CONSTRUÇÃO DE UM ESTIMADOR PESSIMISTA PARA O PROBLEMA DA SEQUÊNCIA MAIS PRÓXIMA

Keity Yamamoto, Carlos A. Martinhon, Helena C. G. Leitão
Instituto de Computação, Universidade Federal Fluminense
Rua Passo da Pátria 156, Bloco E, Sala 303,
São Domingos, 24210-230, Niterói, RJ, Brasil
{kyama@acol.com.br, mart@dcc.ic.uff.br, hcgl@ic.uff.br}

Resumo

No Problema da Seqüência Mais Próxima-PSMP (*Closest String Problem*) desejamos encontrar uma seqüência de caracteres que mais se aproxime, segundo uma dada métrica, de um dado conjunto S de seqüências de mesmo tamanho. Em outras palavras, desejamos minimizar a maior distância desta seqüência às demais seqüências do conjunto. Este artigo discute uma estratégia de *derandomização* para o PSMP baseada no método das probabilidades condicionais. Apresentamos inicialmente um algoritmo randômico aproximado para o problema, já proposto na literatura, e mostramos como gerar uma solução determinística utilizando-se o método dos estimadores pessimistas. Esta estratégia é utilizada normalmente em situações onde as probabilidades condicionais necessárias na *derandomização* não podem ser determinadas diretamente.

Palavras-chave – Método Probabilístico, Biologia Computacional, *Derandomização*. Algoritmos Randômicos Aproximativos.

Abstract

In the *Closest String Problem* – CSP the objective is to find a sequence of characters that is closer, according to a given metric, to a set *S* of sequences of the same size. In another words, the objective is to minimize the major distance between this sequence and the other sequences of *S*. In this paper we discuss a derandomization strategy for the CSP based on the method of conditional probabilities. We first present a randomized approximation algorithm for the CSP already proposed in the literature, and we show how to produce a deterministic solution through the method of pessimistic estimators. This approach is normally used when the conditional probabilities necessaries in the derandomization process cannot be directly determined.

Keywords – Probabilistic Method, Computational Biology, Derandomization, Randomized Approximation Algorithms.

1 INTRODUÇÃO

Em muitos problemas da biologia molecular deseja-se comparar e encontrar regiões comuns ou que sejam próximas a um conjunto de seqüências de DNA, RNA ou proteínas. Estes problemas são encontrados em várias aplicações importantes como: busca de regiões conservadas em seqüências não alinhadas, identificação de drogas genéticas, formulação de sondas (*probes*) genéticas, entre outras [HS95, LR90, PBPR89, LBMM91, Stor90, SH91, WAG84, WG86, WP84].

No Problema da Seqüência Mais Próxima – PSMP (*Closest String Problem*) desejamos determinar uma seqüência que mais se aproxime, segundo alguma métrica, de um dado conjunto de seqüências. Em outras palavras, desejamos minimizar a maior distância desta seqüência às demais

seqüências do conjunto. O PSMP tem sido bastante estudado na literatura. Frances e Litman [FL97] mostraram que o problema é NP-difícil. Berman *et al.* [BGHMS97] apresentaram um algoritmo exato polinomial de uma versão parametrizada. Neste caso, a distância entre a seqüência a ser determinada e o conjunto dado de seqüências será no máximo uma constante. Ben-Dor *et al.* [BLPR97] e Gasieniec *et al.* [GJL99] apresentaram um algoritmo aproximativo, com razão de performance próxima do valor ótimo quando este é suficientemente grande. Lanctot *et al.* [LLMWZ99] obtiveram um algoritmo $(4/3+e)$-aproximado (para $e > 0$) e Li *et al.* [LMW02] apresentaram um esquema de aproximação polinomial para o problema. Finalmente, Pardalos *et al.* [PMLO04] propuseram métodos exatos baseados em *branch-and-bound* e três novas formulações de programação linear inteira para o problema.

Na Seção 2, definimos formalmente o PSMP e apresentamos alguns conceitos básicos importantes. Posteriormente, nas Seções 3 e 4, discutimos de maneira sucinta as técnicas de arredondamento randômico (*Randomized Rounding*) e uma estratégia de *derandomização* presente na literatura conhecida como método das Probabilidades Condicionais. Na Seção 5, apresentamos o algoritmo randômico aproximativo proposto por Ben-Dor *et. al.* [BLPR97] para o PSMP e mostramos, na Seção 6, como aplicar o método da probabilidades condicionais na construção de um algoritmo determinístico com a mesma razão de performance. Em nosso caso, as probabilidades condicionais necessárias para a aplicação deste método não podem ser determinadas diretamente, apresentamos então estimadores pessimistas (definidos convenientemente) visando a determinação de uma nova solução aproximada determinística para o PSMP. Embora o algoritmo proposto em [BLPR97] possua razão de performance inferior aos algoritmos randômicos apresentados em [LLMWZ99] e [LMW02], a técnica de derandomização aqui apresentada pode ser adaptada similarmente para estes dois últimos casos.

2 DEFINIÇÃO DO PROBLEMA E CONCEITOS BÁSICOS

Várias medidas têm sido propostas para medir a distância (ou diferença) entre duas seqüências de caracteres de mesmo tamanho. A distância de *Hamming*, por exemplo, é calculada simplesmente contando-se o número de posições correspondentes dos caracteres onde estas seqüências diferem. Uma justificativa mais técnica para o uso freqüente da distância de *Hamming* na comparação de seqüências em biologia molecular, pode ser encontrada em [LLMWZ99]. De maneira geral, dado um alfabeto finito de símbolos Σ, representaremos a distância entre duas seqüências de tamanho k por uma função $d_1:\Sigma^k x \Sigma^k \rightarrow [a,b]$, onde a e b são reais positivos. A distância entre dois caracteres será representada fazendo-se $k=1$.

Considere agora um conjunto $S=\{s_1,s_2,...,s_m\}$ de seqüências (todas de tamanho n) sobre Σ e seja $s_i[j]$, o j-ésimo caractere de $s_i \in S$. No PSMP deseja-se encontrar uma seqüência $s=s_H$ de tamanho n que minimize d de forma que, para cada $s_i \in S$, tenhamos $d_1(s_H, s_i) \leq d$, para $i=1...m$. Salvo indicação em contrário, considere $d_1(s_i[j],s_H[j]) \in [a,b]$ (onde $j \in \{1,..,n\}$), a distância entre o j-ésimo caractere de s_i e s_H respectivamente. Formalmente, temos:

$$\begin{aligned} &\min\ d \\ &s.a\ \begin{cases} d_1(s_H, s_i) \leq d, & i = 1,\ldots,m \\ s_H \in \Sigma^n \end{cases} \end{aligned}$$

onde Σ^n representa o conjunto de todas as seqüências com n caracteres.

Apresentamos agora alguns resultados auxiliares (*Chernoff-Hoeffding bounds*), que serão utilizados na análise da razão de aproximação do algoritmo proposto por Ben-Dor *et. al.* [BLPR97].

Lema 1: Sejam X_1, X_2, ..., X_n, variáveis 0-1 aleatórias independentes, onde $X_i = 1$ com probabilidade p_i e $0 < p_i < 1$. Considere ainda $X = \sum_{i=1}^{n} X_i$ e $m = E[X]$. Então, $\forall\ e \in (0,1]$, temos:

$$\Pr\left(X > m + en\right) < \exp\left(-\frac{1}{3}ne^2\right).$$

Lema 2: Sejam $X_1, X_2, ..., X_n$, variáveis aleatórias independentes, onde X_i (p/ i=1,..,n) assume valores no intervalo real [a,b]. Além disso, seja $X = \sum_{i=1}^{n} X_i$ e $d > 0$. Então:

$$\Pr\left(X \geq (1+d)E[X]\right) \leq \exp\left(-E[X]d^2 / 3(b-a)^2\right).$$

Definição 1: (Algoritmo Randômico α-aproximado)

Dizemos que um algoritmo randômico polinomial A para um problema de minimização π é α-aproximado se, e somente se, $E(Z_H) \leq \alpha.opt$, onde $\alpha \geq 1$, Z_H representa o valor da solução heurística obtida por A e opt representa o valor ótimo de π. Equivalentemente, dizemos que A será α-aproximado para π se, e somente se, $\Pr(Z_H \leq \alpha.opt) \geq ½$.

3 ARREDONDAMENTO RANDÔMICO (*Randomized Rounding*)

Na técnica de arredondamento randômico, introduzida inicialmente por Raghavan&Thompson [RT87], formula-se primeiramente um modelo de Programação Linear Inteira – PLI para o problema original. A solução da relaxação linear associada definirá então probabilidades a serem utilizadas na etapa randômica, buscando-se, dessa forma, a determinação de soluções viáveis de boa qualidade para o problema original (solução aproximada).

Sem perda de generalidade, considere o seguinte modelo de Prog. Linear Inteira 0-1:

$$opt = \min c^T x$$
$$s.a \quad \begin{cases} Ax \geq b \\ x \in \{0,1\}^n \end{cases} \qquad (PLI)$$

onde $A \in \Re^{mxn}$, $c \in \Re^{nx1}$ e $b \in \Re^{mx1}$. A Relaxação Linear – RL do PLI acima é obtida substituindo-se simplesmente $x \in \{0,1\}^n$ por $x \in [0,1]^n$. Uma solução ótima polinomial para RL pode ser obtida com a utilização de métodos de pontos interiores [Wrig97]. Assim, se $y \in [0,1]^n$ representa uma solução ótima fracionária de RL, teremos então os seguintes passos no procedimento de Arredondamento Randômico:

1. Formule o problema original como um modelo de Programação Linear Inteira (PLI),
2. $y \leftarrow$ Retorne uma solução de RL (em tempo polinomial);
3. **Repita**
   - 3.1 Utilize y para calcular (através de uma função de arredondamento randômico linear ou não-linear) uma solução inteira para o PLI
   - 3.2 $x_H \leftarrow$ Salva melhor solução viável (de menor custo) do PLI

   Até (condição de parada)
4. Retorna x_H e custo $Z_H = c^T x_H$.

Trata-se na verdade , de uma aplicação do método de Monte Carlo [MR95]. Note que a solução com coordenadas inteiras obtida no Passo 3.1 pode ser inviável. Assim, se B é um evento representando a probabilidade de fracasso (solução inviável para o PLI) devemos repetir o processo até que a probabilidade de inviabilidade seja arbitrariamente pequena (condição de parada). Em outras palavras, se $\varepsilon > 0$ e k representa o número de repetições, a probabilidade de fracasso, deverá ser tal que $\Pr(B)^k < e$. Para isso, obviamente, devemos garantir que $Pr(B) < 1$ em uma iteração qualquer do algoritmo. No caso dos algoritmos randômicos α-aproximados, esperamos ainda que B_C (evento

complementar de B) represente uma solução viável com razão de performance α. Para maiores detalhes sobre a aplicação do método de Monte Carlo vide [MR95].

4 PROBABILIDADES CONDICIONAIS E ESTIMADORES PESSIMISTAS

Em determinadas situações será possível construir algoritmos puramente determinísticos a partir de algoritmos randômicos utilizando-se técnicas ou ferramentas probabilísticas. Em outras palavras, deseja-se construir um algoritmo determinístico sem que se sacrifique muito a qualidade da solução e/ou tempo de processamento obtidos no procedimento randômico. Infelizmente, não se conhece um mecanismo universal de conversão que seja aplicável a todas as situações.

Apesar de não existir um termo apropriado em português para esta técnica, iremos chamá-la simplesmente de *derandomização* (semelhante ao termo inglês *derandomization*). Entre os métodos de *derandomização* mais conhecidos na literatura podemos citar: o método das probabilidades condicionais, método das expectâncias condicionais, método dos estimadores pessimistas, *k-wise independence* entre outros [AS92]. Neste artigo, utilizamos o método das probabilidades condicionais e o método dos estimadores pessimistas introduzido inicialmente por Raghavan [Ragh88]. Em seu trabalho, Raghavan mostra que, uma vez garantida uma solução viável com probabilidade de sucesso estritamente positiva na etapa de arredondamento randômico, uma nova solução viável $x \in \{0,1\}^n$ do PLI poderá, em muitos casos, ser obtida de maneira determinística.

No método das probabilidades condicionais é feita uma analogia com árvores de decisão, construindo-se, deterministicamente, um vetor $x \in \{0,1\}^n$ correspondente ao caminho de descida da raiz da árvore de decisão até uma folha representando um "bom" evento.

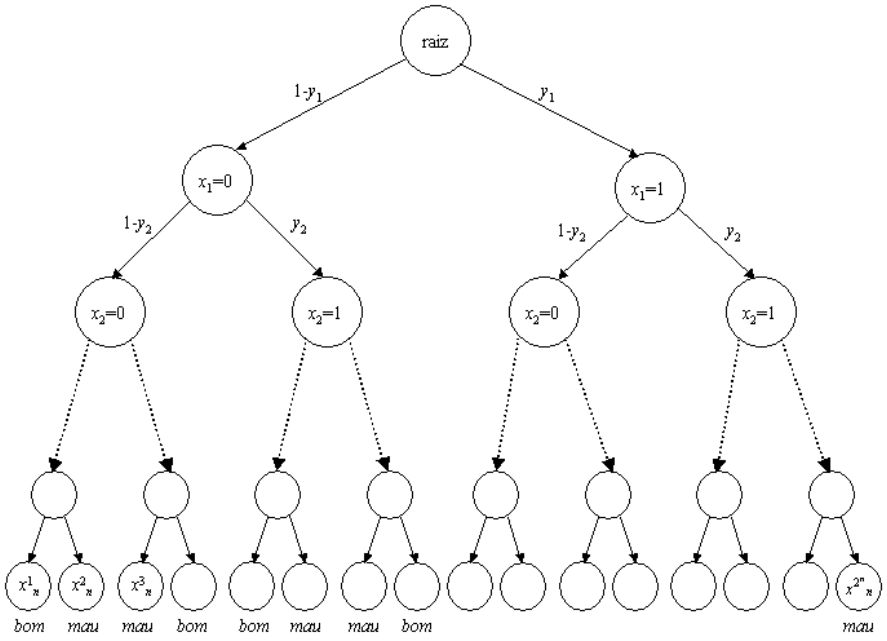

Figura 1: Árvore de decisão para busca de uma solução viável

A Figura 1 ilustra uma árvore binária completa T com n níveis. O j-ésimo nível de T (onde $j \in \{1..n\}$) irá representar uma atribuição de valores 0-1 à variável aleatória x_j. Cada folha da árvore irá corresponder a um *bom* ou *mau* evento. No caso dos algoritmos randômicos α-aproximados, um evento B_C será *bom*, se a solução x obtida for viável e seu custo associado estiver dentro da razão de aproximação α, caso contrário, diremos que o evento B é *mau* evento. Em outras palavras, um *mau* evento B representa uma solução inviável ou uma solução que não satisfaz à razão de performance pretendida. Assim, nosso objetivo será percorrer a árvore T da raiz até uma folha *boa* de T em tempo determinístico polinomial.

A caminhada na árvore de decisão é realizada da seguinte forma. Se à variável aleatória x_1 é atribuído o valor 1, percorre-se da raiz para seu filho à direita. Se x_1=0, percorre-se para o filho à esquerda, e assim sucessivamente até que se chegue a uma folha da árvore *T*.

Note que, cada folha de *T*, corresponde a uma entre 2^n seqüências possíveis de $x=(x_1,x_2,...,x_n)$ (representada na Figura 1 por x^i, onde $i \in \{1..\ 2^n\}$).

No método probabilístico, devemos garantir inicialmente que Pr(*B*)<1 (probabilidade de um *mau* evento), ou equivalentemente, Pr(B_C)>0. A questão natural que se coloca agora é: como percorrer da raiz até uma folha *boa* de *T* ?

Considere então P_1=Pr(*B*) e $y=(y_1, y_{2,...,} y_n)$ uma solução da Relaxação Linear do RL discutida na seção precedente. Da expressão de probabilidade absoluta [Mey83] tem-se que:

$$P_1 = \Pr(x_1=1).P_2(B|x_1=1) + \Pr(x_1=0).P_2(B|x_1=0).$$

No arredondamento randômico, se fazemos Pr(x_1=1) = y_1 e Pr(x_1=0) = 1 - y_1 (arredondamento linear) teremos então que: $P_1 \geq y_1.\min\{P_2(B|x_1=1); P_2(B|x_1=0)\} + (1-y_1).\min\{P_2(B|x_1=1); P_2(B|x_1=0)\} = P_2(B|X_1)$, onde X_1= 1 se, e somente se, $P_2(B|x_1=1) \leq P_2(B|x_1=0)$, e X_1= 0, caso contrário.

De maneiro geral, seja *j* um nível de *T*, para algum $j \in \{1..n\}$, e $P_j(B|X_1,..., X_{j-1})$ a probabilidade condicional de ocorrência de um *mau* evento, dado que os valores $X_1,..., X_{j-1}$ já tenham sido determinados. Tem-se então que:

$$P_j\left(B \middle| X_1,..., X_{j-1}\right) = y_j P_{j+1}\left(B \middle| X_1,..., X_{j-1}, x_j = 1\right) + \left(1 - y_j\right) P_{j+1}\left(B \middle| X_1,..., X_{j-1}, x_j = 0\right)$$

$$P_j\left(B \middle| X_1,..., X_{j-1}\right) \geq \min\left\{P_{j+1}\left(B \middle| X_1,..., X_{j-1}, x_j = 1\right), P_{j+1}\left(B \middle| X_1,..., X_{j-1}, x_j = 0\right)\right\} = P_{j+1}\left(B \middle| X_1,..., X_j\right)$$

Observe finalmente que, se $P_1 = Pr(B) < 1$ então:

$$1 > P_1 \geq P_2\left(B \middle| X_1\right) \geq ... \geq P_{n+1}\left(B \middle| X_1,..., X_n\right) = \Pr(folha)$$

Como Pr(*folha*) < 1, garantimos então que a solução determinística será uma folha *boa* de *T*, visto que $P_{n+1}(B|X_1,...,X_n)$ = Pr(*folha*) = 0. Em outras palavras, uma solução determinística será obtida com probabilidade de erro igual a *zero.*

Como as probabilidades condicionais nem sempre são determinadas facilmente, outras técnicas de *derandomização* poderão ser utilizadas [AS92]. No método dos *Estimadores Pessimistas* por exemplo, introduzido por Raghavan [Ragh88], as probabilidades condicionais são limitadas superiormente por uma função $U:[0,1]^n \rightarrow [0,1)$ (denominada estimador pessimista) e satisfazendo às seguintes condições:

1) $U_1(y_1,..., y_n) < 1$, onde *y* é uma solução da relaxação linear do PLI.
2) $U_{j+1}(X_1,..., X_j, y_{j+1},..., y_n) \geq P_{j+1}(B|X_1,..., X_j)$ onde $j \in \{1,...,n\}$ e X_k (para k=1,...,*j*) é uma atribuição dada às variáveis $x_k \in \{0,1\}$.
3) $U_j(X_1,..., X_{j-1}, y_j,..., y_n) \geq \min\{U_{j+1}(X_1,..., X_{j-1}, x_j=1, y_{j+1},..., y_n)\ ,\ U_{j+1}(X_1,..., X_{j-1}, x_j=0, y_{j+1},..., y_n)\}$.

A estratégia de *derandomização* será idêntica àquela descrita acima, bastando substituir agora as probabilidades condicionais pelos estimadores pessimistas correspondentes. Obviamente, neste caso, a função $U:[0,1]^n \rightarrow [0,1)$ deverá ser determinada e computada facilmente.

5 ALGORITMO RANDÔMICO APROXIMATIVO DE Ben-dor *et al.* [BLPR97]

Descrevemos nesta seção o algoritmo randômico aproximado desenvolvido por [BLPR97] para o PSMP. Apresentamos inicialmente seu modelo de Programação Linear Inteira 0-1 e o arredondamento randômico utilizado.

Seja $\Sigma=\{s_1,s_2,...,s_P\}$ um alfabeto finito com p símbolos, $S=\{s_1,s_2,...,s_m\}$ um conjunto de m seqüências de Σ^n e s_{opt}, uma seqüência ótima com n caracteres. São definidos para cada caracter $s \in \Sigma$ e cada caracter $s_{opt}[j]$ (para $j=1,..,n$), uma variável binária $x_{j,s}$, onde $x_{j,s}=1$ se, e somente se, $s_{opt}[j]=s$, e $x_{j,s}=0$, caso contrário. Considere então o seguinte modelo de Programação Linear Inteira 0-1:

$$d_{opt} = \min d \qquad (1)$$

$$s.a \begin{cases} \sum_{s\in\Sigma} x_{j,s} = 1 & , j=1..n \quad (2) \\ \sum_{j=1}^{n} \sum_{s\in\Sigma} x_{j,s}\, d_1(s, s_i[j]) \le d & , i=1..m \quad (3) \\ x_{j,s} \in \{0,1\} & , j=1..n\,;\, s\in\Sigma \quad (4) \end{cases}$$

No modelo apresentado, (1) representa a função objetivo que se deseja minimizar. As restrições (2) asseguram que somente um símbolo $s \in \Sigma$ será selecionado em cada posição da solução ótima s_{opt}. As desigualdades (3) especificam que a distância entre cada seqüência de S e s_{opt}, deve ser menor ou igual a um valor d. As desigualdades (4) indicam que somente valores 0 ou 1 podem ser atribuídos às variáveis $x_{j,s}$. A seguir apresentamos uma descrição detalhada do algoritmo randômico aproximado de [BLPR97].

Entrada: $s_1,s_2,...,s_m \in \Sigma^n$
Saída: seqüência $s_H \in \Sigma^n$ e a distância d_H.
Início
1. Resolver o modelo de relaxação linear, retornando os valores relaxados y_{js} e d_y
2. Construir a seqüência s_H a partir dos valores relaxados y_{js}
   Para $j=1,...,n$ **faça**
   $\Pr(s_H[j]=s) = y_{js}$, onde $s \in \Sigma$
3. Calcular a distância $d_H = \max_{1\le i\le m}\left\{ d_1(s_H, s_i) = \sum_{1\le j\le n} d_1(s_H[j], s_i[j]) \right\}$
4. Retornar seqüência s_H e distância d_H.
Fim

Algoritmo 1: Algoritmo aproximativo de [BLPR97]

No passo (1), resolve-se o modelo de relaxação linear obtendo-se valores fracionários para $y_{j,s}$, onde $1 \le j \le n$ e $s \in \Sigma$. Com os valores de $y_{j,s}$ no passo (2), constrói-se a solução com coordenadas inteiras s_H através do processo de arredondamento randômico. Finalmente, no passo (3), calcula-se a maior distância d_H, comparando a solução s_H com todas as seqüências do conjunto S.

5.1 Análise de Aproximação de Ben-Dor *et al*. [BLPR97]

Como observado no Algoritmo 1, o vetor y com coordenadas fracionárias $y_{j,s}$ (definindo probabilidades na etapa de arredondamento randômico), representa uma solução da relaxação de (1)-(4) e d_y representa seu valor ótimo associado. Note ainda que uma solução inteira s_H obtida após o arredondamento randômico, será sempre viável já que seu valor associado d_H é calculado posteriormente à obtenção de s_H. Logo, na análise de aproximação, deveremos verificar apenas se o valor da solução inteira d_H satisfaz ou não à razão de performance pretendida.

Suponha que s_H seja uma solução inteira $(1+\delta)$-aproximada do PSMP para algum $\delta > 0$. Da Definição 1, devemos provar então que $E(d_H) \le (1+\delta)d_{opt}$ para algum $\delta>0$, onde $d_H=max\{d_1(s_H,s_i) | i=1,..,m\}$. Equivalentemente, podemos provar que $\Pr(B)<1$ onde $B \equiv (d_H > (1+\delta)d_{opt})$, representa a probabilidade de falha (ou de um mau evento).

Considere agora $X_i = d_1(s_i, s_H)$, variáveis aleatórias representado a distância entre as seqüências s_i e s_H (para $i=1,...,m$). Temos então, o seguinte resultado preliminar demonstrado em [BLPR97]:

Lema 3: $E[X_i] \leq d_y$, para i=1..m.

Note que o *mau* evento B ocorre se, e somente se, $B_i \equiv (X_i > (1+d)d_{opt})$ ocorre para pelo menos um índice $i \in \{1..m\}$. Assim, dado $0<e<1$, devemos provar que:

$$\Pr(B) = \Pr\left(\bigcup_{i=1}^{m} B_i \right) \leq e < 1$$

Como $d_y \leq d_{opt}$ temos do Lema 3 que: $E[X_i] \leq d_{opt}$, $\forall\ i \in \{1,..,m\}$. Além disso, como $d>0$, espera-se então que: $\Pr(B_i) = \Pr(X_i > (1+\delta)d_{opt}) \leq \Pr(X_i > (1+d)E(X_i)) \leq e/m$, $\forall\ i$=1,..,m.

Por outro lado, da desigualdade de *Chernoff-Hoeffding* (Lema 2), tem-se que:

$$\Pr(X_i \geq (1+d)E[X_i]) \leq \exp\left(-E[X_i]d^2/3D^2\right),\ \forall\ i=1..m. \qquad (5)$$

onde $X_i = \sum_{j=1}^{n} X_{ij}$, $X_{ij} = d_1(s_i[j], s_H[j])$ para j=1,..,n e $D = max_{a,b \in \Sigma}\{d_1(a, b)\}$.

Note da desigualdade (5) que as variáveis aleatórias X_{ij} (para j=1..n) são independentes e assumem valores no intervalo real [0,D], onde D representa a maior distância entre dois caracteres do alfabeto Σ (neste caso, a = 0 e b = D).

Finalmente, esperamos escolher d e e de maneira que:

$$\exp\left(-E[X_i]d^2/3D^2\right) \leq e/m\ ,\ \forall\ i=1..m.$$

Novamente do Lema 3, como $E[X_i] \leq d_{opt}$, temos:

$$\Pr(B_i) = \Pr(X_i > (1+d)d_{opt}) \leq \frac{1}{\exp\left(d_{opt}d^2/3D^2\right)} \leq \frac{1}{\exp\left(E[X_i]d^2/3D^2\right)} \leq \frac{e}{m},\ \forall\ i=1..m. \qquad (6)$$

Portanto, de (6) devemos ter $d \geq D\left(3\ln(m/e)/d_{opt}\right)^{1/2}$, $\forall\ i$=1..m.

Note que a qualidade da razão de aproximação melhora (pequenos valores de d) para aquelas instâncias onde o valor da solução ótima d_{opt} é maior. Note ainda que, como não conhecemos d_{opt} e como $d_y \leq d_{opt}$, ao fazermos $d = D(3ln(m/e)/d_y)^{1/2}$ obtemos um algoritmo randômico (1+d)-aproximado, com probabilidade de sucesso maior ou igual a 1-e. Equivalentemente, $\Pr(B)=\Pr(d_H > (1+d)d_{opt}) \leq e < 1$. Maiores detalhes sobre este assunto podem ser encontrados em [BLPR97].

6 DERANDOMIZAÇÃO

Vejamos agora como construir uma solução determinística $s \in \Sigma^n$ satisfazendo a mesma razão de aproximação obtida na Seção 5.1. Entretanto, antes de desenvolvermos um algoritmo *derandomizado* pelo método das probabilidades condicionais, considere a seguinte notação auxiliar (definida a partir de (1)-(4)). Representaremos por $x_j = (x_{js_1}, x_{js_2}, ..., x_{js_p})$ onde $j \in \{1,...,n\}$ e $|\Sigma|=p$, o vetor de variáveis binárias associado à j-ésima posição de uma seqüência $s \in \Sigma^n$. Se todas as componentes de x_j já forem conhecidas teremos então $\hat{x}_j = (\hat{x}_{js_1}, \hat{x}_{js_2}, ..., \hat{x}_{js_p})$. Note neste caso, que cada vetor $\hat{x}_j$ irá representar uma seqüência de valores onde apenas uma coordenada é 1 e as demais são 0. Para construção de uma solução determinística através da técnica de *derandomização* (como na Seção 4), é fundamental que se calcule as probabilidades condicionais $\Pr(B|A_k(x))\ \forall x \in \Sigma$, onde $A_k(x) \equiv \left(\hat{x}_1, \hat{x}_2, ..., \hat{x}_{k-1}, x_{kx} = 1\ e\ x_{ks} = 0\ \forall\ s \neq x\right)$ e k=1,..,n. Dessa forma, o caracter $x \in \Sigma$ escolhido na k-ésima etapa deverá ser tal que:

$$\Pr\left(B \middle| \hat{x}_1,\ldots,\hat{x}_{k-1}\right) \geq \min_{x\in\Sigma}\left\{\Pr\left(B \middle| A_k(x)\right)\right\} = \Pr\left(B \middle| \hat{x}_1,\ldots,\hat{x}_{k-1},\hat{x}_k\right) \qquad (7)$$

O processo deverá ser repetido até que tenhamos uma solução determinística $\hat{x} = (\hat{x}_1,...,\hat{x}_n)$ onde $\Pr(B|\hat{x}_1,...,\hat{x}_n) = 0$.

Vejamos agora o seguinte modelo de Programação Linear PL(k,x) auxiliar associado a um índice $k \in \{1,..,n\}$ e um símbolo $\xi \in \Sigma$:

$$\bar{d}_{k,x} = \min d \qquad (8)$$

s.a

$$\sum_{s\in\Sigma} x_{js} = 1 \quad , j = k,..,n \qquad (9)$$

$$\sum_{j=k}^{n}\sum_{s\in\Sigma} x_{js}\, d_1(\boldsymbol{s}, s_i[j]) \leq d - \hat{d}_{i,k-1} \quad , i = 1,..,m \qquad (10)$$

$$x_{kx} = 1 \quad , para \;\; a\,\text{lg}\,um\, \boldsymbol{x} \in \Sigma \qquad (11)$$

$$x_{ks} = 0 \quad , \forall \;\; \boldsymbol{s} \neq \boldsymbol{x} \qquad (12)$$

$$x_{js} \in [0,1] \quad , j = k,..,n \;\; e \;\; \boldsymbol{s} \in \Sigma \qquad (13)$$

onde: $\hat{d}_{i,0} = 0$ e $\hat{d}_{i,k-1} = \sum_{j=1}^{k-1}\sum_{s\in\Sigma}\hat{x}_{js}\, d_1(\boldsymbol{s}, s_i[j])$, para i=1,...,m e k=1,..,n.

Este modelo é bastante semelhante àquele apresentado em (1)-(4). Neste caso entretanto, deve ser observado que as constantes $\hat{d}_{i,k-1}$ são obtidas em função das atribuições dadas aos k-1 primeiros caracteres de uma solução $s \in \Sigma^n$. As restrições (11) e (12) asseguram que somente o símbolo $x \in \Sigma$ será selecionado na k-ésima posição s. Note ainda que resolvendo-se a relaxação PL(k,ξ) teremos uma solução do tipo $(\hat{x}_1,...,\hat{x}_{k-1},\hat{x}_k,\bar{x}_{k+1},...,\bar{x}_n)$ onde $\hat{x}_{kx} = 1$ e $\hat{x}_{ks} = 0$, para todo $s \neq \xi$. Além disso, para cada índice $k \in \{1,...,n\}$ considere p problemas de programação linear, ou seja, faremos $x_{kx} = 1$ para cada símbolo ξ em Σ.

Agora, sejam dados os valores $\hat{x}_{js}$ para j=1,..,k-1 e $\hat{x}_{kx} = 1$ para algum $x \in \Sigma$. Representaremos por:

$$X_{i,k+1} = \sum_{j=k+1}^{n} d_1(s[j], s_i[j]) \quad \text{para cada } i=1,...,m \text{ e } k=1,..,n\text{-}1,$$

as variáveis aleatórias associadas a uma solução heurística $s|P \in \Sigma^{n-k}$ (onde P={k+1,..,n}) obtida randômicamente através da solução relaxada de PL(k,x). Considere então o seguinte resultado preliminar:

Lema 4: $E[X_{i,k+1}] \leq \bar{d}_{k,x} - \hat{d}_{i,k-1} - d_1(\boldsymbol{x}, s_i[k])$, $\forall$ i=1,...,m; $k \in \{1,...,n\text{-}1\}$ e $\xi \in \Sigma$.

Prova: Como $X_{i,k+1} = \sum_{j=k+1}^{n} d_1(s[j], s_i[j])$, temos que:

$$E[X_{i,k+1}] = E\left[\sum_{j=k+1}^{n} d_1(s[j], s_i[j])\right]$$

Das desigualdades (10)-(13) e da linearidade do valor esperado temos:

$$E[X_{i,k+1}] = \sum_{j=k+1}^{n}\sum_{s\in\Sigma}\Pr(s[j] = \boldsymbol{s})\, d_1(\boldsymbol{s}, s_i[j]) = \sum_{j=k+1}^{n}\sum_{s\in\Sigma}\bar{x}_{j,s}\, d_1(\boldsymbol{s}, s_i[j]) \leq \bar{d}_{kx} - \hat{d}_{i,k-1} - d_1(\boldsymbol{x}, s_i[k]). \quad \square$$

Conforme discutido anteriormente, para aplicação do método das probabilidades condicionais é fundamental que se garanta inicialmente Pr(B) < 1, onde B representa um *mau* evento. Assim, se 1+δ é razão de aproximação para o PSMP (onde $\delta = D(3ln(m/e)/d_y)^{1/2}$), um *mau* evento associado ao Algoritmo 1 ocorre, sempre que $B \equiv (d_H > (1+\delta)d_{opt})$, sendo d_H variável aleatória representando o valor da solução heurística s_H (Seção 5.1).

Sem perda de generalidade, poderemos assumir que um *mau* evento B ocorre sempre que $B \equiv (d_H > (1+\delta)d_y)$ onde $d_y \leq d_{opt}$ é o valor da relaxação linear de (1)-(4). Seguindo-se o mesmo raciocínio desenvolvido na Seção 5.1, obtemos um algoritmo (1+δ)-aproximado para o PSMP fazendo-se $\delta = D(3ln(m/e)/d_y)^{1/2}$, ou seja, teremos Pr($B$)=Pr($d_H > (1+\delta)d_y$)<1.

Observe que um *mau* evento (B |$A_k(\xi)$) ocorre para $k \in \{1,..,n\}$ e $\xi \in \Sigma$, se e somente se, (B_i |$A_k(\xi)$) ocorre para pelo menos um índice $i \in \{1,..,m\}$ onde:

$$\left(B_i \middle| A_k(\boldsymbol{x})\right) \equiv \left(\hat{d}_{i,k-1} + d_1(\boldsymbol{x}, s_i[k]) + X_{i,k+1} > (1+\boldsymbol{\delta})\bar{d}_{k,x}\right) \equiv \left(X_{i,k+1} > (1+\boldsymbol{\delta})\bar{d}_{k,x} - \hat{d}_{i,k-1} - d_1(\boldsymbol{x}, s_i[k])\right)$$

Temos então que:

$$\Pr\left(B_i \middle| A_k(\boldsymbol{x})\right) = \Pr\left(X_{i,k+1} > (1+\boldsymbol{\delta})\bar{d}_{k,x} - \hat{d}_{i,k-1} - d_1(\boldsymbol{x}, s_i[k])\right), \text{ p/ } i=1,...,m \text{ e } k \in \{1,...,n\}.$$

Logo do Lema 4, temos para i=1,...,m:

$$\Pr\left(B_i \middle| A_k(\boldsymbol{x})\right) = \Pr\left(X_{i,k+1} > \bar{d}_{k,x} - \hat{d}_{i,k-1} - d_1(\boldsymbol{x}, s_i[k]) + \boldsymbol{\delta}\,\bar{d}_{k,x}\right) \leq \Pr\left(X_{i,k+1} > E[X_{i,k+1}] + \boldsymbol{\delta}\,\bar{d}_{k,x}\right)$$

Note que as probabilidades condicionais acima não são determinadas explicitamente. Entretanto, limites superiores podem ser obtidos utilizando-se, por exemplo, a desigualdade de *Chernoff-Hoeffding* como descrita no Lema 2. Contudo, neste caso, observe que as expectâncias condicionais não são conhecidas explicitamente, significando portanto que os estimadores pessimistas não podem ser determinados diretamente. Para contornar esse problema, consideraremos inicialmente o caso binário onde $d_1(a,b) \in \{0,1\}, \forall a,b \in \Sigma$ (distância de Hamming). Neste caso, teremos:

$$X_{i,k+1} = \sum_{j=k+1}^{n} d_1(s[j], s_i[j]) = \sum_{j=k+1}^{n} Y_j \qquad (15)$$

Note que $X_{i,k+1}$ representa uma soma de variáveis aleatórias independentes $Y_j \in \{0,1\}$ para $j=k+1,..,n$. Assim Y_j = 1 sempre que $s[j] \neq s_i[j]$ e, Y_j = 0 caso contrário. Portanto, da desigualdade de *Chernoff-Hoeffding* conforme descrito no Lema 1, temos:

$$\Pr\left(B_i \middle| A_k(\boldsymbol{x})\right) = \Pr\left(X_{i,k+1} > E[X_{i,k+1}] + \delta\,\bar{d}_{k,x}\right) \leq \exp\left(-\frac{1}{3}\bar{d}_{k,x}\delta^2\right) \leq \frac{e}{m}, \quad \text{para } i=1,...,m.$$

Segue então que:

$$\Pr\left(B \middle| A_k(\boldsymbol{x})\right) = \Pr\left(\bigcup_{i=1}^{m} B_i \middle| A_k(\boldsymbol{x})\right) \leq \sum_{i=1}^{m} \Pr\left(B_i \middle| A_k(\boldsymbol{x})\right) \leq \frac{m}{\exp\left(\bar{d}_{k,x}\boldsymbol{\delta}^2/3\right)} < \boldsymbol{e} < 1$$

onde, $\delta = D\sqrt{3\ln(m/e)/d_y}$. (16)

Logo, um estimador pessimista U:$[0,1]^{pn} \rightarrow [0,1)$ pode ser obtido diretamente para as probabilidades condicionais fazendo-se simplesmente:

$$U(A_k(x)) = U\left(\hat{x}_1,...,\hat{x}_{k-1}, x_{kx}=1 \; e \; x_{ks}=0 \; \forall \; s \neq x\right) = \frac{m}{\exp\left(\bar{d}_{k,x}\delta^2/3\right)}, \; \forall \; k \in \{1,...,n\} \text{ e } x \in \Sigma.$$

Da análise de aproximação da Seção 5.1, é fácil ver que: $U(\bar{x}_1,...,\bar{x}_n) = m/\exp\left(d_y\delta^2/3\right) \leq e < 1$ e, portanto, a condição 1 da definição de estimador pessimista é satisfeita (Seção 4).

A condição 2 também pode ser verificada diretamente pois:

$$\Pr\left(B \middle| A_k(x)\right) \leq U(A_k(x)), \; \forall \; k \in \{1,...,n\} \text{ e } x \in \Sigma.$$

Finalmente, devemos provar que:

$$U\left(\hat{x}_1,...,\hat{x}_{k-1},\bar{x}_k,...,\bar{x}_n\right) \geq \min_{x \in \Sigma}\{U(A_k(x))\}, \text{ onde } k \in \{1,...,n\}.$$

De fato, note que:

$$U\left(\hat{x}_1,...,\hat{x}_{k-1},\bar{x}_k,...,\bar{x}_n\right) = U\left(A_{k-1}(x), \; para \; a \lg um \; x \in \Sigma\right) = \frac{m}{\exp\left(\bar{d}_{k-1,x}\delta^2/3\right)}$$

Como $\bar{d}_{k-1,x} \leq \bar{d}_{k,x} \; \forall x \in \Sigma$, temos em particular que $\bar{d}_{k-1,x} \leq \max\{\bar{d}_{k,x} \; \forall x \in \Sigma\} = \bar{d}_{k,\bar{x}}$. Segue então que:

$$U\left(\hat{x}_1,...,\hat{x}_{k-1},\bar{x}_k,...,\bar{x}_n\right) = \frac{m}{\exp\left(\bar{d}_{k-1,x}\delta^2/3\right)} \geq \min\{U(A_k(x)), \forall x \in \Sigma\} = \frac{m}{\exp\left(\bar{d}_{k,\bar{x}}\delta^2/3\right)}.$$

Observe portanto que a função $U:[0,1]^{pn} \rightarrow [0,1)$ define um estimador pessimista para nossas probabilidades condicionais. Observe ainda que, dado $k \in \{1,..,n\}$, para calcular $min\{U(A_k(\xi), \forall x \in \Sigma\}$ basta calcularmos $max\{\bar{d}_{k,x} \; \forall x \in \Sigma\}$, ou seja, resolvemos $|\Sigma|$ problemas de Programação Linear a cada passo. Logo, temos o seguinte algoritmo de *derandomização* sintetizado a seguir.

Entrada: $s_1, s_2, ..., s_m \in \Sigma^n$
Saída: seqüência $s_H \in \Sigma^n$ e a distância d_H.
Início
  Para $k=1,...,n$ **faça**
    1. **Para cada** $x \in \Sigma$ **faça**
       a. $\hat{x}_{kx} = 1$
       b. $\hat{x}_{ks} = 0$, para todo $\sigma \neq x$
       c. Resolver RL(k,ξ) e retornar $\bar{d}_{k,x}$
       fim para
    2. Retorna $\bar{x}$ tal que $d_{k,\bar{x}} = \max_{x \in \Sigma}\{\bar{d}_{kx} : x \in \Sigma\}$
    3. $s_H[k] \leftarrow \bar{x}$
  fim para
Fim

Algoritmo 2: Algoritmo Determinístico (*Derandomizado*)

Como cada problema de programação linear tem complexidade $O(n^3L)$ o algoritmo acima terá complexidade total igual a $O(n^4L|\Sigma|)$ onde L representa o número total de *bits* utilizado na entrada do problema de programação linear [Wrig97].

Observe agora que, no caso geral onde $d_1(a,b) \in [0,D]$, $a,b \in \Sigma$ e $D \in Z^+$, não poderemos considerar o Lema 1 utilizado para construção do estimador pessimista $U:[0,1]^{pn} \rightarrow [0,1)$, como descrito acima. Note que, nas hipóteses do Lema 1, deveremos considerar apenas variáveis 0-1, aleatórias e independentes entre si. O caso geral pode ser convertido em binário utilizando-se as variáveis $x_{j,s} \in \{0,1\}$, para $j=1,...,n$ e $s \in \Sigma$. Entretanto, as variáveis $x_{j,s}$ para j fixo não são independentes entre si, o que inviabiliza, neste caso, a aplicação do Lema 1.

Outra alternativa seria a utilização do Lema 2, onde temos:

$$\Pr(X_i \geq (1+\delta)E(X_i)) \leq \exp\left(-E(X_i)\delta^2/3D^2\right)$$

onde $X_i = \sum_{j=1}^{n} X_{ij}$ e $X_{ij} = d_1(s_H[j], s_i[j])$, para $j=1,...,n$.

Embora tenhamos X_{ij} independentes entre si (onde $X_{ij} \in [0,D]$) não conhecemos explicitamente $E[X_i]$ e não obtemos portanto um estimador pessimista para o PMSP (o Lema 4 não se aplica neste caso). Como trabalho futuro uma outra alternativa é pesquisar o caso geral utilizando-se outras desigualdades (majorações) distintas daquelas apresentadas por *Chernoff-Hoeffding*.

7 CONCLUSÕES

Neste trabalho, fazemos inicialmente uma descrição das técnicas de arredondamento randômico e *derandomização* e estudamos o método das probabilidades condicionais aplicado ao PSMP. Mostramos como implementar a *derandomização* sugerida por Ben-Dor *et. al* [BLPR97]. Utilizamos o método dos estimadores pessimistas determinando limitantes superiores para as probabilidades condicionais associadas. Esta abordagem permitiu a construção de um novo algoritmo puramente determinístico para o problema com a mesma razão de performance obtida em [BLPR97].

Como sugestão para trabalhos futuros para o PSMP, podemos citar a determinação de estimadores pessimistas para instâncias com distâncias (entre caracteres) no intervalo $[0,D]$ onde $D=max_{a,b\in\Sigma}\{d_1(a,b)\}$. Neste caso, outras desigualdades (*tail inequalities*) deverão ser utilizadas visando-se a determinação de limitantes superiores para as probabilidades condicionais associadas.

Outra possibilidade, é trabalhar na construção de algoritmos *derandomizados* de melhor qualidade (com a mesma razão de performance) utilizando-se, por exemplo, o algoritmo randômico $(4/3+\varepsilon)$-aproximado de Lanctot *et al.* [LLMWZ99] ou o Esquema de Aproximação Randômico Polinomial proposto por Li *et. al.* [LMW02].

8 REFERÊNCIAS

[AS92] Alon, N. and Spencer, J.; The Probabilistic Method. *Wiley, New York*, 1992.

[BLPR97] Ben-Dor, A., Lancia, G., Perone, J. and Ravi, R.; Banishing Bias from Consensus Sequences, *Combinatorial Pattern Matching, 8th Annual Symposium, Springer-Verlag, Berlin*, 1997.

[BGHMS97] Berman, P., Gumucio, D., Hardison, R., Miler, W. and Stojanovic, N.; A linear-time algorithm for the 1-mismatch problem, *Workshops on Algorithms and Data Structures,* pp. 126-135, 1997.

[FL97] Frances, M. and Litman, A.; On Covering Problems of Codes. *Theory of Computing Systems*, vol. 30, pp. 113-119, 1997.

[GJL99] Gasieniec, L., Jansson, J. and Lingas, A.; Efficient approximation algorithms for the Hamming center problem, *Proc. 10th ACM-SIAM Symposium on Discrete Algorithms*, pp. S905-S906, 1999.

[HS95] Hertz, G. and Stormo, G.; Identification of Consensus Patterns in Un-aligned DNA and Protein Sequences: A Large-Deviation Statistical Basis for Penalizing Gaps, *in Proceedings of the 3rd International Conference on Bioinformatics and Genome Research*, pp. 201-216, 1995.

[LLMWZ99] Lanctot, K., Li, M., Ma, B., Wang, S. and Zhang, L.; Distinguishing string selection problems. *Proc. 10th ACM-SIAM Symp. On Discrete Algorithms*, pp. 633-642, 1999.

[LR90] Lawrence, C. and Reilly, A.; An expectation maximization (EM) algorithm for the identification and characterization of common sites in unaligned biopolymer sequences. *Proteins*, 7, pp. 41-51, 1990.

[LMW02] Li, M., Ma, B. and Wang, L.; On the closest string and substring problems. *Journal of the ACM*, 49 (2): pp. 157-171, 2002.

[LBMM91] Lucas, K., Busch, M., Mossinger, S. and Thompson, J.A.; An improved microcomputer program for finding gene or gene family-specific Abd-Elsalam 95 oligonucleotides suitable as primers for polymerase chain reactions or as probes. *Comput. Appl. Biosci.* 7: pp. 525-529, 1991.

[Mey83] Meyer, P.; Probabilidade: Aplicações à Estatística. (2ª edição) *Livros Técnicos e Científicos Editora S.A*, 1983.

[MR95] Motwani, R. and Raghavan, P.; Randomized Algorithms, *Cambrigde Univ. Press*, 1995.

[PMLO04] Pardalos, P. M., Meneses, C. N., Lu, Z., Oliveira, C. A. S.; Optimal Solutions for the Closest String Problem via Integer Programming. *To appear in INFORMS Journal on Computing*, 2004.

[PBPR89] Posfai, J., Bhagwat, A. S., Posfai, G. and Roberts, R. J.; Predictive motifs derived from cytosine methyltransferases. *Nucleic Acids Res* 17 (7), pp. 2421-2435, 1989.

[RT87] Raghavan, P., Thompson, C. D.; Randomized Rounding: Provably good algorithms and algorithmics proofs. *Combinatorica* 7, pp. 365-374, 1987.

[Ragh88] Raghavan, P.; A probabilistic construction of deterministic algorithms: Approximating packing integer programs. *Journal of Computer and System Sciences*, 37: pp. 130-143, 1988.

[Stor90] Stormo, G. D.; Consensus patterns in DNA. InDoolittle, R. F., ed., Molecular Evolution: Computer Analysis of Protein and Nucleic Acid Sequences, Methods inEnzymology, volume 183. *Academic Press*. pp. 211-221, 1990.

[SH91] Stormo, G. D. and Hartzell, G. W.; Identifying protein-binding sites from unaligned DNA fragments, *Proc. Natl. Acad. Sci. USA*, 88 : pp. 5699–5703, 1991.

[WP84] Waterman, M. S. and Perlwitz, M. D.; Line Geometries for Sequence Comparisons. *Bull Math Biol*;46 (4): pp. 567-577, 1984.

[WAG84] Waterman, M. S., Arratia, R. and Galas, D. J.; Pattern Recognition in Several Sequences: Consensus and Alignment. *Bull. Math. Biol*. 46, 515-527, 1984.

[WG86] Waterman, M. S. and Griggs, J. R.; Interval graphs and maps of DNA. *Bulletin of Mathematical Biology*, 48(2): pp. 189-195, 1986.

[Wrig97] Wright, S. J.; Primal-Dual Interior-Point Methods. *SIAM – Society for Industrial and Applied Mathematics*, 1997.