BERKELEY LAB

LAWRENCE BERKELEY NATIONAL LABORATORY

U.S. DEPARTMENT OF ENERGY

Distributed Memory Parallel Markov Random Fields Using Graph Partitioning

Colleen Heinemann, Talita Perciano,

Daniela Ushizima, Wes Bethel

December 11, 2017

Overview

- What is MRF-based image segmentation?
  - Why is it a challenging and relevant problem?
- Distributed-memory parallel MRF-based image segmentation algorithm
- How well does it perform on modern platforms?
- How can we improve it in the future?

Image Segmentation for Material Science

- Study the properties of material samples by identifying different phases
- Asset 3D architecture of materials
- Challenges: amount of data, broad variety of sensors, specific characteristics of the image data
- Current single socket approaches unable to accommodate growing data sizes
- Our approach: Markov Random Field model with graph partitioning for parallelization through MPI

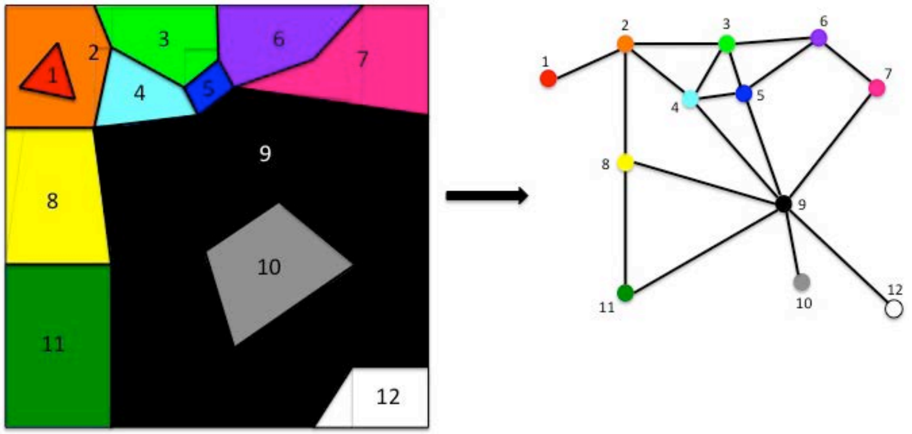
Markov Random Fields (MRF)

- MRF algorithms are accurate and capable of parallelizing
  - Use raw image and oversegmented image
- Problem: application to large data unfeasible due to NP-hard complexity
- Use graph partitioning to assist in making problem parallelizable

Markov Random Fields for Image Segmentation

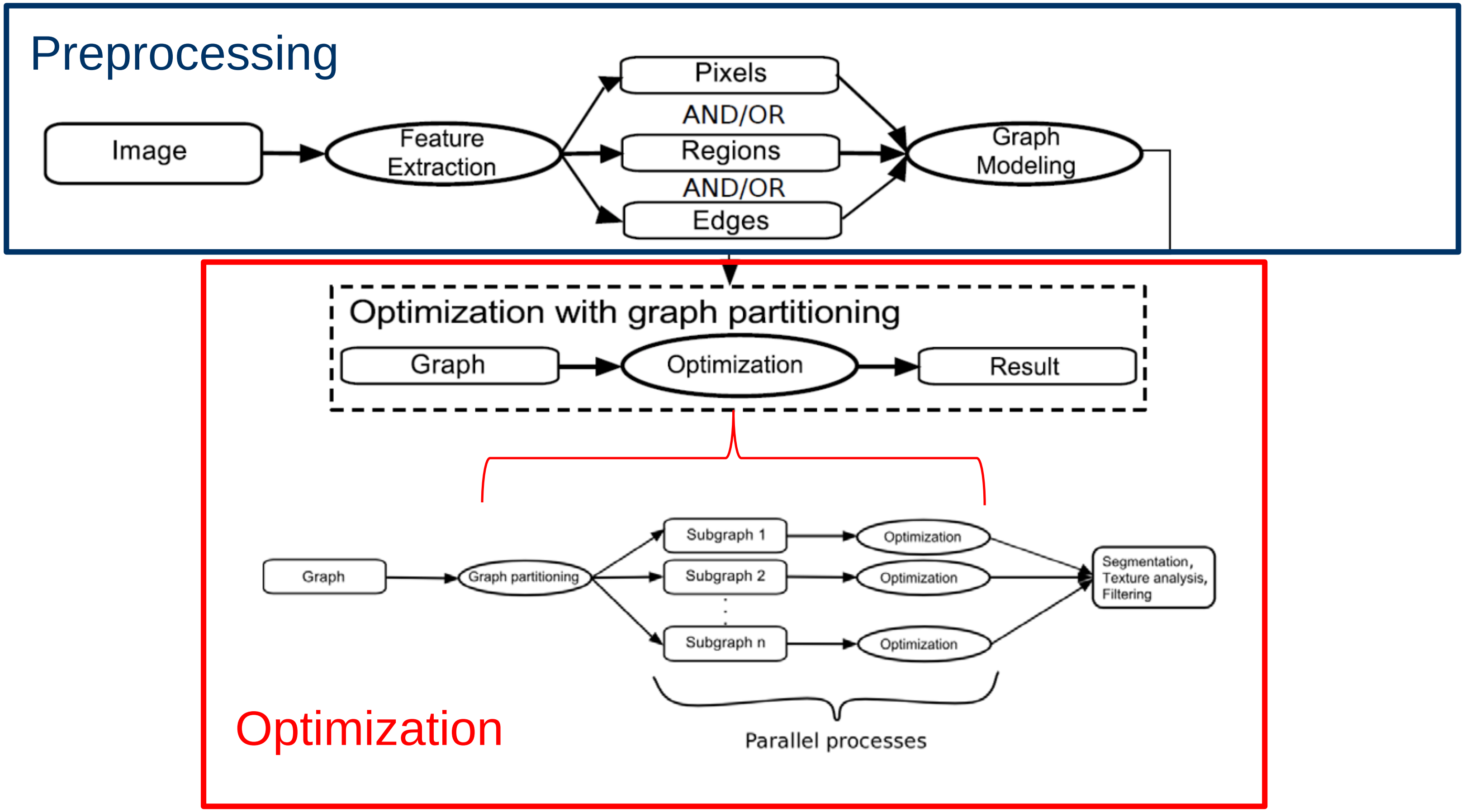
- Given an image represented by $\mathbf{y} = (y_1, \ldots, y_N)$ where each y_i is a region
- Goal: configuration of labels $\mathbf{x} = (x_1, \ldots, x_N)$ where $x_i \in L$ and L is the set of all possible labels, $L = \{0, 1, 2, \ldots, M\}$
- MAP criterion: find a labeling that satisfies:

$$\mathbf{x}^* = \underset{x}{\operatorname{argmin}} \{U(\mathbf{y}|\mathbf{x}, \Theta) + U(\mathbf{x})\}$$



Obtaining a region graph from an oversegmentation

Markov Random Fields for Image Segmentation



T. Perciano, D. Ushizima, E. W. Bethel, Y. D. Mizhahi, and J. A. Sethian, "Reduced-complexity Image Segmentation under Parallel Markov Random Field Formulation using Graph Partitioning," in *2016 IEEE International Conference on Image Processing*, Phoenix, AZ, USA, Sep. 2016, lBNL-1005703.

MPI - Parallel Markov Random Fields (MPI-PMRF)

Algorithm 1 Distributed memory version of the Markov Random Field algorithm using graph partitioning and parameter estimation (MPI-PMRF). Line 8 is run in parallel to distribute the largest amount of work to increase performance.

Input: Original image, oversegmentation, number of classes
Output: Segmented image and estimated parameters

```
 1: K ← number of classes
 2: Initialize parameters and initial labels randomly
 3: Create graph from oversegmentation
 4: for each Expectation Maximization iteration do
 5:     Divide graph into subgraphs (cliques) based on number of
        MPI processes to be used
 6:     Distribute cliques to MPI processes
 7:     for each non-zero clique of the graph do
 8:         Run Expectation Maximization and Maximum a Posteriori
            optimizations on assigned MPI processes
 9:     end for
10:     Gather parameter estimation information for subgraphs
11:     Update parameters
12: end for
13: Generate resulting output image
```

MPI - Parallel Markov Random Fields (MPI-PMRF)

Algorithm 1 Distributed memory version of the Markov Random Field algorithm using graph partitioning and parameter estimation (MPI-PMRF). Line 8 is run in parallel to distribute the largest amount of work to increase performance.

Input: Original image, oversegmentation, number of classes
Output: Segmented image and estimated parameters

```
 1: K ← number of classes
 2: Initialize parameters and initial labels randomly
 3: Create graph from oversegmentation
 4: for each Expectation Maximization iteration do
 5:     Divide graph into subgraphs (cliques) based on number of
        MPI processes to be used
 6:     Distribute cliques to MPI processes
 7:     for each non-zero clique of the graph do
 8:         Run Expectation Maximization and Maximum a Posteriori
        optimizations on assigned MPI processes
 9:     end for
10:     Gather parameter estimation information for subgraphs
11:     Update parameters
12: end for
13: Generate resulting output image
```

MPI - Parallel Markov Random Fields (MPI-PMRF)

Algorithm 1 Distributed memory version of the Markov Random Field algorithm using graph partitioning and parameter estimation (MPI-PMRF). Line 8 is run in parallel to distribute the largest amount of work to increase performance.

Input: Original image, oversegmentation, number of classes
Output: Segmented image and estimated parameters

```
1: K ← number of classes
2: Initialize parameters and initial labels randomly
3: Create graph from oversegmentation
4: for each Expectation Maximization iteration do
5:     Divide graph into subgraphs (cliques) based on number of MPI processes to be used
6:     Distribute cliques to MPI processes
7:     for each non-zero clique of the graph do
8:         Run Expectation Maximization and Maximum a Posteriori optimizations on assigned MPI processes
9:     end for
10:    Gather parameter estimation information for subgraphs
11:    Update parameters
12: end for
13: Generate resulting output image
```

MPI - Parallel Markov Random Fields (MPI-PMRF)

Algorithm 1 Distributed memory version of the Markov Random Field algorithm using graph partitioning and parameter estimation (MPI-PMRF). Line 8 is run in parallel to distribute the largest amount of work to increase performance.

Input: Original image, oversegmentation, number of classes
Output: Segmented image and estimated parameters
1: $K \leftarrow$ number of classes
2: Initialize parameters and initial labels randomly
3: Create graph from oversegmentation
4: **for** each Expectation Maximization iteration **do**
5:     Divide graph into subgraphs (cliques) based on number of MPI processes to be used
6:     Distribute cliques to MPI processes
7:     **for** each non-zero clique of the graph **do**
8:         Run Expectation Maximization and Maximum a Posteriori optimizations on assigned MPI processes
9:     **end for**
10:     Gather parameter estimation information for subgraphs
11:     Update parameters
12: **end for**
13: Generate resulting output image

MPI - Parallel Markov Random Fields (MPI-PMRF)

Algorithm 1 Distributed memory version of the Markov Random Field algorithm using graph partitioning and parameter estimation (MPI-PMRF). Line 8 is run in parallel to distribute the largest amount of work to increase performance.

Input: Original image, oversegmentation, number of classes
Output: Segmented image and estimated parameters

```
 1: K ← number of classes
 2: Initialize parameters and initial labels randomly
 3: Create graph from oversegmentation
 4: for each Expectation Maximization iteration do
 5:     Divide graph into subgraphs (cliques) based on number of
        MPI processes to be used
 6:     Distribute cliques to MPI processes
 7:     for each non-zero clique of the graph do
 8:         Run Expectation Maximization and Maximum a Posteriori
            optimizations on assigned MPI processes
 9:     end for
10:     Gather parameter estimation information for subgraphs
11:     Update parameters
12: end for
13: Generate resulting output image
```

MPI - Parallel Markov Random Fields (MPI-PMRF)

Algorithm 1 Distributed memory version of the Markov Random Field algorithm using graph partitioning and parameter estimation (MPI-PMRF). Line 8 is run in parallel to distribute the largest amount of work to increase performance.

Input: Original image, oversegmentation, number of classes
Output: Segmented image and estimated parameters
1: $K \leftarrow$ number of classes
2: Initialize parameters and initial labels randomly
3: Create graph from oversegmentation
4: **for** each Expectation Maximization iteration **do**
5: Divide graph into subgraphs (cliques) based on number of MPI processes to be used
6: Distribute cliques to MPI processes
7: **for** each non-zero clique of the graph **do**
8: Run Expectation Maximization and Maximum a Posteriori optimizations on assigned MPI processes
9: **end for**
10: Gather parameter estimation information for subgraphs
11: Update parameters
12: **end for**
13: Generate resulting output image

MPI - Parallel Markov Random Fields (MPI-PMRF)

Algorithm 1 Distributed memory version of the Markov Random Field algorithm using graph partitioning and parameter estimation (MPI-PMRF). Line 8 is run in parallel to distribute the largest amount of work to increase performance.

Input: Original image, oversegmentation, number of classes
Output: Segmented image and estimated parameters

```
 1: K ← number of classes
 2: Initialize parameters and initial labels randomly
 3: Create graph from oversegmentation
 4: for each Expectation Maximization iteration do
 5:     Divide graph into subgraphs (cliques) based on number of
        MPI processes to be used
 6:     Distribute cliques to MPI processes
 7:     for each non-zero clique of the graph do
 8:         Run Expectation Maximization and Maximum a Posteriori
            optimizations on assigned MPI processes
 9:     end for
10:     Gather parameter estimation information for subgraphs
11:     Update parameters
12: end for
13: Generate resulting output image
```

MPI - Parallel Markov Random Fields (MPI-PMRF)

Algorithm 1 Distributed memory version of the Markov Random Field algorithm using graph partitioning and parameter estimation (MPI-PMRF). Line 8 is run in parallel to distribute the largest amount of work to increase performance.

Input: Original image, oversegmentation, number of classes
Output: Segmented image and estimated parameters

```
 1: K ← number of classes
 2: Initialize parameters and initial labels randomly
 3: Create graph from oversegmentation
 4: for each Expectation Maximization iteration do
 5:     Divide graph into subgraphs (cliques) based on number of
        MPI processes to be used
 6:     Distribute cliques to MPI processes
 7:     for each non-zero clique of the graph do
 8:         Run Expectation Maximization and Maximum a Posteriori
            optimizations on assigned MPI processes
 9:     end for
10:     Gather parameter estimation information for subgraphs
11:     Update parameters
12: end for
13: Generate resulting output image
```

Experiments Overview

- Types of Experiments
  - Verification testing
  - Performance testing
- Scaling characteristics
  - 2 datasets
  - Large-scale platform

(a
)

(b
)

(a) Result from threaded version; (b) Result from MPI-PMRF

Results are identical

Computation Verification

- 98.99% accurate compared to ground truth
- Identical results with threaded version
- Berkeley Dataset Benchmark
  - Higher accuracy than other image segmentations

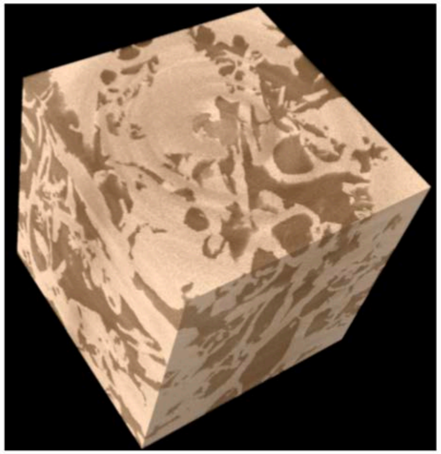
(a) (b)

Synthetic dataset → (a) original image; (b) MPI-PMRF result

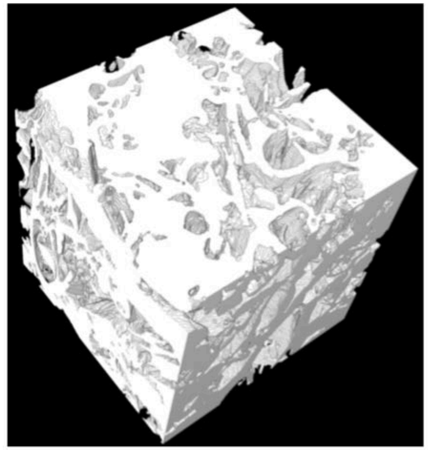
(a) (b)

Experimental dataset → (a) original image; (b) MPI-PMRF result

Results



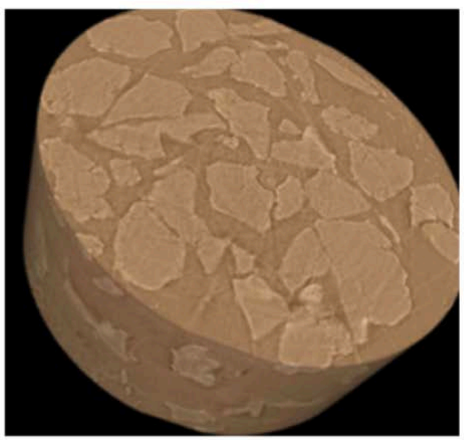
(a)



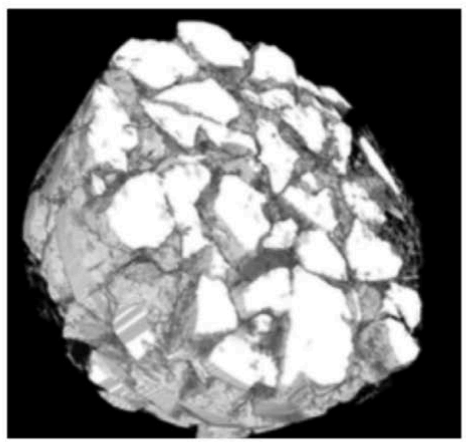
(b)

(a) 3D rendering of original synthetic data

(b) 3D rendering of MPI-PMRF result



(a)



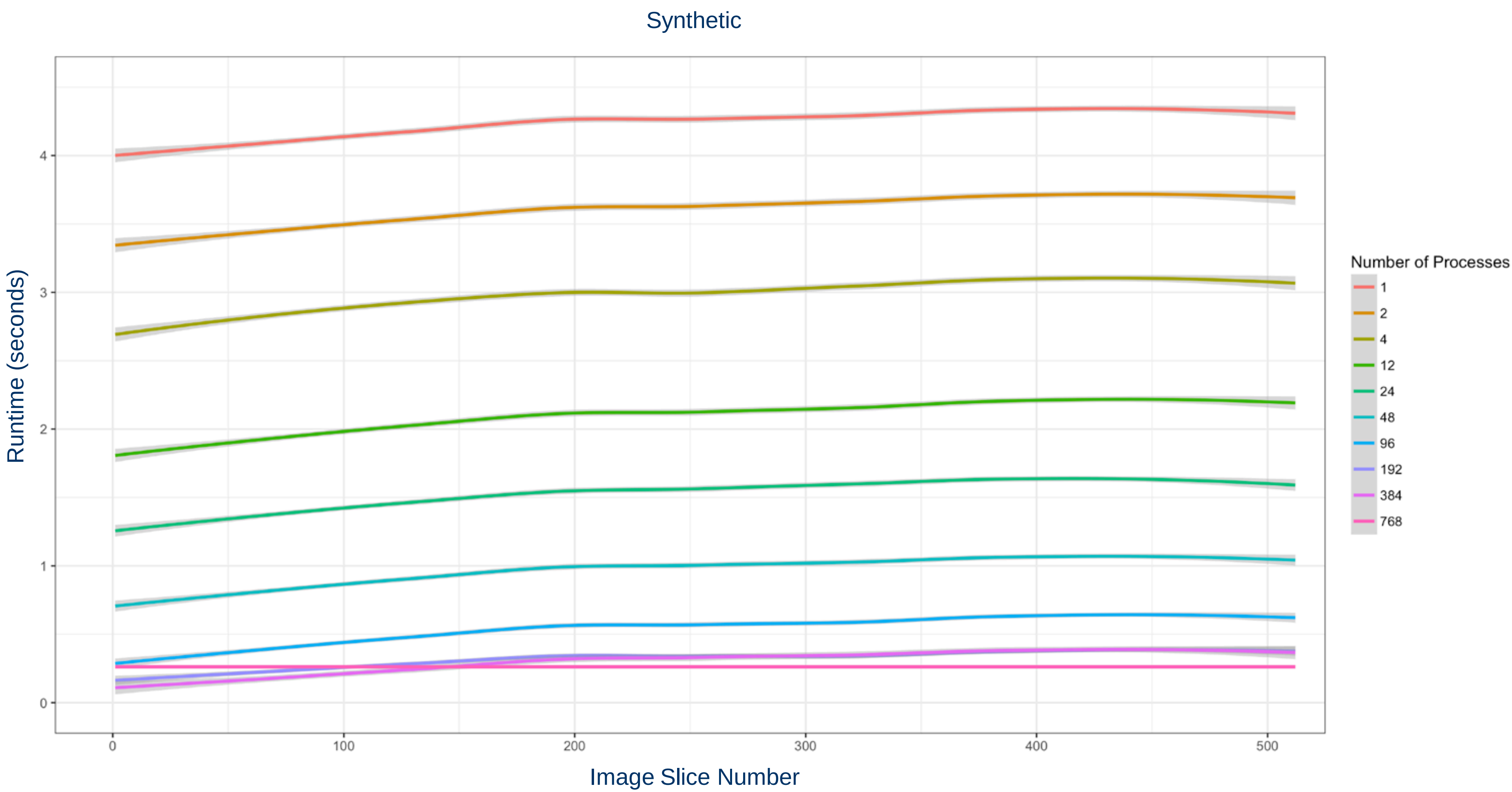
(b)

(a) 3D rendering of original experimental data

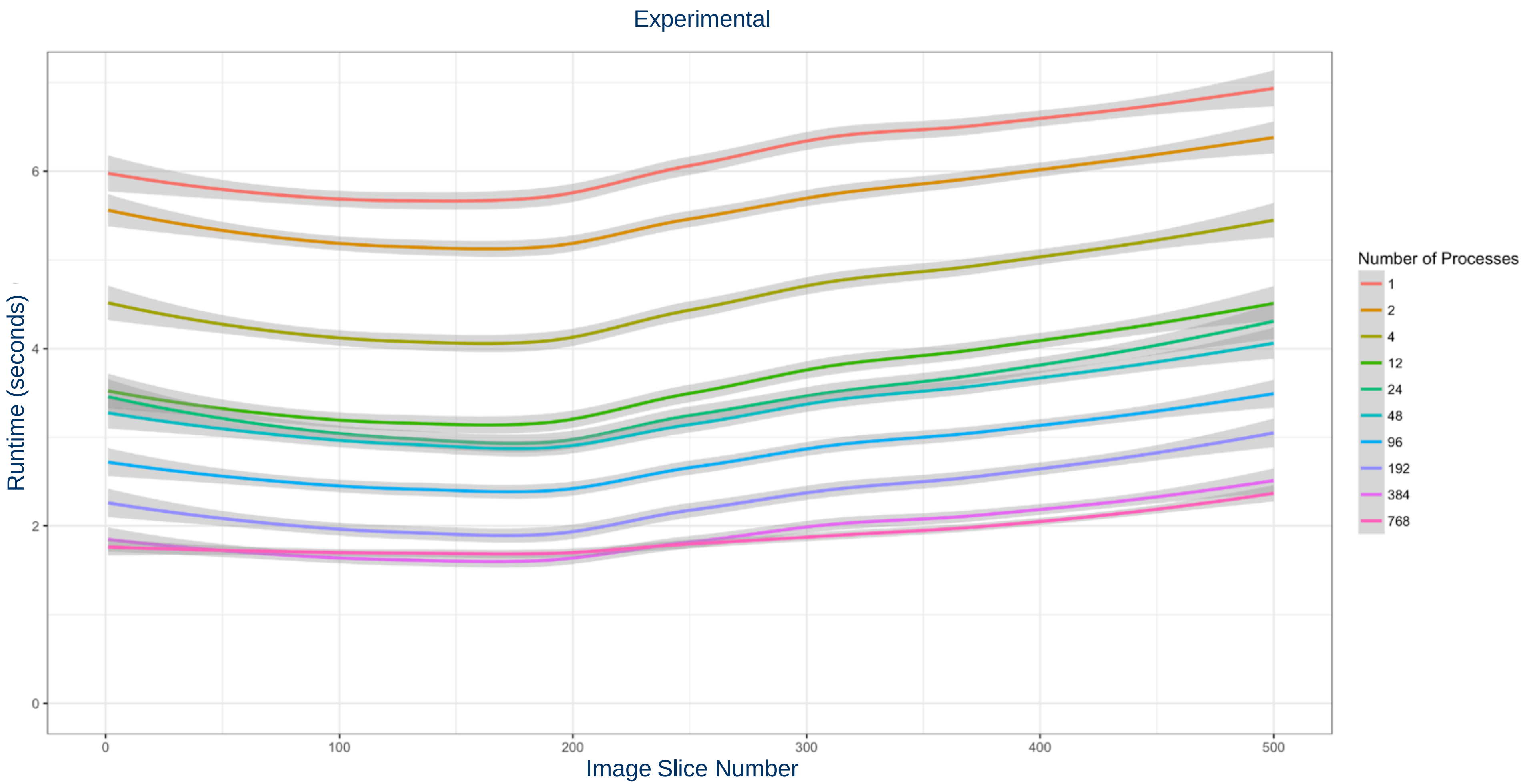
(b) 3D rendering of MPI-PMRF result

Performance Analysis

- Platform
  - Edison supercomputer at NERSC
  - Cray XC30 system
    - 24 cores per node
- Methodology
  - Run 2 datasets
  - Varying levels of concurrency
  - Scalability study
  - Additional performance metrics



When measuring runtime (in seconds) of the synthetic dataset, the results show the overall decrease in runtime as concurrency increases



When measuring runtime (in seconds) of the experimental dataset, the results show the overall decrease in runtime as concurrency increases

Results – Efficiency and Rate

- Use “efficiency” and “rate” metrics to yield insight into scaling performance
- **Efficiency**: measure degree to which code scales compared to serialized version
- **Rate:** measure degree to which performance time increases as function of workload size and concurrency
- Measuring workload imbalance?

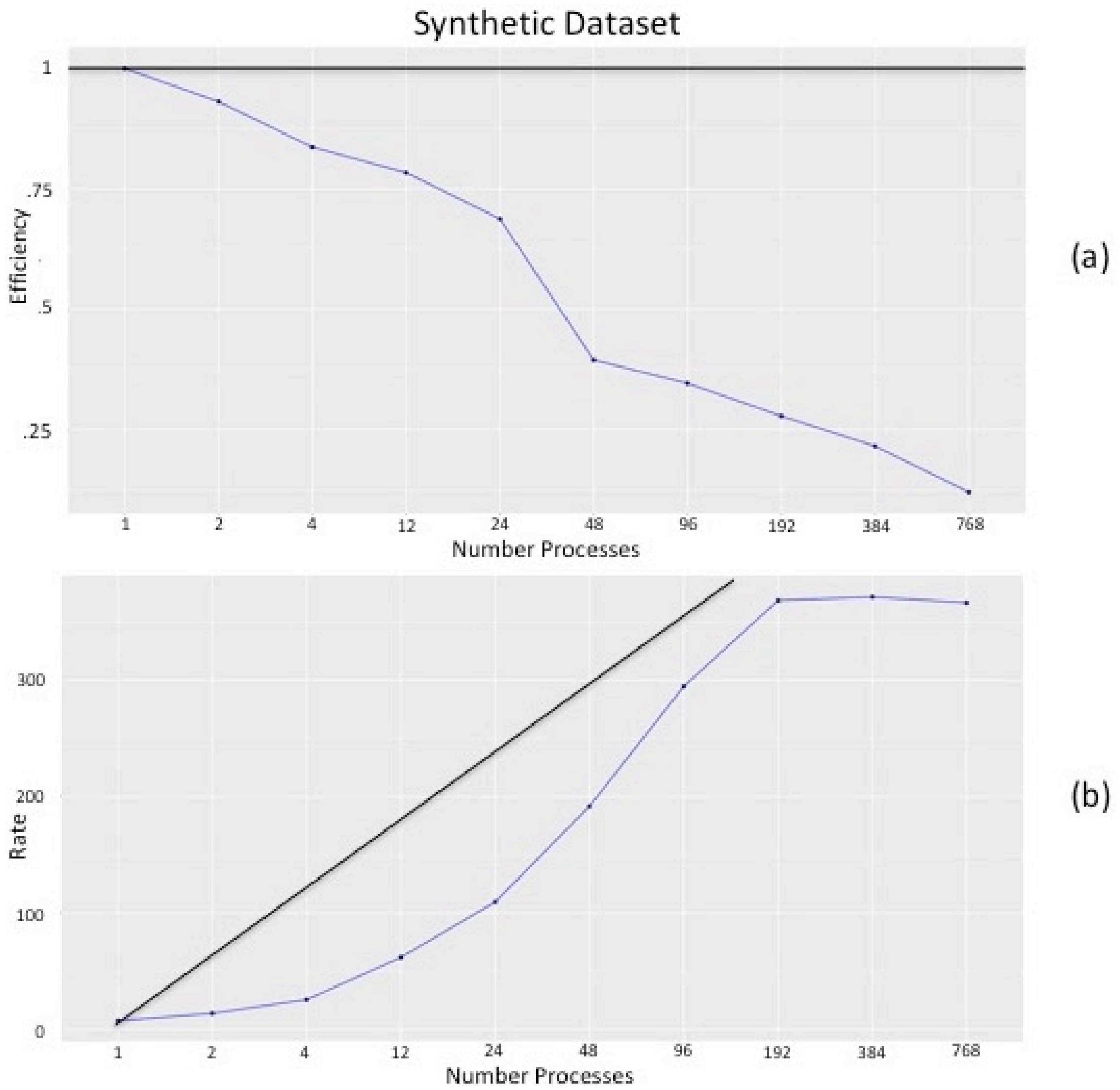
Efficiency

$$E(n,p) = \frac{C^*(n)}{C(n,p)}$$

Rate

$$R(n,p) = \frac{n}{T(n,p)}$$

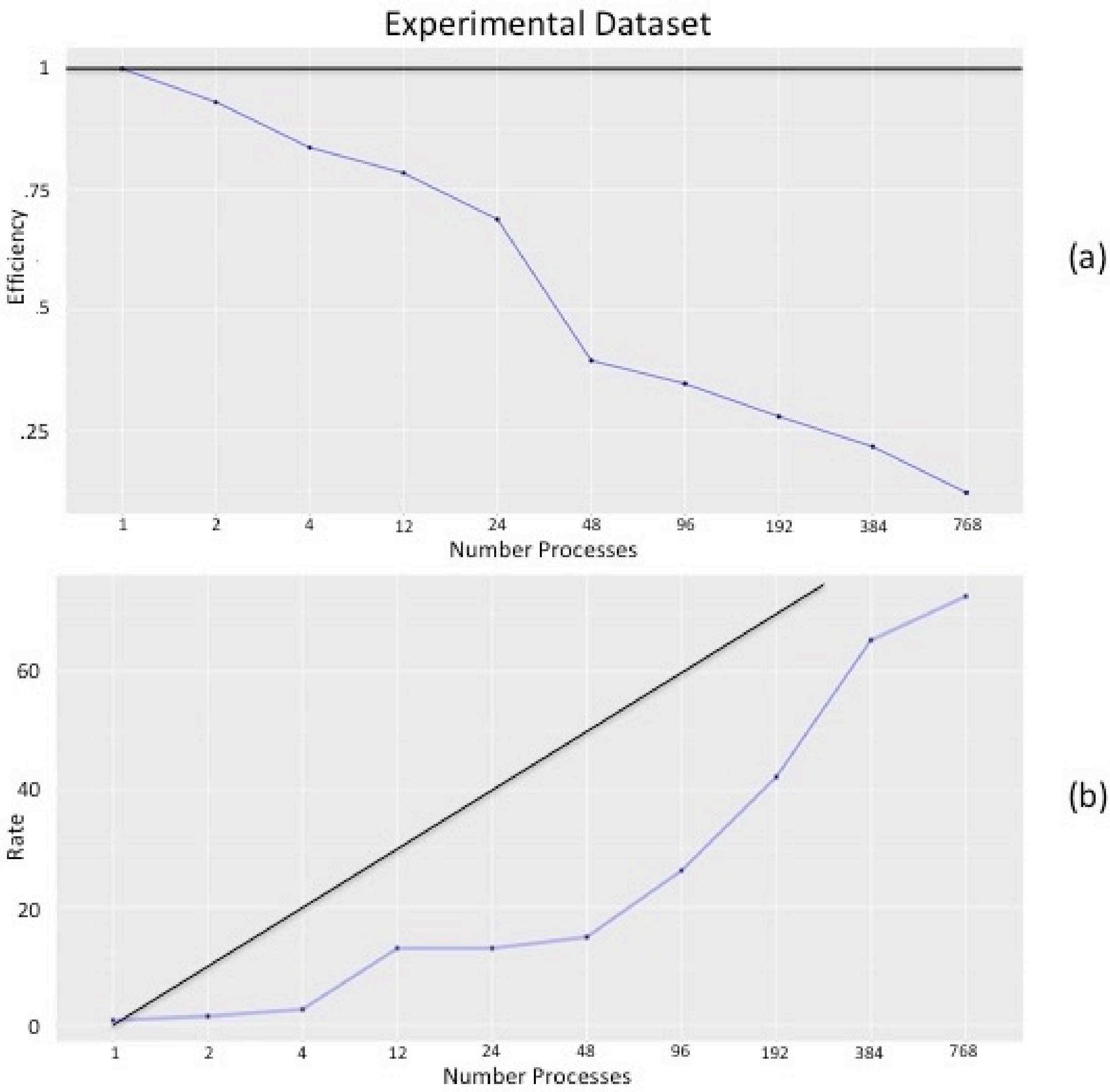
K. Moreland and R. Oldfield, “Formal metrics for large- scale parallel performance,” in *Proceedings of International Supercomputing Conference*, 2015.



(a) Efficiency of the synthetic dataset; does not follow ideal efficiency of 1, but still provides an increase in performance

(b) Rate of the synthetic dataset does not follow ideal rate with a slope equal to 1, but provides a performance increase at different concurrencies and executes faster than when running in serial

Insufficient workload
Workload imbalance
Communication



(a) Efficiency of the experimental dataset; does not follow ideal efficiency of 1, but still provides an increase in performance

(b) Rate of the experimental dataset does not follow ideal rate with a slope equal to 1, but provides a performance increase at different concurrencies and executes faster than when running in serial

Insufficient workload
Workload imbalance
Communication

Summary of Results

- MPI-PMRF shows decrease in runtime as concurrency increases
- Limits to efficiently scaling:
  - Workload imbalance
  - Serialization due to inter-processor communication
  - Insufficient workload
- Future work
  - Increase problem size and complexity
  - Workload balance
  - Extend algorithm to work with 3D image volumes

Conclusion

- Promising new approach for scalable image segmentation to help meet scientific needs
- Take advantage of large computational resources and process large data
- Additional work needed to improve performance

BERKELEY LAB

Lawrence Berkeley National Laboratory

U.S. DEPARTMENT OF ENERGY

UNIVERSITY OF CALIFORNIA