Announcements

- Reading for today's lecture is to finish chapter 2
- Lecture next week
  – Monday afternoon at 4:00 P.M. (room TBA)
- Class can be taken for 1 credit
  – this counts as seminar credit, but not for MS/PhD comps

Serial Computation

- Convert equations to discrete form
- Update from time t to t + delta t

```
foreach longitude, latitude, altitude
        ustar[i,j,k] = n * pi[i,j] * u[i,j,k]
        vstar[i,j,k] = m[j] * pi[i,j] * v[i,j,k]
        sdot[i,j,k] = pi[i,j] * sigmadot[i,j]
end
foreach longitude, latitude, altitude
        D = 4 * ((ustar[i,j,k] + ustar[i-1,j,k]) * (q[i,j,k] + q[i-1,j,k]) +
                        terms in {i,j,k}{+,-}{1,2}
        piq[i,j,k] = piq[i,j,k] + D * delat
        similar terms for piu, piv, piT, and pi
end
foreach longitude, latitude, altitude
        q[i,j,k] = piq[i,j,k]/pi[i,j,k]
        u[i,j,k] = piu[i,j,k]/pi[i,j,k]
        v[i,j,k] = piv[i,j,k]/pi[i,j,k]
        T[i,j,k] = piT[i,j,k]/pi[i,j,k]
end
```

Shared Memory Version

- in each loop nest, iterations are independent
- use a parallel for-loop for each loop nest
- synchronize (barrier) after each loop nest
  – this is overly conservative, but works
  – could use a single sync variabe per item, but would incurr excessive overhead
- potential parallelism is M * N * L
- private variables: D, i, j, k
- Advantages of shared memory
  – easier to get something working (ignoring performance)
- Hard to debug
  – other processors can modify shared data

Distributed Memory Weather

- decompose data to specific processors
  - assign a cube to each processor
    - maximize volume to surface ratio
    - minimizes communication/computation ratio
  - called a <block,block,block> distribution
- need to communicate {i,j,k}{+,-}{1,2} terms at boundaries
  - use send/receive to move the data
  - no need for barriers, send/receive operations provide sync
    - sends earier in computation too hide comm time
- Advantages
  - easier to debug
  - consider data locality explicity with data decomposition
- Problems
  - harder to get the code running

Seismic Code

- Given echo data, compute under sea map
- Computation model
  - designed for a collection of workstations
  - uses variation of RPC model
  - workers are given an independent trace to compute
    - requires little communication
    - supports load balancing (1,000 traces is typical)
- Performance
  - max mfops = $O((F * nz * B^*)^{1/2})$
  - F - single processor MFLOPS
  - nz - linear dimension of input array
  - B^* - effective communication bandwidth
    - $B^* = B/(1 + BL/w) \approx B/7$ for Ethernet (10msec lat., w=1400)
  - real limit to performance was latency **not** bandwidth

Database Applications

- Too much data to fit in memory (or sometimes disk)
  - data mining applications (K-Mart has a 4-5TB database)
  - imaging applications (NASA has a site with 0.25 petabytes)
    - use a fork lift to load tapes by the pallet
- Sources of parallelism
  - within a large transaction
  - among multiple transactions
- Join operation
  - form a single table from two tables based on a common field
  - try to split join attribute in disjoint buckets
    - if know data distribution is uniform its easy
    - if not, try hashing

Speedup in Join parallelism

- Books claims a speed up of1/p^2 is possible
  - split each relation into p buckets
    - each bucket is a disjoint subset of the joint attribute
  - each processor only has to consider N/p tuples per relation
    - join is $O(n^2)$ so each processor does $O((N/p)^2)$ work
    - so spedup is $O(N^2/p^2)/O(N^2) = O(1/p^2)$
- **this is a lie!**
    - could split into 1/p buckets on one processor
    - time would then be $O(p * (N/p)^2) = O(N^2/p)$
    - so speedup is $O(N^2/p^2)/O(N^2/p) = O(1/p)$
      - Amdahls law is not violated