This lecture is about a cool data structure for maintaining rooted trees (potentially very unbalanced) in $O(\log n)$ time per operation. The operations include linking two trees together by adding an edge, and cutting an edge to split a tree into two trees, so the data structure is called link-cut trees. This result is our first solving a dynamic graph problem (where in general we can insert and delete edges); next lecture will see other solutions for trees and generalization from trees to undirected graphs. Link-cut trees have specific advantages in that they can represent information over root-to-node paths; for example, they can easily compute the min/max/sum of values stored in nodes or edges along any root-to-node paths. Most importantly, link-cut trees introduce two powerful tree decompositions: preferred-path decomposition (which we already used in Tango trees) and heavy-light decomposition. As we will cover them, link-cut trees also demonstrate how a surprisingly "care-free" use of splay trees is actually still efficient.

6.851 Advanced Data Structures

Spring 2012