

Nearest constrained circular words

Guillaume Blin

LaBRI, Université de Bordeaux, CNRS UMR 5800, Talence, France
guillaume.blin@labri.fr

Alexandre Blondin Massé¹

Dép. d'informatique, Université du Québec à Montréal, QC, Canada
blondin_masse.alexandre@uqam.ca

Marie Gasparoux²

LaBRI, Université de Bordeaux, CNRS UMR 5800, Talence, France
Dép. d'informatique et de recherche opérationnelle, Université de Montréal, QC, Canada
marie.gasparoux@umontreal.ca

Sylvie Hamel³

Dép. d'informatique et de recherche opérationnelle, Université de Montréal, QC, Canada
sylvie.hamel@umontreal.ca

Élise Vandomme

LaCIM, Université du Québec à Montréal, QC, Canada
elise.vandomme@lacim.ca

Abstract

In this paper, we study circular words arising in the development of equipment using shields in brachytherapy. This equipment has physical constraints that have to be taken into consideration. From an algorithmic point of view, the problem can be formulated as follows: Given a circular word, find a constrained circular word of the same length such that the Manhattan distance between these two words is minimal. We show that we can solve this problem in pseudo polynomial time (polynomial time in practice) using dynamic programming.

2012 ACM Subject Classification Theory of computation → Dynamic programming, Mathematics of computing → Combinatorics on words, Applied computing → Bioinformatics

Keywords and phrases Circular constrained alignments, Manhattan distance, Application to brachytherapy

Digital Object Identifier 10.4230/LIPIcs.CPM.2018.6

Supplement Material A Haskell implementation of the algorithms discussed in this paper is available at <https://gitlab.com/ablondin/nearest-constrained-circular-words>

¹ Supported by the Individual Discovery Grant RGPIN-417269-2013 from National Sciences and Engineering Research Council of Canada (NSERC).

² Partially supported by PoPRA project funded by Conseil Régional d'Aquitaine and European FEDER and IdEx Bordeaux.

³ Supported by the Individual Discovery Grant RGPIN-2016-04576 from National Sciences and Engineering Research Council of Canada (NSERC).



1 Introduction

The profusion of circular molecular DNA sequences (plasmids, mitochondrial DNA, bacterial chromosomes, etc.) and the need to compare them have opened the way to the elaboration of time efficient algorithms for the alignment problem of circular words. The first efficient algorithm for this problem was presented by Maes in 1990 [8] and uses a divide-and-conquer approach to find the edit or Levenshtein distance [6] between two circular words of lengths m and n in $\mathcal{O}(nm \log(n))$ time. It is based on the classical Wagner and Fischer algorithm that finds an alignment and edit distance between two words [12]. Since then, more time efficient algorithms have been developed, using dynamic programming [3], branch-and-bound techniques [9], exploring either approximate or suboptimal solutions [1, 10] or alignment-free methods [2, 4]. In this article, we focus on a related problem which is to find a constrained circular word as close as possible of a given circular word according to the Manhattan distance. This problem comes from currently explored brachytherapy techniques.

Brachytherapy is a form of internal radiotherapy involving short distance (*brachys* in Greek) irradiation. A treatment is performed by placing small sealed radiation sources, also called *seeds*, near or into the tumor site [11]. The seed is moved step by step through a catheter, to irradiate the whole tumor. The radiation dose received at each point depends on the time spent by the seed at this position. The resulting irradiation field is uniform around the seed, regardless of the shape and position of the tumor. Therefore, the use of shields is conceptually considered to modulate more precisely this field. A cylindrical shield encapsulating the seed would block radiation, except through some prescribed openings. Let us associate each stopping position of the seed with the prescription doses for its surrounding circular area. This area of interest is divided, as precisely as possible, in n equal sections. The dose for each section is either a positive integer x (tumor tissue; irradiation prescribed for x units of time) or a 0 (healthy tissue; no irradiation needed) so that each stopping position of the seed is represented by a circular integer word w of length n .

Let us imagine a process, such as 3D printing, allowing us to easily produce customized cylindrical shields for brachytherapy. For a given tumor, we want to irradiate during a selected time through an adequate cylindrical shield, such that each tissue section receives a radiation dose as close as possible to its prescription. Moreover, let us assume that the physical precision of our process is limited: the smallest possible closed (resp. open) sector of a produced shield is still covering several sections of the surrounding area. Then, our algorithmic problem can be formalized as follows: given a circular integer word w of length n , the cylindrical shield to be designed can be seen as a constrained circular binary word of length n where, when we replace each 1 by the selected irradiation time t , the Manhattan distance to w is minimal.

This article is organized as follows. In Section 2, we introduce the notation and mathematically describe our problem. Section 3 is dedicated to computing the minimal distance for a fixed irradiation time, using a dynamic programming algorithm. In Section 4 we provide an algorithm to compute the optimal solutions of the problem. Section 5 considers a particular case of the problem where overdoses are forbidden. Finally, we give perspectives for future work in Section 6.

2 Preliminaries

We now introduce the usual notation on words, as well as a formal definition of the considered problem.

2.1 Words

We briefly introduce the basic terminology on words. More details can be found in [7]. An *alphabet* is a non-empty set Σ ; its elements are called *letters*. It is called *binary* if $|\Sigma| = 2$. A *word* $w = w_1 \cdots w_n$ over Σ is a finite sequence of letters of Σ , where w_i is the i -th letter of w . The *empty word* is denoted by ε . We denote by Σ^* the set of all words over Σ . A *language* L over Σ is a subset of Σ^* .

The *length* of a word w is the number of its letters and is denoted by $|w|$. For a positive integer n , we denote by Σ^n the set of all words of length n over Σ .

Given a word $y \in \Sigma^*$, we say that y is a *factor* of $w \in \Sigma^*$ if there exist two words $x, z \in \Sigma^*$ such that $w = xyz$. In particular, if $x = \varepsilon$ (resp. $z = \varepsilon$), then y is called a *prefix* (resp. *suffix*) of w . For two words w and u , let $\text{LCP}(w, u)$ (resp. $\text{LCS}(w, u)$) denote the *longest common prefix* (resp. *suffix*) of w and u . We denote by $\text{Pref}(w)$ the *set of all prefixes* of a word w . For an integer i , we write $\text{pref}_i(w)$ (resp. $\text{suff}_i(w)$) the *length- i prefix* (resp. *suffix*) of w .

Given an integer $n \geq 0$, we denote by w^n the word $ww \cdots w$ (n times). If $w = xyz$ and $y = a^n$, for some letter $a \in \Sigma$ and some integer $n \geq 1$, we say that y is an *a -run* in w . In the sequel, each a -run y considered is *maximal*, that is, the last letter of x and the first one of z are both different from a . Let $P \subset \Sigma^*$ be a finite set of words over Σ . A language $L \subseteq \Sigma^*$ is said to *avoid* P if, for any $w \in L$ and any $y \in P$, y is not a factor of w .

Two words w and w' are *conjugate*, denoted by $w \sim w'$, if there exist two words x and y such that $w = xy$ and $w' = yx$. For $w = w_1 \cdots w_n \in \Sigma^n$, the j -th conjugate of w , for $1 \leq j \leq n$, is the word $w_j \cdots w_n w_1 \cdots w_{j-1}$. It is easy to verify that the relation \sim is an equivalence relation. Any equivalence class of \sim is called a *circular word*. For instance, $\{01001, 10010, 00101, 01010, 10100\}$ is a circular word.

2.2 Problem definition

For a given tumor, we want to design a cylindrical shield with openings, both enabling us to apply the most accurate treatment possible to the patient and taking into account the production process limitations. Assuming that the irradiation time is t , we represent this shield by a circular word over the alphabet $\{0, t\}$, where each t stands for an opened section and each 0 for a closed section. Let ℓ_1 be the minimal length for an opening, and ℓ_0 the one for a closed sector between two openings. Then, in any word representing a producible shield, each 0-run (resp. t -run) must be of length at least ℓ_0 (resp. ℓ_1). Thus, any such word belongs to the following language of admissible words.

► **Definition 1.** For three positive integers ℓ_0, ℓ_1, t , let A_t be the language over the alphabet $\{0, t\}$ that avoids $t0^{m_0}t$ and $0t^{m_1}0$ for all $0 < m_0 < \ell_0$, $0 < m_1 < \ell_1$. We say that a word u is *t -admissible* if $u \in A_t$.

To reduce the amount of notation, we do not write ℓ_0, ℓ_1 as indices in Definition 1 since ℓ_0, ℓ_1 are fixed. So we write A_t instead of $A_{\ell_0, \ell_1, t}$. Moreover, from now on, we fix $\ell = \max\{\ell_0, \ell_1\}$.

► **Example 2.** Let $\ell_0 = 3$, $\ell_1 = 5$ and $t = 2$. We have $02222200022 \in A_t$, but the word 02222200222 is not t -admissible since it contains the factor 20^22 and $\ell_0 = 3$.

The doses applied through a shield depend on the associated irradiation time t . Therefore, we select the most accurate shield for each relevant value of t .

► **Definition 3.** Let ℓ_0, ℓ_1, t be three positive integers. A word $u \in A_t$ is said to be *t -circularly admissible* if all of its conjugates are in A_t . The set of all t -circularly admissible words is denoted by C_t .

■ **Table 1** Comparison of the results of a naive algorithm (u_{greedy} and $u_{\text{greedy\&circ}}$) with an optimal solution (u_{optimal}) for $\ell = \ell_0 = \ell_1 = 3$ and $w = 111100111000$. Each difference with w is underlined. The last two runs of u_{greedy} are merged into a 0-run in $u_{\text{greedy\&circ}}$. Only u_{optimal} is a proper solution.

| w | 1 | 1 | 1 | 1 | 0 | 0 | 1 | 1 | 1 | 0 | 0 | 0 | $d(u, w)$ | $u \in C_1 ?$ | $u \in \text{Sol}(w) ?$ |
|---------------------------|---|---|---|----------|---|---|----------|----------|----------|----------|---|---|-----------|---------------|-------------------------|
| u_{greedy} | 1 | 1 | 1 | 1 | 0 | 0 | <u>0</u> | 1 | 1 | <u>1</u> | 0 | 0 | 2 | | |
| $u_{\text{greedy\&circ}}$ | 1 | 1 | 1 | 1 | 0 | 0 | <u>0</u> | <u>0</u> | <u>0</u> | 0 | 0 | 0 | 3 | ✓ | |
| u_{optimal} | 1 | 1 | 1 | <u>0</u> | 0 | 0 | 1 | 1 | 1 | 0 | 0 | 0 | 1 | ✓ | ✓ |

► **Example 4.** Let $\ell_0 = 3$, $\ell_1 = 5$ and $t = 2$. We have $02222200022 \in A_t$ but it is not t -circularly admissible since its conjugate 22222000220 is not t -admissible. We have $02222200000 \in C_t$.

At last, we use a distance metric to assess the accuracy of the considered treatments.

► **Definition 5.** The *Manhattan distance* d between two words u and $v \in \mathbb{N}^n$ is $d(u, v) = \sum_{i=1}^n |u_i - v_i|$, where $|u_i - v_i|$ denotes the absolute value of $u_i - v_i$.

Our problem can formally be stated as follows.

► **Problem 1.** Given a word w over the alphabet \mathbb{N} and two integers ℓ_0, ℓ_1 , find $t \in \mathbb{N}^+$ and $u \in C_t$ that minimizes $d(u, w)$.

Note that a solution to Problem 1 always exists (since $u = 0^{|w|}$ is t -circularly admissible for any t), but is not necessarily unique. Therefore, we set the following notation. Let us denote the set of all solutions to Problem 1 for w by

$$\text{Sol}(w) = \{u \in \cup_t \text{Sol}_t(w) \mid d(u, w) \text{ is minimal}\},$$

where $\text{Sol}_t(w) = \{u \in C_t \mid d(u, w) \text{ is minimal}\}$. Observe that if $\ell = \max\{\ell_0, \ell_1\} \leq 1$, Problem 1 is trivial. Hence, in practice, we always consider $\ell \geq 2$.

Also, we always assume in the following that $|w| \geq \ell_0 + \ell_1$. Indeed, the case where $|w| = n < \ell_0 + \ell_1$ is also trivial since any solution for such a word w is either 0^n or t^n .

It is worth mentioning that the following naive, greedy strategy fails to solve Problem 1. Let us consider the simple case where w is a binary word, and hence $t = 1$. Let u_{greedy} be the word obtained by reading through w and beginning a new 0-run (resp. 1-run) as soon as both a 0 (resp. 1) occurs in w and the length of the previous 1-run (resp. 0-run) is at least ℓ_1 (resp. ℓ_0). Else, the last created run is extended. Then, there exist instances of Problem 1 such that $u_{\text{greedy}} \notin C_1$, hence is not a proper solution. For such an instance, let $u_{\text{greedy\&circ}} \in C_1$ be the word obtained by merging the last two runs of u_{greedy} into one run of either 0 or 1 (the one giving the lowest distance to w). An instance such that neither u_{greedy} nor $u_{\text{greedy\&circ}}$ minimizes the distance with w is presented in Table 1.

In what follows, we show that, in practice, Problem 1 can be solved by a polynomial dynamic programming algorithm.

3 Computing the optimal distance for a fixed irradiation time

To solve Problem 1 for a given w , we first fix the irradiation time t and compute the optimal distance for this fixed t . In other words, we compute $d(u, w)$ for some $u \in \text{Sol}_t(w)$.

3.1 Distance matrix

Let $\ell, \ell_0, \ell_1, n, t$ be positive integers such that $\ell = \max\{\ell_0, \ell_1\}$ and $n \geq \ell_0 + \ell_1$. Let w be a word of length n over the alphabet \mathbb{N} . In this section, we describe an algorithm that computes the set $\text{Sol}_t(w)$ for a fixed irradiation time t .

To compute the set $\text{Sol}_t(w)$, we build a dynamic matrix using the possible prefixes of $u \in \text{Sol}_t(w)$, that is, the set

$$\mathcal{P}_t = \{0^i t \mid 1 \leq i \leq \ell_0\} \cup \{t^i 0 \mid 1 \leq i \leq \ell_1\} \cup \{0^{\ell_0+1}, t^{\ell_1+1}\}.$$

In particular, for any word $u \in A_t$, $|\mathcal{P}_t \cap \text{Pref}(u)| = 1$, i.e., there exists a unique word $p \in \mathcal{P}_t$ such that p is a prefix of u . Note that $|\mathcal{P}_t| = \ell_0 + \ell_1 + 2 \leq 2\ell + 2$.

► **Example 6.** For $\ell_0 = 3$ and $\ell_1 = 5$, $\mathcal{P}_t = \{0t, 00t, 000t, 0000, t0, tt0, ttt0, tttt0, ttttt0, tttttt\}$.

The following definition generalizes the notion of t -circularly admissible words, to describe the (shorter) words yet to be extended into ones.

► **Definition 7.** Let $n \in \mathbb{N}$, $i \in \{1, \dots, n\}$, $p \in \mathcal{P}_t$ and $v \in A_t \cap \{0, t\}^\ell$. Then a word u is called t -circularly preadmissible (with respect to p and v) if it satisfies the following properties:

- (i) (admissibility) $u \in A_t$ and $|u| = i$;
- (ii) (circular extendability) There exists a word x such that $ux \in C_t$;
- (iii) (prefix compatibility) $\text{LCP}(u, p) \in \{u, p\}$;
- (iv) (suffix compatibility) $\text{LCS}(u, v) \in \{u, v\}$.

The set of t -circularly preadmissible words of length i with respect to p and v is denoted by $C_{t,p}(v, i)$.

Roughly speaking, $u \in C_{t,p}(v, i)$ if u is t -admissible, starts with p , ends with v (whenever u is long enough) and can be extended into at least one t -circularly admissible word. In particular, we need to consider the longest common prefix (resp. suffix) of u and p (resp. u and v) to take care of the cases where u is shorter than p (resp. shorter than v). Note that each word of \mathcal{P}_t is circularly preadmissible since $n \geq \ell_0 + \ell_1$.

We now define a matrix whose entries indicate the minimum distance between increasingly longer prefixes u' of both w and u , so that u' is preadmissible with respect to its unique prefix $p \in \mathcal{P}_t$. The columns of the matrix correspond to the length of these prefixes while the rows correspond to the possible suffixes of these prefixes.

► **Definition 8 (Distance matrix).** For $p \in \mathcal{P}_t$, let $D_{w,t,p}$ be the matrix of size $|A_t \cap \{0, t\}^\ell| \times n$ such that

$$D_{w,t,p}[v, i] = \min\{d(u, w_1 \cdots w_i) \mid u \in C_{t,p}(v, i)\}.$$

for each $v \in A_t \cap \{0, t\}^\ell$ and each $i \in \{1, \dots, n\}$, with the convention that $\min \emptyset = \infty$.

► **Example 9.** Consider for instance, the word $w = 013331102230313210$ with $\ell_0 = 3$ and $\ell_1 = 5$. Let $t = 2$ and $p = 02$. Figure 3 depicts the matrix $D_{w,t,p}$.

Note that the size of the matrix $D_{w,t,p}$, which is equal to $|A_t \cap \{0, t\}^\ell| \times n$, is polynomial with respect to ℓ and n . This comes from the following lemma.

► **Lemma 10.** The language $A_t \cap \{0, t\}^\ell$ contains $2\ell + \binom{\ell - \min\{\ell_0, \ell_1\}}{2}$ words.

Proof. Let $v \in A_t \cap \{0, t\}^\ell$. By Definition 1, v does not contain $t0^{m_0}t$ and $0t^{m_1}0$ for all $0 < m_0 < \ell_0, 0 < m_1 < \ell_1$. Recall that $\ell = \max\{\ell_0, \ell_1\}$. If $\ell_0 = \ell_1 = \ell$, the fact that $|v| = \ell$ implies that v either is a 1-run followed by a 0-run or vice-versa. Since there are ℓ positions where this first run can end, this gives us ℓ words beginning by 1 and ℓ words beginning by 0. Now, if $\ell = \ell_0 > \ell_1$, v can also begin and end with a 0-run, and have a 1-run of length at least ℓ_1 in the middle. In this case, we need to decide where the first 0-run ends and where the last 0-run begins. Hence we have $\ell - \ell_1$ possible positions to choose from. The case $\ell = \ell_1$ is symmetric and so we have that the language $A_t \cap \{0, t\}^\ell$ contains $2\ell + \binom{\ell - \min\{\ell_0, \ell_1\}}{2}$ words. ◀

Moreover, it follows from Definition 8 that the matrices $D_{w,t,p}$ keep track of the optimal values. More precisely, let

$$D_{w,t} = \min\{D_{w,t,p}[v, n] \mid p \in \mathcal{P}_t, v \in A_t \cap \{0, t\}^\ell\}.$$

Then, the next theorem states that any word u such that $d(u, w) = \min_t D_{w,t}$ is a solution to Problem 1. Indeed, Definition 7 implies that all prefixes of a circularly admissible word u are preadmissible.

► **Theorem 11.** *Let $u \in \{0, t\}^n$ and $i \in \{1, \dots, n\}$. If $u \in C_t$, then there exist $p \in \mathcal{P}_t$ and $v \in A_t \cap \{0, t\}^\ell$ such that $\text{pref}_i(u) \in C_{t,p}(v, i)$. Consequently, for any $u \in \text{Sol}_t(w)$,*

$$d(u, w) = \min\{D_{w,t,p}[v, n] \mid v \in A_t \cap \{0, t\}^\ell, p \in \mathcal{P}_t\}.$$

Proof. Assume first that $u \in C_t$. Let p be the only word in $\mathcal{P}_t \cap \text{Pref}(u)$ and $v \in A_t \cap \{0, t\}^\ell$ such that $\text{LCS}(u, v) \in \{\text{pref}_i(u), v\}$. For $i \geq \ell$, it is obvious that such a v always exists since $u \in C_t$ implies $\text{pref}_i(u) \in A_t$. For $i < \ell$, let a be the first letter of u and $v = a^{\ell-i} \text{pref}_i(u)$. Then, all the conditions of Definition 7 are satisfied and $\text{pref}_i(u) \in C_{t,p}(v, i)$.

For the last part of the statement, we only need to show that $u \in C_{t,p}(v, n)$ for some $v \in A_t \cap \{0, t\}^\ell$ and $p \in \mathcal{P}_t$ implies $u \in C_t$. Assume that $u \in C_{t,p}(v, n)$ for adequate p and v . Thus, u is admissible and $|u| = n$, so that $u \cdot \varepsilon = u \in C_t$, by the circular extendability property. The result then follows from the definition of $\text{Sol}_t(w)$. ◀

In the next subsection, we show that, for a fixed t , for all $p \in \mathcal{P}_t$, $v \in A_t$ and $i \in \{|p| + 1, \dots, n\}$, the value $D_{w,t,p}[v, i]$ can be computed in constant time from at most two values of the form $D_{w,t,p}[v', i - 1]$, where $v' \in A_t$.

3.2 Dynamic programming

We describe now the dynamic computation of such matrices $D_{w,t,p}$. We begin by taking care of the initialization.

► **Lemma 12.** *Let $p \in \mathcal{P}_t$ and $D = D_{w,t,p}$. For $v \in A_t \cap \{0, t\}^\ell$ and $1 \leq i \leq |p|$, we have*

$$D[v, i] = \begin{cases} d(\text{pref}_i(w), \text{pref}_i(p)), & \text{if } \text{LCS}(v, \text{pref}_i(p)) \in \{v, \text{pref}_i(p)\}; \\ \infty, & \text{otherwise.} \end{cases}$$

Proof. We first prove that $u \in C_{t,p}(v, i)$ implies $u = \text{pref}_i(p)$. Indeed, assume that there exists some $u \in C_{t,p}(v, i)$. Then, by Definition 7, we have in particular that u is admissible, $|u| = i$ and $\text{LCP}(u, p) \in \{u, p\}$. Since $i \leq |p|$, we deduce that $\text{LCP}(u, p) = u$, i.e. $u = \text{pref}_i(p)$. Therefore, either $C_{t,p}(v, i) = \{\text{pref}_i(p)\}$ or $C_{t,p}(v, i) = \emptyset$.

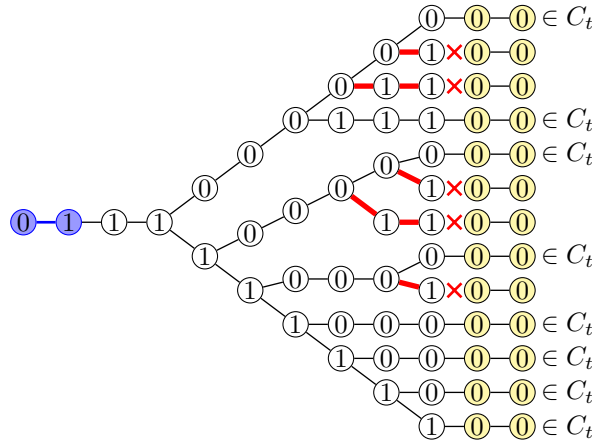


Figure 1 Circularly admissible words of length 12 with $\ell_0 = \ell_1 = 3$, $t = 1$ that have $p = 01$ as prefix. They are constructed by successive extensions of admissible words complying with the prefix. In blue is highlighted the prefix p ; in yellow is the suffix of any such circularly admissible words; and in red are the discarded extensions.

It remains to check under which conditions $\text{pref}_i(p) \in C_{t,p}(v, i)$. Clearly, $\text{pref}_i(p)$ is admissible, since p is, and $|\text{pref}_i(p)| = i$. Also, note that $\text{pref}_i(p)$ is circularly extendable, since $p \in \mathcal{P}_t$ is circularly extendable whenever $n \geq \ell_0 + \ell_1$. The prefix compatibility is trivially verified since $\text{LCP}(\text{pref}_i(p), p) = \text{pref}_i(p)$. Therefore, if $\text{LCS}(v, \text{pref}_i(p)) \in \{v, \text{pref}_i(p)\}$, then $C_{t,p}(v, i) = \{\text{pref}_i(p)\}$ and the result follows. Otherwise, $C_{t,p}(v, i) = \emptyset$. ◀

To compute the other part of the table, we need to focus on words that are admissible for w and that comply with the given prefix $p \in \mathcal{P}_t$. Near the end of the table, we need to add extra conditions to take into account the fact that suffixes of the circularly admissible words are prescribed by p .

Example 13. Let us consider the word $w = 001100001111$, $t = 1$, $\ell_0 = \ell_1 = 3$ and the prefix $p = 01 \in \mathcal{P}_t$. Figure 1 depicts all words considered in the computation of $D_{w,t,p}$, that is, all admissible words that can be extended to a circularly admissible word complying with the prefix p . This prefix condition is highlighted in blue. Starting from the left, we write the possible extensions of an admissible word of length i to one of length $i + 1$. Observe that this rule of extension sometimes leads to dead ends. Indeed, to be circularly admissible, any admissible word of length $|w|$ has to end with two 0's since $\ell_0 = 3$ and the prescribed prefix $p = 01$ begins with only one 0 (this condition is highlighted in yellow). Therefore, the extensions highlighted in red are not prefixes of any circularly admissible words, so that such words need to be discarded.

In Theorem 14, the yellow condition corresponds to the case $n - c_p < i \leq n$ and the red dead ends are taken care of in the case $n - c_p - \ell_a < i \leq n - c_p$. In the remaining case, Equation (1) translates the fact that any admissible word is the extension of a shorter admissible word.

Theorem 14. Let $p = p_1 \dots p_k \in \mathcal{P}_t$ and $D = D_{w,t,p}$. If $p_1 = 0$, we set $c_p = \ell_0 - |p| + 1$, $\bar{p}_1 = t$ and $a = 1$. Otherwise, we set $c_p = \ell_1 - |p| + 1$, $\bar{p}_1 = 0$ and $a = 0$. For $v = v_1 \dots v_\ell \in A_t$ and $|p| + 1 \leq i \leq n$, we have

$$D[v, i] = d(w_i, v_\ell) + \min\{D[v', i - 1] \mid v' \in V'\}$$

where

$$V' = \begin{cases} \emptyset, & \text{if } n - c_p < i \leq n \text{ and } \text{suff}_{i-n+c_p}(v) \notin p_1^* \\ \text{or if } n - c_p - \ell_a < i \leq n - c_p, \text{ suff}_{i-(n-c_p-\ell_a)}(v) \notin (\overline{p_1})^* \text{ and } \text{suff}_1(v) \neq p_1; \end{cases}$$

and otherwise

$$V' = \begin{cases} \{v' \in \{0v_1 \dots v_{\ell-1}, tv_1 \dots v_{\ell-1}\} \mid v'v_\ell \in A_t \cap \{0, t\}^{\ell+1}\}, & \text{if } i > \ell; \\ \{(v_{\ell-i+1})^{\ell-i+2} v_{\ell-i+2} \dots v_{\ell-1}\}, & \text{if } i \leq \ell. \end{cases} \quad (1)$$

Proof. By Definition 8, $D[v, i]$ is the minimum distance $d(u, w_1 \dots w_i)$ over words $u \in C_{t,p}(v, i)$. We begin by focusing on the circular extendability of such words (Condition (ii) in Definition 7).

For any circularly admissible word $x \in C_t$, if x has prefix p with first letter p_1 , then x must end with a suffix $p_1^{c_p}$ since p begins with $|p| - 1$ times the letter p_1 and the constant $c_p = \ell_p - (|p| - 1)$. Therefore, for $n - c_p < i \leq n$, any $u \in C_{t,p}(v, i)$ is the prefix of a circularly admissible word and it must end with $p_1^{i-n+c_p}$. Thus, if $\text{suff}_{i-n+c_p}(v) \notin p_1^*$, then $C_{t,p}(v, i) = \emptyset$ and $D[v, i] = \infty$. In this case, we set $V' = \emptyset$.

Consider now the case where $n - c_p - \ell_a < i \leq n - c_p$. Any circularly admissible word $x \in C_t$ which has prefix p , has suffix $p_1^{c_p}$. So the suffix s of length $c_p + \ell_a$ of x ends with c_p times the letter p_1 . Since s has to be admissible, the factor $p_1 \overline{p_1}$ can not occur in s . In particular, it can not occur in position in $\{n - c_p - \ell_a + 1, \dots, n - c_p\}$ in x . Thus, $u \in C_{t,p}(v, i)$ implies that u has a suffix of the form $(\overline{p_1})^{i-(n-c_p-\ell_a)}$ or u ends with p_1 . Hence, if $\text{suff}_{i-(n-c_p-\ell_a)}(v) \notin (\overline{p_1})^*$ and $\text{suff}_1(v) \neq p_1$, then $C_{t,p}(v, i) = \emptyset$ and $D[v, i] = \infty$. So we set $V' = \emptyset$ in this case too.

Note that when $|p| < i \leq n - c_p - \ell_a$, then any word u satisfying Conditions (i,iii,iv) of Definition 7 always satisfies Condition (ii). Indeed, if u ends with p_1 , then we can concatenate u and p_1^{n-i} to obtain a circularly admissible word. If u ends with $\overline{p_1}$, we can concatenate u with $(\overline{p_1})^{\ell_a} p_1^{n-i-\ell_a}$ to get a circularly admissible word since $c_p \leq n - i - \ell_a$.

We now turn our attention to the other cases where Condition (ii) is always satisfied. We assume now that i and v are such that one of the following holds:

- $|p| < i \leq n - c_p - \ell_a$;
- $n - c_p - \ell_a < i \leq n - c_p$ and $(\text{suff}_{i-(n-c_p-\ell_a)}(v) \in (\overline{p_1})^* \text{ or } \text{suff}_1(v) = p_1)$;
- $n - c_p < i \leq n$ and $\text{suff}_{i-n+c_p}(v) \in p_1^*$.

We discuss two cases according to whether $i \leq \ell$ or $i > \ell$.

Firstly, if $i \leq \ell$, then $u \in C_{t,p}(v, i)$ implies that u is admissible such that $\text{LCS}(u, v) = u$ and $\text{LCP}(u, p) = p$ (as $i \geq |p| + 1$). In particular, u has a prefix $\text{pref}_{i-1}(u)$ that belongs to $C_{t,p}(v', i - 1)$ for some $v' \in A_t \cap \{0, t\}^\ell$. We claim that we can always choose v' such that $v'v_\ell \in A_t$. Indeed, v' has suffix $\text{pref}_{i-1}(u)$ and $\text{pref}_{i-1}(u)$ begins with at least $|p| - 1$ times the letter p_1 . Therefore, we can choose v' to be the length- ℓ word of $p_1^* \text{pref}_{i-1}(u)$. Since $u \in A_t$, we have $\text{pref}_{i-1}(u)v_\ell \in A_t$ and $v'v_\ell \in A_t$. As $p_1 = v_{\ell-i+1}$ and $\text{pref}_{i-1}(u) = v_{\ell-i+1} \dots v_{\ell-1}$, the result follows. Similarly if $C_{t,p}(v, i) = \emptyset$, then $C_{t,p}(v', i) = \emptyset$ for the length- ℓ word $v' \in v_{\ell-i+1}^* v_{\ell-i+1} \dots v_{\ell-1}$. Hence, the equation $\infty = D[v, i] = d(v_\ell, w_i) + D[v', i - 1] = \infty$ holds also in this case.

Thus, we set $V' = \{v' \in \{0, t\}^{\ell-i+1} \text{pref}_{i-1}(u) \mid v'v_\ell \in A_t \cap \{0, t\}^{\ell+1}\}$ and we have

$$\begin{aligned} D[v, i] &= \min\{d(u, w_1 \dots w_i) \mid u \in C_{t,p}(v, i)\} \\ &= d(v_\ell, w_i) + \min\{d(\text{pref}_{i-1}(u), w_1 \dots w_{i-1}) \mid u \in C_{t,p}(v, i)\} \\ &= d(v_\ell, w_i) + \min\{d(u', w_1 \dots w_{i-1}) \mid u' \in C_{t,p}(v', i - 1) \text{ s.t. } v' \in V'\} \end{aligned}$$

and the result follows.

Algorithm 1 Finding an optimal solution

```

1: function FINDSOLUTION( $w$  : integer word,  $\ell_0, \ell_1$ : integers): integer word
2:   Let  $t_{\max}$  be the largest letter occurring in  $w$ 
3:    $\ell \leftarrow \max\{\ell_0, \ell_1\}$ ;  $m \leftarrow \min\{\ell_0, \ell_1\}$ 
4:   Let  $D_{\min}$  be a table of size  $2\ell + \binom{\ell-m}{2} \times |w|$ 
5:    $d_{\min} \leftarrow +\infty$ 
6:   for  $t \leftarrow 1, \dots, t_{\max}$  do  $\triangleright t_{\max}$  iterations
7:     for  $p \in \mathcal{P}_t$  do  $\triangleright \mathcal{O}(\ell)$  iterations
8:        $D \leftarrow \text{COMPUTEMATRIX}(w, t, p, \ell_0, \ell_1)$   $\triangleright \mathcal{O}(|w| \cdot \ell^2)$ 
9:       for  $v \in A_t \cap \{0, t\}^\ell$  do  $\triangleright \mathcal{O}(\ell^2)$  iterations
10:         $dist \leftarrow D[v, |w|]$ 
11:        if  $dist < d_{\min}$  then
12:           $(optimal, time, prefix, d_{\min}, D_{\min}) \leftarrow (v, t, p, dist, D)$ 
13:        end if
14:      end for
15:    end for
16:  end for
17:   $u \leftarrow \text{BACKTRACKING}(D_{\min}, v, time, prefix)$   $\triangleright \mathcal{O}(|w|)$ 
18:  return  $u$ 
19: end function

```

Secondly, if $i > \ell$, then $u \in C_{t,p}(v, i)$ implies that $u \in A_t \cap \{0, t\}^i$, $\text{LCP}(u, p) = p$ and $\text{LCS}(u, v) = v$. In particular u ends either with $0v_1 \dots v_\ell$ or with $tv_1 \dots v_\ell$ where $v = v_1 \dots v_\ell$. As $u \in A_t$, its suffix of length $\ell + 1$ belongs to A_t . Therefore we have $D[v, i] = d(w_i, v_\ell) + \min\{D[v', i-1]\}$ where $v' \in \{0v_1 \dots v_{\ell-1}, tv_1 \dots v_{\ell-1}\}$ such that $v'v_\ell \in A_t \cap \{0, t\}^{\ell+1}$ as required. \blacktriangleleft

4

 Computing an optimal solution

Now, given any circular word with representative w , we are ready to solve Problem 1, *i.e.* to give an algorithm to find a word $u \in \text{Sol}(w)$. For each irradiation time t , we compute the optimal distance $d(u, w)$ for $u \in \text{Sol}_t(w)$. We consider the minimum value obtained while t is varying and we store the distance matrix associated to this minimum value (Algorithm 1).

As for the complexity of the algorithm, note that line 8 is computed using Lemma 12 and Theorem 14 in $\mathcal{O}(|w| \cdot \ell^2)$ since $|A_t \cap \{0, t\}^\ell| = 2\ell + \binom{\ell - \min\{\ell_0, \ell_1\}}{2}$. This gives us a pseudo polynomial time algorithm of complexity $\mathcal{O}(|w| \cdot t_{\max} \cdot \ell^3)$ to compute an optimal distance. An important note is that since t_{\max} represents, in our context, the maximal time of an irradiation, it should be kept as small as possible and in fact be a lot smaller than $|w|$. So, in practice, we have a polynomial time algorithm. Finally, we obtain an optimal solution by backtracking in $\mathcal{O}(|w|)$ with Algorithm 2.

A simple experimentation shows (see Figure 2) that, as expected, our algorithm becomes quickly faster than a naive approach, which enumerates all the words in A_t of length $|w|$, checks whether they are circularly admissible and computes their distance from w , and which is exponential.

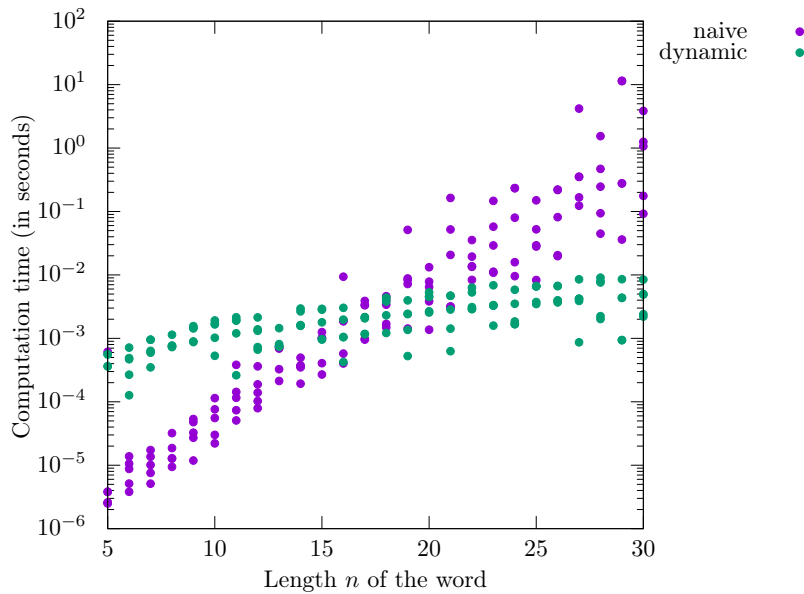
► Example 15 (Example 9 continued). Let $w = 013331102230313210$, $\ell_0 = 3$ and $\ell_1 = 5$. For $t \in \{1, 2, 3\}$, let $\mathcal{P}_t = \{0t, 00t, 000t, 0000, t0, tt0, ttt0, tttt0, ttttt\}$. Among the matrices $D_{w,t,p}$, the one with $t = 2$ and $p = 02$ contains the minimum value in its last column and is

Algorithm 2 From distance to solution

```

1: function BACKTRACKING( $D : \text{table}, v = v_1 \dots v_\ell : \text{integer word}, t : \text{integer}, p : \text{integer}$ 
   word): integer word
2:    $n \leftarrow$  number of columns of  $D$ 
3:    $u \leftarrow \lfloor v_\ell \rfloor$ 
4:   for  $i \leftarrow n - 1, \dots, \ell + 1$  do
5:      $x \leftarrow v_1 \dots v_\ell$ 
6:     if  $D[0x_1 \dots x_{\ell-1}, i] \leq D[tx_1 \dots x_{\ell-1}, i]$  then
7:        $x \leftarrow 0x_1 \dots x_{\ell-1}; u \leftarrow \lfloor x_{\ell-1} \rfloor + u$ 
8:     else
9:        $x \leftarrow tx_1 \dots x_{\ell-1}; u \leftarrow \lfloor x_{\ell-1} \rfloor + u$ 
10:    end if
11:  end for
12:  for  $i \leftarrow \ell, \dots, |p| + 1$  do
13:     $x \leftarrow \underbrace{x_{\ell-i+1} \dots x_{\ell-i+1}}_{\ell-i+2 \text{ times}} x_{\ell-i+2} \dots x_{\ell-1}$ 
14:     $u \leftarrow \lfloor x_{\ell-1} \rfloor + u$ 
15:  end for
16:   $u \leftarrow p + u.$ 
17:  return  $u$ 
18: end function

```



■ **Figure 2** Comparison of the efficiency between Algorithm 1 and a naive algorithm, based on the exhaustive enumeration of all circularly admissible factors. The computation times were obtained by generating 5 random conformations for each $n \in \{5, 6, \dots, 30\}$, choosing the parameters (w, t, ℓ_0, ℓ_1) in each case with uniform probability such that $0 \leq w_i \leq 20$ for $i = 1, 2, \dots, n$, $1 \leq t \leq 20$, $2 \leq \ell_0 \leq 5$ and $2 \leq \ell_1 \leq 5$. Although the dynamic programming strategy is initially more costly, it rapidly becomes faster than the naive strategy, as expected.

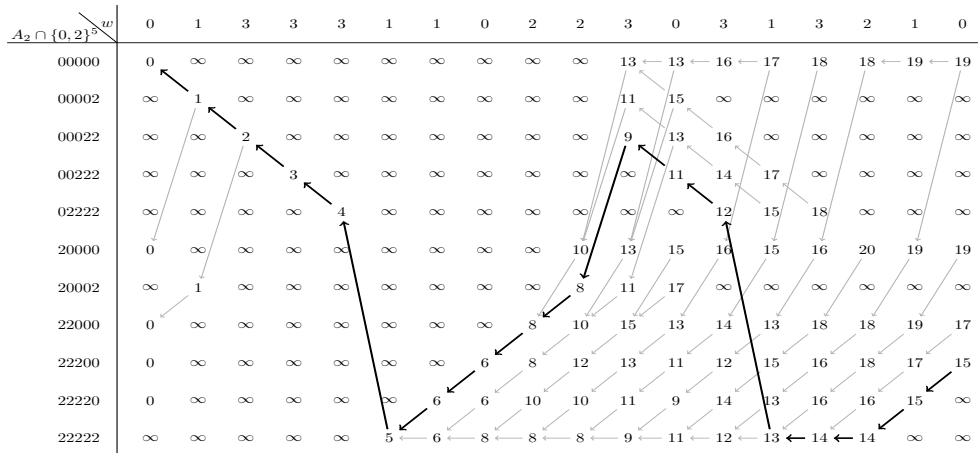


Figure 3 Matrix for the word $w = 013331102230313210$ with $\ell_0 = 3$, $\ell_1 = 5$, $t = 2$ and $p = 02$. An arrow between two cells indicates that the value in the arrival cell is calculated from the one in the origin cell. For ease of reading, the corresponding arrows of cells containing $+\infty$ are not drawn. In bold is a path corresponding to an optimal solution (not necessarily unique) for the irradiation time 2.

represented in Figure 3. The path corresponding to the optimal solution, 022222000222222200 is drawn in bold. Note that it is not the only optimal solution. For instance, we also have $0222222222222222200 \in \text{Sol}(w)$.

5 Forbidding overdoses

Let us now consider a subproblem by imposing an additional condition: no tissue section can receive a dose greater than the one prescribed. That is, *overdoses* are forbidden. We then need the following stronger condition on our previous admissible words. By abuse of notation, and for the sake of simplicity, we use, in what follows, the same notation as in Sections 2 and 3.

Let ℓ, ℓ_0, ℓ_1, n be positive integers such that $\ell = \max\{\ell_0, \ell_1\}$ and $n \geq \ell_0 + \ell_1$. Let w be a word of length n over the alphabet \mathbb{N} . In this section, we show that the algorithm presented in Section 4 can also be used here, but with more efficiency, when we replace Definitions 1 and 3 by the following one.

► **Definition 16.** Let us write $w = w_1 \cdots w_n$ with $w_i \in \mathbb{N}$. Let t be a letter in \mathbb{N} . A word $u \in \mathbb{N}^n$ is *t-admissible* for w if $u \in A_t$ and $u_i \leq w_i$ for $i = 1, \dots, n$. The set of all *t-admissible* words for w is denoted by $A_t(w)$. In addition, if $u \in C_t$, then u is said to be *t-circularly admissible* for w . The set of all *t-circularly admissible* words for w is denoted by $C_t(w)$.

This stronger requirement affects the values of the distance matrices. We have $D_{w,t,p}[v, i] = \infty$ if the last letter of v is greater than the i -th letter w_i of w . Otherwise, $D_{w,t,p}[v, i]$ is computed according to Lemma 12 and Theorem 14.

By abuse of notation, we denote the set of all solutions to Problem 1 for w , under these new conditions, by $\text{Sol}(w) = \{u \in \cup_t \text{Sol}_t(w) \mid d(u, w) \text{ is minimal}\}$, where $\text{Sol}_t(w) = \{u \in C_t(w) \mid d(u, w) \text{ is minimal}\}$ as before.

The “overdose-free” condition gives us the opportunity to do an efficient preprocessing of w and have a smaller set of valid prefixes \mathcal{P}_t which produces a significant gain in time

complexity. Firstly, the “overdose-free” and ℓ_1 constraints imply that we cannot irradiate specific sections of w . So we apply the following preprocessing on w that does not affect the solutions of Problem 1.

► **Preprocessing.** The first step is to set to 0 in w all letters that are smaller than t . Then all non-zero factors⁴ that are too short, *i.e.* of length less than ℓ_1 , are replaced by 0-runs.

Secondly, since it is best to work with words beginning with a 0-run and the preprocessed w clearly does not always have that property, we need the following trivial lemma showing that optimal solutions to Problem 1 are preserved by conjugacy.

► **Lemma 17.** *Let t be a positive integer and $u \in \text{Sol}_t(w)$. Let v be the j -th conjugate of u and w' be the j -th conjugate of w , for an integer j such that $1 \leq j \leq n$. Then $v \in \text{Sol}_t(w')$.*

Proof of Lemma 17. We have $u \in \text{Sol}_t(w)$. Then, $d(u, w) = d(u_1, w_1) + \dots + d(u_n, w_n)$ is minimal. Since v is the j -th conjugate of u and w' be the j -th conjugate of w we have that $v = u_j \dots u_n u_1 \dots u_{j-1}$ and $w' = w_j \dots w_n w_1 \dots w_{j-1}$, which gives us $d(v, w') = d(u_j, w_j) + \dots + d(u_n, w_n) + d(u_1, w_1) + \dots + d(u_{j-1}, w_{j-1}) = d(u, w)$, which is minimal. So $v \in \text{Sol}_t(w')$. ◀

As a consequence, we can choose to work with a particular conjugate of the preprocessed input. If it contains at least one 0, let $j \in \mathbb{N}$ be such that the j -th conjugate w' of w begins with a 0-run of length x , where x is maximal. The “overdose-free” condition implies that the possible prefixes of a solution $u' \in \text{Sol}_t(w')$ begin with a 0-run of length at least x . So we can now compute an optimal solution u' for this w' using Algorithm 1 with the smaller set of prefixes $\mathcal{P}_t = \{0^i t \mid x \leq i \leq \ell_0\} \cup \{0^{\ell_0+1}\}$ of length $|\mathcal{P}_t| = \ell_0 - x + 2$. Then, our optimal solution $u \in \text{Sol}_t(w)$ is obtained by taking the $(n - j + 2)$ -th conjugate of u' . Finally, if the preprocessed input w does not contain any 0, then $\text{Sol}_t(w) = \{t^n\}$.

Though the complexity in the worst case is not improved, in many cases the number of distance matrices to be computed is significantly lowered. For instance, the following example shows that instead of 30 distance matrices needed to find the optimal solution in the original settings, only one distance matrix is needed for the “overdose-free” problem.

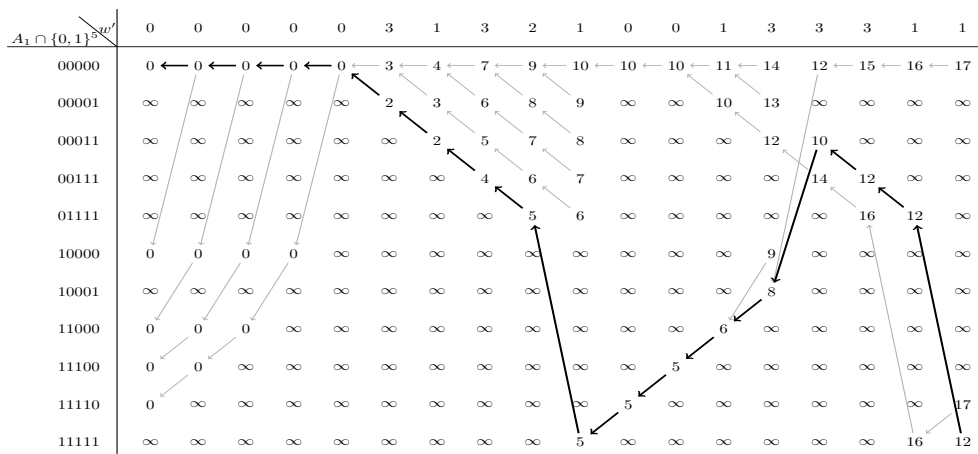
► **Example 18** (Example 9 continued). For the word $w = 013331102230313210$ with $\ell_0 = 3$ and $\ell_1 = 5$, the preprocessing for the irradiation time $t = 1$ gives the word 013331100000313210 . We apply our dynamic programming algorithm to the 8-th conjugate of w that is $w' = 000003132100133311$. Then \mathcal{P}_t contains only one word and the associated matrix $D_{w',1,0000}$ is given in Figure 4. The word $u' = 000001111100011111$ belongs to $\text{Sol}_1(w')$. Since the minimum value in the last column of $D_{w',1,0000}$ is 12 and $d(w', w) = 7$, any $u \in \text{Sol}_1(w)$ is at distance $12 + 7$ from w .

For $t = 2$, the first step of the preprocessing of w leads to the word 00333000000303200 and the second step gives 0^{18} . Hence, we have $\text{Sol}_2(w) = \{0^{18}\}$. It follows that $\text{Sol}_3(w) = \{0^{18}\}$ too. Hence, $\text{Sol}(w) = \text{Sol}_1(w)$ and it contains the 12-th conjugate of u' .

6 Perspectives

Interesting extensions of the original problem could be considered. First, say that we have several stopping positions of our irradiation seed and that each of these stopping positions

⁴ Here, factors are considered circularly. For instance, for $\ell_1 = 3$ and $t = 1$, the word $w = 100340022$ has only one too short non-zero factor, the one occurring in position 4.



■ **Figure 4** Matrix for the word $w' = 000003132100133311$ with $\ell_0 = 3$, $\ell_1 = 5$, $t = 1$ and $p = 0000$, in the case where overdoses are forbidden. An arrow between two cells indicates that the value in the arrival cell is calculated from the one in the origin cell. For ease of reading, the corresponding arrows of cells containing $+\infty$ are not drawn. In bold is a path corresponding to an optimal solution (not necessarily unique) for the irradiation time 1.

is represented by a different word, depicting the conformation of the tumor in that exact position. If our cylindrical shield allows only one conformation of open and close sectors, the problem considered here is to find the shield conformation that is as close as possible to all the given tumor conformation words. A good heuristic here could be to find a consensus of these given words and then to use our dynamic programming algorithm to find the best shield conformation for this word. Some works has already been done on this idea of consensus of circular words. Indeed, in [5], Lee *et al.* give an $\mathcal{O}(n^2 \log n)$ algorithm to compute, given a set of three strings, a consensus for this set under the Hamming distance, i.e. a string minimizing the sum of distances to all strings in the given set.

Another extension would be to be able to use several different irradiation times on one tumor conformation (integer word w). Note that to be able to apply various irradiation doses, we would either need a modified device able to close each open sector after a certain irradiation time, or need to use several shield configurations (optimally, a minimal number of them) for each stopping position among the catheter.

References

- 1 Horst Bunke and Urs Bühler. Applications of approximate string matching to 2d shape recognition. *Pattern Recognition*, 26(12):1797–1812, 1993. doi:10.1016/0031-3203(93)90177-X.
- 2 Maxime Crochemore, Gabriele Fici, Robert Mercas, and Solon P. Pissis. Linear-time sequence comparison using minimal absent words & applications. In Evangelos Kranakis, Gonzalo Navarro, and Edgar Chávez, editors, *LATIN 2016: Theoretical Informatics - 12th Latin American Symposium, Ensenada, Mexico, April 11-15, 2016, Proceedings*, volume 9644 of *Lecture Notes in Computer Science*, pages 334–346. Springer, 2016. doi:10.1007/978-3-662-49529-2_25.
- 3 Jens Gregor and Michael G. Thomason. Dynamic programming alignment of sequences representing cyclic patterns. *IEEE Trans. Pattern Anal. Mach. Intell.*, 15(2):129–135, 1993. doi:10.1109/34.192484.

- 4 Roberto Grossi, Costas S. Iliopoulos, Robert Mercas, Nadia Pisanti, Solon P. Pissis, Ahmad Retha, and Fatima Vayani. Circular sequence comparison: algorithms and applications. *Algorithms for Molecular Biology*, 11:12, 2016. doi:10.1186/s13015-016-0076-6.
- 5 Taehyung Lee, Joong Chae Na, Heejin Park, Kunsoo Park, and Jeong Seop Sim. Finding consensus and optimal alignment of circular strings. *Theor. Comput. Sci.*, 468:92–101, 2013. doi:10.1016/j.tcs.2012.11.018.
- 6 V. I. Levenshtein. Binary codes capable of correcting insertions and reversals. *Sov. Phys. Dokl.*, 10:707–710, 1966.
- 7 M. Lothaire. *Combinatorics on words*. Cambridge Mathematical Library. Cambridge University Press, Cambridge, 1997. doi:10.1017/CB09780511566097.
- 8 Maurice Maes. On a cyclic string-to-string correction problem. *Inf. Process. Lett.*, 35(2):73–78, 1990. doi:10.1016/0020-0190(90)90109-B.
- 9 Andrés Marzal and Sergio Barrachina. Speeding up the computation of the edit distance for cyclic strings. In *15th International Conference on Pattern Recognition, ICPR'00, Barcelona, Spain, September 3-8, 2000.*, pages 2891–2894. IEEE Computer Society, 2000. doi:10.1109/ICPR.2000.906217.
- 10 Ramón Alberto Mollineda, Enrique Vidal, and Francisco Casacuberta. Cyclic sequence alignments: Approximate versus optimal techniques. *IJPRAI*, 16(3):291–299, 2002. doi:10.1142/S0218001402001678.
- 11 R. Pötter, C. Haie-Meder, E. Van Limbergen, I. Barillot, M. De Brabandere, J. Dimopoulos, I. Dumas, B. Erickson, S. Lang, A. Nulens, P. Petrow, J. Rownd, and C. Kirisits. Recommendations from gynaecological (GYN) GEC-ESTRO working group (ii): Concepts and terms in 3D image-based treatment planning in cervix cancer brachytherapy—3D dose volume parameters and aspects of 3D image-based anatomy, radiation physics, radiobiology. *Radiotherapy and Oncology*, 78(1):67–77, 2006. doi:10.1016/j.radonc.2005.11.014.
- 12 Robert A. Wagner and Michael J. Fischer. The string-to-string correction problem. *J. ACM*, 21(1):168–173, 1974. doi:10.1145/321796.321811.