Universidade de Brasília
Departamento de Estatística

Otimização para o Problema da Cadeia de Caracteres Mais Próxima

Ramon Moreira Gonçalves

Projeto apresentado para o Departamento de Estatística da Universidade de Brasília como parte dos requisitos necessários para obtenção do grau de Bacharel em Estatística.

Brasília
2022

Ramon Moreira Gonçalves

Otimização para o Problema da Cadeia de Caracteres Mais Próxima

Orientador: Prof. Dr. Peter Zörnig

Projeto apresentado para o Departamento de Estatística da Universidade de Brasília como parte dos requisitos necessários para obtenção do grau de Bacharel em Estatística.

Brasília
2022

Resumo

Este trabalho aborda o Problema da Cadeia de Caracteres Mais Próxima (PCCP) - *Closest String Problem* (CSP). O objetivo do problema é encontrar uma cadeia de caracteres que minimiza a máxima distância para um conjunto de cadeias com mesmo comprimento e alfabeto fixo. A métrica utilizada é a distância de *Hamming*, que é definida como o número de posições nas quais duas sequências diferem. O CSP tem muitas aplicações, principalmente na biologia computacional e na teoria dos códigos. O objetivo dessa dissertação é implementar dois modelos presentes na literatura que buscam solucionar o CSP por meio de programação linear inteira, e então verificar as vantagens de um modelo em relação ao outro. O primeiro modelo aparece em Meneses et al. (2004, sec. 2.4) e também em Festa (2007, p. 228). Já o segundo modelo é proposto por Zörnig (2011, p. 5612). Para verificar qual modelo é melhor, foram feitas simulações com diversos comprimentos de sequências, a fim de observar o número de restrições e variáveis nos modelos. O segundo modelo se saiu melhor nos testes, pois o número de restrições e variáveis não depende do comprimento das sequências.

Palavras-chave: Problema da Cadeia de Caracteres Mais Próxima, Distância de *Hamming*, Relaxamento LP, Teoria dos códigos, Biologia Computacional.

Abstract

This work addresses the Closest String Problem (CSP). The purpose of the problem is to find a string of characters that minimizes the maximum distance for a set of strings with same length and fixed alphabet.The metric used is the Hamming distance, which is defined as the number of positions at which two sequences differ. The CSP has many applications, mainly in computational biology and theory of codes. The objective of this dissertation is to implement two models present in the literature that seek to solve the CSP through integer linear programming, and then check the advantages of one model over the other.The first model appears in Meneses et al. (2004, sec. 2.4) and also in Festa (2007, p. 228). The second model is proposed by Zörnig (2011, p. 5612). To verify which model is better, simulations were performed with different sequence lengths, in order to observe the number of constraints and variables in the models. The second model performed better in the tests, as the number of constraints and variables do not depend on the length of the sequences.

Keywords: Closest String Problem, Hamming Distance, LP Relaxation, Theory of the Codes, Computational Biology.

Sumário

1 Introdução

Desde que a estrutura do DNA foi desvendada em 1953, a biologia molecular testemunhou grandes avanços. Uma enorme quantidade de dados foi e está sendo gerada desde então devido ao aumento da capacidade de manipular sequências biomoleculares. A necessidade de processar a informação gerada criou problemas inteiramente novos, para os quais foi necessária a ajuda de outras disciplinas como Matemática e Ciência da Computação. Essa necessidade fez surgir um novo campo chamado Biologia Molecular Computacional (SETUBAL; MEIDANIS, 1997).

De acordo com Festa (2007) o surgimento do Projeto Genoma Humano em 1986 elevou a quantidade de pesquisas científicas dedicadas à biologia molecular e à compreensão das interações entre os vários sistemas de uma célula, incluindo interações de DNA, RNA e síntese de proteínas.

Apenas nos últimos anos, modelos de otimização foram analisados e propostos pela comunidade de pesquisa operacional, mostrando que um grande número de problemas de biologia molecular podem ser formulados como problemas de otimização combinatória. Pesquisadores da área fizeram uma observação fundamental no que diz respeito à abstração da estrutura tridimensional real do DNA e sua representação como uma sequência unidimensional de caracteres de um alfabeto de quatro símbolos. O mesmo tipo de suposição envolve também a proteína representada como uma sequência de caracteres de um alfabeto de vinte símbolos. Como resultado da codificação linear de DNA e proteínas, muitos problemas de biologia molecular foram formulados como problemas computacionais e de otimização envolvendo sequências, como, por exemplo, reconstruir longas sequências de DNA a partir de fragmentos sobrepostos, para comparar duas ou mais sequências procurando por suas semelhanças, para procurar padrões que ocorrem com uma certa frequência em sequências de DNA e/ou proteínas (FESTA, 2007).

Assim sendo, este trabalho aborda o Problema da Cadeia de Caracteres Mais Próxima (PCCP) - *Closest String Problem* (CSP), que busca encontrar uma cadeia de caracteres que minimiza a máxima distância para um conjunto de cadeias com mesmo comprimento e alfabeto fixo. Esse alfabeto pode ser formado pelas bases do DNA, RNA ou proteínas. O CSP tem muitas aplicações, principalmente na biologia computacional e na teoria dos códigos. Na teoria dos códigos, o objetivo é encontrar sequências de caracteres que são mais próximas de um determinado conjunto de caracteres. Isso é útil para determinar a melhor maneira de codificar um conjunto de mensagens (LIU et al., 2011).

Além do CSP, existem outros problemas de otimização combinatória, como o Problema da Cadeia de Caracteres Mais Distante - *Farthest String Problem* (FSP), cujo

objetivo é encontrar uma sequência que seja mais distante de um conjunto de cadeias de caracteres. Esse problema é tratado por Zörnig (2015) e também por Festa (2007, p. 229).

2 Objetivos

2.1 Objetivo Geral

Implementar dois modelos presentes na literatura que buscam solucionar o CSP por meio de programação linear inteira, e então verificar as vantagens de um modelo em relação ao outro. O primeiro modelo aparece em Meneses et al. (2004, sec. 2.4) e também em Festa (2007, p. 228). Já o segundo modelo é proposto por Zörnig (2011, p. 5612), e requer geralmente menos restrições e variáveis que o primeiro (especialmente quando um pequeno conjunto de sequências longas é dado).

2.2 Objetivos Específicos

- Implementar os modelos de programação linear em *software* LINGO;
- Gerar cadeias de caracteres utilizando a distribuição uniforme como suporte;
- Aplicar os modelos para conjuntos de três sequências com tamanhos variados;
- Analisar o desempenho dos modelos com relação ao número de variáveis e restrições por meio de simulações;
- Expandir e melhorar o segundo modelo para um maior número de sequências.

3 Metodologia

As sequências de caracteres serão geradas através do *software* R, com base em dois alfabetos:

- Alfabeto binário: segundo Meneses et al. (2004) este caso é interessante quando as sequências comparadas representam informações digitais, como por exemplo em dados gerados por computador;
- Alfabeto com quatro símbolos: representa as bases nitrogenadas do DNA ou RNA.

Vale ressaltar que, por meio de normalização o número de elementos do alfabeto pode ser reduzido ao número de sequências, se o primeiro for maior do que o segundo, para o modelo proposto por Zörnig (2011).

As cadeias de caracteres geradas serão agrupadas de acordo com o comprimento que será predeterminado, para que em seguida os modelos possam ser testados.

3.1 CSP

Os problemas de seleção e comparação de cadeias de caracteres pertencem à classe mais geral da biologia molecular que é conhecida como sequências de consenso, em que um conjunto finito de sequências é dado e alguém está interessado em encontrar seu consenso, ou seja, uma nova sequência que concorda tanto quanto possível com todas as sequências determinadas. Em outras palavras, o objetivo é determinar uma sequência chamada consenso, pois ela representa de alguma forma todas as sequências fornecidas. No caso do CSP, o consenso é uma nova sequência cuja distância máxima a todas as sequências fornecidas é mínima (FESTA, 2007).

Inicie com um alfabeto Ω, i.e. um conjunto finito de elementos, chamado caracteres. Sem perda de generalidade, assumimos que $\Omega = \{1, ..., \omega\}$ com $\omega \in \mathbb{N}$. Ω^m denota o conjunto de todas as sequências de comprimento m com elementos escolhidos de Ω. Para quaisquer duas sequências $s, t \in \Omega^m$, a distância de *Hamming* $d(s, t)$ entre s e t é definida como o número de posições nas quais s e t diferem. O CSP é definido do seguinte modo segundo Zörnig (2011, p. 5609):

Dado um conjunto de sequências $\Sigma = \{s^1, ..., s^n\}$ com $s^i = (s^i_1, ..., s^i_m) \in \Omega^m$ para $i = 1, ..., n$, encontre a sequência $t \in \Omega^m$ de modo que $D(t) = \max_{i=1,...,n} d(s^i, t)$ é mínima.

Um exemplo da distância de *Hamming* pode ser apresentado da seguinte forma:

Dadas duas sequências de comprimento igual a 10, $s^1 = (CGGGCCATTA)$ e $s^2 = (CGTGCCATCA)$, a respectiva distância de *Hamming* é calculada pela soma do

número de caracteres que diferem. Portanto a distância de *Hamming* entre s^1 e s^2 é igual a 2, já que $s_3^1 = G$ difere de $s_3^2 = T$, e $s_9^1 = T$ difere de $s_9^2 = C$.

3.2 Número de Stirling de segundo tipo

Em matemática, particularmente em combinatória, um número de Stirling de segundo tipo é o número de maneiras de particionar um conjunto de n objetos em k subconjuntos não vazios e é denotado por $S(n,k)$ ou $\left\{ {n \atop k} \right\}$. Temos que:

- $S(n,n) = 1$ para $n \geq 1$;
- $S(n,1) = 1$.

Os Números de Stirling de segundo tipo podem ser calculados da seguinte forma:

$$S(n,k) = \frac{1}{k!} \sum_{i=0}^{k} (-1)^i \binom{k}{i} (k-i)^n$$

3.3 Formulações dos modelos

De acordo com o objetivo geral deste trabalho, o modelo apresentado por Meneses et al. (2004, sec. 2.4) e também por Festa (2007, p. 228) será chamado de Modelo I, e o modelo proposto por Zörnig (2011, p. 5612) será chamado de Modelo II.

3.3.1 Modelo I

A abordagem mais amplamente utilizada para resolver o CSP que é conhecido por ser NP-difícil consiste em modelá-lo como um problema de programação linear inteira (ZÖRNIG, 2011, p. 5609). Considere a matriz CSP:

$$S = \begin{pmatrix} s_1^1 & \dots & s_m^1 \\ \vdots & & \vdots \\ s_1^n & \dots & s_m^n \end{pmatrix} \tag{3.3.1}$$

As linhas consistem nas sequências em Σ. O modelo é baseado no fato de que os componentes t_j de uma solução ótima $t = (t_1, \dots, t_m)$ do CSP deve pertencer ao conjunto $V_j = \{s_j^i | i = 1, \dots, n\}$ de diferentes caracteres que aparecem na j-ésima coluna da matriz (3.3.1). Assim, podemos restringir nossa atenção a tais sequências que serão chamadas

de soluções viáveis do CSP. Observe que os números v_j de elementos em V_j podem variar entre 1 e $min(\omega, n)$. A fim de codificar qualquer solução viável dado $t = (t_1, \ldots, t_m)$, definimos uma variável $\mathrm{x}_{i,j}$ para qualquer $j = 1, \ldots, m$ e qualquer $i \in V_j$. Para um índice fixo j, definimos (ZÖRNIG, 2011, p. 5610):

$$\mathrm{x}_{i,j} = \begin{cases} 1 \text{ se } t_j = i \\ 0 \text{ caso contrário.} \end{cases}$$

O CSP pode ser formulado como em Meneses et al. (2004, sec. 2.4) ou Festa (2007, p. 228):

$$\begin{aligned} &\min d \\ \text{s.a } & m - \sum_{j=1}^{m} \mathrm{x}_{s_j^i, j} \leq d \text{ para } i = 1, \ldots, n, \\ & \sum_{i \in V_j} \mathrm{x}_{i,j} = 1 \text{ para } j = 1, \ldots, m, \\ & d \geq 0 \text{ inteiro }, \mathrm{x}_{i,j} \in \{0, 1\} \text{ para } j = 1, \ldots, m,\ i \in V_j, \end{aligned} \tag{3.3.2}$$

em que os lados esquerdos das primeiras n restrições representam as distâncias de *Hamming* $d(t, s^i)$ para $i = 1, \ldots, n$. O modelo tem $1 + \sum_{j=1}^{m} v_j$ variáveis , $n + m$ restrições lineares (e uma restrição inteira para cada variável).

Algoritmos de aproximação de melhor desempenho propostos na literatura são baseados em relaxamento de programação linear do modelo de programação inteira anterior. A ideia básica consiste em formular o problema como o programa inteiro descrito acima, resolvendo sua programação linear relaxada (i.e. o problema sem condições de integralidade) e usando o resultado do problema relaxado para encontrar uma solução aproximada para o problema original (FESTA, 2007).

3.3.2 Modelo II

A ideia básica do Modelo II consiste em resumir colunas agrupadas da matriz CSP (3.3.1) em conjuntos que são isomórficos no sentido especificado abaixo. Isso pode resultar em uma redução considerável do número de variáveis e restrições e como um subproduto reduz o acúmulo de erros de arredondamento (ZÖRNIG, 2011, p. 5610).

Definição 1 *Dada uma matriz CSP S em (3.3.1), com* $s_j^\bullet = \left(s_j^1, s_j^2, \ldots, s_j^n\right)^T$ *sendo uma coluna de S. A partição induzida do conjunto* $s_j^\bullet$ *é definida como a partição não ordenada do conjunto de linhas* $\{1, \ldots, n\}$*, cujos componentes correspondem a valores iguais de* $s_j^\bullet$*. Duas colunas de S são chamadas de isomórficas se induzem ao mesmo conjunto de*

partição.

Por exemplo, sejam $s_1^\bullet = (4,1,1,3,4,4,1)^T$, $s_2^\bullet = (3,2,2,4,3,3,2)^T$ e $s_3^\bullet = (3,3,1,2,1,1,2)^T$ três colunas da matriz S. Então $s_1^\bullet$ e $s_2^\bullet$ induzem a mesma partição $\{1,\ldots,7\} = \{1,5,6\} \cup \{2,3,7\} \cup \{4\}$, enquanto $s_3^\bullet$ induz a partição $\{1,\ldots,7\} = \{1,2\} \cup \{3,5,6\} \cup \{4,7\}$. Assim, $s_1^\bullet$ e $s_2^\bullet$ são isomórficas, enquanto $s_1^\bullet$ e $s_3^\bullet$ não são.

Definição 2 *Seja $\{1,\ldots,n\} = C_1 \cup \ldots \cup C_r$ uma partição com r componentes, em que C_i é rotulado de tal forma que $min(C_1) < min(C_2) < \ldots < min(C_r)$. Então o vetor representativo da partição é definido como o vetor tendo o número i nas posições correspondentes ao componente $C_i(i,\ldots,r)$.*

Por exemplo, os vetores representativos das duas partições diferentes acima são $(1,2,2,3,1,1,2)^T$ e $(1,1,2,3,2,2,3)^T$, respectivamente.

Claramente, para um CSP com tamanho de alfabeto ω e n sequências, o número de classes de isomorfismo de vetores representativos é dado por (ZÖRNIG, 2011, p. 5610):

$$\sum_{r=1}^{\min(n,\omega)} S(n,r) \tag{3.3.3}$$

em que $S(n,r)$ são os números de Stirling de segundo tipo (veja Seção 3.2).

Para qualquer matriz CSP S, podemos agora definir uma matriz normalizada correspondente T, substituindo qualquer coluna em S pelo vetor representativo da partição do conjunto induzido. As soluções viáveis/ótimas do CSP original tem correspondência um a um com as soluções viáveis/ótimas do normalizado (ZÖRNIG, 2011, p. 5611).

Em particular, a normalização reduz o tamanho do alfabeto ω para no máximo o número n de sequências em Σ.

Em geral, a sequência $s = (s_1,\ldots,s_m)$ corresponde a $t = (t_1,\ldots,t_m)$ se s_j e t_j ocorrem na mesma posição da respectiva coluna. Assim, para as soluções viáveis correspondentes s e t de S e T, respectivamente, a Distâncias de *Hamming* $d(s,s^i)$ e $d(t,t^i)$ são iguais para qualquer índice i, e s é ótimo para S se e somente se t for ótimo para T. Portanto a solução do CSP original pode ser facilmente recuperada da solução do CSP normalizado.

De forma geral, o CSP normalizado agora pode ser modelado como em Zörnig

(2011, p. 5612):

$$\begin{aligned} &\min d \\ \text{s.a } & m - \sum_{j=1}^{k} \mathrm{y}_{t_j^i,j} \leq d \text{ para } i = 1, \ldots, n, \\ & \sum_{i=1}^{v_j} \mathrm{y}_{i,j} = m_j \text{ para } j = 1, \ldots, k, \\ & d, \mathrm{y}_{i,j} \text{ inteiros não negativos,} \end{aligned} \tag{3.3.4}$$

em que $\mathrm{y}_{i,j}$ representa a frequência do caractere i nas posições de t correspondentes à j-ésima classe de isomorfismo de vetores representativos. Os parâmetros k, m_j, v_j denotam o número de tais classes que ocorrem no CSP normalizado, o tamanho dessas classes e o número de caracteres diferentes nessas classes, respectivamente. O primeiro índice t_j^i nas desigualdades denota o i-ésimo elemento do j-ésimo vetor representativo. O comprimento das sequências é $m = m_1 + \ldots + m_k$.

O número de variáveis e restrições lineares no modelo II são máximos quando todos os vetores representativos possíveis ocorrem na matriz CSP normalizada. Então, para um comprimento de sequência arbitrário m (maior do que o número de classes de isomorfismo), estes números são dados por $1 + \sum_{j=2}^{k} v_j = \sum_{j=1}^{\min(n,\omega)} jS(n,j)$ e $n + k = n + \sum_{j=2}^{\min(n,\omega)} S(n,j)$ (ZÖRNIG, 2011, p. 5612).

4 Resultados comparativos

Primeiro será mostrado como é a aplicação prática dos dois modelos, e em seguida será feita uma simulação para avaliar o desempenho de ambos. A simulação será feita em *software* R.

4.1 Aplicação prática dos modelos

Primeiro são geradas três sequências a partir da distribuição Uniforme, com comprimento $m = 10$ e alfabeto $\Omega = \{1, 2, 3, 4\}$. Essas sequências formam a matriz CSP S e sua respectiva matriz normalizada T.

Tabela 1: Matriz CSP e matriz normalizada

Posição		1	2	3	4	5	6	7	8	9	10
S	s^1	1	4	1	4	3	3	1	4	4	1
	s^2	1	4	4	2	2	3	1	3	3	4
	s^3	1	1	3	4	4	4	1	2	1	4
T	t^1	1	1	1	1	1	1	1	1	1	1
	t^2	1	1	2	2	2	1	1	2	2	2
	t^3	1	2	3	1	3	2	1	3	3	2

Claramente, qualquer coluna j com caracteres idênticos pode ser eliminada de uma matriz CSP, já que, neste caso, o j-ésimo elemento de qualquer solução viável é determinado de forma única. Assim podemos eliminar as colunas 1 e 7 da matriz CSP.

4.1.1 Modelo I

Temos que:

$$S = \begin{pmatrix} 4 & 1 & 4 & 3 & 3 & 4 & 4 & 1 \\ 4 & 4 & 2 & 2 & 3 & 3 & 3 & 4 \\ 1 & 3 & 4 & 4 & 4 & 2 & 1 & 4 \end{pmatrix} \tag{4.1.1}$$

Com base no modelo 3.3.2 queremos:

$$
\begin{aligned}
& \min d \\
\text{s.a } & 8 - \mathrm{x}_{4,1} - \mathrm{x}_{1,2} - \mathrm{x}_{4,3} - \mathrm{x}_{3,4} - \mathrm{x}_{3,5} - \mathrm{x}_{4,6} - \mathrm{x}_{4,7} - \mathrm{x}_{1,8} \leq d \\
& 8 - \mathrm{x}_{4,1} - \mathrm{x}_{4,2} - \mathrm{x}_{2,3} - \mathrm{x}_{2,4} - \mathrm{x}_{3,5} - \mathrm{x}_{3,6} - \mathrm{x}_{3,7} - \mathrm{x}_{4,8} \leq d \\
& 8 - \mathrm{x}_{1,1} - \mathrm{x}_{3,2} - \mathrm{x}_{4,3} - \mathrm{x}_{4,4} - \mathrm{x}_{4,5} - \mathrm{x}_{2,6} - \mathrm{x}_{1,7} - \mathrm{x}_{4,8} \leq d \\
& \mathrm{x}_{4,1} + \mathrm{x}_{1,1} = 1 \\
& \mathrm{x}_{1,2} + \mathrm{x}_{4,2} + \mathrm{x}_{3,2} = 1 \\
& \mathrm{x}_{4,3} + \mathrm{x}_{2,3} = 1 \\
& \mathrm{x}_{2,4} + \mathrm{x}_{3,4} + \mathrm{x}_{4,4} = 1 \\
& \mathrm{x}_{3,5} + \mathrm{x}_{4,5} = 1 \\
& \mathrm{x}_{2,6} + \mathrm{x}_{3,6} + \mathrm{x}_{4,6} = 1 \\
& \mathrm{x}_{1,7} + \mathrm{x}_{3,7} + \mathrm{x}_{4,7} = 1 \\
& \mathrm{x}_{1,8} + \mathrm{x}_{4,8} = 1 \\
& d \geq 0 \text{ inteiro }, \mathrm{x}_{i,j} \in \{0,1\} \text{ para } j = 1, \ldots, 8, \ i \in V_j, \qquad (4.1.2)
\end{aligned}
$$

O modelo acima tem 21 variáveis e 11 restrições lineares, além de uma restrição inteira para cada variável. Resolvendo o problema relaxado do modelo acima via *Lingo*, obtemos os seguintes resultados:

- d = 4;
- $t = (1, 4, 1, 4, 4, 3, 1, 3, 1, 4)$, já com valores nas posições 1 e 7. Ou seja, a sequência t minimiza $D(t) = \max_{i=1,2,3} d(s^i, t)$.

4.1.2 Modelo II

Pela equação (4.2.1) temos que o número de classes de isomorfismo de vetores representativos é dado por:

$$\sum_{r=1}^{\min(3,4)} S(3,r) = S(3,1) + S(3,2) + S(3,3) = 5$$

Os vetores representativos são:

$$\begin{pmatrix} 1 \\ 1 \\ 1 \end{pmatrix}, \begin{pmatrix} 1 \\ 1 \\ 2 \end{pmatrix}, \begin{pmatrix} 1 \\ 2 \\ 1 \end{pmatrix}, \begin{pmatrix} 1 \\ 2 \\ 2 \end{pmatrix}, \begin{pmatrix} 1 \\ 2 \\ 3 \end{pmatrix} \qquad (4.1.3)$$

Como o primeiro vetor acima é composto por caracteres iguais, ele pode ser removido.

Para o modelo II, iremos resolver o CSP na forma normalizada. Considerando a matriz T na Tabela 1 podemos reorganizar as colunas resultando em quatro grupos de colunas associadas a quatro classes de isomorfismo de vetores representativos. Isso resulta na Tabela 2.

Tabela 2: Matriz normalizada com colunas reordenadas

Posição		1	2	3	4	5	6	7	8
T	t^1	1	1	1	1	1	1	1	1
	t^2	1	1	2	2	2	2	2	2
	t^3	2	2	1	2	3	3	3	3
Grupo da coluna		1		2	3		4		

Qualquer solução viável t pode agora ser representada por meio das frequências $y_{i,j}$ do caractere i no grupo j.

Uma solução do CSP normalizado acima pode agora ser obtida, resolvendo o problema de programação linear inteira:

$$\begin{aligned} & \min d \\ \text{s.a } & 8 - y_{1,1} - y_{1,2} - y_{1,3} - y_{1,4} \leq d \\ & 8 - y_{1,1} - y_{2,2} - y_{2,3} - y_{2,4} \leq d \\ & 8 - y_{2,1} - y_{1,2} - y_{2,3} - y_{3,4} \leq d \\ & y_{1,1} + y_{2,1} = 2 \\ & y_{1,2} + y_{2,2} = 1 \\ & y_{1,3} + y_{2,3} = 1 \\ & y_{1,4} + y_{2,4} + y_{3,4} = 4 \\ & d \text{ e } y_{i,j} \text{ inteiros não negativos.} \end{aligned} \tag{4.1.4}$$

Observe que o primeiro índice de $y_{i,j}$ nas desigualdades é o i-ésimo elemento do vetor representativo correspondente (veja 4.1.3).

O modelo acima possui 10 variáveis e 7 restrições lineares, além de uma restrição inteira para cada variável. Resolvendo o problema relaxado do modelo acima via *Lingo*, obtemos os seguintes resultados:

- $$\begin{aligned} &y_{1,1} = 2, \quad y_{1,2} = 1, \quad y_{1,3} = 0, \quad y_{1,4} = 1, \\ &y_{2,1} = 0, \quad y_{2,2} = 0, \quad y_{2,3} = 1, \quad y_{2,4} = 1, \\ &\qquad\qquad\qquad\qquad\qquad\quad y_{3,4} = 2 \end{aligned}$$

- d = 4;

- $t = (1, 4, 1, 4, 2, 3, 1, 2, 1, 4)$, já com valores nas posições 1 e 7. Ou seja, a sequência t minimiza $D(t) = \max_{i=1,2,3} d(s^i, t)$.

Os valores da sequência t foram obtidos por meio das frequências $y_{i,j}$ e da reordenação dos valores para as posições originais (da Tabela 2 para a Tabela 1).

Comparando ambos os modelos é possível notar que o modelo II requer menos variáveis e restrições que o primeiro. Agora será feita uma simulação ainda com 3 sequências, porém, o comprimento delas será maior, a fim de se verificar qual dos modelos é melhor.

4.2 Simulações

Primeiro foram feitas 1000 simulações com 3 sequências de comprimento $m = 100$ e outras 1000 com $m = 1000$. Todas simulações foram feitas com base em um alfabeto $\Omega = \{1, 2, 3, 4\}$. Em seguida retiram-se todas as colunas com caracteres idênticos e foi calculado o número de restrições e variáveis para o Modelo I, já que no Modelo II esses números não se alteram com o comprimento das sequências.

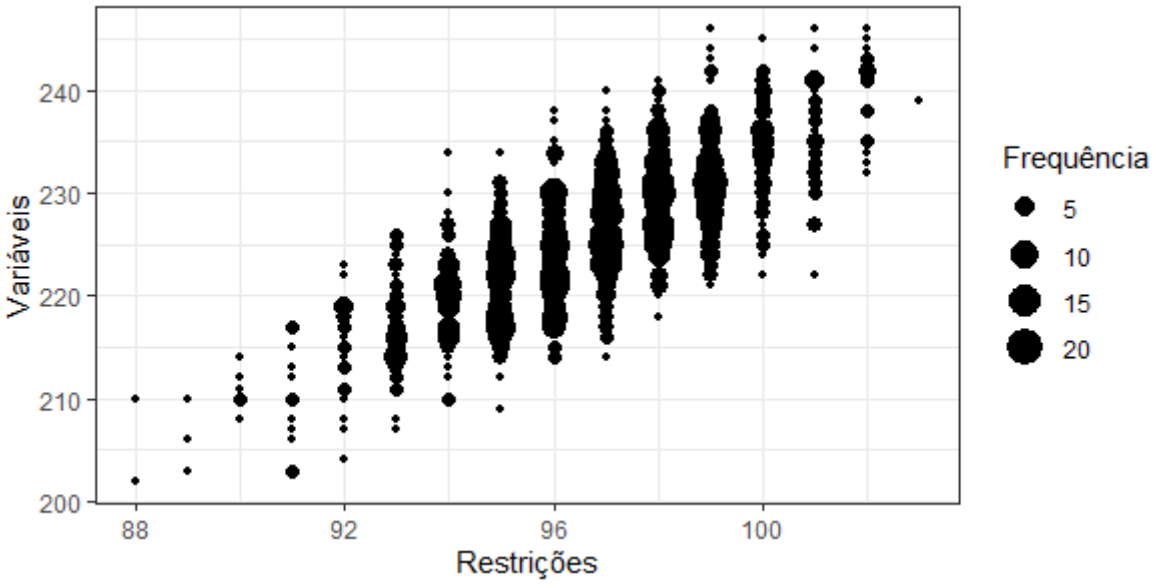

Figura 1: Simulações para o modelo I com $m = 100$

Pela figura acima vemos que na grande maioria dos casos o modelo I teve mais de 200 variáveis e mais de 90 restrições, enquanto no Modelo II são necessárias 10 variáveis e 7 restrições. Por fim veremos as simulações para $m = 1000$.

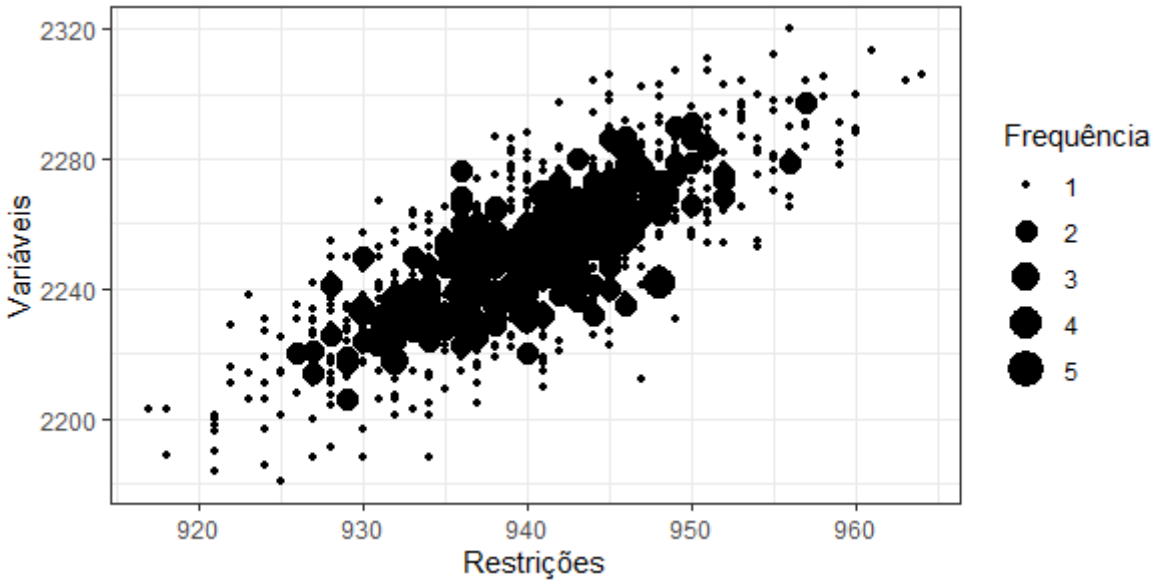

Figura 2: Simulações para o modelo I com $m = 1000$

No caso com $m = 1000$ a diferença entre os modelos é ainda maior, com mais de 900 restrições e 2000 variáveis no modelo I contra 7 restrições e 10 variáveis do Modelo II. Com isso é possível notar que o Modelo II é muito superior ao modelo I, especialmente nos casos onde o comprimento das sequências é grande (o que geralmente ocorre na prática). Na próxima seção o Modelo II será aprofundado, a fim de otimizá-lo ainda mais.

5 Aprofundamento do Modelo II

Nesta seção, o estudo do Modelo II será dividido em dois ramos. O primeiro tratará do caso com alfabeto binário, que é muito importante quando se estuda informações digitais. O segundo ramo será voltada para o caso em que $\min(n, \omega) = n$, ou seja, quando o número de sequências é menor ou igual ao número de elementos do alfabeto Ω.

Os gráficos com o número de variáveis e restrições desta seção foram construídos com base em Zörnig (2011, p. 5612).

5.1 Caso com alfabeto binário

Recapitulando, o Modelo II pode ser escrito da seguinte forma:

$$\begin{aligned} &\min d \\ \text{s.a } & m - \sum_{j=1}^{k} \mathrm{y}_{t_j^i,j} \leq d \text{ para } i = 1, \ldots, n, \\ & \sum_{i=1}^{v_j} \mathrm{y}_{i,j} = m_j \text{ para } j = 1, \ldots, k, \\ & d, \mathrm{y}_{i,j} \text{ inteiros não negativos,} \end{aligned} \tag{5.1.1}$$

Para o caso binário temos que t_j^i pode assumir os valores 0 ou 1. Com isso, $\mathrm{y}_{t_j^i,j}$ pode ser escrito como x_j se $t_j^i = 0$ e y_j se $t_j^i = 1$. As restrições do tipo $\sum_{i=1}^{v_j} \mathrm{y}_{i,j} = m_j$ passam a ser $\mathrm{y_j} + \mathrm{x_j} = \mathrm{m_j}$. Além disso, $\mathrm{x_j}$ e $\mathrm{y_j}$ são inteiros não negativos.

Tome como exemplo as 3 sequências a seguir com $m = 8$ e alfabeto $\Omega = \{0, 1\}$.

$$S = \begin{pmatrix} 0 & 1 & 1 & 0 & 1 & 0 & 0 & 1 \\ 0 & 0 & 0 & 1 & 0 & 1 & 1 & 0 \\ 1 & 1 & 1 & 1 & 0 & 1 & 0 & 1 \end{pmatrix} \tag{5.1.2}$$

Em sua forma normalizada:

$$T = \begin{pmatrix} 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 1 & 1 & 1 & 1 & 1 & 1 \\ 1 & 0 & 0 & 1 & 1 & 1 & 0 & 0 \end{pmatrix} \tag{5.1.3}$$

Pela equação (4.2.1) temos que o número de classes de isomorfismo de vetores

representativos é dado por:

$$\sum_{r=1}^{\min(3,2)} S(3,r) = S(3,1) + S(3,2) = 4$$

Os vetores representativos são:

$$\begin{pmatrix}0\\0\\0\end{pmatrix}, \begin{pmatrix}0\\0\\1\end{pmatrix}, \begin{pmatrix}0\\1\\0\end{pmatrix}, \begin{pmatrix}0\\1\\1\end{pmatrix} \tag{5.1.4}$$

Como o primeiro vetor acima é composto por caracteres iguais, ele pode ser removido.

Uma solução pode ser obtida resolvendo o problema de programação linear inteira:

$$\begin{aligned}
&\min d \\
\text{s.a } & 8 - \mathrm{x}_1 - \mathrm{x}_2 - \mathrm{x}_3 \leq d \\
& 8 - \mathrm{x}_1 - \mathrm{y}_2 - \mathrm{y}_3 \leq d \\
& 8 - \mathrm{y}_1 - \mathrm{x}_2 - \mathrm{y}_3 \leq d \\
& \mathrm{x}_1 + \mathrm{y}_1 = 1 \\
& \mathrm{x}_2 + \mathrm{y}_2 = 4 \\
& \mathrm{x}_3 + \mathrm{y}_3 = 3 \\
& d, \mathrm{x}_j \text{ e } \mathrm{y}_j \text{ inteiros não negativos.}
\end{aligned} \tag{5.1.5}$$

Observando o problema acima, é possível perceber que as variáveis do tipo y_j podem ser escritas como $\mathrm{m_j} - \mathrm{x_j}$. As restrições do tipo $\mathrm{y_j} + \mathrm{x_j} = \mathrm{m_j}$ passam a ser $\mathrm{x_j} \leq \mathrm{m_j}$, com os valores de $\mathrm{x_j}$ sendo inteiros não negativos. Com essas modificações o modelo passa a ter k variáveis a menos, ou seja, uma variável a menos para cada classe de isomorfismo de vetor representativo, sem considerar o vetor formado por caracteres iguais. Para o caso com alfabeto binário, o número de variáveis cai praticamente pela metade, com o mesmo número de restrições. O exemplo acima fica da seguinte maneira:

$$
\begin{aligned}
& \min d \\
\text{s.a } & 8 - x_1 - x_2 - x_3 \leq d \\
& 8 - x_1 - (4 - x_2) - (3 - x_3) \leq d \\
& 8 - (1 - x_1) - x_2 - (3 - x_3) \leq d \\
& x_1 \leq 1 \\
& x_2 \leq 4 \\
& x_3 \leq 3 \\
& d \text{ e } x_j \text{ inteiros não negativos.}
\end{aligned} \tag{5.1.6}
$$

Em notação LINGO:

```
!3 Sequências binárias;

SETS:
      NUMBERS /1..3/: X,M ;
 ENDSETS

min=d;

8-x(1)          -x(2)          -x(3)         <=d;
8-x(1)          -(m(2)-x(2))   -(m(3)-x(3))<=d;
8-(m(1)-x(1))   -x(2)          -(m(3)-x(3))<=d;

m(1) = 1; m(2) = 4; m(3) = 3;

@FOR( NUMBERS(I):          x(I) <= m(I));
```

Figura 3: Equação 5.1.6 escrita sem as restrições de integralidade.

Resolvendo o problema acima por meio de relaxamento LP chegamos em $d = 4$.

A figura abaixo mostra o número de variáveis para o Modelo II e para sua versão reduzida conforme o número de sequências cresce.

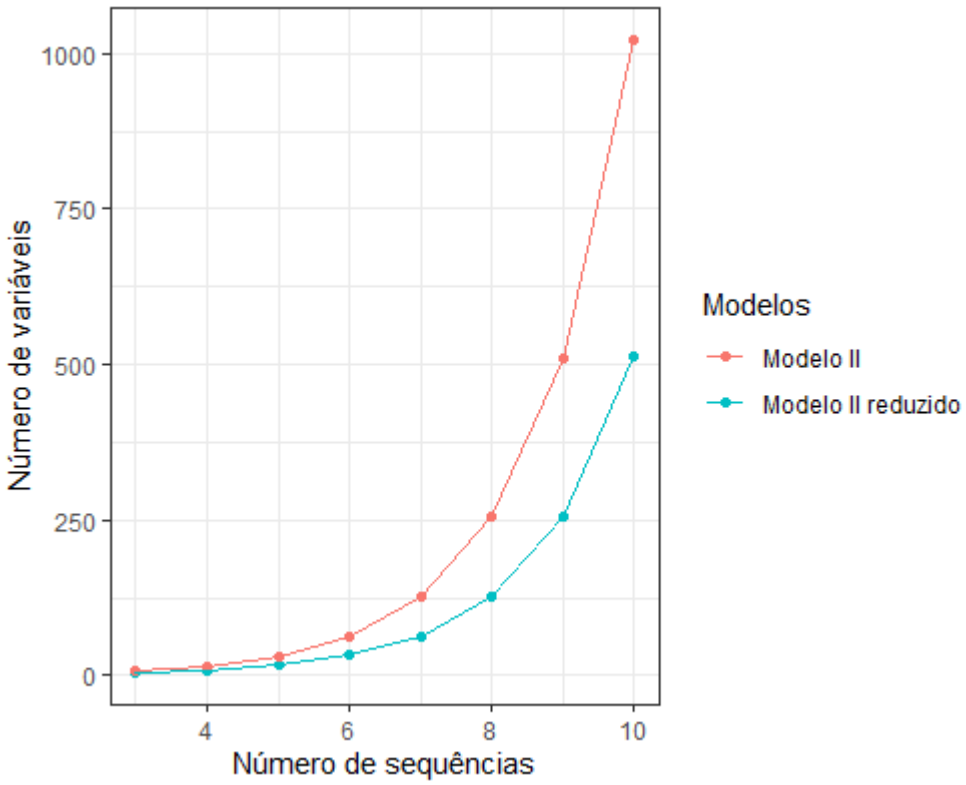

Figura 4: Número de variáveis para as duas versões com alfabeto binário do Modelo II

Pela figura acima é possível notar que as manipulações algébricas reduziram significativamente o número de variáveis, porém, o número de variáveis cresce exponencialmente conforme o número de sequências aumenta, principalmente para $n > 8$.

Abaixo é possível notar como o número de restrições cresce no caso binário.

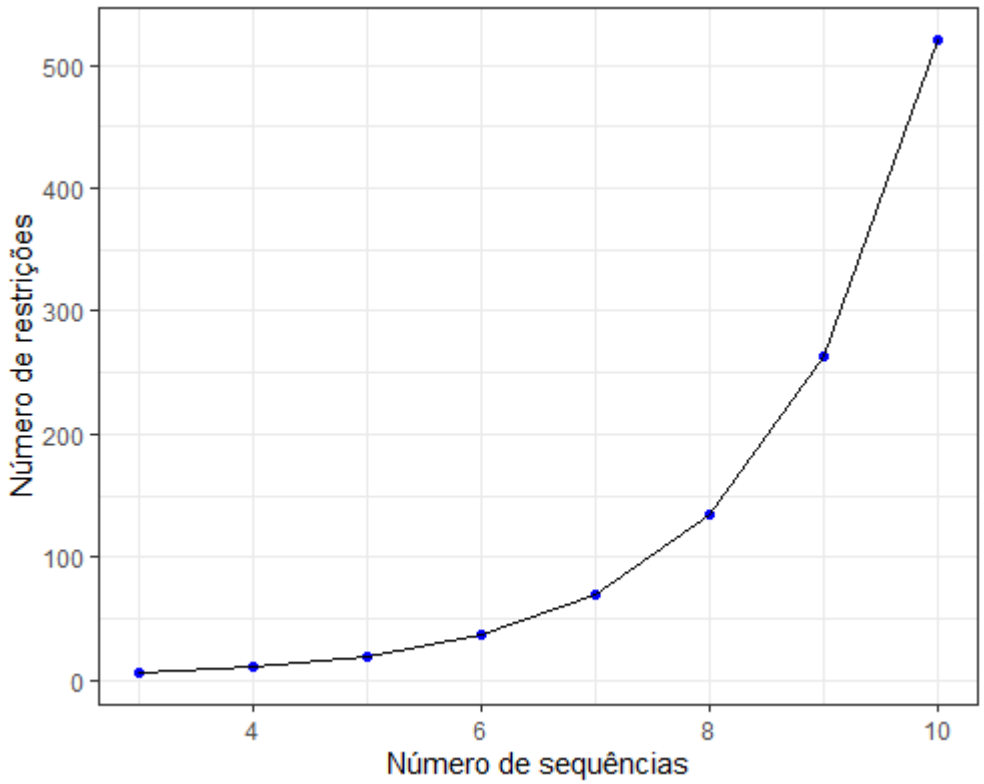

Figura 5: Número de restrições para o Modelo II com alfabeto binário

5.2 Segundo caso

Agora será visto o em que onde $\min(n, \omega) = n$, ou seja, o caso com o maior número de variáveis e restrições, dado um determinado número de sequências. Seguindo a mesma linha de raciocínio do caso binário, as variáveis mais à direta antes da igualdade que aparecem nas restrições do tipo $\sum_{i=1}^{v_j} \mathrm{y_{i,j}} = m_j$ para $j = 1, \ldots, k$ no modelo 5.1.1 serão substituídas pelo $\mathrm{m_j}$ menos as demais variáveis em cada restrição para cada $\mathrm{j} = 1, \ldots, \mathrm{k}$. Pegando o exemplo 4.1.4:

$$
\begin{aligned}
& \min d \\
\text{s.a } & 8 - y_{1,1} - y_{1,2} - y_{1,3} - y_{1,4} \leq d \\
& 8 - y_{1,1} - y_{2,2} - y_{2,3} - y_{2,4} \leq d \\
& 8 - y_{2,1} - y_{1,2} - y_{2,3} - y_{3,4} \leq d \\
& y_{1,1} + y_{2,1} = 2 \\
& y_{1,2} + y_{2,2} = 1 \\
& y_{1,3} + y_{2,3} = 1 \\
& y_{1,4} + y_{2,4} + y_{3,4} = 4 \\
& d \text{ e } y_{i,j} \text{ inteiros não negativos.}
\end{aligned} \tag{5.2.1}
$$

Fazendo as manipulações algébricas:

$$
\begin{aligned}
& \min d \\
\text{s.a } & 8 - y_{1,1} - y_{1,2} - y_{1,3} - y_{1,4} \leq d \\
& 8 - y_{1,1} - (1 - y_{1,2}) - (1 - y_{1,3}) - y_{2,4} \leq d \\
& 8 - (2 - y_{1,1}) - y_{1,2} - (4 - y_{1,3}) - (1 - y_{1,4} - y_{2,4}) \leq d \\
& y_{1,1} \leq 2 \\
& y_{1,2} \leq 1 \\
& y_{1,3} \leq 1 \\
& y_{1,4} + y_{2,4} \leq 4 \\
& d, y_{1,1}, y_{1,2}, y_{1,3}, y_{1,4} \text{ e } y_{2,4} \text{ inteiros não negativos.}
\end{aligned} \tag{5.2.2}
$$

Com essas modificações o modelo passa a ter k variáveis a menos. A figura abaixo mostra a diferença.

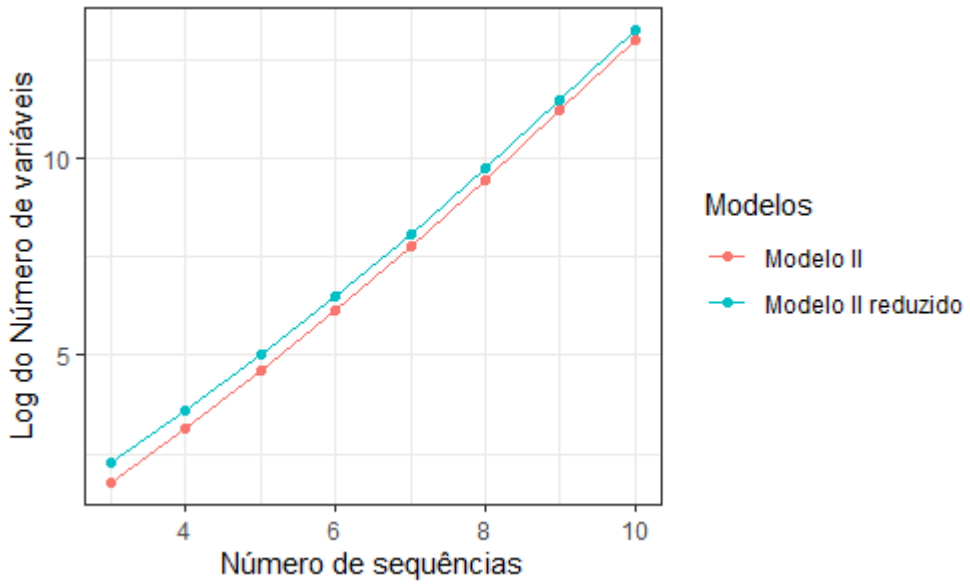

Figura 6: *Log* do número de variáveis para as duas versões do Modelo II

Pela figura acima é possível notar que o número de variáveis reduziu consideravelmente, porém, o ganho não foi tão significativo como no caso com alfabeto binário. Também é possível notar que o número de variáveis ultrapassa 10000 com $n = 8$.

Para finalizar, será visto como o número de restrições cresce no modelo II.

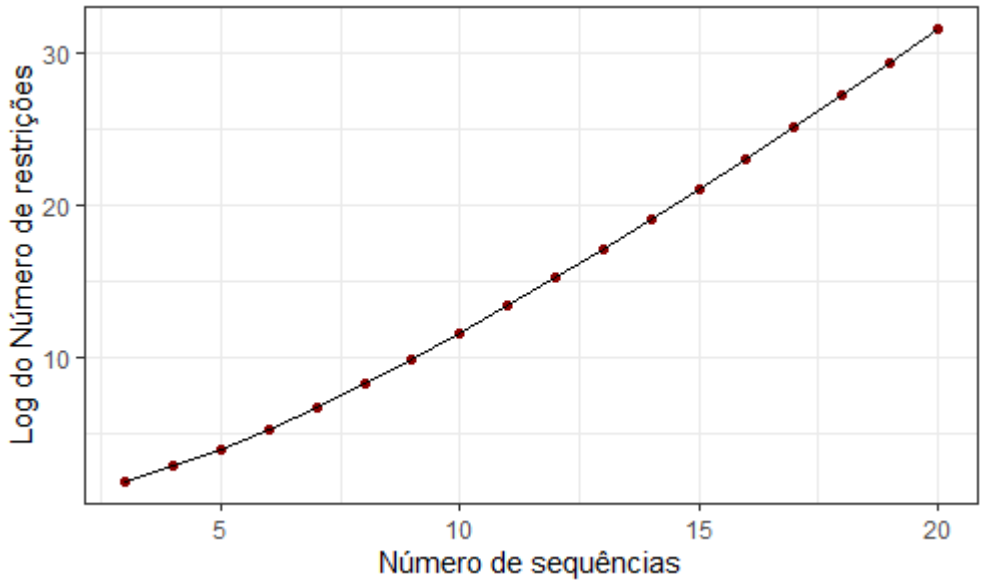

Figura 7: *Log* do número de restrições para o Modelo II

A partir da figura acima é visto que o número de restrições é muito alto a partir de $n = 8$, com $n = 10$ o número ultrapassa 100.000. Também é possível notar que no modelo binário o número de restrições é muito menor se comparado com o caso acima.

5.2.1 Caso com duas sequências

O caso com duas sequências é o mais simples, pois há apenas o vetor representativo $(0, 1)^T$. O valor d que minimiza a máxima distância entre as sequências é dado por:

- $d = m_1/2$, se m_1 é par;
- $d = (m_1 + 1)/2$, se m_1 é impar.

Chega-se nesse resultado pois como há apenas duas sequências, o consenso entre as duas é formado por metade dos valores de cada sequência, sem considerar os casos onde os valores são iguais nas duas sequências.

Para ver outros programas em LINGO basta acessar ⟨https://drive.google.com/drive/folders/14Uxp-nuDAkYZPuqrSyDOzeO8wjX5_5QT?usp=sharing⟩.

6 Conclusão

Esta dissertação apresentou o Problema da Cadeia de Caracteres Mais Próxima (PCCP) - *Closest String Problem* (CSP), com aplicações na biologia computacional e na teoria dos códigos.

Para resolver o CSP por meio de programação linear inteira, foram utilizados dois modelos presentes na literatura: O Modelo I aparece em Meneses et al. (2004, sec. 2.4) e também em Festa (2007, p. 228). Já o Modelo II é proposto por Zörnig (2011, p. 5612).

Para testar os modelos foram feitas simulações em R, com três sequências com diversos comprimentos. O Modelo II foi muito superior ao primeiro, já que o número de variáveis e restrições não mudam de acordo com o comprimento das sequências. Como na prática o objetivo é comparar sequências muito longas, o Modelo II é fortemente recomendado para encontrar a solução exata do CSP. Vale lembrar que a solução ótima exata nos dois modelos é a mesma, porém, as soluções relaxadas podem variar, já que há arredondamentos. Além disso, no Modelo II há menos arredondamentos que no Modelo I, pois há menos variáveis com restrições de integralidade.

Em seguida foi feito um aprofundamento no Modelo II em dois ramos. Primeiro foi visto o caso com alfabeto binário, que é muito utilizado na teoria dos códigos. Através de manipulações algébricas foi possível reduzir o número de variáveis praticamente pela metade, o número de restrições permaneceu o mesmo.

O segundo caso teve um foco quando o número de sequências é menor ou igual ao tamanho do alfabeto. As manipulações algébricas reduziram o número de variáveis, porém o ganho não foi tão significativo.

Por fim, foi possível notar que para sequências muito grandes o número de variáveis e restrições cresce exponencialmente. O *software* LINGO, por exemplo, limita o número de restrições e variáveis em quase todas as versões, como mostra a figura abaixo:

Version	Total Variables	Integer Variables	Nonlinear Variables	Global Variables	Constraints
Demo/Web	300	30	30	5	150
Solver Suite	500	50	50	5	250
Super	2,000	200	200	10	1,000
Hyper	8,000	800	800	20	4,000
Industrial	32,000	3,200	3,200	50	16,000
Extended	*Unlimited*	*Unlimited*	*Unlimited*	*Unlimited*	*Unlimited*

Figura 8: Limitações das versões do LINGO

Uma das formas de tentar contornar essa situação é a utilização de outros métodos heurísticos, baseados no Modelo II, para solucionar o CSP.

Referências

CHEN, Z.-Z.; MA, B.; WANG, L. A three-string approach to the closest string problem. *Journal of Computer and System Sciences*, Elsevier, v. 78, n. 1, p. 164–178, 2012.

FESTA, P. On some optimization problems in molecular biology. *Mathematical biosciences*, Elsevier, v. 207, n. 2, p. 219–234, 2007.

KELSEY, T.; KOTTHOFF, L. Exact closest string as a constraint satisfaction problem. *Procedia Computer Science*, Elsevier, v. 4, p. 1062–1071, 2011.

LANCTOT, J. K. et al. Distinguishing string selection problems. *Information and Computation*, Elsevier, v. 185, n. 1, p. 41–55, 2003.

LI, M.; MA, B.; WANG, L. On the closest string and substring problems. *Journal of the ACM (JACM)*, ACM New York, NY, USA, v. 49, n. 2, p. 157–171, 2002.

LIU, X. et al. Exact algorithm and heuristic for the closest string problem. *Computers & operations research*, Elsevier, v. 38, n. 11, p. 1513–1520, 2011.

MENESES, C. N. et al. Optimal solutions for the closest-string problem via integer programming. *INFORMS Journal on Computing*, INFORMS, v. 16, n. 4, p. 419–429, 2004.

SETUBAL, J. C.; MEIDANIS, J. *Introduction to computational molecular biology*. [S.l.]: PWS Pub. Boston, 1997.

ZÖRNIG, P. Improved optimization modelling for the closest string and related problems. *Applied Mathematical Modelling*, Elsevier, v. 35, n. 12, p. 5609–5617, 2011.

ZÖRNIG, P. Reduced-size integer linear programming models for string selection problems: Application to the farthest string problem. *Journal of Computational Biology*, Mary Ann Liebert, Inc. 140 Huguenot Street, 3rd Floor New Rochelle, NY 10801 USA, v. 22, n. 8, p. 729–742, 2015.