Information Sciences 181 (2011) 106–114

Contents lists available at ScienceDirect

Information Sciences

journal homepage: www.elsevier.com/locate/ins

Time-series forecasting using a system of ordinary differential equations

Yuehui Chen [a,*], Bin Yang [a], Qingfang Meng [a], Yaou Zhao [a], Ajith Abraham [b]

[a] *Computational Intelligence Lab, School of Information Science and Engineering, University of Jinan, 106 Jiwei Road, 250022 Jinan, PR China*
[b] *Machine Intelligent Research Labs (MIR Labs), Scientific Network for Innovation and Research Excellence, USA*

ARTICLE INFO

Article history:
Received 7 July 2009
Received in revised form 30 August 2010
Accepted 1 September 2010

Keywords:
Hybrid evolutionary method
Network traffic
Small-time scale
The additive tree models
Ordinary differential equations
Particle swarm optimization

ABSTRACT

This paper presents a hybrid evolutionary method for identifying a system of ordinary differential equations (ODEs) to predict the small-time scale traffic measurements data. We used the tree-structure based evolutionary algorithm to evolve the architecture and a particle swarm optimization (PSO) algorithm to fine tune the parameters of the additive tree models for the system of ordinary differential equations. We also illustrate some experimental comparisons with genetic programming, gene expression programming and a feed-forward neural network optimized using PSO algorithm. Experimental results reveal that the proposed method is feasible and efficient for forecasting the small-scale traffic measurements data.

1. Introduction

Network traffic analysis and modeling play a major role in characterizing network performance and hence it has been a recent focus of many research works. Models that accurately capture the salient characteristics of the traffic is useful for analysis and simulation, and they also provide a better understanding of network dynamics. It is also useful for network design and engineering problems, e.g., the traffic balance scheme, router, switcher designing, the management of devices and its supporting software development etc.

Complexity is a key issue in network geometry and information traffic. Evidence of traffic complexity appears in many forms, such as the long-range correlations and self-similarities found in the statistical analysis of traffic measurements. There is also strong evidence of these phenomena occurring at several different time scales. The complexity revealed from the traffic measurements has led to the suggestion that the network traffic cannot be analyzed within the framework of available traffic models [9,13,15]. Alternative reliable traffic models and tools for quality assessment and control should be developed [7,18,19,26].

Recently, communication and network technologies are developing rapidly, which prompts the traffic characteristics to change abruptly. The research emphasis of the network traffic analysis and modeling has to deal with large-time scale to smaller-time scale systems. Some recent research works have also illustrated that the traffic characteristics of the small-time scale systems were different from those of the large-time scale systems [25,27]. So the large-time scale network traffic models cannot be suitable for the small-time scale network traffic.

Some researchers have proposed reasonable mathematical models based on the observed time series data so as to provide system analysis and prediction in various application domains [1,4,5,8,11,12,16,20]. Mathematical modeling is the art of

* Corresponding author.
E-mail addresses: yhchen@ujn.edu.cn (Y. Chen), batsi@126.com (B. Yang), ise_mengqf@ujn.edu.cn (Q. Meng), ise_zhaoyo@ujn.edu.cn (Y. Zhao), ajith.abraham@ieee.org (A. Abraham).

0020-0255/$ - see front matter © 2010 Elsevier Inc. All rights reserved.
doi:10.1016/j.ins.2010.09.006

translating problems from an application area into tractable mathematical formulations, whose theoretical and numerical analysis provides insight, answers, and guidance, useful to understand the original application [12]. The system of differential equations can describe the dynamic properties of a system, which changes with time quite well and predict the future states of the system very conveniently. Cao et al. [3] used the ordinary differential equations (ODE) to predict the populations of the United States from 1790 to 1950, and the result indicate that the ODE was a powerful model in the discovery of sciential laws for dynamic data. In this research, we use the ODE model to predict the small-time scale traffic measurements data.

Various methods are proposed to infer ODEs during the last few years [3,10,24]. Most of these works can be classified into two groups: the first group is to identify the parameters of the ODEs and the second group is to identify the structure. The former is illustrated by genetic algorithms (GA), and the latter by the genetic programming (GP) approach. Cao et al. used GP to evolve the ODEs from the observed time series [3]. The main idea is to embed a genetic algorithm (GA) in genetic programming (GP), where GP is employed to discover and optimize the model's structure, while the GA is employed to optimize its parameters. Authors illustrated that the GP-based approach introduce numerous advantages over other modeling methods. Tsoulos and Lagar proposed a novel method based on the grammatical evolution [24]. The method forms generations of trial solutions expressed in an analytical closed form. Iba proposed an ODE identification method by using the least mean square (LMS) along with the ordinary GP [10]. Some individuals are created by the LMS method at some intervals of generations and they replace the worst individuals in the population.

We also proposed a new representation scheme of the additive models for the system identification especially the reconstruction of polynomials and the identification of linear/nonlinear systems. This model is robust, and it is easy to analyze by traditional techniques. This is because the evolved additive tree model is simple and is very near to the traditional representation of the system to be reconstructed [6] and the computational complexity is similar to the GP.

In this paper, we proposed a hybrid evolutionary method, in which the tree-structure based evolution algorithm and particle swarm optimization (PSO) are employed to evolve the architecture and the parameters of the additive tree models for system of ordinary differential equation identification. The partitioning [2] is used in the process of identification of the structure of the system. Thus a ODE (Eq. (1)) containing some other variables can be evolved and is convenient and effective for a simple variable prediction.

$$\dot{Y} = f(X_1, X_2, \ldots, X_n). \tag{1}$$

The paper is organized as follows. In Section 2, we describe the details of the proposed method. In Section 3, four examples are used to examine the effectiveness of the proposed method and finally Conclusions are drawn in Section 4.

2. Representation of additive tree model

We use the tree-structure based evolutionary algorithm to evolve the architecture of the additive tree models for the system of ordinary differential equation identification. For this purpose, we encode the right-hand side of an ODE into an additive tree individual as illustrated in Fig. 1.

Two instruction/operator sets I_0 and I_1 are used to generate the additive tree.

$$\begin{cases} I_0 = \{+_2, +_3, , \ldots, +_N\}, \\ I_1 = F \cup T = \{, /, sin, cos, exp, rlog, x, R\}, \end{cases} \tag{2}$$

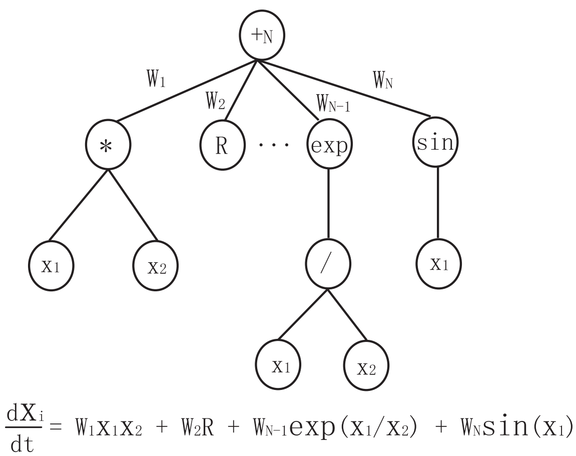

Fig. 1. Example of a ODEs in the form of the additive tree model.

where $F = \{*, /, sin, cos, exp, rlog\}$ and $T = \{x, R\}$ are functions and terminal sets. $+N$, $*$, $/$, sin, cos, exp, $rlog$, x, *and* R denote the addition, multiplication, protected division ($\forall\ x, y \in R$: when $y = 0, x/0 = 1$), sine, cosine, exponent, protected logarithm ($\forall x \in R, x \neq 0$: $rlog(x) = log(abs(x))$ and $rlog(0) = 0$), system inputs, and random constant number, and taking N, 2, 2, 1, 1, 1, 1, 0 and 0 arguments respectively [6].

N is an integer number (the maximum number of ODE terms), I_0 is the instruction set and the root node, and the instructions of other nodes are selected from the instruction set I_1. Note that if the right-hand side of ODEs is a polynomial, then the instruction set I_1 can be defined as $I_1 = \{*2, *3, \ldots, *n, x_1, x_2, \ldots, x_n, R\}$.

We infer the system of ODEs using a partitioning scheme, in which equations describing each variable of the system is inferred separately. This method can also significantly reduce the research space. When using partitioning, a candidate equation for a signal variable is integrated by substituting references to other variables with data from the observed time series [2]. Thus each right-hand side of the ODE system, is evolved independently in parallel. The ODE can contain some unknown variables and it is convenient and effective for simple variable prediction.

3. The proposed hybrid method

3.1. Structure optimization of models

Finding an optimal or near-optimal additive tree model is formulated as an evolutionary search process. We used the additive tree operators as following:

(1) Mutation. We choose three mutation operators to generate offsprings from the parents and the operation is described as follows:
   (1) Change one terminal node: randomly select one terminal node in the tree and replace it with another terminal node, which is generated randomly.
   (2) Grow: select a random leaf in the hidden layer of the tree and replace it with a newly generated subtree.
   (3) Prone: randomly select a function node in a tree and replace it with a terminal node selected in the set T.
   The additive tree operators are applied to each parent in the population to generate an offspring using the following steps: (a) A Poission random number N, with mean λ is generated. (b) N random mutation operators are uniformly selected with replacement from the above mentioned mutation operator set. (c) These N mutation operators are applied in sequence one after the other to the parent to get new offsprings.
(2) Crossover. First two parents are selected according to the predefined crossover probability P_c and select one nonterminal node in the hidden layer for each additive tree randomly, and then swap the selected subtree.
(3) Selection. Evolutionary programming (EP) style tournament selection [17] is applied to select the parents for the next generation. Pairwise comparison is conducted for the union of μ parents and μ offsprings. For each individual, q opponents are chosen uniformly at random from all the parents and offspring. For each comparison, if the individual's fitness is not smaller than the opponent's, it receives a selection. Select μ individuals from the parents and offsprings, which has most wins to form the next generation. This is repeated in each generation until a predefined number of generations has reached or the best structure is found.

3.2. Parameter optimization of models using PSO

According to Fig. 1, we check all the parameters contained in each equation, namely by counting their number n_i ($i = 1, 2, \ldots, N$, N is the number of the equations).

According to n_i, the particles are randomly generated initially. Each particle x_i represents a potential solution. A swarm of particles moves through space, with the moving velocity of each particle represented by a velocity vector v_i. At each step, each particle is evaluated and keep track of its own best position, which is associated with the best fitness it has achieved so far in a vector $Pbest_i$. The best position among all the particles is kept as Gbest [14]. A new velocity for particle i is updated as follows:

$$v_i(t+1) = v_i(t) + c_1 r_1 (Pbest_i - x_i(t)) + c_2 r_2 (Gbest(t) - x_i(t)), \tag{3}$$

where c_1 and c_2 are positive constants and r_1 and r_2 are uniformly distributed random numbers within the range of $[0, 1]$. Based on the updated velocities, each particle changes its position according to the following equation:

$$x_i(t+1) = x_i(t) + v_i(t+1). \tag{4}$$

3.3. Fitness definition

A fitness function maps ODE to a scalar and real fitness value that reflects the ODE's performances on a given task. The normalized mean squared error (*NMSE*) and the root mean square error (*RMSE*) are usually used as the fitness function for functions approximation type problems an we also used them to evaluate the performance of the ODE.

$$NMSE = \frac{\frac{1}{N-1}\sum_{k=1}^{N-1}\left(x'(t_0+k\Delta t)-x(t_0+k\Delta t)\right)^2}{\frac{1}{N}\sum_{k=1}^{N-1}\left(x(t_0+k\Delta t)-\bar{x}\right)^2}, \tag{5}$$

$$RMSE = \sqrt{\frac{1}{N-1}\sum_{k=1}^{N-1}\left(x'(t_0+k\Delta t)-x(t_0+k\Delta t)\right)^2}, \tag{6}$$

where N is the number of the time series, t_0 is the starting time, $\triangle t$ is the step size, T is the number of the data point, $x(t_0+k\triangle t)$ is the actual output of the ODE sample, and $x'(t_0+k\triangle t)$ is the ODE output with the initial condition $x'(t_0+(k-1)\triangle t)$ integrating the best ODE, and $\bar{x}$ is the average output traffic data. All outputs are calculated by using the approximate forth-order Runge–Kutta method. When calculating the outputs, some individuals may cause overflow. In this case, the individuals with inappropriate fitness values are weeded out from the population.

3.4. Summary of algorithm

The algorithmic steps for the optimal design of each ODE is summarized as follows:

(1) Set the initial values of parameters used in the proposed hybrid method. Create the initial population randomly (structures and their corresponding parameters);
(2) Structure optimization is achieved by the additive tree variation operators as described in Subsection 2.1 in which the fitness function is calculated by the normalized mean squared error (*NMSE*) or root mean square error (*RMSE*);
(3) At some interval of generations, select the better structures to optimize parameters. Parameter optimization is achieved by PSO as described in Subsection 2.2. During the parameter optimization process, the structure is fixed. All the parameters used To form a better structure now formulates a parameter vector, which is to be optimized by PSO.
(4) If the maximum number of generations is reached or a satisfactory solution is found, then stop; otherwise go to step (2).

4. System for Prediction

After obtaining the best model, we then input the last line (feature vector) of the training data as the initial conditions of the best ODE to get the predicted time series of the system. The error is calculated by (5), (6). The process is described in Fig. 2.

5. Experimental results and analysis

To test the effectiveness of the proposed method, we uses the TCP traffic data, which is published by the Lawrence Berkeley Laboratory. This traffic data contain an hour's worth of all wide-area traffic between Digital Equipment Corporation

Procedure System of ODE for Prediction

begin
  let X^* , X be $m*n$ matrice as the targeted test data, assign the last row of the training data as the initial condition to Y
  for i:=1 to m do
    begin
      integrate the system of the best ODE for a step with some numerical integration methods;
      assign the solution to the i-th row of X;
      take the i-th row of X^* as the Y;
    end
  The error can be calculated integrating eqs.5,6 with the matic X and X^*;
end

Fig. 2. Procedure system of ODE for prediction.

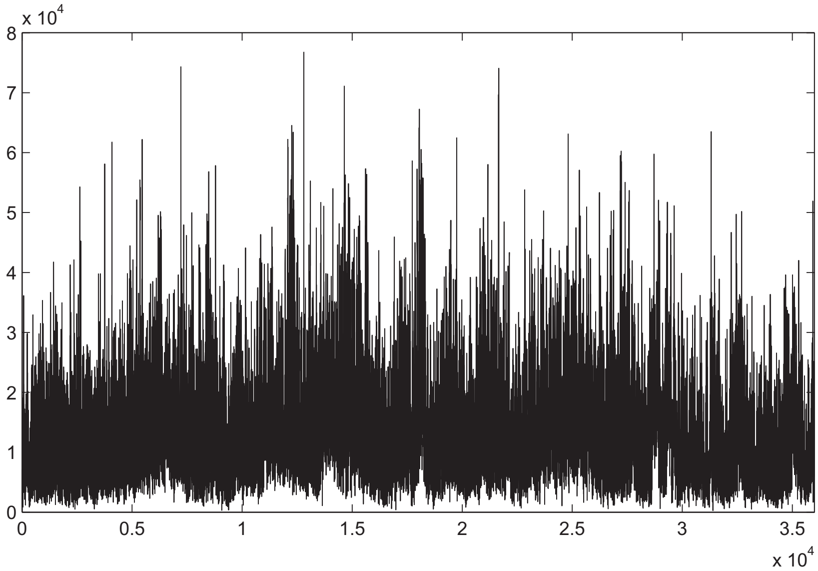

Fig. 3. Actual traffic measurements data.

and the rest of the world. The data package used is DEC-Pkt1, and the time stamps have millisecond precision [23].The traffic data aggregated with time bin 0.1 s, that is the number of packages arrived within the 0.1 s time interval, are shown in Fig. 3.

In general, the traffic measurements can be considered as a sum of a regular process and a stochastic part, which are related to the high-frequency noise. The elimination of the noise may simplify the analyzed time series, so we apply the wavelet soft threshold noise reduction method to this data. The difference between the original time series and the filtered signal, corresponds to the noisy component. Fig. 3 illustrates the original traffic series and Fig. 4 presents the corresponding filtered signal and Fig. 5 depicts the noisy component. The filtered traffic measurements data are normalized within the interval $[0,1]$ using the following equation.

$$x' = \frac{x - x_{min}}{x_{max} - x_{min}}. \tag{7}$$

Some of our past research illustrated that the network traffic time series are of nonlinear nature [21,22], so we use the system of ordinary differential equations to predict the network traffic time series. The length of this traffic measurements data is 36,000. The first 33,000 data points are used as the training set, and the last 3000 data points are used as the test set.

The parameter settings in this experiment is shown in Table 1. *Population size* is the number of individuals of initial population, *Generation* is the maximum number of generations, *Crossover rate* is the predefined crossover probability P_c, *Mutation rate* is the predefined mutation probability P_m, *Stepsize* is the step size for four Runge–Kutta, c_1 and c_2 are positive constants used in the PSO algorithm. These parameters are chosen experientially and different parameter choices affect the convergence rate of algorithm. We used 10 input variables to construct an ordinary differential equation model. Namely we use the first 10 variables to predict the current variable. The used instruction set $I_0 = \{+2,+3,+4,+5,+6,+7,+8\}$ and $I_1 = \{*, /, log, exp, rlog, sin, cos, X_1, X_2, X_3, \ldots, X_{10}\}$. After several runs, we obtained the best model for prediction network traffic measurements data, which is shown in (8). The computational time required is about 2.5 h on a single-CPU personal computer (Pentium III 933 MHz). We gain that *Population size*, P_c, P_m, and *stepsize* closely affect the convergence rate of the algorithm and the precision of the traffic prediction. The bigger *Population size* leads to the probability for having better solutions. In our experiments, computation of the fitness evaluation takes the largest amount of time, so although the number of generations needed becomes small, the time spent with a bigger *Population size* increases significantly. For example, with a *Population size* of 100 we must spend about 2.9 h in the same computer. With a small *Population size* of 20, the time is 2.85 h. So we selected a *Population size* of 50. P_c and P_m are given generally. In order to test the effect of *stepsize*, we tried experiments with different *stepsize* of 0.0001, 0.005, 0.1, 0.5, 1 and 2. Fig. 6 illustrates the *NMSEs* of training data for different *stepsizes* of 0.005, 0.1, and 2 for every 20 generations. As evident, the error of ODE increases significantly when the *stepsize* is greater than 1, while too small *stepsize* such as 0.0005, can provide smaller errors in the beginning, but offer little improvement in the evolutionary process of ODE. With *stepsizes* of 0.005 and 2, the *NMSEs* for test data are 0.013172 and 0.014876, respectively. For a tradeoff, we set the *stepsize* as 0.1. The *NMSEs* for training data is 0.062074. *NMSEs* and *RMSE* for the test data are 0.011147 and 0.024109, respectively. The time series predicted by this system is depicted in Fig. 8 along with the target value. The difference between actual network traffic time series data and the predicted values are illustrated in Fig. 7. From Figs. 7

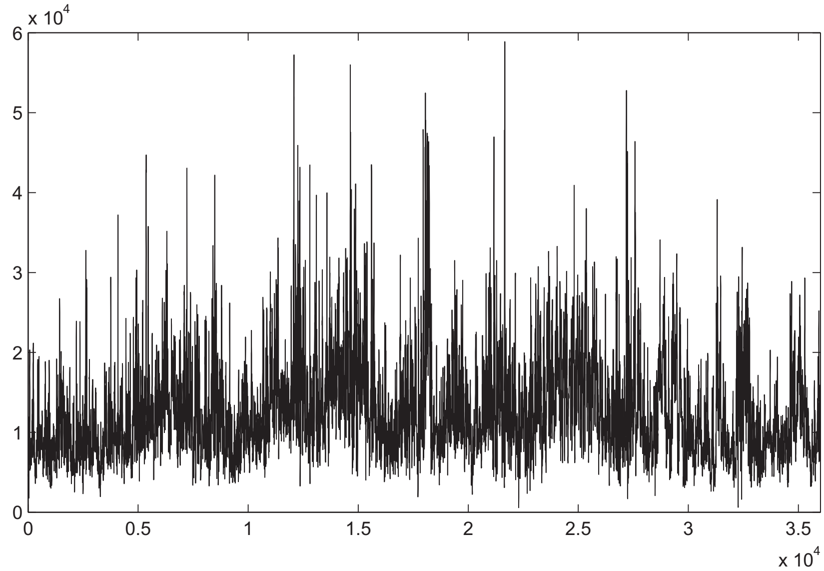

Fig. 4. Filtered traffic measurements data.

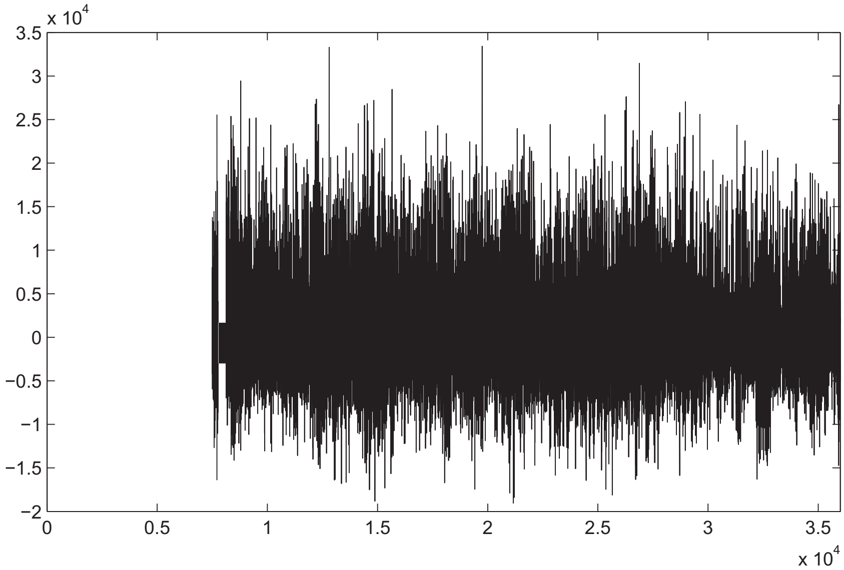

Fig. 5. Noisy component.

Table 1
Parameters for experiment.

Population size	50
Generation	200
Crossover rate	0.7
Mutation rate	0.3
Stepsize	0.1
c_1	2.0
c_2	2.0

and 8, it is evident that the system of ordinary differential equation can effectively predict the traffic data, and the error is very low.

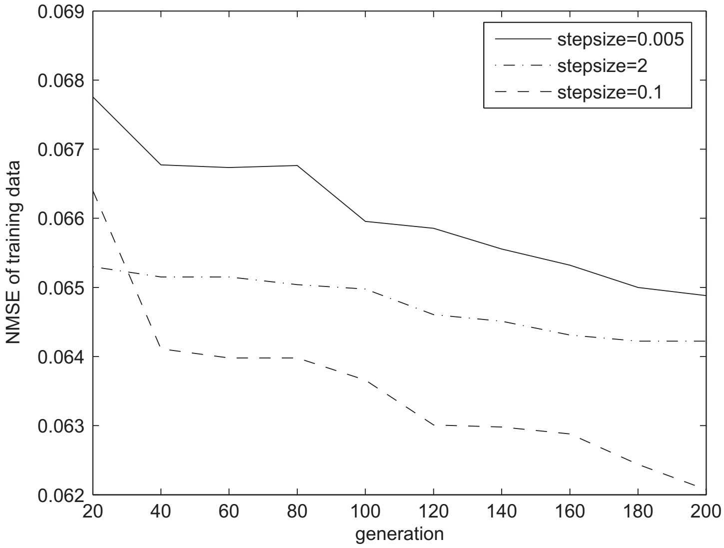

Fig. 6. *NMSE* of training data with different step sizes.

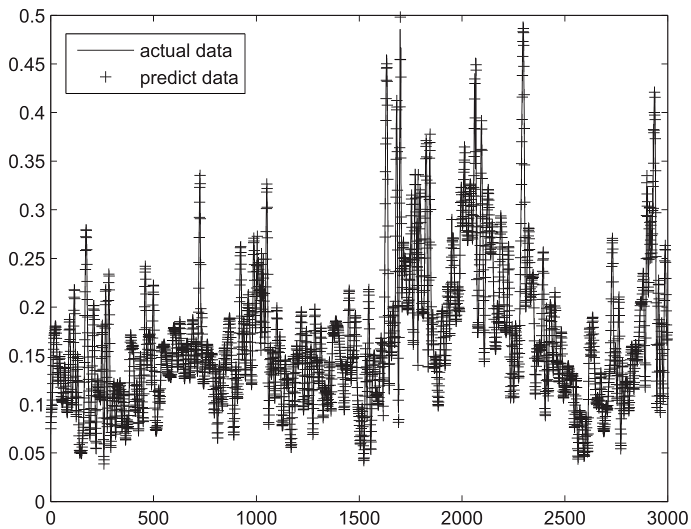

Fig. 7. Comparison of actual and predicted time series.

$$\dot{f} = -2.082257sin(X_6) + 1.010300sin(X_1) + 1.393565log(1 + X_{10}X_{10}) - \frac{1.900762}{(1 + exp(-X_{10}))} + 0.962096cos(X_2). \quad (8)$$

We also made a comparison with a feedforward neural network trained using a PSO algorithm. The *NMSEs* for training and testing data sets are 0.092304 and 0.073236, respectively[7]. As evident, the prediction performance of ODE is better than the neural network model. Fig. 9, illustrates the statistical distributions of the absolute difference between the actual time series and the predicted data by our method. As evident, the prediction error of the ODE model mainly concentrates on the vicinity of zero.

To test the validity of the additive tree-structured evolutionary algorithm, we also compared the traffic prediction using gene expression programming (GEP) and genetic programming (GP). In these experiments, GEP and GP used the same parameters with the proposed method except *Generation*, and use the same operator set $I = \{*, /, log, exp, rlog, sin, cos, X_1, X_2, X_3, \ldots, X_{10}\}$ and run conditions. After searching the best ODE the results of GEP and GP are listed in Table 2. The best model obtained using GEP and GP for prediction network traffic measurements data are given by (9) and (10) respectively.

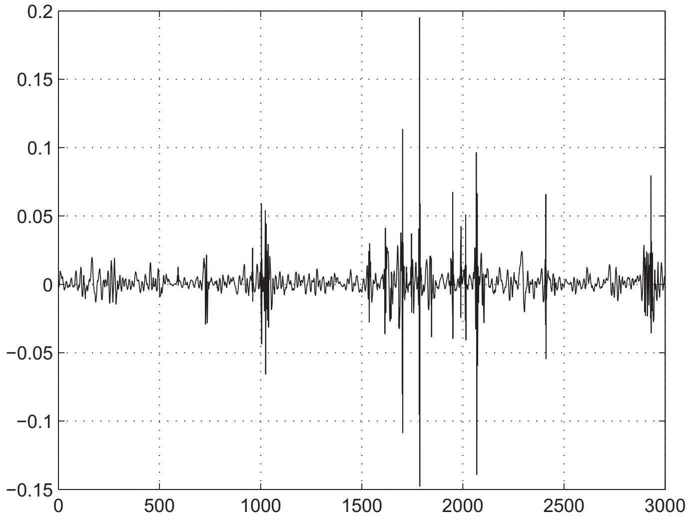

Fig. 8. Errors of actual and predicted time series.

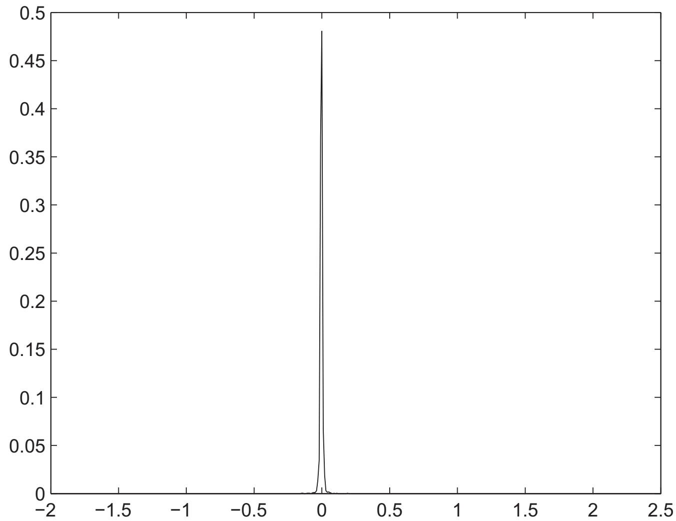

Fig. 9. The statistical distributions of the predicted errors.

Table 2
Comparison results using the proposed method, GP and GEP.

	our method	GP	GEP
runtime	2.5 h	3.7 h	3.16 h
NMSE for training data	0.062074	0.063113	0.062737
NMSE for test data	0.011147	0.013198	0.012007

$$\dot{f} = 1.395673log(1 + x_{11}x_9) + \frac{2.2024131}{1 + x_{11}x_7} - 1.896417x_5 - 2.204957cos(x_5) + 1.087188log(1 + x_1x_8), \tag{9}$$

$$\dot{f} = 2.231005cos(x_9) - \frac{1.585037}{1 + exp(-x_9)} + 1.499770sin(x_{10}) - 2.141063log(1 + x_6x_9) - \frac{1.341362}{1 + x_1x_1}. \tag{10}$$

From the empirical results, it is evident that the proposed additive tree mode is powerful than the GEP and GP models, both in the aspects of accuracy and runtime.

6. Conclusions

In this paper, a hybrid evolutionary method for evolving ODEs is proposed to predict the small-time scale traffic measurements data. Tree-structure based evolutionary algorithm and particle swarm optimization (PSO) algorithm were used to evolve the architecture and the parameters of the additive tree models for identifying a System of ordinary differential equations. The experiment results clearly illustrate that the ODE model can effectively predict the traffic measurements data. The prediction error mainly concentrates on the vicinity of zero and the accuracy of the ODE model is superior to that of genetic programming, gene expression programming and feedforward neural network optimized using PSO.

Acknowledgments

This research was partially supported by the Natural Science Foundation of China (61070130), the Natural Science Foundation of Province (Y2007G33), the Shandong Distinguished Middle-aged and Young Scientist Encourage and Reward Foundation of China (BS2009SW003), and the Key Subject Research Foundation of Shandong Province.

References

[1] F. Braghin, R. Lewis, R.S. Dwyer-Joyce, S. Bruni, A mathematical model to predict railway wheel profile evolution due to wear 261 (11-12) (2006) 1253–1264.
[2] J. Bongard, H. Lipson, Automated reverse engineering of nonlinear dynamical systems, Proceedings of the National Academy of Science 104 (24) (2007) 9943–9948.
[3] H. Cao, L. Kang, Y. Chen, J. Yu, Evolutionary modeling of systems of ordinary differential equations with genetic programming, Genetic Programming and Evolvable Machines1 (40) (2000) 309–337.
[4] B. Cuartas-Uribe, M.C. Vincent-Vela, S. lvarez-Blanco, M.I. Alcaina-Miranda, E. Soriano-Costa, Prediction of solute rejection in nanofiltration processes using different mathematical models, Desalination 200 (1–3) (2006) 144–145.
[5] Y.H. Chen, B. Yang, A. Abraham, Flexible neural trees ensemble for stock index modeling, Neurocomputing 70 (4–6) (2007) 697–703.
[6] Y.H. Chen, J. Yang, Y. Zhang, J.W. Dong, Evolving additive tree models for system identification, International Journal of Computational Cognition 3 (2) (2005) 19–26.
[7] A.D. Doulamis, N.D. Doulamis, S.D. Kollias, An adaptable neural- network model for recursive nonlinear traffic prediction and modeling of MPEG video sources, Neural Networks, IEEE Transactions on (2003) 150–166.
[8] Raquel P.F. Guin, Pear drying, Experimental validation of a mathematical prediction model, Food and Bioproducts Processing 86 (4) (2008) 248–253.
[9] I. Gasser, G. Sirito, B. Werner, Bifurcation analysis of a class of car-following traffic models, Physica D 197 (2004) 222–241.
[10] H. Iba, Inference of differential equation models by genetic programming, Information Sciences 178 (23) (2008) 4453–4468.
[11] G. Kou, Y. Peng, Zh.X. Chen, Y. Shi, Multiple criteria mathematical programming for multi-class classification and application in network intrusion detection, Information Sciences 179 (4) (2009) 371–381.
[12] A. Neumaier, Mathematical Model Building, Chapter 3, in: J. Kallrath (Ed.), Modeling Languages in Mathematical Optimization, Kluwer, Boston, 2004.
[13] G. Orosz, B. Krauskopf, R.E. Wilson, Bifurcations and multiple traffic jams in a car-following model with reaction-time delay, Physica D 211 (3–4) (2005) 277–293.
[14] M. Oltean, C. Grosan, Evolving digital circuits using multi expression programming, in: R. Zebulum et al. (Eds.), NASA/DoD Conference on Evolvable Hardware, 24–26 June, Seattle, IEEE Press, NJ, 2004, pp. 87–90.
[15] G. Orosz, R.E. Wilson, B. Krauskopf, Bifurcations in a car-following model with delay in Fifth IFAC Workshop on Time-Delay Systems, in: W. Michiels, D. Roose (Eds), IFAC, 2004.
[16] S.K. Sadrnezhaad, A. Ferdowsi, H. Payab, Mathematical model for a straight grate iron ore pellet induration process of industrial scale, Computational Materials Science 44 (2) (2008) 296–302.
[17] K. Chellapilla, Evolving computer programs without subtree crossover, IEEE Transactions on Evolutionary Computation 1 (1997) 209–216.
[18] P. Shang, X. Li, S. Kamae, Chaotic analysis of traffic time series, Chaos, Solitons and Fractals 25 (2005) 121–128.
[19] P. Shang, X. Li, S. Kamae, Nonlinear analysis of traffic time series at different temporal scales, Physics Letters A 357 (4-5) (2006) 314–318.
[20] S. Sundar, R. Naresh, A.K. Misra, J.B. Shukla, A nonlinear mathematical model to study the interactions of hot gases with cloud droplets and raindrops, Applied Mathematical Modelling 33 (7) (2009) 3015–3024.
[21] P. Shang, X. Li, S. Kamae, Chaotic analysis of traffic time series, Chaos Solitons Fractals 25 (2005) 121–128.
[22] P. Shang, X. Li, S. Kamae, Nonlinear analysis of traffic time series at different temporal scales, Physics Letter A 357 (2006) 314–318.
[23] traffic archive, <http://ita.ee.lbl.gov/>.
[24] I.G. Tsoulos, I.E. Lagaris, Solving differential equations with genetic programming, Genetic Programming and Evolvable Machines 7 (1) (2006) 1389–2576.
[25] S. Uglig, Non-stationarity and high-order scaling in TCP flow arrivals:a methodological analysis, ACM SIGCOMM computer communications 34 (2) (2004) 9–24.
[26] Y.B. Xie, W.X. Wang, B.H. Wang, Modeling the coevolution of topology and traffic on weighted technological networks, Physical Review E 75 (026111) (2007).
[27] Z.L. Zhang, V.J. Ribeiro, S. Moon, C. Diot, Small-time scaling behaviors of Internet backbone traffic: an empirical study, IEEE Computer and Communications Societies 3 (2003) 1826–1836.