

Small Candidate Set for Translational Pattern Search

Ziyun Huang

Department of Computer Science and Software Engineering
Penn State Erie, The Behrend College
zxh201@psu.edu

Qilong Feng

School of Computer Science and Engineering
Central South University, P.R. China
csufeng@mail.csu.edu.cn

Jianxin Wang

School of Computer Science and Engineering
Central South University, P.R. China
jxwang@csu.edu.cn

Jinhui Xu

Department of Computer Science and Engineering
State University of New York at Buffalo
jinhui@buffalo.edu

Abstract

In this paper, we study the following pattern search problem: Given a pair of point sets A and B in fixed dimensional space \mathbb{R}^d , with $|B| = n$, $|A| = m$ and $n \geq m$, the pattern search problem is to find the translations \mathcal{T} 's of A such that each of the identified translations induces a matching between $\mathcal{T}(A)$ and a subset B' of B with cost no more than some given threshold, where the cost is defined as the minimum bipartite matching cost of $\mathcal{T}(A)$ and B' . We present a novel algorithm to produce a small set of candidate translations for the pattern search problem. For any $B' \subseteq B$ with $|B'| = |A|$, there exists at least one translation \mathcal{T} in the candidate set such that the minimum bipartite matching cost between $\mathcal{T}(A)$ and B' is no larger than $(1 + \epsilon)$ times the minimum bipartite matching cost between A and B' under any translation (*i.e.*, the optimal translational matching cost). We also show that there exists an alternative solution to this problem, which constructs a candidate set of size $O(n \log^2 n)$ in $O(n \log^2 n)$ time with high probability of success. As a by-product of our construction, we obtain a weak ϵ -net for hypercube ranges, which significantly improves the construction time and the size of the candidate set. Our technique can be applied to a number of applications, including the translational pattern matching problem.

2012 ACM Subject Classification Theory of computation \rightarrow Pattern matching; Theory of computation

Keywords and phrases Bipartite matching, Alignment, Discretization, Approximate algorithm

Digital Object Identifier 10.4230/LIPIcs.ISAAC.2019.29

Funding The research of the first and last authors was supported in part by NSF through grant CCF-1716400. The research of the last author was also supported by NSF through grant IIS-1910492. The research of the second and third authors was supported in part by NSFC through grants 61872450, 61828205, and 61672536.



© Ziyun Huang, Qilong Feng, Jianxin Wang and Jinhui Xu;
licensed under Creative Commons License CC-BY

30th International Symposium on Algorithms and Computation (ISAAC 2019).

Editors: Pinyan Lu and Guochuan Zhang; Article No. 29; pp. 29:1–29:18

Leibniz International Proceedings in Informatics



LIPICs Schloss Dagstuhl – Leibniz-Zentrum für Informatik, Dagstuhl Publishing, Germany

43 **1** Introduction

44 Pattern search/matching is an important problem in computer science and finds applications
 45 in many different domains such as computer vision, pattern recognition, robotics, autonomous
 46 driving, and surveillance. In this paper, we consider a special variant of the problem, where
 47 the objective is to find a small pattern (*e.g.*, the image of a car) from a large environment
 48 (called background; *e.g.*, the image of a road with traffic) which may contain multiple copies
 49 of the pattern or its deformations. The problem is often encountered in our daily life. For
 50 example, most of the smart phones have the capability of identifying human faces (or other
 51 types of objects) in their camera softwares. The core problem for such a functionality is
 52 to efficiently find all appearances of a given object from the pictures. The pattern search
 53 problem may also appear in higher ($d > 2$) dimensional space. One of such examples is the
 54 pattern extraction problem arising in biological image analysis, where the objective is to
 55 identify the 3D spatial positioning patterns of chromosomes [9, 10, 22].

56 We approach this pattern search problem from a geometric perspective, using a formulation
 57 from [1] with some slight modifications. The pattern is represented by a point set A and the
 58 background by a point set B in \mathbb{R}^d space for some fixed d . The sizes of A and B are m and
 59 n , respectively, with $n \geq m$ (note that n could be significantly larger than m). If there is a
 60 translation \mathcal{T} which moves A to a new location $\mathcal{T}(A)$ such that the *difference* between $\mathcal{T}(A)$
 61 and some $B' \subseteq B$ with $|B'| = |A|$ is minimized, we say that an instance B' of pattern A is
 62 discovered by \mathcal{T} , where the difference of two sets is measured by their bipartite matching
 63 cost (see Section Preliminaries).

64 The pattern search problem is a natural extension of the pattern matching problem,
 65 whose aim is to find a rigid transformation that minimizes the difference of two given sets A
 66 and B . Extensive research has been done for the pattern matching problem using different
 67 metrics as the measurement for the similarity/distance of two sets [13, 11, 14]. Commonly
 68 used metrics include Euclidean distance in 1-to-1 matching, Hausdorff distance in 1-to-many
 69 or many-to-1 matching, and Earth's Mover Distance (EMD) in many-to-many matching. An
 70 early result on this problem is the paper [18] which provides an $\tilde{O}(mn^2)$ -time solution to
 71 the matching problem under translation and Hausdorff distance in \mathbb{R}^2 . A more recent result
 72 is the one in [11] which approximates (with ratio $(1 + \epsilon)$) the pattern matching problem
 73 under rigid transformations and EMD metric in \mathbb{R}^d . The running time of their algorithm is
 74 $\tilde{O}((mn)^{2d})$, which is near the lower bound (*i.e.*, $\Omega(mn^{\Omega(d)})$) [5] of the problem.

75 For the pattern search problem under translations, there is a number of results [21, 4] that
 76 are closely related to the work in this paper. Most of them use the concept of *partial-matching*
 77 *Voronoi diagram*.

78 For two given point sets A and B , their partial-matching Voronoi diagram (PMVD) is a
 79 partition of the translation space into regions so that each of them consists of translations
 80 \mathcal{T} sharing the same locally optimal bipartite matching between $\mathcal{T}(A)$ and a subset of B .
 81 The PMVD uses the sum of squared distances as the measurement for the matching cost.
 82 Clearly, such a Voronoi diagram is capable of solving the pattern search/matching problem,
 83 as only one translation from every cell needs to be determined for finding the optimal
 84 translational alignment of A and B . The best known upper bound on the size of PMVD is
 85 $O(m!m^d n^{2d})$ [17]. Ben-Avraham *et al.* [4] constructed a partial-matching Voronoi diagram in
 86 \mathbb{R}^2 of complexity $O(n^2 m^{3.5} \log^m m)$, and found *locally* min-cost translations in $O(m^6 n^3 \log n)$
 87 time.

88 In this paper, we develop a novel method for finding a small set \mathbb{T} of candidate translations
 89 so that for any instance $B' \subseteq B$ of A , there is at least one translation \mathcal{T} in the candidate

90 set that matches $\mathcal{T}(A)$ and B' approximately. A subset B' of B is called an instance of A if
 91 $|B'| = |A|$. Note that B' can be any subset of B as long as it has the same size as A and
 92 may have a large difference with A ; this is somewhat different from the normal meaning of
 93 instance. We say that a translation \mathcal{T} *discovers* an instance B' if it minimizes the bipartite
 94 matching cost of $\mathcal{T}(A)$ and B' . For any $\epsilon > 0$, a translation \mathcal{T} $(1 + \epsilon)$ -approximately discovers
 95 B' , if the bipartite matching cost between $\mathcal{T}(A)$ and B' is no more than $(1 + \epsilon)$ times the
 96 minimum difference between B' and A under any translation. Clearly, with such a candidate
 97 set \mathbb{T} , we are able to find all instances B' which are similar to A , where the level of similarity
 98 is controlled by some threshold on the difference of B' and the translations of A . Note that
 99 if a value of the threshold is given in advance, it is possible to further reduce the size of
 100 the candidate set by removing (during the execution of our algorithm) those translations
 101 which induce higher matching cost (see the remark in Section 6 for more details). Also, if
 102 B has some exact (or congruent) instances of A , \mathbb{T} will contain all translations inducing
 103 zero-difference matchings of A .

104 The problem of finding the translations that match pattern A to all its exact instances
 105 could be quite challenging, as suggested by the exponential size of PMVD in [21, 4]. However,
 106 we are able to show that if approximation and implicit representation are allowed, the
 107 problem can be solved much more efficiently through identifying a small candidate set of
 108 translations with a (surprisingly) near linear size.

109 Particularly, we show that it is possible to build a candidate set \mathbb{T} with size $O_{d,\epsilon}(n \log n)$
 110 for A and B in $O(mn \log mn)$ deterministic time. This bound is asymptotically near optimal,
 111 since it is easy to construct an example that needs $O(n)$ different translations to yield all
 112 perfect matches of a pattern (such an example will be given later). A trade-off between the
 113 running time and the size of \mathbb{T} can also be made, which provides a probabilistic algorithm to
 114 build a \mathbb{T} with a slightly larger size (*i.e.*, $O(n \log^2 n)$) but a better time complexity (*i.e.*,
 115 $O(n \log^2 n)$) which is independent of m . Our construction is based on a weak ϵ -net technique
 116 and a space discretization technique from [7]. Our approach shows a non-obvious connection
 117 between weak ϵ -net and the pattern search problem. A fast algorithm is also provided to
 118 build a small-size ϵ -net for ranges of axis-aligned hypercubes.

119 In some sense, candidate set \mathbb{T} can be viewed as an implicit and approximate representation
 120 of the exponential-size PMVD. Thus, it has the potential to be used in various applications
 121 of PMVD, such as moving object tracking and autonomous driving. Note that in such
 122 applications, all translations in \mathbb{T} (rather than those inducing better matchings) are needed.

123 2 Preliminaries

124 In this paper, we do not distinguish a point and its corresponding vector in \mathbb{R}^d , *i.e.*, a point
 125 is equivalent to a vector that points from the origin to it. In this way, a point in \mathbb{R}^d naturally
 126 defines a translation (along the corresponding vector) in \mathbb{R}^d . Following the basic vector
 127 arithmetics, operators $+$ and $-$ can be applied to points in \mathbb{R}^d .

128 Given two point sets A_1 and A_2 of the same size, their bipartite matching is represented
 129 by a bijective mapping $\phi : A_1 \rightarrow A_2$. That is, each point $a \in A_1$ is matched to a distinct point
 130 $\phi(a) \in A_2$ and the cost C_{ϕ, A_1, A_2} of the matching is defined as $C_{\phi, A_1, A_2} = \sum_{a \in A_1} \|a - \phi(a)\|$.
 131 The difference of A_1 and A_2 , denoted by $\Delta(A_1, A_2)$, is then $\Delta(A_1, A_2) = \min_{\phi} C_{\phi, A_1, A_2}$.

132 Let $A, B \subset \mathbb{R}^d$ be the two point sets in the pattern search problem, with $|B| = n, |A| = m$
 133 and $n \geq m$. We label points in A and B , respectively, as $A = \{a_1, a_2, \dots, a_m\}$ and
 134 $B = \{b_1, b_2, \dots, b_n\}$. The **reference set** P of A and B is defined as follows.

135 ► **Definition 1.** Let $p_{i,j} = b_j - a_i$ for any $a_i \in A$ and $b_j \in B$. The reference set P of A and
 136 B is the multi-set that contains all $p_{i,j}$ for $i = 1, 2, \dots, m$ and $j = 1, 2, \dots, n$.

137 We use an injective mapping $\phi : \{1, 2, \dots, m\} \rightarrow \{1, 2, \dots, n\}$ to represent a perfect
 138 matching for A (under a certain translation) and a subset of B . $\phi(i) = j$ means that a_i is
 139 matched to b_j . The matching cost C_ϕ for a matching ϕ is defined as $\sum_{i=1}^m \|a_i - b_{\phi(i)}\|$.

140 From the definition of P , it is clear that for any $a_i \in A$, $b_j \in B$ and translation
 141 $\mathcal{T} \in \mathbb{R}^d$, $\|\mathcal{T}(a_i) - b_j\| = \|\mathcal{T} - p_{i,j}\|$. The matching cost $C_\phi(\mathcal{T})$ between $\mathcal{T}(A)$ and B for
 142 matching ϕ is then $\sum_{i=1}^m \|\mathcal{T} - p_{i,\phi(i)}\|$. In other words, $C_\phi(\mathcal{T})$ is the sum of distances from
 143 \mathcal{T} to m points in P . For convenience, we let $P(\phi) = \{p_{1,\phi(1)}, p_{2,\phi(2)}, \dots, p_{m,\phi(m)}\}$. Thus
 144 $C_\phi(\mathcal{T}) = \sum_{p \in P(\phi)} \|\mathcal{T} - p\|$.

145 In the rest of the paper, we study all possible matchings between A and B based on the
 146 relationship of $\mathcal{T} \in \mathbb{R}^d$ and P . This means that our algorithms work in the translational
 147 space of P , instead of the original space of A and B .

148 For any pair of point sets X and Y with the same cardinality, we use $\Delta(X, Y)$ to denote
 149 the minimum bipartite matching cost of X and Y . A set $B' \subseteq B$ is called an instance of A if
 150 $|B'| = |A|$. For any instance B' , let \mathcal{T} be the translation that minimizes $\Delta(\mathcal{T}(A), B')$. Then,
 151 we say that \mathcal{T} *discovers* B' , or B' is discoverable at \mathcal{T} . If there is another translation \mathcal{T}'
 152 satisfying the following inequality, $\Delta(\mathcal{T}'(A), B') \leq (1 + \epsilon)\Delta(\mathcal{T}(A), B')$, then, we say that B'
 153 is $(1 + \epsilon)$ -approximately discoverable at \mathcal{T}' .

154 3 Main Results

155 The main results of this paper are the following theorems which show that for any given
 156 pair of point sets A and B , and any constant $\epsilon > 0$, it is possible to efficiently construct
 157 a small-size candidate set \mathbb{T} of translations in \mathbb{R}^d such that any instance $B' \subseteq B$ of A is
 158 approximately discoverable at some $\mathcal{T} \in \mathbb{T}$. In other words, \mathbb{T} is a small-size candidate set of
 159 translations to find all instances of A approximately.

160 ► **Theorem 2.** For any pair of point sets A and B in fixed dimensional space \mathbb{R}^d with
 161 size m and n ($n \geq m$), respectively, and any small constant $0 < \epsilon < 1$, it is possible to
 162 construct, deterministically, a candidate set $\mathbb{T} \subset \mathbb{R}^d$ of size $O(n \log n)$ in $O(mn \log mn)$
 163 time such that for any given instance $B' \subseteq B$ of A , there exists a translation $\mathcal{T} \in \mathbb{T}$ that
 164 $(1 + \epsilon)$ -approximately discovers B' .

165 ► **Theorem 3.** For any pair of point sets A and B in fixed dimensional space \mathbb{R}^d with size
 166 m and n ($n \geq m$), respectively, and any small constant $0 < \epsilon < 1$, it is possible to construct
 167 a candidate set $\mathbb{T} \subset \mathbb{R}^d$ of size $O(n \log^2 n)$ in $O(n \log^2 n)$ time, with success probability at
 168 least $1 - 1/n$. For any given instance $B' \subseteq B$ of A , there exists a translation $\mathcal{T} \in \mathbb{T}$ that
 169 $(1 + \epsilon)$ -approximately discovers B' .

170 With the above theorems, we immediately have the following corollary as their application
 171 to the classical pattern matching problem. (See the appendix for the proof.)

172 ► **Corollary 4.** It is possible to generate a set of $O(n \log n)$ candidate translations in \mathbb{R}^d
 173 in $O(mn \log mn)$ time such that one of them induces a $(1 + \epsilon)$ -approximation for the optimal
 174 translational matching between A and B .

175 Another interesting conclusion from the above discussion is that if there exists an instance
 176 $B' \subseteq B$ which is identical to A under translation \mathcal{T} (i.e., the bipartite matching cost between
 177 B' and $\mathcal{T}(A)$ is 0), then $\mathcal{T} \in \mathbb{T}$.

178 The above theorems and corollary suggest an efficient way to identify a small number
 179 of translations that enable us to obtain approximate solutions to the translational pattern
 180 matching problem. The size of the candidate set is near optimal. This can be easily seen
 181 from the following simple example in 1-D: Consider $A = \{1, 2\}$ and $B = \{1, 2, \dots, n\}$; then
 182 all the n translations that align 1 in A to any of the $n - 1$ points $\{1, 2, \dots, n - 1\}$ in B are
 183 optimal translations.

184 After obtaining all the candidate translations, the approximate optimal matching can
 185 then be computed by solving a min-cost partial matching problem for fixed point sets $\mathcal{T}(A)$
 186 and B for every candidate \mathcal{T} .

187 3.1 Overview of Techniques

188 Our main idea for constructing a small candidate set is via space discretization. We are able
 189 to prove a locality property for the pattern search problem: if the distance between two
 190 translations \mathcal{T}_1 and \mathcal{T}_2 (viewed as points in \mathbb{R}^d) is close compared to their closest distance to
 191 any point in the reference set P , then for any instance $B' \subseteq B$, $\Delta(\mathcal{T}_1(A), B')$ and $\Delta(\mathcal{T}_2(A), B')$
 192 are also close. This suggests that we can decompose \mathbb{R}^d into “small” regions, so that the
 193 every region has a small diameter, comparing to its distance to P . For any $B' \subseteq B$, let \mathcal{T}_O be
 194 the translation that discovers B' , and \mathcal{T}_O lies in some “small” region C . Then, any $\mathcal{T}' \in C$
 195 approximately discovers B' . This means that, if we choose one arbitrary point from each
 196 “small” region to form the candidate set \mathbb{T} , it is guaranteed that any instance B' will be
 197 approximately discoverable by some translation in \mathbb{T} . The details will be shown in Section 4.

198 However, making all regions of the entire space “small” seems to be challenging, if not
 199 impossible. Existing space discretization techniques, such as [7, 3, 15], are only able to ensure
 200 that regions distant from the points in P are “small”. For regions close to points in P (called
 201 “close” regions), new techniques are needed to select their candidate translations. In Section
 202 5, we discuss how to choose translations from “close” regions so that every instance B' whose
 203 corresponding optimal translation falls in some “close” region can also be approximately
 204 discoverable.

205 In Section 4.4, we also describe how to use weak ϵ -net to “sketch” P (with size mn) using
 206 a much smaller set Q of size $O(n)$. This will allow us to significantly reduce the size of
 207 discretization (and thus the size of the candidate set) from $\tilde{O}(mn)$ to $\tilde{O}(n)$, which is near
 208 optimal. We are able to show that such a sketching still preserves the locality property of
 209 the pattern search problem.

210 4 Locality Based Discretization for Pattern Search

211 In this section, we present a discretization approach for the pattern search problem, based
 212 on the locality property of the problem.

213 4.1 Locality of Pattern Search

214 In the context of pattern search, locality refers to the following observation: for any instance
 215 $B' \subseteq B$ and two translations \mathcal{T}_1 and \mathcal{T}_2 , if \mathcal{T}_1 and \mathcal{T}_2 are close to each other, their induced
 216 minimum bipartite matching costs between the translations of A and B' are also close. Let
 217 $\Delta(X, Y)$ denote the minimum bipartite matching cost between two point sets X and Y . The
 218 following lemma shows the locality property with respect to the distance between translations
 219 and a point in the reference set P . (See Appendix for the proof.)

220 ► **Lemma 5.** Let \mathcal{T}_1 and \mathcal{T}_2 be two translations in \mathbb{R}^d , and $p \in P$ be the nearest neighbor
 221 of \mathcal{T}_1 in the reference set P . If $\|\mathcal{T}_1 - \mathcal{T}_2\| \leq \epsilon\|p - \mathcal{T}_1\|$ for some constant $0 < \epsilon < 1$, then
 222 $|\Delta(\mathcal{T}_1(A), B') - \Delta(\mathcal{T}_2(A), B')| \leq \epsilon\Delta(\mathcal{T}_1(A), B')$ for any instance $B' \subseteq B$.

223 The above locality property suggests the following discretization approach to find a
 224 candidate set for the pattern search problem. The idea is to decompose \mathbb{R}^d into a number of
 225 “small” regions. Each region C is “small” enough in the sense that the minimum distance r
 226 between C and the points in P is large, comparing to the diameter $D(C)$ of C , i.e., $D(C) \leq \epsilon r$
 227 for some constant $0 < \epsilon < 1$. With such a discretization, we may then simply choose one
 228 arbitrary translation from each region to form the candidate set \mathbb{T} . To see that this is indeed
 229 the desired candidate set, consider any instance $B' \subseteq B$. Let \mathcal{T} be the translation that
 230 discovers B' , C' be the region containing \mathcal{T} , and $\mathcal{T}_C \in \mathbb{T}$ be the translation chosen from C .
 231 Then, by Lemma 5, we know that B' is $(1 + \epsilon)$ -approximately discoverable at \mathcal{T}_C .

232 **4.2 Space Discretization and Close Regions**

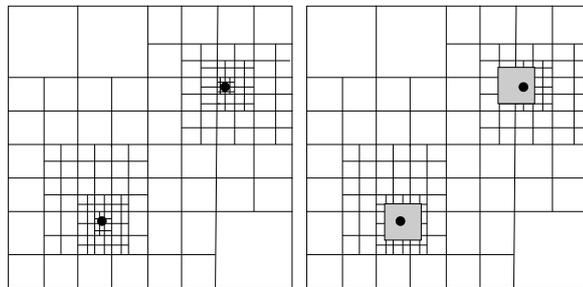
233 Unfortunately, an exact implementation of the above space discretization is not possible.
 234 This is because the size of some regions can be infinitely small if the distance of the region to
 235 a point in P is small enough. This means that an infinite number of regions can be generated
 236 around every point in P (see Figure 1). To overcome this difficulty, a possible way is to
 237 utilize some of the known space discretization techniques, such as [7, 3, 15]. However, a
 238 common issue of such techniques is that only part of the resulting regions can be viewed as
 239 “small” (i.e., $D \leq \epsilon r$, where D is the diameter of the region and r is the minimum distance
 240 between the a point in P and the region). Such regions are distant from points in P . All
 241 other regions are in close proximity to points in P , and cannot be viewed as “small”, even
 242 though their diameters might be small. We call such regions as “close” regions.

243 Clearly, for “small” regions, it
 244 will be sufficient (by Lemma 5) to
 245 choose one point arbitrarily from
 246 each of them and include it into
 247 the candidate set \mathbb{T} . The main
 248 issue is, thus, how to select candi-
 249 date translations from the “close”
 250 regions. We will discuss our ideas
 251 on close regions in next section.

252 In this paper, we use the techni-
 253 que in [7] for space discretization.
 254 The main reason is that using this
 255 technique, we have some good geo-
 256 metric properties on close regions.
 257 This will help us ensure the correct-
 258 ness of our proposed approach and
 259 simplify the analysis.

260 For self completeness, below we summarize the space discretization technique in [7]. The
 261 main technique in [7] is an algorithm $\text{AIDecomposition}(P, \beta, \gamma)$, where P is a point set in \mathbb{R}^d
 262 and $0 < \beta, \gamma < 1$ are two small constants (to be determined in later analysis). The following
 263 lemma is the main result of the algorithm.

264 ► **Lemma 6.** [7] $\text{AIDecomposition}(P, \beta, \gamma)$ generates a partition of \mathbb{R}^d in $O_{d,\beta,\gamma}(|P| \log|P|)$
 265 time, where each region C of the partition satisfies one of the following conditions.



252 **Figure 1** Illustrative figures for “small” and “close” regions. Left Figure: If every region needs to be “small”, then
 253 an infinite number of regions will be generated around each
 254 point in P . Right Figure: Possible “close” regions to prevent
 255 from yielding an infinite number of “small” regions.

- 266 1. C is associated with a subset V of P and a point $v \in V$, such that
 267 a. The diameter $D(V)$ of V is no larger than βr , where r is the closest distance between
 268 a point in C and a point in V .
 269 b. $\|v - u\| \leq \gamma \|v' - u\|$ and $D(C) \leq \beta \|v' - u\|$, for any point $u \in C$ and any point
 270 $v' \in P \setminus V$
 271 2. $D(C) \leq \beta r$, where r is the closest distance between a point in C and a point in P .

272 The regions that satisfy condition 1 are the close regions in our previous discussion,
 273 and the regions that satisfy condition 2 correspond to the small regions. Note that for a
 274 close region C generated by AIDecomposition, it is close to the associated point set $V \subset P$,
 275 comparing to points in $P \setminus V$, where the closeness is controlled by the parameter γ . There
 276 are also some other interesting properties of close regions generated by AIDecomposition
 277 shown in Lemma 6. These properties will prove to be useful in later analysis.

278 4.3 Reducing the Number of Regions

279 If we directly apply the above AIDecomposition technique or any other space discretization
 280 technique (such as [3, 15]) to the reference set P , at least $\Omega(mn)$ regions will be generated,
 281 since $|P| = mn$. This results in a candidate set of size $\Omega(mn)$, which could be significantly
 282 larger than $O(n)$, *i.e.*, the maximum number of possible translations that could yield a
 283 perfect matching for $\mathcal{T}(A)$ and B . Thus, it is tempting to ask whether it is possible to
 284 construct a discretization with size only near $O(n)$.

285 A natural approach for size reduction is to use a smaller set Q to “sketch” P , and build a
 286 discretization for Q , instead of P . Let $\xi > 1$ and $\mu > 0$ be some given constants. We require
 287 that Q be (ξ, μ) -dense for P , defined as follows.

288 ► **Definition 7.** A point set $Q \subset \mathbb{R}^d$ is called (ξ, μ) -dense for P , if for any point $\mathcal{T} \in \mathbb{R}^d$, it
 289 satisfies $\mu \|p_\xi - \mathcal{T}\| \geq \|q - \mathcal{T}\|$, where p_ξ is the m/ξ -th closest point in P to \mathcal{T} , and $q \in Q$ is
 290 the nearest neighbor of \mathcal{T} in Q .

291 In other words, points in Q are “dense” enough, so that for any $\mathcal{T} \in \mathbb{R}^d$, it is possible to
 292 find a point $q \in Q$ that is closer to, or not much farther away from \mathcal{T} than p_ξ . Using such a
 293 formulation for “dense” allows us to use $\|q - \mathcal{T}\|$ to effectively lower bound $\Delta(\mathcal{T}(A), B')$ for
 294 any B' . Let ϕ be the matching realizing the minimum cost bipartite matching between $\mathcal{T}(A)$
 295 and B' . Then, we have

$$296 \quad \|q - \mathcal{T}\| \leq \mu \|p_\xi - \mathcal{T}\| \leq \sum_{p \in P(\phi)} \mu \|p - \mathcal{T}\| / ((1 - 1/\xi)m) = \mu \Delta(\mathcal{T}(A), B') / ((1 - 1/\xi)m). \quad (1)$$

297 Using an argument similar to the one in Lemma 5, we have the following improved version
 298 of locality property.

299 ► **Lemma 8.** Let \mathcal{T}_1 and \mathcal{T}_2 be two translations in \mathbb{R}^d , and $p \in Q$ be the nearest neighbor of \mathcal{T}_1
 300 in Q . If $\|\mathcal{T}_1 - \mathcal{T}_2\| \leq \beta \|p - \mathcal{T}_1\|$ for constant $0 < \beta < 1$, then $|\Delta(\mathcal{T}_1(A), B') - \Delta(\mathcal{T}_2(A), B')| \leq$
 301 $\beta \mu (1 - 1/\xi)^{-1} \Delta(\mathcal{T}_1(A), B')$ for any instance $B' \subseteq B$.

302 The above lemma enables us to use Q for the space discretization. More specifically, we
 303 run AIDecomposition(Q, β, γ) on an (ξ, μ) -dense Q (with β, ξ, μ and γ to be determined
 304 later). This yields a near linear size (in terms of the size of Q) discretization. From previous
 305 discussion, we know that for any instance $B' \subseteq B$, if the translation that discovers B' lies
 306 in a small region (note that since the discretization is based on Q instead of P , “small

307 region” now means that their diameters are small compared to the distances to their nearest
 308 neighbors in Q), then any translation in the small region will $(1 + \epsilon)$ -approximately discover
 309 B' , if parameters β , ξ , μ and γ are properly chosen according to the desired approximate
 310 ratio ϵ . A formal argument will be provided later when analyzing the correctness of our
 311 algorithm. The remaining main challenge is then to deal with the case that the translation
 312 discovering B' lies in a close region. This will be covered in the next section.

313 4.4 Finding (ξ, μ) -dense Q for P

314 To conclude this section, we briefly describe how to find a (ξ, μ) -dense set Q for the discretiz-
 315 ation.

316 A set Q that is (ξ, μ) -dense can actually be found by constructing a weak ϵ -net of P
 317 [16]. ϵ -net is an important concept in combinatorial and computational geometry, and has
 318 been extensively studied in the past. For our problem, we use weak ϵ -net for axis-aligned
 319 hypercubes (*i.e.*, hyper-boxes with equal edge length in every direction). Below is the
 320 definition.

321 ► **Definition 9.** *Let P be any point set in \mathbb{R}^d , and ϵ be any small constant between 0 and 1.
 322 A point set $Q \subset \mathbb{R}^d$ is called a weak ϵ -net of P for axis-aligned hypercubes, if any axis-aligned
 323 hypercube G containing $\epsilon|P|$ or more points in P also contains at least one point in Q .*

324 Note that ϵ -net is a much more general concept than the above definition. In this
 325 paper, we only consider weak ϵ -net for axis-aligned hypercubes. For convenience, we will use
 326 thereafter the term “ ϵ -net” without specifying “weak” and “axis-aligned hypercubes”. For
 327 any constant $c > 1$, it is easy to see that a $1/cn$ -net Q of P is (c, \sqrt{d}) -dense.

328 A small-size ϵ -net for the reference set P can be built efficiently. (We leave the proofs of
 329 the following lemmas to the end of the paper.)

330 ► **Lemma 10.** *For any point multi-set $P \subset \mathbb{R}^d$ with size mn and any constant $c > 0$, a
 331 $1/cn$ -net of P with size $O_d(n)$ can be generated in $O_d(nm \log nm)$ time.*

332 ► **Lemma 11.** *A $1/cn$ -net of the reference set P with size $O_d(n \log n)$ can be generated in
 333 $O_d(n \log n)$ time for any constant $c > 0$ with probability at least $1 - 1/n$.*

334 5 Handling Close Regions

335 In last section, we discuss how to use (ξ, μ) -dense Q to obtain a space discretization and how
 336 to select translations from small regions. In this section, we show how to pick translations
 337 from close regions (*i.e.* regions that satisfy condition 1 in Lemma 6) so that they will be
 338 good approximations for all those translations that fall in close regions and discover some
 339 instances of B .

340 Below, we assume that γ , β , μ and ξ are chosen such that $\gamma < (16\sqrt{d} + 1)^{-1}$, $4\sqrt{d}\beta < 1$
 341 and $\xi = 12d$. Let $B' \subseteq B$ be an instance of A and \mathcal{T}_O be the translation that discovers B'
 342 and lies in some close region C . Denote by $V \subset Q$ and $v \in V$, respectively, the subset of
 343 points in Q and its representative associated with C , as indicated in condition 1 of Lemma
 344 6. Let ϕ_O be the matching between $\mathcal{T}_O(A)$ and B' that realizes the minimum bipartite
 345 matching cost. Let G_v be the smallest axis-aligned box containing V . We consider 2 cases:
 346 $|G_v \cap P(\phi_O)| \geq 2m/3$ or $|G_v \cap P(\phi_O)| < 2m/3$.

347 ► **Lemma 12.** *If $|G_v \cap P(\phi_O)| \geq 2m/3$, $\mathcal{T}_O = v$.*

348 **Proof.** Assume by contradiction that $\mathcal{T}_O \neq v$. For any matching ϕ and any translation \mathcal{T} ,
 349 we use notation $C_\phi(\mathcal{T})$ to denote the matching cost between $\mathcal{T}(A)$ and B' under ϕ . In the
 350 following, we analyze how the value of $C_{\phi_O}(\mathcal{T})$ changes, where variable \mathcal{T} is initially \mathcal{T}_O ,
 351 and then changed to v . Note that $C_{\phi_O}(\mathcal{T}) = \sum_{p \in P(\phi_O)} \|p - \mathcal{T}\| = \sum_{p \in G_v \cap P(\phi_O)} \|p - \mathcal{T}\| +$
 352 $\sum_{p \in P(\phi_O) \setminus G_v} \|p - \mathcal{T}\|$.

353 To estimate the change of $\sum_{p \in G_v \cap P(\phi_O)} \|p - \mathcal{T}\|$, we note that for any $p \in G_v \cap P(\phi_O)$,
 354 $\|p - v\| \leq \sqrt{d}D(V) \leq \sqrt{d}\beta\|\mathcal{T}_O - v\|$, where the last inequality is from Condition 1 of
 355 Lemma 6 and the fact that \mathcal{T}_O is in C . From triangle inequality, we have $\|p - \mathcal{T}_O\| \geq$
 356 $\|v - \mathcal{T}_O\| - \|v - p\| \geq (1 - \sqrt{d}\beta)\|v - \mathcal{T}_O\|$. Thus, we get $\|p - \mathcal{T}_O\| - \|p - v\| \geq (1 - 2\sqrt{d}\beta)\|v - \mathcal{T}_O\|$.
 357 This means that moving \mathcal{T} from \mathcal{T}_O to v reduces the value of $\sum_{p \in G_v \cap P(\phi_O)} \|p - \mathcal{T}\|$ by at
 358 least $(2m/3)(1 - 2\sqrt{d}\beta)\|v - \mathcal{T}_O\|$.

359 For the term of $\sum_{p \in P(\phi_O) \setminus G_v} \|p - \mathcal{T}\|$, we know (from triangle inequality and the as-
 360 sumption that $|P(\phi_O) \setminus G_v| \leq m/3$) that its change is smaller than $(m/3)\|v - \mathcal{T}_O\|$. Since
 361 $1 - 2\sqrt{d}\beta > 1/2$ (by the assumption that $4\sqrt{d}\beta < 1$), we get $(2m/3)(1 - 2\sqrt{d}\beta)\|v - \mathcal{T}_O\| >$
 362 $(m/3)\|v - \mathcal{T}_O\|$ when $\mathcal{T}_O \neq v$. Therefore, we have $C_{\phi_O}(v) < C_{\phi_O}(\mathcal{T}_O)$ (from previous discus-
 363 sion). However, this results in a contradiction, since from definition, \mathcal{T}_O discovers B' and
 364 thus should have the minimum matching cost between A and B' under any translation.

365 Thus, the lemma follows. ◀

366 For the case $|G_v \cap P(\phi_O)| < 2m/3$,
 367 we have the following lemma.

368 **► Lemma 13.** *If $|G_v \cap P(\phi_O)| < 2m/3$,
 369 then for any $\mathcal{T}' \in C$, $\Delta(\mathcal{T}'(A), B') \leq$
 370 $(1 + 48\beta\sqrt{d})\Delta(\mathcal{T}_O(A), B')$.*

371 **Proof.** For any matching ϕ and any
 372 translation \mathcal{T} , we use notation $C_\phi(\mathcal{T})$
 373 to denote the matching cost between
 374 $\mathcal{T}(A)$ and B' under ϕ . We prove this
 375 lemma by showing that $C_{\phi_O}(\mathcal{T}') \leq$
 376 $(1 + 48\beta\sqrt{d})C_{\phi_O}(\mathcal{T}_O)$; the lemma then
 377 follows, since $C_{\phi_O}(\mathcal{T}') \geq \Delta(\mathcal{T}'(A), B')$.

378 Let P' be the closest $5m/6$ points
 379 to \mathcal{T}_O in $P(\phi_O)$. Since $|G_v \cap P(\phi_O)| <$
 380 $2m/3$, we have $|P' \setminus G_v| \geq m/6$.

381 For analysis purpose, imagine that
 382 we “attach” $2d$ axis-aligned boxes
 383 $\{G_1, G_2, \dots, G_{2d}\}$ to each face of G_v , with centers aligned in G_v (see Figure 2 for an example
 384 in 2D), and each box has equal edge length r , where r is the smallest positive number such that
 385 P' is contained in the union of $\{G_1, G_2, \dots, G_{2d}\}$ and G_v . Let $F = G_v \cup G_1 \cup G_2 \dots \cup G_{2d}$.

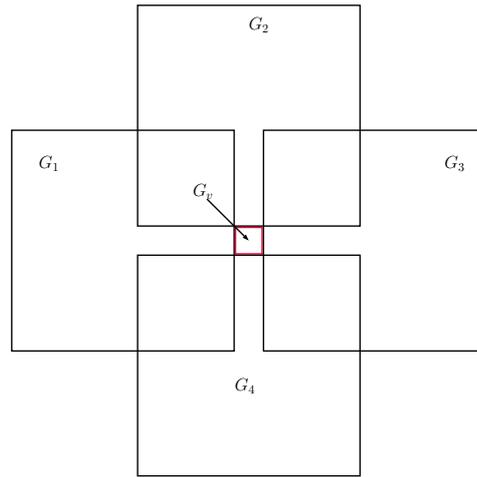
386 By the fact that $|P' \setminus G_v| \geq m/6$, we know that one box of $\{G_1, G_2, \dots, G_{2d}\}$ contains
 387 more than $m/12d$ points in P . Since $\xi = 12d$ and Q is a $1/n\xi$ -net of P , we also know that
 388 the box contains a point q_t from Q . Thus, F contains a point q_t in $Q \setminus V$.

389 From Lemma 6, we have $\|\mathcal{T}_O - v\| \leq \gamma\|\mathcal{T}_O - q_t\|$. Thus,

$$390 \quad \|v - q_t\| \geq (1/\gamma - 1)\|\mathcal{T}_O - v\|. \tag{2}$$

391 Let L_v denote the edge length of G_v . Then, we have $L_v \leq D(V) \leq \beta\|\mathcal{T}_O - v\|$. Since
 392 $\beta < 1/4\sqrt{d} < (1/4\sqrt{d})(1/\gamma - 1)$, from (2) we get

$$393 \quad L_v \leq \|v - q_t\|/4\sqrt{d}. \tag{3}$$



394 **■ Figure 2** Illustration of the arrangement of $\{G_1, G_2, \dots, G_{2d}\}$ and G_v .

29:10 Small Candidate Set for Translational Pattern Search

394 Let L denote the length of F : $L = 2r + L_v$. L is clearly no smaller than $\|v - q_t\|/\sqrt{d}$ in
 395 order to contain both v and q_t . From (3), we have $L_v \leq r$ and $r \geq 3\|v - q_t\|/8\sqrt{d}$. By (2)
 396 and the assumption that $1/\gamma - 1 \geq 16\sqrt{d}$, we get

$$397 \quad r \geq 3(1/\gamma - 1)\|\mathcal{T}_O - v\|/8\sqrt{d} \geq 6\|\mathcal{T}_O - v\|. \quad (4)$$

398 Let O_v denote the center of G_v . Then, $\|O_v - v\| \leq \sqrt{d}L_v \leq \sqrt{d}D(V) \leq \sqrt{d}\beta\|\mathcal{T}_O - v\| \leq$
 399 $\|\mathcal{T}_O - v\|/4$. Thus, from triangle inequality we have

$$400 \quad \|\mathcal{T}_O - v\| \leq 5\|\mathcal{T}_O - v\|/4. \quad (5)$$

401 Combining (4) and (5) gives us $\|O_v - \mathcal{T}_O\| \leq 5r/24$.

402 From the definition of r , we know that there must exist a point $p' \in P'$ on the boundary
 403 of F . From the arrangement of $\{G_1, G_2, \dots, G_{2d}\}$ and G_v , we have $\|O_v - p'\| \geq r/2$. Thus,
 404 $\|\mathcal{T}_O - p'\| \geq r/2 - 5r/24 > r/4$ (by triangle inequality).

405 Also, from the fact that $L_v \leq r$, we know that F can be covered by a box centered at O_v
 406 and with edge length $3r$. Since $q_t \in F$, we have $\|O_v - q_t\| \leq 3\sqrt{d}r/2$. Combining this with
 407 the fact that $\|O_v - \mathcal{T}_O\| \leq 5r/24$, we get $\|\mathcal{T}_O - q_t\| \leq 3\sqrt{d}r/2 + 5r/24 \leq 2\sqrt{d}r$. Thus, we
 408 have $\|\mathcal{T}_O - q_t\|/\|\mathcal{T}_O - p'\| \leq 8\sqrt{d}$.

409 In summary, from the above discussion and Lemma 6, we have the following.

- 410 1. There exists $q_t \in Q$ such that the following inequality holds $\|\mathcal{T}_O - q_t\|/\|\mathcal{T}_O - p'\| \leq 8\sqrt{d}$,
 411 where $p' \in P(\phi_O)$ satisfies the condition that less than $5m/6$ points in $P(\phi_O)$ are closer
 412 to \mathcal{T}_O than it.
- 413 2. Inequality $\|\mathcal{T}_O - \mathcal{T}'\| \leq \beta\|\mathcal{T}_O - q_t\|$ holds for any $\mathcal{T}' \in C$.

414 Following a similar argument given in Lemma 5, we can show that $|C_{\phi_O}(\mathcal{T}_O) - C_{\phi_O}(\mathcal{T}')| \leq$
 415 $48\beta\sqrt{d}C_{\phi_O}(\mathcal{T}_O)$. This concludes the proof. \blacktriangleleft

416 Now for any $\epsilon > 0$, if we set $48\beta\sqrt{d} \leq \epsilon$, then by Lemmas 12 and 13, we know that
 417 if \mathcal{T}_O lies in a close region C generated by AIDecomposition, either v discovers B' (if
 418 $|G_v \cap P(\phi_O)| \geq 2m/3$), or an arbitrary $\mathcal{T}' \in C$ $(1 + \epsilon)$ -approximately discovers B' (if
 419 $|G_v \cap P(\phi_O)| < m/3$). Therefore, we may put v and an arbitrary point in C into the
 420 candidate set. This ensures that B' is $(1 + \epsilon)$ -approximately discoverable by at least one of
 421 these two points when \mathcal{T}_O lies in C .

422 **6** The Algorithm and Analysis

423 In this section, we summarize the discussion so far and provide the algorithm to generate the
 424 candidate set of translations. The following Algorithm 1 shows in details how to generate
 425 the candidate set \mathbb{T} .

426 Depending on the method chosen to construct the ϵ -net in Step 3 (Lemma 10 or Lemma
 427 11), the size of Q is $O(n)$ or $O(n \log n)$, the size of the discretization (in terms of number of
 428 regions) generated in step 4 is $O(n \log n)$ or $O(n \log^2 n)$, and the total running time of the
 429 algorithm is $O(mn \log mn)$ or $O(n \log^2 n)$. No matter which algorithm is chosen to construct
 430 \mathbb{T} , we have the following lemma.

431 **► Lemma 14.** *For any instance $B' \in B$ of A , there exists at least one translation $\mathcal{T} \in \mathbb{T}$,
 432 such that \mathcal{T} $(1 + \epsilon)$ -approximately discovers B' .*

Algorithm 1 Generate-Candidate-Set

Input: Point sets A and B of \mathbb{R}^d with $|A| \leq |B|$. Approximate factor $0 < \epsilon < 1$.

Output: A set \mathbb{T} of translations in \mathbb{R}^d , such that each instance B' of A is $(1+\epsilon)$ -approximately discoverable.

- 1: Initialize \mathbb{T} to be \emptyset .
- 2: Initialize constants β, γ, ξ , such that: $\xi = 12d$, $48\beta\sqrt{d} \leq \epsilon$, $\gamma < (16\sqrt{d} + 1)^{-1}$, $4\sqrt{d}\beta < 1$ and $\sqrt{d}\beta(1 - 1/\xi)^{-1} \leq \epsilon$.
- 3: Build a $1/\xi n$ -net Q for P , where P is the reference set.
- 4: Run $\text{AIDecomposition}(Q, \beta, \gamma)$ to generate a discretization which decomposes \mathbb{R}^d into close regions and small regions.
- 5: For each small region C , pick an arbitrary point from C and put it into \mathbb{T} .
- 6: For each close region C , suppose it is associated with point set V and $v \in V$. Pick an arbitrary point p in C . Put both v and p into \mathbb{T} .
- 7: Output \mathbb{T} as the result.

Proof. Let \mathcal{T}_O be the translation that discovers B' .

If \mathcal{T}_O lies in a small region C , let $\mathcal{T}_C \in \mathbb{T}$ be the point chosen from C in step 5 of Algorithm 1. Let q be the nearest neighbor of \mathcal{T}_O in Q . By Lemma 6, we know that $\|\mathcal{T}_C - \mathcal{T}_O\| \leq \beta\|q - \mathcal{T}_O\|$. By Lemma 8 and the fact that Q is (ξ, \sqrt{d}) -dense, we have $\Delta(\mathcal{T}_C(A), B') \leq (1 + \sqrt{d}\beta(1 - \xi)^{-1})\Delta(\mathcal{T}_O(A), B') \leq (1 + \epsilon)\Delta(\mathcal{T}_O(A), B')$ (The last inequality comes from choice of parameters in Algorithm 1). Thus, we know that B' is $(1 + \epsilon)$ -approximately discoverable at $\mathcal{T}_C \in \mathbb{T}$.

If \mathcal{T}_O lies in a close region C , let $v \in V \subset Q$ be the representative point associated with C as stated in Lemma 6. Let $\mathcal{T}_C \in \mathbb{T}$ be the point chosen from C in step 6 of Algorithm 1. Then, by Lemmas 12 and 13, we know that either v discovers B' , or $\Delta(\mathcal{T}_C(A), B') \leq (1 + 48\beta\sqrt{d})\Delta(\mathcal{T}_O(A), B') \leq (1 + \epsilon)\Delta(\mathcal{T}_O(A), B')$. This means that either v or \mathcal{T}_C $(1 + \epsilon)$ -approximately discovers B' .

This completes the proof. \blacktriangleleft

From the above analysis, we immediately have our main results, Theorems 2 and 3.

7 Constructing ϵ -net for Hypercubes for P

To conclude this paper, we introduce efficient algorithms to construct a weak ϵ -net for the reference set P with axis-aligned hypercubic ranges. From the well known ϵ -net theorem, we know that a random sample of size $O((d'/\epsilon) \log(d'/\epsilon) + \log n/\epsilon)$ from P , where d' is the VC-dimension of the range space defined by hypercubes in \mathbb{R}^d (it is known that $d' \leq 2d$), is an ϵ -net with probability at least $1 - 1/n$. There are several previous results on ϵ -net for simple shapes like axis aligned rectangles, halfspace and disks in 2 or 3 dimension [20, 8, 2], which provide methods to build smaller size ϵ -nets. [19] provides a mathematical construction of ϵ -nets for axis-aligned hypercubes of size $O(1/\epsilon)$, which is optimal in size, although its efficient (*i.e.* in near $O(|P|)$ time) algorithmic implementation is unknown. [12] introduces a method to construct ϵ -nets for axis-aligned rectangles in any fixed dimension, which can be applied to generate the (ξ, μ) -dense subset Q . The running time of this method is $O(|P| \log^d |P|)$.

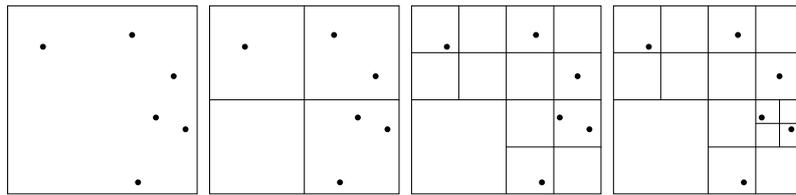
We show that if we further restrict the shapes to hypercubes, we are able to obtain an optimal size (*i.e.* $O(1/\epsilon)$) weak ϵ -net more efficiently. In the following we show how to deterministically construct a linear size $1/n$ -net Q for any multi-set P of size $O(mn)$, *i.e.*,

29:12 Small Candidate Set for Translational Pattern Search

463 a point set Q of size $O(n)$ such that if an axis-aligned hypercube G contains m points in
 464 P , it then contains at least one point in Q . The time of the construction is $O(nm \log nm)$
 465 (Recall that the size of P is mn). Clearly, the same space and time complexity bounds for a
 466 $1/cn$ -net Q for any constant c are also achievable, thus proving Lemma 10. We also note that
 467 by applying the ϵ -net theorem, it is possible to construct such a Q of larger size ($O(n \log n)$),
 468 but in shorter ($O(n \log n)$) time, with high probability. This allows us to make a trade-off
 469 between the size of Q and the time complexity of the construction. We leave the discussion
 470 of the alternative construction to the end of section, and focus on the deterministic linear
 471 size ϵ -net construction in the following.

472 The construction is based on the quad-tree decomposition technique, which recursively
 473 partitions the regions inside the quad-tree boxes, and uses a 2^d -way tree structure to represent
 474 the partition. To build a quad-tree for P , we first start with a bounding box G which contains
 475 all points in P and is the root of the quad-tree. We then decompose G into 2^d smaller boxes
 476 with equal size, with each of them being a child of G . For each child box, we recursively
 477 perform the same decomposition. The recursion stops when a box contains no more than 1
 478 point in P . The quad tree decomposition for P can be performed within $O(|P| \log |P|)$ time
 479 by maintaining a sorted list of P for each of the d axes [6], and compressing the tree properly
 480 to handle empty boxes during the decomposition.

481 Figure 3 below shows an example of quad-tree decomposition.



■ **Figure 3** Example of quad-tree decomposition.

482 A quad-tree decomposition may produce a large number of empty boxes, when a large
 483 number of points are aggregated in some region (see left of Figure 4 for an example). To
 484 resolve this issue, when decomposing a box G in the quad-tree decomposition, we first
 485 perform a quad-tree compression, which directly computes the smallest quad-tree box G_C
 486 that contains all the points in $P \cap G$ (see the right side of Figure 4). Then the quad-tree
 487 decomposition can continue on G_C . This will avoid generating many unnecessary empty
 488 boxes. Note that this compression step is not required if points are not concentrated, *i.e.*, if
 489 decomposing G yields at least 2 nonempty boxes (*i.e.*, containing points in P). In this case,
 490 we decompose G in the standard fashion.

491 With this compression step, the running time of the quad-tree decomposition is still
 492 $O(|P| \log |P|)$ [6].

493 Algorithm 2 and Algorithm 3 describe our quad-tree decomposition-based method for
 494 producing a weak ϵ -net Q . The decomposition scheme is a modification of the standard
 495 quad-tree decomposition. The main routine Algorithm 2 outputs the ϵ -net Q , together with
 496 a set U of boxes which is for analysis purpose. Algorithm 3 is the body for the recursion.

497 The Algorithm 2 and 3 are essentially trimmed versions of the standard quad-tree
 498 decomposition (by not decomposing some of the boxes in the process). Given a box G
 499 that contains multiple points in P , instead of simply decomposing it into 2^d sub-boxes and
 500 recursively building the quad-tree on them (which could generate boxes with few points in
 501 it and thus results in a quad-tree with high complexity), Algorithm 3 iteratively performs
 502 the quad-tree decomposition on *only one* sub-box which contains the maximum number of

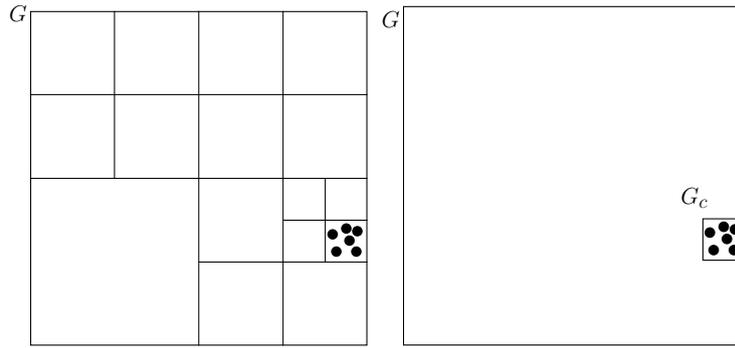


Figure 4 Example of quad-tree compression. It is possible that points in a box are aggregated at some location. Directly applying the quad-tree decomposition will generate many empty boxes. We can directly compute a box to contain all these points without really performing the decomposition.

Algorithm 2 Construct- ϵ -Net

Input: A set $P \subset \mathbb{R}^d$

Output: An ϵ -net Q . A set U of \mathbb{R}^d boxes.

- 1: Initialize Q, U as empty sets. Initialize G as a box that contains P .
 - 2: Start recursion by running Decompose-Single-Box subroutine on G
-

503 points in P , and tries to identify a box G' with the following properties. When the iteration
 504 (from step 4 to step 8) stops (at step 4 or 7), G' satisfies the following 2 conditions

- 505 1. There are less than $m/2^{d+1}$ points in $P \cap G \setminus G'$.
- 506 2. (a) All points in $P \cap G'$ have the same location, OR (b) There are at least $m/2^{d+1}$ points
 507 in $P \cap G \setminus G''$, where G'' is the child box of G' with the most number of points in P .

508 Only in case 2(b) we perform the recursion on the boxes generated by the decomposition of
 509 G' . See Figure 5 for illustration. In addition, we do not decompose G when there are only a
 510 small number ($\leq m/2^{d+1}$) of points in it. Since the algorithm is a trimmed version of the
 511 standard quad-tree decomposition, the running time is thus $O(|P| \log |P|) = O(mn \log mn)$.

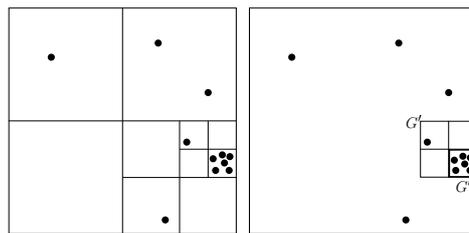


Figure 5 To achieve better performance, Algorithm 3 uses an iteration to find out G' with the desired properties. Recursion continues only (on sub-boxes of G') if $P \cap G \setminus G''$ contains an enough number of points, where G'' is the quad-tree child box of G' with the most number of points in P . This can greatly reduce the number of boxes generated.

512 It is quite clear that the size of Q is $O(n)$. The Decompose-Single-Box procedure stops
 513 immediately once the condition $|G \cap P| \leq m/2^{d+1}$ is satisfied. The procedure also makes sure
 514 that for any G'' of the 2^d child boxes generated for G (if the decomposition and recursion
 515 occur), inequality $|G \setminus G''| \geq m/2^{d+1}$ holds. This implies that the size of the recursion tree,
 516 and thus the size of U and Q , is $O(mn/(m/2^{d+1})) = O(n)$.

Algorithm 3 Decompose-Single-Box

Input: A box G

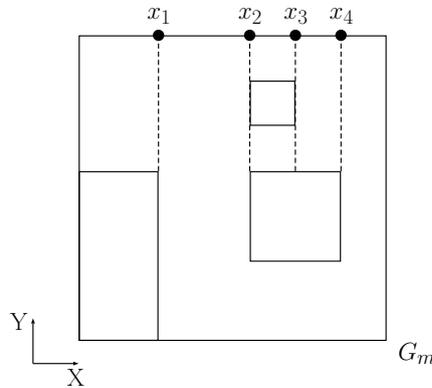
Output: A sub-quad tree with G as the root.

- 1: Add G into U . Add all the vertices of G to Q .
 - 2: If G contains $\leq m/2^{d+1}$ points in P , return.
 - 3: Initialize variable G' to be the box G .
 - 4: If all points in P that lie in G' coincide at point p . Put p and all vertices of G' into Q . Put G' into U . Return.
 - 5: Update variable G' to be the resulting box from the quad-tree compression of the current G' , if necessary (See Appendix A.3).
 - 6: Decompose G' equally into 2^d sub-boxes. Let G'' be the one that contains the most number of points in P .
 - 7: If $G \setminus G''$ contains $\geq m/2^{d+1}$ points in P , recursively call Decompose-Single-Box on the 2^d sub-boxes generated in the above step. Then return.
 - 8: Otherwise, update variable G' to be G'' . Go to step 4.
-

517 ► **Lemma 15.** *Set Q generated by the above algorithm is a weak $1/n$ -net for P , i.e., if any*
 518 *axis-aligned hypercube G_m contains at least m points in P , then G_m contains at least 1 point*
 519 *in Q .*

520 **Proof.** Let U_m denote the subset $U_m \subseteq U$ of hyperboxes G' in U such that the interior of
 521 G_m intersects G' .

522 For each coordinate axis e of \mathbb{R} , let $F(e)$ be the set of faces f of boxes in U_m , such that f
 523 is perpendicular to e and intersects the interior of G_m . Let the *cutting number* $x(e)$ of e be
 524 the possible number of distinct coordinate values of faces in $F(e)$ in the e axis. (See Figure 6
 525 for an example.) We consider two cases.



■ **Figure 6** Example of cutting number for X axis in an configuration of interior G_m . In this example there are 3 boxes of U_m intersecting G_m . There are 4 different x values for faces of these boxes that lies in G_m and are perpendicular to the X axis. Thus the cutting number for X axis in this example is 4.

526 **Case 1.** $x(e) \leq 1$ for any axis e . Then boxes in U partitions G_m into no more than
 527 2^d regions, since in every direction G_m is “cut” by boxes in U at most once. Since G_m
 528 contains at least m points in P , one of the regions will contain strictly more than $m/2^d$
 529 points. However, from Algorithm 3, we know that all of the regions formed by U can only
 530 have no more than $m/2^{d+1}$ points, with the exception that for some regions, all its contained

531 points in P have the same location p . Thus, G_m intersects such a region and contains p .
 532 Since $p \in Q$ from Algorithm 3 (see Step 4), this case is proved.

533 **Case 2.** $x(e) \geq 2$ for some axis e . Since the quad-tree decomposition always divides
 534 boxes equally, the following fact is clear.

535 **Fact.** Let f_1, f_2 be the faces of boxes G_1 and G_2 in U , respectively, such that they are
 536 facing the same direction e . If the distance between f_1 and f_2 in the direction of e is $l > 0$,
 537 one of G_1 and G_2 has edge length $\leq l$.

538 If $x(e) \geq 2$, then there exist faces f_1 and f_2 facing the direction of e and intersect the
 539 interior of G_m . If these 2 faces belongs to the same box $G_f \in U$, then G_f is smaller than
 540 G_m in size. Thus, one of the vertices p_f of G_f must lie in G_m . From Algorithm 3, we know
 541 that $p_f \in Q$. Therefore, $G_m \cap Q \neq \emptyset$. If f_1 and f_2 belong to different boxes, say G_1 and
 542 G_2 in U , from the above fact, we know that the edge length of one of the boxes will be no
 543 larger than the distance between f_1 and f_2 in the direction of e . Since the box size is smaller
 544 than G_m , one of its vertices is in G_m . This again leads to the fact that $G_m \cap Q \neq \emptyset$. This
 545 completes the proof. \blacktriangleleft

546 **Alternative Construction Using ϵ -net Theorem:** Recall that the ϵ -net Theorem allows
 547 us to build a $1/cn$ -net of P with size $O(n \log n)$ by using $O(n \log n)$ samples, where $c > 0$ is
 548 a constant. Also note that, it is not necessary to explicitly compute P (whose size is mn)
 549 before conducting the sampling. From the definition of P (in Section 2, Definition 1), a
 550 random sample from P can be obtained by first sampling a from A , b from B , and then
 551 computing $b - a$ as the sample. This allows us to build a $1/cn$ -net of P in $O(n \log n)$ time.

552 ——— References ———

- 553 1 Helmut Alt and Leonidas J Guibas. Discrete geometric shapes: Matching, interpolation, and
 554 approximation. In *Handbook of computational geometry*, pages 121–153. Elsevier, 2000.
- 555 2 Boris Aronov, Esther Ezra, and Micha Sharir. Small-size ϵ -nets for axis-parallel rectangles
 556 and boxes. *SIAM Journal on Computing*, 39(7):3248–3282, 2010.
- 557 3 Sunil Arya, Theocharis Malamatos, and David M Mount. Space-time tradeoffs for approximate
 558 nearest neighbor searching. *Journal of the ACM (JACM)*, 57(1):1, 2009.
- 559 4 Rinat Ben-Avraham, Matthias Henze, Rafel Jaume, Balázs Keszegh, Orit E Raz, Micha Sharir,
 560 and Igor Tubis. Minimum partial-matching and hausdorff rms-distance under translation:
 561 combinatorics and algorithms. In *European Symposium on Algorithms*, pages 100–111. Springer,
 562 2014.
- 563 5 Sergio Cabello, Panos Giannopoulos, and Christian Knauer. On the parameterized complexity
 564 of d-dimensional point set pattern matching. In *International Workshop on Parameterized
 565 and Exact Computation*, pages 175–183. Springer, 2006.
- 566 6 Paul B Callahan and S Rao Kosaraju. A decomposition of multidimensional point sets
 567 with applications to k-nearest-neighbors and n-body potential fields. *Journal of the ACM*,
 568 42(1):67–90, 1995.
- 569 7 Danny Z Chen, Ziyun Huang, Yangwei Liu, and Jinhui Xu. On clustering induced voronoi
 570 diagrams. *SIAM Journal on Computing*, 46(6):1679–1711, 2017.
- 571 8 Kenneth L Clarkson and Kasturi Varadarajan. Improved approximation algorithms for
 572 geometric set cover. *Discrete & Computational Geometry*, 37(1):43–58, 2007.
- 573 9 Hu Ding, Ronald Berezney, and Jinhui Xu. k-prototype learning for 3d rigid structures. In
 574 *Advances in Neural Information Processing Systems*, pages 2589–2597, 2013.
- 575 10 Hu Ding, Branislav Stojkovic, Ronald Berezney, and Jinhui Xu. Gauging association patterns
 576 of chromosome territories via chromatic median. In *Proceedings of the IEEE Conference on
 577 Computer Vision and Pattern Recognition*, pages 1296–1303, 2013.

29:16 Small Candidate Set for Translational Pattern Search

- 578 **11** Hu Ding and Jinhui Xu. Fpts for minimizing earth mover's distance under rigid transfor-
579 mations. In *European Symposium on Algorithms*, pages 397–408. Springer, 2013.
- 580 **12** Esther Ezra. A note about weak ϵ -nets for axis-parallel boxes in d -space. *Information*
581 *Processing Letters*, 110(18-19):835–840, 2010.
- 582 **13** Martin Gavrilov, Piotr Indyk, Rajeev Motwani, and Suresh Venkatasubramanian. Combin-
583 atorial and experimental methods for approximate point pattern matching. *Algorithmica*,
584 38(1):59–90, 2004.
- 585 **14** Michael T Goodrich, Joseph SB Mitchell, and Mark W Orletsky. Practical methods for approx-
586 imate geometric pattern matching under rigid motions:(preliminary version). In *Proceedings*
587 *of the tenth annual symposium on Computational geometry*, pages 103–112. ACM, 1994.
- 588 **15** Sariel Har-Peled. A replacement for voronoi diagrams of near linear size. In *Proceedings 42nd*
589 *IEEE Symposium on Foundations of Computer Science*, pages 94–103. IEEE, 2001.
- 590 **16** David Haussler and Emo Welzl. ϵ -nets and simplex range queries. *Discrete & Computational*
591 *Geometry*, 2(2):127–151, 1987.
- 592 **17** Matthias Henze, Rafel Jaume, and Balázs Keszegh. On the complexity of the partial least-
593 squares matching voronoi diagram. In *Proc. 29th European Workshop on Computational*
594 *Geometry*, pages 193–196, 2013.
- 595 **18** Daniel P Huttenlocher, Klara Kedem, and Micha Sharir. The upper envelope of voronoi
596 surfaces and its applications. *Discrete & Computational Geometry*, 9(3):267–291, 1993.
- 597 **19** Janardhan Kulkarni and Sathish Govindarajan. New ϵ -net constructions. In *Proceedings of*
598 *the 22nd Annual Canadian Conference on Computational Geometry, Winnipeg, Manitoba,*
599 *Canada*, pages 159–162. Citeseer, 2010.
- 600 **20** Jiří Matoušek, Raimund Seidel, and Emo Welzl. How to net a lot with little: Small ϵ -nets
601 for disks and halfspaces. In *Proceedings of the sixth annual symposium on Computational*
602 *geometry*, pages 16–22. ACM, 1990.
- 603 **21** Günter Rote. Partial least-squares point matching under translations. In *Proc. 26th European*
604 *Workshop on Computational Geometry*, pages 249–251. Citeseer, 2010.
- 605 **22** Nitasha Sehgal, Andrew J Fritz, Jaromira Vecerova, Hu Ding, Zihe Chen, Branislav Sto-
606 jkovic, Sambit Bhattacharya, Jinhui Xu, and Ronald Berezney. Large-scale probabilistic 3d
607 organization of human chromosome territories. *Human molecular genetics*, 25(3):419–436,
608 2015.

A Appendix

A.1 Proof of Corollary 4

Proof. Suppose that translation \mathcal{T} and $B' \subseteq B$ realize the minimum cost bipartite matching with A . Let C_{OPT} be the minimum bipartite matching cost of B' and $\mathcal{T}(A)$. C_{OPT} is then the optimal minimum bipartite matching cost between A and B under translations. Since B' is a C_{OPT} -instance of A , there exists $\mathcal{T}' \in \mathbb{T}$ such that the matching cost between B' and $\mathcal{T}'(A)$ is no larger than $(1 + \epsilon)C_{OPT}$. Thus, \mathcal{T}' induces a $(1 + \epsilon)$ -approximation for the optimal translational matching between A and B . \blacktriangleleft

A.2 Proof of Lemma 5

Proof. Let ϕ_1 (or ϕ_2) be the corresponding bipartite matching which gives rise to the minimum cost between B' and $\mathcal{T}_1(A)$ (or $\mathcal{T}_2(A)$). Then, $\Delta(\mathcal{T}_1(A), B') = \sum_{q \in P(\phi_1)} \|q - \mathcal{T}_1\|$, and $\Delta(\mathcal{T}_2(A), B') = \sum_{q \in P(\phi_2)} \|q - \mathcal{T}_2\|$. Note that

$$\begin{aligned}
 \sum_{q \in P(\phi_1)} \|q - \mathcal{T}_2\| &= \sum_{q \in P(\phi_1)} \|q - \mathcal{T}_1 - \mathcal{T}_2 + \mathcal{T}_1\| \\
 &\leq \sum_{q \in P(\phi_1)} \|\mathcal{T}_2 - \mathcal{T}_1\| + \sum_{q \in P(\phi_1)} \|q - \mathcal{T}_1\| \\
 &= m\|\mathcal{T}_2 - \mathcal{T}_1\| + \Delta(\mathcal{T}_1(A), B') \\
 &\leq m\epsilon\|p - \mathcal{T}_1\| + \Delta(\mathcal{T}_1(A), B') \\
 &\leq \epsilon\Delta(\mathcal{T}_1(A), B') + \Delta(\mathcal{T}_1(A), B') \\
 &= (1 + \epsilon)\Delta(\mathcal{T}_1(A), B'),
 \end{aligned}$$

where the first inequality comes from the triangle inequality, and the third inequality comes from the fact that p is the nearest neighbor of \mathcal{T}_1 in P , which implies that $m\|p - \mathcal{T}_1\| \leq \sum_{q \in P(\phi_1)} \|q - \mathcal{T}_1\| = \Delta(\mathcal{T}_1(A), B')$. From the assumption that ϕ_2 is the minimum cost bipartite matching between $\mathcal{T}_2(A)$ and B' , we know that the value $\sum_{q \in P(\phi_1)} \|q - \mathcal{T}_2\|$, which is the matching cost between $\mathcal{T}_2(A)$ and B' under ϕ_1 , must be no smaller than $\Delta(\mathcal{T}_2(A), B')$. Therefore, from the above inequality, we have $\Delta(\mathcal{T}_2(A), B') \leq (1 + \epsilon)\Delta(\mathcal{T}_1(A), B')$.

Following a similar argument, we also have

$$\begin{aligned}
 \Delta(\mathcal{T}_1(A), B') &\leq \sum_{q \in P(\phi_2)} \|q - \mathcal{T}_1\| \\
 &= \sum_{q \in P(\phi_2)} \|q - \mathcal{T}_2 - \mathcal{T}_1 + \mathcal{T}_2\| \\
 &\leq \sum_{q \in P(\phi_2)} \|\mathcal{T}_2 - \mathcal{T}_1\| + \sum_{q \in P(\phi_2)} \|q - \mathcal{T}_2\| \\
 &= m\|\mathcal{T}_2 - \mathcal{T}_1\| + \Delta(\mathcal{T}_2(A), B') \\
 &\leq m\epsilon\|p - \mathcal{T}_1\| + \Delta(\mathcal{T}_2(A), B') \\
 &\leq \frac{\epsilon}{1 - \epsilon}\Delta(\mathcal{T}_2(A), B') + \Delta(\mathcal{T}_2(A), B') \\
 &= (1 - \epsilon)^{-1}\Delta(\mathcal{T}_2(A), B'),
 \end{aligned}$$

where the fourth inequality comes from the following argument. The closest distance from a point in P to \mathcal{T}_1 is $\|p - \mathcal{T}_1\|$. Since $\|\mathcal{T}_1 - \mathcal{T}_2\| \leq \epsilon\|p - \mathcal{T}_1\|$, we know that the closest

29:18 Small Candidate Set for Translational Pattern Search

⁶⁴⁵ distance from a point in P to \mathcal{T}_2 is at least $(1 - \epsilon)\|p - \mathcal{T}_1\|$. Therefore, $\Delta(\mathcal{T}_2(A), B') =$
⁶⁴⁶ $\sum_{q \in P(\phi_2)} \|q - \mathcal{T}_2\| \geq m(1 - \epsilon)\|p - \mathcal{T}_1\|$.

⁶⁴⁷ Putting everything together, we have that $(1 - \epsilon)\Delta(\mathcal{T}_1(A), B') \leq \Delta(\mathcal{T}_2(A), B') \leq (1 +$
⁶⁴⁸ $\epsilon)\Delta(\mathcal{T}_1(A), B')$. Thus, the lemma follows. \blacktriangleleft