CS2040 Tutorial 9

Week 11, starting 24 Oct 2022

Q1 Graph Modelling and Searching

You have a graph **G** of **N** persons and **M** *friendships* between two people:

- If person **A** is a *friend* of person **B**, then **B** is also a *friend* of **A**
- If **A** is a *friend* of **B** and **B** is a *friend* of **C**, it does **NOT** mean that **A** and **C** must be *friends*
- If **X** and **Y** are *friends*, or **X** is *friend* of one *in a relationship* with **Y**, then **X** and **Y** have a *relationship*
- It is possible that for 2 people **X** and **Y**, there is no (direct/indirect) *relationship* between **X** and **Y**

Find the:

- best data structure / representation for **G** to be in
- the algorithm to solve each problem

and

- its time complexity

to solve each of these parts:

(a) Ali wants to find out if two distinct given people are *friends*
This query may be run **repeatedly Q** times

(b) Bob wants to find out, just once, if two distinct given people are in a *relationship*

(c) Bob wants to find out if 2 distinct given people are in a *relationship*
This query may be run **repeatedly Q** times

Q2 Cycle Detection

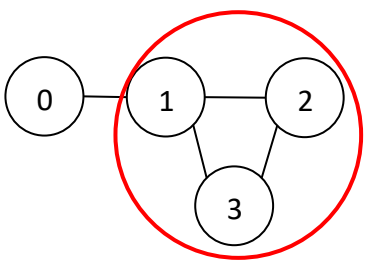
An *undirected* graph **G** with **V** vertices and **E** edges is a tree if any one of the following properties hold:

1. **G** is connected and **E** = **V** - 1
2. **G** has no cycles (is acyclic) and **E** = **V** - 1
3. There exists a unique path between every pair of vertices in **G**

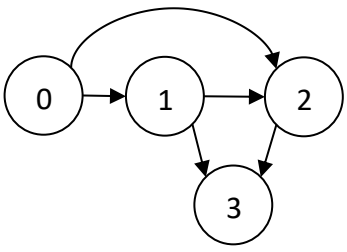
You are quite interested in property #2 and want to explore further. **Implement efficient algorithms** to find if a non-empty graph **G**, represented using an **adjacency matrix**, contains a cycle when **G** is:

(a) undirected (remember that **G** may not be connected)
(b) directed (remember that **G** may not be connected - even if it is, it may not be strongly connected)

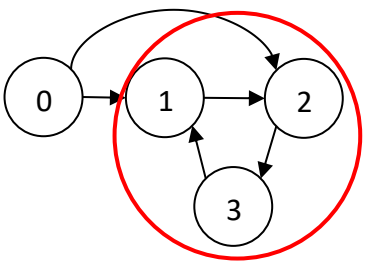
Determine the time complexity of your algorithm that works on the adjacency matrix, as well as what the time complexity would be if an adjacency list was used instead.



cycle in **undirected** graph



NO cycle in **directed** graph



cycle in **directed** graph

Question 3 (Online Discussion) – We're Under Attack!

In a computer game, there are **P** players. Each player is identified by an *id* in 0..(**P**-1), and has a *race* (in the game), either Orc or Human. Orcs are **supposed** to *attack* Humans only, and Humans are **supposed** to *attack* Orcs only. However, **some joker** will sometimes choose to *attack* someone of his own *race*.

Given the positive integer **P**, as well as a list of **A** distinct *attacks*, each of the form (attacking player id, victim player id), determine and output if:

- We're Under Attack! – some player is DEFINITELY *attacking* another of the same *race*
- Are we Under Attack? – it is possible that some player is *attacking* another of the same *race*
- Attack? What Attack? – it is DEFINITELY true that no player is *attacking* another of the same *race*

Unfortunately, you have NO information about the *race* of any of the **P** players =(