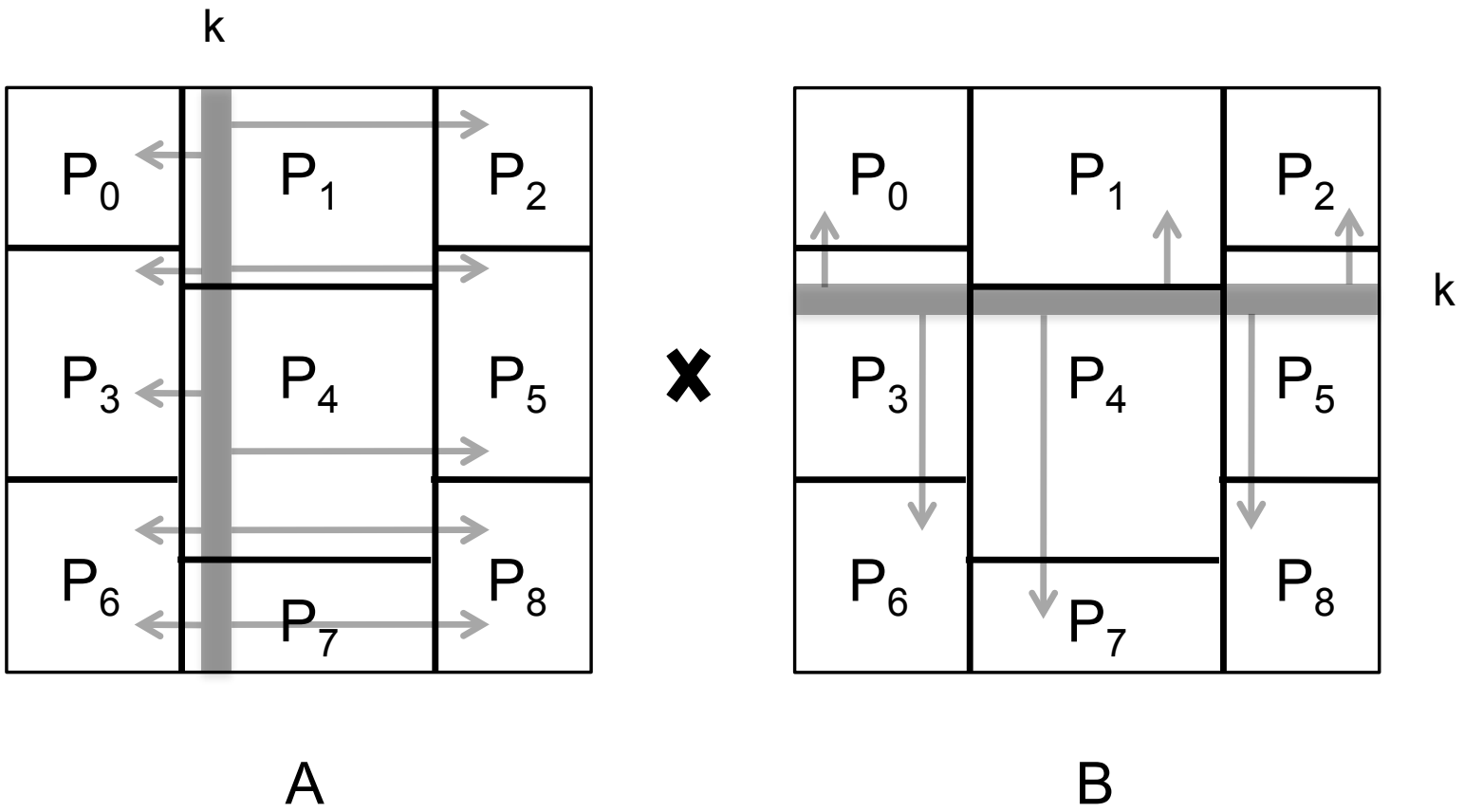
The Optimal Partitioning Shapes for Parallel Matrix Computation on Two and Three Heterogeneous Processors

Ashley DeFlumere, Alexey Lastovetsky

Heterogeneous Computing Laboratory, University College Dublin, Ireland

Matrix Multiplication - SUMMA



SUMMA algorithm, broadcast columns of A and rows of B, shown using column based data partition for 9 heterogeneous processors

Data Partition Shapes

Data Partitioning

- Traditional data partitioning for matrix multiplications generally assigns a rectangular submartrix to each processor
- Clearly optimal for homogeneous systems
- Current data partitioning for heterogeneous systems has been adapted from homogeneous algorithms
- What is the optimal shape for heterogeneous systems? Could it be non-rectangular?

Modelling MMM - Assumptions

Define the problem,

Computation

- Each Matrix A, B, C is square and identically partitioned
- Each Processor has a defined computation speed, expressed as a ratio $P_r : 1$ (2 processor) or $P_r : R_r : 1$ (3 processor), and overall speed $T = P_r + R_r + 1$
- Modelled by kij algorithm (like SUMMA)

Modelling MMM - Assumptions

Define the problem,

Computation

- Each Matrix A, B, C is square and identically partitioned
- Each Processor has a defined computation speed, expressed as a ratio $P_r : 1$ (2 processor) or $P_r : R_r : 1$ (3 processor), and overall speed $T = P_r + R_r + 1$
- Modelled by kij algorithm (like SUMMA)

Communication

- Modelled by Hockney, $\alpha + \beta \times M$
- Each Processor communicates with other processors, and all links are of the same speed

MMM Algorithm Description

Execution Time: Communication and Computation

- **Serial Communication with Barrier:** All serial communication first, then computation
- **Parallel Communication with Barrier:** All parallel communication first, then computation
- **Serial Communication with Overlap:** Serial communication and any computation not requiring communication first, then remaining computation
- **Parallel Communication with Overlap:** Parallel communication and any computation not requiring communication first, then remaining computation
- **Parallel Interleaving Overlap:** Communication and computation overlapped in k steps (compute k, send $k+1$)

Note for each algorithm, decreasing the volume of communication also decreases (or leaves unchanged) the execution time

Searching for Candidate Partition Shapes

Motivation: We believe that optimal shapes should be condensed, *i.e.* not random, arbitrary arrangements of elements
Goal: Find a small number of shapes, candidates, which no arrangement of elements can be superior to

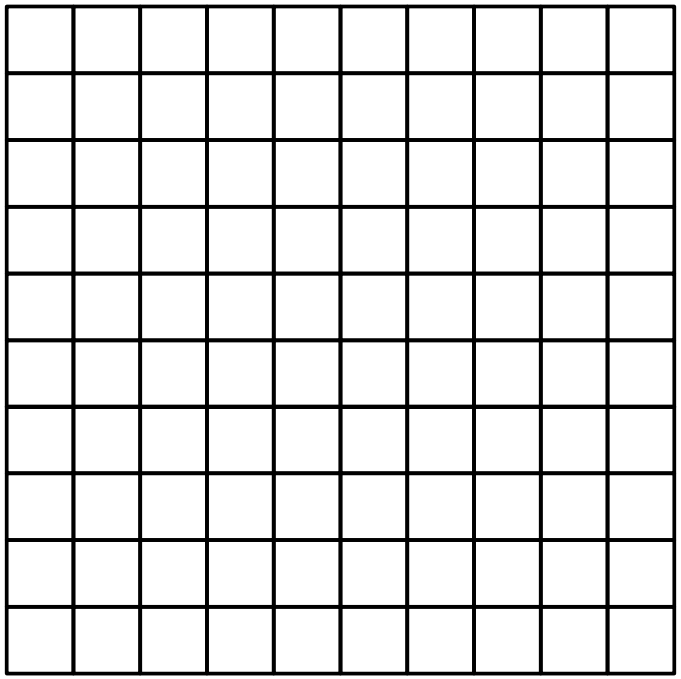
Push Technique

- Act on elements of a single processor, Q, in a single row or column, k
- Re-assign elements of Q into rows and columns other than k
- Follow rules in reassignment which guarantee lower or same total volume of communication

Push in Two Processor System

Two Processor Example

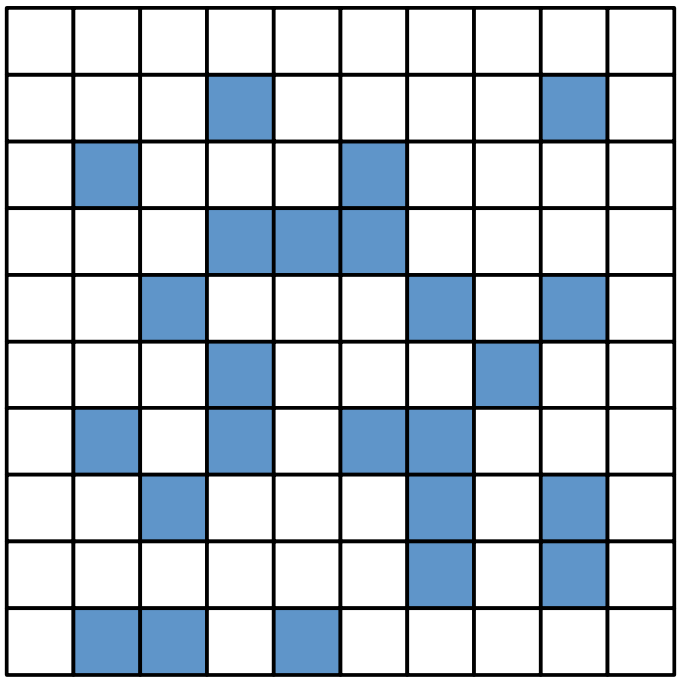
Push elements of slower processor, incrementally improving the volume of communication with each step



Push in Two Processor System

Two Processor Example

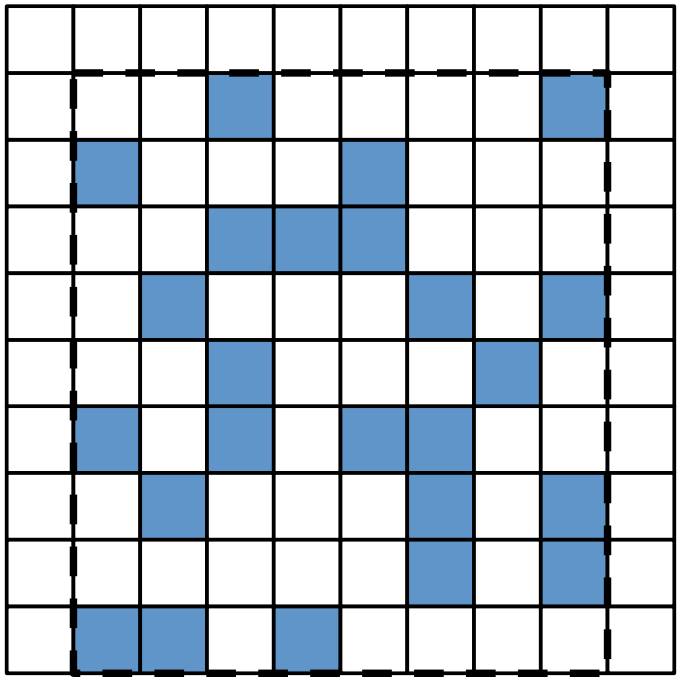
Push elements of slower processor, incrementally improving the volume of communication with each step



Push in Two Processor System

Two Processor Example

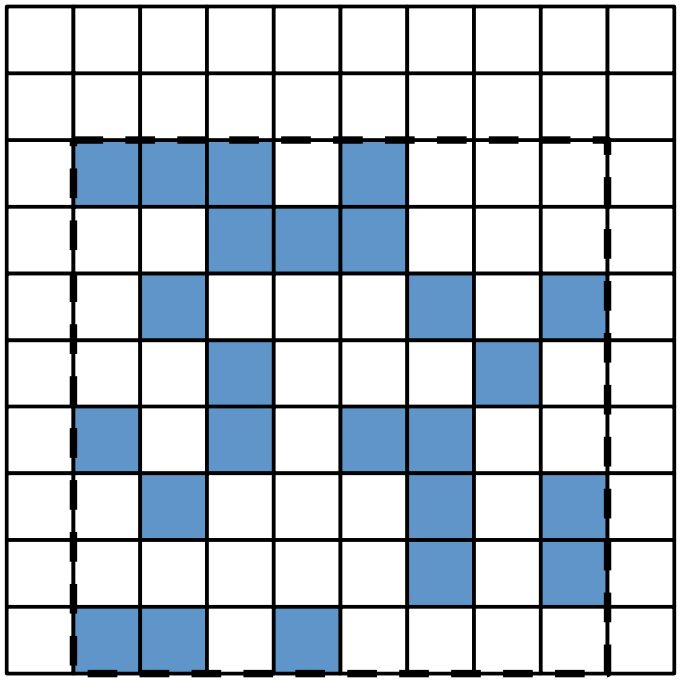
Push elements of slower processor, incrementally improving the volume of communication with each step



Push in Two Processor System

Two Processor Example

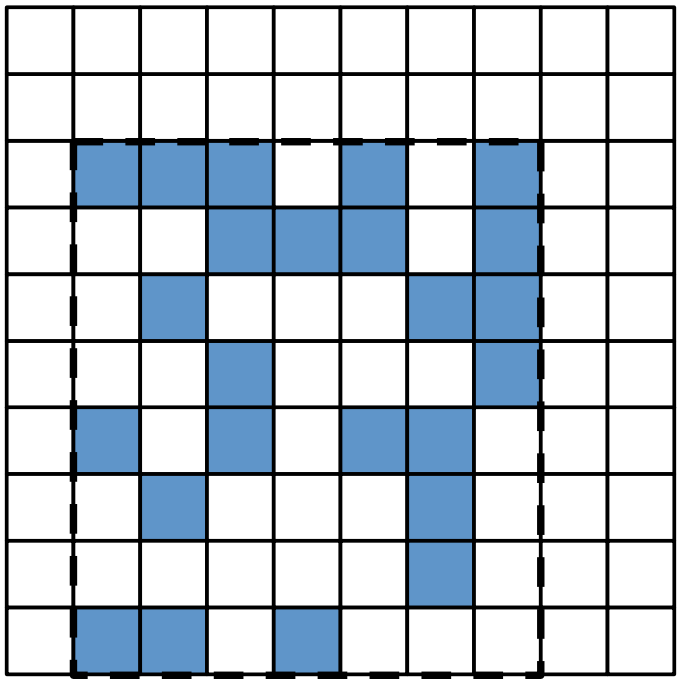
Push elements of slower processor, incrementally improving the volume of communication with each step



Push in Two Processor System

Two Processor Example

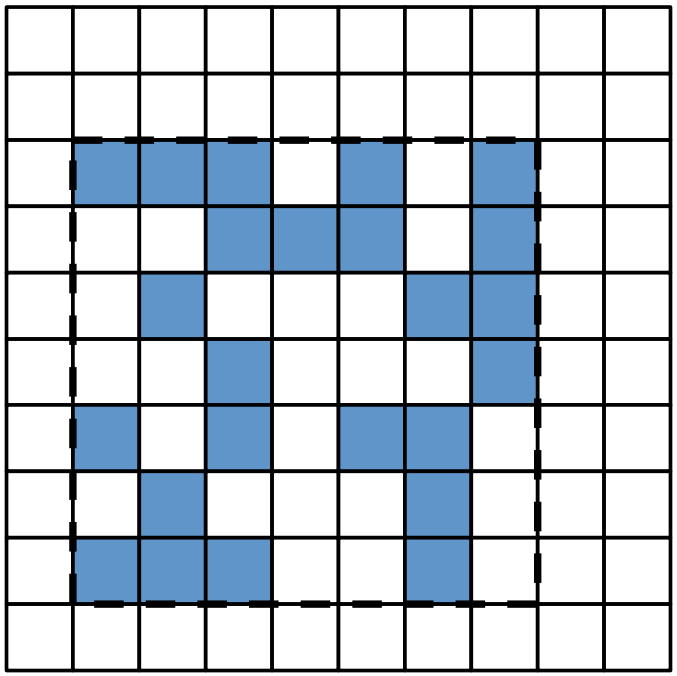
Push elements of slower processor, incrementally improving the volume of communication with each step



Push in Two Processor System

Two Processor Example

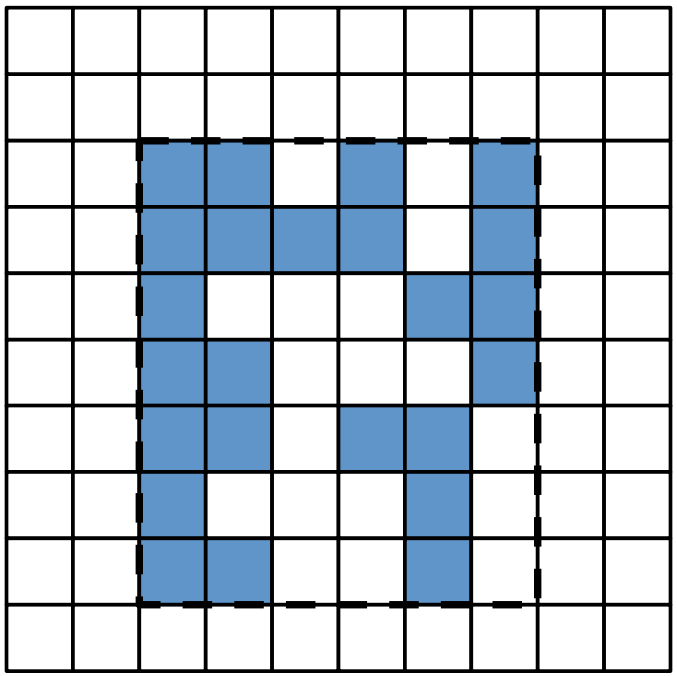
Push elements of slower processor, incrementally improving the volume of communication with each step



Push in Two Processor System

Two Processor Example

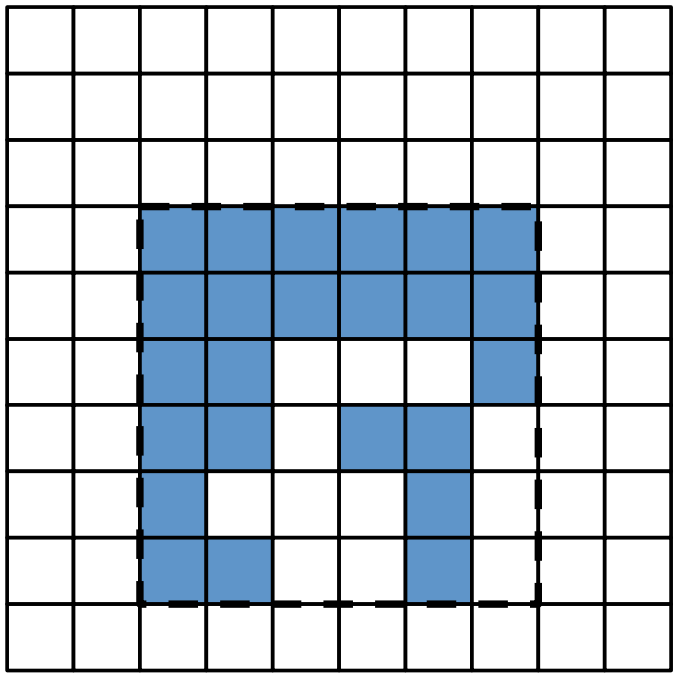
Push elements of slower processor, incrementally improving the volume of communication with each step



Push in Two Processor System

Two Processor Example

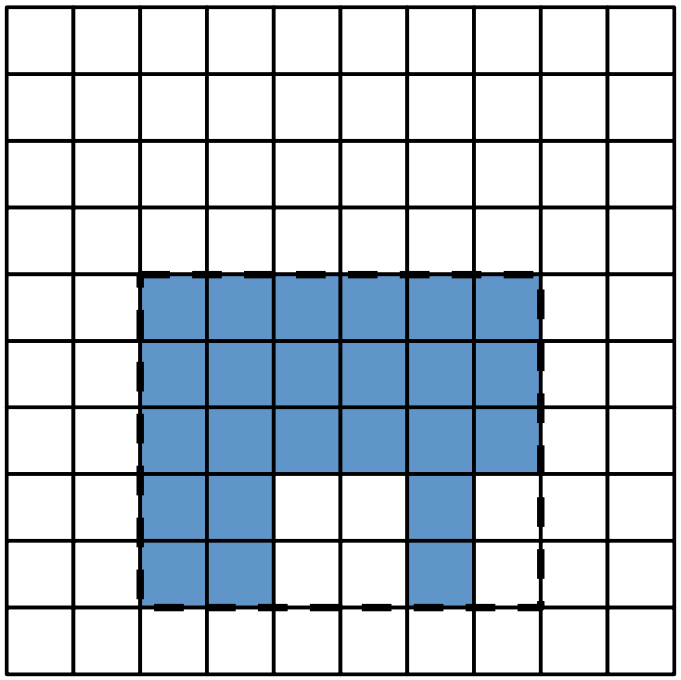
Push elements of slower processor, incrementally improving the volume of communication with each step



Push in Two Processor System

Two Processor Example

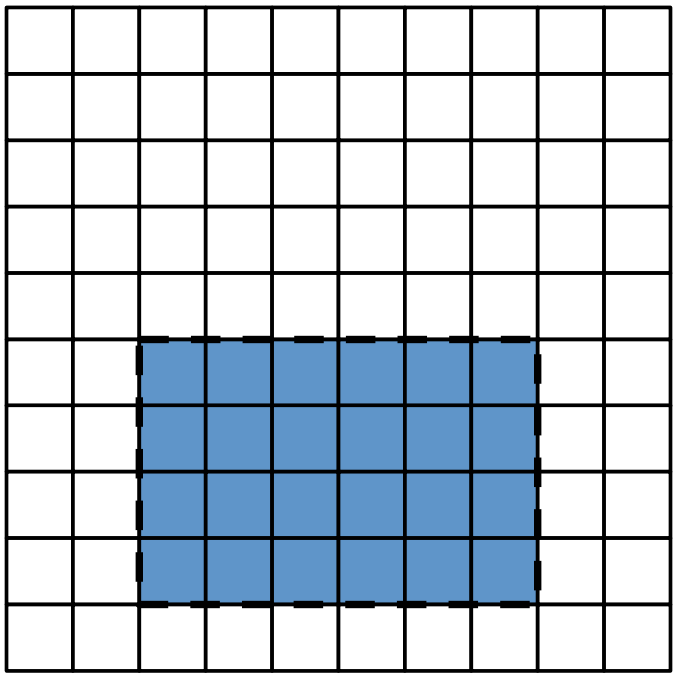
Push elements of slower processor, incrementally improving the volume of communication with each step



Push in Two Processor System

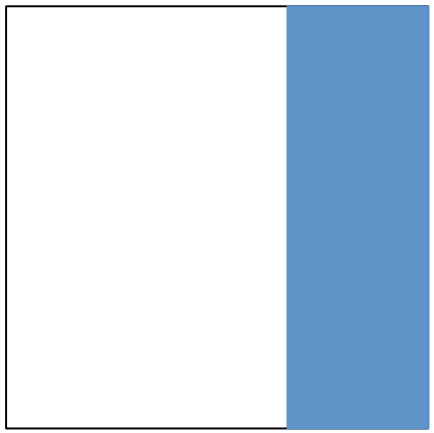
Two Processor Example

Push elements of slower processor, incrementally improving the volume of communication with each step

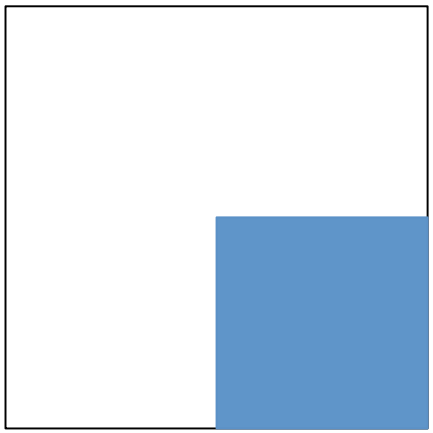


Optimal Candidates for Two Processors

Proved analytically that no shape is superior to the Straight Line and the Square Corner, these are the optimal candidates

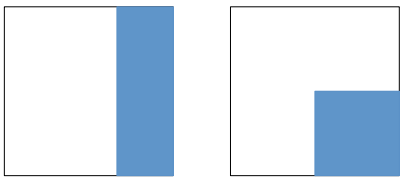


Straight Line



Square Corner

Optimal Shape for Two Processors



Straight Line
Square Corner

Analyse using 5 MMM algorithms

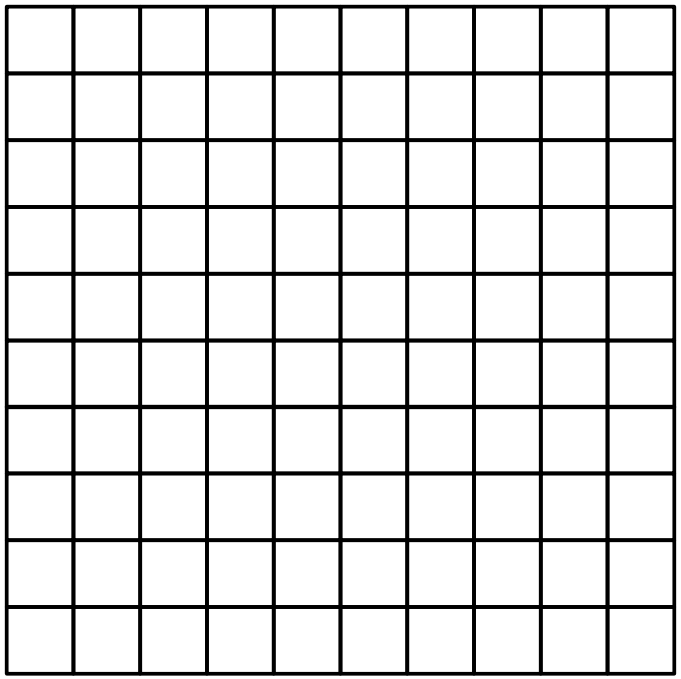
Square Corner Optimality

- **Serial Communication with Barrier:** For processor ratios greater than $3:1$
- **Parallel Communication with Barrier:** For processor ratios greater than $3:1$
- **Serial Communication with Overlap:** For all processor ratios
- **Parallel Communication with Overlap:** For all processor ratios
- **Parallel Interleaving Overlap:** For processor ratios greater than $3:1$

Finding Partition Shapes in Three Processor System

Three Processor Push

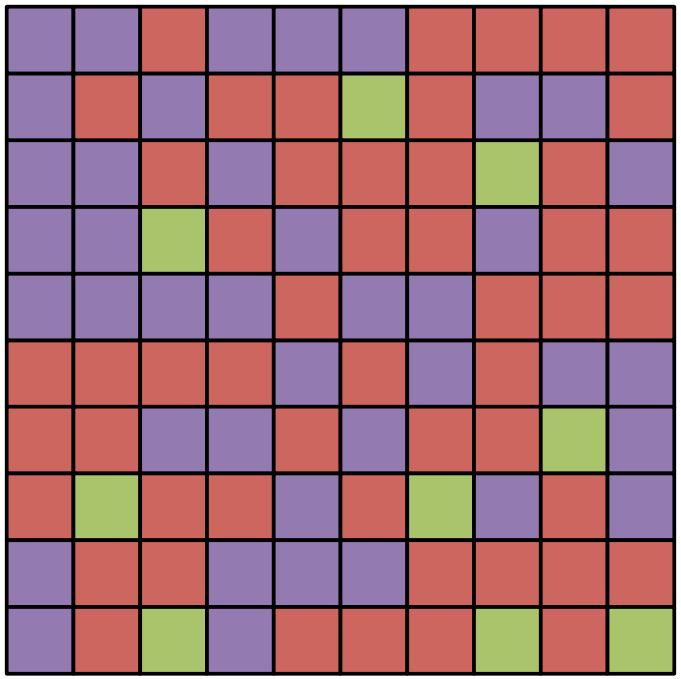
Push elements of slower processor, incrementally improving the volume of communication with each step



Finding Partition Shapes in Three Processor System

Three Processor Push

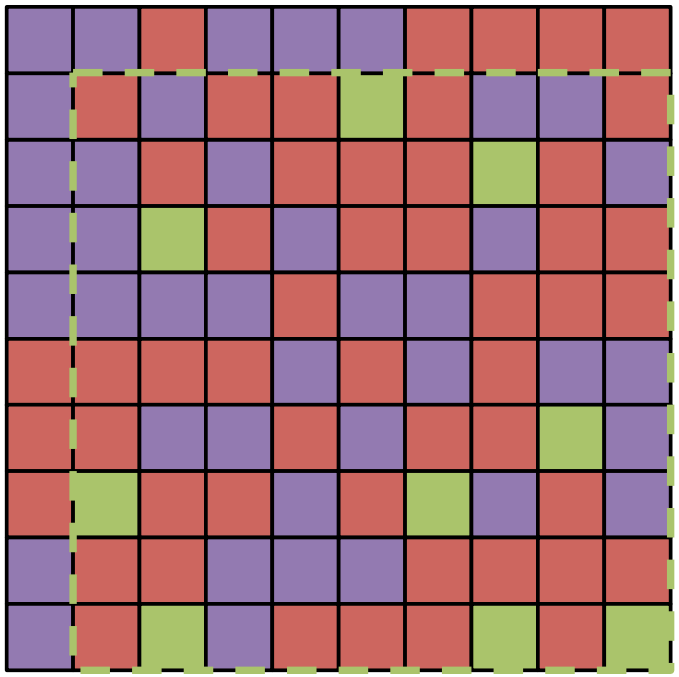
Push elements of slower processor, incrementally improving the volume of communication with each step



Finding Partition Shapes in Three Processor System

Three Processor Push

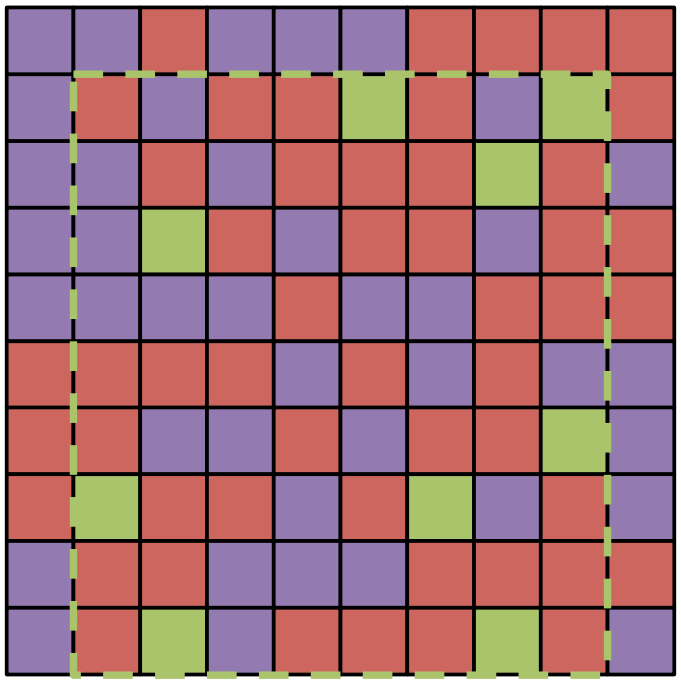
Push elements of slower processor, incrementally improving the volume of communication with each step



Finding Partition Shapes in Three Processor System

Three Processor Push

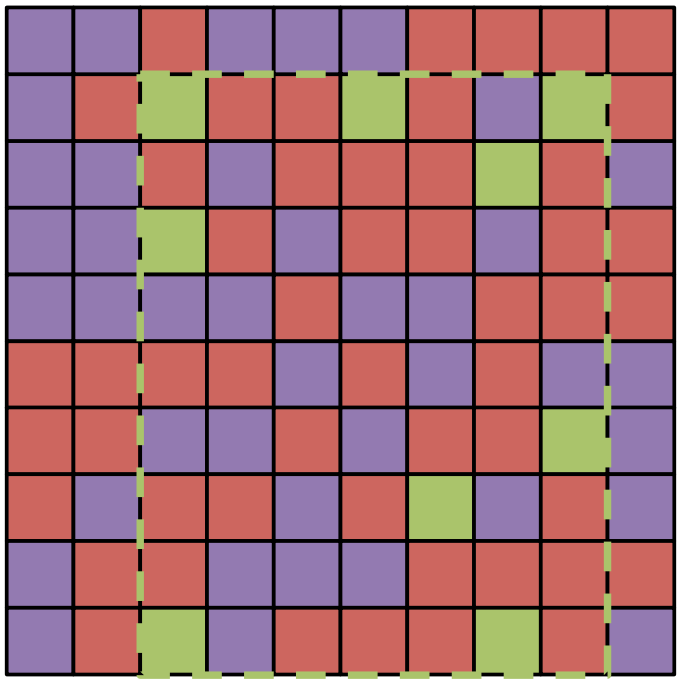
Push elements of slower processor, incrementally improving the volume of communication with each step



Finding Partition Shapes in Three Processor System

Three Processor Push

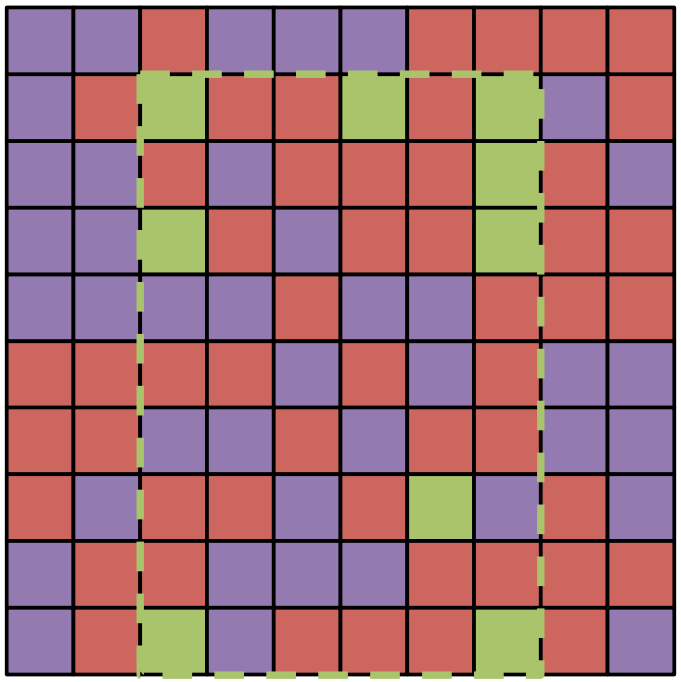
Push elements of slower processor, incrementally improving the volume of communication with each step



Finding Partition Shapes in Three Processor System

Three Processor Push

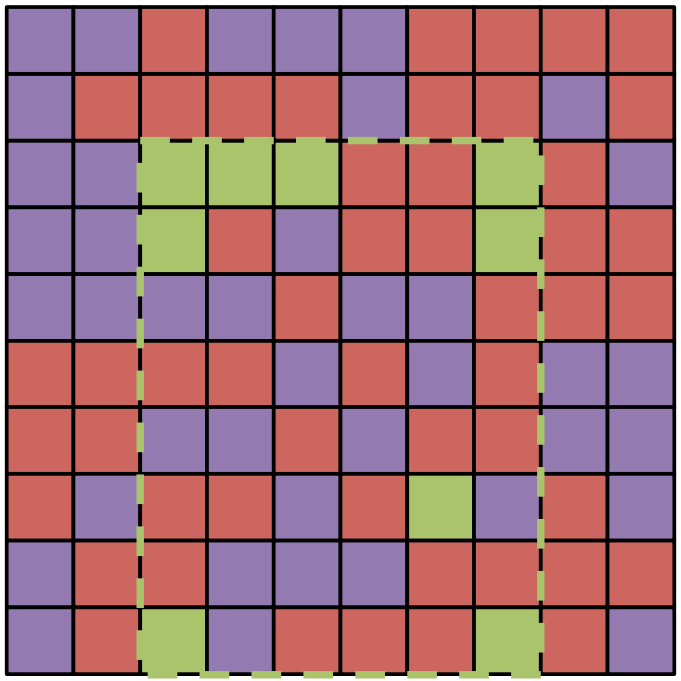
Push elements of slower processor, incrementally improving the volume of communication with each step



Finding Partition Shapes in Three Processor System

Three Processor Push

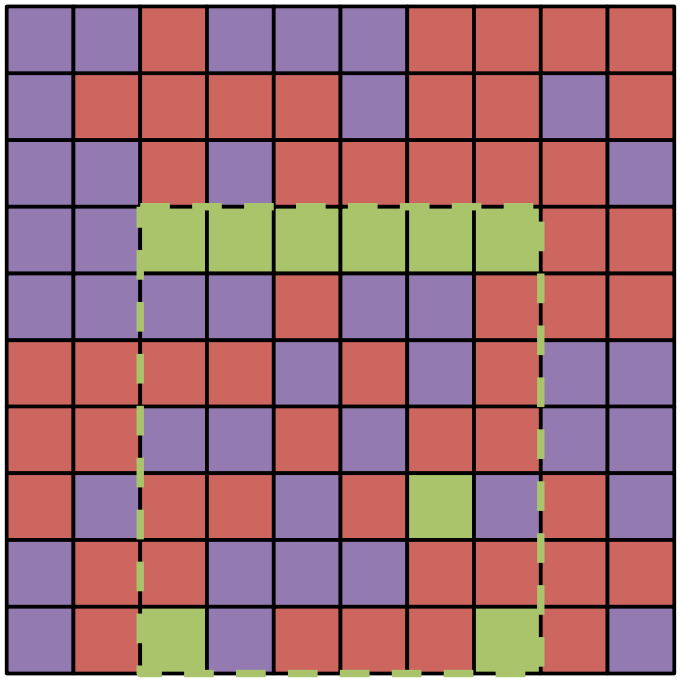
Push elements of slower processor, incrementally improving the volume of communication with each step



Finding Partition Shapes in Three Processor System

Three Processor Push

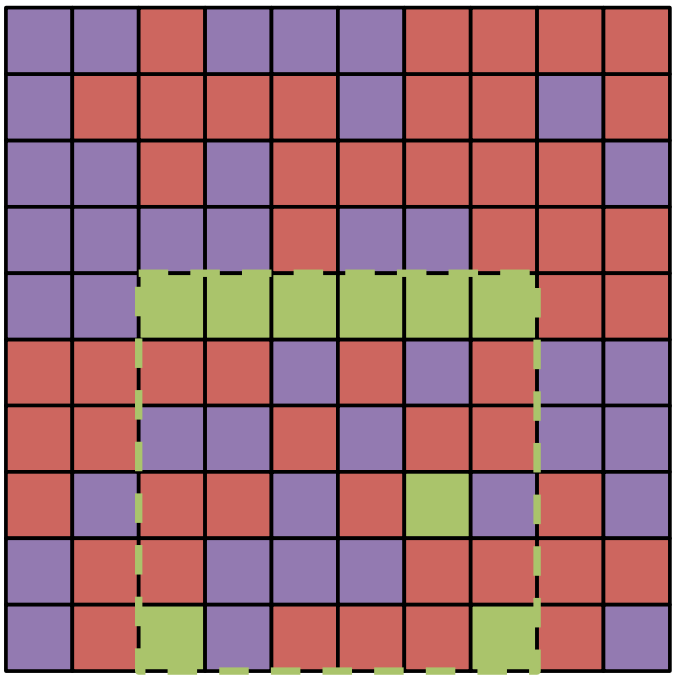
Push elements of slower processor, incrementally improving the volume of communication with each step



Finding Partition Shapes in Three Processor System

Three Processor Push

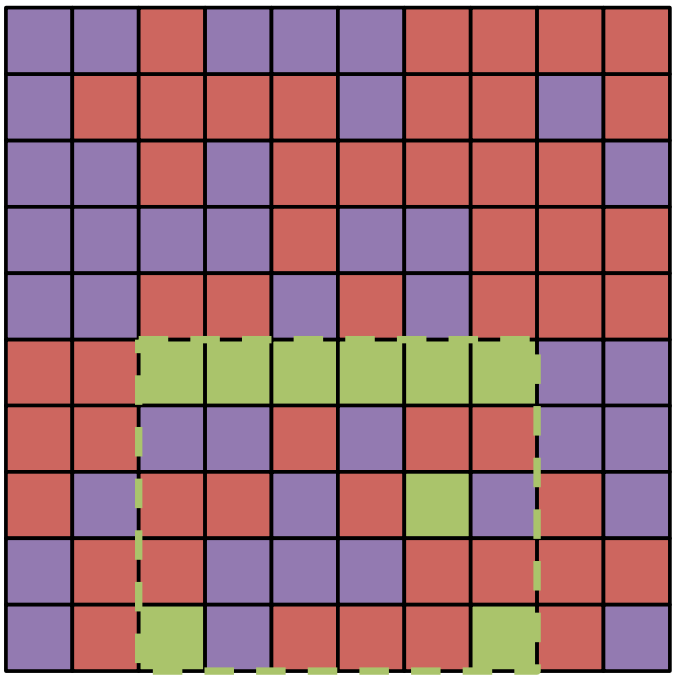
Push elements of slower processor, incrementally improving the volume of communication with each step



Finding Partition Shapes in Three Processor System

Three Processor Push

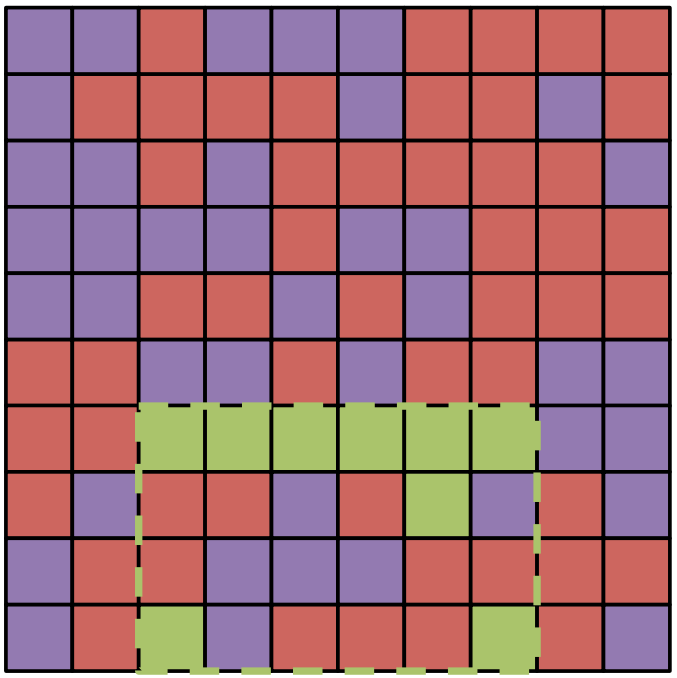
Push elements of slower processor, incrementally improving the volume of communication with each step



Finding Partition Shapes in Three Processor System

Three Processor Push

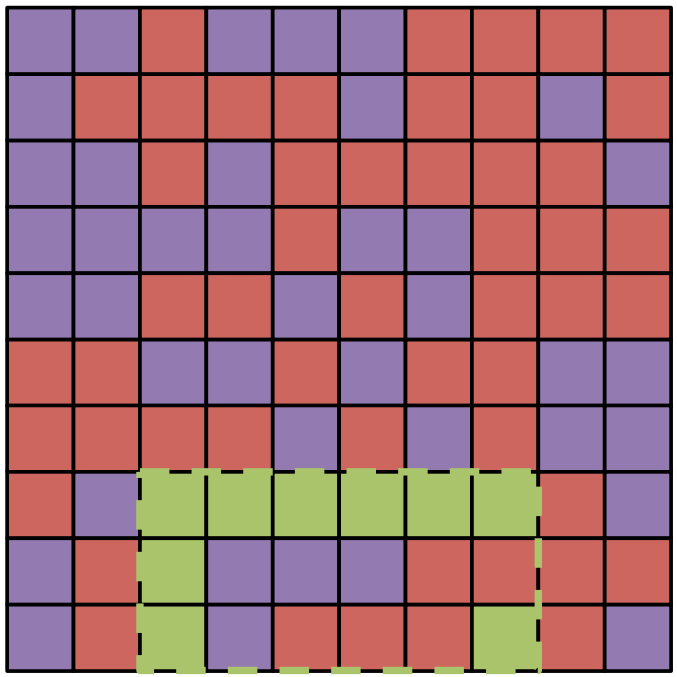
Push elements of slower processor, incrementally improving the volume of communication with each step



Finding Partition Shapes in Three Processor System

Three Processor Push

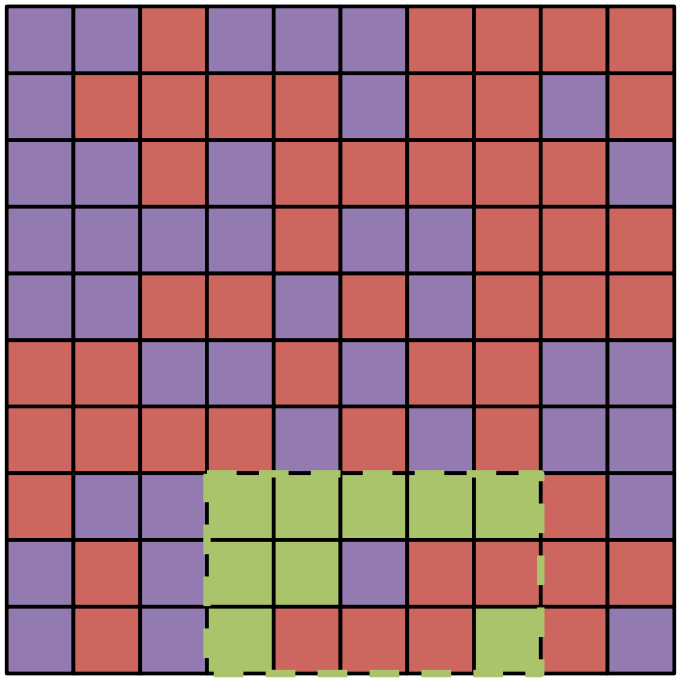
Push elements of slower processor, incrementally improving the volume of communication with each step



Finding Partition Shapes in Three Processor System

Three Processor Push

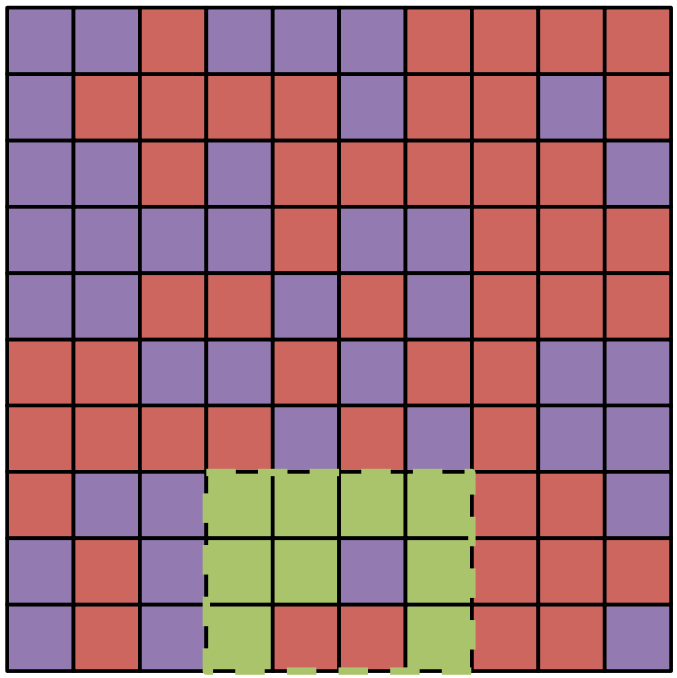
Push elements of slower processor, incrementally improving the volume of communication with each step



Finding Partition Shapes in Three Processor System

Three Processor Push

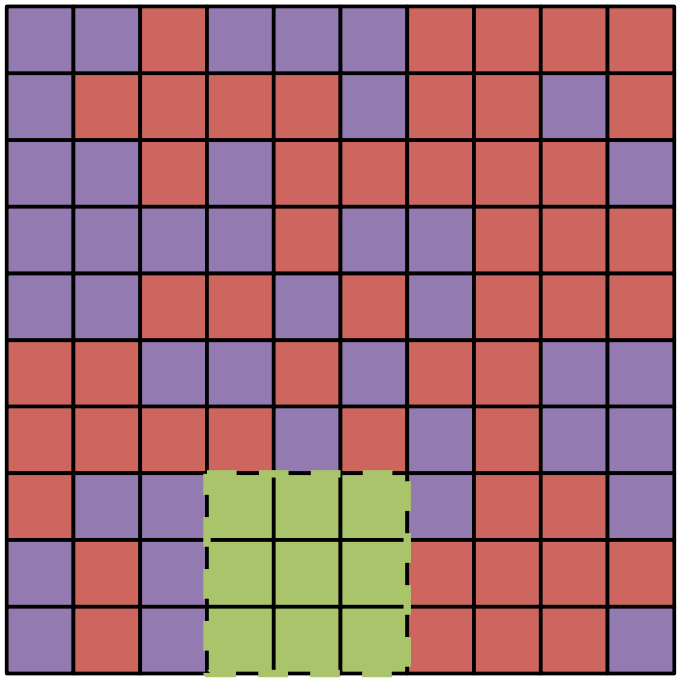
Push elements of slower processor, incrementally improving the volume of communication with each step



Finding Partition Shapes in Three Processor System

Three Processor Push

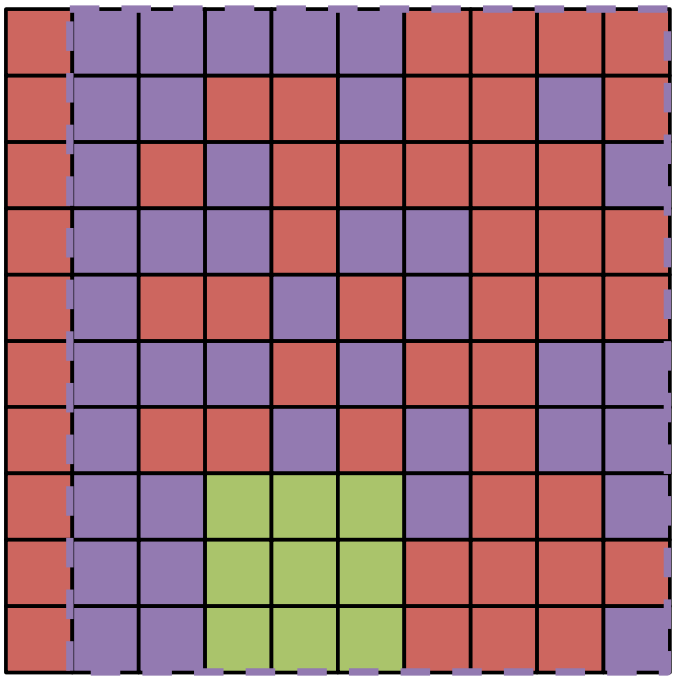
Push elements of slower processor, incrementally improving the volume of communication with each step



Finding Partition Shapes in Three Processor System

Three Processor Push

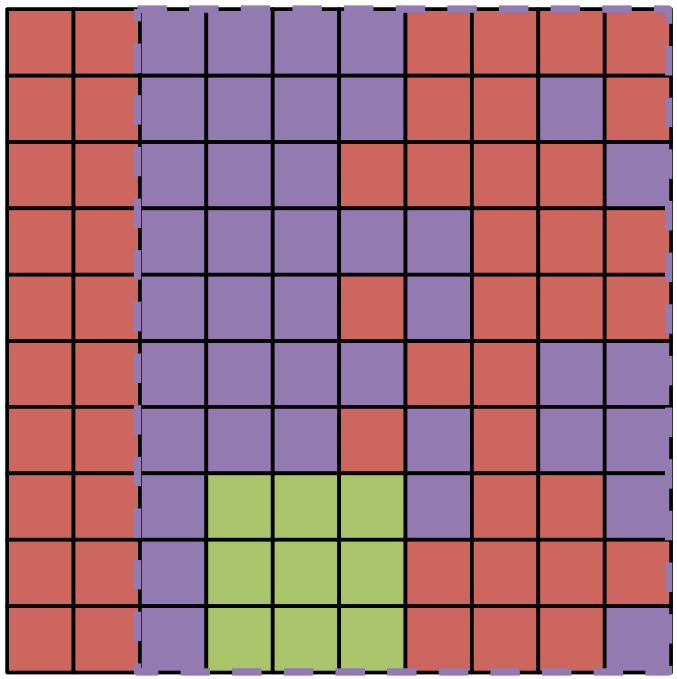
Push elements of next slowest processor, incrementally improving the volume of communication with each step



Finding Partition Shapes in Three Processor System

Three Processor Push

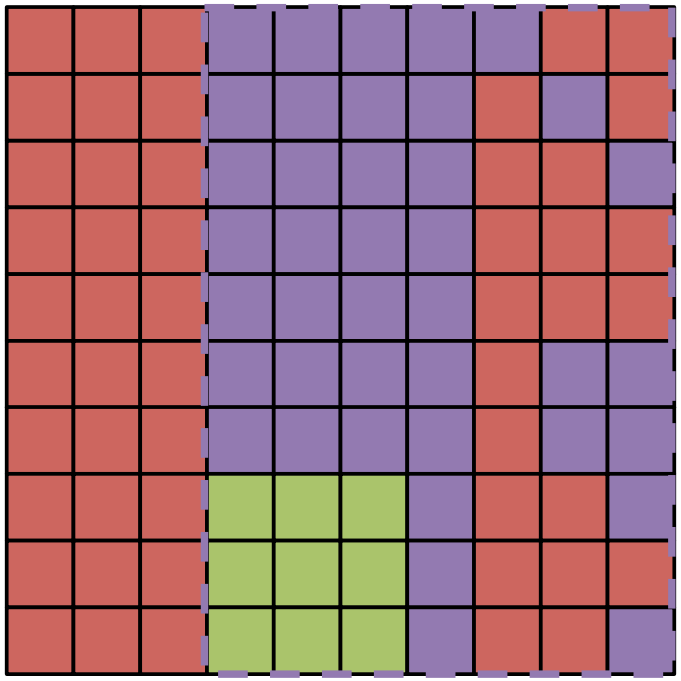
Push elements of next slowest processor, incrementally improving the volume of communication with each step



Finding Partition Shapes in Three Processor System

Three Processor Push

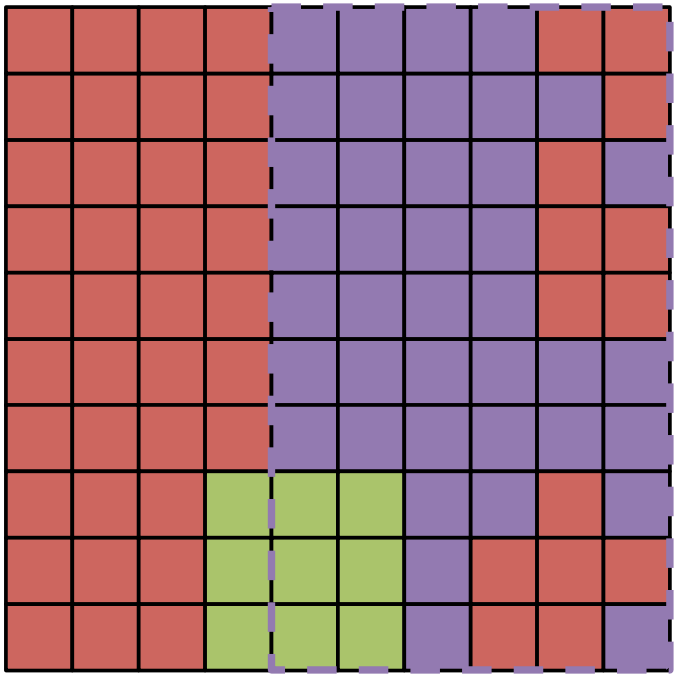
Push elements of next slowest processor, incrementally improving the volume of communication with each step



Finding Partition Shapes in Three Processor System

Three Processor Push

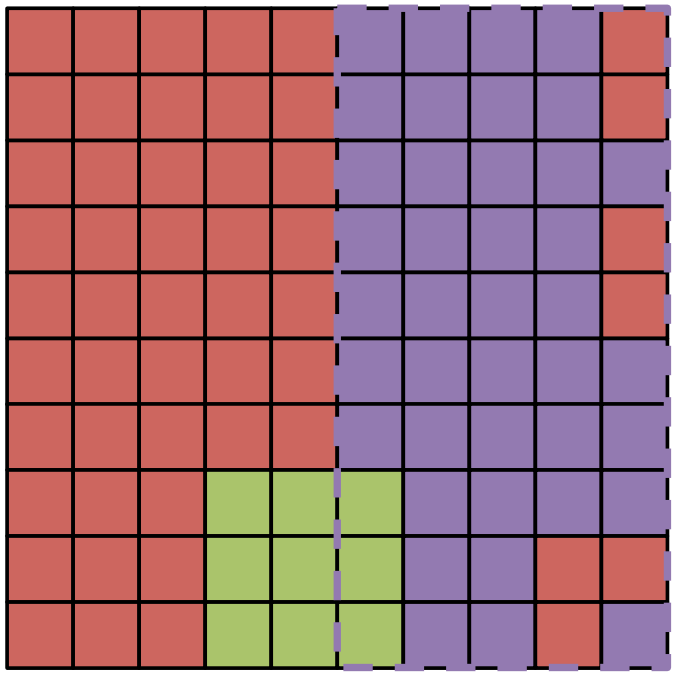
Push elements of next slowest processor, incrementally improving the volume of communication with each step



Finding Partition Shapes in Three Processor System

Three Processor Push

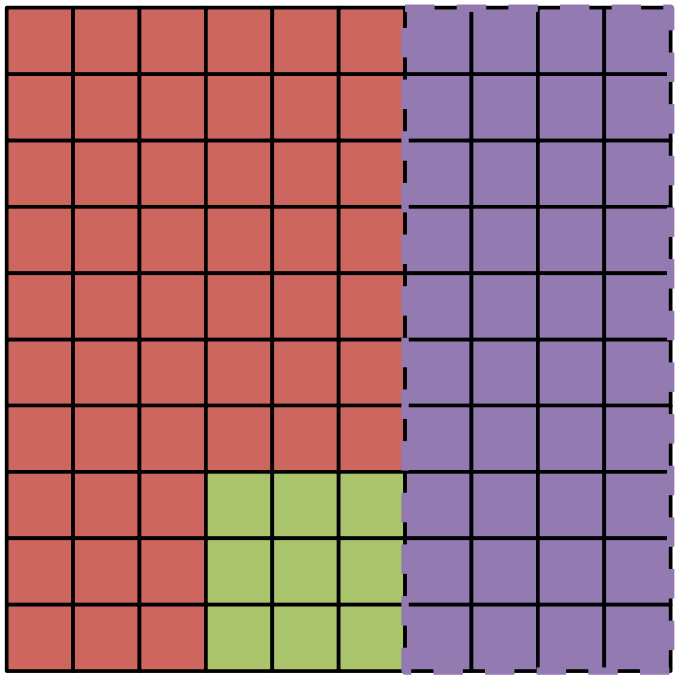
Push elements of next slowest processor, incrementally improving the volume of communication with each step



Finding Partition Shapes in Three Processor System

Three Processor Push

Push elements of next slowest processor, incrementally improving the volume of communication with each step



Convergence - DFA Program

Three Processor Challenges

- Two Processor Push can be mathematically shown to always converge to recognisable shapes
- Three Processor Push is more complex
- Consider legality of moving both processors, not simply the active processor being Pushed
- Must show that Three Processor Push always forms some recognisable shape
- Use a hybrid of analytical and experimental approaches to convince ourselves this is possible

DFA Program Definition

- Present problem as a Deterministic Finite Automaton, $(Q, \Sigma, \delta, q_0, F)$
- Q - the finite set of states, possible data partition shapes
- Σ - the finite set of the alphabet, the processors and directions of Push
- δ - $Q \times \Sigma \rightarrow Q$, the transition function, the Push operation
- q_0 - the start state, chosen at random
- F - $F \subseteq Q$, the accept states, candidates to be the optimum

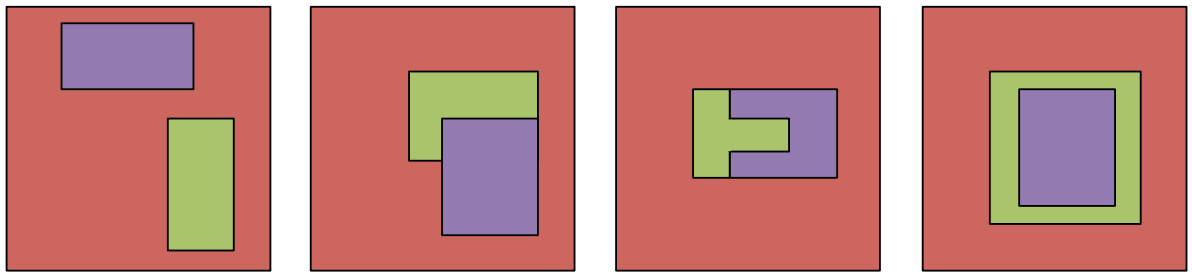
Postulate 1 - Three Processor Push

There exists no arrangement of elements among three heterogeneous processors in an $N \times N$ matrix which cannot be improved with the Push operation, except those arrangements of shapes defined as Archetypes A, B, C and D.

Analysis

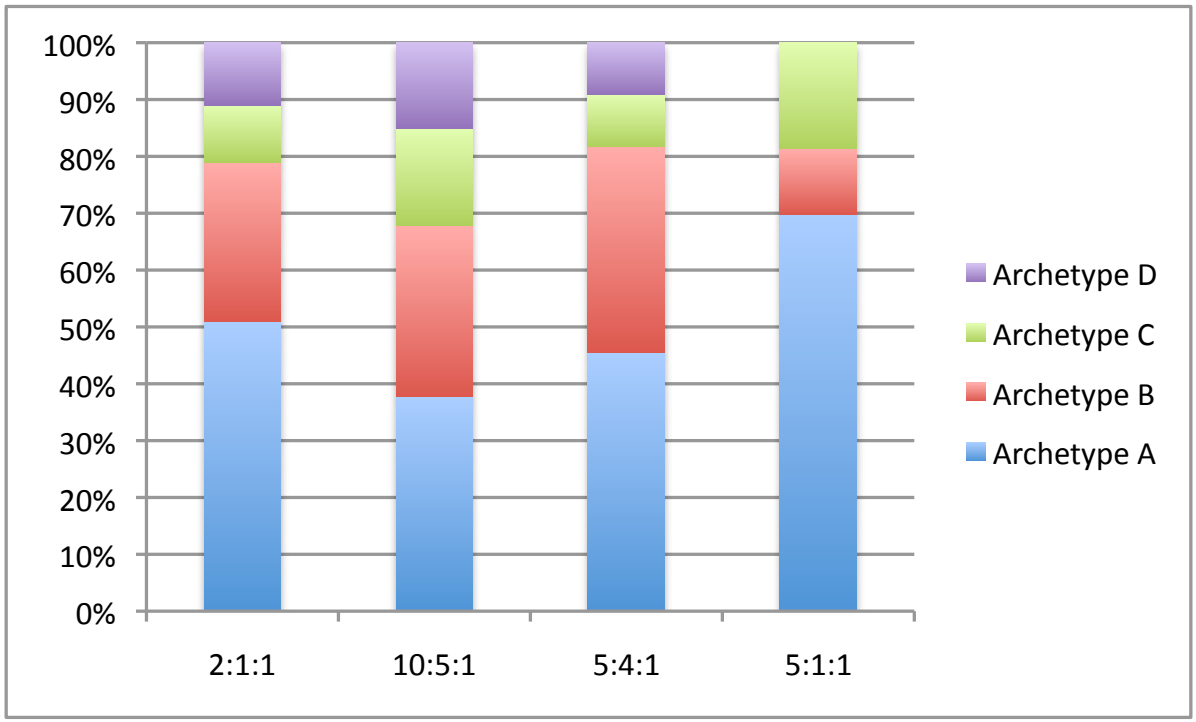
Four Shape Archetypes

- Categorised by Enclosing Rectangles and number of Corners
- **Archetype A:** Slower processors have non-overlapping enclosing rectangles
- **Archetype B:** Slower processors have partially overlapping enclosing rectangles, (1 extra corner)
- **Archetype C:** Slower processors have partially overlapping enclosing rectangles, (more than 1 extra corner)
- **Archetype D:** Enclosing rectangle of one slower processor complete surrounds the other



Experimental Setup

- Set $N = 1000$, use variety of ratios of $P_r : R_r : S_r$
- Run DFA program minimum 10,000 times per processor ratio



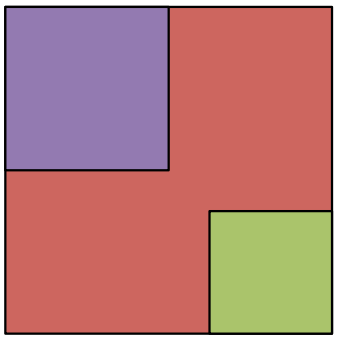
Shape Archetypes

Reducing all Archetypes to Archetype A

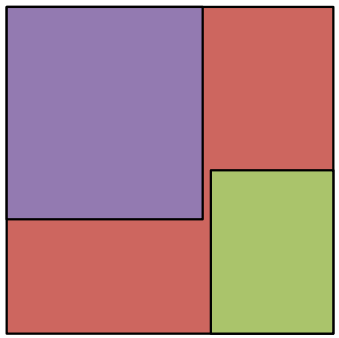
- **B** $\rightarrow$ **A:** A non-Push transformation, guaranteed not to raise volume of communication
- **C** $\rightarrow$ **A:** Execute Push operations in direction(s) not chosen randomly by DFA program (no example found where Push was not possible)
- **D** $\rightarrow$ **A:** A non-Push transformation, guaranteed not to raise volume of communication

Three Processor Candidate Shapes

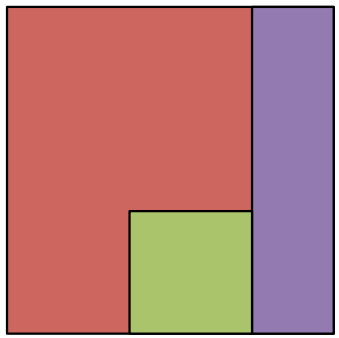
Archetype A has many constituent partition shapes, we create a canonical form for each:



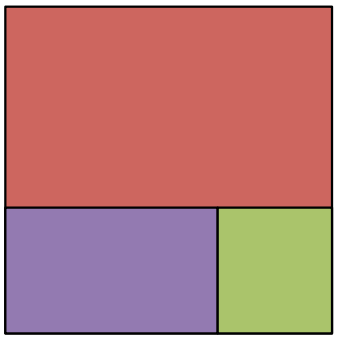
(1) Square Corner



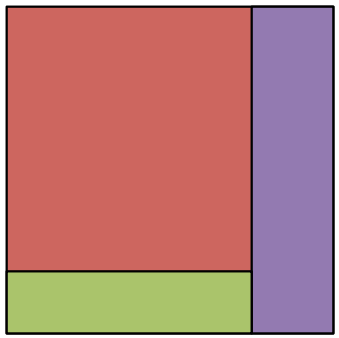
(2) Rectangle Corner



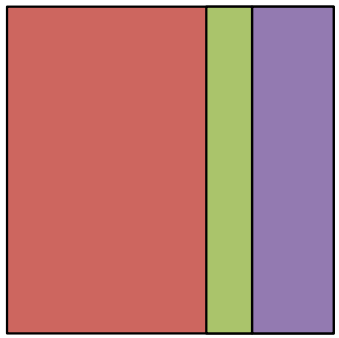
(3) Square Rectangle



(4) Block Rectangle



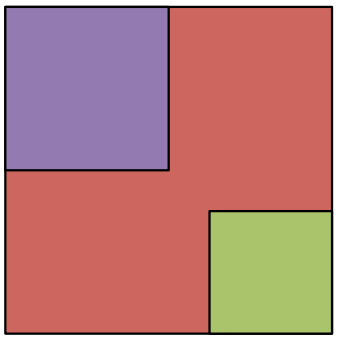
(5) L Rectangle



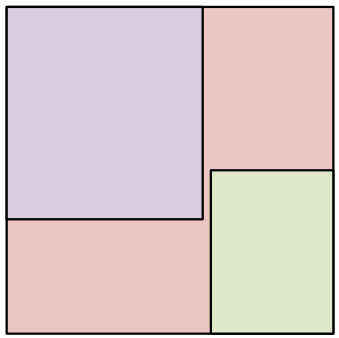
(6) Traditional Rectangle

Three Processor Candidate Shapes

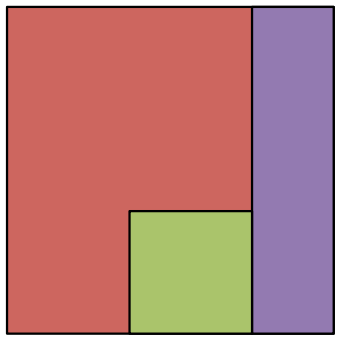
Proved analytically that three are superior to others, and should be analysed further:



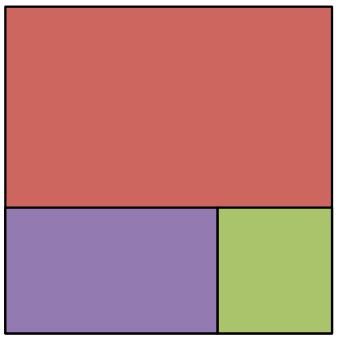
(1) Square Corner



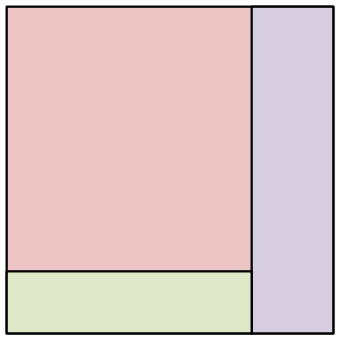
(2) Rectangle Corner



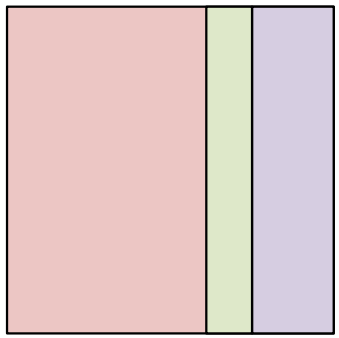
(3) Square Rectangle



(4) Block Rectangle



(5) L Rectangle



(6) Traditional Rectangle

Detailed Analysis for Three Fully Connected Processors

Optimal Shapes by MMM Algorithm

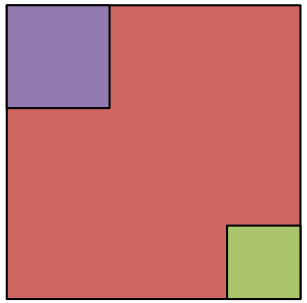
- **Serial Communication with Barrier:**
  - Square Corner : $P_r < 2T - 2\sqrt{R_r T} - 2\sqrt{T}$
  - Rectangle Corner : $P_r < T - 2\sqrt{T}$
  - Block Rectangle Otherwise
- **Parallel Communication with Barrier:**
  - Square Corner : $P_r > 2(\sqrt{R_r T} - R_r + \sqrt{T} - 1)$
  - Rectangle Corner : $P_r < 2R_r + \frac{R_r}{\sqrt{T}} - 2\sqrt{T} - 1$
  - Block Rectangle Otherwise
- **Serial Communication with Overlap:**
  - Square Corner : $P_r > 2\frac{c}{N}(\sqrt{R_r T} + \sqrt{T}) + 2T(r - r^2 - \frac{r^2}{\sqrt{R_r}} + \frac{r}{\sqrt{R_r}} - \frac{r^2}{R_r}) - \frac{Tc}{N} - \frac{2c}{N}\sqrt{T}$
  - Rectangle Corner : $P_r < T - 2\sqrt{T}$
  - Block Rectangle Otherwise
- **Parallel Communication with Overlap:**
  - Square Rectangle : $P_r < \frac{1+\frac{2}{\sqrt{T}} - \frac{R_r}{T} - \frac{R_r}{T\sqrt{T}} - \frac{3}{T} - 2r^2}{N(\frac{r^2}{cR_r} - \frac{1}{Tc})}$
  - Square Corner Otherwise
- **Parallel Interleaving Overlap:**
  - Block Rectangle : $P_r < 4\sqrt{T}$
  - Square Corner Otherwise

Optimal Shape for Three Processor

Summary of Analysis

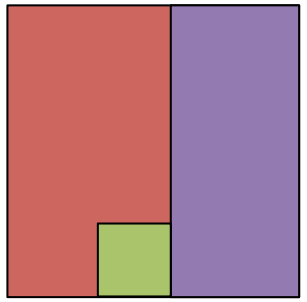
Square Corner Optimality

Optimal for systems with 1 fast processor, and two relatively slow processors



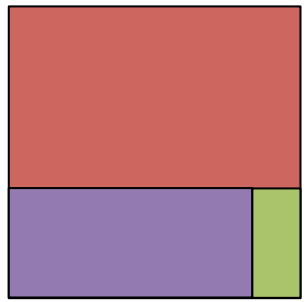
Square Rectangle Optimality - *(A Shape Never Considered Before)*

Optimal for systems with 2 fast processors, and one relatively slow processor



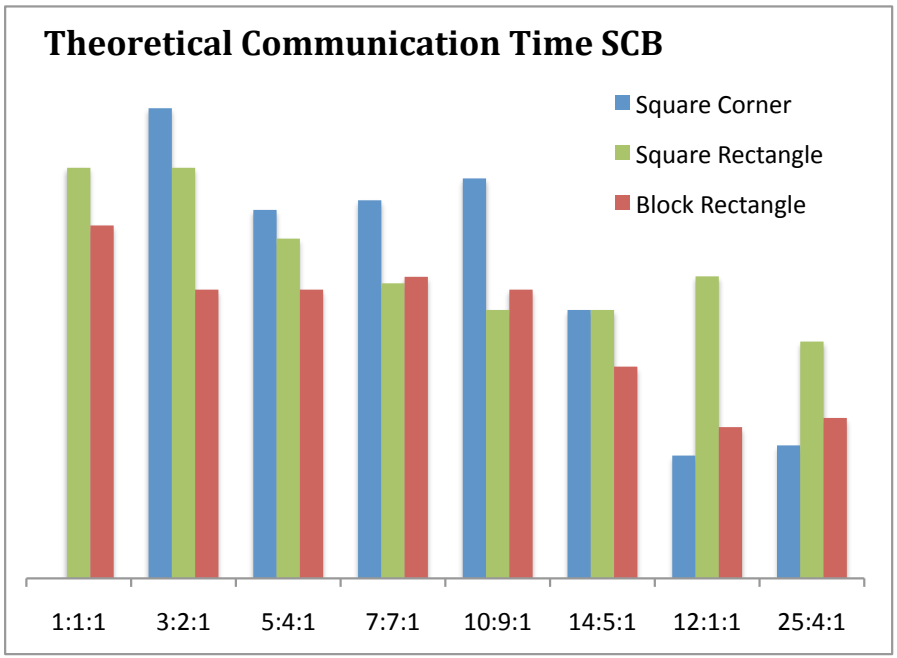
Block Rectangle Optimality

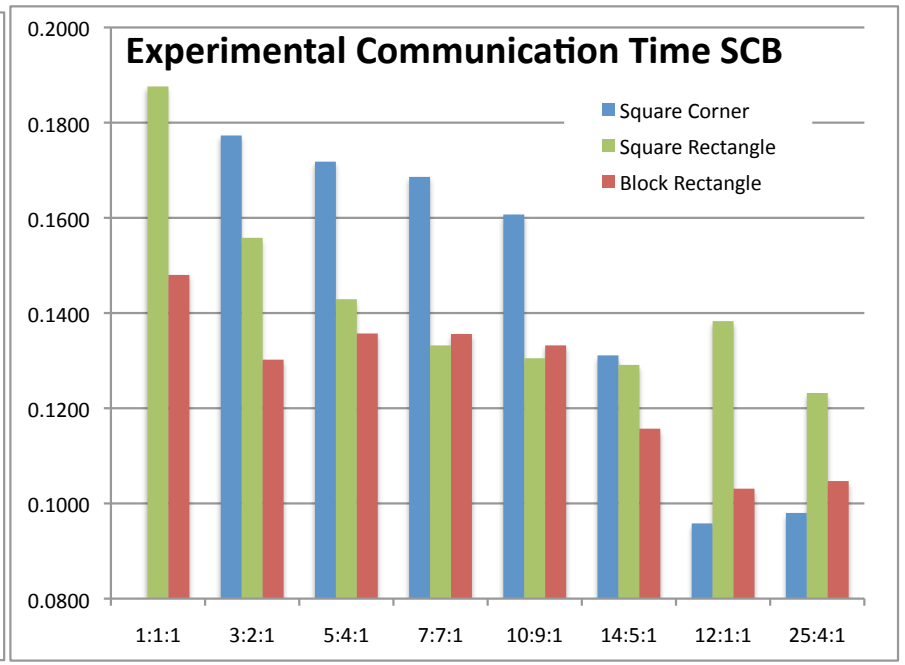
Optimal for systems with 1 fast, 1 medium and 1 slow processor, as well as relatively homogeneous systems



Three Processor Experimental Results

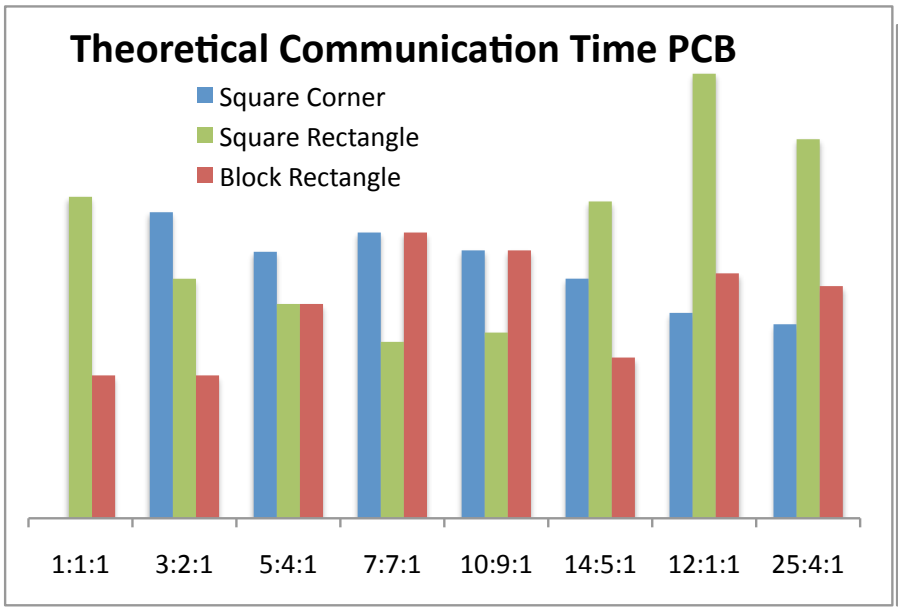
Serial Communication with Barrier

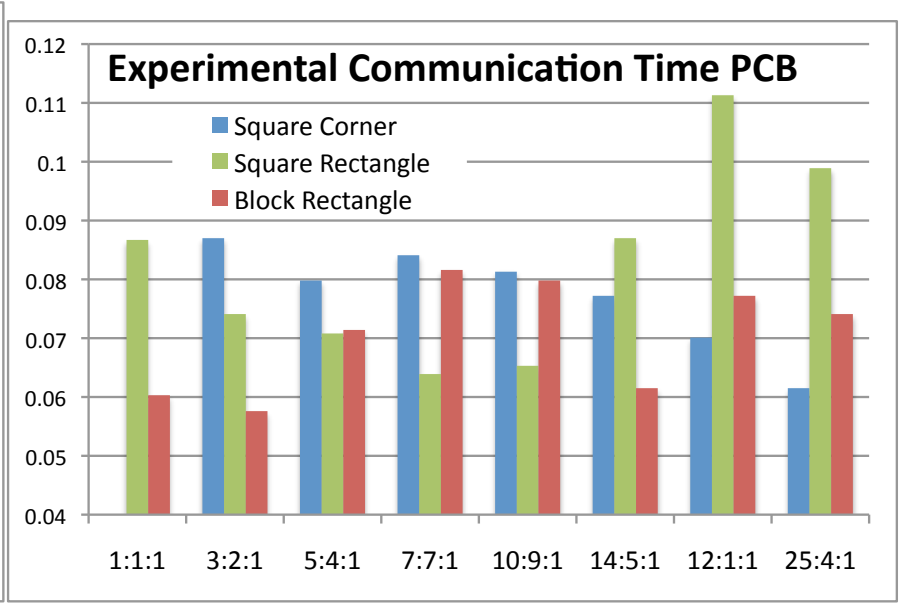




Three Processor Experimental Results

Parallel Communication with Barrier





Thank You