COMP26120: Algorithms and Imperative Programming

Graph Algorithms
Lecture 1: Graph traversal algorithms

Graph algorithms lectures outline

This part consists of 3 lectures with the following topics:

Lecture 1: Graph traversal algorithms

- Basic terminology and representations;
- Data structures for representing graphs;
- Graph traversal algorithms:
  - Depth first search (DFS);
  - Breadth first search (BFS);

Lecture 2: Directed graphs

- Transitive closure and the Floyd-Warshall algorithm;
- Digraph traversal;
- The realistic example: Garbage collection;

Lecture 3: Shortest paths in graphs

- Dijkstra's algorithm;
- The Bellman-Ford algorithm;

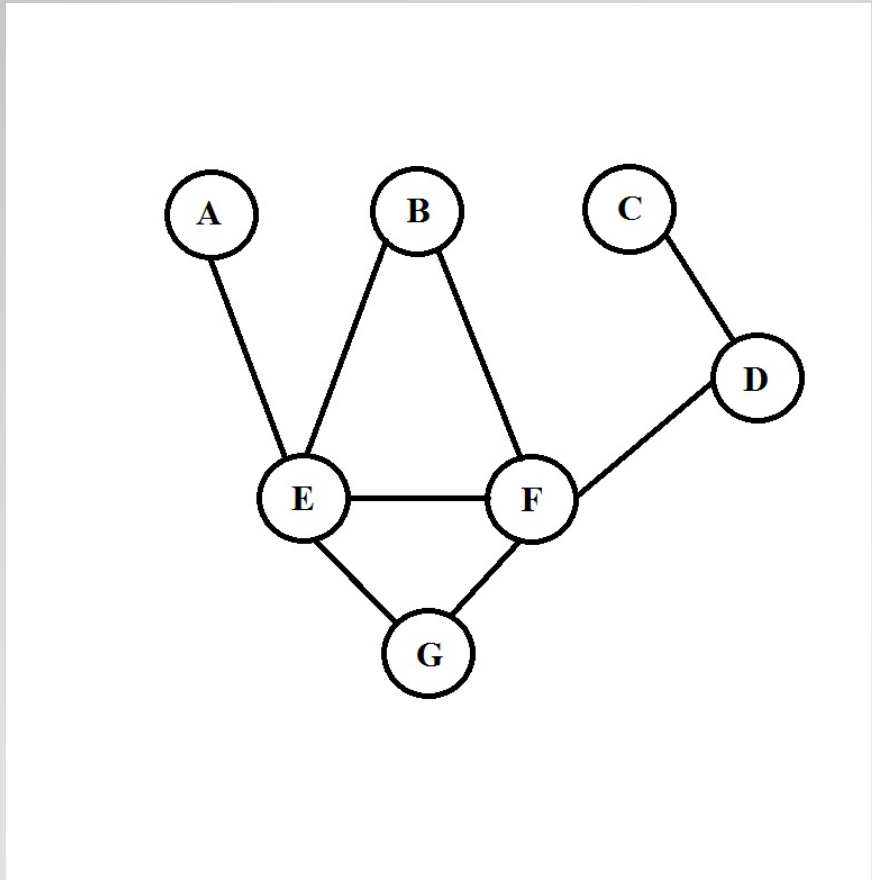
Graph traversal algorithms

- How would you define a graph?
  - A set of nodes containing some useful information connected by edges.
- But then, how would you define a tree?
  - A set of nodes containing some useful information connected by edges.
- What is the difference between them?

Graph traversal algorithms
Basic definitions

- Connectivity information is relevant to many areas in real life: computer networks, electricity and telecom grid, road and railway networks, social networks, gaming theory…
- We are interested in paths that exist in such structures.
- Graphs are associated with representations and algorithms for dealing with relationships that represent the connectivity between objects.
- A graph is a set of objects, called vertices, and a set of pairwise connections between them, called edges.
- A graph *G(E,V)* is a set *V* of vertices and a collection *E* of pairs of vertices from *E*, called edges.

Graph traversal algorithms
Basic definitions



- *G=G(V,E)*
- *V={A,B,C,D,E,F,G}*
- *E={(A,E),(B,E),(B,F),(C,D),*
  *(D,F),(E,F),(E,G),(F,G)}

Graph traversal algorithms
Basic definitions

- Edges in a graph can be directed or undirected. In the former case the pair (u, v) means that the edge originates in the vertex u and terminated in the vertex v.
- *The degree* of a vertex is the number of edges incident to it. In a directed graph we can distinguish in and out index.
- There can be multiple edges between two nodes, refereed to as *parallel edges*.
- Another special type of an edge is from a node to itself (*a self-loop*).
- If *G* is a simple undirected graph (no parallel edges or self loops) with n nodes and e edges, then $e \leq n(n-1)/2$.

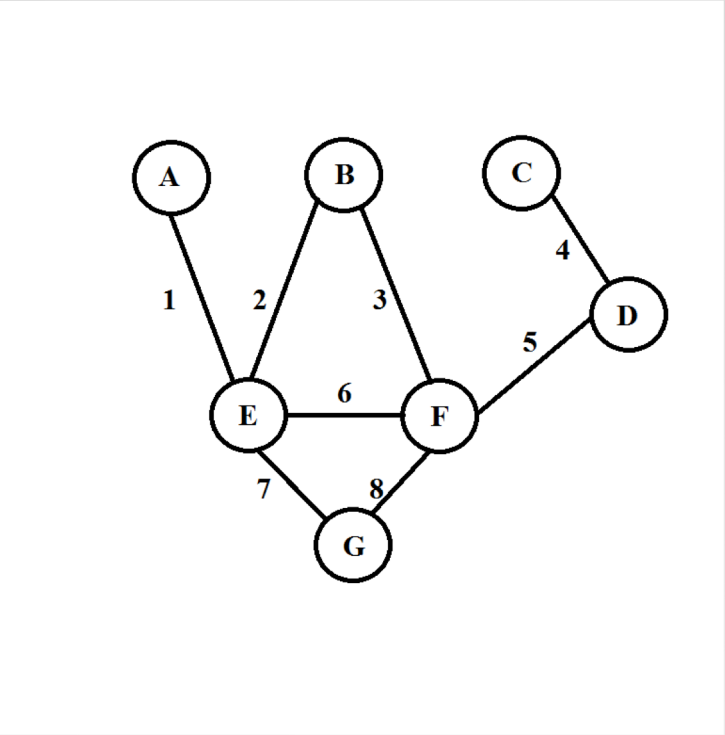
Graph traversal algorithms
Basic definitions

- A path in a graph is a succession of edges between a number of nodes. A cycle in a graph is a path with the same start and end vertex.
- A subgraph *H* of a graph *G* has vertices and edges that are subsets of those in a graph *G.* A spanning subgraph has the same nodes as the original graph, but a subset of edges.
- A forest is a graph without cycles. A tree is a connected forest.

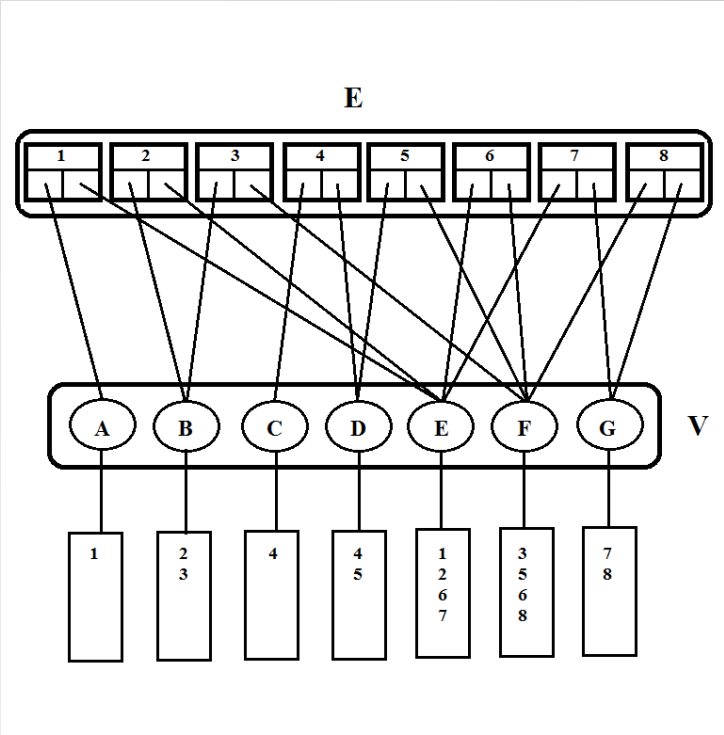
Graph traversal algorithms
Data structures for graphs

- The two most commonly used data structures for representing graphs are the adjacency list and the adjacency matrix. Different representations may lead to different asymptotic execution times for graph operations.
- Homework: Read Operations on Graphs from GT (p. 360).

Graph traversal algorithms
Data structures for graphs

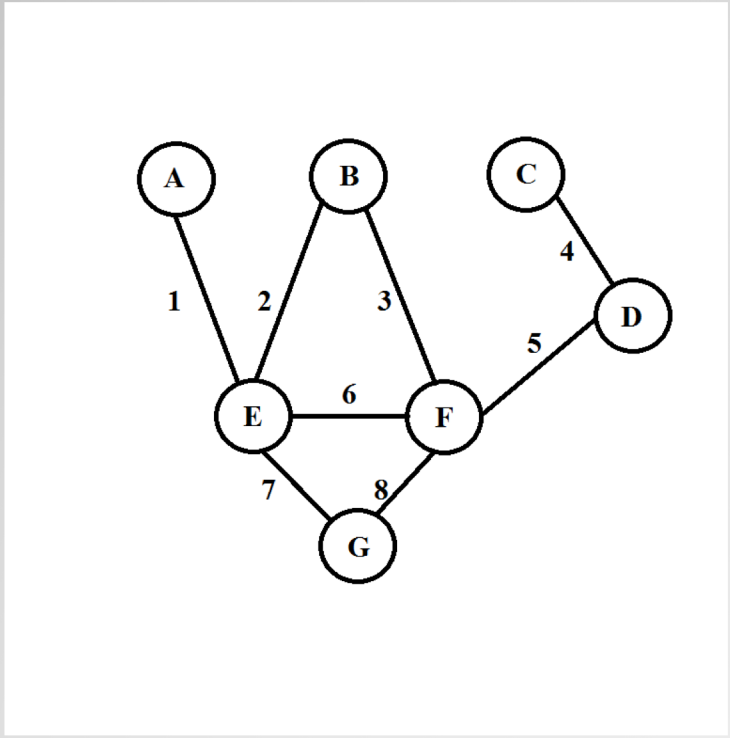


An example graph



Its adjacency list representation

Graph traversal algorithms
Data structures for graphs



An example graph

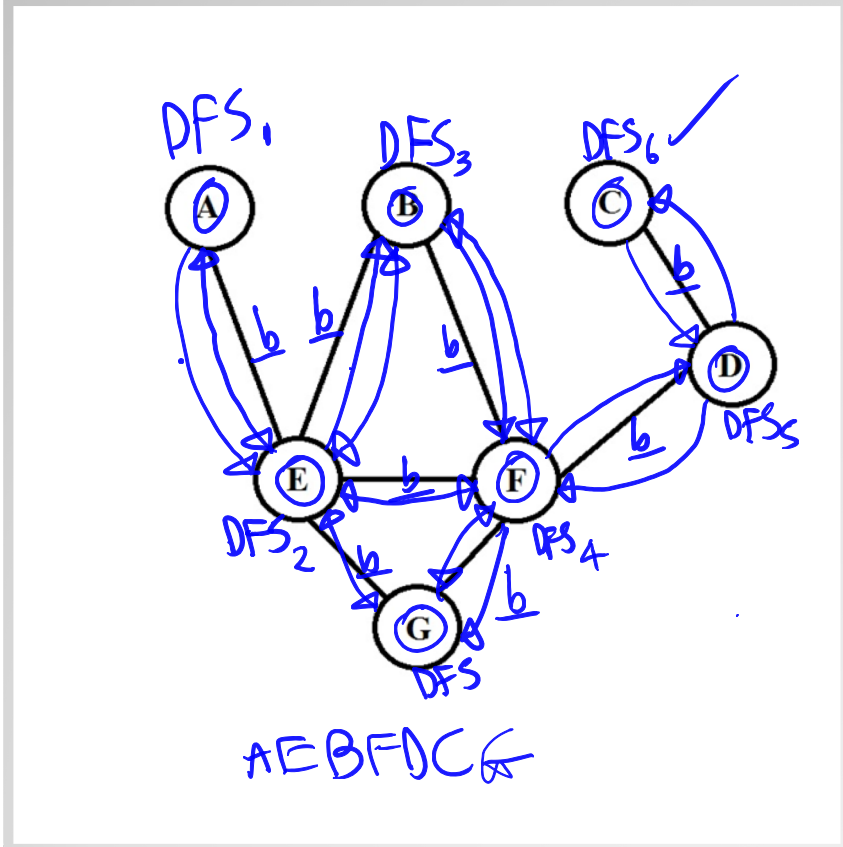
	A	B	C	D	E	F	G
A	0	0	0	0	1	0	0
B	0	0	0	0	2	3	0
C	0	0	0	4	0	0	0
D	0	0	4	0	0	5	0
E	1	2	0	0	0	6	7
F	0	3	0	5	6	0	8
G	0	0	0	0	7	8	0

Its adjacency matrix representation

Graph traversal algorithms
Depth first search

- A traversal is a systematic procedure for exploring graph by visiting his vertices and edges in a certain order.
- Depth first search of an undirected graph G applies backtracking technique (similar to systematically going through a labyrinth).
- We start from a node v, mark it as visited, and choose one edge to the neighbouring node. If that node is marked as visited, we backtrack. When this procedure leads to a dead end, i.e. to the node w, which all the neighbours are marked as visited, we backtrack, checking along the way whether the visited nodes have any edges leading to unexplored nodes.

Graph traversal algorithms
Depth first search



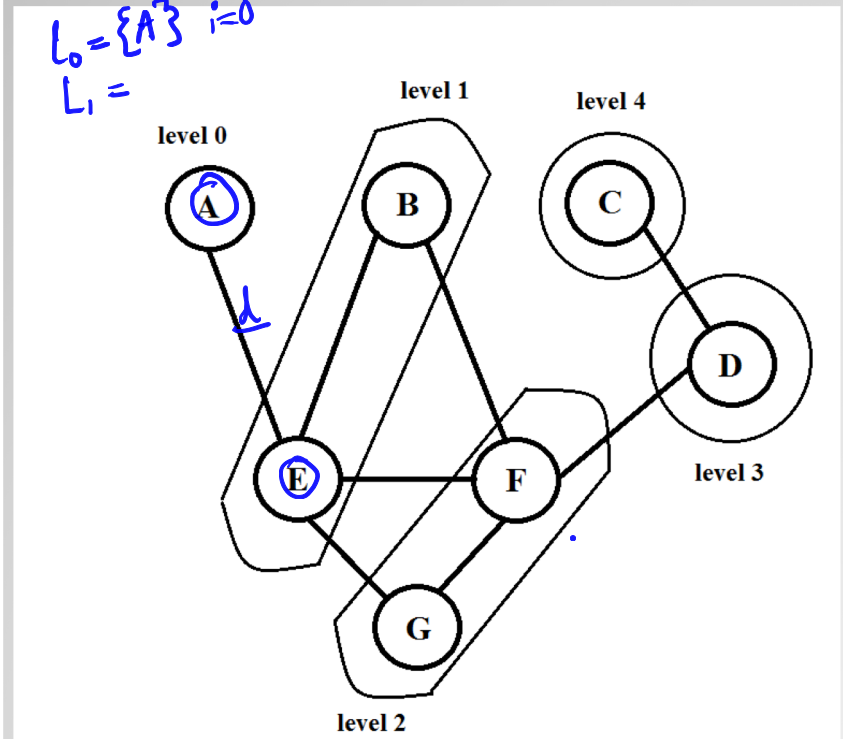
Algorithm DFS (G,v);
Input: Graph G, vertex v;
Output: discovery and backtrack edges and visited nodes;
Label v as explored;
For all e incident to v **do**
  if e is unexplored **then**
    go to w along e;
    if w is unexplored **then**
      d label e as a discovery edge;
      DFS(G,w);
    else
      b label e as a backtrack edge;
      go back along it to v;
    end if
  end if
end for

Complexity of the DFS algorithm for a graph with n vertices and e edges is $O(n+e)$.

Graph traversal algorithms
Breadth first search

- The Breadth first search (BFS) traverses a connected component of a graph and defines a spanning tree in the process.
- It does not search recursively, but subdivides the graph into levels, where all the nodes that belong to the same level have the paths of the same length that connect them to a fixed node.

Graph traversal algorithms
Breadth first search



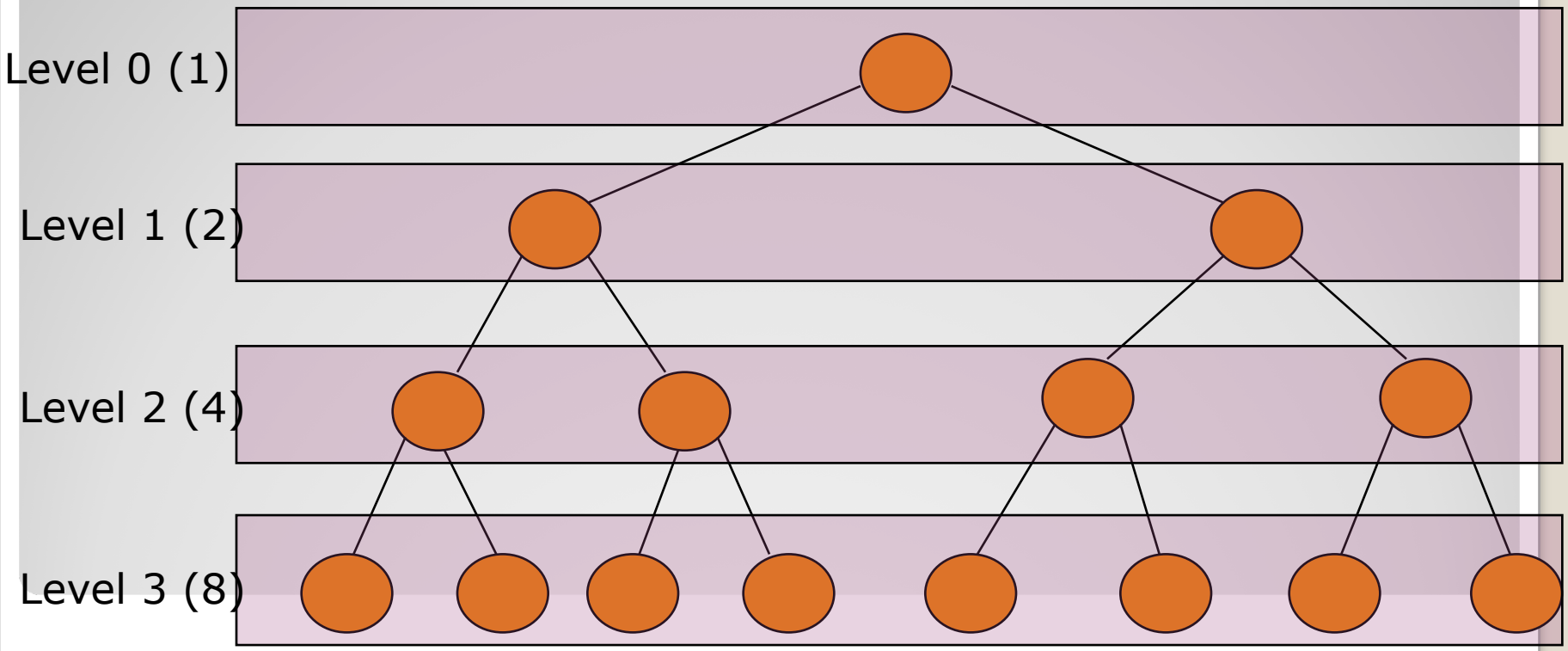
Algorithm BFS (G,s);
Input: Graph G, vertex s in G;
Output: A labelling of the edges as discovery or cross edges;

Create an empty list L_0. Put s into L_0 and mark it as explored; $i = 0$;
While L_i is not empty **do**
  Create an empty list L_{i+1};
  for each vertex $v \in L_i$ **do**
    for each edge $e = (v, w)$ **do**
      if e is not explored **then**
        if vertex w is not explored **then**
          Label e as a discovery edge;
          Mark $w \in L_{i+1}$ as explored;
        else
          Label e as a cross edge;
        end if
      end if
    end for
  end for
  $i = i + 1$;
end while

Complexity of the BFS algorithm for a graph with *n* vertices and *e* edges is $O(n + e)$.

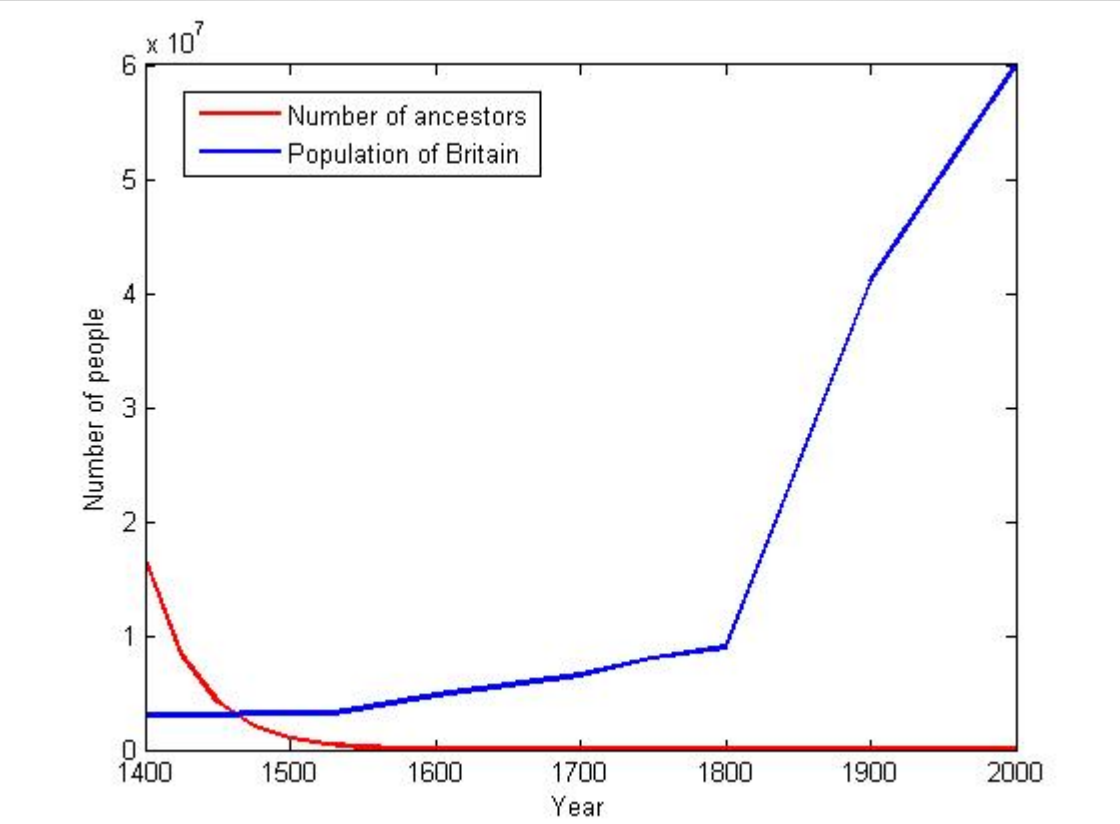
Problem solving
The ancestry tree

- The ancestry relations can be represented as a complete binary tree.



Problem solving
The ancestry tree

- But, is it a tree? Let's make the following assumption:
  - Each new generation emerges after 25 years (i.e. we have 4 generations in a century).



Problem solving
The ancestry tree

- The problem with a model: we are running out of people who are candidates for the ancestors.
- Looking further back we would have:
  - At the start of the new era (0 AD) according to our perfect binary tree model we would have 80th generation of our ancestors and there would be $2^{80} \approx 1.2 \cdot 10^{24}$ of them, if they are considered to be unique.
  - For comparison, todays' population of the world is 7.7 billion $\approx 2^{33}$.
  - At 0 AD the population of the Roman Empire was less than 65 million people.
  - Where are the ancestors come from?

Problem solving
The ancestry tree

graph

- Some of the nodes in the ancestry tree are not unique!