A Linear-Time Parameterized Algorithm for Node Unique Label Cover*

Daniel Lokshtanov[1], M. S. Ramanujan[2], and Saket Saurabh[3]

1 **University of Bergen, Bergen, Norway**
daniello@ii.uib.no

2 **Algorithms and Complexity Group, TU Wien, Vienna, Austria**
ramanujan@ac.tuwien.ac.at

3 **University of Bergen, Bergen, Norway; and**
The Institute of Mathematical Sciences, Chennai, India
saket@imsc.res.in

Abstract

The optimization version of the Unique Label Cover problem is at the heart of the Unique Games Conjecture which has played an important role in the proof of several tight inapproximability results. In recent years, this problem has been also studied extensively from the point of view of parameterized complexity. Chitnis et al. [FOCS 2012, SICOMP 2016] proved that this problem is fixed-parameter tractable (FPT) and Wahlström [SODA 2014] gave an FPT algorithm with an improved parameter dependence. Subsequently, Iwata, Wahlström and Yoshida [SICOMP 2016] proved that the *edge* version of Unique Label Cover can be solved in *linear* FPT-time, and they left open the existence of such an algorithm for the *node* version of the problem. In this paper, we resolve this question by presenting the first linear-time FPT algorithm for Node Unique Label Cover.

1998 ACM Subject Classification G.2.1 Combinatorics, F.2.2 Nonnumerical Algorithms and Problems

Keywords and phrases Algorithms and data structures, Fixed Parameter Tractability, Unique Label Cover, Linear Time FPT Algorithms

Digital Object Identifier 10.4230/LIPIcs.ESA.2017.57

1 Introduction

In the Unique Label Cover problem we are given an undirected graph G, where each edge $uv = e \in E(G)$ is associated with a permutation $\phi_{e,u}$ of a constant size alphabet Σ. The goal is to construct a labeling $\Psi : V(G) \setminus X \to \Sigma$ maximizing the number of edge constraints, that is, edges for which $(\Psi(u), \Psi(v)) \in \phi_{uv,u}$ holds. For some $\epsilon > 0$ and given Unique Label Cover instance L, Unique Label Cover(ϵ) is the decision problem of distinguishing between the following two cases: (a) there is a labeling Ψ under which at least $(1-\epsilon)|E(G)|$ edges are satisfied; and (b) for every labeling Ψ at most $\epsilon|E(G)|$ edges are satisfied. This problem is at the heart of famous Unique Games Conjecture (UGC) of Khot [27]. Essentially, UGC says that for any $\epsilon > 0$, there is a constant M such that it is NP-hard to decide Unique Label Cover(ϵ) on instances with label set of size M. The

* Daniel Lokshtanov acknowledges support from *Pareto-Optimal Parameterized Algorithms*, ERC Starting Grant 715744. M. S. Ramanujan acknowledges support from FWF, project P26696. Saket Saurabh acknowledges support from *Parameterized Approximation*, ERC Starting Grant 306992.

Unique Label Cover(ϵ) problem over the years has become a canonical problem to obtain tight inapproximability results. We refer the reader to a survey of Khot [28] for more detailed discussion on UGC.

In recent times Unique Label Cover has also attracted a lot of attention in the realm of parameterized complexity. In particular two parameterizations, namely, Edge Unique Label Cover and Node Unique Label Cover have been extensively studied. These problems are, not only, interesting combinatorial problems on its own but they also generalize several well-studied problems in the realm of parameterized complexity. The objective of this paper is to study the following problem.

Node Unique Label Cover **Parameter:** $|\Sigma| + k$
Input: A simple graph G, finite alphabet Σ, integer k and for every edge $e = (u, v) \in E(G)$, permutations $\phi_{e,u}$ and $\phi_{e,v}$ of Σ such that $\phi_{e,u} = \phi_{e,v}^{-1}$ and a function $\tau : V(G) \to 2^{\Sigma}$.
Question: Does there exist a set $X \subseteq V(G)$ and a function $\Psi : V(G) \setminus X \to \Sigma$ such that $|X| \leq k$ and for any $v \in V(G) \setminus X$, for any $(u, v) \in E(G - X)$, we have $(\Psi(u), \Psi(v)) \in \phi_{uv,u}$ where $\Psi(v) \in \tau(v)$ for every $v \in V(G)$?

We remark that the standard formulation of this problem excludes the function τ. However, this formulation is a clear generalization of the standard formulation (simply set $\tau(v) = \Sigma$ for every vertex v) and the way we describe our algorithm makes it notationally convenient to deal with this statement. To make the presentation simpler, we assume that $\Sigma = [|\Sigma|] = \{1, \ldots, |\Sigma|\}$.

The parameterized complexity of the Node Unique Label Cover problem was first studied by Chitnis et al. [5] who proved it is FPT by giving an algorithm running in time $2^{\mathcal{O}(k^2 \cdot \log |\Sigma|)} n^4 \log n$. They complemented this result by proving that an FPT algorithm for this problem parameterized only by k is unlikely to exist. Subsequently, Wahlström [38] (see also [25]) improved the parameter dependence by giving an algorithm running in time $\mathcal{O}(|\Sigma|^{2k} n^{\mathcal{O}(1)})$. The *edge* version of this problem was proved to be solvable in FPT-*linear* time by Iwata et al. [25] who gave an algorithm running in time $\mathcal{O}(|\Sigma|^{2k}(m + n))$. However, their approach does not apply to the much more general *node* version of the problem and they asked whether there is an FPT algorithm for the node version with a linear time dependence on the input size. In this paper, we answer this question in the affirmative by giving a linear time FPT algorithm for this problem. Note that we have stated the problem in a slightly more general form than is usually seen in literature. However, this modification does not affect the solvability of the problem in linear FPT time. We now state our theorem formally.

► **Theorem 1.1.** *There is a $2^{\mathcal{O}(k \cdot |\Sigma| \log |\Sigma|)}(m + n)$ algorithm solving* Node Unique Label Cover*, where m and n are the number of edges and vertices respectively in the input graph.*

Not only does our result answer the open question of Iwata et al. [25], when the label set Σ is of constant-size for some fixed constant, our algorithm also achieves optimal asymptotic dependence on the budget k under the Exponential Time Hypothesis [22].

By its very nature, the Node Unique Label Cover problem is a problem about breaking various types of dependencies between vertices. Since these dependencies are propagated along edges, it is reasonable to view the problem as breaking these dependencies by hitting appropriate sets of paths in the graph. Chitnis et al. [5] used this idea to argue that highly connected pairs of vertices will always remain dependent on each other and hence one can recursively solve the problem by first designing an algorithm for graphs that are 'nearly' highly connected and then use this algorithm as a base case in a divide and conquer type approach. However, the polynomial dependence of their algorithm is $\mathcal{O}(n^4 \log n)$ where n is the number of vertices in the input. Subsequently, Wahlström [38] improved the parameter dependence by using a branching algorithm based on the solution to a specific linear program. However, since this algorithm requires solving linear programs, the dependence on the input is far from linear. Iwata et al. [25] showed that for several special kinds of LP-relaxations, including those involved in the solution of the *edge* version of Unique Label Cover, the corresponding linear program can be solved in linear-time using flow-based techniques and hence they were able to obtain the first linear-time FPT algorithm for the edge version of Unique Label Cover. However, their approach fails when it comes to the *node* version of this problem.

Our Techniques. In this paper, we view the Node Unique Label Cover problem as a problem of hitting paths between certain pairs of vertices in an appropriately designed *auxiliary* graph H whose size is greater than that of the input graph G by a factor depending only on the parameter. The high level road map for the solution follows those in the algorithms developed for solving graph separation problems via important separators in [31, 4], the LP-guided branching in [14, 6, 29, 23], the Valued CSP-based algorithms in [38, 25], the skew-symmetric branching algorithm for 2-SAT Deletion in [33] and most recently, the branching algorithm for the *edge* version of Group Feedback Vertex Set [32]. We show that for any prescribed labeling on the vertices of G, it is possible to select (in linear time) a constant-size set of vertices of G such that after guessing the intersection of this set with a hypothetical solution, if we augment the labeling by branching over all permitted labelings of the remaining vertices in this set then we reduce a pre-determined measure of the input which depends only on the parameter. By repeatedly doing this, we obtain a branching algorithm for this problem where each step requires linear time. The main technical content of the paper is in proving that

(a) there exists a constant-size vertex set and an appropriate measure for the instance such that the measure 'improves' in each step of the branching and
(b) such a vertex set can be computed in linear time.

Related work on improving dependence on input size in FPT algorithms. Our algorithm for Node Unique Label Cover belongs to a large body of work where the main goal is to design linear time algorithms for NP-hard problems for a fixed value of k. That is, to design an algorithm with running time $f(k) \cdot \mathcal{O}(|I|)$, where $|I|$ denotes the size of the input instance. This area of research predates even parameterized complexity. The genesis of parameterized complexity is in the theory of graph minors, developed by Robertson and Seymour [35, 36, 37]. Some of the important algorithmic consequences of this theory include $\mathcal{O}(n^3)$ algorithms for Disjoint Paths and $\mathcal{F}$-Deletion for every fixed values of k. These results led to a whole new area of designing algorithms for NP-hard problems with as small dependence on the input size as possible; resulting in algorithms with improved dependence on the input size for Treewidth [1, 2], FPT approximation for Treewidth [3, 34], Planar $\mathcal{F}$-Deletion [1, 2, 8, 10, 9], and Crossing Number [11, 12, 19], to name a few.

The advent of parameterized complexity started to shift the focus away from the running time dependence on input size to the dependence on the parameter. That is, the goal became designing parameterized algorithms with running time upper bounded by $f(k)n^{\mathcal{O}(1)}$, where the function f grows as slowly as possible. Over the last two decades researchers have tried to optimize one of these objectives, but rarely both at the same time. More recently, efforts have been made towards obtaining linear (or polynomial) time parameterized algorithms that compromise as little as possible on the dependence of the running time on the parameter k. The gold standard for these results are algorithms with linear dependence on input size as well as provably optimal (under ETH) dependence on the parameter. New results in this direction include parameterized algorithms for problems such as Odd Cycle Transversal [24, 33], Subgraph Isomorphism [7], Planarization [26, 15], Subset Feedback Vertex Set [30] as well as a single-exponential and linear time parameterized constant factor approximation algorithm for Treewidth [3]. Other recent results include parameterized algorithms with improved dependence on input size for a host of problems [13, 16, 17, 18, 20, 21].

2 Preliminaries

We fix a label set Σ and assume that all instances of Node Unique Label Cover we deal with are over this label set. When we refer to a set X being a *solution* for a given instance of Node Unique Label Cover, we implicitly assume that X is a set of *minimum* size. We denote the set of functions $\{\phi_{e,u}\}_{e\in E(G),u\in e}$ simply as ϕ (without any subscript).

Before we proceed to describe our algorithm for Node Unique Label Cover, we make a few remarks regarding the representation of the input. We assume that the input graph is given in the form of an adjacency list and for every edge $e = (u, v)$ the permutations $\phi_{e,v}$ and $\phi_{e,u}$ are included in the two nodes of the adjacency list corresponding to the edge e. This is achieved by representing the permutations as $|\Sigma|$-length arrays over the elements in $[|\Sigma|]$. It is straightforward to check that given the input to Label Cover in this form, the decision version of the problem can be solved in time $\mathcal{O}(|\Sigma|^{\mathcal{O}(1)}(m + n))$. We assume that the input to Node Unique Label Cover is also given in the same manner.

3 Setting up the tools

3.1 Defining the auxiliary graph

▶ **Definition 3.1.** Let (G, k, ϕ, τ) be an instance of Node Unique Label Cover and let $\Psi : V(G) \to \Sigma$. We say that Ψ is a **feasible labeling** for this instance if for all $(u, v) \in E(G)$, $(\Psi(u), \Psi(v)) \in \phi_{uv,u}$. For $\tau : V(G) \to 2^{\Sigma}$, we say that Ψ is **consistent with** τ if for every $v \in V(G)$, $\Psi(v) \in \tau(v)$.

For an instance $I = (G, k, \phi, \tau)$ of Node Unique Label Cover, we define an associated auxiliary graph H_I as follows. The vertex set of H_I is $V(G) \times \Sigma$. For notational convenience, we denote the vertex (v, i) by v_i. The vertex v_i is meant to represent the (eventual) labeling of v by the label i. The edge set of H_I is defined as follows. For every edge $e = (u, v)$ and for every $i \in \Sigma$, we have an edge $(u_i, v_{\phi_{e,u}(i)})$. That is, we add an edge between u_i and u_j where j is the image of i under the permutation $\phi_{e,u}$.

We now prove certain structural lemmas regarding this auxiliary graph which will be used in the design as well as analysis of our algorithm. For ease of description, we will treat instances of Label Cover as instances of Node Unique Label Cover. To be precise, we represent an instance (G, ϕ) of Label Cover as the trivially equivalent instance $(G, 0, \phi, \tau^0)$ of Node Unique Label Cover where, $\tau^0(v) = \Sigma$ for every $v \in V(G)$. The first observation follows from the definition of H_I and the fact that since G is a simple graph, for every edge $e \in E(G)$, the set of edges in H_I that correspond to this edge form a matching.

▶ **Observation 3.2.** *Let $I = (G, 0, \phi, \tau)$ be an instance of* Node Unique Label Cover*. Then, for every $v \in V(G)$, for every distinct $i, j \in \Sigma$, v_i and v_j have no common neighbors in H_I.*

▶ **Observation 3.3.** *Let $I = (G, 0, \phi, \tau)$ be a* Yes *instance of* Node Unique Label Cover *and let Ψ be a feasible labeling for this instance. Let $v \in V(G)$ and $i = \Psi(v)$. Then, for any vertex $u \in V(G)$ and $j \in \Sigma$, if u_j is in the same connected component as v_i in H_I then $\Psi(u) = j$.*

The above observation describes the 'dependency' between pairs of vertices which are in the same connected component of G. Moving forward, we will characterize the dependencies between vertices when subjected to additional constraints.

▶ **Definition 3.4.** Let $I = (G, k, \phi, \tau)$ be an instance of Node Unique Label Cover. For $v \in V(G)$, we use $[v]$ to denote the set $\{v_1, \ldots, v_{|\Sigma|}\}$. For a subset $S \subseteq V(G)$, we use $[S]$ to denote the set $\bigcup_{v \in S}[v]$. Similarly, for $e = (u, v) \in E(G)$, we use $[e]$ to denote the set $\big\{(u_i, v_j)\big\}_{i \in \Sigma, j = \phi_{e,u}(i)}$ of edges and for a subset $X \subseteq E(G)$, we use $[X]$ to denote the set $\bigcup_{e \in X}[e]$. For the sake of convenience, we also reuse the same notation in the following way. For $v \in V(G)$ and $\alpha \in \Sigma$, we also use $[v_\alpha]$ to denote the set $\{v_1, \ldots, v_{|\Sigma|}\}$. This definition extends in a natural way to sets of vertices and edges of the auxiliary graph H_I. Finally, for a set $S \subseteq V(H_I) \cup E(H_I)$, we denote by S^{-1} the set $\{s | s \in V(G) \cup E(G) : [s] \cap S \neq \emptyset\}$.

▶ **Definition 3.5.** Let $I = (G, k, \phi, \tau)$ be an instance of Node Unique Label Cover. We say that a set $Z \subseteq V(H_I) \cup E(H_I)$ is **regular** if $|Z \cap [v]| \leq 1$ for any $v \in V(G)$ and $|Z \cap [e]| \leq 1$ for any $e \in V(G)$ and **irregular** otherwise. That is, regular sets contain at most 1 copy of any vertex and edge of G.

Now that we have defined the notion of regularity of sets, we prove the following lemma which shows that the auxiliary graph displays a certain symmetry with respect to regular paths. This will allow us to transfer arguments which involve a regular path between vertices v_i and u_j to one between vertices v_{i_1} and u_{j_1} where $i \neq i_1$ and $j \neq j_1$.

▶ **Lemma 3.6.** *Let $I = (G, k, \phi, \tau)$ be an instance of* Node Unique Label Cover. *Let P be a regular path in H_I from v_i to u_j. Let $V(P)$ denote the set of vertices of G in P and let U denote the set $[V(P)]$. Then, there are vertex disjoint paths $P_1, \ldots, P_{|\Sigma|}$ in H_I and a partition of U into sets $U_1, \ldots, U_{|\Sigma|}$ such that for each $r \in [|\Sigma|]$, $V(P_r) = U_r$ and P_r is a path from v_{i_1} to u_{i_2} for some $i_1, i_2 \in \Sigma$.*

In the next lemma, we describe additional structural properties of the auxiliary graph. In particular, we establish the relation between various copies of the same vertex set. Intuitively, the following lemma says that for every connected and regular set of vertices Z, simply observing the set $N[Z]$ can allow one to make certain useful assertions about the set of vertices in the neighborhood of the set $Z' = [Z] \setminus Z$. Note that for a graph H and set $Z \subseteq V(H)$, we use $N_H[Z]$ and $N_H(Z)$ to denote the closed and open neighborhoods of Z in H respectively. If H is clear from the context, then we drop the subscript.

▶ **Lemma 3.7.** *Let $Z \subseteq V(H_I)$ be a connected regular set of vertices and let $Y = N(Z)$. Further, suppose that $N[Z]$ is regular. Let $Z' = [Z] \setminus Z$ and $Y' = [Y] \setminus Y$. Then, $Y' \subseteq N(Z') \subseteq [Y]$. Furthermore, for every connected component C in $H_I[Z']$, $N(C) \cap [v] \neq \emptyset$ for every $v \in V(G)$ for which there is a $j \in \Sigma$ such that $v_j \in Y$.*

Using the observations and structural lemmas proved so far, we will now give a forbidden-structure characterization of Yes instances of Node Unique Label Cover.

▶ **Lemma 3.8.** *Let $I = (G, 0, \phi, \tau)$ be a* Yes *instance of* Node Unique Label Cover *where G is connected. Let $v \in V(G)$ and $i \in \Sigma$. Then, there is a feasible labeling Ψ such that $\Psi(v) = i$ if and only if there is no $j \in \Sigma$ such that v_i and v_j are in the same connected component of H_I.*

So far, we have studied the structure of YES instances of this problem when the budget $k = 0$. The next lemma is a direct consequence of Lemma 3.8 and allows us to characterize YES instances of the problem for values of k greater than 0.

▶ **Lemma 3.9.** *Let $I = (G, k, \phi, \tau)$ be an instance of* NODE UNIQUE LABEL COVER. *Then, I is a* YES *instance if and only if there is a set $S \subseteq V(G)$ of at most k vertices such that for every $v \in V(G) \setminus S$, there is an $i_v \in \Sigma$ such that $[S]$ intersects all paths from v_{i_v} to v_j for every $\Sigma \ni j \neq i_v$ in the graph H_I. Moreover if there is a feasible labeling for $G - S$ consistent with τ that labels v with the label $i \in \tau(v)$ then for every $u \in V(G)$ and $j \in \Sigma \setminus \tau(u)$, $[S]$ intersects all v_i-u_j paths.*

Using the above lemma, we will interpret the NODE UNIQUE LABEL COVER problem as a parameterized cut-problem and use separator machinery to design a linear-time FPT algorithm for this problem.

3.2 Defining the associated cut-problem

We begin by recalling standard definitions of separators in undirected graphs.

▶ **Definition 3.10.** Let G be a graph and X and Y be disjoint vertex sets. A set S disjoint from $X \cup Y$ is said to be an X-Y **separator** if there is no X-Y path in the graph $G - S$. We denote the vertices in the components of $G - S$ which intersect X by $R(X, S)$ and we denote by $R[X, S]$ the set $R(X, S) \cup S$. We say that an X-Y separator S_1 **covers** an X-Y separator S_2 if $R(X, S_1) \supseteq R(X, S_2)$.

▶ **Definition 3.11.** Let I be an instance of NODE UNIQUE LABEL COVER and let X and Y be disjoint vertex sets of H_I. We say that a minimal X-Y separator S is **good** if the set $R[X, S]$ is regular and **bad** otherwise.

Note that if S is a minimal X-Y separator then $N(R(X, S)) = S$. We are now ready to prove the *Persistence* Lemma which plays a major role in the design of the algorithm. In essence this lemma says that if we are guaranteed the existence of a solution whose deletion leaves a graph with a feasible labeling Ψ and if we are given a vertex v excluded from the deletion set which has a single label α in its allowed label set, then we can define a set T such that the solution under consideration *must* separate v_α from T. Furthermore, if we find a *good* minimum v_α-T separator S, then we can correctly fix the labels of all vertices which have exactly one copy in $R(v_\alpha, S)$. It will be shown later that once we fix the labels of these vertices, the subsequent exhaustive branching steps will decrease a pre-determined measure of the input instance.

▶ **Lemma 3.12** (Persistence Lemma). *Let $I = (G, k, \phi, \tau)$ be a* YES *instance of* NODE UNIQUE LABEL COVER. *Let $X \subseteq V(G)$ be a minimal set of size at most k such that $G - X$ has a feasible labeling and let Ψ be a feasible labeling for $G - X$ consistent with τ. Let v be a vertex not in X with $|\tau(v)| = 1$ and let $\alpha \in \Sigma$ be such that $\alpha = \Psi(v)$ and $\tau(v) = \{\alpha\}$. Let T denote the set $\bigcup_{u \in V(G)} \bigcup_{\gamma \in \Sigma \setminus \tau(u)} \{u_\gamma\}$.*

- *$[X]$ is a v_α-T separator in H_I.*
- *Let S be a good v_α-T minimum separator in H_I and let $Z = R(v_\alpha, S)$. Then, there is a solution for the given instance disjoint from Z^{-1}.*

Proof. The first statement follows from Lemma 3.9. We now prove the second statement. We begin by observing that T contains the set $[v] \setminus \{v_\alpha\}$. This is because $\tau(v)$ is a singleton and only contains the label α. As a result, we know that the set $[X]$ must intersect all v_α-v_β

paths for $\alpha \neq \beta$. Let X_1 denote the set $X \cap Z^{-1}$. If X_1 is empty then we are already done. Therefore, $X_1 \neq \emptyset$. Let S' denote the subset of $S \setminus [X]$ which is not reachable from v_α in the graph $H_I - [X]$ via paths whose internal vertices lie in Z. We now have 2 cases depending on S' being empty or non-empty. We will argue that the first case cannot occur since it contradicts the minimality of X. In the second case we use very similar arguments but show that we can modify X to get an alternate solution X' which is disjoint from the set Z.

Case 1: S' is empty. That is, every vertex in $S \setminus [X]$ is reachable from v_α in $H_I - [X]$ via paths whose internal vertices lie in Z. Let $u \in X_1$ and let $b \in \Sigma$ such that $u_b \in Z$. Since Z is regular, $Z \cap [u]$ must in fact be equal to $\{u_b\}$. We now claim that $X' = X \setminus \{u\}$ is also a set such that $G - X'$ has a feasible labeling, contradicting the minimality of X.

Suppose that this is not the case. That is, $G - X'$ does not have a feasible labeling. Since every connected component of $G - X'$ which does not contain u is also a connected component of $G - X$, all such components do have a feasible labeling. Indeed any feasible labeling of $G - X$ restricted to the vertices in these components is a feasible labeling for these components. Therefore, there is a single component in $G - X'$ which does not have a feasible labeling – the component containing u.

By Lemma 3.8, if there is no $b' \in \Sigma \setminus \{b\}$ such that the connected component of $H_I - [X']$ containing u_b also contains $u_{b'}$, then there is a feasible labeling of the component of $G - X'$ which contains u, a contradiction. Therefore, there is a $b' \in \Sigma \setminus \{b\}$ such that there is a u_b-$u_{b'}$ path in $H_I - [X']$. If this path contains vertices of $[u]$ other than u_b and $u_{b'}$, then we pick the vertex of $[u] \setminus \{u_b\}$ which is closest to u_b on this path and call it $u_{b'}$. Therefore, the path P from u_b to $u_{b'}$ is internally disjoint from $[u]$. We now have the following claim regarding P.

▶ **Claim 3.13.** *The path P is internally regular.*

We now return to the proof of the first case. Since $u_b \in Z$ and $u_{b'} \notin Z$ (as $N[Z]$ is regular), P must intersect $N(Z)$ which is the same as S, in $S \setminus [X]$. Furthermore, P must intersect $N(C)$ where C is the connected component of $Z' = [Z] \setminus Z$ containing the vertex $u_{b'}$. We now have the following 2 subcases based on the intersection of P with the(not necessarily non-empty) set $S \cap N(C)$. In both subcases we will demonstrate the presence of a v_α-v_β path in $H_I - [X]$ for some $\beta \in \Sigma \setminus \{\alpha\}$.

Case 1.1: *P contains a vertex in $S \cap N(C)$.* Let w_ℓ be a vertex in $S \cap N(C)$ which appears in P. We let P_1 denote the subpath of P from u_b to w_ℓ and P_2 denote the subpath of P from w_ℓ to $u_{b'}$. Furthermore, since P is internally regular, P_1 and P_2 are regular. We apply Lemma 3.6 to the regular path P_2 to get a path P_2' with u_b as one endpoint and w_h as the other endpoint, where $w_h \neq w_\ell$. Now, since $W_\ell \in N[Z]$ and $N[Z]$ is regular by our assumption, it must be the case that $w_h \notin Z$. Therefore the path P_2' must intersect S at a vertex other than w_ℓ. Let x_r be such a vertex, where $x \in V(G)$ and $r \in \Sigma$. However, in the case we are in, we know that x_r (which is contained in $S \setminus [X]$) is reachable from v_α in $H_I - [X]$ by a path Q whose internal vertices lie in Z. We let the subpath of P_2' from x_r to w_h be denoted by J. Furthermore, the case we are in guarantees that w_ℓ is reachable from v_α in $H_I - [X]$ via a path L whose internal vertices lie in Z. Since L lies completely in $N[Z]$, it is regular and we may apply Lemma 3.6 on this path to obtain a path L' with w_h s one endpoint and v_β as the other endpoint for some $\beta \in \Sigma$. Since we have already argued that $w_h \neq w_\ell$, it follows that $\beta \neq \alpha$. Therefore, we get a concatenated walk $Q + J + L'$ which is a walk that is present in the graph $H_I - [X]$ and contains v_α and v_β, contradicting the premise of the lemma that there is a feasible labeling for $G - X$ setting v to α. This completes the argument for this subcase.

Case 1.2: *P does not contain a vertex in $S \cap N(C)$.* Let x_r be the last vertex of S which is encountered when traversing P from u_b to $u_{b'}$ and let w_ℓ be the last vertex of $N(C)$ encountered in the same traversal. Observe that since the previous subcase does not hold, it must be the case that x_r occurs before w_ℓ in this traversal. We let J denote the subpath of P between x_r and w_ℓ. Now, Lemma 3.7 implies that there is a $h \in \Sigma \setminus \{\ell\}$ such that $w_h \in S$. This is because $N(C) \subseteq [S]$. Now, the case we are in guarantees the presence of paths L and Q from v_α to w_h and x_r respectively such that L and Q both lie strictly inside $N[Z]$ and hence are regular. Now, we apply Lemma 3.6 on the regular path J to get a path J' with w_h as one endpoint and x_{r_1} as the other for some $r_1 \in \Sigma$. Since we have already argued that $w_h \neq w_\ell$, it must be the case that $r_1 \neq r$. Now, we apply Lemma 3.6 on the regular path Q to get a path Q' with x_{r_1} as one endpoint and v_β as the other for some $\beta \in \Sigma$. Since we have shown that $r_1 \neq r$, we infer that $\beta \neq \alpha$. Now, the concatenated walk $L + J' + Q'$ implies the presence of a v_α-v_β path in $H_I - [X]$, a contradiction to the premise of the lemma. This completes the argument for this subcase.

Thus we have concluded that $G - X'$ has a feasible labeling, contradicting the minimality of X. This completes the argument for the first case.

Case 2: S' is non-empty. Let $\mathcal{Q}$ be a set of $|S|$-many v_α-S paths contained entirely in $N[Z]$ which are vertex disjoint except for the vertex v_α. Since S is a minimum v_α-T separator, such a set of paths exists. Recall that X_1 denotes the set $X \cap Z^{-1}$. We let $\hat{X}_1$ denote the set $[X] \cap Z$. That is, those copies of X_1 present in Z. Due to the presence of the set of paths $\mathcal{Q}$ and the fact that v is disjoint from X, it must be the case that $\hat{X}_1$ contains at least one vertex in each path in $\mathcal{Q}$ that connects v and S'. Furthermore, since S is a good separator, we conclude that $|X_1| = |(\hat{X}_1)^{-1}| \geq |(S')^{-1}|$. We now claim that $X' = (X \setminus X_1) \cup (S')^{-1}$ is also a solution for the given instance. That is, $|X'| \leq |X|$ and $G - X'$ has a feasible labeling. By definition, $|X'| \leq |X|$ holds. Therefore, it remains to prove that $G - X'$ has a feasible labeling.

Again, it must be the case that any connected component of $G - X'$ which does not have a feasible labeling must intersect the set X_1. Any other component of $G - X'$ is contained in a component of $G - X$ and already has a feasible labeling by the premise of the lemma.

By Lemma 3.8, there must be a vertex $u^1 \in X_1$ and distinct labels $b, b' \in \Sigma$ such that $u^1_b \in Z$ and there is a $u^1_b - u^1_{b'}$ path P in $H_I - [X']$. We now consider the intersection of P with the set $[X_1]$ and let p_{γ_1} and q_{γ_2} be vertices on P such that $p_{\gamma_1}, q_{\gamma_2} \in [Z]$, the subpath of P from p_{γ_1} to q_{γ_2} is internally disjoint from $[X_1]$ *and* $p_{\gamma_1} \in Z$ and $q_{\gamma_2} \notin Z$. We first argue that such a pair of vertices exist.

We begin by setting $p_{\gamma_1} = u^1_b$ and $q_{\gamma_2} = u^1_{b'}$. If the path P is already internally disjoint from $[X_1]$ then we are done. Otherwise, let u^2_c be the vertex of $[X_1]$ closest to p_{γ_1} along the subpath between p_{γ_1} and q_{γ_2}. Now, if u^2_c is not in Z then we are done by setting $q_{\gamma_2} = u^2_c$. Otherwise, we continue by setting $p_{\gamma_1} = u^2_c$. Since this process must terminate, we conclude that the vertices p_{γ_1} and q_{γ_2} with the requisite properties must exist.

For ease of notation we will now refer to the path between p_{γ_1} and q_{γ_2} as P. Note that by definition, P is internally disjoint from $[X_1]$. We now have a claim identical to that in the previous case.

▶ **Claim 3.14.** *The path P is internally regular.*

We now complete the proof of this case. Since $p_{\gamma_1} \in Z$ and $q_{\gamma_2} \in [Z] \setminus Z$, P must intersect $N(Z)$ in $(S \setminus [X]) \setminus S'$. Furthermore, P must also intersect $N(C)$ where C is the connected component of $H_I[Z']$ containing q_{γ_2}, where $Z' = [Z] \setminus Z$. We again consider 2 subcases based on the intersection of the path P with the (not necessarily non-empty) set $N(C) \cap S$.

Case 2.1: *P contains a vertex in $S \cap N(C)$.* Let w_ℓ be a vertex in $S \cap N(C)$ which appears in P. We let P_1 denote the subpath of P from p_{γ_1} to w_ℓ and P_2 denote the subpath of P from w_ℓ to q_{γ_2} . Since P is internally regular, P_1 and P_2 are regular. Furthermore, since $q_{\gamma_2} \notin Z$, there is a $\gamma_3 \in \Sigma \setminus \{\gamma_2\}$ such that $q_{\gamma_3} \in Z$. We now apply Lemma 3.6 on the regular path P_2 to get a path P_2' with q_{γ_3} as one endpoint and w_h as the other, where $h \neq \ell$ since $\gamma_2 \neq \gamma_3$. Furthermore, since $w_\ell \in N[Z]$ and $N[Z]$ is regular, it must be the case that $w_h \notin Z$. Therefore the path P_2' must intersect $N(Z)$ at a vertex x_r. Let J be the subpath of P_2' from x_r to w_h. Now, since $x_r \in (S \setminus [X]) \setminus S'$, we know that there is a v_α-x_r path in $H_I - [X]$ which lies entirely in $N[Z]$. Let Q be such a path. Similarly, we know that there is a v_α-w_ℓ path L in $H_I - [X]$ which also lies entirely in $N[Z]$ and hence is regular. We now apply Lemma 3.6 on L to get a path L' with w_h as one endpoint and v_β as the other endpoint for some $\beta \in \Sigma$. Since we have already argued that $w_h \neq w_\ell$, we conclude that $\beta \neq \alpha$. However, the concatenated walk $Q + J + L'$ is present in $H_I - [X]$, implying a v_α-v_β path in $H_I - [X]$, a contradiction to the premise of the lemma. We now address the second subcase under the assumption that this subcase does not occur.

Case 2.2: *P does not contain a vertex in $S \cap N(C)$.* Let x_r be the last vertex of S which is encountered when traversing P from p_{γ_1} to q_{γ_2} and let w_ℓ be the last vertex of $N(C)$ encountered in the same traversal. Since the previous subcase is assumed to not hold, x_r must occur before w_ℓ in this traversal. We let J denote the subpath of P between x_r and w_ℓ. Lemma 3.7 implies the existence of a label $h \in \Sigma \setminus \{\ell\}$ such that $w_h \in S$. This follows from the fact that $N(C) \subseteq [S]$. Also, since w_ℓ occurs in P, w_h is not contained in S' or $[X]$. The same holds for x_r Therefore, the case we are in guarantees the presence of paths L and Q from v_α to w_h and x_r respectively, where L and Q are contained within the set $N[Z]$ and hence they must be regular and amenable to applications of Lemma 3.6. We begin by applying Lemma 3.6 on the regular path J to get a path J' with w_h as one endpoint and x_{r_1} as the other for some $r_1 \in \Sigma$. However, since $h \neq \ell$, we conclude that $r_1 \neq r$. Therefore, we now apply Lemma 3.6 on the path Q to obtain a path Q' with x_{r_1} as one endpoint with the other endpoint being v_β for some $\beta \in \Sigma$. Again, since $r_1 \neq r$, we conclude that $\beta \neq \alpha$. Now, observe that the concatenated walk $L + J' + Q'$ implies the presence of a v_α-v_β path in $H_I - [X]$, a contradiction to the premise of the lemma. This completes the argument for this subcase as well and consequentially that for Case 2.

We have thus proved that Case 1 cannot occur at all and in Case 2, there is an exchange argument which constructs an alternate solution X' which is disjoint from Z. This completes the proof of the lemma. ◀

The main consequence of the above lemma is that at any point in the run of our algorithm solving an instance $I = (G, k, \phi, \tau)$, if there is a vertex v whose label is 'fixed', i.e. $\tau(v) = \{\alpha\}$ for some $\alpha \in \Sigma$ and there is a good v_α-T separator S where T is defined as in the premise of the above lemma, then we can correctly 'fix' the labelings of all vertices in the set $(R(v_\alpha, S))^{-1}$. That is, we can define a new function τ' as follows. For every $u \in V(G)$ and $\gamma \in \Sigma$, we set $\tau'(u) = \{\gamma\}$ if $u_\gamma \in R(v_\alpha, S)$ and $\tau'(u) = \tau(u)$ otherwise. Lemma 3.12 implies that the given graph has a deletion set of size at most k which leaves a graph with a feasible labeling consistent with τ if and only if the graph has deletion set of size at most k which leaves a graph with a feasible labeling consistent with τ'.

3.3 Computing good separators

▶ **Lemma 3.15.** *Let $I = (G, k, \phi, \tau)$ be an instance of* Node Unique Label Cover*, v be a vertex in G and let $\alpha \in \Sigma$. Let T_v^α denote the set $[v] \setminus \{v_\alpha\}$ and $T \supseteq T_v^\alpha$ be a set not containing v_α. There is an algorithm that, given I, v, α, and T runs in time $\mathcal{O}(|\Sigma| \cdot k(m+n))$ and either*

- *correctly concludes that there is no v_α-T separator of size at most $|\Sigma| \cdot k$ or*
- *returns a pair of minimum v_α-T separators S_1 and S_2 such that S_2 covers S_1, S_1 is good, S_2 is bad and for any vertex $u \in R(v_\alpha, S_2) \setminus R[v_\alpha, S_1]$, the size of a minimal v_α-T separator containing u is at least $|S_1| + 1$ or*
- *returns a good minimum v_α-T separator S such that no other minimum v_α-T separator covers S or*
- *correctly concludes that there is no good v_α-T minimum separator.*

▶ **Lemma 3.16.** *Let $I = (G, k, \phi, \tau)$ be an instance of* Node Unique Label Cover*, v be a vertex in G, $\alpha \in \Sigma$, $T \supseteq [v] \setminus \{v_\alpha\}$ be a set not containing v_α and let $\ell > 0$ be the size of a minimum v_α-T separator in H_I. Let S_1 and S_2 be a pair of minimum v_α-T separators such that S_1 is good, S_2 is bad, and for any vertex $y \in R(v_\alpha, S_2) \setminus R[v_\alpha, S_1]$, the size of a minimal v_α-T separator containing y is at least $\ell + 1$. Let $u \in V(G)$ and $\gamma_1, \gamma_2 \in \Sigma$ such that $u_{\gamma_1}, u_{\gamma_2} \in R[v_\alpha, S_2]$. Then,*

1. *$R[v_\alpha, S_2]$ contains a pair of paths P_1 and P_2 such that for each $i \in \{1, 2\}$, the path P_i is a v_α-u_{γ_i} path and both paths are internally vertex disjoint from S_2 and contain at most one vertex of S_1.*
2. *Given I, v_α, S_1 and S_2, there is an algorithm that, in time $\mathcal{O}(|\Sigma| \cdot k(m + n))$, computes a pair of paths with the above properties.*
3. *For $i \in \{1, 2\}$, any minimum v_α-$T \cup \{u_{\gamma_i}\}$ separator disjoint from $V(P_i) \cap (S_1 \cup S_2)$ and $R(v_\alpha, S_1)$ has size at least $\ell + 1$, where ℓ is the size of a minimum v_α-T separator.*

We are now ready to prove Theorem 1.1 by describing our algorithm for Node Unique Label Cover. Before doing so, we make the following important remark regarding the way we use the algorithms described in this subsection. In the description of our main algorithm, there will be points where we make a choice to *not* delete certain vertices. That is, we will choose to exclude them from the solution being computed. At such points, we say that we make these vertices *undeletable*.

All the above algorithms also work when given an undeletable set of vertices in the graph and the minimum separators we are looking for are the minimum *among* those separators disjoint from the undeletable set of vertices. Regarding the running time of these algorithms, there will be a multiplicative factor of $|\Sigma| \cdot k$ which arises due to potentially blowing up the size of the graph by a factor of $|\Sigma| \cdot k$ by making $(|\Sigma| \cdot k) + 1$ copies of every undeletable vertex.

4 The Linear time algorithm for Node Unique Label Cover

Before we describe our algorithm, we state certain assumptions we make regarding the input. We assume that at any point, we are dealing with a connected graph G. Furthermore, we assume that instances of Node Unique Label Cover are given in the form of a tuple – $(G, k, \phi, \tau, w^*, V^\infty)$ where the element w^* denotes either a vertex from $V(G)$ or it is undefined. If w^* denotes a vertex then, $|\tau(w^*)| = 1$ and we will attempt to solve the problem on the tuple $(G, k, \phi, \tau, w^*, V^\infty)$ under the assumption that w^* is not in the solution (which is required to be disjoint from V^∞). Furthermore the definition of the problem allows us to assume that if there is a feasible labeling for this instance (after deleting a solution) then there is one consistent with τ. Since $\tau(w^*)$ is singleton, any feasible labeling consistent with τ must set w^* to the unique label in $\tau(w^*)$.

We first check if G already has a feasible labeling (not necessarily one consistent with τ). If so, then we are done. If not and $k = 0$ then we return No. If any connected component of G has a feasible labeling then we remove this component. Otherwise, we check if w^* is defined. If w^* is undefined, then we pick an arbitrary *deletable* vertex $v \in V(G)$. That is $v \notin V^\infty$. We then recursively solve the problem on the instances $I_{q_0}, \dots, I_{q_r}$ where $\{q_1, \dots, q_r\} = \tau(v)$ and for each q_i where $i \geq 1$, the instance I_{q_i} is defined to be $(G, k, \phi, \tau_{v=q_i}, w^*, V_1^\infty)$ with $\tau_{v=q_i}$ defined as the function obtained from τ by restricting the image of v to the singleton set $\{q_i\}$, w^* defined as $w^* = v$ and V_1^∞ defined as $V_1^\infty = V^\infty \cup \{v\}$. The instance I_{q_0} is defined as $(G - \{v\}, k-1, \phi', \tau', w^*, V^\infty)$ where ϕ' and τ' are restrictions of ϕ and τ to the graph $G - \{v\}$. This will be the only branching rule which has a branching factor depending on the parameter (in this case the size of the label set Σ) and we call this rule, $\mathbf{B_0}$.

We now describe the steps executed by the algorithm in the case when w^* is defined. Suppose that $w^* = v$, $\tau(v) = \alpha$. Recall that by our assumption regarding well-formed inputs, if w^* is defined then $\tau(w^*)$ must be a singleton set. We set $T = \bigcup_{u \in V(G)} \bigcup_{\gamma \in \Sigma \setminus \tau(u)} u_\gamma$. Intuitively, T is the set of all vertices u_γ such that if there is a feasible labeling of G (after deleting the solution) which sets v to α then it cannot be consistent with τ unless the solution hits all paths in H_I (where I is the given instance) between v_α and u_γ. We remark that since T depends only on the input instance I, we use $T(I)$ to denote the set T corresponding to any input instance I. Once we set T as described we first check if there is a v_α-T path in H_I. If not, then the algorithm deletes the component of G containing v and recurses by setting w^* to be undefined. The correctness of this operation is argued as follows. Observe that T contains all vertices of $[v] \setminus \{v_\alpha\}$ and excludes v_α. Therefore, Lemma 3.8 implies that the component of G containing v already has a feasible labeling and hence can be removed.

Otherwise if there is a v_α-T path in H_I, then we execute the algorithm of Lemma 3.15 with this definition of v, α and T and undeletable set $[V^\infty]$. Observe that T contains all vertices of $[v] \setminus \{v_\alpha\}$ but excludes v_α. This is because $\tau(v) = \{\alpha\}$. The next steps of our algorithm depend on the output of this subroutine. For each of the four possible outputs, we describe an exhaustive branching.

Case 1: *The subroutine returns that there is no v_α-T separator of size at most $|\Sigma| \cdot k$ which is disjoint from $[V^\infty]$.* In this case, our algorithm returns No. The correctness of this step follows from Lemma 3.12.

Case 2: *The subroutine returns a good v_α-T separator S which is smallest among all v_α-T separators disjoint from $[V^\infty]$ such that no other v_α-T separator disjoint from $[V^\infty]$ and having the same size as S, covers S.* In this case, we do the following. For each vertex u_γ in the set $R(v_\alpha, S)$ where $u \in V(G)$ and $\gamma \in \Sigma$, we set $\tau(u) = \{\gamma\}$ and add u to V^∞. That is, we set $V^\infty = V^\infty \cup (R(v_\alpha, S))^{-1}$. Note that prior to this operation, $\gamma \in \tau(u)$ since otherwise u_γ would belong to T. We then pick an arbitrary vertex $x_\delta \in S$ and recursively solve the problem on 2 instances I_1 and I_2 defined as follows. The instance I_1 is defined to be $(G - \{x\}, k-1, \phi', \tau', V^\infty)$ where ϕ' and τ' are restrictions of ϕ and τ to $G - \{x\}$. The instance I_2 is defined to be $(G, k, \phi, \tau', V_1^\infty)$ where $V_1^\infty = V^\infty \cup \{x\}$ and τ' is defined to be the same as τ on all vertices but x and $\tau'(x) = \{\delta\}$. We call this branching rule, $\mathbf{B_1}$. The exhaustiveness of this branching step follows from the fact that once the vertices in $(R(v_\alpha, S)^{-1})$ are made undeletable, unless the vertex x is deleted, Observation 3.3 forces any feasible labeling that labels v with α to label x with δ.

Case 3: *The subroutine correctly concludes that there is no good v_α-T separator which is also smallest among all v_α-T separators disjoint from* $[V^\infty]$. In this case, we compute S, the minimum v_α-T separator that is disjoint from V^∞ and closest to v_α. Since S is not good, $R[v_\alpha, S]$ contains a pair of vertices u_{γ_1} and u_{γ_2} for some $u \in V(G)$ and $\gamma_1, \gamma_2 \in \Sigma$. Furthermore, since S is a v_α-T separator, it must be the case that u_{γ_1} and u_{γ_2} are not in T. This implies that $\{\gamma_1, \gamma_2\} \subseteq \tau(u)$. We now recursively solve the problem on 3 instances I_0, I_1, I_2 defined as follows. The instance I_0 is defined as $(G - \{u\}, k-1, \phi', \tau', w^*, V^\infty)$, where ϕ' and τ' are defined as the restrictions of ϕ and τ to the graph $G - \{u\}$. The instance I_1 is defined as $(G, k, \phi, \tau', w^*, V_1^\infty)$ where $V_1^\infty = V^\infty \cup \{u\}$ and τ' is defined to be be the same as τ on all vertices but u and $\tau'(u) = \tau(u) \setminus \{\gamma_1\}$. Similarly, the instance I_2 is defined as $(G, k, \phi, \tau', w^*, V_1^\infty)$ where $V_1^\infty = V^\infty \cup \{u\}$ and τ' is defined to be be the same as τ on all vertices but u and $\tau'(u) = \tau(u) \setminus \{\gamma_2\}$. We call this branching rule $\mathbf{B_2}$.

The exhaustiveness of this branching follows from the fact that if u is not deleted (the first branch) then any feasible labeling of $G - X$ for a hypothetical solution X must label u with at most one label out of γ_1 and γ_2. Therefore, if I is a YES instance then for at least one of the 2 instances I_1 or I_2, there is a feasible labeling of $G - X$ consistent with the corresponding τ'.

Case 4: Finally, we address the case when the subroutine returns a pair of minimum (among those disjoint from $[V^\infty]$) v_α-T separators S_1 and S_2 such that S_2 covers S_1, S_1 is good, S_2 is bad and there is no minimum (among those disjoint from V^∞) v_α-T separator which covers S_1 and is covered by S_2. In this case, $R[v_\alpha, S_2]$ contains a pair of vertices u_γ, u_δ for some vertex $u \in V(G)$ and $\gamma, \delta \in \Sigma$.

We execute the algorithm of Lemma 3.16 to compute in time $\mathcal{O}(|\Sigma| \cdot k(m+n))$, a v_α-u_γ path P_1 and a v_α- u_δ path P_2 such that both paths are internally vertex disjoint from S_2 and contain at most one vertex of S_1 each. Let $x^1, x^2 \in V(G)$ and $\beta_1, \beta_2 \in \Sigma$ be such that $x^1_{\beta_1}$ and $x^2_{\beta_2}$ are the vertices of S_1 in P_1 and P_2 respectively. Note that P_1 or P_2 may be disjoint from S_1. If P_i ($i \in \{1,2\}$) is disjoint from S_1 then we let $x^i_{\beta_i}$ be undefined. We now recurse on the following (at most) 5 instances $I_1, \ldots, I_5$ defined as follows.

- $I_1 = (G - x^1, k-1, \phi', \tau', w^*, V^\infty)$ where ϕ' and τ' are restrictions of ϕ and τ to $G - \{x^1\}$.
- $I_2 = (G - x^2, k-1, \phi', \tau', w^*, V^\infty)$ where ϕ' and τ' are restrictions of ϕ and τ to $G - \{x^2\}$.
- $I_3 = (G - u, k-1, \phi', \tau', w^*, V^\infty)$ where ϕ' and τ' are restrictions of ϕ and τ to $G - \{u\}$.
- $I_4 = (G, k, \phi, \tau', w^*, V_1^\infty)$ where $V_1^\infty = V^\infty \cup (R(v_\alpha, S_1))^{-1} \cup \{x^1\}$ and τ' is the same as τ on all vertices of G except u and $\tau'(u) = \tau(u) \setminus \{\gamma\}$.
- $I_5 = (G, k, \phi, \tau', w^*, V_1^\infty)$ where $V_1^\infty = V^\infty \cup (R(v_\alpha, S_1))^{-1} \cup \{x^2\}$ and τ' is the same as τ on all vertices of G except u and $\tau'(u) = \tau(u) \setminus \{\delta\}$.

This branching rule is called $\mathbf{B_3}$ and we now argue the exhaustiveness of the branching. The first three branches cover the case when the solution intersects the set $\{x^1, x^2, u\}$. Suppose that a hypothetical solution, say X, is disjoint from $\{x^1, x^2, u\}$. By Lemma 3.12, we may assume that X is disjoint from $R(v_\alpha, S_1)$. Since any feasible labeling of $G - X$ sets u to at most one of $\{\gamma_1, \gamma_2\}$, branching into 2 cases by excluding γ_1 from $\tau(u)$ in the first case and excluding γ_2 from $\tau(u)$ in the second case gives us an exhaustive branching. This completes the description of the algorithm. The correctness follows from the exhaustiveness of the branchings. We will now prove the running time bound.

Analysis of running time. It follows from the description of the algorithm and the bounds already proved on the running time of each subroutine, that each step can be performed in time $\mathcal{O}((\Sigma + k)^{\mathcal{O}(1)}(m+n))$. Therefore, we only focus on bounding the number of nodes in the search tree resulting from this branching algorithm. In order to analyse this number, we introduce the following measure for the instance $I = (G, k, \phi, \tau, w^*, V^\infty)$ corresponding to any node of the search tree. We define $\mu(I) = (\Sigma + 1)k - \lambda(I)$ where $\lambda(I)$ is $\lambda(w^*, T(I))$ if w^* is defined and 0 otherwise.

Note that $\lambda(w^*, T(I))$ denotes the size of the smallest w^*-$T(I)$ separator in H_I among those disjoint from $[V^\infty]$. Furthermore, observe that $\mu(I) \leq (|\Sigma| + 1) \cdot k$ for any instance on which the algorithm can potentially branch. We now argue that this measure strictly decreases in each branch of every branching rule and since the number of branches in any branching rule is bounded by *max* $\{|\Sigma| + 1, 5\}$ (Rules $\mathbf{B_0}$ and $\mathbf{B_3}$), the time bound claimed in the statement of Theorem 1.1 follows.

References

1 Hans L. Bodlaender. A linear time algorithm for finding tree-decompositions of small treewidth. In *Proceedings of the Twenty-Fifth Annual ACM Symposium on Theory of Computing, May 16-18, 1993, San Diego, CA, USA*, pages 226–234, 1993.

2 Hans L. Bodlaender. A linear-time algorithm for finding tree-decompositions of small treewidth. 25(6):1305–1317, 1996.

3 Hans L. Bodlaender, Pål Grønås Drange, Markus S. Dregi, Fedor V. Fomin, Daniel Lokshtanov, and Michal Pilipczuk. An $O(c^k n)$ 5-approximation algorithm for treewidth. In *FOCS*, pages 499–508, 2013. `doi:10.1109/FOCS.2013.60`.

4 Jianer Chen, Yang Liu, and Songjian Lu. An improved parameterized algorithm for the minimum node multiway cut problem. *Algorithmica*, 55(1):1–13, 2009. `doi:10.1007/s00453-007-9130-6`.

5 Rajesh Hemant Chitnis, Marek Cygan, MohammadTaghi Hajiaghayi, Marcin Pilipczuk, and Michal Pilipczuk. Designing FPT algorithms for cut problems using randomized contractions. In *53rd Annual IEEE Symposium on Foundations of Computer Science, FOCS 2012, New Brunswick, NJ, USA, October 20-23, 2012*, pages 460–469, 2012. `doi:10.1109/FOCS.2012.29`.

6 Marek Cygan, Marcin Pilipczuk, Michal Pilipczuk, and Jakub Onufry Wojtaszczyk. On multiway cut parameterized above lower bounds. *TOCT*, 5(1):3, 2013. `doi:10.1145/2462896.2462899`.

7 Frederic Dorn. Planar subgraph isomorphism revisited. In *STACS*, pages 263–274, 2010.

8 Michael R. Fellows and Michael A. Langston. Nonconstructive tools for proving polynomial-time decidability. *J. ACM*, 35(3):727–739, 1988. `doi:10.1145/44483.44491`.

9 Fedor V. Fomin, Daniel Lokshtanov, Neeldhara Misra, M. S. Ramanujan, and Saket Saurabh. Solving d-sat via backdoors to small treewidth. In *Proceedings of the Twenty-Sixth Annual ACM-SIAM Symposium on Discrete Algorithms, SODA 2015, San Diego, CA, USA, January 4-6, 2015*, pages 630–641, 2015.

10 Fedor V. Fomin, Daniel Lokshtanov, Neeldhara Misra, and Saket Saurabh. Planar f-deletion: Approximation, kernelization and optimal FPT algorithms. In *53rd Annual IEEE Symposium on Foundations of Computer Science, FOCS 2012, New Brunswick, NJ, USA, October 20-23, 2012*, pages 470–479, 2012.

11 Martin Grohe. Computing crossing numbers in quadratic time. In *Proceedings on 33rd Annual ACM Symposium on Theory of Computing, July 6-8, 2001, Heraklion, Crete, Greece*, pages 231–236, 2001.

12 Martin Grohe. Computing crossing numbers in quadratic time. *J. Comput. Syst. Sci.*, 68(2):285–302, 2004.

13 Martin Grohe, Ken ichi Kawarabayashi, and Bruce A. Reed. A simple algorithm for the graph minor decomposition – logic meets structural graph theory. In *SODA*, pages 414–431, 2013. doi:10.1137/1.9781611973105.30.

14 Sylvain Guillemot. FPT algorithms for path-transversal and cycle-transversal problems. *Discrete Optimization*, 8(1):61–71, 2011.

15 Ken ichi Kawarabayashi. Planarity allowing few error vertices in linear time. In *FOCS*, pages 639–648, 2009. doi:10.1109/FOCS.2009.45.

16 Ken ichi Kawarabayashi, Yusuke Kobayashi, and Bruce A. Reed. The disjoint paths problem in quadratic time. *J. Comb. Theory, Ser. B*, 102(2):424–435, 2012. doi:10.1016/j.jctb.2011.07.004.

17 Ken ichi Kawarabayashi and Bojan Mohar. Graph and map isomorphism and all polyhedral embeddings in linear time. In *STOC*, pages 471–480, 2008. doi:10.1145/1374376.1374443.

18 Ken ichi Kawarabayashi, Bojan Mohar, and Bruce A. Reed. A simpler linear time algorithm for embedding graphs into an arbitrary surface and the genus of graphs of bounded tree-width. In *FOCS*, pages 771–780, 2008. doi:10.1109/FOCS.2008.53.

19 Ken ichi Kawarabayashi and Bruce A. Reed. Computing crossing number in linear time. In *STOC*, pages 382–390, 2007. doi:10.1145/1250790.1250848.

20 Ken ichi Kawarabayashi and Bruce A. Reed. A nearly linear time algorithm for the half integral parity disjoint paths packing problem. In *SODA*, pages 1183–1192, 2009. doi:10.1145/1496770.1496898.

21 Ken ichi Kawarabayashi and Bruce A. Reed. An (almost) linear time algorithm for odd cycles transversal. In *SODA*, pages 365–378, 2010.

22 Russell Impagliazzo, Ramamohan Paturi, and Francis Zane. Which problems have strongly exponential complexity? *Journal of Computer and System Sciences*, 63(4):512–530, 2001.

23 Yoichi Iwata, Keigo Oka, and Yuichi Yoshida. Linear-time fpt algorithms via network flow. In *SODA*, pages 1749–1761, 2014. doi:10.1137/1.9781611973402.127.

24 Yoichi Iwata, Keigo Oka, and Yuichi Yoshida. Linear-time fpt algorithms via network flow. In *SODA*, pages 1749–1761, 2014. doi:10.1137/1.9781611973402.127.

25 Yoichi Iwata, Magnus Wahlström, and Yuichi Yoshida. Half-integrality, lp-branching, and FPT algorithms. *SIAM J. Comput.*, 45(4):1377–1411, 2016. doi:10.1137/140962838.

26 Bart M. P. Jansen, Daniel Lokshtanov, and Saket Saurabh. A near-optimal planarization algorithm. In *SODA*, pages 1802–1811, 2014. doi:10.1137/1.9781611973402.130.

27 Subhash Khot. On the power of unique 2-prover 1-round games. In *Proceedings on 34th Annual ACM Symposium on Theory of Computing, May 19-21, 2002, Montréal, Québec, Canada*, pages 767–775, 2002.

28 Subhash Khot. On the unique games conjecture (invited survey). In *Proceedings of the 25th Annual IEEE Conference on Computational Complexity, CCC 2010, Cambridge, Massachusetts, June 9-12, 2010*, pages 99–121, 2010.

29 Daniel Lokshtanov, N. S. Narayanaswamy, Venkatesh Raman, M. S. Ramanujan, and Saket Saurabh. Faster parameterized algorithms using linear programming. *ACM Transactions on Algorithms*, 11(2):15:1–15:31, 2014. doi:10.1145/2566616.

30 Daniel Lokshtanov, M. S. Ramanujan, and Saket Saurabh. Linear time parameterized algorithms for subset feedback vertex set. In *Automata, Languages, and Programming – 42nd International Colloquium, ICALP 2015, Kyoto, Japan, July 6-10, 2015, Proceedings, Part I*, pages 935–946, 2015. doi:10.1007/978-3-662-47672-7_76.

31 Dániel Marx. Parameterized graph separation problems. *Theor. Comput. Sci.*, 351(3):394–406, 2006. doi:10.1016/j.tcs.2005.10.007.

32 M. S. Ramanujan. A faster parameterized algorithm for group feedback edge set. In *Graph-Theoretic Concepts in Computer Science – 42nd International Workshop, WG*

2016, Istanbul, Turkey, June 22-24, 2016, Revised Selected Papers, pages 269–281, 2016. doi:10.1007/978-3-662-53536-3_23.

33 M. S. Ramanujan and Saket Saurabh. Linear time parameterized algorithms via skew-symmetric multicuts. In *SODA*, pages 1739–1748, 2014. doi:10.1137/1.9781611973402.126.

34 B. Reed. Finding approximate separators and computing tree-width quickly. In *Proceedings of the 24th Annual ACM symposium on Theory of Computing (STOC'92)*, pages 221–228. ACM, 1992.

35 Neil Robertson and Paul D. Seymour. Graph minors .xiii. the disjoint paths problem. *J. Comb. Theory, Ser. B*, 63(1):65–110, 1995.

36 Neil Robertson and Paul D. Seymour. Graph minors. XVIII. tree-decompositions and well-quasi-ordering. *J. Comb. Theory, Ser. B*, 89(1):77–108, 2003.

37 Neil Robertson and Paul D. Seymour. Graph minors. XX. wagner's conjecture. *J. Comb. Theory, Ser. B*, 92(2):325–357, 2004.

38 Magnus Wahlström. Half-integrality, LP-branching and FPT algorithms. In *Proceedings of the Twenty-Fifth Annual ACM-SIAM Symposium on Discrete Algorithms, SODA 2014, Portland, Oregon, USA, January 5-7, 2014*, pages 1762–1781, 2014. doi:10.1137/1.9781611973402.128.