Parameter Optimization Algorithms for Evolving Rule Models Applied to Freshwater Ecosystems

Hongqing Cao, Friedrich Recknagel, and Philip T. Orr

***Abstract*—Predictive rule models for early warning of cyanobacterial blooms in freshwater ecosystems were developed using a hybrid evolutionary algorithm (HEA). The HEA has been designed to evolve IF-THEN-ELSE model structures using genetic programming and to optimize the stochastical constants contained in the model using population-based algorithms. This paper intensively investigates the performances of the following six alternative population-based algorithms for parameter optimization (PO) of rule models within this hybrid methodology: 1) hill climbing (HC); 2) simulated annealing (SA); 3) genetic algorithm (GA); 4) differential evolution (DE); 5) covariance matrix adaptation evolution strategy (CMA-ES); and 6) estimation of distribution algorithm (EDA). The comparative study was carried out by predictive modeling of chlorophyll-*a* concentrations and the potentially toxic cyanobacterium *Cylindrospermopsis raciborskii* cell concentrations based on water quality time-series data in Lake Wivenhoe, Queensland, Australia, from 1998 to 2009. The experimental results demonstrate that with these PO methods, the rule models discovered by the HEA proved to be both predictive and explanatory whose IF condition indicates threshold values for some crucial water quality parameters. When comparing different PO algorithms, HC always performed best followed by DE, GA, and EDA, while CMA-ES performed worst and the performance of SA varied with different data sets.**

***Index Terms*—Cyanobacterial blooms, evolutionary algorithm, genetic programming, population-based algorithms.**

I. Introduction

RECURRENT cyanobacterial blooms have become a common feature of Australian rivers and lakes [1], [2] and pose significant risks to human health through water consumption [3], [4]. Harmful algal blooms have a high economic cost worldwide and a report in 1999 [5] estimated the annual cost to Australia to be between AUD$180 and AUD$240 million. Therefore, there is an immediate need to develop suitable tools for predictive modeling to facilitate early warning and informed management of cyanobacterial blooms.

Traditionally, process-based models that allow simulations of food web dynamics and nutrient cycles over time by using ordinary differential equations (ODEs) [6], [7] have been widely used. However, these ODEs are typically calibrated, and are usually only applicable for one specific sampling site.

With the rapid development of advanced computing technology, data mining (DM) techniques [8] have been explored to complement or even replace process-based models. With the aim of prediction, extensive study of machine learning techniques has been conducted in ecological modeling, which typically include artificial neural network (ANN) models [9], [10], and fuzzy logic and neuro-fuzzy models [11], [12]. A major drawback of ANN models is the lack of transparency by not explicitly representing the underlying models. In contrast, fuzzy models attempt to overcome this by introducing fuzzy rules, but the acquisition and reliability of these rules largely depends on the experts' empirical knowledge.

In recent years, the use of evolutionary algorithms (EAs) has gained wide popularity in domains, such as machine learning, pattern recognition, economic prediction, optimization control, and parallel processing [13], due to their capability of self-adaptation, self-organization, self-learning, intrinsic parallelism, and generality. The first application of EAs for modeling cyanobacterial blooms using simple equations was reported in [14]. Recently, we proposed a hybrid evolutionary algorithm (HEA) [15] to model algal blooms using single IF-THEN-ELSE rules that provide high transparency of the ecological driving forces and relationships driving cyanobacterial growth. Even though the IF-THEN-ELSE rule structures are similar to rule-based classifiers [16], they are more flexible in terms of representing input relationships by complex nonlinear functions in both the condition branch and the result branch. Moreover, the IF-THEN-ELSE rules calculate the output by a mathematical expression directly from the selected result branch rather than by execution of a set of rules in order.

Since we found that the IF condition randomly generated by genetic programming (GP) indicates ecological thresholds of crucial water quality parameters, we carried out parameter optimization for each rule. In addition, the optimized random constants in the THEN and ELSE branches help improve the predictive accuracy and explanitary power of the model.

The previous HEA version used only a genetic algorithm (GA) based on multiparent crossover [17] to optimize the parameters of rule models, but did not compare with other optimization algorithms. In this paper, we systematically studied six population-based algorithms for parameter optimization

Manuscript received August 13, 2012; revised March 5, 2013, May 13, 2103, and September 3, 2013; accepted September 24, 2013. Date of publication October 18, 2013; date of current version November 26, 2014.

H. Cao and F. Recknagel are with the School of Earth and Environmental Sciences, University of Adelaide, Adelaide 5005, Australia (e-mail: hongqing.cao@adelaide.edu.au; friedrich.recknagel@adelaide.edu.au).

P. T. Orr is with Seqwater, City East Qld 4002, Australia (e-mail: philip.orr@monash.edu).

Digital Object Identifier 10.1109/TEVC.2013.2286404

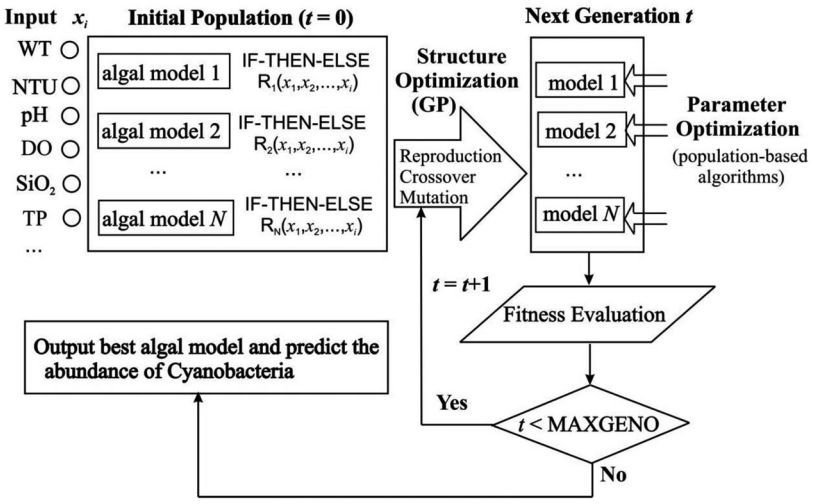

Fig. 1. Conceptual diagram of HEA for evolving the IF-THEN-ELSE rule models.

within the hybrid methodology. They are: 1) hill climbing (HC); 2) simulated annealing (SA); 3) GA; 4) differential evolution (DE); 5) covariance matrix adaptation evolution strategy (CMA-ES); and 6) estimation of distribution algorithm (EDA). The effectiveness of each algorithm was tested by the validity of the prediction of chlorophyll-*a* (Chl-*a*) concentrations and cell concentrations of the potentially toxic cyanobacterium *Cylindrospermopsis raciborskii* (Wolonszynska) Seenaya and Subba Raju (*C. raciborskii*) for 11 years of data monitored in Lake Wivenhoe, Queensland, Australia. In this paper, we provide a systematic analysis and comparison of different parameter optimization algorithms, as well as the resulting predictive rule models.

The remainder of this paper is structured as follows. In the next section, we briefly describe the hybrid evolutionary algorithm to develop rule models for predicting the abundance of cyanobacteria that includes the structure optimization using genetic programming and the six population-based algorithms applied to optimize the random constants in the rule. In Section III, we present the experimental results using the six algorithms on two data sets and statistically compare their performances. In Section IV, further experiments are described based on the best models achieved in Section III. Finally, we draw some conclusions and identify future work in Section V.

II. Hybrid Evolutionary Algorithm for Evolving If-Then-Else Rules

A HEA was proposed to evolve the rule model for predicting the abundance of cyanobacteria. The HEA contains two layers. The first or outer layer searches for model structures using GP, while the second or inner layer searches for the optimal continuous parameters of the model using some population-based optimization algorithms. Fig. 1 shows a conceptual diagram of HEA based on water-quality data as input and cyanobacterial abundance data as output. The details of the GP and the population-based algorithms are briefly described in what follows.

```
IF (((WT>22.5) AND (pH<10.8)) OR (TP*DO>=523.5))
THEN Chl-a = DO*exp(pH)+NTU*SiO2*34.5
ELSE  Chl-a = WT/2.1+ln(TP*pH-25.6)
```

Fig. 2. Example of a rule model for predicting Chl-*a* concentrations.

A. Structure Optimization of Rule Models

1) *Model Representation:* We used GP [18], [19] as the main technique for evolving the rule model structure. Because GP typically operates on parse trees instead of bit strings as traditional GA does, it is well suited to evolving an equation or formula relating the output and input variables. First, we defined the following three function sets.

1) Logic function set: $F_{\mathrm{L}} = \{AND, OR\}$;
2) Comparison function set: $F_{\mathrm{C}} = \{>, <, \geq, \leq\}$;
3) Arithmetic function set: $F_{\mathrm{A}} = \{+, -, *, /, exp, ln\}$.

In our case, a rule model with an IF-THEN-ELSE structure is represented as a vector of multiple trees in GP with the form (Tree1, Tree2, Tree3). Tree1 denotes the IF condition branch, and Tree2 and Tree3 denote the result branches of the THEN and ELSE branch, respectively. Their function sets are

$$\mathrm{F_{Tree1}} = \mathrm{F_L} \cup \mathrm{F_C} \cup \mathrm{F_A} \text{ and } \mathrm{F_{Tree2/Tree3}} = \mathrm{F_A}.$$

They have the same terminal set as

$$T = \{x_1, \ldots, x_n, c\}$$

where n is the number of input variables and c is a random constant.

Fig. 2 shows an example of a rule model for predicting Chl-*a* concentrations. The rule model shows the relationship between Chl-*a* and water temperature (WT), pH, total phosphorus (TP), dissolved oxygen (DO), turbidity (NTU), and SiO_2. Note that we provided this example for the purpose of showing the rule structure only, which may not have any biological meaning.

2) *Genetic Operators:* We used the standard GP crossover and mutation operators [18] to recombine the rules in the

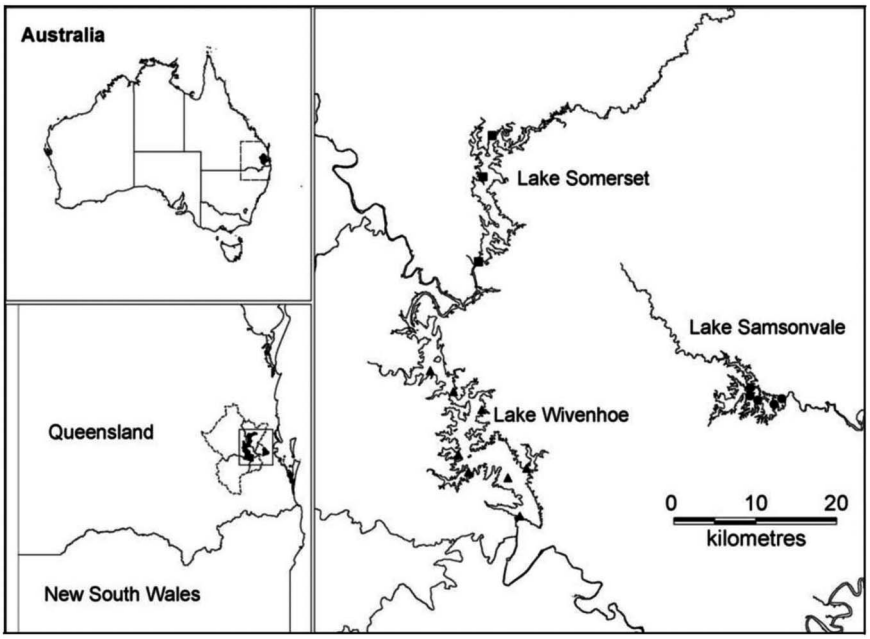

Fig. 3. Locations of three reservoirs in southeastern Queensland, Australia, showing the sampling sites on each reservoir (from [50]).

population to produce new models. The crossover is always performed between the same type of trees (Tree1/Tree2/Tree3). Note that in Tree1 there are three different types of function nodes that come from F_L, F_C, F_A, respectively. Only the same types of nodes are selected as the crossover points that ensure the crossover always produces valid condition branches for a rule.

3) *Fitness Evaluation:* The entire data set was divided into two parts: training data and testing data. As the 11 years recorded water quality data from Lake Wivenhoe reveal different annual patterns, it is difficult to choose which years should be used for training and which years should be left out for testing. To improve the robustness and stability of the model, we used a bootstrap method [20] to choose the training and testing data points randomly. That is, we defined a percentage first (75% in this paper) and in each run that percentage of the whole data set was randomly selected as the training data set and the remainder used as the testing data set. We realized that as bootstrapping requires independent data samples, when applied to time-series data, it inevitably overestimates the model accuracy to some extent, and the model performance is likely to be biased due to this sampling.

The goodness of a model in the population is evaluated by the root mean squared error (RMSE) between the measured training data and the predicted data. The fitness function is defined as

$$\text{Fitness} = \sqrt{\frac{1}{k}\sum_{i=1}^{k}(\hat{y}_i - y_i)^2}$$

where k is the number of training data points, and y_i and $\hat{y}_i$ are the measured value and the predicted value of the ith data point, respectively. Obviously, here, the lower the fitness value is, the better the model. In addition, as the training data set and testing data set can vary in each run, to make a fair comparison, we validated the best-evolved rule in each run on all the data (both training and testing data) and calculated its total RMSE as the comparable value among different runs and different algorithms.

B. Parameter Optimization of Rule Models

1) *Six Parameter Optimization Algorithms:* One of the problems inherent in ecosystem modeling is that when a model is built to describe an ecosystem, there are always parameters in the model whose values have large uncertainty and can affect the model accuracy significantly. Traditionally, a wide range of mathematical optimization techniques have been applied to ecosystem models to adjust model parameters by minimizing the misfit between the model output and a set of *in situ* observations. To date, those include among others, least-squares methods [21], [22], modified simplex algorithm [23], conjugate gradient methods [24], [25], and adjoint methods [26], [27]. A drawback common to each of the above techniques is that they are only able to detect a local minimum, and the minimum they find may depend on the parameter settings of the initial values of the model. It is clearly more desirable to have techniques that are able to identify the global minimum. The global stochastic techniques supposedly capable of doing this have been developed and applied to ecosystem models. They include SA [28], [29] and GA [30]. More recently, some new population-based methods have been developed, such as DE, CMA-ES, and EDA. They are considered to be the most competitive representatives of evolutionary computation but so far have not been applied to ecosystem models. In this paper, we selected several state-of-the-art population-based global optimization techniques to optimize the random constants contained in a rule model and compared their performance and efficiency. Those methods

include: 1) HC; 2) SA; 3) GA; 4) DE; 5) CMA-ES; and 6) EDA.

To ensure a fair comparison, for each algorithm we defined the stopping (termination) criterion as being when the total number of fitness evaluation *evalnum* reaches a maximum number N_{eval} (=1000). Below we provide brief descriptions of each method as well as the settings of some key control parameters for each algorithm. The parameter values were set based on previous experience and in consideration of available computational time. Readers are strongly recommended to refer to relevant literature for a better understanding of each method.

a) *HC:* HC was originally a local search technique that starts with a random (potentially poor) solution, and iteratively makes small changes to the solution, each time improving it a little. When the algorithm cannot detect any improvement, it terminates. In this paper, we modified the simple HC method by enabling it to do hill-climbing iteratively, each time with an initial starting point from the best solution in the previous run. The pseudocode of HC is presented in Appendix A. A key control parameter for the algorithm is the neighborhood size $N_{\text{neighbors}}$, which was set as 50 for this paper.

b) *SA:* SA [31] is a random-search technique that exploits an analogy between the way in which a metal cools and freezes into a minimum energy crystalline structure (the annealing process) and the search for a minimum in a more general system. It forms the basis of an optimization technique for combinatorial and other problems. The pseudocode of SA is presented in Appendix B. The three key control parameters in the algorithm were set as follows: initial temperature $T_0 = 100$, the epoch length $L = 100$ (the number of trials allowed at each temperature level), and the maximum Markov chains length $K = 10\,000$ (the total number of trials allowed in the whole SA process).

c) *GA:* A GA [32] is a search heuristic that mimics the process of biological evolution and the mechanisms of natural selection and genetic variation. Suitable codings are used to represent possible solutions to a problem, and the search is guided by using some genetic operators (crossover, mutation, etc.) and the principle of survival of the fittest. Due to the merits of self-adaptation, self-organization, self-learning, intrinsic parallelism and generality, GAs have been applied successfully in a wide range of economic, engineering, and scientific disciplines [33]. The GA we used in this paper is based on multiparent crossover [17]. The pseudocode of GA and the details of multiparent crossover are presented in Appendix C. The three control parameters (M, a, b) in the crossover enable the offspring to skip out of the range of parents. In this paper, they were set as $M = 8$, $a = -0.5$, $b = 1.5$, and the population size *Popsize* was set as 50.

d) *DE:* DE is an EA [34] proposed in [35] for solving real-parameter optimization problems. It is an effective, robust, and simple global optimization algorithm that extracts the differential information (i.e., distance and direction information) from the current population of solutions to guide its further search. In this way, no separate probability distribution has to be used which makes the scheme completely self-organizing. According to frequently reported comprehensive studies [36], [37], DE outperforms many other optimization methods in terms of convergence speed and robustness over common benchmark functions and real-world problems. In this paper, we used the DE algorithm and five alternative DE schemes from [38]. The pseudocode of DE and the details of five alternative DE schemes are presented in Appendix D. The parameter settings in the algorithm are $Popsize = 50$, $\lambda = F = 0.5$.

e) *CMA-ES:* CMA-ES is one of the most powerful evolutionary algorithms for real-valued optimization [39], [40] with many successful applications. For an overview see [41]. The main advantage of CMA-ES lies in its invariance properties, which are achieved by carefully designed variation and selection operators, and in its efficient (self-) adaptation of the mutation distribution. The CMA-ES consists of two adaptation phases: the cumulative step-size adaptation (CSA), which is based on the path length control, as well as the actual CMA, which is based on the evolution path. See [42] for an analysis of the two components. The website http://www.lri.fr/∼hansen/cmaes_inmatlab.html provides implementations of the CMA-ES and links. The reader can download the source code with various programming languages (ANSI C, C++, Fortran, Java, MATLAB, etc). The implementation of the CMA-ES algorithm used in this paper was based on the pseudocode shown in [40].

f) *EDA:* EDA [43], [44] is a relatively new branch of EAs. Unlike other EAs, EDAs do not use crossover or mutation. Instead, they explicitly extract global statistical information from the selected solutions and build a probability model of promising solutions based on the extracted information. New solutions are sampled from the model and fully or in part replace solutions in the current population.

Different continuous EDAs have been proposed for the global continuous optimization problem, including univariate marginal distribution algorithm (UMDA_c) [45], population based incremental learning (PBIL_c) [46], mutual information maximization for input clustering (MIMIC_c) [47], and iterated density evolutionary algorithm (IDEA) [48]. In this paper, we used an improved IDEA called iterated density evolutionary algorithm with adapted maximum-likelihood Gaussian models (AMaLGaM-IDEA) [49] that uses the combination of standard-deviation ratio, adaptive variance scaling, and anticipated mean shift to adaptively change both the covariance and the mean-shift to prevent inefficient sampling. In addition, truncation selection, rejection sampling, and elitist replacement were used in the algorithm. The implementation of the EDA algorithm in this paper was based on the pseudocode shown in [49]. Among these, some parameter settings are $Popsize = 50$ and truncation_percentage = 30%.

2) *Parameter Set Encoding and Fitness Evaluation:* The encoding of the parameter set for each of the six parameter optimization algorithms above is the same. Regarding a specific rule model, we first checked all the constants contained in Tree1, Tree2, and Tree3, including counting the number of constants l and recording their positions. Each individual in the parameter population can then be represented as an l-dimensional row vector $(c_1, c_2, \ldots, c_l)$ where each

component c_i for $i = 1, 2, \ldots, l$ is encoded as a floating number and generated randomly ranging from 0 to a predefined maximum integer (500 in this paper) during the initialization of the parameter population.

We chose positive initial values based on the following considerations. First, all the input variables in the test data sets (WT, pH, TP, etc.) allow for positive values only, so if the parameter to optimize appears in Tree1 and indicates a threshold for some specific variable, it is reasonable to initialize it with a positive value. Second, in most cases, the parameter may appear in any position of Tree1, Tree2, or Tree3. This does not matter because we have defined subtraction in the arithmetic function set. If the parameter has a negative value, it is equivalent to subtracting the absolute value of the parameter. Hence, there is no effect caused by defining a positive data range when initializing each parameter as negative values of the parameter can be achieved by evolving a model structure with subtraction operations. Moreover, because most random constants in the rule do not represent any biological meanings, for simplicity, we defined an identical maximum for each parameter. Note that this maximum only applies when initializing the parameter population and when the population is evolved generation by generation, the parameter is no longer restricted by this maximum. Its value will change adaptively by using genetic operators in the parameter optimization (PO) algorithm. In addition, if the parameter indicates a threshold, it will ultimately reach a valid value for the specific water quality parameter. Later on, experiments will verify this.

Before the fitness evaluation of an individual in the parameter population, we first returned to the original rule model and replaced all constants with the corresponding components of the row vector (i.e., the individual) and then followed the same procedure as in Section II-A to calculate the fitness.

III. Experiments

A. Study Sites and Data

The three drinking water reservoirs—Lake Samsonvale, Lake Somerset, and Lake Wivenhoe—shown in Fig. 3 are the primary raw water sources of drinking water for about 1.5 million people living in the City of Brisbane and the surrounding areas in southeastern Queensland, Australia. In recent years, toxic cyanobacteria have been regularly recorded in each of the reservoirs. In this paper, we used the Chl-*a* concentration and *C. raciborskii* abundance data from Lake Wivenhoe to test and compare the performance of different parameter optimization algorithms based on the hybrid evolutionary modeling framework. A number of limnological variables have been collected over an 11 year period (1998–2009), as shown in Table I. As the sampling intervals of the raw data are highly irregular, and sampling dates for physical, chemical, and biological variables do not always coincide, the data need to be interpolated to meet daily time steps for forecasting required by the models. To achieve this, we have evaluated a number of interpolation methods, such as random, Lagrange, Parabola, cubic spline, Aitken, and linear interpolation but we found the difference between these interpolation methods did not affect the modeling results. Therefore for simplicity, we used linear interpolation to fill missing values to produce a complete daily time series for this period. In addition, to develop 7-day-ahead predictive models for cyanobacterial abundance using our modeling algorithm, we time-shifted the interpolated daily data by 7 days to use as input data.

TABLE I
Parameters Used as Input and Output Variables for Evolutionary Modeling With the Hea Methodology

Division	Categories	Variables	Unit	Mean±SD
Input Variables	Physical	Electrical Conductivity (EC)	µS/cm	354.90±72.64
		Turbidity (NTU)	Nephelometric Turbidity Unit (NTU)	12.92±56.76
		Water Temperature (WT)	°C	22.66±3.70
	Chemical	Dissolved Oxygen (DO)	mg/L	8.67±1.25
		pH (pH)	pH Unit	8.21±0.32
		Reactive SiO_2 (SiO_2)	mg/L	3.24±2.69
		Total Nitrogen (TN)	mg/L	0.48±0.078
		Total Phosphorus (TP)	mg/L	0.019±0.012
Output Variables	Biological	Chlorophyll-*a* (Chl-*a*)	µg/L	7.82±4.54
		Cylindrospermopsis raciborskii (*C. raciborskii*)	cells/mL	10108±20698

B. Parameter Settings and Measures

For both data sets (Chl-*a* and *C. raciborskii*) 100 runs were conducted for each algorithm independently. All the experiments were performed on a Corvus supercomputer (SGI, Altix XE1300) with 69 nodes, each with two Intel quad-core processors provided by *eResearch* SA. All algorithms were programmed in C++. The GP parameter settings of HEA for structure optimization are: *Popsize* = 100, maximum tree depth = 4, and maximum number of generations (MAXGENO) = 80.

C. Results and Discussion

Table II shows the statistical results of six PO algorithms for two test data sets after 100 runs of each algorithm. For Chl-*a* data, based on the mean total RMSE of 100 runs, we ranked the average performance of the six algorithms from best to worst. The rank order was HC, DE, GA, EDA, SA, and CMA-ES. When applied to *C. raciborskii* abundance data, SA was better than DE and the rank order was HC, SA, DE, GA, EDA, and CMA-ES.

Surprisingly, both data sets demonstrated that HC was the most time-consuming in terms of computational efforts although intuitively, HC should be faster than other more complex algorithms. The high execution cost of HC implies that it is generating more complex individuals. We rechecked each model obtained by the six PO algorithms over 100 runs and obtained the average numbers of total tree nodes and optimized parameters that are shown in Table II. For both data sets, the rule models obtained by HC were most complicated with on average about 29 nodes and 7 parameters for Chl-*a* data and 37 nodes and 9 parameters for *C. raciborskii* data.

TABLE II
STATISTICAL COMPARISON OF THE SIX PO ALGORITHMS FOR THE TWO TEST DATA SETS AFTER 100 RUNS OF EACH ALGORITHM

Data Sets	No.	PO Algo.	Total RMSE (Mean±SD)	Run-Time (h)	No. of Total Tree Nodes	No. of Parameters to Optimize
Chl-*a* Data	A1	HC	3.75±0.25	1.75	28.7	6.5
	A2	SA	4.33±0.44	0.92	16.6	3.3
	A3	GA	4.04±0.18	1.13	19.7	3.9
	A4	DE	3.97±0.16	1.19	20.5	4.1
	A5	CMA-ES	4.38±0.084	1.31	17.2	3.4
	A6	EDA	4.17±0.16	1.01	16.9	3.6
C. raciborskii Data	A1	HC	14320.77±1280.43	3.5	36.8	9.0
	A2	SA	14467.44±915.88	2.8	24.7	4.7
	A3	GA	15674.24±573.03	2.4	26.5	6.2
	A4	DE	15217.4±510.57	2.6	28.7	6.7
	A5	CMA-ES	18772.74±846.83	3.3	31.4	7.8
	A6	EDA	16373.12±1005.84	2.7	25.7	5.6

In contrast, the rule models obtained by SA on average had 17 nodes and 3 parameters for Chl-*a* data, 25 nodes and 5 parameters for *C. raciborskii* cell data. Undoubtedly, the increased number of parameters and model complexity resulted in the higher execution cost of HC. We have not determined the reason that causes HC to lead to more complex individuals, but this will be the subject of future work.

Table III shows the best rule models obtained by the six PO algorithms for Chl-*a* data after 100 runs of each algorithm. The IF condition branches of all those rule models clearly indicate threshold values of some water quality parameters (i.e., SiO_2, EC, WT, TN, etc.) which stimulate phytoplankton growth and result in higher Chl-*a* concentrations. This demonstrates that the HEA has the most advantage of automatically discovering the easily understandable IF-THEN-ELSE rules hidden in the measured data, and can be regarded as a powerful intelligent data mining tool for ecological data.

It is interesting to see that each threshold turning up in the six best models has a valid value for the specific water quality parameter. As mentioned previously, all the parameters were initialized with the same range [0, 500]. However after parameter optimization, they adaptively reached different values within the valid range of the associated water quality parameter. For example, for the best HC model, the threshold values for SiO_2 were small values as 3.431 and 7.114, whereas the threshold for EC was larger value at 356.748.

For the six best models found by the different PO algorithms, the HC model obviously was the best with the lowest RMSE (3.21), followed by the GA, DE, and EDA models. Although the total RMSEs of those three models were very close, their model structures were quite different. Comparatively, the worst models were the SA and CMA-ES models, both of which had RMSEs greater than 3.9.

The validation results for the six best Chl-*a* models on the whole data set are illustrated in Fig. 4. The graph of measured data shows that the two most prominent summer peaks occurred in 2000 and 2002, respectively. Moderate peak events were observed annually in the remaining years except 2003. As shown in Fig. 4, generally all six best models predicted the timing of peak events in every year quite well and their differences lie in the predicted magnitudes of the peaks. In terms of their predictive power, the HC model appeared best being able to match magnitudes of measured summer peaks very well for most years from 1998 to 2009, although it did slightly overestimate the magnitudes in 2007 and 2008. In contrast, the five other models predicted the magnitudes of peak events reasonably well for the years prior to 2003 but underestimated most peaks from 2003 to 2009.

Table IV shows the best rule models for *C. raciborskii* data for each of the six PO algorithms over 100 runs. Similar to the Chl-*a* models, each IF condition branch indicates threshold values of some water quality parameters (i.e., WT, TP, NTU, SiO_2, DO, EC) or their ratios (i.e., SiO_2/pH). However, the logic combination of variable conditions becomes more complicated by the significantly increased lengths of their IF branches. The HC and SA models are significantly superior to other models because they both had much lower RMSE. The second group consists of the DE, EDA and GA models whose RMSE values were also very close. While the CMA-ES model performed worst with the largest RMSE (>15 000).

The modeling results of the six best models validated on the whole *C. raciborskii* data set are illustrated in Fig. 5. The measured data show the annual summer blooms during the years from 1998 to 2009. Among those, the biggest blooms occurred in 2000, 2001, 2002, 2003, 2006, and 2009. Results from all six models show that the predicted timing of these prominent peaks corresponds very well to the measured data. This confirms the models predictive capability to achieve our aim of giving early warning of cyanobacterial blooms. Moreover, the HC and SA models were best able to forecast the magnitudes of the summer peaks in four successive years from 2000 to 2003, where the predicted values correlated well with the measured data. In contrast, other models underestimated those peaks in most cases. Due to the fact that all six models were unable to accurately predict the magnitudes of the summer peaks in 2006 and 2009, it seems challenging work for the HEA to get good predictions for those two years.

Finally, the results of the experiments were analyzed using a one-way ANOVA test combined with a Tukey's Post Hoc Test for multiple comparisons using a 95% confidence level as suggested by [51]. The data we used for statistical analysis were the total RMSE values in 100 runs for the six PO

TABLE III
BEST MODELS OBTAINED FOR CHL–*a* AFTER 100 RUNS OF EACH OF THE SIX PO ALGORITHMS

PO Algo.	Best Model	Total RMSE
HC	IF ((SiO_2<3.431)OR((SiO_2<7.114)AND(EC<=356.748))) THEN Chl-*a* = ln(|(((DO/TP-EC*2.177)/((TN-0.508)/SiO_2))|) ELSE Chl-*a* = ln(|(SiO_2+TN/(pH- SiO_2))|)	3.21
SA	IF (EC>=357.207) THEN Chl-*a* = pH+(ln(|TN|)- SiO_2+2.941)+ln(|(TP*(17.145-NTU))|) ELSE Chl-*a* = ln(|ln(|(TP* SiO_2)|)|)+WT-15.12	3.90
GA	IF ((SiO_2>=7.114)OR((SiO_2>=3.376)AND(EC>355.991))) THEN Chl-*a* = pH-ln(|(EC-39.083)|) ELSE Chl-*a* = ln(|ln(|(NTU/0.895)|)|)+3*TN+pH	3.42
DE	IF (EC<362.372) THEN Chl-*a* = (99.57-pH)*pH*pH*WT*8.0469-55.399/WT ELSE Chl-*a* = pH-(SiO_2-pH)/11.346-(SiO_2-2.348)*(SiO2-WT/23.117)	3.57
CMA -ES	IF (SiO_2>7.27) THEN Chl-*a* = ln(|(pH+WT- SiO_2-TP*pH)|) ELSE Chl-*a* = ln(|(exp(SiO_2)*NTU*WT-exp(pH))|)	3.96
EDA	IF ((WT>=24.078)AND(TN<=0.574)) THEN Chl-*a* = ln(|(pH*pH-exp(SiO_2)*(89.096-NTU*110.222))|) ELSE Chl-*a* = ln(|((pH/NTU)*(pH*pH+NTU*21.429))|)	3.72

TABLE IV
BEST MODELS OBTAINED FOR *C. raciborskii* AFTER 100 RUNS OF EACH OF THE SIX PO ALGORITHMS

PO Algo.	Best Model	Total RMSE
HC	IF((SiO_2<6.412)AND((WT*SiO_2)>126.629)AND(EC>=257.975) AND(NTU>2.365)) THEN *C. raciborskii* = (2*NTU+exp(pH)-498.514)*16.653 ELSE *C. raciborskii* = (WT-ln(|TN|)-21.73)*(WT-TN/DO-20.543) *(NTU*5.376+296.446)	11567
SA	IF ((TP<=0.028)AND((SiO_2/pH)<=0.734)) THEN *C. raciborskii* = (NTU+exp(SiO_2)+WT*15.933-355.932) *(WT*21.906-445.928) ELSE *C. raciborskii* = (2*WT+152.225)*(WT*19.638-452.542)	12053
GA	IF ((SiO_2<=6.424)AND((NTU>=310.704)OR(WT>=25.713))) THEN *C. raciborskii* = ((((WT*pH)+(SiO_2*44.835))*20.755)*SiO_2) ELSE *C. raciborskii* = WT*(EC*TN+4.009)	14135
DE	IF ((NTU<=268.448)AND(((WT*NTU)<=66.423)OR(WT<=25.82))) THEN *C. raciborskii* = (WT+9.31)*pH*WT ELSE *C. raciborskii* = SiO_2*10059.73	13861
CMA -ES	IF (WT*ln(|WT|)<=82.596) THEN *C. raciborskii* = NTU*96.246+2612.5 ELSE *C. raciborskii* = (WT+NTU+WT*SiO_2+78.5)*131.266	15234
EDA	IF ((DO<4.464)OR(DO>9.568)OR(SiO_2<=4.464) OR((WT-SiO_2)<19.189)) THEN *C. raciborskii* = exp(pH) ELSE *C. raciborskii* = exp(pH)*14.743-2*exp(DO)	14029

algorithms and two data sets. The results show that the performance order of the six algorithms is slightly different for Chl-*a* and *C. raciborskii*.

1) For the Chl-*a* data, the order is A1(HC)→(A3(GA), A4(DE))→A6(EDA) →(A2(SA), A5(CMA-ES)).
2) For the *C. raciborskii* data, the order is (A1(HC), A2(SA))→A4(DE)→A3(GA)→A6(EDA) →A5(CMA-ES).

Their main difference is the rank of SA. So it can be concluded that based on the two ecological data sets, HC is statistically the best method, followed by DE, GA, EDA whose performances are comparable, whereas the CMA-ES always performs worst and the performance of SA may vary with different case studies.

D. Model Selection

In this paper, we presented six parameter optimization algorithms developed under the framework of HEA. We produced a set of candidate ecologically plausible rule models that comparably match measured data making it difficult to select the best one.

In this section, we applied model selection theory [52] to compare and rank the best models obtained by the six alternative PO algorithms as applied to the two test data sets. We allocated two scores to each model using small-sample corrected Akaike's information criterion (AIC) [53] and minimum description length (MDL) [52] criterion according to the following formulas:

$$\text{AIC} = D\log(\text{RSS}) + 2K\left(\frac{D}{D-(K+1)}\right)$$

and

$$\text{MDL} = D\log(\text{RSS}) + K\log(D)$$

where

$$\text{RSS} = \sum_{i=1}^{D} (\hat{y}_i - y_i)^2$$

is the residual sum of square errors, D the number of total data points, and K the number of GP tree nodes for each rule model.

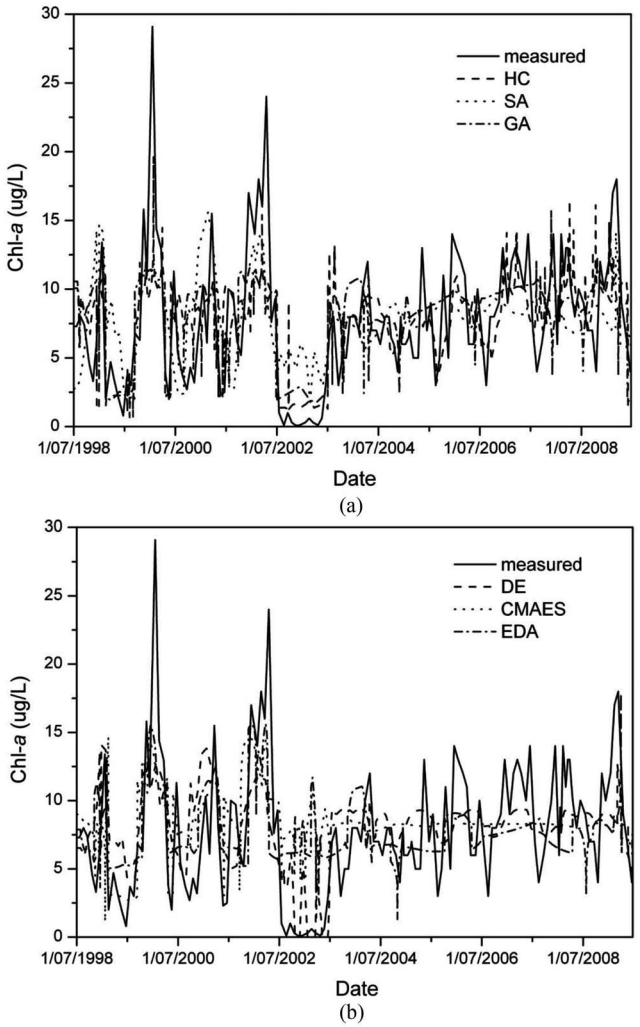

Fig. 4. Validation results on the whole data set of Chl-*a* for the best models obtained from 100 runs of each of the six PO algorithms. (a) HC, SA, and GA. (b) DE, CMA-ES, and EDA.

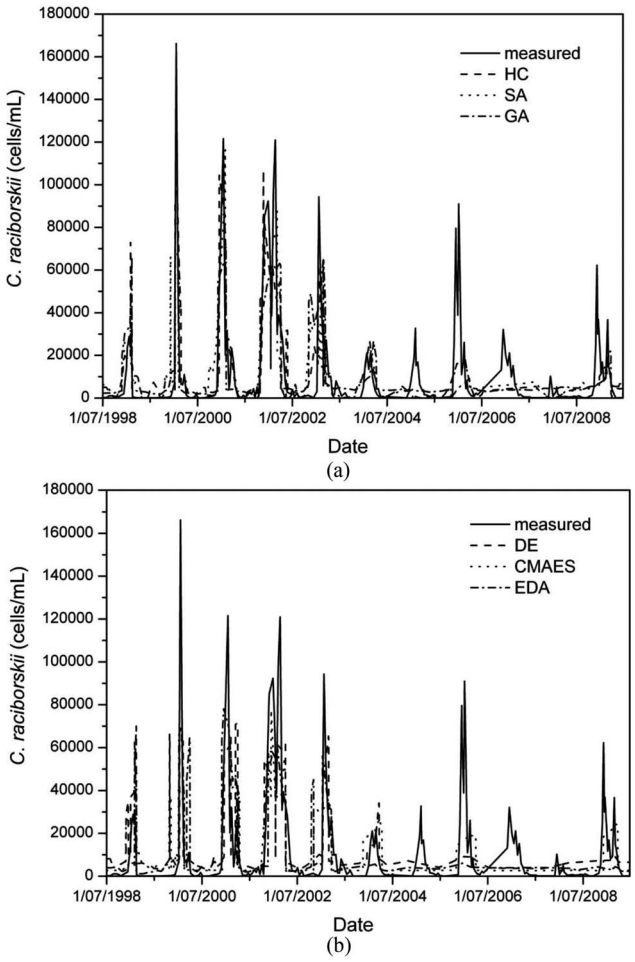

Fig. 5. Validation results on the whole data set of *C. raciborskii* for the best models obtained from 100 runs of each of the six PO algorithms. (a) HC, SA, and GA. (b) DE, CMA-ES, and EDA.

y_i and $\hat{y}_i$ are the measured value and the predicted value of the ith data point, respectively. Both scores are approximations of theoretical measures that are not computable in general. AIC is an approximation of Kullback–Leibler divergence that measures the amount of information lost when building a model. Whereas the MDL criterion aims at capturing the general idea of choosing the model that provides the shortest description of the data. These two scores share the same first part, $D\log(\text{RSS})$, which evaluates how good a model replicates the prefixed behavior. However, they differ in the second part that penalizes complex models with more parameters over simpler ones. This second part aims at preventing the overfitting of the sample data when using complex models. AIC penalizes models with a number of parameters approaching the number of data points more strongly than MDL does.

The scores associated with each model are presented in Table V and they determine the following rank order for the models.

1) For the Chl-*a* data:
   Model(HC)→Model(GA)→Model(DE)→Model(EDA)
   →Model(SA)→Model(CMA-ES).
2) For the *C. raciborskii* data:
   Model(HC)→Model(SA)→Model(DE)→Model(EDA)
   →Model(GA)→Model(CMA-ES).

IV. Further Experiments

We conducted some further experiments in two aspects. First, all the experiments described above show that when applying different PO algorithms at each generation, it produced a variety of models with different accuracy and structures when the evolution process was completed. So, we wondered what the models would look like if we did not carry out parameter optimization during evolution except during the final generation. That is, only structure optimization using GP was performed during the evolution. When the maximum number of generations MAXGENO was reached, a specific algorithm was applied to optimize the model parameters for each individual in the final population. For each test data set, we conducted 100 independent runs for each PO algorithm and present the statistical results in Table VI. For both data

TABLE V
AIC AND MDL SCORES ASSOCIATED WITH THE BEST MODELS OBTAINED USING THE SIX PO ALGORITHMS ON THE TWO TEST DATA SETS

Data Sets	Chl-*a* Data				*C. raciborskii* Data			
PO Algo.	RSS	*K*	AIC	MDL	RSS	*K*	AIC	MDL
HC	41349	33	18583	18636	5.37×10^{11}	47	47144	47218
SA	60910	27	19246	19289	5.83×10^{11}	36	47265	47322
GA	46829	28	18790	18834	8.01×10^{11}	29	47804	47850
DE	51096	35	18956	19011	7.71×10^{11}	23	47725	47762
CMA-ES	62811	23	19291	19328	9.31×10^{11}	22	48052	48087
EDA	55367	32	19090	19140	7.89×10^{11}	28	47776	47820

TABLE VI
STATISTICAL COMPARISON OF THE SIX PO ALGORITHMS FOR THE TWO TEST DATA SETS AFTER 100 RUNS OF EACH ALGORITHM WHEN PARAMTER OPTIMIZATION IS CARRIED OUT AFTER EVOLUTION

Data Sets	No.	PO Algo.	Total RMSE (Mean±SD)	Run-Time (sec.)	No. of Total Tree Nodes	No. of Parameters to Optimize
Chl-*a* Data	A1	HC	4.36±0.093	63	17.4	3.5
	A2	SA	4.39±0.084	40	17.1	3.2
	A3	GA	4.37±0.074	61	17.3	3.6
	A4	DE	4.35±0.081	58	17.1	3.2
	A5	CMA-ES	4.40±0.097	68	16.7	3.1
	A6	EDA	4.38±0.095	61	17.1	3.2
C. raciborskii Data	A1	HC	18475.36±3502.37	74	23.7	4.3
	A2	SA	18350.51±1165.99	52	23.3	4.8
	A3	GA	18502.92±912.61	73	23.3	4.2
	A4	DE	18406.53±2158.65	69	23.9	4.2
	A5	CMA-ES	19541.71±6190.18	64	25.2	4.3
	A6	EDA	18588.74±807.39	80	25.0	4.2

TABLE VII
STATISTICAL COMPARISON OF 10 000 RANDOM PARAMETER MODELS BASED ON THE BEST CHL-*a* MODEL STRUCTURES OBTAINED FOR EACH OF THE SIX PO ALGORITHMS

Best Model	Total RMSE		
	Min	Max	Mean±SD
HC	4.08	8.93	5.14±0.53
SA	4.94	497.1	196.73±140.84
GA	4.60	244.88	122.25±69.4
DE	4.31	2057.66	632.87±574.26
CMA-ES	3.91	6.38	4.92±0.10
EDA	3.98	6.36	4.57±0.25

sets, the model accuracy for different PO algorithms measured by the mean total RMSE was very close with similar model complexity in terms of the mean number of total tree nodes and parameters to optimize. By comparing with the results in Table II, we conclude that optimizing the model parameters for rule models during evolution improves the model accuracy significantly, especially when using some better-performing algorithms such as HC and DE. However, it is notable that both the model complexity and computational time increases significantly when the parameter optimization is carried out for every generation. Typically for the HC algorithm, the number of total tree nodes and parameters to optimize increases by 1.5–2 fold compared with only optimizing parameters at the final generation. Moreover, it takes several hours to finish one run instead of a few minutes without parameter optimization during the evolution.

Second, based on the best Chl-*a* models shown in Table III, we conducted some experiments to further investigate the model structures and their linkage to the constant values contained in the model.

As in the first step, for each best model, we froze the model structure and randomly generated a set of floating numbers to fill the corresponding constants in the model within the range of 0 to 500 and then calculated its total RMSE. This procedure was repeated 10 000 times and the statistical results are shown in Table VII. Obviously, the six best models were divided into two groups in terms of distinct RMSE values. The HC, CMA-ES, and EDA models (the first group) had lower mean RMSEs (<6.0) and standard deviations (<1.0). This indicates that those models had relatively stable model structures and the change of the random constants had little effect on the model accuracy. On the contrary, the SA, GA, and DE models (the second group) had larger mean RMSEs (>200) and higher standard deviations (>60) and their total RMSE changed significantly with the random constant values. For example, for the best DE model, the minimal RMSE was 4.31 while the maximal RMSE could reach 2057.66. Because it is possible for each model in the second group to reach a small minimal RMSE, we considered that these model structures did not perform poorly but their accuracy could be improved significantly by parameter optimization.

By following the above experiments, the next step is that without loss of representability, we chose a model from the second group with moderate RMSE and moderate number of parameters to optimize. In our case, we chose the best GA model that has a mean RMSE of 122.25 and consists of six parameters to be optimized. We rewrote the model structure as follows:

TABLE VIII
STATISTICAL COMPARISON OF THE SIX PO ALGORITHMS AFTER 100 RUNS OF EACH ALGORITHM BASED ON THE MODEL STRUCTURE OF THE BEST GA CHL-*a* MODEL

PO Algo.	Total RMSE (Mean±SD)	TRAINING RMSE (Mean±SD)	TESTING RMSE (Mean±SD)	Run-Time (sec.)
HC	3.81±0.13	3.82±0.14	3.78±0.13	12.76
SA	4.27±0.078	4.28±0.078	4.23±0.078	12.98
GA	3.96±0.082	3.97±0.082	3.92±0.085	12.92
DE	3.87±0.23	3.88±0.24	3.85±0.22	12.80
CMA-ES	6.43±3.25	7.43±3.76	12.84±6.51	13.09
EDA	5.05±0.14	5.83±0.17	10.07±0.29	13.24

TABLE IX
STATISTICAL COMPARISON OF OPTIMIZED PARAMETERS FOR THE SIX PO ALGORITHMS AFTER 100 RUNS OF EACH ALGORITHM BASED ON THE MODEL STRUCTURE OF THE BEST GA CHL-*a* MODEL

PO Algo.	P1	P2	P3	P4	P5	P6
HC	61.97±167.90	70.77±180.39	338.96±120.65	273.86±108.89	2.12±0.20	1.26±0.64
SA	356.46±0.078	386.41±243.90	412.52±248.83	412.84±255.01	2.18±0.34	0.51±0.47
GA	27.42±95.43	189.28±251.17	397.53±156.34	269.92±163.48	2.16±0.10	0.93±0.39
DE	227.65±255.19	109.88±226.04	323.88±124.01	319.87±322.41	1.36±0.61	1.61±1.02
CMA-ES	211.77±148.95	225.46±143.38	275.39±146.03	244.40±116.33	231.06±130.05	6.49±11.15
EDA	932.73±88.42	588.31±113.70	621.16±236.78	668.62±276.14	743.22±82.54	0.14±0.53

TABLE X
BEST PARAMETER SETS OBTAINED BY THE SIX PO ALGORITHMS AFTER 100 RUNS OF EACH ALGORITHM BASED ON THE MODEL STRUCTURE OF THE BEST GA CHL-*a* MODEL

PO Algo.	P1	P2	P3	P4	P5	P6	Total RMSE
HC	7.11	3.37	356.76	24.73	0.92	3.14	3.41
SA	733.02	3.34	358.04	651.62	1.12	2.15	3.87
GA	7.10	3.40	342.94	61.48	2.13	1.51	3.73
DE	7.31	3.37	354.41	75.89	0.95	3.83	3.47
CMA-ES	982.10	621.07	428.92	431.39	780.58	0.027	5.03
EDA	163.17	3.38	355.09	302.96	386.04	0.03	4.26

IF $((SiO_2 >= P1) OR ((SiO_2 >= P2) AND (EC > P3)))$
THEN Chl-a = pH-ln(|(EC-P4)|)
ELSE Chl-a = ln(|ln(|(NTU/P5)|)|) + P6*TN + pH.

Using the same random training and test data sets as when the best GA model was obtained, we froze the model structure and applied six PO algorithms to optimize the six parameters (P1∼P6) separately. One hundred independent runs were conducted for each algorithm. All the parameter settings for each PO algorithm are the same as previous experiments except $N_{eval} = 10\,000$.

Table VIII shows the RMSE values after 100 runs of each algorithm when using the six PO algorithms to optimize the six parameters. Based on the three average RMSEs (total, training, and testing) the HC, DE, GA models performed best followed by the SA and EDA models. Again, the CMA-ES was the worst performer. These results compare well with previous results in Section III. As expected, in terms of execution time, HC runs fastest, with the two more complex algorithms, EDA and CMA-ES, running slowest.

Table IX shows the results of the statistical comparison of the six parameters based on 100 runs of each of the six PO algorithms. We found that for each algorithm the parameters P1∼P4 changed significantly in different runs but not so for P5 and P6. This suggests that a large number of different optimal combinations exist for the six parameters to achieve a small RMSE. Table X shows the best parameter sets obtained by the six PO algorithms. The results prove again that based on the total RMSE, the HC model performed best followed by the SA, GA, and EDA models, and that the CMA-ES model performed worst. In addition, the best parameter sets for the HC, DE, and GA models were very similar.

V. CONCLUSION AND FUTURE WORK

This paper proposes a novel evolutionary computation method for modeling cyanobacterial blooms. It has the following main features:

1) explanatory single rules with IF-THEN-ELSE structures to model the general algal growth and abundance of the tropical cyanobacterium *C. raciborskii*;
2) a HEA that performs structural and parameter optimization of rule models simultaneously;

3) six population-based algorithms that are applied to perform parameter optimization within the hybrid methodology including HC, SA, GA, DE, CMA-ES, and EDA.

The effectiveness of the methodology was tested by time-series modeling of Chl-*a* concentrations and *C. raciborskii* abundance in Lake Wivenhoe. Based on the 7-day-ahead forecasting of Chl-*a* and *C. raciborskii*, we drew the following conclusions.

1) Whatever PO method was applied in the hybrid EA, our modeling algorithm always found explanatory rule models whose IF conditions indicate threshold values of water quality conditions that trigger or suppress phytoplankton growth, which is reflected in Chl-*a* concentrations and *C. raciborskii* cell concentrations. These threshold values provide crucial insights about water conditions that most likely favor growth of cyanobacteria. More interestingly, our algorithm is able to propose a variety of alternative rule models with distinctive IF condition branches. These results can then inform laboratory ecological experiments to test a range of hypothesis implied by the evolved models.
2) Both the experiments and their statistical analysis suggest that when comparing different PO methods, HC always performed best followed by DE, GA, and EDA as a group. CMA-ES always performed worst and the performance of SA was dependent on the data set being used. This result was surprising and we surmised that this is mainly driven by the high complexity and uncertainty of the model parameters contained in the rule model. Unlike some benchmark functions, the number of random constants in a rule is arbitrary and their value ranges can be quite different. Hence, it is hard for some sophisticated methods with complex mathematical computation (i.e., CMA-ES) to capture the hidden relationship among multiple model variables and find the global optima with a small population size. In this paper, we found that the default population size λ in CMA-ES was less than 10 in most cases. On the contrary, some classical optimization techniques, such as HC and SA, have the advantage of exploring the irregular search space by using multiple starting points and iterative random searches. They seem to work better for these rule models when optimizing the random constants with arbitrary numbers that are evolved based on real-world ecological data.

To further improve the predictive accuracy of rule models, we will focus our future research on:

1) developing an ensemble of alternative rules and using the averaged result of the alternative rules as output;
2) designing rule set models with multilevel embedded IF-THEN-ELSE structures.

Comparing advantages and disadvantages of model structures based on single rules, an ensemble of multiple single rules, and a rule set with embedded single rules by means of real-world ecological data will further enhance predictive modeling by evolutionary computation.

APPENDIX A

Pseudo-code of HC

```
BEGIN
  check all constants contained in current rule;
  evalnum ← 0;
  best rule ← current rule;
  best fitness ← fitness of current rule;
  while(evalnum < N_eval)
  {
    n ← 0;
    current rule ← best rule;
    current fitness ← best fitness;
    while(n < N_neighbors)
    {
      produce a temporary rule by performing
      Gaussian mutations on the constants
      randomly chosen from Tree1,Tree2 and
      Tree3 respectively;
    calculate the fitness of the temporary rule;
    evalnum ← evalnum + 1;
    if (temporary fitness is better than best fitness)
    {
      best fitness ← temporary fitness;
      best rule ← current rule;
    }
      n ← n + 1;
   }
  }
  output the best rule and best fitness;
END
```

APPENDIX B

Pseudo-code of SA

```
BEGIN
  check all constants contained in current rule;
  evalnum ← 0;
  best rule ← current rule;
  best fitness ← fitness of current rule;
  T ← T_0;
  frozen ← false;
  totaln ← 0;
  while ((!frozen || totaln < K) && evalnum < N_eval)
  {
    cooling ← false;
    k ← 0;
    while (k < L)
    {
    produce a temporary rule by performing
      Gaussian mutations on the constants
      randomly chosen from Tree1, Tree2 and Tree3
      respectively;
    calculate the fitness of the temporary rule;
    evalnum ← evalnum + 1;
    △E ← temporary fitness - current fitness;
    if (△E < 0)
    {
      cooling ← true;
      current rule ← temporary rule;
```

```
        current fitness ← temporary fitness;
        best fitness ← temporary fitness;
        best rule ← current rule;
      }
      else if (e^((−1)∗ΔE/T) > random(0,1))
      {
        current rule ← temporary rule;
        current fitness ← temporary fitness;
      }
      k ← k + 1;
      totaln ← totaln + 1;
    }
    if (cooling)
      T ← 0.9 * T;
    else frozen ← true;
    }
  output the best rule and best fitness;
END
```

APPENDIX C

Pseudo-code of GA

```
BEGIN
  t ← 0;
  evalnum ← 0;
  randomly initialize a parameter population P(0);
  calculate the fitness of the individuals in P(0);
  evalnum ← evalnum + 1 (for each calculation);
  while (evalnum < N_eval)
  {
    sort population P(t) in terms of fitness;
    randomly choose M individuals from P(t) to do
    multiparent crossover;
    calculate the fitness of the new produced
    individual;
    evalnum ← evalnum + 1;
    if(new fitness is better than the worst fitness)
    replace the worst one with the new individual;
    P(t) ← P(t + 1);
    t ← t + 1;
  }
output the best rule and best fitness;
END
```

Multiparent crossover [17]

Randomly select M different individuals from the population ($M > 2$) denoted as $P_1, P_2, \ldots, P_M$ where $P_i = (p_1\ p_2 \ldots p_k)$ (i:1∼M) and k is the number of parameters to optimize. Accordingly produce M random coefficientsα_i, which satisfies $a \le \alpha_i \le b$ ($a < 0$, $b > 1$) and $\sum_{i=1}^{M} \alpha_i = 1$, then generate a new individual, P_{new}, by the non-convex linear combination of those M individuals as the following:

$$P_{new} = \sum_{i=1}^{M} \alpha_i P_i.$$

APPENDIX D

Pseudo-code of DE

```
BEGIN
  t ← 0;
  evalnum ← 0;
  randomly initialize a parameter population P(0);
  calculate the fitness of the individuals in P(0);
  evalnum ← evalnum + 1(for each calculation);
  while(evalnum < N_eval)
  {
    sort population P(t) and get the best
      individual bestp(t);
    for (i = 0; i < Popsize; i++)
    {
    produce a new offspring p_i*(t) for individual
      p_i(t) by randomly choosing one of the five
      alternative DE schemes;
    calculate the fitness of p_i*(t);
    evalnum ← evalnum + 1;
    if (fitness(p_i*(t)) is better than fitness(p_i(t))
      p_i(t + 1) ← p_i*(t);
    else p_i(t + 1) ← p_i(t);
  }
  t ← t + 1;
 }
 output the best rule and best fitness;
END
```

Five Alternative DE Schemes ([38])

1. DE-rand1:
   $p_i*(t) = p_{r1}(t) + F*(p_{r2}(t) - p_{r3}(t))$
2. DE-rand2:
   $p_i*(t) = p_{r1}(t) + F*(p_{r2}(t) + p_{r3}(t) - p_{r4}(t) - p_{r5}(t))$
3. DE-best1:
   $p_i*(t) = bestp(t) + F*(p_{r1}(t) - p_{r2}(t))$
4. DE-best2:
   $p_i*(t) = bestp(t) + F*(p_{r1}(t) + p_{r2}(t) - p_{r3}(t) - p_{r4}(t))$
5. DE-randtobest1:
   $p_i*(t) = p_i(t) + \lambda*(bestp(t) - p_i(t)) + F*(p_{r1}(t) - p_{r2}(t))$

Notes:

r1, r2, r3, r4, r5 ∈[0,*Popsize*-1]: randomly chosen integer and mutually different.

λ, $F \in [0,2]$ (usually set $\lambda = F$).

REFERENCES

[1] G. M. Hallegraeff, "Harmful algal blooms in the Australian region," *Marine Pollution Bull.*, vol. 25, nos. 5–8, pp. 186–190, 1992.

[2] T. H. Donnelly, M. R. Grace, and B. T. Hart, "Algal blooms in the Darling-Barwon River, Australia," *Water Air Soil Pollution*, vol. 99, nos. 1–4, pp. 487–496, 1997.

[3] S. E. Shumway, "A review of the effects of algal blooms on shellfish and aquaculture," *J. World Aquaculture Soc.*, vol. 21, no. 2, pp. 65–104, 1990.

[4] S. J. Pittman and K. M. Pittman, "Short-term consequences of a bentic cyanobacterial bloom (Lyngbya majuscule Gomont) for fish and penaeid prawns in Moreton Bay (Queesland, Australia)," *Estuarine Coastal Shelf Sci.*, vol. 63, no. 4, pp. 619–632, 2005.

[5] Atech Group, "Cost of algal bloom," LWRRDC Occasional Paper 26/99. Land and Water Resources Research and Development Corporation, Canberra, ACT, Australia, 2000, pp. 1–42.

[6] R. A. Park, "A generalized model for simulating lake ecosystems," *Simulation*, vol. 23, no. 2, pp. 33–50, 1974.

[7] H. Pei and J. Ma, "Study on the algal dynamic model for West Lake, Hangzhou," *Ecol. Modelling*, vol. 148, no. 1, pp. 67–77, 2002.

[8] U. Fayyad, G. Piatetsky-Shapiro, and P. Smyth, "From data mining to knowledge discovery: An overview," in *Advances in Knowledge Discovery and Data Mining*, U. Fayyad, G. Piatetsky-Shapino, and P. Smyth, Eds. Boston, MA, USA: MIT Press, 1996, pp. 1–34.

[9] B. Wei, N. Sugiura, and T. Maekawa, "Use of artificial neural network in the prediction of algal blooms," *Water Res.*, vol. 35, no. 8, pp. 2022–2028, 2001.

[10] F. Recknagel, M. French, P. Harkonen, and K. Yabunaka, "Artificial neural network approach for modelling and prediction of algal blooms," *Ecol. Modelling*, vol. 96, nos. 1–3, pp. 11–28, 1997.

[11] Q. Chen and A. Mynett, "Integration of data mining techniques and heuristic knowledge in fuzzy logic modelling of eutrophication in Taihu Lake," *Ecol. Modelling*, vol. 162, nos. 1–2, pp. 55–67, 2003.

[12] H. R. Maier, T. Sayed, and B. J. Lence, "Forecasting cyanobacterium *Anabaena spp*. in the River Murray, South Australia, using B-spline neurofuzzy models," *Ecol. Modelling*, vol. 146, nos. 2–3, pp. 85–96, 2001.

[13] T. Bäck, U. Hammel, and H.-P. Schwefel, "Evolutionary computation: Comments on the history and current state," *IEEE Trans. Evol. Comput*., vol. 1, no. 1, pp. 5–16, Jan. 1997.

[14] F. Recknagel, J. Bobbin, P. Whigham, and H. Wilson, "Comparative application of artificial neural networks and genetic algorithms for multivariate time series modelling of algal blooms in freshwater lakes," *J. Hydroinformatics*, vol. 4, no. 2, pp. 125–133, 2002.

[15] H. Cao, F. Recknagel, B. Kim, and N. Takamura, "Hybrid evolutionary algorithm for rule set discovery in time-series data to forecast and explain algal population dynamics in two lakes different in morphometry and eutrophication," in *Ecological Informatics,* 2nd ed., F. Recknagel, Ed. Berlin, Germany: Springer-Verlag, 2006, ch. 17, pp. 347–367.

[16] G. B. Fogel and M. Cheung, "Derivation of quantitative structure-toxicity relationships for ecotoxicological effects of organic chemicals: Evolving neural networks and evolving rules," in *Proc. Congr. Evol. Comput*., vol. 1. 2005, pp. 274–281.

[17] J. Yu, H. Cao, Y. Chen, L. Kang, and H. Yang, "A new approach to estimation of the electrocrystallization parameters," *J. Electroanal. Chem.*, vol. 474, no. 1, pp. 69–73, 1999.

[18] J. R. Koza, *Genetic Programming: On the Programming of Computers by Means of Natural Selection*. Cambridge, MA, USA: MIT Press, 1992.

[19] J. R. Koza, *Genetic Programming II: Automatic Discovery of Reusable Programs*. Cambridge, MA, USA: MIT Press, 1994.

[20] A. C. Davison and D. V. Hinkley, *Bootstrap Methods and Their Application*. Cambridge, U.K.: Cambridge Univ. Press, 1997.

[21] R. U. Murnane, "Determination of the thorium and particulate matter cycling parameters at Station P: A reanalysis and comparison of least squares techniques," *J. Geophys. Res*., vol. 99, no. C2, pp. 3393–3405, Feb. 1994.

[22] P. Prunet, J.-F. Minster, D. Ruiz-Pino, and I. Dadou, "Assimilation of surface data in a one-dimensional physical-biogeochemical model of the surface ocean: 1. Method and preliminary results," *Glob. Biogeochem. Cycles*, vol. 10, no. 1, pp. 111–138, 1996.

[23] S. Marsili-Libelli and E. Giusti, "Water quality modelling for small river basins," *Environmental Modelling Softw.*, vol. 23, no. 4, pp. 451–463, 2008.

[24] G. T. Evans, "The role of local models and data sets in the Joint Global Ocean Flux Study," *Deep-Sea Res.*, vol. 46, no. 8, pp. 1369–1389, 1999.

[25] M. J. R. Fasham and G. T. Evans, "The use of optimization techniques to model marine ecosystem dynamics at the JGOFS station at 47°N 20°W," *Phil. Trans. R. Soc. Lond*., vol. 348, no. 1324, pp. 203–209, 1995.

[26] R. J. Matear and G. Holloway, "Modeling the inorganic phosphorus cycle of the North Pacific using an adjoint data assimilation model to assess the role of dissolved organic phosphorus," *Glob. Biogeochem. Cycles*, vol. 9, no. 1, pp. 101–119, 1995.

[27] B. A. Ward, M. A. M. Friedrichs, T. R. Anderson, and A. Oschlies, "Parameter optimisation techniques and the problem of underdetermination in marine biogeochemical models," *J. Marine Syst.*, vol. 81, nos. 1–2, pp. 34–43, 2010.

[28] G. C. Hurtt and R. A. Armstrong, "A pelagic ecosystem model calibrated with BATS and OWSI data," *Deep-Sea Res.*, vol. 46, no. 1, pp. 27–61, 1999.

[29] R. J. Matear, "Parameter optimization and analysis of ecosystem models using simulated annealing: A case study at station P," *J. Marine Res.*, vol. 53, no. 4, pp. 571–607, 1995.

[30] J. J. Vallino, "Improving marine ecosystem models: Use of data assimilation and mesocosm experiments," *J. Marine Res.*, vol. 58, no. 1, pp. 117–164, 2000.

[31] L. Ingber, "Simulated annealing: Practice versus theory," *Math. Comput. Modelling*, vol. 18, no. 11, pp. 29–57, 1993.

[32] M. Mitchell, *An Introduction to Genetic Algorithm*. Cambridge, MA, USA: MIT Press, 1996.

[33] D. E. Goldberg, *Genetic Algorithms in Search, Optimization and Machine Learning*. Reading, MA, USA: Addison-Wesley, 1989.

[34] T. Bäck, "Introduction to evolutionary algorithms," in *Evolutionary Computation 1: Basic Algorithms and Operators*, T. Bäck, D. B. Fogel, and Z. Michalewicz, Eds. Bristol, U.K.: Instit. Phys., 2000, ch. 7, pp. 59–63.

[35] R. Storn and K. Price, "Differential evolution: A simple and efficient heuristic for global optimization over continuous spaces," *J. Global Optimization*, vol. 11, no. 4, pp. 341–359, 1997.

[36] O. Hrstka and A. Kučcerovà, "Improvement of real coded genetic algorithm based on differential operators preventing premature convergence," *Advance Eng. Software*, vol. 35, nos. 3–4, pp. 237–246, 2004.

[37] J. Vesterstroem and R. Thomsen, "A comparative study of differential evolution, particle swarm optimization, and evolutionary algorithms on numerical benchmark problems," in *Proc. Congr. Evol. Comput*., vol. 2. 2004, pp. 1980–1987.

[38] R. Storn, "On the usage of differential evolution for function optimization," in *Proc. Biennial Conf. North Amer. Fuzzy Inf. Process. Soc*., 1996, pp. 519–523.

[39] N. Hansen, S. D. Müller, and P. Koumoutsakos, "Reducing the time complexity of the derandomized evolution strategy with covariance matrix adaptation (CMA-ES)," *Evol. Comput.*, vol. 11, no. 1, pp. 1–18, 2003.

[40] N. Hansen, "The CMA evolution strategy: A comparing review," in *Towards a New Evolutionary Computation: Advances in Estimation of Distribution Algorithms*, J. A. Lozano, P. Larrañaga, I. Inza, and E. Bengoetxea, Eds. Berlin, Germany: Springer, 2006, pp. 75–102.

[41] N. Hansen. (2005). *References to CMA-ES Applications* [Online]. Available: http://www.bionik.tu-berlin.de/user/niko/cmaapplications.pdf

[42] H.-G. Beyer and D. V. Arnold, "Qualms regarding the optimality of cumulative path length control in CSA/CMA-evolution strategies," *Evol. Comput.*, vol. 11, no. 1, pp. 19–28, 2003.

[43] H. Mühlenbein and G. Paaß, "From recombination of genes to the estimation of distributions: I. Binary parameters," in *Proc. PPSN IV*, 1996, LNCS 1411, pp. 178–187.

[44] P. Larrañaga, "Review on estimation of distribution algorithms," in *Estimation of Distribution Algorithms: A New Tool for Evolutionary Computation*, P. Larranãga and J. A. Lozano, Eds. Boston, MA, USA: Kluwer Academic, 2002, ch. 3, pp. 57–100.

[45] P. Larrañaga, R. Etxeberria, J. A. Lozano, and J. M. Peña, "Optimization in continuous domains by learning and simulation of Gaussian networks," in *Proc. Genet. Evol. Comput. Conf. Workshop Program*, 2000, pp. 201–204.

[46] S. Rudlof and M. Köppen, "Stochastic hill climbing with learning by vectors of normal distributions," in *Proc. WSC1*, 1996, pp. 60–70.

[47] M. Pelikan and H. Mühlenbein, "The bivariate marginal distribution algorithm," in *Advances in Soft Computing—Engineering Design and Manufacturing*, R. Roy, T. Furuhashi, and P. K. Chawdhry, Eds. London, U.K.: Springer, 1999, pp. 521–535.

[48] P. A. N. Bosman and D. Thierens, "Expanding from discrete to continuous estimation of distribution algorithms: The IDEA," in *Proc. PPSN VI*, 2000, LNCS 1917, pp. 767–776.

[49] P. A. N. Bosman, J. Grahl, and D. Thierens, "Adapted maximum-likelihood Gaussian models for numerical optimization with continuous EDAs," Centre Math. Comput. Sci. (CWI), Amsterdam, The Netherlands, Tech. Rep. SEN-E0704, Dec. 2007.

[50] P. T. Orr, J. P. Rasmussen, M. A. Burford, G. K. Eaglesham, and S. M. Lennox, "Evaluation of quantitative real-time PCR to characterise spatial and temporal variations in cyanobacteria, *Cylindrospermopsis raciborskii* (Woloszynska) Seenaya et Subba Raju and cylindrospermopsin concentrations in three subtropical Australian reservoirs," *Harmful Algae,* vol. 9, no. 3, pp. 243–254, 2010.

[51] P. L. Lanzi, D. Loiacono, S. W. Wilson, and D. E. Goldberg, "Prediction update algorithms for XCSF: RLS, Kalman filter, and gain adaptation," in *Proc. 8th Annu. Conf. Genet. Evol. Comput.*, 2006, pp. 1505–1512.

[52] K. P. Burnham and D. R. Anderson, *Model Selection and Multimodel Inference—A Practical Information-Theoretic Approach*, 2nd ed. Berlin, Germany: Springer, 2002.

[53] P. Grumwald, "Model selection based on minimum description length," *J. Math. Psychol*., vol. 44, no. 1, pp. 133–152, 2000.

Hongqing Cao received the B.Eng. degree in computer software and the M.Eng. degree in computer science from Chongqing University, Chongqing, China, in 1993 and 1996, respectively, and the Ph.D. degree in computer software and theory from Wuhan University, Wuhan, China, in 1999.

She is a Visiting Scientist with the School of Earth and Environmental Sciences (EES), University of Adelaide, Adelaide, Australia. Her research interests include evolutionary computation, ecological modeling, computational biology, and bioinformatics.

Friedrich Recknagel received the Ph.D. and Habilitation degrees from the Dresden University of Technology, Dresden, Germany.

He is currently an Associate Professor with the School of Earth and Environmental Sciences, University of Adelaide, Adelaide, Australia, in the areas of freshwater ecology and ecological modeling.

Dr. Recknagel has been the Editor-in-Chief of *Elsevier Journal Ecological Informatics* since 2006.

Philip T. Orr received the Ph.D. degree from the University of Technology, Sydney, Australia, in 2006.

He joined CSIRO, Griffith, NSW, Australia, in 1977 as a Plant Physiologist focusing on physiology of aquatic plants. He has authored 32 papers, four book chapters, and 17 consultancy reports.

Dr. Orr is a member of the Australasian Society for Phycology and Aquatic Botany (ASPAB) and was an Associate Editor of *Phycologia*, the journal of the International Phycological Society.