

A Parallel Data Mining Package Using MatlabMPI

Parna Khot

Ashok Krishnamurthy

Stan Ahalt

John Nehrbass

Juan Carlos Chaves

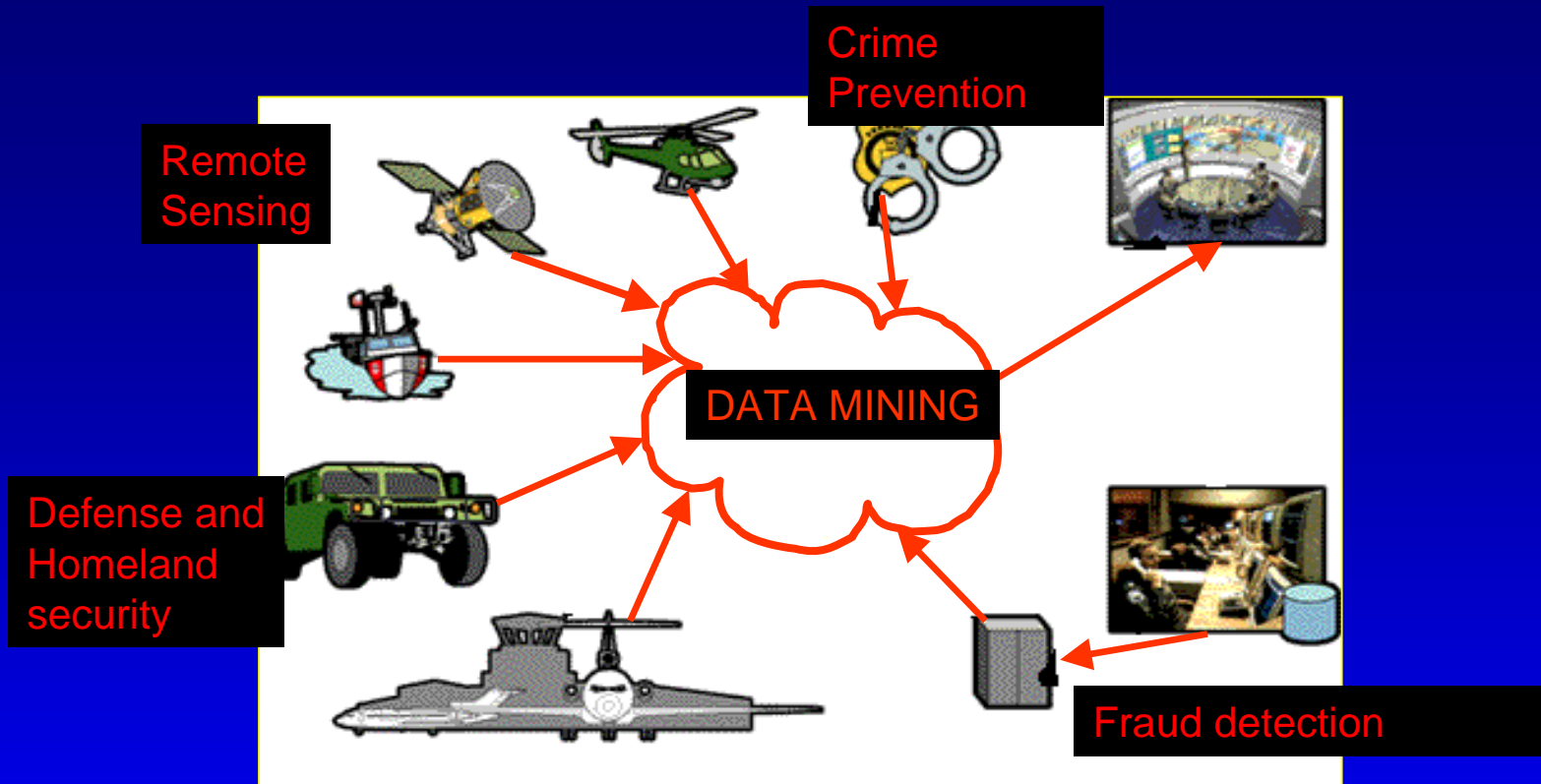
The Ohio State University



Outline

- **Motivation**
 - Why parallel data mining toolbox?
- **MatlabMPI**
 - What is MatlabMPI?
- **Parallel data mining toolbox**
 - K-Means Clustering
 - CART
- **Results of MatlabMPI implementation**
- **Conclusions**
- **Future Work**

Motivation



- Today, the amount of data that is collected from sensors and computerized transactions is huge.
- Data Mining algorithms arise in many different fields and typically are used to search through this data to look for patterns.
- Parallel data mining algorithms can help handle the huge datasets in a timely manner.

Typical Data Mining Tasks

- Clustering.
- Classification.
- Association Rules.
- Regression.
- Pattern Recognition

We will consider only Clustering and Classification in this presentation.

MatlabMPI Overview

The latest MatlabMPI information, downloads, documentation, and information may be obtained from:

<http://www.ll.mit.edu/MatlabMPI>

Parallelization using MPI

- **The Message Passing Interface (MPI) is a general method of parallelization by including explicit calls within the code to a library for exchanging messages between the processing elements.**
 - **MPICH**
 - **Implementation of Message Passing Interface standard for C, C++, Fortran77, Fortran90.**
 - **MatlabMPI**
 - **A Matlab implementation of MPI.**

MPI & MATLAB

- Message Passing Interface (MPI):
 - A message-passing library specification.
 - Specific libraries available for almost every kind of HPC platform: shared memory SMPs, clusters, NOWs, Linux, Windows.
 - Fortran, C, C++ bindings.
 - Widely accepted standard for **parallel computing**.
- MATLAB:
 - Integrated computation, visualization, programming, and programming environment.
 - Easy matrix based notation, many toolboxes, etc
 - Used extensively for technical and scientific computing.
 - Currently: mostly **SERIAL code**.

What is MatlabMPI?

- It is a **MATLAB implementation** of the MPI standards that allows any MATLAB program to exploit multiple processors.
- It implements, **the basic MPI functions** that are the core of the MPI point-to-point communications with extensions to other MPI functions. (Growing)
- **MATLAB *look and feel*** on top of standard MATLAB **file I/O**.
- Pure **M-file implementation**: about 100 lines of MATLAB code.
- It runs **anywhere MATLAB runs**.
- Principal developer: **Dr. Jeremy Kepner** (MIT Lincoln Laboratory)

General Requirements

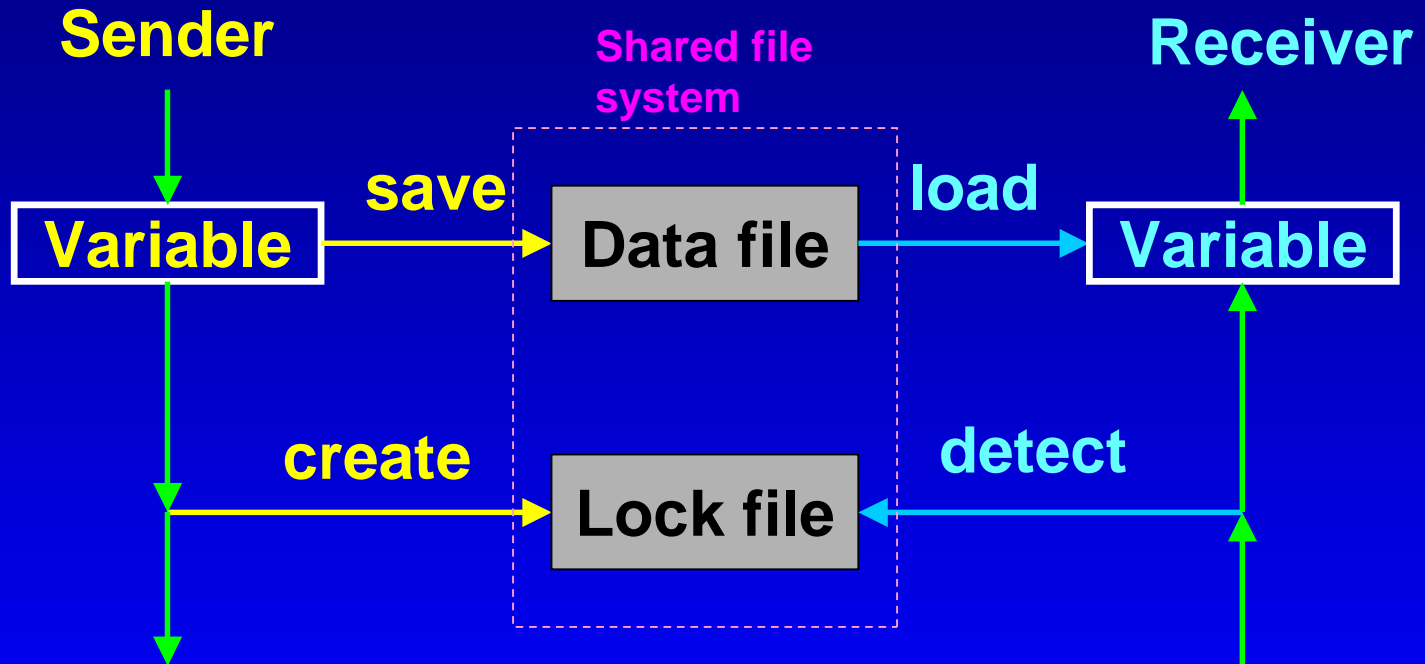
- As MatlabMPI uses file I/O for communication, a **common file system** must be visible to every machine/processor.
- **On shared memory platforms:** single MATLAB license is enough since any user is allowed to launch many MATLAB sessions.
- **On distributed memory platforms:** one MATLAB license per machine / node.
- Currently **Unix** based platforms only, but **Windows** support coming soon.

Basic Concepts

- **Basic Communication:**

- **Messages: MATLAB variables** transferred from one processor to another
- One processor sends the data, another receives the data
- **Synchronous transfer:** call does not return until the message is sent or received
- **SPMD model:** usually MatlabMPI programs are parallel SPMD programs. The same program is running on different processors/data.

Communication architecture



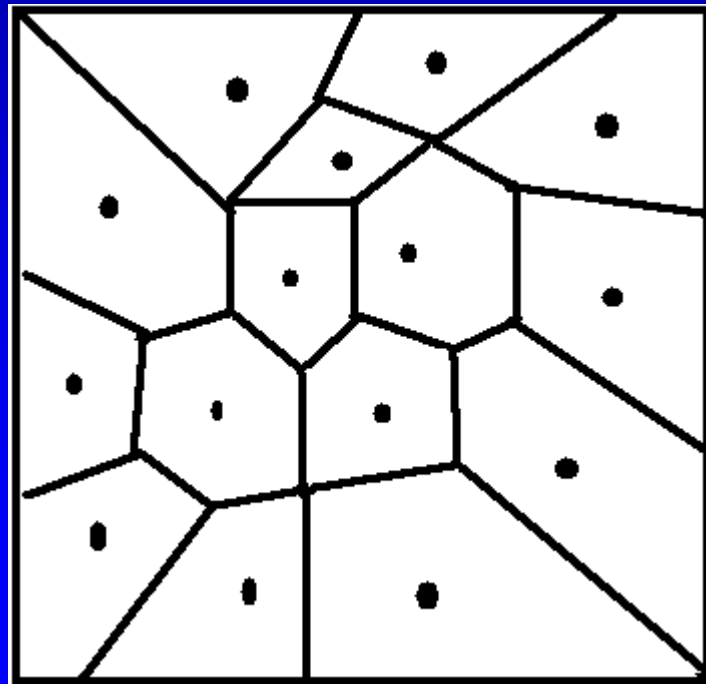
- Receiver waits until it detects the existence of the lock file.
- Receiver deletes the data and lock file, after it loads the variable from the data file.

Possible modifications/customizations

- **ssh vs rsh.**
- **Path variables.**
- **System dependent information required to run MATLAB.**

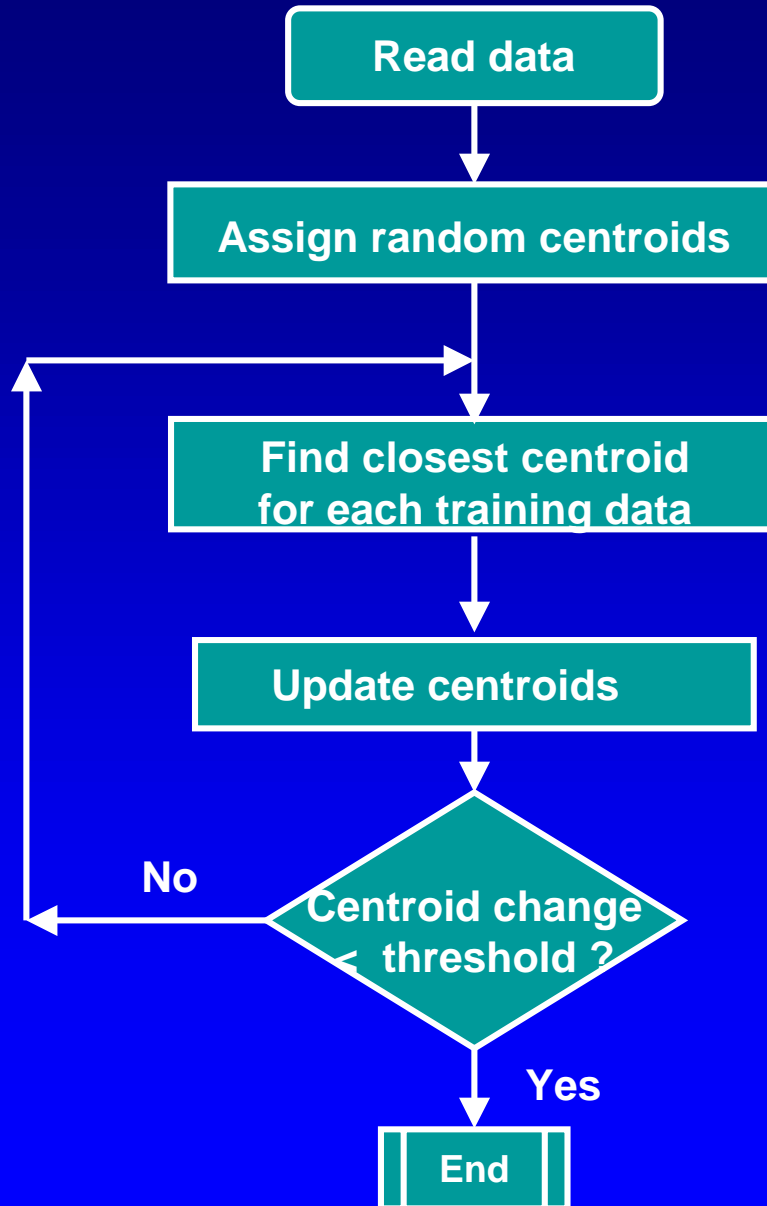
Data Mining Toolbox: Clustering

- Clustering divides the data into disjoint subsets based on a similarity measure.
- Each subset (cluster) is characterized by its centroid.
 - Training data is used to estimate the centroids.
- K-Means is a commonly used clustering algorithm.
 - The number of clusters is assumed to be known apriori.



Voronoi Diagram

K-Means Clustering



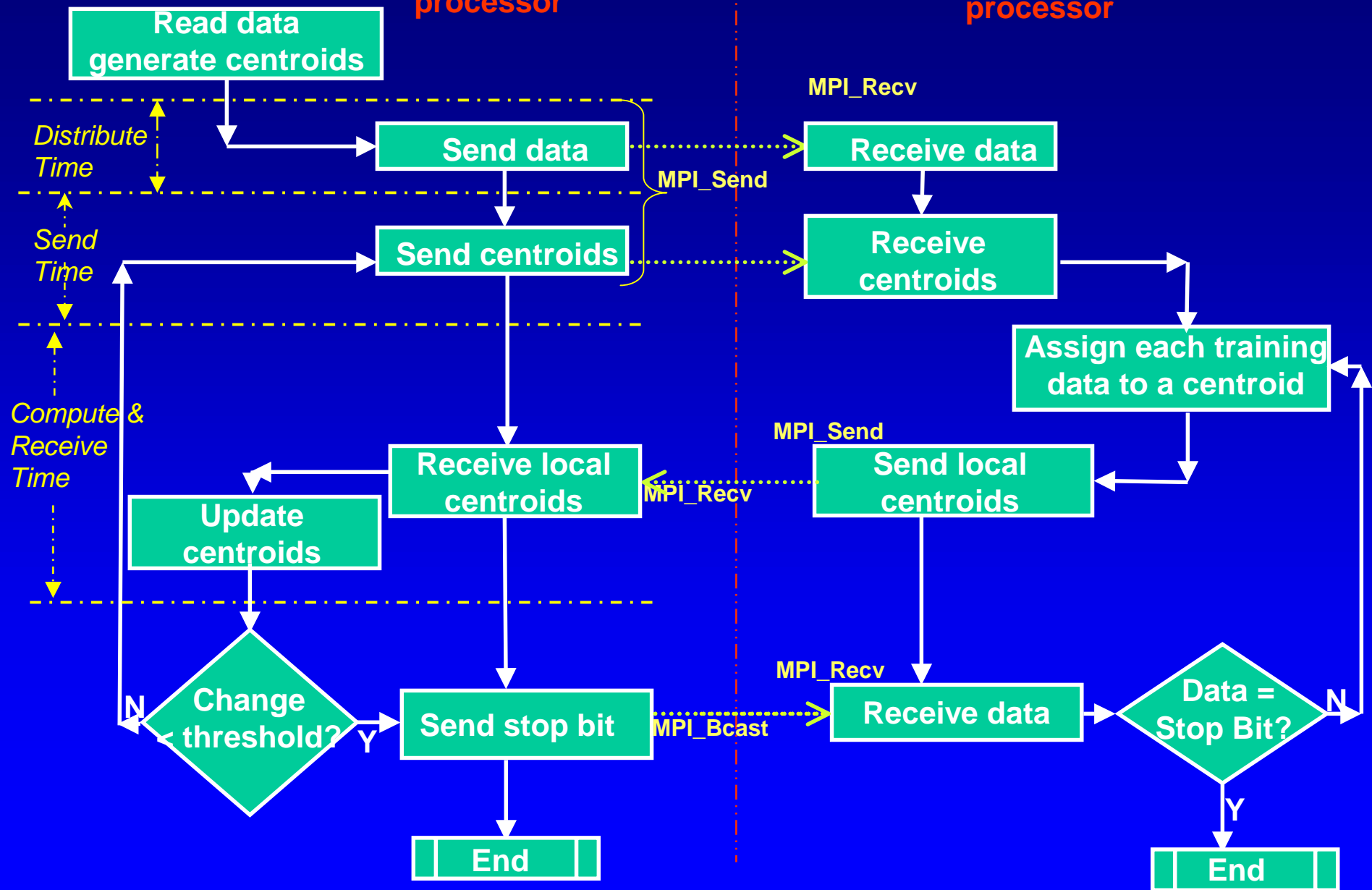
Parallel K-Means Clustering

- We have considered two approaches:
 - **Master-Slave Method**– The rank-0 processor determines when clustering is done.
 - **Peer-to-Peer Method** – All the processing elements communicate among themselves to decide when clustering is done.

Master – Slave Method

rank - 0
processor

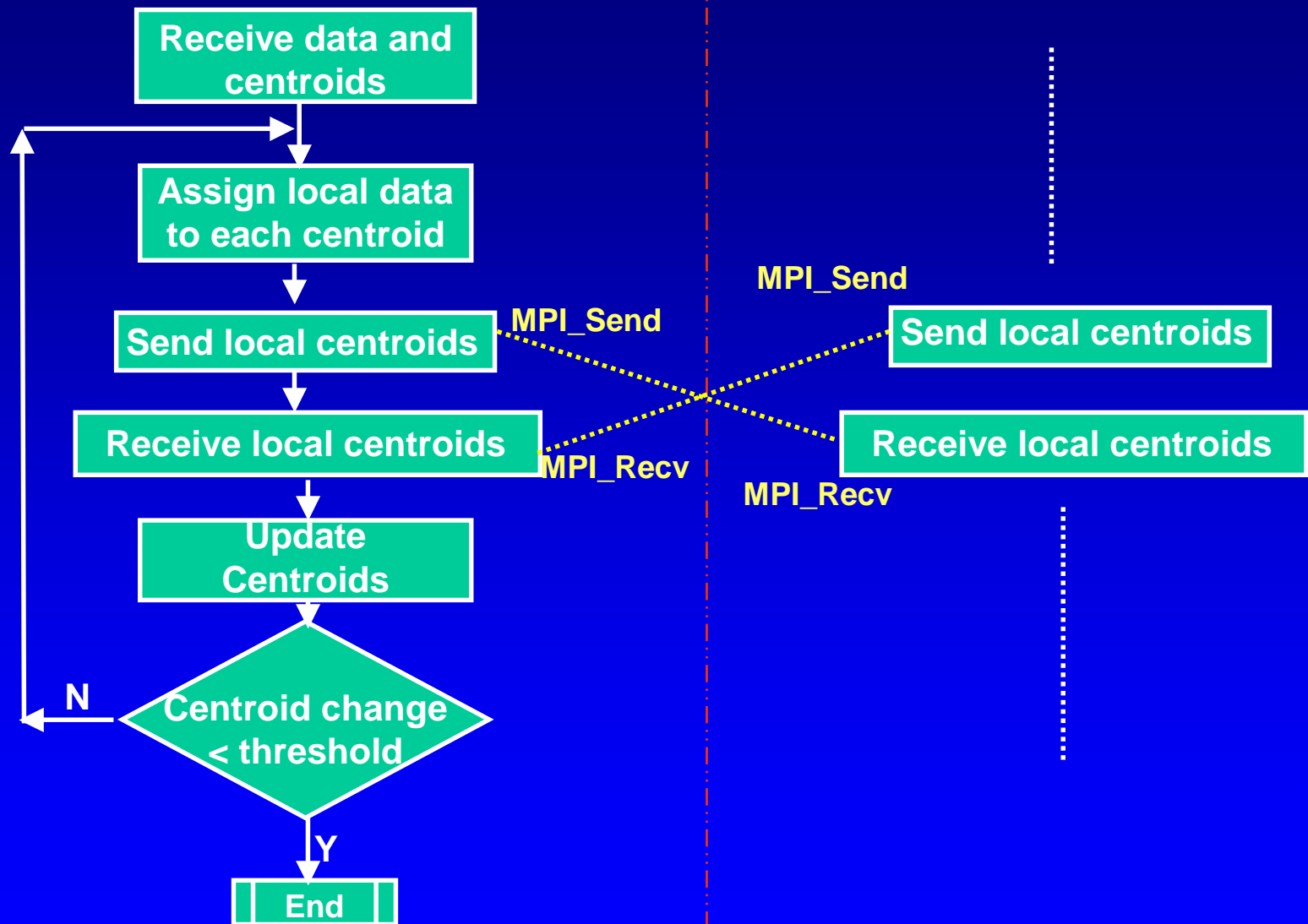
rank -n
processor



Peer-to-Peer Method

Rank-n Processor

Other Processors



Communication And Compute Times

- Consider clustering of N vectors of dimension D into K clusters. Assume that clustering takes L iterations through the data, and P processors are used.
- **Serial Method**
 - Communication Time – N/A
 - Communication Data Size – N/A
 - Compute Time – $O(NKL)$
- **Master Slave Method**
 - Communication Time – $(N-1) * (P+1) T_{\text{MPI_Send}} + (N-1) * P T_{\text{MPI_Recv}}$
 - Communication Data size
 - Initial – $(N+K)/(P-1)$
 - Per loop = K
 - Compute Time / Processor – $O((N/(P-1))K)$
- **Peer-to-Peer Method**
 - Communication Time – $(N) * (P) (T_{\text{MPI_Send}} + T_{\text{MPI_Recv}})$
 - Communication Data size
 - Initial – $(N+K)/(P-1)$
 - Per loop = K
 - Compute Time / Processor – $O((N/P)K)$

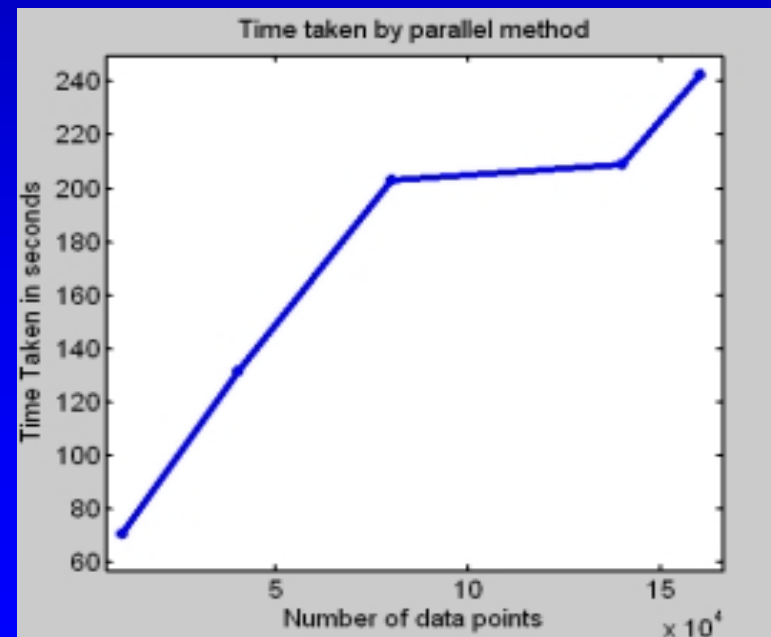
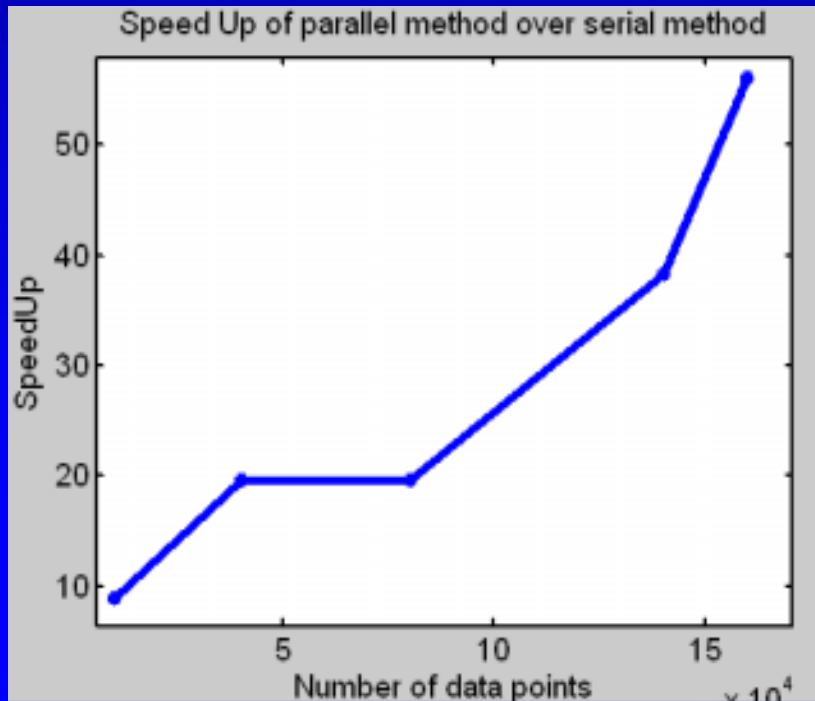
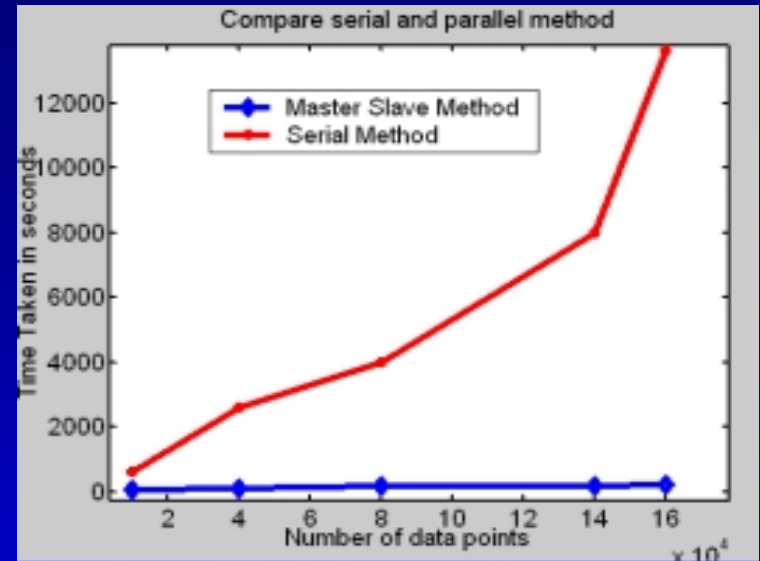
Parallelization Effectiveness

- We studied the effects of following parameter variations on the Master-Slave parallel K-means algorithm
 - Number of data points.
 - To observe the effect of increase in total data size.
 - Number of centroids.
 - Scalability.
 - To observe the effect of change in number of processing elements.

Effect of varying number of data points

- Data Set

- Number of data points: 1M – 16M
- Number of centroids: 30
- Number of processors: 16
- Dimensionality of data: 3
- As number of data points is increased speed up of parallel process over serial process increases.



Effect of varying number of centroids

- **Data Set**

 - Number of data points – 0.4M

 - Number of centroids – varied

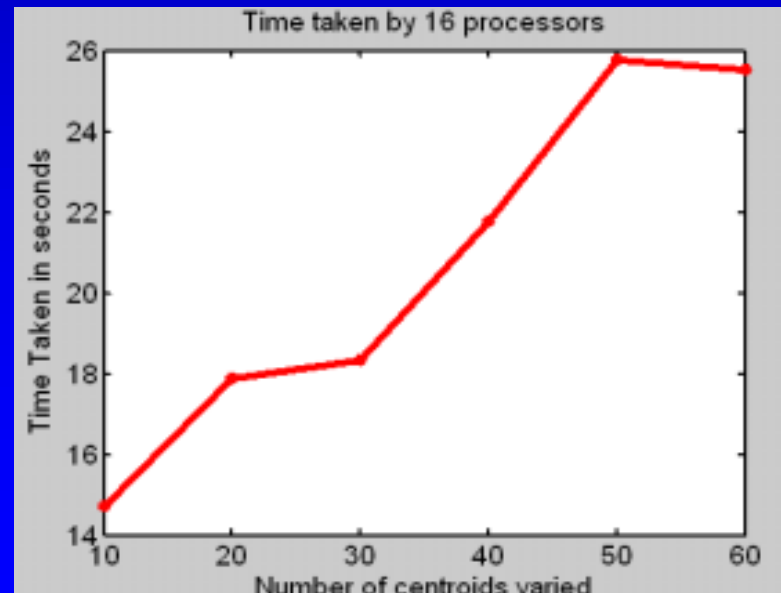
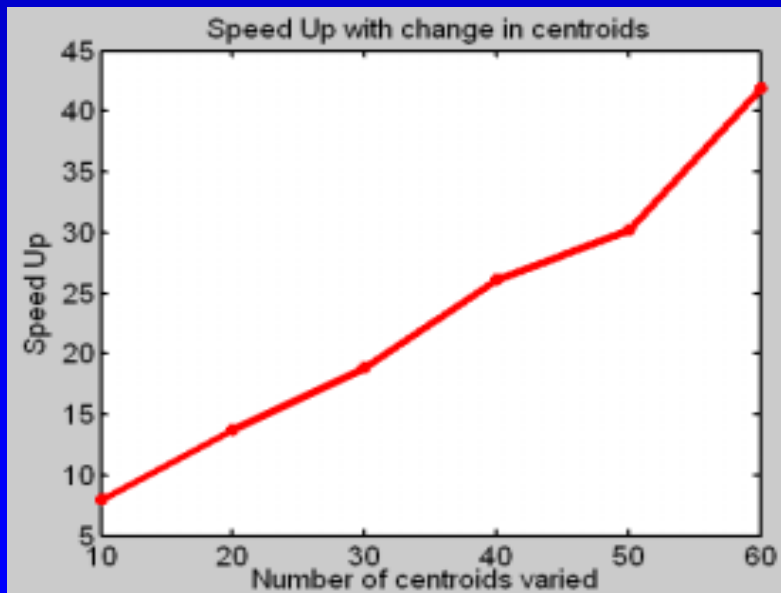
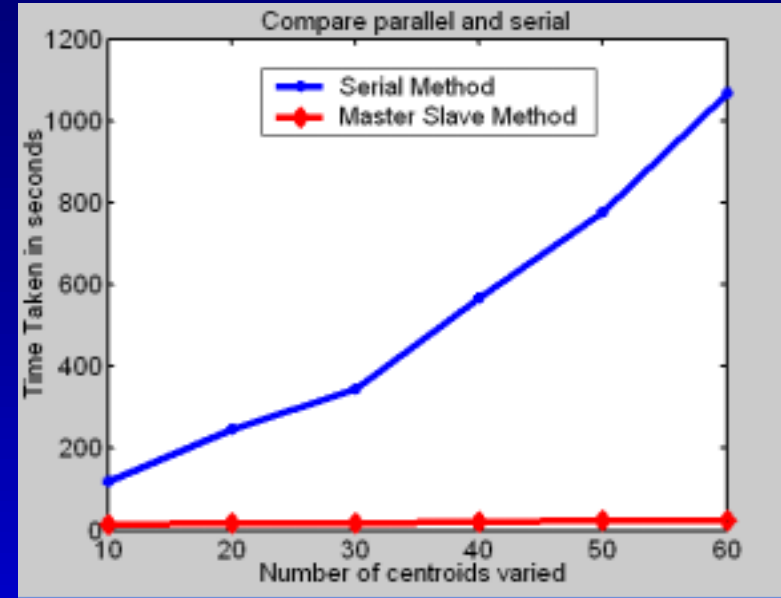
 - Number of processors – 16

 - Dimensionality - 8

- **Effect of increase in number of centroids with constant number of data points**

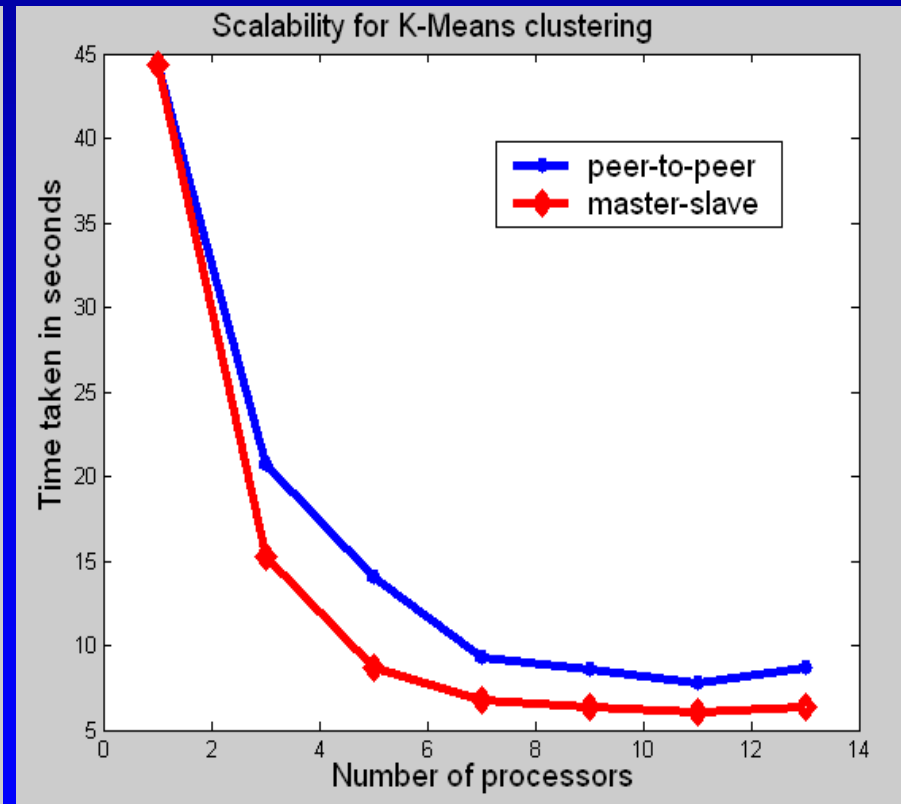
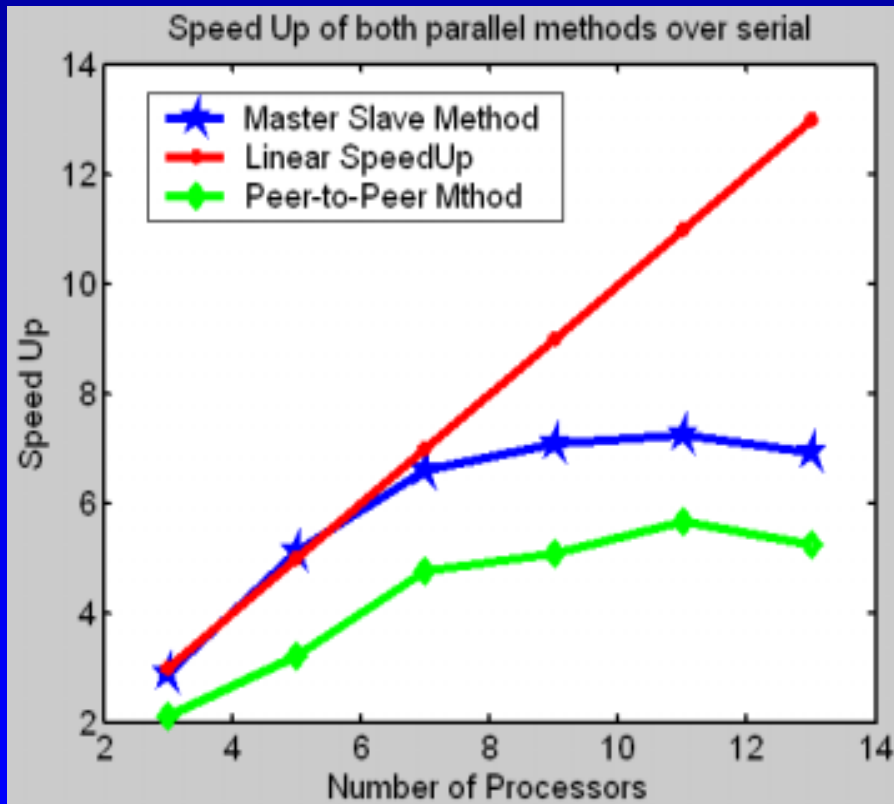
 - The number of data points per process is constant.

 - Speed up observed since compute time is of the order of NK .



Scalability Results

- As number of processors is increased the time taken decreases
 - number of data points: 0.2M
 - number of clusters: 30
 - Dimensionality: 3



Dependence on data size

- The decrease in time as the number of processors is increased is not true for all cases

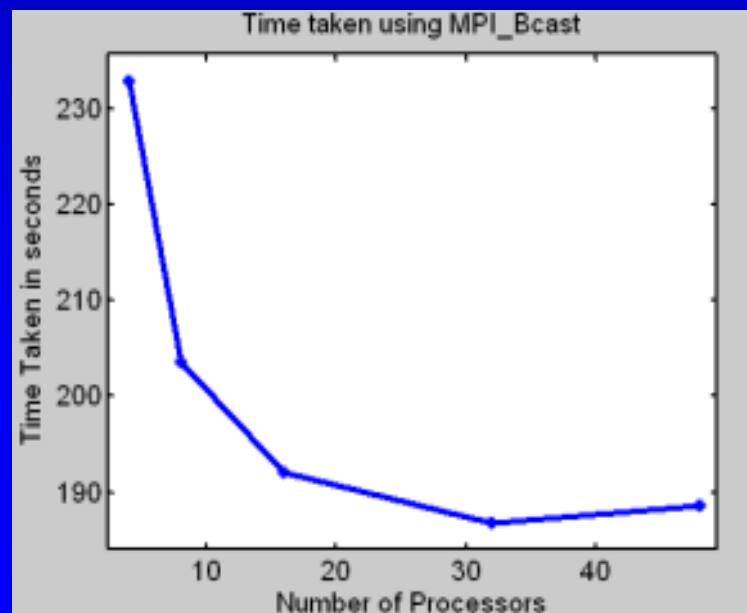
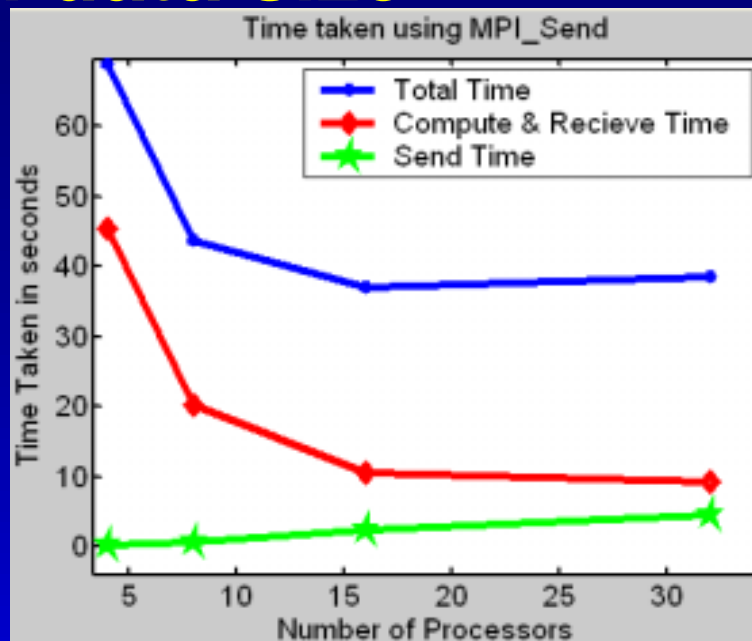
- Data Set for figure :

- Number of data points: 1M
- Number of clusters: 16
- Dimensionality: 8

- For 32 processors increase in time taken to send data is greater than the decrease in computation and receive time.

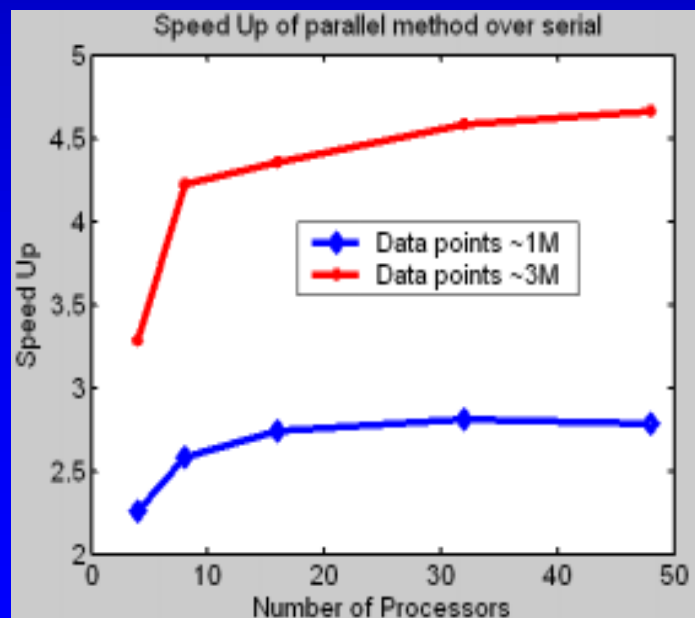
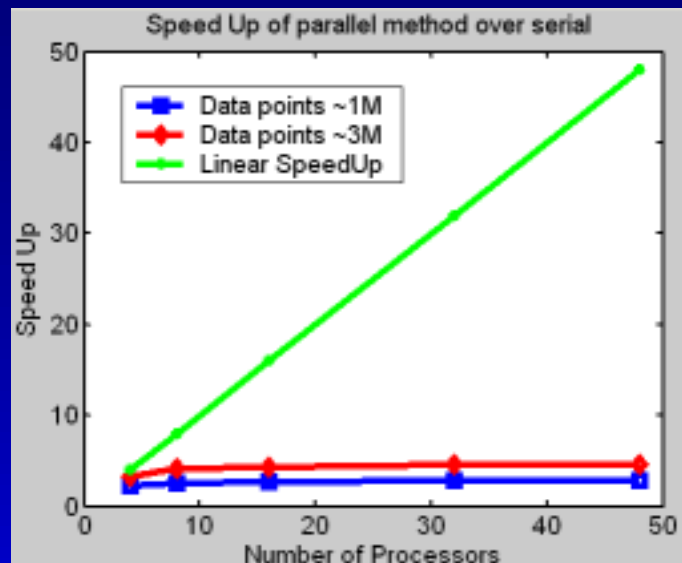
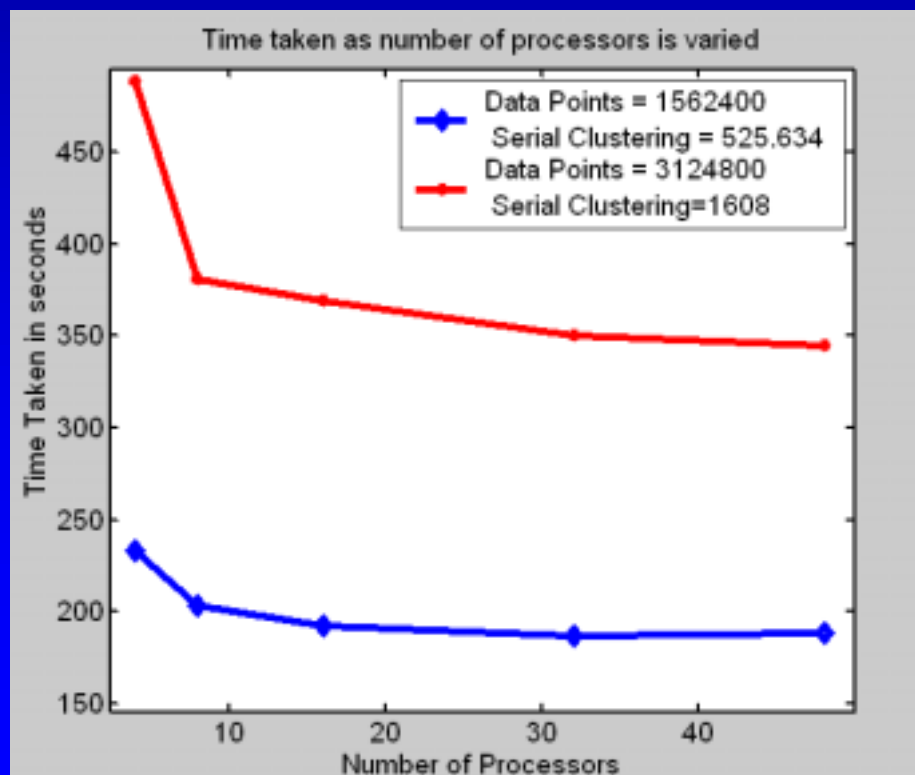
- Rank-0 needs to write 31 files to send data to other processors.

- Using MPI_Bcast instead of MPI_Send shows scalability for 32 processors also, but overall time taken is more.



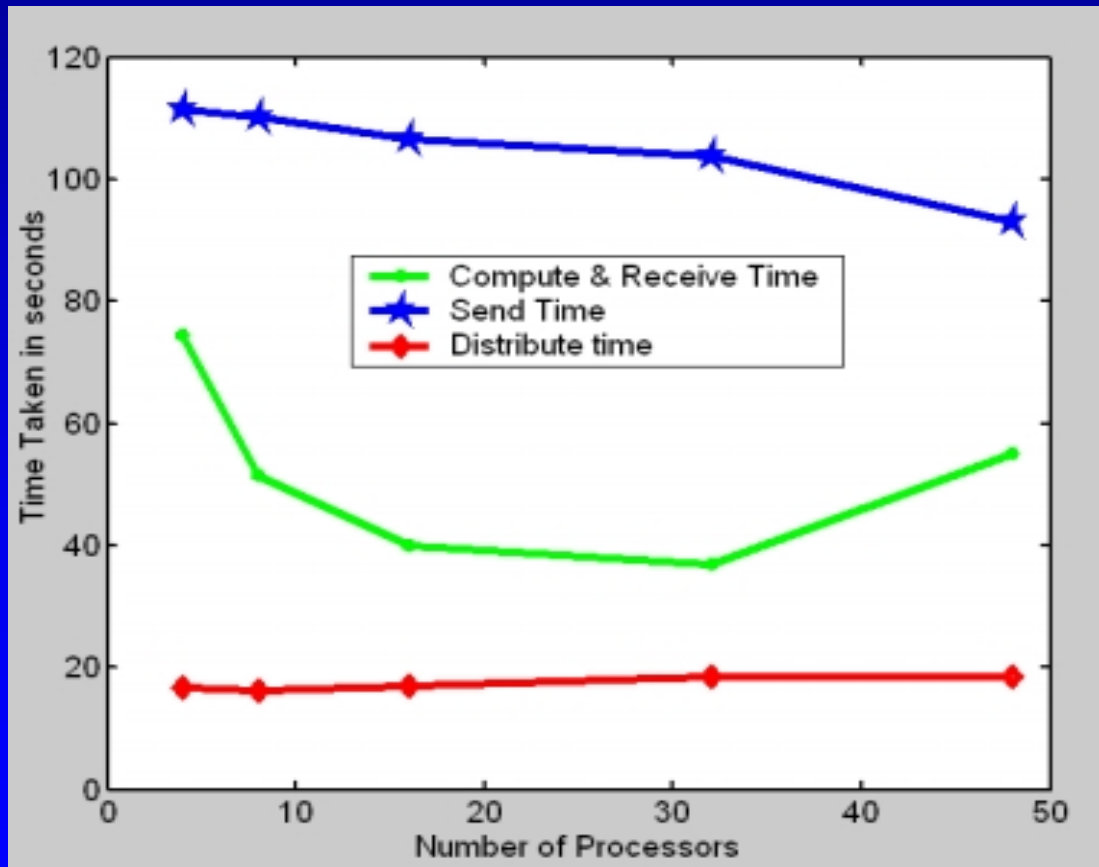
Effect of MPI_Bcast

- Time taken for parallel process decreases as number of processors is increased.
- For 3M the time taken decreases as number of processors is increased.
- Observe for ~1M
 - time taken by 48 processors > time taken by 32 processors



Why this behavior with MPI_Bcast?

- Time taken to read data from 47 processors is reduced
- Time taken to distribute the data is modestly increased.
- But Rank-0 processor receives data from 47 processors and this time increases significantly



Conclusion

- **For K-Means Clustering**
 - Speedup is observed as number of data points is increased.
 - Speedup is observed as number of centroids is increased
 - For given data size as the number of processors is increased time taken decreases only to the point that the increase in communication cost overshadows the decrease in computation cost
- **The advantage of using MatlabMPI is observed if data size is large.**

Data Mining Toolbox: Classification

Classification and Regression Tree (CART)

- **Classification Tree**
 - A tree structured classifier obtained by systematic splitting of training data samples using attribute values.
- **Regression Tree**
 - A tree structured model to predict values (get function description) of a continuous valued variable based on values of other variables.

Classification Tree

- A tree structured classifier is built in two phases:
 - 1) **Growth Phase** : In this the tree is built by recursively partitioning the data until a threshold condition is reached.
 - 2) **Prune Phase** : If the tree obtained in the growth phase is too large or too small then the misclassification rate will be high as compared to the right sized tree. The pruning of the tree is done to obtain a right sized tree.
- Only the Growth Phase of CART has been parallelized.

Example

- We explain the steps to build a classification tree using a smaller example.

Attr1	Attr2	Attr3	Class
0	0	0	1
0	1	0	2
1	1	0	3

- Training data
 - Classes – 3
 - Attributes – 3

- Size of training data (Elements per class)
 - Class 1 = 3
 - Class 2 = 5
 - Class 3 = 7

Sequential Classification tree

- Steps:
 1. The selection of the splits.
 2. The decisions when to declare a node terminal or to continue splitting it.
 3. The assignment of each terminal node to a class.

Selection of Splits

- **Split Question (X-attribute, C-integer value)**
 - continuous attributes : {Is X<C?}
 - categorical attributes : {Is X=C?}
 - In above example Q- {Is X=0?}
- **Split Criterion:**
Best split minimizes impurity at a node
 - eg: Gini index is given by:

$$i(t) = 1 - \sum_j p_j^2$$

where p_j is the proportion of class 'j' at node 't'.

- At a node with 'n' elements if split 'S' divides the data into S_1 (n_1 elements) and S_2 (n_2 elements)

$$gini_{split}(S) = \frac{n_1(gini(S_1)) + n_2(gini(S_2))}{n_1 + n_2}$$

- The split that maximizes $\Delta i(s, t)$ is selected to be the best split.

Splitting the main node

- Gini Index at root node

- Count matrix for each attribute

- If attribute value – 0 then data goes to left node

- Attribute –1

Value	C-1	C -2	C-3
0	3	5	0
1	0	0	7

Gini Index: $n_1=8, n_2=7$

$\text{gini}(s_1)=0.46857$

$\text{Gini}(s_2)=0$

$\text{Gini}_{\text{split}}=0.25$

- Attribute - 2

Value	C-1	C -2	C-3
0	0	5	7
1	3	0	0

Gini Index: $n_1=12, n_2=3$

$\text{gini}(s_1)=0$

$\text{gini}(s_2)=0.486$

$\text{Gini}_{\text{split}}=0.388$

- Attribute – 3

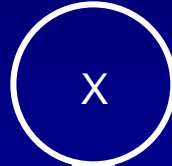
Value	C-1	C -2	C-3
0	3	5	7
1	0	0	0

No use splitting with this attribute since $n_2=0$

- The best splitting attribute is 1 since it has minimum gini index.

Split Tree

Complete
Training
Data Set



Attribute 1 = 0

Attribute 1 = 1

Class	1	2
Count	3	5



Class	3
Count	7

Attribute 2 = 0

Attribute 2 = 1

Class	2
Count	5



Class	1
Count	3

Serial Growth Phase - contd.

- **Decision to stop splitting**
 - A node is decided to be a terminal node if the Gini index is lower than a threshold.
 - Splitting is stopped at node 't' if

$$\max_{s \in S} \Delta i(s, t) < \beta$$

Or if the node is pure (as in above example.)

- **Assign class to each terminal node.**
 - Class j is assigned to terminal node $t \in \tilde{T}$ if

$$j = \arg \max_i p(i | t)$$

Parallel CART

(For Categorical Attributes)

1. Suppose the size of the given data set is N and number of processors is P .
2. The rank-0 processor
 - Reads the training data
 - Distributes the data equally among all the processors.
3. All other processors
 - Calculate and send the count matrices for all attributes.
4. Rank-0 processor
 - Receives count matrices
 - Finds best splitting attribute

Parallel CART – contd.

5. Rank-0 process

- Stops if all terminal nodes are pure.
- Else sends best splitting attribute to all other processors.

6. All other processors

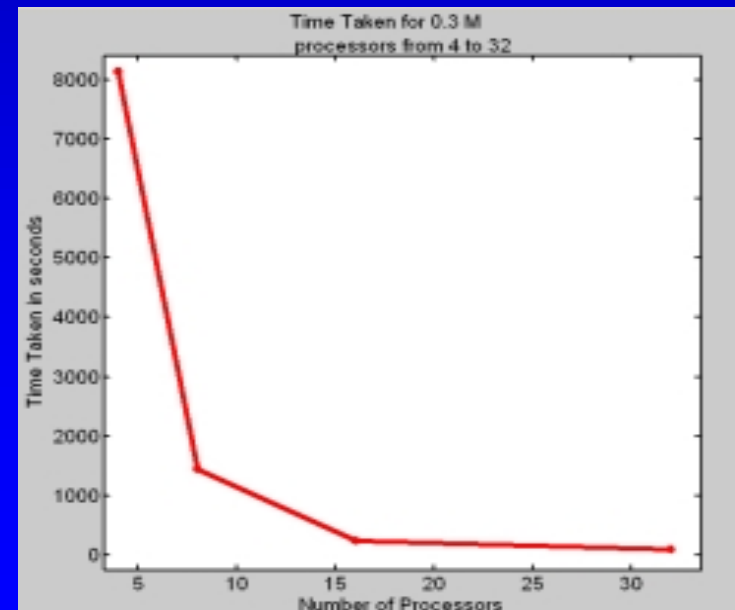
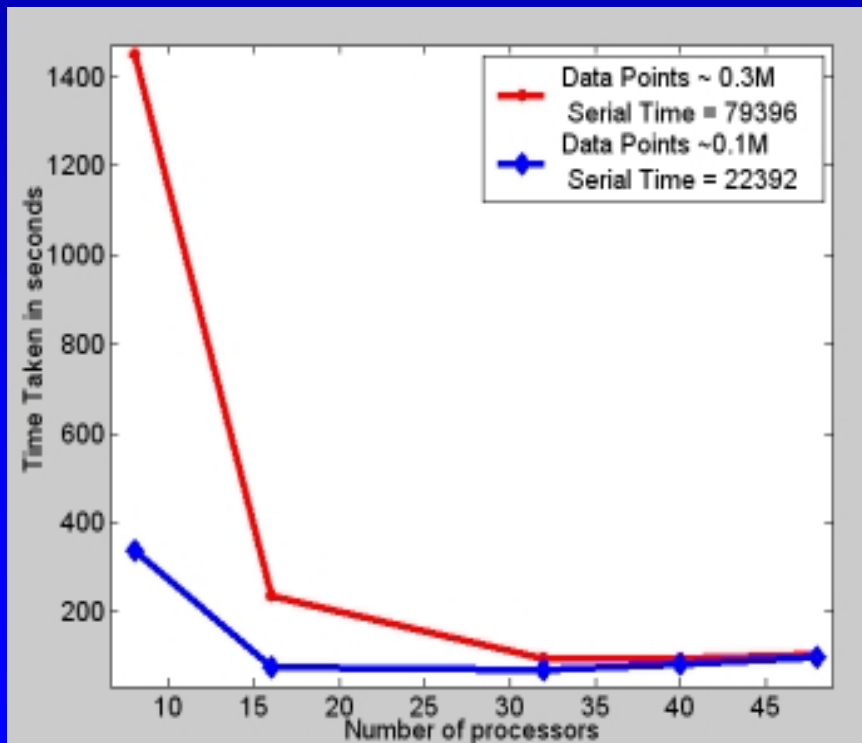
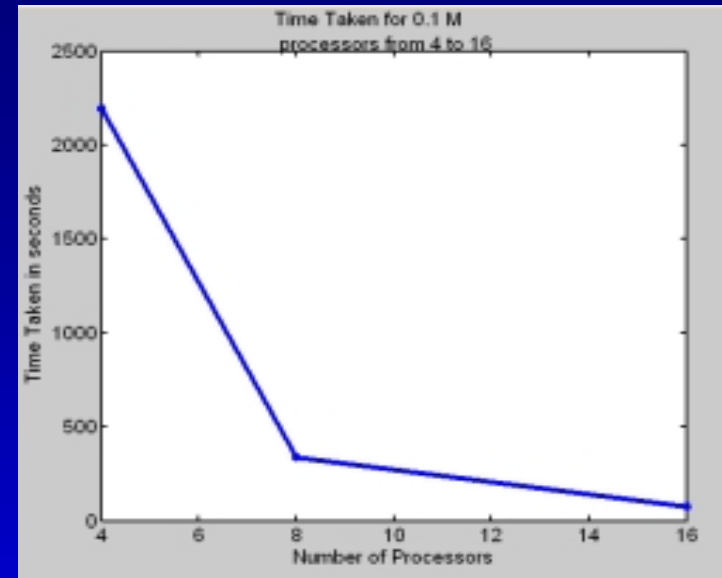
- Split the data into the left and right node using the best splitting attribute.
- Steps 3-6 are repeated for each of the leaves.

Effects Of Parallelization of categorical CART

- We studied the performance of the parallel algorithm with the variation in number of processing elements.
 - As the number of processors is increased the number of training samples per processor decreases.
 - Time taken per processor decreases hence total time taken decreases.

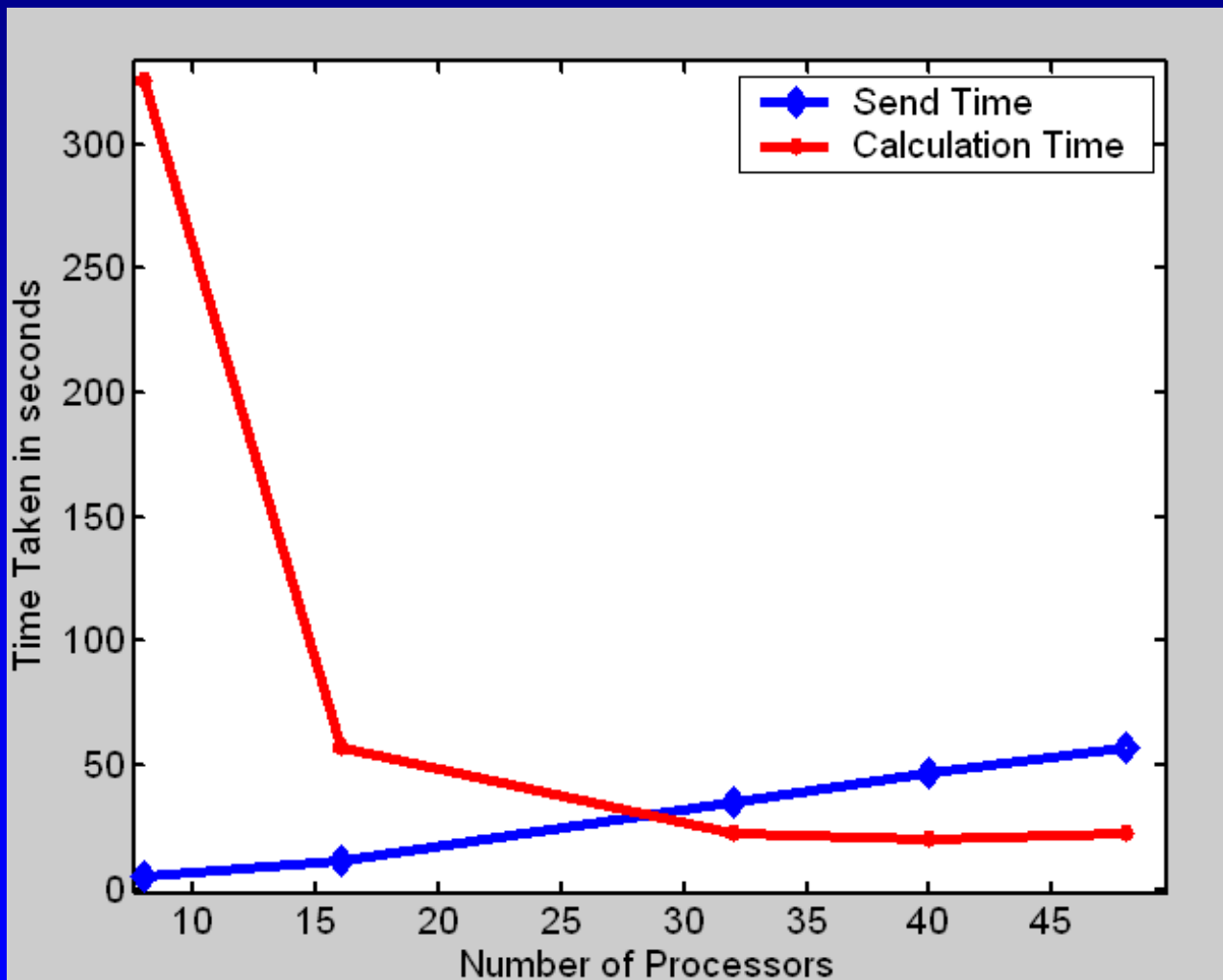
Scalability Results

- Time taken to get classification tree using 0.3M and 0.1M training data points. Number of attributes: 7 Number of classes: 10
- Serial process takes very long.
 - **For 0.3M data points with 32 processors, speedup is about 845**
- But for number of processors greater than 32 time taken increases



Reason For Increase In Time

Increase in time taken to send messages is greater than decrease in computation time



Conclusions

- **Parallel processing takes less time than serial process**
- **For large data sizes the increase in communication cost is less than the decrease in calculation cost.**
- **Parallel CART using MatlabMPI can be used with very large data sets**

Future Work

- **Optimize the use of MPI_Bcast.**
- **Generalize CART algorithm for continuous type of attributes.**
- **Parallelize Prune Phase.**
- **Add Support Vector Machines to the parallel data mining toolbox.**