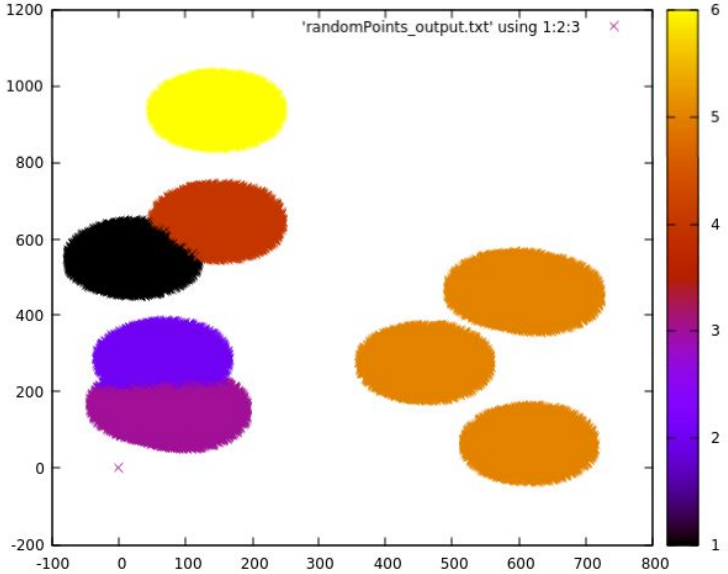
Using Parallelism to Speed Up Cluster Validation

The Purpose of Clustering:

- “…clustering analyzes data objects without consulting class labels.” Data Mining: Concepts and Techniques.
- Associates unlabeled (unclassified) data into groups based on some distance measure.
- Distance, proximity, close-ness and similarity all measure roughly the same thing and there are many ways to measure distance.
- There are *many* decisions to make when clustering data.

Cluster Validity:



6-Means clustering of 100,000 points generated around 10 random centers.

- How can we tell that a clustering is “good”?
- Judgement is easy in low dimensional data but much harder when visualization is more difficult.
- Validation methods can either be external (which rely on external knowledge) or internal (which only relies on the measures from the data)

Dunn Index:

- First proposed by JC Dunn in the Journal of Cybernetics in 1974.
- Indicates the ratio between the minimum distance between any two clusters and the the maximum of the mean distance between points in any cluster.
- **High value** indicates well separated clusters.
- Because it relies on a “max value in the denominator” it is very sensitive to one bad cluster.
- □(Ci, Cj) is the inter-cluster distance measure.
  - Center to center for this presentation.
- Δk is the max distance between 2 points is a cluster.
  - Can be other measures.

$$DI_m = \frac{\min\limits_{1 \leqslant i < j \leqslant m} \delta(C_i, C_j)}{\max\limits_{1 \leqslant k \leqslant m} \Delta_k}$$

Davies-Bouldin Index:

- Defined by David Davies and Donald Bouldin in 1979 in *IEEE Transactions on Pattern Analysis and Machine Intelligence.*
- Represents a ratio between distances between points to centroids and centroids to centroids.
- **Low values** indicate strongly dissimilar clusters.
- S_i is the mean of all distances from a point to its own centroid i.
- $M_{i,j}$ is the distance between cluster i and j.
- N is the number of clusters.

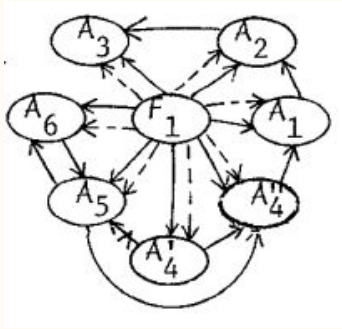
$$R_{i,j} = \frac{S_i + S_j}{M_{i,j}}$$

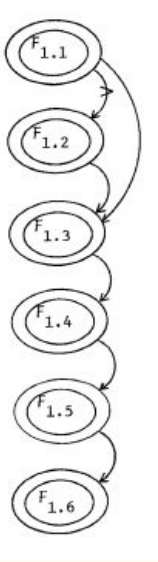
$$D_i \equiv \max_{j \neq i} R_{i,j}$$

$$DB \equiv \frac{1}{N} \sum_{i=1}^{N} D_i$$

Parallelizing These Approaches:

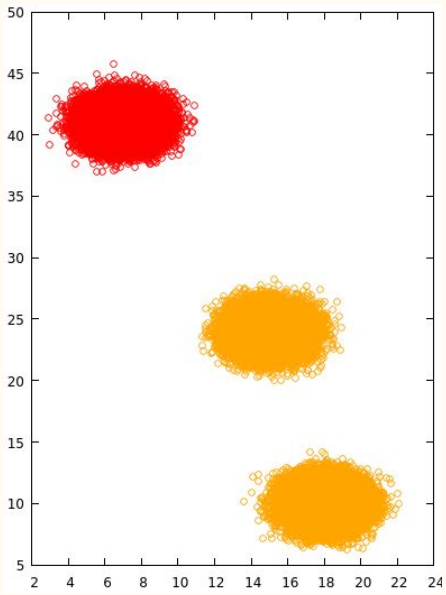
- Seperate loops to minimize dependencies.
- Introduce OpenMP
- Merge parallel executions as necessary to maintain data integrity.
- All code was compiled with gcc at at -O0.

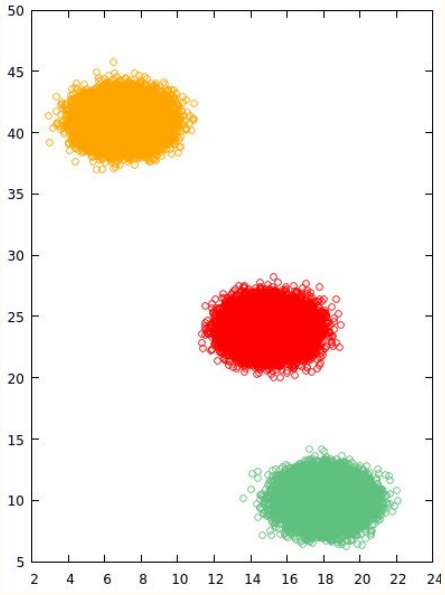




Test Data Set 1:

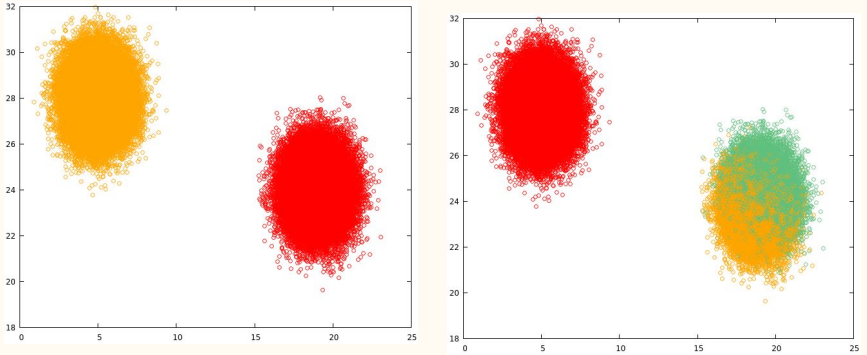
- 100,000 points
- 2 dimensions
- Generated in gaussian distributions around 3 centers.
- 2 test clustering:
  - k=2
  - k=3



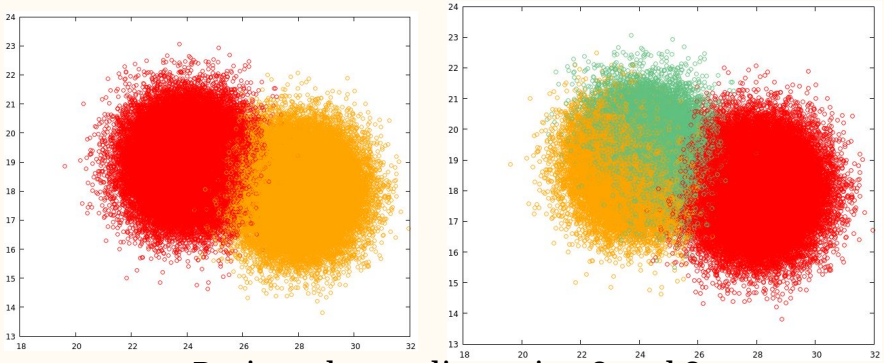


Test Data Set 2:

- 100,000 points
- 10 dimensions
- Generated in gaussian distributions around 2 centers.
- 2 test clustering:
  - k=2
  - k=3



Projected onto dimension 1 and 2

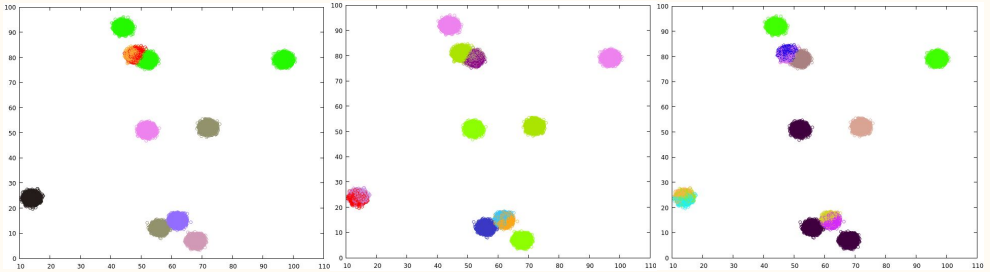


Projected onto dimension 2 and 3

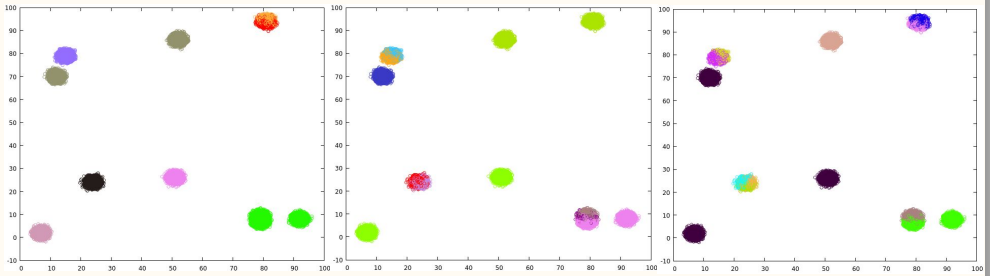
Test Data Set 2:

- 100,000 points
- 10 dimensions
- Generated in gaussian distributions around 10 centers.
- 3 test clustering:
  - k=8
  - k=10
  - k=12

8 clusters 10 clusters 12 clusters



Projected onto dimensions 1 and 2

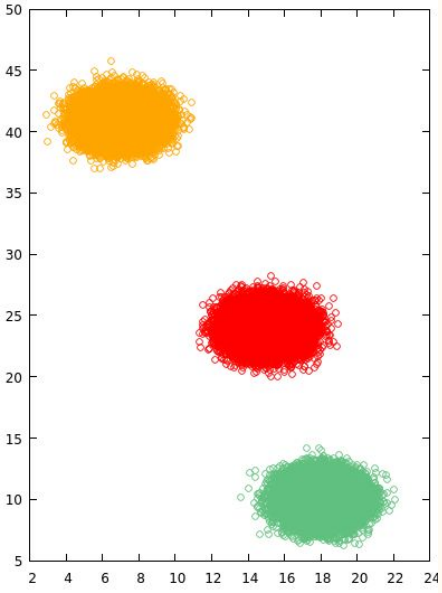


Projected onto dimensions 2 and 3

Set 1 Results:

	2 clusters	3 clusters
Dunn Index	2.287	2.988
DB Index	0.164	0.084

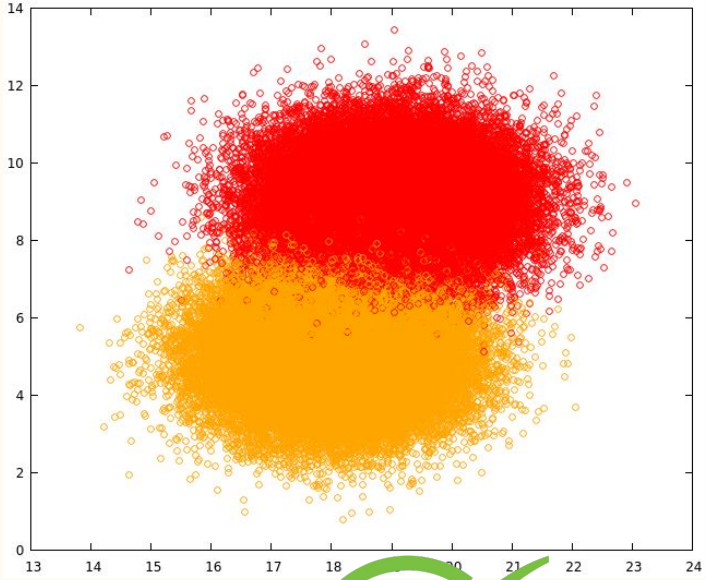
Best Fit



Set 2 Results:

	2 clusters	3 clusters
Dunn Index	3.668	0.239
DB Index	0.124	1.308

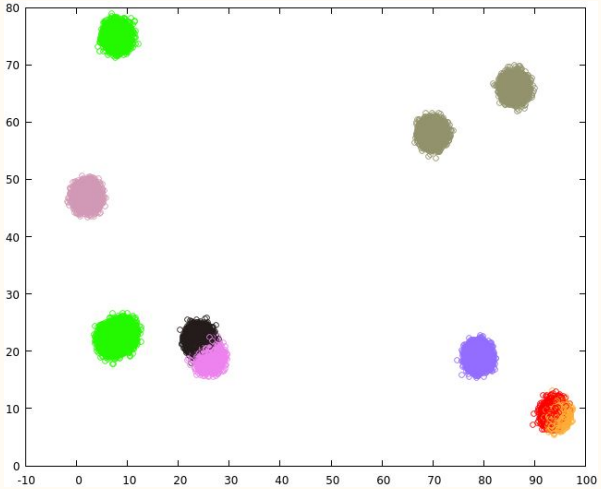
Best Fit



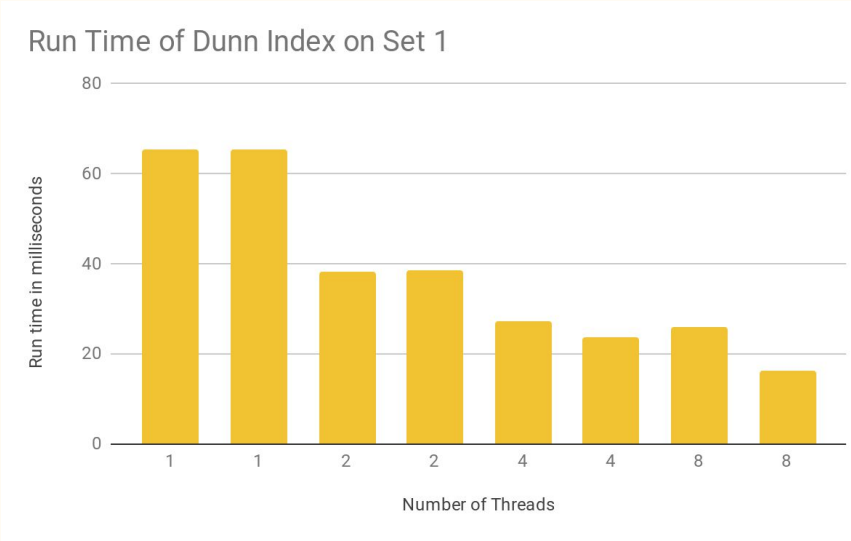
Set 3 Results:

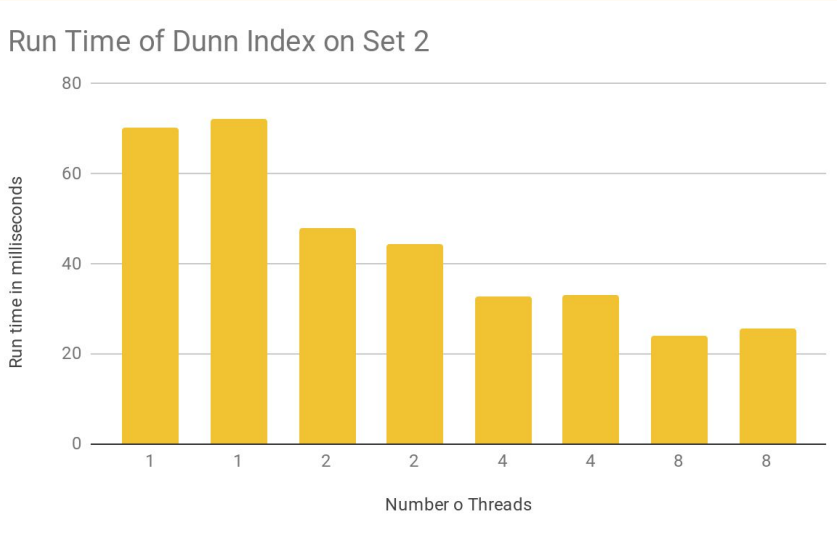
	8 clusters	10 clusters	12 clusters
Dunn Index	0.021	0.025	0.027
DB Index	0.674	1.5	1.327

Best Fit

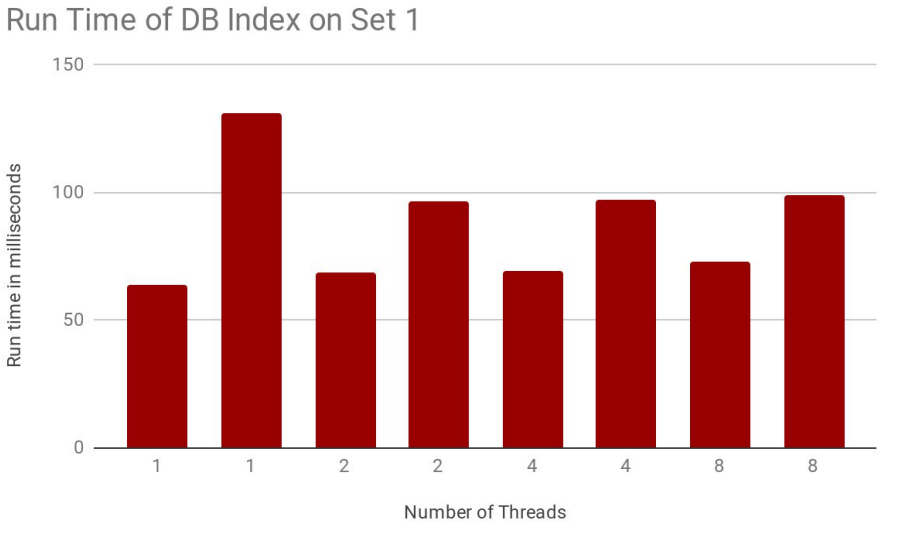


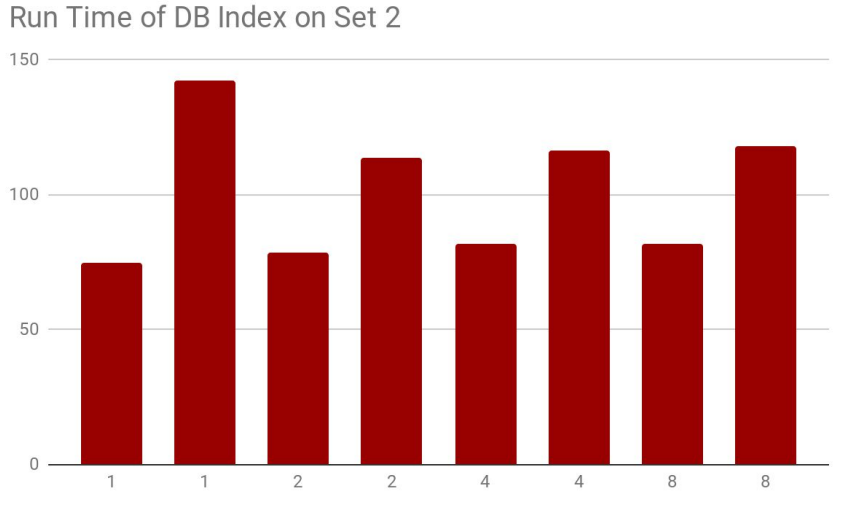
Dunn Index Performance:





Davies-Bouldin Index Performance:



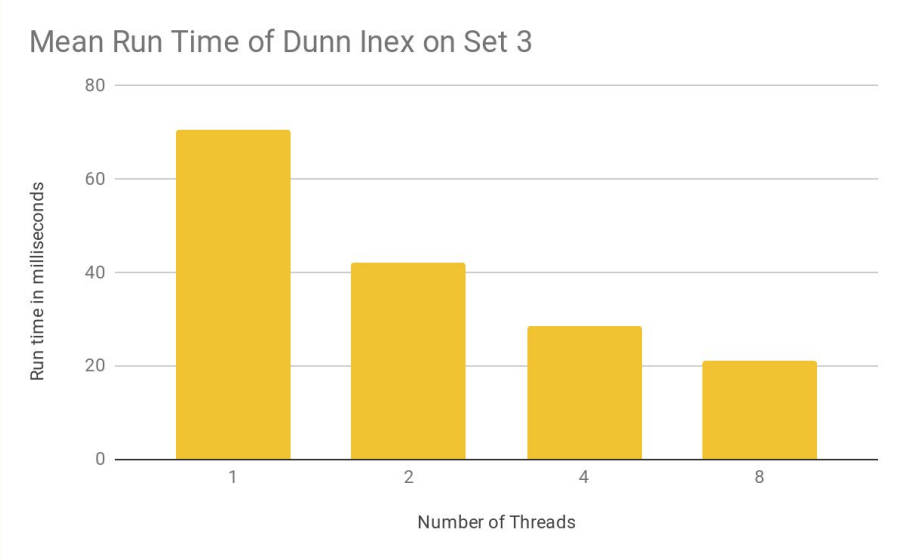


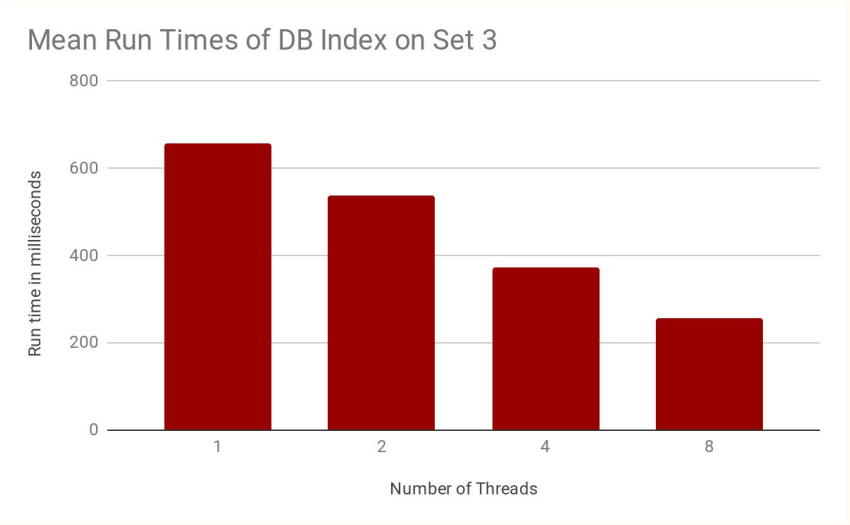
What's Wrong with DB?

- Outer loop iterating across all clusters means the benefit of parallelism is lost.

```
#pragma omp parallel for
for (i = 0; i < numClusters; ++i){ //divide sum by points to get centers
    centers[i].values = new float[dimensions];
    int j;
    for (j = 0; j < dimensions; ++j){
        centers[i].values[j] = (sums[i][j] / totalPoints[i]);
    }
    centers[i].cluster = i+1;
}
```

Set 3 Results:



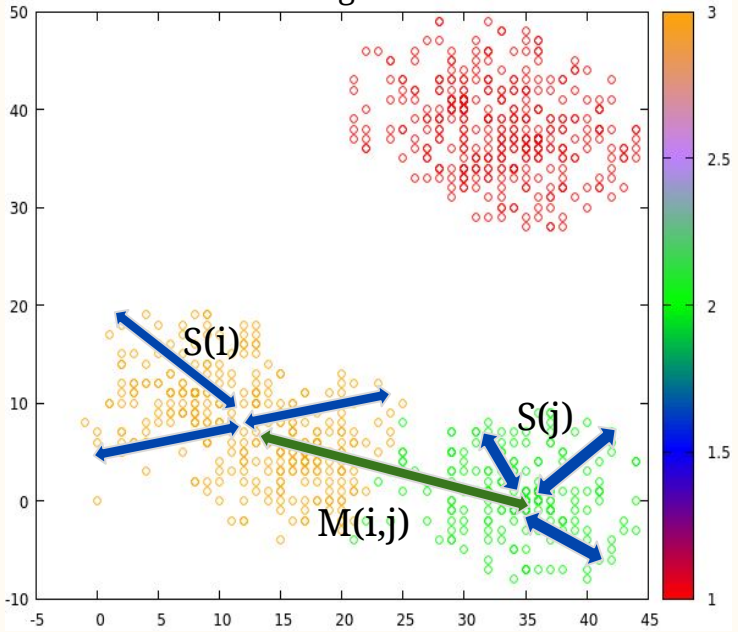


Run times averaged across three runs for three values of k display the expected increase in performance.

Applications:

- Performance improvement will be multiplied in hierarchical systems and the results will be more meaningful.
- Non-deterministic clustering algorithms could have several runs evaluated quickly and easily.
- Selecting a clustering algorithm is often a very difficult problem. More powerful and convenient tools to evaluate clusters makes the problem easier.

Calculating a DB index:



Future Work and Refinements:

- Re-run tests on distinct architectures.
- Re-run tests with LLVM.
- Continue to refactor code to derive more benefit from parallelism.

Thank you for your attention.

Questions or Comments?