MCMD L12: Parallel (Prefix) Sum

PRAM

1 disk
P processors
n input items

Each time step a processor can:
read, write, operate (+, -, *, <<, ...)

shared memory: CRCW (although CREW more realistic)

-------------------------
2 key resources to minimize
+ Ptime == total number of time needed to wait to complete
+ Work == total time if run sequentially
Let T = sequential time
Goal: Work = O(T)  Ptime << T  e.g.  O(log T)

Assume infinite number of processors.
Q: Why is this ok for algorithm design?
A: If need P and have p, split up work into P/p parallel chunks, run sequentially

-------------------------
Sum (n):
INPUT A = [a_1, a_2, ..., a_n]

Sequential?  O(n)

PRAM:  O(n/p + log n)

#############################
for i=1 to n PARDO
  B(0,i) := A(i)
for h = 1 to log n DO
  for i=1 to n/2^h PARDO
    B(h,i) := B(h-1,2i-1) + B(h-1,2i)
return B(log n, 1)
#############################

(log n) rounds:
A=B0 = 7 4 2 5 1 4 9 2
B1 = 11 7 5 11
B2 = 18 16
B3 = 34

O(n) work, O(log n) time

--------------------------
PRAM = BSP
--------------------------

Prefix Sum

INPUT A = [a_1, a_2, ..., a_n]
OUTPUT B = [a_1, a_1+a_2, a_1+a_2+a_3, ...]
b_i = sum_{j=1}^i a_i

Sequential? O(n)

########################
for i=1 to n PARDO
  B(0,i) := A(i)
for h = 1 to log n DO
  for i=1 to n/2^h PARDO
    B(h,i) := B(h-1,2i-1) + B(h-1,2i)
for h = log n to 0 DO
  for i=1 to n/2^h, even PARDO
    C(h,i) := C(h+1,i/2)
  C(h,1) := B(h,1)
  for i=3 to n/2^h, odd PARDO
    C(h,i) := C(h+1, (i-1)/2) + B(h,i)
Output C (PAROUT)
########################

Builds sum, then distributes back down.

(log n) rounds up, log n rounds down.

(log n) rounds:
A=B0 = 7 4 2 5 1 4 9 2
B1 = 11 7 5 11
B2 = 18 16
B3 = 34
C3 = 34
C2 = 18 34
C1 = 11 18 23 34
C0 = 7 11 13 18 19 23 32 34
$O(n)$ work, $O(\log n)$ time