AN OPTIMAL XP ALGORITHM FOR HAMILTONIAN CYCLE ON GRAPHS OF BOUNDED CLIQUE-WIDTH

BENJAMIN BERGOUGNOUX, MAMADOU MOUSTAPHA KANTÉ, AND O-JOUNG KWON

Abstract. For MSO_2-expressible problems like Edge Dominating Set or Hamiltonian Cycle, it was open for a long time whether there is an algorithm which given a clique-width k-expression of an n-vertex graph runs in time $f(k) \cdot n^{\mathcal{O}(1)}$ for some function f. Recently, Fomin et al. (*SIAM. J. Computing*, 2014) presented several lower bounds; for instance, there are no $f(k) \cdot n^{o(k)}$-time algorithms for Edge Dominating Set and for Hamiltonian Cycle unless the Exponential Time Hypothesis (ETH) fails. They also provided an algorithm running in time $n^{\mathcal{O}(k)}$ for Edge Dominating Set, but left open whether Hamiltonian Cycle can be solved in time $n^{\mathcal{O}(k)}$.

In this paper, we prove that Hamiltonian Cycle can be solved in time $n^{\mathcal{O}(k)}$. This improves the naive algorithm that runs in time $n^{\mathcal{O}(k^2)}$ by Espelage et al. (WG 2001). We present a general technique of representative sets using two-edge colored multigraphs on k vertices. The essential idea behind is that for a two-edge colored multigraph, the existence of an Eulerian trail that uses edges with different colors alternatively can be determined by two information, that are the number of colored edges incident with each vertex and the number of connected components containing an edge. This allows to avoid storing all possible graphs on k vertices with at most n edges, which gives the $n^{\mathcal{O}(k^2)}$ running time. We can apply this technique to other problems such as q-Cycle Covering or Directed Hamiltonian Cycle as well.

1. Introduction

Tree-width is one of the graph width parameters that plays an important role in graph algorithms. Various problems which are NP-hard on general graphs, have been shown to be solvable in polynomial time on graphs of bounded tree-width [1, 2]. A celebrated algorithmic meta-theorem by Courcelle [3] states that every graph property expressible in monadic second-order logic which allows quantifications over edge and vertex sets (MSO_2) can be decided in linear time on graphs of bounded tree-width. Minimum Dominating Set, q-Coloring, and Hamiltonian Cycle problems are such graph problems.

Courcelle and Olariu [5] defined the notion of *clique-width* of graphs, whose modeling power is strictly stronger than tree-width. The motivation of clique-width came from the observation that many algorithmic problems are tractable on classes of graphs that can be recursively decomposable along vertex partitions (A, B) where the number of neighbourhood types between A and B are small. Courcelle, Makowsky, and Rotics [4] extended the meta-theorem on graphs of bounded tree-width [3] to graphs of bounded clique-width, at a

Date: February 21, 2017.

Key words and phrases. XP-algorithm, Hamiltonian cycle, clique-width.

B. Bergougnoux and M.M. Kanté are supported by French Agency for Research under the GraphEN project (ANR-15-CE-0009). O. Kwon is supported by the European Research Council (ERC) under the European Union's Horizon 2020 research and innovation programme (ERC consolidator grant DISTRUCT, agreement No. 648527).

cost of a smaller set of problems, namely, the class of problems expressible in MSO_1, which allows quantifications on vertex sets only. Some of the known examples of graph problems that are MSO_2-definable, but not MSO_1-definable are Max-Cut, Edge Dominating Set, and Hamiltonian Cycle problems.

A natural question is whether such problems allow an algorithm with running time $f(k)\cdot n^{\mathcal{O}(1)}$ for some function f, when a clique-width k-expression of an n-vertex input graph is given. This question has been carefully answered by Fomin et al. [8, 9]. In particular, they showed that for Max-Cut and Edge Dominating Set, there is no $f(k)\cdot n^{o(k)}$-time algorithm unless the Exponential Time Hypothesis (ETH) fails, and proposed for both problems algorithms with running time $n^{\mathcal{O}(k)}$. They argued that Hamiltonian Cycle also cannot be solved in time $f(k)\cdot n^{o(k)}$, unless ETH fails, but left open the question of finding an algorithm running in time $n^{\mathcal{O}(k)}$. Until now, the best algorithm is the one by Espelage, Gurski, and Wanke [7] which runs in time $n^{\mathcal{O}(k^2)}$.

Our Contribution. In this paper, we prove that Hamiltonian Cycle can be solved in time $n^{\mathcal{O}(k)}$, thereby resolving the open problem in [9]. A *Hamiltonian cycle* in a graph is a cycle containing all vertices of the graph. We formally define clique-width in Section 2.

Hamiltonian Cycle **Input :** A graph G **Question :** Does G have a Hamiltonian cycle?

Theorem 1.1. *Given an n-vertex graph G and its clique-width k-expression, one can solve* Hamiltonian Cycle *in time $n^{\mathcal{O}(k)}$.*

A k-labeled graph is a graph whose vertices are labeled by integers in $\{1,\dots,k\}$. Clique-width k-expressions are expressions which allow to recursively construct a graph with the following graph operations: (1) creating a graph with a single vertex labeled i for some $i\in\{1,\dots,k\}$, (2) taking the disjoint union of two k-labeled graphs, (3) adding all edges between vertices labeled i and vertices labeled j, for $j\neq i$, (4) renaming all vertices labeled i into j. The clique-width of a graph is the minimum k such that it can be constructed using labels in $\{1,\dots,k\}$. One observes that if a graph contains a Hamiltonian cycle C, then each k-labeled graph introduced in the clique-width k-expression admits a partition of its vertex set into pairwise vertex-disjoint paths, *path-partitions*, which is a restriction of the Hamiltonian cycle C. A natural approach is to enumerate all such partitions into paths. Furthermore, since the adjacency relations between this k-labeled graph and the remaining part only depend on the labels, it is sufficient to store for each pair of labels (i,j), the number of paths whose end vertices are labeled by i and j. As the number of paths between two label classes is bounded by n, there are at most $n^{\mathcal{O}(k^2)}$ possible such information. This is the basic idea of the XP algorithm developed by Espelage, Gurski, and Wanke [7].

The essential idea is to introduce an equivalence relation between two path-partitions. Given a path-partition $\mathcal{P}$ that is a restriction of a Hamiltonian cycle C, we consider the maximal paths in $C-\bigcup_{P\in\mathcal{P}}E(P)$ as another path-partition $\mathcal{Q}$. As depicted in Figure 1, we can construct a multigraph associated with $\mathcal{P}$ and $\mathcal{Q}$ on the vertex set $\{v_1,\dots,v_k\}$, by adding a red edge v_iv_j (an undashed edge) if there is a path in $\mathcal{P}$ with end vertices labeled by i and j, and by adding a blue edge v_iv_j (a dashed edge) if there is a path in $\mathcal{Q}$ with end vertices labeled by i and j. A crucial observation is that this multigraph admits

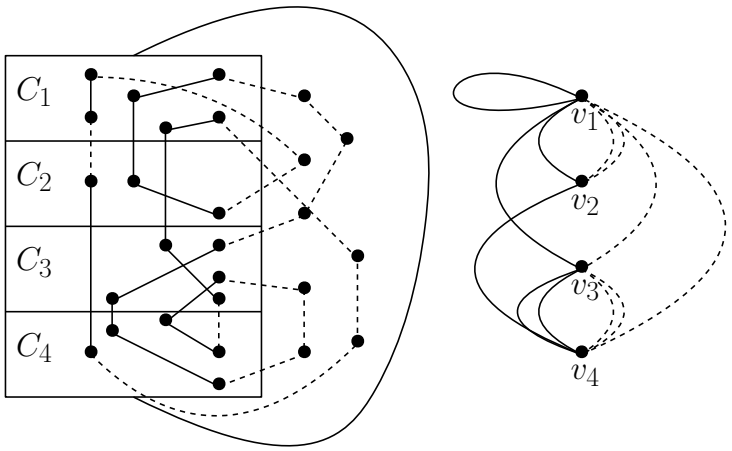

FIGURE 1. The restriction of a Hamiltonian cycle to a k-labeled graph. The complement part can be considered as another set of paths.

an Eulerian trail where red edges and blue edges are alternatively used. This is indeed a characterisation of the fact that two such path-partitions can be joined into a Hamiltonian cycle. To determine the existence of such an Eulerian trail, it is sufficient to know the degree of each vertex and the connected components of the corresponding multigraphs of the two path-partitions. This motivates an equivalence relation between path-partitions. As a byproduct, we can keep in each equivalence class a representative and since the number of equivalence classes is bounded by $2^{k \log(k)} \cdot n^k$, we can turn the naive algorithm into an $n^{\mathcal{O}(k)}$-time algorithm. A more detailed explanation of our algorithm is provided in Section 3, after the required definitions.

The paper is organized as follows. Section 2 contains the necessary preliminaries and required notions. Section 3 is devoted to the overview of the algorithm and the proof of the existence of Eulerian trails in two-edge colored multigraphs. We then introduce in Section 4 the equivalence relation between multigraphs on the vertex set $\{v_1, \ldots, v_k\}$, and introduce operations related to the update of path-partitions in clique-width k-expressions, and prove that they preserve the equivalence relation. We define the notion of representatives in Section 5.1 and give the algorithm in Section 5.2. We conclude with more applications of our notion of representatives in Section 6.

2. Preliminaries

The size of a set V is denoted by $|V|$, and we write $[V]^2$ to denote the set of all subsets of V of size 2. We denote by $\mathbb{N}$ the set of non-negative integers.

We essentially follow [6] for our graph terminology, but we deal only with finite graphs. The vertex set of a graph G is denoted by $V(G)$ and its edge set by $E(G) \subseteq [V(G)]^2$. As usual, we write xy to denote an edge $\{x, y\}$. Let G be a graph. For $X \subseteq V(G)$, we denote by $G[X]$ the subgraph of G induced by X, and for $F \subseteq E(G)$, we write $G - F$ for the subgraph $(V(G), E(G) \setminus F)$. The *degree* of a vertex x, denoted by $\deg_G(x)$, is the number of edges incident with x. For two sets $A, B \subseteq V(G)$, A is *complete* to B if for every $v \in A$ and $w \in B$, $vw \in E(G)$. A *cut-edge* in a connected graph is an edge e such that $G - \{e\}$ is disconnected. We recall that a cycle in a graph G is a *Hamiltonian cycle* if it covers all vertices of G.

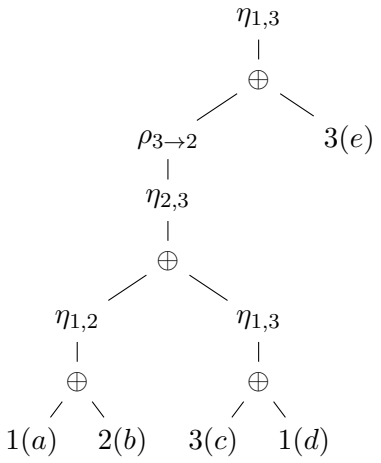

FIGURE 2. 3-expression of C_5.

A graph is *non-trivial* if it contains an edge, otherwise it is said *trivial*. A *walk* of a graph is an alternating sequence of vertices and edges, starting and ending at some vertices, where for every consecutive pair of a vertex x and an edge e, x is incident with e. A *trail* of a graph is a walk where each edge is used at most once. A trail is *closed* if its first and last vertices are the same.

A *multigraph* is essentially a graph, but we allow two edges to be incident with the same set of vertices. Formally, a *multigraph* G is a pair $(V(G), E(G))$ of disjoint sets, also called sets of vertices and edges, respectively, together with a map $\text{mult}_G : E(G) \to V(G) \cup [V(G)]^2$, which maps every edge to one or two vertices, still called its end vertices. Note that we admit loops in multigraphs, while we do not in our definition of graphs. If there is $e \in E(G)$ such that $\text{mult}_G(e) = \{x, y\}$ (or $\text{mult}_G(e) = \{x\}$), we use the term *multiedge* to refer to $\{x, y\}$ (or $\{x\}$). The degree of a vertex x in a multigraph G, is defined analogously as in graphs, except that each loop is counted twice, and similarly for other notions. If there are exactly k edges e such that $\text{mult}_G(e) = \{x, y\}$ (or $\text{mult}_G(e) = \{x\}$), then we denote these distinct edges by $\{x, y\}_1, \ldots, \{x, y\}_k$ (or $\{x\}_1, \ldots, \{x\}_k$); if $k = 1$, then for the sake of clarity, we write $\{x, y\}$ (or $\{x\}$) instead of $\{x, y\}_1$ (or $\{x\}_1$).

An *Eulerian trail* in a multigraph is a closed trail containing all edges.

2.1. **Clique-width.** A graph is *k-labeled* if there is a labeling function $f : V(G) \to \{1, \ldots, k\}$, and we call $f(v)$ the *label* of v. For a k-labeled graph G, we simply call the set of all vertices with label i as the *label class* i of G.

The *clique-width* of a graph G is the minimum number of labels needed to construct G using the following four operations:

(1) Creation of a new vertex v with label i (denoted by $i(v)$).
(2) Disjoint union of two labeled graphs G and H (denoted by $G \oplus H$).
(3) Joining by an edge each vertex with label i to each vertex with label j ($i \neq j$, denoted by $\eta_{i,j}$).
(4) Renaming label i to j (denoted by $\rho_{i\to j}$).

Every graph can be defined by an algebraic expression using these four operations. Such an expression is called a *clique-width k-expression* or shortly a *k-expression* if it uses at most k distinct labels.

For instance, the cycle *abcdea* of length 5 can be constructed using the following 3-expression: $\eta_{1,3}(\rho_{3\to 2}\eta_{2,3}(\eta_{1,2}(1(a)\oplus 2(b))\oplus\eta_{1,3}(3(c)\oplus 1(d)))\oplus 3(e))$. We can represent this expression as a tree-structure, depicted in Figure 2. Such trees are known as *syntactic trees* associated with k-expressions.

A clique-width k-expression is called *irredundant* if whenever $\eta_{i,j}$ is applied, the constructed graph contains no prior edges between vertices with label i and vertices with label j. Courcelle and Olariu [5] proved that given a clique-width k-expression, it can be transformed into an irredundant k-expression in linear time. Therefore, we can assume that a given clique-width expression is irredundant.

2.2. **Path-partition.** For a graph G, a set $\mathcal{P}=\{P_1,\dots,P_m\}$ of pairwise vertex-disjoint paths in G is called a *path-partition* of G if $\bigcup_{1\le i\le m}V(P_i)=V(G)$. A path-partition $\mathcal{P}$ is *k-labeled* if each end vertex of a path in $\mathcal{P}$ is labeled by some integer in $\{1,\dots,k\}$. For a path-partition $\mathcal{P}$ of a k-labeled graph G, the labeling of $\mathcal{P}$ *induced by the labeling of G* consists in assigning to each end vertex of a path in $\mathcal{P}$ its label in G. Lastly, for a k-labeled path-partition $\mathcal{P}$ of a graph, we define the auxiliary multigraph $Aux(\mathcal{P})$ with vertex set $\{v_1,\dots,v_k\}$ and edge set $\bigcup_{i,j\in\{1,\dots,k\}}\{\{v_i,v_j\}_1,\dots,\{v_i,v_j\}_{\ell_{ij}}\}$ where ℓ_{ij} is the number of paths in $\mathcal{P}$ with end vertices labeled i and j respectively.

3. Overview of the algorithm

In an irredundant clique-width k-expression ϕ defining a given graph G, G is recursively constructed using k-labeled graphs. Such k-labeled graphs H arising in the k-expression defining G are subgraphs of G and satisfy the following properties:

(1) for two vertices $v,w\in V(H)$ with same labels in H, $N_G(v)\cap(V(G)\setminus V(H))=N_G(w)\cap(V(G)\setminus V(H))$,
(2) for some label class i, say L_i, and label class j, say L_j, in H with $i\neq j$, if there exist $v\in L_i$, $w\in L_j$ with $vw\in E(G)\setminus E(H)$, then L_i is complete to L_j in G, and there are no edges between L_i and L_j in H.

In (2), the former statement is because when we add vw, all vertices in each set of L_i or L_j have same labels, and the latter statement is because of the irredundancy of ϕ. Given such a k-labeled subgraph H of G, for any Hamiltonian cycle C, the restriction of C to H induces a k-labeled path-partition of H. Because of the two properties (1) and (2), it is sufficient to store the end vertices of each path. This is naturally represented as a multigraph on k vertices, motivating the definition of the auxiliary multigraphs associated with k-labeled path-partitions. Our algorithm will be based on the following characterization of equivalent path-partitions.

We formally prove the following relations between two path-partitions in Section 5.

Proposition 5.2. *Let $\mathcal{P}_1$ and $\mathcal{P}_2$ be k-labeled path-partitions of H whose labelings are induced by the labeling of H. If*

- $(\deg_{Aux(\mathcal{P}_1)}(v_1),\dots,\deg_{Aux(\mathcal{P}_1)}(v_k))=(\deg_{Aux(\mathcal{P}_2)}(v_1),\dots,\deg_{Aux(\mathcal{P}_2)}(v_k))$ *and*
- $\{V(C)\mid C \text{ is a component of } Aux(\mathcal{P}_1)\}=\{V(C)\mid C \text{ is a component of } Aux(\mathcal{P}_2)\}$,

then the following are equivalent.

(1) G has a Hamiltonian cycle C_1 containing each path in $\mathcal{P}_1$ as a subpath and such that every edge in $C_1 - (\bigcup_{P\in\mathcal{P}_1} E(P))$ is contained in $E(G)\setminus E(H)$.
(2) G has a Hamiltonian cycle C_2 containing each path in $\mathcal{P}_2$ as a subpath and such that every edge in $C_2 - (\bigcup_{P\in\mathcal{P}_2} E(P))$ is contained in $E(G)\setminus E(H)$.

Suppose there are two such path-partitions $\mathcal{P}_1$ and $\mathcal{P}_2$, and G has a Hamiltonian cycle C_1 containing each path in $\mathcal{P}_1$ as a subpath and such that every edge in $C_1-(\bigcup_{P\in\mathcal{P}_1} E(P))$ is contained in $E(G)\setminus E(H)$. Let $\mathcal{Q}$ be the subpaths of C_1 connecting two consecutive paths in $\mathcal{P}_1$. See Figure 1 for an illustration, where dashed edges denote the paths of $\mathcal{Q}$. Note that if a path in $\mathcal{Q}$ is an edge, then it is an edge between two label classes, and since ϕ is an irredundant k-expression, these two label classes are completely adjacent to each other.

One can observe that there is a one-to-one correspondence between the end-vertices of $\mathcal{P}_1$ and those of $\mathcal{Q}$. Let us label each end vertex of a path in $\mathcal{Q}$ as the label of H, We consider the auxiliary multigraph $Aux(\mathcal{P}_1)$ and the auxiliary multigraph $Aux(\mathcal{Q})$ by considering $\mathcal{Q}$ as a path-partition of the underlying graph on $\bigcup_{Q\in\mathcal{Q}} V(Q)$. We obtain a multigraph F from the disjoint union of $Aux(\mathcal{P}_1)$ and $Aux(\mathcal{Q})$ by identifying each v_i. Following the Hamiltonian cycle C, one easily checks that there is an Eulerian trail which alternates between edges in $Aux(\mathcal{P}_1)$ and edges in $Aux(\mathcal{Q})$.

We will prove that if we replace $Aux(\mathcal{P}_1)$ with $Aux(\mathcal{P}_2)$ in F, then the new graph also admits an Eulerian trail, because of the given conditions in Proposition 5.2. To see this, we observe the following, which is a strengthen of Euler's theorem on Eulerian trails. It is well known that a connected multigraph contains an Eulerian trail if and only if every vertex has even degree. Moreover, when edges are colored by two colors, say red and blue, and each vertex is incident with the same number of edges for both colors, then we can find an Eulerian trail where the two colors appear alternatively. We call such an Eulerian trail a *red-blue alternating Eulerian trail*. For a multigraph G colored by red and blue and $v\in V(G)$, let $\mathrm{rdeg}_G(v)$ denote the number of red edges incident with v, and let $\mathrm{bdeg}_G(v)$ denote the number of blue edges incident with v.

Lemma 3.1. *Let G be a connected multigraph whose edges are colored by red and blue. Then the following are equivalent. Then G has a red-blue alternating Eulerian trail if and only if for every vertex v, $\mathrm{bdeg}_G(v)=\mathrm{rdeg}_G(v)$.*

Proof. It is clear that if G has a red-blue alternating Eulerian trail, then for every vertex v, $\mathrm{bdeg}_G(v)=\mathrm{rdeg}_G(v)$. We prove the other direction. Let m be the number of edges in G. We choose a red-blue alternating trail $v_1e_1v_2e_2\cdots v_ie_iv_{i+1}$ with maximum i such that e_1 is colored red, and there is exactly one non-trivial component C of $G-\left(\bigcup_{j\in\{1,\ldots,i\}}\{e_j\}\right)$, and C contains v_1 and v_{i+1}, and in $G_i := G-\left(\bigcup_{j\in\{1,\ldots,i\}}\{e_j\}\right)$,

- for each $v\in V(G)\setminus\{v_1,v_{i+1}\}$, $\mathrm{rdeg}_{G_i}(v)=\mathrm{bdeg}_{G_i}(v)$,
- if $v_{i+1}\neq v_1$, and e_i is colored red (resp. blue), then $\mathrm{bdeg}_{G_i}(v_{i+1})=\mathrm{rdeg}_{G_i}(v_{i+1})+1$ (resp. $\mathrm{rdeg}_{G_i}(v_{i+1})=\mathrm{bdeg}_{G_i}(v_{i+1})+1$) and $\mathrm{bdeg}_{G_i}(v_1)=\mathrm{rdeg}_{G_i}(v_1)+1$.
- if $v_{i+1}=v_1$, then $\mathrm{bdeg}_{G_i}(v_1)=\mathrm{rdeg}_{G_i}(v_1)$ or $\mathrm{bdeg}_{G_i}(v_1)=\mathrm{rdeg}_{G_i}(v_1)+2$.

We claim $i=m-1$. Suppose $i<m-1$. Note that there are at least two edges in $E(G)\setminus E(G_i)$.

We observe that there are at most one cut-edge in G_i incident with v_{i+1}. If there are two cut-edges incident with v_{i+1}, then one cut-edge f satisfies that G_i-f has a connected

component C that does not contain v_1 and v_{i+1}. However, all vertices v in C satisfy $\mathrm{rdeg}_{G_i}(v) = \mathrm{bdeg}_{G_i}(v)$ by the assumption, and in particular, they have even degrees, and this is not possible. Therefore, there are at most one cut-edge in G_i incident with v_{i+1}.

Suppose $v_{i+1} \neq v_1$ and e_i is colored red. The proof is symmetric when e_i is colored blue. If there is a blue edge e incident with v_{i+1} in G_i, then we can choose e as a next edge to extend the red-blue alternating trail. Thus, we may assume there is only one edge e incident with v_{i+1} and it is a cut-edge of G_i. By the condition, v_{i+1} is not adjacent to any red edge in G_i, and thus it has degree 1 in G_i. Therefore, $G_i - e$ has only one non-trivial component, and it implies that we can choose e as a next edge to extend the red-blue alternating trail.

Suppose $v_{i+1} = v_1$ and $\mathrm{bdeg}_{G_i}(v_1) = \mathrm{rdeg}_{G_i}(v_1) + 2$. In this case, e_i should be a red edge. Since there are at least two blue edges incident with v_{i+1} in G_i, we can choose a blue edge e that is not a cut-edge. Suppose $v_{i+1} = v_1$ and $\mathrm{bdeg}_{G_i}(v_1) = \mathrm{rdeg}_{G_i}(v_1)$. In this case, e_i should be a blue edge. Note that $\mathrm{rdeg}_{G_i}(v_1) \neq 0$ by the condition that there is at least one non-trivial connected component containing v_1 and v_{i+1}, and thus there is at least one red edge incident with v_{i+1} in G_i. We claim that there is at least one red edge among them that is not a cut-edge. Suppose for contradiction that there is only one red edge incident with v_{i+1} in G_i and it is a cut-edge in G_i. By the assumption, there is exactly one blue edge incident with v_{i+1} in G_i. Since one of two edges incident with v_{i+1} is not a cut-edge, the other one is also not a cut-edge. It contradicts to the assumption. Thus, we can choose a red edge e that is not a cut-edge.

In each case, we can extend the red-blue alternating trail using e, which contradicts to the maximality. Therefore $i = m - 1$, and v_{i+1} should be connected to v_1 by a blue edge $f := v_{i+1}v_1$. Thus, $v_1e_1v_2e_2 \cdots v_ie_iv_{i+1}fv_1$ is a required red-blue alternating Eulerian trail. □

Indeed, when we replace $Aux(\mathcal{P}_1)$ with $Aux(\mathcal{P}_2)$ in F, the set of components does not change (thus consists of one non-trivial component), and each vertex is incident with same number of red and blue edges, and by Lemma 3.1, the resulting graph has an Eulerian trail. We will argue that one can construct a Hamiltonian cycle of G from paths of $\mathcal{P}_2$ using the properties (1) and (2) at the beginning of this section.

Motivated by Proposition 5.2, we define in Section 4 an equivalence relation between two sets of multigraphs on the same vertex set $\{v_1, \ldots, v_k\}$. We further define operations on those multigraphs, corresponding to procedures of updating path-partitions, and prove that the equivalence between two sets is preserved under such operations. These results will form the skeleton of the main algorithm.

4. An equivalence relation between families of k-vertex multigraphs

For two multigraphs G and H on the same vertex set $\{v_1, \ldots, v_k\}$ and with disjoint edge sets, we denote by $G//H$ the multigraph with vertex set $\{v_1, \ldots, v_k\}$ and edge set $E(G) \cup E(H)$.

For families $\mathcal{F}, \mathcal{F}_1, \mathcal{F}_2$ of multigraphs on the vertex set $\{v_1, \ldots, v_k\}$ and two distinct integers $i, j \in \{1, \ldots, k\}$, we define the following operations:

(1) $\mathcal{F} + (i, j)$ is the set of all multigraphs F' where F' can be obtained from a multigraph F in $\mathcal{F}$ as follows[1]: choose two distinct edges $\{v_i, v_i'\}_t$ and $\{v_j, v_j'\}_s$,

[1]We allow v_i' (or v_j') to be equal to v_i (or v_j).

and let F' be the multigraph on vertex set $\{v_1, \dots, v_k\}$ and edge set $(E(F) \setminus \{\{v_i, v_i'\}_t, \{v_j, v_j'\}_s\}) \cup \{e\}$ with $e \notin E(F)$ mapped to $\{v_i', v_j'\}$.

(2) $\mathcal{F} + t(i,j)$ is the set constructed from $\mathcal{F}$ by doing the operation $+(i,j)$ t times.

(3) $\mathcal{F}|_{i \to j}$ is the set of all multigraphs F where F can be obtained from a multigraph in $\mathcal{F}$ by replacing every edge by an end vertex v_i with an edge with an end vertex v_j.

(4) $\mathcal{F}_1 \uplus \mathcal{F}_2 := \{F_1//F_2 : F_1 \in \mathcal{F}_1, F_2 \in \mathcal{F}_2\}$.

Let $\mathcal{F}_1$ and $\mathcal{F}_2$ be two families of multigraphs on the vertex set $\{v_1, \dots, v_k\}$. We write $\mathcal{F}_1 \lesssim \mathcal{F}_2$ if for every multigraph H on the vertex set $\{v_1, \dots, v_k\}$,

- whenever there exists $G_2 \in \mathcal{F}_2$ such that $(\deg_{G_2}(v_1), \dots, \deg_{G_2}(v_k)) = (\deg_H(v_1), \dots, \deg_H(v_k))$ and $G_2//H$ has at most one non-trivial component, there exists $G_1 \in \mathcal{F}_1$ such that $(\deg_{G_1}(v_1), \dots, \deg_{G_1}(v_k)) = (\deg_H(v_1), \dots, \deg_H(v_k))$ and $G_1//H$ has at most one non-trivial component.

We say that $\mathcal{F}_1$ is *equivalent* to $\mathcal{F}_2$, written $\mathcal{F}_1 \equiv \mathcal{F}_2$, if $\mathcal{F}_1 \lesssim \mathcal{F}_2$ and $\mathcal{F}_2 \lesssim \mathcal{F}_1$.

We prove that the equivalence between two families is preserved by the operation $+(i,j)$.

Proposition 4.1. *Let $\mathcal{F}_1$ and $\mathcal{F}_2$ be two families of multigraphs on the vertex set $\{v_1, \dots, v_k\}$. If $\mathcal{F}_1 \equiv \mathcal{F}_2$, then $\mathcal{F}_1 + (i,j) \equiv \mathcal{F}_2 + (i,j)$.*

Proof. Suppose $\mathcal{F}_1 \equiv \mathcal{F}_2$. It is sufficient to prove that $\mathcal{F}_1 + (i,j) \lesssim \mathcal{F}_2 + (i,j)$. For this, suppose there exist a graph H on $\{v_1, \dots, v_k\}$ and $G_2 \in \mathcal{F}_2 + (i,j)$ such that $(\deg_{G_2}(v_1), \dots, \deg_{G_2}(v_k)) = (\deg_H(v_1), \dots, \deg_H(v_k))$ and $G_2//H$ has at most one non-trivial component. Since $G_2 \in \mathcal{F}_2 + (i,j)$, there exist $F_2 \in \mathcal{F}_2$, edges $\{v_i, v_i'\}_t$, $\{v_j, v_j'\}_s$ in F_2 such that $G_2 = (V(F_2), E(F_2) \setminus \{\{v_i, v_i'\}_t, \{v_j, v_j'\}_s\} \cup \{e\})$ with $e \notin E(F_2)$ mapped to $\{v_i', v_j'\}$ in G_2. Let $H' := (V(H), E(H) \cup \{e'\})$ with $e' \notin E(H)$ mapped to $\{v_i, v_j\}$ in H. We claim that

- $(\deg_{F_2}(v_1), \dots, \deg_{F_2}(v_k)) = (\deg_{H'}(v_1), \dots, \deg_{H'}(v_k))$ and
- $F_2//H'$ has at most one non-trivial component.

By the construction of G_2 from F_2, for every $v_\ell \in V(F_2) \setminus \{v_i, v_j\}$, v_ℓ has the same degree in F_2 and G_2, and the degrees of v_i and v_j in G_2 are one less than the degrees in F_2, respectively. Since the degrees of v_i and v_j in H' are one more than the degrees in H, we have $(\deg_{F_2}(v_1), \dots, \deg_{F_2}(v_k)) = (\deg_{H'}(v_1, \dots, \deg_{H'}(v_k))$. Assume now that $F_2//H'$ has at least two non-trivial components. First observe that the four vertices v_i, v_j, v_i', v_j' are in the same non-trivial component C of $F_2//H'$, and $\{v_i, v_i'\}$ are in a same non-trivial component of $G_2//H$. If C' is another non-trivial component of $F_2//H'$, then it does not intersect $\{v_i, v_i', v_j, v_j'\}$, *i.e.*, C' is non-trivial component in $G_2//H$ that does not intersect the one containing $\{v_i, v_i'\}$, yielding a contradiction.

Since $\mathcal{F}_1 \equiv \mathcal{F}_2$, there exists $F_1 \in \mathcal{F}_1$ such that $(\deg_{F_1}(v_1), \dots, \deg_{F_1}(v_k)) = (\deg_{H'}(v_1), \dots, \deg_{H'}(v_k))$ and $F_1//H'$ has at most one non-trivial component. By Lemma 3.1, $F_1//H'$ contains an Eulerian trail where edges in F_1 and edges in H' are alternatively used. Let $\{v_i, v_i''\}_{t'}$ and $\{v_j, v_j''\}_{s'}$ be the edges where $\{v_i, v_i''\}_{t'}, e', \{v_j, v_j''\}_{s'}$ appear in the Eulerian trail in this order (recall that e' is mapped to $\{v_i, v_j\}$). Clearly, if we remove the edges $\{v_i, v_i''\}_{t'}, e', \{v_j, v_j''\}_{s'}$ and add an edge f mapped to $\{v_i'', v_j''\}$ in $F_1//H'$, then the obtained multigraph K still admits an alternating Eulerian trail. Let $G_1 = (V(F_1), E(F_1) \setminus \{\{v_i, v_i''\}_{t'}, \{v_j, v_j''\}_{s'}\} \cup \{f\})$ with $f \notin E(F_1)$ mapped to $\{v_i'', v_j''\}$ in G_1. One easily checks that $K = G_1//H$, and since K has an Eulerian trail where edges in G_1 and edge in H are alternatively used, by Lemma 3.1, $(\deg_{G_1}(v_1), \dots, \deg_{G_1}(v_k)) = (\deg_H(v_1), \dots, \deg_H(v_k))$

and $G_1//H$ has at most one non-trivial component. Because $G_1 \in \mathcal{F}_1 + (i,j)$, we can thus conclude that $\mathcal{F}_1 + (i,j) \lesssim \mathcal{F}_2 + (i,j)$. □

We prove a similar property for the other operations.

Proposition 4.2. *Let $\mathcal{F}_1$ and $\mathcal{F}_2$ be families of multigraphs on the vertex set $\{v_1, \ldots, v_k\}$ and let $i, j \in \{1, \ldots, k\}$ be two distinct integers. If $\mathcal{F}_1 \equiv \mathcal{F}_2$, then $\mathcal{F}_1|_{i\to j} \equiv \mathcal{F}_2|_{i\to j}$.*

Proof. Suppose $\mathcal{F}_1 \equiv \mathcal{F}_2$. It is sufficient to prove that $\mathcal{F}_1|_{i\to j} \lesssim \mathcal{F}_2|_{i\to j}$. Suppose there exist a graph H on $\{v_1, \ldots, v_k\}$ and $G_2 \in \mathcal{F}_2|_{i\to j}$ such that $(\deg_{G_2}(v_1), \ldots, \deg_{G_2}(v_k)) = (\deg_H(v_1), \ldots, \deg_H(v_k))$ and $G_2//H$ has at most one non-trivial component. By Lemma 3.1, there is an Eulerian trail C where edges of G_2 and edges of H are alternatively used. Since $G_2 \in \mathcal{F}_2|_{i\to j}$, there exists $F_2 \in \mathcal{F}_2$ such that G_2 is obtained from F_2 by replacing every edge with an end vertex v_i with an edge with an end vertex v_j. Clearly, we can construct F_2 from G_2 by reversing each of these operations. We construct the multigraph H' from H as follows :

- following the construction of F_2 from G_2, whenever an edge $\{v_\ell, v_j\}_t$ is replaced with $\{v_\ell, v_i\}_s$ for some v_ℓ, we replace the edge $\{v_{\ell'}, v_j\}_t$ with $\{v_{\ell'}, v_i\}_p$ in H where $\{v_\ell, v_j\}_t$ and $\{v_j, v_{\ell'}\}_s$ are consecutive edges in C.

This operation preserves the existence of an alternating Eulerian trail. It implies that $(\deg_{F_2}(v_1), \ldots, \deg_{F_2}(v_k)) = (\deg_{H'}(v_1), \ldots, \deg_{H'}(v_k))$ and $F_2//H'$ also has at most one non-trivial component.

Since $\mathcal{F}_1 \equiv \mathcal{F}_2$, there exists $F_1 \in \mathcal{F}_1$ such that $(\deg_{F_1}(v_1), \ldots, \deg_{F_1}(v_k)) = (\deg_{H'}(v_1), \ldots, \deg_{H'}(v_k))$ and $F_1//H'$ has at most one non-trivial component. In other words, $F_1//H'$ has an Eulerian trial where edges in F_1 and edges in H' are alternatively used. Similar to the previous procedure, we can obtain a graph G_1 form F_1 by following the construction of H from H', such that $G_1 \in \mathcal{F}_1|_{i\to j}$ and $G_1//H$ has an Eulerian trail where edges in G_1 and edges in H are alternatively used. By Lemma 3.1 $(\deg_{G_1}(v_1), \ldots, \deg_{G_1}(v_k)) = (\deg_H(v_1), \ldots, \deg_H(v_k))$ and $G_1//H$ has at most one non-trivial component, and we conclude that $\mathcal{F}_1|_{i\to j} \lesssim \mathcal{F}_2|_{i\to j}$. □

Proposition 4.3. *Let $\mathcal{F}_1, \mathcal{F}_2, \mathcal{F}_3$ be families of multigraphs on the vertex set $\{v_1, \ldots, v_k\}$. If $\mathcal{F}_1 \equiv \mathcal{F}_2$, then $\mathcal{F}_1 \uplus \mathcal{F}_3 \equiv \mathcal{F}_2 \uplus \mathcal{F}_3$.*

Proof. Suppose $\mathcal{F}_1 \equiv \mathcal{F}_2$. It is sufficient to prove that $\mathcal{F}_1 \uplus \mathcal{F}_3 \lesssim \mathcal{F}_2 \uplus \mathcal{F}_3$. Suppose there exist a graph H on $\{v_1, \ldots, v_k\}$ and $G_2 \in \mathcal{F}_2$ and $G_3 \in \mathcal{F}_3$ such that $(\deg_{G_2//G_3}(v_1), \ldots, \deg_{G_2//G_3}(v_k)) = (\deg_H(v_1), \ldots, \deg_H(v_k))$ and $(G_2//G_3)//H$ has at most one non-trivial component. Thus, $(G_2//G_3)//H$ has an Eulerian trail C where edges in $G_2//G_3$ and edges in H are alternatively used. Clearly, $(G_2//G_3)//H = G_2//(G_3//H)$. Now, we obtain a graph H' from $G_3//H$ by successively repeating the following: if P is a maximal path in C which alternates between edges of G_3 and those of H, remove all the edges of P and add an edge e_P between the two endpoints of the P. Notice that by replacing in C each maximal path P alternating between edges of G_3 and of H, by the edge e_P, we obtain an Eulerian trail alternating between edges of G_2 and edges of H'.

Since $\mathcal{F}_1 \equiv \mathcal{F}_2$, there exists $G_1 \in \mathcal{F}_1$ such that $G_1//H'$ has an Eulerian trail C_1 where edges in G_1 and edges in H' are alternatively used. We can replace each edge e_P of H' with P in C_1, obtaining an Eulerian trail of $G_1//(G_3//H)$ which alternates between edges of $G_1//G_3$ and of H. Therefore, $(\deg_{G_1//G_3}(v_1), \ldots, \deg_{G_1//G_3}(v_k)) =$

$(\deg_H(v_1), \ldots, \deg_H(v_k))$ and $(G_1//G_3)//H$ has at most one non-trivial component. This concludes the proof of the statement. □

5. Hamiltonian Cycle problem

We prove the main result of this paper.

Theorem 1.1. *Given a graph G and its k-expression, one can solve* Hamiltonian Cycle *in time $n^{\mathcal{O}(k)}$.*

5.1. **Equivalence between partial solutions.** We now define formally our notion of representatives based only on the degree sequence and connected components of auxiliary multigraphs associated with k-labeled path-partitions.

Definition 5.1 (Representatives by auxiliary multigraphs)**.** *Let G and H be multigraphs on vertex set $\{v_1, \ldots, v_k\}$. We write $G \simeq H$ whenever $(\deg_G(v_1), \ldots, \deg_G(v_k)) = (\deg_H(v_1), \ldots, \deg_H(v_k))$ and $\{V(C) \mid C$ is a component of $G\}$ is equal to $\{V(C) \mid C$ is a component of $H\}$. One easily checks that $\simeq$ is an equivalence relation on any set $\mathcal{F}$ of multigraphs on vertex set $\{v_1, \ldots, v_k\}$.*

For a family $\mathcal{F}$ of multigraphs on vertex-set $\{v_1, \ldots, v_k\}$, let reduce$(\mathcal{F})$ *be the operation which takes in each equivalence class of $\mathcal{F}/\simeq$ a representative.*

The goal of this section is to prove the following.

Proposition 5.2. *Let G be a graph with its irredundant clique-width k-expression ϕ, and let t be a node in the syntactic tree. Let G_t be the k-labeled graph constructed at t, and let $\mathcal{P}_1$ and $\mathcal{P}_2$ be k-labeled path-partitions of G_t whose labelings are induced by the labeling of G_t. If $Aux(\mathcal{P}_1) \simeq Aux(\mathcal{P}_2)$, then the following are equivalent.*

(1) G has a Hamiltonian cycle C_1 containing each path in $\mathcal{P}_1$ as a subpath and such that every edge in $C_1 - (\bigcup_{P \in \mathcal{P}_1} E(P))$ is contained in $E(G) \setminus E(G_t)$.
(2) G has a Hamiltonian cycle C_2 containing each path in $\mathcal{P}_2$ as a subpath and such that every edge in $C_2 - (\bigcup_{P \in \mathcal{P}_2} E(P))$ is contained in $E(G) \setminus E(G_t)$.

Proof. By symmetry, it is sufficient to prove that (1) implies (2). Suppose G has a Hamiltonian cycle C_1 containing each path in $\mathcal{P}_1$ as a subpath and such that every edge in $C_1 - (\bigcup_{P \in \mathcal{P}_1} E(P))$ is contained in $E(G) \setminus E(G_t)$. Let $\mathcal{Q}$ be the set of all maximal paths in $C_1 - (\bigcup_{P \in \mathcal{P}_1} E(P))$, and let $H := G[\bigcup_{Q \in \mathcal{Q}} V(Q)]$. We consider $\mathcal{Q}$ as the path-partition of H where the end vertices of paths in $\mathcal{Q}$ are labeled by their labels in G_t.

Since C_1 is a Hamiltonian cycle, $Aux(\mathcal{P}_1)//Aux(\mathcal{Q})$ has an Eulerian trail where edges in $Aux(\mathcal{P}_1)$ and edges in $Aux(\mathcal{Q})$ are alternatively used. Thus $Aux(\mathcal{P}_1)//Aux(\mathcal{Q})$ has at most one non-trivial connected component, and

$$(\deg_{Aux(\mathcal{P}_1)}(v_1), \ldots, \deg_{Aux(\mathcal{P}_1)}(v_k)) = (\deg_{Aux(\mathcal{Q})}(v_1), \ldots, \deg_{Aux(\mathcal{Q})}(v_k)).$$

Since $Aux(\mathcal{P}_1) \simeq Aux(\mathcal{P}_2)$ by the assumption, $Aux(\mathcal{P}_2)//Aux(\mathcal{Q})$ has at most one non-trivial connected component and

$$(\deg_{Aux(\mathcal{P}_2)}(v_1), \ldots, \deg_{Aux(\mathcal{P}_2)}(v_k)) = (\deg_{Aux(\mathcal{Q})}(v_1), \ldots, \deg_{Aux(\mathcal{Q})}(v_k)).$$

Therefore, by Lemma 3.1, $Aux(\mathcal{P}_2)//Aux(\mathcal{Q})$ admits an Eulerian trail where the edges in $Aux(\mathcal{P}_2)$ and the edges in $Aux(\mathcal{Q})$ are alternatively used.

Now, we show that G has a Hamiltonian cycle C_2 containing each path in $\mathcal{P}_2$ as a subpath and such that every edge in $C_2 - (\bigcup_{P \in \mathcal{P}_2} E(P))$ is contained in $E(G) \setminus E(G_t)$. Let

$e_1, e_2, \ldots, e_{2m}$ be the sequence of the edges in an Eulerian trail of $Aux(\mathcal{P}_2)//Aux(\mathcal{Q})$ where edges in $Aux(\mathcal{P}_2)$ and edges in $Aux(\mathcal{Q})$ are alternatively used such that $e_1 \in E(Aux(\mathcal{P}_2))$. For convenience, let $e_0 := e_{2m}$ and $e_{2m+1} := e_1$. For each $i \in \{1, \ldots, 2m\}$, let $L(i)$ be the vertex incident with e_i and e_{i-1}, and let $R(i)$ be the vertex incident with e_i and e_{i+1}. For each integer $i \in \{1, \ldots, m\}$, let P_{2i-1} be the path in $\mathcal{P}_2$ corresponding to e_{2i-1}, and for each integer $i \in \{1, \ldots, m\}$, let Q_{2i} be the path in $\mathcal{Q}$ corresponding to e_{2i}. We construct a Hamiltonian cycle of G from $(\bigcup_{P\in\mathcal{P}_2} V(P), \bigcup_{P\in\mathcal{P}_2} E(P))$ as follows. Let $i \in \{1, \ldots, m\}$.

(1) If Q_{2i} consists of an edge, then we add an edge between the end vertex of P_{2i-1} corresponding to $R(2i-1)$ and the end vertex of P_{2i+1} corresponding to $L(2i+1)$.
(2) Suppose Q_{2i} is a path of length at least 2. Let q_1^{2i}, q_2^{2i} be the end vertices of Q_{2i} corresponding to $L(2i)$ and $R(2i)$, respectively. Let q_3^{2i}, q_4^{2i} be the neighbors of q_1^{2i}, q_2^{2i} in Q_{2i}, respectively. Then, we add $Q_{2i} - \{q_1^{2i}, q_2^{2i}\}$ to the subgraph, and add an edge between q_3^{2i} and the end vertex of P_{2i-1} corresponding to $R(2i-1)$, and add an edge between q_4^{2i} and the end vertex of P_{2i+1} corresponding to $L(2i+1)$.

We verify that this is always possible. Suppose Q_{2i} consists of an edge. By the assumption of C_1, the edge in Q_{2i} is contained in $E(G) \setminus E(G_t)$; in other words, this edge is added in some ascendant node of t. This also implies that $R(2i-1) \neq L(2i+1)$, and by the definition of the clique-width expression, the label class with respect to the index of $R(2i-1)$ and the label class with respect to the index of $L(2i+1)$ are completely adjacent to each other in G. Therefore, there is an edge between the end vertex of P_{2i-1} corresponding to $R(2i-1)$ and the end vertex of P_{2i+1} corresponding to $L(2i+1)$, and furthermore, this edge is not in G_t because ϕ is irredundant. So, we can add it to connect two paths.

Now suppose Q_{2i} has length at least 2. By the definition of the clique-width expression, all vertices in a label class in G_t have the same neighborhood in $V(G) \setminus V(G_t)$. So, there are an edge between q_3^{2i} and the end vertex of P_{2i-1} corresponding to $R(2i-1)$, and an edge between q_4^{2i} and the end vertex of P_{2i+1} corresponding to $L(2i+1)$, and we can add them.

It implies that G has a Hamiltonian cycle C_2 containing each path in $\mathcal{P}_2$ as a subpath and such that every edge in $C_2 - (\bigcup_{P\in\mathcal{P}_2} E(P))$ is contained in $E(G) \setminus E(G_t)$. □

We can now prove the following which essentially tells us that if $\mathscr{F}$ is the set of possible k-labeled path-partitions at a node t of the syntactic tree, it is enough to store $\text{reduce}(\{Aux(\mathcal{P}) \mid \mathcal{P} \in \mathscr{F}\})$.

Proposition 5.3. *Let $\mathcal{F}$ be a family of graphs on the vertex set $\{v_1, \ldots, v_k\}$. Then $\mathcal{F} \equiv \text{reduce}(\mathcal{F})$.*

Proof. Since $\text{reduce}(\mathcal{F})$ is a subset of $\mathcal{F}$, it is sufficient to prove that $\text{reduce}(\mathcal{F}) \lesssim \mathcal{F}$. Let H be a multigraph on $\{v_1, \ldots, v_k\}$ and let $G_2 \in \mathcal{F}$ such that $(\deg_{G_2}(v_1), \ldots, \deg_{G_2}(v_k)) = (\deg_H(v_1), \ldots, \deg_H(v_k))$ and $G_2//H$ has at most one non-trivial component. By the definition of $\text{reduce}(\mathcal{F})$, there is $G_1 \in \text{reduce}(\mathcal{F})$ so that $G_1 \simeq G_2$. By definition of $\simeq$, we have $(\deg_{G_1}(v_1), \ldots, \deg_{G_1}(v_k)) = (\deg_H(v_1), \ldots, \deg_H(v_k))$. Since the components are the same in G_1 and in G_2, and the edges in H join components of G_2 in $G_2//H$ as in $G_1//H$, we can conclude that $G_1//H$ has at most one non-trivial component. □

5.2. An $n^{\mathcal{O}(k)}$-algorithm for Hamiltonian Cycle.

Proof of Theorem 1.1. We assume that G has at least 3 vertices, otherwise we can automatically say it is a No-instance. Since every k-expression can be transformed into an

irredundant k-expression in linear time, we may assume that G is given with an irredundant k-expression. Let ϕ be the given irredundant k-expression defining G, and T be the syntactic tree of ϕ. For every node t of T, let G_t be the subgraph of G defined at node t, and for each $i \in \{1, \ldots, k\}$, let $G_t[i]$ be the subgraph of G_t induced on the vertices with label i.

For each node t and each vector $(a_1, \ldots, a_k) \in \{0, 1, \ldots, n\}^k$, let $c[t, (a_1, \ldots, a_k)]$ be the set of all graphs F on the vertex set $\{v_1, \ldots, v_k\}$ where

- $F = Aux(\mathcal{P})$ for some k-labeled path-partition $\mathcal{P}$ of G_t,
- for each $i \in \{1, \ldots, k\}$, a_i is the degree of v_i in F.

Instead of computing the whole set $c[t, (a_1, \ldots, a_k)]$, we will compute a subset $r[t, (a_1, \ldots, a_k)]$ of $c[t, (a_1, \ldots, a_k)]$ of size $2^{\mathcal{O}(k \log k)}$ such that $r[t, (a_1, \ldots, a_k)] \equiv c[t, (a_1, \ldots, a_k)]$.

We explain how to decide whether G has a Hamiltonian cycle. Let t_{root} be the root node of T, and let $t_{lastjoin}$ be the node that is a node taking the disjoint union of two graphs and closest to the root node. We can observe that G has a Hamiltonian cycle if and only if there are some node t between t_{root} and $t_{lastjoin}$ with child t' and a path-partition $\mathcal{P}$ of $G_{t'}$ such that

- t is a join node labeled by $\eta_{i,j}$,
- $\deg_{Aux(\mathcal{P})}(v_i) = \deg_{Aux(\mathcal{P})}(v_j) > 0$ and $\deg_{Aux(\mathcal{P})}(v_{i'}) = 0$ for all $i' \in \{1, \ldots, k\} \setminus \{i, j\}$.

This is equivalent to that $c[t', (a_1, \ldots, a_k)] \neq \emptyset$ for some vector $(a_1, \ldots, a_k)$ where $a_i = a_j > 0$ and $a_{i'} = 0$ for all $i' \in \{1, \ldots, k\} \setminus \{i, j\}$. Therefore, if there is a Hamiltonian cycle, then $r[t', (a_1, \ldots, a_k)] \neq \emptyset$ for such a tuple of t, t', and $(a_1, \ldots, a_k)$, and we can correctly say that G has a Hamiltonian cycle, and otherwise, there are no such tuples, and we can correctly say that G has no Hamiltonian cycles.

Now, we explain how to recursively generate $r[t, (a_1, \ldots, a_k)]$.

(1) (Creation of a vertex v with label i)
If $a_i = 2$ and $a_j = 0$ for all $j \neq i$, then $c[t, (a_1, \ldots, a_k)]$ consists of one graph on the vertex set $\{v_1, \ldots, v_k\}$ with a loop incident with v_i, and otherwise, it is an empty set. So, we add the graph $(\{v_1, \ldots, v_k\}, \{v_i v_i\})$ to $r[t, (a_1, \ldots, a_k)]$ when $a_i = 2$ and $a_j = 0$ for all $j \neq i$, and set $r[t, (a_1, \ldots, a_k)] := \emptyset$ otherwise.

(2) (Disjoint union node with two children t_1 and t_2)
Since every path-partition of G_t is obtained by taking the disjoint union of a path-partition of G_{t_1} and a path-partition of G_{t_2}, we have

$$c[t,(a_1, \ldots, a_k)] := \bigcup_{(a_1^1,\ldots,a_k^1)+(a_1^2,\ldots,a_k^2)=(a_1,\ldots,a_k)} c[t_1, (a_1^1, \ldots, a_k^1)] \uplus c[t_2, (a_1^2, \ldots, a_k^2)].$$

We assign

$$r[t,(a_1, \ldots, a_k)] := \text{reduce}\left(\bigcup_{(a_1^1,\ldots,a_k^1)+(a_1^2,\ldots,a_k^2)=(a_1,\ldots,a_k)} r[t_1, (a_1^1, \ldots, a_k^1)] \uplus r[t_2, (a_1^2, \ldots, a_k^2)] \right).$$

(3) (Join node with the child t' such that each vertex with label i is joined to each vertex with label j)

Note that every path-partition of G_t is obtained from a path-partition of $G_{t'}$ by adding some edges between end vertices of label i and end vertices of label j. We can observe that when we add an edge between an end vertex v of a path P_1 with label i, and an end vertex w of a path P_2 with label j, these two paths P_1 and P_2 will be unified into a path whose end vertices are end vertices of P_1 and P_2 other than v and w. Thus, it corresponds to the operation $+(i,j)$ on auxiliary multigraphs. We observe that

$$c[t,(a_1,\dots,a_k)] = \bigcup_{\substack{a'_i-a_i=a'_j-a_j=\ell\geq 0\\ a'_x=a_x \text{ for } x\neq i,j}} (c[t',(a'_1,\dots,a'_k)] + \ell(i,j)).$$

We take all possible vectors $(a'_1,\dots,a'_k)$ where $a'_i-a_i=a'_j-a_j\geq 0$, and for all $t\in\{1,\dots,k\}\setminus\{i,j\}$, $a'_t=a_t$. Assume $\ell=a'_i-a_i$. For each $\ell\in\{0,1,\dots,n\}$, we assign

$$r_\ell := \mathrm{reduce}(\cdots \mathrm{reduce}(\mathrm{reduce}(r[t',(a'_1,\dots,a'_k)]+(i,j))+(i,j))\cdots+(i,j)),$$

where we repeat ℓ times, and assign

$$r[t,(a_1,\dots,a_k)] := \mathrm{reduce}(r_0\cup r_1\cup\cdots\cup r_n).$$

(4) (Renaming node with a child t' such that the label of each vertex with label i is changed to j)

Every path-partition of G_t is also a path-partition of $G_{t'}$, and vice versa. Since just labelings of vertices are changed, we can observe that if $a_i\neq 0$, then $c[t,(a_1,\dots,a_k)]$ is the empty set, and otherwise, we have

$$c[t,(a_1,\dots,a_k)] := \bigcup_{\substack{a_x=a'_x \text{ for all } x\neq i,j\\ a'_i+a'_j=a_j}} c[t',(a'_1,\dots,a'_k)]|_{i\to j}.$$

If $a_i\neq 0$, then we assign the empty set to $r[t,(a_1,\dots,a_k)]$, and otherwise, we assign

$$r[t,(a_1,\dots,a_k)] := \mathrm{reduce}\left(\bigcup_{\substack{a_x=a'_x \text{ for all } x\neq i,j\\ a'_i+a'_j=a_j}} r[t',(a'_1,\dots,a'_k)]|_{i\to j}\right).$$

To confirm the correctness of our algorithm, we claim the following.

Claim 1. *Let t be a node and $(a_1,\dots,a_k)\in\{0,1,\dots,n\}^k$. Then $r[t,(a_1,\dots,a_k)]\equiv c[t,(a_1,\dots,a_k)]$.*

Proof. It is clear when t is a node that creates a new vertex.

Suppose t is a disjoint union node with two children t_1 and t_2. For two vectors $(a^1_1,\dots,a^1_k)$ and $(a^2_1,\dots,a^2_k)$ where $(a^1_1,\dots,a^1_k)+(a^2_1,\dots,a^2_k)=(a_1,\dots,a_k)$, we have

$$r[t_1,(a^1_1,\dots,a^1_k)]\uplus r[t_2,(a^2_1,\dots,a^2_k)]\equiv c[t_1,(a^1_1,\dots,a^1_k)]\uplus c[t_2,(a^2_1,\dots,a^2_k)]$$

by Proposition 4.3. Thus, we have $r[t,(a_1,\dots,a_k)]\equiv c[t,(a_1,\dots,a_k)]$.

Suppose t is a join node with the child t' such that each vertex with label i is joined to each vertex with label j. By applying Propositions 4.1 and 5.3 recursively, we obtain that $r[t,(a_1,\ldots,a_k) \equiv c[t,(a_1,\ldots,a_k)]$.

Lastly, suppose t is a renaming node with a child t' such that the label of each vertex with label i is changed to j. We may assume $a_i = 0$. In that case, by Propositions 4.2 and 5.3, we have $r[t,(a_1,\ldots,a_k) \equiv c[t,(a_1,\ldots,a_k)]$. ◊

By Claim 1, we correctly update a representative set $r[t,(a_1,\ldots,a_k)]$ of $c[t,(a_1,\ldots,a_k)]$ for each pair of t and $(a_1,\ldots,a_k)$. Therefore, we can correctly decide whether G has a Hamiltonian cycle or not using sets $r[t,(a_1,\ldots,a_k)]$.

Running time. Each constructed set $r[t,(a_1,\ldots,a_k)]$ consists of at most $2^{\mathcal{O}(k\log k)}$ graphs, as we keep at most one graph for each partition of $\{v_1,\ldots,v_k\}$ after the reduce operation. For the node taking the disjoint union of two graphs, for a fixed vector $(a_1,\ldots,a_k)$, there are $n^{\mathcal{O}(k)}$ ways to take two vectors A_1 and A_2 such that $A_1 + A_2 = (a_1,\ldots,a_k)$. So, we can update $r[\cdot,\cdot]$ in time $n^{\mathcal{O}(k)} \cdot 2^{\mathcal{O}(k\log k)}$. For the node joining edges between two classes, the value ℓ can be taken from 0 to n. Since each operation $+(i,j)$ take $k^2 \cdot 2^{\mathcal{O}(k\log k)}$ time, we can update $r[\cdot,\cdot]$ in time $n^2 \cdot 2^{\mathcal{O}(k\log k)}$. Clearly, we can update $r[\cdot,\cdot]$ in time $n \cdot 2^{\mathcal{O}(k\log k)}$ for the relabeling nodes. Therefore, we can solve Hamiltonian Cycle for G in time $n^{\mathcal{O}(k)}$. □

6. More applications and Concluding Remarks

We discuss two variants of Hamiltonian Cycle, where we can apply the same technique.

Let q be a positive integer. The q-Cycle Covering problem asks for a given graph G whether there is a set of at most q pairwise vertex-disjoint cycles in G whose union contains all vertices of G. Definitely, 1-Cycle Covering is the Hamiltonian Cycle problem. In the q-Cycle Covering problem, we relax the definition of path-partitions so that it may contain at most q cycles, and we keep the number of cycles in the path-partition. Also, we define its auxiliary multigraph $Aux(\mathcal{P})$ using those remaining paths. One can easily check that two such modified path-partitions $\mathcal{P}_1$ and $\mathcal{P}_2$ are equivalent for q-Cycle Covering if they contain the same number of cycles and $Aux(\mathcal{P}_1) \simeq Aux(\mathcal{P}_2)$.

The second application is for Directed Hamiltonian Cycle. Clique-width was also considered for directed graphs by Courcelle and Olariu [5]. The clique-width operations for directed graphs are the same as for the undirected graphs, except the one that add edges between two label classes, defined as follows:

(3*) Adding an arc (u,v) for each vertex u with label i to each vertex v with label j ($i \neq j$, denoted by $\eta_{i,j}$).

The *clique-width* of a directed graph G is the minimum number of labels needed to construct G using these operations. In this case, we use directed auxiliary multigraphs. Similar to Lemma 3.1, we can show the following.

Lemma 6.1. *Let G be a connected directed multigraph whose arcs are colored by red and blue. Then the following are equivalent.*

(1) For every vertex v, the number of blue edges leaving v is the same as the number of red edges entering v, and the number of red edges leaving v is the same as the number of blue edges entering v.

(2) G has a red-blue alternating Eulerian directed trail.

Using Lemma 6.1, we can proceed same as Theorem 1.1.

We conclude with one question. A digraph D is an *out-tree* if D is an oriented tree with only one vertex of in-degree zero (called the root). The vertices of out-degree zero are called leaves of D. The Min Leaf Out-Branching problem asks for a given digraph D and an integer ℓ, whether there is a spanning out-tree of D with at most ℓ leaves. This problem generalizes Hamiltonian Path by taking $\ell = 1$. Ganian, Hlinený, and Obdrzálek [10] showed that there is a $n^{2^{\mathcal{O}(k)}}$-time algorithm for solving Min Leaf Out-Branching problem, when a clique-width k-expression of a digraph D is given. We ask whether it is possible to drop down the exponential blow-up $2^{\mathcal{O}(k)}$ to $\mathcal{O}(k)$.

References

[1] S. Arnborg and A. Proskurowski. Linear time algorithms for NP-hard problems restricted to partial k-trees. *"Discrete Applied Mathematics"*, 23:11–24, 1989.

[2] H. L. Bodlaender. A partial k-arboretum of graphs with bounded treewidth. *Theoretical Comput. Sci.*, 209:1–45, 1998.

[3] B. Courcelle. The monadic second-order logic of graphs I: Recognizable sets of finite graphs. *"Information and Computation"*, 85:12–75, 1990.

[4] B. Courcelle, J. A. Makowsky, and U. Rotics. Linear time solvable optimization problems on graphs of bounded clique width. *Theoretical Comput. Sci.*, 33:125–150, 2000.

[5] B. Courcelle and S. Olariu. Upper bounds to the clique width of graphs. *Discrete Appl. Math.*, 101(1-3):77–114, 2000.

[6] R. Diestel. *Graph Theory*. Number 173 in Graduate Texts in Mathematics. Springer, third edition, 2005.

[7] W. Espelage, F. Gurski, and E. Wanke. How to solve NP-hard graph problems on clique-width bounded graphs in polynomial time. In *Graph-theoretic concepts in computer science (Boltenhagen, 2001)*, volume 2204 of *Lecture Notes in Comput. Sci.*, pages 117–128. Springer, Berlin, 2001.

[8] F. V. Fomin, P. A. Golovach, D. Lokshtanov, and S. Saurabh. Intractability of clique-width parameterizations. *SIAM J. Comput.*, 39(5):1941–1956, 2010.

[9] F. V. Fomin, P. A. Golovach, D. Lokshtanov, and S. Saurabh. Almost optimal lower bounds for problems parameterized by clique-width. *SIAM J. Comput.*, 43(5):1541–1563, 2014.

[10] R. Ganian, P. Hlinený, and J. Obdrzálek. Clique-width: When Hard Does Not Mean Impossible. In T. Schwentick and C. Dürr, editors, *28th International Symposium on Theoretical Aspects of Computer Science (STACS 2011)*, volume 9 of *Leibniz International Proceedings in Informatics (LIPIcs)*, pages 404–415, Dagstuhl, Germany, 2011. Schloss Dagstuhl–Leibniz-Zentrum fuer Informatik.

(Kanté and Bergougnoux) Université Clermont Auvergne, LIMOS, CNRS, Aubière, France.
E-mail address: {mamadou.kante,benjamin.bergougnoux}@uca.fr

(Kwon) Logic and Semantics, Technische Universität Berlin, Berlin, Germany.
E-mail address: ojoungkwon@gmail.com