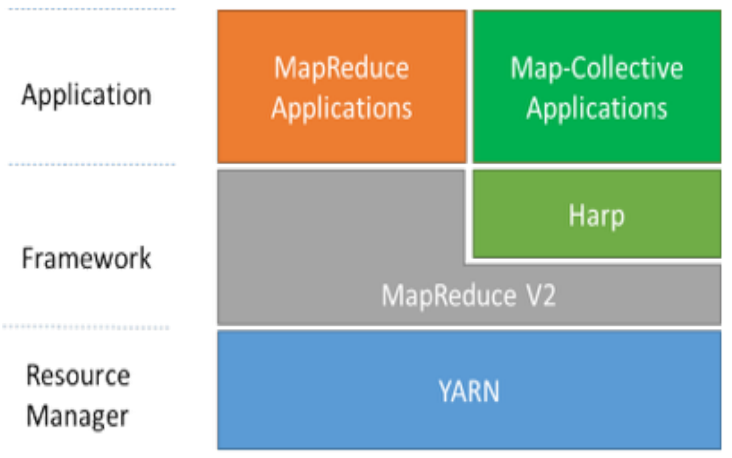
Multinomial Logistic Regression using Harp

Chao-Hong Chen and Qiuwei Shou, CSCI-B534 Distributed System, Indiana University

Introduction

- Large scale data analysis challenges the performance of machine learning algorithm
- Exploring parallelization of Multinomial Logistic Regression[2,4] in large scale dataset by using Harp[3]
- Harp is a communication collective library working as a plugin in Hadoop
- Project is inspired by [Genkin 2007]paper[7]

1



3

Dataset

RCV1v2[6]

- 800,000 manually categorized newswire stories made by Reuters, Ltd. for search purposes.
- 23,149 training documents and 781,265 test documents
- 47,236 terms in each document
- 103 topics over 4 hierarchical groups
- We use the data in the vector format. Each vector in a file represented by the form <did> [<tid>:<weight>].
  - <did>: An unique document id.
  - <tid>: A positive term id which is between 1 and 47,236
  - <weight>: The number feature value within document weight.

<u>Example of the vector file format:</u>

9995 1:0.03 3:0.047 8:0.38749738478937479 14:0.1 2748:0.03
999996 7:0.13 19:0.138 255:0.58588 314:0.28101 18800:0.005
999998 2:0.00001 3:0.108 184:0.228 488:0.0821 40917:0.111

Methodology

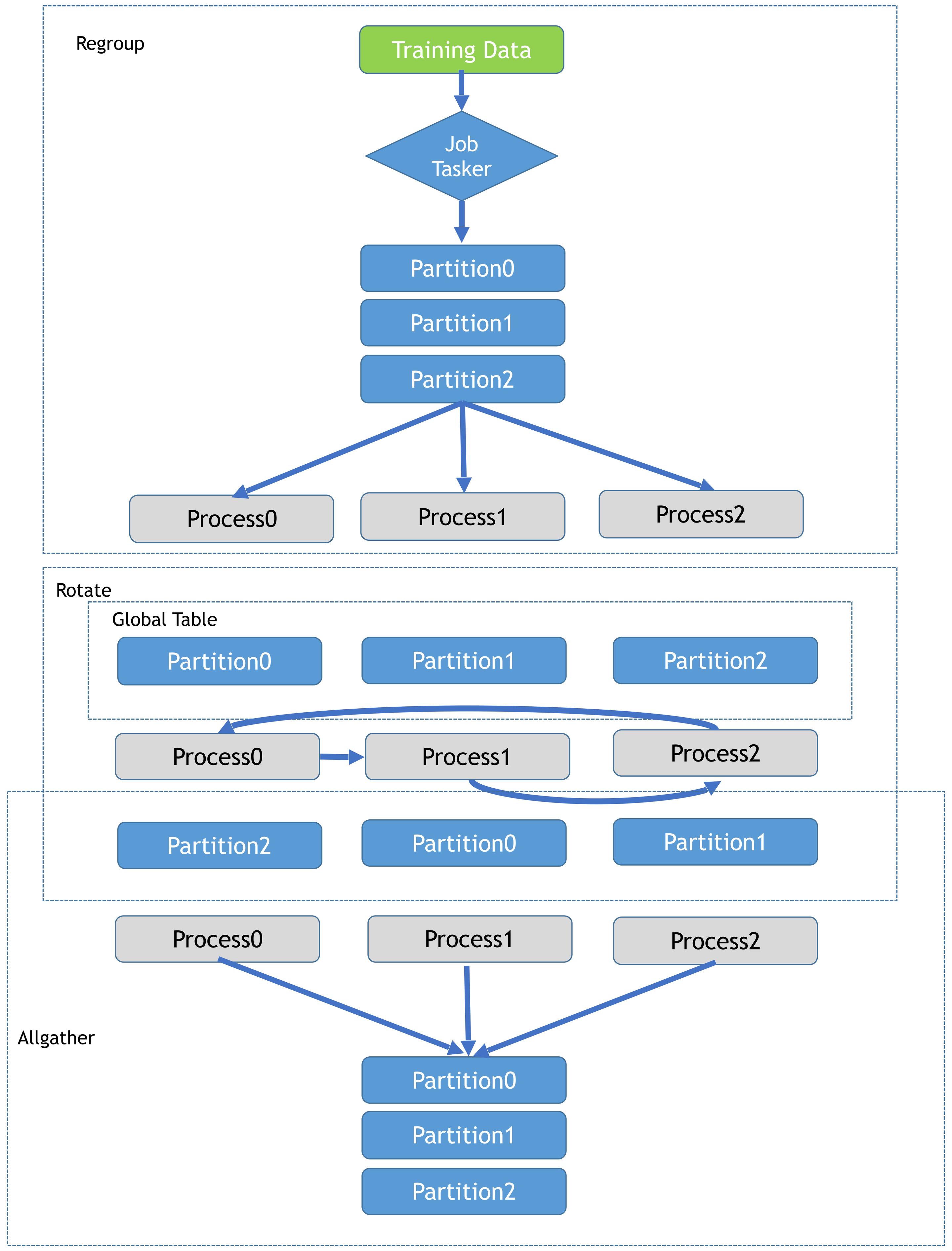
- We use SGD to train the MLR model
  - Define a lost function J(θ)

$$J(\overline{\theta}) = -\frac{1}{m}\sum_{i=1}^{m} y_i \log h_{\overline{\theta}}(\overline{x_i}) + (1-y_i)\log(1-h_{\overline{\theta}}(\overline{x_i}))$$

  - Minimize the lost function and update θ with given learning rate by calculating the partial derivative

initialize $\overline{\theta}$
for $j = 1$ to $ITER$ **do**
  for $i = 1$ to m **do**
    $\overline{\theta} := \overline{\theta} - \alpha \frac{1}{m} \sum_{i=1}^{m} (h_{\overline{\theta}}(\overline{x_i}) - y_i) \cdot \overline{x_i}$
  end for
end for

- SGD is parallelized by using regroup/allgather model in Harp.
  - Regroup distributes data and task to all mappers
  - Rotate passes the updated computation results from one mapper to the others
  - Allgather collects the output from each mapper



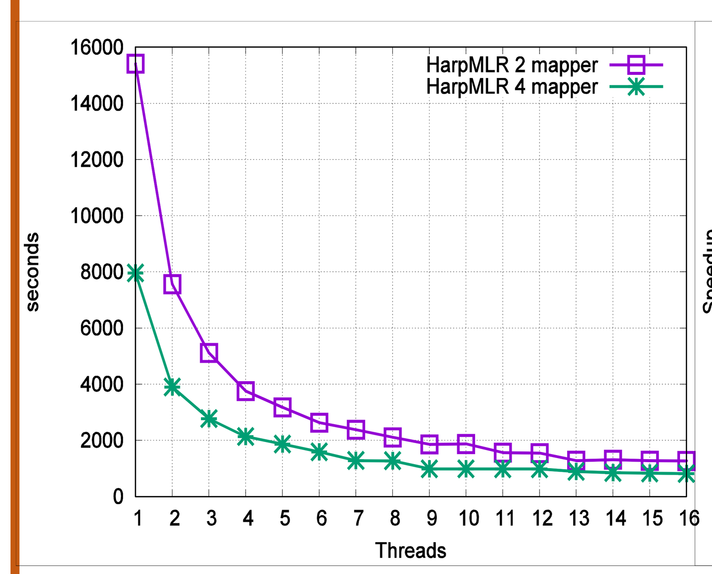
Evaluation

- Perform text categorization using the trained MLR model
  - We use two machines each has Intel(R) Xeon(R) CPU E52670 v3 with 128GB ram
  - Analyze the effectiveness of the algorithm
  - Compare the runtime with increasing number of mappers and threads

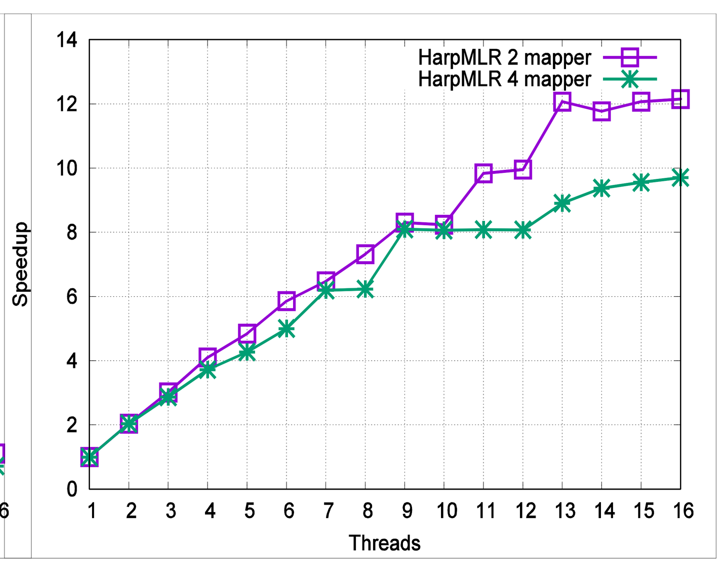
Results

- Increasing local thread number gives almost linear speedup.
- Increasing local thread number is slightly faster than increasing number of mappers
- We also use the training set in [6] to calculate the effectiveness of the output results, the macroaveraged F1:0 (defined in [5]) is 0:62.

Harp SGD



Runtime of Harp MLR with 2 mappers



Conclusion

- Propose a parallel version of SGD to solve MLR using Harp
- Evaluate algorithm in RCV1v2
- Achieve expected speedup from increase number of mappers and increase number of local threads.

Reference

[1] https://iu.instructure.com/files/65089925/download?download_frd=1
[2] https://github.com/tpeng/logistic-regression.
[3] Harp. http://salsaproj.indiana.edu/harp/index.html.
[4] scikit-learn. http://scikit-learn.org/.
[5] D. D. Lewis. Evaluating and optimizing autonomous text classifciation systems. In Proceedings of the 18th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval, SIGIR '95, pages 246-254, New York, NY, USA, 1995. ACM.
[6] D. D. Lewis, Y. Yang, T. G. Rose, and F. Li. Rcv1: Anew benchmark collection for text categorization research. J. Mach. Learn. Res., 5:361{397, Dec. 2004.
[7] Genkin, A., Lewis, D.D., Madigan, D., 2007. Large-scale Bayesian logistic regression for text categorization. Technometrics 49, 291–304.