1

Hello, my project for this year is named "graphs' decompositions and resolutions of combinatorial problems" and this one is supervised by Florent Madeleine

2

Here is the scheeme of my presentation. First, I will briefly expose the objectives of my project
Secondly I will introduce you two concepts : treewidth and nice tree.
Finally, We will see an example of a problem's resolution which is using a nice tree and the treewidth.

3

4

This project deals with a type of tree which is representing graph.
A graph is more complex than a tree but sometimes it is possible to reduce it with a tree representation to solve problems because a tree is often easier to manipulate.
That is the reason why my first task will be to study some classicals' problems where there is a graph by using a tree representation of it.
After that, I will have to implement a program to solve some of these problems.
Once this first step is done, I will have to implement a program which is able to give a tree representation of a graph.
Now, as I have said previously, I am going to expose some concepts and show how we can solve a problem with them.

5

6

A graph is not always as simple as a tree but it's possible to resume it in a particular tree while keeping some graph's informations into the tree.
To do that, the nodes of the tree represent a collection of nodes of the graph such that :
LIRE 1 :
You can see that each edge of G is present at least in one of the node (bag) of T.
LIRE 2 :
this means that all the bags where is present a node (for example the node 7) forms a tree (here a very simple one because it has only one branch).
It's important to realise two facts :
* lots of very different trees can represent a same graph * all graph has at least one tree representation. As a matter of fact, you can see that a tree as the tree T3 can always be made, even with a very complex graph.

7

The more a graph is strange (for example with a lot of loops), the more its tree representation has to contain some big node.
So it seams important to determine how far is a graph from a tree : this is the role of the treewidth
LIRE LE PREMIER POINT :
Remember that a tree representation of a graph is not unique and so there is no reason that the width of one of tree representation is the same of another one.
LIRE LE DEUXIÈME POINT
We can understand that we want to consider the tree with the smallest nodes.
In the example, each bag of T has a width of 3 so the treewidth is 2.
In comparaison, the graph of the previous slide has a treewidth of 3 and it seams obvious (at least to me) that its structure is more far from a tree than the present graph.

8

Now that we can evaluate the treewidth of a graph, I will expose a particular tree representation which is useful to implement algorithms : the well-named nice trees
In a nice tree, we can only find this four situations :
Leaf : no children
Introduce and Forget : in a bottom-up approach, there is one lettre more or one lettre less.
Join : in a bottom-up approach, two same nodes give a unique node with the same content.
The remark will be more explained in the two next slides.
It serves two purposes :
* can we always obtain a nice tree ? The response is yes and we will show you in the next slide * why is it useful ? It's to decompose complex mechanisms and thereby to answer to some problems

9

Here is a tree representation of a graph and we want to obtain a nice tree of this graph.
I want to draw your attention that I will constuct my nice tree from the root to the leaf and in the definition of a nice tree we are beginning from the leaf to the root.
So the "FORGET" method, the "INTRODUCE" method and the "JOIN" method will seem reverse action.
We are starting from the root and it's the same node for the two trees.
In the tree on the left, each son of the root looses a letter and wins another one.
This can be done with a join (to have two branches), a introduce (to loose a letter) and a forget (to win a letter).
So we obtain the good nodes in the nice tree.
We can make some similar steps and we obtain all the nodes of the starting tree.
It is not over because in a nice tree the leafs have to contain a single letter. But it is easy to obtain with some introduces.
Finally we obtain a nice tree of the previous graph.
We will use it in the next situation.

10

11

The k-color problem is a well known problem where we want to know if we can color a graph with k colors so that two neighbors never have the same color.
We have seen two concepts before and each of them is usuful here :
LIRE
In other words, treewidth tells us if it is possible and nice tree gives us a way to do it.
It is a good moment to study an example : we want to color this graph we have seen before and for which we have its treewidth and a nice tree.
I draw your attention that it is easy for a human to see how many colors he needs to resolve this particular problem and how to color it.
But this is not so easy to program a computer to do it especialy if the graph is bigger and the way to do it can be very long : this problem is NP-complet.
And our goal is to implement a program which is able to find the solution in a short time.

12

the biggest bag contains 3 letters so the treewidth is 2 so we need 3 colors to solve the problem.
We can start from the "a" leaf : for example it could be colored with red.
when we make an introduce we add a color but when we make a forget we can reuse the color of the letter which is gone because we know that this letter never come back latter : this is one of the property of a tree representation.
So this alternation of forget and introduce insures that we can make the coloration with 3 colors.
FAIRE DEFILER
Finally, the nice tree is colored with 3 colors and we can color the graph with this combination.

13

14

Thank you for your attention