Evolutionary Practice Problems Generation: More Design Guidelines

Alessio Gaspar, A. T. M. Golam Bari
Dept. Computer Science & Engineering
University of South Florida
Tampa, FL, USA
alessio@usf.edu
bari@mail.usf.edu

R. Paul Wiegand
Institute for
Simulation & Training
University of Central Florida
Orlando, FL, USA
wiegand@ist.ucf.edu

Anthony Bucci
119 Armory St.
Cambridge, MA, USA
anthony@bucci.onl

Amruth N. Kumar
Ramapo College of New Jersey
Mahwah, NJ, USA
amruth@ramapo.edu

Jennifer L. Albert
The Citadel
171 Moultrie Street
Charleston, SC, USA
jalbert@citadel.edu

Abstract

We propose to further extend preliminary investigations of the nature of the problem of evolving practice problems for learners. Using a refinement of a previous simple model of interaction between learners and practice problems, we examine some of its properties and experimentally highlight the role played by the number of values each gene may take in our encoding of practice problems. We then experimentally compare both a traditional - **P-CHC** - and Pareto-based - **P-PHC** - variants of coevolutionary algorithms. Comparisons are conducted with respect to the presence of noise in fitness evaluations, the number of values genes may take, and two distinct fitness functions. Each fitness captures an aspect of the nature of learner-problem interaction but one has been shown to induce overspecialization pathologies. We then summarize our findings in terms of guidelines on how to adapt evolutionary algorithms to tackle the task of evolving practice problems.

Introduction

Intelligent Tutoring Systems have been able to provide new opportunities for students of all disciplines to get automated feedback during unsupervised practice sessions. While much research has been dedicated to modeling learners' performance in order to adapt practice problems to their needs, less work has explored the possibility for intelligent tutors to automatically design practice problems.

The lack of a formal model of the underlying optimization problem makes Evolutionary Algorithms natural candidates for the task at hand. More specifically, Coevolutionary Algorithms - **CEA** - feature so-called "pathological dynamics" which are analogous to those observed in educational settings where a population of learners interacts with a set of practice problems. For instance, the *loss of gradient* phenomenon occurring when a population significantly outperforms its coevolutionary counterpart, is analogous to situations where assignments become too hard for a group of students. Recent theoretical advances in CEA theory open new opportunities to improve our understanding of the dynamic occurring when adapting a set of practice problems alongside a population of learners. Such insights are essential to efficient evolutionary practice problem generation.

Preliminary work in investigating the challenges posed by practice problem generation to Evolutionary Algorithms (Gaspar et al. 2016) led us to apply a state-of-the-art coevolutionary algorithm to a suite of increasingly complex approximations of the target application. We propose to revisit one of these early models, formally analyze its intrinsic limitations, and compare it to another well-known Coevolutionary problem (e.g. COMPARE-ON-ONE (De Jong and Pollack 2004)). These steps supplement our previous work and allow us to refine our understanding of the design guidelines to design suitable EA variants for the task of practice problem generation. This, in turn, is essential to establish the role, if any, that evolutionary algorithms *may* be able to play in this task before they are even applied to real students populations or compared to over approaches previously used to generate practice problems.

Background

There have been limited applications of evolutionary techniques to educational domain in general, and to automated generation of practice problems in particular. Instead, previous work focused on their potential to help personalize the delivery of content, e.g. (Huang, Huang, and Chen 2007; Chen 2008), or data-mine educational data, e.g. (Romeroa et al. 2009). While focused on coevolutionary learning in the context of the Tron light-cycle game, this approach led to interesting applications in the educational domain; e.g. (Sklar and Pollack 1998) and established the foundations for the game-theoretic study of coevolutionary learning involving human learners (Bader-Natal 2008). Even more recently, theoretical results explaining pathological coevolutionary dynamics (Bucci 2007) have helped gain insights about the difficulties encountered in an introductory programming course (Wiegand et al. 2016).

Previous work led us to identify several characteristic problems or dynamics in the EA literature, that are directly relevant to evolving practice problems in general (Gaspar et al. 2016):

Overspecialization occurs in multi-objectives optimization when some highly competitive candidate solutions only improve a subset of their objectives. In educational settings, it is analogous to learners who master a subset of practice problems without acquiring skills in all learning objectives.

Noisy evaluation is inherent to problems where external factors may affect evaluations' outcomes. In our target application, each practice problem must be evaluated via its interaction with students. However, the outcome may be influenced by learner distraction, learner fatigue or technology-related issues. Even if we were able to somehow re-expose a learner to the same problem for the first time, the outcome would unlikely be identical.

User fatigue (Llorá et al. 2005) is a serious impediment to the applicability of Interactive Evolutionary Algorithms. In such algorithms, evaluation of candidate solutions is performed by human agents who quickly become unreliable as the number of specimens they have to inspect increases. In educational applications, this problem is further exacerbated as the time and cognitive effort required to work through a single practice problem are much greater than in typical IEA applications where evaluating often boils down to expressing a subjective preference; e.g. computer-generated art.

Experiment #1 - Noisy Teacher-Learner

Problem

In order to gain insights about how co-evolutionary techniques in general, and the **P-PHC** algorithm in particular, would fare on the practice-problem evolution task, we adapted the simple stochastic model of teacher-learner co-evolution that was used in (Gaspar et al. 2016).

As in the original work, we used fixed-length integer vectors as genotypes for both learners (candidates) and practice problems (tests): $\langle g_1, g_2, g_3, g_4 \rangle$ with each of the 4 genes taking value in $[1..N_G]$. The rationale for using integer values taken from a specified range is based on the implementation requirements of our proof of concept implementation[1]. The latter is meant to have an EA variant evolve specific practice problems for novice programmers, known as Parsons puzzles (Parsons and Haden). The puzzle-like exercises have shown to be particularly helpful in developing programming skills in learners. In each Parsons puzzle, an already written correct program, accompanied by a plain English description of its goals is used. It is broken down into fragments, generally corresponding to one line of code, which are then randomly shuffled. A few of the fragments are selected and transformed so as to introduce a bug. For instance, replacing "<" by "<=" in the condition of a FOR loop would introduce a off-by-one bug. These erroneous versions of the original program fragments, which we will refer to as "distracters", are then shuffled with all the other fragments. Learners are then presented with the description of the program, along with a list of all the valid fragments and distracters, shuffled together. Their goal is to drag and drop the valid fragments from this list in order so as to reconstitute the original program while avoid using the distracters. In our proof of concept implementation, a practice problem genotype is a fixed-length integer vector. The first integer is the index of the program to be used, taken from a predefined library of programs which we wrote to be suitable for our students' level. The following integers each represent the index of a transform also taken from such a library. Each transform uses regular expressions to match specific program fragments and modify them so as to introduce bugs we have observed among students.

In our simplified model, we wanted to also use fixed-size sequences of integers as the focus of the evolutionary techniques. However, we used a simplified way to interpret them while still establishing a meaningful coevolutionary influence between the practice problems and learners entities.

To this end, we compute the sum of the genes for a learner noted by S^L, and the sum of the genes for a practice problem by S^P. S^L represents the expected number of attempts taken by the corresponding learner to solve an arbitrary practice problem. The higher this number, the more the learner is struggling. Similarly, S^P represents the difficulty level for the corresponding practice problem, also expressed as an expected number of attempts needed by an arbitrary learner to solve it. Based on these, the outcome of the interaction of a given practice problem with a specific learner is the number of attempts taken by that learner to solve it.

$$N = S^L + S^P + \text{rand}(r) \qquad (1)$$

where $\text{rand}(r)$ returns a random integer in $[0 : r - 1]$, thus capturing the variability of students' performance. Both the learner and practice problem fitnesses are respectively derived from this quantity; $F^L = -N$ and $F^P = N$. Therefore, these fitness measures are opposite for learners and practice problems but both revolve around the concept of difficulty. The number of attempts necessary for a given learner to solve a given practice problem is the fitness of the latter; the higher meaning that the practice problem is more difficult. Reciprocally, the lower this number of attempts, the higher is the fitness of the learner.

It should be clarified before to go any further that the above model should not be misconstrued as a claim that solely relying on a difficulty metric is a suitable way to measure the worth of practice problems. We do plan on investigating more pedagogically-oriented metrics when we use our system with students. However, difficulty is often found to be an essential component of more elaborate approaches such as the Zone of Proximal Development. As such, we felt that integrating difficulty measures in a minimalist model would be a reasonable approach which, while clearly not meant to capture the complex nature of real students' performance, would introduce relevant coevolutionary interaction between our two populations.

Algorithms

Interestingly, this model captures the fact that students would primarily focus on improving their performance and

[1] Source code for the proof of concept implementation under development is available on the project's sourceforge repository at https://sourceforge.net/projects/evotutoring/

skills, rather than any other metrics, such as informativeness, that would primarily facilitate the evolution of practice problems. For this reason, we explored two variants of **P-PHC**; **P-PHC-I** and **P-PHC-P**. The first one uses informativeness, as in the original **P-PHC**, to drive the evolution of the learners. The second one relies on performance in terms of minimizing the number of attempts needed to solve practice problems. In both the versions, the practice problems' outcome vectors are now composed of values representing number of attempts (f_P) rather than binary outcomes. When evaluating learners in **P-PHC-I**, we averaged the difference, in the number of attempts, induced by a given learner between each pairs of practice problems.

Results

As in our previous experiments, we used *objective fitnesses* to track improvements in quality. For both practice problems and learners, we simply summed all the genes. Given how problems' genotypes are used to compute the outcome of the interaction with learners, the larger this sum, the better. We therefore expected the system to converge to the individual with maximal allowed value on each of its genes. Similarly, we expected learners to converge toward an optimal learner solving problems in as few attempts as possible, thus minimizing the value of all the genes.

Please note that, for the learners, this metric is only relevant when considering the performance-driven versions of the learners' fitness; i.e. **P-PHC-P** and **P-CHC**. In the informativeness-based version, **P-PHC-I**, it would not make sense to track the sum of genes for learners as the selective pressure applied to their population does not encourage them to minimize their number of attempts but rather to help identify good from bad practice problems. For this reason, such results are omitted in table 2 which focuses solely on tracking improvement of practice problems. We also opted to focus on comparing **P-CHC** with **P-PHC-I** only based on previous results that strongly suggested the Informative variant to be much more beneficial (Gaspar et al. 2016).

Implications Regarding Design Guidelines

For each of our experiments, we will interpret and summarize the results from the perspective of their significance in terms of how we should design an evolutionary approach to practice problems generation. Keep in mind that such interpretation will remain, by necessity, at a certain level of abstraction in so far that it is meant to be applicable to a wide range of EAs and any specific learning domain for which the practice problems may be targeted; e.g. discrete mathematics, programming...

The results presented in row ”RL #8” from Table 2 suggests that our target application can benefit from Pareto coevolution when $N_G = 100$. Equation 1 defines an interaction model between learners and practice problems. Both the mean fitness and dispersion metrics show that **P-PHC-I** outperforms **P-CHC** in a statistically significant manner.

However, when bounding the genes values at $N_G = 10$, these benefits vanish, as shown on row ”RL #7” in Table 2, where the mean values are very close but the dispersion is significantly higher for **P-PHC-I**. This observation motivated us to take a closer look at the nature of the interactions taken place in **P-PHC-I**.

Analysis of the interaction in Experiment #1

We propose to revisit previous section's findings in order to identify the dynamics responsible for the results detailed in Table 2. To this end, we identify and quantify three properties that explain the algorithms' behavior for trivial and Pareto-based coevolution of practice problems.

Property 1 - Mutation Effect

In both the **PHC** and **CHC** algorithms, each parent practice problem p undergoes mutations in order to generate a child practice problem c. The mutation operator increases or decreases the value of each gene by 1 with equal probability.

However, whether the fitness of a child practice problem (F_c) becomes higher than its parent's (F_p) is based on their respective sum of genes value, along with that of the $rand(r)$ term (see Equation 1). As both p and c interact with same set of learners within a generation, S^L is constant across interactions, therefore we focus on S^P for both the parent and child, which we will note as S_p and S_c for short.

Let us first inspect the probability that $S_c > S_p$ due to mutation assuming N as genotype size;

$$P^{\mu}_{S_c>S_p} = \left(\sum_{i=\lceil N/2+0.5\rceil}^{N} \binom{N}{i}\right)/2^N$$

If N is odd then $P^{\mu}_{S_c>S_p} = P^{\mu}_{S_c<S_p} = 0.5$ and $P^{\mu}_{S_c=S_p} = 0$, otherwise $P^{\mu}_{S_c>S_p} = P^{\mu}_{S_c<S_p} = 0.3125$ and $P^{\mu}_{S_c=S_p} = 1 - (P^{\mu}_{S_c>S_p} + P^{\mu}_{S_c<S_p}) = 0.375$.

Proof: Probability Mass Function - PMF -, of applying ”+1” operation out of ”+1” and ”-1” operations on a genome of N size, of a random variable $X = x \ where \ 1 \leq x \leq N$ is $p(x) = \binom{N}{x}/2^N$. To satisfy the condition, $P^{\mu}_{S_c>S_p}$, ”+1” operations need to be applied on $(N/2+1)^{th}$ to N^{th} genes on p's genotype. So, the Cumulative Distribution Function - CDF -, such that $P^{\mu}_{S_c>S_p}$ is $P(X > N/2)$.

$$P^{\mu}_{S_c>S_p} = \left(\sum_{i=\lceil N/2+0.5\rceil}^{N} \binom{N}{i}\right)/2^N$$

Explanation: To satisfy $S_c > S_p$, the frequency of ”+1” mutation operation need to exceed than that of ”-1” operation. A parent p, with even genome size, can have half of its genes increased by 1 and the other half decreased by 1, thus producing a child c such that $S_p = S_c$. However, if the genome size is odd, there is no chance that $S_p = S_c$.

In the current settings, if $S_c > S_p$ then the former may exceed the latter by either $\delta^{\mu}_{min} = 2$ or by $\delta^{\mu}_{max} = 4$. So, based on the effects of mutations, the difference between parent and child is $\delta = \{\delta^{\mu}_{min}, \delta^{\mu}_{max}\}$.

Property 2 - rand(r) Effect

As mentioned earlier, the probability that $F_c > F_p$ is also affected by the $rand(r)$ term found in Equation 1. We therefore quantify the impact of this random term on the probability that the child practice problem feature a higher fitness than its parent.

First, PMF of having a child c receives a greater (smaller) random value than its parent p is based on $X = x\ where\ 1 \leq x \leq r-1$ is $p_1(x) = P(X = x) = \frac{r-x}{r^2}$. Consequently, the CDF of X is $F_1(x) = P(X \leq x) = \sum_{x=1}^{x} P(X = x) = \frac{x}{r} - \frac{x(x+1)}{2r^2}, r \neq 1$.

The PMF of both parent and child receiving an equal random number $Y = y\ where\ 1 \leq y \leq r-1$ is $P(Y = y) = \frac{1}{r^2}$. Consequently, the CDF of Y is $P(Y \leq y) = \sum_{r=1}^{r} P(Y = y) = \frac{1}{r}$

The probability of a child c getting a random value larger or equal than that of its parent p is therefore $P_{R_c \geq R_p}^{rand} = \frac{1}{2} + \frac{1}{2r}$, where $r = 2, 3, ...$

Proof :

$$F_1(x) = \sum_{n=1}^{x} \frac{r-n}{r^2} = \frac{x}{r} - \frac{x(x+1)}{2r^2}$$

$$P_{R_c \geq R_p}^{rand} = \sum_{i=1}^{r} \frac{1}{r^2} + F_1(r-1) = \frac{1}{2} + \frac{1}{2r}$$

Explanation: $rand(r)$ returns an integer in $[0..r-1]$. So, it can be seen as picking a pair (p_a, p_b) from r^2 pairs where r of them satisfy p_a = p_b, $\frac{r^2-r}{2}$ pairs follow $p_a > p_b$ and rest of the $\frac{r^2-r}{2}$ are obliged by $p_a < p_b$.

As we did with the δ values for the *Mutation Effect*, we define $\delta_{min}^{rand} = 1$ and $\delta_{max}^{rand} = r - 1$. In addition, $\delta_{=}^{rand}$ corresponds to $p_a = p_b$. So, when $S_c > S_p$ the difference is bounded by $\delta_{min}^{\mu} + \delta_{=}^{rand} = 2 + 0 = 2 \leq \delta_{\mu}^{rand} \leq \delta_{max}^{\mu} + \delta_{max}^{rand} = 4 + (r-1) = 3 + r = 5$.

Property 3 - Combined Effect

We have so far examined the probability of both the mutation operator, and the random noise, to contribute separately. In this section, we devote our attention to the combination of both effects on the relation between F_c and F_p. A child c Pareto-dominates its parent p based on how much it balances the gain or loss from the *Mutation Effect* by that of the $rand(r)$*Effect*. For instance, if $S_c > S_p$ by δ after the *Mutation Effect*, then p needs to get an equal or smaller value by at most $\delta - 1$ in order for $F_c > F_p$ to hold.

Let us start by defining three outcomes, *win, loss* and *draw* for the combination of both effects. We say that *c wins against p*, i.e. we have a *c win* when $S_c > S_p$ after *Mutation Effect* or $R_c > R_p$ in $rand(r)$*Effect*. Similarly, a *c loss* is defined for the ”<” relationship and the outcome is termed as *draw* for the ”=” relationship.

When combining the effect of the mutation and random term, these ”*win/loss/draw*” outcomes affect the values of F_c and F_p which, in turn, determine whether the child practice problem is ”strictly better” than its parent. Therefore, the combined effect of Property 1 and Property 2 need to be examined with respect to the cases listed in Table 1 for $r = 2$. These actually determine the three relations ,”>”, ”<” and ”=”, between F_c and F_p assuming both of them interact with the same learner. There are four cases where $F_c > F_p$, four cases for $F_c < F_p$ and one case for $F_c = F_p$. The probabilities listed in the ”Combined” column are obtained by multiplying the probabilities of the two independent effects it combines. Assuming the total probability for $F_c > F_p$ to be $P_{F_c > F_p}$. Then, for $1 \leq k_1 \leq r-1$; $P_{F_c > F_p} \geq 0.3125 \times P_{R_c \geq R_p}^{rand} + 0.6875 \times F_1(k_1)$

Similarly, for $k_1, k_2 \in \delta$; $P_{F_c = F_p} = \sum_{i=1}^{r} \frac{1}{r^2} - \frac{k_1 + k_2}{r^2}$ and $P_{F_c < F_p} = 1 - (P_{F_c > F_p} + P_{F_c = F_p})$

Proof : We can derive the following equation for $P_{F_c > F_p}$ using all the five cases of $F_c > F_p$ listed in Table 1.

$$\begin{aligned} P_{F_c > F_p} &= 0.3125 \times F_1(k) + 0.3125 \times F_1(k_1) \\ &= 0.15625 + 0.15625 \times \frac{1}{r} + 0.6875 \times F_1(k_1) \end{aligned}$$

To satisfy $F_c = F_p$, the three conditions listed in Table 1 can be summarized as follows for $k_3 = p1(k) \times k + p2 \times k_2$. $P_{F_c = F_p} = \sum_{i=1}^{r} \frac{1}{r^2} - 0.3125 \times \frac{k_3}{r^2}$

Implications regarding Design Guidelines

The analysis of Exp#1 results presented in this section suggested that having a high number of objectives causes problems to our algorithms. This finding is aligned with the literature on evolutionary multi-objectives optimization - EMOO - where problems with more than about five objectives are much more difficult to tackle by state of the art algorithms (He and Yen 2016). This led to the definition of many-objectives optimization as a field of study of its own, dedicated to investigate solution to EMOO problems featuring a non-trivial number of objectives.

In terms of our target application, these findings suggest that, rather than attempting to evaluate every evolved practice problem on as many students as feasible, we should instead restrict the number of students exposed to a given problem. Coupled with the need to mitigate user-fatigue, this means that the policy assigning each evolved practice problem to a ”suitable” learner may be very selective and still benefit the overall dynamics of our evolutionary system.

Experiment #2 - Genes’ Bounds

Problem

We adapt COMPARE-ON-ONE to the *practice problem - learner* interaction defined in Equation 2 as follows;

$$G_{one}(P, L) = \begin{cases} +1 & \text{if } P_m \geq L_m \\ -1 & \text{otherwise} \end{cases} \text{where}, m = \text{arg max } L_i \tag{2}$$

where L is learner, P is practice problem and x_i denotes the value of individual x in dimension i.

Algorithms

We measure performance of **P-CHC**, **P-PHC-P** and **P-PHC-I**, for two different payoff functions defined in Equations 1 and 2 while the genes of entities are bounded in $[1, N_G]$ for $N_G = 10$ and $N_G = 100$. It is worth pointing out that this is a significant departure from the number games commonly used in the literature, e.g. (Bucci 2007) or (De Jong and Pollack 2004). The latter does not limit the values taken by a gene.

Table 1: Different cases and status for combined effect in three relational comparisons between c and p, $r = 2$. Z_p^c refers to an outcome under condition c with p probability. W, L and D denote *win, loss and draw* respectively. Prop#1, Prop#2 and Both stand for mutation, random and combined effect respectively.

$F_c > F_p$				$F_c < F_p$				$F_c = F_p$			
Prop#1	Prop#2	Both	Status	Prop#1	Prop#2	Both	Status	Prop#1	Prop#2	Both	Status
$W_{0.31}^{\delta}$	$L_{0.25}^{1\leq k<\delta}$	$W_{0.08}^{\delta-k}$	OK	$W_{0.31}^{\delta}$	$L_{0}^{k>\delta}$	$L_{0}^{k-\delta}$	NA				
$W_{0.31}^{\delta}$	$W_{0.25}^{1\leq k\leq r-1}$	$W_{0.08}^{\delta+k}$	OK	$L_{0.31}^{\delta}$	$W_{0.25}^{k<\delta}$	$L_{0.08}^{\delta-k}$	OK	$W_{0.31}^{\delta}$	L_{0}^{δ}	D_{0}	NA
$W_{0.31}^{\delta}$	$D_{0.50}$	$W_{0.16}^{\delta}$	OK	$L_{0.31}^{\delta}$	$D_{0.50}$	$L_{0.16}^{\delta}$	OK	$D_{0.37}$	$D_{0.50}$	$D_{0.18}$	OK
$D_{0.37}$	$W_{0.25}^{1\leq k\leq r-1}$	$W_{0.09}^{k}$	OK	$L_{0.31}^{\delta}$	$L_{0.25}^{1\leq k\leq r-1}$	$L_{0.08}^{k+\delta}$	OK	$L_{0.31}^{\delta}$	W_{0}^{δ}	D_{0}	OK
$L_{0.31}^{\delta}$	$W_{0}^{k>\delta}$	$W_{0}^{k-\delta}$	NA	$D_{0.37}$	$L_{0.25}^{1\leq k\leq r-1}$	$L_{0.09}^{k}$	OK				
Total		0.41				0.41				0.18	

In our target application, genes represent selections of specific characteristics or components of practice problems and hence be necessarily bounded in value. Therefore, it is particularly relevant to investigate further whether differences in N_G impact the need for us to rely on Pareto-based coevolutionary algorithms as opposed to traditional ones.

Results

Table 2, "RL #1-4" show the performance of **P-CHC** and **P-PHC-I** under two payoff functions. Let us label Equation 1a as the version of Equation 1 in which the $rand(r)$ term is discarded .

The result for COMPARE-ON-ONE indicates that when we use a low bound value for genes, we do not need **P-PHC** abilities to overcome overspecialization.

Implications regarding Design Guidelines

In terms of design guidelines, if overspecialization is possible then Pareto coevolution of practice problem based on learner's informativeness is preferable. Algorithms, such as **P-PHC-I** indeed prevent learners from overspecializing on some aspects of practice problems at the detriment of increasing their skills across all learning objectives.

Overall, these findings suggest that recent breakthrough in coevolutionary computation theory (Bucci 2007) is applicable to our target application; i.e. competitive coevolution in a teacher-learner scenario may be improved by prioritizing informativeness in one of the populations.

In addition, the target application should be capable to keep practice problem's genotype intact but using recycled gene value while building phenotype from that genotype.

Experiment #3 - Noise & Genes' Bounds

Problems & Algorithms

In this experiment, we use both Equation 1 and a noisy version of Equation 2 in which the outcome is flipped ($+1$ by -1 or vice et versa) with a 5% probability.

We measure the performance of the same algorithms than in our previous experiments on both problems. The genes values are bound in $[1, N_G]$ for $N_G = 10$ and $N_G = 100$.

Results

Table 2, "RL #5-8", show the performance of **P-CHC** and **P-PHC-I** under noisy environment.

Results suggest that Pareto coevolution is still better at getting rid of overspecialization for the noisy COMPARE-ON-ONE. It is also better when genes can take more values.

Implications regarding Design Guidelines

If learner vs. practice problems interactions may yield overspecialization, then it is preferable to rely on a Pareto-based coevolutionary algorithm, even in a noisy environment, regardless gene's bound range. On the other hand, smaller bound of gene values in proposed interaction model is expected to benefit from trivial coevolution. No matter which fitness function we use, bounding the gene in upper value is expected to produce practice problems that will refrain the learners to be expert in only one learning objective.

Conclusion & Future Work

This paper allowed us to extend previous work focused on identifying design guidelines for leveraging evolutionary algorithms to generate practice problems.

The first, Experiment #1 confirmed previous results (Gaspar et al. 2016) with a modified model of our target problem that enabled us to conduct a more thorough analysis of the intrinsic properties of **P-PHC**. The combined effects of the mutation operator and the random term integrated in our fitness function revealed that, in accordance with the evolutionary multi-objective optimization literature, the larger the number of objectives in our problems, the more difficult it is for our algorithms to achieve decent performance. With respect to our target application, this suggests that not only using only a few students to evaluate each practice problem may be necessary to mitigate user fatigue, but it might also be beneficial to achieve productive coevolutionary dynamics. Furthermore, the previous results regarding the benefits of informativeness over performance in driving coevolution (Gaspar et al. 2016) mean that we already have a good candidate as criterion to select the students to use for evaluations.

However, while the results highlighted the suitability of Pareto-based Coevolutionary techniques to our target problem, they also revealed the unexpected relevance of the number of values that each gene may take in our encoding. We

Table 2: Performance of **P-CHC** and **P-PHC-I** under Pathology(overspecialization), noise and different bounds on two Pay off functions defined in Equations 1 and 2.

			Mean Objective Fitness			**Mean Dispersion**			
Property	**Fitness**	**Bounded**	**P-CHC**	**P-PHCI**	p	**P-CHC**	**P-PHC-I**	p	**RL**
Pathology	Eq 2	$[1, 10]$	35.99	35.41	> 0.05	7.13	7.10	> 0.05	1
		$[1, 100]$	67.60	45.52	< 0.01	31.42	7.51	< 0.01	2
	Eq 1a	$[1, 10]$	38.50	38.47	> 0.05	3.66	3.71	> 0.05	3
		$[1, 100]$	82.67	81.70	> 0.05	12.05	11.86	> 0.05	4
Noise	Eq 2a	$[1, 10]$	36.75	35.18	< 0.01	7.20	6.89	< 0.01	5
		$[1, 100]$	63.94	38.60	< 0.01	20.13	7.65	< 0.01	6
	Eq 1	$[1, 10]$	38.72	38.47	< 0.01	1.60	3.72	< 0.01	7
		$[1, 100]$	82.19	83.50	> 0.05	15.16	11.96	< 0.01	8

investigated only two extreme values so far, $N_G = 10$ and $N_G = 100$, but also considered a classic coevolutionary number game, COMPARE-ON-ONE. The latter allowed us to compare the results obtained with our fitness function against a baseline for which we know that overspecialization is encouraged by the environment. Experiments #2 and #3 suggest that, regardless of whether overspecialization is likely to occur, a low value for N_G means that we may achieve comparable or even better performance by adopting a traditional coevolution approach, e.g. **P-CHC**, rather than a Pareto-based one, e.g. **P-PHC**.

A priority for our future work will therefore be to quantify the minimal such value, everything else being equal, for which Pareto Coevolution shows benefits over traditional approaches like **CHC**. Last but not least, we will apply the design guidelines we gathered so far to conduct a preliminary evaluation of our proof of concept implementation software with real students in order to validate the proposed guidelines, and thus gain insights on which coevolutionary pathologies are most pronounced in this specific application.

Acknowledgments

This material is based in part upon work supported by NSF (awards #1504634, #1502564, and #1504634) and by the ACM SIGCSE Special Projects 2015 award.

References

Bader-Natal, A. 2008. *The Teacher's Dilemma: A game-based approach for motivating appropriate challenge among peers*. Ph.D. Dissertation, Michtom School of Computer Science, Brandeis University.

Bucci, A. 2007. *Emergent Geometric Organization and Informative Dimensions in Coevolutionary Algorithms*. Ph.D. Dissertation, Brandeis University, Boston, MA.

Chen, C. M. 2008. Intelligent web-based learning system with personalized learning path guidance. *Computers & Education* 51:787–814.

De Jong, E. D., and Pollack, J. B. 2004. Ideal evaluation from coevolution. *Evolutionary Computation* 12(2):159–192.

Gaspar, A.; Bari, G.; Kumar, A. N.; Wiegand, R. P.; Bucci, A.; and Albert, J. L. 2016. Evolutionary practice problems generation: Problem characterization. In *28th IEEE ICTAI*.

He, Z., and Yen, G. G. 2016. Many-objective evolutionary algorithm: Objective space reduction and diversity improvement. *IEEE Trans. Evo. Comp.* 20(1):145–160.

Huang, M.-J.; Huang, H.-S.; and Chen, M.-Y. 2007. Constructing a personalized e-learning system based on genetic algorithm and case-based reasoning approach. *in Expert Syst Appl* 33:551–564.

Llorá, X.; Sastry, K.; Goldberg, D. E.; Gupta, A.; and Lakshmi, L. 2005. Combating user fatigue in igas: Partial ordering, support vector machines, and synthetic fitness. In *GECCO*, 1363–1370.

Parsons, D., and Haden, P. Parson's programming puzzles: A fun and effective learning tool for first programming courses. In *Proc. of 8th ACE, 2006 - Volume 52*, 157–163. Darlinghurst, Australia: Australian Computer Society, Inc.

Romeroa, C.; Gonzalez, P.; Ventura, S.; del Jesusb, M.; and Herrerac, F. 2009. Evolutionary algorithms for subgroup discovery in e-learning: A practical application using moodle data. *in Expert Syst Appl* 36:1632–1644.

Sklar, E., and Pollack, J. 1998. Toward a community of evolving learners. In *ICLS*.

Wiegand, R. P.; Bucci, A.; Kumar, A. N.; Albert, J. L.; and Gaspar, A. 2016. A data-driven analysis of informatively hard concepts in introductory programming. In *SIGCSE*.