Image Segmentation using OpenMP

Aniket R Rane
CSE 702
Instructor – Dr. Russ Miller

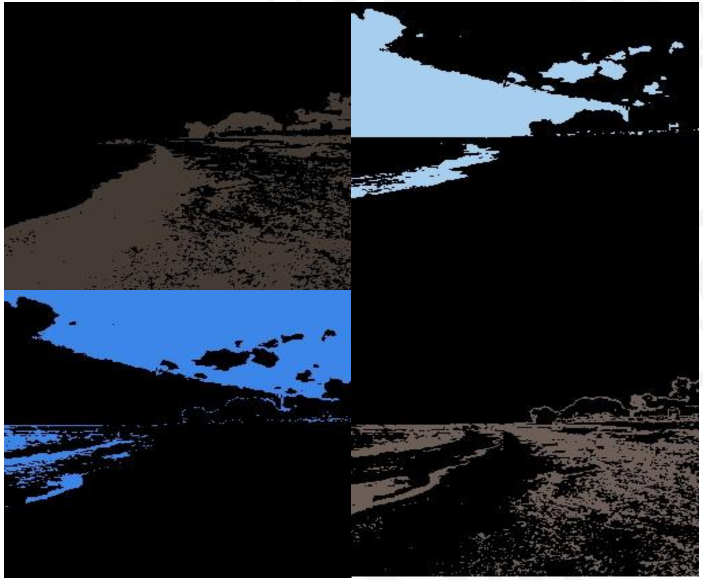
University at Buffalo
Department of Computer Science
and Engineering
School of Engineering and Applied Sciences

OUTLINE

1. Proposed Project
2. K Means
3. Clustering
4. Parallel Model
5. Results
6. Inferences
7. Conclusion

Proposed Project

- Image Segmentation using K-means : Break up the image into meaningful or perceptually similar images.



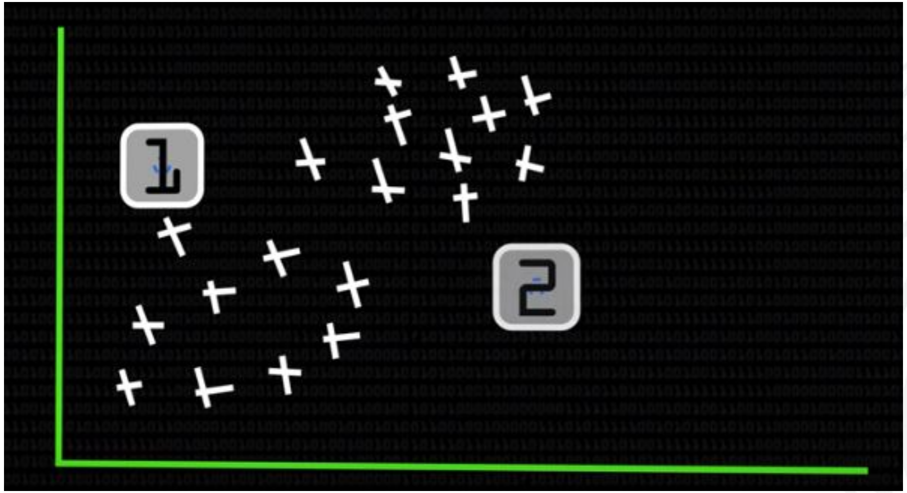
Clustering

- Partitioning of data
- Similar elements placed in same cluster. Similarity is calculated based on some distance metric such as Euclidean distance
- Unsupervised Learning – Useful – Don’t Know What you’re looking for
- Requires data, but no labels
- Types
    - Partition Algorithms
    - Hierarchical Algorithms

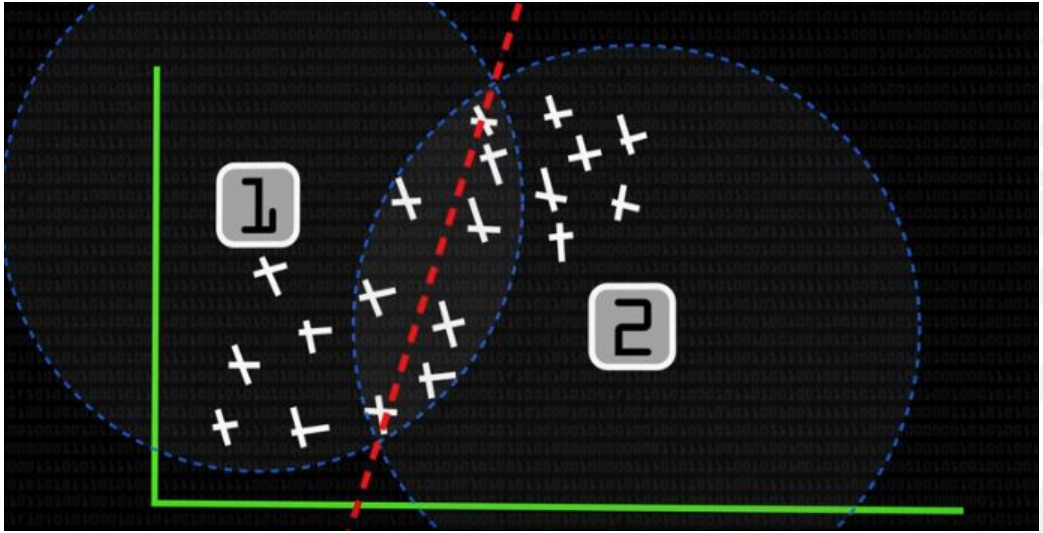
K-Means

1. Select k i.e. the number of clusters
2. Use a strategy to select k points to be cluster centers.
3. Put each point in the data set in the cluster which has its center closest to the point
4. Calculate new cluster centers by taking means of all points in a cluster
5. Repeat 3 and 4 until convergence

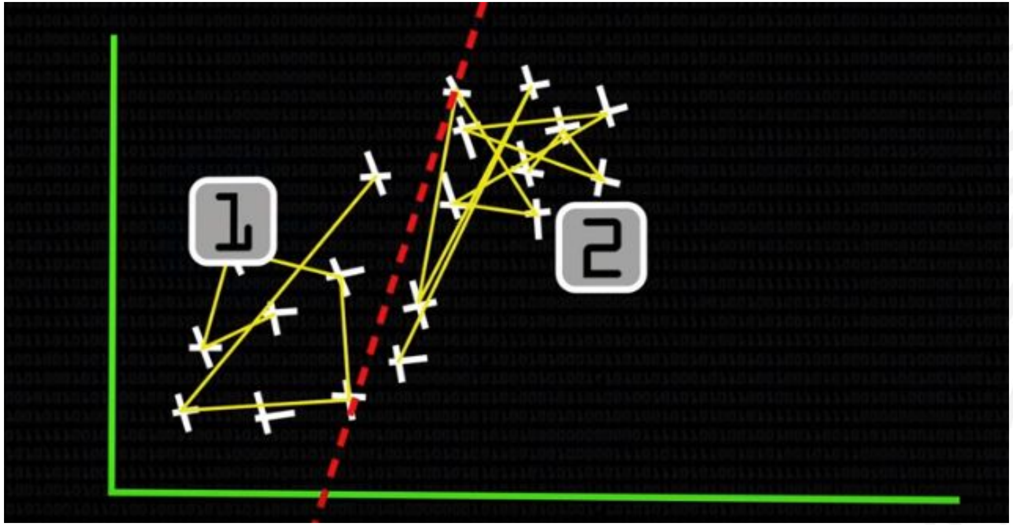
K = 2



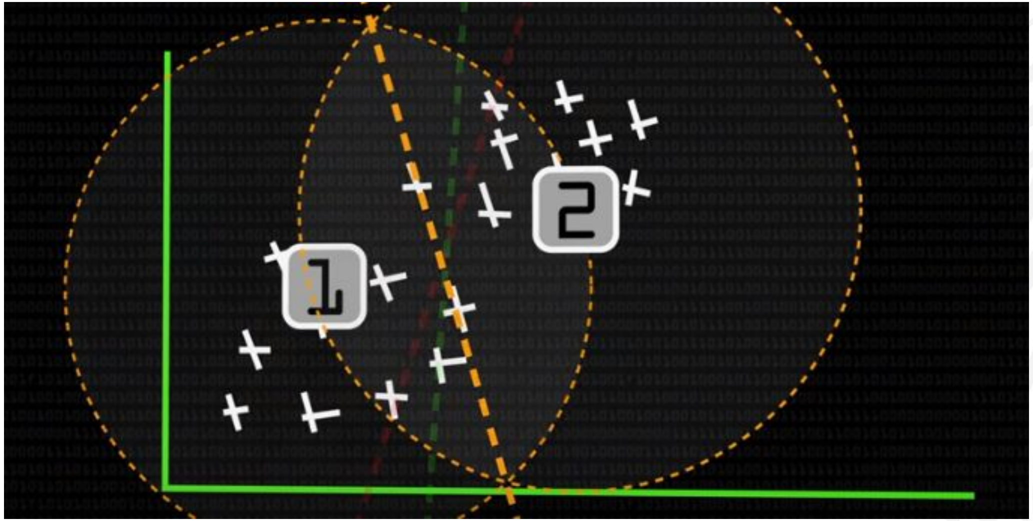
K = 2



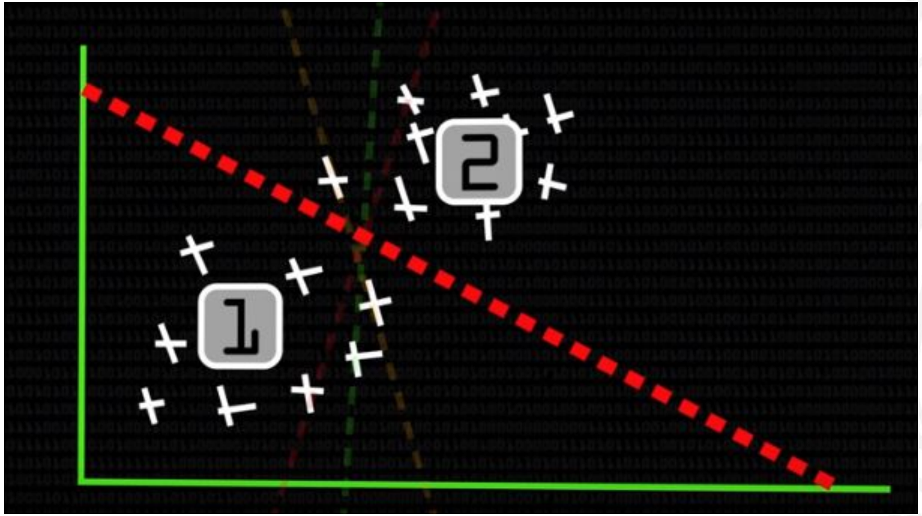
K = 2



K = 2



K = 2



Parallel Implementation – Image to Dataset

- Read the image using OpenCV for Python.
- Append the R, G, and B values of the pixels to a string one by one.
- Saving the string to a .txt file.

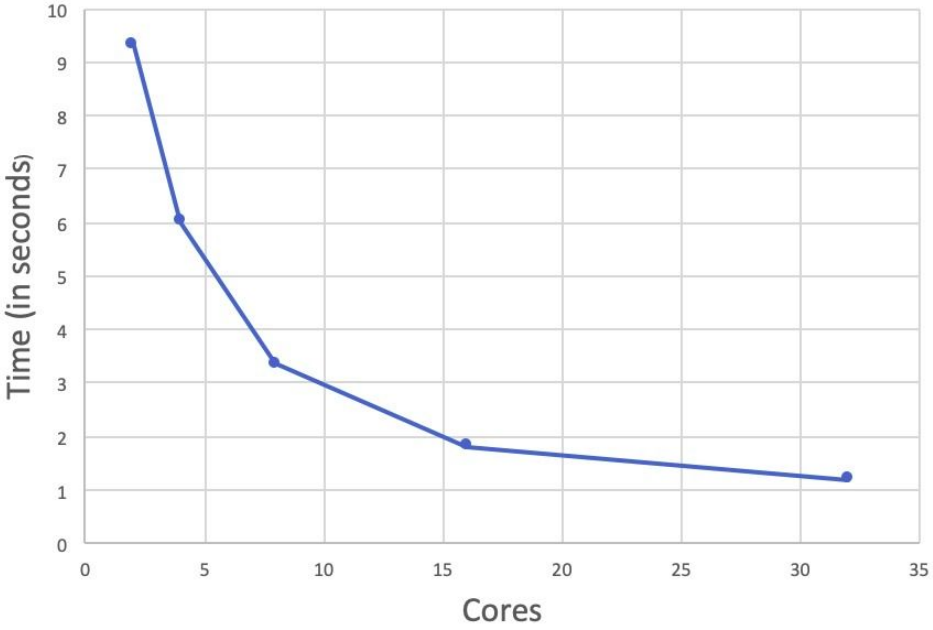
Parallel Model

- Consider N data points and P cores..
- Assign N/P data points to each core using the text file.
- Core 0 randomly chooses k points as cluster centroids.
- Each core for each of its points, finds the cluster to which the point belongs.
- Recalculate local sums for each cluster in each core.
- Add all local sums for each core to find global means.
- Repeat the clustering for number of iterations.
- Save the cluster means of the final iteration.

Parallel Implementation – Independent Images of Clusters

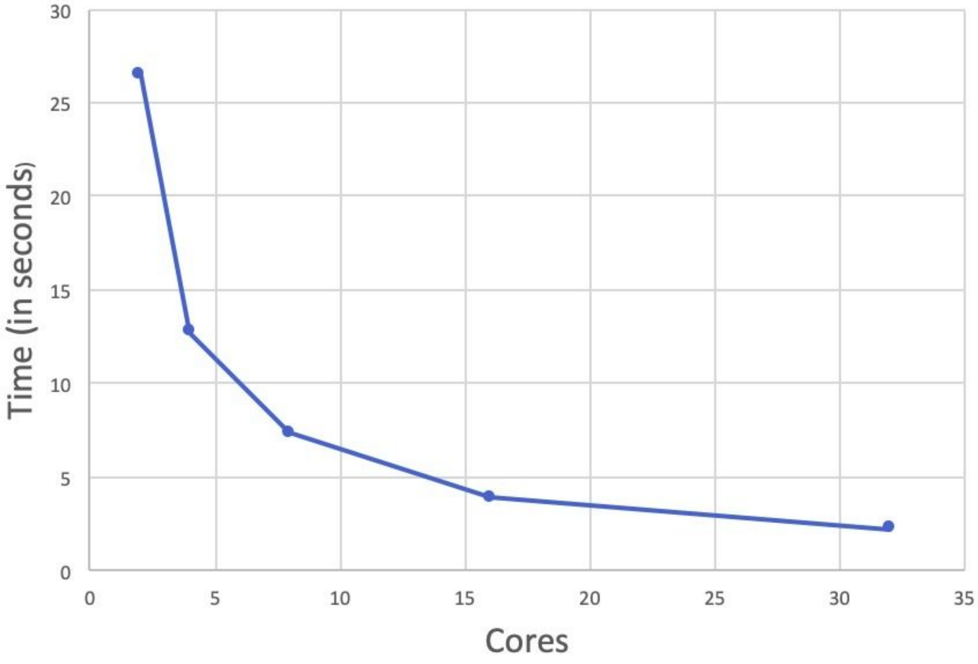
- Read the file with final cluster means.
- Read the image.
- For each pixel, determine the cluster it belongs to and form a different image for every cluster with pixel values equal to the respective cluster means.
- Save the resulting images.

Results
3 Cluster 20 Iterations



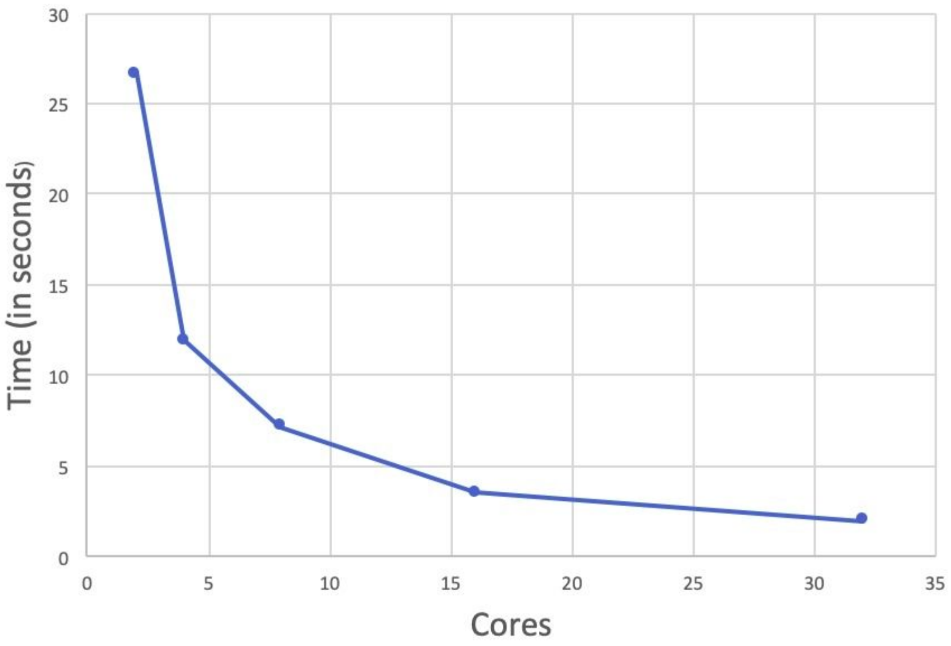
Number of threads	Time in seconds
2	9.30
4	6.01
8	3.35
16	1.81
32	1.2

3 Clusters 40 Iterations



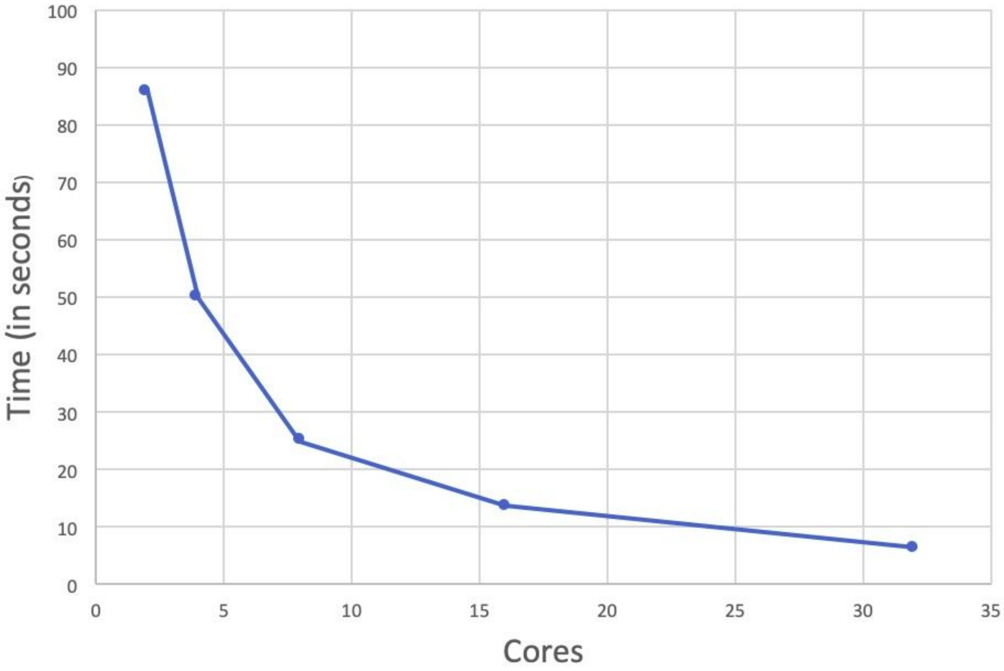
Number of threads	Time in seconds
2	26.42
4	12.76
8	7.35
16	3.88
32	2.2

4 Clusters 20 Iterations



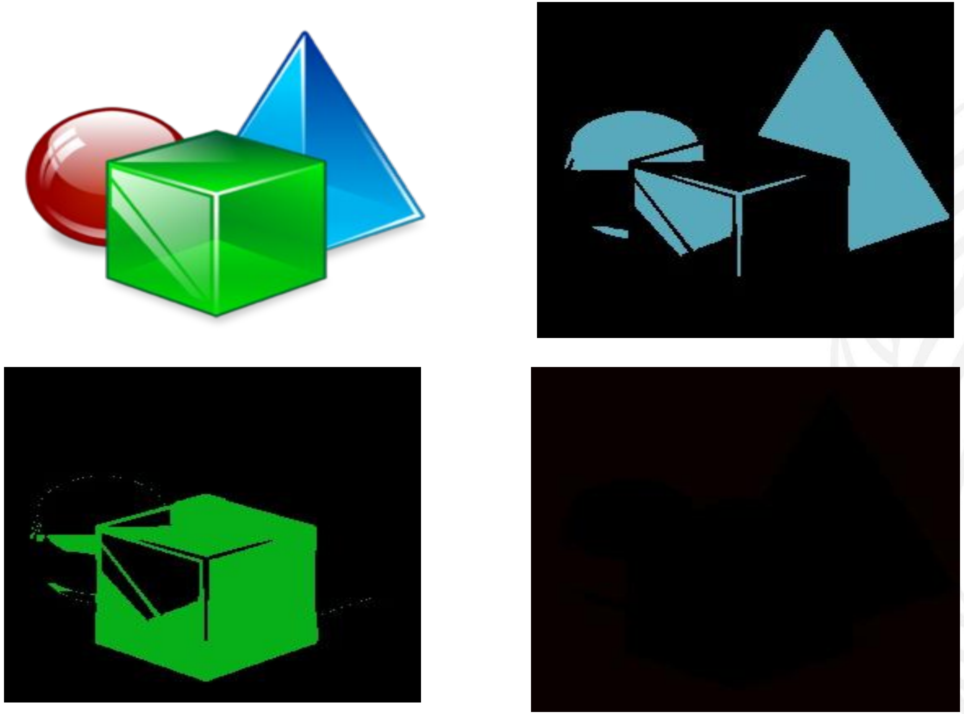
Number of threads	Time in seconds
2	26.64
4	11.92
8	7.15
16	3.55
32	1.99

4 Clusters 40 Iterations

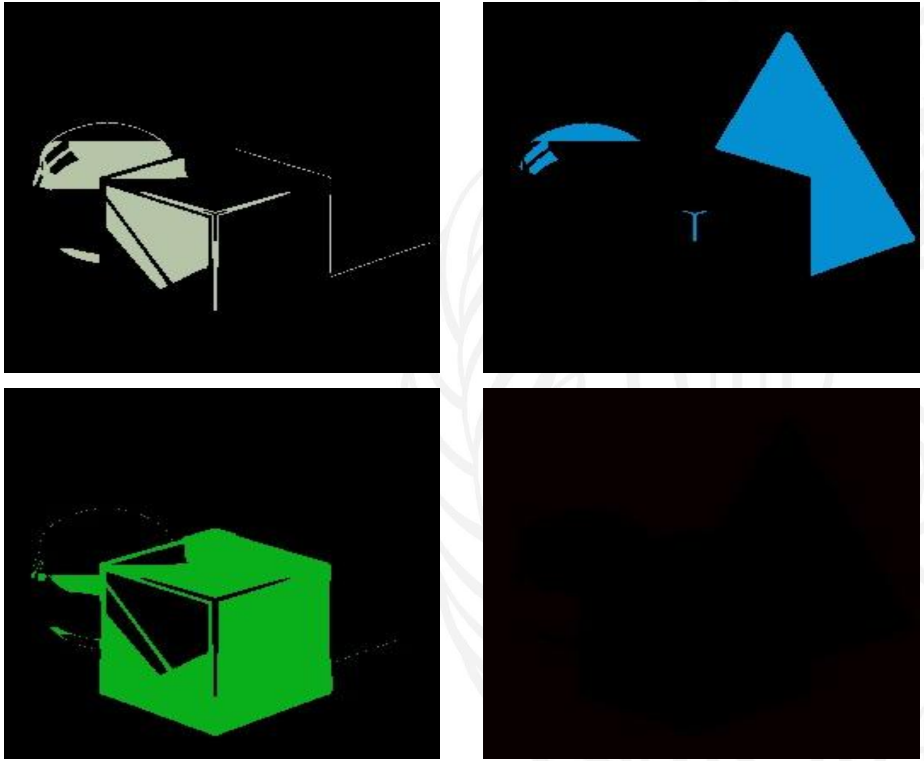


Number of threads	Time in seconds
2	85.74
4	49.8
8	24.85
16	13.58
32	6.29

Independent 3 Clusters



Independent 4 Clusters



Inferences

- Significant speedup observed only up to 16 cores.
- Number of clusters has a big impact on the image segmentation results.
- Convergence is better after 30 iterations.

Challenges

- The input text files and output images had to be created serially.
- Images compatible with K Means.

References

- Algorithms Sequential & Parallel: A Unified Approach (Dr. Russ Miller, Dr. Laurence Boxer)
- https://ubccr.freshdesk.com/support/solutions/articles/13000026245-tutorials-and-training-documents (Dr. Matthew Jones)
- https://www.openmp.org/wp-content/uploads/omp-hands-on-SC08.pdf
- http://people.csail.mit.edu/dsontag/courses/ml12/slides/lecture14.pdf
- Stackoverflow for general OpenMP questions