2018 IEEE 59th Annual Symposium on Foundations of Computer Science

Approximating Edit Distance Within Constant Factor in Truly Sub-Quadratic Time

Diptarka Chakraborty
Computer Science Institute of Charles University
Malostranské náměstí 25, 118 00
Praha 1, Czech Republic
Email: diptarka@iuuk.mff.cuni.cz

Debarati Das
Computer Science Institute of Charles University
Malostranské náměstí 25, 118 00
Praha 1, Czech Republic
Email: debaratix710@gmail.com

Elazar Goldenberg
The Academic College Of Tel Aviv-Yaffo
School of Computer Science
Tel Aviv-Yaffo, Israel
Email: elazargo@mta.ac.il

Michal Koucký
Computer Science Institute of Charles University
Malostranské náměstí 25, 118 00
Praha 1, Czech Republic
Email: koucky@iuuk.mff.cuni.cz

Michael Saks
Department of Mathematics
Rutgers University
Piscataway, NJ, USA
Email: msaks30@gmail.com

***Abstract*—Edit distance is a measure of similarity of two strings based on the minimum number of character insertions, deletions, and substitutions required to transform one string into the other. The edit distance can be computed exactly using a dynamic programming algorithm that runs in quadratic time. Andoni, Krauthgamer and Onak (2010) gave a nearly linear time algorithm that approximates edit distance within approximation factor poly$(\log n)$.**

In this paper, we provide an algorithm with running time $\widetilde{O}(n^{2-2/7})$ that approximates the edit distance within a constant factor.

***Keywords*-Edit distance; Approximation algorithm; Sub-quadratic time algorithm; Randomized algorithm;**

I. Introduction

Exact computation of edit distance. The *edit distance* (aka *Levenshtein distance*) [1] between strings x, y, denoted by $d_{\text{edit}}(x, y)$, is the minimum number of character insertions, deletions, and substitutions needed to convert x into y. It is a widely used distance measure between strings that finds applications in fields such as computational biology, pattern recognition, text processing, and information retrieval. The problems of efficiently computing $d_{\text{edit}}(x, y)$, and of constructing an optimal *alignment* (sequence of operations that converts x to y), are of significant interest.

Edit distance can be evaluated exactly in quadratic time via dynamic programming (Wagner and Fischer [2]). Landau *et al.* [3] gave an algorithm that finds an optimal alignment in time $O(n + d_{\text{edit}}(x, y)^2)$, improving on a previous $O(n \cdot d_{\text{edit}}(x, y))$ algorithm of Ukkonen [4]. Masek and Paterson [5] obtained the first (slightly) sub-quadratic $O(n^2/\log n)$ time algorithm, and the current asymptotically fastest algorithm (Grabowski [6]) runs in time $O(n^2 \log\log n/\log^2 n)$.

The research leading to these results has received funding from the European Research Council under the European Union's Seventh Framework Programme (FP/2007-2013)/ERC Grant Agreement no. 616787.

Supported in part by Simons Foundation under award 332622

Backurs and Indyk [7] showed that a *truly sub-quadratic algorithm* ($O(n^{2-\delta})$ for some $\delta > 0$) would imply a $2^{(1-\gamma)n}$ time algorithm for CNF-satisfiabilty, contradicting the Strong Exponential Time Hypothesis (SETH). Abboud et al. [8] showed that even shaving an arbitrarily large polylog factor from n^2 would have the plausible, but apparently hard-to-prove, consequence that NEXP does not have non-uniform NC^1 circuits. For further "barrier" results, see [9], [10].

Approximation algorithms. There is a long line of work on *approximating* edit distance. The exact $O(n+k^2)$ time algorithm (where k is the edit distance of the input) of Landau *et al.* [3] yields a linear time $\sqrt{n}$-factor approximation. This approximation factor was improved, first to $n^{3/7}$ [11], then to $n^{1/3+o(1)}$ [12] and later to $2^{\widetilde{O}(\sqrt{\log n})}$ [13], all with slightly superlinear runtime. Batu *et al.* [14] provided an $O(n^{1-\alpha})$-approximation algorithm with runtime $O(n^{\max\{\frac{\alpha}{2}, 2\alpha-1\}})$. The strongest result of this type is the $(\log n)^{O(1/\epsilon)}$ factor approximation (for every $\epsilon > 0$) with running time $n^{1+\epsilon}$ of Andoni *et al.* [15]. Abboud and Backurs [16] showed that a truly sub-quadratic deterministic time $1 + o(1)$-factor approximation algorithm for edit distance would imply new circuit lower bounds.

Independent of our work, Boroujeni *et al.* [17] obtained a truly sub-quadratic *quantum* algorithm that provides a constant factor approximation. Their latest results [18] are a $(3+\epsilon)$ factor with runtime $\widetilde{O}(n^{2-4/21}/\epsilon^{O(1)})$ and a faster $\widetilde{O}(n^{1.708})$-time with a larger constant factor approximation.

Andoni and Nguyen [19] found a randomized algorithm that approximates Ulam distance of two permutations of $\{1, \ldots, n\}$ (edit distance with only insertions and deletions) within a (large) constant factor in time $\widetilde{O}(\sqrt{n}+n/k)$, where k is the Ulam distance of the input; this was improved by Naumovitz *et al.* [20] to a $(1+\varepsilon)$-factor approximation (for any $\varepsilon > 0$) with similar runtime.

Our results. We present the first truly sub-quadratic time

2575-8454/18/$31.00 ©2018 IEEE
DOI 10.1109/FOCS.2018.00096

classical algorithm that approximates edit distance within a constant factor.

Theorem 1. *There is a randomized algorithm* **ED-UB** *that on input strings x, y of length n over any alphabet Σ outputs an upper bound on $d_{edit}(x, y)$ in time $\widetilde{O}(n^{12/7})$ that, with probability at least $1 - n^{-5}$, is at most a fixed constant multiple of $d_{edit}(x, y)$.*

If the output is U, then the algorithm has implicitly found an alignment of cost at most U. The algorithm can be modified to explicitly output such an alignment.

The approximation factor proved in this preliminary version is 1680, can be greatly improved by tweaking parameters. We believe, but have not proved, that with sufficient care the algorithm can be modified (with no significant increase in runtime) to get $(3 + \epsilon)$ approximation.

Theorem 1 follows from:

Theorem 2. *For every $\theta \in [n^{-1/5}, 1]$, there is a randomized algorithm* **GAP-UB**$_\theta$ *that on input strings x, y of length n outputs $u =$* **GAP-UB**$_\theta(x, y)$ *such that: (1) $d_{edit}(x, y) \le u$ and (2) on any input with $d_{edit}(x, y) \le \theta n$, $u \le 840\theta n$ with probability at least $1 - n^{-7}$. The runtime of* **GAP-UB**$_\theta$ *is $\tilde{O}(n^{2-2/7}\theta^{4/7})$.*

The name **GAP-UB**$_\theta$ reflects that this is a "gap algorithm", which distinguishes inputs with $d_{\text{edit}}(x, y) \le \theta n$ (where the output is at most $840\theta n$), and those with $d_{\text{edit}}(x, y) > 840\theta n$ (where the output is greater than $840\theta n$).

Theorem 1 follows via a routine construction of **ED-UB** from **GAP-UB**$_\theta$, presented in Section V. The rest of the paper is devoted to proving Theorem 2.

The framework of the algorithm. We use a standard two-dimensional representation of edit distance. Visualize x as lying on a horizontal axis and y as lying on a vertical axis, with horizontal coordinate $i \in \{1, \ldots, n\}$ corresponding to x_i and vertical component j corresponding to y_j. The width $\mu(I)$ of interval $I \subseteq \{0, 1, \ldots, n\}$ is $\max(I) - \min(I) = |I| - 1$. Also, x_I denotes the substring of x indexed by $I - \{\min(I)\}$. (Note: $x_{\min(I)}$ is not part of x_I, e.g., $x = x_{\{0,\ldots,n\}}$. This convention is motivated by Proposition 3.) We refer to I as an *x-interval* to indicate that it indexes a substring of x, and J as a *y-interval* to indicate that it indexes a substring of y. A *box* is a set $I \times J$ where I is a x-interval and J is a y-interval; $I \times J$ corresponds to the substring pair (x_I, y_J). $I \times J$ is a w-box if $\mu(I) = \mu(J) = w$. We often abbreviate $d_{\text{edit}}(x_I, y_J)$ by $d_{\text{edit}}(I, J)$. A *decomposition* of an x-interval I is a sequence $I_1, \ldots, I_\ell$ of subintervals with $\min(I_1) = \min(I)$, $\max(I_\ell) = \max(I)$ and for $j \in [\ell - 1]$, $\max(I_j) = \min(I_{j+1})$.

Associated to x, y is a directed graph $G_{x,y}$ with edge costs called a *grid graph* with vertex set $\{0, \ldots, n\} \times \{0, \ldots, n\}$ and all edges of the form $(i - 1, j) \to (i, j)$ (H-steps), $(i, j - 1) \to (i, j)$ (V-steps) and $(i - 1, j - 1) \to (i, j)$ (D-steps). Every H-step or V-step costs 1, and D-steps cost 1 if $x_i \ne y_j$ and 0 otherwise. There is a 1-1 correspondence that maps a path from $(0, 0)$ to (n, n) to an *alignment* from x to y, i.e. a set of character deletions, insertions and substitutions that changes x to y, where an H-step $(i - 1, j) \to (i, j)$ means "delete x_i", a V-step $(i, j - 1) \to (i, j)$ means "insert y_j between x_i and x_{i+1}" and a D-step $(i-1, j-1) \to (i, j)$ means replace x_i by y_j, unless they are already equal. We have:

Proposition 3. *The cost of an alignment, cost(τ), is the sum of edge costs of its associated path τ, and $d_{edit}(x, y)$ is equal to cost($G_{x,y}$), the min cost of an alignment path from $(0, 0)$ to (n, n).*

For $I, J \subseteq \{0, \ldots, n\}$, $G_{x,y}(I \times J) \cong G_{x_I, y_J}$ is the grid graph induced on $I \times J$, and $d_{\text{edit}}(I, J) = \text{cost}(G_{x,y}(I \times J))$.

The natural high-level idea of **GAP-UB**$_\theta$ appears (explicitly or implicitly) in previous work. The algorithm has two phases. First, the *covering phase* identifies a set $\mathcal{R}$ of *certified boxes* which are pairs $(I \times J, \kappa)$, where κ is an upper bound on the *normalized edit distance* $\Delta_{\text{edit}}(x_I, y_J) = d_{\text{edit}}(x_I, y_J)/\mu(I)$. ($\Delta_{\text{edit}}(I, J)$ is more convenient than $d_{\text{edit}}(I, J)$ for the covering phase.) Second, the *min-cost path phase*, takes input $\mathcal{R}$ and uses a straightforward customized variant of dynamic programming to find an upper bound $U(\mathcal{R})$ on $d_{\text{edit}}(x, y)$ in time quasilinear in $|\mathcal{R}|$. The central issue is to ensure that the covering phase outputs $\mathcal{R}$ that is sufficiently informative so that $U(\mathcal{R}) \le c \cdot d_{\text{edit}}(x, y)$ for constant c, while running in sub-quadratic time.

Simplifying assumptions. The input strings x, y have equal length n. (It is easy to reduce to this case: pad the shorter string to the length of the longer using a new symbol. The edit distance of the new pair is between the original edit distance and twice the original edit distance. This factor 2 increase in approximation factor can be avoided by generalizing our algorithm to the case $|x| \ne |y|$, but we won't do this here.) We assume n is a power of 2 (by padding both strings with a new symbol, which leaves edit distance unchanged). We assume that θ is a (negative) integral power of 2. The algorithm involves integer parameters w_1, w_2, d, all of which are chosen to be powers of 2.

Organization of the paper. Section II is a detailed overview of the covering phase algorithm and its analysis. Section III presents the pseudo-code and analysis for the covering phase. Section IV presents the min-cost path phase algorithm. Section V summarizes the full algorithm and discusses improvements in runtime via recursion.

II. Covering algorithm: Detailed overview

We give a detailed overview of the covering phase and its time analysis and proof of correctness, ignoring minor technical details. The pseudo-code in Section III corresponds to the overview, with technical differences mainly to improve runtime. We will illustrate the sub-quadratic time analysis

with the sample input parameter $\theta = n^{-1/50}$ and algorithm parameters $w_1 = n^{1/10}$, $w_2 = n^{3/10}$ and $d = n^{1/5}$.

The covering phase outputs a set $\mathcal{R}$ of certified boxes. The goal is that $\mathcal{R}$ includes an *adequate approximating sequence* for some min-cost path τ in $G_{x,y}$, which is a sequence σ of certified boxes $(I_1 \times J_1, \kappa_1), \ldots, (I_\ell \times J_\ell, \kappa_\ell)$ that satisfies:

1) $I_1, \ldots, I_\ell$ is a decomposition of $\{0, \ldots, n\}$.
2) $I_i \times J_i$ is an *adequate cover of* τ_i, where $\tau_i = \tau_{I_i}$ denotes the minimal subpath of τ whose projection to the x-axis is I_i, and adequate cover means that the (vertical) distance from the start vertex (resp. final vertex) of τ_i and the lower left (resp. upper right) corner of $I_i \times J_i$, is at most a constant multiple of $\text{cost}(\tau_i) + \theta$.
3) The sequence σ is *adequately bounded*, i.e., $\sum_i \mu(I_i)\kappa_i \leq c(\text{cost}(\tau) + \theta n)$, for a constant c.

This is a slight oversimplification of Definition 3 of (k, ζ)-*approximation of* τ by a sequence of certified boxes.

The intuition for the second condition is that τ_i is "almost" a path between the lower left and upper right corners of $I_i \times J_i$. Now τ_i might have a vertical extent J' that is much larger than its horizontal extent I_i, in which case it is impossible to place a square $I_i \times J_i$ with corners close to both endpoints of τ_i. But in that case, τ_i has a very high cost (at least $|\mu(J') - \mu(I_i)|$. The closeness required is adjusted based on $\text{cost}(\tau_i)$, with relaxed requirements if $\text{cost}(\tau_i)$ is large.

The output of the min-cost path phase should satisfy the requirements of **GAP-UB**$_\theta$. Lemma 17 shows that if the min-cost path phase receives $\mathcal{R}$ that contains a (k, θ)-approximating sequence to some min-cost path τ, then it will output an upper bound to $d_{\text{edit}}(x, y)$ that is at most $k'(d_{\text{edit}}(x, y) + \theta n)$ for some k'. So that on input x, y with $d_{\text{edit}}(x, y) \leq \theta n$, the output is at most $2k'\theta n$, satisfying the requirements of **GAP-UB**$_\theta$. This formalizes the intuition that an adequate approximating sequence captures enough information to deduce a good bound on $\text{cost}(\tau)$.

Once and for all, we fix a min-cost path τ. Our task for the covering phase is that, with high probability, $\mathcal{R}$ includes an adequate approximating sequence for τ.

A τ-*match* for an x-interval I is a y-interval J such that $I \times J$ is an adequate cover of τ_I. It is easy to show (Proposition 7) that this implies $d_{\text{edit}}(I, J) \leq (\text{cost}(\tau_I) + \theta\mu(I))$. A box $I \times J$ is said to be τ-*compatible* if J is a τ-match for I and a box sequence is τ-*compatible* if every box is τ-compatible. A τ-compatible certified box sequence whose distance upper bounds are (on average) within a constant factor of the actual cost, satisfies the requirements for an adequate approximating sequence. Our cover algorithm will ensure that $\mathcal{R}$ contains such a sequence.

A natural decomposition is $\mathcal{I}_{w_1}$, with all parts of width w_1 (think of w_1 as a power of 2 that is roughly $n^{1/10}$) so $\ell = n/w_1$ and $I_j = \{(j-1)w_1, \cdots, (j)w_1\}$. The naïve approach to building $\mathcal{R}$ is to include certified boxes for enough choices of J to guarantee a τ-match for each I_j. An interval of width w_1 is δ-*aligned* if its upper and lower endpoints are both multiples of δw_1 (which we require to be an integral power of 2). We restrict attention to x-intervals in $\mathcal{I}_{w_1}$, called x-*candidates* and θ-aligned y-intervals of width w_1 called y-*candidates*. It can be shown (see Proposition 8) that an x-interval I always has a τ-match J that is θ-aligned. (In this overview we will fix δ to θ; the actual algorithm has $O(\log n)$ iterations during which the value of δ varies, giving improvements in runtime that are unimportant in this overview.) For each x-candidate I, designate one such τ-match as the *canonical* τ-*match*, $J^\tau(I)$ for I, and $I \times J^\tau(I)$ is the *canonical* τ-*compatible box for* I.

In the *exhaustive approach*, for each (x-candidate, y-candidate)-pair (I, J), its edit distance is computed in time $O(w_1^2)$, and the certified box $(I \times J, \Delta_{\text{edit}}(I, J))$ is included. There are $\frac{n}{w_1}\frac{n}{\theta w_1}$ boxes, so the time for all edit distance computations is $O(\frac{n^2}{\theta})$, which is worse than quadratic. (The factor $\frac{1}{\theta}$ can be avoided by standard techniques, but this is not significant to the quest for a sub-quadratic algorithm, so we defer this until the next section.) Note that $|\mathcal{R}|$ is $\frac{n^2}{\theta(w_1)^2}$ (which is $n^{1.82}$ for our sample parameters) so at least the min-cost path phase (which runs in time quasi-linear in $\mathcal{R}$) is truly sub-quadratic.

Two natural goals that will improve the runtime are: (1) Reduce the amortized time per box needed to certify boxes significantly below $(w_1)^2$ and (2) Reduce the total number of certified boxes created significantly below $\frac{n^2}{\theta(w_1)^2}$. Neither goal is always achievable, and our covering algorithm combines them. In independent work [17], [18], versions of these two goals are combined, where the second goal is accomplished via Grover search, thus yielding a constant factor sub-quadratic time quantum approximation algorithm.

Reducing amortized time for certifying boxes: the *dense case* algorithm. We aim to reduce the amortized time per certified box to be much smaller than $(w_1)^2$. We divide our search for certified boxes into iterations $i \in \{0, \ldots, \log n\}$. For iteration i, with $\epsilon_i = 2^{-i}$, our goal is that for all candidate pairs I, J with $\Delta_{\text{edit}}(I, J) \leq \epsilon_i$, we include the certified box $(I \times J, c\epsilon_i)$ for a fixed constant c. If we succeed, then for each I_j and its canonical τ-match $J^\tau(I_j)$, and for the largest index i for which $\Delta_{\text{edit}}(I_j, J^\tau(I_j)) \leq \epsilon_i$, iteration i will certify $(I_j \times J^\tau(I_j), \kappa_j)$ with $\kappa_j \leq c\epsilon_i \leq 2c\Delta_{\text{edit}}(I_j, J^\tau(I_j))$, as needed.

For a string z of size w_1, let $\mathcal{H}(z, \rho)$ be the set of x-candidates I with $\Delta_{\text{edit}}(z, x_I) \leq \rho$ and $\mathcal{V}(z, \rho)$ be the set of y-candidates J with $\Delta_{\text{edit}}(z, y_J) \leq \rho$. In iteration i, for each x-candidate I, we will specify a set $\mathcal{Q}_i(I)$ of y-candidates that includes $\mathcal{V}(x_I, \epsilon_i)$ and is contained in $\mathcal{V}(x_I, 5\epsilon_i)$. The set of certified boxes $(I \times J, 5\epsilon_i)$ for all x-candidates I and $J \in \mathcal{Q}_i(I)$ satisfies the goal of iteration i.

Iteration i proceeds in rounds. In each round we select an x-candidate I, called the *pivot*, for which $\mathcal{Q}_i(I)$ has not yet been specified. Compute $\Delta_{\text{edit}}(x_I, y_J)$ for all y-candidates

J and $\Delta_{\text{edit}}(x_I, x_{I'})$ for all x-candidates I'; these determine $\mathcal{H}(x_I, \rho)$ and $\mathcal{V}(x_I, \rho)$ for any ρ. For all $I' \in \mathcal{H}(x_I, 2\epsilon_i)$, set $\mathcal{Q}_i(I') = \mathcal{V}(x_I, 3\epsilon_i)$. By the triangle inequality, for each $I' \in \mathcal{H}(x_I, 2\epsilon_i)$, $\mathcal{V}(x_I, 3\epsilon_i)$ includes $\mathcal{V}(x_{I'}, \epsilon_i)$ and is contained in $\mathcal{V}(x_{I'}, 5\epsilon_i)$ so we can certify all the boxes with upper bound $5\epsilon_i$. Mark intervals in $\mathcal{H}(x_I, 2\epsilon_i)$ as *fulfilled* and proceed to the next round, choosing a new pivot from among the unfulfilled x-candidates.

The number of certified boxes produced in a round is $|\mathcal{H}(x_I, 2\epsilon_i)| \times |\mathcal{V}(x_I, 3\epsilon_i)|$. If this is much larger than $O(\frac{n}{\theta w_1})$, the number of edit distance computations, then we have significantly reduced amortized time per certified box. (For example, in the trivial case $i = 0$, every candidate box will be certified in a single round.) But in worst case, there are $\frac{n}{w_1}$ rounds each requiring $\Omega(\frac{nw_1}{\theta})$ time, for an unacceptable total time $\Theta(n^2/\theta)$.

Here is a situation where the number of rounds is much less than $\frac{n}{w_1}$. Since any two pivots are necessarily greater than $2\epsilon_i$ apart, the sets $\mathcal{V}(x_I, \epsilon_i)$ for distinct pivots are disjoint. Now for some parameter d (think of $d = n^{1/5}$) an x-candidate is *d-dense* for ϵ_i if $|\mathcal{V}(x_I, \epsilon_i)| \geq d$, i.e., x_I is ϵ_i-close in edit distance to at least d y-candidates; it is d-sparse otherwise. If we manage to select a d-dense pivot I in each round, then the number of rounds is $O(\frac{n}{w_1 d\theta})$ and the overall time will be $\Theta(\frac{n^2}{d\theta^2})$. For the sample parameters this is $\Theta(n^{1.84})$. But there's no reason to expect that we'll only choose dense pivots; indeed there need not be any dense pivot.

Let's modify the process a bit. When choosing potential pivot I, first test whether or not it is (approximately) d-dense. This can be done with high probability, by randomly sampling $\tilde{\Theta}(\frac{n}{\theta w_1 d})$ y-candidates and finding the fraction of the sample that are within ϵ_i of x_I. If this fraction is less than $\frac{\theta w_1 d}{2n}$ then I is *declared sparse* and abandoned as a pivot; otherwise I is *declared dense*, and used as a pivot. With high probability, all d-dense intervals that are tested are declared dense, and all tested intervals that are not $d/4$-dense are declared sparse, so we assume this is the case. Then all pivots are processed (as above) in time $O(\frac{n^2}{d\theta^2})$ (under sample parameters: $O(n^{1.84})$). We pay $\tilde{O}(\frac{n}{w_1 d\theta})(w_1)^2$ to test each potential pivot (at most $\frac{n}{w_1}$ of them) so the overall time to test potential pivots is $\tilde{O}(\frac{n^2}{d\theta})$ (with sample parameters: $\tilde{O}(n^{1.82})$).

Each iteration i (with different ϵ_i) splits x-candidates into two sets, $\mathcal{S}_i$ of intervals that are declared sparse, and all of the rest for which we have found the desired set $\mathcal{Q}_i(I)$. With high probability every interval in $\mathcal{S}_i$ is indeed d-sparse, but a sparse interval need not belong to $\mathcal{S}_i$, since it may belong to $\mathcal{H}(x_I, 2\epsilon_i)$ for some selected pivot I.

For every x-candidate $I \notin \mathcal{S}_i$ we have met the goal for the iteration. If $\mathcal{S}_i$ is very small for all iterations, then the set of certified boxes will suffice for the min-cost path algorithm to output a good approximation.

But if $\mathcal{S}_i$ is not small, another approach is needed.

Reducing the number of candidates explored: the *diagonal extension algorithm*. For each x-candidate I, although it suffices to certify the single box $(I, J^\tau(I))$ with a good upper bound, since τ is unknown, the exhaustive and dense case approaches both include certified boxes for all y-candidates J. The potential savings in the dense case approach comes from certifying many boxes simultaneously using a relatively small number of edit distance computations.

Here's another approach: for each x-candidate I try to quickly identify a relatively small subset $\mathcal{Y}(I)$ of y-candidates that is guaranteed to include $J^\tau(I)$. If we succeed, then the number of boxes we certify is significantly reduced, and even paying quadratic time per certified box, we will have a sub-quadratic algorithm.

We need the notion of *diagonal extension* of a box. The *main diagonal* of box $I \times J$, is the segment joining the lower left and upper right corners. The square box $I' \times J'$ is a *diagonal extension* of a square subbox $I \times J$ if the main diagonal of $I \times J$ is a subsegment of the main diagonal of $I' \times J'$. (see Definition 2.) Given square box $I \times J$ and $I' \subset I$ the *diagonal extension of $I \times J$ with respect to I'* is the unique diagonal extension of $I \times J$ having x-interval I'. The key observation (Proposition 9) is: if $I \times J$ is an adequate cover of τ_I then any diagonal extension $I' \times J'$ is an adequate cover of $\tau_{I'}$.

Now let w_1, w_2 be two numbers with $w_1 | w_2$ and $w_2 | n$. (Think of $w_1 = n^{1/10}$ and $w_2 = n^{3/10}$.) We use the decomposition $\mathcal{I}_{w_2}$ of $\{0, \ldots, n\}$ into intervals of width w_2. The set of y-candidates consists θ-aligned vertical intervals of width w_2 and has size $\frac{n}{\theta w_2}$. To identify a small set of potential matches for $I' \in \mathcal{I}_{w_2}$, we will identify a set (of size much smaller than $\frac{n}{w_2}$) of w_1-boxes $\mathcal{B}(I')$ having x-interval in $\mathcal{I}_{w_1}(I')$ (the decomposition of I' into width w_1 intervals). For each box in $\mathcal{B}(I')$ we determine the diagonal extension $I' \times J'$ with respect to I', compute $\kappa = \Delta_{\text{edit}}(I', J')$ and certify $(I' \times J', \kappa)$. Our hope is that $\mathcal{B}(I')$ includes a τ-compatible w_1-box $I'' \times J^\tau(I'')$, then the observation above implies that its diagonal extension provides an adequate cover for $\tau_{I'}$.

Here's how to build $\mathcal{B}(I')$: Randomly select a polylog(n) size set $\mathcal{H}(I')$ of w_1-intervals from $\mathcal{I}_{w_1}(I')$. For each $I'' \in \mathcal{H}(I')$ compute $\Delta_{\text{edit}}(I'', J'')$ for each y-candidate J'', and let $\mathcal{J}(I'')$ consist of the d candidates J'' with smallest edit distance to I''. Here d is a parameter; think of $d = n^{1/5}$ as before. $\mathcal{B}(I')$ consists of all $I'' \times J''$ where $I'' \in \mathcal{H}(I')$ and $J'' \in \mathcal{J}(I'')$.

To bound runtime: Each $I' \in \mathcal{I}_{w_2}$ requires $\tilde{O}(\frac{n}{\theta w_1})$ width-w_1 $\Delta_{\text{edit}}()$ computations, taking time $\tilde{O}(\frac{nw_1}{\theta})$. Diagonal extension step requires $\tilde{O}(d)$ width-w_2 $\Delta_{\text{edit}}()$ computations, for time $\tilde{O}(dw_2^2)$. Summing over $\frac{n}{w_2}$ choices for I' gives time $\tilde{O}(n^2 \frac{w_1}{\theta w_2} + ndw_2)$ (with sample parameters: $\tilde{O}(n^{1.82})$).

Why should $\mathcal{B}(I')$ include a box that is an adequate approximation to $\tau_{I'}$? The intuition behind the choice of $\mathcal{B}(I')$ is that an adequate cover for $\tau_{I'}$ should typically be among the cheapest boxes of the form $I' \times J'$, and if $I' \times J'$ is cheap then for a randomly chosen w_1-subinterval I'', we should also have $I'' \times J^\tau(I'')$ is among the cheapest boxes for I''.

Clearly this intuition is faulty: I' may have many inexpensive matches J' such that $I' \times J'$ is far from $\tau_{I'}$, which may all be much cheaper than the match we are looking for. In this bad situation, there are many y-intervals J' such that $\Delta_{\text{edit}}(I', J')$ is smaller than the match we are looking, and this is reminiscent of the *good* situation for the dense case algorithm, where we hope that I' has lots of close matches. This suggests combining the two approaches, and leads to our full covering algorithm.

The full covering algorithm. This is now easy to describe. The parameters w_1, w_2, d are as above. We iterate over $i \in \{0, \ldots, \log n\}$ with $\epsilon_i = 2^{-i}$. In iteration i, we first run the dense case algorithm, and let $\mathcal{S}_i$ be the set of intervals declared sparse. Then run the diagonal extension algorithm described earlier (with small modifications): For each w_2-interval I', select $\mathcal{H}(I') = \mathcal{H}_i(I')$ to consist of $\theta(\log^2 n)$ independent random selections from $\mathcal{S}_i$. For each $I'' \in \mathcal{H}_i(I')$, find the set of vertical candidates J'' for which $\Delta_{\text{edit}}(I'', J'') \leq \epsilon_i$. Since I'' is (almost certainly) d-sparse, the number of such J'' is at most d. Proceeding as in the diagonal extension algorithm, we produce a set $\mathcal{P}_i(I')$ of $\tilde{O}(d)$ certified w_2-boxes with x-interval I'. Let $\mathcal{R}_D$ (resp. $\mathcal{R}_E$) be the set of all certified boxes produced by the dense case iterations, resp. diagonal extension iterations. The output is $\mathcal{R} = \mathcal{R}_D \cup \mathcal{R}_E$. (See Figure 1 for an illustration of the output $\mathcal{R}$.)

The runtime is the sum of the runtimes of the dense case and diagonal extension algorithms, as analyzed above. Later, we will give a more precise runtime analysis for the pseudocode.

To finish this extended overview, we sketch the argument that $\mathcal{R}$ satisfies the covering phase requirements.

Claim 4. *Let I' be an interval in the w_2-decomposition. Either (1) the output of the dense case algorithm includes a sequence of certified w_1-boxes that adequately approximates the subpath $\tau_{I'}$, or (2) with high probability the output of the sparse case algorithm includes a single w_2-box that adequately approximates $\tau_{I'}$.*

(This claim is formalized in Claim 14.) Stitching together the subpaths for all I' implies that $\mathcal{R}$ will contain a certified box sequence that adequately approximates τ.

To prove the claim, we establish a sufficient condition for each of the two conclusion and show that if the sufficient condition for the second conclusion fails, then the sufficient condition for the first holds.

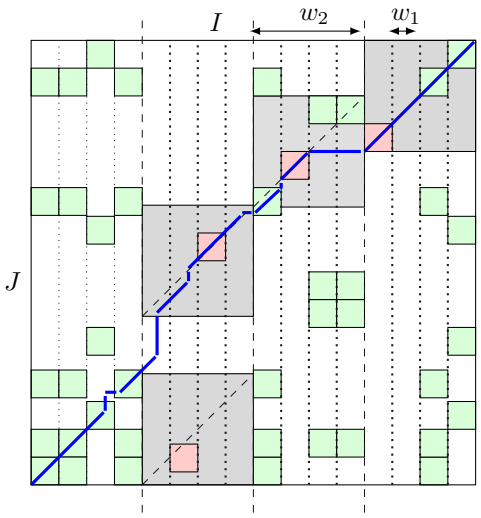

Figure 1. Illustration of the Covering Algorithm: Green boxes are low cost boxes in dense w_1-strips, while the pink ones are in sparse w_1-strips. The blue line corresponds to the path τ that we are trying to cover. In each w_2-strip, τ is covered by either a collection of many w_1-boxes or it is covered by a diagonal extension of a low cost w_1-box. The various boxes might overlap vertically which is not shown in the picture.

Let $\mathcal{I}'$ denote the w_1-decomposition $\mathcal{I}_{w_1}(I')$ of I'. Every interval $I'' \in \mathcal{I}'$ has a θ-aligned τ-match $J^\tau(I'')$. It will be shown (see Proposition 8), that $\Delta_{\text{edit}}(I'', J^\tau(I'')) \leq 2\frac{\text{cost}(\tau_{I''})}{\mu(I'')} + \theta$. Let $u(I'')$ denote this upper bound. Consider the first alternative in the claim. During the dense case iteration $i = 0$, every interval is declared dense, and $(I'' \times J^\tau(I''), 5)$ is in $\mathcal{R}_D$ for all I''. To get an adequate approximation, we try to show that later iterations provide much better upper bounds on these boxes, i.e., $(I'' \times J^\tau(I''), \gamma(I'')) \in \mathcal{R}_D$ for a small enough value of $\gamma(I'')$. By definition of adequate approximation, it is enough that $\sum_{I'' \in \mathcal{I}'} \gamma(I'') \leq c \sum_{I'' \in \mathcal{I}'} u(I'')$, for some c. Let $t(I'')$ be the last (largest) iteration for which $\epsilon_{t(I'')} \geq u(I'')$ and $I'' \notin \mathcal{S}_{t(I'')}$ (which is well defined since $\mathcal{S}_0 = \emptyset$). Let $b(I'') = \epsilon_{t(I'')}$. Since $b(I'') \geq u(I'') \geq \Delta_{\text{edit}}(I'', J^\tau(I''))$, the box $(I'' \times J^\tau(I''), 5b(I''))$ is certified. The collection $\{(I'' \times J^\tau(I''), 5b(I''))\}$ is a sequence of certified boxes that satisfies the first two conditions for an adequate approximation of τ. The third condition will follow if:

$$\sum_{I'' \in \mathcal{I}'} 5b(I'') \leq c \sum_{I'' \in \mathcal{I}'} u(I'') \tag{1}$$

so this is sufficient to imply the first condition of the claim.

Next consider what we need for the second alternative to hold. Let $\mathcal{S}_i(I')$ be the set of intervals declared sparse in iteration i. An interval $I'' \in \mathcal{S}_i(I')$ is a *winner* (for iteration i) if $\Delta_{\text{edit}}(I'', J^\tau(I'')) \leq \epsilon_i$, and $\mathcal{W}_i(I')$ is the set of winners. In iteration i of the diagonal extension

algorithm, we sample $\theta(\log^2 n)$ elements of $\mathcal{S}_i(I')$. If for at least one iteration i our sample includes a winner I'' then the second condition of the claim will hold: $I'' \times J^\tau(I'')$ is extended diagonally to a w_2-box, and by the diagonal extension property, the extension is an adequate cover of $\tau_{I'}$, which we will certify with its exact edit distance.

Thus for the second alternative to fail with nonnegligible probability:

$$\text{For all } i, |\mathcal{W}_i(I')| < |\mathcal{S}_i(I') - \mathcal{W}_i(I')|, \tag{2}$$

We argue that if (2) holds, then the success condition (1) holds. Multiply (2) by ϵ_i and sum on i to get:

$$\sum_{I'' \in \mathcal{I}'} \sum_{i: I'' \in \mathcal{W}_i(I')} \epsilon_i < \sum_{I'' \in \mathcal{I}'} \sum_{i: I'' \in \mathcal{S}_i(I') - \mathcal{W}_i(I')} \epsilon_i. \tag{3}$$

For $I'' \in \mathcal{I}_{w_1}(I')$, consider the iterations i for which $I'' \in \mathcal{W}_i(I')$ and those for which $I'' \in \mathcal{S}_i(I') - \mathcal{W}_i(I')$. First of all if $\epsilon_i \geq u(I'')$ and $I'' \in \mathcal{S}_i(I')$ then since $\Delta_{\text{edit}}(I'', J^\tau(I'')) \leq u(I'') \leq \epsilon_i$ we conclude $I'' \in \mathcal{W}_i(I')$. So $I'' \in \mathcal{S}_i(I') - \mathcal{W}_i(I')$ implies that $\epsilon_i < u(I'')$, so the inner sum of the right side of (3) is at most $2u(I'')$ (by summing a geometric series).

Furthermore, for i with $u(I'') \leq \epsilon_i < b(I'')$, $I'' \in \mathcal{S}_i$ by the choice of $t(I'')$. Either $b(I'')/2 \leq u(I'')$ or $u(I'') < b(I'')/2$. The latter implies $I'' \in \mathcal{W}_{t(I'')+1}(I')$, and then $b(I'')/2$ is upper bounded by the inner sum on the left of (3). Therefore:

$$\begin{aligned}
\sum_{I''} b(I'') &\leq \sum_{I''} \left(2u(I'') + \sum_{i: I'' \in \mathcal{W}_i(I')} 2\epsilon_i \right) \\
&< \sum_{I''} \left(2u(I'') + 2 \sum_{i: I'' \in \mathcal{S}_i(I') - \mathcal{W}_i(I')} \epsilon_i \right) \\
&\leq 6 \sum_{I''} u(I''),
\end{aligned}$$

as required for (1).

This completes the overview of the covering algorithm.

III. Covering Algorithm: pseudo-code and analysis

The pseudo-code consists of *CoveringAlgorithm* which calls procedures *DenseStripRemoval* (the dense case algorithm) and *SparseStripExtensionSampling* (the diagonal extension algorithm). These are abbreviated, respectively by CA, DSR and SSES. The technical differences between the pseudo-code and the informal description, are mainly to improve runtime analysis.

A. Pseudo-code

The parameters of CA are as described in the overview: x, y are input strings of length n, θ comes from **GAP-UB**$_\theta$, $w_1 < w_2 < n$ and $d < n$ are integral powers of 2, as are the auxiliary input parameters. The output is a set $\mathcal{R}$ of certified boxes. The algorithm uses global constants $c_0 \geq 0$ and $c_1 \geq 120$, where the former one is needed for Proposition 11.

We use a subroutine **SMALL-ED** which takes strings z_1, z_2 of length w and parameter κ and outputs ∞ if $\Delta_{\text{edit}}(z_1, z_2) > \kappa$ and otherwise outputs $\Delta_{\text{edit}}(z_1, z_2)$. The algorithm of [4] implements **SMALL-ED** in time $O(\kappa w^2)$.

One technical difference from the overview, is that the pseudo-code saves time by restricting the search for certified boxes to a portion of the grid close to the main diagonal. Recall that **GAP-UB**$_\theta$ has two requirements, that the output upper bounds $d_{\text{edit}}(x, y)$ (which will be guaranteed by the requirement that $\mathcal{R}$ contains no falsely certified boxes), and that if $d_{\text{edit}}(x, y) \leq \theta n$, the output is at most $c\theta n$ for some constant c. We therefore design our algorithm assuming $d_{\text{edit}}(x, y) \leq \theta n$, in which case every min-cost $G_{x,y}$-path τ consists entirely of points within $\frac{\theta}{2}n$ steps from the main diagonal, i.e. $|i - j| \leq \frac{\theta}{2}n$. So we restrict our search for certified boxes as follows: set $m = \frac{1}{4}\theta n$, and consider the $\frac{n}{m}$ overlapping equally spaced boxes of width $8m = 2\theta n$ lying along the main diagonal. Together these boxes cover all points within θn of the main diagonal.

The algorithm of the overview is executed separately on each of these n/m boxes. Within each of these executions, we iterate over $i \in \{0, \ldots, \log \frac{1}{\theta}\}$ (rather than $\{0, \ldots, \log n\}$ as in the overview). In each iteration we apply the dense case algorithm and the diagonal extension algorithm as in the overview. The output is the union over all n/m boxes and all iterations, of the boxes produced.

In the procedures DSR and SSES, the input G is an induced grid graph corresponding to a box $I_G \times J_G$, as described in the "framework" part of Section I. The procedure DSR on input G, sets $\mathcal{T}$ to be the w_1-decomposition of I_G (the x-candidates) and $\mathcal{B}$ to be the set of $\frac{\epsilon_i}{8}$-aligned y-candidates. As in the overview, the dense case algorithm produces a set of certified boxes (called $\mathcal{R}_1$ in the pseudo-code) and a set $\mathcal{S}$ of intervals declared sparse. SSES is invoked if $\mathcal{S} \neq \emptyset$ and iterates over all x-intervals I' in the decomposition $\mathcal{I}_{w_2}(I_G)$. The algorithm skips I' if $\mathcal{S}$ contains no subset of I', and otherwise selects a sample $\mathcal{H}$ of $\theta(\log^2 n)$ subintervals of I' from $\mathcal{S}$. For each sample interval I'' it finds the vertical candidates J'' for which $\Delta_{\text{edit}}(I'', J'') \leq \epsilon_i$, does a diagonal extension to I' and certifies each box with an exact edit distance computation.

There are a few parameter changes from the overview that provide some improvement in the time analysis: During each iteration i, rather than take our vertical candidates to be from a θ-aligned grid, we can afford a coarser grid that is $\epsilon_i/8$-aligned. Also, the local parameter d in DSR and SSES

is set to d/ϵ_i during iteration i.

There is one counterintuitive quirk in SSES: each certified box is replicated $O(\log n)$ times with higher distance bounds. This is permissible (increasing the distance bound cannot decertify a box), but seems silly (why add the same box with a higher distance bound?). This is just a convenient technical device to ensure that the second phase min-cost path algorithm gives a good approximation.

Algorithm 1 CA$(x, y, n, w_1, w_2, d, \theta)$
CoveringAlgorithm

Input: Strings x, y of length n, $w_1, w_2, d \in [n]$, $w_1 < w_2 < \theta n/4$, and $\theta \in [0, 1]$. n, w_1, w_2, θ are powers of 2.
Output: A set $\mathcal{R}$ of certified boxes in G.

Initialization: $G = G_{x,y}$, $\mathcal{R}_D = \mathcal{R}_E = \emptyset$.
Let $m = \frac{\theta n}{4}$
for $k = 0, \ldots, \frac{4}{\theta}$ **do**
  Let $I = J = \{km, km + 1, \ldots, (k + 8)m\}$.
  for $i = \lceil \log 1/\theta \rceil, \ldots, 0$ **do**
    Set $\epsilon_i = 2^{-i}$.
    Invoke DSR$(G(I \times J), n, w_1, \frac{d}{\epsilon_i}, \frac{\epsilon_i}{8}, \epsilon_i)$ to get $\mathcal{S}$ and $\mathcal{R}_1$.
    if $\mathcal{S} \neq \emptyset$ **then**
      Invoke SSES$(G(I \times J), \mathcal{S}, n, w_1, w_2, \frac{d}{\epsilon_i}, \frac{\epsilon_i}{8}, \epsilon_i, \theta)$ to get $\mathcal{R}_2$.
    else
      $\mathcal{R}_2 = \emptyset$.
    end if
    Add items from $\mathcal{R}_1$ to $\mathcal{R}_D$ and from $\mathcal{R}_2$ to $\mathcal{R}_E$.
  end for
end for
Output $\mathcal{R} = \mathcal{R}_D \cup \mathcal{R}_E$.

For the analysis we must prove that $\mathcal{R}$ contains an "adequate approximation" of some min-cost alignment path τ. To state this precisely, we start with definitions and observations that formalize intuitive notions from the overview.

Cost and normalized cost. The *cost* of a path τ, cost(τ), from (u_1, u_2) to (v_1, v_2) in a grid-graph (see Section I), is the sum of the edge costs, and the *normalized cost* is ncost$(\tau) = \frac{\text{cost}(\tau)}{v_1 - u_1}$. cost$(G(I \times J))$ (or simply cost$(I \times J)$), the *cost of subgraph* $G(I \times J)$, is the min-cost of a path from the lower left to the upper right corner. The *normalized cost* is ncost$(I \times J) = \frac{1}{\mu(I)}$cost$(I \times J)$.

We note the following simple fact without proof:

Proposition 5. *For $I, J, J' \subseteq \{0, \ldots, n\}$, $|d_{edit}(x_I, y_J) - d_{edit}(x_I, y_{J'})| \leq |J \Delta J'|$, where Δ denotes symmetric difference.*

Projections and subpaths. The *horizontal projection* of a path $\tau = (i_1, j_1), \ldots, (i_\ell, j_\ell)$ is the set of $\{i_1, \ldots, i_\ell\}$. We say that τ *crosses box* $I \times J$ if the vertices of τ belong to $I \times J$ and its horizontal projection is I. If the horizontal projection of τ contains I', $\tau_{I'}$ denotes the (unique) minimal subpath of τ whose projection is I'.

Algorithm 2 DSR$(G, n, w, d, \delta, \epsilon)$
DenseStripRemoval

Input: $G = G_{x,y}(I_G \times J_G)$ for some $I_G, J_G \subseteq \{0, 1, \ldots, n\}$, $w, d \in [n]$, the endpoints of I_G and J_G are multiples of w and $\delta, \epsilon \in [0, 1]$.
Output: Set $\mathcal{S}$ which is a subset of the w-decomposition of I_G and a set $\mathcal{R}$ of δ-aligned certified w-boxes all with distance bound $5\epsilon_i$.

Initialization: $\mathcal{S} = \mathcal{R} = \emptyset$. $\mathcal{T} = \mathcal{I}_w(I_G)$.
$\mathcal{B}$, the set of y-candidates, is the set of width w δ-aligned subintervals of J_G (having endpoints a multiple of δw.)
while $\mathcal{T}$ is non-empty **do**
  Pick $I \in \mathcal{T}$
  Sample $c_0|\mathcal{B}|\frac{1}{d}\log n$ intervals $J \in \mathcal{B}$ uniformly at random and for each test if $\Delta_{\text{edit}}(x_I, y_J) \leq \epsilon$.
  if for at most $\frac{c_0}{2}\log n$ sampled J's, **SMALL-ED**$(x_I, y_J, \epsilon) < \infty$ **then**
    $\mathcal{S} = \mathcal{S} \cup \{I\}$; $\mathcal{T} = \mathcal{T} - \{I\}$. (*$I$ is declared sparse*)
  else
    (*I is declared dense and used as a pivot*)
    Compute:
    $\mathcal{Y} = \{J \in \mathcal{B};$ **SMALL-ED**$(x_I, y_J, 3\epsilon) < \infty\}$.
    $\mathcal{X} = \{I' \in \mathcal{T};$ **SMALL-ED**$(x_I, x_{I'}, 2\epsilon) < \infty\}$.
    Add $(I', J', 5\epsilon)$ to $\mathcal{R}$ for all pairs $(I', J') \in \mathcal{X} \times \mathcal{Y}$.
    $\mathcal{T} = \mathcal{T} - \mathcal{X}$.
  end if
end while
Output $\mathcal{S}$ and $\mathcal{R}$.

Proposition 6. *Let τ be a path with horizontal projection I, and let $I_1, \ldots, I_\ell$ be a decomposition of I. Then the τ_{I_j} are edge-disjoint and so:*

$$\begin{aligned} cost(\tau) &\geq \sum_{i=1}^{\ell} cost(\tau_{I_i}) \\ ncost(\tau) &\geq \sum_{i=1}^{\ell} \frac{\mu(I_i)}{\mu(I)} ncost(\tau_{I_i}). \end{aligned}$$

Definition 1. $(1-\delta)$**-cover.** *Let τ be a path with horizontal projection I and let $I' \times J'$ be a (not necessarily square) box with $I' \subseteq I$. For $\delta \in [0, 1]$ the box $I' \times J'$ $(1 - \delta)$-covers τ if the initial, resp. final, vertex of the subpath $\tau_{I'}$ is within $\delta\mu(I')$ vertical units of $(\min(I'), \min(J'))$, resp. $(\max(I'), \max(J'))$.*

Proposition 7. *Let $I' \times J'$ be a (not necessarily square) box that $(1 - \delta)$-covers path τ.*

1) *ncost$(I' \times J') \leq$ ncost$(\tau_{I'}) + 2\delta$.*

Algorithm 3 SSES$(G, \mathcal{S}, n, w_1, w_2, d, \delta, \epsilon, \theta)$ SparseStripExtensionSampling

Input: $G = G_{x,y}(I_G, J_G)$ with $I_G, J_G \subseteq \{0, 1, \ldots, n\}$, w_1, w_2, d, n are powers of 2, with $w_1, w_2, d < n$ and $w_1 < w_2$. Endpoints of I_G and J_G are multiples of w_2, $\mathcal{S}$ is a subset of the w_1-decomposition of I_G and δ, ϵ, θ are non-positive integral powers of 2.

Output: A set $\mathcal{R}$ of certified w_2-boxes in G.

Initialization: $\mathcal{R} = \emptyset$.
$\mathcal{B}$, the set of y-candidates, is the set of width w δ-aligned subintervals of J_G (endpoints are multiples of δw.)
for $I' \in \mathcal{I}_{w_2}(I_G)$ **do**
  if $\mathcal{S}$ includes a subset of I' **then**
    Select $c_1 \log^2 n$ intervals $I \in \mathcal{S}$ independently and uniformly at random from $\mathcal{I}_{w_1}(I') \cap \mathcal{S}$, to obtain $\mathcal{H}$.
    for each $I \in \mathcal{H}$ and each $J \in \mathcal{B}$ **do**
      if SMALL-ED$(x_I, y_J, \epsilon) < \infty$ **then**
        Let J' be such that $I' \times J'$ is the diagonal extension of $I \times J$ in $I' \times J_G$.
        Let $p =$ **SMALL-ED**$(x_{I'}, y_{J'}, 3\epsilon)$
        if $p < \infty$ **then**
          For $k = 0, \ldots, \log n$, add $(I', J', p + \theta + 2^{-k})$ to $\mathcal{R}$.
        end if
      end if
    end for
  end if
end for
Output $\mathcal{R}$.

2) *If J'' is any vertical interval, then $I' \times J''$ $(1 - \delta - |J' \Delta J''|/\mu(I'))$ covers τ.*

The routine proof is in the full version.

δ-aligned boxes A y-interval J of width w is *δ-aligned* for $\delta \in (0, 1)$ if its endpoints are multiples of δw (which we require to be an integer). The easy proof of the following is in the full version:

Proposition 8. *Let τ be a path that crosses $I \times J$. Suppose that $I' \subseteq I$ has width w, and $\mu(J) \geq w$.*

1) *There is an interval J^1 with $\mu(J^1) = \mu(I')$ so that ncost$(I' \times J^1) \leq 2$ncost$(\tau_{I'})$ and $I' \times J^1$ $(1 -$ ncost$(\tau_{I'}))$-covers τ.*
2) *There is a δ-aligned interval $J' \subseteq J$ of width w so that ncost$(I' \times J') \leq 2$ncost$(\tau_{I'}) + \delta$ and $I' \times J'$ $(1 -$ ncost$(\tau_{I'}) - \delta)$-covers τ.*

(J^1, J' are "τ-matches" for I', in the sense of the overview.)

Definition 2. 1) *The* main diagonal *of a box is the segment joining the lower left and upper right corners.*
2) *For a square box $I' \times J'$, and $I' \subseteq I$, the* true diagonal extension *of $I' \times J'$ to I is the square box $I \times \hat{J}$ whose main diagonal contains the main diagonal of $I' \times J'$.*
3) *For a w-box $I' \times J'$ contained in strip $I \times J$, the* adjusted diagonal extension of $I' \times J'$ within $I \times J$ *is the box $I \times J''$ obtained from the true diagonal extension of $I' \times J'$ to I by the minimal vertical shift so that it is a subset of $I \times J$. (The adjusted diagonal extension is the true diagonal extension if the true diagonal extension is contained in $I \times J$; otherwise it's lower edge is $\min(J)$ or its upper edge is $\max(J)$.)*

Proposition 9. *Suppose path τ crosses $I \times J$ and ncost$(\tau_I) \leq \epsilon$. Let $w = \mu(I)$. Let $I' \times J'$ be a w'-box that $(1 - \delta)$-covers $\tau_{I'}$. Then the adjusted diagonal extension $I \times J''$ of $I' \times J'$ within $I \times J$ $(1 - (\epsilon + \delta \frac{w'}{w}))$-covers τ and satisfies ncost$(I \times J'') \leq 3\epsilon + 2\delta \frac{w'}{w}$.*

The straightforward proof appears in the full version.

(k, ζ)-approximation of a path. This formalizes the notion of adequate approximation of a path by a certified box sequence.

Definition 3. *Let G be a grid graph on $I \times J$. Let $\zeta, \epsilon \in [0, 1]$. Let τ be a path that crosses G. A sequence of certified boxes $\sigma = \{(I_1 \times J_1, \epsilon_1), (I_2 \times J_2, \epsilon_2), \ldots, (I_\ell \times J_\ell, \epsilon_\ell)\}$* (k, ζ)-approximates *τ provided that:*

1) *$I_1, \ldots, I_\ell$ is a decomposition of I.*
2) *For each $i \in [\ell]$, $I_i \times J_i$ $(1 - \epsilon_i)$-covers τ.*
3) *$\sum_{i \in [\ell]} \epsilon_i \mu(I_i) \leq (k \cdot$ ncost$(\tau) + \zeta)\mu(I)$.*

Proposition 10. *Suppose path τ crosses $I \times J$ and $I_1, \ldots, I_m$ is a decomposition of I, and for $i \in [m]$, σ_i is a certified box sequence that (k, ζ)-approximates τ_{I_i}. Then $\sigma_1, \ldots, \sigma_m$ (k, ζ)-approximates τ.*

The routine proof appears in the full version

(d, δ, ϵ)-dense and -sparse. Fix a box $I \times J$. An interval $I' \subseteq I$ of width w is *(d, δ, ϵ)-sparse* (wrt $I \times J$) for integer d and $\epsilon, \delta \in (0, 1]$ if there are at most d δ-aligned w-boxes in $I' \times J$ of ncost at most ϵ, and is *(d, δ, ϵ)-dense* otherwise.

The sets $\mathcal{S}_i$ and $\mathcal{S}_i(I')$. For fixed k in the outer loop of CA, the set $\mathcal{S}$ created in iteration i of CA is denoted by $\mathcal{S}_i$. For any interval I', $\mathcal{S}_i(I')$ is the set of subintervals of I' belonging to $\mathcal{S}_i$.

Successful Sampling. The algorithm uses random sampling in two places, in the i loop inside CA and within the conditional on $\mathcal{S}$ containing a set from $\mathcal{I}_{w_1}(I')$ in SSES. We now specify what we need from the random sampling.

Definition 4. *A run of the algorithm has* successful sampling *provided that for every $k \in \{0, \ldots, 4/\theta\}$ and $i \in \{0, \ldots, \log \frac{1}{\theta})\}$ in the nested CA loops:*

- *For every w_1 interval I with endpoints a multiple of w_1, if I is $(\frac{d}{\epsilon_i}, \frac{\epsilon_i}{8}, \epsilon_i)$-dense interval (in terms of global parameters), DSR does not assign I to $\mathcal{S}$ and if I is $(\frac{d}{4\epsilon_i}, \frac{\epsilon_i}{8}, \epsilon_i)$-sparse, DSR places I in $\mathcal{S}$.*

- *On all calls to SSES, for every w_2 interval I with endpoints a multiple of w_2, if $|\mathcal{W}_i(I)|$ has size at least $|\mathcal{S}_i(I) - \mathcal{W}_i(I)|/32$ then the sample $\mathcal{H}$ selected contains an element of $\mathcal{W}_i(I)$. (Here $\mathcal{S}_i(I)$ and $\mathcal{W}_i(I)$ are that defined in the proof of Claim 14, whose definitions don't depend on the randomness used to select $\mathcal{H}$.)*

The proof of the following (via standard tail bounds) appears in the full version:

Proposition 11. *For large enough n, a run of CA has* successful sampling *with probability at least $1 - n^{-7}$*

We assume that coins are fixed in a way that gives successful sampling.

B. Properties of the covering algorithm

The main property of CA to be proved is:

Theorem 12. *Let x, y be strings of length n, $1/n \le \theta \le 1$ be a real. Let w_1, w_2, d satisfy $w_1 \le \theta w_2$, $w_2 \le \frac{\theta n}{4}$ and $1 \le d \le \frac{\theta n}{w_1}$. Assume n, w_1, w_2, d, θ are powers of 2. Let $\mathcal{R}$ be the set of weighted boxes obtained by running CA$(x, y, n, w_1, w_2, d, \theta)$ with $c_1 > 120$. Then (1) Every $(I \times J, \epsilon) \in \mathcal{R}$ is correctly certified, i.e., $\Delta_{edit}(x_I, y_J) \le \epsilon$, and (2) In a run that satisfies successful sampling, for every path τ from the source to the sink in $G = G_{x,y}$ of cost at most θ there is a subset of $\mathcal{R}$ that $(45, 15\theta)$-approximates τ.*

Proof: All boxes output are correctly certified: Each box in $\mathcal{R}_E$ comes from SSES which only certifies boxes with atleast their exact edit distance. For $(I \times J, \epsilon) \in \mathcal{R}_D$, there must be an I' such that $\Delta_{\text{edit}}(x_{I'}, y_J) \le \frac{3}{5} \cdot \epsilon$ and $\Delta_{\text{edit}}(x_{I'}, x_I) \le \frac{2}{5} \cdot \epsilon$ and so $\Delta_{\text{edit}}(x_I, y_J) \le \epsilon$.

It remains to establish (2). Fix a source-sink path τ of normalized cost κ. By Proposition 10 it is enough to show that for each $I' \in \mathcal{I}_{w_2}$, $\mathcal{R}$ contains a box sequence that $(45, 15\theta)$-approximates $\tau_{I'}$. So we fix $I' \in \mathcal{I}_{w_2}$.

The main loop (on k) of CA processes G in overlapping boxes. Since $\text{ncost}(\tau) \le \theta$, one of these boxes, which we'll call $I \times J$, must contain $\tau_{I'}$. (See the full version for a proof.)

We note:

Claim 13. *Let $i \in \{0, \ldots, \log 1/\theta\}$. Suppose $I'' \in \mathcal{I}_{w_1}(I)$ and $J'' \subseteq J$ is $\epsilon_i/8$-aligned. If $I'' \notin \mathcal{S}_i$ and cost$(I'' \times J'') \le \epsilon_i$ then $(I'' \times J'', 5\epsilon_i) \in \mathcal{R}_D$.*

Proof: If $I'' \notin \mathcal{S}_i$ then in the call to DSR$(G(I \times J), n, w_1, d/\epsilon_i, \epsilon_i/8, \epsilon_i)$ there is an iteration of the main loop,where the selected interval $\tilde{I}$ from $\mathcal{T}$ is declared dense and $\Delta_{\text{edit}}(x_{\tilde{I}}, x_{I''}) \le 2\epsilon_i$. Since $\Delta_{\text{edit}}(x_{I''}, y_{J''}) \le \epsilon_i$, $\Delta_{\text{edit}}(x_{\tilde{I}}, y_{J''}) \le 3\epsilon_i$ and so $I'' \in \mathcal{X}$ and $J'' \in \mathcal{Y}$. Thus, DSR certifies $(I'' \times J'', 5\epsilon_i)$, which is added to $\mathcal{R}_D$. ∎

The theorem follows from:

Claim 14. *For an interval $I' \in \mathcal{I}_{w_2}$, assuming successful sampling $\mathcal{R}_E$ or $\mathcal{R}_D$ contains a $(45, 15\theta)$-approximation of $\tau_{I'}$.*

The proof is similar to that of Claim 4, with adjustments for some technicalities.

Proof: Let $\tau' = \tau_{I'}$ and $\kappa = \text{ncost}(\tau')$. Let $\mathcal{I}' = \mathcal{I}_{w_1}(I')$. For $I'' \in \mathcal{I}'$, let $\kappa_{I''} = \text{ncost}(\tau_{I''})$. By Proposition 8, for all $I'' \in \mathcal{I}'$ and $\epsilon_i \ge \kappa_{I''}$ there is an $\epsilon_i/8$-aligned vertical interval $J_i^\tau(I'')$, such that $\text{ncost}(I'' \times J_i^\tau(I'')) \le 2\kappa_{I''} + \epsilon_i/8$ and $I'' \times J_i^\tau(I'')$ $(1 - \kappa_{I''} - \epsilon_i/8)$-covers $\tau_{I'}$.

Let $s(I'')$ be the largest integer such that $\epsilon_{s(I'')} \ge 3\kappa_{I''} + \kappa + \theta$. Let $t(I'') \le s(I'')$ be the largest integer such that $I'' \notin \mathcal{S}_{t(I'')}$. (Since $\theta n/w_1 \ge d$, $\mathcal{S}_0 = \emptyset$, so $t(I'')$ is well-defined.) Let $a(I'') = \epsilon_{s(I'')}$ (this plays a similar role to $u(I'')$ in Section II) and $b(I'') = \epsilon_{t(I'')}$.

For all $\epsilon_i \in [a(I''), b(I'')]$, $\text{ncost}(I'' \times J_i^\tau(I'')) \le \epsilon_i$ and $I'' \times J_i^\tau(I'')$ $(1 - \epsilon_i)$-covers τ'. By the definition of $b(I'')$ and Claim 13, $\mathcal{R}_D$ contains the certified box $(I'' \times J_{t(I'')}^\tau(I''), 5b_{I''})$. So $\mathcal{R}_D$ contains a $(45, 15\theta)$-approximation of τ' provided that:

$$\sum_{I'' \in \mathcal{I}'} 5b(I'') \le \frac{45}{8} \sum_{I'' \in \mathcal{I}'} a(I'') \tag{4}$$

since $a(I'') \le 2(3\kappa_{I''} + \kappa + \theta)$.

Next we determine a sufficient condition that $\mathcal{R}_E$ contain a box sequence (consisting of a single box) that $(5, 4\theta)$-approximates τ'. Let $\mathcal{S}_i(I') = \mathcal{S}_i \cap \mathcal{I}'$. Interval $I'' \in \mathcal{S}_i(I')$ is a *winner for iteration i* if $\epsilon_i \ge a(I'')$. This set of winners is denoted by $\mathcal{W}_i(I')$. It suffices that during iteration i, the set of $c_1 \log^2 n$ samples taken in SSES includes a winner I''; then since $\Delta_{\text{edit}}(I'', J_i^\tau(I'')) \le \epsilon_i$, the (adjusted) diagonal extension $I' \times \tilde{J}$ of $I'' \times J_i^\tau(I'')$ will be certified. By Proposition 9, $I' \times \tilde{J}$ has normalized cost at most $3\kappa + 2\epsilon_i w_1/w_2 \le 3\kappa + 2\theta \le 3\epsilon_i$ and it $(1 - (\kappa + \theta))$-covers τ'. If $\kappa = 0$ then $(I' \times \tilde{J}, \text{ncost}(I' \times \tilde{J}) + \theta + 2^{-\log n})$ is in $\mathcal{R}_E$ by the behavior of SSES and it $(5, 4\theta)$-approximates τ'. Otherwise $\kappa \ge 1/n$; so set $k = \lfloor \log 1/\kappa \rfloor$. Thus, $k \le \log n$ and $2^{-k} \in [\kappa, 2\kappa)$. Then $(I' \times \tilde{J}, \text{ncost}(I' \times \tilde{J}) + \theta + 2^{-k})$ is in $\mathcal{R}_E$ and it $(5, 4\theta)$-approximates τ'.

Under successful sampling if $|\mathcal{W}_i(I')| \ge \frac{1}{32}|\mathcal{S}_i(I') - \mathcal{W}_i(I')|$, at least one interval from $\mathcal{W}_i(I')$ will be included in our $c_1 \log^2 n$ samples during SSES and $\mathcal{R}_E$ will contain a $(5, 4\theta)$-approximation of τ' as above. So suppose this fails:

$$\text{For all } i,\ |\mathcal{W}_i(I')| < \frac{1}{32}|\mathcal{S}_i(I') - \mathcal{W}_i(I')|. \tag{5}$$

We show that this implies (4). Multiplying (5) by ϵ_i and summing on i yields:

$$\sum_{I'' \in \mathcal{I}'} \sum_{i: I'' \in \mathcal{W}_i(I')} \epsilon_i < \frac{1}{32} \sum_{I'' \in \mathcal{I}'} \sum_{i: I'' \in \mathcal{S}_i(I') - \mathcal{W}_i(I')} \epsilon_i. \tag{6}$$

$I'' \in \mathcal{S}_i(I') - \mathcal{W}_i(I')$ implies $\epsilon_i < a(I'')$. Summing the geometric series:

$$\sum_{i: I'' \in \mathcal{S}_i(I') - \mathcal{W}_i(I')} \epsilon_i \le 2a(I''). \tag{7}$$

Either $a(I'') = b(I'')$ or $a(I'') < b(I'')$. If the latter, then $I'' \in \mathcal{W}_i(I')$ for $\epsilon_i = b(I'')/2$. So:

$$\begin{aligned}\sum_{I'' \in \mathcal{I}'} b(I'') &\leq \sum_{I''} \Big(a(I'') + \sum_{i: I'' \in \mathcal{W}_i(I')} 2\epsilon_i \Big) \\ &< \sum_{I''} \Big(a(I'') + \frac{1}{16} \sum_{i: I'' \in \mathcal{S}_i(I') - \mathcal{W}_i(I')} \epsilon_i \Big) \\ &\leq \frac{9}{8} \sum_{I'' \in \mathcal{I}'} a(I'')\end{aligned}$$

which implies Equation 4. (The second inequality follows from (6) and the last inequality from (7).) ■

■

C. Time complexity of CA

We write $t(w, \epsilon)$ for the time of **SMALL-ED**(z_1, z_2, ϵ) on strings of length w. We assume $t(w, \epsilon) \geq w$, and that for $k \geq 1$, there is a constant $c(k)$ such that for all $\epsilon \in [0, 1]$ and all $w > 1$, $t(w, k\epsilon) \leq c(k) \cdot t(w, \epsilon) + c(k)$. As mentioned earlier, by [4], we can use $t(w, \epsilon) = O(w^2\epsilon)$.

Theorem 15. *Let n be a sufficiently large power of 2 and $\theta \in [1/n, 1]$ be a power of 2. Let x, y be strings of length n. Let $\log n \leq w_1 \leq w_2 \leq \theta n/4$, $1 \leq d \leq n$ be powers of 2, where $w_1 | w_2$ and $w_2 | n$, and $w_1/w_2 \leq \theta$. The size of the set $\mathcal{R}$ output by CA is $O((\frac{n}{w_1})^2 \log^2 n)$ and in any run that satisfies successful sampling, CA runs in time:*

$$O\Bigg(|\mathcal{R}| + \sum_{\substack{k=\log 1/\theta, \ldots, 0 \\ \epsilon = 2^{-k}}} \Big(\frac{\theta n^2 \log n}{d\epsilon w_1^2} \cdot t(w_1, \epsilon) + \frac{\theta n^2 \log^2 n}{w_1 w_2 \epsilon} \cdot t(w_1, \epsilon) + \frac{nd \log^2 n}{w_2 \epsilon} \cdot t(w_2, \epsilon) \Big) \Bigg).$$

Proof: To bound $|\mathcal{R}|$ note that for each choice of k, i in the outer and inner loops of CA, the set of candidate boxes of width w_1 has size $O(\frac{\theta n}{w_1} \frac{\theta n}{w_1 \epsilon_i})$. This upper bounds the number of boxes certified by DSR. The call to SSES constructs at most one diagonal extension for each such candidate box, and each diagonal extension gives rise to at most $O(\log n)$ certified boxes. Thus, for each (k, i) there are $O(\frac{\theta^2 n^2 \log n}{(w_1)^2 \epsilon_i})$ certified boxes. Summing the geometric series over i, noting that $\min(\epsilon_i) = \theta$, and summing over $O(1/\theta)$ values of k gives the required bound on $|\mathcal{R}|$.

The steps in the algorithm that actually construct certified boxes (13 of DSR, 11 of SSES, 13 of CA) cost $O(1)$ per box giving the first term in the time bound.

We next bound the other contributions to runtime. The outer loop of CA has $\frac{4}{\theta} + 1$ iterations on k's. The inner loop has $1 + \log \frac{1}{\theta}$ iterations on i. Each iteration invokes DSR and SSES on $I \times J$ with I and J of width at most $4\theta n$.

We bound the time of a call to DSR. To distinguish between local variables of DSR and global variables of CA, we denote local input variables as $\hat{G}, \hat{n}, \hat{w}, \hat{d}, \hat{\delta}, \hat{\epsilon}$. For $\mathcal{B}$ and $\mathcal{T}$ as in DSR, $|\mathcal{B}| \leq \frac{\mu(I_{\hat{G}})}{\hat{\delta}\hat{w}}$. since $\mu(I_{\hat{G}}) = \mu(J_{\hat{G}})$. The main while loop of DSR repeatedly picks intervals $I \in \mathcal{T}$ and samples $c_0 |\mathcal{B}| \frac{\log \hat{n}}{\hat{d}} \leq \frac{c_0 \mu(I_{\hat{G}}) \log \hat{n}}{\hat{d}\hat{\delta}\hat{w}}$ vertical intervals J and tests whether $\Delta_{\text{edit}}(x_I, y_J) \leq \hat{\epsilon}$. Each such test takes time $t(\hat{w}, \hat{\epsilon})$. This is done at most once for each of the $\mu(I_{\hat{G}})/\hat{w}$ horizontal candidates for a total time of $O(\frac{\mu(I_{\hat{G}})^2 \log \hat{n}}{\hat{w}^2 \hat{\delta}\hat{d}}) t(\hat{w}, \hat{\epsilon})$. We next bound the cost of processing a pivot I. This requires testing $\Delta_{\text{edit}}(x_I, y_J) \leq 3\hat{\epsilon}$ for $J \in \mathcal{B}$ and $\Delta_{\text{edit}}(x_I, x_{I'}) \leq 2\hat{\epsilon}$ for $I' \in \mathcal{T}$. Each test costs $O(t(\hat{w}, \hat{\epsilon}))$ (by our assumption on $t(\cdot, \cdot)$), and since $|\mathcal{T}| \leq |\mathcal{B}| = \frac{\mu(I_{\hat{G}})}{\hat{w}\hat{\delta}}$, I is processed in time $O(\frac{\mu(I_{\hat{G}})}{\hat{w}\hat{\delta}} t(\hat{w}, \hat{\epsilon}))$. This is multiplied by the number of intervals declared dense, which we now upper bound. If I is declared dense then at the end of processing I, $\mathcal{X}$ is removed from $\mathcal{T}$. This ensures $\Delta_{\text{edit}}(I, I') > 2\epsilon$ for any two intervals I, I' declared dense. By the triangle inequality the sets $\mathcal{B}(I) = \{J \in \mathcal{B};\ \Delta_{\text{edit}}(x_I, y_J) \leq \epsilon\}$ are disjoint for different pivots. By successful sampling, for each pivot I, $|\mathcal{B}(I)| \geq \frac{\hat{d}}{4}$, and thus at most $|\mathcal{B}|/(\hat{d}/4) = \frac{4\mu(I_{\hat{G}})}{\hat{d}\hat{\delta}\hat{w}}$ intervals are declared dense, so all intervals declared dense are processed in time $O(\frac{\mu(I_{\hat{G}})^2}{\hat{w}^2 \hat{d}\hat{\delta}^2}) t(\hat{w}, \hat{\epsilon})$.

The time for dense/sparse classification of intervals and for processing intervals declared dense is at most $O(\frac{\mu(I_{\hat{G}})^2 \log \hat{n}}{\hat{w}^2 \hat{d}\hat{\delta}^2}) t(\hat{w}, \hat{\epsilon})$. During iteration i of the inner loop of CA, the local variables of DSR are set as $\hat{n} = n$, $\mu(I_{\hat{G}}) \leq 4\theta n$, $\hat{w} = w_1$, $\hat{d} = d/\epsilon_i$, $\hat{\delta} = \epsilon_i/8$. Substituting these parameters yields time $O(\frac{\theta^2 n^2 \log n}{(w_1)^2 d\epsilon_i}) t(w_1, \epsilon_i)$. Multiplying by the $O(1/\theta)$ iterations on k gives the first summand of the theorem.

Next we turn to SSES. The local input variables $n, w_1, w_2, \mathcal{S}, \theta$ are set to their global values so we denote them without $\hat{}$. The other local input variables are denoted as $\hat{G}, \hat{d}, \hat{\delta}, \hat{\epsilon}$. The local variable $\mathcal{B}$ has size $\frac{\mu(I_{\hat{G}})}{\hat{\delta} w_1}$. By successful sampling, we assume that on every call, every interval in $\mathcal{S}$ is $(\hat{d}, \hat{\delta}, \hat{\epsilon})$- sparse. The outer loop enumerates the $\mu(I_{\hat{G}})/w_2$ intervals I' of $\mathcal{I}_{w_2}(I_{\hat{G}})$. We select $\mathcal{H}$ to be $c_1 \log^2 n$ random subsets from subsets of I' belonging to $\mathcal{S}$. For each $I \in \mathcal{H}$ and $J \in \mathcal{B}$, we call **SMALL-ED**$(x_I, y_J, \hat{\epsilon})$, taking time $t(w_1, \hat{\epsilon})$. The total time of all tests is $O(\frac{\mu(I_{\hat{G}})^2 \log^2 n}{\hat{\delta} w_1 w_2}) t(w_1, \hat{\epsilon})$. Using $\hat{d} = d/\epsilon_i$, $\hat{\delta} = \epsilon_i/8$ and $\hat{\epsilon} = \epsilon_i$ from the ith call to SSES gives $O(\frac{\theta^2 n^2 \log^2 n}{\epsilon_i w_1 w_2}) t(w_1, \epsilon_i)$. Multiplying by the $O(1/\theta)$ iterations on k gives the second summand in the theorem.

Assuming successful sampling, all intervals in the set $\mathcal{S}$ passed from DSR to SSES are $(\hat{d}, \hat{\delta}, \hat{\epsilon})$-sparse. Therefore, for each sampled I, at most $\hat{d}$ intervals J are within $\hat{\epsilon}$ of I. For each of these we do a diagonal extension of $I \times J$ to a w_2-box $I' \times J'$, and call **SMALL-ED**$(x_{I'}, y_{J'}, 3\hat{\epsilon})$ at cost $O(t(w_2, \hat{\epsilon}))$ for each call. The number of such calls is $O(\frac{\mu(I_{\hat{G}}) \hat{d} \log^2 n}{w_2})$. Using the parameter $\hat{d} = d/\epsilon_i$ in the ith call of the inner iteration of CA, we get a cost of

$O(\frac{\theta n d \log^2 n}{\epsilon_i w_2})t(w_2, \epsilon_i)$ and multiplying by the $O(1/\theta)$ gives the third summand in the theorem. ∎

Choosing the parameters to minimize the maximum term in the time bound, subject to the restrictions of the theorem and using $t(w, \epsilon) = O(\epsilon w^2)$ we have:

Corollary 16. *For all sufficient large n, and for $\theta \geq n^{-1/5}$ (both powers of 2) choosing w_1, w_2, and d to be the largest powers of two satisfying: $w_1 \leq \theta^{-2/7}(n)^{1/7}$, $w_2 \leq \theta^{1/7}(n)^{3/7}$, and $d \leq \theta^{3/7}(n)^{2/7}$, with probability at least $1 - n^{-1/7}$, CA runs in time $\tilde{O}(n^{12/7}\theta^{4/7})$, and outputs the set $\mathcal{R}$ of size at most $\tilde{O}(n^{12/7}\theta^{4/7})$.*

IV. Min-cost Paths in Shortcut Graphs

We now describe the second phase of our algorithm, which uses the set $\mathcal{R}$ output by CA to upper bound $d_{\text{edit}}(x, y)$. A *shortcut graph* on vertex set $\{0, \ldots, n\} \times \{0, \ldots, n\}$ consists of the H and V edges of cost 1, together with an arbitrary collection of *shortcut* edges $(i, j) \to (i', j')$ where $i < i'$ and $j < j'$, also denoted by $e_{I,J}$ where $I = \{i, \ldots, i'\}$ and $J = \{j, \ldots, j'\}$, along with their costs. A *certified graph* (for x, y) is a shortcut graph where every shortcut edge $e_{I,J}$ has cost at least $d_{\text{edit}}(x_I, y_J)$. The min cost path from $(0, 0)$ to (n, n) in a certified graph upper bounds $d_{\text{edit}}(x, y)$. The second phase algorithm uses $\mathcal{R}$ to construct a certified graph, and computes the min cost path to upper bound on $d_{\text{edit}}(x, y)$.

A certified box $(I \times J, \kappa)$ corresponds to the $e_{I,J}$ with cost $\kappa\mu(I)$. (In the certified graph we use non-normalized costs.) However, the certified graph built from $\mathcal{R}$ in this way may not have a path of cost $O(d_{\text{edit}}(x, y) + \theta n)$. We need a modified conversion of $(I \times J, \kappa)$. If $\kappa \geq 1/2$ we add no shortcut. Otherwise $(I \times J, \kappa)$ converts to the edge $e_{I,J'}$ with cost $3\kappa\mu(I)$ where J' is obtained by shrinking J: $\min(J') = \min(J) + \ell$ and $\max(J') = \max(J') - \ell$ where $\ell = \lfloor \kappa\mu(I) \rfloor$. By Proposition 5, this is a certified edge. Call the resulting graph $\widetilde{G}$. The following straightforward claim is proved in the full version:

Lemma 17. *Let τ be a path from source to sink in $G_{x,y}$. If $\mathcal{R}$ contains a sequence σ that (k, θ)-approximates τ then there is a source-sink path τ' in $\widetilde{G}$ that consists of the shortcuts corresponding to σ together with some H and V edges with $cost_{\widetilde{G}}(\tau') \leq 5(k \cdot cost_{G_{x,y}}(\tau) + \theta n)$.*

Computing the min-cost. We present an $O(n+m\log(mn))$ algorithm to find a min cost source-sink path in a shortcut graph $\widetilde{G}$ with m shortcuts. It's easier to switch to the max-benefit problem: Let $\widetilde{H}$ be the same graph with cost c_e of $e = (i, j) \to (i', j')$ replaced by *benefit* $b_e = (i' - i) + (j' - j) - c_e$, (so H and V edges have benefit 0). The min-cost path of $\widetilde{G}$ is $2n$ minus the max-benefit path of $\widetilde{H}$. To compute the max-benefit path of $\widetilde{H}$, we use a binary tree data structure with leaves $\{1, \ldots, n\}$, where each node v stores a number b_v, and a collection of lists $L_1, \ldots, L_n$, where L_i stores pairs $(e, q(e))$ where the head of e has x-coordinate i and $q(e)$ is the max benefit of a path that ends with e.

We proceed in $n - 1$ rounds. Let the set A_i consist of all the shortcuts whose tail has x-coordinate i. The preconditions for round i are: (1) for each leaf j, the stored value b_j is the max benefit path to (i, j) that includes a shortcut whose head has y-coordinate j (or 0 if there is no such path), (2) for each internal node v, $b_v = \max\{b_j : j \text{ is a leaf in the subtree of } v\}$. and (3) for every edge $e = (i', j') \to (i'', j'')$ with $i' < i$, the value $q(e)$ has been computed and $(e, q(e))$ is in list $L_{i''}$. During round i, for each shortcut $e = (i, j) \to (i', j')$ in A_i, $q(e)$ equals the max of $b_v + b_e$ over tree leaves v with $v \leq j$. This can be computed in $O(\log n)$ time as max $b_v + b_e$, over $\{j\}$ union the set of left children of vertices on the root-to-j path that are not themselves on the path. Add $(e, q(e))$ to list $L_{i'}$. After processing A_i, update the binary tree: for each $(e, q(e)) \in L_{i+1}$, let j be the y-coordinate of the head of e and for all vertices v on the root-to-j path, replace b_v by $\max(b_v, q(e))$. The tree then satisfies the precondition for round $i + 1$. The output of the algorithm is b_n at the end of round $n - 1$. It takes $O(n)$ time to set up the data structure, $O(m \log m)$ time to sort the shortcuts, and $O(\log n)$ processing time per shortcut (computing $q(e)$ and later updating the data structure).

V. Summing up and speeding up

To summarize, the algorithm **GAP-UB**$_\theta$ runs CoveringAlgorithm of Section III, converts the output into a shortcut graph, and runs the min-cost path algorithm of Section IV. By Corollary 16, and the quasilinear runtime (in the number of shortcuts) of the min-cost path algorithm, the algorithm **GAP-UB**$_\theta$ runs in time $\tilde{O}(n^{12/7}\theta^{4/7})$. The construction of the main algorithm **ED-UB** from **GAP-UB** is standard:

Proof of Theorem 1 from Theorem 2: Given **GAP-UB**$_\theta$, we construct **ED-UB**: Run the aforementioned exact algorithm of [3] with runtime $O(n + k^2)$ time on instances of edit distance k, for $O(n + n^{2-2/5})$ time. If it terminates then it outputs the exact edit distance. Otherwise, the failure to terminate implies $d_{\text{edit}}(x, y) \geq n^{4/5}$. Now run **GAP-UB**$_{\theta_j}(x, y)$ for $\theta_j = (1/2)^j$ for $j = \{0, \ldots, \frac{\log n}{5}\}$ and output the minimum of all upper bounds obtained. Let j be the largest index with $\theta_j n \geq d_{\text{edit}}(x, y)$ (such an index exists since $j = 0$ works). The output is at most $840\theta_j n \leq 1680 d_{\text{edit}}(x, y)$. We run at most $O(\log n)$ iterations, each with runtime $\tilde{O}(n^{2-2/7})$. ∎

Speeding up the algorithm. The runtime of **ED-UB** is dominated by the cost of **SMALL-ED**(z_1, z_2, ϵ) on pairs of strings of length $w \in \{w_1, w_2\}$. We use Ukkonen's algorithm [4] with $t(w, \epsilon) = O(w^2\epsilon)$. In the full paper we describe a revised algorithm **ED-UB**$_1$, replacing the Ukkonen's algorithm with **ED-UB**. This worsens the approximation factor (roughly multiplying it by the approximation factor of **ED-UB**) but improves runtime. The internal parameters

w_1, w_2, d are adjusted to maximize savings. One can iterate this process any constant number of times to get faster algorithms with worse (but still constant) approximation factors. Because of the dependence of the analysis on θ, we do not get a faster edit distance algorithm for all $\theta \in [0,1]$ but only for θ close to 1. (This may be an artifact of our analysis rather than an inherent limitation.)

Theorem 18. *For $\epsilon > 0$, there are constants $c > 1$ and $\beta \in (0,1)$ and an algorithm with runtime $O(n^{\frac{1+\sqrt{5}}{2}+\epsilon})$ that on input x, y of length n, outputs u such that $d_{edit}(x,y) \leq u \leq cd_{edit}(x,y) + n^{1-\beta}$ with probability at least $1 - 1/n$.*

Acknowledgements. We thank the FOCS 2018 committee and especially several anonymous referees for helpful comments and suggestions. Michael Saks thanks C. Seshadhri for insightful discussions on the edit distance problem.

REFERENCES

[1] V. Levenshtein, "Binary Codes Capable of Correcting Deletions, Insertions and Reversals," *Soviet Physics Doklady*, vol. 10, p. 707, 1966.

[2] R. A. Wagner and M. J. Fischer, "The string-to-string correction problem," *J. ACM*, vol. 21, no. 1, pp. 168–173, Jan. 1974.

[3] G. M. Landau, E. W. Myers, and J. P. Schmidt, "Incremental string comparison," *SIAM J. Comput.*, vol. 27, no. 2, pp. 557–582, Apr. 1998.

[4] E. Ukkonen, "Algorithms for approximate string matching," *Inf. Control*, vol. 64, no. 1-3, pp. 100–118, Mar. 1985.

[5] W. J. Masek and M. S. Paterson, "A faster algorithm computing string edit distances," *Journal of Computer and System Sciences*, vol. 20, no. 1, pp. 18 – 31, 1980.

[6] S. Grabowski, "New tabulation and sparse dynamic programming based techniques for sequence similarity problems," *Discrete Applied Mathematics*, vol. 212, pp. 96–103, 2016.

[7] A. Backurs and P. Indyk, "Edit distance cannot be computed in strongly subquadratic time (unless SETH is false)," in *Proceedings of the Forty-Seventh Annual ACM on Symposium on Theory of Computing*, ser. STOC '15. New York, NY, USA: ACM, 2015, pp. 51–58.

[8] A. Abboud, T. D. Hansen, V. V. Williams, and R. Williams, "Simulating branching programs with edit distance and friends: or: a polylog shaved is a lower bound made," in *Proceedings of the 48th Annual ACM SIGACT Symposium on Theory of Computing, STOC 2016, Cambridge, MA, USA, June 18-21, 2016*, 2016, pp. 375–388.

[9] A. Abboud, A. Backurs, and V. V. Williams, "Tight hardness results for LCS and other sequence similarity measures," in *IEEE 56th Annual Symposium on Foundations of Computer Science, FOCS 2015, Berkeley, CA, USA, 17-20 October, 2015*, 2015, pp. 59–78.

[10] K. Bringmann and M. Künnemann, "Quadratic conditional lower bounds for string problems and dynamic time warping," in *IEEE 56th Annual Symposium on Foundations of Computer Science, FOCS 2015, Berkeley, CA, USA, 17-20 October, 2015*, 2015, pp. 79–97.

[11] Z. Bar-Yossef, T. Jayram, R. Krauthgamer, and R. Kumar, "Approximating edit distance efficiently," in *Foundations of Computer Science, 2004. Proceedings. 45th Annual IEEE Symposium on*, Oct 2004, pp. 550–559.

[12] T. Batu, F. Ergun, and C. Sahinalp, "Oblivious string embeddings and edit distance approximations," in *Proceedings of the Seventeenth Annual ACM-SIAM Symposium on Discrete Algorithm*, ser. SODA '06. Philadelphia, PA, USA: Society for Industrial and Applied Mathematics, 2006, pp. 792–801.

[13] A. Andoni and K. Onak, "Approximating edit distance in near-linear time," in *Proceedings of the Forty-first Annual ACM Symposium on Theory of Computing*, ser. STOC '09. New York, NY, USA: ACM, 2009, pp. 199–204.

[14] T. Batu, F. Ergün, J. Kilian, A. Magen, S. Raskhodnikova, R. Rubinfeld, and R. Sami, "A sublinear algorithm for weakly approximating edit distance," in *Proceedings of the Thirty-fifth Annual ACM Symposium on Theory of Computing*, ser. STOC '03. New York, NY, USA: ACM, 2003, pp. 316–324.

[15] A. Andoni, R. Krauthgamer, and K. Onak, "Polylogarithmic approximation for edit distance and the asymmetric query complexity," in *51th Annual IEEE Symposium on Foundations of Computer Science, FOCS 2010, October 23-26, 2010, Las Vegas, Nevada, USA*, 2010, pp. 377–386.

[16] A. Abboud and A. Backurs, "Towards hardness of approximation for polynomial time problems," in *8th Innovations in Theoretical Computer Science Conference, ITCS 2017, January 9-11, 2017, Berkeley, CA, USA*, 2017, pp. 11:1–11:26.

[17] M. Boroujeni, S. Ehsani, M. Ghodsi, M. T. Hajiaghayi, and S. Seddighin, "Approximating edit distance in truly subquadratic time: Quantum and MapReduce," in *Proceedings of the Twenty-Ninth Annual ACM-SIAM Symposium on Discrete Algorithms, SODA 2018, New Orleans, LA, USA, January 7-10, 2018*, 2018, pp. 1170–1189.

[18] ——, "Approximating edit distance in truly subquadratic time: Quantum and MapReduce (extended version of [17])," 2018.

[19] A. Andoni and H. L. Nguyen, "Near-optimal sublinear time algorithms for Ulam distance," in *Proceedings of the Twenty-First Annual ACM-SIAM Symposium on Discrete Algorithms, SODA 2010, Austin, Texas, USA, January 17-19, 2010*, 2010, pp. 76–86.

[20] T. Naumovitz, M. E. Saks, and C. Seshadhri, "Accurate and nearly optimal sublinear approximations to Ulam distance," in *Proceedings of the Twenty-Eighth Annual ACM-SIAM Symposium on Discrete Algorithms, SODA 2017, Barcelona, Spain, Hotel Porta Fira, January 16-19*, 2017, pp. 2012–2031.