Sender Initiated LD Algorithms

- The overloaded node attempts to send tasks to lightly loaded node
  - Transfer Policy: If new Tasks takes you above threshold, become sender. If receiving task will not lead to crossing over threshold, then become receiver
  - Selection Policy: Newly arrived tasks
  - Location Policy
    - Random – still better than no sharing. Constrain by limiting the number of transfers
    - Threshold – chose nodes randomly but poll them before sending task. Limited no. of polls. If process fails execute locally.
    - Shortest – Poll all randomly selected nodes and transfer to least loaded. Doesn't improve much over threshold.
  - Information Policy, Stability

Receiver initiated

- Load sharing process initiated by a lightly loaded node
  - Transfer Policy: Threshold based.
  - Selection Policy: Can be anything
  - Location Policy: Receiver selects upto N nodes and polls them, transferring task from the first sender. If none are found, wait for a predetermined time, check load and try again
  - Information Policy
  - Stability: At high loads, few polls needed since senders easy to find. At low loads, more polls but not a problem. However, transfers will tend to be preemptive.

Symmetric Algorithms

- Simple idea – combine the previous two. One works well at high loads, the other at low loads.
- Above Average Algorithm: Keep load within a range
  - Transfer Policy: maintain 2 thresholds equidistant from average. Nodes with load > upper are senders, Nodes with load < lower are receivers.
  - Location Policy: Sender Initiated:
    - Sender broadcasts “toohigh” message and sets up toohigh alarm
    - Receiver getting toohigh message replies with accept, cancels its toolow alarm, starts an awaitingtask alarm, and increments load value
    - Sender which gets accept message will transfer task as appropriate. If it gets toolow message, it responds with a toohigh to the sender.
    - If no accept has been received within timeout, send out changeaverage message.

- Location Policy: Receiver Initiated
  - A receiver broadcasts a toolow message and sets toolow alarm.
  - Upon receiving toohigh message, do as in sender initiated
  - If toolow alarm expires, send changeaverage message
- Selection Policy
- Information Policy is demand driven, and has low overhead. Each node can change the range individually.

Adaptive Algorithms

- Stable Symmetric Algorithm.
  - Use information gathered during polling to change behaviour. Start by assuming that everyone is a receiver.
  - Transfer Policy: Range based with Uppper and Lower Threshold
  - Location Policy: Sender Initiated component polls node at head of receiver list. Depending on answer, either a task is transferred or node moved to OK or sender list. Same thing happens at the receiving end. Receiver initiated component polls senders list in order, OK list and receivers list in reverse order. Nodes are moved in and out of lists at sender and receiver.
  - Selection Policy – any, Information Policy – Demand driven
  - At high loads, receiver lists get empty preventing future polling and “deactivating” sender component. At low loads, receiver initiated polling is deactivated, but not before updating receiver lists.
  - **Read Section 11.7, 11.8**

Requirements for load distribution

- Scalability
- Location Transparency
- Determinism
- Preemption
- Heterogeniety