Tutorial 3

1 Scheduling Courses

We'll start by finishing up the problem begun in lecture. We discussed a set of MIT courses and the prerequisite relationships between them. The problem was to determine how a student could complete all of the courses in the fewest terms, ignoring factors like course availability, human tolerance for problem sets, etc. The approach was to model the problem as a directed acyclic graph or DAG. Each course was represented by a vertex. If course a was a prerequisite for course b, then there was a directed arc from a to b.

We let t be the number of vertices in the longest path in the graph. At least t terms are necessary to complete all the courses, because at most one course on the longest path can be taken each term. The remaining difficulty is to show that t terms are sufficient to complete all courses. In particular, we want to prove the following theorem:

Theorem 1 *If the longest path in a DAG G has length t, then the vertices can be partitioned into t antichains $A_1, A_1, \ldots, A_t$ such that for all $a \in A_i$, if there is an arc from b to a, then $b \in A_1 \cup A_1 \cup \ldots \cup A_{i-1}$.*

This theorem would imply that t terms are sufficient. In term i, the student should take the set of courses A_i. Since the sets $A_1, A_2, \ldots, A_t$ partition the set of all courses, the student will complete every course. Furthermore, the theorem ensures that every prerequisite for a course was completed in a previous term.

Proof. Let A_i consist of all vertices a such that the longest path ending at a contains i vertices. This partitions the vertices of the graph into sets $A_1, A_2, \ldots, A_t$.

First, we have to prove that each set A_i is an antichain. The proof is by contradiction. Assume for the purpose of contradiction that A_i is not an antichain. Then there exist distinct vertices $a, b \in A_i$ such that there is a path from a to b. By definition of the set A_i, the longest path ending at a contains i vertices. If we append to this the path from a to b, then we obtain a path ending at b with more than i vertices. This contradicts the assertion that $b \in A_i$.

Second, we have to show that for all $a \in A_i$, if there is an arc from b to a, then $b \in A_1 \cup A_1 \cup \ldots \cup A_{i-1}$. Again, the proof is by contradiction. Assume for the purpose of contradiction that $b \notin A_1 \cup A_1 \cup \ldots \cup A_{i-1}$. Then the longest path ending at b must contain more than $i - 1$ vertices. If we append the arc from b to a, then there is a path ending at a containing more than i vertices. This contradicts the assertion that $a \in A_i$. ∎

2 Buildup Error

"Buildup error" is a common source of mistakes in inductive proofs. We'll look closely at the anatomy of this error, but remember that this is an error to *avoid*, not a tool to *use*.

Buildup error occurs when the induction hypothesis $P(n)$ is itself an implication of the form $A(n) \Rightarrow B(n)$. In the inductive step of a correct proof, we assume $A(n) \Rightarrow B(n)$ in order to prove $A(n+1) \Rightarrow B(n+1)$. In an erroneous proof, we assume $B(n)$ in order to prove $A(n+1) \Rightarrow B(n+1)$. Here is how the error looks in a complete, false proof:

False Theorem 2 *If every node in a graph is adjacent to a node of the same color, then the graph is monochromatic.*

To see that the theorem is false, consider a cycle on four nodes with two red nodes followed by two blue nodes. Each node is adjacent to some node of the same color, but the graph is not monochromatic.

Proof. The proof is by induction on the number of nodes in the graph. Let $P(n)$ be the predicate, "If every node in an n-node graph is adjacent to a node of the same color, then the graph is monochromatic."

In the base case, $P(1)$, the graph contains only one node and so is certainly monochromatic.

In the inductive step, assume $P(n)$ in order to prove $P(n+1)$. Consider a graph with $n+1$ nodes and pull out one special node. By induction, the remaining n-node graph is monochromatic. Now if the special node is adjacent to a node of the same color, then it must be the same color as all other nodes in the graph. Therefore, the $n+1$-node graph is monochromatic. ■

The flaw is in the statement, "By induction, the remaining n-node graph is monochromatic." This is incorrect. By induction, we can assume only that *if every node in the remaining n-node graph is adjacent to a node of the same color*, then the remaining n-node graph is monochromatic." The correct assumption does not lead to a proof. Good thing, since the theorem is false!

3 Well-Ordering

There is another variation of induction that is sometimes useful: well-ordering. The well-ordering axiom looks nothing like the induction axiom:

Axiom 1 *Every nonempy set $S \subseteq \mathcal{N}$ has a smallest element.*

The well-ordering axiom is often called the *well-ordering principle*. It may seem obvious and useless!

As for obvious, note that this axiom would be false if the set of non-negative integers, $\mathcal{N}$, were replaced by, say, the set of non-negative rational numbers. The subset S consisting of positive rational numbers has no smallest element.

As for useless, it may be surprising that everything that can be proven by induction can also be proven using the well-ordering principle!

An example of a well-ordering proof is given below. Define a *cycle* in a tournament to be a sequence of players such that each defeats the next and the last defeats the first. The *length* of a cycle is the length of the sequence of players. A *n-cycle* is a cycle of length n. For example, p_1 defeats p_2, p_2 defeats p_3, p_3 defeats p_4, and p_4 defeats p_1 is a 4-cycle.

Theorem 3 *If a tournament contains a cycle, then it contains a 3-cycle.*

Proof. The proof is by contradiction. Assume for the purpose of contradiction that the tournament contains a cycle, but no 3-cycle. By the well-ordering axiom we can let n be the length of the shortest cycle. By assumption $n > 3$. Consider three consecutive players in the cycle, p_1, p_2, and p_3. If p_1 defeats p_3, then p_2 can be removed to create a cycle of length $n - 1$. This contradicts the assumption that the shortest cycle had length n. If p_3 defeats p_1, then p_1, p_2, and p_3 form a 3-cycle, again a contradiction. ■

4 Bipartite Graphs

Recall that a graph is k-colorable if every vertex can be assigned one of the k colors so that no adjacent vertices have the same color. A 2-colorable graph is called *bipartite.* Many common types of graph are bipartite, and they often arise in computer science.

Theorem 4 *Every tree is bipartite.*

Before proving this theorem, we'll first prove a lemma about trees.

Lemma 1 *Every tree has a vertex of degree 1.*

Proof. The proof is by contraction. Suppose that there is an n-vertex tree that has no vertex of degree 1. Since a tree is connected, no vertex can have degree 0. As a result, every vertex must have degree at least 2. Since the number of edges in a graph is half the sum of the degrees, the number of edges must be at least $\frac{1}{2}(2n) = n$. This contradicts the fact that an n-vertex tree has exactly $n - 1$ edges. ■

Now we can use the lemma to prove the theorem that every tree is bipartite.

Proof. The proof is by induction. Let $P(n)$ be the predicate that every tree with n vertices is bipartite. In the base case, $P(1)$ is true because a graph consisting of a single vertex can be colored with just one color. In the inductive step, for $n \geq 1$ assume that every tree with n vertices is bipartite in order to prove that every tree with $n + 1$ vertices is bipartite.

Let G be a tree with $n+1$ vertices. By the lemma, we can pull off a vertex v with degree 1. The remaining n-node graph is a tree. (The reason is that we removed one node and one edge and did not introduce a cycle. Therefore, the remaining graph is acyclic and has n nodes and $n-1$ edges. This defines a tree.) By induction, the remaining graph is bipartite. The extra vertex v can take whatever color that its single neighbor does not. $\blacksquare$

Team Problems

1. Prove that a graph is bipartite iff it does not contain an odd cycle.

2. An *Euler cycle* of a graph is a cycle that contains every edge in the graph exactly once. Prove that a connected graph has an Euler cycle iff every vertex has even degree.