Discuss set cover

• When take union bound, also include one term for odds that too many sets are chosen

0.1 Treewidth

Imagine a canonical recursive solution on a graph

• Pick a vertex

• For each possible setting of it, compute optimum solution on rest of graph

• Kind of what we did in bounded search tree for FPT

• What goes wrong? Exponential search space. New problem for each setting of variable

Perhaps we can “memoize” the recursion, turn into dynamic programming?

• In many graph problems, feasibility is determined by “local” constraints on vertices and who they interact with
  – Graph coloring
  – Max independent set
  – vertex cover
  – matching

• edges in graph represent “interactions” constraining joint behavior of neighboring variables

• Perhaps only need to memoize one answer for each state of neighbors, ignoring rest of graph

• If no variable interacts with many others, get something exponential in the degree

• Intuition: maximum matching in a tree
  – Root tree anywhere
  – Compute, for $v$ with subtree $T$, max matching in $T$ with $v$ matched and unmatched
  – Can evaluate for $v$ given children tables of $v$

Elimination orderings

• Represent an “unravelling” of the graph one vertex at a time, with plans to memoize
Problem to be solved: when I eliminate a vertex, I create hidden interactions between neighbors of that vertex.

To represent those interactions, need to add edges between all neighbors.

If can do so without ever creating large neighbor set, there is hope!

Treewidth

- **Induced Treewidth**: size of largest neighbor set created by given elimination ordering
- **Graph Treewidth**: induced treewidth of best (smallest) elimination ordering

Treewidth 1: tree

Treewidth 2: series-parallel graphs

SAT

- Treewidth not just for problems on graphs
- Use graph to reflect interactions between any variables
- eg, edge between vars if share a clause
- Maintain truth-table for each clause—list of satisfying assignments for it
- Take eliminated vertex/variable \( v \)
- Combine its clause’s truth tables—combined clause is happy with a given assignments if original set are all happy for some setting of \( v \)
- Reduced formula is SAT iff original is
- Size of new clause: degree of \( v \)
- So, if small treewidth, never create large clause
- In which case, easy to maintain tables
- Runtime: \( n \) eliminations, each involving about \( 2^w \) work.