Fast FPT-algorithms for cleaning grids*

Josep Díaz Dimitrios M. Thilikos

Abstract

We consider the problem that, given a graph G and a parameter k, asks whether the edit distance of G and a rectangular grid is at most k. We examine the general case where the edit operation are vertex/edge removals and additions. If the dimensions of the grid are given in advance, then we give a parameterized algorithm that runs in $2^{O(\log k \cdot k)} + O(n^3)$ steps. In the case where the dimensions of the grid are not given we give a parameterized algorithm that runs in $2^{O(\log k \cdot k)} + O(k^2 \cdot n^3)$ steps. We insist on parameterized algorithms with running times where the relation between the polynomial and the non-polynomial part is additive. For this, our algorithm design is based on the technique of kernelization. In particular we prove that for each version of the above problem there exists a kernel of size $O(k^4)$.

1 Introduction

We consider the problem of measuring the *degree of similarity* of two graphs G and H, where the degree of similarity is considered to be the minimum number of *edit operations* needed to transform one graph into the other. This measure of similarity is also known as the *edit distance* between graphs. The problem has received a lot of attention due to its multiple applications in patten recognition and computer vision among others topics. In the more usual setting the edit operations are: contract an edge, applying an inverse contraction[1] of a vertex and (in case of labeled graphs) relabel an edge. In general, the problem is *NP*-hard and can be computed in polynomial time on trees (see for ex. [11]). For a recent update on the heuristics developed for the edit distance problem see [9].

In this paper, we examine decision problems associated with the editing distance between G and H when H is a grid of size $p \times q$ (we call such a grid (p,q)-*grid and we denote it as* $H_{p,q}$), where the edit operations are either the removal or the insertion of either an edge or a vertex. We represent these operations by the members of the set $\mathcal{U} = \{\mathsf{e}\text{-out}, \mathsf{e}\text{-in}, \mathsf{v}\text{-out}, \mathsf{v}\text{-in}\}$. Given $\mathcal{E} \subseteq \mathcal{U}$, we denote $\mathcal{E}$-**dist**(G,H) as the *edit distance of* G *and* H *with respect to* $\mathcal{E}$, which is the minimum number of operations in the set $\mathcal{E}$ that when applied to G can transform it to H.

*Departament de Llenguatges i Sistemes Informàtics, Universitat Politècnica de Catalunya, Campus Nord – Edifici Ω, c/Jordi Girona Salgado 1-3, E-08034, Barcelona, Spain. This research was supported by the spanish CICYT project TIN-2004-07925 (GRAMMARS). The first author was partially supported by the *Distinció per a la Promoció de la Recerca de la GC, 2002*. Emails: {diaz,sedthilk}@lsi.upc.edu

[1] H' is the result of the inverse contraction of a vertex v of H if H is the result of the contraction of an edge to vertex v.

Our problem setting has been motivated by the problem of *regularity extraction* in digital systems, where we are looking for regular patterns in complicated circuits (usually VLSI circuits) indicating ways to organize or embed them (for an example, see [8]). In particular, we study the problem that given a graph G, a pair $p, q \in \mathbb{N}$ and a set $\mathcal{E} \subseteq \mathcal{U}$, compute the minimum number of operations needed to transform G to a (p, q)-grid. This problem is NP-complete, as can be seeing considering the particular case where $q = 1$ and $\mathcal{E}$ contains only the *erasing a vertex* operation, is equivalent to the Longest Induced Path problem, i.e. the problem of given a graph G and a constant $k \geq 0$ decide if G contains a simple path of length at least k as an induced subgraph (we just set $k \leftarrow |V(G)| - k$).

Our study adopts the point of view of parameterized complexity introduced by Downey and Fellows (see [2]). We consider parameterizations of hard problems, i.e. problems whose input contains some (in general small) parameter k and some main part. A parameterized problem belongs in the class FPT if it can be solved by an algorithm of time complexity $g(k) \cdot n^{O(1)}$ where g is a non-polynomial function of k and n is the size of the problem (we call such an algorithm *FPT-algorithm*). A popular technique on the design of parameterized algorithms is *kernelization*. Briefly, this technique, consists in finding a polynomial-time reduction of a parameterized problem to itself in a way that the sizes of the instances of the new problem, we call it *kernel*, depend *only* on the parameter k. The function that bounds the size of the main part of the reduced problem determines the *size* of the kernel and is usually of polynomial (on k) size. Notice that if a parameterized problem admits a reduction to a problem kernel, then it is in FPT because any brute force algorithm can solve the reduced problem in time that does not depend on the main part of the original problem. Notice also that this technique provides FPT-algorithms of time complexity $g(k)+n^{O(1)}$ where the non-polynomial part $g(k)$ is just additive to the overall complexity.

In Section 3, we prove that the k-Almost grid defined below is FPT[2] by giving a kernel of size $O(k^4)$ or $O(k^3)$ depending on the size of the grid we are looking for.

k-Almost grid
Input: A graph G, two positive integers p, q, a non-negative integer k, and a set $\mathcal{E} \subseteq \mathcal{U}$.
Parameter: A non-negative integer k.
Question: Can G be transformed to a (p, q)-grid after at most k edit operations from $\mathcal{E}$?

Notice that our parameterization also includes the "dual" parameterization of the Longest Induced Path problem, in the sense that now the parameter is not the length of the induced path but the number of vertices in G that are *absent* in such a path. (The parameterized complexity of the "primal" parameterization of the Longest Induced Path problem remains, to our knowledge, an open problem.)

In Section 4 we consider the following more general problem:

k-Almost some Grid
Input: A graph G and a set $\mathcal{E} \subseteq \mathcal{U}$.
Parameter: A non-negative integer k.
Question: Decide if there exist some pair p, q such that $\mathcal{E}$-$\mathbf{dist}(G, H_{p,q}) \leq k$.

[2]In an abuse of notation, we indistinctly refer to FPT problem or to problem in the FPT class

The above problem can be solved applying the algorithm for k-Almost Grid for all possible values of p and q. This implies an algorithm of time complexity $O(\log n(g(k) + n^{O(1)}))$. In Section 4, we explain how to avoid this $O(\log n)$ overhead and prove that there exists also a time $O(g(k)+n^{O(1)})$ algorithm for the k-Almost some Grid problem. That way, both of our algorithms can be seen as pre-processing algorithms that reduce the size of the problem input to a function depending *only* on the parameter k and not on the main part of the problem.

A different but somehow related sets of problems, which have received plenty of attention is the following: Given a graph G and a property Π, decide what is the minimum number of edges (nodes) that must be removed in order to obtain a subgraph of G with property Π. In general all these problems are NP-hard [10, 4, 7]. Some of those problems have been studied from the parameterized point of view [5], however, all these problems have the characteristic that the property Π must be an hereditary property. We stress that the k-Almost Grid and k-Almost any Grid problems completely different nature, as the property of *containing a grid* is not hereditary.

2 Definitions and basic results

All graphs we consider are undirected, loop-less and without multiple edges. Define the *neighbourhood* of a vertex $v \in V(G)$ as $N_G(v) = \{u \in V(G) \,|\, (u, v) \in E(G)\}$, and let $\Delta(G) = \max\{|N_G(v)| \mid v \in V(G)\}$.

Denote by (p, q)-*grid*, any graph $H_{p,q}$ that is isomorphic to a graph with vertex set $\{0, \ldots, p-1\} \times \{0, \ldots, q-1\}$ and edge set

$$\{\{(i, j), (k, l)\} \mid (i, j), (k, l) \in V(H) \text{ and } |i - k| + |j - l| = 1\}.$$

In the above definition p is the column number and q is the row number of the grid. The r-*border* of $H_{s,r}$ is defined as $B^{(r)}(H_{s,r}) = ((i, 0), (i, r-1) \mid i = 1, \ldots, s-1)$. We call a vertex/edge/column) of $H_{s,r}$ *internal* if it is not in/none of its endpoints is in/none of its vertices is in $B^{(r)}(H_{s,r})$.

All the algorithms and results in this paper work for the general case where $\mathcal{E} = \mathcal{U}$. The other cases are straightforward simplifications of our results[3]. Subsequently, we will also drop the $\mathcal{E}$ part in the $\mathcal{E}$-**dist**(G, H) notation. If **dist**$(G, H) \leq k$ we will denote as $V_{\text{dead}}/E_{\text{dead}}$ the vertices/edge that should be removed and $V_{\text{add}}/E_{\text{add}}$ the vertices/edges that should be added towards transforming G to H (in order to well define $V_{\text{add}}/E_{\text{add}}$ we always fix one of the possible ways that such a transformation can be accomplished). Without loss of generality, we assume that the transformation procedure gives priority first to vertex removals then to edge removals, then to vertex insertions, and finally to edge insertions. We also use the notation $k_1 = |V_{\text{dead}}|$, $k_2 = |E_{\text{dead}}|$, $k_3 = |V_{\text{add}}|$ and $k_4 = |E_{\text{add}}|$. Finally, we observe that $k_1 + k_2 + k_3 + k_4 \leq k$.

We say that a graph G contains an r-*band* J of *width* s or, alternatively, *an* (r, s)-*band*, if the following conditions are satisfied:

a) G contains $H_{s,r}$ as induced subgraph.

b) $\nexists\{x, y\} \in E(G)$ such that $x \in V(G) - V(H_{s,r})$ and $y \in V(H_{s,r}) - B^{(r)}(J)$.

[3]Our results also hold for more general sets of operations given that they locally change the structure of the input graph (See Section 5).

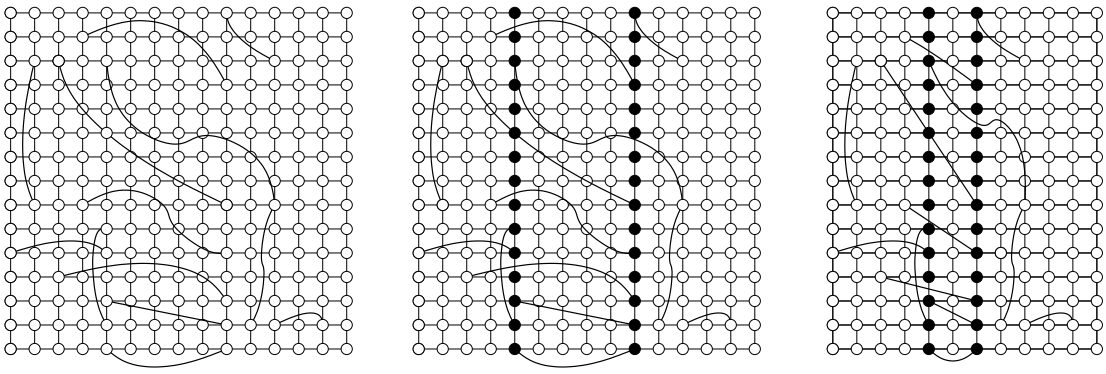

Figure 1: A graph G where $\mathbf{dist}(G, H_{15,15}) \leq 13$ with a $(6, 15)$-band J in it and the result of a 3-contraction of J in G.

If G is a graph and $H_{p,q}$ is a (p, q)-band in G where $p \geq 3$ and $q \geq 1$, for $0 \leq w \leq p-3$, a *w-contraction of $H_{p,q}$ in G* is the graph obtained by contracting in G all the edges of $H_{p,q}$ that are in the set $\{\{(i, m), (i+1, m)\} \mid 2 \leq i \leq w+1, 1 \leq m \leq q\}$. We use the notation $\mathsf{contract}(G, H_{p,q}, w)$ to denote the result of the operation just described. Notice that the routine $\mathsf{contract}(G, H_{p,q}, w)$ runs in $O(wq)$ in steps. For an example of a c-contraction, see Figure 1.

Lemma 1 *Let G be a graph containing a $(3+c, q)$-band $H_{3+c,q}$ for some $c \geq \lceil \frac{k}{q} \rceil$, and let $G' = \mathsf{contract}(G, H_{3+c,q}, w)$, where $w = c - \lceil \frac{k}{q} \rceil + 1$. Then $\mathbf{dist}(G, H_{p,q}) \leq k$ iff $\mathbf{dist}(G', H_{p-w,q}) \leq k$.*

Given a positive integer q, we call an edge of a graph *q-extremal* if $q > 1$ and both its endpoints have degree 3 or $q = 1$ and both its endpoints have degree 2. For the set of q-extremal edges of a graph G we use the notation $E^{(q)}_{\text{ext}}(G)$.

Lemma 2 *There exists an algorithm (we call it $\mathsf{find\text{-}edge\text{-}max\text{-}band}(G, q, e)$) that, given a graph G, a positive integer q and a q-extremal edge e of G, returns a maximal length $(3+c, q)$-band $H_{3+c,q}$ where $c \geq 0$ and e is also an extremal edge of H. If such a $(3+c, q)$-band does not exists, then return "NO". If the answer is "NO" the algorithm finishes in $O(q)$ steps. If the answer is a vertex set with $(3+c)q$ vertices, then the algorithm finishes in $O(qc)$ steps.*

Proof sketch: Algorithm $\mathsf{find\text{-}edge\text{-}max\text{-}band}(G, q, e)$ checks first whether e is the "upper" edge of some pair of neighboring columns by "guessing" neighbors of degree that agree to the corresponding sub-grid pattern. If this guess finishes successfully, then the algorithm ties to extend this $(q, 2)$-band to the right or to the left. If this extension is possible to some of the two directions then the output is the corresponding (q, c)-band, where $c \geq 2$. □

The proof of the following lemma is based on a contradiction argument.

Lemma 3 *If G is a graph where $\Delta(G) \leq d$ and $\mathbf{dist}(G, H_{p,q}) \leq k$ where $p \geq q$, then G contains at most $2(p-2) + 2(q-2) + 2dk$ q-extremal edges.*

The algorithm of the following lemma makes use of Lemma 3 and Algorithm find-edge-max-band(G, q, e) in Lemma 2.

Lemma 4 *There exists an algorithm (we call it* find-max-band(G, q, c)*) that, given a graph G, and two positive integers q, c returns a maximal length $(3 + c', q)$-band $H_{3+c,q}$ where $c' \geq c$. If such a $(3+c', q)$-band does not exists, then* find-max-band(G, q, c) *returns "*NO*". In case of negative answer* find-max-band(G, q, c) *runs in $O(q|E^q_{\text{ext}}(G)|)$ steps. In case of a positive answer* find-max-band(G, q, c) *runs in $O(qc|E^q_{\text{ext}}|)$ steps.*

Suppose that G can be transformed to H after a sequence of edit operations. We call a vertex $v \in G$ *safe* if is not removed by a vertex removal operation in this sequence. If a vertex of G is not safe then we call it *dead*. We also say that an safe vertex v is *dirty* if some of the edit operations alters the set of edges incident to v in G (i.e. either adds or removes some edge incident to v). If a safe vertex is not dirty, then it is *clean*. A vertex of H is *new* if it was introduced during some vertex insertion operation. Notice that if $\mathbf{dist}(G, H) \leq k$, then there are at most k new vertices in H and at most k dead vertices in G, which implies the following lemma,

Lemma 5 *Let G and H be two graphs where $\mathbf{dist}(G, H) \leq k$ and such that the transformation of G to H involves k_1 vertex removals and k_2 vertex additions. Then, $|V(H)| - k_2 \leq |V(G)| \leq |V(H)| + k_1$.*

A straightforward counting argument gives the following lemma.

Lemma 6 *Let G be a graph such that $\mathbf{dist}(G, H_{p,q}) \leq k$. Supose also that G does not contain any $(3 + c, q)$-band where $q \cdot c > k$ and $H_{p,q}$ contains $\leq d$ dirty vertices. Then $p \leq (\lceil \frac{k}{q} \rceil + 3)(d + k + 1)$.*

Lemma 7 *Let G be a graph with a vertex v of degree more than $k + 4$ in G and let $H_{p,q}$ be a grid. Then any transformation of G to $H_{p,q}$ should involve the operation of the removal of v. In particular, $\mathbf{dist}(G, H_{p,q}) \leq k$ iff $\mathbf{dist}(G[V(G) - v], H) \leq k - 1$.*

Proof sketch: Notice that more than k of the edges of v should be removed towards transforming G to $H_{p,q}$ while it is possible to apply at most k edit operations on G. Therefore, any sequence on $\mathcal{E}$ able to transform G to $H_{p,q}$, should include the removal of v. □

Notice that if v is a dirty vertex of H then v is either adjacent to a vertex $v' \in V_{\text{dead}}$ or is incident to an edge in $E_{\text{add}} \cup E_{\text{dead}}$. As $v' \in V_{\text{dead}}$ can be adjacent to at most $\Delta(G)$ dirty vertices of H and an edge in $E_{\text{add}} \cup E_{\text{dead}}$ can be incident to at most two dirty vertices of H, then each edit operation creates at most $\max\{2, \Delta(G)\}$ dirty vertices. Therefore,

Lemma 8 *Let G be a graph and, let H be some grid. If $\mathbf{dist}(G, H) \leq k$, then H contains at most $\max\{\Delta(G), 2\} \cdot k$ dirty vertices.*

3 Looking for a given grid $H_{p,q}$

In this section, we show that the k-Almost grid problem is in FPT. For this, we first combine Lemmata 5, 8, and 6 and prove the following bound f to the size of the kernel.

Lemma 9 *Let G be a graph where $\mathbf{dist}(G, H_{p,q}) \leq k$. Suppose also that $\Delta(G) \leq b$ and that G does not contain any $(3+c,q)$-band where $c \geq \lceil k/q \rceil$ or any $(3+c,p)$-band where $c \geq \lceil k/p \rceil$. Then $|V(G)| \leq f(k,b,p,q)$ where*

$$f(k,b,p,q) = \begin{cases} k + k^2 & \text{if } p \leq k \text{ and } q \leq k, \\ k + 5k(b \cdot k + k + 1) & \text{if } \min\{p,q\} \leq k < \max\{p,q\}, \\ k + 16(b \cdot k + k + 1)^2 & \text{if } p > k \text{ and } q > k. \end{cases}$$

Next, we present the algorithm to construct the kernel.

Kernel-for-Check-Grid(G,p,q,k)
Input: A graph G, two positive integers p,q, and a non-negative integer k
Output: Either returns "NO", meaning that that $\mathbf{dist}(G, H_{p,q}) > k$, or returns a graph G' and a triple p', q', k' such that $\mathbf{dist}(G, H_{p,q}) \leq k$ iff $\mathbf{dist}(G', H_{p',q'}) \leq k'$.

0. Set $k' = k$, $G' = G$, $p' = p$ and $q' = q$.
1. As long as G' has a vertex of degree $\geq k' + 4$ remove it from G' and set $k' \leftarrow k' - 1$. If after the end of this proccess $k' < 0$ then **return** "NO".
2. Apply any of the following procedures as long as this is possible:
- As long as find-max-band$(G', q', \lceil \frac{k'}{q'} \rceil) = H_{3+c,q'}$,
  then set $w = c - \lceil \frac{k'}{q'} \rceil + 1$, $G' = \mathsf{contract}(G', H_{3+c,q'}, w)$ and set $p' \leftarrow p' - w$.
- As long as find-max-band$(G', p', \lceil \frac{k'}{p'} \rceil) = H_{3+c,p'}$,
  then set $w = c - \lceil \frac{k'}{p'} \rceil + 1$, $G' = \mathsf{contract}(G', H_{3+c,p'}, w)$ and set $q' \leftarrow q' - w$.

3 If $|V(G')| > f(k', k+4, p', q')$ then **return** "NO".
4. return G', p', q', k'.

Theorem 1 *Algorithm* Kernel-for-Check-Grid(G,p,q,k) *is correct and if it outputs the graph G' then $|V(G')| \leq f(k, k+4, p, q)$. Moreover, it runs in $O(pq(p+q+k^2))$ steps.*

Proof: Step **1** is justified by Lemma 7. Notice that before the algorithm enters Step **2**, $\Delta(G) \leq k+4$. Step **2** creates equivalent instances of the problem because of Lemma 1. From Lemmata 3 and 4, each time a contraction of step **2** is applied, the detection and contraction of the corresponding q-band (p-band) requires $O(q'c(p'+q'+k'^2))$ ($O(p'c(p'+q'+k'^2))$) steps. As the the sum of the lengths of the bands cannot exceed $q(p)$ we obtain that step **2** requires, in total, $O(pq(p+q+k'^2))$ steps. Moreover, before the check of Step **3**, all the conditions of Lemma 9 are satisfied. Therefore, if the algorithm returns G', p', q', k', then $|V(G')| \leq f(k', k+4, p', q') \leq f(k, k+4, p, q)$. □

Theorem 2 *There exists an algorithm (we call it* Check-Grid(G,p,q,k)*) that given a graph G, two positive integers p,q, and a non-negative integer k either returns "NO", meaning that that $\mathbf{dist}(G, H_{p,q}) > k$, or returns a sequence of at most k operations that transforms G to $H_{p,q}$.* Check-Grid(G,p,q,k) *runs in $O(16^k \cdot (2n+3k)^{2k})$ steps.*

We stress that Algorithm Check-Grid(G, p, q, k) can be replaced by a faster one when we do not consider edge additions in the set of edit operations.
Now comes the main algorithm of this section which implements the kernelization.

Almost-grid(G, p, q, k)
Input: A graph G, two positive integers p, q, and a non-negative integer k
Output: Either returns "NO", meaning that that $\mathbf{dist}(G, H_{p,q}) > k$, or returns a sequence of at most k operations that transforms G to $H_{p,q}$.

1. If Kernel-for-Check-Grid$(G, p, q, k) = (G', p', q', k')$ then return Check-Grid(G, p, q, k)
2. return "NO"

We conclude with the following.

Theorem 3 *Algorithm* Almost-grid(G, p, q, k) *solves the* k-Almost-Grid *problem in* $2^{O(k \log k)} + O(n^3)$ *steps.*

4 Looking for any grid

To solve the k-Almost some Grid problem it is enough to apply Check-Grid(G, p, q, k) for all $(p, q) \in \mathbf{A}(n, k) = \bigcup_{n-k \leq i \leq n+k} \{(p, q) \mid p \cdot q = i\}$. As $|\mathbf{A}(n, k)| \leq 2k \log(k + n)$, k-Almost some Grid can be solved after $O(k \log(n+k))$ calls of Almost-grid(G, p, q, k) which gives a running time of $O(\log n)(2^{O(k \log k)}) + O(k^2 n^3 \log n)$ steps. We call this algorithm check-all-cases(G, k). We stress that there are cases where $|\mathbf{A}(n, k)| \geq \frac{1}{2} \log n$ and therefore, that way, we may not avoid the "$\log n$"-overhead. In this section, we will give an alternative approach that gives running times of the same type as the case of k-Almost Grid.

We call *r-band collection* $\mathcal{C}$ of a graph G a collection of $|\mathcal{C}|$ r-bands in G, with widths $s_1, \ldots, s_{|\mathcal{C}|}$, where no pair of them have common interior vertices. The *width* of such a collection is $\sum_{i=1,\ldots,|\mathcal{C}|}(s_i - 2)$.

The following algorithm uses the Algorithm find-edge-max-band in order to identify the possible components of an r-band collection. Notice that the algorithm has two levels of "guessing" edges: The first level guesses a starting r-extremal edge, if the edge belongs to some $(3 + c, q)$-band, the algorithm extends the collection by guessing r-extremal edges of new components, excluding r-extremal edges that already belong to bands included in the collection.

The algorithm of the following lemma makes use of Algorithm find-edge-max-band(G, q, e).

Lemma 10 *There exists an algorithm (we call it* find-max-band-collection(G, r, w)*) that given a graph G and two positive integers r and w returns "YES" if G contains an r-band collection of width $\geq w$; otherwise, returns "NO". The algorithm* find-max-band-collection(G, r, w) *runs in $O(rw|E_{\text{ext}}(G)| + n)$ steps.*

The following lemma identifies a bound on the size of a graph without a q-band collection of sufficient big width.

Lemma 11 *Let G be a graph where $\Delta(G) \leq b$ and let $H_{p,q}$ be a grid where* $\mathbf{dist}(G,H) \leq k$. *Then if G does not contain any q-band collection of width $> k$ we have that $|V(H)| \leq (b \cdot k + 2)^2$.*

Proof: $H_{p,q}$ contains a single element q-band collection $\mathcal{C}_H$ of width $p-2$. We apply on G the at most k operations that transformed it to H. If we remove an edge in Y_{in}, in the worst case, its endpoints may belong in neighboring columns of $\mathcal{C}_H$ and this can remove at most 2 units from the width of $\mathcal{C}_H$. If we add back an edge in Y_{out}, in the worst case, its endpoints may belong in different columns of $\mathcal{C}_H$ and this can remove at most 4 units from the width of $\mathcal{C}_H$. If finally we add back a vertex in X_{out}, in the worst case, all its neighbours will belong to different columns and this may reduce the width of $\mathcal{C}_H$ by $\leq \Delta(G) \leq b$. So G contains a q band collection of width at least $p-2-e_{\text{in}}-e_{\text{out}}-b \cdot x_{\text{out}} \geq p-2-b \cdot k$, which implies that $p \leq b \cdot k + 2$. Working symmetrically, we conclude that $q \leq b \cdot k + 2$ and the result follows. □

The following gives some conditions on when it is possible to make an estimation of the dimensions of the grid.

Lemma 12 *Let G be a graph where $\Delta(G) \leq b$ and let H be a grid where* $\mathbf{dist}(G,H) \leq k$. *Then, if G contains an r-band collection of width $> 3k$ then $H = H_{s,r}$ for some $s \in [\frac{n-k}{r}, \frac{n+k}{r}]$. Moreover such an r-band collection can exist for at most two distinct values of r and s will be unique when $r > 2k$.*

Proof: Let $\mathcal{C}$ be the interior columns in such a collection of bands. We will apply the edit operations that transform G to H. The removal of a vertex in X_{out} can reduce the width of $\mathcal{C}$ by at most 3. The removal of an edge in Y_{out} can reduce the width of $\mathcal{C}$ by at most 3. The addition of an edge in Y_{in} can reduce the width of $\mathcal{C}$ by at most 2. Therefore, in the worst case, H still contains a collection of bands of width $3k+1-3k > 1$. So, there exist at least one r-band of width at least 1 in H and this means that $H = H_{s,r}$ for some $s \geq 1$.

Suppose now that there are three integers $r_1 < r_2 < r_3$ and, that G contains an r_i-band collection of width $> 3k$, for $i = 1,2,3$. Let also $I_i = 1,2,3$ be the interior columns of each of these band collections. Clearly, for any $i < j, I_i \cap I_j \neq \emptyset$. According to the previous analysis, after the edit operations, three, different, size r_i-bands of width al least 1 will survive which is impossible to exist in a rectangular grid.

Notice that $s \cdot r = n - x_{\text{out}} + x_{\text{in}}$. Let $\delta = |x_{\text{out}} - x_{\text{in}}|$. We obtain that $s \cdot r \in [n-\delta, n+\delta]$, which implies $s \in [\frac{n-k}{r}, \frac{n+k}{r}]$, as $\delta \leq k$. If the interval $[\frac{n-k}{r}, \frac{n+k}{r}]$ contains two integers, then $\frac{n+k}{r} - \frac{n-k}{r} \geq 1$, which implies that $r \leq 2k$. □

We now proceed with the main algorithm of this Section.

Check-some-Grid(G,k)

Input: A graph $G$and a non-negative integer k
Output: The answer to the k-Almost-some-Grid problem with instance G and parameter k.

1. As long as G has a vertex of degree $\geq k+4$ remove it from G and set $k \leftarrow k-1$. If after the end of this proccess $k<0$ then **return** "NO".
2. $R \leftarrow \emptyset$
3. for $i=1,\ldots,n$, if find-max-band-collection$(G,i,3k+1)$ ="YES", then set $R \leftarrow R \cup \{i\}$
4. **If** $|R|=0$ then **if** $V(G) > k+((k+4)k+2)^2$ then **return** "NO", **otherwise, goto** Step **7**
5. **For** any $r \in R$,
If $r > 2k$, **then**
begin
If $\mathbb{N} \cap [\frac{n-k}{r}, \frac{n+k}{r}] = \emptyset$, **then return** "NO", **otherwise**,
return Almost-Grid(G,r,s,k) where $\{s\} = \mathbb{N} \cap [\frac{n-k}{r}, \frac{n+k}{r}]$.
end
otherwise, **for** $(i,j) \in \bigcup_{n-k \leq i \leq n+k} \{(p,q) \mid p \leq 2k, p \cdot q \leq i\}$,
begin
If Almost-Grid(G,i,j,k) = YES, **then return** Almost-Grid(G,i,j,k).
end
6. **Return** "NO".
7. check-all-cases(G,k).

Theorem 4 *Algorithm* Check-some-Grid(G,k) *is correct and the* k-Almost-any-Grid *problem is in* FPT *and can be solved in time* $2^{O(k \log k)} + O(k^2 \cdot n^3)$.

Proof: Step **1** is justified by Lemma 7 and we may assume that before the algorithm enters Step **2**, $\Delta(G) \leq k+4$. Steps **2**–**4** are based on Lemma 11. Step **3** involves $O(n)$ calls of find-max-band-collection$(G,i,3k+1)$ which requires $O(kn^2)$ steps because of Lemma 10. Theorefore, Step **3** requires $O(kn^3)$ steps. Finally the loop in Step **5** is correct because of Lemma 12. Note that the first loop in step **5** is applied at most 2 times and that $|\bigcup_{n-k \leq i \leq n+k} \{(p,q) \mid p \leq 2k, p \cdot q \leq i\}| = O(k^2)$. Therefore, step **5** runs in $2^{O(k \log k)} + O(k^2 \cdot n^3)$ steps. As we noticed in the begining of this section, check-all-cases(G,k) requires $O(\log n')(2^{O(k \log k)} + O(\log n' \cdot n'^3 \cdot k^2))$ steps where n' is the number of vertices of G in step **7**. As $n' \leq O(k^4)$, this means that step **7** requires $2^{O(k \log k)}$ steps. □

5 Extensions

The ideas of the kernelization algorithm Kernel-for-Check-Grid can be also used for more general "cleanning" problems. The directions in which our results can be extended are the following:

Other patterns. Algorithm Find-edge-max-band intents to find "regions of regularity" (r-bands) in the input graph that can be safely contracted. This is used by the kernelization Algorithm Kernel-for-Check-Grid in order to output an equivalent instance of the

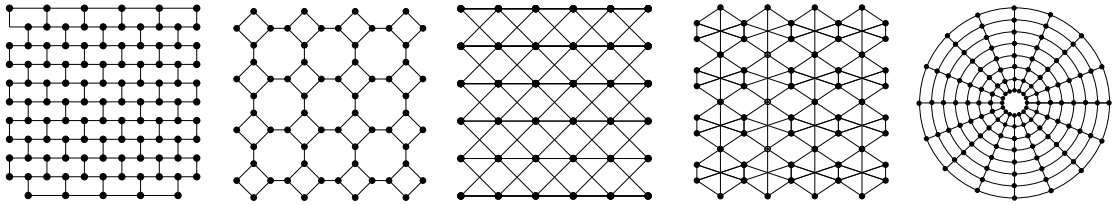

Figure 2: Patterns of graphs for which the edit distance problem is in FPT.

problem (of small size) containing the "essential" part of the "dirtinesss" of the input graph. While Algorithm Find-edge-max-band works for grids, it can be adapted for other "regular" patterns as well. In Figure 2 we depict some examples of such patterns. In each of them one has to define the adequate notion of regularity pattern and design the analogous of Algorithm Find-edge-max-band for its detection. It is enough to adapt Lemma 1 in order to justify the compression of this regular patterns and to modify Lemma 9 to prove that when no further compression is possible. Therefore, the graph has a size that depends only on the parameter k (the proofs are based on the same arguments as in the case of grid).

Other operations. Observe that the set $\mathcal{U} = \{\mathsf{e\text{-}out}, \mathsf{e\text{-}in}, \mathsf{v\text{-}out}, \mathsf{v\text{-}in}\}$ is not the most general set of edit operations, where the arguments of our proofs can be applied. In fact, all the proofs in this paper exploit the "locality" of these operations. All our results hold also for any set of operations, where each operation is equivalent to the detection of a constant size subgraph H, together with the application of a constant number of edge removals, edge additions, edge contractions, vertex removals, vertex additions (with the addition of an arbitrary number of incident edges), or inverse contractions on H. Some examples of such transformations are depicted in Figure 3.

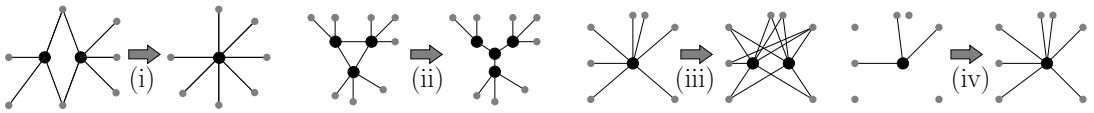

Figure 3: (i) identification of two vertices, (ii) Δ-Y transformation, (iii) vertex duplication, and (iv) addition of any number of incident edges.

References

[1] N. Alon, R. Yuster and U. Zwick. Color-Coding. *Journal of the ACM*, 42(4), 844–856, 1995.

[2] R. Downey and M. Fellows. *Parameterized complexity*. Springer-Verlag, 1999.

[3] H. Kaplan, R. Shamir and R. Tarjan. Tractability of parameterized completion problems on chordal, strongly chordal and proper interval graphs. *SIAM Journal of Computing*, 28, 1906–1922, 1999.

[4] J. Lewis and M. Yannakakis. The node-deletion problem for hereditary properties is NP-Complete. *Journal Comput. and Systems Sci.* 20(2), 219–230, 1980.

[5] Leizhen Cai. Fixed-parameter tractability of graph modification problems for hereditary properties. *Information Processing Letters*, 58(4)171–176, 1996

[6] B. Monien. How to find paths efficiently. *Annals of Discrete Mathematics*, 25, 239–254, 1985.

[7] A. Natanzon, R. Shamir and R. Sharan. Complexity classification of some edge modification problems. *Discrete Applied Mathematics*, 113(1), 109–128, 2001.

[8] Rao, S. D. and Kurdahi, F. J. On clustering for maximal regularity extraction. *IEEE transactions on Computer-aided Design of Integrated Circuits and Systems*, 12(8), 1198–1208, 1993.

[9] A. Robles-Kelly, E. Hankok. Graph edit-distance from spectral seriation. *IEEE Transactions on Pattern analysis and Machine Intelligence*, 27, 365–378, 2005.

[10] M. Yannakakis. Node and edge deletion NP-complete problems. *ACM Symposium on Theory of Computing (STOC)*, 253–264, 1978.

[11] K. Zhang and D. Sasha. Simple fast algorithms for the editing distance between trees and related problems. *SIAM Journal on Computing* 18, 1245–1262, 1989.