USC
UNIVERSITY
OF SOUTHERN
CALIFORNIA

Energy Efficient Adaptive Beamforming on Sensor Networks

Viktor K. Prasanna

Bhargava Gundala, Mitali Singh

Dept. of EE-Systems
University of Southern California
email: prasanna@usc.edu
http://ceng.usc.edu/~prasanna
http://pacman.usc.edu

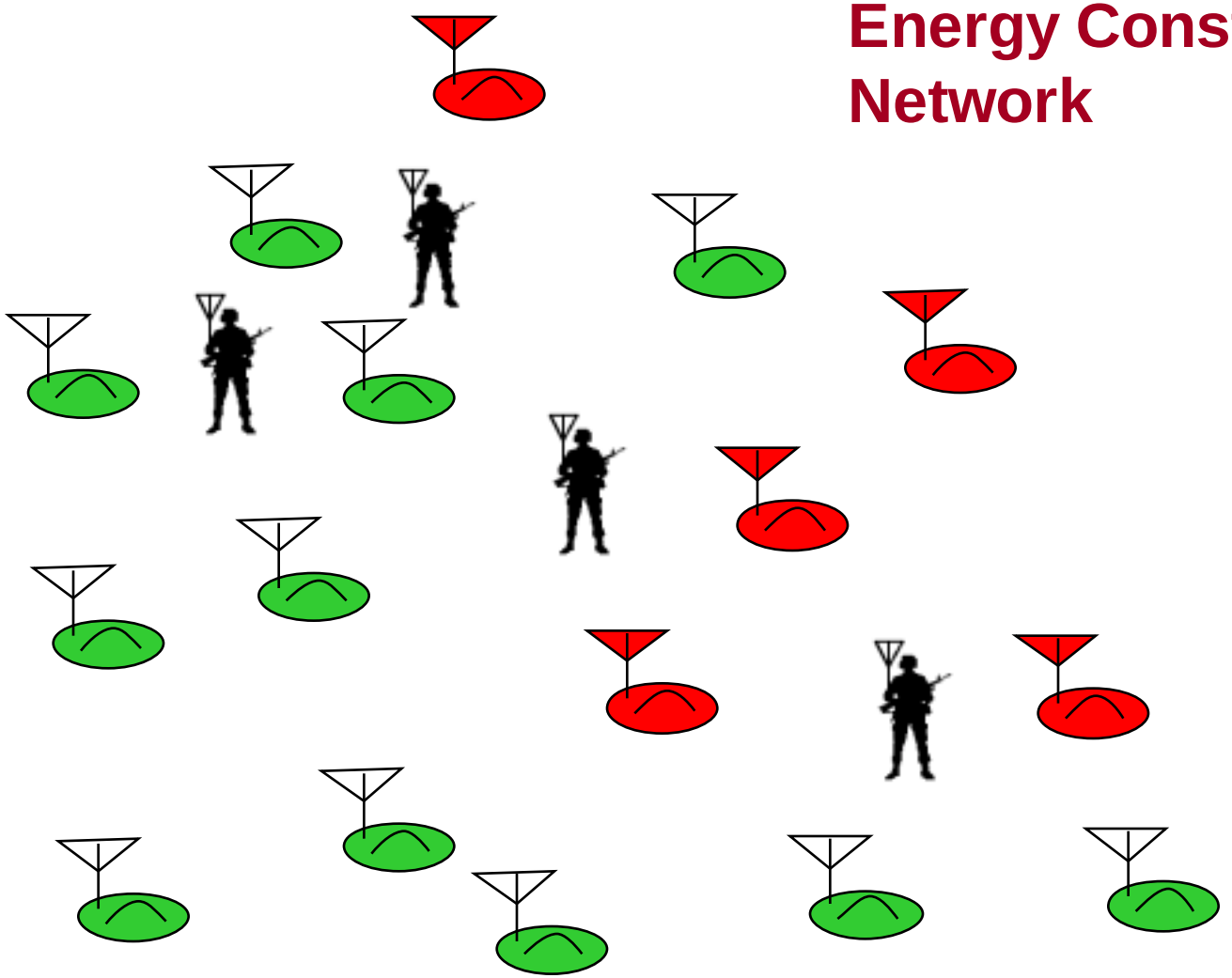
USC UNIVERSITY OF SOUTHERN CALIFORNIA

Outline

- Problem Definition
- Computational Characteristics
- Prior Solution
- Power Optimizations
  - Sensor Node Level
  - Inter Node Level
- Challenges/Discussion

Problem Scenario

Energy Constrained Network



Passive

Active

USC UNIVERSITY OF SOUTHERN CALIFORNIA

Beamforming

Def: The technique which spatially filters the signals received from an array of sensors and estimates the spatial features of the sources

Procedure:

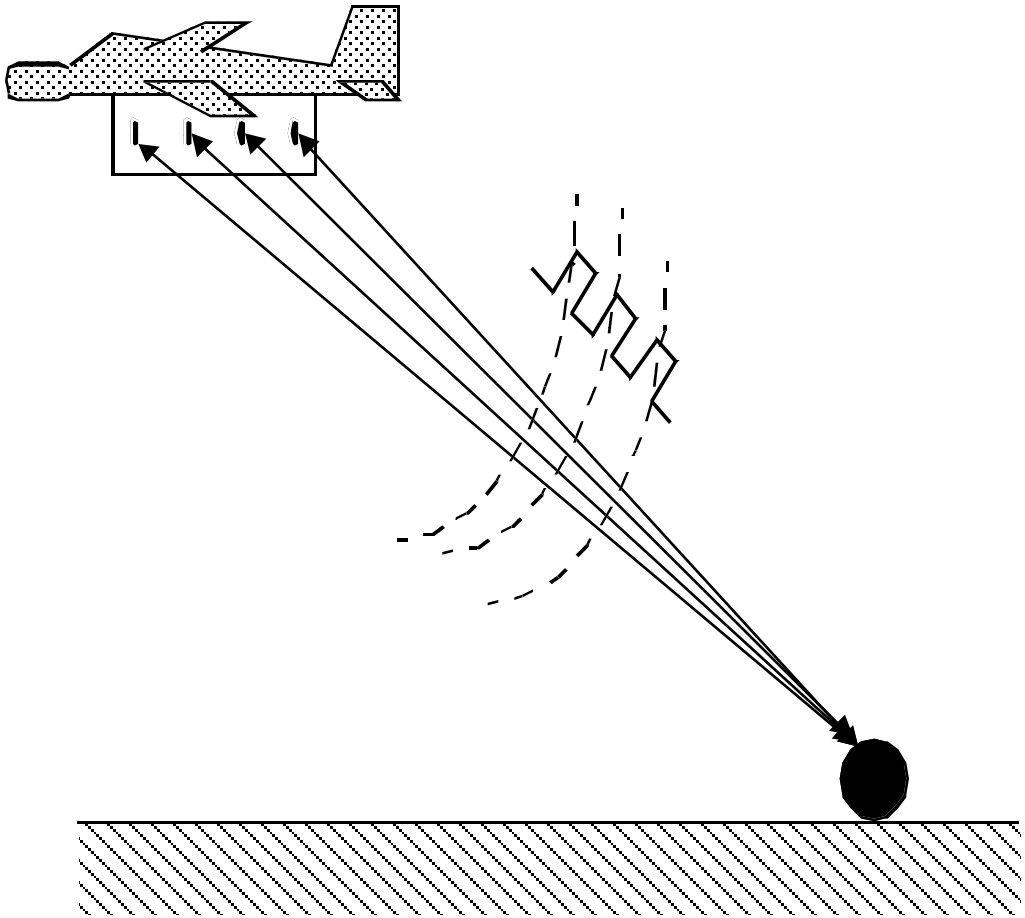
1. passively and repeatedly sample acoustic propagation wave field signals
2. input data, linearly combined with a weight matrix to form a sonar beam for a particular direction of look

Adaptive Sonar Beamforming:

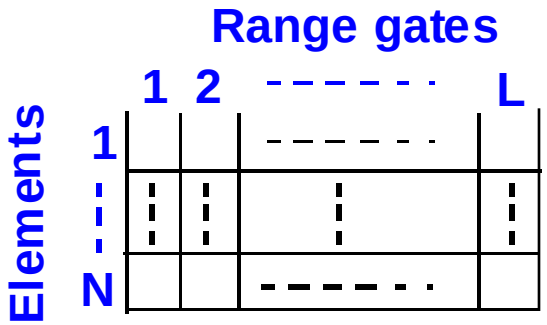
For High SNR and High resolution

Time changing signal and noise properties included in the derivation of weights, making them adapt accordingly

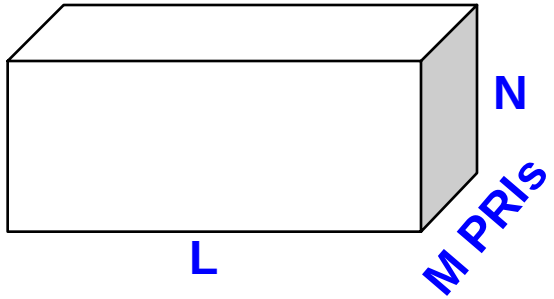
Space Time Adaptive Processing



Target Detection

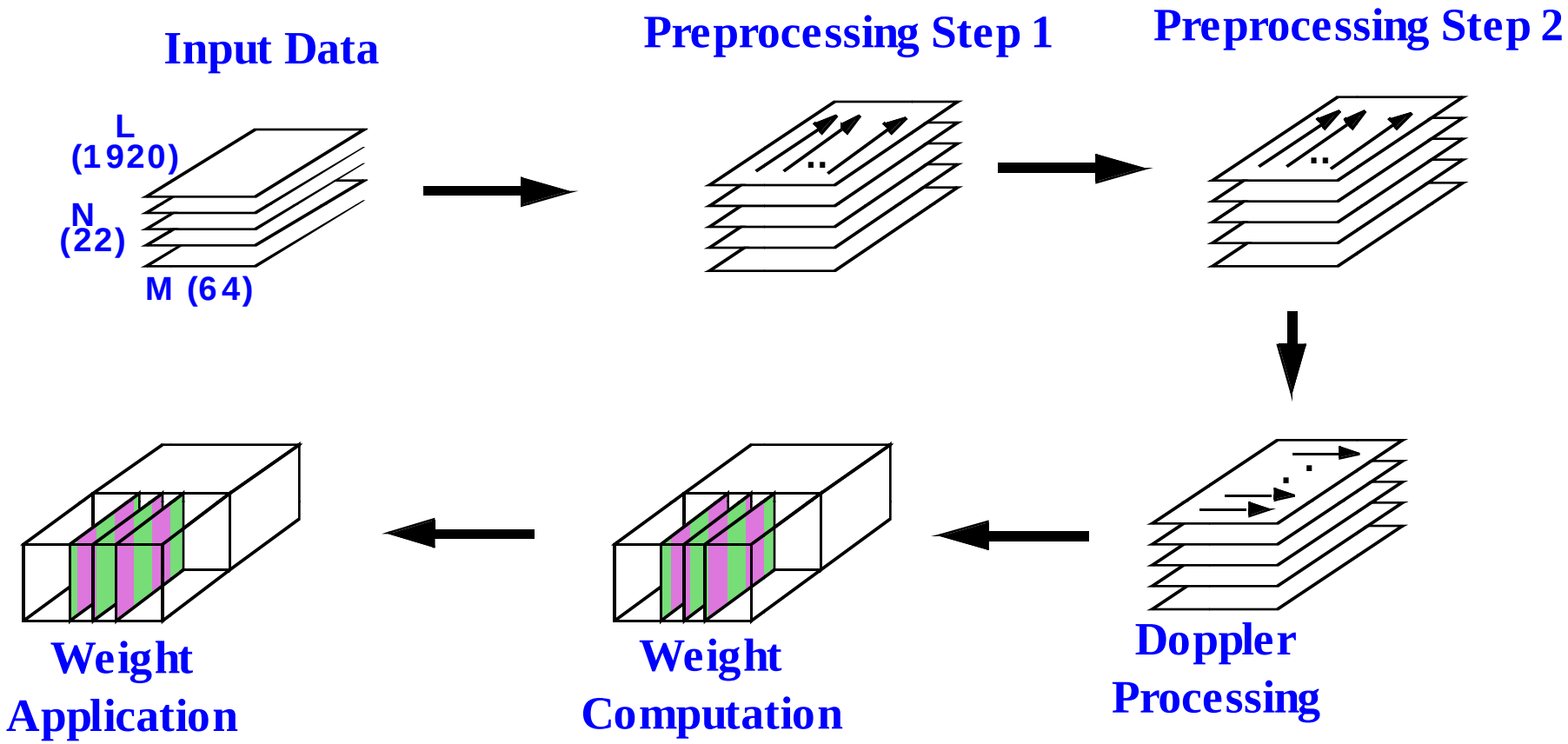


Pulse Repetition Interval



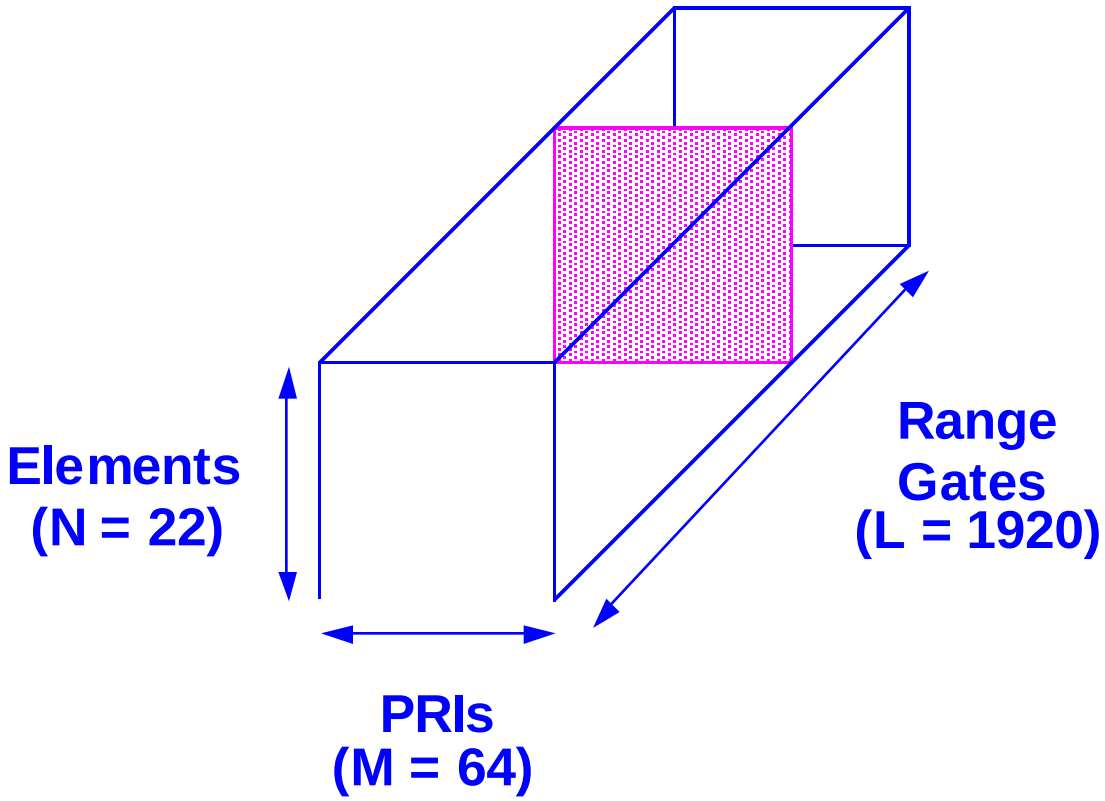
Each CPI
(Coherent Processing Interval)

MITRE RT_STAP Benchmark



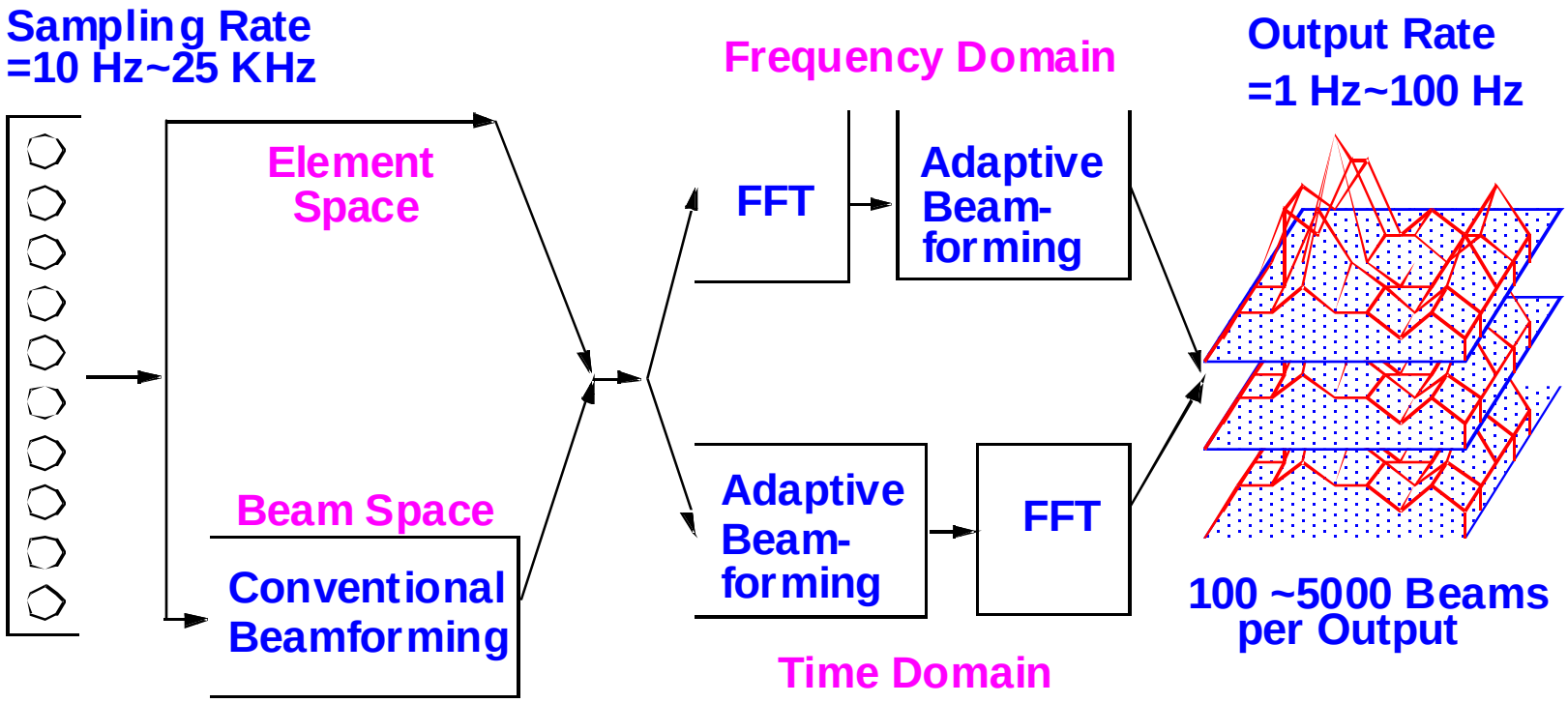
$T_{latency}$ = 161.25 msec & T_{period} = 32.25 msec

Input Data Cube



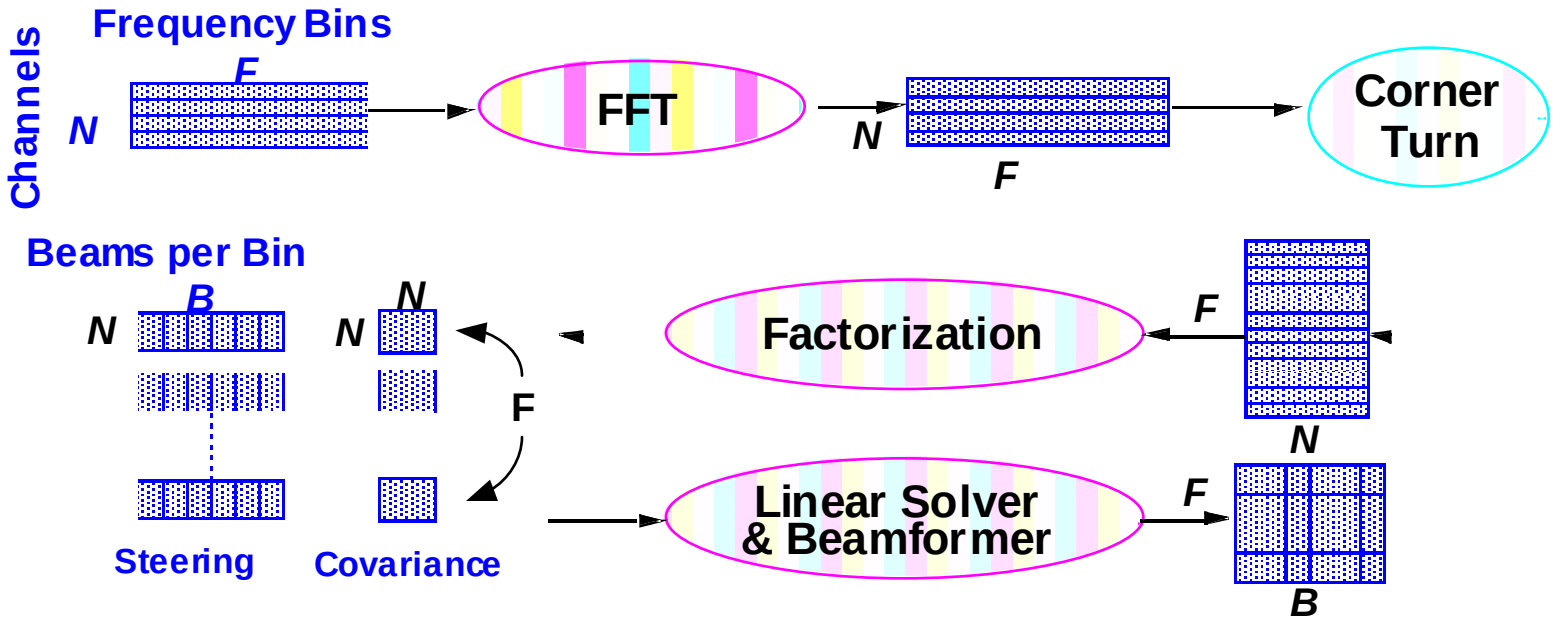
Sonar Signal Processing

Adaptive Beamforming

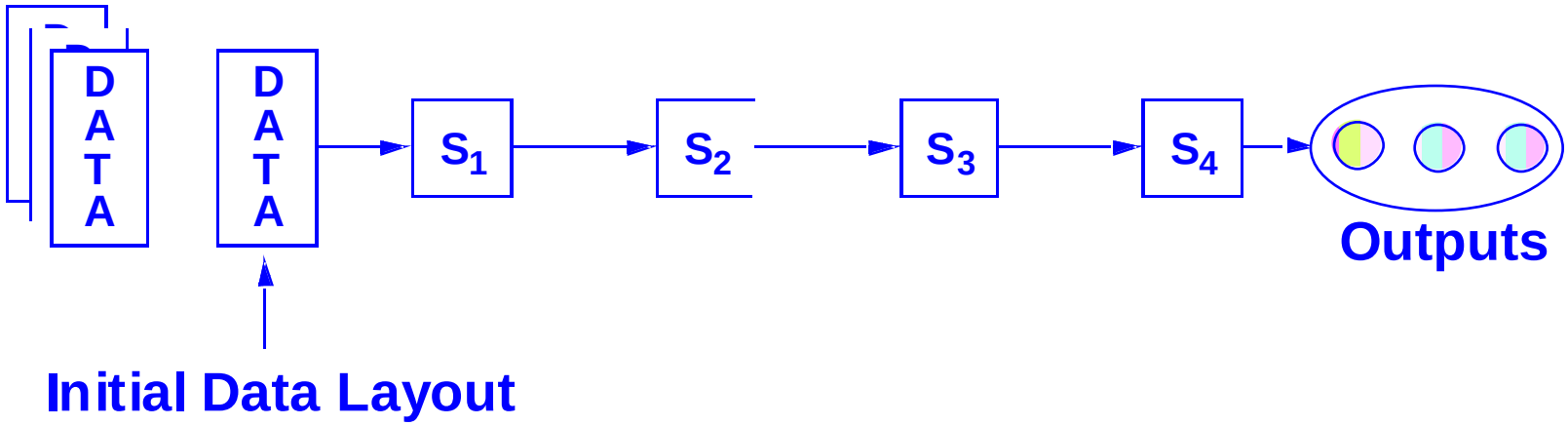


An Example Adaptive Beamformer

MVDR (Minimum Variance Distortionless Response)



Computational Characteristics



- Overall processing consists of sequence of subproblems
- Computational requirements are different for each subproblem
- Large amount of data is repeatedly processed in real-time
- Data access patterns change from subproblem to subproblem
- Throughput and latency performance requirements

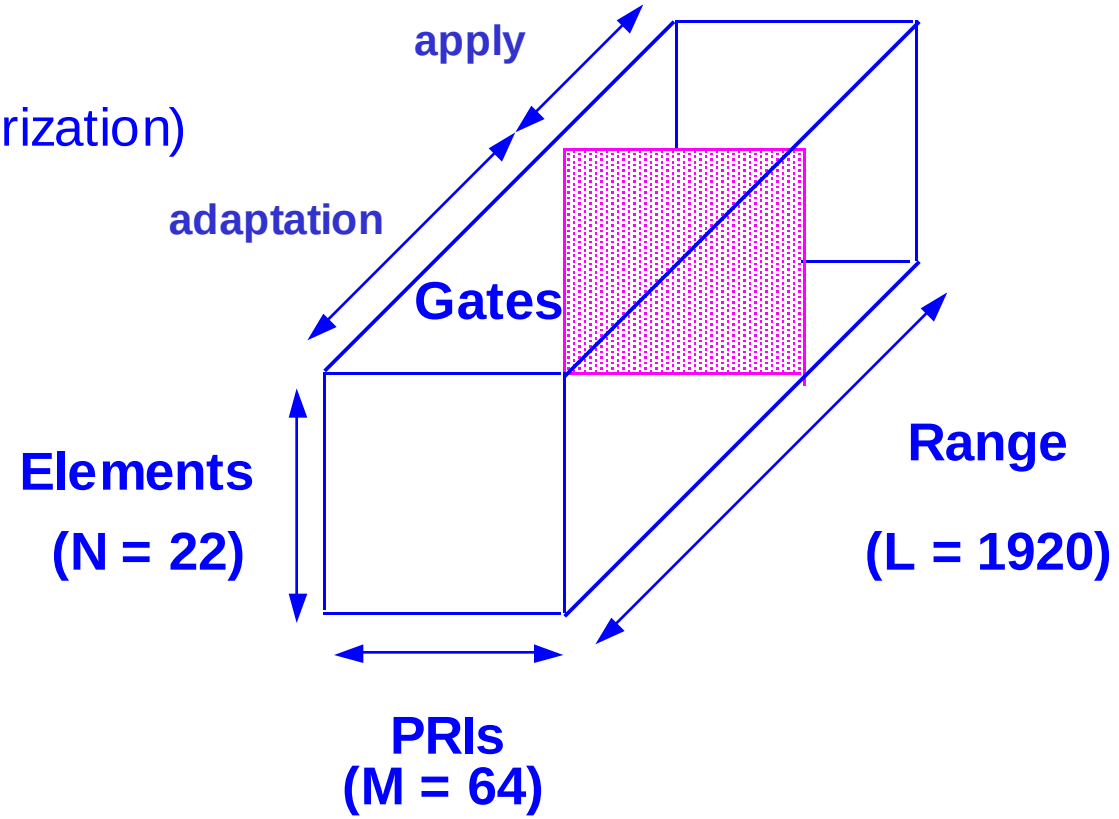
Adaptive Processing

Key Problems

- Doppler Processing (FFT)
- Weight Computation

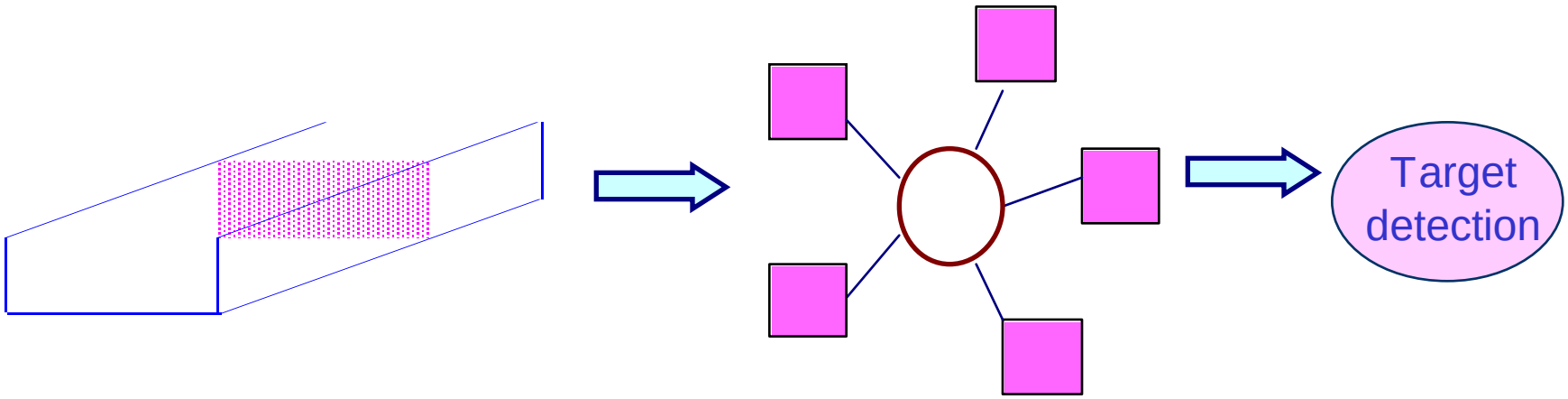
  (Co Variance matrix factorization)
- Weight Application

  (Matrix Vector Product)



Prior Solution

Architecture= tightly coupled collection of processors



High bandwidth, low latency network

Key Issue: Communication Cost

Coarse grain machines : Powerful processing nodes

-SP-2: Typical Configuration

- 640 Mflops/node
- 64 MB – 4 GB Memory
- 4.5 – 36.2 GB Internal Disk

- T3E: Typical Configuration

- 1200 Mflops/node (T3E- 1200)
- Local Memory Access Time: 87 ~ 253 nsec
- Global Memory Access Time: 1~2μsec (SHMEM)

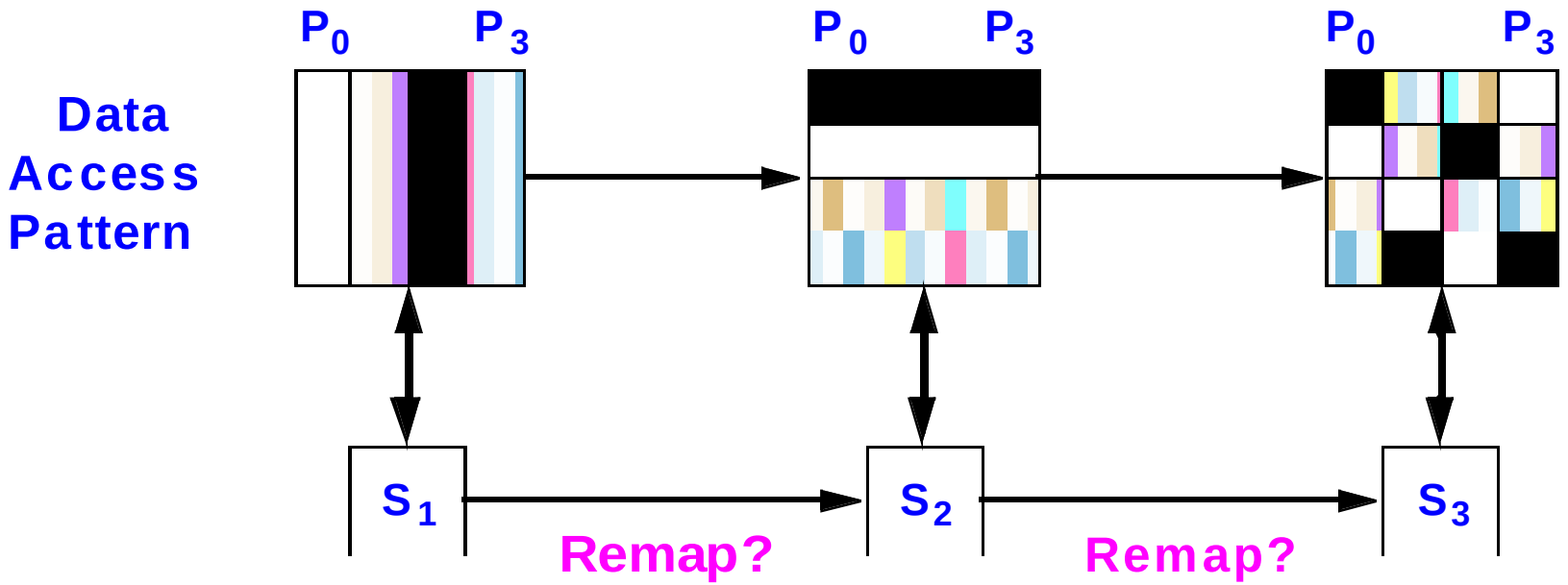
❑ Large software overhead for message transfer

- SP-2: ~39 μsec overhead/message using MPL/MPI

  ~ 9 nsec/byte/node transfer rate

- local memory access: 100's of nsec

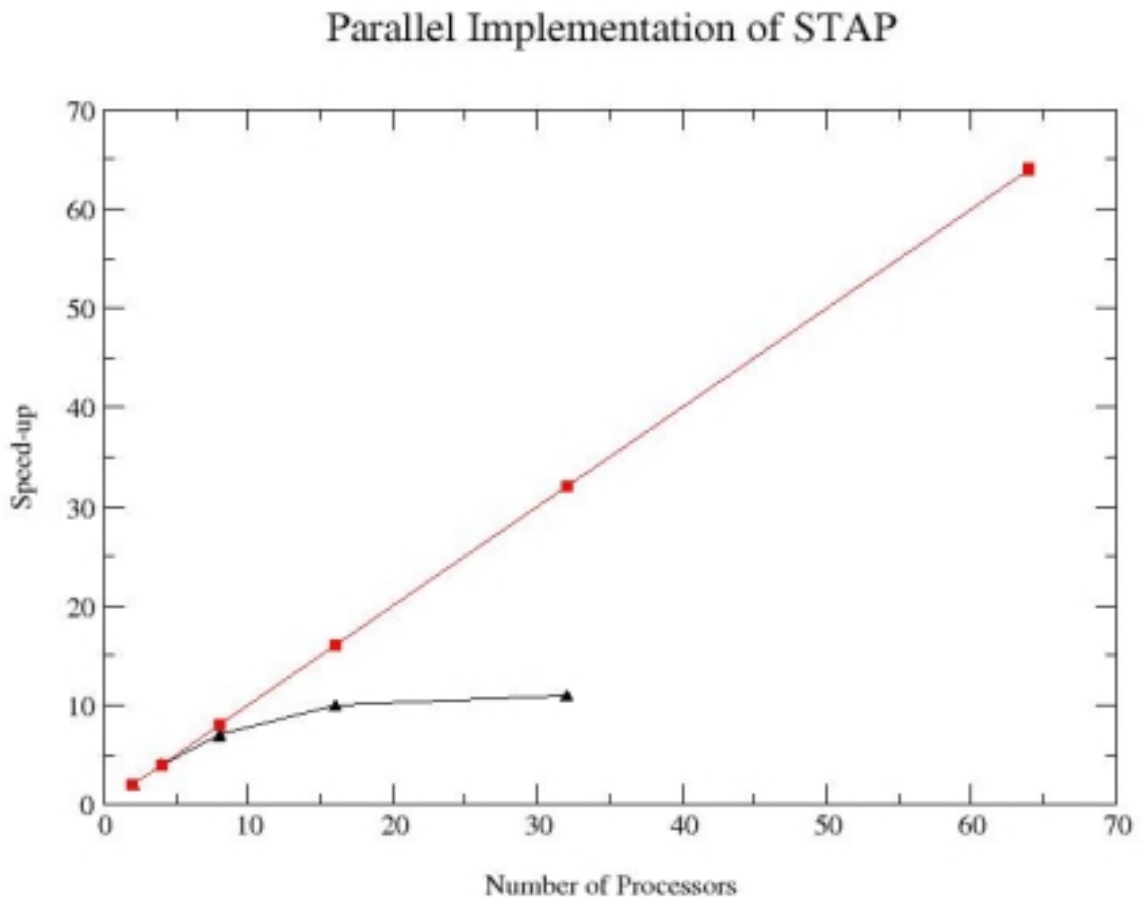
Key Idea- Data Remapping



Benefits of Remapping Must Exceed the Overhead

USC UNIVERSITY OF SOUTHERN CALIFORNIA

Impact of Data Remapping



Implementation performed on IBM SP-2 at MHPCC

Code developed using C, MPI and ESSL

USC UNIVERSITY OF SOUTHERN CALIFORNIA

Lessons learnt

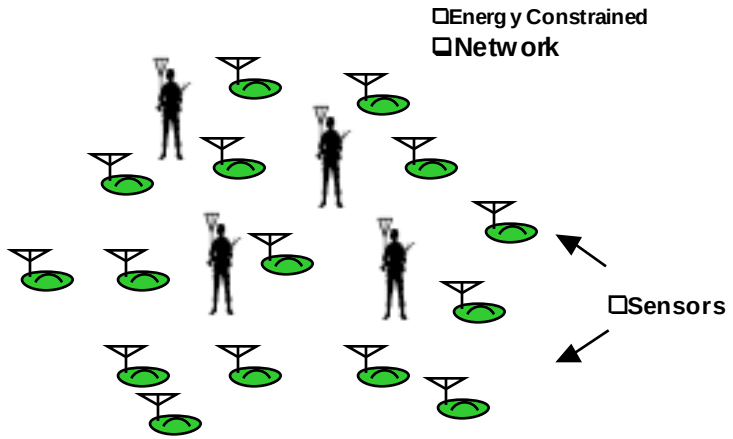
Objective : Adaptive beamforming on parallel machines

- Task level parallelism
- Minimize communication cost
- Data Remapping

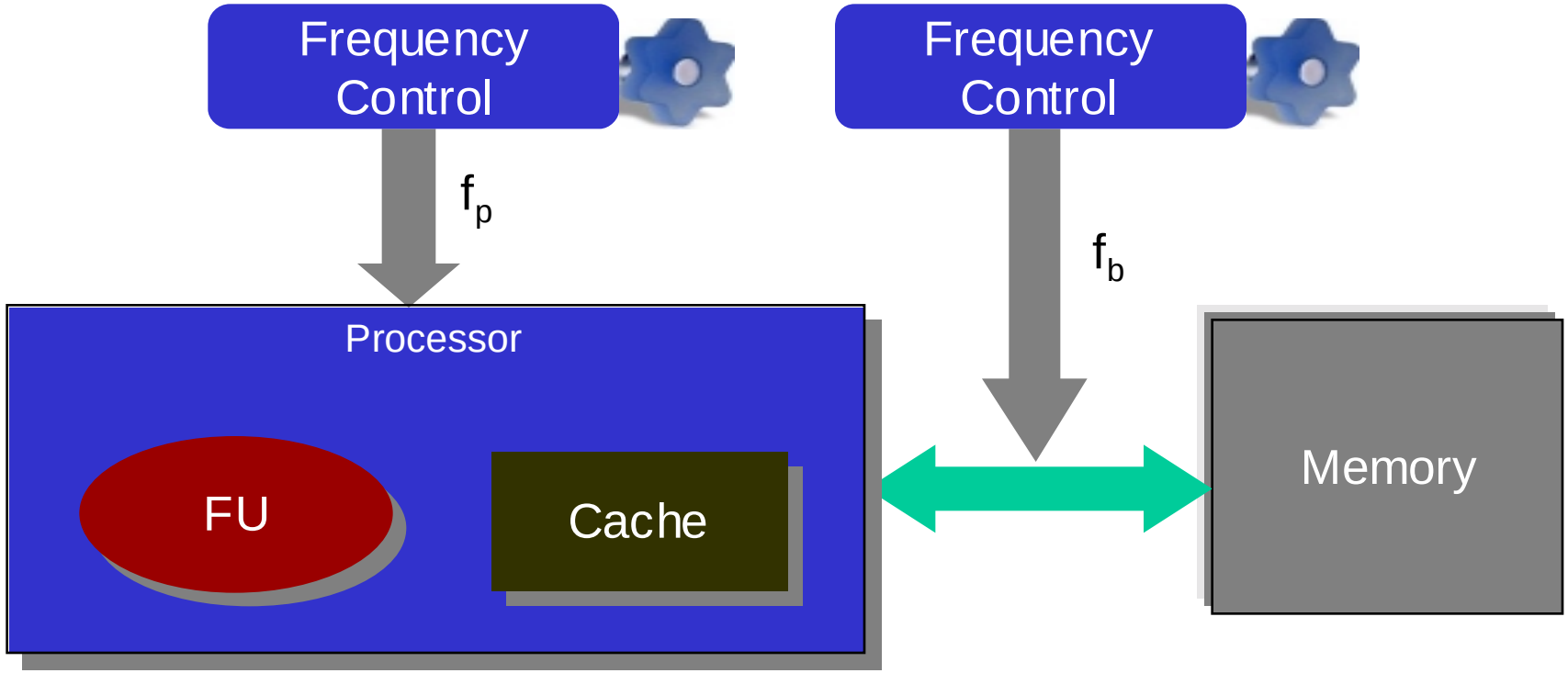
Energy Efficiency

Power is critical and must be conserved

- Reduce power dissipation at sensor node level
  - energy efficient algorithms
- Decrease power dissipation at inter-node level
  - Optimize on communication cost between sensors



Power Model for a Processing Element



$$\text{Power}_{\text{Total}} = \text{Power}_{\text{Processor}} + \text{Power}_{\text{Data bus}} + \text{Power}_{\text{Memory}}$$

$$\text{Power}_{\text{unit}} = \text{Power}_{\text{Dynamic}} + \text{Power}_{\text{Static}}$$

$$= 0.5f(n)CV^2f_{\text{Active}} + VI_{\text{Leakage}}$$

$$F_{max} \propto (V-V_t)/V$$

Reduce Processor-Memory Data Traffic

Instructions for Memory access consume lot of power

Instruction (Intel 486DX2)	Energy (10^{-8} Joules)
MOV DX BX	2.49
MOV DX [BX]	3.53
MOV [BX] DX	4.30

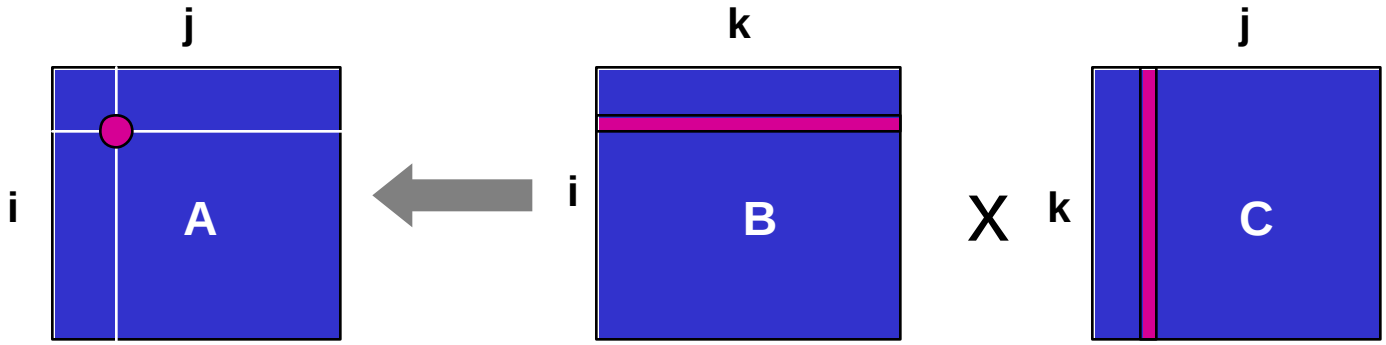
Reduce # of memory accesses

- reduce cache misses
- high data reuse in cache
- use registers

Reduce power consumed on the data bus

Example: Matrix Multiplication

Cache size =n



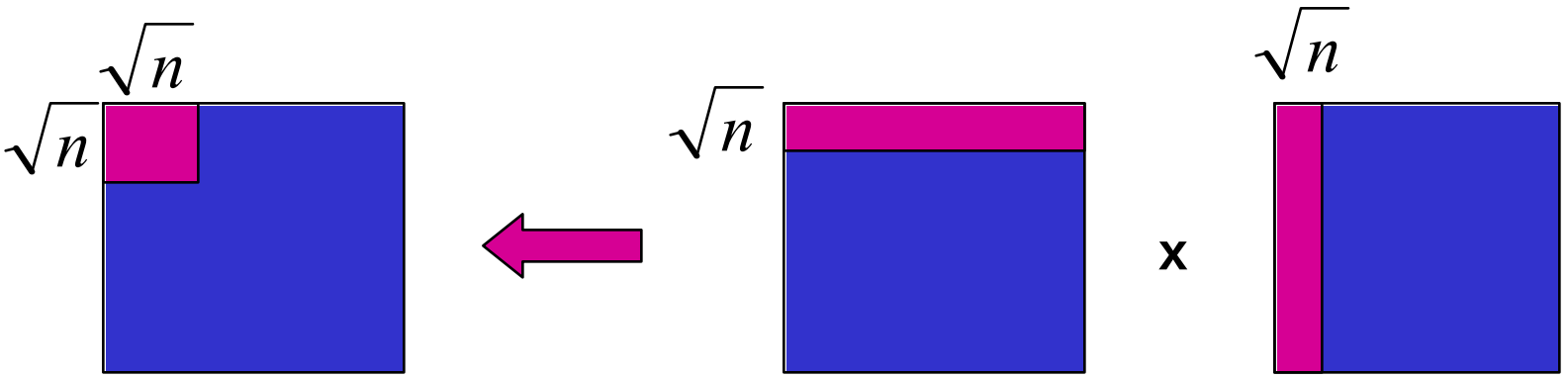

```
Do i = 0 ;
    Do j = 0 ;
        A[i,j] ← 0 ;
    Do k = 0 ;
        A[i, j] ← A[i,j] + B[i,k] x C[k,j] ;
        k++; j++; i++ ;
```

Energy = $\alpha n^3 + \beta(n+n^2)n + \gamma(3n^2) \approx (\alpha + \beta)n^3$

Time = n^3 + lower order terms

Optimization I: Reduce Bus Traffic

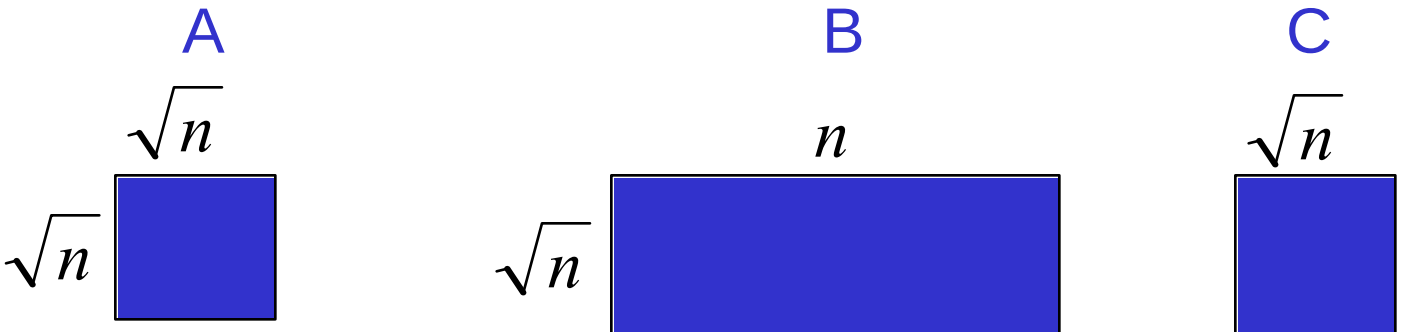
Block Matrix Multiply

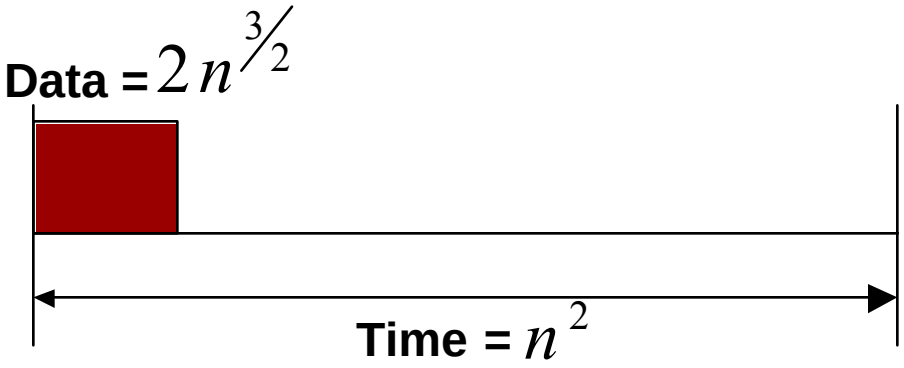


Energy = $\alpha n^3 + 2\beta(n.n^{1/2})n + \gamma(3n^2)$

Time = n^3 + lower order terms

Optimization II: Reduce Peak Bus Bandwidth

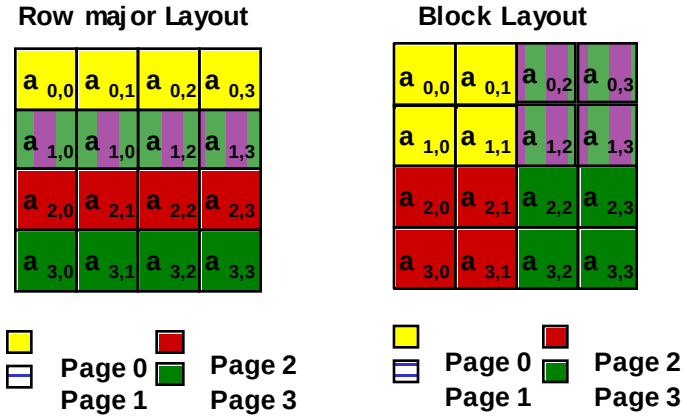




Bus Data Rate $\propto 1/\sqrt{n}$ Processor Rate!

Optimization III: Application directed Data Layouts

- **Applications have different data access patterns**
  - Matrices accessed by rows, columns, diagonals, sub-squares
  - Tree structures accessed along paths, sub-trees
- **"Naive" data layouts degrade performance**
  - Large working sets cause capacity misses
  - Improper alignment in memory causes conflict misses



USC UNIVERSITY OF SOUTHERN CALIFORNIA

Cache Friendly Algorithms

Cache friendly

- High data reuse
- Low cache pollution
- Regular access patterns Data layouts
  - Static data layouts (Matrix Multiply)
  - Dynamic data layouts (FFT)

USC UNIVERSITY OF SOUTHERN CALIFORNIA

Fast Fourier Transform

DFT: Cooley-Tukey Algorithm

- Compute DFT of size $N = N_1 * N_2$
  - Step1: compute N_2 DFTs of size N_1
  - Step2: multiply twiddle factors
  - Step3: compute N_1 DFTs of size N_2
- Divide and conquer recursively

Current Approach

- MIT FFTW
  - Determine optimal factorization
  - Perform low level optimizations for kernels
  - Construct larger size FFTs from kernels

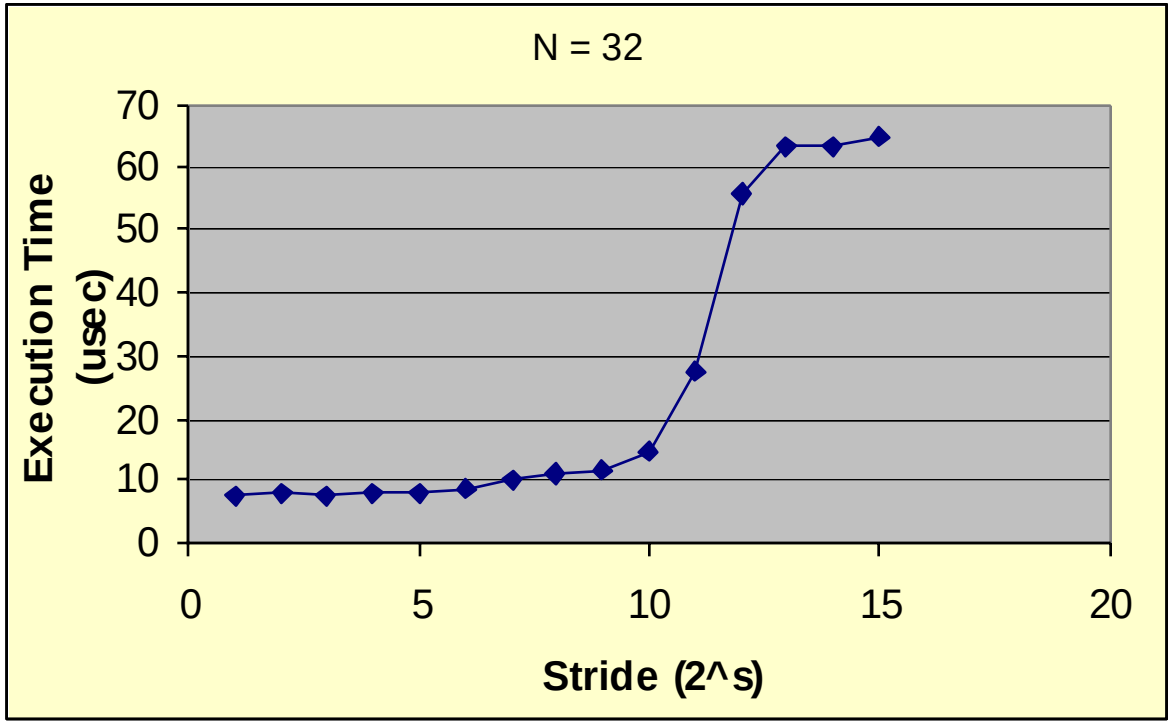
- Key Assumption
  - All DFTs of same size have same execution time

Problem with Current Approach

All N-point DFTs do not have the same cost!

- different data access patterns with various strides
- stride affects execution time

32-point FFT with Strided Access - Experimental Results



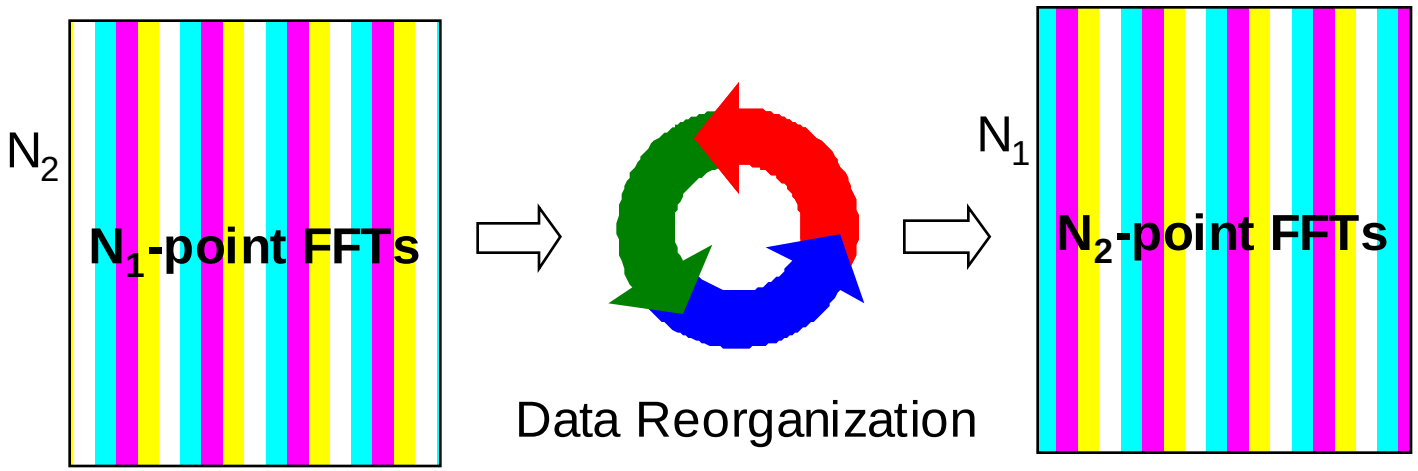
Sun Ultra 1: 167MHz, L2 Cache = 512 KB = 32 K points

Our Approach

Reorganize input data layout to change non-unit stride to unit stride

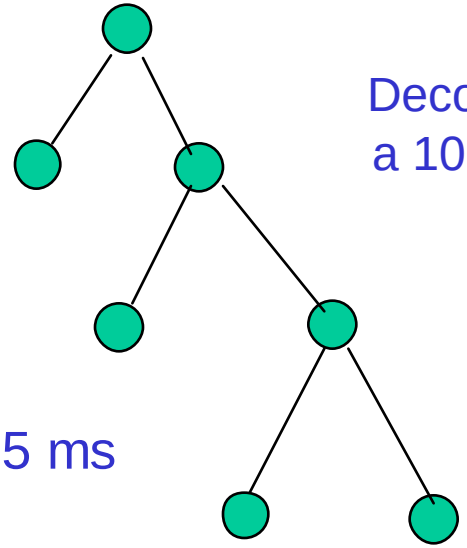
Dynamic Data Layout

Perform data reorganization during computation



Example

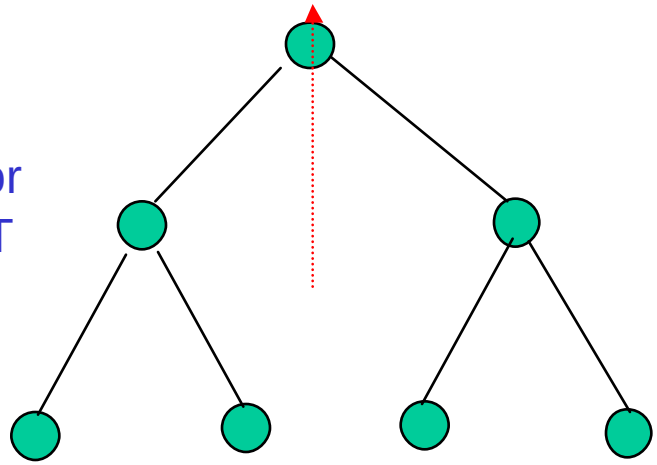
FFTW



1611.125 ms

Decomposition trees for
a 1024*1024 point FFT

USC approach



1039.6496 ms

54.96% improvement over

state-of-the-art FFTW package on DEC Alpha

Other Techniques for Node Level Power Optimizations ?

- Voltage frequency scaling

  $f_{max} \alpha (V-V_t)/V$

- Power management (idle/sleep/active states)

- Reduce precision

- Clock Gating

Instruction (Fujitsu Sparc'934)	Energy (10^{-8} Joules)
OR	3.26
MUL	3.26

USC
UNIVERSITY
OF SOUTHERN
CALIFORNIA

Current Work

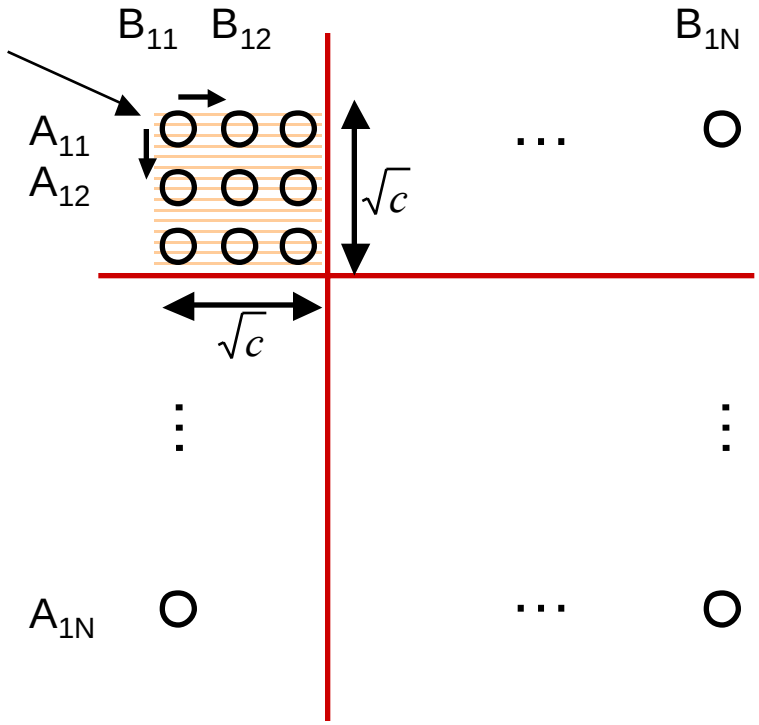
- Development and Verification of techniques proposed for power optimization
- Existing simulators
  - Simple Power(based on Simple Scalar architecture)
  - Joule Track (Code Length Limitations)
- Board level Power Measurements
  - Brutus Evaluation Board (SA-1100)
- Build a functional level power simulation
  - Fast with acceptable level of accuracy.
  - Develop a multiprocessor power model

Space Time Representation

Compute results in each block

- Schedule blocks row-major
  - $\frac{N^2}{c}$ steps
  - Data per step $\propto N\sqrt{c}$
  - Operations per step $\propto Nc$
  - Data reuse per step $\propto \sqrt{c}$
  - Total traffic $\propto \frac{N^2}{c} * N\sqrt{c} = \frac{N^3}{\sqrt{c}}$

A ⊗ B for N x N matrices



O = computation for result (i,j)
c = cache size

USC
UNIVERSITY
OF SOUTHERN
CALIFORNIA

Theorem

Unidirectional Space-Time representation leads to cache friendly algorithms

=> Energy Efficient Algorithms

Network level Energy Optimization

- Computation cost is much lower than communication cost
- Radio interface consumes a large amount of power

POWER Consumed	WINS sensor Node
Transmission(100m)	600mw (at 100kbits/sec)
Reception	300mw
Processor (SA1100)	250MIPS/watt

- Energy to transfer 32 bits over 100m in WINS sensor node
  =((600 +300)mw ÷ 100kbits/s) x 32 = **288 x 10^{-6} Joules**

- Energy to execute a 32 bit instruction using SA1100 processor
  = 1 ÷ 250 MIPS/watt = **0.004 x 10^{-6} Joules**

- Additional overhead for bits added for error correction

- Retransmissions are frequent due to unreliable links(e.g.wireless)

USC UNIVERSITY OF SOUTHERN CALIFORNIA

Reduce Communication Cost

- Exploit data redundancy to reduce data traffic
- Improve locality of computation while assigning subtasks to node
- Communication limited to closely placed nodes
  - Larger distance requires higher transmission power
  - Reduces reliability of link

Network Level Power Optimization Issues

- Topology of network is unknown
  - Estimation of Communication cost
  - Task allocation
- Broadcast Communication Model
- Need: Framework for Energy Efficient Computation in Adhoc Networks