MAU22C00 Lecture 19

John Stalker

Trinity College Dublin

Strong extensionality

What is wrong with the strong version of Extensionality?

Pick a set.

If it's non-empty, pick a member.

In the strong version of Extensionality this member must be a set.

If it's non-empty, pick a member.

Continue as long as you can.

No matter what choices you make, either this never stops or you end up with the empty set.

If you assume Foundation then in fact it always stops with the empty set.

Really, there's only the empty set and sets constructed from it, e.g. $\{\emptyset, \{\emptyset\}, \{\emptyset, \{\emptyset\}\}\}$.

Zermelo-Fraenkel

The usual version of set theory is called Zermelo-Fraenkel, after Zermelo, who had a somewhat reasonable axiomatic system, and Fraenkel, who vandalised it.

- Extensionality (strong version)
- Elementary sets
- Selection
- Union
- Power sets
- Infinity
- Replacement
- Foundation
- Choice?

The version without Choice is called ZF while the version with it is called ZFC.

Banach-Tarski

This version of set theory has some unfortunate consequences.

There are sets A_1, A_2, A_3, A_4, A_5, B_1, B_2, B_3, C_1, C_2, C_3, C_4 and C_5 in three dimensional Euclidean space with the following properties.

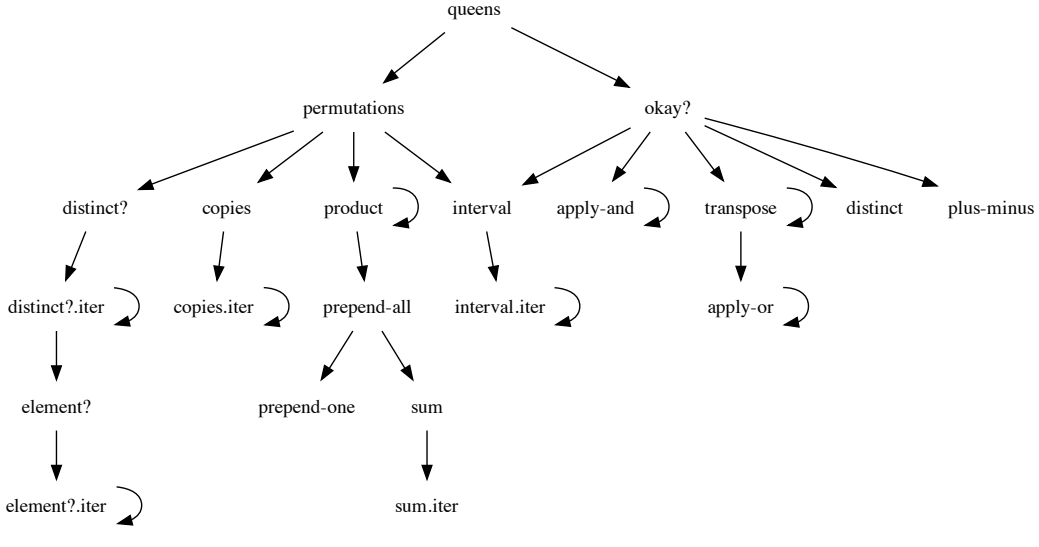
- B_1, B_2 and B_3 are disjoint balls of radius 1.
- A_1 is congruent to C_1, A_2 is congruent to C_2, A_3 is congruent to C_3, A_4 is congruent to C_4, and A_5 is congruent to C_5.
- A_1, A_2, A_3, A_4 and A_5 are disjoint and their union is $B_1 \bigcup B_2$.
- C_1, C_2, C_3, C_4 and C_5 are disjoint and their union is B_3.

In other words, we can take a ball, split it into five pieces, move those pieces via a rigid motion, i.e. a combination of translations, reflections and rotations, and reassemble them to form two balls of the same radius as the original one.

The principal culprit here is the (unrestricted) Axiom of Choice. This doesn't happen with weaker versions like Dependent Choice.

Graph Theory

Graphs in this module are not what they are in other maths modules. They're things like this directed graph:



Call graphs

The preceding was the call graph of a Scheme program to list all solutions of the 8 queens problem in chess.

There's a "vertex" for each defined function (excluding ones provided by the language) and an "edge" from each function to each function that it calls.

Some functions call themselves, leading to edges from a vertex to itself, called self-loops.

Scheme allows mutual recursion as well, which would give more complicated loops.

A program without recursion would have a call graph which is a tree.

This is actually a fairly simple program, just 61 lines. For comparison LLVM had 23,852,309 lines of code when I last checked. LLVM has a call graph but not one a human could expect to understand.

Other graphs

Graphs are ubiquitous in programming, not just for code but for data. Objects typically have pointers to other objects. There's a graph with a vertex for each object in memory and an edge between any pair of objects where one points to another.

One approach to garbage collection is to traverse this graph, marking all reachable objects, and then free anything left unmarked.

Module dependencies at a university also give a graph. I kept such a graph handy when I was the school undergraduate director.

Directed and undirected graphs

All the graph examples so far were *directed* graphs. There was a direction to all the edges.

Sometimes there is no preferred direction, in which case we have an *undirected* graph. Network connectivity graphs are often, but not always, undirected.

One of the early computer games, Hunt the Wumpus (1973), had a cave with twenty rooms, each connected to three other rooms by passageways. The passageways were bidirectional, so this is an undirected graph, shown below.

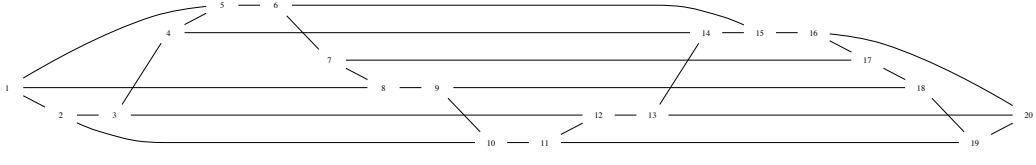


Figure 2: The Wumpus graph

Graph theories

Graph theory isn't really one subject but several, depending on the following choices:

- Are our graphs directed or undirected?
- How many vertices and edges do we allow? Finitely many? Countably many, uncountably many?
- Are self-loops, i.e. edges connecting a vertex to itself, allowed?
- Can there be more than one edge between a pair of vertices?

We'll consider directed and undirected graphs, with undirected graphs as a special case of directed graphs. Our graphs will be finite. Self-loops are generally allowed, but will be excluded in some theorems. There will be at most one edge from one vertex to another.

Definitions

A graph is a finite set together with a relation on that set.

We interpret the members of the set as vertices and interpret the pairs in the relation as edges, with the first element as the initial vertex and the last element as the final vertex.

With this definition an undirected graph corresponds to a symmetric relation.

A graph is called complete if it has no self-loops but otherwise has edges between any two distinct vertices.

A graph is called bipartite if there are two disjoint sets of vertices such that every edge runs from a vertex in one set to a vertex in the other.

A graph isomorphism is a bijective function from the vertices of one graph to another which preserves edges.

An automorphism is an isomorphism from a graph to itself.

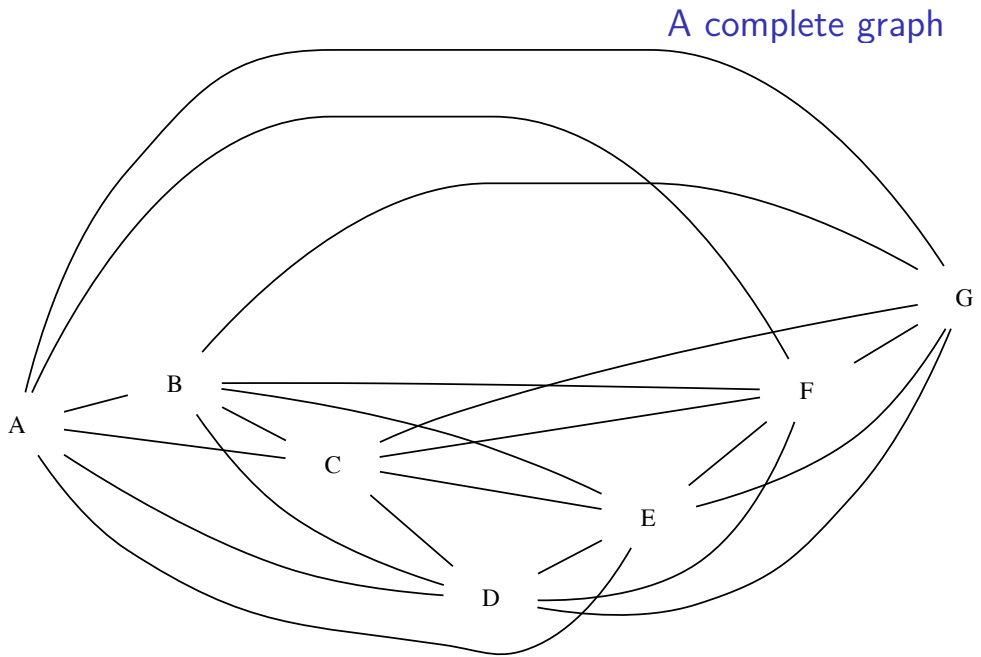


Figure 3: A complete graph

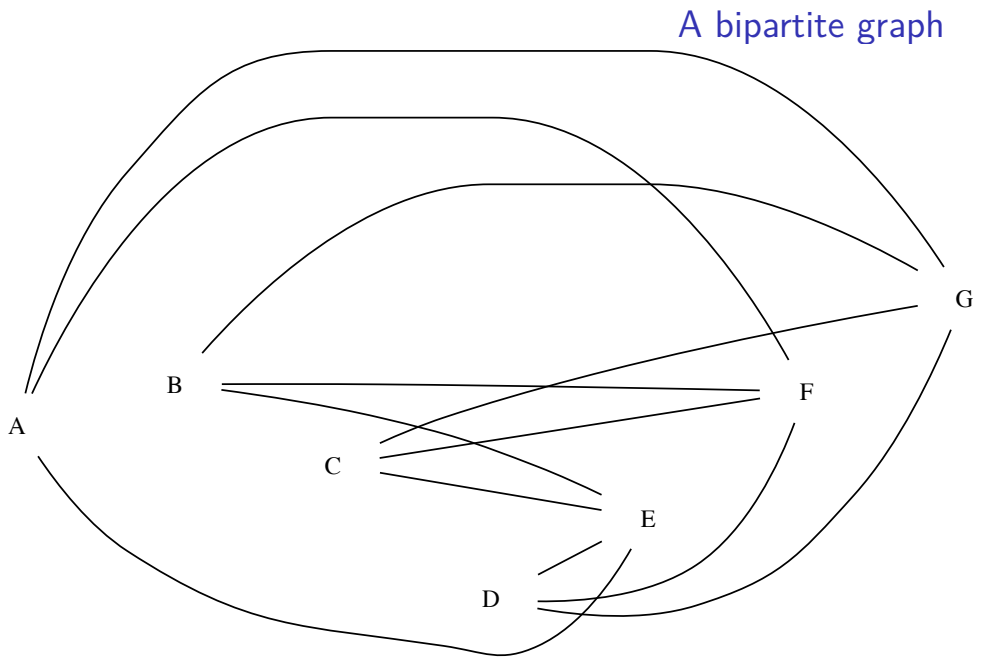


Figure 4: A bipartite graph

Isomorphisms, automorphisms

Two complete graphs are isomorphic if and only if they have the same number of vertices.

A (the?) complete graph with n vertices is called K_n, so the one given a moment ago is K_7.

K_n has $n!$ automorphisms, since any permutation of the vertices is an automorphism.

$7! = 5040$.

The bipartite graph example has $4! \cdot 3! = 144$ automorphisms, since we can permute the vertices A, B, C, D and E, F, G separately.

This bipartite graph has an edge between each vertex in one subset and each in the other, but that's not required.

A bipartite graph with vertices divided into a set of p and a set of q and edges between each possible pair of vertices is called a $K_{p,q}$, so that example is a $K_{4,3}$.

Subgraphs

A subgraph is a subset of the vertices together with a subset of the restriction of the edge relation to that set.

Our bipartite graph example is a subgraph of the complete graph example.

The complete graph example is also a subgraph of itself.

The graph with vertices A, B, C, D, E, F, G and no edges is a subgraph of each.

So is the graph with no vertices and no edges.