Fachbereich Mathematik & Informatik
AG Algorithmik

Sommersemester 2020

Bachelorarbeit zum Thema

On Strong Triadic Closure with Edge Insertion

Betreuer: Prof. Dr. Christian Komusiewicz
M.Sc. Niels Grüttemeier
M.Sc. Nils Morawietz

eingereicht von:	Béla Neuendorf
Matrikelnummer:	2827931
Datum der Abgabe:	28. Mai 2020

Abstract

Strong Triadic Closure (STC) was introduced by Sintos and Tsaparas [24] to characterize the strength of relationships in social networks. The strong triadic closure property states that if two individuals in a network each have a strong relationship with a third individual, then they have to be connected in the network as well. In the computational problem of STC, the goal is to label the edges in E of a graph $G = (V, E)$ either strong or weak, such that for every induced P_3 in G, which is a path consisting of three vertices, at least one edge is labeled weak. Furthermore, it is required that the number of weak edges is at most k. Strong Triadic Closure with Edge Insertion (STC+) is an extension of STC. Here, the goal is to find a set of additional weak edges and a labeling such that for every induced P_3 in G at least one edge is labeled weak or the missing edge is inserted and the number of weak edges, including the inserted edges, is at most k. In this work, we study the parameterized and classical complexity of STC+ and prove that it is NP-hard. Furthermore, we provide a $4k$-vertex kernel and an FPT-algorithm for STC+. Cluster Editing is a problem that is stated in terms of induced P_3s as well. To obtain more information about its relation to STC+, we compare Subgraph-free Editing, the generalization of Cluster Editing, with the generalization of STC+, that is Strong Subgraph Closure with Edge Insertion. Given the graphs G and H, in both problems the goal is to "destroy" induced H-graphs in G by at most k modifications, in Subgraph-free Editing by deleting and adding edges, in Strong Subgraph Closure with Edge Insertion by giving an additional edge set and a labeling. We compare the two problems for graphs H on three or four vertices.

Zusammenfassung

Strong Triadic Closure (STC) wurde von Sintos and Tsaparas [24] vorgestellt, um die Stärke von Beziehungen in sozialen Netzwerken zu charakterisieren. Die Strong Triadic Closure Eigenschaft ist, dass wenn zwei Individuen in einem Netzwerk beide eine starke Beziehung zu einem dritten Individuum haben, dann müssen sie auch in dem Netzwerk verbunden sein. Im Berechnungsproblem STC ist das Ziel, die Kanten in E eines Graphen $G = (V, E)$ entweder stark oder schwach zu markieren, so dass von jedem induzieren P_3 in G, dies ist ein Pfad bestehend aus drei Knoten, mindestens eine Kante schwach markiert ist. Außerdem darf die Anzahl der schwach markierten Kanten höchstens k betragen. Strong Triadic Closure with Edge Insertion (STC+) ist eine Erweiterung von STC mit dem Ziel, eine Menge von zusätzlichen schwach markierten Kanten sowie einer Markierung der Kanten in E zu finden, so das von jedem induzieren P_3 mindestens eine Kante schwach markiert ist, oder die die fehlende Kante eingefügt wurde. Zusätzlich darf die Anzahl der schwach markierten Kanten, inklusive der eingefügten Kanten, höchstens k betragen. Wir untersuchen die parametrisierte und klassische Komplexität von STC+

und beweisen, dass es NP-hart ist. Weiterhin erarbeiten wir einen $4k$-Knoten Kern sowie einen FPT-Algorithmus für STC+. CLUSTER EDITING ist ein Problem, das auch auf induzierten P_3s formuliert werden kann. Um mehr über dessen Beziehung zu STC+ zu erfahren, vergleichen wir dessen Verallgemeinerung, SUBGRAPH-FREE EDITING, mit STRONG SUBGRAPH CLOSURE WITH EDGE INSERTION, der Verallgemeinerung von STC+. In beiden Problem ist das Ziel für zwei Graphen G und H, induzierte H-Graphen in G zu "zerstören". In SUBGRAPH-FREE EDITING soll dies mit höchstens k Kantenlöschungen und -einfügungen gemacht werden, in STRONG SUBGRAPH CLOSURE WITH EDGE INSERTION mit einer Menge zusätzlichen schwach markierten Kanten und einer Markierung der Kanten, so dass es höchstens k schwach markierte Kanten gibt. Wir vergleichen die beiden Probleme für Graphen H bestehend aus drei und vier Knoten.

Contents

1 Introduction

Online social networks have massively grown since their introduction. That has revolutionized how social scientists study the structure of human relationships [2]. As individuals transfer their social relations and interactions online, the internet evolved from a network of documents and data to additionally a network of people. Social structures that have been previously invisible are now recorded at large scale in online social networks like Facebook [25].

Understanding the strength and nature of online relationships is key to anyone that is interested in making use of these data [8, 9, 14, 29] and can benefit sociology, advertisement and friendship suggestion algorithms, just to state some examples [24].

Social networks are mathematically represented by a graph that consists of vertices representing individuals and edges representing the relationships between these individuals. These relationships are not all of the same nature: Some people are close friends, others are just distant relatives or forgotten high school classmates. To distinguish between different kind of ties in a network graph, labeling the edges either *strong* or *weak* is one option. The principle of *strong triadic closure* is an important approach for this problem [24], which goes back to Granovetter's sociological work [11]. Informally, the *strong triadic closure property* is that if two persons both have a strong relationship with a third person, they have to know each other at least weakly. Sociologists first used the strong triadic closure principle when analyzing human friendship, choices and came to the conclusion that friends of friends tend to become friends themselves [16, 28]. The goal of Strong Triadic Closure (STC), the computational problem associated to the strong triadic closure principle, is to label the edges of a graph such that the strong triadic closure property is satisfied and number of strong edges is maximized (equivalently the number of weak edges is minimized). A labeling for a graph G that satisfies the STC property is called an STC-labeling.

Consider a group of friends in the real world, which also connected online in a social network. In the graph of the social network, they form a clique, that is each pair of vertices are adjacent. We could label all edges between these vertices strong and satisfy the strong triadic closure property in the clique. But if one edge in the clique is missing, for example because two persons did not connect already in the social network, we could not label all other edges strong. In fact, if the clique is of size n, we would need to label $n-2$ edges weak. Inserting the missing edge would benefit the minimization of weak edges. Therefore an extending computational problem of STC was introduced: Strong Triadic Closure with Edge Insertion (STC+) [24]. In this problem, given a graph G and an integer $k \geq 0$, the question is whether there exists a set of additional edges E' and an STC-labeling of the edges of G and E' such that all edges in E' are labeled weak and the number of weak edges is at most k.

Both STC and STC+ can be specified in terms of induced subgraphs instead of "conflict triples": In STC, for every induced P_3, which is a path over three vertices, at least one of the edges has to be labeled weak. In STC+, for every induced P_3 at least one edge has to be labeled weak or the missing edge has to be inserted instead, turning the P_3 into a clique of size three. The following two computational problems, Cluster Deletion (CD) and Cluster Editing (CE), are working on induced P_3s as well. In Cluster Deletion, the goal is to turn a graph into a cluster graph, that is a graph consisting of one or more disjoint cliques, by deleting a minimum number of edges. A graph G is a cluster graph if and only if there is no induced P_3 in G. It has been noticed that STC is closely related to Cluster Deletion [18, 12]. In Cluster Editing, the goal is to turn a graph into cluster graph with a minimum number of modifications, not only by edge deletions, but by insertions as well.

In this work, we study the parameterized and classical computational complexity of STC+, provide an approximation algorithm, a fixed-parameter algorithm, a linear-vertex kernel for the parameter k and an exponential-vertex kernel for the parameter $\ell := |E| - k$. During our work the question arose, whether STC+ and CE correspond. That is, for a graph G and an integer k, there is an additional set of edges E' and an STC-labeling of the edges of G and E', such that all edges in E' are labeled weak and the number of weak edges is at most k if and only if G can be turned into a cluster graph by at most k edge modifications. To that end, we analyse the relation of the generalizations of the two problems, Subgraph-free Editing and Strong Subgraph Closure with Edge Insertion for other induced H-graphs beside P_3s.

1.1 Known Results and Related Work

STC is NP-hard [24], even when restricted to graphs with maximum degree four [18] or to split graphs [19]. In contrast, it is solvable in polynomial time when the input graph is bipartite [24], subcubic [18], a proper interval graph [19] or a cograph [18]. STC can be solved in $\mathcal{O}(1.28^k + nm)$ time and admits a $4k$-vertex kernel [12, 24]. Furthermore, there is a fixed-parameter algorithm for STC parameterized by $\ell := |E| - k$, but the problem does not admit a polynomial kernel with respect to ℓ, unless NP $\subseteq$ coNP/poly [12, 10].

Cluster Deletion is NP-hard [23], even when restricted to graphs with maximum degree four [17] or to $(2K_2, 3K_1)$-free graphs [7]. Cluster Editing is NP-hard even on graphs with maximum degree six [17], can be solved in $\mathcal{O}(1.62^k + n + m)$ time [1] and admits a $2k$-vertex kernel [4].

There is a polynomial-time reduction from STC+ to 3-Hitting Set leading to an $\mathcal{O}(\log n)$-approximation algorithm for STC+ [24].

Golovach et al. [10] introduced the Strong F-Closure problem, whitch is a generalization of STC. For a graph F, the goal of Strong F-Closure is to find a labeling $L = (S_L, W_L)$ of a graph $G = (V, E)$ such that no induced subgraph of G is isomorphic to F and consists only of strong edges and the number of weak edges is at most k. They show that there is a polynomial kernel for Strong F-Closure parameterized by k but does not admit a polynomial kernel for the parameter $\ell := |E| - k$.

During their work about STC, Sintos and Tsaparas introduce Multi-STC as a generalization of STC, where edges of a graph can be labeled to weak or to c different strong types and the question is whether there is a Multi-STC-labeling for the graph such that there is no P_3 labeled the same kind of strong, with at most k edges labeled weak. It is harder than STC since for all $c \geq 3$ it is NP-hard even with $k = 0$ [3].

1.2 Our Results

First, we show that STC+, STC, CD and CE are corresponding on graphs that are *diamond*- and C_4-free. The problems are corresponding on *diamond*- and C_4-free graphs, if for every *diamond*- and C_4-free graph $G = (V, E)$ and every integer $k \geq 0$ there is a set of additional edges E' and an STC-labeling of the edges in E and E' such that all edges in E' are labeled weak and there are at most k weak edges if and only if there is an STC-labeling of the edges in E with at most k weak edges if and only if G can be turned into a cluster graph by at most k edge deletions if and only if G can be turned into a cluster graph by at most k edge modifications. With that correspondence, we can prove the NP-hardness of STC+ even when restricted to *diamond*- and C_4-free graphs. Secondly, we give a 3-approximation algorithm for the problem.

Then, we provide the first linear-vertex kernel for STC+ parameterized by k. More precisely, we show that any instance of STC+ can be reduced to an equivalent instance with at most $4k$ vertices in $\mathcal{O}(n^2m)$ time. Furthermore, we study the parameterized complexity of STC+ for the parameter ℓ. We show that for a graph $G = (V, E)$ and an integer $k \geq 0$, searching for a set of additional weak edges E' and an STC-labeling of the edges in E and E' such that all edges in E' are labeled weak and the number of weak edges is at most k is equivalent to searching for such an additional edge set E' and an STC-labeling of the edges in E and E' such that all edges in E' are labeled weak and the number of strong edges is at least ℓ. We provide an $\mathcal{O}(\ell \cdot 2^{\ell})$-vertex kernel for STC+.

Moreover we propose a branching algorithm with branching size $\mathcal{O}(2.57^k)$ and show the existence of an FPT-algorithm running in $2.076^k \cdot n^{\mathcal{O}(1)}$ time. This can be achieved by using a polynomial-time reduction from STC+ to 3-Hitting Set and using the algorithm of Wahlstöm solving 3-Hitting

Set [27].

Finally, we study the correspondence of Strong Subgraph Closure with Edge Insertion and Subgraph-free Editing on other induced H-graphs beside P_3s to obtain more information for the question, whether STC+ and CE correspond or not.

2 Preliminaries

In this section, the basic notations and definitions for graphs that we use during this work are introduced. Furthermore, we give the formal definitions of the computational problems that we consider and a short summary of the aspects of parameterized algorithms that are fundamental for our work. In the end we provide an overview about approximation algorithms.

2.1 Graph Theory

All graphs considered in this work are simple and undirected. A graph consists of vertices and edges, each connecting exactly two vertices. We let $G = (V, E)$ denote a graph where $V = \{v_1, \ldots, v_n\}$ is the set of vertices and $E = \{e_1, \ldots, e_m\}$ is the set of edges. In an undirected graph an edge $e \in E$ connecting the two vertices $u, v \in V$ is a set consisting of those vertices, that is, $e = \{u, v\}$. The number of vertices and edges is denoted by $n := |V|$ and $m := |E|$. For a set of vertices V, $\binom{V}{2} := \{\{u, v\} : u, v \in V\}$ denotes the set of all possible edges consisting of those vertices.

For every vertex $v \in V$, the *open neighborhood* in the graph G, denoted by $N_G(v)$, is the set of vertices which are adjacent to v. Formally it is defined as $N_G(v) := \{u \in V : \{u, v\} \in E\}$. The *degree* of a vertex v is the size of its open neighborhood and is defined as $d(v) := |N_G(v)|$. The *closed neighborhood* of v in G, denoted by $N_G[v]$, is the open neighborhood of v plus v. Hence $N_G[v] := N_G(v) \cup \{v\}$. The set of vertices in G which have a distance of exactly two to v is denoted by $N^2_G(v)$ and is defined as $N^2_G(v) := \bigcup_{u \in N_G(v)} N_G(u) \setminus N_G[v]$.

For a subset $S \subseteq V$ of the vertices of G, by $N_G(S)$ we denote the open neighborhood of S which is defined as $N_G(S) := \bigcup_{v \in S} N_G(v) \setminus S$. The closed neighborhood of S is defined as $N_G[S] := N_G(S) \cup S$. The set of vertices in G which have a distance of exactly two to some vertex in S and at least two to every vertex of S is denoted by $N^2_G(S)$ and defined as $N^2_G(S) := \bigcup_{u \in N_G(S)} N_G(u) \setminus N_G[S]$.

For any two vertex sets $V_1, V_2 \subseteq V$, let $E_G(V_1, V_2) := \{\{v_1, v_2\} \in E : v_1 \in V_1 \land v_2 \in V_2\}$ denote the set of edges which connect vertices of V_1 and V_2 in G. Furthermore, for any vertex set $V' \subseteq V$ we denote the set of edges between vertices in V' by $E_G(V') := E_G(V', V')$. In all of the above notations, the subscript G may be omitted if the referred graph is clear from the context.

For a graph $G = (V, E)$, we denote the set of vertices of G by $V(G)$ and the set of edges by $E(G)$. A graph H is a *subgraph* of G if $V(H) \subseteq V(G)$ and $E(H) \subseteq E(G)$. For a subset $S \subseteq V(G)$ of the vertices in G the *induced subgraph* of G by S is denoted by $G[S]$ and defined as $G[S] := (S, E_G(S))$.

For any vertex set $V' \subseteq V$ let $G - V'$ denote $G[V \setminus V']$ which is the graph obtained by removing all vertices of V' and their incident edges from G. For an edge set $E' \subseteq E$ let $G - E'$ denote the graph $G' = (V, E \setminus E')$ that results from removing the edges E' from G. Furthermore, let $G + E'$ denote the graph $G' = (V, E \cup E')$ that results from adding the edges E' to the graph.

A graph $G = (V, E)$ is *complete* if for every pair of vertices $v_i, v_j \in V$, $i \neq j$, it holds that $\{v_i, v_j\} \in E$, that is, all vertices in V are pairwise adjacent. A *clique* in G is a subset of vertices $K \subseteq V$ such that $G[K]$ is complete. G is a *cluster graph* if every vertex $v \in V$ lies in exactly one maximum clique. Equivalently, G is a cluster graph if there exists no induced P_3 in G. A vertex set $I \subseteq V$ is an *independent set* in G if for each vertices $u, v \in I$ it holds that $\{u, v\} \notin E$, that is, no two vertices of I are adjacent.

For a vertex set $X \subseteq V$ in a graph $G = (V, E)$, having a certain property, for example being a clique or an independent set, we say that X is *maximal* under that property if there is no $v \in V \setminus X$ such that $X \cup \{v\}$ has the property as well. Furthermore, we say that X is a *maximum* clique (independent set) in G if there is no other clique (independent set) $Y \subseteq V$ in G with $|Y| > |X|$.

A 2-*partition* of a set (of vertices or edges) X, is a grouping of the elements of X into two subsets P_1, P_2 of X such that each element of X is included in exactly one of these subsets. A *cut* $C = (V_1, V_2)$ of a graph $G = (V, E)$ is a 2-partition of the vertices in V. The *cut-set* $E_C := E_G(V_1, V_2)$ is the set of edges having each one endpoint in V_1 and the other in V_2. A *matching* $M \subseteq E$ in a graph $G = (V, E)$ is a set of pairwise disjoint edges.

A graph $G = (V_G, E_G)$ is *isomorphic* to a graph $H = (V_H, E_H)$ if and only if there is a bijective function $f : V_G \to V_H$ such that for any two vertices $u, v \in V_G$ there exists the edge $\{u, v\} \in E_G$ if and only if there exists the edge $\{f(u), f(v)\} \in E_H$. We say that G is H-*free* if it does not contain an induced subgraph that is isomorphic to H. The *complement graph* $\overline{G} = (V, E^{-1})$ of G is obtained by inverting the edges: for every non-existing edge in G, there is an edge in $\overline{G}$ and vice versa. Thus, the edges of the complement graph are defined as $E^{-1} := \binom{V}{2} \setminus E$.

2.2 Problem Definitions

Definition 1. *A labeling $L = (S_L, W_L)$ of an undirected graph $G = (V, E)$ is a 2-partition of the edge set E. The edges in S_L are called strong and the edges in W_L are called weak.*

Definition 2. A labeling $L = (S_L, W_L)$ is an STC-*labeling* of a graph G if there exists no pair of strong edges $\{u, v\} \in S_L$ and $\{v, w\} \in S_L$ such that $\{u, w\} \notin E$.

For any vertices $u, v \in V$ of a graph $G = (V, E)$ we call v a *weak neighbor* of u under a labeling $L = (S_L, W_L)$ if $\{u, v\} \in W_L$ and a *strong neighbor* if $\{u, v\} \in S_L$. An STC-labeling $L = (S_L, W_L)$ for a graph G is *optimal* if the number $|W_L|$ of weak edges is minimal. That is, there exists no other STC-labeling $L' = (S'_L, W'_L)$ such that $|W'_L| < |W_L|$.

Now we define the computational problems described informally in the previous chapter.

STRONG TRIADIC CLOSURE (STC)
Input: An undirected graph $G = (V, E)$ and an integer $k \in \mathbb{N}$.
Question: Is there an STC-labeling $L = (S_L, W_L)$ of G such that $|W_L| \leq k$?

STRONG TRIADIC CLOSURE WITH EDGE INSERTION (STC+)
Input: An undirected graph $G = (V, E)$ and an integer $k \in \mathbb{N}$.
Question: Is there an additional edge set $E' \subseteq \binom{V}{2} \setminus E$ such that there exists an STC-labeling $L = (S_L, W_L)$ of $G' = (V, E \cup E')$ with $E' \subseteq W_L$ and $|W_L| \leq k$?

For an STC+ instance (G, k) we call (L, E') a *solution* if it satisfies the conditions described in the problem definition. A solution (L, E') is *optimal* if there is no other solution $(L^* = (S_{L^*}, W_{L^*}), E^*)$ with $|W_{L^*}| < |W_L|$. For a graph $G = (V, E)$ and a labeling $L = (S_L, W_L)$ we say that a tuple (u, v, w) of vertices of V is a *strong* P_3 if the induced subgraph $G[\{u, v, w\}]$ is a P_3 consisting of the edges $\{u, v\} \in S_L$ and $\{v, w\} \in S_L$.

CLUSTER DELETION and CLUSTER EDITING are defined as follows.

CLUSTER DELETION (CD)
Input: An undirected graph $G = (V, E)$ and an integer $k \in \mathbb{N}$.
Question: Can we transform G into a cluster graph by at most k edge deletions?

CLUSTER EDITING (CE)
Input: An undirected graph $G = (V, E)$ and an integer $k \in \mathbb{N}$.
Question: Can we transform G into a cluster graph by deleting and adding at most k edges?

2.3 Parameterized Algorithms

In *parameterized algorithms* one analyzes discrete combinatorial problems that are conventionally NP-hard. If a problem is NP-hard, then an algorithm solving it is of exponential running time unless P = NP. The function expressing the running time of a parameterized algorithm depends not only on the input size, but on one or more problem-specific parameters as well. It is hoped that these parameters take only relatively “small” values, making the exponential growth of running time affordable. An algorithm whose running time is expressed by a function $f(k) \cdot n^c$ for a constant c and a computable function $f : \mathbb{N} \to \mathbb{N}$ is called an *fixed-parameter tractable* (FPT)-algorithm [5]. For every fixed parameter value, an FPT-algorithm yields an answer in polynomial time [22].

Formally we define a *parameterized problem* and *fixed-parameter tractability* as follows:

Definition 3. [5] A *parameterized problem* is a language $Q \subseteq \Sigma^* \times \mathbb{N}$, where Σ is a finite alphabet. For an instance $(x, k) \in \Sigma^* \times \mathbb{N}$, k is called the *parameter.*

For a parameterized problem Q we call an instance (I, k) a *yes-instance* if $(I, k) \in Q$ and a *no-instance* otherwise.

A parameterized problem defines as above is often called a *decision problem.* In the *optimization problem* of a parameterized problem Q, the goal for an instance x is to find a solution for the instance that is of minimum (or maximum) size. It is easy to see that given an algorithm for a decision problem of a parameterized problem Q that not only returns yes or no for an instance (x, k) but a feasible solution of size k, one can obtain an algorithm for the optimization problem of Q by performing a linear or any other kind of search on $1 \leq k \leq X$ for some upper bound X of k. The other way round, given an optimization algorithm and an integer k, one can compare the output of the optimization algorithm to k to solve the decision problem.

Definition 4. [5] A parameterized problem Q is *fixed-parameter tractable* (FPT) if it can be determined in $f(k) \cdot n^{\mathcal{O}(1)}$ time whether $(x, k) \in Q$ or $(x, k) \notin Q$, where f is a computable function only depending on k.

A *polynomial-time reduction* is a good tool for cases where it is hard find an FPT-algorithm for a parameterized problem with a desired running time.

Definition 5. [15] A *polynomial-time reduction* from a parameterized problem A to another parameterized problem B is an algorithm that, given an instance (x, k) of A, outputs an instance (x', k') of B such that

1. (x, k) is a yes-instance of A if and only if (x', k') is a yes-instance of B,
2. $k' \leq g(k)$ for some computable function g, and
3. the algorithm runs in polynomial time in $|I| + k$.

It is easy to see that if there is a polynomial-time reduction algorithm that given an instance (x, k) of A, outputs an instance (x', k') of B and if there is an algorithm for B running in time $f(|x'| + k')$ for some computation function f, then A can be solved in $f(|x'| + k')$ plus the running time of the algorithm. Thus, if B can be solved in polynomial time, than A can be solved in polynomial time as well. Conversely, if A is NP-hard, then B is NP-hard too.

One of the most important techniques in designing fixed-parameter algorithms is *kernelization* [5]. In practice, many input instances of computationally hard problems contain some parts that are relatively easy to handle and other parts that form the "hard part" of the input. So before starting an algorithm that is quite cost-intensive to solve the problem, one may apply a *data reduction* algorithm that runs in polynomial time and shrinks the input data as much as possible [22].

We say that two instances I and I' of a parameterized problem Q are *equivalent* when it holds that $(I, k) \in Q$ if and only if $(I', k') \in Q$. Now we give a formal definition that captures the idea of kernelization.

Definition 6. A *data reduction rule*, or simply, *reduction rule*, for a parameterized problem Q is a function $\phi : \Sigma^* \times \mathbb{N} \to \Sigma^* \times \mathbb{N}$ such that ϕ maps an instance (I, k) of Q to an equivalent instance (I', k') of Q and ϕ is computable in time polynomial in $|I| + k$.

To formalize the requirement that the output instance has to be relatively small, we apply the main principle of parameterized complexity: measuring the complexity in terms of the parameter. Thus, the *size* of a kernelization algorithm $\mathcal{A}$ is the function $\text{size}_{\mathcal{A}} : \mathbb{N} \to \mathbb{N} \cup \infty$ defined as $\text{size}_{\mathcal{A}}(k) := \sup\{|I'| + k' : (I', k') = \mathcal{A}(I, k), I \in \Sigma^*\}$.

Definition 7. [5] A *kernelization algorithm*, or simply a *kernel*, for a parameterized problem Q is an algorithm $\mathcal{A}$ that takes as input an instance (I, k) of Q and returns an equivalent instance (I', k') of Q. Moreover, it is required that $\text{size}_A(k) \leq g(k)$ for some computable function $g : \mathbb{N} \to \mathbb{N}$.

Reformulating the size requirement of this definition, we get the following: There exists a computable function $g(\cdot)$ such that for each output $(I', k') = \mathcal{A}(I, k)$ of a kernelization algorithm $\mathcal{A}$, it has to hold that $|I'| + k' \leq g(k)$. If an upper bound of $g(\cdot)$ is a polynomial (linear) function of the parameter k, we say that Q admits a *polynomial (linear) kernel.* The output of a kernelization algorithm is often called a kernel as well [5].

2.4 Approximation algorithms

Approximation algorithms are a separate way of dealing with algorithmically hard problems, next to parameterized algorithms [20].

An approximation algorithm $\mathcal{A}$ of an optimization problem Q returns a feasible solution whose size is approximating the size of an optimal solution, for example within a guaranteed factor of the optimal size, and runs in polynomial time [26].

3 Complexity Analysis

Even though it was claimed that STC+ is NP-hard when it was introduced [24], there is no proof showing it. Thus, in this chapter we establish the NP-hardness of STC+. To that end, we show that the polynomial-time reduction of 3-SAT to Cluster Editing shown in [17] is not only a reduction to CE, but to STC+ as well. In the reduction, a graph is constructed as an input for CE, which has maximum vertex degree six and is C_4- and *diamond*-free. The graphs C_4 and *diamond* are shown in Fig. 1.

Recall that two problems P_A and P_B *correspond* on C_4- and *diamond*-free graphs Π if for every graph $G \in \Pi$ and every non-negative integer k it holds that (G, k) is a yes-instance of P_A if and only if (G, k) is a yes-instance of P_B.

The following chain of lemmas provides the reduction from 3-SAT to STC+.

- There is a polynomial-time reduction from 3-SAT to CE on a C_4- and *diamond*-free graph [17].
- On C_4- and *diamond*-free graphs, CE and CD are corresponding. This is left to show.
- On *diamond*-free graphs, CD and STC are corresponding [12].
- On C_4- and *diamond*-free graphs, STC and STC+ are corresponding. This is left to show.

First, let us show that the graph created in the reduction of 3-SAT to CE is C_4- and *diamond*-free. In the 3-SAT problem, the input is a boolean formula ϕ in conjunctive normal form with at most three literals per clause (3-CNF) and it is asked whether there is an assignment to the variables of ϕ that satisfies all clauses of ϕ. Without loss of generality, the assumption is made that every clause contains exactly three literals and each variable is contained in at least two clauses.

Lemma 1. *The graph $G = (V, E)$ constructed in the reduction of* 3-SAT *to* Cluster Editing *in* [17] *is C_4- and* diamond*-free.*

Proof. Given a 3-CNF formula ϕ with n variables $\{x_0, \ldots, x_{n-1}\}$ and m clauses $C_0, \ldots, C_{m-1}$, the graph G is constructed as a Cluster Editing instance as follows.

For each variable x_i, $0 \leq i < n$, with m_i being the count of clauses containing x_i, a *variable cycle* is added to G consisting of the vertices $V_i^v := \{i_0, \ldots, i_{4m_i-1}\}$ and the edges $E_i^v := \{\{i_k, i_{k+1}\} : 0 \leq k \leq 4m_i - 2\} \cup \{\{i_{4m_i-1}, i_0\}\}$.

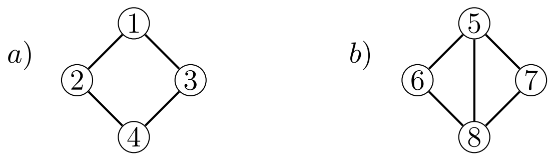

Figure 1: a) shows a C_4 and b) shows a *diamond*.

For each clause C, a fixed set of vertices needs to be "reserved" in the corresponding variable cycles of the variables contained by C. To this end, suppose that for each variable x_i there is an arbitrary but fixed ordering of the clauses that contain x_i and let $\pi(i,j) \in \{0,\ldots,4m_i-1\}$ denote the position of a clause C_j that contains x_i in this ordering. If two clauses C_j and $C_{j'}$ contain the same variable x_i such that $C_{j'}$ follows C_j in the ordering of clauses containing x_i, then $\pi(i,j') = \pi(i,j)+4$. Thus, for each clause C_j, the four vertices $v_{\pi(i,j)},\ldots,v_{\pi(i,j)+3}$ are "reserved'.

For a clause C_j containing the variables x_p, x_q, and x_r, the clause gadget that connects the three corresponding variable cycles is constructed by adding a new vertex a_j to the graph G. Let E^c_j denote the edge set of the clause gadget. It contains for each $i \in \{p,q,r\}$ the edges $\{a_j, i_{4\pi(i,j)}\}$ and $\{a_j, i_{4\pi(i,j)+1}\}$ if x_i is contained non-negated in C_j or the edges $\{a_j, i_{4\pi(i,j)+1}\}$ and $\{a_j, i_{4\pi(i,j)+2}\}$ otherwise. Now the construction of the graph $G=(V,E)$ is completed by setting $V := \bigcup_{i=0}^{n-1} V^v_i \cup \bigcup_{j=0}^{m-1}\{a_j\}$ and $E := \bigcup_{i=0}^{n-1} E^v_i \cup \bigcup_{j=0}^{m-1} E^c_j$.

By that construction G consists of the disjoint variable cycles and the clause vertices connecting those cycles. For two arbitrary clause vertices a_i and a_j it holds that they do not have a common neighbor since if they contain a common variable x, then by the construction of the graph, they have pairwise different neighbors in the variable cycle of x. Furthermore, for any variable cycle (V^v_i, E^v_i) and a neighboring clause vertex a_j being connected to two vertices v_p, v_{p+1} of the variable cycle, it holds that v_p, v_{p+1} have only a_i as a common neighbor. As mentioned above, all variables are contained in at least two clauses, so each variable cycle has at least eight vertices and thus two vertices in a cycle have at most one common neighbor in the cycle. That leaves a pair of a clause vertex and a vertex of a variable having two common neighbors as the only possible existence of C_4 or *diamond*. But if a vertex a_j is connected to two vertices $p_{4\pi(p,j)}$ and $p_{4\pi(p,j)+1}$ for a variable p, then there is no vertex in the variable cycle that is neighbor to both $p_{4\pi(p,j)}$ and $p_{4\pi(p,j)+1}$, since the vertices $p_{4\pi(p,j)}$ and $p_{4\pi(p,j)+1}$ are adjacent.

Therefore, for two arbitrary vertices u and v in G it holds that $|N(u) \cup N(v)| \leq 1$ and thereby G is C_4- and *diamond*-free. $\square$

Secondly, we show that CE and CD correspond on C_4- and *diamond*-free graphs.

Lemma 2. *Given a C_4- and* diamond*-free graph $G = (V, E)$ and an integer $k > 0$, then (G, k) is a yes-instance of* CD *if and only if (G, k) is a yes-instance of* CE.

Proof. ($\Rightarrow$) This implication is trivial: Any set of edges that are deleted of a graph to turn it into a cluster graph is not only a solution for CD but for CE as well.

($\Leftarrow$) Let $G = (V, E)$ be a C_4- and *diamond*-free graph. It holds for all nonadjacent vertices $u, v \in V$ that $|N(u) \cap N(v)| \leq 1$, that is they have at most one vertex $w \in V$ as a common neighbor as otherwise with another common neighbor $x \in V$, the graph $G[\{u, v, w, x\}]$ would form a C_4 or a *diamond*. Hence [17, Lemma 1] implies that there is an optimal CE solution that only deletes edges and therefore that solution is one for CD as well. $\square$

Now we show that STC and STC+ correspond on C_4- and *diamond*-free graphs as well.

Lemma 3. *Given a C_4- and* diamond*-free graph $G = (V, E)$ and an integer $k > 0$, then (G, k) is a yes-instance of* STC *if and only if (G, k) is a yes-instance of* STC+.

Proof. ($\Rightarrow$) This implication is trivial. If (G, k) is a yes-instance of STC then it is a yes-instance of STC+ as well.

($\Leftarrow$) Given a C_4- and *diamond*-free graph $G = (V, E)$ and an STC+ solution $(L = (S_L, W_L), E')$ such that $E' \subseteq W_L$ and L is an STC-labeling for $G' = (V, E \cup E')$, we can build an STC-labeling $L' = (S_{L'}, W_{L'})$ for G as follows: At first, we set $S_{L'} := S_L$ and $W_{L'} := W_L \setminus E'$. Then for each edge $e' = \{u, w\} \in E'$ there is at most on vertex $v \in V$ such that (u, v, w) is a strong P_3 under L in G, since G is C_4- and *diamond*-free. Therefore, we add $\{u, v\}$ to $W_{L'}$ and remove $\{u, v\}$ from $S_{L'}$. Afterwards, L' is an STC-labeling for G, since for every strong P_3 (u, v, w) in G under L with $\{u, w\} \in E'$ we labeled on of the edges $\{u, v\}$ or $\{v, w\}$ weak. Furthermore, we have that $|W_{L'}| = |W_L| \leq k$. $\square$

Lemma 1, Lemma 2, Lemma 3 and the fact that on C_4- and *diamond*-free graphs CD and STC correspond [12] imply the following.

Corollary 1. *On C_4- and* diamond*-free graphs the problems* CE,CD,STC *and* STC+ *correspond.*

This gives us the means to prove the main result of this section.

Theorem 1. STC+ *is NP-hard even when restricted to graphs with maximum vertex degree six and to C_4- and* diamond*-free graphs.*

Proof. Let ϕ be a 3-SAT formula. As analyzed in Lemma 1, there is a polynomial-time reduction from 3-SAT to CE constructing a C_4- and *diamond*-free graph $G = (V, E)$ such that ϕ is satisfiable if and only if G can be transformed into a cluster graph by at most $k := 10m$ edge modifications. By Lemma 2, it follows that

(G, k) is a yes-instance of CE if and only if
(G, k) is a yes-instance of CD.

Moreover, G is *diamond*-free and Cluster Editing and STC correspond on *diamond*-free graphs [12] and thus:

(G, k) is a yes-instance of CD if and only if
(G, k) is a yes-instance of STC.

Since G is C_4- and *diamond*-free, Lemma 3 implies the following:

(G, k) is a yes-instance of STC if and only if
(G, k) is a yes-instance of STC+.

With these equivalences we finally get the following, which proves the correctness of the reduction from 3-SAT to STC+ and the NP-hardness of STC+:

ϕ is satisfiable if and only if there is an
STC+ solution (L', E') for G with $|W_{L'}| + |E'| \leq k := 10m$.

$\square$

4 An Approximation Algorithm

With STC+ being NP-hard, the computational expense of solving instances of the problem in given time might exceed the available resources. One way of circumventing the presumed exponential running time of an algorithm providing an optimal solution for an NP-hard problem is to use an approximation algorithm running in polynomial time. In the following we provide an 3-approximation algorithm for STC+ by relating to the polynomial-time reduction of STC+ to 3-Hitting Set in the $\mathcal{O}(\log(n))$-approximation of STC+ in [24]. The 3-Hitting Set problem is defined as follows:

3-Hitting Set (3HS)

Input: A universe of elements U and a collection of subsets of U, each of size three, $\mathcal{S} = \{S_1, \ldots, S_n\}$.

Question: Is there a subset $C \subseteq U$ of size at most k such that for each $S_i \in \mathcal{S}$ it holds that $S_i \cap C \neq \emptyset$, that is, each set of $\mathcal{S}$ is *hit* by C?

Lemma 4. *Let $G = (V, E)$ be a graph and k a non-negative integer. There is a* 3HS *instance (U, S, k) such that there is an* STC+ *solution $(L = (S_L, W_L), E')$ for (G, k) if and only if there is a hitting set C for (U, S, k).*

Proof. Let $G = (V, E)$ be a graph and k a non-negative integer. One can compute in polynomial time a universe of elements U and a collection of subsets of U, $\mathcal{S} = \{S_1, \ldots, S_n\}$, such that (G, k) is a yes-instance of STC+ if and only if (U, S, k) is a yes-instance of 3HS [24]. This can be done as follows: The universe U consists of all vertex pairs $\{u, v\}$ where $u, v \in V$. For each P_3 $t = (u, v, w)$ in G, we create the set $S_t = \{\{u, v\},\{v, w\},\{u, w\}\}$.

Let C be a hitting set for the set $\mathcal{S} = \{S_t : t \text{ is a } P_3 \text{ in } G\}$, then let $L = (S_L, W_L)$ be a labeling of E with $W_L := \{\{u, v\} \in C : \{u, v\} \in E\} \cup E'$ and $S_L := E \setminus W_L$. Furthermore, let $E' := \{\{u, v\} \in C : \{u, v\} \notin E\}$. We have that $L = (S_L, W_L)$ is an STC-labeling for the graph $G + E'$, since every P_3 in G is either covered by a weak edge or is closed by an edge $e' \in E'$.

Conversely, given a solution (L, E') for the STC+ instance G, we can define the hitting set C by adding all pairs of vertices $\{u, v\} \in W_L$ to it. Since all P_3s are covered by $E' \cup W_L$, C hits the set $\mathcal{S} = \{S_t : t \text{ is a } P_3 \text{ in } G\}$.

Constructing the sets U and $\mathcal{S}$ can obviously be done in polynomial time. □

Proposition 1. *There is a 3-approximation algorithm for the* STC+ *problem.*

Proof. Given a graph $G = (V, E)$ as input of the optimization version of STC+ and let k be the size of an optimal STC+ solution for G. By

Lemma 4, there is a 3HS instance (U, S) such that for each integer $k \geq 0$ it holds that there is an STC+ solution $(L = (S_L, W_L), E')$ for (G, k) if and only if there is a hitting set C for (U, S, k). It is a well-known fact that there is a 3-approximation for 3HS, but for sake of completeness, we shortly give the details of the approximation:

If for the 3HS instance (U, S) the size of a minimum hitting set C is k, we can find a solution of size at most $3k$ as follows: We greedily search for a maximal matching $M \subseteq \mathcal{S}$, that is a set of hyper edges such that for all edges $M_i, M_j \in M$ with $i \neq j$ it holds that $M_i \cap M_j = \emptyset$. Since C has to contain at least one element of each $M_i \in M$, the size of C is at least $|M|$. Let $X := \bigcup_{M_i \in M} M_i$ be the set of elements that are contained of the hyper edges of M. We have that $|X| = 3 \cdot |M| \leq 3 \cdot |C| = 3k$. Furthermore, we have that X hits all edges of $\mathcal{S}$ as otherwise there would be an edge $S_i \in \mathcal{S}$ not being hit by X, implying that $S_i \cap S_j = \emptyset$ for all $S_j \in M$, being a contradiction to M being a maximal matching. It follows that X is a hitting set of $\mathcal{S}$ of size at most three times the size of an optimal solution.

By Lemma 4, it follows that there is an STC+ solution $(L = (S_L, W_L), E')$ for $(G, 3k)$. $\square$

5 Kernelization for Parameters k and ℓ

In this section, we provide a $4k$-vertex kernel for STC+ parameterized by k and an $\mathcal{O}(\ell \cdot 2^{\ell})$-vertex kernel parameterized by ℓ.

5.1 Data Reduction leading to a $4k$-vertex Kernel

Based on the concept of critical cliques, we obtain a problem kernel for STC+ consisting of at most $4k$ vertices in $\mathcal{O}(n^2 m)$ time. The data reduction rules are inspired by the reduction rules used to obtain a $4k$-vertex kernel for Cluster Editing [13]. Critical cliques are defined as follows.

Definition 8. A critical clique in a graph $G = (V, E)$ is a clique K where the vertices of K all have the same neighbors in $V \setminus K$, and K is maximal under this property.

The basic idea behind introducing critical cliques is as follows. If there is a solution $(L = (W_L, S_L),\ E')$ for G of size at most k, then there are at most $2k$ vertices affected. A vertex being *affected* means that it is an endpoint of an edge that is either labeled weak or added to the set E'. To give an upper bound $4k$ for the number of vertices of V in G, we have to find an upper bound $2k$ for the vertices being unaffected as well. To give such an upper bound, we show that there exists an optimal solution such that all vertices in the same critical clique have the same fate, meaning that the edges to a neighbor of the critical clique are either all labeled weak or all labeled strong. We provide data reduction rules such that an STC+ instance reduced by these rules has at most $2k$ unaffected vertices or is a no-instance.

The first reduction rule is quite obvious:

Rule 1. *Remove all isolated critical cliques.*

Lemma 5. *Rule 1 is safe and is exhaustively applied in $\mathcal{O}(n + m)$ time.*

Proof. Let $G = (V, E)$ be a graph and K a critical clique in G with $N(K) = \emptyset$. We show the following: There exists an optimal STC+ solution (L, E') for G with $|W_L| \leq k$ if and only if there exists an STC+ solution (L^*, E^*) for $G - K$ with $|W_{L^*}| \leq k$.

($\Rightarrow$) Let (L, E') be an STC+ solution for G with $|W_L| \leq k$. We create a labeling $L^* := (S_{L^*}, W_{L^*})$ by setting $S_{L^*} := S_L \setminus \{\{u, v\} \in S_L : u, v \in K\}$, $W_{L^*} := W_L \setminus \{\{u, v\} \in W_L : u, v \in K\}$. Then (L^*, E') is an STC+ solution for $G - K$ with $|W_{L^*}| \leq |W_L| \leq k$.

($\Leftarrow$) Now let (L^*, E^*) be an STC+ solution for $G - K$ with $|W_{L^*}| \leq k$. We create a labeling $L = (S_L, W_L)$ as follows: $S_L := S_{L^*} \cup \{\{u, v\} : u, v \in K\}$ and $W_L := W_{L^*}$. We have that (L, E^*) is an STC+ solution for G with $|W_L| = |W_{L^*}| \leq k$, since there are no P_3s containing vertices of the isolated clique K. Furthermore, there is no strong P_3 (u,v,w) with $u,v,w \in V \setminus K$ under the labeling L since there was no such strong P_3 under the labeling L^*.

Regarding the running time, note that finding and removing all isolated cliques is clearly doable in $\mathcal{O}(n+m)$ time. □

The following lemma provides that for each vertex in the neighborhood of a critical clique the edges connecting this vertex to the critical clique are all labeled the same.

Lemma 6. *Let $(G=(V,E),k)$ be an instance of* STC+ *and K a critical clique of G. There exists an optimal solution $(L=(W_L,S_L),\ E')$ such that for each $v \in N(K)$ we have $E(\{v\},K) \subseteq S_L$ or $E(\{v\},K) \subseteq W_L$.*

Proof. Let K be a critical clique in G and let $(L=(W_L,S_L),\ E')$ be an optimal solution such that $E(\{v\},K) \not\subseteq S_L$ and $E(\{v\},K) \not\subseteq W_L$.

For each vertex $u \in K$, let $W_u = \{v \in V : \{u,v\} \in W_L\}$ and $S_u = \{v \in N_G(K) : \{u,v\} \in S_L\}$. For each $v \in W_u$ it holds that $v \in N_G(K) \cup N_G^2(K)$, since if there would be a $v \in W_u \setminus (N_G(K) \cup N_G^2(K))$ there could not be a P_3 containing u and v, so the insertion of the edge $\{u,v\}$ would contradict (L,E') being an optimal solution. Without loss of generality let $u \in K$ be such that $|W_u| = \min_{v\in K}\{|W_v|\}$. We create a new solution (L^*,E^*) by taking the solution (L,E') but for each $u' \in K\setminus\{u\}$ and for each $w \in W_u$ we set the edge $\{u',w\}$ to weak and add $\{u',w\}$ to E' if $w \in N_G^2(K)$ and for each $w' \in S_u$ we set the edge $\{u',w'\}$ to strong. Assume towards a contradiction that by these modifications we created a strong P_3 (u',w,v) or (w,u',w') with $u' \in K\setminus\{u\}$, $w,w' \in S_u$ and $v \in N^2(K)$. Then (u,w,v) or (w,u,w') would be a strong P_3 as well, which is a contradiction to $(L=(W_L,S_L),\ E')$ being a solution. It follows that we did not create any strong P_3 by building (L^*,E^*). Since for each $u' \in K$ it holds that $|W_u| \le |W_{u'}|$ we have that $|W_{L^*}| \le |W_L|$ and therefore (L^*,E^*) is an optimal solution.

Furthermore, we have that for each $v \in N(K)$ it holds: $E(\{v\},K) \subseteq S_{L^*}$ or $E(\{v\},K) \subseteq W_{L^*}$. □

With the previous lemma we have the means to show that if a critical clique K is “big”, that is, if $|K| \ge |N(K)|$, then we do not need to insert edges between K and $N^2(K)$ as shown in the following.

Lemma 7. *Let K be a critical clique in G with $|K| \ge |N(K)|$, then there exists an optimal solution (L,E') such that for all edges $\{u,v\} \in E'$ it holds: $u \notin K$ and $v \notin K$ and $E(\{v\},K) \subseteq S_L$ or $E(\{v\},K) \subseteq W_L$ for each $v \in N(K)$.*

Proof. Let K be a critical clique in $G=(V,E)$ with $|K| \ge |N(K)|$. By Lemma 6, there exists an optimal solution $(L=(W_L,S_L),\ E')$ such that $E(\{v\},K) \subseteq S_L$ or $E(\{v\},K) \subseteq W_L$ for each $v \in N(K)$. Let $G' = G+E'$.

Assume there exists a vertex set $D \subseteq N^2(K)$ such that $E_{G'}(K, D) \subseteq E'$ with $E_{G'}(K, D) \neq \emptyset$, that is, there exist weak edges having endpoints in K. Clearly, these weak edges have the other endpoints in $N^2(K)$, as otherwise they would not close any strong P_3. This implies that there exist vertices $u \in K$ and $v \in N(K)$ such that for each $w \in D$ we have that (u, v, w) is a P_3 in G or (L, E') was not an optimal solution. Lemma 6 and $\{u, v\} \in S_L$ imply that $E(\{v\}, K) \subseteq S_L$ and therefore there are $|D| \cdot |K|$ many edges inserted between K and D which we notate by $F := \{\{u, w\} : u \in K, w \in D\} \subseteq E'$. We set $E^* := E' \backslash F$ and define an STC-labeling $L^* := (S_{L^*}, W_{L^*})$ as follows: $S_{L^*} := S_L \backslash E_G(N(K), D)$ and $W_{L^*} := (W_L \cup E_G(N(K), D)) \backslash F$. It is obvious that L^* is an STC-labeling for $G + E^*$ since setting the edges $E_G(N(K), D)$ weak implies that $E_G(N(K), N^2(K)) \subseteq W_{L^*}$, so there could not exists a strong P_3 in $G + E^*$ under L^*. Furthermore, we have that we removed $|D| \cdot |K|$ many edges from E' and W_L but only added $|N(K)| \cdot |D|$ to W_L. Since $|N(K)| \cdot |D| \leq |K| \cdot |D|$, we have that $|W_{L^*}| \leq |W_L|$ and therefore (L^*, E^*) is an optimal solution as well, which has the property that no edges of E^* are between K and $N^2(K)$. □

The second reduction rule is based on the idea of removing a critical clique K such that the number of edges between its first and second neighborhood is small and only few edges need to be inserted in the first neighborhood to turn the $N[K]$ into a complete graph. For a critical clique K and a critical clique $K' \subseteq N(K)$, the set of edges which are needed to completely connect K' and $N(K)$ is denoted by $E^f_{K',N(K)}$.

Rule 2. *If there exists a critical clique K such that $|K| \geq |N(K)|$ and for each critical clique $K' \subseteq N(K)$ it holds that $|K| \cdot |K'| \geq |E(K', N^2(K))| + |E^f_{K',N(K)}|$, then remove $N[K]$ from G and decrease k by $|E(N(K), N^2(K))| + |\binom{N[K]}{2}| - |E(N[K])|$.*

Lemma 8. *Rule 2 is safe and can be exhaustively applied in $\mathcal{O}(n^2 m)$ time.*

Proof. Let K be a critical clique such that $|K| \geq |N(K)|$ and for each critical clique $K' \subseteq N(K)$ it holds that $|K| \cdot |K'| \geq |E(K', N^2(K))| + |E^f_{K',N(K)}|$. Let $G_r = (V_r, E_r)$ be the graph that results from removing $N[K]$ from G. We show that (G, k) has an STC+ solution $(L = (S_L, W_L), E')$ with $|W_L| \leq k$ if and only if (G_r, k_r) with $k_r := k - (|E(N(K), N^2(K))| + |\binom{N[K]}{2}| - |E(N[K])|)$ has an STC+ solution $(L_r = (S_{L_r}, W_{L_r}), E'_r)$ with $|W_{L_r}| \leq k_r$.

($\Rightarrow$) By Lemma 7, there exists an optimal solution $(L = (W_L, S_L), E')$ for (G, k) such that $E(K, N^2(K)) \cap E' = \emptyset$ and $E(\{v\}, K) \subseteq S_L$ or $E(\{v\}, K) \subseteq W_L$ for each $v \in N(K)$. Let $G' = G + E'$.

Assume there exists a critical clique $K' \subseteq N(K)$ such that there exists a vertex set $X \subseteq K'$ defined as $X := \{v \in K' : E(K, \{v\}) \subseteq W_L\}$ that are weak neighbors of K in K'. Let $S_{K',N^2(K)} := E_{G'}(K', N^2(K)) \cap S_L$ be the set of strong edges between K' and $N^2(K)$. It holds that there is no vertex $v \in K' \setminus X$ that has a strong neighbor $w \in N^2(K)$, since there would be a vertex $u \in K$ such that (u, v, w) is a strong P_3 in G' under L which would contradict that (L, E') is a solution.

Let $E^f_{X,N(K)} := \{\{u, v\} : u \in X \wedge v \in N(K) \setminus K' \wedge E(K, \{v\}) \in S_L \wedge \{u, v\} \notin E\}$ be the set of edges that would be needed to completely connect X and the vertices of $N(K) \setminus K'$ that are strong neighbors of K.

We construct a new solution $(L^* = (S_{L^*}, W_{L^*}), E^*)$ as follows: $S_{L^*} := S_L \setminus S_{K',N^2(K)} \cup E(K, X)$ and $W_{L^*} := W_L \setminus E(K, X) \cup S_{K',N^2(K)} \cup E^f_{X,N(K)}$ and $E^* := E' \cup E^f_{X,N(K)}$. We have that (L^*, E^*) is a solution, since by labeling the edges $S_{K',N^2(K)}$ weak and by inserting the edges $E^f_{X,N(K)}$, the edges $E(K, X)$ cannot be part of any strong P_3 in $G + E^*$ under L^*. Since K satisfies the conditions of Rule 2, it holds that $|K| \cdot |K'| \geq |E(K', N^2(K))| + |E^f_{K',N(K)}|$. Thus, we have the following.

$$\begin{aligned}
&|K| \cdot |K'| \geq |E(K', N^2(K))| + |E^f_{K',N(K)}| \\
\Leftrightarrow &|K| \cdot |K'| \cdot \frac{|X|}{|K'|} \geq |E(K', N^2(K))| \cdot \frac{|X|}{|K'|} + |E^f_{K',N(K)}| \cdot \frac{|X|}{|K'|} \\
\overset{*}{\Leftrightarrow} &|K| \cdot |X| \geq |E(X, N^2(K))| + |E^f_{X,N(K)}| \\
\Leftrightarrow &|E(K, X)| \geq |E(X, N^2(K))| + |E^f_{X,N(K)}|
\end{aligned}$$

The inequation on the right hand side of the equivalence $(*)$ follows since X is a subset of K' and $|E^f_{K',N(K)}| = |K'| \cdot |Y|$ for $Y := \{v \in N(K) \setminus K' : E(K, \{v\}) \in S_L \wedge E(K', v) = \emptyset\}$ and $|E(K', N^2(K))| = |K'| \cdot |Z|$ for $Z \subseteq N^2(K) \cap N(K')$. This implies $W_{L^*} \leq W_L$, thus (L^*, E^*) is an optimal solution.

We can apply that procedure for every critical clique $K' \subseteq N(K)$ and obtain an optimal solution $(\tilde{L} = (S_{\tilde{L}}, W_{\tilde{L}}), \tilde{E})$ such that for each critical clique $K' \subseteq N(K)$ it holds that $E(K, K') \subseteq S_{\tilde{L}}$, $E(K', N^2(K)) \subseteq W_{\tilde{L}}$ and $E^f_{K',N(K)} \subseteq \tilde{E}$.

We define an STC+ solution $(L_r = (S_{L_r}, W_{L_r}), E'_r)$ for (G_r, k_r) by defining the labeling L_r with $S_{L_r} := S_{\tilde{L}} \setminus \{\{u, v\} \in S_{\tilde{L}} : u, v \in N[K]\}$ and $W_{L_r} := W_{\tilde{L}} \setminus \{\{u, v\} \in W_{\tilde{L}} : u \in N[K] \vee v \in N[K]\}$. Furthermore, we set $E'_r := \tilde{E} \setminus \{\{u, v\} \in \tilde{E} : u, v \in N[K]\}$. It holds that (L_r, E'_r) is an STC+ solution for G_r, since if there would be a strong P_3 (u, v, w) in $G_r + E'_r$ under L_r, it would hold that $u, v, w \in V \setminus N[K]$, and since all edges in $G[V \setminus N[K]]$ are labeled the same in L_r and $\tilde{L}$, (u, v, w) would be a strong P_3 in $G + \tilde{E}$ under $\tilde{L}$ as well, which is a contradiction to $(\tilde{L}, \tilde{E})$ being an STC+ solution for (G, k). Furthermore, we have that $|W_{L_r}| = |W_{\tilde{L}}| - (|E(N(K), N^2(K))| + |\binom{N[K]}{2}| - |E(N[K])|) \leq k - (|E(N(K), N^2(K))| + |\binom{N[K]}{2}| - |E(N[K])|) = k_r$.

($\Leftarrow$) Now let $(L_r = (S_{L_r}, W_{L_r}), E'_r)$ be an STC+ solution for (G_r, k_r). We define a labeling $L = (S_L, W_L)$ by setting $S_L := S_{L_r} \cup (\binom{N[K]}{2} \setminus E^f_{K',N(K)} \subseteq$

$\tilde{E}$) and $W_L := W_{L_r} \cup E^f_{K',N(K)} \cup E(N(K), N^2(K))$. The additional edge set is $E' := E'_r \cup E^f_{K',N(K)}$. We have that L is an STC+-labeling for $G+E'$, since for every induced P_3 (u,v,w) in G with $u,v,w \in E_r$ it holds that (u,v,w) is not a strong P_3 under L_r in G_r and therefore not strong under L in G. Furthermore, by the insertion of $E^f_{K',N(K)}$ we have that $G(N[K])$ is complete in $G + E'$ and therefore cannot contain an induced P_3. Moreover, since $E(N(K), N^2(K)) \subseteq W_L$, there cannot be a strong P_3 in $G + E'$ under L with vertices in $N[K]$ and in $E \setminus N[K]$. Furthermore, we have that $|W_L| = |W_{L_r}| + |E(N(K), N^2(K))| + |\binom{N[K]}{2}| - |E(N[K])| \leq k_r + |E(N(K), N^2(K))| + |\binom{N[K]}{2}| - |E(N[K])| = k$. Thus, (L, E') is an STC+ solution for (G, k).

Concerning the running time: For a fixed critical clique K, we can compute for all critical cliques $K' \subseteq N(K)$ the sizes of the two sets of edges $E(K', N^2(K))$ and $E^f_{K',N(K)}$ in $\mathcal{O}(m)$ time. To decide, whether Rule 2 can be applied, one iterates over all critical cliques K and computes the sets $E(K', N^2(K))$ and $E^f_{K',N(K)}$ for all critical cliques $K' \in N(K)$. Thereby, the applicability of Rule 2 can be decided in $\mathcal{O}(n \cdot m)$ time. Clearly, Rule 2 can be applied in $\mathcal{O}(n+m)$ time. This gives us a running time of $\mathcal{O}(n^2 m)$. $\square$

If there is an instance to which none of the two above reduction rules applies, we call it *reduced* with respect to these rules. On that base we can achieve a problem kernel for STC+ with at most $4k$ vertices. The proof of the following theorem works mostly analogously to the one showing a $4k$ vertex kernel for Cluster Editing [13].

Lemma 9. *If a graph $G = (V, E)$ that is reduced with respect to Rules 1 and 2 has more than $4k$ vertices, then (G, k) is a no-instance of* STC+.

Proof. Let (G, k) be a yes-instance of the STC+ problem reduced by Rule 1 and Rule 2 and let (L, E') be an optimal solution. We partition V into two sets: V_1, the set of vertices which are endpoints of edges of W_L and $V_2 = V \setminus V_1$. Recall, that the vertices $v \in V_1$ are called *affected* and $v \in V_2$ *unaffected*. Clearly $|V_1| \leq 2k$, as $|V_1| > 2k \Leftrightarrow |W_L| > k$ would contradict (L, E') being a solution. It remains to show that $|V_2| \leq 2k$. The basic idea of the proof is the following: We show that in a critical clique either all vertices are affected or all vertices are unaffected. Then we can conclude that the number of vertices in an unaffected critical clique K is limited by the number of weak edges incident with vertices in $N(K)$. Since we can show that there is an assignment of unaffected vertices to weak edges such that there are at most two unaffected vertices assigned to a weak edge and there are at most k weak edges in any yes-instance, it follows that there are at most $2k$ unaffected vertices.

First, let us make following observation:

Observation 1. *For each critical clique $K \subseteq V$ it holds that $K \subseteq V_1$ or $K \subseteq V_2$, that is, either all vertices of K are affected or all vertices are unaffected.*

Proof. Let $v \in V$ be an unaffected vertex and let K be the critical clique containing v, that is for each vertex $w \in N(K)$ the edge $\{v, w\}$ is labeled strong. Lemma 6 implies $E(K, w) \subseteq S_L$. Thus we have $E(K, N(K)) \subseteq S_L$ and thereby all vertices of K are unaffected.

The vertices of each $K' \in N(K) \cup N^2(K) \setminus K$ are affected, since all edges between K and $N(K)$ are strong and no edges between K and $N^2(K)$ were added to E', all edges of $E(N(K), N^2(K))$ have to be labeled weak. Furthermore, if there is a critical clique $K_i \subseteq N(K)$ that has no edges to $N^2(K)$, then there is another critical clique $K_j \subseteq N(K)$ such that $E(K_i, K_j) \cap E = \emptyset$ since otherwise $K_i \subseteq K$. Thereby, the vertices of K_i are endpoints of edges in E'. $\Diamond$

Let $K_1, \ldots, K_l$ be the unaffected critical cliques. Let $\mathcal{K}_1 = \{K_i : |K_i| < |N(K_i)| : 1 \leq i \leq l\}$ be the set of unaffected critical cliques having less vertices than their neighborhood and let $\mathcal{K}_2 = \{K_1, \ldots, K_l\} \setminus \mathcal{K}_1$ be the set of unaffected critical cliques having at least as many vertices as their neighborhood.

Let $K_i \in \mathcal{K}_1$. Since the instance is reduced with respect to Rule 1, K_i is not isolated. As analyzed in Observation 1, each critical clique $K' \in N(K_i) \cup N^2(K_i) \setminus K_i$ is affected. That is, each vertex in K' is an endpoint of an edge inserted between vertices of $N(K)$, of a weak edge between $N(K)$ and $N^2(K)$, or both. This implies that $|N(K_i)| \leq 2|E'_i| + |E_i^W|$, with E'_i being the edges of E' added between the vertices in $N(K_i)$ and E_i^W being the edges between $N(K_i)$ and $N^2(K_i)$ that are labeled weak. Since $|K_i| < |N(K_i)|$ it holds that $|K_i| < 2|E'_i| + |E_i^W|$.

Now let be $K_i \in \mathcal{K}_2$. Since we cannot apply Rule 2 and $|K_i| \geq |N(K_i)|$, it holds that there exists $K' \in N(K_i)$ such that $|K_i| \cdot |K'| < |E(K', N^2(K_i))| + |E^f_{K',N(K_i)}|$. Since K_i is unaffected, all edges of $E(K', N^2(K_i))$ are labeled weak and the edges $E^f_{K',N(K_i)}$ have to be added to E'. Recall that $E^f_{K',N(K)}$ is the set of edges needed to completely connect K' and $N(K_i)$. It follows that

$$\begin{aligned}
|K_i| &< \frac{|E_{K',N^2(K_i)}| + |E^f_{K',N(K_i)}|}{|K'|} \\
&\leq |E_{K',N^2(K_i)}| + |E^f_{K',N(K_i)}| \\
&\leq |E'_i| + |E_i^W|.
\end{aligned}$$

The two inequalities above that provide upper bounds for the number of vertices in each $K_i \in \mathcal{K}_1 \cup \mathcal{K}_2 = V_2$ leads to the following.

$$\begin{aligned} |V_2| &= \sum_{i=1}^{l} K_i \\ &\overset{*}{\leq} \sum_{i=1}^{l} (2|E'_i| + |E_i^W|) \\ &\overset{**}{\leq} 2|E'| + \sum_{i=1}^{l} (|E_i^W|) \\ &\overset{***}{\leq} 2|E'| + 2|E^W| \\ &\leq 2|W_L| \leq 2k \end{aligned}$$

Here, E^W denotes the edges of E that are labeled weak. Inequality (*) follows from the analysis in the above two cases. Note that any two unaffected critical cliques K_i and K_j are not neighbored. Since they are not the same critical cliques, without loss of generality there exists a vertex $v \in N(K_i) \setminus N(K_j)$ which would imply the existence of a strong P_3 (v, u, w) in $G + E'$ under L for any $u \in K_i$ and $w \in K_j$. Furthermore, for two unaffected critical cliques K_i and K_j it holds that $N(K_i) \cup N(K_j) = \emptyset$, as otherwise for a vertex $v \in N(K_i) \cup N(K_j)$, $u \in K_i$ and $w \in K_j$ there would be the strong P_3 (u, v, w). Thus, the sets E'_i and E'_j are disjoint, implying inequality (**). In the worst case an edge $e = \{u, v\} \in W_L$ is counted two times, as there can be $u \in N(K_i)$ and $v \in N(K_j)$ for two unaffected critical cliques K_i, K_j. The edge e cannot be counted more than two times, since if it is counted by the unaffected critical cliques K_i, K_j, K_s, then without loss of generality we may assume $u \in N(K_i)$ and $u \in N(K_s)$, which would imply that there are $v \in K_i$ and $w \in K_s$ such that (v, u, w) is a strong P_3. This provides the safeness of inequality (***). It follows that $|V| = |V_1| + |V_2| \leq 4k$. □

Theorem 2. STC+ *admits a* $4k$*-vertex kernel that can be computed in* $\mathcal{O}(n^2 m)$ *time.*

Proof. By Lemma 9, any yes-instance of STC+ that is reduced with respect to Rule 1 and Rule 2 has at most $4k$ vertices. Since the reduction rules take $\mathcal{O}(n + m)$ and $\mathcal{O}(n^2 m)$ time to exhaustively apply, we have an overall running time of $\mathcal{O}(n^2 m)$. □

5.2 Data Reduction Leading to an $\mathcal{O}(\ell \cdot 2^\ell)$-Vertex Kernel

The parameter $\ell := |E| - k$ is the dual parameter of k for STC+. That is, given a graph $G = (V, E)$ and an integer k, looking for an additional edge set E' and an STC-labeling $L = (S_L, W_L)$ for the graph $G+E'$ with $E' \subseteq W_L$ and $|W_L| \leq k$ is equivalent to looking for an additional edge set and labeling such that $|S_L| - |E'| \geq \ell$. This is made clear by the following equivalences with W_L^G being the edges of E labeled weak.

$$\begin{aligned}|S_L| - |E'| \geq \ell \Leftrightarrow |S_L| - |E'| \geq |E| - k \Leftrightarrow (|S_L| - |E|) - |E'| \geq -k \\ \Leftrightarrow -|W_L^G| - |E'| \geq -k \Leftrightarrow |W_L^G| + |E'| \leq k \Leftrightarrow |W_L| \leq k\end{aligned}$$

Looking for an STC+ solution such that the number of strong edges is at least ℓ, which is done for STC, would not be appropriate, since one could simply label all edges of a graph strong and insert weak for every strong P_3 to maximize the number of strong edges.

We show that STC+ admits an $\mathcal{O}(\ell \cdot 2^\ell)$-vertex kernel for the parameterization by $\ell := |E| - k$ which works mostly analogously to the $\mathcal{O}(\ell \cdot 2^\ell)$-vertex kernel for STC [12].

Let $G = (V, E)$ be a graph and k a non-negative integer. First, we find a maximum matching $M \subseteq E$ of G in $\mathcal{O}(\sqrt{n} \cdot m)$ time [21]. If $|M| \geq \ell$, then we already have an additional edge set $E' = \emptyset$ and an STC-labeling $L = (M, E \setminus M)$ for $G + E'$ with $|M| - |E'| \geq \ell$. Therefore, we can subsequently assume that the size of a maximum matching in G is smaller than ℓ. The basic idea of the kernelization is to show that there are superfluous vertices in the independent set that is left by removing the edges of M and their endpoints from G. To that end we partition the vertex set V of G into the sets V_M, I_1 and I_2 as follows:

- $V_M := \{v \in V : v \text{ is an endpoint of an edge } e \in M\}$
- $I_2 := \{v \in V \setminus V_M : \exists \{u, w\} \in M : u \text{ and } w \text{ are both neighbors of } v\}$
- $I_1 := V \setminus (V_M \cup I_2)$

Furthermore, we say that two vertices $u, v \in I_1$ are in the same *family* F if $N(u) = N(v)$. Given a family F, we refer to the neighborhood of the vertices in F by $N(F)$.

First, we show that the number of vertices in I_2 is upper-bounded by ℓ:

Observation 2. *$|I_2| < \ell$.*

Proof. Assume there is an edge $\{u, v\} \in M$ such that there are two vertices w and w' that both have u and v as neighbors. Then we have that $M' := M \cup \{\{w, u\}, \{w', v\}\} \setminus \{\{u, v\}\}$ is a matching as well and it holds that $|M'| > |M|$, which is a contradiction to M being a maximum matching. Thus, since $|M| < \ell$ and each edge in M has at most one neighbor in I_2, it holds that $|I_2| < \ell$. □

Now we give a reduction rule to remove superfluous vertices from the graph. This rule decreases the parameter k, but it reduces it by the amount of edges that are removed from the graph and therefore the parameter ℓ does not change.

Rule 3. *For each family F of vertices in I_1 do the following: If $|N(F)| < |F|$, remove $|F|-|N(F)|$ vertices from F and decrease k by $(|F|-|N(F)|)\cdot|N(F)|$.*

To prove the safeness of the rule above, we use the concept of *weak cuts* [12], which is defined as follows.

Definition 9. Let $L = (S_L, W_L)$ be an STC-labeling for the graph $G = (V, E)$. A *weak cut* under L is a cut C of the graph G such that $E_C \subseteq W_L$.

In [12], the following crucial proposition was shown:

Proposition 2. *Let $L = (S_L, W_L)$ be an* STC*-labeling for the graph $G = (V, E)$. If there is a weak cut $C = (V_1, V_2)$ with the cut set E_C, then there is an* STC*-labeling $L' = (S_{L'}, W_{L'})$ for $G' = (V, E \setminus E_C)$ such that $|S_{L'}| = |S_L|$.*

Note that since Proposition 2 works for a graph and an STC-labeling, we only use it on a graph that is obtained after inserting the additional edge set in STC+. Now we have the means to prove the correctness of Rule 3.

Proposition 3. *Rule 3 is safe.*

Proof. Let $G^* = (V^*, E^*)$ be the instance resulting from an application of Rule 3 to an instance (G, k) for a graph $G = (V, E)$. Furthermore, let E_r be the edges we removed during the reduction in Rule 3. We show that there is an STC+ solution (L^*, E^*) for $G^* + E^*$ with $|S_{L^*}| - |E^*| \geq \ell$ if and only if there is an STC+ solution (L, E') for $G + E'$ with $|S_L| - |E'| \geq \ell$.

($\Rightarrow$) Let $(L' = (S_{L'}, W_{L'}, E'')$ be an optimal STC+ solution for G^* with $|S_{L'}| - |E''| \geq \ell$. We define the labeling $L = (S_L, W_L)$ with $S_L := S_{L'}$ and $W_L := W_{L'} \cup E_r$. The labeling L is an STC-labeling for $G + E''$ since we added only weak edges to the graph G^* and thereby could not create any strong P_3s. Furthermore, $|S_L| - |E''| = |S_{L'}| - |E''| \geq \ell$.

($\Leftarrow$) Let $(L = (S_L, W_L), E')$ be an optimal STC+ solution for G with $|S_L| - |E'| \geq \ell$. Assume there is a family of vertices $F = \{u_1, \ldots, u_{|F|}\}$ and $N(F) := \{v_1, \ldots, v_{|N(F)|}\}$ with $|N(F)| < |F|$ and there are more than $|N(F)|$ vertices $\{u_1, \ldots, u_p\}$ in F having a strong neighbor in $N(F)$. Let v_r be such a vertex and let $E_s \subseteq S_L$ be the set of strong edges and $E_+ \subseteq E'$ be the set of additional edges incident with v_r. It holds that $|E_s| \leq |N(F)|$ and $|E_+| \geq r - 1 \geq |N(F)|$. By labeling all edges in E_s weak, v_r has no strong neighbors and therefore we can remove all edges E_+. This implies that (L^*, E^*) with $S_{L^*} := S_L \setminus E_s$, $W_{L^*} := W_L \cup E_s \setminus E_+$ and $E^* := E' \setminus E_+$ is an optimal STC+ solution for G as well with $|W_{L^*}| \leq |W_L|$. We can repeat this procedure leading to an optimal STC+ solution $(\tilde{L}, \tilde{E})$ such that for each family F with $|N(F)| < |F|$, there are $|F| - |N(F)|$ vertices in F having only weak neighbors under the labeling $\tilde{L}$. Let $X := \{v \in V : v \text{ has only weak neighbors under } \tilde{L}\}$. We have that $(V \setminus X, X)$ is a weak cut C for $G + \tilde{E}$ and therefore Proposition 2 implies that there is an STC-labeling L' for $G' = (V, E \setminus E_C \cup \tilde{E})$ such that $|S_{L'}| = |S_L|$. Since $|E_C| = (|F| - |N(F)|) \cdot |N(F)| = |E_r|$, it holds $|W_{L'}| = |W_L| - |E_r| \leq k$. Therefore, $(L', \tilde{E})$ is an optimal STC+ solution for G' so we conclude that there is an optimal STC+ solution (L'', E'') with $|S_{L''}| - |E''| = |S_{L'}| - |E'| \geq \ell$.

□

Theorem 3. STC+ *admits a kernel with* $\mathcal{O}(\ell \cdot 2^\ell)$ *vertices that can be computed in* $\mathcal{O}(\sqrt{n}m)$.

Proof. Let (G, k) be an STC+ instance reduced with respect to Rule 3. As argued above, the number of vertices V_M that are endpoints of some edge $e \in M$, is less than $2 \cdot |M| = 2\ell$ since any instance with $|M| \geq \ell$ is a yes-instance. By Observation 2, it holds that $|I_2| < \ell$. Hence, it remains to show the upper bound of $|I_1|$.

First, note that every edge $\{u, w\} \in M$ has at most one endpoint with neighbors in I_1 [12, Observation 1]. Furthermore, there are no edges between I_1 and I_2, since $I_1 \cup I_2$ is an independent set. Since $|M| < \ell$, there are less than ℓ vertices in V_M that are adjacent to vertices of I_1. Therefore, there are less than 2^ℓ different families F of vertices in I_1. Since the graph G is reduced, each family is of size at most ℓ. This implies that there are less than $\ell \cdot 2^\ell$ vertices in I_1 which leads to the upper bound of $\mathcal{O}(\ell \cdot 2^\ell)$ vertices for G.

Concerning the running time regard the following: A maximum matching can be found in $\mathcal{O}(\sqrt{n}m)$ time [21] and executing Rule 3 exhaustively can be done in linear time. □

6 Fixed-Parameter Algorithms for the Parameterization by k

In the analysis of data reduction rules leading to a linear vertex problem kernel, the main idea was to "cut away the trivial parts" of the input instance, leaving behind the problem kernel, the hard part, which presumably cannot be solved in polynomial time. Thus, we cannot avoid using some exponential time methods to obtain an optimal solution. Depth-bounded search trees are one of the most common approaches in parameterized algorithms that originates in the general idea of backtracking [5]. It is a standard way of performing a systematic exhaustive search in the huge search space related to a computationally hard problem [22].

The basic idea behind a systematic search using a depth-bounded search tree algorithm is to find in polynomial time a "small subset" of the input instance such that at least one element of this subset is part of an optimal solution [22]. The algorithm then *branches* into a number of subproblems that are obtained by the possibilities of adding some elements of the subset to the solution [5]. This strategy is recursively applied, until no more branching is possible. If we found a feasible solution in the branch we can output it, otherwise the algorithm backtracks to a previous state.

A branching algorithm is a good tool to provide an FPT-algorithm for a parameterized problem. During the branching, in each step the parameter is reduced by some integer at least one, thus, the number of branches is limited by an exponential function depending on k, but each branch can be processed in time polynomial in $|V|$.

Formally, in a branching step for an instance I of a computational problem, we generate from I simpler instances $I_1, \ldots, I_r$ ($r \geq 2$) of the same problem such that.

$$I \text{ is a yes-instance if and only if } \exists i \in \{1, \ldots, r\} : I_i \text{ is a yes-instance.}$$

If for an instance with the parameter k a branching algorithm calls itself recursively with decreased parameters $k-d_1, k-d_2, \ldots, k-d_p$, then $(d_1, \ldots, d_p)$ is called the *branching vector* of this recursion [5].

In Section 6.1 we provide a basic bounded search tree algorithm leading to a search tree of size $\mathcal{O}(3^k)$ and in Section 6.2 we give more complex branching strategies improving the worst-case size of the search tree to $\mathcal{O}(2.57^k)$. In Section 6.3 we provide a fixed-parameter algorithm by reducing STC+ to 3HS and use an algorithm for 3HS having the worst case running time of $2.076^k \cdot n^{\mathcal{O}(1)}$ [27].

6.1 Basic Branching Strategy

The fundamental idea of the following branching algorithm is to recursively find a P_3 (u, v, w) in the input graph $G = (V, E)$ and branch on the subcases to solve this conflict by labeling one of the edges $\{u, v\}$ or $\{v, w\}$ weak or adding the weak edge $\{u, w\}$ to the additional edge set.

In the following we describe a recursive algorithm for STC+. Since the algorithm calls itself recursively, we need to deal with an additional edge set and a labeling as further input. Given as input a graph $G = (V, E)$ and a non-negative integer k', the algorithm outputs an additional edge set E' and an STC-labeling $L = (S_L, W_L)$ for $G' = (V, E \cup E')$ with $E' \subseteq W_L$ and $|W_L| \leq k'$, or returns false if no such solution (L, E') exists. Note that for an STC+ instance (G, k) the first call of the algorithm is done with the following input: $k' := k$, $E' = \emptyset$ and the labeling $L = (S_L, W_L)$ with $S_L = E$ and $W_L = \emptyset$. Obviously, (L, E') is most likely not an STC+ solution for (G, k), but is step by step turned into one during the branching algorithm.

Input: A graph $G = (V, E)$, an integer $k' \geq 0$, an additional edge set $E' \subseteq \binom{V}{2} \setminus E$ and a labeling $L = (S_L, W_L)$.

- If L is already an STC-labeling for $G + E'$, then we are done and output (L, E') as a solution.
- Otherwise, if $k \leq 0$, then it is not possible to find a solution in this branch of the search tree: return to the parent in the search tree.
- Otherwise, find a strong P_3 (u, v, w) in $G + E'$. Recursively call the branching algorithm on the following three subcases consisting of the input G, $k' \geq 0$, E'' and $L' = (S_{L'}, W_{L'})$ as specified below:

  (B1) $W_{L'} = W_L \cup \{\{u, v\}\}$, $S_{L'} = S_L \setminus \{\{u, v\}\}$, $E'' = E'$ and $k' = k - 1$.

  (B2) $W_{L'} = W_L \cup \{\{v, w\}\}$, $S_{L'} = S_L \setminus \{\{v, w\}\}$, $E'' = E'$ and $k' = k - 1$.

  (B3) $W_{L'} = W_L \cup \{\{u, w\}\}$, $S_{L'} = S_L$, $E'' = E' \cup \{\{u, w\}\}$ and $k' = k - 1$.

Theorem 4. STC+ *can be solved in* $\mathcal{O}(3^k \cdot k + n^2 m)$ *time.*

Proof. The recursive algorithm described above is checking in its exhaustively search every combination of labeling edges of P_3s and adding additional edges that might be a solution of the input graph and thus is correct.

Regarding the running time, we make the following observations. The preprocessing before starting the algorithm, to obtain a problem kernel of the input instance respecting the reduction rules Rule 1 and Rule 2 can be done in $\mathcal{O}(n^2 m)$ time (Lemma 9). In the algorithm there are three branching cases and in each one the parameter is reduced by one, thereby the branching vector is $(1, 1, 1)$. Thus, the size of the search tree build by the algorithm is bounded by 3^k [6]. The computational overhead related to every node of the search tree is $\mathcal{O}(k)$. Before starting the search tree algorithm, we can set up a linked list of all P_3s, this clearly can be done in $\mathcal{O}(mn)$ time which is dominated by the $\mathcal{O}(n^2m)$ term. In each vertex of the search tree we need to update the list of P_3, after labeling an edge weak or adding a new edge to E'. Since we applied the reduction rules of Section 5.1, we have at most $4k$ vertices in the graph. It follows that by labeling an edge weak or adding a new weak edge, there are at most $4k$ many strong P_3 removed and by using a double linked list, one can update it in $\mathcal{O}(k)$ time. That can be achieved as follows: after labeling an edge $\{u, v\}$ weak or adding an edge $\{u, v\}$ to E', we can iterate over all $\mathcal{O}(k)$ many vertices $w \in V$ with $w \neq u$ and $w \neq v$ and update the status of that triple if it was a strong P_3 but now not any more. This update can be done in constant time by using an array of size $|V|^3 = (4k)^3$ to store for each vertex triple a pointer to the possible strong P_3 in the list of P_3s. $\square$

6.2 Refined Branching Strategy

In the following we improve the search tree algorithm described in Section 6.1. The basic idea of finding vertices $u, v, w \in V$ with such that (u, v, w) is a strong P_3 in $G + E'$, remains. Now, however, we are going to distinguish into more specific cases and provide additional branching steps for every possible situation. We branch on the case of u and w not sharing another common neighbor but v and on the two cases that emerge when u and w share another common neighbor $x \in V$. By that case distinction we can improve the worst-case running time bound to $\mathcal{O}(2.57^k \cdot k + n^2m)$.

We start with describing the three cases that can emerge when considering a strong P_3 (u, v, w) in $G + E'$.

(C1) u and w do not share a common neighbor $x \in V$ with $x \neq v$.

(C2) u and w have a common neighbor $x \in V$ with $x \neq v$ and $\{x, v\} \in E$. That is, u, v, w, x form a *diamond*.

(C3) u and w have a common neighbor $x \in V$ with $x \neq v$ and $\{x, v\} \notin E$. That is, u, v, w, x form a C_4.

Dealing with the case (C1) one can easily see that branching into the cases (B1) and (B2) suffices, since adding an additional edge to E' closes only one strong P_3, but labeling one of the two edges weak closes at least one strong P_3. In the complexity analysis in Section 3, we already stated out that in C_4- and *diamond*-free graphs, there is an optimal STC+ solution without any edge insertions. The following lemma is based on the same intuition.

Lemma 10. *Given a graph $G = (V, E)$, an integer $k \geq 0$, a labeling $L_P = (S_{L_P}, W_{L_P})$, an additional edge set E'_P and vertices $u, v, w \in V$ such that (u, v, w) is a strong P_3 in $G + E'_P$ under L_P and u and w do not share a common neighbor $x \in V$ with $x \neq v$. Let $(L^* = (S_{L^*}, W_{L^*}), E^*)$ be a solution that is yielded by branching into the cases* (B1) *or* (B2) *such that* $|W_{L^*}| := \min\{|W_{L^x}| : (L^x = (S_{L^x}, W_{L^x}), E^x) \text{ is a solution yielded by branching into subcases (B1) and (B2)}\}$. *Then branching into the case* (B3) *cannot yield a solution $(L = (S_L, W_L), E')$ with $|W_L| < |W_{L^*}|$ and thus can be omitted.*

Proof. Consider the optimal solution $(L = (S_L, W_L), E')$ for the STC+ instance (G, k) where $\{u, w\}$ was added to E'. Assume that $\{u, v\} \in S_L$ and $\{v, w\} \in S_L$ as otherwise (L, E') was not optimal since (u, v, w) is the only P_3 in G having u and w as endpoints. Let $L^* = (S^*_L, W^*_L)$ with $S^*_L = S_L \setminus \{\{u, v\}\}$ and $W^*_L = W_L \cup \{\{u, v\}\} \setminus \{\{u, w\}\}$ and let $E^* = E' \setminus \{\{u, w\}\}$. Since (u, v, w) is the only P_3 having u and w as endpoints in G, we did not create any strong P_3 in $G + E^*$. Therefore, (L^*, E^*) is a solution as well and $|W^*_L| = |W_L|$. □

Lemma 10 shows that in case (C1) a branching into the two cases (B1) and (B2) is sufficient. Thus, since each case reduces the parameter by one, we have a branching vector of $(1, 1)$ with the branching number 2 for the case (C1).

In case (C2) u and w have a common neighbor $x \in V$, with $x \neq v$ and $\{x, v\} \in E$ and therefore $\{u, v, w, x\}$ form a *diamond* which is presented in Fig. 2. When branching into the case (B3), the edge $\{u, w\}$ is added to E' and there is no strong P_3 left consisting of the vertices u, v, w, x. This is branching case (D1) and is labeled by (2) in Fig. 2.

In both branching cases (B1) and (B2) the strong P_3 (u, x, w) left, which leads to the subcases labeled by (3) and (4). On these cases we can branch again, respectively labeling one of the edges $\{u, x\}$ and $\{x, w\}$ weak, as labeled by (5), (6), (7) and (8). This leads to following branching for the case (C2):

(D1) $W_{L'} = W_L \cup \{\{u, w\}\}$, $E'' = E' \cup \{\{u, w\}\}$ and $k' = k - 1$.

(D2) $W_{L'} = W_L \cup \{\{u, v\}, \{u, x\}\}$, $E'' = E'$ and $k' = k - 2$.

(D3) $W_{L'} = W_L \cup \{\{u, v\}, \{x, w\}\}$, $E'' = E'$ and $k' = k - 2$.

(D4) $W_{L'} = W_L \cup \{\{v, w\}, \{u, x\}\}$, $E'' = E'$ and $k' = k - 2$.

(D5) $W_{L'} = W_L \cup \{\{v, w\}, \{x, w\}\}$, $E'' = E'$ and $k' = k - 2$.

This branching leads to a branching vector of $(1, 2, 2, 2, 2)$ with the branching number 2.57 [6].

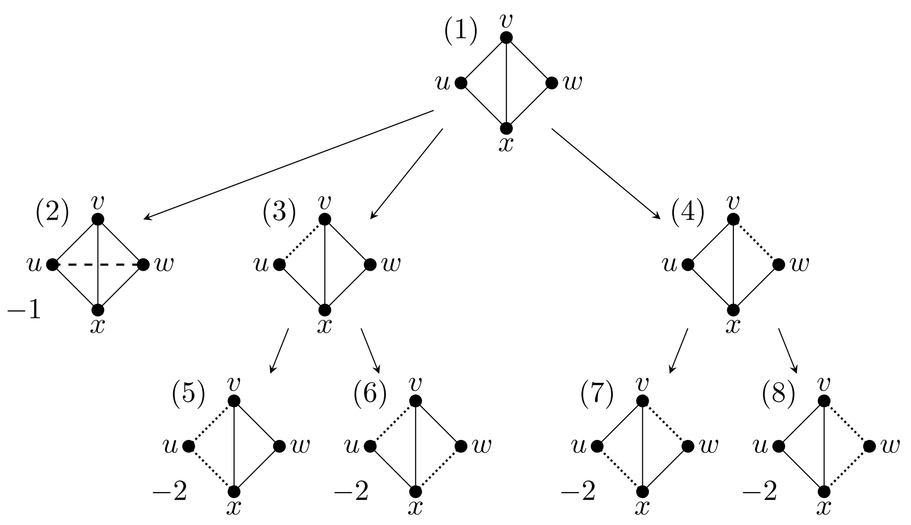

Figure 2: The branching for case (C2). Dotted lines denote weak labeled edges, dashed lines denote inserted weak edges.

In case (C3) u and w have a common neighbor $x \in V$ with $x \neq v$ and $\{x, v\} \notin E$ and thereby form a C_4 which is presented in Fig. 3. We leave branching case (B1) as it is. This branching is (E1) and is labeled by (2). We focus on refining the subcases emerging when branching into the cases (B2) and (B3).

When branching into (B2) as shown in (3), the strong P_3s (v, u, x) and (u, x, w) are left. We branch into the subcase of labeling $\{u, x\}$ weak, shown by (5) where we close all strong P_3s in $G[\{u, v, w, x\}]$ and into the subcase of not labeling $\{u, x\}$ weak, shown by (6). In the second case we need to label $\{x, w\}$ weak and to add $\{v, x\}$ to E'.

In the branching case (B3), as labeled by (4), the strong P_3s (v, u, x) and (v, w, x) are left. We branch into the subcase of adding $\{v, x\}$ to E', shown by (7), which closes all strong P_3s in $G[\{u, v, w, x\}]$ and into the subcase of not adding $\{v, x\}$ to E', shown by (8). In the second case, labeling only one of the edges $\{u, v\}$ or $\{v, w\}$ weak is no option since that would be subcases of (B1) or (B2). That leads to the necessity of labeling both $\{u, x\}$ and $\{x, w\}$ weak. Thus, we define the following branching:

(E1) $W_{L'} = W_L \cup \{\{u, v\}\}$, $E'' = E'$ and $k' = k - 1$.

(E2) $W_{L'} = W_L \cup \{\{v, w\}, \{u, x\}\}$, $E'' = E'$ and $k' = k - 2$.

(E3) $W_{L'} = W_L \cup \{\{v, w\}, \{x, w\}, \{v, x\}\}$, $E'' = E' \cup \{\{v, x\}\}$ and $k' = k - 3$.

(E4) $W_{L'} = W_L \cup \{\{u, w\}, \{v, x\}\}$, $E'' = E' \cup \{\{u, w\}, \{v, x\}\}$ and $k' = k - 2$.

(E5) $W_{L'} = W_L \cup \{\{u, x\}, \{x, w\}, \{u, w\}\}$, $E'' = E' \cup \{\{u, w\}\}$ and $k' = k - 3$.

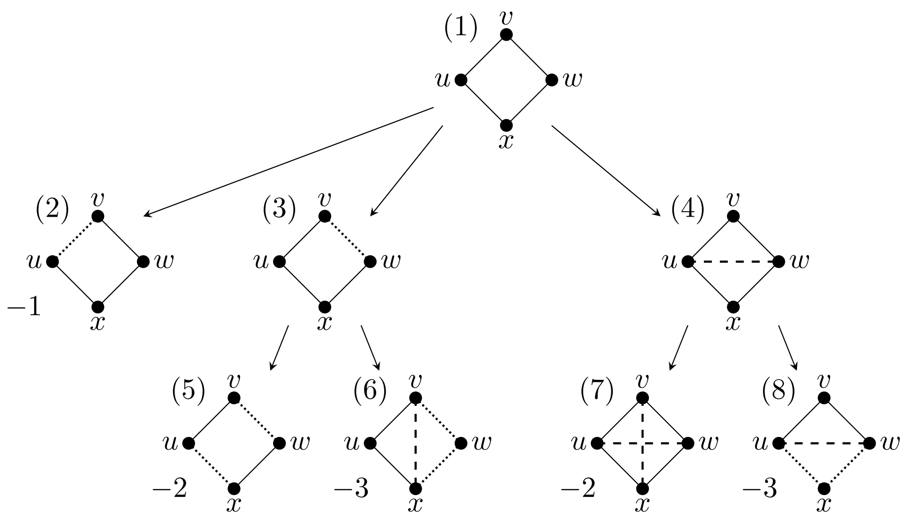

Figure 3: The branching for case (C2). Dotted lines denote weak labeled edges, dashed lines denote inserted weak edges.

This branching leads to a branching vector of $(1, 2, 2, 3, 3)$ with the branching number 2.27 [6].

In summary, the refined branching leads to a worst-case branching vector of $(1, 2, 2, 2, 2)$ yielding the branching number 2.57. Thus, we have a search tree of size $\mathcal{O}(2.57^k)$ for our refined branching algorithm. This leads to the following theorem:

Theorem 5. STC+ *can be solved in* $\mathcal{O}(2.57^k \cdot k + n^2 m)$ *time.*

6.3 Improved Running Time by Reducing STC+ to 3-Hitting Set

Using an FPT-algorithm for 3HS [27], we obtain the following theorem:

Theorem 6. STC+ *can be solved in* $2.076^k \cdot n^{\mathcal{O}(1)}$ *time.*

Proof. Given a graph $G = (V, E)$ and a non-negative integer k. By Lemma 4, there is a polynomial-time reduction from the STC+ instance (G, k) to a 3HS instance (U, S, k), with the running time $\mathcal{O}(n^3)$. This instance can be solved in $\mathcal{O}(2.076^k \cdot n^{\mathcal{O}(1)})$ time [27]. Since the running time of the reduction bounded by $\mathcal{O}(n^3)$ is dominated by the term $n^{\mathcal{O}(1)}$, the overall running time to solve STC+ in that way is $2.076^k \cdot n^{\mathcal{O}(1)}$. □

7 On The Relation of Strong Subgraph Closure with Edge Insertion and Subgraph-free Editing

During the research for STC+ we arrived at the presumption thatSTC+ may correspond to Cluster Editing. Recall that STC+ and CE correspond if for every graph $G = (V, E)$ and every integer $k \geq 0$ it holds that (G, k) is a yes-instance of STC+ if and only if (G, k) is a yes-instance of CE. In this section we analyze, whether the generalizations of the two problems, Strong Subgraph closure with Edge Insertion and Subgraph-free Editing, correspond for other H-graphs on three or four vertices, besides P_3.

We say that the two problems *correspond* for a certain graph H that shall be "destroyed", if for every graph G and every non-negative integer k it holds that (G, H, k) is a yes-instance of SSC+ if and only if (G, H, k) is a yes-instance of Subgraph-free Editing.

Before giving the formal definitions of the two problems, we need do define when a graph G is H-free under a labeling.

Definition 10. Given the graphs G and H, an additional edge set $E' \subseteq \binom{V_G}{2} \setminus E_G$ and a labeling L, we say that G is *H-free* under L if there exists no set of vertices $X \subseteq V_G$ such that $(G + E')[X]$ contains only strong edges and is isomorphic to H.

In the following we formally define the two problems introduced above.

Strong Subgraph closure with Edge Insertion (SSC+)
Input: The undirected graphs $G = (V_G, E_G)$, $H = (V_H, E_H)$ and a non-negative integer $k \in \mathbb{N}$.
Question: Is there a labeling $L = (S_L, W_L)$ and an additional edge set $E' \subseteq \binom{V}{2} \setminus E$ with $E' \subseteq W_L$ and $|W_L| \leq k$ such that the graph $G' = (V_G, E_G \cup E')$ is H-free under L?

Given a SSC+ instance (G, H, k), a labeling $L = (S_L, W_L)$ and an addition edge set $E' \subseteq \binom{V}{2} \setminus E$. We call (L, E') a *solution* for (G, H, k) if the graph $G' = (V_G, E_G \cup E')$ is H-free under L and $|W_L| \leq k$.

Subgraph-free editing (SE)
Input: The undirected graphs $G = (V_G, E_G)$, $H = (V_H, E_H)$ and a non-negative integer $k \in \mathbb{N}$.
Question: Can we transform G into an H-free graph with at most k edge deletions and insertions?

Given an SE instance (G, H, k), an addition edge set $E^+ \subseteq \binom{V}{2} \setminus E$ and an edge set $E^- \subseteq E$. We call (E^+, E^-) a *solution* for (G, H, k) if the graph $G' = (V_G, E_G \cup E^+ \setminus E^-)$ is H-free and $|E^+| + |E^-| \leq k$.

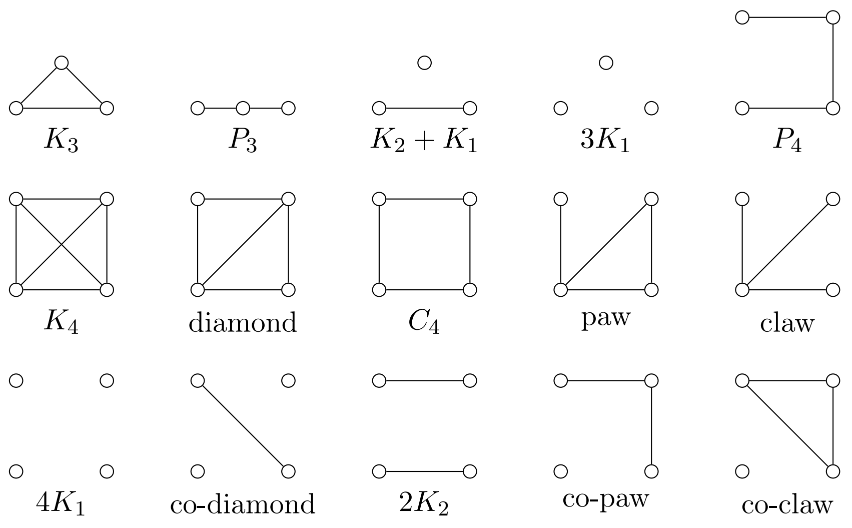

Figure 4: The small graphs H on three or four vertices.

The graphs on which we analyzed whether the two problems correspond or not are the graphs K_r and rK_1 for $r \in \mathbb{N}$ and all possible graphs on three or four vertices, except P_3 and $K_2 + K_1$, listed in Fig. 4. The graph K_r is a clique of size r and the graph rK_1 is an independent set of size r.

To simplify the proof of the crucial theorem in this section, showing the correspondence between SSC+ and SE for certain H graphs, the following three lemmas need to be provided:

Lemma 11. *Let G and H be arbitrary graphs and k a non-negative integer. If (G, H, k) is a yes-instance of* SE*, then (G, H, k) is a yes-instance of* SSC+ *as well.*

Proof. Let $G = (V_G, E_G)$ and $H = (V_H, E_H)$ and let (E^-, E^+) be a solution for the SE instance (G, H, k), with E^- the set of edges being removed from G and E^+ the set of edges being added to G, to turn G into the H-free graph $G^* = (V_H, (E_H \cup E^+) \setminus E^-)$. We define the following sets of edges as: $E' := E^+$, $W_L := E^+ \cup E^-$ and $S_L := E_G \setminus E^-$. We show that $(L = (S_L, W_L), E')$ is a solution for the SSC+ instance (G, H, k). Assume towards a contradiction that there exists a set of vertices $X \subseteq V_G$ such that for the graph $G' = (V_G, E_G \cup E')$ the induced subgraph $G'[X]$ contains only strong edges under the labeling L and is isomorphic to H. Then for every edge $e \in G'[X]$ it holds that $e \notin W_L$ and by definition of the sets $e \notin E^- \cup E^+$ as well. Then $G^*[X]$ is isomorphic to H which is a contradiction to (E^-, E^+) being a solution for the SE instance (G, H, k). It follows that the SSC+ instance (G, H, k) is a yes-instance with the solution $(L = (S_L, W_L), E')$. $\square$

The following is a well-known fact, but for sake of completeness, we provide a short proof.

Lemma 12. *Let $G = (V_G, E_G)$ and $H = (V_H, E_H)$ be arbitrary graphs. It holds that G is H-free if and only if $\overline{G}$ is $\overline{H}$-free.*

Proof. ($\Rightarrow$) Let G be H-free. Assume that there exists a set of vertices $X \subseteq V_G$ such that the induced subgraph $\overline{G}[X]$ is isomorphic to $\overline{H}$. Then $G[X]$ is the complement graph of $\overline{G}[X]$ and obviously isomorphic to H. This contradicts G being H-free.

($\Leftarrow$) Since the complement graph of $\overline{G}$ is G, this direction of the proof works analogously to the other direction and therefore can be omitted. □

The following lemma provides that if a graph G is H-free under a labeling L, then we can easily build another labeling L' such that $\overline{G}$ is $\overline{H}$-free under L'.

Lemma 13. *Let $G = (V_G, E_G)$ and $H = (V_H, E_H)$ be arbitrary graphs, $L = (S_L, W_L)$ be a labeling for $G+E'$ and let $E' \subseteq \binom{V_G}{2} \setminus E_G$ be an additional set of edges with $E' \subseteq W_L$. It holds that $G + E'$ is H-free under a labeling $L = (S_L, W_L)$ if and only if $\overline{G}+E''$ is $\overline{H}$-free under the labeling $L' = (S'_L, W'_L)$ with $E'' = W_L$, $W'_L = E'$ and $S'_L = \binom{V_G}{2} \setminus (E_G \cup E')$.*

Proof. ($\Rightarrow$) Let $G + E'$ be H-free under the labeling L. Assume towards a contradiction that there exists a set of vertices $X \subseteq V$ such that the induced subgraph $(\overline{G} + E'')[X]$ consists only of strong edges and is isomorphic to $\overline{H}$ under the labeling L' (recall that $\overline{G} + E'' := (V(\overline{G}), E(\overline{G}) \cup E''))$. The fact that $(\overline{G} + E'')[X]$ consists only of strong edges implies that $(\overline{G} + E'')[X] = \overline{G}[X]$. By definition of the sets $W_{L'}$ and E'' it follows that $(G + E')[X]$ consists only of strong edges and therefore $(G + E')[X] = G[X]$. Thus, we have that $\overline{(\overline{G} + E'')[X]} = \overline{\overline{G}}[X]) = G[X] = (G + E')[X]$. In other words, $(\overline{G} + E'')[X]$ is the complement graph of $(G + E')[X]$. It follows that $(\overline{G} + E'')[X]$ being isomorphic to $\overline{H}$ under L' implies that $(G + E')[X]$ is isomorphic to H under L. This contradicts $G + E'$ being H-free under L.

($\Leftarrow$) Since the complement graph of $\overline{G}$ is G, this direction of the proof works analogously to the other direction and therefore can be omitted. □

Now we have the means to show the central theorem for this section.

Theorem 7. *The problems* SSC+ *and* SE

- *do not correspond on $H \in \{P_4$, diamond, co-diamond, C_4, $2K_2$, paw, co-paw, claw, co-claw$\}$ and*
- *correspond on $H \in \{K_r, rK_1\}$ for $r \in \mathbb{N}$.*

Proof. For $H \in \{P_4$, *co-diamond*, C_4, *co-paw*, *claw*$\}$, Fig. 5 - Fig. 9 provide examples that SSC+ and SE do not correspond.

For $H \in \{$*diamond*, $2K_2$, *paw*, *co-claw*$\}$, we have that the graphs are respectively the complement graphs of *co-diamond*, C_4, *co-paw* and *claw*. Lemma 12 and Lemma 13 imply that the complement graphs of the graphs in Fig. 5 - Fig. 9 provide examples that SSC+ and SE do not correspond on the graphs $H \in \{$*diamond*, $2K_2$, *paw*, *co-claw*$\}$.

It remains to show that the problems correspond on $H \in \{K_r, rK_1\}$ for $r \in \mathbb{N}$. Lemma 11 implies that every optimal solution of an SE instance (G, H, k) provides a solution for the SSC+ instance of the same input. Therefore it is sufficient to show that for every graph G and $H \in \{K_r, rK_1 : r \in \mathbb{N}\}$ and a non-negative integer k it holds that the SSC+ instance (G, H, k) having a solution implies that the corresponding SE instance has a solution as well.

Case 1: $H = K_r$. Let G be an arbitrary graph, k a non-negative integer and let $(L = (S_L, W_L),\ E')$ be an optimal solution for the SSC+ instance (G, H, k) which implies that $G + E'$ under L does not contain a subgraph consisting only of strong edges and being isomorphic to H. First, note that E' is empty. That is because H is a K_r and thereby a complete graph, every subraph $G[\{u_1, \ldots, u_r\}]$ of G for arbitrary vertices $u_1, \ldots, u_r \in V$ can only be isomorphic to H if it is complete as well. Adding a weak edge to the graph could never destroy any K_r subgraph and because of that, $E' = \emptyset$ or $(L = (S_L, W_L),\ E')$ is not an optimal solution.

Let the following sets be defined as: $E^- := W_L$ and $E^+ := E' = \emptyset$. We have that (E^-, E^+) is a solution for the SE instance (G, H, k), since for every subgraph X of G that is isomorphic to H, there is at least one edge $e \in E(X)$ labeled weak, which as well is in E^- and thereby is removed from G. Since E^+ is empty, there are no edges added to the graph that could create a K_r.

Case 2: $H = rK_1$. Let G be an arbitrary graph, k a non-negative integer and and $(L = (S_L, W_L),\ E')$ be an optimal solution for the SSC+ instance (G, H, k). By Lemma 13, there is a labeling $\underline{L' = (S'_L, W'_L)}$ and a set of additional weak edges E'' such that $\overline{G} + E''$ is $\overline{H}$ under L'. In case 1 we showed that this implies the existence of a solution (E^-, E^+) for the SE instance $(\overline{G}, \overline{H}, k)$ such that $\overline{G} - E^- + E^+$ is $\overline{H}$-free. The complement graph of $\overline{G} - E^- + E^+$ is $G + E^- - E^+$ and by Lemma 12, it is H-free.

□

In Theorem 7 we did not consider the case $H = K_2 + K_1$. Since this graph is the complement graph of a P_3, by Lemma 12 and Lemma 13, it follows that proving or disproving the correspondence of SSC+ and SE on $K_2 + K_1$ also settles the case $H = P_3$. This is left as an open question for now.

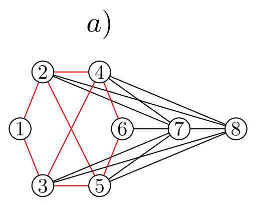

Figure 5: An example that SSC+ and SE do not correspond on $H = P_4$. Red edges are part of a P_4, blue dashed edges are (weak) inserted edges and blue dotted are weak labeled / deleted edges.

In Fig. 5, *a*) shows a graph G containing the P_4s $(1, 2, 4, 5)$, $(1, 3, 5, 6)$, $(1, 2, 5, 6)$, $(1, 3, 4, 6)$. For the SSC+ solution in *b*), adding the weak edge $\{1, 6\}$ and labeling all other edges strong provides an optimal solution of size 1. For the SE in *c*), adding the edge $\{1, 6\}$ would create the P_4 $(1, 6, 7, 8)$. There are at least two edge modifications in SE required to solve all P_4s.

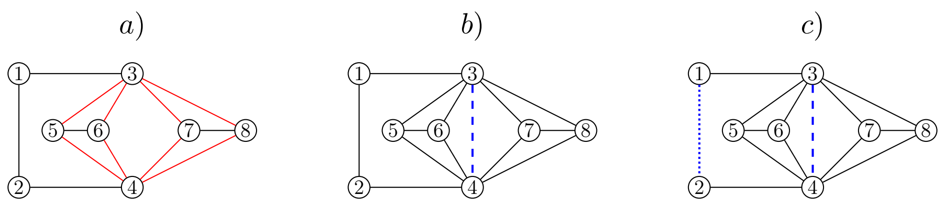

Figure 6: An example that SSC+ and SE do not correspond on $H = C_4$. Red edges are part of a C_4, blue dashed edges are (weak) inserted edges and blue dotted are weak labeled / deleted edges.

In Fig. 6, *a*) shows a graph G containing the C_4s $(3, 7, 4, 6)$, $(3, 8, 4, 6)$, $(3, 7, 4, 5)$, $(3, 8, 4, 5)$. For the SSC+ solution in *b*), adding the weak edge $\{3, 4\}$ and labeling all other edges strong provides an optimal solution of size 1. For the SE in *c*), adding the edge $\{3, 4\}$ would create the C_4 $(1, 3, 4, 2)$. There are at least two edge modifications in SE required to solve all C_4s.

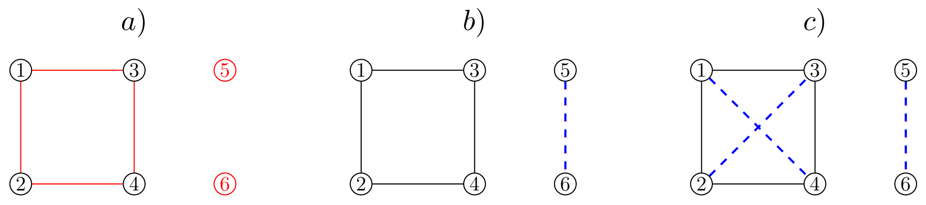

Figure 7: An example that SSC+ and SE do not correspond on $H =$ *co-diamond*. Red edges and vertices are part of a *co-diamond*, blue dashed edges are (weak) inserted edges and blue dotted are weak labeled / deleted edges.

In Fig. 7, *a*) shows a graph G containing the *co-diamonds* $(1, 2, 5, 6)$, $(1, 3, 5, 6)$, $(3, 4, 5, 6)$, $(2, 4, 5, 6)$. For the SSC+ solution in *b*), adding the weak edge $\{5, 6\}$ and labeling all other edges strong provides an optimal solution of size 1. For the SE in *c*), adding the edge $\{5, 6\}$ would create the *co-diamonds* $(5, 6, 2, 3)$ and $(5, 6, 1, 4)$. There are at least three edge modifications in SE required to solve all *co-diamonds*.

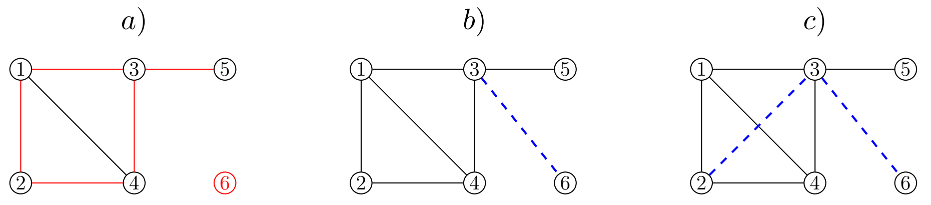

Figure 8: An example that SSC+ and SE do not correspond on H = *co-paw*. Red edges and vertices are part of a *co-paw*, blue dashed edges are (weak) inserted edges and blue dotted are weak labeled / deleted edges.

In Fig. 8, *a*) shows a graph G containing the *co-paws* $(2, 1, 3, 6)$, $(2, 4, 3, 6)$, $(4, 3, 5, 6)$, $(1, 3, 5, 6)$. For the SSC+ solution in *b*), adding the weak edge $\{3, 6\}$ and labeling all other edges strong provides an optimal solution of size 1. For the SE in *c*), adding the edge $\{3, 6\}$ would create the *co-paw* $(5, 3, 6, 2)$. There are at least two edge modifications in SE required to solve all *co-paws*.

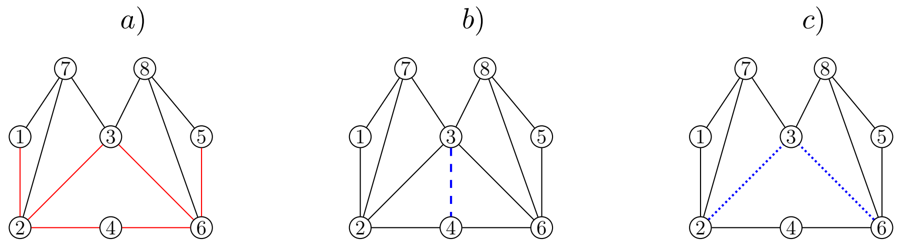

Figure 9: An example that SSC+ and SE do not correspond on H = *claw*. Red edges are part of a *claw*, blue dashed edges are (weak) inserted edges and blue dotted are weak labeled / deleted edges.

In Fig. 9, *a*) shows a graph G containing the *claws* $(2, 1, 3, 4)$ and $(3, 4, 5, 6)$. For the SSC+ solution in *b*), adding the weak edge $\{3, 4\}$ and labeling all other edges strong provides an optimal solution of size 1. For the SE in *c*), adding the edge $\{3, 4\}$ would create the *claw* $(7, 8, 3, 4)$. There are at least two edge modifications in SE required to solve all *claws*.

8 Conclusion and Outlook

Using a polynomial-time reduction of 3SAT to Cluster Editing and showing the correspondence of CE, CD, STC and STC+ on C_4- and *diamond*-free graphs, we proved that STC+ is NP-hard, even when restricted restricted to $C4$- and *diamond*-free graphs with maximum vertex degree six.

Furthermore, we provided a 3-approximation algorithm for STC+ by using a polynomial-time reduction of STC+ to 3-Hitting Set.

Using the concept of critical cliques, we provided the first linear vertex kernel for STC+. Furthermore we provided an $\mathcal{O}(\ell \cdot 2^{\ell})$-vertex kernel for the parameterization by ℓ.

To give an FPT-algorithm, we developed a basic branching algorithm with the search tree size of $\mathcal{O}(3^k)$ and refined the branching strategies leading to a search tree size of $\mathcal{O}(2.57^k)$. Moreover we showed that there is a fixed-parameter algorithm for STC+ running in $2.076^k \cdot n^{\mathcal{O}(1)}$ time, by using a polynomial-time reduction to 3HS.

During our work the question arose, whether STC+ and CE correspond. This was considered for the first time, when we developed the $4k$-vertex kernel for STC+, where the basic idea is very similar to the one for the $4k$-vertex kernel of CE shown in [13]. Since we were not able to prove or disprove the correspondence of STC+ and CE, we analyzed the correspondence of the generalizations of the two problems, Strong Subgraph Closure with Edge Insertion and Subgraph-free Editing, for other graphs H of three or four vertices than P_3s. We showed that SSC+ and SE correspond for the subgraphs $H \in \{K_r, rK_1\}$ for $r \in \mathbb{N}$, where the correspondence is quite trivial. For $H \in \{P_4$, *diamond*, *co-diamond*, C_4, $2K_2$, *paw*, *co-paw*, *claw*, *co-claw*$\}$, the problems do not correspond. Thus, for subgraphs on three or four vertices only for the cases P_3 and its complement $K_1 + K_2$ it is left to show whether SSC+ and SC correspond or not.

For any graph $G = (V, E)$ and a non-negative integer k, any CE solution (E^+, E^-) easily provides an STC+ solution $(L = (S_L, W_L), E')$ with $E' := E^+$, $W_L := E^+ \cup E^-$ and $S_L := E \setminus E^-$. Hence, the two problems would correspond if there is an optimal STC+ solution that provides in some way a CE solution. An STC+ solution would provide a CE solution with $E^+ := E'$ and $E^- := W_L \setminus E'$, if for all vertices $u, v, w \in V$ it holds:

- If $G[\{u, v, w\}]$ is a P_3 in G, then if the missing edge is inserted, either none of the other two edges is in W_L, or both are in W_L.
- If without loss of generality only u and v are adjacent, then:
  - If $\{u, v\} \in S_L$, either none of the two missing edges is in E', or both are in E'.
  - If $\{u, v\} \in W_L$, then at most one of the two missing edges is in E'.

- If $G[\{u,v,w\}]$ is a K_3, then at least two edges connecting the three vertices are in W_L, or all three are in S_L.
- If $G[\{u,v,w\}]$ is a $3K_1$, then at most one of the two missing edges is in E', or all three are in E'.

In Observation 1, we showed that in any optimal STC+ solution $(L = (S_L, W_L), E')$ the unaffected vertices lie in critical cliques consisting only of unaffected vertices. Therefore, any unaffected vertex cannot be part of a vertex set that violates the conditions above. Thus, it remains to show that there exists an optimal STC+ solution such that the affected vertices, which are at most $2k$ many, are not part of a vertex set that violates the conditions above.

A further task is the analysis of STC WITH STRONG EDGE INSERTION, where inserted edges are not labeled weak but strong, so they could be part of a strong P_3. For a graph $G = (V, E)$ and an integer $k \geq 0$, the task is to find an additional edge set E' and an STC-labeling of $G + E'$ such that $E' \subseteq S_L$ and $|W_L| + |E'| \leq k$. If STC+ and CE correspond in the way as discussed above, then STC WITH STRONG EDGE INSERTION and STC+ should also correspond, since then there would be an optimal STC+ solution to every graph G such that the inserted edges are not part of any induced P_3.

Furthermore, one could define and analyze for MULTI-STC, introduced by Sintos and Tsaparas [24] and analyzed by Bulteau et. al. [3], an extending problem, MULTI-STC WITH EDGE INSERTION.

References

[1] Sebastian Böcker. A golden ratio parameterized algorithm for cluster editing. *J. Discrete Algorithms*, 16:79–89, 2012.

[2] Danah Boyd and Nicole B. Ellison. Social network sites: Definition, history, and scholarship. *J. Computer-Mediated Communication*, 13(1):210–230, 2007.

[3] Laurent Bulteau, Niels Grüttemeier, Christian Komusiewicz, and Manuel Sorge. Your rugby mates don't need to know your colleagues: Triadic closure with edge colors. In Pinar Heggernes, editor, *Algorithms and Complexity - 11th International Conference, CIAC 2019, Rome, Italy, May 27-29, 2019, Proceedings*, volume 11485 of *Lecture Notes in Computer Science*, pages 99–111. Springer, 2019.

[4] Jianer Chen and Jie Meng. A 2k kernel for the cluster editing problem. *J. Comput. Syst. Sci.*, 78(1):211–220, 2012.

[5] Marek Cygan, Fedor V. Fomin, Lukasz Kowalik, Daniel Lokshtanov, Dániel Marx, Marcin Pilipczuk, Michal Pilipczuk, and Saket Saurabh. *Parameterized Algorithms.* Springer, 2015.

[6] Fedor V. Fomin and Dieter Kratsch. *Exact Exponential Algorithms.* Texts in Theoretical Computer Science. An EATCS Series. Springer, 2010.

[7] Yong Gao, Donovan R. Hare, and James Nastos. The cluster deletion problem for cographs. *Discret. Math.*, 313(23):2763–2771, 2013.

[8] Eric Gilbert. Predicting tie strength in a new medium. In Steven E. Poltrock, Carla Simone, Jonathan Grudin, Gloria Mark, and John Riedl, editors, *CSCW '12 Computer Supported Cooperative Work, Seattle, WA, USA, February 11-15, 2012*, pages 1047–1056. ACM, 2012.

[9] Eric Gilbert and Karrie Karahalios. Predicting tie strength with social media. In Dan R. Olsen Jr., Richard B. Arthur, Ken Hinckley, Meredith Ringel Morris, Scott E. Hudson, and Saul Greenberg, editors, *Proceedings of the 27th International Conference on Human Factors in Computing Systems, CHI 2009, Boston, MA, USA, April 4-9, 2009*, pages 211–220. ACM, 2009.

[10] Petr A. Golovach, Pinar Heggernes, Athanasios L. Konstantinidis, Paloma T. Lima, and Charis Papadopoulos. Parameterized aspects of strong subgraph closure. In David Eppstein, editor, *16th Scandinavian Symposium and Workshops on Algorithm Theory, SWAT 2018, June 18-20, 2018, Malmö, Sweden*, volume 101 of *LIPIcs*, pages 23:1–23:13. Schloss Dagstuhl - Leibniz-Zentrum für Informatik, 2018.

[11] Mark S. Granovetter. The strength of weak ties. *American Journal of Sociology*, 78(6):1360–1380, 1973.

[12] Niels Grüttemeier and Christian Komusiewicz. On the relation of strong triadic closure and cluster deletion. *Algorithmica*, 82(4):853–880, 2020.

[13] Jiong Guo. A more effective linear kernelization for cluster editing. *Theor. Comput. Sci.*, 410(8-10):718–726, 2009.

[14] Indika Kahanda and Jennifer Neville. Using transactional information to predict link strength in online social networks. In Eytan Adar, Matthew Hurst, Tim Finin, Natalie S. Glance, Nicolas Nicolov, and Belle L. Tseng, editors, *Proceedings of the Third International Conference on Weblogs and Social Media, ICWSM 2009, San Jose, California, USA, May 17-20, 2009.* The AAAI Press, 2009.

[15] Jon M. Kleinberg and Éva Tardos. *Algorithm design.* Addison-Wesley, 2006.

[16] Charlotte Knorr. *Holland/Leinhardt (1971): Transitivity in Structural Models of Small Groups*, pages 267–270. 01 2019.

[17] Christian Komusiewicz and Johannes Uhlmann. Cluster editing with locally bounded modifications. *Discret. Appl. Math.*, 160(15):2259–2270, 2012.

[18] Athanasios L. Konstantinidis, Stavros D. Nikolopoulos, and Charis Papadopoulos. Strong triadic closure in cographs and graphs of low maximum degree. *Theor. Comput. Sci.*, 740:76–84, 2018.

[19] Athanasios L. Konstantinidis and Charis Papadopoulos. Maximizing the strong triadic closure in split graphs and proper interval graphs. In Yoshio Okamoto and Takeshi Tokuyama, editors, *28th International Symposium on Algorithms and Computation, ISAAC 2017, December 9-12, 2017, Phuket, Thailand*, volume 92 of *LIPIcs*, pages 53:1–53:12. Schloss Dagstuhl - Leibniz-Zentrum für Informatik, 2017.

[20] Dániel Marx. Parameterized complexity and approximation algorithms. *Comput. J.*, 51(1):60–78, 2008.

[21] Silvio Micali and Vijay V. Vazirani. An $\mathcal{O}(\sqrt{|V|} \cdot |e|)$ algorithm for finding maximum matching in general graphs. In *21st Annual Symposium on Foundations of Computer Science, Syracuse, New York, USA, 13-15 October 1980*, pages 17–27. IEEE Computer Society, 1980.

[22] Rolf Niedermeier. *Invitation to Fixed-Parameter Algorithms.* Oxford University Press, 2006.

[23] Ron Shamir, Roded Sharan, and Dekel Tsur. Cluster graph modification problems. *Discret. Appl. Math.*, 144(1-2):173–182, 2004.

[24] Stavros Sintos and Panayiotis Tsaparas. Using strong triadic closure to characterize ties in social networks. In Sofus A. Macskassy, Claudia Perlich, Jure Leskovec, Wei Wang, and Rayid Ghani, editors, *The 20th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, KDD '14, New York, NY, USA - August 24 - 27, 2014*, pages 1466–1475. ACM, 2014.

[25] Johan Ugander, Brian Karrer, Lars Backstrom, and Cameron Marlow. The anatomy of the facebook social graph. *CoRR*, abs/1111.4503, 2011.

[26] Vijay V. Vazirani. *Approximation algorithms.* Springer, 2001.

[27] Magnus Wahlström. *Algorithms, measures and upper bounds for satisfiability and related problems.* PhD thesis, Linköping University, Sweden, 2007.

[28] Stanley Wasserman and Katherine Faust. *Social network analysis - methods and applications*, volume 8 of *Structural analysis in the social sciences.* Cambridge University Press, 2007.

[29] Rongjing Xiang, Jennifer Neville, and Monica Rogati. Modeling relationship strength in online social networks. In Michael Rappa, Paul Jones, Juliana Freire, and Soumen Chakrabarti, editors, *Proceedings of the 19th International Conference on World Wide Web, WWW 2010, Raleigh, North Carolina, USA, April 26-30, 2010*, pages 981–990. ACM, 2010.