas para a sua execução. Isto é refletido no tempo de execução do mesmo, o que permite avaliar quando um dado algoritmo é útitl, em termos práticos, para resolver determinado problema. Para dar uma medida dessa complexidade faremos antes uma breve discussão sobre notação assintótica e classes de complexidade.

3.3.1 Notação assintótica

Notação Θ

Dada uma função real g, definimos

$$\Theta(g) = \{f : \exists c_1, c_2 > 0, \exists n_0 \in \mathbb{N} \text{ com } 0 \leq c_1 g(n) \leq f(n) \leq c_2 g(n),\ \forall n \geq n_0\}$$

Intuitivamente, podemos pensar em $\Theta(g)$ como sendo o conjunto de todas as funções que têm crescimento limitado inferiormente por $c_1 g$ e superiormente por $c_2 g$ a partir de um certo ponto. Quando $f \in \Theta(g)$ é comum escrever $f = \Theta(g)$.

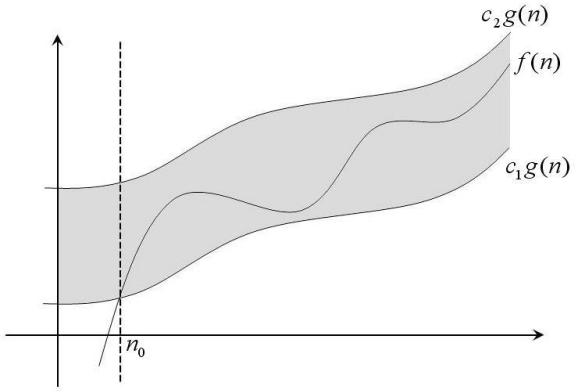

Figura 3.6 $f = \Theta(g)$

Notação O

Dada uma função real g, definimos

$$O(g) = \{f : \exists c > 0, \exists n_0 \in \mathbb{N} \text{ com } 0 \leq f(n) \leq cg(n),\ \forall n \geq n_0\}$$

Podemos pensar em $O(g)$ como sendo o conjunto de todas as funções que têm crescimento limitado superiormente por cg a partir de um certo ponto. A notação $f = O(g)$ indica que $f \in O(g)$.

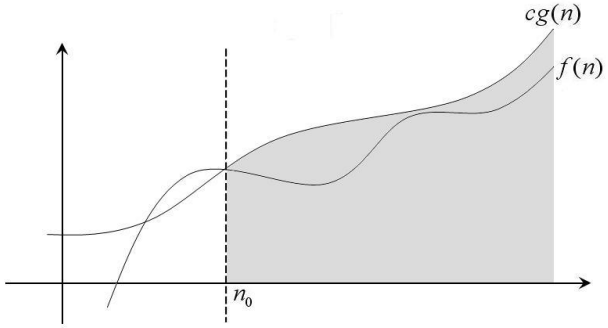

Figura 3.7 $f = O(g)$

Notação Ω

Dada uma função real g, definimos

$$\Omega(g) = \{f : \exists c > 0, \exists n_0 \in \mathbb{N} \text{ com } 0 \leq cg(n) \leq f(n),\ \forall n \geq n_0\}$$

Nesse caso, $\Omega(g)$ é o conjunto de todas as funções que têm crescimento limitado inferiormente por cg a partir de um certo ponto. Também utiliza-se a notação $f = \Omega(g)$ para indicar $f \in \Omega(g)$.

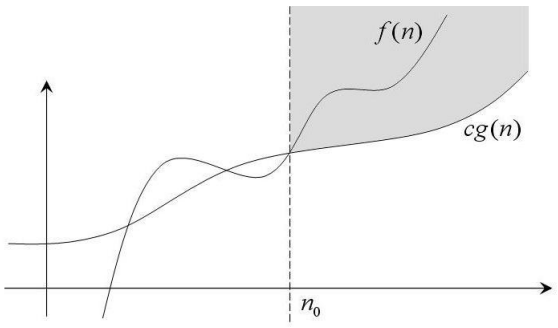

Figura 3.8 $f = \Omega(g)$

Relações entre Θ, O e Ω

Das definições de Θ, O e Ω seguem as seguintes conclusões:

- $f \in \Theta(g) \Rightarrow f \in O(g)$. Portanto, $\Theta(g) \subseteq O(g)$.
- $f \in \Theta(g) \Rightarrow f \in \Omega(g)$. Dessa forma, $\Theta(g) \subseteq \Omega(g)$.
- $f \in \Theta(g) \Leftrightarrow f \in O(g)$ e $f \in \Omega(g)$. Logo, $\Theta(g) = O(g) \cap \Omega(g)$.

3.3.2 Classes de complexidade

Para formalizar a noção de complexidade, pensaremos em um dado problema como sendo uma cadeia de caracteres que o descreve. Um conjunto de cadeias com alguma propriedade em comum será chamado de linguagem. Nesse contexto, a complexidade de uma linguagem (e também do problema a ela associado) será medida pela quantidade de recursos necessários para decidir se uma determinada cadeia pertence a essa linguagem. Normalmente essa medida é dada em função do número de certas operações fundamentais realizadas para isso.

3.3.2.1 Alfabetos, cadeias e linguagens

Um *alfabeto* é um conjunto finito de símbolos, que serão chamados de *letras*. Uma *cadeia de caracteres* (ou *string*) sobre um alfabeto é uma sequência finita de letras concatenadas desse alfabeto. O conjunto de todas as cadeias definidas sobre um alfabeto Σ é denotado por Σ^*. Frequentemente utiliza-se $\Sigma = \{0,1\}$.

O *comprimento* de uma cadeia w, denotado por $|w|$, é o número de letras que foram concatenadas para formá-la (contadas as repetições). Finalmente, um conjunto de palavras definidas sobre um alfabeto será chamado de *linguagem*.

3.3.2.2 Máquinas de Turing

Uma *máquina de Turing* é um dispositivo teórico, criado pelo matemático Alan Turing em 1936, com o objetivo de simular a lógica de algoritmos. Informalmente, uma máquina de Turing consiste de [41]:

1. Uma fita *infinita* dividida em células adjacentes.
2. Um *cabeçote* que pode ler, escrever ou apagar símbolos de uma alfabeto finito sobre a fita, bem como mover-se para a esquerda ou para a direita sobre a fita.
3. Uma tabela de instruções indicando que ação será executada a partir da configuração atual da máquina e do símbolo lido na fita. Essa tabela é conhecida como *função de transição*.
4. Um *registrador* que armazena o estado atual da máquina.

Definição 3.4. *(Máquina de Turing)*

Um máquina de Turing M é uma 7-upla $(Q, \Sigma, \Gamma, \delta, q_0, q_{ac}, q_{rej})$, onde

1. Q é o conjunto (finito) de estados possíveis da máquina.
2. Σ é um alfabeto que contém os símbolos que compõem as cadeias que são entradas para M. Esse alfabeto não contém o símbolo $\square$ (espaço em branco).
3. Γ é o alfabeto da fita, onde $\square \in \Gamma$ e $\Sigma \subseteq \Gamma$.
4. $\delta : Q \times \Gamma \to Q \times \Gamma \times \{E, D\}$ é a função de transição, onde E e D indicam, respectivamente, movimento do cabeçote para a esquerda ou para a direita.
5. $q_0 \in Q$ é o estado inicial de M.
6. q_{ac} é o estado de *aceitação*.
7. q_{rej} é o estado de *rejeição* ($q_{ac} \neq q_{rej}$).

Atingindo um dos estados q_{ac} ou q_{rej} a máquina M pára. Caso isto não ocorra, M entra em *loop*.

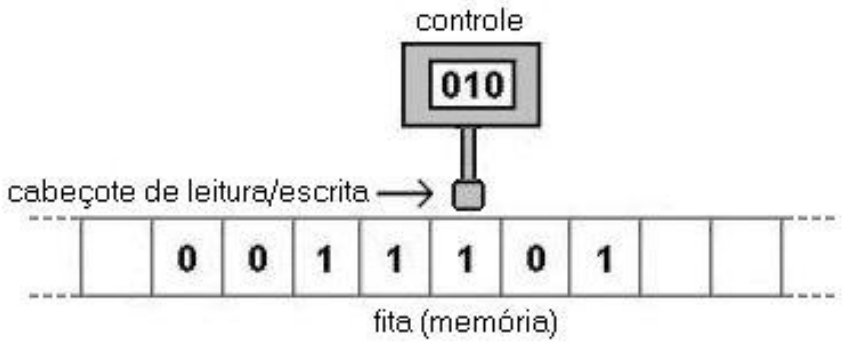

Figura 3.9 Máquina de Turing

3.3.2.3 Classe de complexidade P

Dizemos que uma linguagem L é reconhecida em *tempo polinomial determinístico* se existe uma máquina de Turing M, que retorna os valores 0 ou 1, e uma função polinomial $p : \mathbb{N} \to \mathbb{N}_+$ tais que

(1) Sobre uma entrada x, M pára após no máximo $p(|x|)$ passos, onde $|x|$ é o comprimento da cadeia x;

(2) $M(x) = 1$ se, e somente se, $x \in L$.

A classe de complexidade P é o conjunto das linguagens reconhecidas em tempo polinomial determinístico. De outro modo, podemos dizer que P é o conjunto dos problemas de decisão que podem ser resolvidos em tempo polinomial.

3.3.2.4 Classe de complexidade NP

Uma linguagem L está em NP quando existe uma relação binária $R_L \subseteq \{0,1\}^* \times \{0,1\}^*$ e uma função polinomial $p : \mathbb{N} \to \mathbb{N}_+$ tais que

1. R_L pode ser reconhecida em tempo polinomial determinístico;
2. $x \in L$ se, e somente se, existe $y \in \{0,1\}^*$ tal que $(x,y) \in R_L$ e $|y| \leq p(|x|)$.

Um tal y é chamado de *testemunha* da pertinência de x a L. A definição acima pode ser interpretada do seguinte modo:

Uma linguagem L está em NP se existem "provas curtas" de pertinência a L que podem ser eficientemente verificadas . O termo "prova curta" está associado à condição

$$\exists y \in \{0,1\}^* \text{ tal que } (x,y) \in L, \text{ com } |y| \leq p(|x|).$$

A eficiência vem do fato que isto pode ser verificado em tempo polinomial.

Uma outra maneira de interpretar NP é como sendo o conjunto de todos os problemas de decisão que podem ser resolvidos por uma máquina de Turing não-determinística em tempo polinomial.

Pode-se (a grosso modo) dizer que P é a classe dos problemas cujas soluções são "fáceis de encontrar" e NP é a classe dos problemas cujas soluções são "fáceis de serem verificadas". Muitos dos teóricos em teoria da complexidade acreditam que P$\neq$NP (ver [28]). Este é o mais famoso problema em aberto em Ciência da Computação.

3.3.2.5 NP-Completude

Dizemos que uma linguagem L é NP-completa se L está em NP e se toda linguagem em NP é redutível a L em tempo polinomial. Uma linguagem L' é redutível a uma linguagem L se existe uma função computável f tal que $x \in L'$ se, e somente se, $f(x) \in L$; para indicar esse fato usamos a notação $L \leq_{poly} L'$.

Alguns dos problemas NP-completos mais conhecidos são [30, 36, 44]:

Satisfatibilidade de uma fórmula proposicional (SAT)

Chama-se *literal* uma variável proposicional ou a negação de uma variável proposicional. Uma disjunção de literais recebe o nome de *cláusula*. Uma fórmula proposicional está na *forma normal conjuntiva* (FNC) se esta é uma conjunção de cláusulas. Sabe-se que toda fórmula proposicional é equivalente a uma fórmula na forma normal conjuntiva.

Chamaremos de SAT ao conjunto de todas as fórmulas na forma normal conjuntiva que são satisfatíveis, ou seja, existe uma atribuição de valores às variáveis proposicionais que tornam a fórmula verdadeira. O problema SAT consiste, por definição, em verificar se uma expressão booleana na FNC é satisfatível. Em 1971 Stephen Cook provou que o SAT é NP-completo.

Problema da mochila (knapsack problem)

Dado um conjunto de ítens, cada um tendo peso p_i e valor v_i, determinar a quantidade de cada ítem que se pode tomar de modo que o peso total não exceda um dado valor p e o valor total seja o máximo possível.

Problema do caixeiro viajante

Um caixeiro viajante, que se desloca entre pares de cidades a um custo que depende apenas do par escolhido, recebe um prêmio por cada cidade visitada e paga uma multa por cada cidade que deixa de visitar. Deseja-se estabelecer uma rota que minimize os custos de viagem e a soma das multas pagas, além de permitir que ele receba um dado prêmio pré-fixado.

Problema da clique máxima em um grafo

Uma *clique*[1] em um grafo não-direcionado é um conjunto C de vértices que são dois a dois adjacentes. Dizemos que uma clique C é *maximal* se não existe outra clique que a contenha. Quando não existe uma clique C_1 que seja maior (em quantidade de vértices) que C, então C é *máxima*. O problema de encontrar uma clique máxima em um grafo não-direcionado é NP-completo.

Problema da soma de subconjuntos

Dado um conjunto X de inteiros e um inteiro s, existe um subconjunto não vazio de X cuja soma de seus elementos seja s?

[1] Aqui o termo clique tem um significado pouco conhecido. Significa conspiração; subgrupo organizado no interior de um grupo cujos membros partilham preferências específicas; "panelinha".

Coloração de grafos (usando no mínimo 3 cores)

Assumiremos que *colorir* um grafo significa atribuir cores aos seus vértices de modo que dois vértices adjacentes tenham cores distintas. O problema da coloração de grafos está associado a diversos problemas de ordem prática como, por exemplo, programação, registro de alocação em compiladores e reconhecimento de padrões.

Determinar se é possível colorir os vértices de um grafo usando no máximo k cores é um problema NP-completo.

3.3.2.6 Problemas NP-difíceis

Um problema é chamado de NP-difícil quando existe algum problema NP-completo que pode ser reduzido a ele em tempo polinomial. O mais conhecido dos problemas NP-difíceis é o *problema da parada*. Este foi um dos primeiros problemas que se provou ser indecidível. A prova desse fato, dada por Alan Turing, foi publicada em 1936 [41].

Problema da parada

Dada a descrição de um programa e uma entrada finita, decidir se o programa pára ou entra em loop para aquela entrada.

Turing provou que não existe um algoritmo geral para resolver o problema da parada para todos os possíveis pares programa-entrada.

3.4 ALGORITMOS QUE RESOLVEM O SVP

Nosso objetivo nesta seção é estudar algoritmos que resolvem o problema do vetor mais curto. A abordagem adotada seguirá de perto as idéias expostas em [6]. Um algoritmo pode resolver o SVP de forma exata ou aproximada. Em geral, algoritmos que resolvem exatamente o SVP são de alto custo computacional, o que os torna extremamente lentos. Por outro lado, algoritmos de aproximação são mais rápidos, mas podem produzir soluções não tão próximas de uma solução exata.

3.4.1 O algoritmo de Gauss (SVP em dimensão 2)

O primeiro algoritmo que estudaremos será o algoritmo de Gauss. Sua importância reside no fato que este produz uma solução exata para o SVP em dimensão 2. Em sua versão generalizada, a construção independe da norma escolhida.

Por toda essa subseção adotaremos uma norma qualquer $||\cdot||$ em $\mathbb{R}^m$. O algoritmo de Gauss baseia-se nas propriedades das chamadas bases reduzidas de reticulados bidimensionais.

Definição 3.5. *(Bases reduzidas)*

Seja Λ um reticulado contido em $\mathbb{R}^2$. Uma base $[u,v]$ de Λ é reduzida, com relação à norma $||\cdot||$, se

$$||u||, ||v|| \leq ||u+v||, ||u-v||$$

Geometricamente, essa definição significa que, no paralelogramo gerado por u e v, os lados não são maiores que as diagonais. A utlização de bases reduzidas é justificada pelo resultado a seguir, cuja prova pode ser encontrada em [6]:

Teorema (Bases reduzidas e mínimos sucessivos)

A base $[u,v]$ de um reticulado $\Lambda \subset \mathbb{R}^2$, de posto 2, é reduzida se, e somente se, os comprimentos de u e v são os dois mínimos sucessivos λ_1 e λ_2 do reticulado.

Para descrever o algoritmo de Gauss precisamos ainda do conceito de base bem ordenada, que será definido a seguir:

Definição 3.6. *(Base bem ordenada)*

Dizemos que uma base $[u,v]$ de um reticulado Λ é bem ordenada quando

$$||u|| \leq ||u-v|| < ||v||$$

A seguir, temos uma descrição do algoritmo generalizado de Gauss:

Algoritmo 3.1: Redução de base generalizado (Gauss)

Entrada: $u, v \in \Lambda$, vetores linearmente independentes
$||\cdot||$, norma em $\mathbb{R}^m$

Saída: Base reduzida para $\Lambda([u,v])$

início
(1) **se** $||u|| > ||v||$ **então** troca(u,v)
(2) **se** $||u-v|| > ||u+v||$ **então** $v \leftarrow -v$
(3) **se** $||v|| \leq ||u-v||$ **então**
(4) retorne $[u,v]$
sair
fim se
(5) **se** $||u|| \leq ||u-v||$ **então**
(6) **enquanto** $[u,v]$ não é reduzida **faça**
(7) Encontre $\mu \in \mathbb{Z}$ tal que $||v-\mu u||$ é mínimo
(8) **se** $||u-v|| > ||u+v||$ **então** $v \leftarrow -v$
(9) troca(u,v)
fim enquanto
(10) retorne $[u,v]$
sair
fim se
(11) **se** $||u|| = ||v||$ **então**
(12) retorne $[u,u-v]$
sair
fim se
(13) retorne $[v-u,u]$
fim

Esse algoritmo é bastante eficiente. De fato, ele pode ser executado em tempo polinomial conforme afirma o teorema a seguir (ver [6]):

Teorema (Complexidade do algoritmo de Gauss)

Dada uma norma eficientemente computável (isto é, em tempo polinomial) $||\cdot||$, o algoritmo de Gauss, recebendo como entradas dois vetores linearmente independentes u e v, produz, em tempo polinomial, uma base $B = \{u_1, v_1\}$ para $\Lambda([u,v])$ tal que $||u_1|| = \lambda_1$ e $||v_1|| = \lambda_2$. Em particular, o SVP em dimensão 2 está em P.

A prova desse teorema, embora simples, requer uma análise detalhada das diversas situações que podem ocorrer com a base dada e, por esse motivo, será omitida.

3.4.2 SVP em dimensão maior que 2

Nesta seção apresentaremos o principal algoritmo que resolve o SVP em dimensões maiores que 2, porém de forma aproximada. Desenvolvido por A. Lenstra, H. Lenstra e L. Lovász em 1982, esse algoritmo foi batizado de LLL. O tamanho do candidato a vetor mais curto produzido pelo LLL, em um reticulado de posto n, está numa faixa que vai de λ_1 até $\gamma(n)\lambda_1$, onde $\gamma(n) = (2/\sqrt{3})^n$.

O fator exponencial $\gamma(n)$ faz com que o erro se torne cada vez maior à medida que a dimensão n cresce. Em dimensão $n = 20$, por exemplo, o vetor produzido pode ser quase 18 vezes maior do que o procurado; enquanto que, em dimensão $n = 30$, esse fator é quase 75, conforme pode ser visto na tabela abaixo:

Fator de erro ($\gamma(n)$) no algoritmo LLL

n	$\gamma(n)$	n	$\gamma(n)$	n	$\gamma(n)$	n	$\gamma(n)$	n	$\gamma(n)$
1	1.1547	7	2.7371	13	6.4879	19	15.3786	25	36.4531
2	1.3333	8	3.1605	14	7.4915	20	17.7577	26	42.0924
3	1.5396	9	3.6494	15	8.6505	21	20.5049	27	48.6041
4	1.7778	10	4.2140	16	9.9887	22	23.6770	28	56.1232
5	2.0528	11	4.8659	17	11.5340	23	27.3398	29	64.8055
6	2.3704	12	5.6187	18	13.3183	24	31.5693	30	74.8309

Apesar de ter um fator de erro exponencial associado, o LLL é um dos principais algoritmos utilizados na busca de soluções para o SVP. Por que isto ocorre? A resposta para essa pergunta reside no fato que o LLL é um algoritmo absolutamente geral, no sentido de não exigir nenhuma particularidade com relação à base do reticulado.

Para descrever o LLL é necessário redefinir a noção de base reduzida, descrita quando estudamos o algoritmo de Gauss. Vamos fixar a norma ℓ_2 e analisar geometricamente o significado da condição $||u||, ||v|| \leq ||u+v|| \, ||u-v||$:

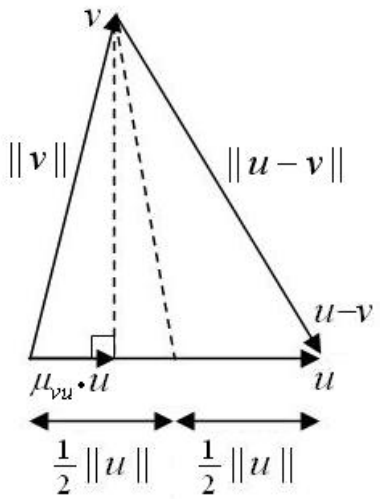

Figura 3.10 Bases reduzidas: Interpretação geométrica

Consideraremos apenas a condição $||v|| \leq ||u-v||$; os demais casos são análogos. Nesse caso, o ponto de encontro da altura no triângulo da figura acima (que delimita a projeção de v sobre u) ocorrerá mais próximo de v que de $u-v$ ou, no melhor dos casos, exatamente mo ponto médio da base u (quando $||v|| = ||u-v||$). Portanto, o coeficiente que indica a fração de u que compõe a projeção será $\mu_{vu} \leq 1/2$.

Agora podemos redefinir o conceito de bases reduzidas:

Definição 3.7. *(Redefinindo bases reduzidas em dimensão 2)*

A base $B = [u, v]$ de um reticulado é reduzida se

(i) $||u|| \leq ||v||$ (ii) $\mu_{vu} \leq 1/2$

onde μ_{vu} é o coeficiente de Gram-Smidt $\frac{\langle v, u \rangle}{\langle u, u \rangle}$.

Pode-se provar que as duas definições de base reduzida são equivalentes, desde que se utilize a norma ℓ_2.

O próximo passo é generalizar esse conceito para reticulados em dimensão superior. Nosso ponto de partida será o processo de ortogonalização de Gram-Schmidt. Vimos no capítulo anterior que, dada a base $B = [b_1, \ldots, b_n]$ do reticulado Λ, o pocesso de ortogonalização de Gram-Schmidt produz vetores mutuamente ortogonais $b_1^*, \ldots, b_n^*$ dados por

$$b_i^* = b_i - \sum_{j<i} \mu_{i,j} b_j^* \qquad \text{onde} \qquad \mu_{i,j} = \frac{\langle\, b_i,\, b_j^* \,\rangle}{\langle\, b_j^*,\, b_j^* \,\rangle}$$

Agora seja $S_i = \text{spam}(b_i^*, b_{i+1}^*, \ldots, b_n^*)$, com $1 \leq i \leq n$, o espaço gerado pelos vetores b_i^*, b_{i+1}^*, b_n^* sobre $\mathbb{R}$. Para cada i defina a projeção $\pi_i : \mathbb{R}^m \longrightarrow S_i$ por

$$\pi_i(x) = \sum_{j=i}^{n} \frac{\langle\, x,\, b_j^* \,\rangle}{\langle\, b_j^*,\, b_j^* \,\rangle}\, b_j^*$$

Note que, dado $x \in S_i$, a projeção $\pi_i(x)$ é a componente de x ortogonal a $b_1, \ldots, b_{i-1}$. Além disso, como $b_i = b_i^* + \sum_{j<i} \mu_{i,j} b_j^*$, temos

$$\pi_i(b_i) = \sum_{j=i}^{n} \frac{\langle\, b_i,\, b_j^* \,\rangle}{\langle\, b_j^*,\, b_j^* \,\rangle}\, b_j^* = \frac{\langle\, b_i^*,\, b_i^* \,\rangle}{\langle\, b_i^*,\, b_i^* \,\rangle} b_i^* = b_i^*$$

Finalmente, dispomos de todos os elementos necessários para estender a definição de base reduzida a reticulados em dimensões maiores que 2.

Definição 3.8. *(Bases reduzidas gerais)*

Uma base $B = [b_1, \ldots, b_n]$ de um reticulado é *LLL*-reduzida com parâmetro $\delta \in \mathbb{R}_+^*$, ou δ*LLL*-reduzida, se

- $|\mu_{i,j}| \leq 1/2$ para todo $i > j$
- $\delta ||\pi_i(b_i)||^2 \leq ||\pi_i(b_{i+1})||^2$ para todo $i < n$

Observamos que esta definição é, de fato, uma extensão do conceito de base reduzida em dimensão 2. Para $\delta = 1$ as duas condições podem ser reescritas como

$$\mu_{i+1,i} \leq 1/2 \quad \text{e} \quad ||\pi_i(b_i)|| \leq ||\pi_i(b_{i+1})||,$$

o que significa que a base $[\pi_i(b_i), \pi_i(b_{i+1})]$ é reduzida.

O resultado a seguir estabelece um intervalo ao qual deve pertencer o parâmetro δ, bem como uma cota superior para o tamanho do menor vetor de uma base δ*LLL*-reduzida.

Teorema (Estimativas para δ e $||b_1||$)

Seja $B = [b_1, \ldots, b_n]$ uma base δLLL-reduzida.

(i) Se $\delta > \frac{1}{4}$, então $||b_1|| \leq \left(\frac{2}{\sqrt{4\delta - 1}}\right)^{n-1} \lambda_1$.

(ii) Para $\delta = \frac{1}{4} + \left(\frac{3}{4}\right)^{\frac{n}{n-1}}$ tem-se $||b_1|| \leq \left(\frac{2}{\sqrt{3}}\right)^{n} \lambda_1$.

Prova. Como B é reduzida temos, para cada i,

$$\begin{aligned}
\delta||b_i^*||^2 &= \delta||\pi_i(b_i)||^2 && \text{(propriedade da projeção } \pi_i\text{)} \\
&\leq ||\pi_i(b_{i+1})||^2 && \text{(definição de base reduzida)} \\
&= \left|\left|\sum_{j=i}^{n} \frac{\langle\, b_{i+1},\, b_j^* \,\rangle}{\langle\, b_j^*,\, b_j^* \,\rangle} b_j^*\right|\right|^2 && \text{(definição de } \pi_i\text{)} \\
&= ||b_{i+1}^* + \mu_{i+1,i} b_i^*||^2 && \text{(pois } b_{i+1} = b_{i+1}^* + \sum_{j<i+1} \mu_{i,j} b_j^* \text{)} \\
&= ||b_{i+1}||^2 + \mu_{i+1,i}^2 ||b_i^*|| && \text{(teorema de Pitágoras: } b_i^* \text{ e } b_{i+1}^* \text{ são ortogonais)} \\
&\leq ||b_{i+1}||^2 + \frac{1}{4}||b_i^*|| && \text{(pois } \delta \in (\frac{1}{4}, 1)\text{)}
\end{aligned}$$

o que nos permite escrever

$$\left(\delta - \frac{1}{4}\right) ||b_i^*||^2 \leq ||b_{i+1}^*||^2.$$

Para $j < i$, considere as desigualdades

$$\begin{aligned}
\left(\delta - \frac{1}{4}\right) ||b_1^*||^2 &\leq ||b_2^*||^2 \\
\left(\delta - \frac{1}{4}\right) ||b_2^*||^2 &\leq ||b_3^*||^2 \\
&\cdots \\
\left(\delta - \frac{1}{4}\right) ||b_{i-1}^*||^2 &\leq ||b_i^*||^2
\end{aligned}$$

Como $b_i \neq 0$, para todo i, podemos multiplicar essas $i-1$ desigualdades para obter

$$\left(\delta - \frac{1}{4}\right)^{i-1} ||b_1^*||^2 \leq ||b_i^*||^2.$$

Assim,

$$|b_i^*|| \geq \left(\delta - \frac{1}{4}\right)^{\frac{i-1}{2}} ||b_1^*|| \geq \left(\delta - \frac{1}{4}\right)^{\frac{i-1}{2}} ||b_1||.$$

Utilizando a cota inferior para λ_1 estabelecida no Teorema 3.1, obtemos

$$\lambda_1 \geq \min_i ||b_i^*|| \geq \left(\delta - \frac{1}{4}\right)^{\frac{n-1}{2}} ||b_1^*||$$

Segue que se $\delta > \frac{1}{4}$, então $||b_1|| \leq \left(\frac{2}{\sqrt{4\delta-1}}\right)^{n-1} \lambda_1$, o que prova (i).

Finalmente, fazendo $\delta = \frac{1}{4} + \left(\frac{3}{4}\right)^{\frac{n}{n-1}}$ em (i), encontramos $||b_1|| \leq \left(\frac{2}{\sqrt{3}}\right)^n \lambda_1$, o que prova (ii). ♢

Para descrever o algoritmo de redução de base LLL, o qual estende o algoritmo de Gauus, inicialmente observaremos a estrutura geral deste último. O algoritmo de Gauss é, basicamente, composto de três passos:

- Passo 1 (Redução): $b_2 \leftarrow b_2 - cb_1$, com $c = \left\lceil \frac{\langle b_2, b_1 \rangle}{\langle b_1, b_1 \rangle} \right\rfloor$ (inteiro mais próximo).
- Passo 2 (Troca): Se $||b_1|| > ||b_2||$, fazer a troca $b_1 \leftrightarrow b_2$.
- Passo 3 (Loop): Se $[b_1, b_2]$ não é reduzida, repetir os passos 1 e 2.

Seguiremos as mesmas idéias na descrição do algoritmo de redução de base generalizado:

Algoritmo 3.2: Redução de base LLL

Entrada: $B = [b_1, \ldots, b_n]$ base de um reticulado
Parâmetro $\delta > \frac{1}{4}$

Saída: Base δLLL-reduzida para o reticulado $\Lambda(B)$

início

(1) **para** $i = 1, 2, \ldots, n$
  (2) **para** $j = i-1, \ldots, 1$
    (3) $\mu_{i,j} = \left\lceil \frac{\langle b_i, b_j \rangle}{\langle b_j, b_j \rangle} \right\rfloor$
    (4) $b_i \leftarrow b_i - \mu_{i,j} b_j$
(5) **se** $\delta ||\pi_i(b_i)|| > ||\pi_i(b_{i+1})||$, para algum i, **então**
    (6) troca: $b_i \leftrightarrow b_{i+1}$
(7) **senão** retorne B

fim

D. Micciancio e S. Goldwasser, em [6], fazem uma análise detalhada do funcionamento desse algoritmo. Nessa análise são estabelecidos três pontos importantes: Primeiro, $\delta \in (\frac{1}{4}, 1)$; segundo, o algoritmo produz um vetor mais curto no intervalo $[\lambda_1, \left(\frac{2}{\sqrt{3}}\right)^n \lambda_1]$. Finalmente, uma nova condição, $\delta < 1$, é adicionada para garantir que o algoritmo seja executado em tempo polinomial. Saber se isto também ocorre para $\delta = 1$ ainda é um problema em aberto.

No início desta seção foi destacada a importância do algoritmo LLL, embora produzindo soluções com uma grande margem de erro. Enfatizamos a característica de que ele não impõe nenhuma condição especial sobre a base inicial do reticulado. Entretanto, o fato de ser executado em tempo polinomial tem peso ainda maior na escolha desse algoritmo para resolver o SVP.

3.4.3 Outros algoritmos

Nesta seção veremos os principais algoritmos desenvolvidos para resolver o SVP. Alguns deles, como o LLL e o BKZ, foram formulados a partir de alguma teoria de redução de bases; porém, uma de suas consequências é a solução do SVP. Nosso interesse no momento é, principalmente, conhecer a complexidade desses algoritmos.

Na tabela abaixo são utilizados os seguintes parâmetros:

- n: posto do reticulado Λ.
- k: utilizado na definição de k-BKZ redução.
- ε: constante de ajuste; deve ser escolhido de modo que multiplicar dois números de ℓ-bits tenha complexidade $O(\ell^{\varepsilon})$.

Principais algoritmos que resolvem o SVP				
Nome	Ano	Exato/Aprox.	Complexidade	Fator de aprox.
LLL (Lenstra-Lenstra-Lovàsz), [13]	1982	Aproximado	$O(n^{5+2\varepsilon})$	$\left(\frac{2}{\sqrt{3}}\right)^n$
LLL ponto-flutuante (Schnorr-Euchner), [15]	1994	Aproximado	$O(n^{5+\varepsilon})$	$\left(\frac{2}{\sqrt{3}}\right)^n$
LLL segmento-forte (Koy-Schnorr), [16]	2002	Aproximado	$O(n^{3+\varepsilon}\log n)$	$\left(\frac{2}{\sqrt{3}}\right)^n$
semi $2k$-BKZ (Schnorr), [17]	1987	Aproximado	$O(n^3 k^{k+O(k)} + n^4)$	$\left(\frac{4}{3}k\right)^{\frac{n-1}{2k-1}}$
Random Sampling Reduction (Schnorr), [18]	2003	Aproximado	$O(n^3 \left(\frac{k}{6}\right)^{\frac{k}{4}} + n^4)$	$\left(\frac{k}{6}\right)^{\frac{n}{2k}}$
Primal-dual reduction (Koy), [19]	2004	Aproximado	$O(n^3 k^{\frac{k}{2}+O(k)} + n^4)$	$\left(\frac{k}{6}\right)^{\frac{n}{k}}$
Minkowski convex body (Kannan), [14]	1983	Exato	$2^{O(n\log n)}$	1
Crivo (Ajtai-Kumar-Sivakumar), [12]	2001	Exato (Random)	$2^{O(n)}$	1

Nos próximos capítulos será desenvolvido um novo algoritmo de aproximação para o SVP. Na sua construção utilizaremos o conceito de otimização em conônias de formigas, desenvolvido por M. Dorigo [22]. Esta técnica tem sido aplicada com sucesso em diversos problemas de otimização, permitindo o design de algoritmos probabilísticos que produzem “boas suluções” a custos computacionais “não muito elevados”.

CAPÍTULO 4

OTIMIZAÇÃO EM COLÔNIAS DE FORMIGAS

The ant is knowing and wise, but it doesn't know enough to take a vacation.
– CLARENCE DAY

—

Algumas sociedades de insetos e, em particular, colônias de formigas são caracterizadas por possurem uma estrutura social altamente organizada, embora sejam compostas de indvíduos bem simples. Essa organização permite que nessas sociedades sejam realizadas tarefas que excedem bastante a capacidade de qualquer um de seus indivíduos. Neste capítulo descreveremos em detalhes o modo como algumas dessas tarefas são coordenadas em colônias de formigas de modo a produzirem os resultados esperados. Nosso objetivo é estudar formas de simular esse comportamento em colônias de agentes artificiais, seguindo um modelo proposto em 1990 por Deneubourg e outros (ver [23]) para colônias de formigas.

Com essas idéias nasceu uma nova geração de algoritmos, conhecidos como "Algoritmos em colônias de formigas". Entre estes, uma classe destacou-se pela variedade de problemas onde podem ser aplicados: "Otimização em colônias de formigas" ou "Ant colony optimization" (ACO), desenvolvido por Marco Dorigo em sua tese de PhD [21], e que será nosso principal objeto de estudo neste capítulo.

4.1 FORMIGAS E OTIMIZAÇÃO

Grande parte das espécies de formigas tem visão pouco desenvolvida, embora existam exceções como a *Aenictus ceylonicus* que é completamente cega e a *Myrmecia* (ou *formiga bulldog*) australiana que tem uma visão excepcional. O modo como é contornado o problema de visão parcial ou sua ausência na execução de tarefas tem sido objeto de estudo de vários pesquisadores, dentre os quais se destacam o trabalho de Aron, Daneubourg, Goss e Pasteels [23].

Em uma colônia de formigas, a comunicação entre indivíduos ou entre estes e o meio ambiente, é feita através de substâncias químicas, chamadas de *feromônios*. Tipos especiais de feromônio são utilizados para criar as *trilhas de feromônio*, que marcam caminhos no chão (ou superfície onde se deslocam) e serão seguidas pelos indivíduos. É esse comportamento coletivo de "deixar feromônio" e "seguir feromônio" que faz com que tarefas como localizar uma fonte de alimento e trazê-lo até o formigueiro, entre outras, sejam apropriadamente executadas

e servem de inspiração para a construção de colônias artificiais.

Vários pesquisadores investigaram esse comportamento de "deixar feromônio" e "seguir feromônio" durante o deslocamento formigueiro - fonte de alimentos - formigueiro. Em [23], Aron, Deneubourg, Goss e Pasteels descrevem um experimento por eles desenvolvido e executado em uma colônia de formigas da espécie *I. humilis*, encontrada na Argentina. Nesse experimento, foi construída uma "ponte dupla" ligando formigueiro e fonte de alimentos e durante sua execução variou-se a razão $r = \ell_l/\ell_c$ entre a ramificação mais longa da ponte (ℓ_l) e a mais curta (ℓ_c). Descreveremos a seguir esses experimentos.

Experimento 1: Ramificações de igual comprimento

Inicialmente Deneubourg e sua equipe estudaram o caso $r = 1$, onde ambas as ramificações tem o mesmo comprimento. A figura a seguir mostra os resultados obtidos com esse experimento.

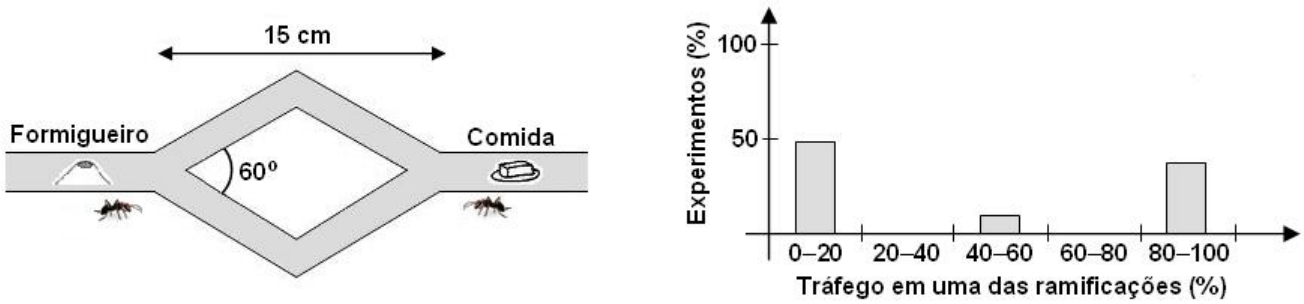

Figura 4.1 Experimento da ponte dupla com $r = 1$

Estes resultados podem ser interpretados da seguinte forma: Inicialmente as ramificações ainda não receberam feromônio. Assim, as formigas escolhem livremente uma dessas ramificações sem nenhuma influência do ambiente nem das outras formigas da colônia. Eventualmente, uma das ramificações irá se tornar um pouco mais povoada que a outra. Nesse momento, uma quantidade ligeiramente superior de feromônio em uma ramificação atrairá uma quantidade ligeiramente maior de formigas e esse processo, com o decorrer do tempo, tornará essa ramificação cada vez mais povoada. Conforme pode ser visto no gráfico acima, na maior parte dos experimentos as formigas escolhem uma ou outra ramificação em aproximadamente o mesmo número de experimentos.

Experimento 2: Uma ramificação é duas vezes maior que a outra

Em um segund experimento, eles estudaram o caso $r = 2$, onde uma ramificação tem o dobro do comprimento da outra. A figura a seguir mostra os resultados obtidos com esse novo experimento.

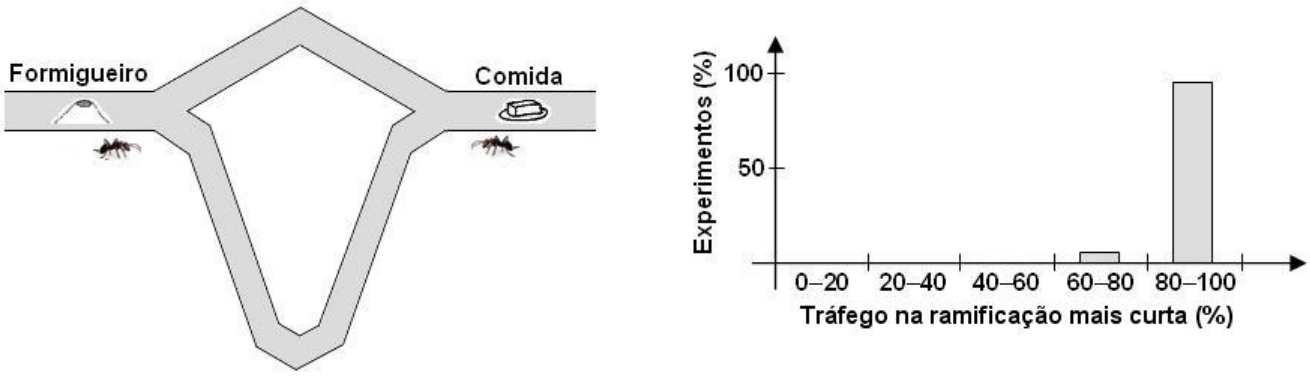

Figura 4.2 Experimento da ponte dupla com $r = 2$

Como no caso anterior, nesse experimento as formigas, inicialmente, não têm preferência por qualquer uma das ramificações enquanto estas não possuem feromônio. Entretanto, como será percorrida mais rapidamente, a ramificação menor passará a acumular quantidades cada vez maiores de feromônio em comprarção com a outra. Ao longo do tempo, esse processo atrairá mais e mais formigas para esta ramificação. No final do experimento, a maior parte das formigas terá convergido para a ramificação menor.

Uma característica importante foi observada por Deneubourg e sua equipe durante esse experimento; independentemente do valor de r, a convergência para uma das ramificações não era total. Uma pequena porcentagem das formigas ainda percorria o caminho mais longo. Esse fato foi interpretado por esses pesquisadores como uma forma encontrada pela colônia de, mesmo após encontrar o "menor caminho", ainda continuar a exploração do ambiente.

Nas próximas seções estudaremos dois modelos que reproduzem o comportamento de uma colônia de formigas no experimento da ponte dupla. O primeiro modelo representa um sistema estocástico contínuo e será descrito por um sistema de equações diferenciais, enquanto que o segundo é discreto e considera apenas o comportamento médio da colônia.

4.2 MODELANDO A COLÔNIA: SISTEMA CONTÍNUO

Com base nos resultados obtidos em seus experimentos, Deneubourg e sua equipe propuseram um modelo que descreve o comportamento das formigas durante o processo de procura e coleta de alimento. Alguns parâmetros serão necessários para a descrição desse modelo.

μ: Taxa de formigas por segundo que cruzam a ponte dupla em cada ramificação (supostamente constante).

v: Velocidade de deslocamento (em cm/s) de uma formiga; também considerada constante.

l_l, l_c: São o comprimento da ramificação mais longa e mais curta, respectivamente.

r: Razão entre o comprimento da ramificação mais longa e o da mais curta; $r = l_l/l_c$.

t_c: Tempo gasto por uma formiga para percorrer a ramificação mais curta; $t_c = l_c/v$.

t_l: Tempo gasto por uma formiga para percorrer a ramificação mais longa; $t_l = l_l/v = r \cdot t_c$.

Vamos enumerar os pontos inicial e final das ramificações, denotando por 1 o formigueiro e por 2 a fonte de comida. Uma formiga, chegando no ponto $i \in \{1,2\}$, tem a opção de escolher um dos caminhos l ou c, que chamaremos de mais longo e mais curto, respectivamente. Em um dado instante, a quantidade φ_{ia} de feromônio na ramificação a é proporcional ao número de formigas que por ali passaram até aquele momento. Por sua vez, a probabilidade p_{ia} que uma formiga, chegando no ponto i, tome a decisão de percorrer a ramificação a será funcção dessa quantidade de feromônio.

A partir de seus experimentos (descritos em [23]), Deneubourg e equipe propuseram a seguinte expressão para a probabilidade acima descrita:

$$p_{ia}(t) = \frac{(t_a + \varphi_{ia}(t))^\alpha}{(t_c + \varphi_{ic}(t))^\alpha + (t_c + \varphi_{il}(t))^\alpha}, \quad a \in \{c,l\}$$

Ainda com base nos resultados obtidos nesses experimentos, ficou estabelecido que o problema da ponte dupla pode ser modelado adequadamente com $\alpha = 2$. Este sistema estocástico pode ser descrito pelo sistema de equações diferenciais abaixo:

$$\frac{d\varphi_{ia}}{dt} = \mu p_{ja}(t - t_a) + \mu p_{ia}(t), \qquad (i,j) \in \{(1,2),(2,1)\}, \quad a \in \{c,l\}$$

Uma breve discussão sobre esse sistema, bem como a análise de algumas simulações são descritas em [24].

4.3 MODELANDO A COLÔNIA: SISTEMA DISCRETO

Na seção anterior estudamos o modelo proposto por Deneubourg para descrever uma colônia real de formigas. Em particular, estudamos um sistema de equações diferenciais que descreve o comportamento da colônia no que diz respeito à procura e coleta de alimentos, embora em experimento particular que é o da ponte dupla.

Estudaremos agora uma forma de adaptar as idéias de Deneubourg na construção de colônias artificiais de formigas, simulando o experimento da ponte dupla. Para isso precisamos formular uma série de condições, as quais são descritas em [24]:

Modelo discreto do experimento da ponte dupla:

1. Construção de um grafo com dois nós (ou vértices) 1 e 2, representando o formigueiro e a fonte de comida, respectivamente. Esses nós são conectados por dois arcos, onde um deles é r vezes maior que o outro, sendo $r \geq 1$ um número inteiro. A figura a seguir ilustra essa condição:

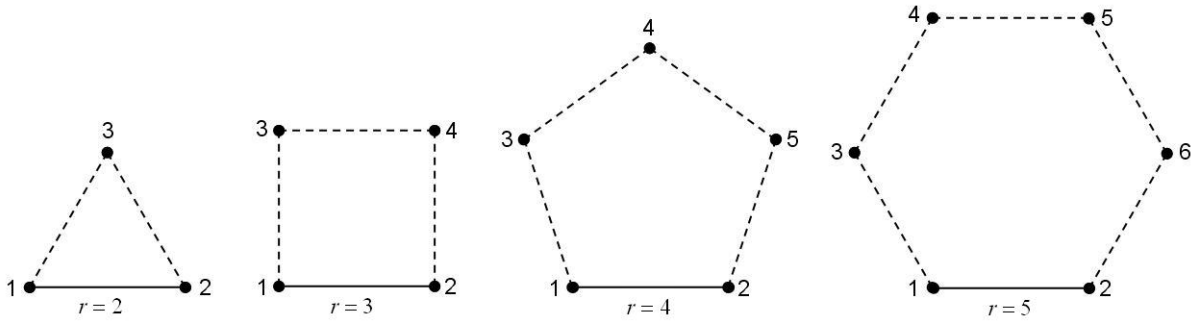

Figura 4.3 Grafos da ponte dupla artificial com $r = 2,3,4,5$, respectivamente

2. Devemos assumir que o tempo é discreto e que a cada e que a cada passo de tempo $(t = 1,2,3\ldots)$ cada formiga move-se de um nó do grafo para um nó vizinho com velocidade constante $v = 1$unidade de comprimento/unidade de tempo.

3. Em cada passagem por um arco, uma formiga deposita nele 1 unidade de feromônio.

4. O movimento das formigas é determinado probabilisticamente, onde p_{ia} é a probabilidade de que uma formiga que se encontra no nó $i \in \{1,2\}$ escolha a ramificação $a \in \{c,l\}$.

5. A probabilidade p_{ia} é função da quantidade de feromônio φ_{ia} que uma formiga localizada no nó i encontra na ramificação a. Fazendo uma analogia com o modelo real, para $i \in \{1,2\}$, são definidas as probabilidades de escolha da ramificação mais curta (c) ou mais longa (l):

$$p_{ic}(t) = \frac{(\varphi_{ic}(t))^{\alpha}}{(\varphi_{ic}(t))^{\alpha} + (\varphi_{il}(t))^{\alpha}}, \qquad p_{il}(t) = \frac{(\varphi_{il}(t))^{\alpha}}{(\varphi_{ic}(t))^{\alpha} + (\varphi_{il}(t))^{\alpha}}$$

6. Assumimos que inicialmente todas as formigas estão no formigueiro e assim a quantidade de formigas nos nós 1 e 2, nesse instante, será $m_1(0) = m$ e $m_2(0) = 0$, respectivamente. Em um dado instante $t > 0$, a quantidade de formigas presentes no nó i é dada por

$$m_i(t) = p_{jc}(t-1)m_j(t-1) + p_{jl}(t-r)m_j(t-r), \text{ com } (i,j) \in \{(1,2),(2,1)\}$$

Os termos que compõem a expressão para $m_i(t)$ podem ser interpretados da seguinte forma:

(i) A parcela $p_{jc}(t-1)m_j(t-1)$ mede a quantidade de formigas que no instante anterior $(t-1)$ estavam no extremo oposto e escolheram viajar pelo caminho mais curto; dessa forma, gastaram 1 unidade de tempo para chegar ao nó onde estão.

(ii) Por outro lado, a parcela $p_{jl}(t-r)m_j(t-r)$ indica a quantidade de formigas que, estando no extremo oposto $t-r$ unidades de tempo antes, decidiram viajar pelo camingo mais longo. Como este caminho é r vezes maior que o mais curto, elas precisam de r unidades de tempo para percorrê-lo.

7. A atualização de feromônio será feita da seguintes forma:

$$\begin{cases} \varphi_{ic}(t) = \varphi_{ic}(t-1) + p_{ic}(t-1)m_i(t-1) + p_{jc}(t-1)m_j(t-1), & i \neq j \\ \varphi_{il}(t) = \varphi_{il}(t-1) + p_{il}(t-1)m_i(t-1) + p_{jl}(t-r)m_j(t-r), & i \neq j \end{cases}$$

Como no item anterior, descreveremos o significado de cada uma das parcelas que compõem as expressões acima. Para a ramificação mais curta, $\varphi_{ic}(t)$, temos

(i) $\varphi_{ic}(t-1)$:
quantidade de feromônio existente nessa ramificação no instante anterior ;

(ii) $p_{ic}(t-1)m_i(t-1)$:
quantidade de formigas que estavam no nó i no instante anterior e decidiram viajar pelo caminho mais curto, cada uma contribuindo com 1 unidade para o feromônio total dessa raminficação;

(iii) $p_{jc}(t-1)m_j(t-1)$:
quantidade de formigas que estavam no nó j no instante anterior e decidiram viajar pelo caminho mais curto, cada uma também contribuindo com 1 unidade para o feromônio total dessa raminficação.

Analogamente, para $\varphi_{il}(t)$ temos

(i) $\varphi_{il}(t-1)$:
quantidade de feromônio existente nessa ramificação no instante anterior ;

(ii) $p_{il}(t-1)m_i(t-1)$:
quantidade de formigas que estavam no nó i no instante anterior e decidiram viajar pelo caminho mais longo, cada uma contribuindo com 1 unidade para o feromônio total dessa raminficação;

(iii) $p_{jl}(t-r)m_j(t-r)$:
quantidade de formigas que acabaram de chegar ao nó i, vindo do outro extremo pelo caminho mais longo; essas contribuem com uma unidade para o feromônio total dessa ramificacção apenas r unidades de tempo após sua partida.

O experimento da ponte dupla, tanto em seu modelo contínuo abordado na seção anterior, quanto o modelo discreto estudado nesta seção, foram simulados por M. Dorigo e os resultados obtidos, bem como uma análise detalhada desses resultados são descritos em [24]. A partir desses resultados, M. Dorigo conclui que os dois modelos produzem resultados bastante similares.

4.4 IMPERFEIÇÕES DO SISTEMA E O MODELO S-ACO

O modelo ACO descrito anteriormente ajusta-se adequadamente ao experimento da ponte dupla, produzindo soluções com razoável grau de precisão. Nesse experimento, entretanto, o grafo utilizado possui extrutura extremamente simples. Situações que normalmente passariam desapercebidas poderiam levar a resultados imprecisos na modelagem de grafos mais complexos. Duas situações, em particular, poderiam produzir resultados inesperados:

1. **Formigas "andando em círculos"**

   A cada movimento, tanto de ida à fonte, quanto na volta ao formigueiro, uma formiga deposita feromônio nas arestas que percorre. Como seu movimento é aleatório, ela pode criar em seu deslocamento caminhos, ou parte de caminhos, compostos de circuitos fechados (também chamados de ciclos ou loops). Devido ao acúmulo de feromônio, um ciclo poderia tornar-se cada vez mais atrativo de forma que a formiga não mais pudesse sair dele.

2. **Desfavorecimento de boas soluções**

   Devido ao caráter aleatório do processo, em dado momento poderia ocorrer que uma boa solução não tivesse até então recebido uma quantidade suficiente de feromônio que a tornasse atrativa. Consequentemente, outras soluções atrairiam mais formigas enquanto esta seria gradativamente esquecida.

Para evitar a ocorrência de quaisquer das situações mencionandas anteriormente, serão efetuadas algumas alterações no modelo original (ACO). O novo modelo obtido será chamado de ACO simples e denotado por S-ACO (ver [24]).

4.4.1 Descrição do modelo S-ACO

Em sua composição geral, o S-ACO segue os mesmos princípios do ACO. Porém, com o objetivo de eliminar os problemas destacados anteriormente, novas caracteríistivas serão adiconadas ao modelo enquanto algumas das já existentes serão ligeiramente modificadas.

A seguir descreveremos as características que definem o modelo S-ACO em um grafo conexo e finito $G = (N, A)$, onde N é o conjunto de nós do grafo G enquanto que A é o conjunto de suas arestas.

1. **Ida aleatória até a fonte sem atualização de feromônio:**

   Enquanto constrói uma solução, em seu deslocamento do formigueiro até a fonte de alimento, o movimento de cada formigas será probabilístico, guiado apenas pela concentração de feromônio nas arestas que incidem em cada nó por onde ela passa.

   Inicialmente cada aresta $(i, j) \in G$ recebe uma quantidade fixa $\tau_{ij} = \tau_0$ de feromônio. Denotaremos por v_i^k a vizinhança que a formiga k encontra quando está localizada no nó i. Esta vizinhança será composta dos nós diferentes de i que são extremidade de alguma aresta incidente em i, exceto o nó visitado antes de i. A probabilidade de que a formiga k, estando no nó i, escolha mover-se para o nó $j \in v_i^k$ será calculada da seguinte forma:

$$p_{ij}^k = \begin{cases} \frac{\tau_{ij}^\alpha}{\sum_{l \in v_i^k} \tau_{il}^\alpha}, & \text{se } j \in v_{ij}^k \\ 0, & \text{se } j \notin v_{ij}^k \end{cases}$$

   onde α é um parâmetro (constante) a ser determinado e cuja função é ajustar o modelo para assegurar sua convergência.

   Nesta fase as formigas não depositarão feromônio. Dessa forma, podemos evitar o favorecimento de ciclos que possam ter sido criados durante o deslocamento.

2. **Volta determinística ao formigueiro com eliminação de ciclos:**

   A próxima modificação no sistema inicial ACO não possui análogo no mundo real; entretanto, esta mostra-se extremamente útil na implementação de modelos artificiais eficientes de colônias de formigas. A cada formiga será dada uma capacidade limitada de memória onde ela armazanará apenas o caminho percorrido desde o formigueiro até a fonte e, quando necessário, alguma medida associada às arestas percorridas.

   Essa habilidade de memorizar caminhos permite que, ao final do deslocamento até a fonte, sejam eliminados possíveis ciclos criados durante a construção de um caminho.

3. **Atualização de feromônio associada à qualidade do caminho encontrado:**

   Na volta ao formigueiro a formiga depositará feromônio em um caminho livre de ciclos. Por outro lado, memorizando as medidas associadas às arestas percorridas, poderá ser feita ao final da construção de um caminho uma avaliação de sua qualidade bem como compará-lo com outras soluções já encontradas.

   Em seu camingo de volta ao formigueiro a formiga percorrerá o mesmo caminho que construiu na ida, porém na direção oposta, isto após procedida a eliminação de todos os possíveis ciclos. A formiga k deposita uma quantidade de feromônio $\Delta\tau^k$ em cada arco por onde passa. Dessa forma, um arco (i, j) em seu caminho ele será atualizado de acordo com a expressão

   $$\tau_{ij} \leftarrow \tau_{ij} + \Delta\tau^k$$

   O valor escolhido para $\Delta\tau^k$ depende do problema que está sendo abordado. Sua escolha precisa evidenciar características do problema que são importantes de se levar em consideração ao avaliar a qualidade de um caminho e, ao mesmo tempo, penalizar caminhos que não possuam essa característica.

   Embora, aparentemente, não havendo capacidade similar de memória em formigas reais, Beckers, Deneubourg e Goss publicaram uma artigo (ver [5]) relatando a descoberta de uma espécie de formiga (Lasius niger, encontrada em quase toda a Europa e em partes da América do Norte e Ásia) que depositam maiores quantidades de feromônio quando encontram maiores fontes de alimento. Isto reforça a idéia da necessidade de se avaliar a qualidade de um caminho encontrado.

4. **Evaporação de feromônio:**

   Uma última característica será adicionada ao sistema, finalizando a descrição do modelo S-ACO. Na descrição de como problemas de otimização são resolvidos em colônias reais de formigas, a evaporação de feromônio não influenciou de forma significativa o processo. Em colônias artificiais de formigas essa característica será adicionada com a intenção de evitar uma rápida convergência para soluções que não sejam ótimas. Ao perder parte do feromônio que contém, um caminho torna-se um pouco menos atrativo, permitindo assim que as formigas tenham a chance de explorar novos caminhos na busca por melhores soluções.

   Após cada formiga produzir sua solução, ocorre o fenômeno de evaporação de feromôno, que será aplicado a todas as arestas do grafo. Escolhido um parâmetro $\rho \in (0,1]$, a quantidade total de feromônio na aresta (i,j) será atualizada de acordo com a expressão

$$\tau_{ij} \leftarrow (1-\rho)\tau_{ij}, \qquad \forall (i,j) \in A$$

   Deve-se notar que este processo ocorre na volta da fonte ao formigueiro, uma vez que no modelo S-ACO foi eliminada a atualização de feromônio durante a ida do formigueiro até a fonte. Após completado o processo de evaporação será então efetuada a atualização de feromônio devido à qualidade da solução produzida, conforme discutido no item anterior.

No próximo capítulo descreveremos como a utilização de colônias de formigas artificiais podem ser aplicads na busca de soluções do problema do vetor mais curto. Os resultados serão produzidos através de simulações e analisaremos a qualidade desses resultados. Ao final, vamos comparar os métodos mais importantes que são utilizados na solução do SVP, concentrando nossa análise em dois pontos fundamentais: qualidade das soluções e tempo de execução dos algoritmos.

CAPÍTULO 5

UM MODELO ACO PARA O SVP

Life, as the most ancient of all metaphors insists, is a journey;
and the travel book, in its deceptive simulation of the journey's
fits and starts, rehearses life's own fragmentation.
More even than the novel, it embraces the contingency of things.
– JONATHAN RABAN

—

5.1 INTRODUÇÃO

Neste capítulo desenvolveremos um modelo ACO para resolver de modo aproximado o problema do vetor mais curto. Esta classe de algoritmos indica a direção que devemos seguir, porém cada problema a ser modelado possui especificidades próprias e isto nos obriga a adaptar o modelo geral a cada situação encontrada. Algumas características dos modelos ACO necessitam de uma reinterpretação no contexto do SVP para que possam continuar funcionais nesse novo ambiente. Descreveremos a seguir alguns pontos importantes que precisam ser adaptados:

O reticulado é um espaço de busca infinito

O conceito de otimização em colônias de formigas modela problemas e procura as possíveis soluções sobre um grafo finito. Para resolver o problema do vetor mais curto será necessário reduzir o reticulado, que é um espaço de busca infinito, a um grafo finito. Nesse caso, a obtenção de resultados precisos está diretamente associada a uma boa formulação dos critérios utilizados para limitar a região de busca. Além disso, existe também uma forte relação entre o tamanho do grafo e o custo computacional para determinar solução do problema em questão. Sendo assim, é necessário estabelecer um equilibrio entre a qualidade das soluções que queremos e o quanto estamos dispostos a pagar por isso.

Posição do formigueiro e da fonte de alimentos

No modelo geral ACO são conhecidos o ponto de partida, normalmente chamado de formigueiro, e o ponto de chegada, que representa a fonte de alimento. Nesse caso, deseja-se computar alguma informação que se encontra entre esses dois pontos como, por exemplo, o

menor caminho entre eles ou o menor custo para se transportar algo da fonte até o formigueiro. Ao modelar o problema do vetor mais curto, embora existam inúmeras cópias desse vetor espalhadas por todo o reticulado, não sabemos onde elas se encontram. Para encontrar um dado vetor, uma formiga precisa percorrer um caminho entre suas extremidades que, nesse caso, representarão o formigueiro e a fonte. No nosso problema não conhecemos a localização do formigueiro nem da fonte de alimentos, isto é, não sabemos onde a jornada de uma formiga começa nem onde termina.

Critérios de convergência do modelo

Na abordagem tradicional de sistemas modelados por colônias de formigas o fator determinante para a escolha de um caminho como sendo solução de um dado problema é o fato de que um grande percentual das formigas da colônia concentra-se sobre aquele caminho. Uma das características marcantes de um reticulado é o fato de que este é composto de infinitas cópias da região fundamental em todas as direções definidas pelos vetores de uma base. Sendo assim, mesmo que o espaço de busca seja limitado a uma dessas cópias da região fundamental, por exemplo, um grande número de simetrias ainda deve persistir. Cada caminho que leva a um vetor mais curto poderia, em princípio, ser igualmente atraente para uma formiga. Portanto, poderemos encontrar boa parte das formigas de uma colônia distribuídas, preferencialmente, entre alguns ou todos esses caminhos. Nesse caso, teremos que lidar não apenas com um caminho levando ao vetor mais curto, mas sim com toda uma classe de tais caminhos.

As próximas seções deste capítulo serão dedicadas à solução desses problemas de adaptação do modelo original ACO, descrição de um novo modelo adequado a reticulados, simulação do modelo proposto e à análise dos resultados obtidos dessas simulações.

5.2 ADAPTAÇÕES NO MODELO ACO

Vimos anteriormente que o problema do vetor mais curto não permite que o modelo original de colônias artificiais de formigas seja diretamente aplicado em sua solução. Ao invés disso, algumas modificações precisam ser efetuadas para que nosso objetivo possa ser atingido. Sendo assim, precisamos propor possíveis soluções para cada um dos problemas destacados na seção anterior.

5.2.1 Reduzindo o espaço de busca para um grafo finito

Esta é uma questão crucial na modelagem de problemas em reticulados através de colônias de formigas. Uma escolha inicial poderia ser o grafo formado pelos vértices e arestas da própria região fundamental. Para reticulados de posto n, o número de vértices e de arestas dessa região são dados, respectivamente, por

$$V_n = 2^n \qquad \text{e} \qquad A_n = n \cdot 2^{n-1}$$

Partindo da região fundamental, adicionando uma cópia de um vetor da base na respectiva direção, aumentamos o grafo em mais um paralelepípedo naquela direção. Dessa forma, o número de vértices passaria a ser

$$2 \cdot V_n - V_{n-1} = 3 \cdot 2^{n-1} = \frac{3}{2} V_n,$$

enquanto que o novo número de arestas seria

$$2 \cdot A_n - A_{n-1} = (3n+1)2^{n-2} = \left(\frac{3}{2} + \frac{1}{2n}\right) A_n.$$

Portanto, aumentando apenas uma unidade cada coordenada, em todas as direções dadas pelos vetores da base, a quantidade de vértices e arestas do novo grafo seriam pelo menos $(1.5)^n$ vezes maiores que as do paralelepípedo fundamental.

Como vimos em capítulos anteriores, podemos encontrar limitantes para o tamanho do vetor não nulo mais curto. O comprimento λ_1, de um vetor mais curto em um reticulado Λ de posto n, satisfaz $\lambda_1 \leq \sqrt{n}.\mathrm{Det}(\Lambda)^{1/n}$. Isto nos permite delimitar uma região em torno da origem onde um tal vetor pode ser encontrado. Entretanto, considerando que um pequeno aumento de seu tamanho em qualquer direção aumenta exponencialmente a quantidade de vértices de de arestas do grafo, optaremos inicialmente por escolher o espaço de busca como sendo o paralelepípedo fundamental do reticulado.

A escolha do paralelepípedo fundamental como região onde vamos encontrar um vetor que aproxima v_0, o vetor mais curto do reticulado, evita os problemas citados acima. Entretanto, efeitos colaterais indesejáveis dessa escolha podem aparecer quando o uma das extremidades de v_0 está longe dos vértices dessa região. Informalmente, podemos ver que isto ocorre, por exemplo, quando um dos vetores da base é bem menor que um outro estes formam entre sí um ângulo pequeno. A figura a seguir ilustra esta situação:

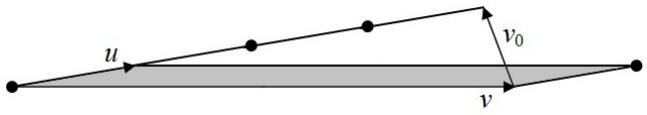

Figura 5.1 Vetor mais curto fora da região fundamental

Considere, por exemplo, os vetores $u = (7,0), v = (3,1) \in \mathbb{R}^2$ como base de um reticulado. Os portos que estão, na região fundamental são $\overrightarrow{0}$ e as extremidades dos vetores u, v, $u+v$. Com esses pontos podemos obter ainda os vetores $v-u$ e $u-v$. Temos

$$||u|| = 7, \quad ||v|| = \sqrt{10}, \quad ||u+v|| = \sqrt{101}, \quad ||u-v|| = ||v-u|| = \sqrt{17}$$

O vetor $u-2v = (1,-2)$ mede $\sqrt{5}$ e assim não pode estar na região fundamental.

Quando o vetor mais curto estiver fora da região fundamental o modelo que estamos propondo, tomando a região fundamental como espaço de busca, não conseguirá encontrá-lo. Além disso, como um reticulado é um grupo discreto, haverá um limite para o grau de aproximação que poderemos obter, não sendo possível ultrapassar esse limite.

5.2.2 Definição dos análogos de formigueiro e fonte de alimento

No modelo original ACO os conceitos de "formigueiro" e "fonte" são fundamentais para sua aplicacão. O formigueiro é o ponto a partr do qual todas as formigas da colônia iniciam a construção de qualquer caminho. Por outro lado, a fonte de alimento (ou, simplesmente, fonte) é concebida como o ponto final, onde qualquer formiga encerra todo caminho que constrói.

Uma vez definidos os pontos do grafo que representarão o formigueiro e a fonte, todo o processo ocorre em caminhos que os ligam. A partir das informações extraídas desses caminhos, pode-se inferir sobre possíveis soluções do problema que está sendo modelado.

Esta característica não pode ser estendida ao problema do vetor mais curto, pois não possuímos informações sobre a localização exata de suas extremidades. De fato, são estas informações que estamos buscando. No modelo que estamos propondo, todas as formigas partirão da origem do reticulado, mas este ponto não será considerado como inicio de qualqur caminho. Quando cada formiga concluir a construção de seu caminho, consideraremos estes aos pares e uniremos cada par através do ponto comum a todos os caminhos, produzindo assim um novo caminho.

Ao final, quando cada formiga conclui sua construção, avaliamos a qualidade de cada caminho produzido. Quanto mais próximos os extremos de um caminho, melhor será sua qualidade. Nesse caso, os candidatos a vetor mais curto serão os vetores que ligam esses extremos, pois

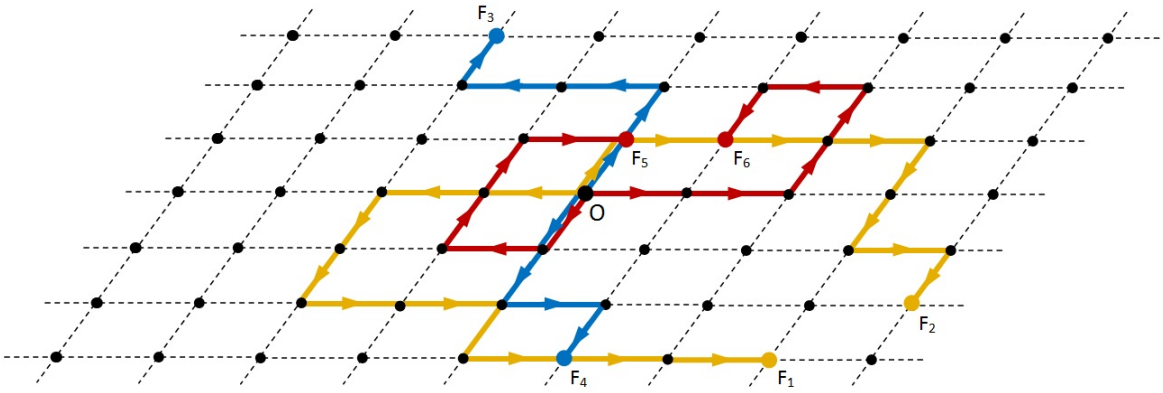

Figura 5.2 Caminhos construidos por pares de formigas: $(F_1,F_2),(F_3,F_4),(F_5,F_6)$

são as extremidades desses novos caminhos que fornecem as informações que procuramos sobre vetores curtos. Além disso, essas extremidades fazem o papel da fonte de alimento, pois será a partir delas que as formigas voltam até o formigueiro (representado pela origem do reticulado) depositando feromônio pelo caminho.

No nosso modelo, não serão considerados apenas os caminhos produzidos por pares fixos de formigas. Como a medida que importa é a distância entre as extremidades, consideraremos todas as distâncias entre todas as extremidades produzidas, tomadas duas a duas. Isto assemelha-se, de certo modo, a tomar um subconjunto dos vértices do grafo e construir vetores entre pares desses vértices e tomar o menor deles como uma aproximação para o vetor mais curto. Entretanto, a escolha desses vértices não é completamente aleatória pois serão os melhores que a operação "constrói caminho-deposita feromônio" pode produzir.

5.2.3 Qualidade de um caminho e atribuição de feromônio

Um outro componente característico dos modelos ACO tem influência direta no sucesso ou fracasso na modelagem de um dado problema. Quando uma formiga retorna ao formigueiro depositando feromônio pelo caminho, sua intenção é atrair mais formigas para aquele caminho. No modelo artificial entretanto, o depósito de feromônio deve ser feito de modo a beneficiar caractetísticas relevantes do problema, ou seja, vetores curtos. Não podemos beneficiar caminhos que não produzem boas soluções, isto é, caminhos que levam a vetores "grandes". Uma maneira de implementar isto é depositar em caminhos que produzem vetores pequenos quantidades maiores de feromônios, enquanto que caminhos que geram vetores maiores receberão menos feromônio. Portanto, deve existir uma relação inversa entre quantidade de feromônio depositada e tamanho do vetor produzido.

No nosso modelo a quantidade de feroônio depositada em um caminho é calculada da se-

guinte forma: dado um caminho $C = v_1 v_2 \cdots v_n$ livre de ciclos (ou loops), onde os v_i's são pontos do reticulado, a quantidade de feromônio que C receberá é dada por

$$\text{feromonio}(C) = \frac{1}{||v_n - v_1||}$$

Note que nesse modelo o vetor $\overrightarrow{v_1 v_n}$, que ume as extremidades de um caminho, é um candidato a vetor mais curto. Quanto menor seu tamanho, melhor será a avaliação do caminho que o produziu que por sua vez receberá maior quantidade de feromônio. A expressão acima traduz essa idéia em números.

5.2.4 Convergência do modelo

Conforme dissemos na seção anterior, o que faz com que um caminho seja visto como possuidor de alguma informação sobre uma possível solução para um problema é o fato de que um grande percentual das formigas da colônia estar concentrado nesse caminho. No problema do vetor mais curto, entretanto, um grande número de simetrias existentes no reticulado faz com que as formigas fiquem concentradas em diversos caminhos ao invés de em um único. Uma característica particularmente interessante da nossa versão ACO para o SVP é que estamos interessados exatamente nos caminho, mas sim em seus extremos. Em termos de convergência agora não importa qual o caminho mais frequentado, mas sim que vértices aparcem mais vezes como extremidades de caminhos. Havendo necessidade de definir preferĉia entre os vértices do grafo, este será o primeiro critério de escolha.

5.3 UM ALGORITMO ACO PARA O SVP

Nesta seção descreveremos em detalhes os parâmetros utilizados no modelo ACO que aproxima o vetor mais curto em um reticulado, bem como o algoritmo para realizar essa tarefa. Denotaremos esse algoritmo por SVP-ACO.

5.3.1 Parâmetros do modelo para o problema do vetor mais curto

Dimensão (dim)**:**
Esta é a dimensão do espaço onde está imerso o reticulado.

Posto (n)**:**
O posto de um reticulado é a quantidade de vetores que possui qualquer de duas bases. É, ao lado do volume da região fundamental, a medida mais importante associada a um reticulado. O posto geralmente é o componente principal no cálculo da complexidade de algoritmos que

tratam de questões sobre reticulados.

Quantidade de formigas (q)**:**
É a quantidade de formigas na colônia artificial. Sua importância para o modelo reside no simples fato que muitas formigas produzem muitos caminhos. Muitos desses caminhos devem coincidir produzindo boas soluções para o problema. Entretanto, um grande aumento na quantidade de formigas tornará o processo lento.

Passos por formiga (pf)**:**
Na construção de um caminho o número de passos dados por uma formiga é definido por esse parâmetro. No entanto, este não é necessariamente o "tamanho" final de um caminho, pois cada caminho terá todos os ciclos (loops) eliminados ao final de sua construção.

Passeios da colônia (pc)**:**
A escolha de um caminho como sendo uma "boa" solução para o problema está associada à quantidade de formigas presentes nesse caminho. Entretanto, a quantidade de feroônio presente em um caminho é a única forma de atrair uma formiga. Um "passeio" é um ciclo completo de toda colônia, onde cada formiga parte do formigueiro construindo seu caminho e, em seguida, retorna depositando feromônio no caminho (agora livre de loops).

Constante de ajuste do modelo (α)**:**
No capítulo anterior descrevemos o modelo ACO que modelava o experimento da ponte dupla. Na ocasião, vimos que um dos parâmetros do modelo era uma constante α que exercia influência no cálculo das probabilidades de escolha de uma determinada direção. Sendo $\varphi(P,v,t)$, onde v é um dos vetors da base B do reticulado, a quantidade de feromônio presente no instante t na aresta que parte de P na direção v, a probabilidade de que uma formiga estando em P escolha a direção v é dada por

$$p(P,v,t) = \frac{[\varphi(P,v,t)]^{\alpha}}{\sum_{u \in B} [\varphi(P,u,t)]^{\alpha}}$$

Nos estudos realizados por Deneubourg, ficou estabelecido que para o problema da ponte dupla poderia ser utilizado $\alpha = 2$. Entretanto, esse valor pode não ser adequado ao problema do vetor mais curto.

Percentual de evaporação de feromônio (e)**:**
Na discussão sobre os modelos ACO vimos que uma rápida convergência do modelo poderia levar a soluções sub-ótimas. Isto ocorre porque as formigas da colônia seriam todas atraídas muito cedo para alguns caminhos enquanto que muitos outros não seriam mais visitados. O conceito de evaporação faz com que a concentração de feromônio aumente mais lentamente, permitindo assim que as formigas possam explorar novos caminhos.

Feromônio inicial (f)**:**
É a quantidade inicial de feromônio que estará presente em todas as arestas do grafo no inicio do processo.

Repetições (rep)**:**
Como os modelos ACO são processos estocásticos, diferentes resultados podem ser obtidos em diferentes simulações do modelo, mesmo que todos os parâmetros permaneçam com seus valores inalterados. Sendo assim, o modelo será avaliado por seu comportamento médio em diversas repetições de ensaios para conjuntos de valores fixos dos parâmetros.

O restante deste capítulo será dedicado à formulação de um algoritmo que resolve de forma aproximada o problema do vetor mais curto. Além disso, faremos simulações do modelo proposto com o objetivo de compreender a influência de cada parâmetro no resultado final obtido.

5.3.2 Algoritmo SVP-ACO

Agora que foram definidos os parâmetros necessários à descrição do modelo, podemos propor um algoritmo que resolve, de forma aproximada, o problema do vetor mais curto. Isto será nosso objeto de estudo no próximo capítulo.

CAPÍTULO 6

SIMULAÇÕES DO MODELO ACO-SVP

Life, as the most ancient of all metaphors insists, is a journey;
and the travel book, in its deceptive simulation of the journey's
fits and starts, rehearses life's own fragmentation.
More even than the novel, it embraces the contingency of things.
– JONATHAN RABAN
(Escritor e romancista inglês)

—

6.1 INTRODUÇÃO

Nos capítulos anteriores estudamos em detalhes o problema do vetor mais curto. Esse estudo culminou com a proposta de um modelo, inspirado nos algoritmos de otimização em colônias de formigas. Sua formulaçãoo foi feita de modo que representasse da melhor forma possível as características mais relevantes do problema de encontrar um vetor mais curto em um reticulado. Neste capítulo vamos observar o comportamento desse modelo através de simulações e verificar sua eficácia na solução desse problema.

Cada uma das simulações aqui descrita terá o objetivo específico de compreender a influência de um determinado parâmetro do modelo no resultado final obtido, isto é, no tamanho do vetor mais curto produzido pelo algoritmo SVP-ACO. Conforme estudado anteriormente, esse problema é classificado como NP-hard e assim, com exceção de alguns casos especiais, não temos acesso direto ao vetor mais curto em um dado reticulado. Nossa abordagem então será, necessariamente, indireta e nossos resultados serão comparados aos que são produzidos pelo algoritmo LLL.

O algoritmo LLL, conforme estudado em capítulos anteriores, produz vetores com alguma margem de erro. Embora essa margem seja medida por uma expressÃ£o exponencial no posto do reticulado, o LLL produz, na prática, resultados significativamente melhores do que sua margem de erro teórica prediz.

6.2 SIMULANDO O MODELO SVP-ACO

Cada uma das simulações que descreveremos a seguir, tem o obejtivo de estudar um dos parâmetros do modelo descrito no capítulo anterior. Consideramos em nossas simulações apenas reticulados conhecidos como "full rank", isto é, reticulados cujo posto é igual à dimensão do espaço $\mathbb{R}^m$ onde ele está imerso.

Para cada simulação foi considerado o posto n em estudo e foram geradas aleatoriamente 10 bases de vetores em $\mathbb{R}^n$, produzindo assim 10 reticulados distintos. Devido ao caráter aleatório do modelo, para cada um desses 10 reticulados foram executadas 10 repetições de todo o processo. Dessa forma, para cada um dos valores do parâmetro que está sendo estudado (e fixados todos os outros parâmetros), temos um total de 100 candidatos a vetor mais curto, produzidos por 100 simulações. Entretanto, muitos desses vetores são coincidentes.

A comparação entre os resultados obtidos pelo SVP-ACO e LLL será feita da seguinte forma:

Sejam υ_{ACO} e υ_{LLL} os vetores produzidos pelo SVP-ACO e pelo LLL, respectivamente. Estudaremos o valor da razão $r = ||\upsilon_{ACO}||/||\upsilon_{vLLL}||$. Dessa forma, teremos $r > 1$, $r < 1$ ou $r = 1$, conforme ocorra $||\upsilon_{ACO}|| > ||\upsilon_{LLL}||$, $||\upsilon_{ACO}|| < ||\upsilon_{LLL}||$ ou $||\upsilon_{ACO}|| = ||\upsilon_{LLL}||$.

Como visto anteriormente, pode ocorrer que o vetor mais curto de um reticulado esteja "longe" da região fundamental. Caso isto ocorra e o LLL consiga "capturá-lo" haverá uma diferença significativa entre os tamanhos dos vetores produzidos pelo LLL e pelo SVP-ACO. Nesse caso, os dois algoritmos em questão estariam procurando vetores mais curtos em regiões possivelmente bastante distintas, o que poderia criar aparentes distorções na comparação dos resultados obtidos. Como forma de evitar essas distorções, vamos gerar apenas instâncias de reticulados onde o vetor mais curto encontrado pelo LLL está na região fundamental e, dessa forma, tanto o LLL quanto o SVP-ACO estarão rodando sobre o mesmo espaço de busca.

6.2.1 Posto do reticulado $\times$ vetor mais curto

Estudaremos inicialmente como o posto do reticulado pode afetar o vetor mais curto produzido pelo nosso modelo. Esse parâmetro assumirá os valores 4, 6, 8 e 10. Os outros parâmetros do modelo serão fixados conforme descrito na tabela a seguir e esses valores serão utilizados sempre que um parâmetro precisar ter seu valor fixado. IncluÃmos ainda o desvio padrão de cada amostra para termos uma idéia da dispersão da amostra de r obtida com a simulação. Os dados obtidos podem ser visualizados na tabela abaixo:

Posto do reticulado × vetor mais curto			
Parâmetros fixos	Posto	*r* médio	Desvio padrão
Quantidade de formigas: 10	4	1.000	0.000
Passos por formiga: 10	6	1.134	0.295
Passeios da colônia: 10	8	1.149	0.165
Constante de ajuste: 2	10	1.219	0.175
Taxa de evaporação: 0.01			

Tabela 6.1 Relação entre posto e vetor mais curto

A figura abaixo mostra graficamente valor médio de r em 10 repetições da simulação para cada base gerada.

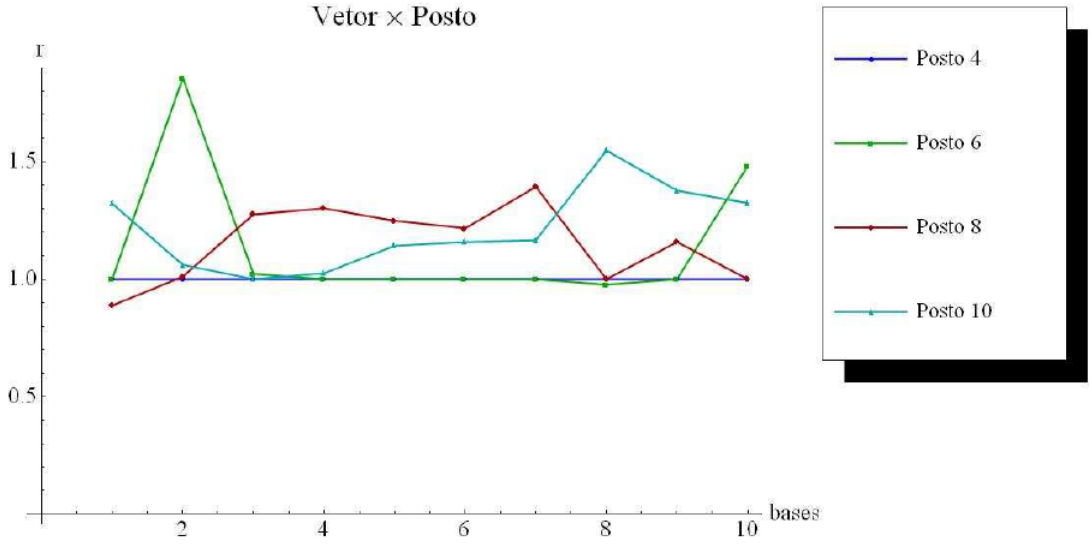

Figura 6.1 Influência da do posto no cálculo do vetor mais curto

Análise:
O parâmetro posto tem influência decisiva na aplicaçãoo do modelo. Fixando todos os outros parâmetros, um aumento no posto do reticulado faz o modelo produzir candidatos a vetores curtos ligeiramente maiores que os produzidos pelo LLL. Os resultados obtidos pelo SVP-ACO encontram-se em uma faixa entre 1 e 1.6 vezes o valor obtido pelo LLL.

Uma possível melhoria nesses resultados pode ser obtida através de um aumento na quantidade de formigas da colônia. Estudaremos essa propriedade a seguir.

6.2.2 Quantidade de formigas × vetor mais curto

Nesta seção estudaremos a contribuição do parâmetro "quantidade de formigas" no cálculo do vetor mais curto. Nas simulações consideramos os postos 4, 6, 8, 10, enquanto que a quantidade de formigas tomou os valores 5, 10, 15, 20 e 25. Nas próximas tabelas, cada célula mostrará duas informações: acima está a *média* e abaixo o *desvio padrão* da amostra *r* relativa ao conjunto de dados em estudo.

Quantidade de formigas × vetor mais curto						
Parâmetros fixos	Posto	Quantidade de formigas				
		5	10	15	20	25
Passos por formiga: 10	4	1.010	1.000	0.923	0.994	0.982
		0.031	0.000	0.196	0.020	0.050
Passeios da colônia: 10	6	1.140	1.133	1.037	1.005	1.059
		0.255	0.295	0.290	0.060	0.049
Constante de ajuste: 2	8	1.277	1.149	1.084	1.000	1.019
		0.328	0.164	0.162	0.117	0.097
Taxa de evaporação: 0.01	10	1.604	1.212	1.051	1.079	1.016
		0.300	0.175	0.094	0.092	0.122

Tabela 6.2 Relação entre quantidade de formigas e vetor mais curto

Os dados contidos nessa tabela podem ser visualizados no gráfico a seguir:

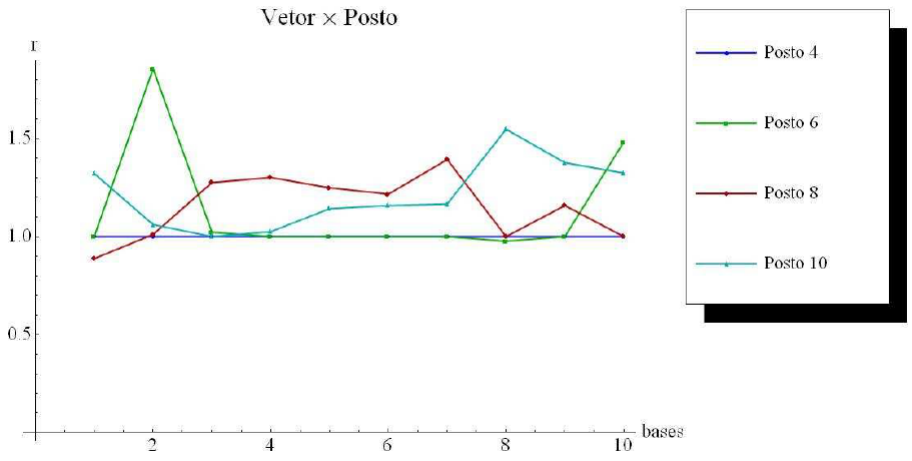

Figura 6.2 Influência da quantidade de formigas no cálculo do vetor mais curto

Análise:
Conforme esperado, à medida que cresce o número de formigas da colônia, o valor de r diminui. Na região fundamental não ocorreu de o SVP-ACO produzir resultados significativamente melhores que o LLL, porém conseguu produzir boas soluções (r pequeno) mesmo com um número bem pequeno de formigas (25 no máximo). Entretanto, isso muda bastante quando nosso espaçoo de busca não fica restrito à região fundamental.

Em um novo experimento, escolhemos apenas um valor para o posto ($n = 6$) e permitimos que o LLL procurasse soluções em todo o reticulado, enquanto o SVP-ACO continuará restrito à região fundamental. Obtivemos os resultados dados pela tabela a seguir, onde mostramos o valor médio de r para cada uma das bases do reticulado geradas. Cada entrada da tabela foi obtida de uma base completamente diferente de todas as outras; portanto, elementos de uma mesma coluna não estão associados a uma mesma base.

Quantidade de formigas × vetor mais curto (experimento 2)											
Formigas	Base 1	Base 2	Base 3	Base 4	Base 5	Base 6	Base 7	Base 8	Base 9	Base 10	r médio
5	1.126	1.001	1.672	0.791	1.000	1.087	1.000	1.000	1.394	1.325	1.140
10	1.000	1.855	1.022	1.000	1.000	1.000	1.000	0.977	1.000	1.481	1.133
15	1.000	1.000	1.000	0.664	1.000	1.809	1.000	1.000	0.953	0.949	1.037
20	0.888	1.000	1.022	1.000	1.000	1.000	1.000	1.000	1.000	1.142	1.005
25	1.000	1.000	1.031	1.000	1.471	1.000	0.980	1.000	1.105	1.000	1.059

Tabela 6.3 Relaçõ entre quantidade de formigas e vetor mais curto - experimento 2

Note que, mesmo restringindo o SVP-ACO à região fundamental e não impondo restrição alguma ao LLL, ainda é possível obter melhores resultados que o LLL, isto é $r < 1$.

O próximo parâmetro a ser investigado é o número de passos que uma formiga dá na construção de um caminho.

6.2.3 Número de passos × vetor mais curto

A construção de cada caminho no modelo SVP-ACO está condicionada ao número de passos que uma formiga tem permissão para caminhar durante a construção. Para um dado reticulado, avaliar a quantidade de passos que uma formiga pode dar constitui uma questão difícil de resolver. Por um lado, quando o número de passos é "pequeno", pode ser que nenhuma formiga consiga aproximar-se de um vetor mais curto, caso estes estejam localizados distante da origem. Entretanto, quando o número de passos é "grande" as formigas facilmente criarão caminhos contendo ciclos e isso faz com que o caminho produzido seja, depois de eliminados os ciclos, igual a um caminho construído com menos passos. Isto ocorre especialmente quando uma formiga está localizada em uma região com alta concentração de feromônio, pois o modelo a induz a continuar próximo daquela região.

A próxima tabela foi obtida simulando o ACO-SVP para reticulados com posto 4, 6, 8, 10, 12 e onde o número de passos assume os valores 3, 6, 9, 12 e 15 enquanto que todos os outros parâmetros permanecem constantes. Convém lembrar que cada dois caminhos aqui construídos darão origem a um caminho final, obtido pela união desses dois através de suas origens comuns. Dessa forma, os caminhos a serem analisados podem ter no final uma quantidade entre 6 e 30 passos.

Número de passos × vetor mais curto						
Parâmetros fixos	Posto	Passos por formiga				
		3	6	9	12	15
Quantidade: 10	4	1.000	0.920	0.988	1.000	1.000
		0.000	0.254	0.037	0.000	0.000
Passeios: 10	6	1.378	1.025	1.022	0.973	0.941
		0.987	0.076	0.174	0.169	0.130
Ajuste: 2	8	1.125	1.124	1.048	1.067	1.156
		0.270	0.205	0.124	0.090	0.316
Evaporação: 0.01	10	1.230	1.200	1.304	1.183	1.312
		0.214	0.198	0.332	0.205	0.329

Tabela 6.4 Relação entre número de passos e vetor mais curto

Graficamente temos:

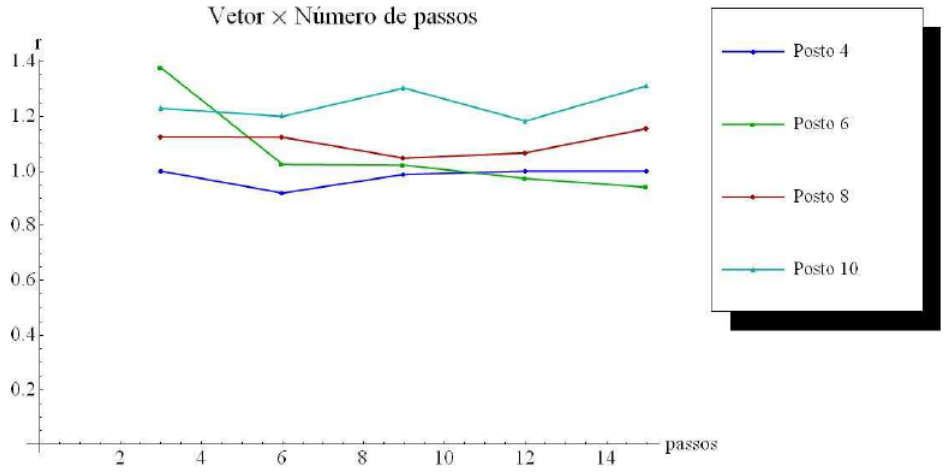

Figura 6.3 Influência do número de passos no cálculo do vetor mais curto

Análise:
Para cada valor do posto do reticulado escolhido para observação a qualidade das soluções produdidas tem margem de erro *r* que oscila sem que possa ser notado algum padrão que comprove que as soluções melhoram ou pioram, à medida que o número de passos cresce. Embora, para todos os valores observados para o posto, sejam encontrados os melhores resultados (menor *r* médio) entre 8 e 12 passos, os dados obtidos não permitem determinar uma relação entre a quantidade de passos dados por uma formiga e a qualidade do caminho que ela produz.

6.2.4 Passeios da colônia × vetor mais curto

Estudaremos agora a relação entre o número de passeios que as formigas da colônia realizam e a qualidade do menor vetor produzido. Um passeio da colônia é um ciclo composto das seguites operações (nessa ordem):

1. Cada formiga constrói um caminho
2. São removidos todos os ciclos de cada caminho
3. Todos os caminhos, agora livres de ciclos, são avaliados (distância entre seus extremos)
4. Cada formiga retorna ao "formigueiro" depositando feromônio no caminho em quantidades calculadas por uma função convenientemente definida.

O conceito de passeio da colônia foi criado com a finalidade principal de adicionar feromônio nas arestas do grafo e, dessa forma, atrair mais formigas para os caminhos que são considerados melhores de acordo com a função de avaliação. A próxima tabela é composta de dados obtidos por simulação do modelo SVP-ACO para reticulados com posto 4, 6, 8, 10, onde foram considerados 4, 6, 9, 12 e 15 passeios de toda a colônia.

Passeios da colônia × vetor mais curto						
Parâmetros fixos	Posto	Passeios da colônia				
		3	6	9	12	15
Quantidade: 10	4	0.933	0.893	1.000	0.968	0.966
		0.121	0.236	0.000	0.101	0.109
Passos: 10	6	1.002	1.033	1.035	0.972	1.066
		0.005	0.093	0.096	0.083	0.157
Ajuste: 2	8	1.051	1.093	1.250	1.147	1.093
		0.115	0.136	0.237	0.153	0.133
Taxa de evaporação: 0.01	10	1.215	1.313	1.121	1.119	1.267
		0.172	0.229	0.120	0.131	0.299

Tabela 6.5 Relação entre número de passeios e vetor mais curto

A figura a seguir representa graficamente essa tabela:

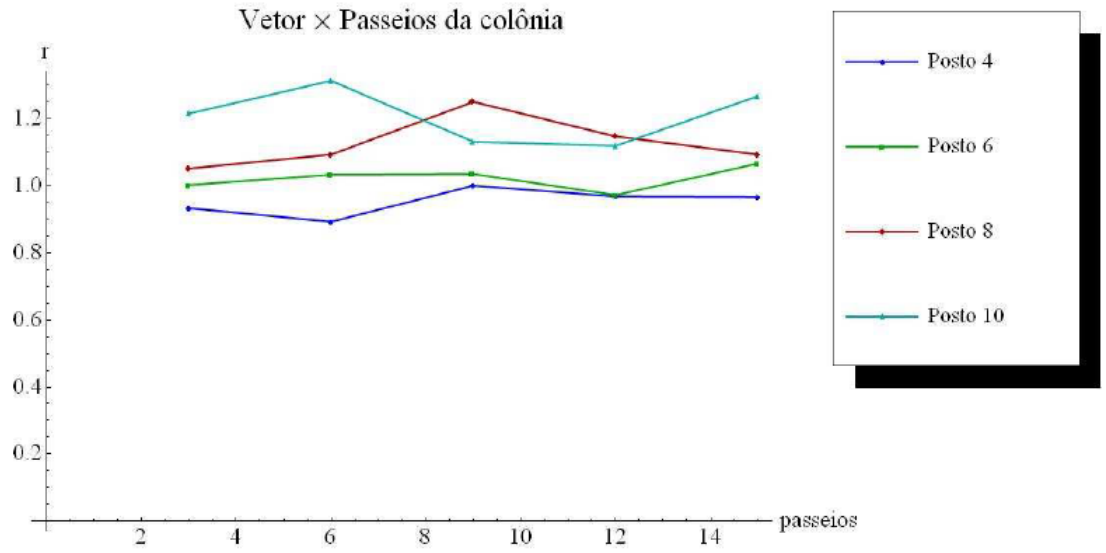

Figura 6.4 Influência do número de passeios no cálculo do vetor mais curto

Análise:
De acordo com os resultados dessa simulação, verificamos que existe uma maior concentração dos menores valores de r no intervalo entre 8 e 12 passeios da colônia. Entretanto, essa faixa de valores pode depender do posto do reticulado. Mesmo que haja uma compensação devido à evaporação, uma maior quantidade de passeios concentrará mais feromônio em algumas regiões do grafo dificultando a busca de possíveis melhores soluções.

6.2.5 Ajuste do modelo × vetor mais curto

Nesta seção estudaremos o modo pelo qual a constante de ajuste do modelo (α), proposta por Deneubourg e sua equipe, pode afetar o vetor mais curto produzido pelo algoritmo SV-PACO. Nesse experimento, simulamos o modelo para os valores $\alpha = 1, 2, ..., 10$ em reticulados de posto 4, 6, 8, 10 e 12.

Constante de ajuste × vetor mais curto											
Parâmetros fixos	Posto	Constante de ajuste									
		1	2	3	4	5	6	7	8	9	10
Quantidade: 10	4	1.000	1.000	0.980	0.965	1.000	0.964	1.000	0.959	1.000	0.979
		0.000	0.000	0.065	0.110	0.000	0.105	0.000	0.130	0.000	0.140
Passos: 10	6	1.000	1.133	1.040	0.969	1.095	1.001	0.952	1.003	0.988	1.009
		0.000	0.295	0.069	0.185	0.273	0.025	0.101	0.009	0.041	0.023
Passeios: 10	8	0.975	1.149	1.073	1.207	1.117	1.081	1.184	1.222	1.320	1.053
		0.102	0.164	0.136	0.248	0.244	0.120	0.273	0.227	0.440	0.115
Evaporação: 0.01	10	1.175	1.212	1.257	1.218	1.200	1.255	1.173	1.389	1.432	1.267
		0.150	0.175	0.278	0.226	0.169	0.292	0.166	0.376	0.500	0.358

Tabela 6.6 Relação entre constante de ajuste e vetor mais curto

Análise:
No modelo proposto por Deneubourg para explicar o experimento da ponte dupla, o parâmetro α (constante de ajuste) era parte importante do modelo e suas pesquisas apontaram o valor $\alpha = 2$ como o mais apropriado para assegurar a convergência. Entretanto, no problema do vetor mais curto, esse parâmetro não mostrou interferência no que diz respeito ao cálculo desse vetor. Sua variação não criou nenhum padrão de comportamento das soluções obtidas, mesmo em reticulados com posto pequeno. Isto pode ser visto nos dados na tabela acima, obtidos por simulações do modelo, bem como na visualização gráfica desses dados através da figura abaixo:

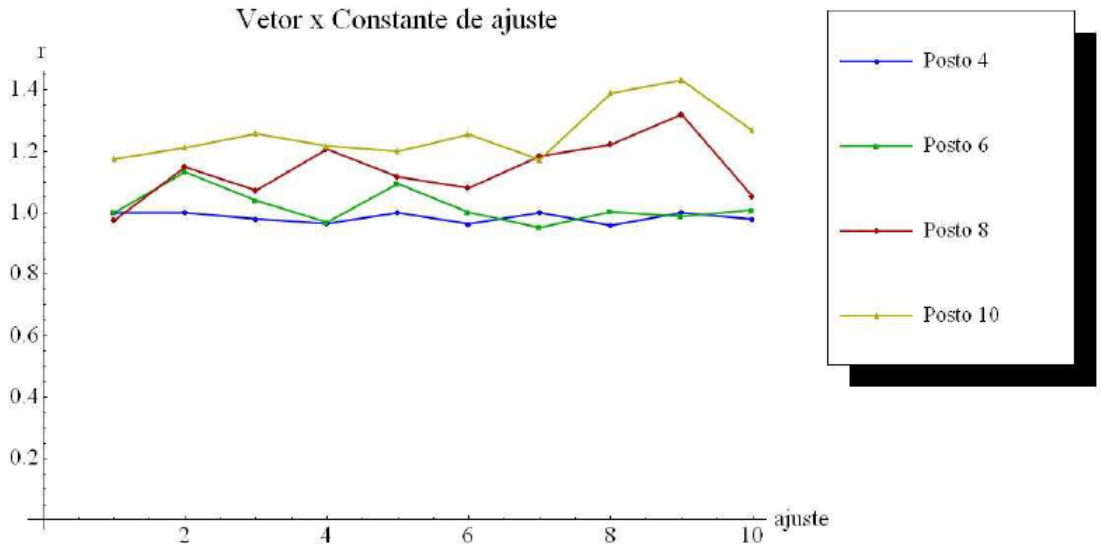

Figura 6.5 Influência da constante de ajuste no cálculo do vetor mais curto

Como não foi observada nenhuma influência da constante de ajuste no valor de r, esse parâmetro não será considerado como parte importante do modelo.

6.2.6 Taxa de evaporação × vetor mais curto

O parâmetro conhecido como taxa de evaporação foi criado com um objetivo bem definido: evitar uma rápida convergência do modelo, permitindo que as formigas artificiais possam continuar explorando o ambiente, mesmo que já tenham encontrado uma solução avaliada como boa. Pequenas taxas de evaporação dão ao modelo essa característica. Por outro lado, taxas maiores podem remover grande parte do feromônio de caminhos com boa avaliação, de modo que se tornem menos atrativo. Nesta seção consideraremos incialmente reticulados de posto 4, 6, 8 e 10; atribuiremos valores entre 0% e 10% a esse parâmetro e estudamos sua interferência sobre os candidatos a vetor mais curto produzidos pelo SVP-ACO. Os resultados da simulação do modelo para essas taxas de evaporação estão na tabela a seguir:

Taxa de evaporação × vetor mais curto (experimento 2)												
Parâmetros fixos	Posto	Taxa de evaporação										
		0	0.01	0.02	0.03	0.04	0.05	0.06	0.07	0.08	0.09	0.1
Quantidade: 10	4	1.000	1.000	0.991	1.000	0.917	1.000	1.051	0.910	1.000	1.032	1.000
		0.000	0.000	0.029	0.000	0.228	0.001	0.328	0.224	0.000	0.185	0.000
Passos: 10	6	1.014	1.133	0.971	1.058	1.141	1.109	1.182	1.092	0.969	1.085	1.099
		0.044	0.295	0.136	0.150	0.264	0.267	0.452	0.210	0.065	0.144	0.262
Passeios: 10	8	1.207	1.149	1.151	1.070	1.158	1.167	1.093	1.038	1.006	1.163	1.176
		0.346	0.164	0.152	0.119	0.403	0.262	0.131	0.156	0.167	0.239	0.294
Ajuste: 2	10	1.181	1.212	1.279	1.259	1.343	1.387	1.225	1.187	1.198	1.283	1.415
		0.106	0.175	0.314	0.223	0.224	0.438	0.230	0.215	0.165	0.295	0.483

Tabela 6.7 Relação entre de evaporação e vetor mais curto

Na figura abaixo podemos ver a representação gráfica dos dados obtidos com esse experimento:

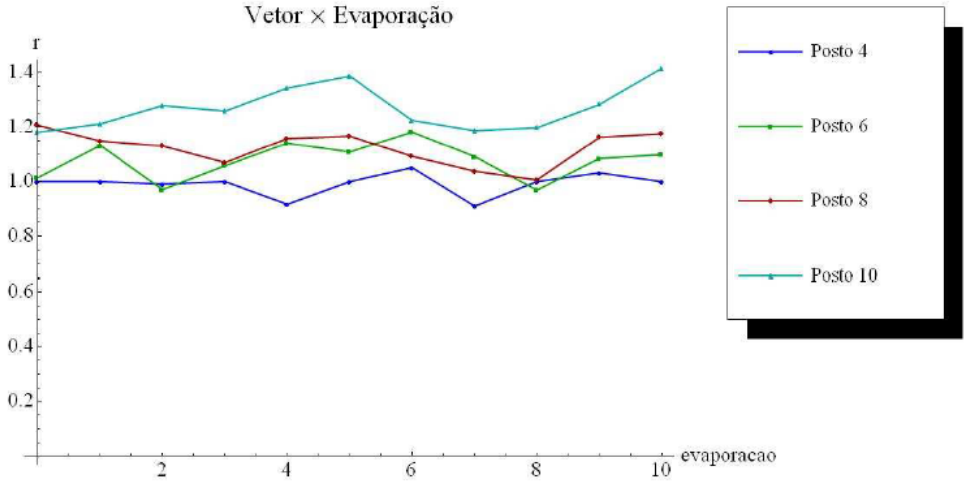

Figura 6.6 Influência da taxa de evaporaçø̃no cálculo do vetor mais curto

Análise:
Diferente do que acontece com a constante de ajuste, a função desse parâmetro no modelo não permite que ele seja ignorado. Os dados obtidos das simulações mostram que os valores de r sofrem uma diminuição se a taxa de evaporação está entre 2% e 4%. Entretanto, quando essa atinge valores entre 7% e 8% podemos observar o mesmo fenômeno e com mais intensidade.

Devido à importância desse parâmetro e às observações anteriores, vamos realizar uma nova simulação com um volume maior de dados para serem observados. Agora observaremos reticulados com posto 4, 5,..., 10 e manteremos a taxa de feromônio entre 0% e 10%. Os resultados obtidos com a simulação podem ser vistos na tabela e no gráfico a seguir:

Taxa de evaporação × vetor mais curto (experimento 2)												
Parâmetros fixos	Posto	Taxa de evaporação										
		0	0.01	0.02	0.03	0.04	0.05	0.06	0.07	0.08	0.09	0.1
Quantidade: 10	4	1.000	1.000	0.991	1.000	0.917	1.000	1.051	0.910	1.000	1.032	1.000
		0.000	0.000	0.029	0.000	0.228	0.001	0.328	0.224	0.000	0.185	0.000
	5	1.000	1.119	0.980	0.950	1.000	0.964	0.932	1.009	0.977	1.030	0.967
		0.000	0.375	0.064	0.094	0.000	0.112	0.181	0.029	0.073	0.177	0.104
Passos: 10	6	1.014	1.133	0.971	1.058	1.141	1.109	1.182	1.092	0.969	1.085	1.099
		0.044	0.295	0.136	0.150	0.264	0.267	0.452	0.210	0.065	0.144	0.262
	7	1.096	1.066	1.101	0.954	1.076	1.055	1.088	0.946	1.088	1.034	1.059
		0.252	0.130	0.169	0.181	0.317	0.162	0.186	0.099	0.163	0.155	0.111
Passeios: 10	8	1.207	1.149	1.151	1.070	1.158	1.167	1.093	1.038	1.006	1.163	1.176
		0.346	0.164	0.152	0.119	0.403	0.262	0.131	0.156	0.167	0.239	0.294
	9	1.201	1.087	1.164	1.299	1.215	1.042	1.271	1.100	1.129	1.113	1.045
		0.254	0.122	0.171	0.319	0.248	0.129	0.155	0.184	0.215	0.161	0.095
Ajuste: 2	10	1.181	1.212	1.279	1.259	1.343	1.387	1.225	1.187	1.198	1.283	1.415
		0.106	0.175	0.314	0.223	0.224	0.438	0.230	0.215	0.165	0.295	0.483

Tabela 6.8 Relação entre taxa de evaporação e vetor mais curto (tabela 2)

Com a intenção de fazer uma análise mais precisa, vamos observar o valor médio dos resultados obtidos para cada valor da taxa de evaporação, ou seja, vamos analisar as colunas da tabela anterior.

Taxa de evaporação × vetor mais curto (valor médio))											
Evaporação	0	0.01	0.02	0.03	0.04	0.05	0.06	0.07	0.08	0.09	0.1
Média	1.100	1.109	1.088	1.084	1.121	1.103	1.120	1.040	1.052	1.106	1.109

Tabela 6.9 Relação entre taxa de evaporação e vetor mais curto (tabela 3)

Esta última tabela confirma o que havia sido observado anteriormente. De acordo com as simulações feitas, os menores valores de r são obtidos quando utilizamos taxas de evaporação entre 7% e 8%.

A partir do conjunto completo dos dados gerados pela simulação obtemos o gráfico a seguir:

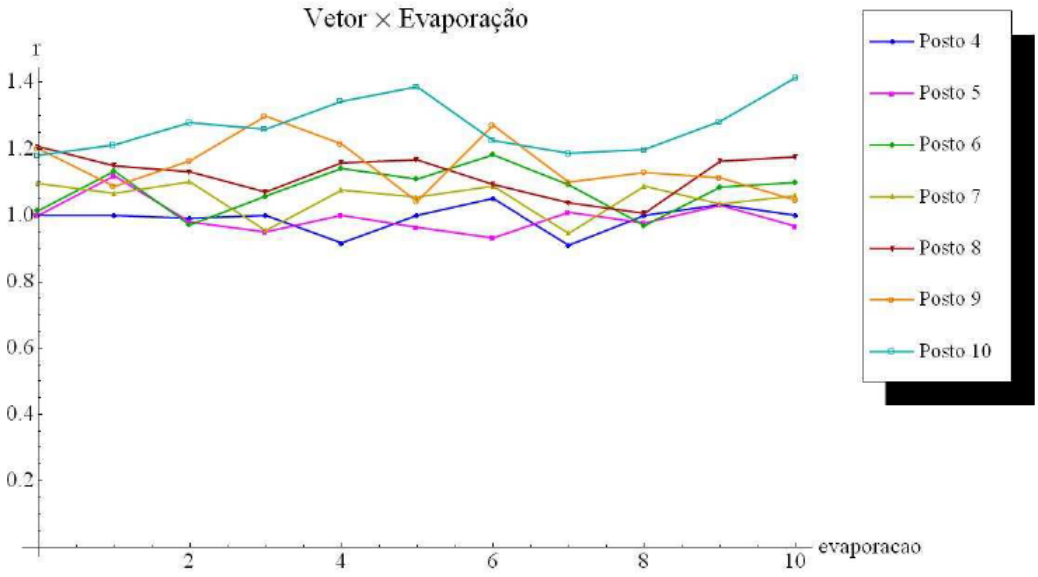

Figura 6.7 Influência da taxa de evaporaçǫ̃no cálculo do vetor mais curto (simulação 2)

6.3 RESUMO DAS INFORMAÇÕES OBTIDAS

Nas simulações realizadas do modelo proposto, algumas características relativas à influência de certos parâmetros ficaram bastante evidenciadas, enquanto que outros praticamente não interferiram nos resultados obtidos. Nesta seção faremos um resumo do que foi observado sobre a contribuição de cada parâmetro. Em seguida, simularemos novamente o modelo observando, para cada parâmetro, seu conjunto de valores que minimizam $r = ||\upsilon_{ACO}||/||\upsilon_{LLL}||$.

Influência observada dos parâmetros do modelo SVP-ACO

- **Posto:**
  À medida que aumenta o posto dos reticulados, ocorre um rápido aumento no valor de r. Além disso, quanto maior o posto, mais dispersos serão os dados obtidos.

- **Quantidade de formigas:**
  Age de forma contrária ao posto. Aumentando gradativamente a quantidade de formigas de uma colônia, observamos uma rápida diminuição no valor de r. Outro dado importante observado foi que em muitos dos casos encontramos variância zero para a amostra de r gerada. Dessa forma, um aumento no número de formigas pode fazer o modelo convergir sempre para o menor valor possível de r.

- **Número de passos:**
  Poucos passos podem fazer com que as formigas não encontrem as extremidades de vetores mais curtos quando estes estão afastados da origem. Muitos passos faz com que as formigas criem caminhos com loops, o que produz caminhos finais com menos passos. De qualquer forma, ainda é preferível utilizar um maior número de passos, pois dará mais opções de construção para as formigas. No modelo simulado não foi possível estabelecer uma quantidade ideal de passos que deva ser utilizada.

- **Passeios:**
  Sucessivas iterações do modelo tem como finalidade destacar caminhos no grafo que produzem melhores soluções para o problema em estudo. Para reticulados com posto até 12, observamos que podemos obter menores erros fazendo entre 8 e 12 passeios.

- **Constante de ajuste:**
  Importante para o experimento da ponte dupla de Deneubourg, esse parâmetro não se mostrou relevante no modelo SVP-ACO.

- **Percentual de evaporação:**
  Para os casos simulados, obtivemos menores valores para r quando a cada passeio evapora entre 7% e 8% do feromônio presente em cada aresta do grafo.

6.4 UMA ÚLTIMA SIMULAÇÃO

Como verificação das informações obtidas, faremos uma nova simulação que utilizará os valores dos parâmetros que indicam a obtenção dos menores valores para r. O algoritmo SVP-ACO será alimentado com os seguintes dados:

- ✓ Dimensão: 8
- ✓ Passeios: 12
- ✓ Posto: 8
- ✓ Ajuste: 2 (irrelevante)
- ✓ Quantidade de formigas: 30
- ✓ Evaporação: 0.07
- ✓ Passos: 10
- ✓ Repetições: 10

Na simulação foram escolhidas aleatoriamente 10 bases $B \subset \mathbb{R}^8$ cada uma com 10 vetores, gerando assim 10 reticulados de posto 10. Para cada base foram realizadas 10 repetições do experimento de onde foram escolhidos os menores vetores gerados. Os resultados obtidos dessa simulação serão mostrados a seguir:

Simulação final	
Base 1	
$(560,887,88,-404,348,542,-855,-125)$	$(-876,345,-616,623,-97,244,-74,-435)$
$(-384,983,686,-688,175,824,911,-121)$	$(-696,-99,335,648,789,-491,296,455)$
$(-465,-664,130,-119,191,-935,61,248)$	$(841,-585,807,-906,229,774,-226,-39)$
$(367,804,-500,-181,-779,526,122,-501)$	$(-567,530,484,-943,355,-37,880,-379)$
LLL: $v = (-168,224,236,241,185,171,93,53)$	$\|\|v\|\| = 516.82$
ACO: $w = (-168,224,236,241,185,171,93,53)$	$\|\|w\|\| = 516.82(r = 1.00)$
Base 2	
$(572,494,-379,-271,-529,252,974,-902)$	$(-784,-551,-364,142,690,-268,-530,866)$
$(950,-684,-130,-310,919,-212,-214,-320)$	$(-771,903,-1000,561,-37,308,280,610)$
$(-143,246,-56,835,-226,246,805,245)$	$(-105,-897,-616,10,-139,-407,-544,-398)$
$(564,-656,959,711,-479,643,219,580)$	$(346,717,702,403,244,693,258,197)$
LLL: $v = (-212,-57,-743,-129,161,-16,444,-36)$	$\|\|v\|\| = 917.33$
ACO: $w = (-212,-57,-743,-129,161,-16,444,-36)$	$\|\|w\|\| = 917.33(r = 1.00)$
Base 3	
$(470,-763,377,-461,-125,288,-406,-904)$	$(-870,787,-923,-365,743,-325,197,21)$
$(103,228,-431,-565,918,869,-931,-601)$	$(419,628,73,-412,936,-83,218,841)$
$(447,-502,-388,-447,-963,792,88,933)$	$(454,-739,783,-922,701,-107,-95,689)$
$(616,753,157,-708,865,679,-637,-893)$	$(-943,-895,-714,477,624,864,306,50)$
LLL: $v = (513,525,588,-143,-53,-190,294,-292)$	$\|\|v\|\| = 1056.22$
ACO: $w = (-513,-525,-588,143,53,190,-294,292)$	$\|\|w\|\| = 1056.22(r = 1.00)$
Base 4	
$(-924,364,-663,611,803,135,-913,96)$	$(546,-986,-312,-147,-817,-455,-99,705)$
$(-702,-772,102,130,-79,-612,-842,-57)$	$(930,925,-525,399,-301,-598,941,403)$
$(813,351,-294,283,-546,-651,191,-520)$	$(-571,452,-204,-379,-495,-838,534,565)$
$(617,-668,-396,38,838,875,37,-737)$	$(-381,-378,-635,-974,227,292,-811,-935)$
LLL: $v = (46,-216,-600,-341,343,37,571,-172)$	$\|\|v\|\| = 999.85$
ACO: $w = (46,-216,-600,-341,343,37,571,-172)$	$\|\|w\|\| = 999.85(r = 1.00)$
Base 5	
$(-701,114,-709,600,595,319,110,371)$	$(889,320,-27,206,871,-83,37,456)$
$(-81,405,-188,-344,-120,-699,-623,-439)$	$(434,-595,678,160,804,214,279,-619)$
$(30,408,-182,300,-988,-994,-263,810)$	$(-574,-125,752,-835,-410,275,842,885)$
$(-158,-123,-766,530,-474,-986,-51,-94)$	$(764,-56,496,-14,-187,-140,-922,88)$
LLL: $v = (464,-187,496,460,-184,-780,16,191)$	$\|\|v\|\| = 1177.66$
ACO: $w = (-2,712,-664,619,83,-143,-132,-69)$	$\|\|w\|\| = 1174.96(r = 0.998)$

Tabela 6.10 Simulação final (Parte 1)

Simulação final	
Base 6	
$(281,-523,311,-241,-872,-452,10,230)$	$(-790,888,697,-956,119,-110,-315,-33)$
$(881,-727,-188,209,-304,-96,606,998)$	$(308,-123,740,-805,478,-872,-876,-950)$
$(-181,959,-189,174,285,-808,-102,306)$	$(-610,637,505,884,752,968,-846,594)$
$(935,-386,-392,-657,282,485,-583,-704)$	$(264,-608,15,985,-206,704,456,462)$
LLL: $v = (418,169,-67,52,-207,25,-825,-704)$	$\|\|v\|\| = 1195.91$
ACO: $w = (-526,280,712,29,-87,594,141,429)$	$\|\|w\|\| = 1194.65(r = 0.999)$
Base 7	
$(101,619,290,448,-803,604,441,689)$	$(588,-402,-336,801,-622,989,-720,-649)$
$(201,497,911,-600,649,-239,-579,592)$	$(-981,-177,267,-450,385,-464,-272,-391)$
$(-678,463,943,733,-378,980,323,-401)$	$(-810,-106,971,728,-286,591,698,-24)$
$(625,-74,791,236,409,-267,787,-830)$	$(-449,533,410,-250,-319,-941,-754,920)$
LLL: $v = (-132,-569,28,-5,92,-389,375,377)$	$\|\|v\|\| = 885.74$
ACO: $w = (132,569,-28,5,-92,389,-375,-377)$	$\|\|w\|\| = 885.74(r = 1.000)$
Base 8	
$(605,-966,986,-972,919,353,-652,-504)$	$(993,-513,-189,-40,-41,-223,457,131)$
$(-875,293,-768,-77,47,559,942,-956)$	$(425,972,490,-540,523,365,140,499)$
$(216,598,874,440,-823,538,623,832)$	$(245,-365,767,391,-805,905,-911,-629)$
$(-511,-948,815,591,-422,555,603,-43)$	$(988,704,-403,451,233,-410,863,15)$
LLL: $v = (993,-513,-189,-40,-41,-223,457,131)$	$\|\|v\|\| = 1250.58$
ACO: $w = (993,-513,-189,-40,-41,-223,457,131)$	$\|\|w\|\| = 1250.58(r = 1.000)$
Base 9	
$(953,-298,-615,-573,-268,720,-736,-703)$	$(322,459,-67,-574,-358,343,-36,468)$
$(456,-298,-70,519,-843,539,-878,986)$	$(586,-202,925,-595,-218,271,-154,159)$
$(-368,-969,201,-916,-468,634,-115,-318)$	$(941,-530,971,-852,246,-394,549,288)$
$(-775,156,483,368,-84,349,65,599)$	$(-775,-568,-801,-218,-610,-820,742,224)$
LLL: $v = (322,459,-67,-574,-358,343,-36,468)$	$\|\|v\|\| = 1055.68$
ACO: $w = (35,-402,-142,472,-354,-457,68,129)$	$\|\|w\|\| = 872.472(r = 0.826)$
Base 10	
$(-613,302,581,276,837,-144,-615,277)$	$(-225,28,835,848,67,-769,-749,924)$
$(311,953,422,45,-374,319,-336,180)$	$(-13,877,-943,-690,-754,-697,342,-949)$
$(-907,400,-860,703,-337,-225,-32,-533)$	$(405,349,217,-598,-40,-801,-306,-42)$
$(-418,-168,-336,566,-920,553,45,834)$	$(123,-295,-361,-876,504,393,-535,-675)$
LLL: $v = (-401,455,-373,-604,-190,377,-127,145)$	$\|\|v\|\| = 1042.67$
ACO: $w = (-124,430,281,-519,460,185,65,-97)$	$\|\|w\|\| = 898.92(r = 0.862)$

Tabela 6.11 Simulação final (Parte 2)

CAPÍTULO 7

CONCLUSÃO E TRABALHOS FUTUROS

Nada há como começar para ver como é árduo concluir.
– VICTOR HUGO
(Novelista, poeta, dramaturgo, ensaísta, estadista e ativista pelos direitos humanos)

—

7.1 CONCLUSÃO

Reticulados, estudados há bastante tempo pelos matemáticos, apenas recentemente atraíram o interesse de outras áreas do conhecimento. Atualmente são componentes importantes no desenvolvimento de áreas como criptografia, codificação de sinais, entre outras. Problemas definidos sobre reticulados podem variar em importância e em dificuldade, indo de problemas simples como codificar uma informaç ao, até problemas como encontrar o vetor mais curto ou o design de algortimos criptográficos "quantum resistant", passando ainda por teoria dos números algébricos e otimização combinatória.

O problema do vetor mais curto, definido sobre um reticulado, pertence à classe dos problemas "NP-hard". Essa característica pode ou não ser interessante, dependendo do ponto de vista do observador. Se por um lado é extremamente difícil exibir uma solução exata mesmo em instâncias de reticulados com posto pequeno, é essa característica que atrai o interesse de quem procura esquemas criptográficos altamente seguros e eficientes do ponto de vista das operações em reticulados.

Por que alguém se interessaria em encontrar o vetor mais curto em um reticulado?
Suponha que $b = \{b_1, \ldots, b_n\}$ é uma base do reticulado L, onde b_1 é um vetor mais curto possível. Uma informação escrita como $m = a_1 a_2 \cdots a_n$ poderia ser codificada como $M = \sum a_i b_i$ e transmitida por algum meio eletrônico. Ocorrendo alguma falha no processo de transmissão, esta informação poderia chegar ao receptor como $M = \sum a_i b_i + e$, onde e é um vetor erro decorrente dessa falha. Caso $||e|| < ||b_1||/2$, teremos condições de decodificar corretamente a informação que foi recebida.

Esse é um dos motivos pelos quais existe uma real necessidade de se buscar algoritmos que resolvam, pelo menos de forma aproximada o problema do vetor mais curto. Por um lado, algoritmos como o LLL produzem soluções aproximadas com grande eficiência mas deixam a desejar no requisito exatidão, especialmente em dimensões elevadas. Do outro lado,

novas gerações de algoritmos como “colônias de formigas”, “algoritmos genéticos” e “algoritmos culturais” podem ser alternativas atraentes quando existe a necessidade de resultados mais precisos. A proposta do algoritmo SVP-ACO é uma tentativa de suprir um pouco essa necessidade.

7.2 PROBLEMAS EM ABERTO

No contexto do problema do vetor mais curto, algumas questões importantes permanecem sem resposta. Dentre elas temos:

- **O algoritmo LLL pára quando $\delta = 1$?**

  É conhecido que o LLL roda em tempo polinomial e termina quando o parâmetro δ é estritamente menor que 1. Entretanto, nada se pode dizer quando $\delta = 1$.

- **Existem algoritmos de aproximação, determinísticos ou probabilísticos, que possam produzir soluções com fator de erro polinomial?**

  Conforme visto anteriormente o LLL tem uma margem de erro exponencial no posto do reticulado. Isto é, na verdade, característica de todos os algoritmos de aproximação para o SVP. O algoritmo LLL por exemplo, para $\delta = 0.75$, tem margem de erro de aproximadamente $(2/\sqrt{3})^n$. Para algoritmos como o SVP-ACO uma análise bastante criteriosa deve ser feita a fim de determinar sua margem de erro, uma vez que não temos acesso direto ao menor vetor do reticulado e os resultados são comparados aos do LLL.

  Uma consequência da não existência, até o presente momento, de tais algoritmos é que o SVP pode continuar a ser usado como suporte para a construção de esquemas criptográficos seguros.

- **Como os resultados obtidos são afetados por uma mudança na norma utilizada?**

  Do modo como foi concebido, o LLL depende completamente da norma euclideana ℓ_2. Uma mudanÃ§a na norma utilizada poderia interferir de algum modo nos resultados produzidos pelo LLL. Por outro lado, o algoritmo SVP-ACO pode ser facilmente modificado para que todas as medidas utilizadas possam ser calculadas com relação a qualquer norma. No entanto, não poderíamos mais comparar esses dois algoritmos pois estariam utilizando diferentes maneiras de calcular medidas.

7.3 TRABALHOS FUTUROS

Apresentaremos agora algumas das possíveis sequências do presente trabalho.

- **Ajuste ideal para os parâmetros:**

  Um parâmetro crucial para obter resultados precisos com o modelo proposto é a quantidade de formigas utilizadas. Dessa forma, deve-se estabelecer alguma relação entre o posto de um reticulado e a quantidade ideal de formigas que o modelo precisa para fornecer bons resultados.

- **Estudo da conexão entre o problema do vetor mais curto e transmissão de sinais digitais:**

  Reticulados fornecem um ambiente natural para codificação/decodificação de sinais. O conhecimento do vetor mais curto no respectivo reticulado, por sua vez, permite que a decodificação de sinais seja feita com precisão absoluta, desde que um possível erro que ocorra na transmissão não exceda determinado limite.

- **Implementação eficiente do SVP-ACO em dimensão grande:**

  A implementação do algoritmo SVP-ACO requer a solução de diversos problemas. Como criar um sistema eficiente de enumeração de vértices para permitir um rápido acesso à vizinhança de um vértice dado? Qual seria a melhor maneira de armazenar a matriz de incidência de uma região formada por cópias do paralelepípedo fundamental?

- **Elaborar formas de comparar os resultados obtidos pelo SVP-ACO com o menor vetor no reticulado**

  Atualmente essa comparação é feita de modo indireto através do algoritmo LLL. Existe uma maneira melhor de efetuar a avaliação dos resultados obtidos?

Referências Bibliográficas

[1] U. Erez and R. Zamir, *Achieving 0.5 log(1+SNR) over the additive white Gaussian noise channel with lattice encoding and decoding*, IEEE Trans. Inform. Theory, Vol. 50, pp. 2293-2314, Oct. 2004.

[2] H. El Gamal, G. Caire, and M. O. Damen, *Lattice coding and decoding achieve the optimal diversity-vs-multiplexing tradeoff of MIMO channels*, IEEE Trans. Inform. Theory, Vol. 50, pp. 968-985, June 2004.

[3] R. Zamir, *Lattices are everywhere*, Proc. of ITA, University of California at San Diego, Feb. 2009.

[4] Karen Aardal. *Lattice basis reduction and integer programming.* Technical Report UU-CS-1999-37, Universiteit Utrecht, 1999.

[5] R. Beckers, J. L. Deneubourg, and S. Goss, *Modulation of trail laying in the ant Lasius niger (hymenoptera: Formicidae) and its role in the collective selection of a food source*. Journal of Insect Behavior, 6(6), 751-759, 1993.

[6] S. Goldwasser, D. Micciancio, *Complexity of Lattice Problems: A Cryptographic Perspective*, The Kluwer International Series in Engineering and Computer Science, vol. 671. Kluwer Academic Publishers. March 2002.

[7] Lima, Elon Lages. *Espaços métricos*. Projeto Euclides. IMPA, Rio de Janeiro, 1993.

[8] S. Lang, *Introduction to Linear Algebra*. Springer, New York, 1986.

[9] Jeffrey Hoffstein, Jill Pipher, Joseph H. Silverman. *NTRU: A Ring Based Public Key Cryptosystem*. In Algorithmic Number Theory (ANTS III), Portland, OR, June 1998, J.P. Buhler (ed.), Lecture Notes in Computer Science 1423, Springer-Verlag, Berlin, 1998, 267-288.

[10] Miklós Ajtai and Cynthia Dwork, *A public-key cryptosystem with worst-case/average-case equivalence*. In Proceedings of the twenty-ninth annual ACM symposium on Theory of computing (El Paso, Texas, United States: ACM, 1977), 284-293.

[11] Peter van Emde Boas, *Another NP-complete problem and the complexity of computing short vectors in a lattice.* Tech. rep., University of Amsterdam, Department of Mathematics, Netherlands, 1981.

[12] M. Ajtai, R. Kumar, D. Sivakumar, *A sieve algorithm for the shortest lattice vector problem*. In Proceedings on 33nd Annual ACM Symposium on Theory oc Computing, STOC 2001, pages 266-275, Heraklion, Crete, Greece. ACM.

[13] A. K. Lenstra, H. W. Jr. Lenstra, L. Lovász, Factoring polynomials with rational coefficients. Mathematische Annalen, 261: 513-534, 1982.

[14] R. Kannan, *Minkowski's convex body theorem and integer programming*. Mathematics of operetion research, 12:415-440, 1987.

[15] C.P. Schnorr, M. Euchner. *Lattice Basis Reduction: Improved Practical Algorithms and Solving Subset Sum Problems*. Mathematical Programming, 66: 181-199, 1994.

[16] H. Koy, C.P. Schnorr. *Segment and Strong Segment LLL-Reduction of Lattice Bases*. Preprint, 2002.

[17] C.P. Schnorr, *A Hierarchy of Polynomial Lattice Basis Reduction Algorithms*. Theoretical Computer Science, 53: 201-224, 1987.

[18] C.P. Schnorr. *Lattice Reduction by Random Sampling and Birthday Methods*. Symposium on Theoretical Aspects of Computer Science 2003, 2607: 146-156.

[19] H. Koy, *Primal/duale Segment-Reduktion von Gitterbasen*. Lecture Universität Franfurt 2000, files from Mai 2004.
//www.mi.informatik.uni-frankfurt.de/research/papers.html

[20] Ravi Kannan. *Improved Algorithm for Integer Programming and Related Lattice Problems*. Proceedings of the Fifteenth Annual ACM Symposium on Theory of Computing, pages 193-206, 2003.

[21] M. Dorigo, *Ottimizzazione, apprendimento automatico, ed algoritmi basati su metafora naturale*. Ph.D.Thesis, Politecnico di Milano, Italy, in Italian, 1992.

[22] M. Dorigo, V. Maniezzo, A. Colorni, *The ant system: optimization by a colony of cooperating ants*. IEEE Trans. Syst. Man Cyber n, 26: 29-42, 1996.

[23] J. J. Deneubourg, S. Aron, S. Goss, J. M. Pasteels, *The self-organizing exploratory pattern of the Argentine ant*. Journal of Insect Behavior, 3: 159-168, 1990.

[24] M. Dorigo, T. Stützle, *Ant colony optimization*. MIT press, 2004.

[25] K. Ambos-Spies, *On the relative complexity of hard problems for complexity classes without complete problems*. TCS, 64 pp. 43-61, 1989.

[26] E. Bach, *Discrete logarithms and factoring*. Technical Report CSD-84-186, University of California at Berkeley, 1984.

[27] T. Baker, J. Gill, and R. Solovay, *Relativizations of the P =? NP question*. SICOMP: SIAM Journal on Computing, 1975.

[28] S. Cook, *The P versus NP Problem*. Manuscript prepared for the Clay Mathematics Institute for the Millennium Prize Problems, April 2000 (revised November 2000).

[29] S. C. Coutinho, *Primalidade em tempo polinomial: Uma introdução ao algoritmo AKS*. Coleção Iniciação Científica, SBM, 2004.

[30] M. Garey and D. Johnson, *Computers and Intractability: a Guide to the Theory of NP-Completeness*. Freeman, San Francisco, 1979.

[31] C. F. Gauss, *Disquisitiones Arithmeticae*. Braunschweig, 1801. English Edition, Springer-Verlag, New York, 1986.

[32] O. Goldreich, *Foundations of Cryptography: Basic Tools*. Cambridge Univ. Press, 2001.

[33] O. Goldreich, *Randomized Methods in Computation*. Lecture Notes, Spring 2001. Available at http://www.wisdom.weizmann.ac.il/õded/rnd.html

[34] G. H. Hardy and E. M. Wright, *An introduction to the theory of numbers*. Oxford University Press, 1979.

[35] S. Homer and A. Selman, *Computability and Complexity Theory*. Springer, 2000.

[36] R. M. Karp, *Reducibility Among Combinatorial Problems*. In Complexity of Computer Computations, Proc. Sympos. IBM Thomas J. Watson Res. Center, Yorktown Heights, N.Y.. New York: Plenum, p.85-103, 1972.

[37] S. Lang, *Algebra*. Graduate Texts in Mathematics, Springer, 2002.

[38] C. Papadimitriou, *Computational Complexity*. Addison-Wesley, Reading, MA, 1994.

[39] A. Selman, *Complexity Theory Retrospective*. Springer, New York, 1988.

[40] M. Sipser, *The history and status of the P versus NP question*. In Proceedings of the 24th ACM Symposium on the Theory of Computing, pp. 603-618. ACM, New York, 1992.

[41] M. Sipser, *Introduction to the Theory of Computation, second edition*. Course Technology (Thomson), 2005.

[42] D. Welsh, *Codes and Cryptography*. Oxford University Press, Oxford, 1988.

Internet

[43] Wikipedia, The Free Encyclopedia, *Norm (mathematics)*.
http://en.wikipedia.org/wiki/Norm_(mathematics)

[44] Wikipedia, The Free Encyclopedia, *Complexity classes P and NP*.
http://en.wikipedia.org/wiki/Complexity_ classes_ P_ and_ NP

[45] Wikipedia, The Free Encyclopedia, *NP-Hard*.
http://en.wikipedia.org/wiki/NP-hard