SCIENTIFIC REPORTS

OPEN

Inference of Large-scale Time-delayed Gene Regulatory Network with Parallel MapReduce Cloud Platform

Received: 9 August 2018
Accepted: 16 November 2018
Published online: 12 December 2018

Bin Yang[1], Wenzheng Bao[2], De-Shuang Huang[3] & Yuehui Chen[4]

Inference of gene regulatory network (GRN) is crucial to understand intracellular physiological activity and function of biology. The identification of large-scale GRN has been a difficult and hot topic of system biology in recent years. In order to reduce the computation load for large-scale GRN identification, a parallel algorithm based on restricted gene expression programming (RGEP), namely MPRGEP, is proposed to infer instantaneous and time-delayed regulatory relationships between transcription factors and target genes. In MPRGEP, the structure and parameters of time-delayed S-system (TDSS) model are encoded into one chromosome. An original hybrid optimization approach based on genetic algorithm (GA) and gene expression programming (GEP) is proposed to optimize TDSS model with MapReduce framework. Time-delayed GRNs (TDGRN) with hundreds of genes are utilized to test the performance of MPRGEP. The experiment results reveal that MPRGEP could infer more accurately gene regulatory network than other state-of-art methods, and obtain the convincing speedup.

Inferring gene regulatory network (GRN) is the primary and important biochemical network, which contains the regulatory relationships among genes, proteins and small molecules[1,2]. To infer and analyze gene regulatory network could understand the intracellular physiological activity and function of biology, interaction in the pathway and how to make the organism change[3–5]. Time delay is a very important characteristic in biological regulation mechanism, especially for regulation process[6,7]. The proteins translated by transcription factor (TF) regulate the target gene. This regulation process requires a time lag, which involves the regulation of protein translation, folding, nuclear transport, turnover, and the extension of the target mRNA[8,9]. Thus time-delayed factor is critical to gene regulation process. Inferring time-delayed GRN (TDGRN) is one of the major hotspots in system biology[10].

To design gene regulatory network modeling methods need to consider time-delayed factor. The time-delayed versions of GRN modeling methods have been proposed. Li *et al.* proposed a unified approach based on time-delayed correlation algorithm for design of time-delayed gene expression matrix and inference of TDGRN[11]. Ngom *et al.* proposed a new extending version of Bayesian network, namely Max-Min high-order dynamic Bayesian network, to model the time lags between TFs and target genes[12]. Chueh and Lu presented a new method based on time-delay Boolean networks to infer biological pathways[13]. Kordmahalleh *et al.* proposed a hierarchical recurrent neural network (HRNN) to identify TDGRN with time series gene expression data[14].

To understand deeply the specific mathematical relationships between TFs and target genes, differential equation model was proposed to infer GRN[15–18]. Some research added time-delayed factor into differential equation model for TDGRN inference. Chowdhury *et al.* presented time-delayed S-System (TDSS) model to identify simultaneously both instantaneous and time-delayed interactions of TDGRN[19]. But in Chowdhury's method, differential evolution (DE) algorithm was utilized to optimize all parameters in a TDSS model, and the computing load is very large for the large-scale GRN. In order to reduce computing load, we proposed restricted gene expression programming (RGEP) and particle swarm optimization (PSO) to evolve the TDSS model[20]. This method could select TFs automatically and the number of optimized parameters is reduced greatly. However the

[1]School of Information Science and Engineering, Zaozhuang University, Zaozhuang, China. [2]School of Computer Science and Technology, China University of Mining and Technology, Xuzhou, China. [3]Institute of Machine Learning and Systems Biology, Tongji University, Shanghai, China. [4]School of Information Science and Engineering, University of Jinan, Jinan, China. Correspondence and requests for materials should be addressed to W.B. (email: baowz55555@126.com)

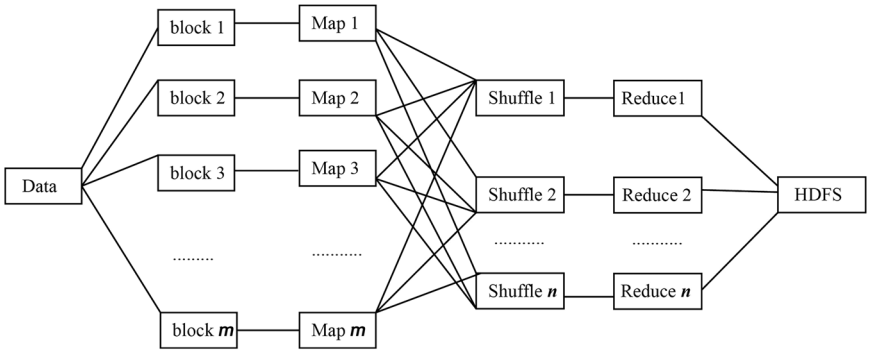

Figure 1. Flowchart of MapReduce framework.

execution time is still unacceptable for GRN inference with hundreds of genes[21–23]. Parallel technology is urgently needed to decrease the computing cost of the algorithm.

MapReduce framework as a parallel programming model is utilized for parallel computation over the past few years[24–26]. Recently many methods based on the MapReduce model have been widely applied in various fields, especially in bioinformatics[27–32]. Hu *et al.* presented a modified variable-length associative sequential pattern discovery (VLASPD) method based on MapReduce model for large-scale protein-protein interactions (PPI) forecasting[33]. Abduallah *et al.* proposed a new MapReduce algorithm based on information-theoretic approach to infer GRN in a cloud environment[34]. You *et al.* e presented a parallel support vector machine (SVM) model based on MapReduce framework to predict the large-scale PPI with the information of protein sequences[35].

In order to decrease the computing cost of large-scale TDGRN identification, this paper proposes a novel MapReduce-based parallel restricted gene expression programming (MPRGEP) algorithm for TDSS model identification. In order to evolve the structure and parameters of TDSS model simultaneously, the structure and parameters are encoded as a chromosome in MPRGEP algorithm. According to partition number, split chromosome population over a cloud computing system's nodes. At each cloud computing node, sub population is optimized iteratively by a novel hybrid evolutionary method based on gene expression programming and genetic algorithm. Then merge them to save as offsprings.

Method

Mapreduce overview. Storage, pretreatment and analysis of biological high-throughput sequencing data have gradually become the main bottleneck of system biology research[36–38]. Hadoop has provided a new solution for big data processing. Hadoop is open-source distributed computing system based on Hadoop Distributed File System (HDFS) and MapReduce framework, and applied to the storage, management and analysis of massive data[39–41]. HDFS is distributed file system, which is utilized to store massive data. MapReduce model is a software framework for big data processing in parallel. MapReduce framework is completed by Map and Reduce operation units, which is described in Fig. 1. In Map phase, input data could be divided into m data blocks. Computing nodes calculate Map function in parallel. The pair output <key, value> of Map function is stored in each computing node. In Reduce phase, all the intermediate results are combined according to key values and generate the final output, which are stored in HDFS.

MapReduce-based restricted gene expression programming algorithm. *Time-Delayed S-system.* Due that time-delayed S-system has high accuracy and flexibility, and contains time-delayed factors, which is suitable for modeling time-delayed systems. *t-th* nonlinear time-delayed differential equation in TDSS model is described as followed[42].

$$\frac{dX_i}{dt} = \alpha_i \prod_{j=1}^{N} X_{j,t-\tau_{g_{ij}}}^{g_{ij}} - \beta_i \prod_{j=1}^{N} X_{j,t-\tau_{h_{ij}}}^{h_{ij}}, \quad i = 1, 2, \ldots N. \tag{1}$$

Where $X_{j,t-\tau_{g_{ij}}}^{g_{ij}}$ is the expression level of gene X_j at $t - \tau_{g_{ij}}$ time point, $\tau_{g_{ij}}$ and $\tau_{h_{ij}}$ are the time-delayed factors, N is the total number of genes in TDGRN, α_i and β_i are rate constants of production function and consumption function, g_{ij} and h_{ij} are kinetic orders.

Chromosome of restricted gene expression programming. In order to better represent and evolve TDSS model, the restricted version of GEP (RGEP) was presented[43]. In RGEP each chromosome of RGEP contains only two genes. An example of RGEP chromosome is described in Fig. 2. The subtraction operator (−) is utilized to connect two genes. Each gene contains head part and tail part, which are created randomly using function set (F) and variable set (T).

$$I_1 = F \bigcup T = \{*1, *2, *3, \ldots, *n\} \bigcup \{x_1, x_2, \ldots, x_m, R\}. \tag{2}$$

Where $*n$ represents the multiplication of n operands, $x_i (i = 1, 2, \ldots m)$ represents the input variable and R denotes constant.

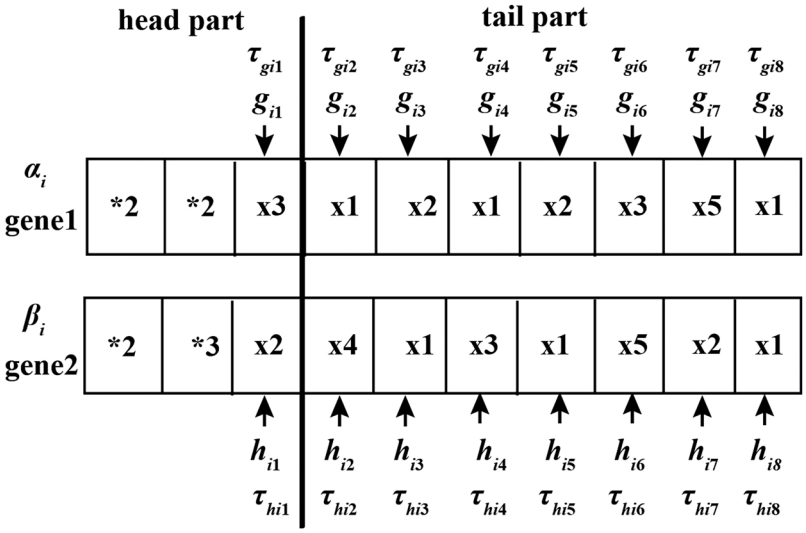

Figure 2. An example of RGEP chromosome with parameters. I_1 is given as $\{*2,\ *3\} \cup \{x_1,\ x_2,\ \ldots,\ x_5\}$.

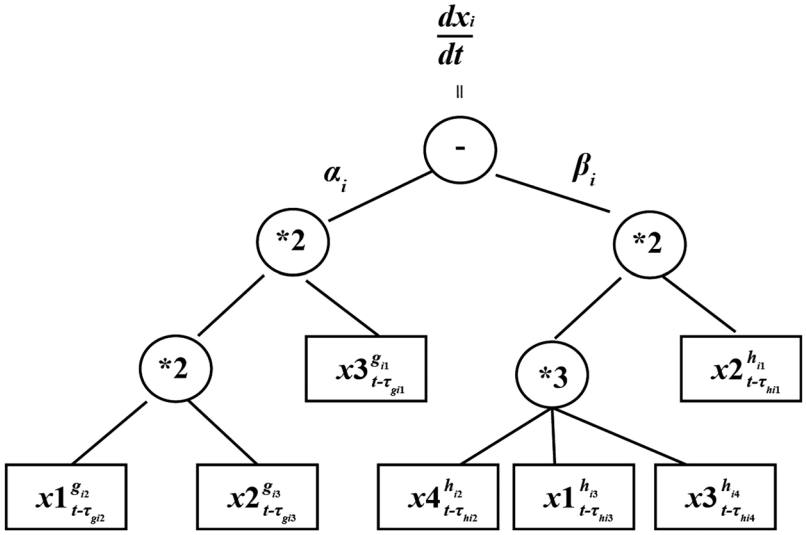

Figure 3. The expression tree of a RGEP chromosome with the parameters.

In each gene, the symbols of head part can be selected from set I_1 randomly. The tail part is created randomly with variable set T only. In advance, the head length (h) is specified for the problems solved. The tail length (t) is calculated according to h.

$$t = (n - 1) \times h + 1 \tag{3}$$

Where n represents the largest number of the operands of functions in set F.

TDSS model has three kinds of parameters: rate constants (α_i and β_i), kinetic orders (g_{ij} and h_{ij}) and time-delayed factor ($\tau_{g_{ij}}$ and $\tau_{h_{ij}}$), so we add these parameters into the chromosome in RGEP. As shown in Fig. 2, gene1 and gene2 are given α_i and β_i, respectively. In each gene, kinetic order (g_{ij} or h_{ij}) and time-delayed factor ($\tau_{g_{ij}}$ or $\tau_{h_{ij}}$) need to be allocated to each terminal node.

Figure 3 describes the arithmetic expression trees (ETs) of Fig. 2. Its decoding differential equation expression is shown as follows.

$$\frac{dx_i}{dt} = \alpha_i x_{3,t-\tau_{g_{i1}}}^{g_{i1}} x_{1,t-\tau_{g_{i2}}}^{g_{i2}} x_{2,t-\tau_{g_{i3}}}^{g_{i3}} - \beta_i x_{2,t-\tau_{h_{i1}}}^{h_{i1}} x_{4,t-\tau_{h_{i2}}}^{h_{i2}} x_{1,t-\tau_{h_{i3}}}^{h_{i3}} x_{3,t-\tau_{h_{i4}}}^{h_{i4}}. \tag{4}$$

Hybrid evolutionary method. In order to search the optimal TDSS model, an original hybrid optimization approach based genetic algorithm[44–46] and gene expression programming[47–49] is proposed in REGP. The structure

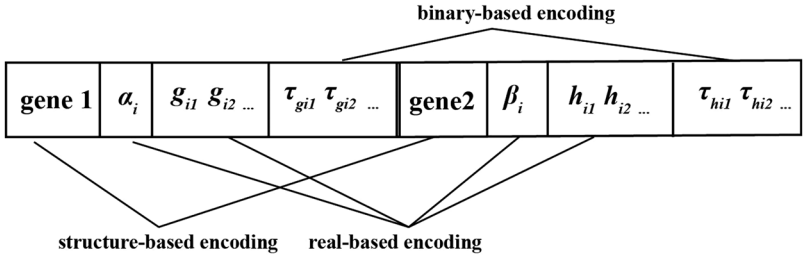

Figure 4. Encoding form of chromosome i in the hybrid evolutionary method.

and parameters in a TDSS model need to be optimized, which are shown in Fig. 3. In our hybrid evolutionary method, two genes of RGEP and parameters are encoded into one chromosome, which is depicted in Fig. 4.

One chromosome contains three kinds of encoding forms. In Fig. 4, gene1 and gene2 are structure-based encoding, rate constants (α_i and β_i) and kinetic orders (g_{ij} and h_{ij}) are real-based encoding, and time-delayed factors ($\tau_{g_{ij}}$ and $\tau_{h_{ij}}$) are binary-based encoding. Single evolution strategy could not reach the optimization purpose, so a hybrid evolutionary method is utilized to reproduce the chromosomes.

(1) Mutation. Mutation probability p_m is defined in advance. According to the encoding case, three mutation strategies are utilized, which are introduced as followed.

(1) Structure-based mutation

- Single-point mutation. The symbols in the head part could be changed to any symbol, which is selected from set I_1 randomly. The symbols in the tail part can only be changed into a symbol from variable set T. Therefore, single-based mutation could create the legal offspring.
- Single-gene mutation. One gene in a chromosome is selected by random, which is replaced by the new gene.
- Change all the variables. All terminal symbols in the structure-coding region are replaced with another terminal symbols.

(2) Real-based mutation

For each real value X in the real-coding region, create a real value r in the interval [0, 1] randomly. If $r < p_m$, real value X could be mutated with the following Equation.

$$X' = X + \delta. \tag{5}$$

Where δ is Gaussian random value.

(3) Binary-based mutation

For each binary value in the binary-coding region, generate a real value r in the interval [0, 1] randomly. If $r < p_m$, the corresponding binary value is inverted.

(2) Crossover. According to the encoding case, three crossover strategies are utilized. First two parents (X and Y) are chosen with the crossover probability p_c, which is defined in advance.

(1) Structure-based crossover

- Single-point recombination. A random point is selected from the structure coding region. Exchange the symbol operators of two parents, which are after this point.
- Single-gene recombination. Two random genes chosen from two parents are swapped.

(2) Real-based crossover

Two parents (X and Y) implement crossover operator with following Equation.

$$X' = X + \gamma(X - Y). \tag{6}$$

$$Y' = Y - \gamma(X - Y). \tag{7}$$

Where $\gamma = 0.99\ \gamma^t$. γ is a variable related to iteration number t. This strategy can change the individuals with a wide range in the early stage of optimization, and protect the better individuals in the later stage.

(3) Binary-based crossover

- Single-point crossover
  A binary point in the binary-coding region is selected randomly. The binary symbols before the point selected are exchanged in order to create the new offsprings.
- Two-point crossover

Select two points in the binary-coding region randomly. The binary string between two points is exchanged between parents.

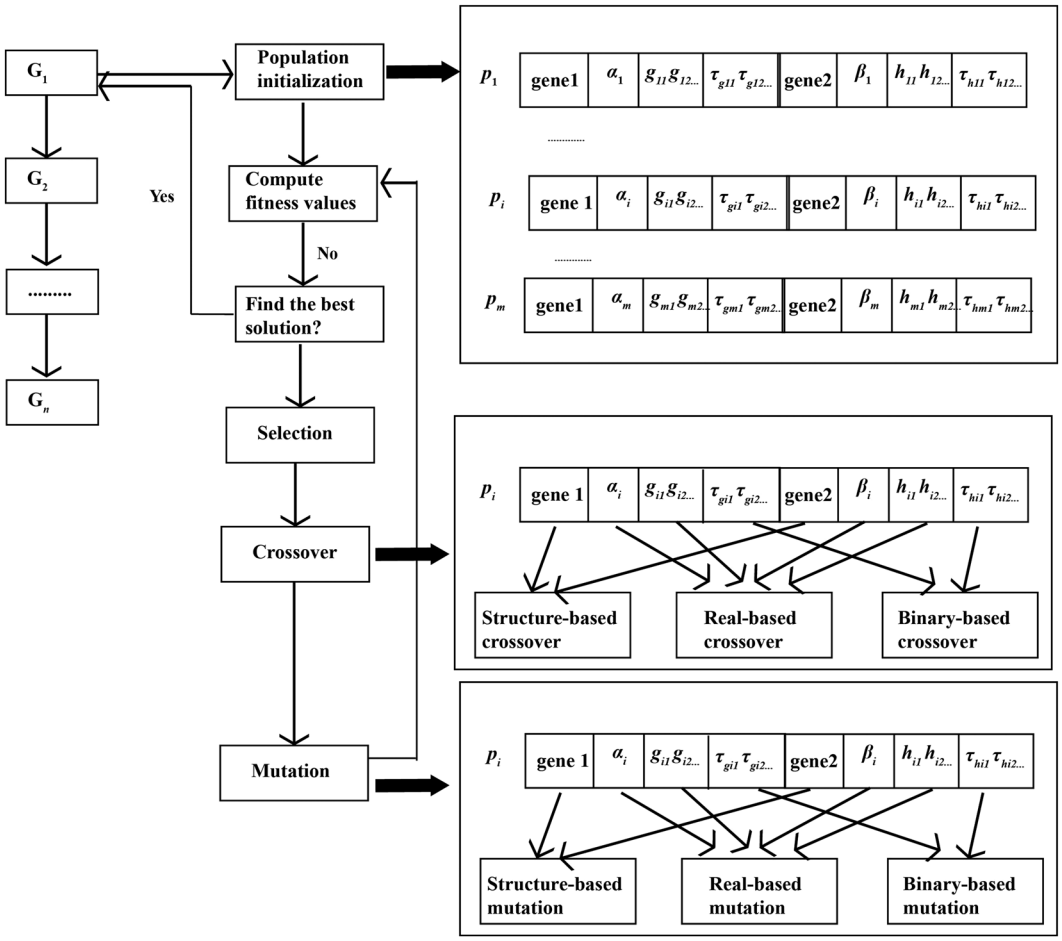

Figure 5. The main flowchart of TDGRN inference.

(3) Selection method. Roulette sampling algorithm is proposed to select the chromosomes to be copied into the next generation according to the fitness values.

Flowchart of time-delayed gene regulatory network inference. Inference flowchart of TDGRN with n genes (G_1, G_2, …, G_n) is depicted in Fig. 5. Decomposition strategy is utilized. From G_1 to G_n, regulatory relationships of each gene are identified by optimizing the TDSS models.

(1) Initialize population (p_1, p_2, …, p_m) containing structure and parameters. The chromosome structure is described in Fig. 5.
(2) The fitness values of all the chromosomes are calculated. If the optimal model is found, stop; otherwise go to (3).
(3) The hybrid evolutionary method is utilized to create the offsprings, which is introduced in Section 2.2.3. According to encoding type, select different crossover and mutation strategies. Go to (2).

Through the optimized TDSS model, gain the regulatory relationships of each gene. Finally the regulatory relationships of all genes constitute gene regulatory network.

MapReduce-based hybrid evolutionary method. To infer large-scale gene regulatory network and reduce high computation load, our hybrid evolutionary method based on Hadoop MapReduce framework is proposed. This framework distributes evolutionary tasks to Map and Reduce modules. Figure 6 shows the hybrid evolutionary framework with the Hadoop MapReduce model.

(1) Input data. The input data are stored on the HDFS, which contain two types of data. The first type of data is chromosome information including the structure and parameters. The second type of data is the fitness value of the corresponding chromosome.
(2) Map phase. Each computation node can operate in Map phase independently, without waiting for other nodes. The task of computing node is to calculate the fitness value f_i of the *i-th* chromosome. The fitness values of all chromosomes are accumulated to obtain *sum_f* for selection operation. According to the input

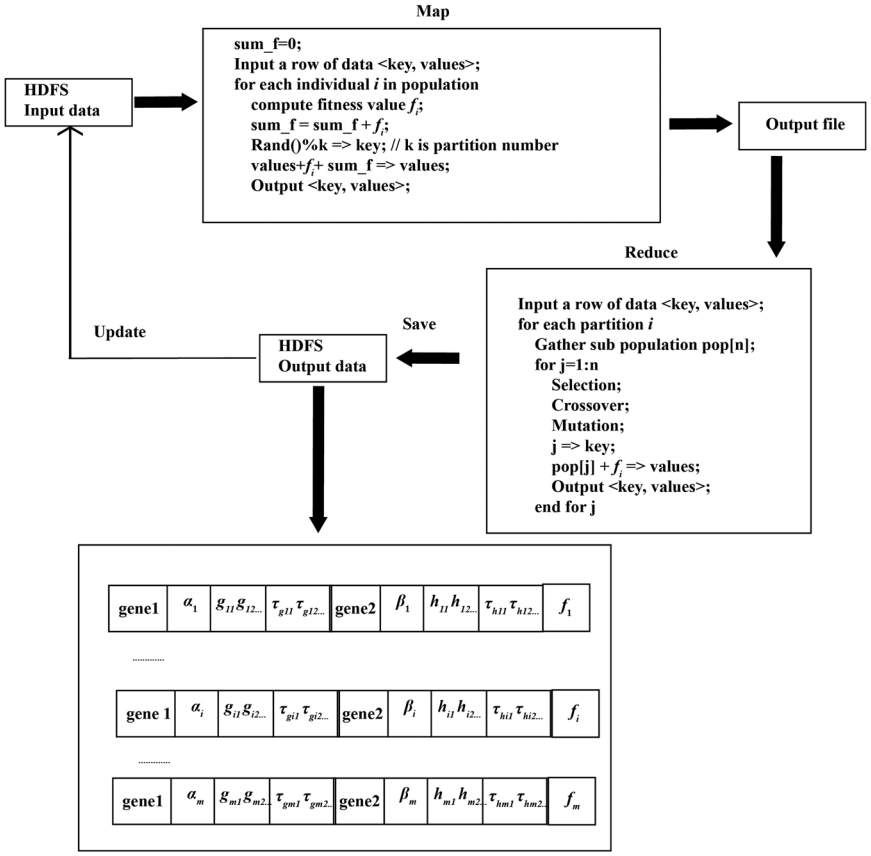

Figure 6. The proposed hybrid evolutionary framework with the Hadoop MapReduce model.

file, the framework divides the chromosome population into computation nodes (Mappers) in order to achieve parallel computing. In order to realize the crossover operation between chromosomes, we randomly divide the population into different partitions. The chromosomes with the same partition id could implement crossover operator. The number of partitions k is defined in advance. The partition id of chromosome *partition_id* is generated randomly, which is set as the key output of Map phase. The chromosome, fitness f_i and total fitness value *sum_f* are set as the value output of Map stage.

(3) Reduce phase. The input data in Reduce phase are from Map phase. After the complete execution of the corresponding Map nodes, the Reduce phase can be executed. In the Reduce phase, the chromosomes with the same *partition_id* are collected into a group, obtaining a sub population. The optimization tasks of sub population are distributed to the same computational node (Reducer). With f_i and *sum_f*, roulette sampling algorithm is utilized to create the offsprings. The individuals in sub population could implement crossover and mutation operator. The gained sub offsprings and fitness values are written to output file of the Reduce phase in order to update the input data on the HDFS. If the number of iterations reaches the termination condition, the algorithm is terminated; otherwise, go to the Map phase.

Experiments

Our proposed parallel algorithm MPRGEP is implemented on MapReduce framework. The hadoop version is 2.6.2 and hadoop cluster consists of one master and 30 slaves. The infrastructure hardware of all nodes is comprised of 3.5 GHz Intel Xeon E5–1620 CPU, 4GB DDR2, and Linux CentOS 6.4 (64-bits). The nodes are connected by local area network with transmission speed of 1,000 Mbps. Three criterions are utilized to evaluate the performance of MPRGEP.

$$S_n = \frac{TP}{TP + FN}. \tag{8}$$

$$S_p = \frac{TN}{FP + TN}. \tag{9}$$

		True GRN: Positive	True GRN: Negative
Inferred GRN	Positive	TP	FP
	Negative	FN	TN

Figure 7. Description of *TP*, *FN*, *FP* and *TN*.

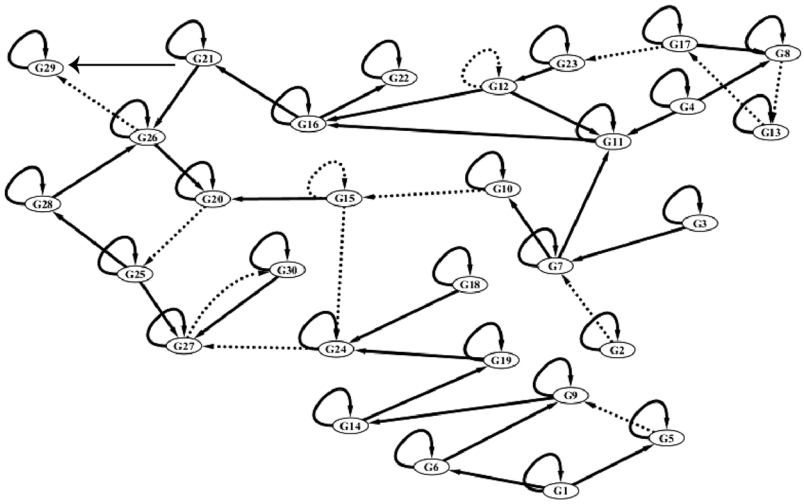

Figure 8. The reconstructed GRN with 30 genes. Solid lines denote the instantaneous regulatory relationships, while dashed lines denote the time-delayed regulatory relationships.

Parameters	Values
Population size	2000
Maximum iteration number	200
Crossover probability p_c	0.7
Mutation probability p_m	0.3
Rate constants (α_i and β_i) interval	[0, 3]
Kinetic orders (g_{ij} and h_{ij}) interval	[0, 1]
Time-delayed factor ($\tau_{g_{ij}}$ and $\tau_{h_{ij}}$) interval	[0, 3]
Partition number k	200

Table 1. Parameters in this experiment.

$$Speedup = \frac{runtime(Single\ node)}{runtime(cluster)}. \tag{10}$$

Where *TP*, *FN*, *FP* and *TN* are presented in Fig. 7.

Artificial dataset. In this part, the parameters of experiments are shown in Table 1, which are selected empirically. The used function set is {*2, *3, *4, *5}. The first artificial dataset is from a 30-gene time-delayed GRN, which is shown in Fig. 8 [19,20]. Kimura's method (S-system model based on decomposition strategy and a cooperative coevolutionary algorithm)[21], DBN (dynamic Bayesian network learned by the likelihood maximization)[22] and TDSS (time-delayed S-system model based on PSO)[23] are also applied for 30-gene artificial TDGRN identification. The averaged performance results of four inferred algorithms are represented in Table 2. From Table 2, it could be seen that MPRGEP has a higher sensitivity (S_n) than other three methods, which reveals that our method can infer more true-positive regulatory relationship. MPRGEP could identify less false-positive regulatory relationships.

	Kimura's method[21]	DBN[22]	TDSS[23]	MPRGEP
S_n	0.8588	0.5662	0.9485	0.971
S_p	0.7096	0.9280	0.9832	0.9832

Table 2. Experiment results for 30-gene artificial TDGRN.

Number of genes	Number of regulatory relationships	Number of time-delayed regulatory relationships
50	128	10
100	212	16
150	345	25

Table 3. Description of three time-delayed gene regulatory networks.

Method	MPRGEP with single node			MPRGEP with computing cluster		
	S_n	S_p	Runtime (m)	S_n	S_p	Runtime (m)
50 genes	0.5781	0.9483	172.3	0.5859	0.9432	44.5
100 genes	0.533	0.9237	728.7	0.5283	0.9223	115.7
150 genes	0.469	0.905	1192.6	0.4753	0.909	159.0

Table 4. Performance of three TDGRNs by running MPRGEP.

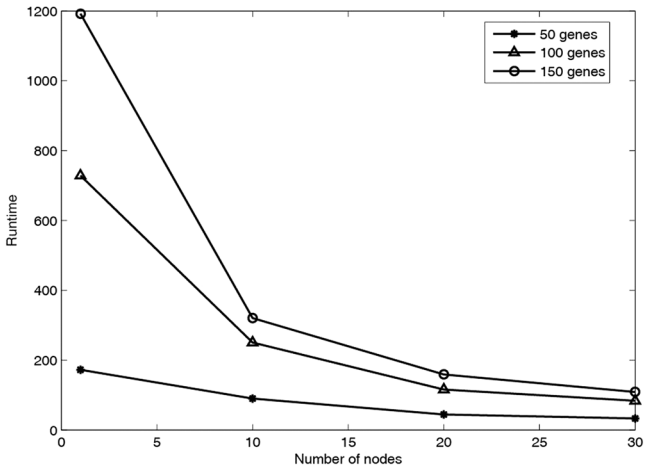

Figure 9. Runtime performance of MPRGEP for three TDGRNs inference.

The open-source software GeneNetWeaver 3.1 is utilized to generate three yeast S.cerevisiae sub gene regulatory networks with 50 genes, 100 genes and 150 genes, respectively. Time-delayed regulatory relationships are created randomly and time-delayed values are selected from [0, 3]. Three time-delayed gene regulatory networks are described in Table 3. Initial conditions are randomly generated. For each network, 10 time-series datasets are generated and each dataset contains 21 time points from 0 to 20.

Our method is executed in the single machine and computing clusters with 20 computing nodes, respectively. Through several runs, the averaged performances are listed in Table 4. From the inference results, we know that MPRGEP not only can solve large-scale time-delayed gene regulatory network, but also perform well in terms of S_n and S_p. Table 4 also reveals that MapReduce framework could reduce running time of GRN inference, which makes it possible to identify large-scale GRN with more genes.

In order to validate the parallel computing performance, MPRGEP algorithm is utilized to identify three above time-delayed GRNs in three computing clusters with 10, 20 and 30 nodes, respectively. The runtime and speedup performance are depicted in Figs 9 and 10. From Fig. 9, it could be seen clearly that as the number of genes rises, the running time also rises. With the increment of computing nodes, the running time decreases. Figure 10 shows that as the number of computing nodes increases, our proposed parallel algorithm accelerates significantly. The best speedup performance of MPRGEP is the case that MPRGEP is run on 30 computing nodes to infer GRN with 150 genes. The speedup curve is not linear because of serial bottlenecks and infrastructure barriers in MapReduce framework.

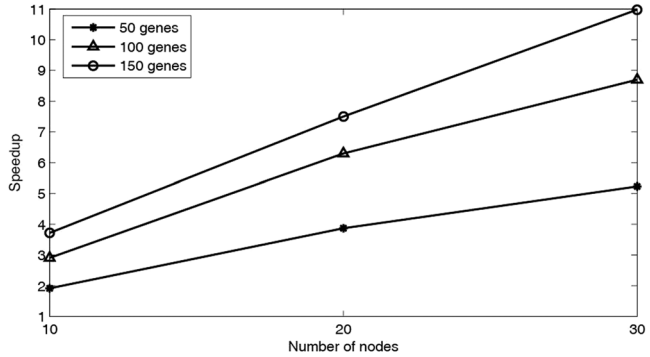

Figure 10. Speedup performance of MPRGEP for three TDGRNs inference.

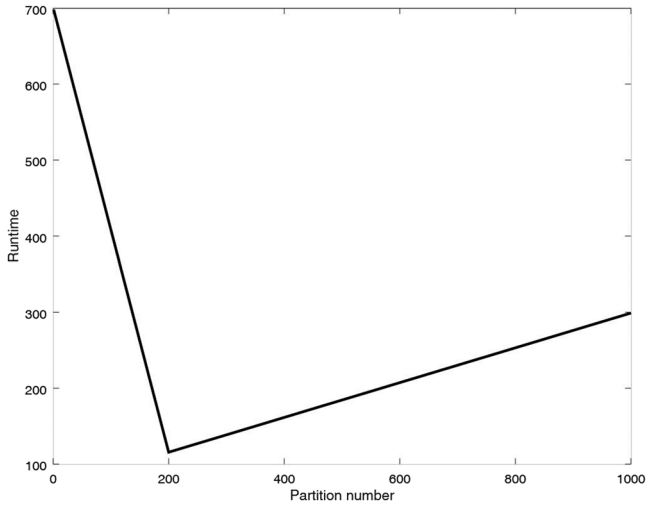

Figure 11. Performance of MPRGEP with different partition numbers.

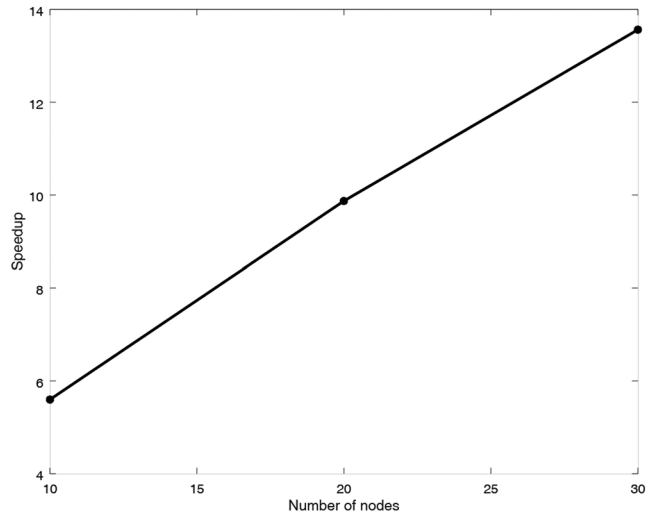

Figure 12. Speedup performance of MPRGEP for GRN inference with 500 genes.

In MPRGEP, the computational tasks of hybrid evolutionary algorithm are mainly concentrated in the Reduce phase. The sub population with the same partition id will be assigned to the same Reduce for optimization. If the number of Reducers is fixed in advance, the number of partitions can affect the speed of parallel computation.

We make the experiments with three partition numbers, 1, 200 and 1000. Node number in the computing cluster is set as 20. The running time is depicted in Fig. 11. From the result, we can see that the hybrid evolutionary algorithm performs best when partition number is set as 200. When the partition number is 1, the sub population contains all the population and is optimized in one Reducer. Parallel strategy doesn't work. When the number of partition number is given to 1000, the number of sub populations is too large. In this case, more Reducers are needed. The allocation and merging of resources could waste more time.

Real biology dataset. In this part, the dataset is from the Gene Expression Omnibus (GEO) at http://www.ncbi.nlm.nih.gov/geo/ (GEO accession: GSE30052)[34,50]. This dataset contains 5,744 probe sets, 10,928 genes and 49 time points. In order to validate the parallel performance of MPRGEP, one subset from this dataset is extracted, containing 500 genes. The experiment is executed in the computing clusters with 20 nodes. The parameters are also from Table 1. The running results are described in Fig. 12. From Fig. 12, it can be seen that our method could be accelerated evidently.

Discussion and Conclusion

With the rapid development of biotechnology, gene regulatory networks inferred contain more genes, so there is necessity for developing advanced computational algorithm to infer gene regulatory network with gene expression data. This paper proposes time-delayed S-system model to model instantaneous and time-delayed regulation interactions in time-delayed gene regulatory network. A novel MapReduce-based parallel restricted gene expression programming (MPRGEP) algorithm is utilized for TDSS model identification. The experiment results reveal that our parallel algorithm is promising in terms of accuracy and speedup when used to infer large-scale TDGRN.

References

1. Kaern, M., Blake, W. J. & Collins, J. J. The engineering of gene regulatory networks. *Annu Rev Biomed Eng.* **5**, 179–206 (2003).
2. Park, J., Ogunnaike, B., Schwaber, J. & Vadigepalli, R. Identifying functional gene regulatory network phenotypes underlying single cell transcriptional variability. *Prog Biophys Mol Bio.* **117**, 87–98 (2015).
3. Schlitt, T. & Brazma, A. Current approaches to gene regulatory network modeling. *BMC Bioinformatics.* **8**, S9 (2007).
4. Madhamshettiwar, P. B., Maetschke, S. R., Davis, M. J., Reverter, A. & Ragan, M. A. Gene regulatory network inference: evaluation and application to ovarian cancer allows the prioritization of drug targets. *Genome Med.* **4**, 41 (2012).
5. Yang, B. *et al*. HSCVFNT: Inference of Time-Delayed Gene Regulatory Network Based on Complex-Valued Flexible Neural Tree Model. *Int. J. Mol. Sci.* **19**, 3178 (2018).
6. Parmar, K., Blyuss, K. B., Kyrychko, Y. N. & Hogan, S. J. Time-Delayed Models of Gene Regulatory Networks. *Comput Math Methods Med.* **2015**, 1–16 (2015).
7. Wang, G., Yin, L., Zhao, Y. & Mao, K. Efficiently mining time-delayed gene expression patterns. *IEEE Trans Syst Man Cybern B Cybern.* **40**, 400–11 (2010).
8. Orosz, G., Moehlis, J. & Murray, R. M. Controlling biological networks by time-delayed signals. *Philos Trans A Math Phys Eng Sci.* **368**, 439–54 (2010).
9. Chaturvedi, I. & Rajapakse, J. C. Detecting robust time-delayed regulation in Mycobacterium tuberculosis. *BMC Genomics.* **10**, S28 (2009).
10. Huang, T. *et al*. Using GeneReg to construct time delay gene regulatory networks. *BMC Res Notes.* **3**, 142 (2010).
11. Li, X. *et al*. Discovery of Time-Delayed Gene Regulatory Networks based on temporal gene expression profiling. *BMC Bioinformatics.* **7**, 26 (2006).
12. Li, Y., Chen, H., Zheng, J. & Ngom, A. The Max-Min High-Order Dynamic Bayesian Network for Learning Gene Regulatory Networks with Time-Delayed Regulations. *IEEE/ACM Trans Comput Biol Bioinform.* **13**, 792–803 (2016).
13. Chueh, T. H. & Lu, H. H. S. Inference of Biological Pathway from Gene Expression Profiles by Time Delay Boolean Networks. *PLoS One.* **7**, e42095 (2012).
14. Kordmahalleh, M. M., Sefidmazgi, M. G., Harrison, S. H. & Homaifar, A. Identifying time-delayed gene regulatory networks via an evolvable hierarchical recurrent neural network. *BioData Min.* **10**, 29 (2017).
15. Cao, J., Qi, X. & Zhao, H. Modeling gene regulation networks using ordinary differential equations. *Methods Mol Biol.* **802**, 185–97 (2012).
16. Gebert, J., Radde, N. & Weber, G. W. Modeling gene regulatory networks with piecewise linear differential equations. *European Journal of Operational Research.* **181**, 1148–1165 (2007).
17. Sakamoto, E. & Iba, H. Identifying gene regulatory network as differential equation by genetic programming. *Genome Informatics.* **11**, 281–283 (2000).
18. Wu, H., Lu, T., Xue, H. & Liang, H. Sparse Additive Ordinary Differential Equations for Dynamic Gene Regulatory Network Modeling. *J Am Stat Assoc.* **109**, 700–716 (2014).
19. Chowdhury, A. R., Chetty, M. & Vinh, N. X. Incorporating time-delays in S-System model for reverse engineering genetic networks. *BMC Bioinformatics.* **14**, 196 (2013).
20. Yang, B., Zhang, W., Wang, H. F., Song, C. D. & Chen, Y. H. TDSDMI: Inference of time-delayed gene regulatory network using S-system model with delayed mutual information. *Computers in Biology and Medicine.* **72**, 218–225 (2016).
21. Kimura, S., Ide, K. & Kashihara, A. Inference of S-system models of genetic networks using a cooperative coevolutionary algorithm. *Bioinformatics.* **21**, 1154–1163 (2005).
22. Perrin, B. E. *et al*. Gene networks inference using dynamic Bayesian networks. *Bioinformatics.* **19**, 138–148 (2003).
23. Yang, B., Zhang, W., Yan, X. F. & Liu, C. X. Reverse engineering of time-delayed gene regulatory network using restricted gene expression programming. *Advances in Intelligent Systems and Computing.* **420**, 155–165 (2016).
24. Babu, S. Towards automatic optimization of MapReduce programs. A*cm Symposium on Cloud Computing*. 137–142 (2010).
25. Dean, J. & Ghemawat, S. MapReduce: A Flexible Data Processing Tool. *Communications of the Acm.* **53**, 72–77 (2010).
26. Liu, Y. *et al*. MapReduce Based Parallel Neural Networks in Enabling Large Scale Machine Learning. *Comput Intell Neurosci.* **2015**, 297672 (2015).
27. Vasciaveo, A. *et al*. A cloud-based approach for Gene Regulatory Networks dynamics simulations. *4th Mediterranean Conference on Embedded Computing*. 72–76 (2015).
28. Langmead, B., Hansen, K. D. & Leek, J. T. Cloud-scale RNA-sequencing differential expression analysis with Myrna. *Genome Biology.* **11**, R83 (2010).
29. Li, Z. *et al*. Enabling big geoscience data analytics with a cloud-based, MapReduce-enabled and service-oriented workflow framework. *PLoS One.* **10**, e0116781 (2015).

30. Liao, R., Zhang, Y., Guan, J. & Zhou, S. CloudNMF: a MapReduce implementation of nonnegative matrix factorization for large-scale biological datasets. *Genomics Proteomics Bioinformatics.* **12**, 48–51 (2014).
31. Kumar, M., Rath, N. K. & Rath, S. K. Analysis of microarray leukemia data using an efficient MapReduce-based K-nearest-neighbor classifier. *J Biomed Inform.* **60**, 395–409 (2016).
32. Mohammed, E. A., Far, B. H. & Naugler, C. Applications of the MapReduce programming framework to clinical big data analysis: current landscape and future trends. *BioData Min.* **7**, 22 (2014).
33. Hu, L., Yuan, X., Hu, P. & Chan, K. C. C. Efficiently predicting large-scale protein-protein interactions using MapReduce. *Comput Biol Chem.* **69**, 202–206 (2017).
34. Abduallah, Y. *et al*. MapReduce Algorithms for Inferring Gene Regulatory Networks from Time-Series Microarray Data Using an Information-Theoretic Approach. *Biomed Res Int.* **2017**, 1–8 (2017).
35. You, Z. H., Yu, J. Z., Zhu, L., Li, S. & Wen, Z. K. A MapReduce based parallel SVM for large-scale predicting protein-protein interactions. *Neurocomputing.* **145**, 37–43 (2014).
36. Wade, J. T. Mapping Transcription Regulatory Networks with ChIP-seq and RNA-seq. *Adv Exp Med Biol.* **883**, 119–34 (2015).
37. Finotello, F. & Di Camillo, B. Measuring differential gene expression with RNA-seq: challenges and strategies for data analysis. *Brief Funct Genomics.* **14**, 130–42 (2015).
38. Liu, Y., Zhou, J. & White, K. P. RNA-seq differential expression studies: more sequence or more replication? *Bioinformatics.* **30**, 301–4 (2014).
39. White, T. Hadoop: the definitive guide 15–362 (O'Reilly Media, Inc., 2009).
40. Shvachko, K., Kuang, H., Radia, S. & Chansler, R. The Hadoop Distributed File System. IEEE 26th Symposium on Mass Storage Systems and Technologies. 1–10 (2010).
41. Taylor, R. C. An overview of the Hadoop/MapReduce/HBase framework and its current applications in bioinformatics. *Bmc Bioinformatics.* **11**, S1 (2010).
42. Chowdhury, A. R., Chetty, M. & Vinh, N. X. Reverse Engineering Genetic Networks with Time-Delayed S-System Model and Pearson Correlation Coefficient. *Lecture Notes in Computer Science.* **8227**, 624–631 (2013).
43. Yang, B., Liu, S. & Zhang, W. Reverse engineering of gene regulatory network using restricted gene expression programming. *J Bioinform Comput Biol.* **14**, 1650021 (2016).
44. Herrera, F., Lozano, M. & Verdegay, J. L. Tackling Real-Coded Genetic Algorithms: Operators and Tools for Behavioural Analysis. *Artificial Intelligence Review.* **12**, 265–319 (1998).
45. Goldberg, D. E. Genetic Algorithm in Search Optimization and Machine Learning 30–254 (Addison-Wesley Longman Publishing Co., Inc, 1989).
46. Gai, K., Qiu, M. & Zhao, H. Cost-Aware Multimedia Data Allocation for Heterogeneous Memory Using Genetic Algorithm in Cloud Computing. *IEEE Transactions on Cloud Computing.* **99**, 1–1 (2016).
47. Ferreira, C. Gene Expression Programming: a New Adaptive Algorithm for Solving Problems. *Computer Science.* **21**, 87–129 (2001).
48. Zhang, Y. *et al*. Using gene expression programming to infer gene regulatory networks from time-series data. *Comput Biol Chem.* **47**, 198–206 (2013).
49. Tang, L., Yang, C. & Li, W. Adopting gene expression programming to generate extension strategies for incompatible problem. *Neural Computing & Applications.* **28**, 1–16 (2016).
50. Chin, S. L., Marcus, I. M., Klevecz, R. R. & Li, C. M. Dynamics of oscillatory phenotypes in Saccharomyces cerevisiae reveal a network of genome-wide transcriptional oscillators. *FEBS Journal.* **279**, 1119–1130 (2012).

Acknowledgements

This work was supported by the Natural Science Foundation of China (No. 61702445), Shandong Provincial Natural Science Foundation, China (No. ZR2015PF007), the PhD research startup foundation of Zaozhuang University (No. 2014BS13), and Zaozhuang University Foundation (No. 2015YY02).

Author Contributions

B.Y. conceived the method. Y.C. and D.H. designed the method. W.B. conducted the experiments and wrote the main manuscript text. All authors reviewed the manuscript.

Additional Information

Competing Interests: The authors declare no competing interests.

Publisher's note: Springer Nature remains neutral with regard to jurisdictional claims in published maps and institutional affiliations.