Concurrent k-means Algorithm

COSC 6490A
Miroslaw Kuc
York University Apr. 7, 2009

Data and Clustering

Scientific experiments generate a lot of data (e.g. through use of microarrays for genetic analysis).

It is difficult to analyze all that data manually (e.g. to find groups of genes with similar structure).

Trying all possible group assignments can be very time consuming. For example, with 25 object and 4 possible clusters can have 4.69×10^{13} possible assignments [1].

Therefore, “brute force” approach is not realistic.

Data and Clustering

The answer: Clustering Algorithms

Trying to maximize similarity within clusters and minimize similarity between clusters.

Variety of difference/distance function used (e.g. Euclidian)

Similarity measure: Root Mean Square

Outline

1. k-means Algorithm
2. “Distributed Memory” k-means Algorithm
3. “Shared Memory” k-means Algorithm
4. Analysis

Outline

1. k-means Algorithm
2. “Distributed Memory” k-means Algorithm
3. “Concurrent Memory” k-means Algorithm
4. Analysis

k-means Algorithm

First described by J. MacQueen in 1967 [2].

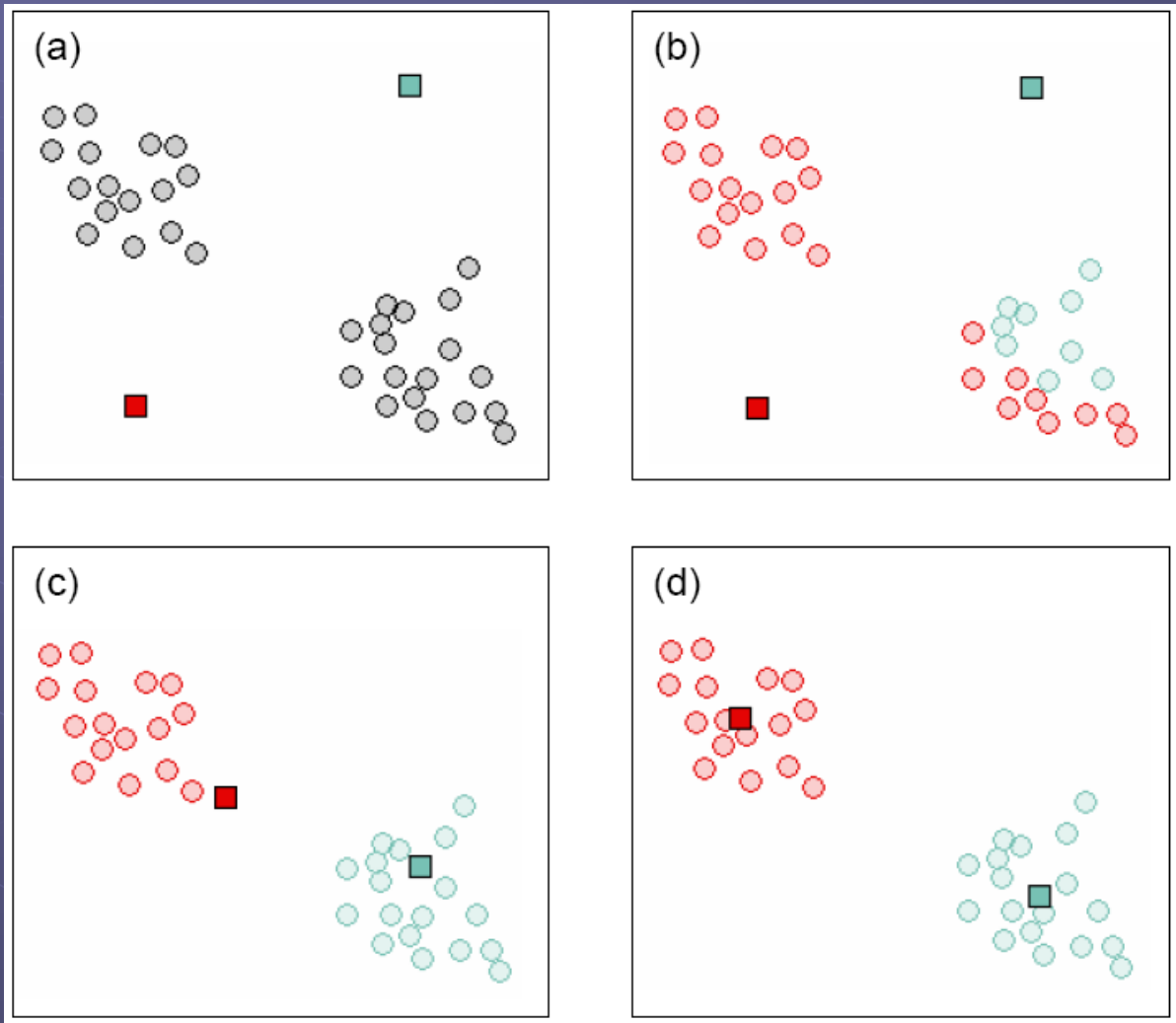
“Unsupervised” learning, number of clusters and the cluster membership is unknown.

Works with a pre-assigned “k” number of clusters
(typically have to try a range of values).

k-means Algorithm

1) Randomly assign cluster centers (e.g. select random points from within the dataset)
2) For each point calculate the distance to the cluster centers and assign the point to the closest “cluster”.
3) Based on the membership calculated in step (2) calculate the center of the new clusters.
4) Repeat steps (2) and (3) until stopping criteria are met (e.g. no point change cluster membership).

k-means Algorithm



k-means Algorithm

Some problems:

- Does not guarantee finding a global solution
- Does not suggest the best “k” (number of clusters)
- Sensitive to outliers

Many solutions proposed for these problems
(outside the scope of this project).

k-means Algorithm

Project:

Implement 2 methods for “parallelization” of the algorithm:

1) Distributed Memory approach

2) Shared Memory approach

Outline

1. k-means Algorithm
2. “Distributed Memory” k-means Algorithm
3. “Shared Memory” k-means Algorithm
4. Analysis

“Distributed” k-means Algorithm

Exploit the “data” parallelism [3]:

- Subdivide the data into equal “chunks”
- Process the data independently
- Bring together partial solutions

“Share-nothing” solution

Benefit: can analyze data on a network of computers for large problems

“Distributed” k-means Algorithm

Intended to run on a single computer (with multiple threads)

Considerations w.r.t. the single-thread version:

1) Each thread processes points “i mod T” where T is the number of threads. The data processed by each thread does not change from iteration to iteration. Each thread may be processing data from all clusters.
2) Some parts of the algorithm will have to be done in the “main” thread (e.g. initial assignment of the cluster centers, combining of the data).
3) The main thread will have to “broadcast” cluster centers for each iteration; treads will have to notify when done.
4) Partial calculations and combining them is relatively easy.

Outline

1. k-means Algorithm
2. “Distributed Memory” k-means Algorithm
3. “Shared Memory” k-means Algorithm
4. Analysis

“Shared” k-means Algorithm

Exploit the process parallelism:

- Data stored in a single location
- Requires mutual exclusion to prevent overwriting each other’s solutions
- Overall solution is built up by all threads

Benefit: some statistics may require all of the data

“Shared” k-means Algorithm

Intended to run on a single computer (with multiple threads)
Considerations w.r.t. the single-thread version:

1) Mutual exclusion for individual points, “processed” flag.
2) Mutual exclusion for location for solutions/statistics.
3) May “spawn” additional threads to update the solutions/statistics instead of waiting for locations to be available.
4) All points processed “in sequence”; therefore, when last point processed and all threads are reach the end of the array, then all points have been processed.
5) Requires less single-treaded operation for combining of the results.

Outline

1. k-means Algorithm
2. “Distributed Memory” k-means Algorithm
3. “Shared Memory” k-means Algorithm
4. Analysis

Analysis

May be beyond the scope of the project

Find out the benefits of parallelization, degree of overhead introduced by the “shared memory” approach

Will require many runs with randomly generated datasets to allow varying:

1) Total number of points
2) Number of clusters
3) Number of points per cluster
4) Cluster shape (circular, elongated) and “extent”
5) Cluster data distribution (normal, uniform, etc)
6) Dimensionality of the data

References

[1] Steinley D. *Local Optima in K-Means Clustering: What You Don't Know May Hurt You*, Psychological Methods 2003, Vol. 8, No. 3, 294-304.

[2] MacQueen J. *Some methods of classification and analysis of multivariate observations*, Proceedings of the fifth Berkeley symposium on mathematical statistic and probability (Vol. 1, pp. 281-297) Berkeley: University of California Press.

[3] Kraj P., Sharma A., Garge N., Podolsky R., McIndoe R. *ParaKMeans: Implementation of a parallelized K-means algorithm suitable for general laboratory use*, BMC Bioinformatics 2008 9:200

References

[4] Jin R., Agrawal G. *Shared Memory Parallelization of Data Mining Algorithms: Techniques, Programming Interface, and Performance, Extended Abstract*, Department of Computer and Information Sciences, University of Delaware, Newark, DE

Thank You!

Questions?