MAP-REDUCE IMPLEMENTATIONS: SURVEY AND PERFORMANCE COMPARISON

Zeba Khanam and Shafali Agarwal

Department of Computer Application, JSSATE, Noida

ABSTRACT

Map Reduce has gained remarkable significance as a prominent parallel data processing tool in the research community, academia and industry with the spurt in volume of data that is to be analyzed. Map Reduce is used in different applications such as data mining, data analytics where massive data analysis is required, but still it is constantly being explored on different parameters such as performance and efficiency. This survey intends to explore large scale data processing using MapReduce and its various implementations to facilitate the database, researchers and other communities in developing the technical understanding of the MapReduce framework. In this survey, different MapReduce implementations are explored and their inherent features are compared on different parameters. It also addresses the open issues and challenges raised on fully functional DBMS/Data Warehouse on MapReduce. The comparison of various Map Reduce implementations is done with the most popular implementation Hadoop and other similar implementations using other platforms.

KEYWORDS

MapReduce, Parallel Data Processing tools, MapReduceFrameworks, Hadoop, DBMS/DataWarehouse.

1. INTRODUCTION

Map reduce is prominently popular in companies, that have huge data sets stored on large number of nodes with an increasing demand to store, retrieve and analyze the rapidly growing data. Different web services generate petabytes of data such as different log files, web search engines using crawlers generate huge archives of web content and large sets of redundant data is also generated from a variety of web services. Earlier the solution was to engage the parallel database systems to deal with such massive amounts of data. But the drawback is that usually such database systems run on expensive high-end servers. When this massive data to be stored and processed increases to the extent of requiring clusters of thousands of nodes, parallel database solutions become extremely expensive [6].

In order to deal with such problems and to process and analyze such large and parallelizable datasets in a cost effective manners different companies have developed distributed data storage and processing systems on large clusters or grids of low-cost commodity machines. Google's Map Reduce [11] is among the first ones to have taken the initiative, gradually other Map reduce frameworks have also evolved such as Hadoop [3] from the open-source community, and Cosmos [24] and Dryad [25] at Microsoft. The framework depicts a programming model that consists of hundred thousands of clusters of commodity machines connected via a high-bandwidth network for running these systems.

DOI:10.5121/ijcsit.2015.7410

2. COMPARATIVE ANALYSIS OF DIFFERENT MAP REDUCE IMPLEMENTATIONS

Map reduce was initially developed by Google as Google Map Reduce and Google File System (GFS).The next invention was a more recently popularized open source technology Apache Hadoop's MapReduce and HDFS components [30]. Hadoop is currently the most popular open source MapReduce implementation written in Java and has been tested in Yahoo's cluster. The term "Hadoop" does not only represent the base modules described above but there are number of add ons incorporated in the base module. The Hadoop "ecosystem" represents all the software packages and the add ons such as Apache Hive, Apache Pig, Apache HBase, Apache Spark, JAQL, Cascading and others [39][40] that serve different purposes and can be installed along with Hadoop. Sailfish [31] is a Map-Reduce framework for large scale data processing. The Sailfish design facilitates batch transmission from mappers to reducers to improve performance. An abstraction called I-files is used for adapting the Map Reduce layer for transporting data from map tasks to reduce tasks supporting network-wide data aggregation. It is implemented as an extension of the distributed filesystem, and works by dividing the data written by multiple writers and read by multiple readers into batches. Skynet is an open-source Ruby implementation of MapReduce [32]. Besides the basic features of MapReduce systems, Skynet can be configured as a fully distributed system with no permanent master node. This facilitates in providing a peer recovery mechanism to reduce the overhead of the master node. All workers are designed and facilitated to perform the task of a master at any time. The worker nodes monitor the status of each other; if one worker is down, another worker will detect it and take over the task. This improves fault-tolerance since the single point of failure is avoided.

Recently many other map/reduce implementations, have been introduced. In this section different implementation environment for MapReduce is highlighted and their performances are compared [1].

Table1: Map Reduce Implementations for different environments

Implementation	**Organization**	**Technology**	**Functionality**
Hadoop	Developed by Apache	Java or arbitrary language (user), Java (service)	Distributed File System, , Scheduling software, Data replication, Fault tolerance,
Spark	Developed by AMPLab at UC Berkeley	APIs for Scala, Java, and Python	Supports faster applications using resilient distributed datasets (RDD). Query large
Phoenix++	Developed at Stanford University	APIs only for C/C++	a MapReduce framework for shared-memory (chip multiprocessor) CMPs and SMP(symmetric
MARISSA (MApReduce Implementation for Streaming Science Applications) (2012)	Developed at SUNY Binghamton	Supports any Executable Binary	Supports variety of POSIX compliant file systems. Iterative application support, Fault-tolerance, •
MARIANE (MApReduce Implementation Adapted for HPC Environments) (2011)	Developed at SUNY Binghamton	Java	Suitable for HPC environments, making use of various cluster, shared-disk, POSIX, and parallel file systems.

MapReduce-MPI	Developed by Steve Plimpton (Sandia)	Bindings for C++ (principal), C, and Python	Uses DDFS
Disco	Developed by Nokia, now open source	Python (user), Erlang (service)	Uses Distributed Index
SASReduce	SAS Technologies	Python	Parameter based Batch Processing, File splitter, Queue based scheduler, Parallel
BitDew	BitDew	Java	Data distribution, replication, Fault tolerance in highly dynamic aggressive task backup and volatile environment,
MARLA(MApReduce with adaptive Load balancing) (2012)	Developed at SUNY Binghamton	Java	Performs better in heterogeneous cluster, load imbalanced environment and cloud environments
DRYAD and DRYADLINQ	Developed by Microsoft	Programmable via C#	Uses Shared directories/ Local disks, Network topology based run time graph optimizations. Monitoring support for execution graphs
Themis	Developed by Rasmussen et al	Java	Meets two I/O property(for jobs that consumes large memory, the I/O operations are kept
MR4C	Developed by Skybox Imaging	Handles libraries in C C++	Algorithms are stored in native shared objects that access data from the local filesystem or any uniform resource identifier (URI), while input/output datasets, runtime parameters,

The table above highlights different MapReduce implementations.

Dryad developed by Microsoft is a general purpose engine for executing parallel and distributed applications [25]. A Dryad application runs in form of data flow graphs by executing the vertices of this graph on a set of available computers. Dryad has been used for simple map reduce style data mining operations and shares a lot of similarity with Google MapReduce [1]. In contrast to using a sequence of map/distribute/sort/reduce operations Dryad application may specify an arbitrary communication Dryad Graph. Therefore the application developer is required to first construct the static job graph and pass it to the runtime to get executed. It promotes easy creation of large-scale distributed applications without requiring them to master any concurrency techniques. Major prerequisite is to construct a graph of the data dependencies of their algorithms.

The drawback being architectural complexity compared with the MapReduce system design. DryadLINQ is another programming model that is a product of Microsoft. The benefit of DryadLINQ generalizes previous execution environments such as Dryad, MapReduce, SQL by adopting an expressive data model of .NET objects; and by supporting general-purpose traditional high-level programming language and hybrid of declarative and imperative programming by exploiting LINQ (Language Integrated Query, a set of .NET constructs for programming with

datasets). The system provides support for flexible and efficient distributed computation in any LINQ-enabled programming language including C#, VB, and F#. The Dryad and DryadLINQ are available [27]. All the DryadLINQ programs are passed to the Dryad execution platform and run on Dryad cluster computing infrastructure. The performance of Dryad and DryadLINQ is compared in [26] by implementing it to the most time consuming query (Q18) from the Sloan Digital Sky Survey database [28]. The query had identified a "gravitational lens" effect and the performance of the two-pass variant of the Dryad program with that of DryadLINQ was done. The results indicated that the DryadLINQ consumed only 100 LOCs of C# and the Dryad program is around 1000 lines of C++ code.

Spark is a new cluster computing framework and proides support for executing the applications much faster than Hadoop. This is achieved by keeping data in memory, and using it interactively to query large datasets with sub-second latency. Spark overcomes the problem of sharing data across multiple MapReduce steps that is required in multipass and interactive applications. For solving this problem Spark provides a new storage primitive called resilient distributed datasets (RDD). RDDs are read and written up to 40 times faster than the distributed filesystem [4]. MARIANE-Hadoop [2] is another implementation of MapReduce based on HDFS. However Hadoop File System is not POSIX compliant [10]. Majority of applications running on the existing HPC environments such as Teragrid and NERSC cannot utilize it. Map Reduce is suitable for HPC environments but results reveal that there could be other design options too that could provide more efficient and better performance in those settings [1]. Another map reduce framework introduced by Fadika etal is MARISSA whose architecture is based on their previous work MARIANE [2]. MARRISA [12] is an implementation for streaming science applications. Hadoop native supports applications written only in Java while Hadoop streaming extends support to non-Java applications and enables scripts and executable binaries to run on its framework. As described [2], Hadoop streaming does not have the features to support scientific applications. MARISSA offers better performance as compared to Hadoop in terms of performance and turnaround time of the applications. The Hadoop streaming model imposes a performance penalty also that was depicted in [12] by using the Hadoop streaming for the execution of C Program. MARISSA also addresses the problem of supporting multiple executables that is not dealt in most of the MapReduce implementations. It is also capable of supporting POSIX compliant file system. MARLA, another MapReduce framework specifically outperforms most of the popular implementations such as Hadoop [6], Twister [7] (does not cater to load balancing) and LEMO-MR [8]. LEMO-MR and Twister both support only Java applications. The results in [9] depict that in handling fault tolerance MARLA and MARIANE perform closely and depicts a better performance over Hadoop. In case of stressed clusters MARLA's overall performance is better whereas MARIANE is slower than Hadoop [9].

MARLA proves to be much faster over other MapReduce implementations in both stress-free clusters and stressed clusters. Phoenix++ is an enhancement over Phoenix that is a MapReduce framework for shared memory CMPs and SMPs. Unlike other shared memory MapReduce frameworks Phoenix++ does not modify the MapReduce interface [11] but uses modularity and function in lining to achieve high locality and low resource usage. Phoenix allows user to write simple, high performance MapReduce code with an increased scalability.

SASReduce Framework is also an implementation of MapReduce in BASE/SAS®. The Map Reduce technology Hadoop, is implemented by SAS technologies such as BASE/SAS®, SAS/MACRO® and

SAS/CONNECT®. The SAS DATASTEP® works as the 'Map' function, and SAS procedures are used to implement the "Reducer". Though it replicates the major functionalities of MapReduce but it does not support Fault tolerance, Data replication and Shared-nothing

architecture [5]. Also the map task is performed on single SMP machine rather than individual machines. Bitdew-MR as stated by Lu Lu is another implementation of MapReduce on Desktop Grid [3]. The results from [3] depict that BitDew-MapReduce performs better when run on Internet Desktop Grid and outperforms Hadoop on several aspects such as fairness, scalability, resilience to node failures and network disconnection. The other extensions to the Map Reduce model is MRPGA (An Extension of MapReduce for Parallelizing Genetic Algorithms)

As MPI's are associated with some drawbacks of being unable to handle heterogeneity, failures etc and that is why it not suitable for cloud applications, another author [2] presented his work of executing PGAs on a MapReduce model. The parallel design pattern support provided by the model eases the task of application development in distributed environments. As this model is not suitable to express PGA's directly, they presented an extension to MapReduce model through an additional reduce phase for global selection, called once at the end of each iteration of the GA loop. To manage faults during execution, they made the master to replicate the optimum individuals selected by MRPGA for each round of evolvement in their architecture.

Themis is another MapReduce implementation designed to have the 2-IO property and focuses on minimizing the I/O operations [33]. Themis performs a wide variety of MapReduce jobs and outperforms Hadoop by a factor of 16 on a variety of common jobs. Most of the map reduce jobs are performed such as click log analysis, DNA read sequence alignment, and PageRank (that is performed with nearly the speed of TritonSort's) [33][34] but it is usually used for small clusters where the there are few chances of node failure. Themis achieves this high performance by reading and writing each record to disk exactly twice. Themis and TritonSort that may also be termed as its predecessor hold four large-scale sorting records as of 2012. Themis is comparable to Triton Sort in terms of the speed that is equivalent to the sequential speed of the disks for I/O-bound jobs, which is approximately the same rate as Triton Sort's record-setting performance. Themis has flexible memory subsystem and provides support to handle large amounts of data skew while ensuring efficient operation record-setting sort performance. Themis accommodates the flexibility of the MapReduce programming model while simultaneously delivering high efficiency. This is achieved by considering fundamentally different points in the design space than existing MapReduce implementations. The comparison of Hadoop version 1.0.3 and Themis done on the Sort-500G and CloudBurst applications shows that Themis has outperformed Hadoop by a factor of 3-16 [33]. Even after the best optimization, Hadoop encountered difficulties in running many large parallel transfers without having the nodes blacklisted for running out of memory. Therefore the performance of a Map Reduce job can be significantly improved, but the ideal candidates for Themis are the clusters that can process petabytes of jobs yet are small enough to experience a lower failure rate than warehouse clusters.

A very recent addition to the Map Reduce Community is the release of MR4C Map Reduce for C that is developed at SkyBox Imaging as an open source framework and allows the execution of native code in Hadoop . It is basically meant to facilitate large scale satellite image processing and geospatial data science, and at the same time also provides the efficiency of natively developed algorithms with the flexibility and scalability present in Hadoop [41].

Other systems have also been developed mainly by the industry and renowned organization to support Big Data analysis, such as Facebook [13], Microsoft's SCOPE [16][24], Yahoo's PNUTS [15], Twitter's Storm [17], LinkedIn's Kafka [18] and Walmart Labs' Muppet [19]. Scope is designed in a way to provide the advantages from both traditional parallel databases and MapReduce execution engines to allow easy programmability and deliver massive scalability and high performance through advanced optimization. The system supports SQL-like declarative scripting language with no explicit parallelism, compliant to efficient parallel execution on large clusters. Microsoft SCOPE, Apache Pig [22, 23] and Apache Hive[20, 21] all aim at supporting

declarative query languages for the MapReduce framework. There has been a spurt in High-level declarative languages, such as Pig, Hive, and Jaql to allow developers to program at a higher level of abstraction. Other runtime platforms, including Nephele/PACTs [4] and Hyracks [6], have been developed to improve the MapReduce execution model.

3. RELATED WORK

Feng etal [29] attempts to present the current research on enhancing MapReduce to better address modern data intensive applications without losing its fundamental advantages and also reviews a number of map reduce implementations. They also discuss ongoing work to provide enhancements to the MapReduce framework to efficiently deal with a richer set of workloads such as streaming data, iterative computations. Other researchers have also explored Map Reduce implementations. Map Reduce implementation in cloud environment is also much researched area [35]. Liu describes Cloud MapReduce for Amazon. This programming model is supported by the Amazon cloud OS. The scalability offered by the cloud OS is an added advantage as compared to the server OS as it is much faster and scalable. The work in [36] compare two implementations of cloud MapReduce in cloud computing environments, AzureMapReduce [38] and Amazon Elastic MapReduce [37].

4. CONCLUSION

The analysis and comparison of these various MapReduce implementations specifically in reference to the Hadoop that has some standard features such as support for various types of input/output format support, compression techniques, a customized scheduler etc, it is observed that there are salient differences between them, for example task scheduling, data management approaches, the language support and the language used to implement the system, the file system employed, the indexing strategies, the design for multiple jobs and the master slave node designing. The discussion in the paper highlights different MapReduce implementations. Some of them are either built on top of Hadoop, or implemented using different scripting languages such as Apache Pig, Apache Hive, Apache HBase, Apache Spark, and others. Spark for example claims to run applications much faster than Hadoop by using a new storage primitive called resilient distributed datasets (RDD) that overcomes the problem of sharing data across multiple MapReduce steps. Similarly Dryad that is a Microsoft product which promotes easy creation of large-scale distributed applications without requiring them to master any concurrency techniques. It runs application in form of data flow graphs and so the user does not need to perform sequence of map/distribute/sort/reduce operations. The popularity of Map Reduce implementations and specifically Hadoop for the analysis of massive data sets , the open-source communities and researchers have developed a number of higher-level languages and programming libraries for Hadoop. Among those approaches are Hive , Pig , JAQL , and Cascading. Data processing tasks written in any of these languages are compiled into one or multiple MapReduce jobs which are executed on Hadoop. All of them aim to ease the development of data parallel program, however, all approaches are applicable on different use-cases and have considerably varying feature sets. For example the Hive system focuses on data warehousing, JAQL is a query language for the JSON data model and Pig's data and processing model is rather designed to fit in somewhere between the declarative style of SQL, and the low-level, procedural style of MapReduce . Then there are certain databases also that offer built in MapReduce support such as MongoDB,Aster etc. Table 1 highlights different MapReduce implementations a nd their comparisons on the basis of language used, file system and task scheduling. The newer systems that are introduced after Hadoop tend to either ease up the task or make it faster than Hadoop but still Hadoop with its richer APIs, features and administrative tools still rules as the most popular and most adopted Map Reduce framework.

REFERENCES

[1] J.Dean and S. Ghemawat, "MapReduce: Simplifed Data Processing on Large Clusters," Operating Systems Design and Implementation, 2004, 137-149.

[2] Z.Fadika, E. Dede, M. Govindaraju, and L. Ramakrishnan. Mariane, " Mapreduce implementation adapted for HPC Environments", IEEE/ACM International Workshop on Grid computing. 0:82–89, 2011.

[3] Lu Lu, Hai Jin, Xuanhua Shi & G. Fedak, "Assessing MapReduce for Internet Computing: A Comparison of Hadoop and BitDew-MapReduce", in GRID '12 Proceedings of the 2012 ACM/IEEE 13th International Conference on Grid Computing, Pages 76-84
IEEE Computer Society Washington, DC, USA ©2012.

[4] M.Zaharia, M.Chowdhury, T.Das, A.Dave. etal, "Fast and Interactive Analytics over Hadoop Data with Spark" USENIX, Aug 2012.

[5] D.Moors, "SASReduce - An implementation of MapReduce in BASE/SAS", Paper 1507-2014 Whitehound Limited, UK, 2014.

[6] Apache Hadoop. [Online]. Available: http://hadoop.apache.org

[7] J.Ekanayake, H. Li, B. Zhang, T. Gunarathne, S.-H. Bae, J. Qiu, and G. Fox, "Twister: a runtime for iterative mapreduce," in HPDC, 2010, pp. 810–818.

[8] Z.Fadika and M.Govindaraju, "Lemo-mr:Low overhead and elastic mapreduce implementation optimized for memory and cpu-intensive applications," IEEE International Conference on Cloud Computing Technology and Science", vol.0,pp. 1-8,2010.

[9] Z.Fadika, E. Dede., Hartog,J., M. Govindaraju. , "MARLA:MapReduce for Heterogenous Clusters", 12th IEEE/ACM International Symposium on Cluster, Cloud and Grid Computing (CCGrid), 2012 .

[10] K.Shvachko, H. Kuang, S. Radia, and R. Chansler. The hadoop distributed file system. In Mass Storage Systems and Technologies (MSST), 2010 IEEE 26th Symposium on, pages 1 –10, May 2010.

[11] J.Talbot, R. M. Yoo, and C. Kozyrakis. Phoenix++: modular mapreduce for shared-memory systems. In Proceedings of the second international workshop on MapReduce and its applications, MapReduce '11, pages 9–16, New York, NY, USA, 2011. ACM.

[12] E.Dede, Z. Fadika, J. Hartog, M. Govindaraju, L. Ramakrishnan, D. Gunter, and R. Canon, "Marissa: Mapreduce implementation for streaming science applications," in E-Science (e-Science), 2012 IEEE 8th International Conference on, Oct 2012, pp. 1–8.

[13] J.Chao, V.Christian and B.Rajkumar, "MRPGA: An Extension of MapReduce for Parallelizing Genetic Algorithms", in Proceedings of the 4th IEEE International Conference on e-Science 2008.

[14] Nokia Research Center. Disco: Massive data- minimal code. http://discoproject.org, 2010.

[15] B.F. Cooper, R. Ramakrishnan, U. Srivastava, A. Silberstein, P. Bohannon, H.-A. Jacobsen, N. Puz, D. Weaver, and R. Yerneni, " PNUTS: Yahoo!'s hosted data serving platform", Proceedings of the VLDB Endowment (PVLDB), 1(2):1277{1288, 2008.

[16] J.Zhou, N. Bruno, M.-C. Wu, P.-˚A. Larson, R. Chaiken, and D. Shakib, " SCOPE: parallel databases meet MapReduce", VLDB Journal, 21(5):611–636, 2012.

[17] J.Leibiusky, G. Eisbruch, and D. Simonassi. "Getting Started with Storm". O'Reilly, 2012.

[18] K.Goodhope, J. Koshy, J. Kreps, N. Narkhede, R. Park, J. Rao, and V. Y. Ye, "Building LinkedIn's real-time activity data pipeline", IEEE Data Engineering Bulletin, 35(2):33–45, 2012.

[19] W.Lam, L. Liu, S. Prasad, A. Rajaraman, Z. Vacheri, and A. Doan. Muppet, " MapReduce-style processing of fast data", Proceedings of the VLDB Endowment (PVLDB), 5(12):1814–1825, 2012.

[20] A.Thusoo et al. "Hive: a warehousing solution over a map-reduce framework", Proceedings of the VLDB Endowment, 2(2):1626–1629, 2009.

[21] A.Thusoo et al. "Hive-a petabyte scale data warehouse using Hadoop", In Proceedings of the 26th IEEE ICDE, pages 996–1005, 2010.

[22] B.He et al. "Mars: a MapReduce framework on graphics processors", In Proceedings of the 17th PACT, pages 260–269, 2008.

[23] C.Olston et al. "Pig latin: a not-so-foreign language for data processing", In Proceedings of the ACM SIGMOD, pages1099–1110, 2008.

[24] R.Chaiken, B. Jenkins, P.Larson, B. Ramsey, D. Shakib. S.Weaver, J. Zhou, "SCOPE: easy and efficient parallel processing of massive data sets", in: Proceedings of VLDB Conference (2008).

[25] Isard, M. et al., "Dryad: distributed data-parallel programs from sequential building blocks", in Proceedings of EuroSys Conference (2007).

[26] Y.Yu, M. Isard, D. Fetterly, M. Budiu, Ú. Erlingsson, P. Gunda, and J. Currey, "DryadLINQ: A System for General-Purpose Distributed Data-Parallel Computing Using a High-Level Language," Symposium on Operating System Design and Implementation (OSDI), CA, December 8-10, 2008.
[27] http://research.microsoft.com/en-us/downloads/03960cab-bb92-4c5c-be23-ce51aee0792c/
[28] J.Gray, A. Szalay, A. Thakar, P. Kunszt, C.Stoughton, D. Slutz, and Vandenberg, J. Data Mining the SDSS SkyServer database. In Distributed Data and Structures 4: Records of the 4th International Meeting, 2002.
[29] L.Feng, Beng Chin Ooi, M. Tamer Özsu, Sai Wu, "Distributed data management using MapReduce", ACM Comput. Surv. 46(3): 31 (2014)
[30] http://hadoop.apache.org/hdfs/
[31] S.Rao, R. Ramakrishnan, A. Silberstein, M. Ovsiannikov, and D. Reeves, "Sailfish: a framework for large scale data processing", In Proc. 3rd ACM Symp. on Cloud Computing. 4:1–4:14.2012
[32] http://skynet.rubyforge.org/
[33] Rasmussen, A., Lam, V. T., Onley, M., Porter, G., Kapoor, R., and Vahdat, a. 2012. Themis: an I/O-efficient MapReduce. In Proc. 3rd ACM Symp. on Cloud Computing. 13:1–13:14.
[34] A.Rasmussen, G. Porter, M. Conley, H. V. Madhyastha, R. N. Mysore, A. Pucher, and A. Vahdat. TritonSort: A Balanced Large-Scale Sorting System. In NSDI, 2011.
[35] H.Liu and D. Orban, "Cloud mapreduce: a mapreduce implementation on top of a cloud operating system," 11th IEEE/ACM International Symposium on Cluster, Cloud and Grid Computing (CCGrid), 2011 .
[36] B.Rashidi, E. Asyabi and J. Talie, "A Comparison of Amazon Elastic Mapreduce and Azure Mapreduce", Elixir International Journal Computer Science and Engineering.
[37] Amazon Web Services LLC, "Amazon Elastic MapReduce (Amazon EMR)," http://aws.amazon.com/elasticMapReduce/,
[38] T.Gunarathne, T. Lon Wu, J. Qiu et al., " MapReduce in the Clouds for Science," 2nd IEEE International Conference on Cloud Computing Technology and Science, Nov. 29- Dec. 1, 2011, Athens, Greece.
[39] K.Beyer, V. Ercegovac, J. Rao, and E. Shekita, "Jaql: A JSON Query Language". URL: http://jaql.org.
[40] Cascading. URL: http://www.cascading.org/.
[41] https://github.com/google/mr4c