Design of Parallel Pipelined STAP on High-Performance Computers

Alok N. Choudhary

Northwestern University

Co-Authors: W. Liao, D. Weiner, P. Varshney

Syracuse University

R. Linderman and M. Linderman

Rome Laboratories

Goals

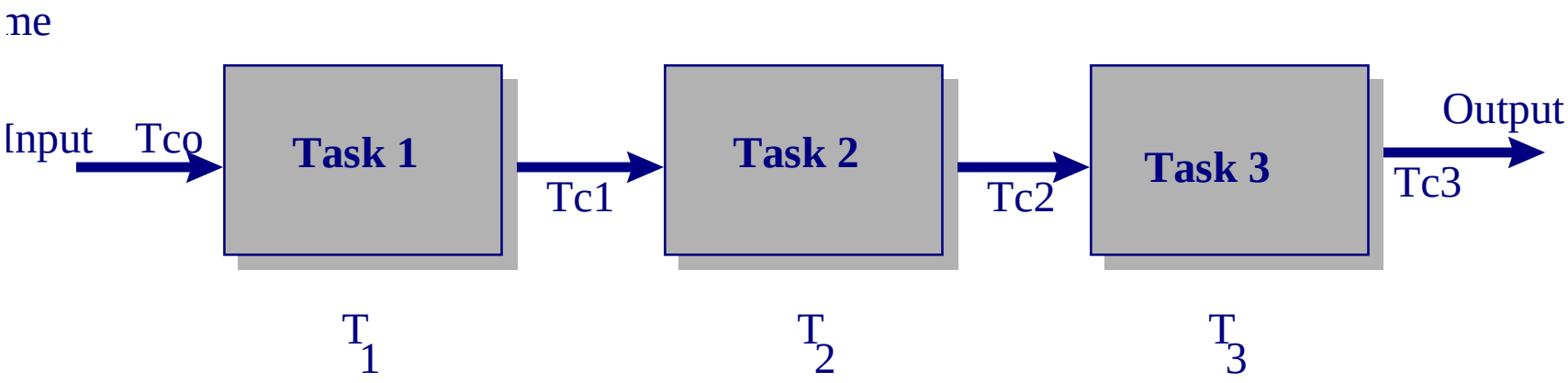
- ***Enable High-Performance implementation of computationally and structurally complex signal processing applications (e.g., STAP)***
- ***Use commercial HPC systems***
- ***Scalability of performance***
- ***Enable/develop daradigms not used/developed before (e.g., parallel pipelined… Beyond SPMD)***

Outline

- *Model of Computation*
- *Issues in Parallelization*
- *STAP Algorithm and Design for Parallel Implementation*
- *Approach to Software Development*
- *Initial Results*
- *Summary*

Model of Computation

- ***Pipelined Computations with Several Tasks***
- ***Unlike traditional parallel computations, tradeoffs in Latency and Thpt***



Resp Time = SUM(Ti+Tci)

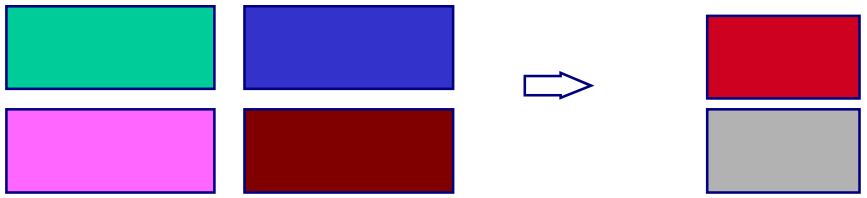
Thpt = 1/Max(Ti)

Issues

- ***Identify Each Task***
  - Potential for Parallelism
  - Scalability
  - Well Defined Computations
- ***Interface***
- ***Communication Requirements (one task to another)***
- ***Balance of Computation***

Data Redistribution

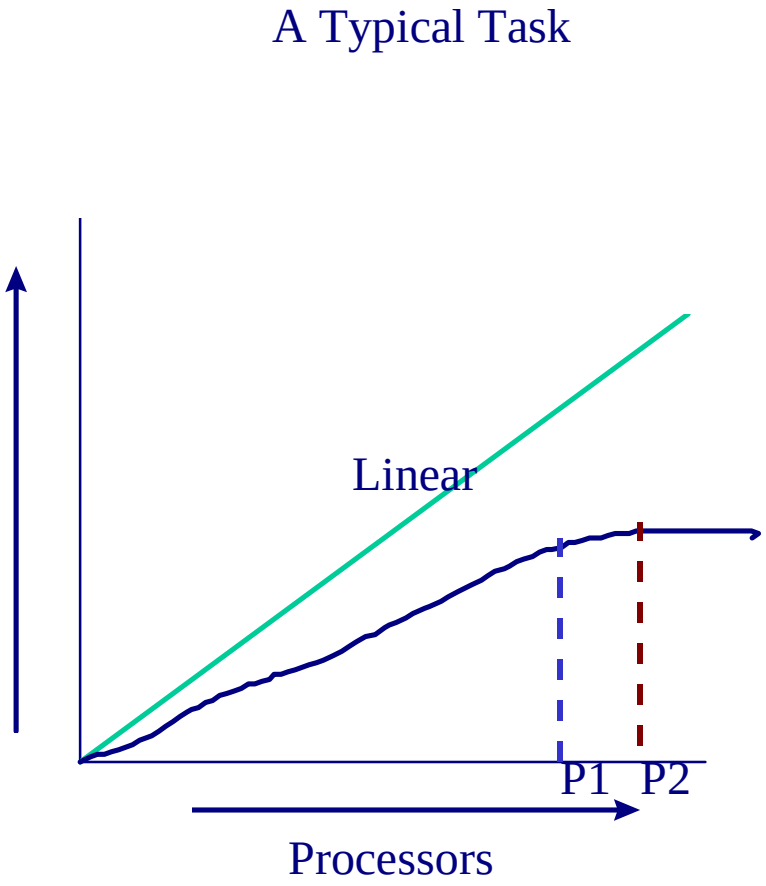
- ***Different distribution of data may be required from one task to next***
  - Different Number of Processors
  - Different data size
  - Better Parallelism (using a different distribution)



Task 1 (4 proc) Task 2 (2 proc)

Redistribution may be required from one phase of computation to another within a task

THPT: RESP Tradeoff



- ***Speed up Knee***
- ***=> Diminishing returns (P1->P2)***
- ***Data Parallel implementation alone not scaleable***
- ***Most tasks exhibit this property***
- ***How to determine NOP on which to execute a task?***

Task Scheduling and Processor Allocation

P	T1	T2	T3
1	100	150	60
2	55	85	32
4	28	45	17
6	20	33	12
8	16	25	11
10	14	21	10.5

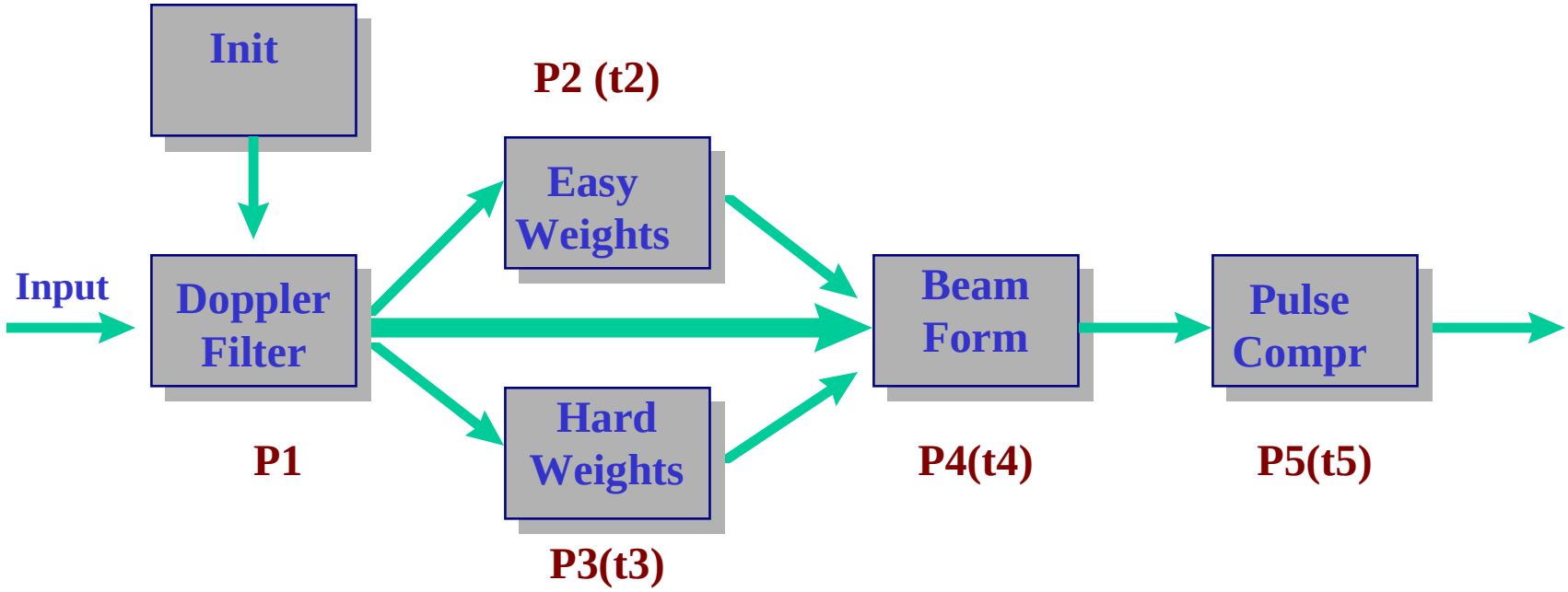
Assume total 20 Processors

- ***Characterization of speedup and execution times***
- ***Given total NOP, come up with allocation such that thpt maximized or response minimized***
- ***Off-line heuristic***
- ***Assumptions:***
  - Communication times included in Ti
  - Convex nature of performance scalability
- ***Only integer (and possibly subsets) of processors assignments possible***

STAP Algorithm

- ***PRI-Staggered Post-Doppler STAP***
- ***Steps***
  - Doppler processing (FFTs)
  - Beamforming (Matrix Multiplications)
  - Pulse Compression (FFTs and Inverse FFTs)
  - Computation of Weights (QR factorization)
    - Has two sub-tasks (easy weights and hard weights)

Overall Design and Macro Data-Flow



Total Time for a step (t) = t1+t4+t5
Throughput = 1/t

Software Design

- ***Each step has this structure***

Initialization
NOP, partition setup, global view..

Instrumentation
setup timers, structures for collection..

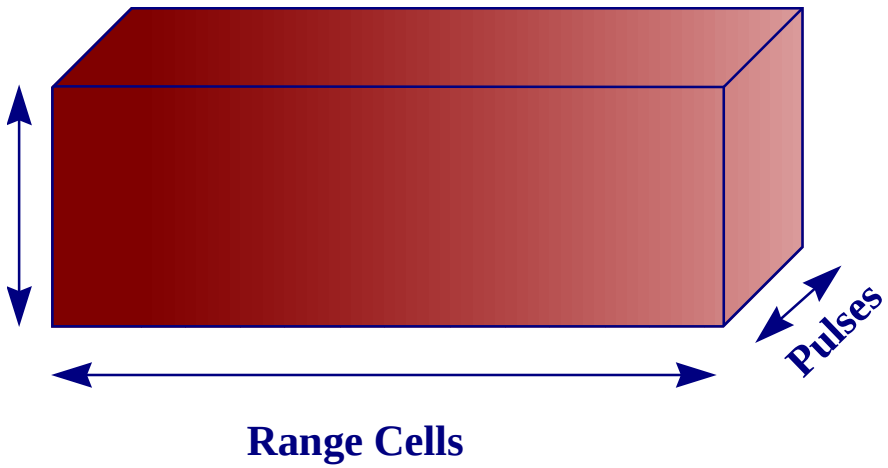
Communication Module
for receiving data, redistr...

Compute Module
Main computations of the step
(Possible instrumentation)

Communication Module
sending data to next task,redist..output

STAP Data Sets

- ***The STAP Data Set***



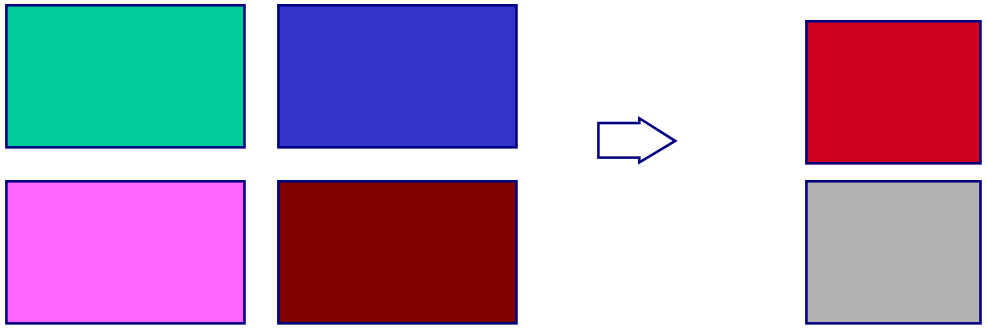
- ***Array Elements <=> Channels***
- ***Typical Data Sizes***
  - Range Cells 500+
  - Channels 16-32
  - Pulses 32-128
-

Initial Performance Results

- ***Platform - Rome Labs Paragon***
- ***Programming Paradigm for each task***
  - Node + Message Passing
  - Communication System - MPI
  - Language - C
- ***Global Paradigm***
  - Parallel Pipelined
  - Multiple parallel tasks communicate
  - communication itself between multiple tasks is parallel and scales

Step: Initialization

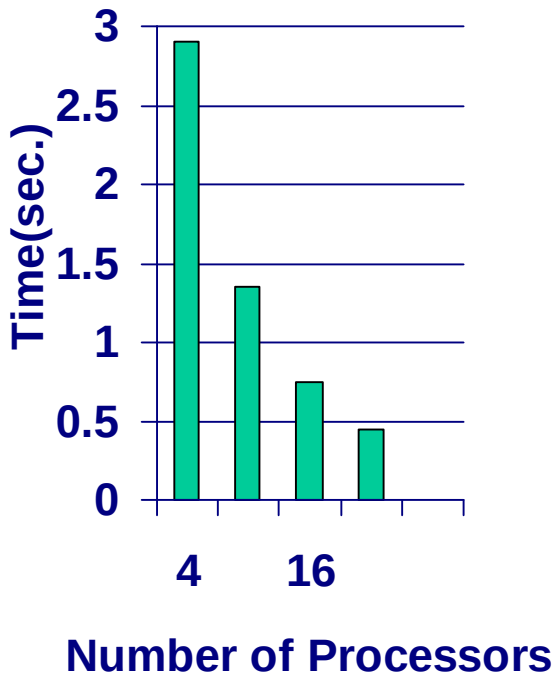
- ***Although the computation involved is the same as instep 1, There are three CPIs.. So more processors may be used for this (But it is one time)***



Initialization

Task 1

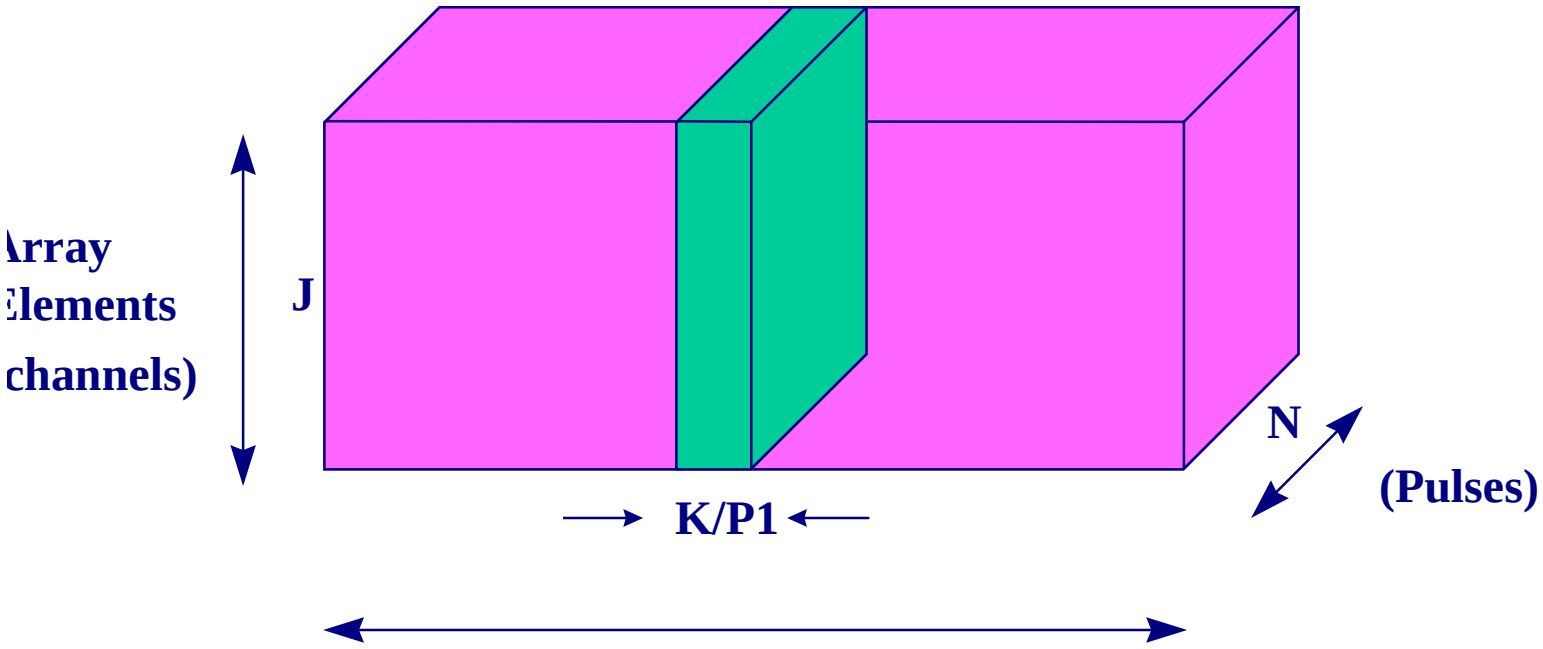
Step: Initialization



- *Data Size*
  - *Range Cells (K) = 512*
  - *No of Channels (J) = 16*
  - *Doppler Bins (N) = 128*
  - *Staggers = 2*
  - *No of Zero padding = 3*
- *Computational Steps*
  - *memory allocation for all arrays*
  - *Windowing, fft coefficient and range correction array initialization*
  - *Parameters setup*
  - *For 3 CPIS compute Doppler bins (next step done thrice here)*

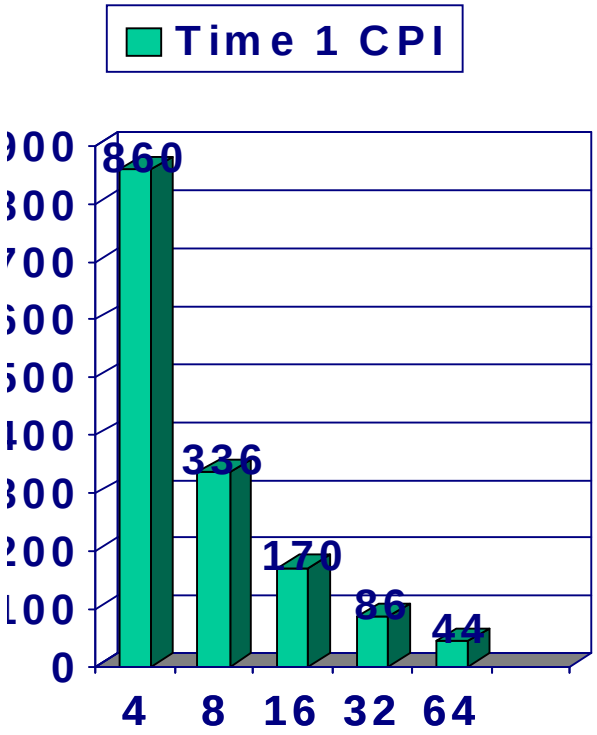
Step1: Doppler Filter

° Assumption: Cube is available at a time (time step)



Range cells are divided across processors

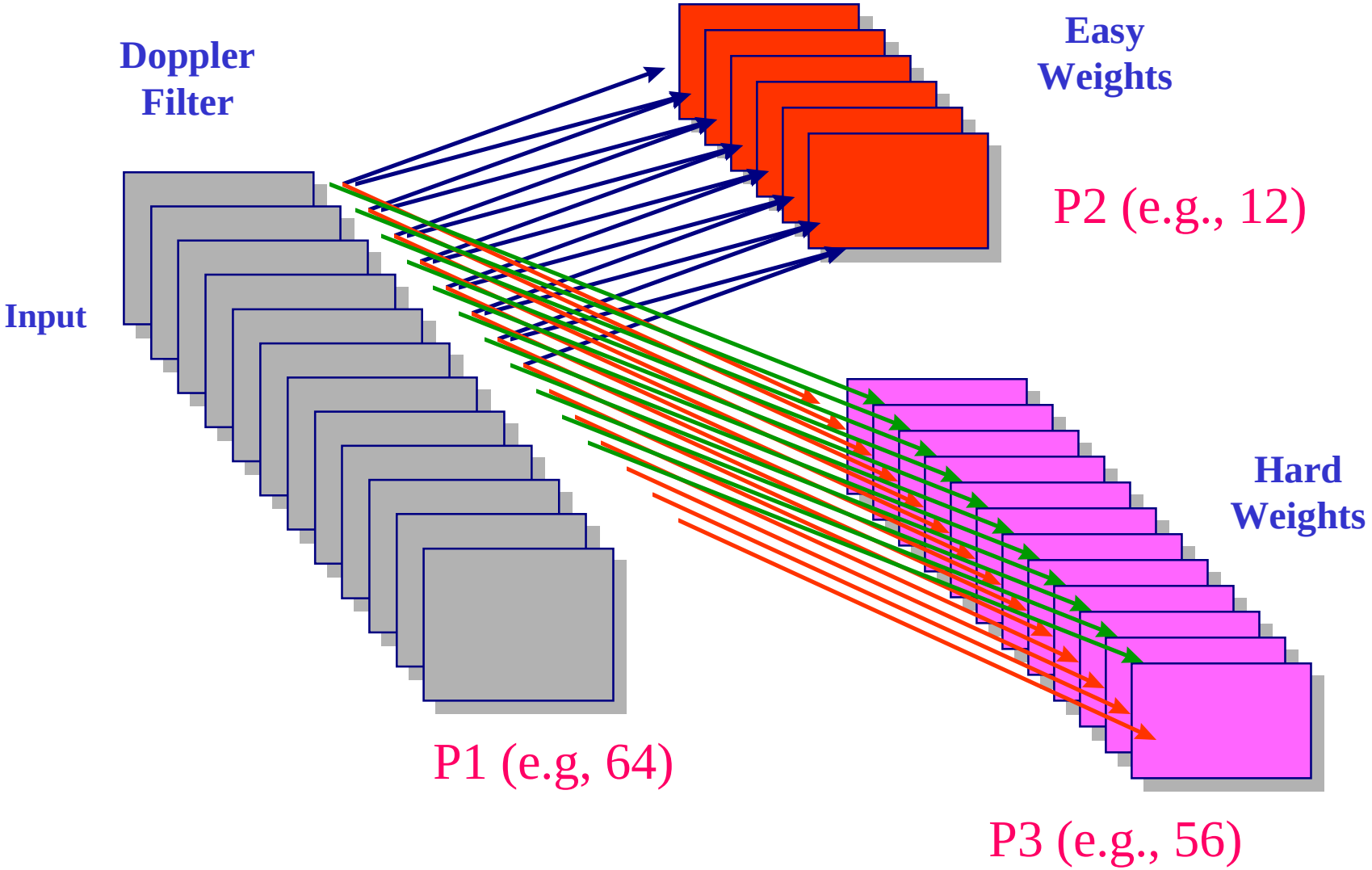
Step 1 : Doppler Filter



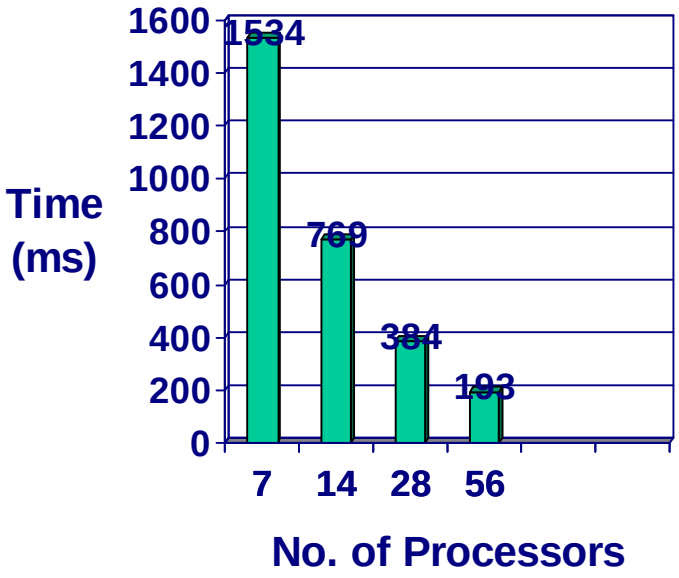
Speedup(4-64) = 19.5

- ***Time for One CPI (ms)***
- ***Data Size***
  - Range Cells (K) = 512
  - No of Channels (J) = 16
  - Doppler Bins (N) = 128
  - Staggers = 2
  - No of Zero padding = 3
- ***Computational Steps***
  - Range correction for each cell
  - Memory copy for second staggered CPI
  - Zero Padding on Both CPIs
  - Windowing on both CPIs
  - FFTs on both CPIs

Parallel Inter-Task Communication and Data Redistribution



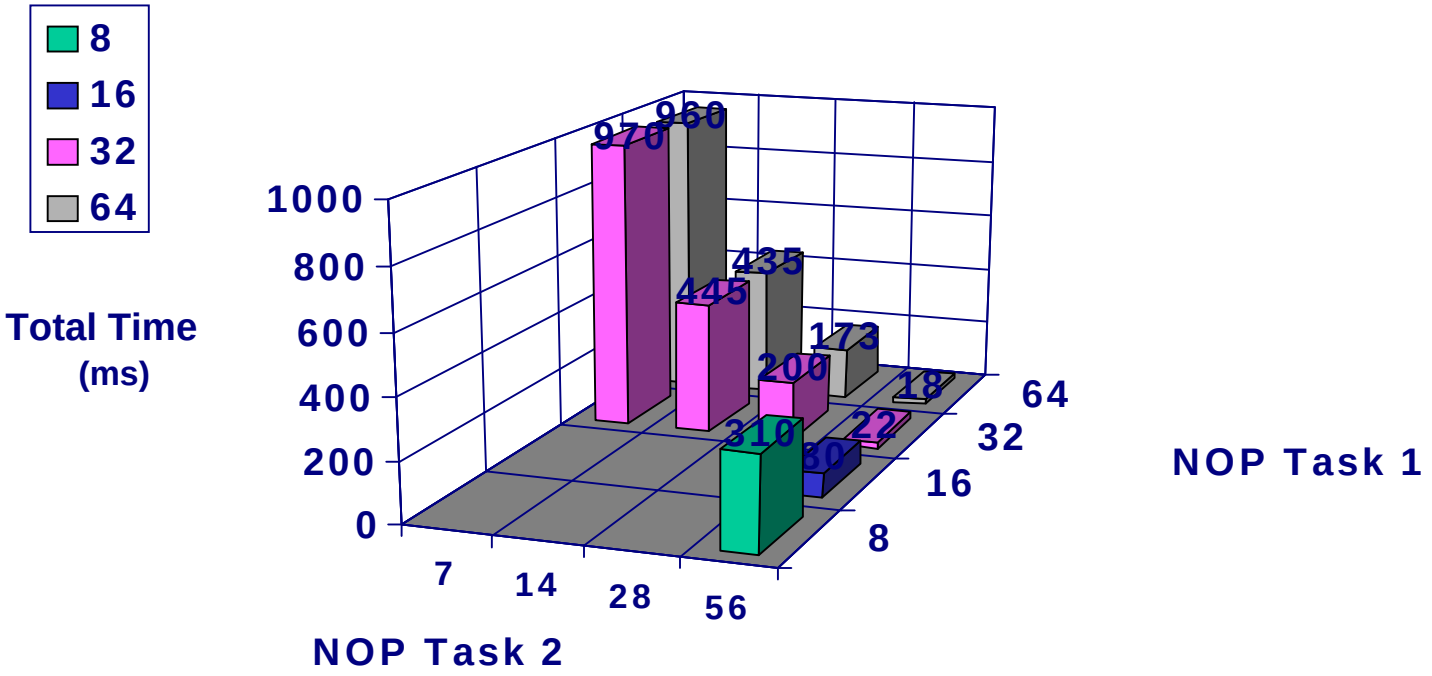
Step 2: Hard Weights



- *Each processor computes QR factorizations for 6 range cells*
- *Each processor works on 1 Doppler bin*

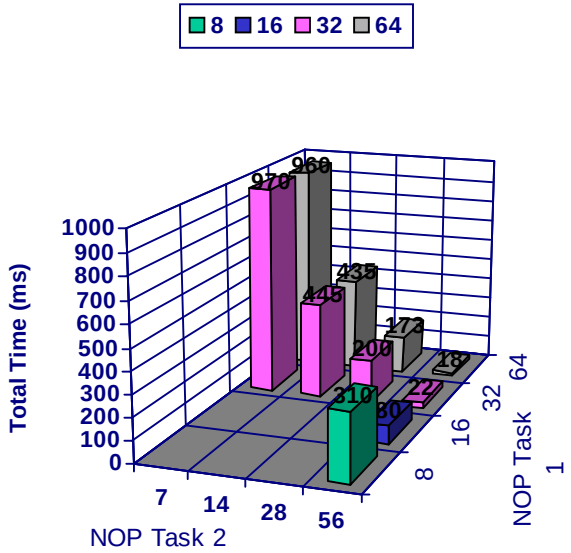
Speedup(7-56) = 7.9

Communication Scaling



Note: Communication scales tremendously to transfer data from one task to next because of parallelism

Communication Scaling



Note: Communication scales tremendously to transfer data from one task to next because of parallelism

Communication Scaling(task 1- task2)

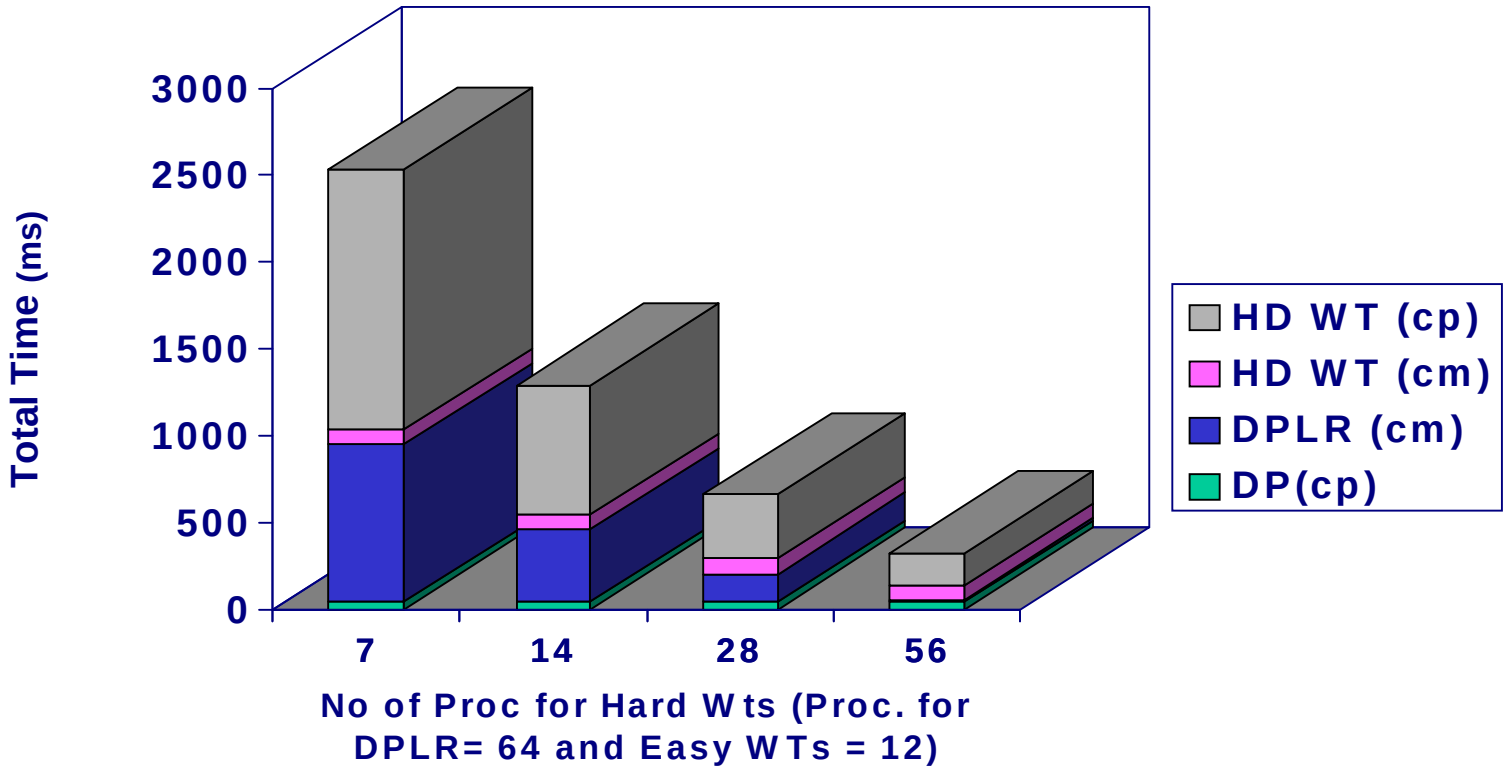
(time in ms)

NOP Task 2

NOP Task 1

(1)/(2)	7 S	R	14 S	R	28 S	R	56 S	R
8							70	177
16							36	43
32	923	46	424	20	191	10	19	3
64	912	47	415	20	163	10	11	6

Computation and Communication



Note: Communication scales tremendously to transfer data from one task to next because of parallelism

Summary

- ***Presented parallel pipelined design of STAP algorithm on HPC Systems***
- ***Modular and scaleable and portable software design***
- ***Three of Five step's initial implementations complete***
- ***Scaleable performance in both computation and communication***
- ***Specific evaluation on Intel Paragon***
- ***Further optimizations identified***