Double Digest Revisited: Complexity and Approximability in the Presence of Noisy Data

TECHNICAL REPORT no. 382
ETH Zurich, Department of Computer Science

Mark Cieliebak* Stephan Eidenbenz† Gerhard J. Woeginger‡

Abstract

We revisit the double digest problem, which occurs in sequencing of large DNA strings and consists of reconstructing the relative positions of cut sites from two different enzymes: we first show that double digest is strongly NP-complete, improving previous results that only showed weak NP-completeness. Even the (experimentally more meaningful) variation in which we disallow coincident cut sites turns out to be strongly NP-complete. In a second part, we model errors in data as they occur in real-life experiments: we propose several optimization variations of double digest that model partial cleavage errors, which occur for various reasons. We then show APX-completeness for most of these variations. In a third part, we investigate these variations with the additional restriction that conincident cut sites are disallowed and we show that it is NP-hard to even find feasible solutions in this case, thus making it impossible to guarantee any approximation ratio at all.

Keywords: Double Digestion, Computational Biology, Noisy Data

*Institute of Theoretical Computer Science, ETH Zurich, cieliebak@inf.ethz.ch
†Institute of Theoretical Computer Science, ETH Zurich, eidenben@inf.ethz.ch
‡Faculty of Mathematical Sciences, University of Twente *and* Department of Mathematics and Computer Science, TU Eindhoven, g.j.woeginger@math.utwente.nl

1 Introduction

Double digest experiments are a standard approach to construct physical maps of DNA. Given a large DNA molecule, which is an unknown string over the alphabet $\{A, C, G, T\}$ for our purposes, the objective is to find the locations of markers, i.e., occurrences of short substrings such as *GAATTC*, on the DNA. Physical maps are required e.g. in DNA sequencing in order to determine the sequence of nucleotides (A, C, G, and T) of large DNA molecules, since current sequencing methods allow only to sequence DNA fragments with tens of thousands of nucleotides, while a DNA molecule can have up to 10^8 nucleotides.

In double digest experiments, two enzymes are used to cleave the DNA molecule. An enzyme is a protein that cuts a DNA molecule at specific patterns, the restriction sites. For instance, the enzyme EcoRI cuts at occurrences of the pattern *GAATTC*. Under appropriate experimental conditions, an enzyme cleaves at all restriction sites in the DNA. This process is called *(full) digestion*. Double digest experiments work in three stages: First, clones (copies) of the unknown DNA string are digested by an enzyme A; then a second set of clones is digested by another enzyme B; and finally a third set of clones is digested by a mix of both enzymes A and B, which we will refer to as C. This results in three multisets of DNA fragments. The lengths of these fragments (i.e., their number of nucleotides) are then measured for each multiset by using gel electrophoresis, a standard technique in molecular biology. This leaves us with three multisets of distances (the number of nucleotides) between all adjacent restriction sites, and the objective is to reconstruct the original ordering of the fragments in the DNA molecule, which is the DOUBLE DIGEST problem.

More formally, the DOUBLE DIGEST problem can be defined as follows, where $\text{sum}(S)$ denotes the sum of the elements in a set S, and $\text{dist}(P)$ is the set of all distances between two neighboring points in a set P of points on a line:

Definition (DOUBLE DIGEST)**.** *Given three multisets A, B and C of positive integers with $sum(A) = sum(B) = sum(C)$, are there three sets P^A, P^B and P^C of points on a line, each starting in 0, such that $dist(P^A) = A$, $dist(P^B) = B$ and $dist(P^C) = C$, and such that $P^A \cup P^B = P^C$?*

For example, given multisets $A = \{5, 15, 30\}$, $B = \{2, 12, 12, 24\}$ and $C = \{2, 5, 6, 6, 7, 24\}$ as an instance of DOUBLE DIGEST, then $P^A = \{0, 5, 20, 50\}$, $P^B = \{12, 14, 26, 50\}$ and $P^C = \{5, 12, 14, 20, 26, 50\}$ is a feasible solution (there may exist more solutions).

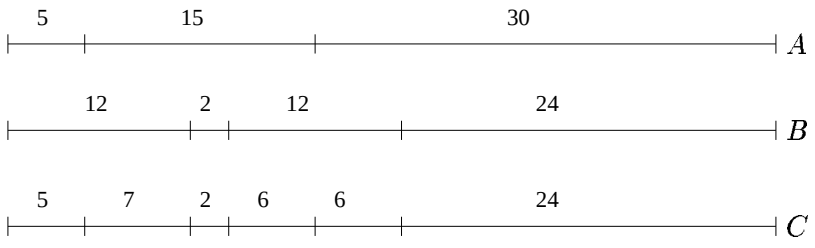

Figure 1: Example for the DOUBLE DIGEST problem.

Due to its importance in molecular biology, the DOUBLE DIGEST problem has been the subject of intense research since the first successful restriction site mappings in the early 1970's [16, 5]. The DOUBLE DIGEST problem is NP-complete [7], and several approaches including exponential algorithms, heuristics, additional experiments or computer-assisted interactive strategies have been proposed (and implemented) in order to tackle the problem [3, 1, 18, 8, 9]. The number of feasible maps for a DOUBLE DIGEST instance can be exponential in the number of fragments [7]. However, some maps can be transformed into each other using cassette transformations [14]. The set of different maps for an instance - modulo cassette transformations - can be characterized by using alternating Eulerian paths in appropriate graph classes [10, 11]. For a survey, see [17] and [12].

The double digest experiment is usually carried out with two enzymes that cut at different restriction sites. For example, enzyme BalI cuts each occurrence of string *TGGCCA* into two substrings *TGG* and *CCA*, while enzyme SalI cuts each occurrence of string *GTCGAC* into two substrings *G* and *TCGAC*. In this case, the two enzymes never cut at the same site. A majority of all possible enzyme pairings of the more than 3000 known enzymes are pairs with such disjoint cutting behavior. On the other hand, some results in the literature rely on enzymes that cut at the same site in some cases (coincidences) [10]. In particular, NP-hardness of the DOUBLE DIGEST problem has so far only been shown using enzymes that allow for coincidences [7, 17, 15]. Indeed, such enzyme pairs exist, for example enzyme HaeIII cuts each *GGCC* string into *GG* and *CC*, and thus cuts at a superset of the sites at which the BalI enzyme cuts. However, having two enzymes that are guaranteed to always cut at disjoint sites seems more natural and might lead – at least intuitively – to easier reconstruction problems. For example, such instances always fulfill $|C| = |A| + |B| - 1$ (where $|S|$ denotes the cardinality of set S). To reflect these different types of experiments, we define the DISJOINT DOUBLE DIGEST problem, which is equivalent to the DOUBLE DIGEST problem with the additional requirement that the two enzymes may never cut at the same site, or, equivalently, that P^A and P^B are disjoint except for the first point (which is 0) and the last point (which is $\mathrm{sum}(A)$).

The NP-hardness results for DOUBLE DIGEST in the literature [7, 17, 15] rely on reductions from weakly NP-complete problems (namely PARTITION). As a first set of results in this paper, we prove in Section 2 that both DOUBLE DIGEST and DISJOINT DOUBLE DIGEST are actually NP-complete in the strong sense by proposing reductions from 3-PARTITION.[1] Thus, no algorithms exist that can solve any of the two problems in pseudopolynomial running time (i.e., in time polynomial in the number of fragments and in the size of the fragments), unless $\mathsf{P} = \mathsf{NP}$.

In a second part of the paper, we try to model reality more closely by taking into account that double digest data usually contains errors. As a matter of fact, all data in double digest experiments is prone to error. Typically, four types of errors can occur [1, 8, 18, 15]:

Partial cleavage An enzyme can fail to cut at some restriction site. Then one large fragment occurs in the data instead of the two (or even more) smaller fragments.

Fragment length Determining the exact length of a fragment from gel electrophoresis is almost impossible. Typical error ranges are between 2% and 7% of the fragment length.

[1] For an introduction to NP-completeness, see [6].

Missing small fragments Small fragments may remain undetected because they travel to far in the gel electropheresis process.

Doublets Two different fragments with almost the same length may generate two spots in the gel electrophoresis that overlap. Thus, only one of the fragments is recognized.

In this paper, we consider the first type of errors, i.e., those due to partial cleavage. They can occur for many reasons, e.g. improper reaction conditions or inaccurate DNA concentration (see e.g. [13] for a list of 13 possible causes). A partial cleavage error occurs e.g. when an enzyme fails to cut at a site where it is supposed to cut in the first (or second) stage of the double digest experiment, but then does cut at this site in the third phase (where it is mixed with the other enzyme). Such an error usually will make it impossible to find a solution for the corresponding DOUBLE DIGEST instance. In fact, only $P^A \cup P^B \subset P^C$ can be guaranteed for any solution. Vice-versa, if an enzyme cuts only in the first (or second) phase, but fails to cut in the third phase, then we can only guarantee $P^C \subset P^A \cup P^B$.

In the presence of errors, usually the data is such that no exact solutions can be expected. Therefore, optimization criteria are necessary in order to compare and gauge solutions. We will define optimization variations of the DOUBLE DIGEST problem taking into account different optimization criteria; our objective will be to find good approximation algorithms. Obviously, an optimal solution for a problem instance with no errors will be a solution for the DOUBLE DIGEST problem itself[2]. Thus, the optimization problem cannot be computationally easier than the original DOUBLE DIGEST problem, and (strong) NP-hardness results for DOUBLE DIGEST carry over to the optimization problem.

An obvious optimization criterion for DOUBLE DIGEST is to minimize the absolute number of partial digestion errors in a solution, i.e., to minimize $e(P^A, P^B, P^C) := |(P^A \cup P^B) - P^C| + |P^C - (P^A \cup P^B)|$ (recall that $|S|$ is the cardinality of set S). Here, points in $(P^A \cup P^B) - P^C$ correspond to errors where enzyme A or B failed to cut in the third phase of the experiment, and points in $P^C - (P^A \cup P^B)$ correspond to errors where either enzyme A or B failed to cut in the first resp. second phase. Unfortunately, the corresponding optimization problem MINIMUM ABSOLUTE ERROR DOUBLE DIGEST in which we try to find point sets P^A, P^B and P^C such that $e(P^A, P^B, P^C)$ is minimum cannot be approximated within any finite approximation ratio (unless P = NP): By contradiction, assume the existence of a polynomial-time approximation algorithm with a finite approximation ratio. Then $\frac{e(\text{solution of algorithm for } I)}{e(\text{optimal solution for } I)} < \infty$ for any instance I. This is also true for instances that actually have no partial cleavage error, and are thus instances of DOUBLE DIGEST. For such instances, an optimal solution has error 0, and therefore the approximation algorithm needs to find a solution with no error as well. Hence, such an algorithm could be used to decide the NP-complete DOUBLE DIGEST problem. A similar argument shows that if we use the number of matching points (i.e, $|P^C \cap (P^A \cup P^B)| - 2$) as a maximization criterion (rather than minimizing the number of errors), the resulting problem cannot be approximated either.

We obtain a more sensible optimization criterion as follows: If we add $|A| + |B| + |C|$ as an offset to the number of errors, we obtain an optimization criterion which turns the

[2]Of course, this only holds if the optimization problem is well-designed.

absolute number of errors into a measure relative to the input size. The corresponding optimization problem is defined as follows:

Definition (MINIMUM RELATIVE ERROR DOUBLE DIGEST). *Given three multisets A, B and C of positive integers with $sum(A) = sum(B) = sum(C)$, find three sets P^A, P^B and P^C of points on a line, each starting in 0, such that $dist(P^A) = A, dist(P^B) = B$ and $dist(P^C) = C$, and such that $r(P^A, P^B, P^C) := |A| + |B| + |C| + e(P^A, P^B, P^C)$ is minimal.*

Instead of counting the number of errors, measuring the total size of a solution is in general a very sensible optimization criterion that does not model cleavage errors exactly but seems very natural to do. In this case, we want to minimize the total number of points in a solution, i.e., to minimize $|P^A \cup P^B \cup P^C|$. This yields the MINIMUM POINT DOUBLE DIGEST problem, which is defined as follows:

Definition (MINIMUM POINT DOUBLE DIGEST). *Given three multisets A, B and C of positive integers with $sum(A) = sum(B) = sum(C)$, find three sets P^A, P^B and P^C of points on a line, each starting in 0, such that $dist(P^A) = A, dist(P^B) = B$ and $dist(P^C) = C$, and such that $|P^A \cup P^B \cup P^C|$ is minimal.*

We show in Section 3 that MINIMUM RELATIVE ERROR DOUBLE DIGEST and MINIMUM POINT DOUBLE DIGEST are APX-hard by proposing gap-preserving reductions[3] from MAXIMUM TRIPARTITE MATCHING to MAXIMUM 4-PARTITION, and from this problem to MINIMUM RELATIVE ERROR DOUBLE DIGESTand MINIMUM POINT DOUBLE DIGEST. The APX-hardness of these problems excludes the possibility of the existence of a polynomial-time approximation scheme (*PTAS*), as there exists a constant $\varepsilon > 0$ such that no polynomial-time algorithm can guarantee to find approximate solutions that are at most a factor $1+\varepsilon$ off the optimum solution (unless P = NP). We then analyze a rather straight-forward approximation algorithm that works for both problems and show that it achieves an approximation ratio of 2 for MINIMUM RELATIVE ERROR DOUBLE DIGEST and an approximation ratio of 3 for MINIMUM POINT DOUBLE DIGEST.

For each optimization problem, a variation can be defined where the enzymes may only cut at disjoint restriction sites (analogous to DISJOINT DOUBLE DIGEST). The corresponding optimization problems are called MINIMUM DISJOINT RELATIVE ERROR DOUBLE DIGEST and MINIMUM DISJOINT POINT DOUBLE DIGEST. In Section 4, we investigate these variations and show that – rather surprisingly – they are even harder than the unrestricted problems: it is NP-hard to even find a feasible solution for a MINIMUM DISJOINT RELATIVE ERROR DOUBLE DIGEST or a MINIMUM DISJOINT POINT DOUBLE DIGEST instance. We establish this result by showing that the problem of disjointly arranging two given sets of numbers is already NP-hard. Any polynomial-time algorithm that claims to achieve a finite approximation ratio for these DOUBLE DIGEST variations will have to be able to find feasible solutions for all instances, which would be equivalent to solving an NP-hard problem. Thus, no finite approximation ratio can be achieved for all DOUBLE DIGEST variations in which we disallow conincident cut sites (unless P = NP). This result would also hold for DOUBLE DIGEST variations with optimization criteria other than the ones we defined, since the proof does not depend on the optimization measure,

[3] For an introduction to gap-preserving reductions, see [2].

but only on the disjointness requirement. We conclude with directions for future research in Section 5.

2 Strong NP-Completeness of (Disjoint) Double Digest

In this section we show strong NP-completeness for the decision problems Double Digest and Disjoint Double Digest. We present reductions from 3-Partition, which is defined as follows: Given $3n$ integers $q_1, \ldots, q_{3n}$ and integer h with $\sum_{i=1}^{3n} q_i = nh$ and $\frac{h}{4} < q_i < \frac{h}{2}$ for all $1 \leq i \leq 3n$, are there n disjoint triples of q_i's such that each triple sums up to h? The 3-Partition problem is NP-complete in the strong sense [6]. Observe that $\frac{h}{4} < q_i < \frac{h}{2}$ implies that each subset of q_i's that sums up to h has exactly three elements. First, we extend the NP-completeness result from [7] for the Double Digest problem.

Lemma 1. Double Digest *is strongly* NP*-complete.*

Proof. Double Digest is NP-complete [7]. To show strong NP-hardness we reduce 3-Partition to Double Digest as follows: Given an instance $q_1, \ldots, q_{3n}$ and h of 3-Partition, let $a_i = c_i = q_i$ for $1 \leq i \leq 3n$, and let $b_j = h$ for $1 \leq j \leq n$. Then the three sets[4] of a_i's, b_j's and c_i's are an instance of Double Digest. If there is a solution for the 3-Partition instance, then there exist n disjoint triples of q_i's (and a_i's as well) such that each triple sums up to h. Starting from 0, we arrange the distances a_i on a line such that each three a_i's that belong to the same triple are adjacent. The same ordering is used for the c_i's. This yields a solution for the Double Digest instance.

On the other hand, if there is a solution for the Double Digest instance, say P^A, P^B and P^C, then there exist n subsets of c_i's such that each subset sums up to h, since each point in P^B must occur in P^C as well, and all adjacent points in P^B have distance h. Then each of these subsets has exactly three elements, since $\frac{h}{4} < q_i < \frac{h}{2}$. Thus, the subsets yield a solution for the 3-Partition instance. □

In the following we show that Double Digest is strongly NP-complete even if we restrict it to enzymes that cut at disjoint restriction sites.

Lemma 2. Disjoint Double Digest *is strongly* NP*-complete.*

Proof. Obviously, Disjoint Double Digest is in NP. Again, we show strong NP-hardness by reducing 3-Partition to it. Given an instance $q_1, \ldots, q_{3n}$ and h of 3-Partition, let $s = \sum_{i=1}^{3n} q_i$ and $t = (n+1) \cdot s$. Recall that $s = nh$. We construct

[4] Technically, these are multisets; however, for sake of simple notation we will not distinguish between sets and multisets, unless crucial.

an instance of DISJOINT DOUBLE DIGEST as follows:

$$
\begin{aligned}
a_i &= q_i && \text{for } 1 \le i \le 3n \\
\hat{a}_j &= 2t && \text{for } 1 \le j \le n-1 \\
b_j &= h + 2t && \text{for } 1 \le j \le n-2 \\
\hat{b}_k &= h + t && \text{for } 1 \le k \le 2 \\
c_i &= q_i && \text{for } 1 \le i \le 3n \\
\hat{c}_j &= t && \text{for } 1 \le j \le 2n-2.
\end{aligned}
$$

Let A consist of the a_i's and $\hat{a}_j$'s, B consist of the b_j's and $\hat{b}_k$'s, and C consist of the c_i's and $\hat{c}_j$'s. Then $\text{sum}(A) = s+(n-1)\cdot 2t = s+(2n-2)\cdot t$, $\text{sum}(B) = (n-2)\cdot(h+2t)+2\cdot(h+t) = s+(2n-2)\cdot t$, and $\text{sum}(C) = s+(2n-2)\cdot t$. Thus, sets A, B and C are an instances of DISJOINT DOUBLE DIGEST.

If there is a solution for the 3-PARTITION instance, then there exist n disjoint triples of q_i's such that each triple sums up to h. Assume w.l.o.g. that the q_i's (and thus the a_i's and c_i's) are ordered such that the three elements or each triple are adjacent.

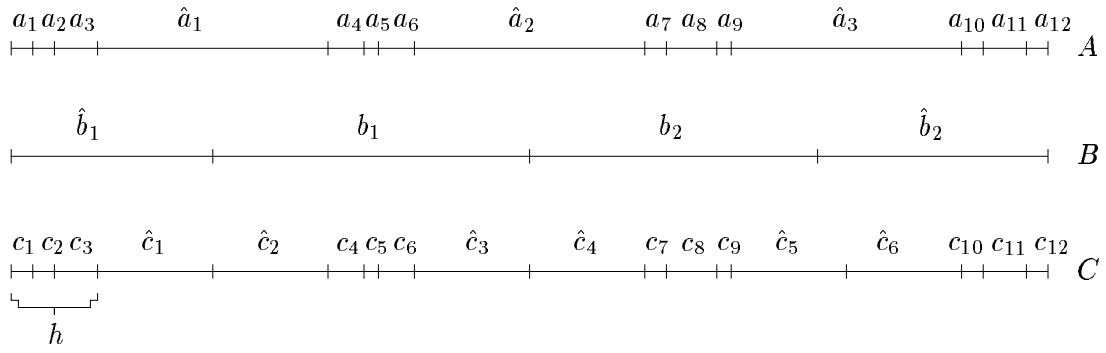

Figure 2: Solution for DISJOINT DOUBLE DIGEST, for $n = 4$.

Starting in 0, we arrange the distances from A on a line such that each three a_i's that belong to the same triple are adjacent, and such that each three a_i's are separated by one $\hat{a}_j$ (cf. Figure 2). Let P^A be the corresponding point set. The distances from B are ordered $\hat{b}_1, b_1, \ldots, b_{n-2}, \hat{b}_2$, and P^B is the corresponding point set. For the distances c_i we use the same ordering as for the distances a_i, and each three c_i's are separated by two $\hat{c}_j$'s. Again, P^C is the corresponding point set. Then P^A, P^B and P^C yield a solution for the DISJOINT DOUBLE DIGEST instance: By construction, the distances in each point set yield exactly the corresponding set of distances. In P^A each point is the sum of an integer less than t and an even multiple of t. On the other hand, in P^B each point except the first and the last is the sum of a multiple of h and an odd multiple of t. Thus, sets P^A and P^B are disjoint except for the first and the last point. Moreover, $P^C = P^A \cup P^B$, hence the three point sets are a solution for the DISJOINT DOUBLE DIGEST instance.

For the opposite direction, let P^A, P^B and P^C be a solution for the DISJOINT DOUBLE DIGEST instance. We consider only P^B and P^C. Each of the $n+1$ points in P^B consists

of a multiple of h and a multiple of t, and each two points in P^B differ in the multiplicity of h. Since $P^B \subseteq P^C$, there must exist $n+1$ points in C that are of the same form. Each point in P^C corresponds to the sum of some distances from C. The distances $\hat{c}_j$ contribute only to the multiplicity of t. Thus, the points in P^C must be such that the distances c_i generate the $n+1$ points with different multiples of h. Starting from zero, this yields n subsets of c_i's that sum up to h. Since $\frac{h}{4} < q_i < \frac{h}{2}$ for all $1 \leq i \leq 3n$, each of the n subsets has exactly three elements. Thus, the corresponding triples of q_i's are a solution for the 3-PARTITION instance. □

3 Approximability of MINIMUM RELATIVE ERROR DOUBLE DIGEST and MINIMUM POINT DOUBLE DIGEST

In this section, we show that MINIMUM RELATIVE ERROR DOUBLE DIGEST and MINIMUM POINT DOUBLE DIGEST are both APX-hard[5]. We introduce a maximization variation of the well–known 4-PARTITION problem [6] which is defined as follows:

Definition (MAXIMUM 4-PARTITION). *Given a multiset $Q = \{q_1, \ldots, q_{4n}\}$ of $4n$ integers and an integer h with $\sum_{i=1}^{4n} q_i = nh$ and $\frac{h}{5} < q_i < \frac{h}{3}$, find a maximum number of disjoint subsets $S_1, \ldots, S_m \subseteq Q$ such that the elements in each set S_i sum up to h.*

While MAXIMUM 4-PARTITION may be an interesting problem per se, we are mainly interested in it as an intermediary problem on our way to proving APX-hardness for our optimization variations of DOUBLE DIGEST. To do so, we propose two gap-preserving reduction, one from MAXIMUM TRIPARTITE MATCHING to MAXIMUM 4-PARTITION, and one from MAXIMUM 4-PARTITION to MINIMUM RELATIVE ERROR DOUBLE DIGEST resp. MINIMUM POINT DOUBLE DIGEST. Here, MAXIMUM TRIPARTITE MATCHINGis the problem where we are given three disjoint sets of integers W, X and Y of equal cardinality and a set $T \subseteq W \times X \times Y$, and we have to find a matching $M \subseteq T$ of maximum cardinality such that no two elements in M agree in any coordinate [6]. Since MAXIMUM TRIPARTITE MATCHING is APX-hard [4] and since our reductions are gap-preserving, this shows APX-hardness for MINIMUM RELATIVE ERROR DOUBLE DIGEST and MINIMUM POINT DOUBLE DIGEST. We also propose and analyze a straight-forward approximation algorithm that achieves an approximation ratio of 3 for MINIMUM RELATIVE ERROR DOUBLE DIGEST and an approximation ratio of 2 for MINIMUM POINT DOUBLE DIGEST.

Lemma 3. MAXIMUM 4-PARTITION *is* APX*-hard.*

Proof. We reduce MAXIMUM TRIPARTITE MATCHING, which is APX-hard, to MaxFour-Partition along the lines of the proof given in [6, pages 97–99]. In fact, the reduction is the same as in [6], we present it here in order to make the paper self-contained; however, the analysis of this reduction as a gap-preserving reduction is new and crucial for our APX-hardness proof.

Let T be a given instance of MAXIMUM TRIPARTITE MATCHING with $T \subseteq W \times X \times Y$, where $W = \{w_1, \ldots, w_d\}, X = \{x_1, \ldots, x_d\}, Y = \{y_1, \ldots, y_d\}$. We assume that each

[5] A problem is said to be APX-hard, if there exists a constant $\varepsilon > 0$ such that no polynomial-time approximation algorithm can guarantee an approximation ratio of $1 + \varepsilon$. See [4] for more details on the complexity class APX.

element from $W \cup X \cup Y$ occurs in at most three triples, since this restricted variation is still APX-hard [4]. W.l.o.g., let $|T| > d$. Let $n = |T|$. From T, we construct an instance Q of MAXIMUM 4-PARTITION that contains $4n$ elements, one for each occurrence of an element of $W \cup X \cup Y$ in a triple in T, and one for each triple in T. The elements in Q corresponding to a particular element $z \in W \cup X \cup Y$ are denoted by $z[1], \ldots, z[N(z)]$, where $N(z)$ is the number of triples in T in which z occurs (thus, $N(z) \leq 3$). Intuitively, $z[1]$ is the "true" element corresponding to z, while $z[2]$ through $z[N(z)]$ are "dummy" elements. All such elements in Q are assigned integer values as follows, where $r = 32d$:

$$
\begin{aligned}
w_i[1] &= 10r^4 + ir + 1 && \text{for } 1 \leq i \leq d \\
w_i[l] &= 11r^4 + ir + 1 && \text{for } 1 \leq i \leq d, 2 \leq l \leq N(w_i) \\
x_j[1] &= 10r^4 + jr^2 + 2 && \text{for } 1 \leq j \leq d \\
x_j[l] &= 11r^4 + jr^2 + 2 && \text{for } 1 \leq j \leq d, 2 \leq l \leq N(x_j) \\
y_k[1] &= 10r^4 + kr^3 + 4 && \text{for } 1 \leq k \leq d \\
y_k[l] &= 8r^4 + kr^3 + 4 && \text{for } 1 \leq k \leq d, 2 \leq l \leq N(y_k)
\end{aligned}
$$

The single integer element u_l corresponding to a triple $(w_i, x_j, y_k) \in T$ is set to $u_l = 10r^4 - ir - jr^2 - kr^3 + 8$, depending on the indices. The key property of the reduction is the following: If we add to u_l the sizes of the three integer elements that correspond to w_i, x_j, y_k, then the total will be $40r^4 + 15$, whenever all three are "true" elements or all three are "dummy" elements. We let bound $h = 40r^4 + 15$. This gives us a correct MAXIMUM 4-PARTITION instance Q. Let OPT denote the size of an optimum solution of the MAXIMUM TRIPARTITE MATCHING instance, and let OPT' denote the size of an optimum solution of the corresponding MAXIMUM 4-PARTITION instance. In order to analyze this reduction as a gap-preserving reduction, we show:

1. If $OPT \geq d$, then $OPT' \geq n$.
2. If $OPT \leq (1 - \frac{4}{3}\varepsilon)d$ for a small constant $\varepsilon > 0$, then $OPT' \leq n - \varepsilon d$.

We will then use these two implications to look at the reduction as a transformation from one promise problem to another.

We first show: If $OPT \geq d$, then $OPT' \geq n$. To see this, we take an optimum solution of the MAXIMUM TRIPARTITE MATCHING instance. Such a solution matches exactly all elements in W, X, Y. The corresponding 4-partition (a partition of Q into sets of four elements) is made up of $n = |T|$ 4-sets (sets of four elements), each containing an element u_l and its corresponding $w_i[\cdot], x_j[\cdot], y_k[\cdot]$. If the triple corresponding to u_l is in the solution for OPT, we group u_l with $w_i[1], x_j[1], y_k[1]$; otherwise, we group u_l with some dummy elements such that it adds up to h. This will use up all dummy elements and we thus have a valid 4-partition of all elements, i.e., a solution of the MAXIMUM 4-PARTITION instance with $OPT \geq n$.

As a second step, we show: If $OPT' > n - \varepsilon d$ for a small constant $\varepsilon > 0$, then $OPT > (1 - \frac{4}{3}\varepsilon)d$. To see this, we take a solution SOL' of the MAXIMUM 4-PARTITION instance with more than $n - \varepsilon d$ different 4-sets that add up to h. Since the total number of elements in Q is $4n$, and since SOL' already contains more than $4(n - \varepsilon d)$ elements,

at most $4\varepsilon d$ elements remain. Assume that all these $4\varepsilon d$ elements are "true" elements, then – since there are $3d$ true elements in total - at least $3d - 4\varepsilon d$ elements are true elements that are contained in the 4-sets of SOL'. Since each 4-set can contain at most three true elements, at least $\frac{3-4\varepsilon}{3}d$ different 4-sets must contain true elements; in fact, each 4-set contains either three or no true elements. As argued in [6], any 4-set that adds up to h must contain exactly one element u_l and either the three corresponding "true" elements $w_i[1], x_j[1], y_k[1]$ or three corresponding dummy elements. We then construct a solution SOL for the MAXIMUM TRIPARTITE MATCHING instance as follows: If a 4-set from OPT' contains three true elements, we let the corresponding triple be in the solution SOL, yielding a feasible solution SOL with at least $(1 - \frac{4}{3}\varepsilon)d$ triples.

Consider the promise problem variation of MAXIMUM TRIPARTITE MATCHING, where we are given an instance and promised that either $OPT \geq d$ (i.e., all elements can be matched) or $OPT \leq (1 - \frac{4}{3}\varepsilon)d$ (i.e., at most a fraction of $1 - \frac{4}{3}\varepsilon$ of the elements can be matched); our task is to find out which one of these two cases is true. Since MAXIMUM TRIPARTITE MATCHING is APX-hard, this promise problem is NP-hard for small enough, but constant values of ε (see [2] for details). The reduction described above transforms the promise problem variation of MAXIMUM TRIPARTITE MATCHING into a promise problem of MAXIMUM 4-PARTITION, where we are promised that either $OPT' \geq n$ or $OPT' \leq n - \varepsilon d$. It is NP-hard to decide which of the two cases is true, because we could use a polynomial-time algorithm for this to also decide the corresponding promise problem for MAXIMUM TRIPARTITE MATCHING. Since in our restricted variation of MAXIMUM TRIPARTITE MATCHING each element from $W \cup X \cup Y$ occurs in at most three triples, and since $|W \cup X \cup Y| = 3d$, there are at most $3 \cdot 3d$ occurrences of elements in all triples. Thus, we have $|T| \leq 3d$, since each triple has three elements. In terms of inapproximability, we have that no polynomial-time algorithm can achieve an approximation ratio of:

$$\frac{|T|}{|T| - \varepsilon d} = 1 + \frac{\varepsilon d}{|T| - \varepsilon d} \geq 1 + \frac{\varepsilon d}{3d - \varepsilon d} \geq 1 + \frac{\varepsilon}{3}.$$

□

We are now ready to prove that MINIMUM RELATIVE ERROR DOUBLE DIGEST and MINIMUM POINT DOUBLE DIGEST are APX-hard.

Lemma 4. MINIMUM POINT DOUBLE DIGEST *is* APX*-hard.*

Proof. We propose a gap-preserving reduction from MAXIMUM 4-PARTITION to MINIMUM POINT DOUBLE DIGEST. For a given MAXIMUM 4-PARTITION instance I, which consists of a set $Q = \{q_1, \ldots, q_{4n}\}$ of integers and an integer h, we construct an instance I' consisting of three sets A, B, and C of MINIMUM RELATIVE ERROR DOUBLE DIGEST as follows: Set A is exactly the same as set Q; set B contains n times the element h; set C is the same as set A.

Let OPT denote the size of an optimum solution of I, and let OPT' denote the size of an optimum solution of I'. We prove two implications that show how this reduction transforms a promise problem into another promise problem:

1. If $OPT \geq n$, then $OPT' \leq 4n + 1$.
2. If $OPT < (1 - \varepsilon)n$ for any $\varepsilon > 0$, then $OPT' > (4 + frac\varepsilon 2)n + 1$

The first implication is straight-forward: If $OPT \geq n$, then all elements from Q can be grouped into correct 4-sets. We then obtain a solution for I' by arranging the elements in A and C such that their order corresponds to the order of the elements in the 4-sets; the elements of B are distributed evenly (there actually is no other choice). This yields a solution for I' with $|Q|+1$ points, thus $OPT' \leq |Q|+1 = 4n+1$.

We prove the second implication by proving its contraposition: if $OPT' \leq (4 + frac\varepsilon 2)n + 1$, then $OPT \geq (1-\varepsilon)n$ for any $\varepsilon > 0$. Let SOL' be a solution for I' with at most $(4 + frac\varepsilon 2)n + 1$ points. We may assume that $P^A = P^C$ in SOL' [If $P^A \neq P^C$ in SOL', we transform the solution by setting P^C equal to P^A; this will not increase the number of points in the solution. To see this, let set S contain all points from P^C that do not have a matching point in P^A; a point $s \in S$ might have a matching point in P^B, but "moving" the matching point from set P^B to P^A will not increase the total number of points or – alternatively speaking – decrease the number of errors.]. Since $4n+1$ points are trivially necessary in any solution (as set A contains $4n$ elements), at most $\frac{\varepsilon}{2}n$ points are caused by points from P^B that have no matching points in P^A. Since there are n elements in B, at least $n - \frac{\varepsilon}{2}n = n(1-\frac{\varepsilon}{2})$ points from P^B are matched with points in P^A. Since each non-matching point can destroy at most two potential 4-sets, the non-matching points from P^B can destroy at most a total of $2\frac{\varepsilon}{2}n = \varepsilon n$ 4-sets in a corresponding solution for I. Thus, we obtain a solution for I with at least $(1-\varepsilon)n$ feasible 4-sets.

Our two implications match an NP-hard promise problem variation of Maximum 4-Partition to an NP-hard promise problem variation of Minimum Point Double Digest. In terms of inapproximability for Minimum Point Double Digest, we have that no polynomial-time algorithm can achieve an approximation ratio of:

$$\frac{|Q|+1+\frac{\varepsilon}{8}|Q|}{|Q|+1} \geq 1 + \frac{\varepsilon}{16}$$

Thus, Minimum Point Double Digest is APX-hard. □

Lemma 5. Minimum Relative Error Double Digest *is* APX*-hard.*

Proof. The proof uses the same reduction as in Lemma 4. The two implications are as follows:

1. If $OPT \geq n$, then $OPT' \leq |A| + |B| + |C|$.
2. If $OPT < (1-\varepsilon)n$ for any $\varepsilon > 0$, then $OPT' > |A| + |B| + |C| + \frac{\varepsilon}{2}n$

Both implications can be shown using essentially the same arguments as in the proof of Lemma 4. Since $|A| + |B| + |C| = \frac{9}{4}|Q| = 9n$, we have the following in terms of inapproximability:

$$\frac{|A| + |B| + |C| + \frac{\varepsilon}{2}n}{|A| + |B| + |C|} = 1 + \frac{\varepsilon}{18}$$

Thus, Minimum Relative Error Double Digest is APX-hard. □

A straight-forward approximation algorithm for both problems simply arranges all distances from A, B and C on a line in a random fashion, starting in 0. If we analyze this algorithm as an approximation algorithm for Minimum Point Double Digest, we see

that this will result in a solution with at most $|A| + |B| + |C| + 1$ points; on the other hand, an optimum solution will always use at least $\max(|A|, |B|, |C|) + 1$ points. Thus, this trivial approximation algorithm achieves an approximation ratio of 3 for MINIMUM POINT DOUBLE DIGEST.

If we consider the same algorithm to be an approximation algorithm for MINIMUM RELATIVE ERROR DOUBLE DIGEST, we see that it will find a solution with an optimization measure of at most $r(P^A, P^B, P^C) = |A|+|B|+|C|+|A|+|B|+|C|-3$, since not a single point might be matched except for the first and the last point. In an optimum solution the optimization measure would be at least $|A| + |B| + |C|$, thus giving an approximation ratio of 2 for this approximation algorithm. This result is in line with our intention of modelling a relative error measure in MINIMUM RELATIVE ERROR DOUBLE DIGEST: the worst feasible solution is 100 per cent off the optimum solution, which corresponds to an approximation ratio of 2.

4 NP-hardness of Finding Feasible Solutions for Optimization Variations of DISJOINT DOUBLE DIGEST

In this section, we show that all DOUBLE DIGEST optimization variations in which we disallow coincidences cannot be approximated by any polynomial-time approximation algorithm with a finite approximation ratio, unless $P = NP$. We achieve this by showing that even finding feasible solutions for these problems is NP-hard. To this end, we introduce the decision problem DISJOINT ORDERING which is defined as follows:

Definition (DISJOINT ORDERING). *Given two multisets A and B of integers with $sum(A) = sum(B)$, find two sets P^A and P^B of points on a line, starting in 0, such that $dist(P^A) = A$, $dist(P^B) = B$, and such that P^A and P^B are disjoint except for the first and the last point.*

First, we will reduce 3-PARTITION to DISJOINT ORDERING, and then we will show how to reduce DISJOINT ORDERING to any optimization variation of DISJOINT DOUBLE DIGEST.

Lemma 6. DISJOINT ORDERING *is* NP*-complete.*

Proof. Obviously, DISJOINT ORDERING is in NP. To show NP-hardness, we reduce 3-PARTITION to it. Given an instance $q_1, \ldots, q_{3n}$ and h of 3-PARTITION, we construct an instance of DISJOINT ORDERING as follows: Let

$$\begin{aligned} a_i &= q_i && \text{for } 1 \le i \le 3n \\ \hat{a}_j &= h && \text{for } 1 \le j \le n+1 \\ b_i &= h+2 && \text{for } 1 \le i \le n \\ \hat{b}_j &= 1 && \text{for } 1 \le j \le (n+1)\cdot h - 2n. \end{aligned}$$

Let A consist of the a_i's and $\hat{a}_j$'s, and let B consist of the b_i's and $\hat{b}_j$'s. Then $\mathrm{sum}(A) = n\cdot h+(n+1)\cdot h = (2n+1)\cdot h$, and $\mathrm{sum}(B) = n\cdot(h+2)+(n+1)\cdot h-2n = (2n+1)\cdot h$. The

number of distances in A is polynomial in n, while the cardinality of B is only polynomial in n and h. However, since 3-PARTITION is NP-complete in the strong sense, it is still NP-complete if h is polynomially bounded in n. In this case, A and B are an instance of DISJOINT ORDERING that can be constructed in time polynomial in n.

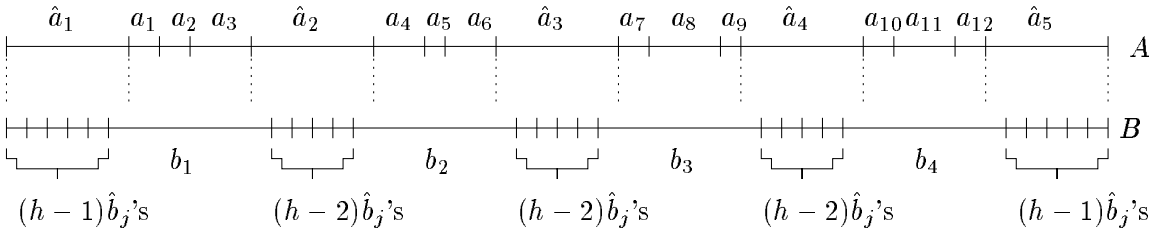

Figure 3: Disjoint ordering of distances in A and B, for $n = 4$. Dotted lines have have distance h.

If there is a solution for the 3-PARTITION instance, then there exist n disjoint triples of q_i's such that each triple sums up to h. W.l.o.g., we assume that the q_i's are ordered such that each three q_i's from a triple are adjacent. We put the distances from A on a line, starting in 0, such that each three a_i's that belong to the same triple are adjacent, and such that each three a_i's are separated by one $\hat{a}_j$ (cf. Figure 3). The distances from B are arranged on a line as follows: first we have $h-1$ distances $\hat{b}_j$, followed by n combinations of one distance b_i and $h-2$ distances $\hat{b}_j$, and at the end there are again $h-1$ distances $\hat{b}_j$. Let P^A and P^B be the corresponding point sets. Then P^A and P^B are disjoint except for the first and the last point, and they yield a solution for the DISJOINT ORDERING instance.

For the opposite direction, assume that P^A and P^B are a solution for the DISJOINT ORDERING instance. First we show that the ordering of the distances from B constructed in the previous paragraph is the only possible arrangement. In P^B, there are n distances b_i. They separate at most $n+1$ blocks of consecutive distances $\hat{b_j}$, including the two margin blocks. Some of the blocks might be empty. Since h is the largest number in A, the length of a margin block is at most $h-1$, and the length of an inner block is at most $h-2$. Thus, the total length of the blocks is at most $2 \cdot (h-1) + (n-1) \cdot (h-2) = (n+1)h - 2n$. This is exactly the number of distances $\hat{b}_j$ (and therefore their total length), thus each of the previous upper bounds has to be tight. This yields the ordering of the distances from B presented above. For the ordering in P^A, the $n+1$ distances $\hat{a}_j$ must be used to cover the $n+1$ blocks of consequtive $\hat{b}_j$. This leaves exactly n gaps, each of length h, which are covered by the distances a_i. This yields a solution for the 3-PARTITION instance, since $\frac{h}{4} < q_i < \frac{h}{2}$ for $1 \leq i \leq 3n$ implies that each gap is covered by exactly three distances. □

We will now show how to reduce DISJOINT ORDERING to MINIMUM DISJOINT RELATIVE ERROR DOUBLE DIGEST: Let A and B be an instance of DISJOINT ORDERING. We "construct" an instance of MINIMUM DISJOINT RELATIVE ERROR DOUBLE DIGEST by simply letting sets A and B be the same sets, and set C be the empty set. If an approximation algorithm for MINIMUM DISJOINT RELATIVE ERROR DOUBLE DIGEST finds a

feasible solution for this instance, this yields immediately a solution for the DISJOINT ORDERING instance, since any solution feasible solution for MINIMUM DISJOINT RELATIVE ERROR DOUBLE DIGEST must arrange the elements from A and B in a disjoint fashion.

The same argument applies for MINIMUM DISJOINT POINT DOUBLE DIGEST, and for any other (reasonable) optimization variation of DISJOINT DOUBLE DIGEST since the reduction is totally independent of the optimization criterion. Thus, we have:

Lemma 7. *No polynomial-time approximation algorithm can achieve a finite approximation ratio for* MINIMUM DISJOINT RELATIVE ERROR DOUBLE DIGEST, MINIMUM DISJOINT POINT DOUBLE DIGEST *and any other reasonable optimization variation of* DISJOINT DOUBLE DIGEST *(unless* $\mathsf{P} = \mathsf{NP}$*).*

5 Conclusion

In this paper, we showed that DOUBLE DIGEST and DISJOINT DOUBLE DIGEST are strongly NP-complete; in a second part, we defined several optimization variations of DOUBLE DIGEST that model partial cleavage errors, proved APX-hardness for MINIMUM RELATIVE ERROR DOUBLE DIGEST and MINIMUM POINT DOUBLE DIGEST, and analyzed straight-forward approximation algorithms for these problems that achieve constant approximation ratios. In a last set of results, we showed for several DOUBLE DIGEST optimization variations, where conincidences are not allowed, that even finding feasible solutions is NP-hard.

While our approximability results are tight for all DISJOINT DOUBLE DIGEST variations, our results leave considerable gaps regarding the exact approximability threshold for MINIMUM RELATIVE ERROR DOUBLE DIGEST and MINIMUM POINT DOUBLE DIGEST, which present challenges for future research. In a different direction of future research, optimization variations of DOUBLE DIGEST that model the three other error types (i.e., fragment length, missing small fragments, and doublets) or even combinations of different error types should be defined and studied. On a meta-level of arguing, it seems unlikely that an optimization variation that models partial cleavage errors and some of the other error types could be any easier than the problems that model only partial cleavage errors, but there is a possibility that some error types might offset each other in a cleverly defined optimization problem.

References

[1] L. Allison and C. N. Yee. Restriction site mapping is in separation theory. *Computer Applications in the Biosciences (CABIOS)*, 4(1):97–101, 1988.

[2] S. Arora and C. Lund. Hardness of approximations. In D. Hochbaum, editor, *Approximation Algorithms for* NP*-Hard Problems*, pages 399–446. PWS Publishing Company, 1996.

[3] B. Bellon. Construction of restriction maps. *Computer Applications in the Biosciences (CABIOS)*, 4(1):111–115, 1988.

[4] P. Crescenzi and V. Kann. A compendium of NP optimization problems. In G. Ausiello, P. Crescenzi, G. Gambosi, V. Kann, A. Marchetti-Spaccamela, and M. Protasi, editors, *Complexity and Approximation. Combinatorial Optimization Problems and their Approximability Properties*. Springer, 1999.

[5] K. J. Danna and D. Nathans. Specific cleavage of simian virus 40 DNA by restriction endonuclease of hemophilus influenzal. *Proc. of the National Academy of Sciences USA*, 68:2913–2917, 1971.

[6] M. R. Garey and D. S. Johnson. *Computers and Intractability: A Guide to the Theory of* NP-*Completeness*. Freeman, San Fransisco, 1979.

[7] L. Goldstein and M. S. Waterman. Mapping DNA by stochastic relaxation. *Advances in Applied Mathematics*, 8:194–207, 1987.

[8] J. Inglehart and P. C. Nelson. On the limitations of automated restriction mapping. *Computer Applications in the Biosciences (CABIOS)*, 10(3):249–261, 1994.

[9] M-Y Kao, J. Samet, and W-K Sung. The enhanced double digest problem for DNA physical mapping. In *Proc. of the* 7^{th} *Scandinavian Workshop on Algorithm Theory (SWAT00)*, pages 383–392, 2000.

[10] D. R. Martin. Equivalence classes for the double-digest problem with coincident cut sites. *Jounal of Computational Biology*, 1(3):241–253, 1994.

[11] P. A. Pevzner. DNA physical mapping and alternating Eulerian cycles in colored graphs. *Algorithmica*, 13:77–105, 1995.

[12] P. A. Pevzner. *Computational Molecular Biology*. MIT Press, 2000.

[13] Promega GmbH, `http://www.promega.com/guides/re_guide/toc.htm`. *Restriction Enzymes Resource*, 2002.

[14] W. Schmitt and M. S. Waterman. Multiple solutions of DNA restriction mapping problems. *Advances in Applied Mathematics*, 12:412–427, 1991.

[15] J. Setubal and J. Meidanis. *Introduction to Computational Molecular Biology*. PWS Boston, 1997.

[16] H. O. Smith and K. W. Wilcox. A restriction enzyme from hemophilus influenza. I. Purification and general properties. *Journal of Molecular Biology*, 51:379–391, 1970.

[17] M. S. Waterman. *Introduction to Computational Biology*. Chapman & Hall, 1995.

[18] L. W. Wright, J. B. Lichter, J. Reinitz, M. A. Shifman, K. K. Kidd, and P. L. Miller. Computer-assisted restriction mapping: an integrated approach to handling experimental uncertainty. *Computer Applications in the Biosciences (CABIOS)*, 10(4):435–442, 1994.