Genetic Programming and Evolvable Machines (2020) 21:87–97
https://doi.org/10.1007/s10710-019-09362-4

The impact of genetic programming in education

Nelishia Pillay[1]

Received: 20 October 2018 / Revised: 23 April 2019 / Published online: 26 July 2019

Abstract
Since its inception genetic programming, and later variations such as grammar-based genetic programming and grammatical evolution, have contributed to various domains such as classification, image processing, search-based software engineering, amongst others. This paper examines the role that genetic programming has played in education. The paper firstly provides an overview of the impact that genetic programming has had in teaching and learning. The use of genetic programming in intelligent tutoring systems, predicting student performance and designing learning environments is examined. A critical analysis of genetic programming in education is provided. The paper then examines future directions of research and challenges in the application of genetic programming in education.

Keywords Genetic programming · Education · Intelligent tutoring systems · Pedagogical agents · Learning analytics

1 Introduction

Artificial intelligence has played a prominent role in education in enhancing teaching and learning. Artificial intelligence techniques have been employed in intelligent tutoring systems, for automated assessment and more recently in data analytics to identify learning problems, in tools to promote collaboration amongst students and in online teaching assistants. For example, Georgia Tech University has recently developed an automated teaching assistant, Jill Watson, that was able to answer student queries in online forums with a 97% accuracy for an online course in artificial intelligence [7]. This paper examines the role that genetic programming has played in education, providing a critical analysis of the application of genetic programming in education. Genetic programming has had an impact on various facets in education including inducing solutions to problems in intelligent tutoring systems, prediction of student performance, the design of online courses, determining student

✉ Nelishia Pillay
npillay@cs.up.ac.za

[1] Department of Computer Science, University of Pretoria, Pretoria, South Africa

perceptions and simulations of concepts in learning environments. One of the areas in which genetic programming has made a major contribution is prediction of student performance. Student performance prediction is essentially a classification problem and genetic programming has proven to be effective at evolving classifiers [4]. One of the advantages that genetic programming has over other techniques used for prediction is that it is a white box technique that produces classifiers that are interpretable, thereby providing explanations for predictions [4]. Genetic programming has also been effective at designing other approaches applied to education [4], e.g. neural networks [30]. As we enter the fourth industrial revolution, it is predicted that artificial intelligence will play an even larger role in education. Given this the paper then presents future research directions for the use of genetic programming for teaching and learning, highlighting potential challenges. Hence, the main contributions of this paper is a survey and analysis of the impact of genetic programming in education and directions for future research with the aim of setting the foundation for promoting further research in this field.

The next section looks at the use of genetic programming in intelligent tutoring systems. The use of genetic programming to predict student performance is presented in Sect. 3. Section 4 provides an overview of using genetic programming for the design of learning environments. A critical analysis of genetic programming in education is provided in Sect. 6. The section goes on to describe future directions and challenges of the application of genetic programming in education as we move into the fourth industrial revolution. A summary discussion of the paper is presented in Sect. 7.

2 Intelligent tutoring systems

Intelligent tutoring systems employ the use of artificial intelligence techniques to provide individualised tuition [16]. These systems model student knowledge and performance to determine what to teach the student at each stage of a tutorial. Intelligent tutoring system architectures are comprised of at least three modules, namely, the expert module, the student module and the tutoring or pedagogical module [16, 19]:

- The *expert module* contains knowledge of the domain to be presented to the student and/or to solve problems.
- The *student module* stores information regarding the knowledge and skills of the student and student preferences.
- The *tutoring module* contains instructional strategies for tutoring the student.

Other modules that have been included in intelligent tutoring system architectures include the *interface module* which provides a user interface via which the student uses the intelligent tutoring system and the *problems module* which stores the tasks or problems to be presented to the student. Genetic programming has essentially been used to automatically generate solutions to problems in the expert module.

Pillay [19] presents a generic architecture for intelligent programming tutors for imperative programming languages. In this architecture genetic programming has been successfully employed to generate solutions to programming problems. One of the challenges experienced in the study is premature convergence of the genetic programming algorithm as a result of fitness function biases. The study presents the iterative structure-based algorithm (ISBA) to overcome this problem. This algorithm uses similarity indices to prevent structurally similar areas from being revisited.

In [21] genetic programming is used to generate finite automata for automata problems in an intelligent tutoring system for finite automata. Solutions were generated in less than a minute. The evolved solutions were found to contain redundant states which were removed using the minimization algorithm for finite automata [13].

3 Predicting student performance

One of the areas in which genetic programming is has made a significant impact is the prediction of student performance. In the study conducted by Orove et al. [18] multi-gene genetic programming is used to predict failure rates at schools. Each element of the genetic programming population represents a weighted prediction model and is comprised of one or more genes and each gene is a parse tree. Each parse tree represents a rule and is a nested if-statement. The evolutionary process evolves the rules and the weights are determined by least squares. The evolved models were successful at predicting student failure rates at schools. One of the challenges indicated by the authors is redundant code and bloat.

A similar study was conducted in [35] to predict student performance in a web-based system using grammar-guided genetic programming. Each element of the genetic programming population is a rule which determines whether the student will pass or not. The proposed approach was able to attain a 74% accuracy in predicting student success. The authors identify an area for future research to be predicting students grades instead of only whether the student would pass.

In [14, 15] grammar-based genetic programming is compared to other classification techniques for predicting student failure in a Mexican high school. In this study they classifiers are composed of if-then-else rules that predict performance. Evolving rules allows for the reason for failures to be determined. The approach achieved a 98.7% accuracy rate in predicting failure, outperforming the other techniques.

Ulloa-Cazarez et al. [29] have employed genetic programming to predict student final grades early in an online learning course in order to provide support for potentially failing students. Data was collected from the online learning management system. The genetic programming approach was found to outperform statistical linear regression models used to perform the same prediction.

In the study conducted by Xing et al. [31] genetic programming has been used to evolve if-then-else rules to predict student performance. The student performance is categorized as excellent, good, average, sufficient and fail by the rules for four geometry modules. Genetic programming was found to outperform other techniques such as neural networks with an 82% prediction accuracy. A further advantage that

genetic programming was found to have over other the other approaches was understandability of the evolved prediction models.

4 Designing learning evironments

This section examines the use of genetic programming for the design of different aspects of learning environments including the content to be presented in web-based course, simulating electronic circuits, adapting of quizzes in an online course and the evolution of pedagogical agents.

In the study conducted by Romero and Ventura [26] grammar-based genetic programming is used to design web-based courseware. Each design specifies the lessons or chapters for a course and different knowledge levels, the number of concepts that each lesson or chapter is comprised of for the different knowledge levels and the activities that will be used to test each concept or chapter. Genetic programming is used to evolve rules that dictate the design. A grammar is used to define the syntax of the rules. A multiobjective function is used to assess fitness. The grammar-based genetic programming approach was found to evolve effective rules.

Li et al. [12] use genetic programming to simulate electronic circuits in a digital learning environment. This is achieved by employing genetic programming to minimimze the difference between theoretical excitation signals and approximation driving pulses.

In [27] genetic programming is used to to design an online course by adapting quizzes and the course to the particular learner. Grammar-based genetic programming is used to evolve association rules to determine how to adapt quizzes. The approach was successfully applied to a course on Clips programming.

One of the challenges with using online systems like intelligent tutoring systems for tutoring is motivating students. This has resulted in the development of Motivationally and Culturally Aware Systems (MOCAs). MOCAs aim to achieve this motivation by use of pedagogical agents which operate in a virtual world. Blanchard and Frasson [2] have used genetic programming to evolve pedagogical agents in a MOCA. The agents are created and dynamically adapted using genetic programming. This allows the pedagogical agent to adapt to the student's needs. Each evolved agent is comprised of different behaviours, e.g. politeness, emotional management.

5 Combining genetic programming with neural networks

This section examines studies in which genetic programming has been combined with neural networks for providing solutions in education. In the study conducted by Fei and Lu [5] genetic programming is combined with neural networks to determine the perceptions of students taking a nautical course to seafaring as a career. The neural network is employed to improve the accuracy of discretization of the data. Genetic programming is applied to the discretized data to produce rules that specify student perceptions. The rules indicate whether the students will follow a career in

seafaring or not and the reasons for this. A similar approach to this can be taken by determine perceptions of topics that students experience difficulty in learning in different courses.

Vrettaros et al. [30] have used genetic programming to design a neural network for assessment in an e-learning environment. The evolved neural network successfully assessed student responses to single select and multiple choice questions.

6 Critical analysis and future research directions

Table 1 provides a summary of the studies reviewed in the previous sections. As can be seen from the table genetic programming in education in the previous sections, there has not been much research in this field. These studies have illustrated the potential of genetic programming in education and set the foundation for future research initiatives. This section firstly provides a critical analysis of previous work and then proposes future research directions and challenges in applying genetic programming in education.

The main contribution that genetic programming has made in the area of intelligent tutoring systems is the automated induction of solutions to problems that are presented to the student. This removes the load from the intelligent tutoring system developer to create and store the solutions to all the problems or exercises that will be presented to the learner. In some domains this may be trivial however in this could be extremely time consuming as in the case of both the domains described in Sect. 2, namely, programming and finite automata.

In order to reduce the time involved in developing intelligent tutoring systems various authoring tools have been made available so that the developer does not have to create the intelligent tutoring system from scratch [3]. Genetic programming libraries need to be incorporated into such authoring tools for automated solution generation. As can been since from the review of the field, genetic programming has not been widely used for the derivation of solutions in intelligent tutoring systems and has been employed in just two intelligent tutoring systems. This can possibly be attributed to the expert knowledge needed to implement genetic programming for solution induction. The availability of genetic programming as part of authoring tools for intelligent tutoring systems will alleviate the need of such expert knowledge.

The generated solutions can be used to assess student solutions as well as to show the student the solution to the problem. In the case of the latter a challenge is the redundant code that may be contained in the solutions evolved by genetic programming which may result in the solutions not being easily readable. For example, the solutions to automata problems generated in [21] contained redundant states which had to be removed using the minimization algorithm for automata. Mechanisms for reducing bloat could be incorporated into the genetic programming approach inducing solutions and editing performed on the evolved solutions. Alternatively, grammatical evolution, a variation of genetic programming aimed at reducing introns and bloat, can be used instead.

Table 1 Summary of genetic programming applications

Area	Application	Genetic programming approach
Intelligent tutoring systems	Generating program problem solutions	Genetic programming
	Generating automata problem solutions	Genetic programming
Predicting student performance	Predicting failure at school [18]	Multi-gene genetic programming
	Predicting failure in a web-based course [35]	Grammar-guided genetic programming
	Predicting failure in school [14, 15]	Grammar-based genetic programming
	Predicting final grades in an online learning course [29]	Genetic programming
	Predicting type of pass in geometry courses [31]	Genetic programming
Designing learning environments	Determining lessons and assessments	Grammar-based genetic programming
	Simulating electronic circuits [12]	Genetic programming
	Adapting quizzes to the learner [27]	Grammar-based genetic programming
	Evolving pedagogical agents [2]	Genetic programming
Determining student perceptions	Student perceptions towards seafaring [5]	Genetic programming and neural networks
Designing approaches	Neural network for assessment [30]	Genetic programming Genetic programming

One of the modules that can be included in an intelligent tutoring system is the problem module which stores problems to be presented to the student at various stages of the tutoring. Creating repositories of problems can be a time consuming process. Genetic algorithms have successfully been used to generate multiple choice questions [34]. Similarly, in the study conducted by Yan et al. [32] a genetic algorithm is used to generate test papers. In these studies the chromosome represents a set of questions and fitness is assessed in terms of whether each question is at the correct level and there is no repetition of questions in previous papers. Genetic programming and variations such as grammar-based genetic programming and grammatical evolution need to be investigated as a means of automatically generating problems. These techniques will allow for more flexibility and instead of generating just a sequence of questions will cater for each question to be evolved. One option would be for genetic programming to evolve rules that produce the problems. The condition of the rules would be the problem specification to be met and the action the component/s to include in the problem.

As we move into the fourth industrial revolution it is anticipated that the use of artificial intelligence will play a major role in automated assessment with the aim of reducing the workload of teachers and lecturers as well as providing a more timely response to students and will be essential for online learning. For example, an artificial intelligence essay marking approach has been able to achieve a 94% accuracy when compared to human marked essays. A further area of investigation is using genetic programming for automated assessment both in intelligent tutoring systems and in standalone assessment tools. For example, in intelligent programming tutors [19] genetic programming can be used to access the accuracy of student code given the success of genetic programming in automated software repair [6].

The use of genetic programming in other modules of the intelligent tutoring system architecture should also be examined. For example, the student module is one such module where genetic programming could have an impact. As in the study conducted by Hong et al. [10] genetic programming can be used to produce an individualised curriculum path for students based on their level of knowledge at each stage of the tutoring process. Learning style is often used in online learning systems such as intelligent tutoring systems as an indicator of which instructional strategies to use when tutoring a student [19, 33]. Genetic programming can be used to induce production rules to determine the instructional strategy given the learning style. Similarly, genetic programming can be used to evolve rules to produce feedback in the pedagogical module in an intelligent tutoring system [19].

Previous research has shown that genetic programming is effective at evolving behaviours of pedagogical agents. Future research should investigate the use of genetic programming for similar online systems such as online automated teaching assistants like Jill Watson, an online teaching assistant used in an artificial intelligence course at Georgia Tech University. Genetic programming can be used for generating behaviours of and feedback provided by automated teaching assistants. In some domains discussions on topics with fellow students may be needed to facilitate learning, however it may not be possible to get together students online at the same time. Given the success of genetic programming in generating learning agents one

possibility would be to use genetic programming to generate the behaviour and communication of automated fellow students.

From Sect. 3 it is evident that genetic programming is effective at predicting student performance. Prediction has generally involved using genetic programming to induce a classifier which is used to predict the performance, e.g. pass or fail. An advantage that genetic programming has over other machine learning techniques when it comes to classification include different representations for classifiers, e.g. rules, decision trees. One of the challenges associated with employing genetic programming is computational cost [4]. This cost can be alleviated with the use of distributed architectures, such as multicore architectures, in the implementation of the genetic programming algorithm. Runtimes can also be reduced by using incremental learning, performing training on subsets of data which are incrementally increased in size [4]. A further challenge which will inhibit the interpretability of the evolved classifier is redundant code, i.e. introns which the evolved classifier may contain. Espejo et al. [4] indicate that use of nondestructive genetic operators and parsimony pressure as means to overcome this. In addition to this, like other machine learning techniques, genetic programming contains various parameters that must be tuned [4]. The initiative by the machine learning community to automate the design of machine learning techniques, namely, autoML, will help overcome this challenge. For example, in the study in [17] the design of the genetic programming algorithm, including determining of parameters, was successfully automated to produce classifiers which performed better than the manually designed classifiers. Other advantages of genetic programming include interpretability of the classifiers produced s well as automatic feature selection/reduction [4]. The latter is important in prediction using educational data as the data usually contains a large number of data. Furthermore, when applying genetic programming to educational data it should be kept in mind that it is highly likely that the data is imbalanced, e.g. more students pass rather than fail, and this must be catered for. This evolution of rules for prediction can be extended to use genetic programming to predict potential learning difficulties. Learning analytics [28] is playing an import role in the mining of educational data to improve teaching and learning and as we move into the fourth industrial revolution this data is becoming big data. Given the success of genetic programming in the prediction of student performance it can play an important role in the mining of data in learning analytics.

The field of educational data mining is a growing field [25] and will play a prominent role in the fourth industrial revolution. The effectiveness of genetic programming for predicting student performance in educational data mining has been established. The successful application of genetic programming for designing courseware has also been illustrated. The use of genetic programming in other areas of educational data mining outlined by Romero et al. [25], namely, the provision of feedback that can be used by instructors to make decisions about how to progress with the learning process, make recommendations to students such as which topic to study next, student modelling in intelligent tutoring systems, identifying undesirable student behaviour such as cheating or low motivation, grouping students for group learning, analyzing social networks to support the learning process, e.g. identifying suitable learning partners, creating concepts maps to get an idea of a

learners' understanding of a topic, needs to be investigated. While the rules evolved by genetic programming for educational data mining does provide some explanation as to reasons for the conclusion/action arrived at, these explanations may not be clear, may not be that readable due to introns or more detail may be required. The combination of explainable artificial [1, 11] and genetic programming needs to be examined to enhance explanations and feedback.

The allocation of resources, especially manpower, is a frequent problem in education [8, 9]. Genetic algorithms have been successfully employed to induce school timetables and university examination timetables [20, 23]. More recently the potential of genetic programming for educational timetabling has been illustrated in [22]. In this study genetic programming has been used to induce heuristics for constructing timetables for university course and examination timetabling. The evolved heuristics were found to outperform human derived heuristics. This study has illustrated the potential of genetic programming and further research into the use of genetic programming for timetabling and resource allocation in education is needed.

An area that needs further investigation is applying genetic programming to promote the various learning theories in learning environments. Reid [24] describes three main learning theories, namely, behaviourism, cognitivism and constructivism. Behaviourism promotes acquiring knowledge though new associations via stimuli and responses resulting in a change in behaviours. In cognitivism on the other hand knowledge is generated by processing information. The learner understands by correlating new knowledge with that in memory. Constructivism promotes the learner creating new knowledge by generating new ideas. The learner constructs knowledge by analyzing his/her own perspective of the world/situation based on previous experiences.

7 Conclusion

This paper provides a review of the role that genetic programming has played in education. Genetic programming has proven to be effective in various facets of education including intelligent tutoring systems, automated pedagogical agents, the design of online courseware and the prediction of student performance. As we move into the fourth industrial revolution it is anticipated that artificial intelligence will play an imperative role in online learning and improving teaching and learning. The paper highlights the contribution that genetic programming can make including playing a larger role in intelligent tutoring systems in terms of automated problem and solution generation, automated assessment and modelling of student knowledge to determine topics and feedback to be presented to students based on their learning needs. Other areas where genetic programming would be effective is the induction of behaviours and feedback of teaching assistants and automated peer students and timetabling and resource allocation in education.

Acknowledgements The author would like to thank the reviewers for their helpful comments and suggestions to improve the quality of the paper.

References

1. A. Adadi, M. Berrada, Peeking inside the black-box: a survey on explainable artificial intelligence (xai). IEEE Access **6**, 52138–52160 (2018)
2. E. Blanchard, C. Frasson, Motivation and evolutionary pedagogical agents, in *Workshop on Motivational and Affrective Issues in ITS, Intelligent Tutoring Systems (ITS 2006)* (2006)
3. D. Dermeval, R. Paiva, I.I. Bittencourt, J. Vassileva, D. Borges, Authoring tools for designing intelligent tutoring systems: a systematic review of the literature. Int. J. Artif. Intell. Educ. **28**, 336–384 (2018)
4. P.G. Espejo, S. Ventura, F. Herrera, A survey on the application of genetic programming to classification. IEEE Trans. Syst. Man Cybern. C Appl. Rev. **40**(2), 121–144 (2010)
5. J. Fei, J. Lu, Analysis of students's perceptions of seafaring career in chinda based on artificial neural network and genetic programming. Marit. Policy Manag. **42**(2), 111–126 (2015)
6. S. Forrest, T.V. Nguyen, W. Weimer, C.L. Goues, A genetic programming approach to automated software repair, in *Proceedings of the 18th International Conference on Genetic Algorithms and the 14th Annual Genetic Programming Conference* (2009), pp. 947–954
7. A.K. Goel, L. Polepeddi, Jill watson: a virtual teaching assistant for online education. Technical Report (Georgia Institute of Technology, 2016)
8. A. Gunawan, K. Ng, A genetic algorithm for the teacher assignment problem for a university in indonesia. Inf. Manag. Sci. **18**(1), 1–16 (2008)
9. W. He, P. Liang, G. Zou, Optimal allocation of higher education resources based on the pareto genetic algorithm. World Trans. Eng. Technol. Educ. **14**(1), 190–197 (2016)
10. C.M. Hong, C.M. Chen, M.H. Chang, Personalized learning path generation approach for web-based learning, in *Proceedings of the 4th WSEAs International Conference on E-Activities* (2005), pp. 62–68
11. D. Howard, M.A. Edwards, Explainable a.i.l the promise of genetic programming multi-run subtree encapsulation, in *Proceedings of the 2018 Conference on Machine Learning and Data Engineering (iCMLDE)* (2018), pp. 158–159
12. Y. Li, C. Yuan, C. Zhang, S. Li, K. Sun, X. Wang, A novel approximation algorithm based on genetic programming in the digital learning environment, in *Proceedings of the 2015 International Conference of Educational Innovation Through Technology* (2015), pp. 33–36
13. P. Linz, *An Introduction to Formal Languages and Automata* (Jones and Bartlett Publishers, Burlington, 2006)
14. C. Marquez-Vera, A. Cano, C. Romero, A.Y.M. Noaman, H.M. Fardoun, S. Ventura, Early dropout predication using data mining: a case study with high school students. Expert Syst. **33**(1), 107–124 (2016)
15. C. Marquez-Vera, A. Cano, C. Romero, S. Ventura, Predicting student failure at school using genetic programming and different data mining approach with high dimnesional and imbalanced data. Appl. Intell. **38**, 315–330 (2013)
16. H.S. Nwana, Intelligent tutoring systems: an overview. Artif. Intell. Rev. **4**(4), 251–277 (1990)
17. T. Nyathi, N. Pillay, Comparison of a genetic algorithm to grammatical evolution for automated design of genetic programming classification algorithms. Expert Syst. Appl. **104**, 213–334 (2018)
18. J. Orove, N. Osegi, B. Eke, A multi-gene genetic programming application for predicting students failure at school. Afr. J. Comput. ICT **7**(3), 21–34 (2014)
19. N. Pillay, An invesitgation into the use of genetic programming for the induction of novice procedural programming solution algorithms in intelligent programming tutors. Ph.D. Thesis (School of Geological and Computer Sciences, University of KwaZulu-Natal, 2004)
20. N. Pillay, A survey of school timetabling research. Ann. Oper. Res. **218**(1), 261–293 (2014)
21. N. Pillay, A. Naidoo, An investigation into the automatic generation of solutions to problems in an intelligent tutoring system for finite automata, in *Proceedings of the 36th Annual Conference of the Southern African Computer Lecturers Association* (2006), pp. 11–20
22. N. Pillay, E. Ozcan, Automated generation of constructive ordering heuristics for educational timetabling. Ann. Oper. Res. **275**, 1–28 (2017)
23. R. Qu, E. Burke, B. McCollum, L. Merlot, S. Lee, A survey of search methodologies and automated system development for examination timetabling. J. Sched. **12**(1), 55–89 (2008)
24. A.J. Reid, What is learning theory? (2019). https://www.instructionaldesigncentral.com/learning-theory. Accessed 20 Oct 2018

25. C. Romero, S. Ventura, Educational data minining: Review of the state of the art. IEEE Transactions on Systems, Man and Cybernetics-Part C: Applications and Reviews **40**(6), 601–618 (2010)
26. C. Romero, S. Ventura, P.D. Bra, Knowledge discovery wtih genetic programming for providing feedback to courseware authors. User Model. User Adapt. Interact. **14**(5), 425–464 (2004)
27. C. Romero, A. Zafra, J.M. Luna, S. Ventura, Associatoin rule mining using genetic programming to provide feedback to instructors from multiple-choice quiz data. Expert Syst. **30**(2), 162–172 (2013)
28. G. Siemens, R. Baker, Learning analytics and educational data mining: Towards communication and collaboration, in *Proceedings of the 2nd International Conference on Learning Analytics and Knowledge* (2012), pp. 252–254
29. R.L. Ulloa-Cazarez, C. Lopez-Martin, A. Abran, C. Yanez-Marquez, Prediction of online students performance by means of genetic programming. Appl. Artif. Intell. **32**, 858–881 (2018)
30. J. Vrettaros, J. Pavlopoulos, A.S. Drigas, K. Hrissagis, GPNN techniques in learning assessment systems. Int. J. Technol. Enhanc. Learn. (2011). https://doi.org/10.1504/IJTEL.2011.041284
31. W. Xing, R. Guo, S.P. Goggins, Participation-based student final performance prediction model through interpretable genetic programming: integrating learning analytics, educational data mining and theory. Comput. Hum. Behav. (2015). https://doi.org/10.1016/j.chb.2014.09.034
32. L. Yan, L. Shuhong, L. Xiurong, Test paper generating method based on genetic algorithm. AASRI Proc. **1**, 549–553 (2012)
33. V. Yannibelli, D. Godoy, A. Amandi, A genetic algorithm approach to recoginise students' learning styles. Interact. Learn. Environ. **14**(1), 55–78 (2006)
34. M. Yildirim, A genetic algorithm for generating test from a question bank. Comput. Appl. Eng. Educ. **18**(2), 298–305 (2009)
35. A. Zafra, S. Ventura, Multi-instance genetic programming for predicting student performance in we based educational environments. Appl. Soft Comput. **12**, 2693–2706 (2012)