

**FACULTY  
OF MATHEMATICS  
AND PHYSICS**  
Charles University

**MASTER THESIS**

Bc. et Bc. Jiří Škrobánek

# **Dyck Edit Distance of Random Strings**

Computer Science Institute of Charles University

Supervisor of the master thesis: Prof. Mgr. Michal Koucký, Ph.D.

Study programme: Theoretical computer science

Prague 2024

I declare that I carried out this master thesis on my own, and only with the cited sources, literature and other professional sources. I understand that my work relates to the rights and obligations under the Act No. 121/2000 Sb., the Copyright Act, as amended, in particular the fact that the Charles University has the right to conclude a license agreement on the use of this work as a school work pursuant to Section 60 subsection 1 of the Copyright Act.

In Prague, July 17th 2024

Jiří Škrobánek

I would like to express my thanks to the supervisor of this thesis professor Michal Koucký for guiding me and providing feedback, despite the substantial geographical distance.

Title: Dyck Edit Distance of Random Strings

Author: Bc. et Bc. Jiří Škrobánek

Institute: Computer Science Institute of Charles University

Supervisor: Prof. Mgr. Michal Koucký, Ph.D., Computer Science Institute of Charles University

Abstract: Dyck languages consist of sequences of opening and closing parentheses of different types which are well-parenthesized. Dyck edit distance problem measures the distance of a string from a Dyck language by counting the number of edits (insertions or deletions of individual characters) required to make it well-parenthesized. In this thesis we study the expected properties of Dyck edit distance for a uniformly selected string. We show the existence of asymptotic properties of Dyck edit distance and establish both lower and upper bounds for Dyck alphabets with different number of symbols.

Keywords: Dyck edit distance, Parenthesized expressions, Random strings

Název práce: Dyckova vzdálenost náhodných řetězců

Autor: Bc. et Bc. Jiří Škrobánek

Ústav: Informatický ústav Univerzity Karlovy

Vedoucí diplomové práce: Prof. Mgr. Michal Koucký, Ph.D., Informatický ústav Univerzity Karlovy

Abstrakt: Dyckovy jazyky se skládají z dobře uzávorkovaných posloupností otevíracích a zavíracích závorek různých typů. Dyckova editační vzdálenost měří vzdálenost daného řetězce od Dyckova jazyka nutným počtem změn jednotlivých znaků (vlození či odstranění), aby byl řetězec dobře uzávorkovaný. Tato práce se zaměřuje na očekávané vlastnosti Dyckovy editační vzdálenosti pro uniformně zvolený řetězec závorek. Ukážeme existenci asymptotických vlastností Dyckovy editační vzdálenosti a představíme jak horní, tak dolní odhady pro Dyckovy abecedy s různým počtem symbolů.

Klíčová slova: Dyckova vzdálenost, uzávorkované výrazy, náhodné řetězce

# Contents

<b>Introduction</b>	<b>6</b>
<b>1 Preliminaries</b>	<b>8</b>
1.1 Definitions . . . . .	8
1.2 Comparing edit distance and Dyck edit distance . . . . .	10
1.3 Calculating Dyck edit distance . . . . .	11
<b>2 Properties of Dyck ratios</b>	<b>13</b>
2.1 Existence and Concentration . . . . .	13
2.2 Upper bounds on Dyck ratio . . . . .	15
<b>3 Distance of Pairs in Matchings</b>	<b>19</b>
3.1 Necessary optimal matching distance of a Dyck string . . . . .	19
3.2 Limited maximum distance for matches . . . . .	22
<b>4 Kolmogorov Lower Bound on Dyck Ratio</b>	<b>23</b>
<b>5 Single Pair of Symbols</b>	<b>26</b>
<b>6 Further Research</b>	<b>30</b>
<b>Bibliography</b>	<b>31</b>
<b>List of Tables</b>	<b>33</b>

# Introduction

The edit distance problem asks the question of how many changes (insertions and deletions of characters) are required to convert one string to another over the same alphabet. The expected properties of edit distance for uniformly selected strings were extensively studied. The initial investigation of Chvátal and Sankoff [1] from 1975 provided some bounds, showed some averages of observed values of edit distance, and established a few basic properties of mean edit distance. In particular they proved that for an alphabet with  $k$  characters the expected length of the longest common subsequence normalized by length of the string converges to some constant  $\gamma_k$  as the length approaches infinity, which directly implies that also edit distance normalized by length converges to some constant.

In some sense we are aiming to obtain similar or equivalent results for the problem of Dyck edit distance in this thesis, as expected properties of Dyck edit distance for uniformly selected string remain mostly unexplored.

Dyck alphabet is a set of pairs of parentheses (each pair consisting of opening and closing character). Dyck language over that alphabet includes the sequences of parentheses which are well-parenthesized. The problem of Dyck edit distance asks the question how far a given string is from being in the Dyck language in terms of number of required insertions and deletions to transform it to a string which is in the Dyck language. (We provide formal definitions later.)

What expected properties of Dyck edit distance are is a natural question to ask. Problems in many settings turn out to involve Dyck languages – parsing many programming and XML-like languages or arithmetic expressions. Dyck edit distance is also related to the problem of RNA folding where opening and closing characters can match even in the opposite order.

Lot of effort has been invested into finding values of  $\gamma_k$ s for edit distance. Yet the precise values remain unknown. Sankoff and Mainville [2] conjectured that  $\gamma_k\sqrt{k} \rightarrow 2$  as  $k$  goes to infinity. Kiwi, Loeb, and Matoušek [3] proved the conjecture in 2004.

Dančík in his doctoral thesis [4] gave both new lower and upper bounds. He defined finite state automata and analyzed their behavior on random strings using Markov chains, getting new lower bounds on  $\gamma_k$ s. For upper bounds so called collations (sequences with matched pairs) and limiting their number by combinatorial analysis were used.

In their 1999 article Baeze-Yates et al. [5] used Markov chains to get upper bounds on edit distance and also used methods based on Kolmogorov complexity to get improved lower bounds on edit distance.

It turns out that some of the methods (mentioned above) that were successfully applied to bound  $\gamma_k$  can also be applied in the area of Dyck edit distance. For example, we try to utilize Kolmogorov complexity to get lower bound on Dyck edit distance. However, many other methods do not seem to be (easily) transferable. Especially approaches that involve studying properties of matchings in bipartite graphs cannot be utilized, since only one string is given on input in the problem of Dyck edit distance, not two.

The structure of the thesis is as follows: In Chapter 1 we formally introduce the required terminology. In Chapter 2 we prove several important asymptotic

# of pairs in alphabet	Observed ratio	Lower bound	Upper bound
2	0.2262	0.0948	0.5115
3	0.3235	0.1700	0.6458
4	0.3875	0.2270	0.7148
5	0.4350	0.2723	0.7469
6	0.4711	0.3094	0.7642
10	0.5636	0.4115	0.8067
100	0.8395	0.7613	0.9489

**Table 1** Overview of our bounds on Dyck edit distance and empirical values

properties about Dyck edit distance and its relation to ordinary edit distance. In particular we show existence of asymptotic constants equivalent to  $\gamma_k$ . In Chapter 3 we explore how Dyck edit distance depends on the maximum allowed distance between characters that can be matched together. In Chapter 4 we use methods based on Kolmogorov complexity to get asymptotic lower bounds. In Chapter 5 we explore the case of Dyck alphabets with only one pair of symbols, where the properties are very different and expected Dyck edit distance is not linear in length of the string. We conclude by suggesting areas that could be investigated in the future.

In Table 1 we show a quick summary of some of the bounds we derived. We measure Dyck edit distance asymptotically by looking at the ratio of count of characters that need to be deleted in order to make the string well-parenthesized to its length. (See Definition 1.1.7) Upper bounds were obtained by the method of Kolmogorov complexity. Lower bound for alphabet with two, three, and four types of parentheses was obtained by the method of Theorem 8. Lower bound for alphabet with one hundred types of parentheses was obtained from Theorem 9. Remaining lower bounds come from Theorem 7.

# 1 Preliminaries

In this chapter we formally define Dyck edit distance and put it in context of classic string edit distance (LCS distance).

## 1.1 Definitions

To begin, we clarify basic terminology of strings. An *alphabet* is just a finite set of *characters* or *symbols*. A *string* is a finite sequence of characters of an alphabet. When we talk about mean properties of strings of some length over an alphabet, we implicitly assume uniform distribution over those strings. We will use capital Greek letters for alphabets, small Greek letters for strings and small Latin letters for characters (symbols).

We will use the following definition of string edit distance:

*Definition 1.1.1* (Edit distance). Let  $\sigma, \chi$  be two strings over the same alphabet. We define *edit distance*  $ed(\sigma, \chi)$  of this pair of strings to be the minimum number of character insertions and deletions required to reach  $\chi$  starting from  $\sigma$ .

We also call edit distance *LCS distance* due to its relation to the length of the longest common subsequence of the two strings  $\sigma, \chi$ .

We should remark here that substitutions of characters are often also allowed along with insertions and deletions. Such edit distance is called Levenshtein distance [6]. We opted not to consider substitutions in this thesis for both ordinary edit distance and Dyck edit distance however.

*Definition 1.1.2* (Maximum (non-crossing) matching between strings). Let  $\sigma, \chi$  two strings over the same alphabet. We say that a set of ordered pairs of indices is a (non-crossing) matching between  $\sigma, \chi$  if

- Indices of  $\sigma$  are used as the first elements of the pairs and each at most once.
- Indices of  $\chi$  are used as the second elements of the pairs and each at most once.
- Pairs are *non-crossing* i.e. for any pair  $(i, j)$  and another  $(k, l)$ , it must hold that  $i < k$  implies  $j < l$  and vice versa.
- For all pairs  $(i, j)$  in the matching the character in position  $i$  in  $\sigma$  and the character in position  $j$  in  $\chi$  must be equal.

We call a matching *maximum* between  $\sigma, \chi$  if it is of maximum size among matchings between  $\sigma, \chi$ . The size of these maximum matchings we denote as  $mm(\sigma, \chi)$ .

It can be seen that the length of the maximum non-crossing matching is also the length of the longest common subsequence of the same strings. The following observation establishes the relationship between edit distance and length of longest common subsequence (LCS):

*Observation 1.* For any two strings  $\sigma, \chi$  of lengths  $a, b$  over the same alphabet

$$mm(\sigma, \chi) = \frac{a + b - ed(\sigma, \chi)}{2}.$$

*Definition 1.1.3.* We say that  $\Delta$  is a Dyck alphabet if it is a finite set of characters that are grouped into ordered pairs (types, or kinds), i.e., the first character of every pair is opening and second is closing character. We say that any string  $\delta$  over a Dyck alphabet  $\Delta$  is a Dyck string.

*Definition 1.1.4* (Dyck edit distance). Let  $\delta$  be a Dyck string over  $\Delta$ . Let  $\chi$  be a well-parenthesized Dyck string over  $\Delta$  with the lowest edit distance from  $\delta$ . We say that  $Ded(\delta) = ed(\delta, \chi)$  is the *Dyck edit distance* of  $\delta$ .

There is often more than one possible choice of  $\chi$  in the previous definition. Imagine Dyck alphabet consisting of opening symbols  $(_1, \dots, (_n$  and closing symbols  $)_1, \dots, )_n$ . Where for all  $i$  the characters  $(_i$  and  $)_i$  form a pair. For example, the string  $(_1(_2 \dots (_n)_1)_2 \dots )_n$  has Dyck edit distance  $2n - 2$ . It may be optimally balanced by deleting all but one pair of parentheses. Another set of options is adding  $2n - 2$  parentheses. This may also be done in a variety of ways, e.g.,  $(_1)_1(_2)_2 \dots (_n)_1)_1)_2 \dots )_n$  or  $(_1(_2 \dots (_{n-1})_{n-1})_2)_1)_n)_1 \dots (_2)_1)_2 \dots )_n$ .

This extra freedom in finding how to optimally balance the string, on top of choosing the individual symbols that should be deleted or added, is also present in the problem of finding the edit distance of two strings.

Matchings for the purposes of Dyck edit distance are quite different as the next definition introduces:

*Definition 1.1.5* (Maximum matching (in a Dyck string)). Let  $\delta$  be a Dyck string over  $\Delta$ . A set of pairs of indices is a *Dyck matching* if all of the following is true:

- Each index is used at most once.
- For any pair  $(i, j)$  it must hold  $i < j$ .
- Pairs are *non-crossing* i.e. for any pair  $(i, j)$  and another  $(k, l)$ , it must hold that  $i < k$  implies  $j > l$  and vice versa.
- Every character at the first index in any pair is opening, the second must be closing, and both must be of the same kind.

We call a matching of  $\delta$  maximum if it is the maximum among  $\delta$ 's matchings.

*Observation 2.* Length of any Dyck  $\delta$  string is equal to its Dyck edit distance plus twice the size of its maximum matching. Or stated precisely

$$mm(\delta) = \frac{|\delta| - Ded(\delta)}{2}.$$

*Observation 3.* The string  $\chi$  in definition 1.1.4 may be obtained from  $\delta$  just by deleting symbols that are not part of a selected maximum matching.

We define Dyck ratio as a convenient measure of normalized Dyck edit distance alike  $\gamma_k$  in case of LCS distance.

*Definition 1.1.6* (Dyck ratio of a string). Let  $\delta$  be a Dyck string of length  $\ell$ . We define  $dr(\delta) = Ded(\delta)/\ell$  to be the *Dyck ratio* of  $\delta$ .

*Definition 1.1.7* (Asymptotic Dyck ratio). Let  $\Delta$  be a Dyck alphabet. We define Dyck ratio of  $\Delta$  for length  $\ell$  to be

$$dr(\Delta, \ell) = \mathbb{E}_{\delta \in \Delta^\ell} [dr(\delta)] = \frac{\sum_{\delta \in \Delta^\ell} dr(\delta)}{|\Delta^\ell|}.$$

And we define the *asymptotic Dyck ratio* of  $\Delta$  to be

$$dr(\Delta, \infty) = \lim_{\ell \rightarrow \infty} dr(\Delta, \ell),$$

if it exists.

We will show that asymptotic Dyck ratio always exists in the next chapter.

## 1.2 Comparing edit distance and Dyck edit distance

We show that it is possible to reduce the edit distance problem to Dyck edit distance problem in linear time, suggesting that Dyck edit distance is a more complex problem. Reduction in the opposite direction is not so quickly obtainable, we use approximate reduction to prove Theorem 7.

*Theorem 1.* Edit distance problem is reducible to Dyck edit distance problem in linear time.

*Proof.* Let  $\Gamma$  be any alphabet and  $\gamma_1, \gamma_2 \in \Gamma^*$ . We define Dyck alphabet  $\Delta_\Gamma$  so that for every character  $g \in \Gamma$  there is a pair of characters  $(g$  and  $)_g$  in  $\Delta_\Gamma$ . For the problem  $ed(\gamma_1, \gamma_2)$  we transform the input in the following way: For all characters  $g$  in  $\gamma_1$  we replace them by  $(g$ , obtaining  $\delta_1$ . Conversely for all characters  $g$  in  $\gamma_2$  we replace them by  $)_g$  and then we reverse that string, obtaining  $\delta_2$ . We then concatenate  $\delta_1$  and  $\delta_2$  to obtain  $\delta$ . Now we claim  $ed(\gamma_1, \gamma_2) = Ded(\delta)$ .

It can be seen that this transformation of  $\gamma_1, \gamma_2$  to  $\delta$  can be done in linear time. So it remains to show  $ed(\gamma_1, \gamma_2) = Ded(\delta)$ .

Let us denote some maximum matching of  $\gamma_1$  and  $\gamma_2$  as  $M$ . We can observe that the transformation  $t : (i, j) \mapsto (i, |\gamma_1| + |\gamma_2| - j - 1)$  yields a set of pairs of indices that constitute a valid Dyck matching in  $\delta$ . (This can be easily verified from definitions of matchings and the transformation used to obtain  $\delta$ .) So there is a (Dyck) matching of the same size as  $M$  in  $\delta$ .

Let us have any Dyck matching  $N$  of  $\delta$ . Apparently, in  $N$  all pairs of matches cross the point where the two original strings were concatenated, simply because there are only opening characters on one side and closing characters on the other. Now we can notice that  $t$  also transforms  $N$  into a matching between  $\gamma_1$  and  $\gamma_2$ .

So for any matching in  $\delta$  there is matching of same size between  $\gamma_1$  and  $\gamma_2$ . We have therefore established that the size of maximum matching between  $\gamma_1$  and  $\gamma_2$  is the same as size of maximum matching in  $\delta$ .

Now by the observations above  $ed(\gamma_1, \gamma_2) = |\gamma_1| + |\gamma_2| - 2mm(\gamma_1, \gamma_2) = |\delta| - 2mm(\delta) = Ded(\delta)$ , which concludes the proof.  $\square$

## 1.3 Calculating Dyck edit distance

To obtain some of the bounds and get empirical values of Dyck ratio, we will need to efficiently determine Dyck edit distance.

Dyck edit distance can be obtained using a simple algorithm based on dynamic programming. Assume we have a Dyck string of length  $n$  and one fixed maximum matching of it. The main idea of the algorithm is that the optimal solution to the problem for length  $n$  can be obtained in one of two ways:

- Either the first character is matched in the maximum matching to the last one and then Dyck edit distance is identical to the Dyck edit distance of a substring of length  $n - 2$  missing the first and last character.
- Or the string can be split into a non-empty prefix and a non-empty suffix such that characters from the prefix do not match into the suffix in the maximum matching.

The running time of this algorithm is  $\mathcal{O}(n^3)$  and we state the full pseudo-code as Algorithm 1. We will return to this algorithm later to make some adjustments. Faster algorithms were developed by Briggmann et al. [7] and improved by Chi et al. [8]. They rely on Boolean matrix multiplication and run in  $\mathcal{O}(n^{2.687})$ . In fact it was proven that calculating Boolean matrix multiplication is at least as hard as calculating Dyck edit distance [9].

There is also a faster algorithm due to Fried et al. [10] running in  $\mathcal{O}(n^2k)$ , or even  $\mathcal{O}(n + k^{4.86})$ , where  $k$  is the Dyck edit distance. However, as we will show later we expect  $k$  to be on the order of  $n$  for a typical Dyck string, so they only offer an improvement in special cases, which we will mostly not address here.

Number of approximation algorithms were also developed with various tradeoffs between speed and accuracy. For example, Das et al. [11] invented a quadratic-time PTAS.

Unfortunately, for our purposes in this thesis none of the discovered algorithms can outperform Algorithm 1 which we will therefore use.

---

**Algorithm 1:** Algorithm to determine Dyck edit distance of a string

---

**Data:** Dyck string  $\delta$  of length  $l$

**Result:**  $e$ , the Dyck edit distance

Initialize array (matrix)  $E$  of size  $l \times l$  to zeroes.

**for**  $i \leftarrow 0$  **to**  $l - 1$  **do**

  |  $E[i, i] \leftarrow 1$

**end**

**for**  $i \leftarrow 2$  **to**  $l$  **do**

  | **for**  $j \leftarrow 0$  **to**  $l - i$  **do**

    |  $b \leftarrow 1 + E[j + 1, j + i - 1]$

    | **for**  $s \leftarrow 1$  **to**  $i - 1$  **do**

      |  $u \leftarrow E[j, j + s - 1] + E[j + s, j + i - 1]$

      | **if**  $b > u$  **then**

        |  $b \leftarrow u$

      | **end**

    | **end**

    | **if**  $\delta[j]$  and  $\delta[j + i - 1]$  can be matched **then**

      |  $u \leftarrow E[j + 1, j + i - 2]$

      | **if**  $b > u$  **then**

        |  $b \leftarrow u$

      | **end**

    | **end**

    |  $E[j, j + i - 1] \leftarrow b$

  | **end**

**end**

$e \leftarrow E[0, l - 1]$

---

## 2 Properties of Dyck ratios

In this chapter we only consider Dyck alphabets with at least two character pairs.

### 2.1 Existence and Concentration

We will now gradually show that the asymptotic Dyck ratio exists for all alphabets and that it is concentrated.

*Theorem 2.* Dyck edit distance is subadditive, i.e., for all Dyck alphabets  $\Delta$  and all even  $n, m \geq 2$  it holds

$$n \cdot dr(\Delta, n) + m \cdot dr(\Delta, m) \geq (n + m) \cdot dr(\Delta, n + m).$$

*Proof.* The fact that concatenation of two strings has Dyck edit distance at most the sum of Dyck edit distances of the two parts is apparent. We can then show:

$$\begin{aligned} (n + m) \cdot dr(\Delta, n + m) &= (n + m) \cdot \mathbb{E}_{\delta \in \Delta^{n+m}}[dr(\delta)] \\ &= \mathbb{E}_{\delta \in \Delta^{n+m}}[(n + m)dr(\delta)] \\ &= \mathbb{E}_{\delta_1 \in \Delta^n} [\mathbb{E}_{\delta_2 \in \Delta^m} [(n + m)dr(\text{concat}(\delta_1, \delta_2))]] \\ &\leq \mathbb{E}_{\delta_1 \in \Delta^n} [\mathbb{E}_{\delta_2 \in \Delta^m} [n \cdot dr(\delta_1) + m \cdot dr(\delta_2)]] \\ &= \mathbb{E}_{\delta_1 \in \Delta^n} [n \cdot dr(\delta_1)] + \mathbb{E}_{\delta_2 \in \Delta^m} [m \cdot dr(\delta_2)] \\ &= n \mathbb{E}_{\delta_1 \in \Delta^n} [dr(\delta_1)] + m \mathbb{E}_{\delta_2 \in \Delta^m} [dr(\delta_2)] \\ &= n \cdot dr(\Delta, n) + m \cdot dr(\Delta, m) \end{aligned}$$

Here *concat* symbolizes the concatenation of its arguments. □

*Theorem 3.* The asymptotic Dyck ratio exists for all Dyck languages.

*Definition 2.1.1* (Subadditive sequence). We say that a sequence  $\{a_n\}_{n=0}^{\infty}$  is *sub-additive* if

$$\forall i, j \in \mathbb{N} : a_i + a_j \geq a_{i+j}.$$

To prove this theorem we will refer to Fekete's subadditivity lemma [12]:

*Theorem 4* (Fekete's subadditivity lemma). For every subadditive sequence  $\{a_n\}_{n=0}^{\infty}$ , the limit  $\lim_{n \rightarrow \infty} \frac{a_n}{n}$  exists and is equal to the infimum  $\inf \frac{a_n}{n}$ .

*Proof of Theorem 3.* Let  $\Delta$  be a Dyck alphabet. Provided  $\lim_{\ell \rightarrow \infty} dr(\Delta, 2\ell)$  exists, the asymptotic ratio also exists. This is apparent since  $dr(\Delta, n) - dr(\Delta, n + 1)$  goes to zero as  $n$  increases.

It suffices to show that the series of  $dr(\Delta, 2\ell)$  converges. This however follows from Fekete's sub-additivity lemma for the sequence  $\{n \cdot dr(\Delta, n)\}$  and Theorem 2. □

# of pairs	Mean observed Dyck ratio
2	0.2262
3	0.3235
4	0.3875
5	0.4350
6	0.4711
10	0.5636
100	0.8395

**Table 2.1** Empirical values of Dyck ratio for alphabets with different number of pairs of symbols obtained by sampling strings of length 3000

We list values of Dyck ratio obtained by sampling in Table 2.1. We generated 1000 strings of length 3000 for each alphabet character by character by uniform sampling. We then used Algorithm 1 to obtain the Dyck edit distance.

We show the dependence of Dyck ratio and string length in Figure 2.1.

*Theorem 5.* The function  $dr(\cdot)$  is concentrated for any Dyck language  $\Delta$ , i.e. for any length  $\ell$ , any positive  $\varepsilon$ , and uniformly selected  $\delta \in \Delta^\ell$  there is a constant  $c$  such that

$$\Pr(|dr(\delta) - dr(\Delta, \ell)| \geq \varepsilon) \leq \exp(-c\varepsilon^2\ell).$$

To prove this theorem we will leverage property of bounded differences and a related inequality due to McDiarmid [13].

*Definition 2.1.2* (Bounded differences property). A function  $f : \mathcal{S}^\ell \rightarrow \mathbb{R}$  satisfies the *bounded differences property* if substituting the value of the  $i$ th coordinate  $x_i$  changes the value of  $f$  by at most  $c$ . More formally for all  $i \in \mathbb{Z}$ ,  $s \in S$  and vector  $X \in S^\ell$

$$|f(X_1, \dots, X_{i-1}, X_i, X_{i+1}, \dots, X_\ell) - f(X_1, \dots, X_{i-1}, s, X_{i+1}, \dots, X_\ell)| \leq c.$$

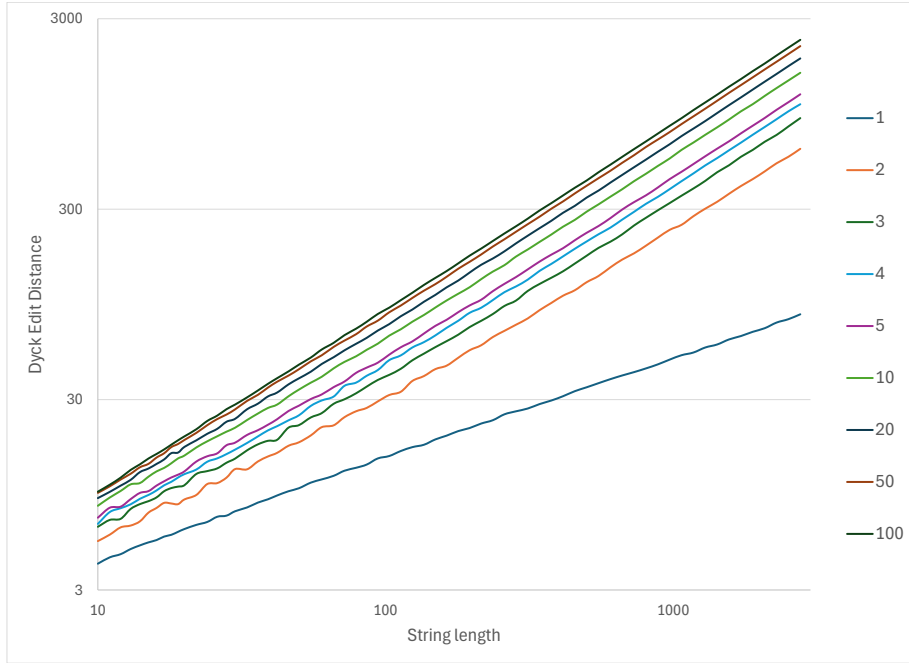
*Theorem 6* (McDiarmid's inequality). Let  $f : \mathcal{X}_1 \times \mathcal{X}_2 \times \dots \times \mathcal{X}_\ell \rightarrow \mathbb{R}$  satisfy the bounded differences property with bound  $c$ . Consider independent random variables  $X_1, X_2, \dots, X_\ell$  where  $X_i \in \mathcal{X}_i$  for all  $i$ . Then, for any  $\varepsilon > 0$ ,

$$\Pr(|f(X_1, X_2, \dots, X_\ell) - \mathbb{E}[f(X_1, X_2, \dots, X_\ell)]| \geq \varepsilon) \leq 2 \exp\left(-\frac{2\varepsilon^2}{lc^2}\right).$$

*Proof of Theorem 5.* For any  $\ell$  and Dyck language  $\Delta$ ,  $dr$  as a vector function satisfies the bounded differences property with  $c = 2/\ell$ . The rest follows from McDiarmid's inequality. I.e., for  $\delta \in \Delta^\ell$  it holds

$$\Pr(|dr(\delta) - dr(\Delta, \ell)| \geq \varepsilon) \leq 2 \exp\left(-\frac{\varepsilon^2\ell}{2}\right).$$

□



**Figure 2.1** Observed mean Dyck edit distances per string length and number of character pairs in the alphabet, based on uniform sampling. Both scales are logarithmic.

## 2.2 Upper bounds on Dyck ratio

We will now try to bound Dyck ratio from above in some way. The following theorem illustrates a relationship between average edit distance and Dyck edit distance for corresponding alphabets.

*Theorem 7.* Let  $\Delta$  be a Dyck alphabet with  $b$  pairs of symbols and  $\Gamma$  an ordinary alphabet with  $b$  symbols. Then

$$dr(\Delta, \infty) \leq 1/2 + \lim_{n \rightarrow \infty} \frac{\mathbb{E}_{\alpha, \beta \in \Gamma^n} [ed(\alpha, \beta)]}{4n}.$$

*Proof.* Let us first put some correspondence between Dyck ratio and edit distance of a pair of strings. Let  $\delta$  be a uniformly random Dyck string of length  $4n$ . Assume that we decompose this string into first half  $\sigma$  and second half  $\chi$ , both of length  $2n$ . Let us denote  $\sigma_1$  the string that is obtained from  $\sigma$  by skipping all closing symbols. Let us denote  $\chi_1$  the string that is obtained by reversing the order of symbols in  $\chi$ , skipping all opening symbols, and replacing all closing symbols by the opening symbol from the same pair. The mean length of  $\sigma_1$  and  $\chi_1$  is  $n$  and the lengths follow binomial distribution,

Now it can be seen that any non-crossing matching between  $\sigma_1$  and  $\chi_1$  naturally translates into a non-crossing matching in  $\delta$ , i.e., for any matched pair  $(s, x)$  of indices from  $\sigma_1$  and  $\chi_1$ , if we look at the original positions of symbols on those

$k$ (# of pairs)	Lower bound on $\gamma_k$	Upper bound on Dyck ratio
2	0.7739	0.6130
3	0.6153	0.6923
4	0.5455	0.7273
5	0.5062	0.7469
6	0.4717	0.7642
10	0.3866	0.8067

**Table 2.2** Upper bounds on Dyck ratio obtained by converting problem to LCS distance

indices in  $\delta$ , we get a pair of indices from  $\delta$ . The size of maximum matching between  $\sigma_1$  and  $\chi_1$  gives us a lower bound on the size of maximum matching in  $\delta$ .

Let us introduce  $\sigma_2, \chi_2$  as the truncation of  $\sigma_1, \chi_1$  to length at most  $\lceil n - n^{0.6} \rceil$ . We can now express the Dyck ratio of  $\delta$  as follows:

$$dr(\delta) \leq \begin{cases} 1/2 + n^{-0.4}/2 + \frac{|\sigma_2| + |\chi_2| - 2mm(\sigma_2, \chi_2)}{4n} & \text{if } |\sigma_1| \geq n - n^{0.6}, |\chi_1| \geq n - n^{0.6} \\ 1 & \text{otherwise.} \end{cases}$$

The term  $1/2 + n^{-0.4}/2 = \frac{2n - 2n^{0.6}}{4n}$  accounts for the characters outside  $\sigma_2$  and  $\chi_2$ . And  $\frac{|\sigma_2| + |\chi_2| - 2mm(\sigma_2, \chi_2)}{4n}$  corresponds to the characters that cannot be matched between  $\sigma_2$  and  $\chi_2$ .

We do not consider the second condition further, because asymptotically it has probability 0 with respect to choice of  $\delta$ . This follows from the Chernoff bound on the lengths of  $\sigma_1$  and  $\chi_1$ .

We may rewrite the term  $\frac{|\sigma_2| + |\chi_2| - 2mm(\sigma_2, \chi_2)}{4n}$  to  $\frac{ed(\sigma_2, \chi_2)}{4n}$ . Furthermore,  $\sigma_2, \chi_2$  are uniformly distributed and apparently the limit of  $n^{-0.4}/2$  goes to 0, so we may write

$$dr(\Delta, \infty) \leq 1/2 + \lim_{n \rightarrow \infty} \frac{\mathbb{E}_{\alpha, \beta \in \Gamma^{\lceil n - n^{0.6} \rceil}} ed(\alpha, \beta)}{4n} = 1/2 + \lim_{n \rightarrow \infty} \frac{\mathbb{E}_{\alpha, \beta \in \Gamma^n} [ed(\alpha, \beta)]}{4n}$$

So the proof is now complete.  $\square$

In the preceding theorem we are essentially matching symbols to symbols in the other half of the Dyck string. One may think that this is unusual and inefficient since most matches would be expected in a short distance. While this is true, we were not able to get any better result trying to find shorter matchings in this way.

Interestingly, if we decided to split the Dyck string in Theorem 7 into more than two parts and then looked at matches between consecutive couples of parts by converting the problems to LCS edit distance in the same fashion, we would not get any better bound.

Using the bounds of Dančák [4] on  $\gamma_k$  we produce upper bounds on Dyck ratio in Table 2.2

We can also try to place an upper bound on the asymptotic Dyck ratio by considering the mean Dyck ratio for strings of some fixed length.

# of pairs	Upper bound on Dyck ratio	Length of segments used
2	0.5354	12
2	0.5229	13
2	0.5115	14
3	0.6458	10
4	0.7331	8
4	0.7148	9

**Table 2.3** Upper bounds on asymptotic Dyck ratio for alphabets with different number of pairs of symbols obtained from short segments of different lengths

*Theorem 8.* Let  $\Delta$  be a Dyck alphabet and  $\ell$  a positive integer. Then  $dr(\Delta, \ell) \geq dr(\Delta, \infty)$ .

*Proof.* Firstly we notice that what suffices to prove is  $\forall \varepsilon > 0 \exists m \forall n : n > m \implies dr(\Delta, m) < dr(\Delta, \ell) + \varepsilon$  and it would follow that  $dr(\Delta, \ell) \geq dr(\Delta, \infty)$  from definition of the asymptotic ratio.

Let us therefore take a positive  $\varepsilon$ , we set  $n$  as  $\lceil \ell/\varepsilon \rceil$ . We can view a string of length  $n$  as some number of segments of length  $\ell$  and final segment (perhaps empty) of length less than  $\ell$ . One way to get an upper bound on Dyck ratio of such string is to only allow matches within the segments and no matches for characters within the final segment. One can see that the marginal distribution of every segment is the same as strings of length  $\ell$ . We can therefore conclude by showing:

$$\begin{aligned} dr(\Delta, n) &\leq \lfloor n/\ell \rfloor \cdot (\ell \cdot dr(\Delta, \ell)) + \ell/n \\ dr(\Delta, n) &\leq dr(\Delta, \ell) + \ell/n \\ dr(\Delta, n) &\leq dr(\Delta, \ell) + \varepsilon \end{aligned}$$

□

We used Theorem 8 to give us some particular upper bounds by enumerating all segments for given alphabets and calculating their Dyck ratios. The results can be seen in Table 2.3. In theory there is no limit to how close to the asymptotic ratio we could get by extending the segments further and further. Since number of strings grows exponentially with length however, this approach is only sustainable for short strings and small alphabets.

For larger alphabets we may instead turn to another approach inspired by the birthday paradox.

*Theorem 9.* There is a function  $f : \mathbb{N} \rightarrow \mathbb{N} \in \Theta(n^{-1/2})$  such that for any  $\Delta$  a Dyck alphabet with  $b \geq 2$  character pairs  $dr(\Delta, \infty) \leq 1 - f(b)$ .

*Proof.* Let  $\delta$  be a uniformly selected string over  $\Delta$  of length  $\ell$ . We slice  $\delta$  into segments of length  $4\lceil\sqrt{b}\rceil$ . We ignore the last segment as it is insignificant with respect to asymptotic.

In each segment the probability that there are at least  $\lceil\sqrt{b}\rceil$  opening symbols in the first  $2\lceil\sqrt{b}\rceil$  symbols is at least  $1/2$ . And also the probability that there are at least  $\lceil\sqrt{b}\rceil$  closing symbols in the remaining  $2\lceil\sqrt{b}\rceil$  symbols is at least  $1/2$ .

The probability that of the  $\lceil\sqrt{b}\rceil$  opening symbols all of them are distinct is at least  $\left(\frac{b-\lceil\sqrt{b}\rceil}{b}\right)^{\lceil\sqrt{b}\rceil}$ , which tends to  $1/e$  as  $b$  goes to infinity.

Conditioned on that, the probability that at least one of the closing symbols in the second half of this segment matches one these opening characters is  $1 - \left(\frac{b-\lceil\sqrt{b}\rceil}{b}\right)^{\lceil\sqrt{b}\rceil}$  which approaches  $1 - 1/e$  as  $b$  goes to  $\infty$ .

So overall there is some probability bounded by a constant from below that there is a match in each segment of length  $4\lceil\sqrt{b}\rceil$ . Thus expected edit distance of  $\delta$  would be  $\ell - \frac{\ell}{\lceil\sqrt{b}\rceil} \cdot c$  for some positive constant shared for all alphabets  $c$ .  $\square$

For example for alphabet  $\Delta$  with 100 character pairs we can show that  $dr(\Delta, \infty) \leq 0.9489$ .

# 3 Distance of Pairs in Matchings

If we knew that most characters can be matched to nearby characters, it would indicate the possibility of obtaining good bounds on Dyck ratio just by looking at local segments of a string. We could also design efficient approximation algorithms to determine the Dyck edit distance if distant matches were not required. This assumption unfortunately does not quite happen to be the case as we discuss in this chapter.

## 3.1 Necessary optimal matching distance of a Dyck string

We already defined the maximum matching in a Dyck string. In this section we will discuss what is perhaps an interesting property of those – the distances between indices within matched pairs. In particular we define maximum distance of a matching.

*Definition 3.1.1* (Maximum distance of a Dyck matching). Let  $D$  be a Dyck matching for string  $\delta \in \Delta^n$ . We define the *maximum distance* of  $D$  to be

$$md(D) = \max_{(i,j) \in D} j - i.$$

It is important to remark that maximum distance is really a property of a matching, not the string itself. We therefore try to define a similar property for Dyck strings.

*Definition 3.1.2* (Necessary optimal matching distance of a Dyck string). Let  $\mathbb{M}$  be the set of optimal Dyck matching for string  $\delta \in \Delta^n$ . We define the *necessary optimal matching distance* of  $\delta$  to be

$$\min_{D \in \mathbb{M}} md(D).$$

We can amend Algorithm 1 for determining Dyck edit distance to also produce the necessary optimal matching distance. We again assume to have a Dyck string of length  $n$  and one fixed maximum matching of it, this time such that it has the least maximum matching distance. We again think of how the problem can be reduced to instances of smaller size:

- If the first character is matched in the fixed maximum matching to the last one and thus the Dyck edit distance is identical to the Dyck edit distance of a substring of length  $n - 2$  missing the first and last character, then clearly the maximum matching distance is  $n - 1$ .
- Otherwise the string can be split into a non-empty prefix and a non-empty suffix such that characters from the prefix do not match with the suffix in the selected maximum matching. In this case the maximum matching distance is the larger of necessary optimal matching distances of the two substrings.

Hence we can try all possible decompositions and among those that produce minimum Dyck edit distance we find such decomposition that produces least maximum matching distance.

The complexity of this modified algorithm remains  $\Theta(n^3)$  when implemented using dynamic programming. The pseudocode is given in Algorithm 2.

---

**Algorithm 2:** Algorithm for determining the necessary optimal matching distance of a Dyck string

---

**Data:** Dyck string  $\delta$  of length  $l$

**Result:**  $e$ , the Dyck edit distance of  $\delta$ ,  $v$ , the necessary optimal matching distance

Initialize arrays (matrices)  $E$  and  $V$  of size  $l \times l$  to zeroes.

**for**  $i \leftarrow 0$  **to**  $l - 1$  **do**

  |  $E[i, i] \leftarrow 1$

**end**

**for**  $i \leftarrow 2$  **to**  $l$  **do**

  | **for**  $j \leftarrow 0$  **to**  $l - i$  **do**

    |  $b \leftarrow 1 + E[j + 1, j + i - 1]$

    |  $d \leftarrow V[j + 1, j + i - 1]$

    | **for**  $s \leftarrow 1$  **to**  $i - 1$  **do**

      |  $u \leftarrow E[j, j + s - 1] + E[j + s, j + i - 1]$

      |  $q \leftarrow \text{Max}(V[j, j + s - 1], V[j + s, j + i - 1])$

      | **if**  $b = u$  **then**

        |  $d \leftarrow \text{Min}(d, q)$

      | **end**

      | **if**  $b > u$  **then**

        |  $b \leftarrow u, d \leftarrow q$

      | **end**

    | **end**

    | **if**  $\delta[j]$  and  $\delta[j + i - 1]$  can be matched **then**

      |  $u \leftarrow E[j + 1, j + i - 2]$  **if**  $b = u$  **then**

        |  $d \leftarrow \text{Min}(d, i - 1)$

      | **end**

      | **if**  $b > u$  **then**

        |  $b \leftarrow u, d = i - 1$

      | **end**

    | **end**

    |  $E[j, j + i - 1] \leftarrow b$

    |  $V[j, j + i - 1] \leftarrow d$

  | **end**

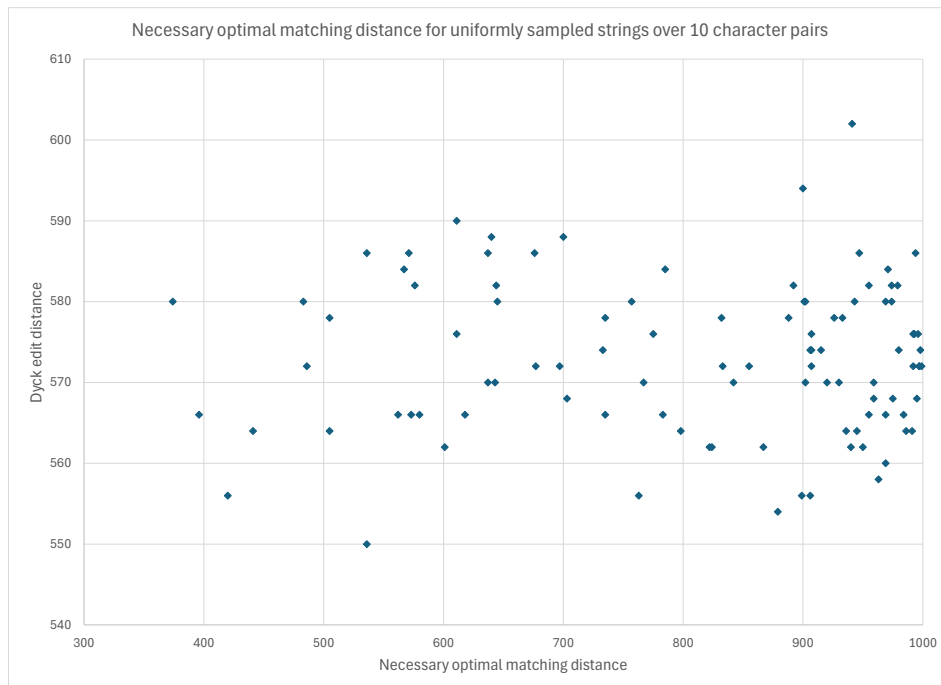
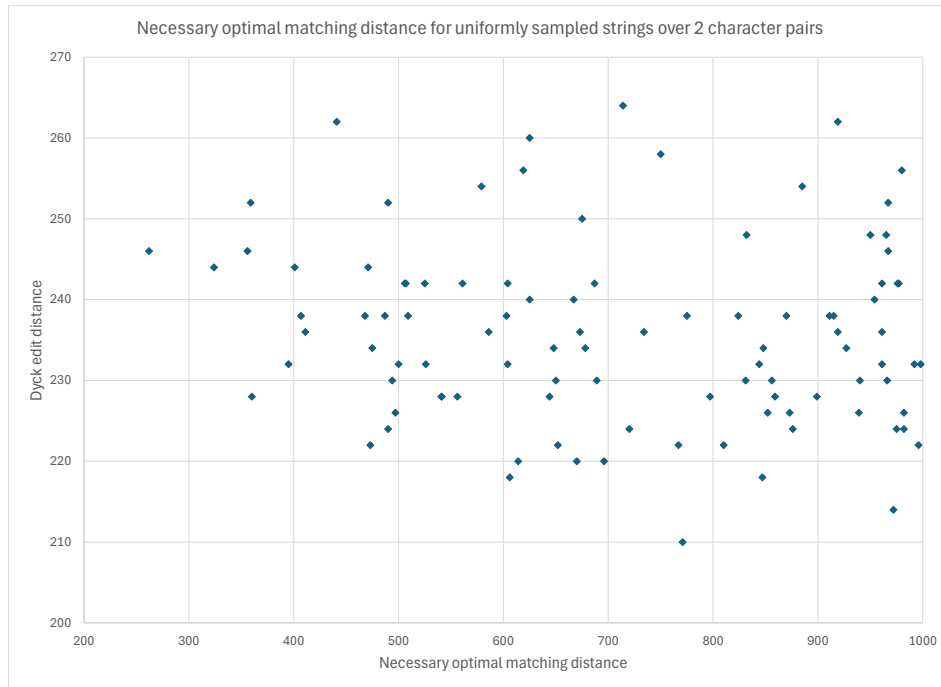
**end**

$e \leftarrow E[0, l - 1]$

$v \leftarrow V[0, l - 1]$

---

We present some observations of necessary optimal matching distance and its dependence on Dyck edit distance in Figure 3.1. We generated strings of length 1000 uniformly and then calculated Dyck edit distance and necessary optimal matching distance. We can notice that there is not much of a correlation. More importantly large distances are quite frequently required.



**Figure 3.1** Scatter plot of Dyck edit distance and necessary optimal matching distance for strings of length 1000 over different alphabets

# of pairs	Distance 10	Distance 33	Distance 100	Unlimited
2	0.4255	0.3128	0.2642	0.2359
3	0.5220	0.4089	0.3604	0.3315
4	0.5844	0.4723	0.4229	0.3955
10	0.7508	0.6473	0.5998	0.5718
100	0.9593	0.9064	0.8709	0.8456

**Table 3.1** Observed Dyck ratio with limited matching distance for alphabets with different numbers of character pairs on strings of length 1000

## 3.2 Limited maximum distance for matches

We can again modify Algorithm 1 for determining Dyck edit distance to only consider matches that have at most a specified maximum distance  $v$ . We again assume to have a Dyck string of length  $n$  and one fixed matching of it with distance at most  $v$  that is maximal given this constraint. In some sense the algorithm now has two phases. We first determine the sizes of optimal matchings for all substrings of length at most  $v + 1$  by checking the two options for decomposition as in Algorithm 1.

After that we use the values for the short substrings to calculate Dyck edit distance for all prefixes of the original string. We do this in order of increasing length. If the prefix has length over  $v + 1$ , it can be seen that there must exist such a decomposition of the prefix into two non-empty strings that matches only exist within the two parts. Furthermore, the last character of the prefix cannot match further than  $v$  characters away, this means that it is sufficient to try decompositions where the second string's length is at most  $v + 1$  – thus the precalculated values can be looked up from the previous phase.

Complexity of the algorithm is  $\Theta(n \cdot v^2)$  for the first phase plus  $\Theta(n \cdot v)$  for the second.

We obtained some data for limited maximum distance experimentally, it can be seen in Table 3.1. We generated 300 strings for each alphabet size and calculated the Dyck edit distance along with the restriction on maximum matching distance using the algorithm above. The values seen are averaged over all generated strings.

# 4 Kolmogorov Lower Bound on Dyck Ratio

We will use Kolmogorov complexity to show that strings that are close to being well-parenthesized can be compressed easily and this implies that there cannot be too many of them. In particular we will define an embedding of strings of some length over a chosen alphabet with Dyck ratio up to some constant into binary strings of limited length. This will give us an upper bound on the count of such strings with low Dyck ratio.

Let us now define a coding that will compress strings with low Dyck ratio.

*Definition 4.0.1 (Match Entropy).* We define *match entropy* for a Dyck ratio  $r$  to be:  $H_m(r) = -r \cdot \log_2(r/2) - 1 - r \cdot \log_2((1-r)/2)$ .

In this definition we assume as is customary with entropy that  $0/\log_2(0) = 0$ .

*Theorem 10.* Let  $\Delta$  be a Dyck alphabet with  $b \geq 2$  pairs of symbols. For any  $r \in (0, 1)$  such that  $\log(b) + 2\log(1-r) - 2\log(r) > 0$  there is a sequence of encodings, one for each length of strings over  $\Delta$  assigning a binary sequence of length at most  $(H_m(r) + (1/2 + r/2)\log_2(b)) \cdot \ell + o(\ell)$  to a string of length  $\ell$  over  $\Delta$  with Dyck ratio of at most  $r$ .

*Proof.* Let  $\delta$  be a string over  $\Delta$  of length  $\ell$  of with Dyck ratio at most  $r$ . We will define the sequence of encodings by describing the encoding for  $\delta$ .

The encoding will have three parts. Firstly, we will encode the Dyck edit distance of the string into  $\lceil \log(\ell) \rceil + 1$  bits.

Then we take an arbitrary maximum matching of  $\delta$  and look at which characters are matched. For every character in  $\delta$  we encode one of four values: opening matched, opening unmatched, closing matched, closing unmatched. This is done with iterated binary Shannon-Fano coding [14] of source  $4^\ell$  for respective probabilities of the four symbols:

opening matched	$dr(\delta)/2$
opening unmatched	$(1 - dr(\delta))/2$
closing matched	$dr(\delta)/2$
closing unmatched	$(1 - dr(\delta))/2$

After that we encode for each character in  $\delta$  the character pair it belongs to skipping closing matched characters. From the first part of the encoding we can already deduce how many characters we will skip, so we can again use iterated Shannon-Fano coding using uniform distribution on pairs.

(Length of the first part of the encoding is known. So given that Shannon-Fano coding produces a prefix code, we can simply concatenate all the parts and decoding will still be unambiguous.)

It can be seen that this encoding is injective, i.e. no two strings over  $\Delta$  of length  $\ell$  of with Dyck ratio at most  $r$  would have identical encoding.

It remains to be shown that the whole encoding is sufficiently short. Length of the first part is  $o(\ell)$ . The length of the second part is  $H_m(\delta) \cdot \ell + o(1)$ . The length of the third part is  $\ell \cdot (1/2 + dr(\delta)/2) \cdot \log_2(b) + o(1)$ .

We now need to reason that

$$\ell \cdot (H_m(\delta) + (1/2 + dr(\delta)/2) \cdot \log_2(b)) \leq (H_m(r) + (1/2 + r/2) \log_2(b)) \cdot \ell.$$

Or equivalently

$$\begin{aligned} H_m(\delta) + (1/2 + dr(\delta)/2) \cdot \log_2(b) &\leq H_m(r) + (1/2 + r/2) \cdot \log_2(b) \\ 2H_m(\delta) + dr(\delta) \log_2(b) &\leq 2H_m(r) + r \log_2(b) \end{aligned}$$

The last inequality holds due to the assumption  $\log_2(b) + 2\log_2(1-r) - 2\log_2(r) > 0$ . (One can take the derivative of  $2H_m(x) + x \log_2(b)$  and see that it is positive on  $(0, r)$ .)

So altogether the length is at most  $(H_m(r) + (1/2 + r/2) \log_2(b)) \cdot \ell + o(\ell)$ , which finished the proof.  $\square$

Finally, the following theorem can be used to obtain lower bounds on asymptotic Dyck ratios.

*Theorem 11.* Let  $\Delta$  be a Dyck alphabet with  $b$  pairs of symbols and let  $r \in (0, 1)$  satisfy  $H_m(r) - (1-r) \log_2(b)/2 < 1$ . Then  $dr(\Delta, \infty) \geq r$ .

*Proof.* We look at the sequence of encodings we defined in Theorem 10. We can conclude that for length  $\ell$  there are at most  $2^{(H_m(r) + (1/2 + r/2) \log_2(b)) \cdot \ell + o(\ell)} = q(\ell)$  Dyck strings of length  $\ell$  with Dyck ratio at most  $r$ . This is simply based on total number of binary sequences of length  $(H_m(r) + (1/2 + r/2) \log_2(b)) \cdot \ell + o(\ell)$  or less.

Since there are  $(2b)^\ell$  Dyck strings of length  $\ell$  and from

$$H_m(r) - (1-r) \log_2(b)/2 < 1$$

it follows that there is  $\varepsilon \in (0, 1)$  and  $n \in \mathbb{N}$  such that

$$\forall \ell > n : \frac{q(\ell)}{(2b)^\ell} < \varepsilon$$

We denote the quantile function by  $Q$ . So in other terms  $Q(\varepsilon, dr(\Delta, \ell)) \geq r$  for all sufficiently large  $\ell$ . From concentration of properties of dyck ratio we also deduce however:

$$\forall \zeta \in (0, 1) : \lim_{\ell \rightarrow \infty} Q(\zeta, dr(\Delta, \ell)) - Q(\varepsilon, dr(\Delta, \ell)) = 0$$

For all  $\zeta \in (0, 1)$  it holds that for every  $\eta \in (0, 1)$  there is  $m \in \mathbb{N}$  such that  $Q(\zeta, dr(\Delta, \ell)) \geq r - \eta$  for all  $\ell > m$ . From here it follows however that  $dr(\Delta, \infty) \geq r$ .  $\square$

Table 4.1 shows some of the calculated lower bounds using Theorem 11.

The bounds we obtained could be improved. One can observe that there we had a considerable degree of freedom when compressing a string. We chose an arbitrary maximum matching. There would often be many matchings to choose from, each of them yielding a different binary sequence of target length that we would not map anything else to. This means that compressed strings are not very

Number of pairs of symbols	Lower bound
2	0.0948
3	0.1700
4	0.2270
5	0.2723
6	0.3094
7	0.3406
8	0.3675
9	0.3909
10	0.4115
100	0.7613
1000	0.9176

**Table 4.1** Lower bound on asymptotic Dyck ratio for different alphabet sizes

dense in the binary sequences. This presents a potential point of improvement of the lower bounds we obtained.

The count of maximum matchings can vary significantly for a given string. For example, there would be always only one for already well-parenthesized Dyck string. But even strings with high Dyck edit distance can still have only one maximum matching, like the string  $))) \dots )))$  only has empty matching.

On the other hand the number of optimal matchings can be exponential. The following simple string has  $\binom{2n}{n}$  maximum matchings:

$$\overbrace{(((\dots (((\ ))) \dots )))}^n$$

## 5 Single Pair of Symbols

Perhaps surprisingly, the properties of Dyck edit distance for alphabets with only one pair of parentheses are different from other alphabets. For larger alphabet, one can observe empirically that expected Dyck edit distance of a random string is asymptotically linear in its length. This is not the case for alphabets with only one pair of symbols however, where Dyck edit distance of a string of length  $n$  is roughly of the order of  $\Theta(\sqrt{n})$ . We will now focus on those.

We will show that we can express Dyck edit distance in terms of random variables of suitable Markov chains. The following theorem will be fundamental for that:

*Theorem 12.* Let  $\Delta$  be a Dyck alphabet with a single pair of symbols and let  $\delta$  be a string over  $\Delta$  of length  $n$ . We define  $q_i$  to be 1 if  $\delta_i$  is the opening symbol and 0 otherwise. And we define  $X_0 = 0$  and  $X_i = X_{i-1} + 1 - 2q_i$ . Then  $3 \max_i |X_i| \geq n \cdot dr(\delta)$ .

*Proof.* We will show that there is such a matching  $M$  that leaves at most  $2 \max_i |X_i|$  indices unmatched. The matching  $M$  is composed of the following: For every opening symbol  $s$  we find the shortest balanced substring of  $\delta$  beginning with this symbol and match it to the last symbol of this substring (which must be closing), if such a substring exists. Remaining symbols are unmatched. One can easily verify that this matching is valid.

It is also optimal. To show this we prove the following claim: If there is an opening character directly followed by a closing character, there is a maximum matching where those two are matched together. To prove it by contradiction, let us take a maximum matching where this is not true, there exactly one of the characters is matched. So we may replace that match by the pair of symbols directly next to each other, we thus get a maximum matching including the desired pair.

This however means that whenever there is an opening character followed by a closing character, we can match them together and determine the rest of the matching by looking at a string with those two characters removed. Clearly, as long as the maximum matching is not empty, there is such a pair, and the converse is also true. Now however the way  $M$  is defined corresponds precisely to the matches obtained by this procedure, so  $M$  is maximum.

Now we must think about which symbols are not matched in  $M$ . Suppose there was an unmatched opening character followed by an unmatched closing character somewhere later in  $\delta$ , we could take closest such a pair of positions  $(i, j)$ . Between these two positions, there would need to be a (possibly empty) balanced string. By definition of  $M$  matches cannot cross over unmatched characters. We would then however need to match positions  $(i, j)$ . So what this implies is that there is a part of the string  $C$  where unmatched closing symbols can exist and after this part there is another part  $O$  where unmatched opening symbols can exist.

$|X_i|$  indicates the difference between in the count of opening and closing characters in the prefix of  $\delta$  of length  $i$ . Let's assume there are  $c$  unmatched closing characters and  $o$  number of unmatched opening characters.

We now want to show that number of unmatched symbols in  $C$  is at most  $\max_i |X_i|$ . Let's take the position of the final unmatched closing character  $p$ . Now  $X_p$  is less than equal to the number of unmatched closing characters. Otherwise the difference between closing and opening characters in the prefix of length  $p$  would be less than  $c$  and character at  $p$  or some other preceding closing character would need to be matched in  $M$ .

If  $o \leq 2c$  and we are finished. Otherwise by analogous logic we can now argue that there is a suffix of  $\delta$  containing all  $o$  unmatched opening characters of some length  $\ell$  that has difference between closing and opening characters of  $\frac{-\max_i |X_i|}{2}$  or more. This is because  $X_{n-\ell} - X_n \leq o$ . □

We see that if we can show that  $X_i$ s will on average be close to zero, Dyck edit distance will be also. So we will now focus on bounding  $\max_i |X_i|$ .

*Theorem 13.* For any integer  $n$  let us define independent random variables  $Z_i \in \{-1, 1\}$  for  $1 \leq i \leq n$ . Based on those we also define  $X_0 = 0$ ,  $X_i = X_{i-1} + Z_i$  for  $1 \leq i \leq n$ . Then  $\mathbb{E}[\max_{i=0, \dots, n} |X_i|] \in \mathcal{O}(n^{1/2+\varepsilon})$  for any positive  $\varepsilon$ .

We will prove this theorem using the Chernoff bound [15] and a few observations.

*Theorem 14* (Chernoff bound). Suppose  $X_1, \dots, X_n$  are independent random variables taking values in  $\{0, 1\}$ . Let  $X$  denote their sum and let  $\mu = E[X]$  denote the sum's expected value. Then for any  $\eta \in (0, 1)$ ,

$$\Pr(|X - \mu| \geq \eta\mu) \leq 2e^{-\eta^2\mu/3}$$

*Proof of Theorem 13.*  $|X_i| > a$  and  $|X_j| > a$  are positively correlated for any pair  $i, j$ .  $\Pr(|X_n| > a)$  is the greatest, so we can multiply this probability by  $n$  and use it as an upper bound for the probability of the event that for some  $i$  it occurs  $|X_i| > a$ .

We take the Chernoff bound for  $Y = X_n/2 + n/2$  which can be seen to be a sum of  $n$  independent random variables  $Z_i/2 + 1/2$  taking value in  $\{0, 1\}$ . We get  $\Pr(|Y - n/2| \geq \eta n/2) \leq 2e^{-\eta^2 n/6}$  and substitute  $\eta = n^{-1/2+\varepsilon}$  yielding  $\Pr(|Y - n/2| \geq n^{1/2+\varepsilon}/2) \leq 2e^{-n^{2\varepsilon}/6}$ .

In terms of  $X_n$  this can be expressed as  $\Pr(|X_n| \geq n^{1/2+\varepsilon}) \leq 2e^{-n^{2\varepsilon}/6}$

Now to bound  $\mathbb{E}[\max_{i=0, \dots, n} |X_i|]$  we can do it as follows:

$$\begin{aligned} \mathbb{E}[\max_{i=0, \dots, n} |X_i|] &\leq n^{1/2+\varepsilon} \Pr\left(\max_{i=0, \dots, n} |X_i| \leq n^{1/2+\varepsilon}\right) \\ &\quad + n \Pr\left(\max_{i=0, \dots, n} |X_i| > n^{1/2+\varepsilon}\right) \\ &\leq n^{1/2+\varepsilon} + 2ne^{-n^{2\varepsilon}/6} \\ &\in \mathcal{O}(n^{1/2+\varepsilon}) \end{aligned}$$

□

We may now notice that  $Z_i$ s can be thought of as indicators of whether a character at position  $i$  is opening or closing, so in fact we just received an upper

bound on expected Dyck edit distance for Dyck alphabets with only a single pair of symbols.

Then the next theorem directly follows:

*Theorem 15.* Let  $\Delta$  be a Dyck alphabet with a single pair of symbols, then  $dr(\Delta) = 0$ .

We should also show a lower bound on Dyck edit distance for single pair of symbols. For that we look at the following property of random walks.

*Theorem 16* (Running maximum of Markov process on a line (found in [16])). Random walk on integer line is a process where in each step with probability  $1/2$  we stay at the same place, with probability  $1/4$  we step to the left, and otherwise we step to the right.

Random walk on an integer line starting at 0 will reach point with value  $v \in \mathbb{N}$  within the first  $n$  steps with probability at least  $1 - 12v/\sqrt{n}$ .

A lower asymptotic bound on the edit distance is an obvious corollary of this theorem.

*Theorem 17.* Let  $\Delta$  be a Dyck alphabet with a single pair of symbols, then  $\mathbb{E}_{\delta \in \Delta^n} [Ded(\delta)] \in \Omega(n^{1/2})$ .

*Proof.* Setting in  $v = \sqrt{n}$  Theorem 16 tells us that we can expect a prefix with  $\sqrt{\ell}$  more closing symbols than opening symbols in a constant fraction of string of length  $\ell$ . In each those strings at least  $\sqrt{\ell}$  symbols must therefore remain unmatched in any matching. This, however, suffices to prove the claim.  $\square$

So we see that the situation for the case described in this section is very different from other Dyck alphabets. Additionally, we can compute Dyck edit distance of a string over alphabet with a single pair of characters with a simpler algorithm.

The idea of this algorithm is to find the prefix that has the greatest difference between closing and opening symbols. This value  $m_1$  will be equal to  $-m$  at the end of the algorithm.

If  $m_1 \leq 0$ , we know that in any prefix there are enough opening symbols to match all closing symbols, therefore the edit distance is the number of excess opening symbols in the whole string, i.e.  $r$  at the end of the algorithm.

Otherwise  $m_1$  closing characters will remain unmatched in some prefix. It can be seen that removing this prefix and running the algorithm on the rest of the string would yield the previous case. Therefore the final Dyck edit distance is equal to  $m_1$  plus the excess opening characters in the rest of the string which can be seen to be equal to  $r - m$  at the end of the algorithm.

---

**Algorithm 3:** Algorithm for calculating Dyck edit distance with single character pair

---

**Data:** Dyck string  $\delta$  of length  $\ell$   
**Result:**  $e$ , the Dyck edit distance of  $\delta$   
 $m \leftarrow 0$   
 $r \leftarrow 0$   
**for**  $i \leftarrow 0$  **to**  $\ell - 1$  **do**  
    **if**  $\delta[i]$  *is opening character* **then**  
         $r \leftarrow r + 1$   
    **else**  
         $r \leftarrow r - 1$   
    **end**  
    **if**  $r < m$  **then**  
         $m \leftarrow r$   
    **end**  
**end**  
 $e \leftarrow r$   
**if**  $m < 0$  **then**  
     $e \leftarrow e - 2m$   
**end**

---

## 6 Further Research

We note here some possible further research directions:

- Sankoff and Mainville found the asymptotic behavior  $\gamma_k \sqrt{k} \rightarrow 2$ . Similar formula may exist for Dyck edit distance. For series of alphabets  $\{\Delta_i\}$  where  $\Delta_i$  contains  $i$  character pairs, it may perhaps be of the form

$$\sqrt{k}(1 - dr(\Delta_i, \infty)) \rightarrow c$$

for some constant  $c$  as  $k$  goes to infinity.

- With more computational power the upper bounds on Dyck ratio that were obtained using analyzing short segments in Table 2.3 could be improved. In theory we could get arbitrarily close to the real Dyck ratio this way.
- It would be interesting to see the expected number of maximum matchings in a Dyck string and also near maximum matchings.
- We may decide to also allow substitutions on top of just insertions and deletions, and thus consider the alternative definition of the Dyck edit distance problem. Most properties would be analogous, however the exact bounds on this modified Dyck ratio would be a bit lower.
- It appears that many of the results of this thesis could be extended to the problem of RNA folding without great difficulty.

# Bibliography

1. CHVÁTAL, Vacláv; SANKOFF, David. *Longest common subsequences of two random sequences*. Vol. 12. Cambridge University Press (CUP), 1975. No. 2. Available from DOI: 10.2307/3212444.
2. KRUSKAL; SANKOFF. Common subsequences and monotone subsequences. 1983, pp. 363–365.
3. KIWI, Marcos; LOEBL, Martin; MATOUSEK, Jiri. Expected length of the longest common subsequence for large alphabets. 2003. Available from DOI: 10.48550/ARXIV.MATH/0308234.
4. DANČÍK, Vladimír. *Expected length of longest common subsequences, Ph.D. thesis*. University of Warwick, 1994.
5. BAEZA-YATES, R. A.; NAVARRO R. Gavaldà, G.; SCHEIHING, R. *Bounding the Expected Length of Longest Common Subsequences and Forests*. Vol. 32. Springer Science and Business Media LLC, 1999. No. 4. Available from DOI: 10.1007/s002240000125.
6. LEVENSHTAIN, V. I. *Binary codes capable of correcting deletions, insertions, and reversals*. 1966.
7. BRINGMANN, Karl; GRANDONI, Fabrizio; SAHA, Barna; WILLIAMS, Virginia Vassilevska. Truly Subcubic Algorithms for Language Edit Distance and RNA Folding via Fast Bounded-Difference Min-Plus Product. *SIAM Journal on Computing*. 2019, vol. 48, no. 2, pp. 481–512. Available from DOI: 10.1137/17M112720X.
8. CHI, Shucheng; DUAN, Ran; XIE, Tianle; ZHANG, Tianyi. *Faster Min-Plus Product for Monotone Instances*. 2022. Available from arXiv: 2204.04500 [cs.DS].
9. ABBOUD, Amir; BACKURS, Arturs; WILLIAMS, Virginia Vassilevska. If the Current Clique Algorithms Are Optimal, so Is Valiant’s Parser. *SIAM Journal on Computing*. 2018, vol. 47, no. 6, pp. 2527–2555. Available from DOI: 10.1137/16M1061771.
10. FRIED, Dvir; GOLAN, Shay; KOCIUMAKA, Tomasz; KOPELOWITZ, Tsvi; PORAT, Ely; STARIKOVSKAYA, Tatiana. An Improved Algorithm for The  $k$ -Dyck Edit Distance Problem. *CoRR*. 2021, vol. abs/2111.02336. Available from arXiv: 2111.02336.
11. DAS, Debarati; KOCIUMAKA, Tomasz; SAHA, Barna. Improved Approximation Algorithms for Dyck Edit Distance and RNA Folding. *CoRR*. 2021, vol. abs/2112.05866. Available from arXiv: 2112.05866.
12. FEKETE, Michael. *Über die Verteilung der Wurzeln bei gewissen algebraischen Gleichungen mit ganzzahligen Koeffizienten*. Vol. 17. Springer Science and Business Media LLC, 1923. No. 1. Available from DOI: 10.1007/bf01504345.
13. MCDIARMID, Colin. On the method of bounded differences. In: *Surveys in Combinatorics, 1989: Invited Papers at the Twelfth British Combinatorial Conference*. Ed. by SIEMONS, J. Editor. Cambridge University Press, 1989, pp. 148–188. London Mathematical Society Lecture Note Series.

14. SHANNON, C. E. *A Mathematical Theory of Communication*. Vol. 27. Institute of Electrical and Electronics Engineers (IEEE), 1948. No. 3. Available from DOI: 10.1002/j.1538-7305.1948.tb01338.x.
15. CHERNOFF, Herman. A Measure of Asymptotic Efficiency for Tests of a Hypothesis Based on the sum of Observations. *The Annals of Mathematical Statistics*. 1952, vol. 23, no. 4, pp. 493–507. Available from DOI: 10.1214/aoms/1177729330.
16. ALDOUS, David. *Markov Chains and Mixing Times (Second Edition)* by David A. Levin and Yuval Peres. Vol. 41. Springer Science and Business Media LLC, 2018. No. 1. Available from DOI: 10.1007/s00283-018-9839-x.

# List of Tables

1	Overview of our bounds on Dyck edit distance and empirical values	7
2.1	Empirical values of Dyck ratio for alphabets with different number of pairs of symbols obtained by sampling strings of length 3000 . .	14
2.2	Upper bounds on Dyck ratio obtained by converting problem to LCS distance . . . . .	16
2.3	Upper bounds on asymptotic Dyck ratio for alphabets with different number of pairs of symbols obtained from short segments of different lengths . . . . .	17
3.1	Observed Dyck ratio with limited matching distance for alphabets with different numbers of character pairs on strings of length 1000	22
4.1	Lower bound on asymptotic Dyck ratio for different alphabet sizes	25