Extending the Tractability of the Clique Problem via Graph Classes Generalizing Treewidth

Philippe Jégou
Aix Marseille Univ, Université de Toulon, CNRS, LIS, Marseille, France
philippe.jegou@univ-amu.fr

Abstract

The study of the Clique problem in algorithmic graph theory is important both because it is a central problem in complexity theory, almost at the same level as SAT (Johnson and Trick 1996), but also because its practical resolution has many applications, notably in Artificial Intelligence (e.g. checking consistency of a binary CSP is equivalent to check the size of maximum clique in its microstructure (Jégou 1993)). A great deal of work has therefore been carried out, and this paper attempts to extend the results obtained in this field. It starts from the observation that this problem is tractable in polynomial time on graphs whose treewidth (Robertson and Seymour 1986) are bounded by a constant (see (Courcelle 1990)). Although this type of result is very interesting from a theoretical point of view, it often remains limited in terms of application. So, we propose here an extension of this type of approach based on the definition of graph classes wider than those of bounded treewidth, but for which we would be able to propose polynomial time algorithms. These graph classes denoted $\mathcal{C}_W^k$ include graphs of treewidth $W + k$, but also contain, even if k and W are constants, graphs known to be of unbounded treewidth. These $\mathcal{C}_W^k$ classes are introduced, their fundamental properties are given and the associated algorithms are presented and evaluated.

Introduction

In Artificial Intelligence algorithms, the Clique Problem in a graph plays a significant role, as it can be found in a large number of application domains. For example, we know that a binary CSP has a solution if and only if its microstructure expression has a clique whose size corresponds to the number of variables in the instance. Beyond this domain, this problem, which is NP-complete, has a wide range of applications in optimization, and is, along with SAT, one of the central problems of Complexity Theory. So this is a problem that can legitimately be the subject of a study in its own right. Its statement is very simple: given a (undirected) graph G and an integer k, the question is whether G has a complete subgraph of k (or more) vertices. The associated optimization problem, called Maximum Clique Problem, consists of determining the largest clique in a graph.

If there are tractable classes (i.e. tractable in polynomial time), as with many other difficult problems, one way of tackling it is to use FPT-type approaches (Downey and Fellows 1999), even if Clique is not FPT. But this is made possible by using the notion of tree-decomposition of graphs and the associated notion of treewidth (Robertson and Seymour 1986). This notion now plays a fundamental role in many fields, notably in the efficient processing of graphical models (CSP, WCSP, cost function networks, Bayesian networks, etc.). Fundamental results with first Arnborg (Arnborg 1985), then Courcelle's meta-theorem (Courcelle 1990), have shown that for the case where a graph has a treewidth bounded by a constant, many problems, including the Clique problem, can then be handled in polynomial time.

In this paper, we address these issues by defining a theoretical tool related to the notion of tree-decomposition, for constructing an infinite number of recusively defined graphs. Under certain assumptions, these classes allow us to propose polynomial time algorithms for solving the Clique problem. These classes of graphs, called $\mathcal{C}_W^k$, are defined by considering two parameters, k and W. We show that any graph G in $\mathcal{C}_W^k$ is recognizable in $O(n^{2k} \times T(G, W))$ where T depends on G and W, W being the treewidth of a graph accessible after k operations on G. We show that if this parameter W is a constant, then $T(G, W)$ is a polynomial factor, which means that instance belonging to $\mathcal{C}_W^k$ can be processed in polynomial time. In the approaches based on tree-decomposition , the difficulty often arises from the fact that classes of graphs have treewidths that are potentially unbounded by constants, as it is the case for planar graphs and complete bipartite graphs. This problem seems to disappear for some of these classes, such as the two mentioned above, with the use of $\mathcal{C}_W^k$ classes. For example, we show that all planar graphs belong to class $\mathcal{C}_3^1$, even though the treewidth of these graphs is in $\Theta(\sqrt{n})$ for graphs of n vertices. We also show that any graph of treewidth $W + k$ belongs to class $\mathcal{C}_W^k$, which leads us to hope that this double parameterization offers a more accurate view of instance difficulty than that indicated by treewidth alone.

This paper is organized as follows. In the next section, we introduce notations and basic notions. In the third section, we define $\mathcal{C}_W^k$ classes and present two fundamental properties, one of which is used in the next section where we exhibit an algorithm for recognizing these classes. In the fifth section, we present an algorithm for solving the clique problem for the case of graphs belonging to a $\mathcal{C}_W^k$ class, and the analysis of its complexity allows us to highlight tractable cases while the sixth section show that some classes of graphs of unbounded treewidth appear in classes

$\mathcal{C}_W^k$ such that k and W are constants. Finally, to conclude this paper, we discuss the relevance of these new classes as a general tool complementary to tree-decomposition.

Preliminaries

First of all, we remind a few notations and definitions. Let $G = (V, E)$ be a finite undirected graph with V the set of vertices and E the set of edges. An edge of E is denoted $\{x, y\}$ with $x, y \in V$. We use n to denote the number of vertices (so $|V| = n$) and e to denote the number of edges (so $|E| = e$). Given a subset of vertices $X \subseteq V$, the graph $G[X]$ denotes the subgraph of G induced by the subset of vertices X. A clique of a graph is a subset K of the vertices such that every two vertices $x, y \in K$ is an edge, i.e. $\{x, y\} \in E$. A complete graph is a graph $G = (V, E)$ such that V is a clique of G (It is usually referred to as K_n). Given a graph $G = (V, E)$, for each vertex $x \in V$, the neighborhood of x is the set $N(x) = \{y \in V : \{x, y\} \in E\}$. If $\sigma = [x_1, x_2, \ldots, x_n]$ is an ordering of V, the successors of a vertex x_i are the elements of the set $N^+(x_i) = \{x_j \in N(x) : i < j\}$. Now, we recall the definition of the *tree-decomposition* of a graph (Robertson and Seymour 1986) and the its associated parameter called *treewidth*:

Definition 1 *A* tree-decomposition *of a graph $G = (V, E)$ is a pair (B, T) where $T = (I, F)$ is a tree (I is a set of nodes and F a set of edges) and $B = \{B_i : i \in I\}$ a family of subsets of V such as every $B_i \in B$ (called* bag*) corresponds to a node i of T and satisfies:*

1. $\cup_{i \in I} B_i = V$,
2. $\forall \{x, y\} \in E$, $\exists i \in I$ *such that* $\{x, y\} \subseteq B_i$, *and*
3. $\forall i, j, k \in I$, *if k is on a path between i and j in T, then* $B_i \cap B_j \subseteq B_k$

The width *of a tree-decomposition is equal to $max_{i \in I}|B_i| - 1$ and the* treewidth *of G denoted $tw(G)$ is equal to the minimum width among all the tree-decompositions of G.*

A Generalization of Treewidth by Defining New Graph Classes

Here we define an infinite number of recursively defined graph classes that generalize graphs of a given treewidth, and therefore also graphs of bounded treewidth. We will see that such a definition can be seen as a generalization of graph treewidth.

Definition 2 *Let $W \in \mathbb{N}$ be a constant. The class of graphs $\mathcal{C}_W^0$ is the set of graphs G whose treewidth is at most W, that is $tw(G) \leq W$. Given a natural number $k > 0$, the class of graphs $\mathcal{C}_W^k$ is the set of graphs $G = (V, E)$ such that there is an ordering $\sigma = [x_1, x_2, \ldots, x_n]$ of V, such that, for $i = 1, 2, \ldots, n$, the subgraph $G[N_\sigma^+(x_i)]$ belongs to $\mathcal{C}_W^{k-1}$. Such an ordering σ is called $\mathcal{C}_W^k$* scheme*.*

We illustrate this definition below, showing in particular that the same graph can belong to different classes of type $\mathcal{C}_W^k$. This is simply made possible by the use of a double parameterization with k and W.

Since the treewidth of the graph in Figure 1 is 4, this graph trivially belongs to $\mathcal{C}_4^0$. Moreover, for this graph, whatever the ordering σ considered on its vertices, the neighborhood of each vertex has a treewidth at most equal to two. Morevover, there will always be at least one vertex whose neighborhood has a treewidth equal to two. So, for each vertex x, the subgraph $G[N_\sigma^+(x)] \in \mathcal{C}_2^0$. Therefore, this graph belongs to $\mathcal{C}_2^1$ since it admits a $\mathcal{C}_2^1$ scheme. Although this is more difficult to observe directly, we can also show that this graph belongs to the class $\mathcal{C}_1^2$. We can see that this graph also belongs to many other classes, by simply changing the value of parameter W. For example, it belongs to any $\mathcal{C}_W^0$ class, taking $W \geq 4$.

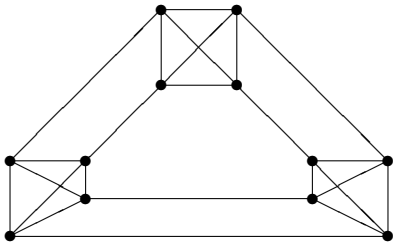

Figure 1. A graph whose treewidth is 4.

We give now a fundamental property about these classes.

Theorem 1 *$\forall G$, if $tw(G) \leq W + k$, then $G \in \mathcal{C}_W^k$.*

Proof. We prove this property by induction on k.

First, for the base case, consider $G \in \mathcal{C}_W^0$, i.e. $k = 0$. By definition, $tw(G) \leq W$. Note that this result holds also for $k = 1$. Indeed, for $G = (V, E)$ such that $tw(G) \leq W + 1$, there is a tree-decomposition of G such as, for all bags B_j, $|B_j| \leq W + 2$. So, there is an ordering $\sigma = [x_1, x_2, \ldots, x_n]$ on V, such that, $\forall i, 1 \leq i \leq n$, $tw(G[N_\sigma^+(x_i)]) \leq W$. To ensure this, it is sufficient to see that if one consider a leaf B_j of the tree-decomposition, its size is at most $W + 2$. In this bag B_j, consider a vertex x which is not included in a neighboring bag. So, its neighborhood $N(x)$ is exactly $B_j \backslash \{x\}$ whose the size is at most $W + 1$, and thus, the treewidth of $G[N(x)]$ is at most W. A such vertex x can be considered as the first one in the ordering σ. A similar reasoning can be extended to all the bags of such a tree-decomposition, and also, to all the vertices of G to complete the ordering σ. Therefore, the ordering σ is then a $\mathcal{C}_W^1$ scheme of G.

Now, assume that the property is true for all k', such as $0 \leq k' < k$, that is, if $tw(G) \leq W + k'$, then $G \in \mathcal{C}_W^{k'}$. We prove that this property also holds for k. Let $G = (V, E)$ be a graph such as $tw(G) \leq W + k$. So, there is a tree-decomposition of G such as, for all bags B_j, $|B_j| \leq W + k + 1$. By the same reasoning as for $k = 1$, we can define an ordering on $\sigma = [x_1, x_2, \ldots, x_n]$ on V, such that, $|N_\sigma^+(x_i)| \leq W + k$, and then, $\forall i, 1 \leq i \leq n$, $tw(G[N_\sigma^+(x_i)]) \leq W + k - 1$. So, using the induction hypothesis, $G \in \mathcal{C}_W^{k-1}$ and by definition, a such ordering σ is a $\mathcal{C}_W^k$ scheme of G, and thus $G \in \mathcal{C}_W^k$. QED.

This basic property on $\mathcal{C}_W^k$ classes thus allows us to consider that this type of class makes it possible to generalize the notion of treewidth of graph using the additional parameter

k. Indeed, we can reduce any graph of treewidth at most w to graphs of parameters W and k, with $W + k = w$, knowing that the graphs of treewidth at most w already belong to the base class $\mathcal{C}_w^0$. And therefore, we can even estimate that the notion of tree decomposition of a given treewidth is generalized by classes with double parameters W and k.

The following property shows the hereditary structure of all these classes, that is, for a graph of a given class $\mathcal{C}_W^k$, all its subgraphs belong to this class. We will see later that this property is of interest from an algorithmic point of view.

Theorem 2 *$\forall W \geq 0$, $\forall k \geq 0$, $\forall G = (V, E) \in \mathcal{C}_W^k$, then $\forall X \subseteq V, G[X] \in \mathcal{C}_W^k$.*

Proof. We prove this property by induction on k. For $k = 0$, the property holds since graphs that belong to $\mathcal{C}_W^0$ are graphs whose the treewidth is bounded by W and since it is well known that every subgraph of a graph of a given treewidth has at most the same treewidth. So every subgraph $G[X]$ of a graph $G \in \mathcal{C}_W^0$ belongs too to $\mathcal{C}_W^0$.

Suppose now that this property holds $\forall k', 0 \leq k' < k$. Let $G = (V, E)$ belonging to $\mathcal{C}_W^k$ and $x \in V$. We prove that $G[V \backslash \{x\}] = G'$ belongs to $\mathcal{C}_W^k$. Let the ordering $\sigma = [x_1, x_2, \ldots, x_n]$ be the associated $\mathcal{C}_W^k$ scheme of G. Consider now σ' the ordering $[x_1, \ldots x_{i-1}, x_{i+1}, \ldots x_n]$ of $V \backslash \{x\}$ where $x = x_i$. We show that the ordering σ' is a $\mathcal{C}_W^k$ scheme of G':

- $\forall j, i+1 \leq j \leq n$, it is clear that with respect to the orderings σ and σ', $G[N^+(x_j)] = G'[N^+(x_j)]$ and since G belongs to $\mathcal{C}_W^k$, then $G'[N^+(x_j)] = G[N^+(x_j)]$ belongs to $\mathcal{C}_W^{k-1}$.
- $\forall j, 1 \leq j \leq i-1$, it is clear that with respect to the orderings σ and σ', $G'[N^+(x_j)]$ is a (non-necessarily strict) subgraph of $G[N^+(x_j)]$ which belongs to $\mathcal{C}_W^{k-1}$. So, by induction hypothesis, every subgraph of a graph belonging to $\mathcal{C}_W^{k-1}$ belongs to $\mathcal{C}_W^{k-1}$. So, $G'[N^+(x_j)]$ belongs to $\mathcal{C}_W^{k-1}$.

So, $\forall j \neq i, 1 \leq j \leq n$ with respect to the ordering σ', the induced subgraph $G'[N^+(x_j)]$ belongs to $\mathcal{C}_W^{k-1}$, and then, the ordering σ' is a $\mathcal{C}_W^k$ scheme of G' and thus, $G' \in \mathcal{C}_W^k$.QED.

This second theorem is useful for testing membership of a $\mathcal{C}_W^k$ class, which is the question addressed in the next section.

Recognition of Graphs of a Given Class $\mathcal{C}_W^k$

The question of recognizing graphs of a given class $\mathcal{C}_W^k$ is now asked. We show first that this is feasible, and we will then see that the complexity is of course linked to the assumptions we can make about the parameter W. Indeed, determining the treewidth of a graph is well known to be an NP-hard problem (Arnborg, Corneil, and Proskuroswki 1987). So, to know whether a graph is in a $\mathcal{C}_W^0$ class, we first have to determine whether its treewidth is at most W. On the other hand, if W is a constant, we can rely on efficient algorithms, since the problem is then known to be tractable (i.e. its time complexity is polynomial (Bodlaender 1996)). In the sequel, let $T(G, W)$ denotes the time complexity to check if the treewidth of a graph G is bounded by W. The value of $T(G, W)$ can be different, depending on whether the value of W is bounded by a constant or not.

Consider an integer W and a graph G. Recognizing whether G belongs to the class $\mathcal{C}_W^0$ can be done directly by calculating the treewidth of G and checking whether it is less than or equal to W. Now consider an integer k. If $k = 1$, a necessary condition for G to belong to $\mathcal{C}_W^1$ is that there exists a first vertex x_1 of G such that its neighborhood has a treewidth less than or equal to W. If no vertex satisfies this condition, then G does not belong to $\mathcal{C}_W^1$. On the other hand, if such a vertex does exist, then it can be the first vertex in a $\mathcal{C}_W^1$ scheme σ. Furthermore, if G belongs to $\mathcal{C}_W^1$, by the heredity property presented in the previous section, the subgraph of $G[V \backslash \{x_1\}]$ belongs to $\mathcal{C}_W^1$ and there exists a vertex x_2 such that its neighborhood in $G[V \backslash \{x_1\}]$ induces a graph whose the treewidth is less than or equal to W. This process can be repeated until all the vertices in G have been eliminated. If all the vertices of G are indeed eliminated, then G belongs to $\mathcal{C}_W^1$. If all the vertices of G are not eliminated, then G does not belong to $\mathcal{C}_W^1$. This approach defines an algorithm whose complexity is bounded by $O(n^2 \times T(G, W))$ since the $\mathcal{C}_W^1$ scheme σ which is computed needs to order the n vertices of G, and for each one, we have to check for the treewidth of at most n sugraphs.

Such an approach can be generalized to other values of k to define the algorithm called $RecoG\text{-}CkW$ for the recognition of graphs of the class $\mathcal{C}_W^k$. If an input graph belongs to $\mathcal{C}_W^k$, this algorithm returns $True$ and an associated scheme σ, otherwise, it returns $False$. In this algorithm, to simplify notations, we consider $N_u(x)$ to denote the subset of unnumbered vertices appearing in the neighbourhood of a vertex x in G. Note that at line 6, the recursive call of $RecoG\text{-}CkW$ in the condition matches to the test $G[N_u(x)] \in \mathcal{C}_W^{k-1}$.

Theorem 3 *For $k \geq 0$, if $G \in \mathcal{C}_W^k$, the algorithm $RecoG\text{-}CkW$ returns True and a $\mathcal{C}_W^k$ scheme (if $k > 0$), otherwise, it returns False.*

Proof. We prove this result by induction on k. This property trivially holds for $k = 0$ since the algorithm is limited to testing the treewidth of G (line 2).

Now, consider a graph $G = (V, E)$ and $k > 0$, and assume that $RecoG\text{-}CkW$ is correct when calling with a parameter k' such that $k' < k$. If $G \in \mathcal{C}_W^k$, necessarily, G possesses a vertex x such that $G[N_u(x)]$ belongs to $\mathcal{C}_W^{k-1}$. This vertex will be the first one in a $\mathcal{C}_W^k$ scheme and by induction hypothesis, the recursive call RecoG-CkW($G[N_u(x)], k - 1, W$) returns True. Using the heredity of graphs belonging to $\mathcal{C}_W^k$, the same reasoning can be applied to the subgraph induced by $G[V \backslash \{x\}]$, And so on, until all the vertices of G are deleted. This will lead to the generation of an ordering σ

Algorithm 1: RecoG-CkW

Input: $G = (V, E)$: **graph**; $k \geq 0$, W : **integer**;
Output: σ : **ordering**; CkW : **boolean**;
// if $G \in \mathcal{C}_W^k$, returns True and σ is the scheme,
// else, returns False

```
1: if k = 0 then
2:    CkW ← ( tw(G) ≤ W );
3: else
4:    σ ← [ ]; CkW ← True; i ← 1;
5:    while CkW and i ≤ n do
6:       if ∃x ∈ V : RecoG-CkW(G[N_u(x)], k − 1, W)
         then
7:          σ[i] ← x;
8:          i ← i + 1;
9:          V ← V\{x};
10:      else
11:         CkW ← False;
12:      end if
13:   end while
14: end if
15: return CkW;
```

on V, and ends the loop with CkW being true, which will then be the value returned by $RecoG\text{-}CkW$.

On the other hand, if $G \notin \mathcal{C}_W^k$, no $\mathcal{C}_W^{k-1}$ scheme can be find. So, during the **while** loop, necessarily, after some deletions of vertices x, none of the vertices belonging to the resulting set V such that $G[N_u(x)] \in \mathcal{C}_W^{k-1}$ will be found, and then, the **while** loop stops after the assignment of CkW to False, which will be the value returned by $RecoG\text{-}CkW$.

For the termination of the algorithm, simply note that with each recursive call, the value of k decreases strictly, so if the algorithm does not stop before then, the case $k = 0$ will be reached and the execution will stop.

Thus, in all cases, this algorithm terminates and the returned result is correct. QED.

The following property indicates the time complexity of this algorithm.

Theorem 4 *The time complexity of the algorithm $RecoG\text{-}CkW$ is $O(n^{2k} \times T(G, W))$.*

Proof. The worst case time complexity occurs for graphs that belongs to $\mathcal{C}_W^k$. In this case, the ***while*** loop will be executed n times. In this case, the cost of the algorithm is given by the cost of the condition in line 6. The condition contained in the ***if*** conditional statement potentially imposes the execution of an additional loop to find a vertex verifying the condition, and this loop will run from the current value i to n, which is bounded by n. This leads to a multiplicative factor n which is cumulated to that of the ***while*** loop, giving a multiplicative factor equal to n^2. So, find a suitable vertex x induces a factor n^2 that must be considered with the cost of the recursive call to $RecoG\text{-}CkW$ associated to the condition $G[V\backslash x] \in \mathcal{C}_W^{k-1}$. So, the time complexity $C(k)$ is given by a recurrence relation based on k:

- For $k = 0$, the cost $C(k) = C(0)$ is $O(T(G, W))$ because the time complexity in line 2 is $O(T(G, W))$
- For $k \geq 1$, the cost $C(k)$ is $O(n^2 \times C(k-1))$

Thus, from this recurrence relation, we can easily prove that the time complexity of the algorithm $RecoG\text{-}CkW$ is $O(n^{2k} \times T(G, W))$. QED.

So, if W and k are defined as constant, the time complexity of the algorithm $RecoG\text{-}CkW$ is then polynomial since we know a linear time algorithms for checking if the treewidth of a graph is equal to a given integer W (Bodlaender 1996), and then, $T(G, W) \in O(|G|)$.

Corollary 1 *If W is bounded by a constant, the time complexity of $RecoG\text{-}CkW$ is in $O(n^{2k} \times |G|)$.*

Class $\mathcal{C}_W^k$ and Tractability of the Maximum Clique Problem

We now consider the problem of finding a maximum size clique in a graph. We will show that this problem, although known to be NP-hard, can be solved in polynomial time on graph classes of type $\mathcal{C}_W^k$, under the assumption that the base class $\mathcal{C}_W^0$ has a parameter W which is a constant while k is also a constant. To solve this problem, we define the algorithm $MaxCliq\text{-}CkW$. We show its correctness and, then, we evaluate its time complexity. We can then see that, under the assumption that the instance processed as input belongs to a given class $\mathcal{C}_W^k$ for which k and W are constants, then this algorithm has a polynomial time complexity.

At the first step of the algorithm (lines 1-2), if $k = 0$, a maximum size clique of G is found, exploiting a tree-decomposition of G whose width is at most W.

For the general case, i.e. $k > 0$, a $\mathcal{C}_W^k$ scheme σ of G is computed with a call to $RecoG\text{-}CkW$. Using this ordering, every vertex x is checked to find a maximum size clique in the associated subgraph $G[N_\sigma^+(x)]$, knowing that this subgraph belongs to $\mathcal{C}_W^{k-1}$. After the **for** loop, we are ensured that K contains the maximum size clique of the graph G.

Theorem 5 *For $k \geq 0$, if $G \in \mathcal{C}_W^k$, the algorithm $MaxCliq$ returns a maximum size clique of G.*

Proof. We prove this result by induction on k.

For $k = 0$, using a basic algorithm, $MaxCliq$ finds a maximum size clique in the graph G assuming that its treewidth is bounded by W.

Now, consider the graph G and $k > 0$, and assume that $MaxCliq$ is correct when it is called with a parameter k' such that $k' < k$. As $G \in \mathcal{C}_W^k$, and since σ is a $\mathcal{C}_W^k$ scheme, for all i, $1 \leq i \leq n$, each subgraph $G[N_\sigma^+(x_i)]$ belongs to $\mathcal{C}_W^{k-1}$ and therefore, applying the inductive hypothesis about the correction of MaxCliq(), each recursive call MaxCliq$(G[N_\sigma^+(x_i)], k-1, W)$ returns a maximum size clique of $G[N_\sigma^+(x_i)]$ which is assigned to K_i (line 8). Thus, $K_i \cup \{x_i\}$ is a maximum size clique of $G[N_\sigma^+(x_i) \cup \{x_i\}]$. Now, consider K', a maximum size clique of G, and

Algorithm 2: MaxCliq

Input: $G = (V, E)$: **graph**; $k \geq 0, W$: **integer**;
Output: K : **set of vertices**;
// K max size clique of $G \in \mathcal{C}_W^k$

```
1: if k = 0 then
2:    K ← Maximum Size Clique of G;
3: else
4:    Find a C_W^k scheme σ using RecoG-CkW;
5:    K ← ∅;
6:    for (i ← 1; i ≤ n; i ← i + 1) do
7:       x ← σ[i];
8:       K_i ← MaxCliq(G[N_σ^+(x)], k − 1, W);
9:       if |K_i ∪ {x}| > |K| then
10:         K ← K_i ∪ {x};
11:      end if
12:   end for
13: end if
14: return K;
```

let $x_j \in K'$, be the first vertex of K' with respect to the ordering σ. Necessarily, $K' \subseteq (N_\sigma^+(x_j) \cup \{x_j\})$. Therefore, after the call MaxCliq($G[N_\sigma^+(x_j)], k-1, W$), we have $|K' \cup \{x_j\})| = |K_j \cup \{x_j\})|$. Note that here, we consider sizes of cliques rather than cliques themselves because it can exists several (so different) maximum size cliques in a given (sub)graph. So, $MaxCliq$ is correct when it is called with the parameters G, k and W. Finally, for the termination of the algorithm, as for the algorithm $RecoG\text{-}CkW$, we can see that with each recursive call, the value of k decreases strictly, and so the case $k = 0$ will be reached and the execution terminated. QED.

We evaluate the complexity of this algorithm by assuming that the time complexity of computing a clique of maximum size in a graph of treewidth W is given by $TC(G, W)$.

Theorem 6 *For $k \geq 0$, if $G \in \mathcal{C}_W^k$, the time complexity of $MaxCliq$ is $O(k.n^{2k} \times T(G, W) + n^k \times TC(G, W))$.*

Proof. For $k = 0$, assume that the time complexity is bounded by $TC(G, W)$ which is the time needed to compute a clique of maximum size in a graph G whose treewidth is W (line 2).

Now, assume $k \geq 1$ and let $Cliq(k)$ be the time complexity of $MaxCliq$ for a value k. In line 4, the time complexity of the call to $RecoG\text{-}CkW$ is $O(n^{2k} \times T(G, W))$. Next, the instructions in lines 7 to 10 are executed n times. In these lines, it is sufficient to consider the running time of the line 8 which is $Cliq(k-1)$ since the recursive call of $MaxCliq$ considers $k-1$ as input. So, the time complexity $Cliq(k)$ is given by a recurrence relation based on k:

- For $k = 0$, $Cliq(k) = Cliq(0)$ is $O(TC(G, W))$
- For $k \geq 1$, $Cliq(k) = n^{2k} \times T(G, W) + n.Cliq(k-1)$

The solution for this relation is :

$$Cliq(k) = n^k \times (T(G, W) \times \Sigma_{i=1}^k n^i + TC(G, W))$$

which can easily be proved by induction. So, the time complexity of the algorithm $MaxCliq$ can be bounded by $O(k.n^{2k} \times T(G, W) + n^k \times TC(G, W))$. QED.

So, if W and k are defined as constant, the time complexity of the algorithm $MaxCliq$ is then polynomial since we know linear time algorithms for checking the treewidth and looking for its maximal size clique, ie. $T(G, W)$ and $TC(G, W)$ can be replaced by $|G|$:

Corollary 2 *If W is bounded by a constant, the time complexity of $MaxCliq$ is in $O(k.n^{2k} \times |G|)$.*

We can see that the parameter k plays a fundamental role here, as the practical tractability of the clique problem will be bound in this approach by the value of k. Moreover, it is also well known that difficult instances of the clique problem, but not only of course, have treewidths not bounded by constants. Nevertheless, we show in the next section that such instances can belong to classes of type $\mathcal{C}_W^k$ for which the value of k is small.

Classes $\mathcal{C}_W^k$ and Graphs of Unbounded Treewidth

It is well known that numerous hard problems, such as the Clique problem for example, are simple to solve as long as their treewidth is bounded by a constant, while they remain hard for the class of graphs of unbounded treewidth. But we show here that classes of graphs of unbounded treewidth can exist, but for which the parameter k can be equal to 1, and the treewidth W can be equal to a small value constant. This is the case of planar graphs for which we know that their treewidth belongs to $\Theta(\sqrt{n})$:

Proposition 1 *Planar graphs belong to class $\mathcal{C}_3^1$.*

Proof. To show that planar graphs belong to the class $\mathcal{C}_3^1$, consider a planar graph $G = (V, E)$. We show that for such graphs, there exists an ordering $\sigma = [x_1, x_2, \ldots, x_n]$ of V such that for $i = 1, 2, \ldots, n$, the subgraph $G[N_\sigma^+(x_i)]$ belongs to $\mathcal{C}_3^0$, that is the class of graphs with a treewidth of 3 or less.

It is well known that for any planar graph, there exists a vertex whose degree is at most 5 (a folklore property derived from Euler's formula). So, consider such a vertex x_1 in G. So, $|N(x_1)| \leq 5$, and since G is planar, $G[N(x_1)] \neq K_5$ and its treewidth is at most 3 because $G[N(x_1)]$ can contain at most 5 vertices and the edges forming 2 cliques of size 4, that is the two cliques K_4. Now consider the subgraph of G defined by $G[V \backslash \{x_1\})]$. Since G is planar, $G[V \backslash \{x_1\})]$ is planar too. So G has a vertex x_2 ,whose degree is at most 5, and the same reasoning as above can be used to show that the subgraph $G[N(x_2) \backslash \{x_1\}]$ belongs to $\mathcal{C}_3^0$. More generally, we can define an ordering $\sigma = [x_1, x_2, \ldots, x_n]$ of V which is a $\mathcal{C}_3^1$ scheme, allowing to prove that G belongs to $\mathcal{C}_3^1$. QED.

If this result seems significant from a theoretical point of view, it is less relevant when we refer to the problem of the clique of maximum size. Indeed, it is well known that a planar graph cannot possess a clique of size greater than or equal to 5, and thus we then have a trivial polynomial algorithm enumerating all subsets of 4 vertices or less to

find a clique of maximum size. So, it is useful to show that there exists other classes of graphs, non-planar for example, for which we have similar properties in terms of parameter k while keeping W equal to a constant. This is the case for the complete bipartite graphs $K_{n,n}$ whose treewidth is n. Indeed, for $K_{n,n}$, it is easy to see that there is a tree-decomposition defined by n bags, each one being associated to one vertex of one on the two sets, and including also the n vertices of the other set. Its treewidth is then n. Moreover, this width is minimal because any optimal triangulation order minimizing the size of the largest induced clique (and therefore the width) will have to start with any vertex from $K_{n,n}$ and since these vertices have the same degree, i.e. n, the first vertex will be include in a clique of size $n+1$, and therefore the width will be at least equal to n. Despite their unbounded treewidth, these graphs belong to the $\mathcal{C}_0^1$ class because the neighborhood of any vertex is a set vertices between which there are no edge. Consequently, the treewidth of this neighborhood is always 0. And this applies regardless of the order of the vertices. So every complete bipartite graph on n vertices belongs to $\mathcal{C}_0^1$ while its treewidth is $\frac{n}{2}$.

Discussion and Conclusion

As we have seen, the definition of $\mathcal{C}_W^k$ classes makes it possible to design polynomial time algorithms for the Clique problem. This is all the more interesting in that, unlike the use of treewidth, for instances where treewidth is not bounded by a constant, the use of this double parameterization allows some of these instances to be processed in polynomial time. So, a natural question arises. "*Can this definition of a new class of graphs be extended to handle other difficult problems* ?" We know that for the case of bounded treewidth, many problems can become tractable because we can obtain efficient approaches for hard problems thanks to FPT algorithms.

Unfortunately, it is not easy to do. Indeed, if we take fairly related graph problems, such as *Independent Set*[1] or *Vertex Cover*[2], the approach used here will not work directly.

In fact, the $\mathcal{C}_W^k$ class seems suited to problems such as Clique due to the definition of the class. Indeed, by constructing a solution as seen in the $MaxCliq\text{-}CkW$ algorithm, every solution to the problem has a first vertex x_i such that a solution is a set of vertices that belongs to N_σ^+, that is in the neighborhood of this first vertex. And consequently, a global solution is defined in adding x_i to a maximum size clique of $G[N_\sigma^+]$, this subgraph being tractable since it satisfies the right properties. Thus, the existence of a solution is intrinsically linked to the construction of the graph class, which is based on the vertex neighborhood.

On the other hand, if we consider the Independent Set problem, every solution is made up of vertices that are precisely absent from the neighborhood of a first vertex x_i. So, treatment using a similar approach to that based on $\mathcal{C}_W^k$ classes would consist in defining this type of class differently, by considering not the neighborhood of vertices, but the vertices that are not in the neighborhood. But what is possible for a problem like Independent Set is not necessarily so for other problems. For example, if we take Vertex Cover, then we need to be sure not only of the nature of the considered σ ordering, but also of the way in which solutions can be constructed. For a given vertex x_i, which vertices should be related to x_i to ensure that a partial solution in this set of vertices can be extended by adding x_i? While in the case of the Clique problem, this is obvious when considering the neighborhood of x_i, in the case of Vertex Cover, the question remains completely open at this stage.

One possible approach is based on the notion of polynomial transformation, transforming any problem to the Clique problem. However, such an approach would only be possible under the assumption that the structural properties related to the treewidth of the graph are preserved. In another context, this possibility was explored using the notion of "guarded reductions", which are reductions defined by first-order logic formulas showing that guarded reductions preserve bounded treewidth (Mitchell 2019).

[1]Given a graph G and an integer k, does G have a subgraph of k or more vertices that has no edges

[2]Given a graph G and an integer k, does G have a subset of k or fewer vertices such that every edge has at least one vertex in that subset

Acknowledgements

The author would like to thank Cyril Terrioux for his critical reading of this paper. This work has been funded by the French Agence Nationale de la Recherche, reference Massal'IA ANR-19-CHIA-0013-01.

References

Arnborg, S. 1985. Efficient Algorithms for Combinatorial Problems with Bounded Decomposability - A Survey. *BIT*, 25(1): 1–23.

Arnborg, S.; Corneil, D.; and Proskuroswki, A. 1987. Complexity of finding embeddings in a k-tree. *SIAM Journal of Discrete Mathematics*, 8: 277–284.

Bodlaender, H. L. 1996. A Linear-Time Algorithm for Finding Tree-Decompositions of Small Treewidth. *SIAM J. Comput.*, 25(6): 1305–1317.

Courcelle, B. 1990. The monadic second-order logic of graphs. I. Recognizable sets of finite graph. *Information and Computation*, 85 (1): 12–75.

Downey, R. G.; and Fellows, M. R. 1999. *Parameterized Complexity*. Monographs in Computer Science. Springer.

Jégou, P. 1993. Decomposition of Domains Based on the Micro-Structure of Finite Constraint Satisfaction Problems. In *AAAI*, 731–736.

Johnson, D.; and Trick, M. 1996. *Cliques, Coloring, and Satisfiability: Second DIMACS Implementation Challenge*. DIMACS Series in Discrete Mathematics and Theoretical Computer Science. vol 26. AMS.

Mitchell, D. 2019. Guarded Constraint Models Define Treewidth Preserving Reductions. In *Principles and Practice of Constraint Programming - Stamford - USA*, volume 11802 of *LNCS*, 350–365.

Robertson, N.; and Seymour, P. 1986. Graph minors II: Algorithmic aspects of treewidth. *Algorithms*, 7: 309–322.