CPA. Introduction to Real-World Graphs

Maximilien Danisch

LIP6 – CNRS and Université Pierre et Marie Curie

first_name.last_name@lip6.fr

Equipe pédagogique :

- Maximilien Danisch (séances 1-5)
- Philippe Aubry, Maryse Pelletier (séances 6-8)
- Binh-Minh Bui-Xuan (séances 9-10)

Contrôles :

- Session 1: projet1 (30) + projet2 (20) + examen écrit (50)
- Session 2 (rattrapage) : uniquement examen écrit fin Juin

Outline

Outline

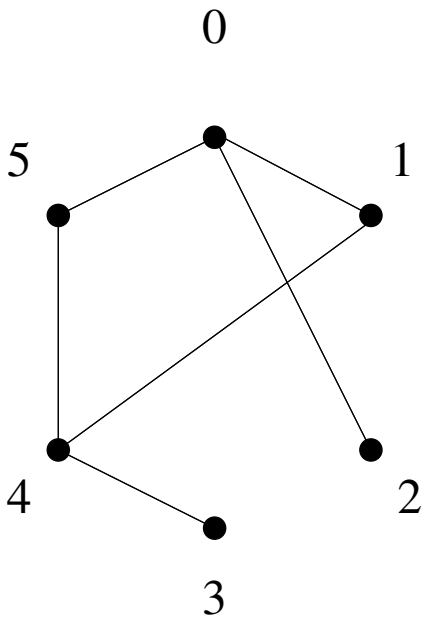
Graph

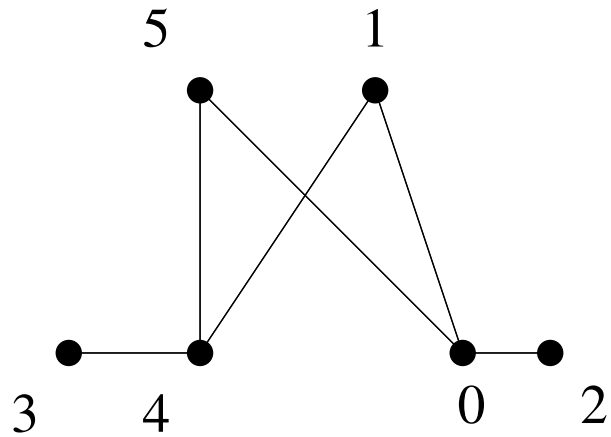
A graph $G = (V, E)$ is a couple of sets.

- V is a set of *vertices* (or *nodes*) ***fr:*** *sommet ou nœud*
- $E \subseteq (V \times V)$ is a set of *edges* (or *links*) ***fr:*** *lien, arête*

Example

- $V = \{0, 1, 2, 3, 4, 5\}$
- $E = \{(0, 1), (0, 2), (3, 4), (4, 5), (5, 0), (1, 4)\}$





Graph theory

Graph theory is a very important and well studied field:
`https://en.wikipedia.org/wiki/Graph_theory`

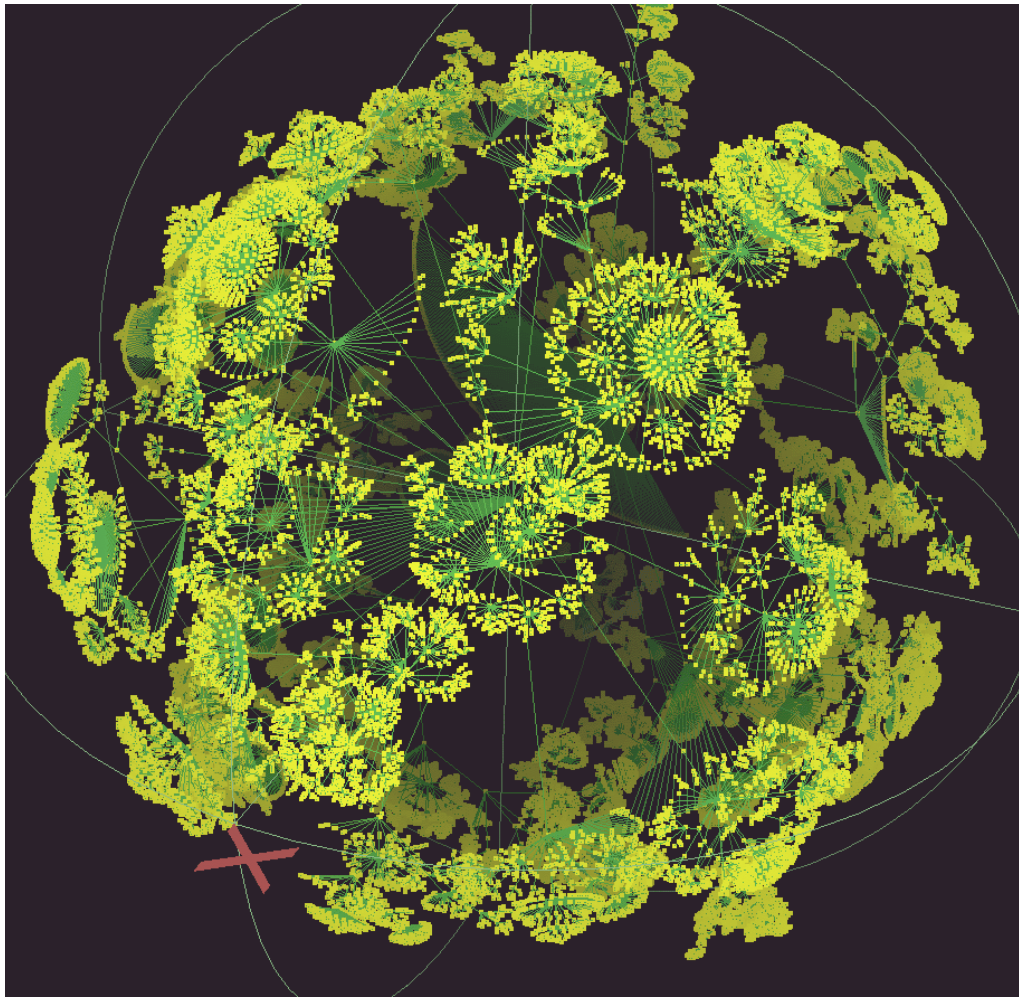
Question: Why is this useful to us?

Graph theory

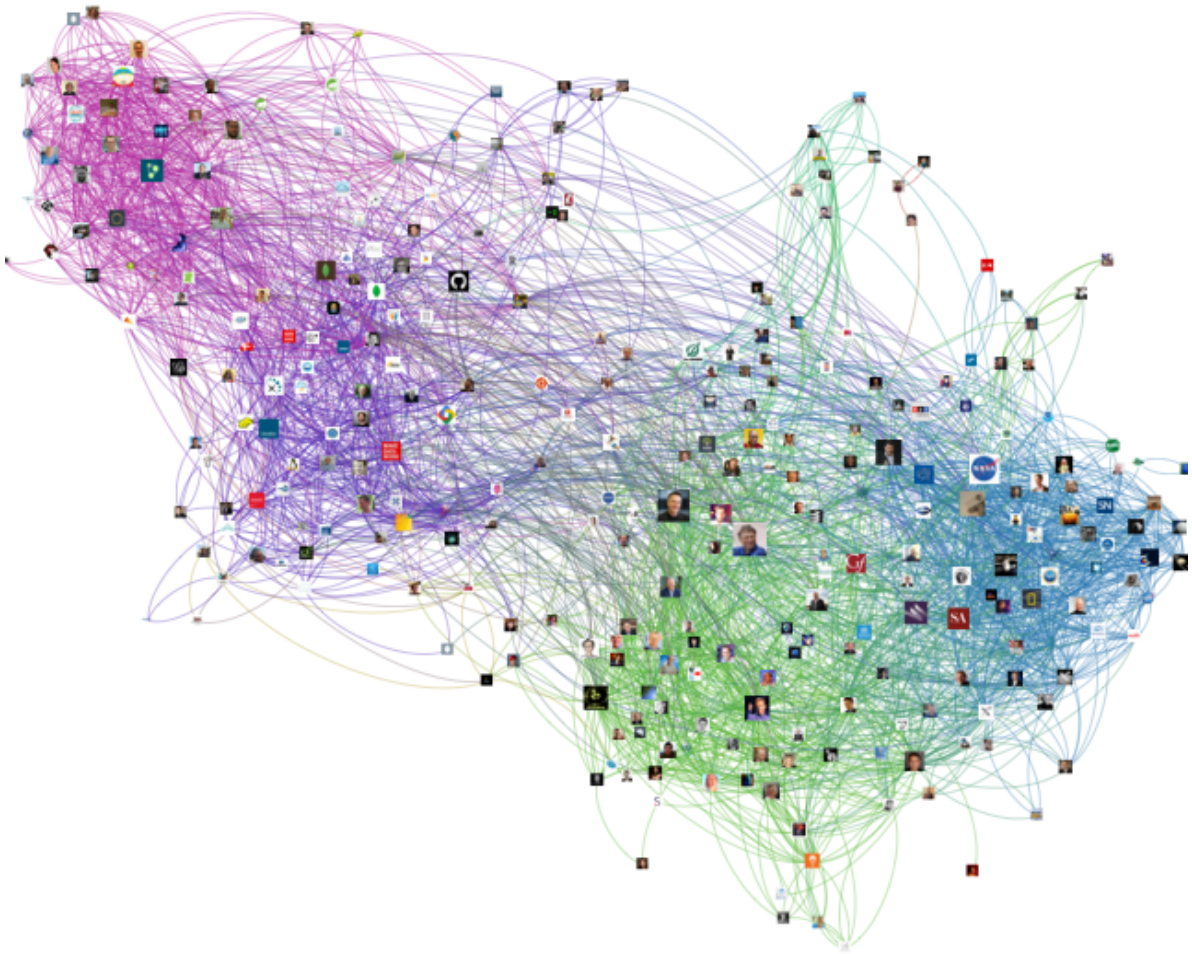
Graph theory is a very important and well studied field:
`https://en.wikipedia.org/wiki/Graph_theory`

Question: Why is this useful to us?
Many systems can be modeled using graphs!

Internet: computers connected by internet connections



Twitter: profiles connected by follow-links



Many more examples

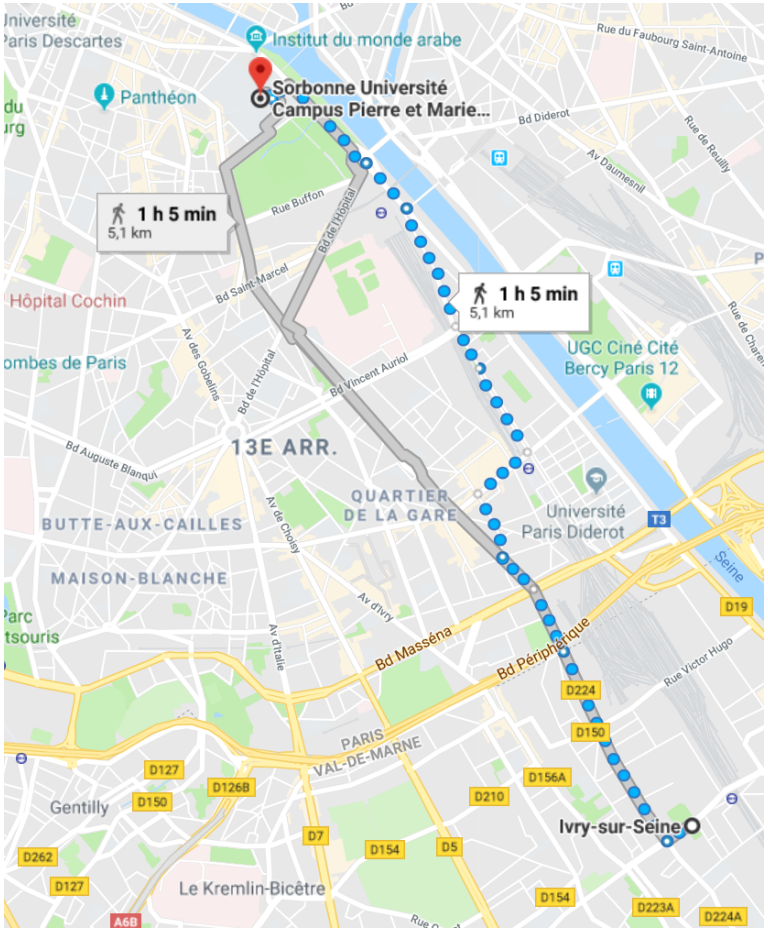
Graphs extracted from the “real-world” are called networks, complex networks or real-world graphs.

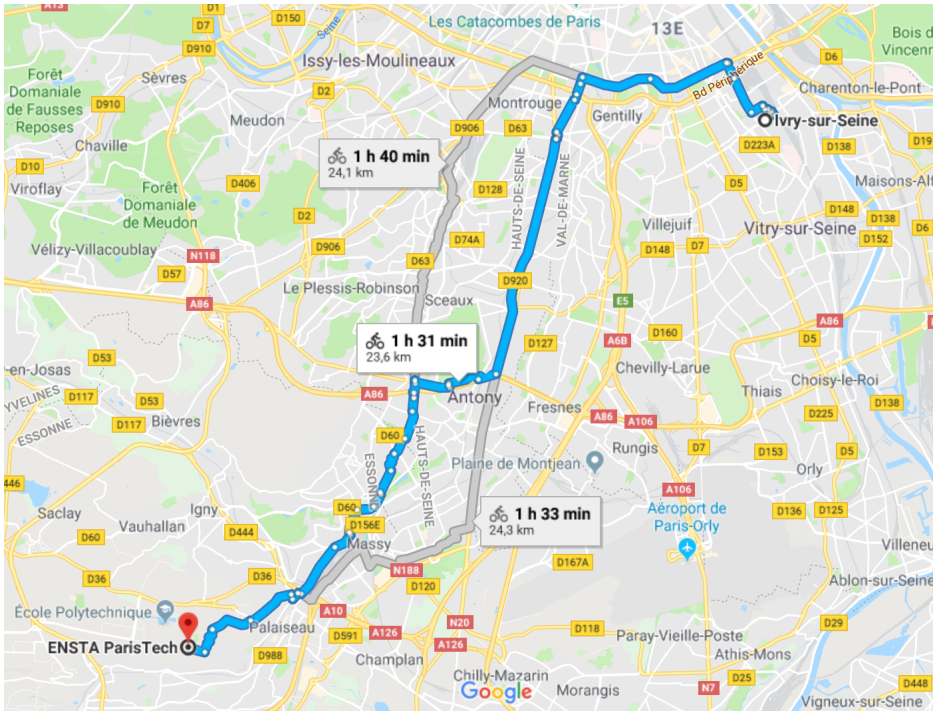
Question: Give some more examples of real-world graphs?

Common properties of real-world graphs

- **Very large:**
  Several G edges
- **Sparse:**
  Not many edges compared to what could be
- **Short distances and diameter:**
  $\sim \log(n)$; Six degrees of separation; Kevin Bacon game
- **Heterogeneous degree distribution:**
  $\sim$ power law; contains hubs (very high degree nodes)
- **Dense locally (many triangles):**
  High clustering coefficient & transitivity ratio
- ...

A real-world problem: GPS





Many more examples

Question: Give some more real-world problems related to graphs?

Outline

Time and memory complexity

Let G be a graph of n nodes and m edges.

Question: What do you think about the following complexities?
$\mathcal{O}(m)$ and $\mathcal{O}(n^2)$

Storing a graph in main memory

Three main formats:

1. List of edges
2. Adjacency matrix
3. Adjacency array

Question: Implementation? Pros and cons? Complexities?

`https://github.com/maxdan94/LoadGraph`

Adjacency Array in C

```
1 //compact adjacency array datastructure
2 typedef struct {
3   unsigned n; //number of nodes
4   unsigned m; //number of edges
5   unsigned *cd; //cumulative degree cd[0]=0 length=n+1
6   unsigned *adj; //concatenated lists of neighbors
7 } adjarray;
```

Question: How to list the neighbors of node u?

Outline

Breadth First Search algorithm (BFS)

Algorithm 1 BFS from node s

```
function BFS(G,s)
    FIFO ← CreateFIFO()
    FIFO.add(s)
    Mark(s)
    while FIFO not empty do
        u ← FIFO.pop()
        output u
        for each v neighbor of u in G do
            if v is not marked then
                FIFO.add(v)
                Mark(v)
```

Question: Implementation? Complexity?

Breadth First Search algorithm (BFS)

- **Question:** How to compute the diameter of a graph? Complexity?
- **Question:** How to compute a good lower bound (and upper bound) to the diameter using only a few BFS? Complexity?
- **Question:** How to generalize BFS if edges have weights?

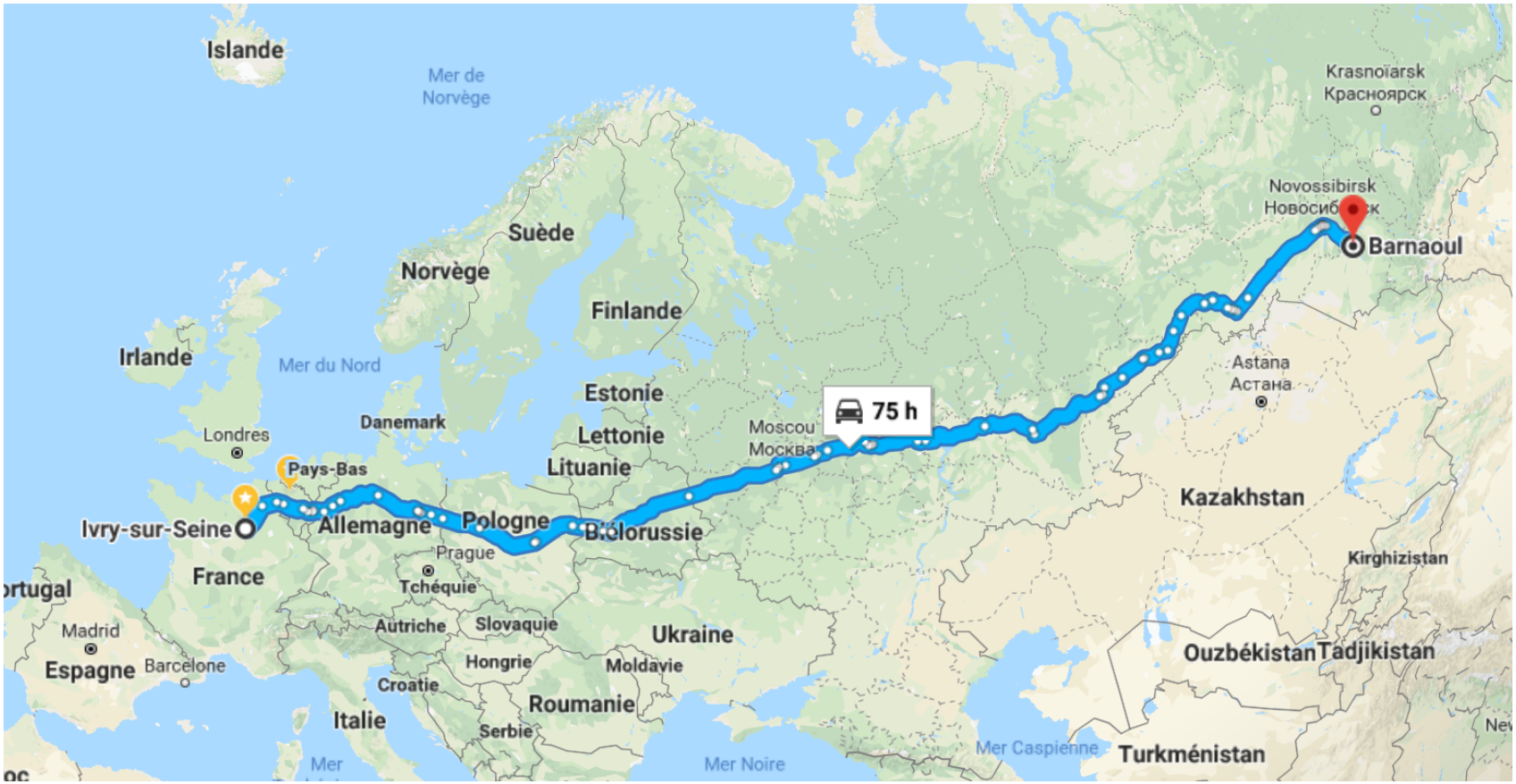
Dijkstra algorithm

Algorithm 2 Dijkstra on a weighted graph G_w from node s

function DIJKSTRA(G_w,s)
  for each node u in G_w **do**
    $D[u] \leftarrow$ INFINITY
  $D[s] = 0$
  while G_w not empty **do**
    $u \leftarrow$ vertex in G_w with minimum $D[u]$
    output u
    for each v neighbor of u in G **do**
      $tmp \leftarrow D[v] + w_{uv}$
      if $tmp < D[v]$ **then**
        $D[v] = tmp$
    remove u from G_w

Question: Remarks? Complexity?

Dijkstra algorithm



Question: Is Dijkstra fast enough to answer that query in 1sec?

Listing triangles

Question: Suggest an algorithm to list all triangles?

Listing triangles

Algorithm: for each edge (u, v) in the graph, compute the intersection of the neighborhoods of u and v.

Remarks:

- Algorithm slow if u or v have high degree
- Triangle (u, v, w) is found 3 times

Idea: reduce as much as possible the size of the lists involved in the computation of the intersection:

- sort the nodes in non-increasing order of degree and re-index the graph (such that $u < v \Rightarrow d(u) \geq d(v)$)
- given an edge (u, v), we consider only the neighbors w of u (resp. v) such that $u < w$ (resp. $v < w$)

Listing triangles

Algorithm 3 List all triangles in G

function TRLIST(G)
  for each node u in G **do**
    $tsl[u] \leftarrow$ truncated and sorted list of neighbors of u
  for each edge (u, v) in G **do**
    $W \leftarrow Intersect(tsl[u], tsl[v])$
    for each node w in W **do**
      output triangle $\{u, v, w\}$

Question: Remarks? Complexity?

Listing k-cliques

Question: Given an integer k, suggest an algorithm to list all k-cliques?

Listing k-cliques

Question: Given an integer k, suggest an algorithm to list all k-cliques?

`https://papers-gamma.link/paper/32`