Spanning Tree

A spanning tree of a connected graph \( G(V, E) \) is a connected acyclic subgraph of \( G \), which includes all the vertices in \( V \) and only (some) edges from \( E \).

Think of a spanning tree as a “backbone”; a minimal set of edges that will let you get everywhere in a graph.

- Technically, a spanning tree isn’t a tree, because it isn’t directed.

- Notice: we could also use DFS and BFS to find spanning trees, as well as Dijkstra’s shortest path algorithm.

Constructing a Spanning Tree

Theorem: Every connected graph has a spanning tree.

Proof: Do the obvious thing: start a some node and grow the tree. The following algorithm does it.

Input \( G(V, E) \) [a connected graph]

Algorithm SpanTree

Choose a vertex \( v \) in \( G \)
\[ V' \leftarrow \{v\} \] [Initialize spanning tree \( T(V', E') \)]
\[ E' \leftarrow \emptyset \]
repeat while \( V' \neq V \)
- Pick \( c \in V' \) and \( c' \in V - V' \) such that \{\( c, c' \)\} \( \in E \)
- \[ V' \leftarrow V' \cup \{c'\} \]
- \[ E' \leftarrow E' \cup \{\{c, c'\}\} \]
endrepeat
Output \( T(V', E') \)

Why does this work:

- After each iteration of the loop, \( T \) is a tree which spans the subgraph containing the vertices in \( V' \).
- If \( V' \neq V \), you can always find a new vertex to add to \( V' \), since \( G \) is connected.

Minimum Spanning Trees

If we have weights on the edges, we often want to find a spanning tree with minimum weight. A slight modification of the previous algorithm does it.

Input \( G(V, E) \) [a connected graph]
\[ w(e) \] for all \( e \in E \) [Weights on edges]

Algorithm MinSpanTree

Choose a vertices \( v, v' \) in \( G \)
such that \{\( v, v' \)\} has minimal weight
(break ties arbitrarily)
\[ V' \leftarrow \{v, v'\} \] [Initialize spanning tree \( T(V', E') \)]
\[ E' \leftarrow \{\{v, v'\}\} \]
repeat while \( V' \neq V \)
- Pick \( c \in V' \) and \( c' \in V - V' \) such that \{\( c, c' \)\} \( \in E \)
and \{\( c, c' \)\} has minimal weight
- \[ V' \leftarrow V' \cup \{c'\} \]
- \[ E' \leftarrow E' \cup \{\{c, c'\}\} \]
endrepeat
Output \( T(V', E') \)
MinSpanTree: Correctness

For simplicity, suppose the edges weights are all unique.

**Lemma:** If the vertices of $G(V, E)$ are divided into two disjoint sets $V_1$ and $V_2$, then any minimum spanning tree of $G$ contains the minimum weight edge $e$ connecting a vertex in $V_1$ to a vertex in $V_2$.

**Proof (sketch):** Suppose not. Then there is a minimum weight spanning tree $T$ that doesn’t contain $e$. Add $e$ to try. There must now be a cycle containing $e$. Take away some other edge $e'$ that a “bridge” between $V_1$ and $V_2$. This gives you a spanning tree $T'$ with less weight than $T$. That’s a contradiction.

Game Trees

Trees are particularly useful for representing and analyzing games.

**Example:** *Daisy* is a game where players alternate picking petals from a daisy. A player gets to pick 1 or 2 petals. Whoever picks the last one wins. (There’s another version where whoever takes the last one loses; both get analyzed the same way.)

Here’s the game tree for 4-petal daisy:

---

Sum and Product Rules

**Example 1:** In New Hampshire, license plates consisted of two letters followed by 3 digits. How many possible license plates are there?

**Answer:** 26 choices for the first letter, 26 for the second, 10 choices for the first number, the second number, and the third number:

$$26^2 \times 10^3 = 676,000$$

**Example 2:** A traveling salesman wants to do a tour of all 50 state capitals. How many ways can he do this?

**Answer:** 50 choices for the first place to visit, 49 for the second, … : 50! altogether.

Chapter 4 gives general techniques for solving counting problems like this. Two of the most important are:

**The Sum Rule:** If there are $n(A)$ ways to do $A$ and, distinct from them, $n(B)$ ways to do $B$, then the number of ways to do $A$ or $B$ is $n(A) + n(B)$.

- This rule generalizes: there are $n(A) + n(B) + n(C)$ ways to do $A$ or $B$ or $C$
- In Section 4.8, we’ll see what happens if the ways of doing $A$ and $B$ aren’t distinct.

**The Product Rule:** If there are $n(A)$ ways to do $A$ and $n(B)$ ways to do $B$, then the number of ways to do $A$ and $B$ is $n(A) \times n(B)$. This is true if the number of ways of doing $A$ and $B$ are independent; the number of choices for doing $B$ is the same regardless of which choice you made for $A$.

- Again, this generalizes. There are $n(A) \times n(B) \times n(C)$ ways to do $A$ and $B$ and $C$
Some Subtler Examples

Example 3: If there are \( n \) Senators on a committee, in how many ways can a subcommittee be formed?

Two approaches:

1. Let \( N_1 \) be the number of subcommittees with 1 senator \((n)\), \( N_2 \) the number of subcommittees with 2 senator \((n(n-1)/2), \ldots \)

According to the sum rule:

\[
N = N_1 + N_2 + \cdots + N_n
\]

- It turns out that \( N_k = \binom{n}{k} \) \((n \text{ choose } k)\); this is discussed in Section 4.4
- A subtlety: What about \( N_0 \)? Do we allow subcommittees of size 0? How about size \( n \)?
  - The problem is somewhat ambiguous.
  - If we allow subcommittees of size 0 and \( n \), then there are \( 2^n \) subcommittees altogether.
  - This is the same as the number of subsets of the set of \( n \) Senators: there is a 1-1 correspondence between subsets and subcommittees.

How many ways can the full committee be split into two sides on an issue?

This question is also ambiguous.

- If we care about which way each Senator voted, then the answer is again \( 2^n \). Each subcommittee defines a split (those who voted Yes; those out vote No); and each split + vote defines a subcommittee.
- If we don’t care about which way each Senator voted, the answer is \( 2^n/2 = 2^{n-1} \).
  - This is an instance of the Division Rule.

2. Simpler method: Use the product rule!

- Each senator is either in the subcommittee or out of it: 2 possibilities for each senator:
  - \( 2 \times 2 \times \cdots \times 2 = 2^n \) choices altogether

General moral: In many combinatorial problems, there’s more than one way to analyze the problem.

Coping with Ambiguity

If you think a problem is ambiguous:

1. Explain why
2. Choose one way of resolving the ambiguity
3. Solve the problem according to your interpretation

- Make sure that your interpretation doesn’t render the problem totally trivial
More Examples

Example 4: How many legal configurations are there in Towers of Hanoi with $n$ rings?

Answer: The product rule again: Each ring gets to “vote” for which pole it’s on.

- Once you’ve decided which rings are on each pole, their order is determined.
- The total number of configurations is $3^n$

Example 5: How many distinguishable ways can the letters of “computer” be arranged? How about “discrete”?

For computer, it’s 8!:

- 8 choices for the first letter, for the second, ...

Is it 8! for discrete? Not quite.

- There are two e’s

Suppose we called them $e_1$, $e_2$:

- There are two “versions” of each arrangement, depending on which e comes first: discreet $e_1 e_2$ is the same as discreet $e_2 e_1$.

Thus, the right answer is $8!/2!

Permutations

A permutation of $n$ things taken $r$ at a time, written $P(n, r)$, is an arrangement in a row of $r$ things, taken from a set of $n$ distinct things. Order matters.

Example 6: How many permutations are there of 5 things taken 3 at a time?

Answer: 5 choices for the first thing, 4 for the second, 3 for the third: $5 \times 4 \times 3 = 60$.

- If the 5 things are $a, b, c, d, e$, some possible permutations are:
  
  \[
  \begin{align*}
  abc & \quad abd & \quad abe & \quad acd & \quad ace \\
  adb & \quad adc & \quad ade & \quad aeb & \quad aec \\
  \end{align*}
  \]

In general

\[
P(n, r) = \frac{n!}{(n-r)!} = n(n-1)(n-2) \cdots (n-r+1)
\]

Combinations

A combination of $n$ things taken $r$ at a time, written $C(n, r)$ or \( \binom{n}{r} \) (“$n$ choose $r$”) is any subset of $r$ things from $n$ things. Order makes no difference.

Example 7: How many ways are there of choosing 3 things from 5?

Answer: If order mattered, then it would be $5 \times 4 \times 3 = 60$. Since order doesn’t matter,

\[
abc, acb, bac, bca, cab, cba
\]

are all the same.

- For way of choosing three elements, there are $3! = 6$ ways of ordering them.

Therefore, the right answer is $(5 \times 4 \times 3)/3! = 10$:

\[
\begin{align*}
abc & \quad abd & \quad abe & \quad acd & \quad ace \\
& \quad ade & \quad bec & \quad bce & \quad cde \\
\end{align*}
\]

In general

\[
C(n, r) = \frac{n!}{(n-r)!r!} = \frac{n(n-1)(n-2) \cdots (n-r+1)}{r!}
\]
More Examples

Example 8: How many full houses are there in poker?

- A full house has 5 cards, 3 of one kind and 2 of another.
- E.g.: 3 5’s and 2 K’s.

Answer: You need to find a systematic way of counting:

- Choose the denomination for which you have three of a kind: 13 choices.
- Choose the three: \( C(4, 3) = 4 \) choices
- Choose the denomination for which you have two of a kind: 12 choices
- Choose the two: \( C(4, 2) = 6 \) choices.

Altogether, there are:

\[ 13 \times 4 \times 12 \times 6 = 3744 \text{ choices} \]