Board 59: Coevolutionary-Aided Teaching: Leveraging the Links Between Coevolutionary and Educational Dynamics

Dr. Alessio Gaspar, University of South Florida

Dr. Alessio Gaspar is an Associate Professor with the University of South Florida's Department of Computer Science & Engineering and director of the USF Computing Education Research & Evolutionary Algorithm Laboratory. He received his Ph.D. in computer science in 2000 from the University of Nice Sophia-Antipolis (France). Before joining USF, he worked as visiting professor at the ESSI polytechnic and EIVL engineering schools (France) then as postdoctoral researcher at the University of Fribourg's Computer Science department (Switzerland). Dr. Gaspar is an ACM SIGCSE, SIGITE and SIGEVO member and regularly serves as reviewer for international journals & conferences and as panelist for various NSF programs. His research interests include Evolutionary Algorithms, Computing Education Research, and applications to Computer-Assisted Teaching & Learning. His technology interests include Linux System Administration, Programming, Web App Development, and open source technologies in general.

Mr. ATM Golam Bari, University of South Florida, Tampa

ATM Golam Bari, student member IEEE, is a Ph.D. student in Computer Science & Engineering Department at University of South Florida, USA. He received the ME and BSc. degree in Computer Science & Engineering from Kyung-Hee University, South Korea and Dhaka University, Bangladesh, in 2013 and 2007, respectively. His main research interest involves Coevolutionary Algorithms, Dynamic Optimization, Bio-data mining.

Dmytro Vitel

I was born in Ukraine, 1988. In 2011 I finished Taras Shevchenko National University of Kyiv and obtained degree Master of Science in Applied Physics. In August 2017, I was accepted into MSIT program at University of South Florida. Eventually, program was changed to MSCS.

Mr. Kok Cheng Tan, University of South Florida

Kok Cheng Tan is a present PhD student of Computer Science at University of South Florida. He tends to work toward data science fields such as machine learning and data mining. He has eight-year teaching experiences and interested in exploring academical present trends.

Dr. Jennifer Albert, The Citadel
Dr. Rudolf Paul Wiegand III, University of Central Florida

R. Paul Wiegand is an Associate Research Professor in the University of Central Florida's School of Modeling, Simulation, and Training. He directs the Advanced Research Computing Center at UCF and directs the Natural Computation & Coadaptive Systems lab. His research interests primarily focus on methods of natural computation, theory of coadaptive and coevolutionary computation, application of coadaptive methods for multiagent learning, and high performance / high throughput computing. More generally, he is interested in designing and applying effective learning algorithms and representations for generating and modeling robust heterogeneous, multiagent team behaviors.

Coevolutionary-Aided Teaching: Leveraging the Links Between Coevolutionary and Educational Dynamics

Alessio Gaspar
alessio@usf.edu

Bari A.T.M. Golam
bari@mail.usf.edu

Dmytro Vitel
dvitel@mail.usf.edu

Paul Wiegand
wiegand@ist.ucf.edu

Jennifer Albert
jennifer.albert@citadel.edu

Kok Cheng Tan
dvitel@mail.usf.edu

Abstract

This paper accompanies the poster presented at the IEEE ASEE 2019 conference's NSF grantees special poster session. Our goal is to provide the reader with an overview of the deliverable and findings that resulted from three collaborative National Science Foundation (NSF) awards from the Improving Undergraduate STEM Education (IUSE) program; #1504634, #1502564, and #1503834.

The so-called EvoParsons project (http://cereal.usf.edu/EvoParsons) resulted in a proof of concept educational software aimed at novice programmers. It applies coevolutionary computations theories and advances to both design Parsons puzzles for students, and enable a dimension based analysis of students vs. puzzles interaction logs.

1 Introduction

The primary inspiration of this work is to approach the learning dynamics occurring between students and practice problems from a domain-independent perspective that has also application in machine learning and co-optimization. More specifically, we leveraged recent theoretical advances in the field of *Coevolutionary Computation* to study student vs. practice problems interactions. The term Coevolutionary-Aided Teaching (CAT) was coined to describe this research agenda. At its core lies the study and application of coevolutionary dynamics to an educational problem-domain. It is worth mentioning that this work is also a significant departure from research focused on Intelligent Tutoring Systems (ITS). While ITS are designed to adapt practice problems to the performance and difficulties of each individual learner, a coevolutionary system opens new perspectives. In a CAT system a population of practice problems coevolve alongside a population of students which learn to overcome them. As such, each evolved practice problem is not meant to address the need of any single student. Instead, the population of problems evolves as a whole to capture the difficulties that are encountered by students, in general. This subtle difference might be explained further by looking at ITS as automated systems meant to teach each individual students, while CAT are automated systems meant to reveal insights about the difficulties encountered by students.

As such, we see Coevolutionary-Aided Teaching as an algorithmic way to support

Computing Education researchers in understanding the difficulties encountered by students. Such endeavor is, for instance, key to establishing concept inventories in our discipline.

However, it is also worth emphasizing that this work is, by its very nature, applicable across many disciplines and topics. This NSF IUSE project was only meant to serve as a proof of concepts for both feasibility and efficiency of the underlying techniques. So far, this research resulted in two distinct, yet complementary, outcomes; First, we applied a coevolutionary algorithm to generate pedagogically-sound practice problems for a population of learners. Second, we applied dimension extraction methods, previously used to study artificial coevolutionary dynamics, to analyze student vs. practice problems interactions.

The remainder of this paper will first introduce the reader to essential literature on the various findings we have been leveraging in this work at the intersection of both the Coevolutionary Computation and Computing Education research communities. We then provide a summary of our findings to date as well as an overview of the EvoParsons software that represents the proof of concept implementation for our representative problem. We hope that this work will motivate others to explore student vs. practice problem interactions from a coevolutionary perspective, at the intersection of these two research communities.

2 Background

EvoParsons brings together both the Coevolutionary Computation and Computing Education Research communities. As such, we review relevant literature from both fields.

2.1 Parsons puzzles

In order to demonstrate the feasibility of coevolving practice exercises against a population of students, we selected a specific representative problem. EvoParsons aims at evolving a type of practice problem meant for novice programmers: Parsons puzzles.

Parsons puzzles are a relatively new type of practice problem aimed at helping novice programmers that were originally introduced in 2006[1]. Each Parsons puzzle consists of a valid implementation of a program, accompanied by a plain-English description of the requirements it fulfills. For instance, a puzzle might be built based on a Java implementation of a program computing the Greatest Common Divisor of two integers, along with a plain English description of what it does. These valid programs are then broken down into fragments. Depending on the specific implementation, each fragment might be as simple as one line of source code. At the same time, a few of these fragments are selected, duplicated, and transformed into so-called distractor fragments that feature a syntactical or logical bug, e.g., forgetting a ';' or transforming a '$<$' into a '$<=$'. These distractors are then added to the list of valid fragments and shuffled together. The requirements are presented to the student, along with the shuffled list of both types of fragments. The learner is expected to identify the valid fragments and re-order them in order to reconstitute the original solution.

Since their original inception, several independent studies have repeatedly illustrated the benefits of Parsons puzzle; e.g.,[2,3,4,5,6] and[7]. Researchers have found that scores on Parsons puzzles correlate with scores on code-writing exercises; and students prefer to solve Parsons puzzles over other activities such as writing code[8,6].

This, along with the simplicity of validating student work and integrate these puzzles in software, made them particularly interesting candidates for our proof of concept implementation.

As such, they confirm that this specific type of practice problem is particularly suitable for this project.

2.2 Interactive Evolution

Evolutionary Algorithms - EAs - are metaheuristics loosely based on neo-Darwinian evolution theories. They have been successfully applied to a wide variety of optimization and machine learning problems.EAs encode potential solutions as data structures manipulated by evolutionary operators; mutation (which randomize a small portion of the encoding), and recombination (which exchanges portions of the encoding between two potential solutions). A fitness function returns, for each possible solution, a measure of its desirability. The EA then iteratively improves an initially random population of N potential solutions, by modifying them with evolutionary operators, evaluating their fitness, then selectively reproducing the highly fit ones into the next generation. This iterative process is applied until the population converges, the best fitness reaches a given threshold, or a number of generations has elapsed.

Since EvoParsons aims at evolving Parsons puzzles against a population of learners, the notion of fitness of any given puzzle will be derived from the outcomes of the interactions that occurred between this particular puzzle and a subset students. The first consequence of this observation is that our application is interactive in nature; i.e, human agents (students) are necessary to evaluate Parsons puzzles. Such evolutionary algorithms are generally referred to as Interactive Evolutionary Algorithms (IEA). IEAs have been successfully applied to the evolution of graphic art , graphical user interfaces, , or criminal caricatures. See[9,10] for illustrative examples. However, such applications are hindered by the *user fatigue phenomenon*, i.e. unreliable evaluations when human agents are presented with excessive numbers of specimens[11]. As such it is essential to minimize interactions[12,13].

2.3 Coevolutionary Pathologies & Algorithms

Coevolutionary algorithms are a variant of Evolutionary Algorithms in which the evaluation of candidate solutions is based on their interaction with another co-adapting population. Typical example of such competitive coevolution may involve a population of solutions being evolved in one population while a population of tests is coevolved in the other population. Candidate solutions are evaluated by being applied to all available tests, and scored on how well they performed. Similarly, tests can be evaluated based on this same interaction and rewarded if they fail many solutions. This kind of archetypal coevolutionary scenario is successful if an arms race dynamics develop, thus allowing each of the two population to improve the quality of its individuals by competing endlessly with the other one.

This kind of scenario is intuitively appealing as it suggest that a system like EvoParsons could coevolve a population of Parsons puzzles, while a population of students serve as tests. The idea of applying techniques inspired from the field of Coevolutionary Computation to have population of artificially evolved artifacts interact with human agents can be dated back to[14]. Shortly after, the foundations for the educational application of this idea were laid in Sklar's pioneering work[15,16]. This work should be credited for having laid the conceptual foundations of what we termed Coevolutionary-Aided Teaching in this project. It is indeed one of the very first application of coevolutionary dynamics to support learning in actual students. This work has also led to further explorations focused on an agent-centric model of the learning environment, combining both human and artificial agents[17,18,19].

This approach also led to the development of BEEWeb[20] which exemplifies the overlap between peer learning and coevolutionary dynamics and brought to attention the need to appropriately motivate how learners evaluate each others[21,22]. The reader is referred to[23] for a comprehensive treatment of the link between educational concerns and game theory.

2.4 Dimension Extraction Algorithms

The very idea of minimizing the number of necessary interactions between candidate solutions and tests in a coevolutionary problem is very appealing to our project. First, it inherently allows to mitigate the above-mentioned user fatigue phenomenon. Second, it also suggests that if an ideal evaluation set can be identified, it might hold some pedagogical insights about the nature of the difficulties encountered by students.

The challenge of defining hard and easy problems arises in the study of *co-optimization* problems and *coevolutionary* algorithms[24]. Co-optimization problems are distinguished from optimization problems by the presence of two or more types of entity (e.g., student or problem) that might vary and can interact with one another with a measurable outcome.

Interestingly, many early coevolution and co-optimization researchers anticipated that simple measures of performance over these outcomes would be sufficient to yield "arms race" dynamics in coevolutionary learning. When such dynamics occur, the population of learners and the population of tests challenge each other causing mutual improvements. However, while some results confirmed the potential of such dynamics[25], others provided indications that many other, less beneficial, coevolutionary dynamics often emerge instead[26]. It is only recently, by adopting more sophisticated notions, such as Pareto dominance when comparing the performance of two learners over more simple aggregated measure of the outcomes, that the coevolutionary learning community has established a successful theoretical framework. It follows that, in order to build successful learning gradients, one must focus on the *information* about how tests distinguish different types of learner performances.

The EvoParsons project features students and computer programming problems (i.e., Parsons puzzles); any student can attempt — interact with — any of the problems, and they receive a score. Therefore, the real-world domain of students solving problems resembles a co-optimization problem, and we believe the tools and techniques used to study those algorithms can also be fruitfully applied to this domain.

One foundational and relevant line of work in co-optimization and coevolutionary algorithms concerns *dimension extraction* methods[27,28,29,30]. Abstractly, this work hypothesizes that the information gleaned from interactions, for instance the scores of students solving problems, can be decomposed into a vector-space-like *coordinate system*. Within a coordinate system, there are potentially *multiple axes*, or *dimensions*, consisting of a linearly ordered subset of the set of all entities. Entities further along a given axis are "no worse than" those preceding it. Across two different axes, entities are incomparable to one another, in the sense that an entity on one axis will be better in some ways, but worse in other ways, than an entity on another axis. This method has been used for a number of purposes, including automatically identifying key conceptual tactics in the game of Nim[30].

3 Project's Outcomes

This section provides an overview of the findings that resulted from developing and studying the EvoParsons system, along with references to our team's publications.

3.1 Encoding Parsons Puzzles as Genotypes

Evolutionary and Coevolutionary Algorithms both operate on representations of candidate solutions that are referred to as ”individuals” or ”genotypes”. The first step in applying such an algorithm to a new problem domain usually consists in devising an encoding for genotypes that is compatible with the evolutionary operators. In EvoParsons, we used fixed-length integer vectors as genotypes, along with two hand-designed reference libraries; programs and transforms. A software component, the Broker, takes genotypes and uses these libraries to reconstruct the corresponding Parsons puzzle. The *programs library* stores implementations of Java programs, along with a text describing the requirements they fulfill. These are indexed by an integer and similar, in content, to exercises found in introductory programming textbooks. Similarly, the *transforms library* holds regular expressions designed to match specific programming statements (e.g., for loop, variables declarations...) and modify them in order to introduce a specific bug (e.g., syntactical error, off-by-one loop initialization...). As such, these transforms have the potential to each model bugs or misconceptions that we expect novice programmers to encounter. These transforms are applied to valid code fragments in order to generate their bugged counterparts; distractor fragments. As with the programs library, all transforms are also indexed by an integer value in a given range. Both libraries are designed by hand and made available to the Broker.

The coevolutionary algorithm evolves fixed-length integer vectors as genotypes. The first value is an index in the programs library, while the remaining values are indexes in the transforms library. Given a genotype, the Broker constructs the corresponding Parsons puzzle by first retrieving the corresponding program from its library and segmenting it into fragments (one per line). The remaining genotype values are used to retrieve the corresponding transforms, match them potentially to valid fragments that are then copied and modified. These distractors are then shuffled alongside the valid fragments.

This simple encoding is key to the ability of providing the EvoParsons system with potentially large catalogs of sample exercises (programs) along with transforms modelling bugs and misconception previously identified in the technical or educational literature. It is then up to the coevolutionary system to explore the space of all possible re-combinations into Parsons puzzles. This space can become quickly extremely large and it would be unthinkable to simply force students through all possible re-combinations in order to establish with certainty which are most beneficial to learners. Similarly, using an algorithm to do so has potential benefits in terms of identifying pedagogically-sound Parsons puzzles while avoiding any human bias or preconception. The reader is referred to the ”Humies” competition (http://www.human-competitive.org/) for examples of applications in which Evolutionary Computation techniques produced original designs rivalling human-engineered ones. It is motivating to try to leverage such automated innovation in the educational domain.

3.2 Pareto-Domination for Parsons Puzzles Coevolution

The second step in applying an evolutionary or coevolutionary algorithm to a new problem generally consists in defining its fitness function that evaluates candidate solutions. Fortunately, recent advances in Coevolutionary Computation provided us with the concept of Pareto-Dominance as a suitable way to guide the coevolution of Parsons puzzles.

Overspecialization is a pathology occurring in coevolutionary and multi-objectives optimization scenarios in which some candidate solutions are highly competitive despite only maximizing their fitness on a subset of all available objectives. This pathology has a direct

counterpart in educational settings, where learners might also improve on only a subset of learning objectives. Our experimental investigation thus started with a synthetic problem meant to help us select an algorithm suitable to handle overspecialization.

To this end, we used the **Focusing Game (FG)**, a number game commonly used in coevolutionary computation to study overspecialization. The interaction outcome of two individuals (i and j), each represented by a 2-tuple of natural numbers, is as follows:

$$FG(\langle i_1, j_1\rangle, \langle i_2, j_2\rangle) = \begin{cases} i_1 > i_2 & \text{if } i_2 > j_2 \\ j_1 > j_2 & \text{otherwise} \end{cases}$$

In other terms, we identify the dimension with the highest value in the second individual and compare it with the corresponding dimension in the first individual. The highest value wins. This design misleads individuals to overspecialize by maximizing a single dimension. While potentially competitive, such individuals fail to maximize both dimensions.

Our early experiments[31] confirmed existing theoretical findings[32] and helped us select the Population-based Pareto Hill Climber algorithm, **P-PHC** for EvoParsons. **P-PHC** relies on selecting better individuals based on the concept of Pareto-Dominance that identifies the best individual in a pair by comparing these individuals' outcome vectors. For a Parsons puzzle, the outcome vector would be built by tracking the outcomes of its interaction with students. If two puzzles interact with the same students, the Pareto-Dominant one is the one that features all favorable outcomes present in the other's outcome vector, plus at least one.

This early part of our work allowed us to establish the relevance of Pareto-dominance as a way to compare Parsons puzzles being coevolved against students). In addition to this first "design guideline", we also investigated the **sensitivity to noise levels** in the interaction outcomes determination process. We fully expect evaluations based on student vs. Parsons puzzles interaction to be noisy due to external factors; e.g., student answering a phone call while using the software. Results confirmed, on a variant of our synthetic problems, that Pareto-based coevolution was indeed also suitable for such scenarios. Similarly, recent breakthroughs[29] also motivated us to explore the potential benefits of **Prioritizing Interactions with Informative Students**. While it is unlikely that real students will strive to improve their informativeness rather than their performance, it is relevant for us to identify only use a subset of students that the system should interact with in order to maximize informativeness when evaluating Parsons puzzles. Our findings also suggested that, in terms of **user fatigue mitigation**, coevolving Parsons puzzles requires careful consideration as to what puzzle is dispatched to a specific student. This led us to integrate puzzle-dispatching policies[33] combining reinforcement learning with evolutionary techniques[12].

3.3 Synthetic EvoParsons Experiment

We then designed a simplistic synthetic teacher-learner model that still captured essential characteristics of our target application. We used fixed-length integer vectors as genotypes for both learners and Parsons puzzles; $\langle g_1, g_2, g_3, g_4\rangle$ with each of the 4 genes taking value in $[1..N_G]$. The student vs. puzzle interaction was approximated by simply summing the values in a learner's genotype (S^L) and summing those of the practice problem (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 Parsons puzzle with a specific learner is the number of attempts taken by that learner to solve it; $N = S^L + S^P + \text{rand}(r)$ where $\text{rand}(r)$ returns a random integer in $[0 : r - 1]$, thus capturing the variability of students' performance. Both the learner and Parsons puzzle fitnesses are respectively derived from this quantity; $F^L = -N \quad \text{and} \quad F^P = N$. Therefore, these fitness measures are opposite for learners and practice problems while both revolving around the concept of difficulty. The number of attempts necessary for a learner to solve a puzzle is the fitness of the latter; higher values denote a difficult puzzle, lower ones denote a good student. This problem is meant to be the simplest to enable a coevolutionary interaction on genotypes very similar to EvoParsons'.

Experimental results[33] confirmed the design guidelines that we established on number games[31]. Further design guidelines were inferred, implying that using only a few students to evaluate each puzzle may not only mitigate user fatigue but also enable a more productive coevolutionary dynamics. If selecting students used for evaluating puzzles is critical, then our previous results on the benefits of informativeness measures over performance measures in driving coevolution[31] already suggested a natural criterion for doing so.

Last but not least, we also investigated the impact of various Parsons puzzles selection policies (all still based on the concept of Pareto-dominance) on the coevolutionary performance. Our findings[34,35] revealed the potential benefits of using alternative selection mechanisms that do by not solely favor Pareto-dominating puzzles, but also give a higher survival chance to those who are not comparable. The results provided us with insights on how to further tune **P-PHC** in order to improve its performance without having to expose students to an early prototype just about yet. In both[35] and then later[36], we also emphasized the development of suitable metrics allowing us to quantify the quality of the evolved Parsons puzzles from a informativeness perspective with respect to the underlying structure of the student vs. puzzles interaction space. This part of our work directly benefited from other research that was led in parallel and will be presented in the last subsection.

3.4 Applying EvoParsons to Actual Students

In order to validate the insights we gained, by using synthetic problems, about the characteristics of the problem consisting in coevolving Parsons puzzles, we integrated the resulting algorithm in the EvoParsons software and used it with actual students. In[37], we linked the previous work to three experiments that were conducted on Information Technology students enrolled in an online offering of an introductory programming course during Spring 2017 at the University of South Florida (USF, COP2512 Programming Fundamentals for IT). We ran two experiments over the course of the semester.

Experiment #1 was conducted at the beginning of the semester, after students were exposed to basic Java concepts: data types, selection and iteration. During this experiment, students were assigned to use our software and work on evolved Parsons puzzles for a minimum of 30 minutes, as practice. A total of 107 students participated in this first experiment. The Broker had 38 items in its distractors library and 40 Java programs in its programs library. The genotypes were set to a length of 10 and the population size to 10 genotypes. The programs library covered three Java topics that had been presented early in the course; data types, selection, and iteration. P-PHC-P ran for a total of six generations as students worked on their assignments and explored 79 unique genotypes.

Experiment #2 was conducted at the end of the semester. At that point in time, students had been exposed to more Java and were therefore more experienced. In this second

experiment, Parsons puzzles were not evolved. Instead, we used the 10 puzzles evolved by P-PHC-P during the first experiment and required participating students to work on all of them. This allowed us to obtain a full interaction matrix featuring most of the students and all 10 Parsons puzzles. CDE was then applied to that full interaction matrix as a way of visualizing the underlying interactions between puzzles and students. This also helped evaluate the quality of the last generation's evolved puzzles, as previously discussed.

In both experiments, students were given a week to work on their assigned Parsons puzzles, then received participation points for doing so, regardless of performance. Experimental results confirmed that the Pareto-based coevolutionary approach in general, and our design guidelines in particular, showed great promise as a viable solution to the problem of evolving practice problems for novice programmers. We also carefully examined the ten Parsons puzzles that resulted from the coevolutionary process. As these were intrinsically characterized by high informativeness in terms of differentiating students' performances, we reviewed the selection and mutation steps that led to them in order to gain insights about what bugs (as captured by the puzzles' distractors) exacerbated the differences in students' performance. This analysis suggested that the Pareto coevolution we used in this work may be further improved in terms of reducing any form of redundancy in the Parsons puzzles found in the last generation. As the student-puzzle interaction space is based on many dimensions, it is more beneficial for the algorithm to maintain populations that effectively sample as many dimensions as feasible. Our algorithm already inherently converged to the best representatives when exploring a given dimension, but it was not yet able to maximize the sampling of as many dimensions as its population size would theoretically allow.

3.5 Dimension Extraction Analysis of the Student-Puzzle Interaction Space

Along with our efforts to develop a coevolutionary algorithm suitable for the automated design of Parsons puzzles, our team also explored the potential of dimension extraction algorithms. In[38], and later[39], our team applied a well-known Dimension Extraction algorithm, **DECA**, to interaction logs captured between students and an Intelligent Tutoring Systems logs, **Problets**. When considering interactions of Parsons puzzles with students, the number of underlying dimensions extracted appeared related to the number of distinct *concepts* that the problem set featured; that is, the number of different ways that problems distinguish student performance. Likewise, the number of dimensions extracted when analyzing the interaction of students with Parsons puzzles provided an implicit measure of the number of distinct *types of learner* that appeared in the student set, or in other words the number of different ways students can be grouped in terms of differences in performance.

Additionally, such analyses highlighted the importance of how informative different problems are. It is simplistic to assign a "difficulty" measure to problems, based purely on how many students solve or do not solve a problem. For instance, if Garry Kasparov were to play a classroom of intermediate chess students, he would likely beat them all, and it would be impossible to determine which students were among the best in the class. Likewise, if someone who never played chess was used as the opponent, this person would likely lose to everyone, and we would be equally uninformed about the relative performance of students. Similarly, a problem that most students miss (or get right) may not be an informative because it does not provide distinctions to help educators know where students are having problems.

Instead, we want challenging problems that distinguish student performance as much as

possible. As educators, we are interested in constructing problems that lie in a *zone of proximal development* — problems of varying levels of difficulty that are on the boundary of where students perform in fundamentally different ways and, consequently, where true performance can best be judged. To begin to address this shortfall, we introduce two new terms grounded in dimension extraction analysis, *informatively hard* and *informatively easy*.

For each dimension extracted in a student analysis, i.e. type of learner, we identify the *most dominant student* as the one who solved the most problems among students on that dimension. We count each problem not passed by such students, for all of the most dominant students. Likewise, we identify the *most dominated student* as the one who solved the fewest problems among students on their dimension. We count each problem solved correctly by such students, for all of the most dominated students. Then, a problem is **informatively hard** if it is most often solved incorrectly by the most *dominant* students. Similarly, a problem is **informatively easy** if it is most often solved correctly by the most *dominated* students.

These two measures do not coincide with their traditional counterparts. For example, it is possible for a problem to be missed by everyone *but* the strongest students, for an otherwise challenging problem to be solved by the weakest students. When such things happen, it is likely that student behavior is confounding a linear expectation of performance, and is therefore interesting. Measures such as informatively easy and hard highlight and preserve the idea that different students understand different concepts in different ways. An informatively easy concept may be understood in the context of Zone of Proximal Development as the concept nearest to some other concept, whereas an informatively hard concept is farthest from any other concept. By contrast, we define a problem as (traditionally) **hard** if it is solved incorrectly by most students and as **easy** if it is solved correctly by most students.

Our results suggested that an informatively easy concept is rarely also easy, or an informatively hard concept also hard, although our intuitive notions of monotonicity of hardness would dictate this. Recall that informatively hard concepts are determined by the most dominant students vis-a-vis the general population, and informatively easy problems are determined by the most dominated students. The more interesting, although rare, cases are when an informatively easy concept is hard, and an informatively hard concept is easy. The former corresponds to when it is easy for the general population to make mistakes while solving a problem. The latter corresponds to when it is easy for the general population to guess the answer. Dimension Extraction thus provides a novel approach to analysing student vs. problems interactions that we so far applied to traditional intelligent tutoring system logs (Problets.org), and as a metric to validate the quality of evolved Parsons puzzles.

4 Discussion & Future Work

Both the dimension extraction analysis and the Parsons puzzles resulting from an EvoParsons run with a population of students hold the key to the difficulties through which these specific students struggled. As such, both components of this project are worth further investigating as first step toward assisting Computing Education researches in the design of concept inventories. Current methods are particularly time-consuming and still strongly dependent on expert consensus; e.g., reliance on Delphi panels[40]. As such, it is relevant to explore alternative approaches and evaluate carefully whether they provide comparable insights. Our latest ongoing work[41] has consisted in revisiting the Parsons puzzles that were coevolved against actual students by the EvoParsons software, and comparing them to both literature on concept inventories for introductory programming[42].

Given the potential of applying DECA analysis to educational interaction logs, we also investigated possibilities to further adapt this algorithm so as to make it usable in practice with the EvoParsons system. As we previously discussed, both the concept of Pareto-dominance and that of underlying dimensions of the students vs. puzzles interaction space, are already key to how EvoParsons coevolves its Parsons puzzles. However, being able to apply a DECA analysis, during evolution itself, and despite the fact that it is difficult to group students on the same puzzles without sacrificing the need to mitigate user fatigue, might be actually feasible. In[43] (submitted, currently under review), we applied machine learning techniques to extrapolate interaction data in synthetic problems in order to support efficient coevolution. Results are so far only preliminary but encouraging and will open possibilities for integration of new techniques in the EvoParsons software.

Acknowledgments

This material is based in part upon work supported by the Association for Computing Machinery's SIGCSE Special Projects 2015 award, and the National Science Foundation under awards #1504634, #1502564, and #1503834. Our team gratefully acknowledges the above funding support, as well as the contributions from dedicated faculty and students; Dr. Amruth Kumar, Dr. Anthony Bucci, Paul Burton, Stephen Kozakoff, and Himank Vats.

References

[1] Dale Parsons and Patricia Haden. Parson's programming puzzles: A fun and effective learning tool for first programming courses. In *Proceedings of the 8th Australasian Conference on Computing Education - Volume 52*, ACE '06, pages 157–163, Darlinghurst, Australia, Australia, 2006. Australian Computer Society, Inc. ISBN 1-920682-34-1.

[2] Paul Denny, Andrew Luxton-Reilly, and Beth Simon. Evaluating a new exam question: Parsons problems. In *Proceedings of the Fourth International Workshop on Computing Education Research*, ICER '08, pages 113–124, New York, NY, USA, 2008. ACM. ISBN 978-1-60558-216-0. doi: 10.1145/1404520.1404532.

[3] Petri Ihantola and Ville Karavirta. Two-dimensional parson's puzzles: The concept, tools, and first observations. *Journal of Information Technology Education*, 10(2):119–132, 2011.

[4] Juha Helminen, Petri Ihantola, Ville Karavirta, and Lauri Malmi. How do students solve parsons programming problems?: An analysis of interaction traces. In *Proceedings of the Ninth Annual International Conference on International Computing Education Research*, ICER '12, pages 119–126, New York, NY, USA, 2012. ACM. ISBN 978-1-4503-1604-0. doi: 10.1145/2361276.2361300.

[5] Barbara J. Ericson. Adaptive parsons problems with discourse rules. In *Proceedings of the Eleventh Annual International Conference on International Computing Education Research*, ICER '15, pages 259–260, New York, NY, USA, 2015. ACM. ISBN 978-1-4503-3630-7. doi: 10.1145/2787622.2787740.

[6] Barbara J. Ericson, James D. Foley, and Jochen Rick. Evaluating the efficiency and effectiveness of adaptive parsons problems. In *Proceedings of the 2018 ACM Conference on International Computing Education Research*, ICER '18, pages 60–68, New York, NY, USA, 2018. ACM. ISBN 978-1-4503-5628-2. doi: 10.1145/3230977.3231000.

[7] Briana B. Morrison, Lauren E. Margulieux, Barbara Ericson, and Mark Guzdial. Subgoals help students solve parsons problems. In *Proceedings of the 47th ACM Technical Symposium on*

Computing Science Education, SIGCSE '16, pages 42–47, New York, NY, USA, 2016. ACM. ISBN 978-1-4503-3685-7. doi: 10.1145/2839509.2844617.

[8] Barbara J. Ericson, Lauren E. Margulieux, and Jochen Rick. Solving parsons problems versus fixing and writing code. In *Proceedings of the 17th Koli Calling International Conference on Computing Education Research*, Koli Calling '17, pages 20–29, New York, NY, USA, 2017. ACM. ISBN 978-1-4503-5301-4. doi: 10.1145/3141880.3141895.

[9] Juan C. Quiroz, Sushil J. Louis, and Sergiu M. Dascalu. Interactive evolution of xul user interfaces. In *Proceedings of the 9th Annual Conference on Genetic and Evolutionary Computation*, GECCO '07, pages 2151–2158, New York, NY, USA, 2007. ACM. ISBN 978-1-59593-697-4. doi: 10.1145/1276958.1277373.

[10] Toshiaki Nakasu, Naiwala P. Chandrasiri, Takeshi Naemura, and Hiroshi Harashima. Nigao: Interactive facial caricature drawing system using genetic algorithm. In *ACM SIGGRAPH 2005 Posters*, SIGGRAPH '05, New York, NY, USA, 2005. ACM. doi: 10.1145/1186954.1186991.

[11] Xavier Llorá, Kumara Sastry, David E. Goldberg, Abhimanyu Gupta, and Lalitha Lakshmi. Combating user fatigue in igas: Partial ordering, support vector machines, and synthetic fitness. In *Proceedings of the 7th Annual Conference on Genetic and Evolutionary Computation*, GECCO '05, pages 1363–1370, New York, NY, USA, 2005. ACM. ISBN 1-59593-010-8. doi: 10.1145/1068009.1068228.

[12] Shimon Whiteson and Peter Stone. On-line evolutionary computation for reinforcement learning in stochastic domains. In *Proceedings of the 8th Annual Conference on Genetic and Evolutionary Computation*, GECCO '06, pages 1577–1584, New York, NY, USA, 2006. ACM. ISBN 1-59593-186-4. doi: 10.1145/1143997.1144252.

[13] Gregory S Hornby and Josh C Bongard. Accelerating human-computer collaborative search through learning comparative and predictive user models. In *Proceedings of the 14th annual conference on Genetic and evolutionary computation*, pages 225–232. ACM, 2012.

[14] P. Funes, E. Sklar, H. Juille, and J. Pollack. The internet as a virtual ecology: Coevolutionary arms races between human and artificial populations. Technical Report CS-97-197, Brandeis University Computer Science, 1997.

[15] E. Sklar, A.D. Blair, and J.B. Pollack. Co-evolutionary learning: Machines and humans schooling together. In *Workshop on Current Trends and Applications of Artificial Intelligence in Education: Fourth World Congress on Expert Systems*, 1998.

[16] E. Sklar and J.B. Pollack. Toward a community of evolving learners. In *Third International Conference on the Learning Sciences (ICLS-98)*, 1998.

[17] Elizabeth Sklar. Agents for education: When too much intelligence is a bad thing. In *Proceedings of the Second International Joint Conference on Autonomous Agents and Multiagent Systems*, AAMAS '03, pages 1118–1119, New York, NY, USA, 2003. ACM. ISBN 1-58113-683-8. doi: 10.1145/860575.860824.

[18] Elizabeth Sklar and Mathew Davies. Multiagent simulation of learning environments. In *Proceedings of the Fourth International Joint Conference on Autonomous Agents and Multiagent Systems*, AAMAS '05, pages 953–959, New York, NY, USA, 2005. ACM. ISBN 1-59593-093-0. doi: 10.1145/1082473.1082617.

[19] Elizabeth Sklar and Debbie Richards. The use of agents in human learning systems. In *Proceedings of the Fifth International Joint Conference on Autonomous Agents and Multiagent Systems*, AAMAS '06, pages 767–774, New York, NY, USA, 2006. ACM. ISBN 1-59593-303-4. doi: 10.1145/1160633.1160768.

[20] Ari Bader-Natal and Jordan Pollack. Motivating appropriate challenges in a reciprocal tutoring system. In *Proceedings of the 2005 Conference on Artificial Intelligence in Education: Supporting Learning Through Intelligent and Socially Informed Technology*, pages 49–56, Amsterdam, The Netherlands, The Netherlands, 2005. IOS Press. ISBN 1-58603-530-4.

[21] Ari Bader-Natal and Jordan B. Pollack. Beeweb: A multi-domain platform for reciprocal peer-driven tutoring systems. In *Proceedings of the 8th International Conference on Intelligent Tutoring Systems*, ITS'06, pages 698–700, Berlin, Heidelberg, 2006. Springer-Verlag. ISBN 3-540-35159-0, 978-3-540-35159-7. doi: 10.1007/11774303_72.

[22] Ari Bader-Natal and Jordan Pollack. Assessing learning in a peer-driven tutoring system. In *Proceedings of the 2007 Conference on Artificial Intelligence in Education: Building Technology Rich Learning Contexts That Work*, pages 530–532, Amsterdam, The Netherlands, The Netherlands, 2007. IOS Press. ISBN 978-1-58603-764-2.

[23] A. Bader-Natal. *The Teacher's Dilemma: A game-based approach for motivating appropriate challenge among peers*. PhD thesis, Michtom School of Computer Science, Brandeis University, 2008.

[24] E. Popovici, A. Bucci, R.P. Wiegand, and E.D. de Jong. Coevolutionary principles. In *Handbook of Natural Computing*, pages 987–1033. Springer, 2012.

[25] D. Hillis. Co-evolving parasites improve simulated evolution as an optimization procedure. *Artificial Life II*, 10:313–324, 1991.

[26] R. Watson and Jordan B. Pollack. Coevolutionary dynamics in a minimal substrate. In L. Spector and *et al*, editors, *Proceedings of the Genetic and Evolutionary Computation Conference, GECCO-2001*, pages 702–709. Morgan Kaufmann, 2001.

[27] A. Bucci, J.B. Pollack, and E.D. de Jong. Automated extraction of problem structure. In *Proceedings of the 7th Annual Conference on Genetic and Evolutionary Computation*, GECCO '04, pages 501–512, 2004.

[28] E.D. de Jong and A. Bucci. DECA: Dimension extracting coevolutionary algorithm. In *Proceedings of the 8th Annual Conference on Genetic and Evolutionary Computation*, GECCO '06, pages 313–320, New York, NY, USA, 2006. ACM.

[29] A. Bucci. *Emergent Geometric Organization and Informative Dimensions in Coevolutionary Algorithms*. PhD thesis, Brandeis University, Boston, MA, 2007.

[30] E.D. de Jong and A. Bucci. Objective set compression: Test-based problems and multi-objective optimization. In *Multi-Objective Problem Solving from Nature: From Concepts to Applications*. Springer, 2008.

[31] Alessio Gaspar, A.T.M. Golam Bari, Amruth N. Kumar, R. Paul Wiegand, Anthony Bucci, and Jennifer L. Albert. Evolutionary practice problems generation: Design guidelines. In *28th IEEE International Conference on Tools with Artificial Intelligence*, ICTAI'16, 2016.

[32] Anthony Bucci and Jordan B. Pollack. Focusing versus intransitivity geometrical aspects of co-evolution. In E. Cantú-Paz and *et al.*, editors, *Proceedings of the Genetic and Evolutionary Computation Conference, GECCO-2003*, pages 250–261. Springer, 2003.

[33] Alessio Gaspar, A.T.M. Golam Bari, Amruth N. Kumar, R. Paul Wiegand, Anthony Bucci, and Jennifer L. Albert. Evolutionary practice problems generation: More design guidelines. In *Proceedings of the* 30^{th} *International Conference of the Florida Artificial Intelligence Research Society*, FLAIRS'17, 2017.

[34] A.T.M. Golam Bari, Alessio Gaspar, R. Paul Wiegand, and Anthony Bucci. Does relaxing strict

acceptance condition improve test based pareto coevolution? In *Proceedings of the* 10^{th} *IEEE Symposium Series on Computational Intelligence*, SSCI'17, 2017.

[35] A.T.M. Golam Bari, Alessio Gaspar, R. Paul Wiegand, and Anthony Bucci. Selection method to relax strict acceptance condition in test-based coevolution. In *Proceedings of the IEEE Congress on Evolutionary Computation*, 2018.

[36] A.T.M. Golam Bari and Alessio Gaspar. Investigating relaxed selection in test-based pareto. In *Proceedings of the 21st International Conference on Computer and Information Technology*, Bangladesh, 21-23 December 2018. United International University.

[37] A.T.M. Golam Bari, Alessio Gaspar, R. Paul Wiegand, Jennifer L. Albert, Anthony Bucci, and Amruth N. Kumar. Evolutionary parsons puzzles: Design, implementation & preliminary evaluation. *Genetic Programming and Evolvable Machines*, 2018.

[38] R. Paul Wiegand, Anthony Bucci, Amruth N. Kumar, Jennifer L. Albert, and Alessio Gaspar. A data-driven analysis of informatively hard concepts in introductory programming. In *Proceedings of the 47th ACM Technical Symposium on Computing Science Education*, SIGCSE'16, pages 370–375, New York, NY, USA, 2016. ACM. ISBN 978-1-4503-3685-7. doi: 10.1145/2839509.2844629.

[39] Anthony Bucci, R. Paul Wiegand, Amruth N. Kumar, Jennifer L. Albert, and Alessio Gaspar. Dimension extraction analysis of student performance on problems. In *Proceedings of the* 29^{th} *International Conference of the Florida Artificial Intelligence Research Society*, FLAIRS'16, 2016.

[40] Holger Danielsiek, Ludger Humbert, and Jan Vahrenhold. Research-based learning revisited: On using a delphi process in informatics teacher education. In *Proceedings of the 6th International Conference on Informatics in Schools: Situation, Evolution, and Perspectives*, ISSEP'13, pages 196–208, Berlin, Heidelberg, 2013. Springer-Verlag. ISBN 978-3-642-36616-1. doi: 10.1007/978-3-642-36617-$8_1$7.

[41] A.T.M. Golam Bari, Alessio Gaspar, Vitel Dmytro, Paul Wiegand, Jennifer Albert, and Kok Cheng Tan. On the potential of evolved parsons puzzles to contribute into concept inventories in computer programming. In *Proceedings of the 126th American Society for Engineering Education Annual Conference and Exposition*, Tampa, Florida, June 15-19 2019. American Society for Engineering Education.

[42] Ricardo Caceffo, Raysa Benatti Guilherme Gama, and Rodolfo Azevedo Tales Aparecida, Tania Caldas. A concept inventory for cs1 introductory programming courses in c. In *In Technical Report 18-06, Institute of Computing, University of Campinas, SP, Brasil*, Brasil, 2018.

[43] A.T.M. Golam Bari, Paul Wiegand, Alessio Gaspar, and Anthony Bucci. Dominance relations of pareto coevolution:artificial to evoparsons's interaction. *IEEE Transactions on Evolutionary Computation*, 2019.