Reading for This Lecture

• Primary
  – Horowitz and Sahni, Chapter 8
  – Grama and Kumar, Parallel Search Algorithms...
Parallel Branch and Bound

- Divide and conquer approach
- "Obvious" approach to parallelization
- Parallelize recursive version
- What are the problems with this?

- How does this compare to other divide and conquer algorithms (such as merge sort)?
A Better Approach

• Master-slave model
• Master process maintains
  – a priority queue of nodes
  – a pool of slave processes to process the nodes
• Whenever a slave finishes processing a node, the master determines its next course of action
  – keep one (or more) of the children
  – get a completely new node
Performance Measures

• Overall running time
• Measures of overhead/redundant work
  – Size of search tree
  – Average time to process a node
• Measures of idle time
  – Time slaves spend waiting for work
  – Percentage load of tree manager
Scalability Issues

• Master process will become a bottleneck
• This could result in idle time for the slaves
• Slaves could end up performing unnecessary work
  – Upper bounds not available as quickly
• Memory usage not distributed -- tree stored centrally
• Run-up time
A Decentralized Model

- Use a crowd computation model.
- Divide the problem into subproblems.
- Each process solves its assigned subproblem.
- What are the problems with this?
Load Balancing

- There are two types of load balancing needed
  - Quantitative
    - Each processor must have enough work to do
  - Qualitative
    - Each processor must have "important" work to do

- Global information is needed to make good load balancing decisions.

- We must make a compromise.
New Approaches

• Try to maintain as much global information as possible without creating bottlenecks.
  – Hierarchical schemes
  – Increased grain size
  – Shared memory

• Completely decentralize
  – Processes periodically give away some of their best nodes to neighbors.
  – Processes request work from each other when they need it.
  – Processes check the quality of their nodes against each other.
Implementing Parallel B and B

- Data structures needed
  - Representation of state
  - Representation subproblems
  - Representation of search tree
- Master-slave model
  - Need a priority queue (easy)
  - Store tree centrally (efficient)
- Crowd computation model
  - Still need to store everything and have some sort of priority queue, but how?
Shameless Plug

- SYMPHONY (Single- or Multi- Process Optimization over Networks) is an object-oriented framework for implementing parallel branch and cut.
- User supplies some subroutines that are specific to the problem-setting
- The remainder (about 90% of the work) is taken care of by SYMPHONY
- Can be easily used to solve a wide variety of discrete optimization problems.