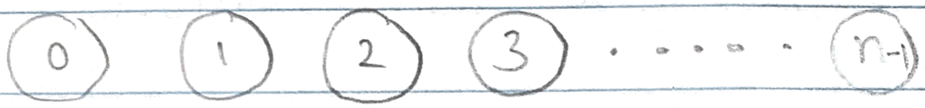
Activity Extracting Parallelism 1D Cases

1. Transform

Loop Iteration

Solu^n→ Complexity $O(n)$

Solu^n→ Granularity breaks down to loop Iteration, given call to function f are independent. and a write operation which involue individual indexes.



Solu^n→ width n $O(1)$

work n $\Theta(n)$

Critical Path, any single task from 1 to n

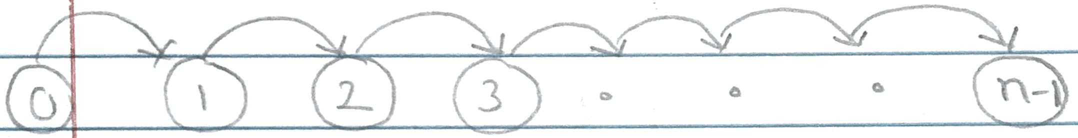
Length of CP = 1 $\Theta(1)$

2. Reduce

Solu^n → Complexity O(n)

Solu^n → Granularity comprises loop Iteration and call to function OP which in case takes two parameters one is result, which is being updated on every Iteratinon hence dependent and on Independent variable index from array.

Case. Read → Write.



0 → assignment

1 → loop

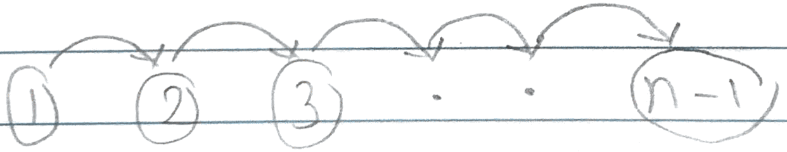
⋮

n

Width 1

Work n, Θ(n) OP is a constant time Operation assuming cost O(1)

Critical Path (whole Graph Itself).



Length of CP = n Θ(n)

3 Prefix Sum

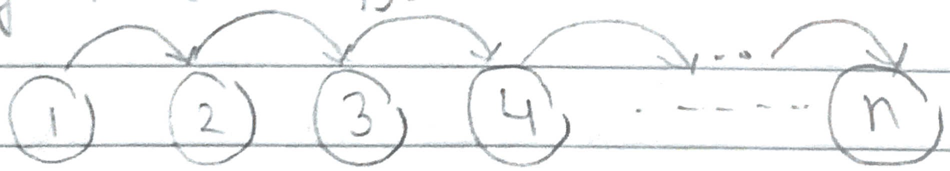
Solu^n→ O(n)

Solu^n→

```
pr[0] = 0                  task 1 (statement)

for (int i=0; i<n; i++)
    pr[i+1] = pr[i] + arr[i]
                           task 2  }
                             :     }  (loop iter..)
                           task n  }
      Case Read → write.
```

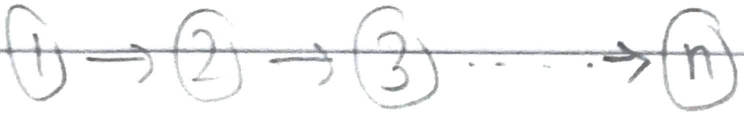
task ① is an assignment statement. and rest of the tasks are performed by loop, inside of which two variables one dependent and other indebendent are being used and written over to next index of pr (Array).



Solu^n→ Width 1

Work n, Θ(n)

Critical Path (whole Graph itself)



Length of CP = n Θ(n)