

Program Design for Parallel Execution

Note Title

2/3/2009

Decomposition

serial alg. \rightarrow collection of tasks
that can execute in parallel

tasks are atomic

$$A[i,j] = f(A[i,j], A[i-1,j], A[i,j-1], \dots)$$

granularity $\overline{\quad}$ static vs. dynamic

Assignment

dividing tasks into processes/threads
aggregating tasks based on proc. count

Orchestration

shared memory vs. message passing
author code

Mapping

parallel prog. \rightarrow who executes what

might be manual

or OS decides

Orchestration

shared addr space (shared mem)

- implicit comm
- read triggered comm
- sync. explicit

message passing

- explicit comm
- send triggered "
- sync implicit

several flavors of explicit comm.

send/recv - pairwise

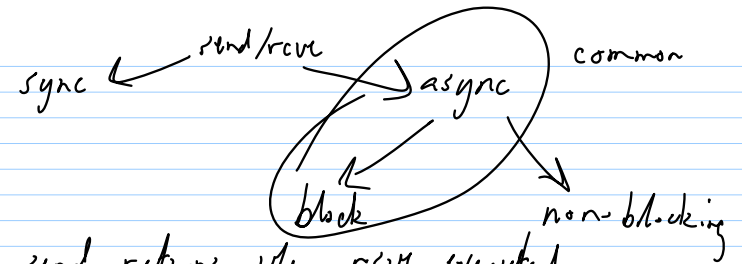
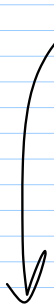
collective ops - group

barrier

reduction

assoc./comm. operator

+, max, min



sync send returns when recv executed

asynch send returns when buffer is free

blocking

non-blocking send the buffer is still unobtainable

non-blocking recv returns indicating whether successful