REOP - Session 1

G. Dalle
22/09/21

I/ Introduction

Guillaume DALLE guillaume.dalle@enpc.fr

Resources : - Educnet page
- Teams workspace (videos)
- My website gdalle.github.io/reop/

II/ Problems & algorithms

General framework of OR

Running example : - Train Platforming Problem
- Inventory Routing Problem

TPP : choosing the platform for each train arriving at a railway station

IRP : managing depot & clients inventory (wrt supply/dem) + shipping / transportation routes

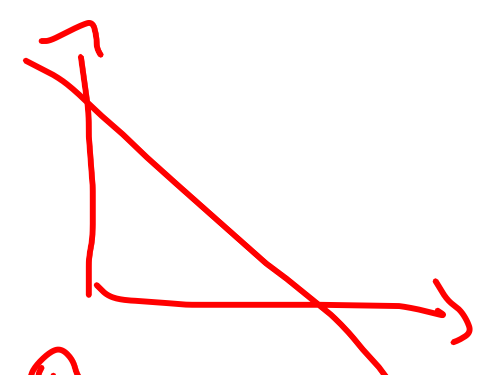
A) Problems

Problem = family of possible inputs + question to answer about an input

* Decision problem : the answer is "yes" or "no"
* Optimization problems : find the best candidate

Quiz : $4 \Rightarrow 1$ feasible sols $\Leftrightarrow \inf_{x \in X} c(x) \neq \inf \emptyset$

$1 \not\Rightarrow 2$ because



$3 \not\Rightarrow 2$ because

$5 \not\Rightarrow 3$: see next slide

Optimization pb : $\min c(x)$ s.t. $x \in X$ (P)

- "st" means "subject to"
- x : decision variable
- c : cost / objective function
- X : set of feasible solutions
- $x \in X$: constraints

Solving the problem :

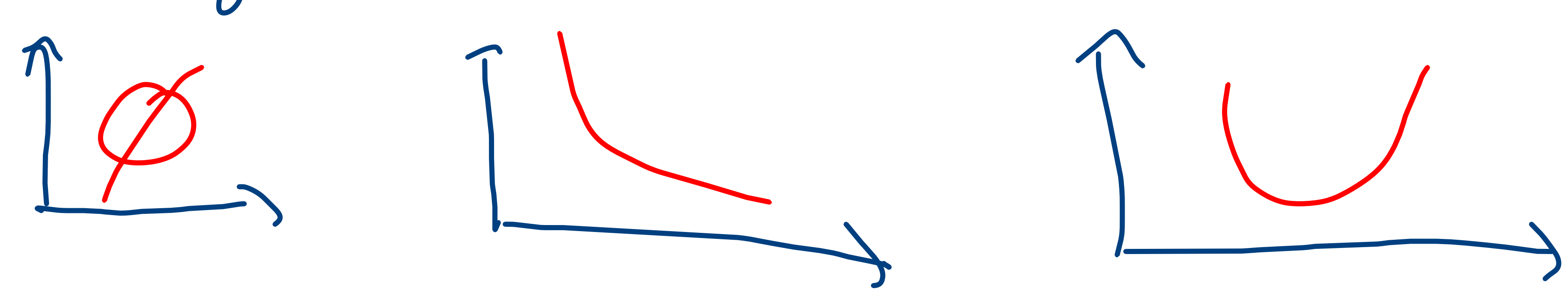
- finding its value $val(P) = \inf\{c(x),\ x \in X\}$
- finding an optimal solution if it exists

$$x^* \in \operatorname{argmin}\{c(x) : x \in X\}$$

$val\ P = +\infty \iff X = \emptyset$

$val\ P = -\infty \iff$ we can go as low as we want

$val\ P > -\infty$ can mean



B) Algorithms

Algorithm = sequence of elementary operations that can be executed by a "computer" (Turing machine)

Time complexity of $\mathcal{A}$: nb of elementary operations necessary for an input of (binary) size n
function $f(n)$

→ polynomial function :) in theory

→ exponential :(

In practice : it depends (LP)

Several types of optimization algorithms

- exact algorithms : compute an optimal solution
- approximation algs : compute a solution with bounded (near optimal) sub-optimality
- heuristics : compute a solution with no guarantee

↳ very useful in the project

How do you assess solution quality with a heuristic?
Find (by other means) a lower bound l

$$l \leq \mathrm{val}(P) \leq c(x^{heur})$$

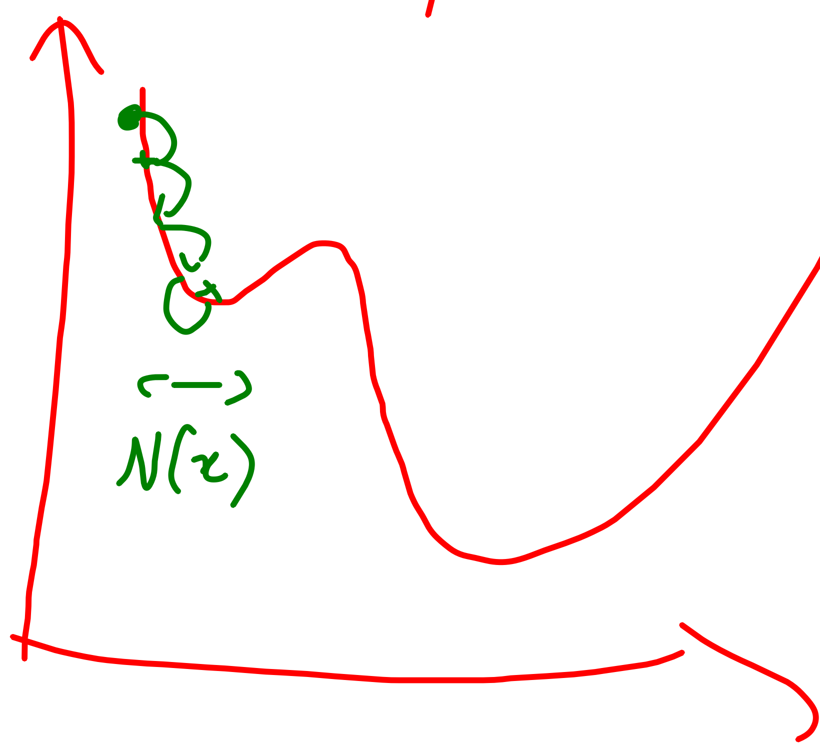
how far

Typical example of heuristic: local search / descent
Iterative algorithm: given a current sol $x_k \in X$,

1) Compute & explore its neighborhood $\mathcal{N}(x_k)$
2) Pick a next solution $x_{k+1} \in X$ such that $c(x_{k+1}) < c(x_k)$

Stop when you don't improve anymore

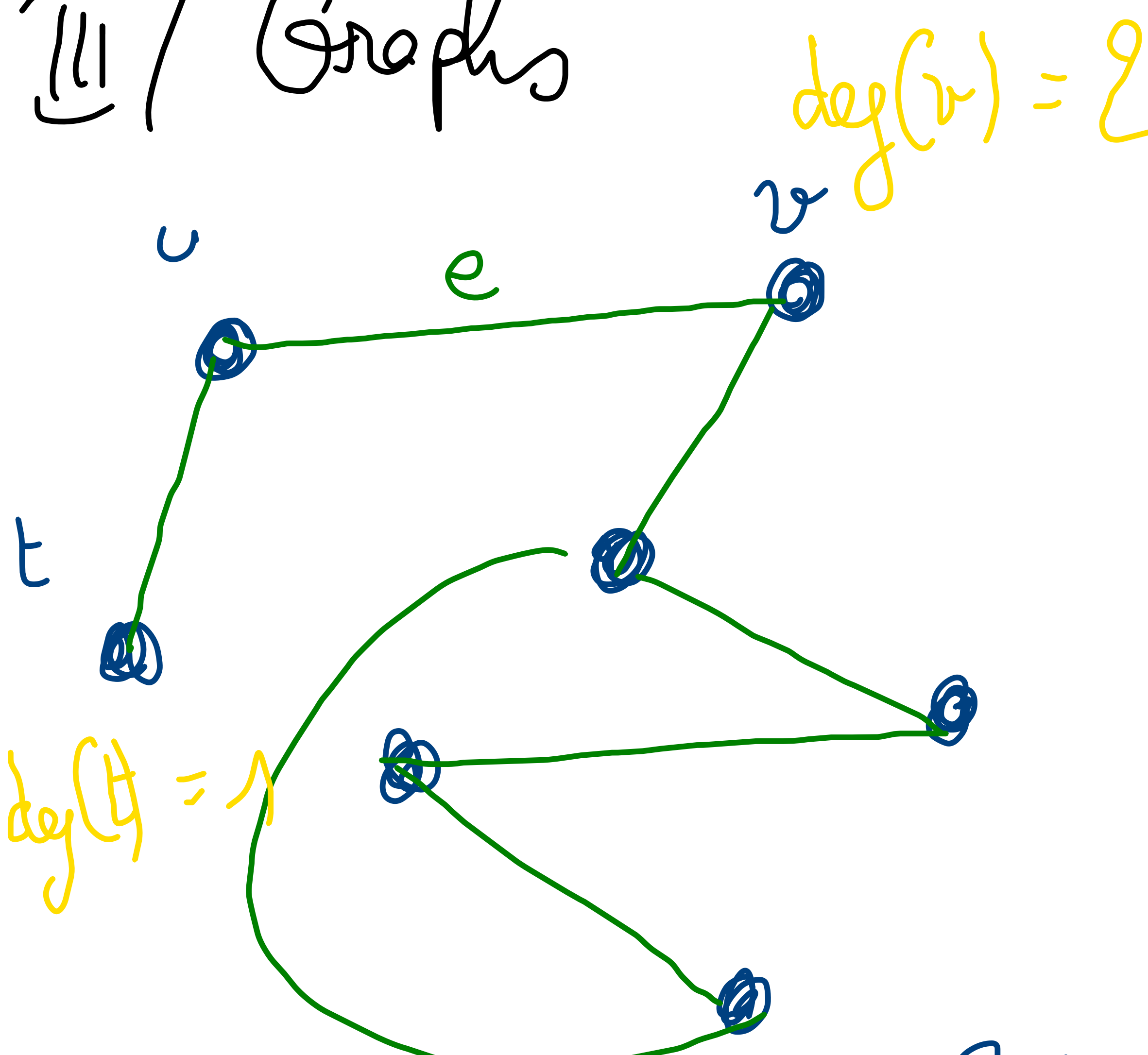
Quiz: LD stops at a local minimum



Compl iterations ? $\nearrow$ size of N

Nb of iterations ?

III/ Graphs



A) Vocabulary

A graph $G = (V, E)$ is a couple

- V = set of vertices — u or v
- E = set of edges — $e = (u, v)$ ←

Undirected $u \bullet\!\!-\!\!-\!\!\bullet v$ — $e = \{u, v\}$

Directed $u \bullet\!\!\longrightarrow\!\!\bullet v$ — $e = u \rightarrow v$

Selected definitions → see the rest in the notes

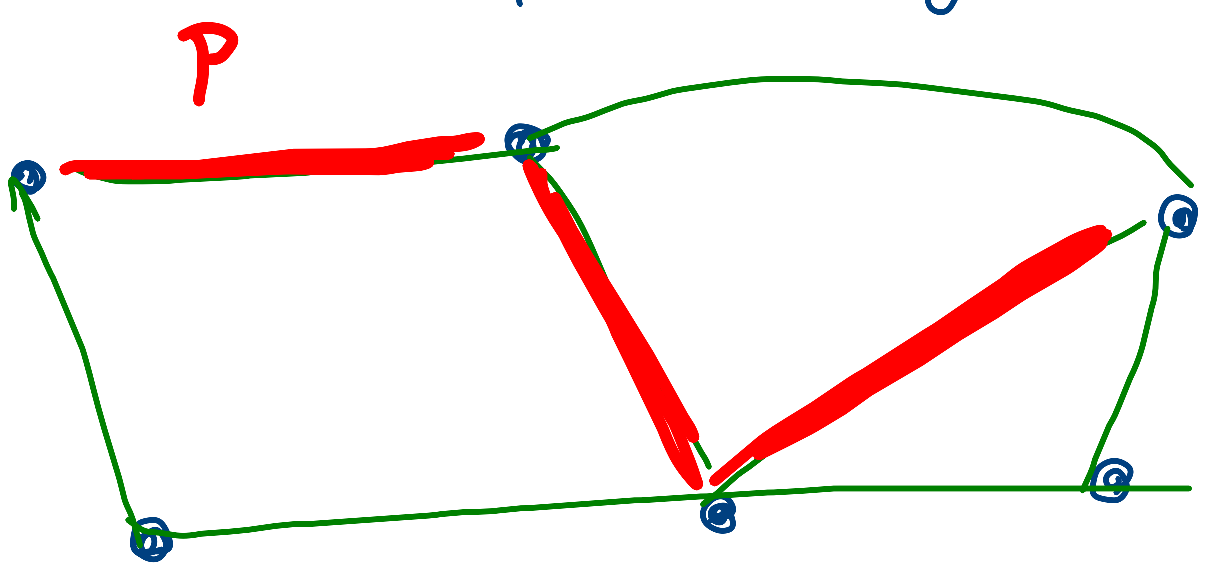
Subgraph: $H = (V', E')$

$V' \subset V$, $E' \subset E[V']$ → the edges within V'

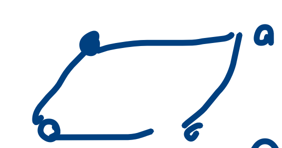
Degree of v: nb of incident edges

B) Paths

A path is a sequence of nodes linked of edges



- simple : no edge is crossed twice
- elementary : no vertex is visited twice
- cycle : start vertex = end vertex

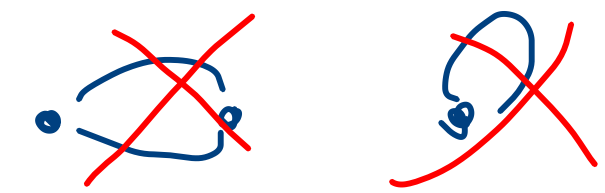


- eulerian : crosses all edges once * (exactly)
- hamiltonian : visits all vertices once (")

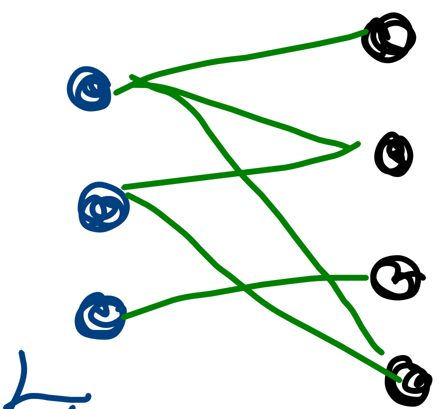
* Königsberg

c) Types of graphs

- simple graph : no duplicate edges & no self-loops

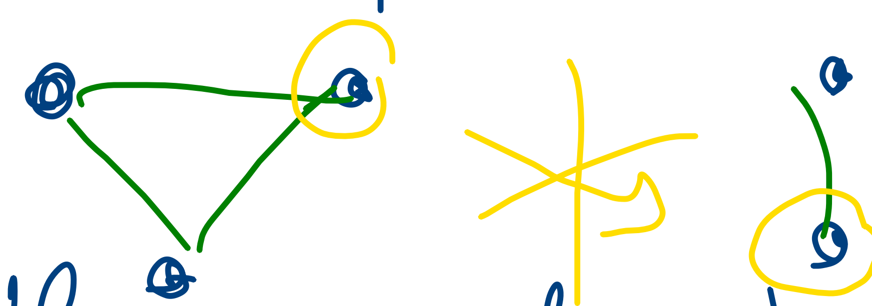


- complete graph : simple graph with all possible edges
- bipartite graphs : two sets of vertices with all edges in the middle



- eulerian / hamiltonian : contain a eul. / ham. path
- connected : there is a path between all pairs of vertices

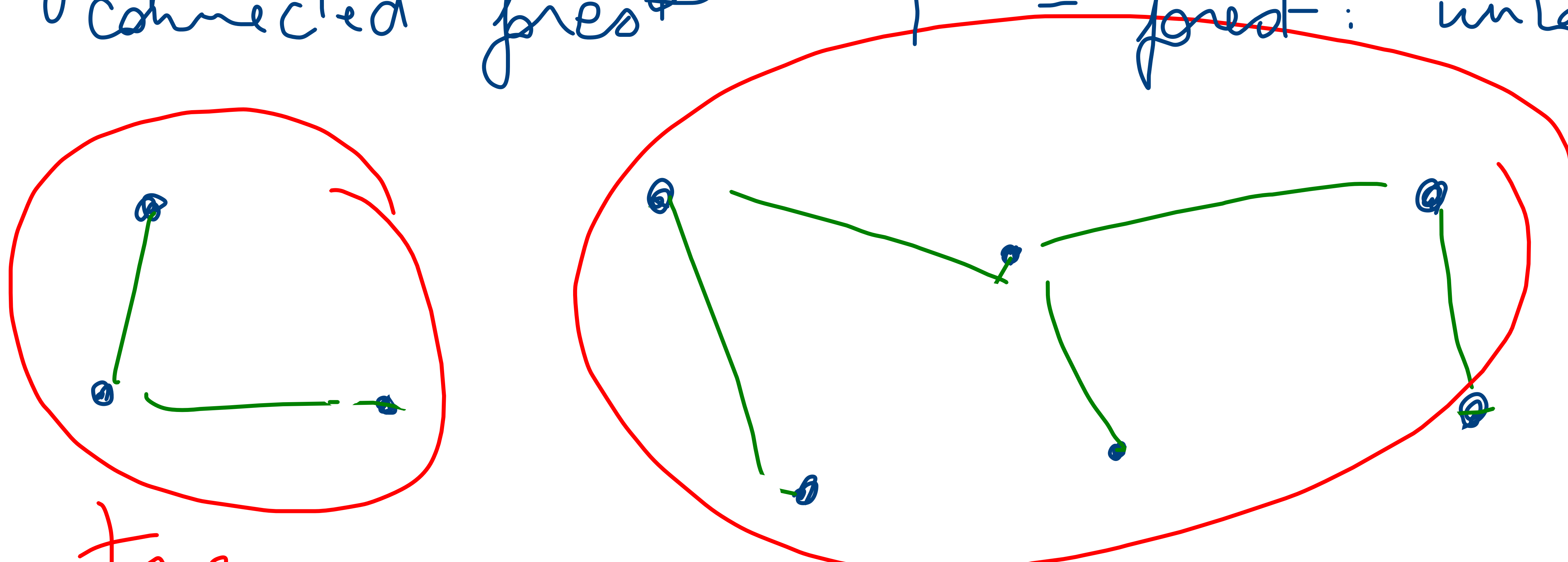
! for directed graphs



two connected component

- forest : graph with no cycle
- tree : connected forest

- tree : connected with no cycle
- forest : union of trees



tree

tree

forest

Quiz:

Elementary is simpler than simple

Complete undirected graph with n vertices has $m = \binom{n}{2} = \frac{n(n-1)}{2}$

Tree has $n-1$ edges $\longrightarrow$ exercise!

Quiz: $\sum_{v \in V} \deg(v) = 2|E|$

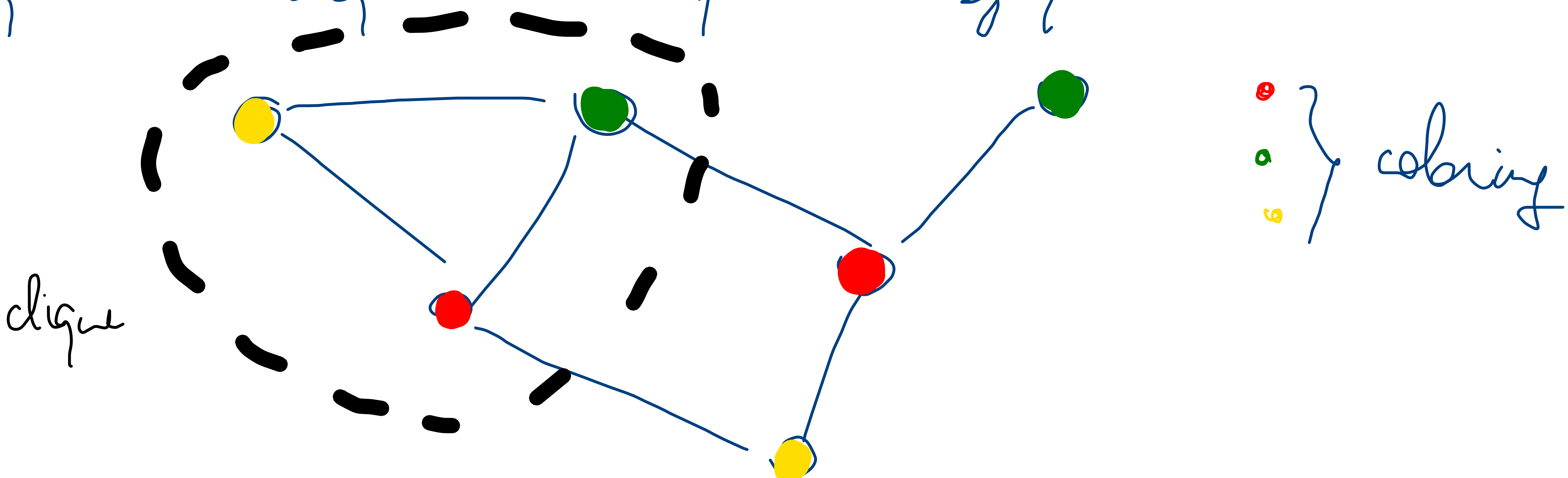
every edge is counted twice
- for its start vertex
- for its end vertex

$e = (u, v)$ adds 1 to degree of u, 1 to degree of v

D) Zoo of graph problems

Colorings: a coloring is a function $c: V \longrightarrow \mathbb{N}$
(one color/integer per vertex)
such that if $e = (u, v) \in E$, then $c(u) \neq c(v)$
(no two adjacent vertices with same color)

Cliques: a clique is a complete subgraph



$\chi(G)$ = smallest number of colors needed to color G
(chromatic nb)

$\omega(G)$ = size of the largest clique (clique nb)

Quiz : If G has a clique of size k,
then you need at least k colors because
each member of the clique needs a $\neq$ color

So $\chi(G) \geq \omega(G)$ but not always equal!

Look up the other animals in the zoo
- matching
- edge & vertex cover
- stable set

IV/ MILP $\longrightarrow$ next time

V/ Homework

- Fill the Wooclap form if you haven't
- Exercises : 3.2, 3.3 & 3.10 in the poly
  + if you want : finish the quiz
- Give feedback