Internship Report

~ Properties of the KLX number ~

Coda BOUROTTE
coda.bourotte@ens-lyon.fr

Research Advisor:
Shinnosuke SEKI
s.seki@uec.ac.jp

Seki Lab — University of Electro-Communications
Chofu, Tokyo, Japan

1. Abstract

The KLX number of a graph is a measurement introduced by [1] to describe the set of graphs that can be build using their method of RNA origami folding. In this internship I analysed different aspects of the KLX number: bounds, counter-examples, NP-Hardness, algorithms etc. I analyzed the relationships between the KLX number of a graph and the treewidth, degree, and cutwidth. I will prove that the Tree-Width is a lower-bound for the KLX number, and that the cutwidth provides a lower bound on the product of the KLX number and the graph's degree. I will show that computing the KLX number is NP-hard, even when restricted to the class of graphs that are planar, bridgeless, bipartite and of degree ≤ 5. Finally, we show that determining whether the KLX number of a graph is less than a fixed constant is MSO_2-expressible, which implies the existence of a linear-time algorithm for graphs of bounded treewidth. This was done using a new trick to store a finite list of neighbours for every vertex of a graph in MSO_2 as a labeling of the edges and vertices.

Contents

2. Introduction

2.1. Motivation

In the field of DNA nanotechnology, researchers are interested in finding means to fold RNA molecules into a predefined shapes (wireframe) [2]. This can be used to create self-assembling biostructure *in vitro*, and even potentially *in vivo*.

DNA and RNA DNA (deoxyribonucleic acid) is a double-stranded structure, meaning that it consists of a series of pairs of *bases*. There are four kinds of base, and Adenine (A) is paired with Thymine (T), while Cytosine (C) is paired with Guanine (G). RNA (ribonucleic acid), on the other hand, is a single-stranded structure that replaces Thymine with Uracil (U). It is obtained by co-transcription: one strand of the DNA double helix is 'copied' by an enzyme into a single RNA strand.

RNA folding RNA is very unstable and will fold itself in a way that minimizes its energy. It will try to fold in a way that pairs A with U and C with G[1]. This process can be utilized to force the folding to create a particular shape. To that extend, [2] introduced a method for creating a DNA molecule that will fold its corresponding RNA strand into a targeted wireframe. The idea is to explore a spanning tree of the graph, writing one strand when going to a child vertex, and writing the second paired strand when coming back to the parent, to make a very rigid and strong two-strand structure for RNA. For all additional edges not in the spanning tree, it will create two half-edges called *kissing loops* that will bind together once both are visited (See Figure 1). Once closed, however, they will bind strongly and will not open again, allowing us to use the pair of half-edges again.

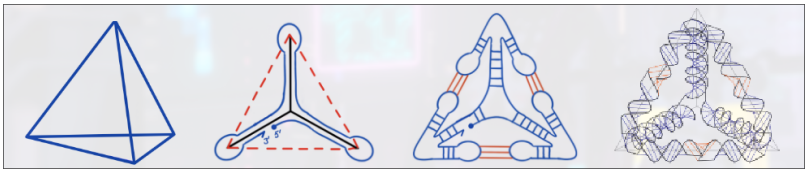


Figure 1: The method described by [2].
From left to right: the target graph, the spanning tree,
spanning tree with the half-edges, and the RNA final molecule[2]

Constraints Two constraints come from this construction, solved by [1]:

- The first is *polymerase trapping*, identified by [3] and studied by [4]. Once the two half-edges bind and close, the RNA breaks if it tries to complete a strand from the newly formed cycle. This is caused by the fact that RNA's double strands have a slight rotation, and therefore requires the enzyme to rotate around the pre-existing single strand. This rotation will be blocked by the cycle as the size of the enzyme and the DNA it reads is very large compared to the RNA that has been written.
  To solve this issue, [1] proposed to close every cycle by a kissing loop. This means that for every cycle, the last time we see a node from the cycle correspond exactly to the last time we visit the edge not in the tree. This is achieved iff the tree is a DFS tree [1], as, in a DFS tree, all edges not in the tree are 'backward edges' from a child to a grandparent, and they are the ones that 'close' the cycles.
- The second is that the number of half-edges that can be present simultaneously is limited and finite (currently limited to 6, but it might grow in the future). We will formally define the *KLX number of a graph* as the minimal number of types of half-edges necessary to create a specific graph using this corrected method.

Tree-depth A second parameters on graphs simmilar to the KLX was also considered by [1], the *tree-depth* (TD), to model another restriction on the number of single-strand tree edges at any given time. This restriction means that the DFS tree must be also bounded in height. Since this is an already a well-established

[1]Some other binds may occur but are weaker
[2]Image by Pekka Orponen & [2] (CC BY 4.0)

measurement of a graph that is also less restictive as the curretn state-of-the-art bound is bigger than the KLX one, I did not particularly study the Tree-Depth. For more information, the reader might be interested in [5].

2.2. Results

Established results When I started my internship, not a lot was known abut the KLX number. [1] defined and proved that computing the KLX-number of a graph was NP-Hard. An exact enumeration-based algorithm was described to compute the KLX number in any graph. The paper also described a few open questions, most notably[3]:

- Efficient combinatorial algorithms for minimising the KLX and TD numbers in some interesting classes of graphs, such as 3-regular and polyhedral graphs, or proving the problems remains NP-hard even in these classes.
- Efficient fixed-parameter or approximation algorithms for minimising the KLX and TD numbers in some relevant classes of graphs.

Informally, a conjecture was shared with me: "if a hamiltonian path exists, then the DFS of best KLX number is a hamiltonian path".

My results Both open questions and the conjecture got answered by my work. Here are my main results:

- The Tree-Width + 1 is a lower bound of the KLX number
- The Cut-Width is a lower bound to the product of the KLX number with the degree of the graph
- The 2×8 rectangular grid with every even vertical edge removed is a counter example to the conjecture
- Computing the KLX number in the class of planar bipartite 4-regular graph is NP-Hard, and similarly for planar bipartite 5-regular bridgless graphs
- For every k, there is a MSO_2 formula to test if a graph have a KLX number less or equal to k. This, by Courcelle's theorem, gives a fixed-parameter-tracable algorithm to for the class of graph of bounded tree-width (aka a linear-time algorithm with a huge constant depending on the KLX number and tree-width)

Some counter-intuitive properties was also found, like how adding an edge to the graph can divide by up to 2 the KLX number, allowing for counter-example to conjectures I came up with. Unfortunatly, no upper-bound (other than trival ones like the number of edges) was found.

Publication It is planned for thoses results to be published into a theoretical computer science journal / algorithmistic one before the end of 2025. We aim at STACS, but might fall back into a less elitist one. During my internship I researched other things that will not be mentionned in this document, but some might lead to publications later this year / next year too. Thoses include:

- Generating RNA langages using circular DNA and optimising for the length of the cycle and number of cropped contextes
- Analyzing the topology of the histogram relation on finite arrays

$$\lim_{n\to\infty} \frac{1}{1+n} \sum_{i=1}^{k} e^i = \{k : k > i\} = \mathcal{P}\lambda \text{ lambda}$$

[3] Thoses have been copied from the article

3. Definitions

In this section I will introduce the different formal definitions that I will use during the rest of the report. We will start with some formal graph theory definitions, introducing the tree-width as a formal concept, informally describe the MSO_2 logic with examples and explain an import link it has with tree-width, before introducing the formal definition of the KLX number and give some examples.

3.1. General Graph Theory Definitions

General We note $|X|$ the cardinal of X. We note $A \sqcup B$ the disjoint union of A and B. We note A^B the set of function from B to A. We note $A^{(\mathbb{N})}$ the set of finite sequences $(a_i)_{i\leq n}$ in A. For $(u_i)_i \in A^{(\mathbb{N})}$ a finite sequence, we note $|(u_i)_i|$ its length. Given two sequence $a, b \in A^{(\mathbb{N})}$, we note $a \parallel b$ the concatenation of a and b. We note $\mathcal{P}(X)$ the power set of X and $\mathcal{P}_k(X)$ the set of all powerset of X of cardinal k. For $k \leq |X|$, $S \underset{k}{\subseteq} X$ means that $S \subseteq X \wedge |S| \leq k$. For $k \in \mathbb{N}$, note $[k] = \{1, 2, 3, .., k\}$. All definition will be indicated using italics in this document *like this*.

Graphs Given a graph G, we note V_G and E_G its set of vertices/edges. We note $\deg_{E_G} v$ the degree of $v \in V_G$ in the graph (V_G, E_G), and we'll denote by $\Delta(G) = \max_{v\in V_G} \deg v$. We denote as $G[S]$ for $S \subseteq V_G$ the subgraph $G' \coloneqq (S, E_G \cap S^2)$. If $S \subseteq E_G$ instead, then $G[S] \coloneqq (V', S)$ with V' the covered vertexes of S (formally, $V' = \{x \in V_G \mid \exists y, (x, y) \in S\}$). For $v \in V_G$ (resp. $e \in E_G$), we note $G - v$ for $G[V_G \setminus \{v\}]$ (respc. $G - e$ for $G[E_G \setminus \{e\}]$). For $A = \{e_1, ..., e_n\} \subseteq E$ (or a subgraph of G composed of the edges A), we can note $G - A$ the graph $G - e_1 - e_2 ... - e_n$. For $k \in \mathbb{N}$, we denote by K_k the complete graph of k vertexes and for all $a, b, \in \mathbb{N}$, we denote by $K_{a,b}$ the complete bipartite graph with two set of vertices of size a and b. A *loop* is a cycle of length 1. For $k > 2$, we denote C_k the cycle of length k. A *simple graph* is a connected, undirected, loop-free graph.

Connectedness A graph is *k-connected* if for all $S \underset{k-1}{\subseteq} V_G$, $G[V_G \setminus S]$ is connected, meaning that removing $k - 1$ vertexes cannot split the graph into multiple connected component. A node is said to be 1-connected if removing it doesn't augment the number of connected components. A graph is said to be *bridgeless* if there doesn't exists an edge $e \in E_G$ such than $G - e$ have more connected component than G.

Remark If a graph is 2-connected, then it is bridgeless: Otherwise, suppose there exists $e = \{x, y\}$ such than $G - e$ split the graph, then $G - x$ splits it too as $G - x$ is a subgraph of $G - e$.

Trees and orderings A *tree* is an acyclic connected subgraph, and the tree T is said to be a *spanning tree of* G if $V_T = V_G$. If a tree T is rooted on $r \in V_G$, for $v \in V_G$, we denote by $T(v)$ the subtree of v in T (now rooted in v). An *ordering* is of V_G is an bijective sequence of vertexes $\varphi : V_G \longrightarrow [\![V_G]\!]$, with the ordering defined be such that for $x, y \in V_G$ we have $x < y \Leftrightarrow \varphi(x) < \varphi(y)$. φ can be seen as a sequence of vertexes $\langle v_1, v_2, ..., v_n\rangle$, and we will often denote the i-th vertex $\varphi^{-1}(i)$ as v_i.

Walks We define a *walk over* G to be a finite sequence $\langle v_i\rangle_{1<i\leq n} \in V_G^{(\mathbb{N})}$ of vertexes such that $\forall i < n, (v_i, v_{i+1}) \in E_G$. A path is an injective walk (we dont pass twice on the same vertex). A *close path* (respc. walk) $\langle v_i\rangle_{i\leq n}$ of G is a path (respc. walk) such that $(v_n, v_1) \in E_G$. We define a *cycle* to be an alias for a close path. If an edge e is apart of a walk v, we denote that "$e \in v$". We will sometime consider a cycle to be the set of edges it contains.

DFS Given a simple graph $G = (V, E)$, we say that T is a *DFS (Depth First Search) spanning-tree of* G (or *Trémaux Tree*) if it is a tree that can be obtained by visiting the graph G using a DFS algorithm. DFS trees are guarantied to only have *backward edges*, meaning that all edges not covered by the tree are always in between a vertex and one of its grand-child. A *DFS walk* is a sequence $\langle v_1, v_2, ..., v_n\rangle$ obtained by the order of visits of vertexes during a DFS algorithm, including when we backtrack to a vertex.

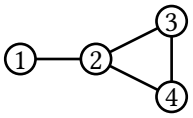


Figure 2: An example graph with 4 DFS starting from vertex 2

Example The possible DFS walks of the example graph Figure 2 starting at node 2 are

$$\langle 2,1,2,3,4,3,2\rangle; \langle 2,1,2,4,3,4,2\rangle; \langle 2,3,4,3,2,1,2\rangle; \langle 2,4,3,4,2,1,2\rangle$$

3.2. Tree-width

Tree-Width Given a graph $G = (V_G, E_G)$, we define a *tree-decomposition of* G as a labeled tree $T = (V_T, E_T)$ with such that (the vertex of V_T are called *bags*):

- Every bag $B \in V_T$ is a subset of V_G
- For all edges $\{a, b\} \in E_G$, there exists $B \in V_T$ such that $a, b \in B$
- For all vertex $v \in V_G$, all bags containing v is a connected subtree of T. Formally, $T[\{B \in V_T : v \in B\}]$ is connected.

The *width* of a tree decomposition T is $\max_{B \in V_T} \mathrm{Card}(B) - 1$. We define the *tree-width* of G (denoted by $\mathrm{TW}(G)$) as the smallest width of a tree-decomposition of G.

Intuition Tree width represent intuitively "how close to a forest" a graph is. A forest have tree-width 1, a $n \times n$ grid have tree-width n and K_n have tree-width $n - 1$. A lot of very strong results have been found around tree-width, the one we will be using being Courcelle's theorem [6], stating that if a property on graph can be written in a specific logic (that is MSO_2), then it is decidable on any familly of graphs of bounded tree-width in polynomial time in the graph.

Remark Property 3 is equivalent to the fact that for all $u \in V_G$, if $u \in B_1 \cap B_2$ two bags, then for all B' in the path from B_1 to B_2 we have $u \in B'$.

Path-Width The definition of the Path-Width is the same as for the tree-width, expect that all trees considered must be paths. We will note the path-width of G as $\mathrm{PW}(G)$. From this definition, it immediatly comes that $\mathrm{TW}(G) \leq \mathrm{PW}(G)$, because every path is a tree.

Remark As the instar of the preceding remark about the tree width, havign a tree width $\leq t$ is equivalent for path-width to finding a sequence of set $B_{i \leq n}$ of cardinal $\leq t$ covering the edges and vertexes such than for all $i \leq k \leq j$, we have $B_k \subseteq B_i \cap B_j$.

Cut-Width Given a graph $G = (V_G, E_G)$ and an ordering of the vertex $\langle v_1, .., v_n\rangle$, the *cutwidth* of the ordering (somtimes called the *folding number*) is the maximal number of edges overlaping at the same time for the given order. Formally, it's $\max_{1 \leq k < n} \mathrm{Card}\big(\{(v_i, v_j) \in E_G \mid i \leq k < j\}\big)$. The *cutwidth* of the graph is the minimal cutwidth of an ordering.

Prop [7] Some inequalities about cut-width, path-width, tree-width and the degree of a graph include:

- $\mathrm{TW} \leq \mathrm{CW}$
- $\mathrm{PW} \leq (\mathrm{TW} + 1) \times \mathrm{CW}$
- T a tree-decomposition of G of minimal Tree-Width, we have $\mathrm{PW}(G) \leq (\mathrm{TW}(G) + 1) \times \Delta(G) \times \mathrm{CW}(T)$

3.3. MSO_2

Only an introduction will be made about MSO, and some formalism will be put under the rug intentionally. For more extended and detailed understanding, especially on the matter of links with langage theory, refer to [8].

The goal of MSO_2 is to express properties about graphs[4] as logical formulas. To that extend, one may quantify over vertices, over edges, and test if two vertices make an edge. We fix a set of *variables* $V = \{x, y, a, x', \alpha_2, ...\}$ and a set of *big variables* $\overline{V} = \{X, U_1, \Gamma, T, ...\}$ (also called *set-variables*).

Formulas The langages of logical formulas in MSO_2 are build from the follwing blocks :
- Quantifying over elements: for all $\psi \in \mathrm{MSO}_2$ and $x \in V$, we have "$\forall x, \psi$", "$\exists x, \psi$" $\in \mathrm{MSO}_2$
- Quantifying over sets of elements: for all $\psi \in \mathrm{MSO}_2$ and $X \in \overline{V}$, we have "$\forall X, \psi$", "$\exists X, \psi$" $\in \mathrm{MSO}_2$
- Equality of elements: for all $x, y \in V$, we have "$x = y$" $\in \mathrm{MSO}_2$
- Propositional operators: for $\psi, \varphi \in \mathrm{MSO}_2$, we have "$\psi \wedge \varphi$", "$\psi \vee \varphi$", "$\neg\psi$", "$\psi \to \varphi$", "$\psi \leftrightarrow \varphi$" $\in \mathrm{MSO}_2$
- Testing if an element is a vertex or edge: for $x \in V$, we have "$\mathrm{IsVertex}(x)$" $\in \mathrm{MSO}_2$
- Testing if an element is the edge described by two others: for $e, u, v \in V$, we have "$e = \{u, v\}$" $\in \mathrm{MSO}_2$

A very important notion here is how quantifiers are over sets of both vertices AND edges.

A lot of notations can be introduced:
- Subset: "$X \subseteq Y$" to denote $\forall x, x \in X \to x \in Y$
- Forall: "$\forall x \in V, \psi$" to denote $\forall x, \mathrm{IsVertex}(x) \to \psi$, (similarly for a set variable)
- Exists: "$\exists X \subseteq E, \psi$" to denote $\exists X, (\forall x \in X, \neg\ \mathrm{IsVertex}(x)) \wedge \psi$ (similarly for a regular variable)
- Edges: "$\{x, y\} \in E$" to denote $\exists e \in E, e = \{x, y\}$
- And others, that I trust the reader will be able to transform into a MSO_2 formulas if necessary.

Satifability Given a formula $\varphi \in \mathrm{MSO}_2$, we can define the set of graphs that satisfies φ. If G satisfies φ, we note it $G \vDash \varphi$, and we say that φ recognise G. We left this definition as an informal one, but we will give examples.

Example 1: Complete Graphs A simple expression recognizing complete graphs would be

$$\mathrm{COMP} \coloneqq \forall x \in V, \forall y \in V, x \neq y \to \{x, y\} \in E$$

It means that "for all pair of two nodes x, y, if x is different to y, then there is an edge from x to y".

Example 2: k-colorable graphs For every fixed $k \in \mathbb{N}^*$, we can express the fact that a graph is k-colorable as a MSO formula. The idea is to put all vertices of the same color in a set (bag), and then make sure that the sets respect 3 propreties :
- **P1:** All vertices are in a bag
- **P2:** Not vertices is in two bags
- **P3:** There is no edges that have both endpoint in the same bag

P1 and **P2** ensure that the bags forms a partitons of the verticices, while **P3** ensure that a bag can correspond to a color with no issues. This gives the resulting formula:

$$\begin{aligned}\mathrm{COLOR}_k \coloneqq \exists X_1, ..., \exists X_k, &\left(\forall x \in V, \bigvee_{i \leq k} x \in X_i\right) \wedge \\ &\left(\forall x \in V, \bigwedge_{i < j \leq k} \neg\big(x \in X_i \wedge x \in X_j\big)\right) \wedge \\ &\left(\forall \{x, y\} \in E, \bigwedge_{i \leq k} \neg(x \in X_i \wedge y \in X_i)\right)\end{aligned}$$

Example 3: Paths We can describe the fact that a set of both vertices AND edges X is a path from u to v as the fact that for every edge in X both of its endpoint are in X, and every vertex other than u, v in X only have two edges in X:

[4] or any *structure*, but we will focus here on the graph case

$$\begin{aligned}\text{IsPath}(X,u,v) \coloneqq\ & u \in X \land v \in X \land \\ & (\forall e \in X \cap E, \exists u, v \in X, e = \{u,v\}) \land \\ & (\forall x \in X \cap V \setminus \{u,v\}, \to \exists e_1, e_2 \in X \cap E, \text{IsEndpoint}(x,e_1) \land \text{IsEndpoint}(x,e_2) \land [\\ & \quad \forall e' \in X \cap E, e' \neq e_1 \land e' \neq e_2 \to \neg\ \text{IsEndpoint}(x,e') \\ &])\end{aligned}$$

with $\text{IsEndpoint}(x,e) \coloneqq \exists y, e = \{x,y\}$. We need to have a set containing edges as a vertex might have more than one neighbour in the path (but only one will be the "next" vertex).

Example 4: Trees An expression recognizing that only a single path exists between two nodes u, v can be:

$$\text{SinglePath}(u,v) \coloneqq \exists X, \text{IsPath}(X,u,v) \land (\forall X', \text{IsPath}(X',u,v) \to X = X')$$

From that, an expression to recognise if a graph G is a tree could be:

$$\text{TREE} \coloneqq \forall u, v \in V, \text{SinglePath}(u,v)$$

Courcelle's Theorem For every $k \in \mathbb{N}$ and formula $\varphi \in \text{MSO}_2$, there is a linear time algorithm $\mathcal{A}$ that, for all graph G of tree-width $\leq k$, test if $G \vDash \varphi$. This complexity hide a huge constant that is hyper-exponential in k and the number of alternating quantifiers in φ. This have been proven by Courcelle in [6].

Other results include that testing if a formula have a satifying graph G of tree-width less than k is decidable, and this implies that testing if a formula is a tautology over graph of bounded tree-width is also decidable [6].

$\mathbf{MSO_1}$ is the same logic as MSO_2 but only restricted to a quantification over vertices. Some formulas (like the existence of a hamiltonian path) are only possible in MSO_2. Other examples include testing the existence of a perfect matching or testing the existence of a spanning tree with maximal degree 3 [8]. We will not study MSO_1 here.

3.4. KLX

For $\langle v_1, v_2, ..., v_{|v|} \rangle$ a DFS walk and $x, y \in V$ some vertexes, we note $\lfloor x \rfloor = \min\{i : 0 < i \leq |v| : v_i = x\}$ the first occurrence of x in the sequence v and $\lfloor x \rfloor_{\geq n} = \min\{i : n \leq i \leq |v| : v_i = x\}$ the first occurrence of x after n in the sequence. Similarly, $\lceil x \rceil = \max\{i : 0 < i \leq |v| : v_i = x\}$ and $\lceil x \rceil_{\leq n} = \max\{i : 0 < i \leq n : v_i = x\}$

A backward edge $\{u,v\}$ is *open at time t* if $\lceil u \rceil \leq t < \lfloor v \rfloor_{\geq \lceil u \rceil}$. In that case, a node (here it is u) has already finished to be visited while a node (here, v) will be visited later. For a backward edge $e = \{u,v\}$, we note e^{st} the node that will be completely visited first and e^{nd} the node that have yet to be fully visited. The set of open edges at time t for the walk v will be denoted $\mathcal{O}_t^v = \{e \in E_G \setminus E_T \mid e \text{ is open at } t \text{ in } v\}$. We will omit v from the notation if it's clear from the context. The KLX number of a DFS walk v is then noted $\text{KLX}(G,v)$ and is defined as the maximal number of same-time open edges, that is $\max_{0<t\leq |v|} \text{Card}(\mathcal{O}_t^v)$

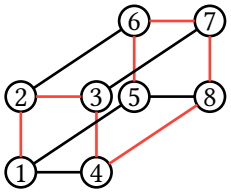
The KLX number of a graph, noted $\text{KLX}(G)$, is the minimal KLX number of a DFS walk. For a root $r \in V_G$, we note $\text{KLX}(G,r)$ the minimal KLX-number of a DFS walk that start at r. We therefore have $\text{KLX}(G) = \min_{v \in V_G} \text{KLX}(G,v)$

Remarks A few remarks:

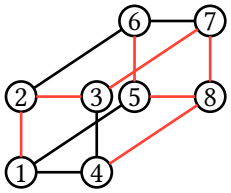
1. The KLX is invariant up to ismorphisms, meaning that we can consider the set of DFS walks up to isomorphisms. Therefore, for vertex-transitive graphs (i.e. graphs for which for all pairs of vertex x, y, there exists an isomorphism of G that maps x to y), we have that $\forall r, \text{KLX}(G) = \text{KLX}(G,r)$.
2. Because we consider a DFS tree, every edge of the tree is taken at most twice: once in a way and once when comming back. Given an edge $e = \{x,y\}$ of the DFS tree, let t be the moment when we do the transition $x \to y$ for the second time (when coming back). We then define $\mathcal{O}_e$ as $\mathcal{O}_t$.

Example 1: Cycle In a cycle, all DFS paths yield the same hamiltonian path (up to isomorphism). For C_k, the DFS might be : $\langle 1, 2, ..., k-1, k, k-1, ..., 1\rangle$, yielding us a KLX of 1, as the edge $(k-1, 1)$ is the only open one from time $t = k$ to $2k-1$

Example 2: Cube The 3D cube has two distinct DFS trees: one is a path, and the other has a split (see Figure 3). Note that there are 3 DFS *walks* as the order for the split matter: since a vertex has two children, going to one first then the other may change the KLX number.



DFS: $\langle 1, 2, 3, 4, 8, 7, 6, 5, 6, 7, 8, 4, 3, 2, 1\rangle$

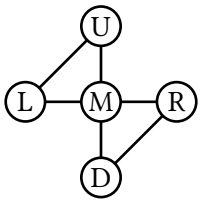


DFS: $\langle 1, 2, 3, 7, 8, 5, 6, 5, 8, 4, 8, 7, 3, 2, 1\rangle$ and $\langle 1, 2, 3, 7, 8, 4, 8, 5, 6, 5, 8, 7, 3, 2, 1\rangle$

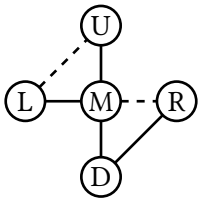
Figure 3: The two DFS (in red) of the cube, up to isomorphism, and their associated DFS tree. For the second one, the split occurs on node 8, and the two DFS show this.

The KLX number of the path is 4: during the transition from 7 to 8, we have the edges $\mathcal{O}_{\{7,8\}} := \{(5,1),(5,8),(6,2),(7,3)\}$ that are open (and it's the maximal). The KLX number of the splitted DFS is 5 in both cases: during the transition from 8 to 7 (the second time), we have the edges $\mathcal{O}_{\{7,8\}} := \{(6,2),(6,7),(5,1),(4,1),(4,3)\}$ that are open (and it's the maximal).

Example 3: This example is to show that the order matter of the KLX matter. Consider the following graph:



We will show that the same DFS tree can have two KLX number depending on the ordering. Consider the following tree starting from U (backward edges are dotted):



Two possible DFS *walks* are possible (in underbrace is the moment where every dotted backward edge is open):

- For walk $\langle U, M, D, \underbrace{R, D, M}_{\{R,M\}}, \underbrace{L, M, U}_{\{L,U\}}\rangle$, the KLX is 1
- For walk $\underbrace{\langle U, M, L, M, D, \underbrace{R, D, M}_{\{R,M\}}, U\rangle}_{\{L,U\}}$, the KLX is 2

Example 4: Complete graph We compute the KLX of K_k to be $\lfloor \frac{k}{2} \rfloor \lceil \frac{k}{2} \rceil$: let v be a DFS walk. Then it is a path, and since all paths are ismorphic in K_k, we will just compute the KLX number of a path in K_k. Denote $v = \langle v_1, ..., v_{k-1}, v_k, v_{k-1}, ..., v_1\rangle$ the DFS walk.

- For $1 \leq t < k$, we have $\mathcal{O}_t = \emptyset$ as we have yet to see a vertex for the last time.
- For $k \leq t < 2k-1$, we have $\mathcal{O}_t = \{(i,j) \in [k]^2, i \leq t < j\}$, with $|\mathcal{O}_t| = (t-k+1)(2k-1-t)$

The maximum is reach in the middle, when $t = k + \lfloor \frac{k}{2} \rfloor$, with a value of $\lfloor \frac{k}{2} \rfloor \lceil \frac{k}{2} \rceil$; therefore $\mathrm{KLX}(K_k) = \lfloor \frac{k}{2} \rfloor \lceil \frac{k}{2} \rceil$

Easy bound We have that $\mathrm{KLX}(G) \leq |E_G| - |V_G| + 1$

Proof The number of open edges at the same time cannot exceed the number of backward edges, and this number is the number of edges minus the number of edges in a DFS tree. □

4. General Properties of the KLX numbers

This section we will introduce :

- 3 useful lemmas of my creation: "1-connected Lemma", the "Cycle lemma" and the "Path lemma"
- A characterisation fo graphs of KLX = 0 and KLX = 1 using cycles
- Some counter-examples of conjectures that were thought to be true.

All the results here are of my creation.

4.1. Useful Lemmas

Thoses lemmas will be useful for the proofs that will follow, and help to build an intuition on how the KLX number can grow.

Prop (The "1-connected Lemma") Let G be a connected graph and x be a not 1-connected vertex (i.e. removing it increases the number of connected components, it is an articulation point). Denote B_1 the connected component of $G - x$ with the biggest $\mathrm{KLX}(G[B^* \cup \{x\}], x)$ and B_2 the second largest. Then, we have

$$\mathrm{KLX}(G[B_1 \cup \{x\}], x) \geq \mathrm{KLX}(G) \geq \mathrm{KLX}(G[B_2 \cup \{x\}], x)$$

Intuition If we start our DFS in B_i, then when we arrive on x, we will have to do $\mathrm{KLX}\big(B_j \cup \{x\}, x\big)$ for every $j \neq i$. So only the "worse" component (B_1) may be improved by starting inside it. In a way, x "split" the DFS walk into a series of independant DFS walks.

Proof

- $\boxed{\mathrm{KLX}(G[B_1 \cup \{x\}], x) \geq \mathrm{KLX}(G)}$ We show that there is a DFS walk of G with $\mathrm{KLX} = \mathrm{KLX}(B^* \cup \{x\}, x)$. Note $B_1, .., B_n$ the different connected component of $G - x$. For every one, let $\langle v_j^{(i)} \rangle_{j \leq |v^{(i)}|}$ be a KLX walk of minimal $\mathrm{KLX}(G[B_i \cup \{x\}], x)$. Consider the concatenation of all walks splitted by x as follows:

  $$v = \langle x, v_1^{(1)}, v_2^{(1)}, ..., v_{|v^{(1)}|}^{(1)}, x, v_1^{(2)}, v_2^{(2)}, ..., v_{|v^{(2)}|}^{(2)}, x, ..., x \rangle$$

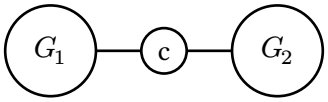
  Then for all $1 \leq t < |v|$, if $v_t = v_j^{(i)}$, we have $\mathcal{O}_t^v = \mathcal{O}_j^{v^{(i)}}$, as once a connected compoennt have been fully visited, no open edge exist as the connected compoent is only connected to x. If $v_t = x$, then $\mathcal{O}_t^v = \emptyset$. Therefore

  $$\max_{1 \leq t < |v|} |\mathcal{O}_t^v| = \max_{1 \leq i \leq n} \max_{1 \leq j < |v^{(i)}|} |\mathcal{O}_j^{v^{(i)}}| = \max_{1 \leq i \leq n} \mathrm{KLX}(G[B_i \cup \{x\}], x) = \mathrm{KLX}(G[B_1 \cup \{x\}], x)$$

  And so $\mathrm{KLX}(G) \leq \mathrm{KLX}(G, v) = \mathrm{KLX}(G[B_1 \cup \{x\}], x)$.

- $\boxed{\mathrm{KLX}(G) \geq \mathrm{KLX}(G[B_2 \cup \{x\}], x)}$ Let v be the walk of best KLX, and note $i_1, ..., i_n$ the instants where x occurs in v. Two cases:
  - If $v_0 \in B_1$, then since a path from v_0 to B_1 must go through x, there is two i_k, i_{k+1} such than $v' := \langle v_{i_k}, v_{i_k+1}, ..., v_{i_{k+1}}, x \rangle$ is a DFS walk of $G[B_1 \cup \{x\}]$, starting from x. Then $\mathcal{O}_t^{v'} \subseteq \mathcal{O}_{t+i_k-1}^v$, so $\mathrm{KLX}(G[B_2 \cup \{x\}], x) \leq \mathrm{KLX}(G)$
  - Otherwise, the path from v_0 to B_1 must pass through x, and by the same reasoning $\mathrm{KLX}(G[B_2 \cup \{x\}], x) \leq \mathrm{KLX}(G[B_1 \cup \{x\}], x) \leq \mathrm{KLX}(G)$ □

Corollary let G be a graph and a node $r \in G$ be a vertex, representing a root. Denote G' the following graph, obtained from two copies of G connected both with r to a new node c :



Then $\mathrm{KLX}(G') = \mathrm{KLX}(G, r)$. The proof is direct since $\mathrm{KLX}(G_1, r) \geq \mathrm{KLX}(G') \geq \mathrm{KLX}(G_2, r)$. We will use this construction because it "forces" the DFS of best KLX to start at some node. For example, we can use this construction to proove that the NP-Hardness stays even if we root the graph.

Prop (The cycle-lemma) For every graph G, DFS walk v, and cycle $C = \langle c_1, ..., c_n \rangle$, for every edge $e \in v \cap C$ that is both in the DFS and the cycle, there is an edge $e' \in C$ such that $e' \in \mathcal{O}_e$

Proof It all comes down to the fact that in a DFS, if there is a non-visited path from x to y when you enter x, you know that $y \in T(x)$ (y will be a child of x). This is because when entering a node x, all of the leftover graph accessible from x will be visited by the DFS, including y.

Take a directed edge $e = (c_i, c_{i+1}) \in v \cap C$ (same reasoning if the edges goes the other way with (c_{i+1}, c_i)), and consider the first vertex c_j in the same direction that have already been visited when taking e for the first time (if we loop all around, we have $c_j = c_i$, see Figure 4). Let $e' = (c_{j-1}, c_j)$, we will show that $e' \in \mathcal{O}_e$: since there is a path from c_{i+1} to c_{j-1}, $c_{j-1} \in T(c_i)$, while $c_i \in T(c_j)$, by transitivity of the parent relation, we have that $c_{j-1} \in T(c_j)$ and therefore is a backward edge the will be open during (at minima) the path from c_{j-1} to c_j in the DFS. Since (c_i, c_{i+1}) is in this path, $e' \in \mathcal{O}_e$. □

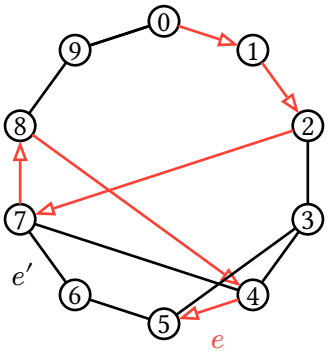


Figure 4: In red the visited edged/ build DFS at the moment of visiting $e = (4, 5)$. Here $e' = (6, 7)$, and the property of a DFS, we know that 6 will be a children of e, meaning that e' will be open when comming back from e.

Prop (The Path Lemma) Let G be a simple graph with u a degree 2 vertex. Let a, b be the two neighbour of u in G, then $G' := G - u + \{a, b\}$ and G have the same KLX.

Proof Denote v the KLX walk in G and v' the KLX walk in G'. First, notice that v can be transformed into a DFS walk of G' by removing u of the sequence, because $\mathcal{O}^{v'}_{(a,v)} = \mathcal{O}^{v}_{(a,u)} = \mathcal{O}^{v}_{(u,b)}$. And on the other way, to transform v' into a DFS walk of G, there is two cases:

- If $(a, b) \in v'$, then we insert u in beetween and we will have the same open set equality
- If $(a, b) \notin v'$, then suppose that a is the first node to be fully visited. We will have $(a, b) \in \mathcal{O}_{(a,p_a)}$ with p_a the parent of a. Replace at $\lceil a \rceil$ the sequence $\langle ...., a, p_a, ... \rangle$ by $\langle ...., a, u, a, p_a, ... \rangle$. The open sets of v will contain (u, b) instead of (a, b) everywhere where it was contained, and $\mathcal{O}^{v}_{(u,a)} = \mathcal{O}^{v}_{(a,p_a)}$ □

Remark This tells us that simple paths in the graph doesn't add complexity, and in a way we can always consider the case where the graph doesn't have any node of degree 2. Since degree 1 vertexes can also be erased as they won't open any edges (see the 1-connected lemma), the complexity of the KLX number reigns in choices for the DFS of who to visit first.

4.2. Characterizations of graphs of KLX=0 and KLX=1

I will here describe graphs of KLX number 0 and graphs of KLX number 1 using cycles. Thoses two results can be used to have a polynomial algorithm to test if a graph is of KLX number 0 or 1.

Theorem 1: For G a simple graph, $\mathrm{KLX}(G) = 0 \Leftrightarrow G$ is a tree.

Proof A tree T have $\mathrm{KLX}(T) = 0$ as no backward edges exist for the unique DFS. Let G be a graph of $\mathrm{KLX}(G) = 0$. Then there is a DFS tree T with no backward edges. Since $|E_T| = |E_G|$, we have $T = G$.

Corrolary There is a linear time algorithm to test if a graph is of KLX = 0 as we just need to test if it is a tree. Test if $|E| = |V| - 1$ and do a BFS in $O(|E| + |V|) = O(|V|)$.

We will say that by *"kissing-edge cycles"* the fact that two cycles share an edge.

Lemma A graph contains two cycles that share an edge iff it contains two cycles that contains a contigueous series of edges and all other edges are distincts.

Proof If it contains two cycles that contain a contigueous series of edges, then they share an edge. The other way is a bit more subtle. Take any two edge-kissing cycle $C = \langle c_1, c_2, ..., c_{|C|}\rangle$ and $C' = \langle c'_1, c'_2, ..., c'_{|C'|}\rangle$. Since they are distinct, suppose $c_1 \neq c'_1$. Take i the smallest integer such than $c_i \in C'$ and j the biggest integer such than $c_j \in C'$ (they exists because they are edge-kissing cycles), and note P a path from c_i to c_j in C'. Then let

$$C^* := \langle c_j, ..., c_{|C|}, c_1, ..., c_i\rangle \parallel P$$

We have C' and C^* that are edge-kissing cycle on P, and only on P by the minimality (respc. maximality) of i (respc. j) □

Theorem 2: For all G a simple graph, $\mathrm{KLX}(G) = 1 \Leftrightarrow G$ doesn't contain two kissing-edge cycles.

Proof:

- $\boxed{\Rightarrow}$ Let v be the DFS walk of $\mathrm{KLX} = 1$, and suppose by contradiction that there is two edge-kissing cycles, say $C = \langle c_1, ..., c_{|C|}\rangle$ and $C' = \langle c'_1, ..., c'_{|C'|}\rangle$ with the common edges $E = (c_1, c_2, ..., c_k) = (c'_1, c'_2, ..., c'_k)$. By the lemma we can assume that all other edges are distincts. We will first show that there exists an edge of the common edges $e \notin v$: Suppose by contradiction that all edges $E \in v$. Then by the cycle lemma on any $e \in E$, we have an edge $e_1 \in C - E$ and $e_2 \in C' - E$ such than $e_1, e_2 \in \mathcal{O}_e$; This is a contradiction as $e_1 \neq e_2$ and $|\mathcal{O}_e| \leq 1$
  Therefore we have $e \in E$ an open edge. Suppose that we visit c_1 before c_2 (the situation is symetric). Because c_2 is connected to c_1, we have $c_2 \in T(c_1)$ and therefore there is a taken path from c_1 to c_2, note it $p = \langle c_1 = p_1, ..., p_n = c_2\rangle$. Two cases:
  - If $p = C$, then by taking the cycle composed by $C - E$ followed by $C' - E$ (in the opposite direction) and applying the cycle lemma on any element x of P, we get that both $e, e' \in \mathcal{O}_x$ for $e \neq e$, a contradiction.
  - Otherwise, we have two path from c_1 to c_2 that are different for at least a vertex: $C - E$ and p. We can therefore find a cycle composed of only edges of element of $P \cup (C - E)$, and by applying the cycle lemma on any edge e' of P in this cycle we get the existence of $x \in \mathcal{O}_{e'}$, while we also have $e \in \mathcal{O}_{e'}$, and $e \neq x$, a contradiction.
- $\boxed{\Leftarrow}$ Let G be a graph without two edge-kissing cycles. Intuitively, will will show that it looks like a tree with some cycles replacing some vertexes. We will consider the DFS walk that start at any vertex, and for any edge in a cycle, visit all neighbour before the next in the cycle. Then for any backward edge e, the moments in wich the edges t is open is exacly the only cycles containing e. Since no two cycles overlap, the KLX is 1. □

Remark They can share a common vertex, just not a common edge. 42 cycles attached on a single vertex, by the 1-connected lemma on said vertex, only have a KLX of 1.

Corrolary There is a $O(V)$ algorithm to test if a graph has $\mathrm{KLX} = 1$.

Proof First, since two cycles cannot share an edge, there cannot be more than $2|V|$ edges. Then, using a DFS in which for every edge we mark if it is or not already in a cycle, we can verify that the graph is connected (all edges has been visited) and that every edge is in a single cycle by checking that is is marked only once.

4.3. Counter-Examples of conjectures

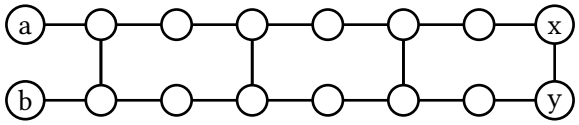
A few conjectures have been formulated during my internship, that I manage to disprove:

- A hamiltonian path (if it exists) is always the DFS tree of best KLX.
- The KLX number is monotone: removing an edge never increases the KLX number.
- There is a function of the Tree-Width that bounds the KLX number.

We shall give a counter example to each of them here.

Prop A hamiltonian path is not always the best KLX DFS tree

Proof Consider the following counter example graph:



The graph only has a single hamiltonian path v, so the best KLX for a hamiltonian path is 3 (See Figure 5)

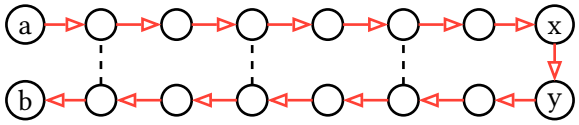
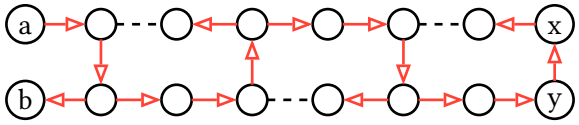


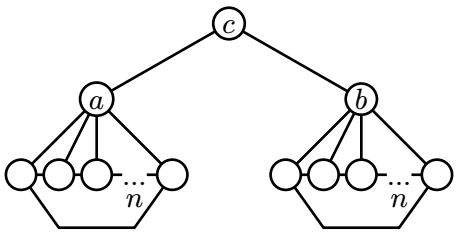
Figure 5: The only hamiltonian path. In dash, the backward edges contained in $\mathcal{O}_{\{x,y\}}$

But the following DFS tree has a KLX of 2 (because the two backward edges on the top will not be open simultaneously):



Prop Removing an edge may grow (asymptotically) the KLX by a factor of 2.

Proof Let n be an even positive number, consider the following graph G_n, formed from two cycles of length n (we will call them C_{left} and C_{right}), each connected on every node to respectfully two nodes a and b, themselved connected by a center node c. :



By the 1-connectedness lemma, we have that $\mathrm{KLX}(G[\{c,a\} \cup C_{\text{left}}], c) \leq \mathrm{KLX}(G) \leq \mathrm{KLX}\big(G\big[\{c,b\} \cup C_{\text{right}}\big], c\big) = \mathrm{KLX}(G[\{c,a\} \cup C_{\text{left}}], c)$, and one can check that $\mathrm{KLX}(G[\{a\} \cup C_{\text{left}}], a)$ is $n-1$: watever first edge e we take from a (it's a cycle so it's all symmetric), all the other edge starting from a going to C_{left} will be open, plus one due to the cycle C_{left}.

Consider adding an edge from a vertex from C_{right} to c, we will show that there is a DFS walk of KLX $= \frac{n}{2} + 2$. Note $C_{\text{left}} = \langle c_1, ..., c_n \rangle$ the vertexes of first cycle (in order) and $C_{\text{right}} = \langle c'_1, ..., c'_n \rangle$. Without loss of generality, suppose that the edge is (c, c'_1). Consider the following DFS (see Figure 7):

$$v = \langle c_1, ..., c_{\frac{n}{2}}, a, c_{\frac{n}{2}+1}, ..., c_{n-1}, c_n, c_{n-1}, ..., _{\frac{n}{2}+1}, a, c, c'_1, ..., c'_{\frac{n}{2}}, b, c'_{\frac{n}{2}+1}, ..., c'_{n-1}, c'_n,$$
$$c'_{n-1}, ..., c'_{\frac{n}{2}+1}, b, c'_{\frac{n}{2}}, ..., c'_1, c, a, c_{\frac{n}{2}}, ..., c_1 \rangle$$

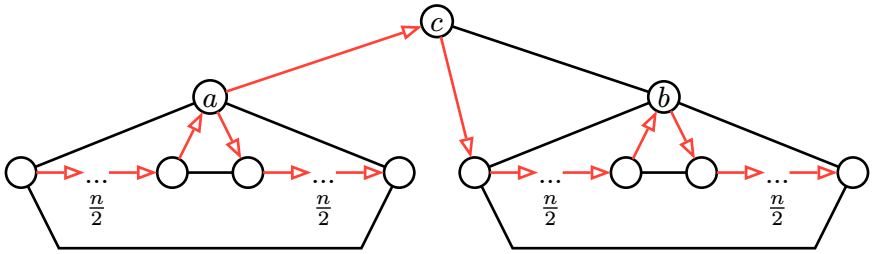


Figure 7: DFS of KLX $= \frac{n}{2} + 2$ in red arrows

The biggest values of the $|\mathcal{O}_e|$ for $e \in v$ are:
- $\frac{n}{2} + 1$ edges on the time corresponding with the edge $\left\{b, c'_{\frac{n}{2}+1}\right\}$
- $\frac{n}{2} + 2$ edges on the time corresponding with the edge $\left\{b, c'_{\frac{n}{2}}\right\}$
- $\frac{n}{2} + 1$ edges on the time corresponding with the edge $\left\{a, c_{\frac{n}{2}+1}\right\}$
- $\frac{n}{2}$ edges on the time corresponding with the edge $\left\{a, c_{\frac{n}{2}}\right\}$

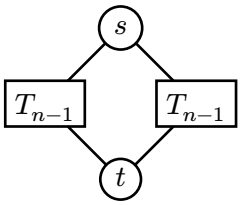
with $\text{KLX}(G_n + \{c, c'_1\}) = \frac{n}{2} + 2$. Removing the $\{c, c'_1\}$ edge from $G_n + \{c, c'_1\}$ will cause the KLX to grow by a factor of $2 - \frac{2}{n}$ □

Prop There is a familly of graphs of bounded Tree-Width and unbounded KLX

Define by induction a serie of graphs T_n for all $n \in \mathbb{N}$ with T_0 being a single edge with two vertex s and t, and T_{n+1} being made by taking two copy of T_n and connecting the two s_1, s_2 to a newly created s and simmilarly for t_1, t_2 to t:

For $n = 0$:

For $n > 0$:



Prop For all $n \in \mathbb{N}$, $\text{TW}(T_n) \leq 2$

Proof Thoses graphs are series-parrallel graph: the curious reader may want to look at [8] for a definiton of this class of graphs. An important properties of this familly: they all have a tree-width ≤ 2

Prop $\forall n > 0, \text{KLX}(T_{n+1}) \geq n$

First let's show that $\text{KLX}(T_{n+1}) \geq \text{KLX}(T_n, s)$. Intuitively, the idea is a bit simmilar to the 1-connected lemma, but instead of having $\text{KLX}(T_n, s)$ we have $\min\{\text{KLX}(T_n, s), \text{KLX}(T_n, t)\}$ (but T_n rooted in s is isomorphic to T_n rooted in t). Given a DFS walk v of T_{n+1}, there is a moment i in which it enter one T_n from either s or t (if it starts on the left T_n it will enter the right one at some point). By symmetery, suppose it enter the left T_s by s. Consider $v' = \langle v_i, ..., v_k \rangle$ from wich we removoved all vertexes that isn't in the left instance of T_n. Then v' is a DFS walk as the moment in question are precisely the subtree $T(t)$ – and removing the vertex from a sub-tree of a DFS keep the fact that it is still a DFS. Since we have for all $e \in v'$, $\mathcal{O}_e^{(v')} \subseteq \mathcal{O}_e^{(v)}$, and since v' is a DFS of T_n starting from s, we have $\text{KLX}(T_{n+1}) \geq \text{KLX}(T_n, s)$

We will show that $\mathrm{KLX}(T_n) \geq n$. For this, we will prove by induction that the path from s to t contains an edge e such than $|\mathcal{O}_e| \geq n$:

- **Initialization** For $n = 0$ we have $\mathrm{KLX}(T_0) = 0$ as T_0 is a tree.
- **Heredity** We'll note s_1, s_2 the two neighbooring vertexes of s, aka the s's of the (respectively) left and right T_n, and by symmetry, suppose the DFS (that we will call v) starts going to s_1. Then the DFS will go do a DFS walk of the left T_n (call it v'). By induction there is an edge e such than $|\mathcal{O}^{(v')}| \geq n$ But we have s_2 that will be visited, and therefore $\{s, s_2\}$ will be open for -at minimum- every edge in the path from s to s_2. Therefore since $\mathcal{O}_e^{(v')} \subseteq \mathcal{O}_e^{(v)}$ and $\{s, s_2\} \in \mathcal{O}_e^{(v)}$, we have $|\mathcal{O}_e^{(v)}| \geq n + 1$

Therefore $\mathrm{KLX}(T_{n+1}) \geq \mathrm{KLX}(T_n, s) \geq n$. □

Corollary KLX is unbounded even for graphs that are planar 3-regular and have a bounded tree width.

5. Upper and lower bounds

We will show 4 bounds of the KLX number here:

- Two NP-Hard lower bounds related to tree-width and cut-width
- An upper bound and a lower bound both relating to cycles.

5.1. $\mathrm{TW}(G) \leq \mathrm{KLX}(G) + 1$

Proof For this we transform the KLX DFS into a edge-tree a bit like a line graph. For G a graph of $\mathrm{KLX}(G) = k$, let T be the directed DFS such than $\mathrm{KLX}(T) \leq k$. Then consider the *line-graph* of T, defined as the graph T' where nodes are T's edges ($V_{T'} = E_T$) and where two edges are connected to each other if one if the direct child of the other ($E_{T'} = \{\{(x, y), (y, z)\} : x, y, z \in T_V\}$).

For an edge e (or a position in the DFS walk), denote $\mathcal{O}_e^{\mathrm{st}} \coloneqq \{o^{\mathrm{st}} : o \in \mathcal{O}_e\}$ the set of all the "deeper side" of the open edges. First, remark that the list of position t such than $u \in \mathcal{O}_e^{\mathrm{st}}$ is continuous: it is a union of interval of the form $[\![\lceil u \rceil; \lfloor v \rfloor_{\geq \lceil u \rceil}]\!]$, all starting at the same point. This will be our bags, and the intuition is that we keep thoses open until we reach the second part of the backward edge.

For every node $e = \{u, v\}$ of T', we define the bag $B_e \coloneqq \mathcal{O}_e^{\mathrm{st}} \cup \{u, v\}$. We therfore have $|B_e| \leq |\mathcal{O}_e| + 2 \leq \mathrm{KLX} + 2$. We then show that T' is a tree decomposition of G:

- By definition, every bag is a subset of V_G
- For every edge $e \in G_E$, there is two cases:
  - If $e \in E_T$, then the bag B_e contains both x and y
  - If $e = \{x, y\} \notin E_T$, then it is a backward edge. Suppose $x \leq y$. Consider $\mathcal{O}_{\lceil y \rceil_{\geq \lfloor x \rfloor} - 1}$: it's when we take the edge $\{c, y\}$ with c the child in the direction of x (such than $x \in V_{T(c)}$). Since $c \neq x$ because $e \notin E_T$, we have $\{x, y\} \in \mathcal{O}_{\lceil y \rceil_{\geq \lfloor x \rfloor} - 1}$, and therefore $x, y \in B_{\{c,y\}}$.
- We use the equivalent property: Let $e_1, e_2 \in E_T$ be two edges and let $e' \neq e_1, e_2$ be an edge in the path from e_1 to e_2. Let $u \in B_{e_1} \cap B_{e_2}$. There is 3 cases:
  - $u \in \mathcal{O}_{e_1}^{\mathrm{st}} \cap \mathcal{O}_{e_2}^{\mathrm{st}}$. Because of the continuous property, $u \in \mathcal{O}_{e'}^{\mathrm{st}}$
  - $u \notin \mathcal{O}_{e_1}^{\mathrm{st}} \wedge u \in \mathcal{O}_{e_2}^{\mathrm{st}}$. This means that e_1 is before e_2 in the DFS ordering, as having a $u \in \mathcal{O}_t^{\mathrm{st}}$ means that $t \geq \lceil u \rceil$. Therefore at least one open edge uses u, and therefore $u \in \mathcal{O}^{\mathrm{st}}_{\mathrm{parent}(e_1)}$ or $u \in \mathcal{O}^{\mathrm{st}}_{\mathrm{parent}(\mathrm{parent}(e_1))}$ for t the moment of wich e_1 occurs, depending of if u is the lower part of the upper part of the edge e_1. So either case 1 apply, or $\mathrm{parent}(e_1) = e'$ or $\mathrm{parent}(\mathrm{parent}(e_1)) = e'$. In all cases, $u \in B_{e'}$
  - $u \in e_1 \wedge u \in e_2$. This means that e_1 is a direct child of e_2 (or vice-versa), and that's in contradiction with the existence of e'

Therfore $\mathrm{TW}(G) \leq \max|B_X| - 1 + 2 \leq \mathrm{KLX} + 1$. The inequality is strict in some cases.

5.2. $\mathrm{CW}(G) \leq \mathrm{KLX}(G)\Delta(G)$

Proof Given a DFS walk $\langle v_i \rangle_i$ of minimum KLX number, pose the ordering $\varphi : V \to \mathbb{N} : x \mapsto \lceil x \rceil$ and normilize it into an ordering $\varphi : V \to [|V|]$. We will show that its cutwidth is less than $\mathrm{KLX} \times \Delta(G)$.

Note how a vertex $x \in V_G$ appears at most $\deg x + 1$ times in $\langle v_i \rangle_i$. For a given vertex $v \in V_G$, let's note $v_1, ..., v_d$ with $d \leq \deg v + 1$ the positions of v in $\langle v_i \rangle_i$. Let $1 \leq t < |V|$; Since φ is bijective, there is a $v \in V$ such than $\varphi(v) = t$, and let $e = \{x, y\} \in E$ be such than $\varphi(x) \leq t < \varphi(y)$, we will show that $e \in \bigcup_{2 \leq i \leq d} \mathcal{O}_{v_i}$. There is two cases:

- If $y = v$, then since e is a backward edge closing on v there is a i such than $e \in \mathcal{O}_{v_i}$. Note how $i \geq 2$ as $\varphi(x) \leq t$, so x cannot be seen before the first time we see v.
- Otherwise, this means that $\varphi(x) \leq t < \varphi(y)$, and therefor $v \in T(y)$, so v is in the path from x to y, therefor $e \in \mathcal{O}_{v_i}$ for all $i \leq d$.

Finally, since we have the cut of instant t that is lower than $|\bigcup_{i \leq d} \mathcal{O}_{v_i}| \leq \deg v \times \mathrm{KLX}(G)$, we have $\mathrm{CW}(G) \leq \mathrm{KLX}(G)\Delta(G)$. □

5.3. $\mathrm{KLX}(G) \leq$ number of touching cycles

Proof For an edge e, call its *cycle number* the number of cycles that passes through e. Then the maximal cycle number of an edge is an upper bound to the KLX number: Take a graph of $\mathrm{KLX} = k$, note e such than $|\mathcal{O}_e| = k$. Take $e' = \{x, y\} \in \mathcal{O}_e$: there is a path $x \rightsquigarrow e \rightsquigarrow y$ in the DFS tree, and therefor a cycle with the edge e'. But two $e', e'' \in \mathcal{O}_e$ cannot have the same cycle as otherwise an edge of a cycle is both open in one and a part of the DFS in the other. So the cycles must be distincts. □

5.4. Number of disjoint cycles $\leq \mathrm{KLX}(G)$

Proof For a given edge, call its *disjoints cycle number* the number of edge-distincts cycles that passes through it. Then the maximal *disjoints cycle number* is a lower bound to the KLX. This can easely be proven by using the Cycle Lemma on the edge of maximal disjoints cycle number and on every distinct cycle, as the open edges are required to be distincts since all cycles are distincts. □

6. NP Hardness

We are intressted in the decision problem associated with the optimisation problem of the KLX number, I will show that the KLX problem stay NP-Hard even in some very restricted class of graphs.

Consider the following two problems:

– KLX –
Input: G a graph and a $k \in \mathbb{N}$ **Output:** Is $\mathrm{KLX}(G) \leq k$?

– Rooted KLX –
Input: G a graph, $r \in V_G$ a root and a $k \in \mathbb{N}$ **Output:** Is $\mathrm{KLX}(G, r) \leq k$?

[1] already prooved that KLX is NP-Complete. We will prove that both KLX is NP-Complete even when restricted to the following classes of graphs:

- planar, bipartite and of $\Delta(G) \leq 4$
- bridgeless, planar, bipartite and of $\Delta(G) \leq 5$

For both cases, the method use will be simmilar, just with two distinct graph gadjet.

We will consider a reduction from a restricted version of HAMILTONIAN PATH, proven NP–Hard by [9][5]:

[5] Actually, they only prooved that the cycle version is NP-Hard, but by transforming an edge we know the cycle will pass through into a gadget forcing an endpoint to be here, we can consider RESTRICTED HAM. PATH instead

– RESTRICTED HAM. PATH –

Input: G a graph that is 2-connected, 3-regular, and bipartite
Output: Does there exists a hamiltonian cycle over V_G?

Since being 2-connected implies being bridgeless and since being 3-regular implies $\Delta(G) \leq 3$, we conclude that the hamiltonian cycle is NP-Hard over the class of bridgless, planar, bipartie graph of $\Delta(G) \leq 3$.

Prop KLX over the class of planar, bipartite graph of $\Delta(G) \leq 4$ is NP-Hard.

Proof Let G be an instance of RESTRICTED HAM. PATH. Let k be any upper bound to $\text{KLX}(G)$ computable in polynomial time (we may take $k = |V_G|$, but any upper bound is valid). Take $\mathbb{G}$ any 3-regular graph such than $\text{KLX}(G, r) = k$ (you can take the T_n familly of graph used for a coutner-example of bounded tree-width but unboundede KLX). We transform G into G' by attaching every node $v \in G$ to its own copy of $\mathbb{G}$ like the following

We then ask if $\text{KLX}(G') \leq k$.

Prop $\text{KLX}(G') \leq k$ iff there is a hamiltonian path.

- $\Rightarrow$ Suppose that $\text{KLX}(G') \leq k$. Then, because of the 1-connected lemma we have $\text{KLX}(G') \geq k$ and therefore $\text{KLX}(G) = k$. Let T' be a DFS tree of G' of $\text{KLX} = k$. We then take $T = G[T']$, the same tree in wich we "forgot" about the $\mathbb{G}_k$ part for every vertex. We will show that this DFS tree is a hamiltonian path. Suppose by contradiction that there exist a node $u \in T$ of degree ≥ 3. Since u may have a parent, this means that u have at least 2 children in T, let's name them x, y. Suppose that we visit x first. Since the graph G is 2-connected, u is 2-connected and therefore both $T(x)$ and $T(y)$ must be connected to somewhere. Since we are in a DFS, this must be connected to a parent of u. Therefore when visiting $T(x)$ we have to open a backward edge to a parent of u that will stay open when visiting y. So we won't be able to complete the $\mathbb{G}$ subgraph of $T(y)$ with 1 less possible open edge.
- $\Leftarrow$ If there is a hamiltonian path $\langle u_i \rangle_{i \leq n}$ in G, consider the DFS walk $\langle v_i \rangle_i$ that, whenever the DFS visit a new node $u \in G$, it will visit the connected graph with the same DDFS of KLX k before continuing with the DFS. Then for every edge in a copy of $\mathbb{G}$, we have $|\mathcal{O}_e| \leq k$ as the only open edges are from $\mathbb{G}$ and $\text{KLX}(\mathbb{G}, r) = k$. For every edge e of the graph G, since $\text{KLX}(G) \leq k$, we have $|\mathcal{O}_e| \leq \text{KLX}(G) \leq k$. □

Prop KLX over the class of planar, bridgeless, bipartite graph of $\Delta(G) \leq 5$ is NP-Hard.

Proof It's the exact same proof, but instead of attaching $\mathbb{G}$ to a node, we merge each $v \in V_G$ with a degree-two node of $\mathbb{G}$, like the following:

For example, you can use T_n and merge each v with the top-most node s of T_n. The same proof apply. □

7. Courcelle's theorem for KLX

We note $u \geq v$ if a u is a grandparent of v and $u > v$ if $u \geq v \wedge u \neq v$.

Testing if a tree is a DFS tree was already proven to be MSO_1-expressible. For example, [10] showed that trees with a DFS of height less than k are MSO_1-expressible. The method described here allows to store with a bounded amont of labels a "choice" of a neighboors for every node (the parent). This method will be built upon after showing that KLX is MSO_2-expressible, and by Courcelle's theorem this will show that for all $k, t \in \mathbb{N}$ there exists a $O(n)$ algorithm that tests if a graph of tree-width $\leq t$ has a KLX $\leq k$. We will only show the proof that $\mathrm{DFS}(T)$ is MSO_2-expressible here, putting the proof that $\mathrm{KLX}_{\leq k}$ is MSO_2 -expressible in the annexes (see Section 9.1).

7.1. DFS(T,G) is MSO_2-expressible

Given a spanning tree $T = (V, T_E)$ of $G = (V, G_E)$, let us label every edge and node in $\{\top, \bot\}$ according to the following 3 rules:

- **R1:** A node (the root) have label $\top$ and all its connected edges are labeled with $\bot$
- **R2:** For all arc $(u, v) \in T$, we have u labeled with $\top$ iff v is labeled with $\bot$
- **R3:** For all node (except the root) labeled with x, only one arc connected to the node (the arc toward the parent) have label x.

Rules **R1** and **R2** impose the label on the node u to be labeled with $\top$ iff the node u is at an even depth from the root.

Prop Given two nodes $u, v \in V$, we have an alternating sequence of $\top/\bot$ on the path from u to v iff $u \leq v$ or $v \leq u$.

Proof

We show by induction on the depth of u a node, that it is labeled $\top$ iff the parent is the only one labeled $\top$:

- **Initialization** For nodes of depth 1, they are all labeled with $\bot$ and due to **R1** all connections to the root are also labeled $\bot$ given a node u of label $\top$, its parent is of label $\bot$.
- **Inductive step** Take a node u and it's parent p. We will do the case where u is labeled $\top$ (exact same reasoning for $\bot$). Then p is labeled with $\bot$ and the only arc from p with label $\bot$ is p and its parent. Therefore the arc (u, p) have label $\top$.

Finally, given two nodes u, v, we have $u \geq v \vee v \leq u$ iff the path doesn't go from a children to a children iff there is no twice the same label.

Corollary A tree T is a DFS tree in the graph $G = (V, E)$ iff there exists a labeling of T and a node r respecting the rules **R1**, **R2** and **R3** such that for all edges $(u, v) \notin T_E$, the path from u to v is an alternating sequence of $\top/\bot$

The idea now to encode our labeling is to consider a set X that contain all nodes/arcs labeled with $\top$ (the others ones being labeled with $\bot$).

Theorem $\mathrm{DFS}(T, r, X)$ that given r a root node and T a tree, test if the set X describe a correct labeling of T that respect **R1**, **R2** and **R3** is MSO-definable.

Proof We define $\varphi_1, \varphi_2, \varphi_3$ that check each of **R1**, **R2** and **R3**, and ψ that test that all edges not covered forms an alternating labeled path of $\top/\bot$:

$$\begin{aligned}
\mathrm{DFS}(T, r, X) &:= r \in X \wedge \varphi_1 \wedge \varphi_2 \wedge \varphi_3 \wedge \psi \\
\varphi_1 &:= \forall e \in E, \forall v, (r, v) = e \Rightarrow e \notin X \\
\varphi_2 &:= \forall (u, v) \in T, u \in X \leftrightarrow v \notin X. \\
\varphi_3 &:= \forall v \in V, v \neq r \Rightarrow \exists (v, n) \in T, \forall (v, u) \in T, ((v, n) \in X \leftrightarrow v \in X) \wedge [((v, u) \in X \leftrightarrow v \in X) \rightarrow u = n]. \\
\mathrm{Parent}(c, p) &:= \exists (c, p) \in T, (c \in X \leftrightarrow (c, p) \in X) \\
\psi &:= \forall (u, v) \in E \cap \overline{T}, \forall (a, b), (b, c) \in \mathrm{Path}(u, v, T), (a, b) \in X \leftrightarrow (b, c) \notin X
\end{aligned}$$

Corollary $\mathrm{DFS}(T)$ that test if a tree T is a DFS tree is MSO-expressible.

Proof The formula is $\mathrm{DFS}(T) := \exists r, \exists X, \mathrm{DFS}(T, r, X)$

7.2. KLX_k is MSO_2 expressible and Courcelle's theorem

Theorem KLX_k that accepts only graphs of $\mathrm{KLX} \leq k$ is MSO_2-definable.

Proof See Section 9.1 in annexes for the proof.

Corollary For all $t, k \in \mathbb{N}$, there exists a $O(n)$ algorithm that test if a graph G of tree-width $\leq \mathrm{tw}$ have a KLX number less or equal to k.

Proof This is the application of Courcelle's theorem on the formula given by the proof.

8. Conclusions

While initially I was supposed to work on the oritatami model, a model of RNA folding and optimizing the turing-complete proof for a small number of bead types, a complete different subject, my intrest has shifted as my reasaerch advisor showed me this KLX number. I proceed to work on it and disprove some conjectures. And we decided to change the subject a month in to be fully around the KLX number. All results in this document are by me unless explicitly stated, including the counter examples.

Results An overview of my results:

- Some lemmas and counter-examples to give a better understanding of the way the KLX number works
- Two lower bounds, both linked to tree-width and cut-width
- A computable lower and upper bound linked to cycles
- A fixed-parameter-tracable (aka polynomial if specific values are bounded) algorithm to test if a graph is of $\mathrm{KLX} \leq k$ using Courcelle's Theorem
- A proof of NP-hardness for a restricted class of graphs

Challenges The biggest challenge I faced was the unknown of the KLX number. DFS are far from intuitive, as some counter examples can show. It's a deeply non-local mesure that's the basis for the KLX number. It's very hard to find good upper bounds as building a DFS from a mesurement that has nothing to do with DFS is very complicated. But once an idea was found, putting it into writing wasn't as hard as it could have been. To this day, I have no idea if an approximation algorithm is possible or not.

8.1. The work environment

The lab ambiance was exeptionally good. We spent multiple evening with the other students and my reasearch advisor in different restaurants. Mr Seki also brought me to a trip to Akita University to work in a group of other researchers on open problems in RNA-sequencing, and to present some of my results. A good ambiance allowed me to feel welcome and even help the other students by teaching them about graph theory, computability and langage theory. The university also had a club for international students that I joined and was a part of every friday evening.

Bibliography

[1] S. Seki, A. Elonen, and P. Orponen, "Secondary Structure Design for Cotranscriptional 3D RNA Origami Wireframes," *31st International Conference on DNA Computing and Molecular Programming (DNA31)*, 2025.

[2] A. Elonen *et al.*, "Algorithmic Design of 3D Wireframe RNA Polyhedra," *ACS Nano*, vol. 16, no. 10, pp. 16608–16616, 2022, doi: 10.1021/acsnano.2c06035.

[3] C. W. Geary and E. S. Andersen, "Design Principles for Single-Stranded RNA Origami Structures," in *DNA Computing and Molecular Programming*, S. Murata and S. Kobayashi, Eds., Cham: Springer International Publishing, 2014, pp. 1–19.

[4] A. Mohammed, P. Orponen, and S. Pai, "Algorithmic Design of Cotranscriptionally Folding 2D RNA Origami Structures," in *Unconventional Computation and Natural Computation*, S. Stepney and S. Verlan, Eds., Cham: Springer International Publishing, 2018, pp. 159–172.

[5] H. L. Bodlaender, "Approximation Algorithms for Treewidth, Pathwidth, and Treedepth—A Short Survey," in *Graph-Theoretic Concepts in Computer Science*, D. Kráľ and M. Milanič, Eds., Cham: Springer Nature Switzerland, 2025, pp. 3–18.

[6] B. Courcelle, "The monadic second-order logic of graphs. I. Recognizable sets of finite graphs," *Information and Computation*, vol. 85, no. 1, pp. 12–75, 1990, doi: https://doi.org/10.1016/0890-5401(90)90043-H.

[7] E. Korach and N. Solel, "Tree-width, path-width, and cutwidth," *Discrete Applied Mathematics*, vol. 43, no. 1, pp. 97–101, 1993, doi: https://doi.org/10.1016/0166-218X(93)90171-J.

[8] B. Courcelle and J. Engelfriet, *Graph Structure and Monadic Second-Order Logic*, Encyclopedia of Mathematics and Its Application 138. Cambridge.

[9] T. Akiyama, T. Nishizeki, and N. Saito, "NP-completeness of the Hamiltonian cycle problem for bipartite graphs," *J. Inform. Process.*, vol. 3, no. 2, pp. 73–76, 1980.

[10] E. Sam, M. Fellows, F. Rosamond, and P. A. Golovach, "On the Parameterized Complexity of the Structure of Lineal Topologies (Depth-First Spanning Trees) of Finite Graphs: The Number of Leaves," in *Algorithms and Complexity*, M. Mavronicolas, Ed., Cham: Springer International Publishing, 2023, pp. 353–367.

9. Annexes

9.1. $\mathrm{KLX}_{\leq i}$ is MSO_2-expressible

In this section, let $k \in \mathbb{N}$ be the KLX number we want to test. Given a tree $T = (V, T_E)$ of $G = (V, E)$, we will note $\overline{T_E} = \mathrm{Compl}_E(T_E)$ the complement of T_E in E (and not in V^2). We will for ease of use use $p(x)$ as a functional symbol in our formulas to denote the parent of x, knowing that $R(p(x))$ can be replaced by $\exists t, \mathrm{Parent}(x,t) \wedge R(t)$.

Important notion The DFS alone is not enough to characterize a KLX DFS walk: the order of the visit of the children of node can change the value of the KLX.

Given a rooted DFS tree T of $G = (V, E)$, we create k sets of edges/nodes $E_1, ..., E_k$ and for every node $u \in V$ and every i such than $u \in E_i$ we choose two specific neighbors of u (called u_i^1 and u_i^2). All choices must respect that:

- **R1** For every node $u \in V$, there is no cycle of the form $u_{i_1}^2 = u_{i_2}^1, u_{i_2}^2 = u_{i_3}^1, ..., u_{i_p}^2 = u_{i_1}^1$ with all $i_1, ..., i_p$ different
- **R2** For every node $u \in V$, there is only two nodes that appears once in the list of all chosen neighbors (over all possible i), one of those is the parent of u, and all other appears twice
- **R3** Every edge $(u, v) \notin T_E$ is in exactly one E_i

Claim Due to **R1** and **R2**, for every $u \in V$ there is a unique way to order all chosen neighbors of the form $p(u) = u_{i_1}^1, u_{i_1}^2 = u_{i_2}^1, ..., u_{i_{p-1}}^2 = u_{i_p}^1$. We will note by $\left(i_1, ..., i_p\right)$ this ordering of the labeled neighbors of u.

For $(u, v) \notin T_E$ define $X_{u,v}$ the set inner edges and nodes of the open edge (u, v) as the smallest set respecting:

- **X1** The path from u to v is in $X_{u,v}$
- **X2** For every u strictly in the path, we have u_i^2 in the path implies that for every j strictly before i in the ordering of node u, we have the full sub-tree of child u_j^2 in $X_{u,v}$

Finally, all must respect the following conditions:

- **R4** For all $(u, v) \in T_E$, we have $(u, v) \in E_i$ iff there exists $(x, y) \notin T_E$ such than $(u, v) \in X_{x,y}$ and $(x, y) \in E_i$
- **R5** For all $(u, v), (u', v') \notin T_E$, if $(u, v), (u', v') \in E_i$ then $X_{u,v} \cap X_{u',v'} = \emptyset$

Proof of Claim Let u be a node, and consider the graph $G_u = (V_u, E_u)$ with V_u all chosen neighbors of u and E the set of edges of the form $\left\{u_i^1, u_i^2\right\}$. Then **R2** directly describe that for all nodes expect two, the degree is two, meaning G_u is a disjoint union of cycles and a path with an endpoint being $p(u)$. Rule **R1** impose the graph to not have any cycle, making it a single long path with $p(u)$ as an endpoint.

Intuition The idea is that for every open edge $(u, v) \in E \setminus T_E$, we have have all moments (edges and node expect endpoint u, v) in which the edge is open (described by $X_{u,v}$) contain in the same set E_i. Since a DFS KLX walk is intransigeantly ordered, and the order will change the KLX number, the chosen nodes is here to make sure that all E_i are consistent with the order of visit. And due to the following remark we can bound the "useful" order by k, making it MSO-definable:

Remark Consider a KLX DFS walk and a node $x \in V$. For every open edge (u, v) (with $u > v$), either it will not be open upon meeting with the node (if $x > u$ or $v > x$), will close as soon as coming back from the node ($u = x$), or will always be open ($u > x \geq v$).

Lemma (Correction of X1, X2) For every $(a, b) \in \overline{T_E}$, for all $(u, v) \in T_E$, we have $(u, v) \in X_{a,b}$ iff (a, b) is open upon the $u \to v$ transition of the DFS.

Proof of lemma

Take $(v_i)_i$ the DFS and $(a,b) \in \overline{T_E}$. Take $i = \lceil a \rceil$ the last position of a in the DFS and $j = \lfloor b \rfloor_{\geq i}$ the first position of b greater than i: they delimit by definition the moments where the edge (a,b) is open. Let's show that $X_{a,b} = \{(v_k, v_{k+1}) : i \leq k \leq j-1\}$. By definition of a DFS, we have that $\lceil t \rceil$ is the position of the transition $(t, p(t))$ in the DFS.

$\boxed{\Rightarrow}$ Take $(u,v) \in X_{a,b}$. Then, we have two cases :
- If (u,v) is in the path from a to b, we have $i \leq \lceil u \rceil < \lceil u \rceil + 1 \leq j$.
- Suppose that (u,v) is in a sub-tree of a inner node w of the path. Then the lowest common parent of a and u is w, and because the tree is a DFS, we explore every children of w before going back, so $\lceil u \rceil < \lceil p(w) \rceil \leq j$ and this conclude this case.

$\boxed{\Leftarrow}$ Take a $(u, v = p(u))$ of position $t, t+1$ in the DFS, with $i \leq t \leq j$. Then $\lceil u \rceil = t$. Take p the lowest common ancestor of u and a. Since both $u \leq b$ and $a \leq b$ we have $p \leq b$. We have two cases:

- if $p = u$, then u is in the path as $a \leq p = u \leq b$, so by **X1** we have $(u,v) \in X_{a,b}$
- Otherwise, both $a, b \leq p \leq b$. Since we have $i \leq \lceil u \rceil \leq \lceil p \rceil$, a is visited before u. Since there is an open edge (a,b), we conclude that both the child going to a and every later child are order in the DFS (and have a p_i^2 associated) and therefore by **X2** we have $(u,v) \in X_{a,b}$. For this case it's not possible that $b = p$, as otherwise $\lceil u \rceil \leq \lfloor b \rfloor_{\geq a} \leq \lfloor u \rfloor_{\geq a}$ (a contradiction)

Corollary Given a DFS, the set $X_{a,b}$ is unique.

Prop Given a a graph and T a DFS tree of G, we have $\mathrm{KLX}(G) \leq k$ iff there exists k sets $E_1, ..., E_k$ and a choice of nodes as described above following all described rules.

Proof Both implication:

$\boxed{\Rightarrow}$ Let G be a graph of $\mathrm{KLX}(G) \leq k$, and take T the DFS tree. Take $(v_i)_i$ the DFS walk. We will define the edge set $(E_i)_{i \leq k}$ with the induction hypothesis that $|\{(a,b) \in E_i\}|$ is the KLX number on the time of the transition of the edge (a,b) in the DFS:
- Initially, $\forall i, E_i = \emptyset$
- For a given transition of the DFS $(x, p(x))$, for all backward edge of the form (x,t), there is a $j \leq k$ such than $(x, p(x)) \notin E_j$ (otherwise we would have more than k open edges at the same time). Add the moments where (x,t) is open in E_j (by the lemma, those are $X_{x,t}$), and add $(x,t) \in E_j$.

Now to define the ordering. Given a node $u \in V$, there is at maximum k open edges during the transition $(u, p(u))$ in the DFS. Take the reverse order of thoses children and encode it in u as per the claim.

Let's verify that $E_1, ..., E_k$ and the labeling verify **R1** to **R5** :
- **R1** and **R2** are by definition, and because of the claim
- **R3** By definition, for every backward edge (u,v), we added it in a edge set when considering the $(u, p(u))$ transition in the DFS
- **R4** For a given backward edge (a,b), we only add in the $E_i \cap T_E$ set the edges in where (a,b) is open. Because we only add an edge in that case and start with the empty set for E_i, by induction, those are the only ones who persists.
- **R5** By contradiction, given two backward edges (u,v) and (u',v'), suppose that $X_{u,v} \cap X_{u',v'} \neq \emptyset$. Then there is an edge (a,b) that have been added twice in a set E_i. Suppose that (u,v) appears before (u',v') in the DFS (the other way being the same reasoning). Then when considering the set (u',v'), we have $(u', p(u')) \notin E_i$ as this is the necessary condition for it to be added in the E_i set. But because of the lemma, we know that $X_{a,b}$ represent a continuous interval of the DFS walk, so (a,b) can't be in $X_{u,v}$ as it comes after $(u', p(u'))$, a contradiction.

$\boxed{\Leftarrow}$ Supposing the existence of all set following the rules described, we will show that there exists a KLX DFS walk such that for all transition (u,v) in the KLX DFS walk, the number of open edges at this point is bounded by k. For this, since we already have the DFS tree, we just need to indicate the order of visit

of every child of a node. Given a node u, note $c_1, ..., c_p$ the sub-tree of child $i_1, ..., i_p$ respectively (with $i_1, ..., i_p$ the order stored at node u that is well defined by rule **R1** and **R2** according to **Claim 1**). We define the DFS KLX order of u as going first to every sub-tree of u that isn't in $\{c_1, ..., c_p\}$ (in whatever order) before going to the sub-trees $c_p, c_{p-1}, ..., c_1$ in that order.

Finally, to show our bound, we will show that for every transition (u, v) in the DFS, we have $|\mathcal{O}_{\{u,v\}}| = |\{i : (u, v) \in E_i\}| \leq k$ (read $\underset{\text{card}}{\equiv}$ as "have the same cardinal"):

$$
\begin{aligned}
\mathcal{O}_{\{u,v\}} &\underset{\text{card}}{\equiv} \left\{(a, b) \in \overline{T_E} \mid (u, v) \in \text{moments where } (a, b) \text{ is open}\right\} \\
&\underset{\text{card}}{\equiv} \left\{(a, b) \in \overline{T_E} \mid (u, v) \in X_{a,b}\right\} && \text{By Lemma} \\
&\underset{\text{card}}{\equiv} \left\{(a, b) \in \overline{T_E} \mid \exists i, (u, v) \in X_{a,b} \wedge (u, v) \in E_i\right\} && \text{By R3} \\
&\underset{\text{card}}{\equiv} \left\{(a, b, i) \in \overline{T_E} \mid (u, v) \in X_{a,b} \wedge (u, v) \in E_i\right\} \\
&\underset{\text{card}}{\equiv} \left\{i \mid \exists (a, b) \in \overline{T_E} \wedge (u, v) \in X_{a,b} \wedge (u, v) \in E_i\right\} && \text{By R5} \\
&\underset{\text{card}}{\equiv} \{i \mid (u, v) \in E_i\} && \text{By R4}
\end{aligned}
$$

Theorem KLX_k that accepts only graphs of $\text{KLX} \leq k$ is MSO-definable.

Proof We need to find a way to encode the choice of nodes as sets of edges/nodes. Let $u \in V$ be a node, because the size of the ordering is bounded by k (can't have more than k open edges when taking $(u, p(u))$), we can store the two nodes u_i^1 and u_i^1 in a set called O_i^u. Since we can't have an unbounded number of sets (they are all quantified) we need to share nodes in-between O_i^u. To avoid ambiguity, for every i we create O_i^0, O_i^1 and O_i^2 for node of a depth of respectively 0, 1 or 2 mod 3.

For a node u at a depth mod 3 of x, we therefore have $\{u_i^1, u_i^2\} = O_i^x \cap N(u)$ with $N(u)$ the set of neighbors of u. We must therefore have the condition that for every node u of depth $x \bmod 3$, $|O_i^x \cap N(u)| = 2$

We use $\varphi_1, \varphi_2, \varphi_3, \psi_1, \psi_2, \varphi_4, \varphi_5$ that each check respectively of **R1**, **R2**, **R3**, **X1**, **X2**, **R4**, **R5**. $\text{DEPTH}_x(u)$ test is the depth of u is $x \bmod 3$. γ test that $O_1^0, O_1^1, ..., O_k^2$ encode the choice. $\text{SEL}_i(u, u^1, u^2)$ is true iff u^1 and u^2 are the two nodes chosen for u and i. That gives us the following formulas:

$$\mathrm{KLX}_{\leq k} \coloneqq \exists T, \exists r, \exists X, \mathrm{DFS}(T, r, X) \wedge \left(\exists E_1, ..., \exists E_k, \exists O_1^0, \exists O_1^0, ..., \exists O_k^2, \gamma \wedge \varphi_1 \wedge \varphi_2 \wedge \varphi_3 \wedge \varphi_4\right)$$

$$\mathrm{DEPTH}_x(u) \coloneqq \exists A_0, \exists A_1, \exists A_2, r \in A_0 \cap \overline{A_1} \cap \overline{A_2} \wedge \left(\forall v \in V, \bigwedge_{i \in \{0,1,2\}} p(v) \in A_i \leftrightarrow v \in A_{(i+1) \bmod 3}\right) \wedge u \in A_x$$

$$\gamma \coloneqq \forall u, \bigwedge_{\substack{x \in \{0,1,2\} \\ 1 \leq i \leq k}} \mathrm{DEPTH}_x(u) \rightarrow \left(\exists u^1, u^2 \in O_i^x, (u, u^1), (u, u^2) \in E \wedge \left(\forall c \in O_i^x, (u, c) \in E \Rightarrow c = u^1 \vee c = u^2\right)\right)$$

$$\mathrm{SEL}_i(u, u^1, u^2) \coloneqq \bigwedge_{x \in \{0,1,2\}} \mathrm{DEPTH}_x(u) \rightarrow u^1, u^2 \in O_i^x \wedge (u, u^1), (u, u^2) \in E$$

$$\varphi_1 \coloneqq \forall u \in V, \neg \bigwedge_{\substack{p < k \\ 1 \leq i_0, ..., i_p \leq k}} \bigwedge_{0 \leq j \leq p} \exists u^2, a, b \in V, \mathrm{SEL}_{i_j}(u, a, u^2) \wedge \mathrm{SEL}_{i_{j+1 \bmod p+1}}(u, u^2, b)$$

$$\varphi_2 \coloneqq \forall u, \exists p, e$$

$$\mathrm{IsAPath}(P, u, v) \coloneqq \exists r \in V, u \in P \wedge v \in P \wedge \left(\forall k \in P, k \neq r \rightarrow p(k) \in P \wedge (k, p(k)) \in P\right)$$

$$\mathrm{IsThePath}(P, u, v) \coloneqq \mathrm{IsAPath}(P, u, v) \wedge \forall P', \mathrm{IsAPath}(P', u, v) \rightarrow P \subseteq P'$$

$$\mathrm{IsChildren}(a, b) \coloneqq \exists P, \mathrm{IsThePath}(P, a, b) \wedge p(a) \in P \wedge \forall x, y \in P, x \neq y \rightarrow p(x) \neq p(y)$$

$$\psi_1\left(u, v, X_{u,v}\right) \coloneqq \forall P, \mathrm{IsThePath}(P, u, v) \rightarrow P \subseteq X_{u,v}$$

$$\psi_2\left(u, v, X_{u,v}\right) \coloneqq \forall P, \mathrm{IsThePath}(P, u, v) \rightarrow \forall x \in P, x \neq u \wedge x \neq v \rightarrow \bigwedge_{1 \leq p \leq k} \bigwedge_{1 \leq i_1, ..., i_p \leq k} \exists u_1^1, ..., \exists u_p^2, \left(\bigwedge_{1 \leq j \leq p} \mathrm{SEL}_{i_j}(u, u_j^1, u_j^2)\right) \wedge \left(\bigwedge_{1 \leq j < p} u_j^2 = u_{j+1}^1\right) \wedge u_1^1 = p(u) \wedge \bigwedge_{1 \leq j < i \leq p} u_i^2 \in X_{a,b} \rightarrow \left(\forall (v, v') \in E, \mathrm{IsChildren}\left(v, u_j^2\right) \rightarrow v \in X_{a,b} \wedge (v, v') \in X_{a,b}\right)$$

$$\mathrm{IsASet}\left(u, v, X_{u,v}\right) \coloneqq \psi_1\left(u, v, X_{u,v}\right) \wedge \psi_2\left(u, v, X_{u,v}\right)$$

$$\mathrm{IsTheSet}\left(u, v, X_{u,v}\right) \coloneqq \mathrm{IsASet}\left(u, v, X_{u,v}\right) \wedge \forall X', \mathrm{IsASet}(u, v, X') \rightarrow X \subseteq X'$$

$$\varphi_3 \coloneqq \forall e \in E, e \notin T \Rightarrow \bigvee_{1 \leq i \leq k} e \in E_i \wedge \left(\bigwedge_{\substack{1 \leq j \leq k \\ i \neq j}} e \notin E_j\right)$$

$$\varphi_4 \coloneqq \forall (u, v) \in T_E, \bigwedge_{1 \leq i \leq k} (u, v) \in E_i \leftrightarrow \exists (x, y) \in E, \exists X_{x,y}, \mathrm{IsSet}\left(x, y, X_{x,y}\right) \rightarrow \notin T \wedge (u, v) \in X_{x,y} \wedge (x, y) \in E_i$$

$$\varphi_5 \coloneqq \forall (u, v), (u', v') \in E, (u, v), (u', v') \notin T_E \rightarrow \bigwedge_{1 \leq i \leq k} (u, v), (u', v') \in E_i \rightarrow \exists X_1, X_2, \mathrm{IsTheSet}\left(u, v, X_{u,v}\right) \wedge \mathrm{IsTheSet}\left(u', v', X_{u',v'}\right) \wedge \left(\forall t, \neg\left(t \in X_{u,v} \wedge t \in X_{u',v'}\right)\right)$$