CSE 332: Data Structures and Parallelism

Section 7: Parallel Primitives

0. Parallel Prefix Sum

Given input array $[8, 9, 6, 3, 2, 5, 7, 4]$, output an array such that each output$[i]$ = sum(array$[0]$, array$[1]$, ... array$[i]$), using the Parallel Prefix Sum algorithm from lecture. Show the intermediate steps. Draw the input and output arrays, and for each step, show the tree of the recursive task objects that would be created (where a node's child is for two problems of half the size) and the fields each node needs. Do not use a sequential cut-off.

1. Parallel Prefix FindMin

Given an input array $[8, 9, 6, 3, 2, 5, 7, 4]$, output an array such that each output$[i]$ = min(array$[0]$, array$[1]$, ... array$[i]$). Show all steps, as above.

2. Parallel Quicksort

(a) Show that Quicksort with sequential partitioning, but parallel recursive sorting, is indeed $\mathcal{O}(n)$, by solving the recurrence relation shown in lecture: T(n) = n+ T$(n/2)$.

(b) Show that a completely parallel Quicksort, with parallel partition and recursion, is $\mathcal{O}(log^2(n))$, by solving the recurrence relation shown in lecture: T(n) = $\mathcal{O}(log(n))$+ T$(n/2)$.