October 2019

Interactive Fitness Domains in Competitive Coevolutionary Algorithm

ATM Golam Bari
University of South Florida

Follow this and additional works at: https://digitalcommons.usf.edu/etd

Part of the Computer Sciences Commons

Scholar Commons Citation

This Dissertation is brought to you for free and open access by the USF Graduate Theses and Dissertations at Digital Commons @ University of South Florida. It has been accepted for inclusion in USF Tampa Graduate Theses and Dissertations by an authorized administrator of Digital Commons @ University of South Florida. For more information, please contact digitalcommons@usf.edu.
Interactive Fitness Domains in Competitive Coevolutionary Algorithm

by

ATM Golam Bari

A dissertation submitted in partial fulfillment of the requirements for the degree of Doctor of Philosophy
Department of Computer Science and Engineering
College of Engineering
University of South Florida

Major Professor: Alessio Gaspar, Ph.D.
Ali Yalcin, Ph.D.
Lu Lu, Ph.D.
R. Paul Wiegand, Ph.D.
Srinivas Katkoori, Ph.D.

Date of Approval:
September 20, 2019

Keywords: Interactive Evolutionary Algorithm, Optimization, Solution Concept, Pareto Dominance, Concept Inventory

Copyright © 2019, ATM Golam Bari
Dedication

I would like to dedicate my dissertation to my parents Mst Rawshan-ara Begum & Late Md. Yakub Ali; to my father in law Late Md. Reazul Haque and my loving wife Mst Rokeya Reaz.
Acknowledgments

I would like to express my heartiest gratitude to my supervisor, Dr. Alessio Gaspar for his never ending support and encouragement during my research period at USF. I would like to express my deepest appreciation to my committee members, Dr. Ali Yalcin, Dr. Lu Lu, Dr. R. Paul Wiegand, Dr. Srinivas Katkoori for providing their valuable insights regarding my research. I am also thankful to Dr. R. Paul Wiegand, Dr. Anthony Bucci, Dr. Jennifer L. Albert, Dr. Amruth Kumar for their feedback during my research paper writings. I would also like to thank my committee Chairperson Dr. John N Kuhn. I am thankful to the members of our CEREAL lab; Stephen Kozakoff, Kok Cheng Tan, Dmytro Vitel, Himank Vats, Paul Burton for providing feedback and discussion on my research. I am very thankful to Gabriel Franco, Laura Owczarek, Mayra Morfin, Kim Blumer, Lashanda Lightbourne who have always helped me without compliant even if I requested some academic administrative assistance at the very last minute. I am also thankful to Catherine Burton to provide feedback on the format of my dissertation. I am extremely grateful to my wife Rokeya Reaz whose encouragement, inspiration and caring support me during this journey. I am indebted to so many.

This dissertation 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. Any opinions, findings, and conclusions or recommendations expressed in this publication are those of the author(s) and do not necessarily reflect the views of the National Science Foundation.

Thank you.
Table of Contents

List of Tables vi

List of Figures viii

Abstract x

Chapter 1: Introduction 1
  1.1 Background 1
    1.1.1 Interactive Evolutionary Algorithms 3
    1.1.2 Competitive Coevolutionary Algorithms 4
    1.1.3 (Co)evolutionary Approaches in Educational Software 6
  1.2 Candidate Solution Evaluation 7
    1.2.1 Evaluation in Interactive Evolutionary Algorithms 8
    1.2.2 Evaluation in Competitive Coevolutionary Algorithms 8
  1.3 Challenges and Opportunities in Candidate Evaluation of Competitive Coevolutionary Algorithm 9
    1.3.1 Number Games 10
    1.3.2 Objective and Subjective Fitness 10
    1.3.3 Coevolutionary Pathologies 11
      1.3.3.1 Loss of Gradient 12
      1.3.3.2 Overspecialization, Also Known as Focusing 12
      1.3.3.3 Cycling, Also Known as Intransitivity 14
    1.3.4 Opportunities in Competitive Coevolutionary Algorithm 15
      1.3.4.1 Solution Concepts 15
      1.3.4.2 Ideal Evaluation 17
      1.3.4.3 Coevolutionary Dimension Extraction (CDE) 18
      1.3.4.4 Archiving 19
  1.3.5 Discussion on Challenges and Opportunities in CCoEA 19
  1.4 Challenges and Opportunities in Candidate Evaluation for Interactive Evolutionary Algorithm 21
    1.4.1 User Fatigue 21
    1.4.2 Techniques to Reduce User Fatigue 22
      1.4.2.1 Selecting Better Solutions 22
      1.4.2.2 Statistical Ranking of Solutions 23
      1.4.2.3 Evaluation of Solutions by Human Evaluator 24
2.8 Experiment #5 - Genes’ Bounds
  2.8.1 Problems
  2.8.2 Algorithms
  2.8.3 Results
  2.8.4 Implications Regarding Design Guidelines
2.9 Experiment #6 - Noise and Genes’ Bounds
  2.9.1 Problems and Algorithms
  2.9.2 Results
  2.9.3 Implications Regarding Design Guidelines
2.10 Guidelines for Evolving Practice Problem by Noisy Teacher-Learner Coevolution
2.11 Minimal Guidelines for Interactive Noisy Teacher-Learner CCoEA
  2.11.1 Measure the Progress in Coevolution
  2.11.2 Design Solution Concepts
  2.11.3 Consider the Role of Solution Concepts
  2.11.4 Determine Minimum Number of Evaluations
  2.11.5 Develop the Outcome of an Interaction
  2.11.6 Develop Techniques to Mitigate User Fatigue
  2.11.7 Focus on Noisy Evaluations
  2.11.8 Validate the Design Guidelines
  2.11.9 Validate the Evolution

Chapter 3: EvoParsons: Evolving Parsons Puzzles Based on Elementary Coevolution
3.1 Evolutionary Parsons Puzzle
3.2 Experiment #7 - Evaluating Evolved Practice Problems by Simulation
  3.2.1 Problems and Algorithms
    3.2.1.1 Evaluating Evolved Practice Problems on CDE Metrics
  3.2.2 Methods
  3.2.3 Results
  3.2.4 Implications Regarding Design Guidelines
3.3 Experiment #8 - Evaluating Evolved Puzzles in EvoParsons
  3.3.1 Problems and Algorithms
    3.3.1.1 Mapping Integer Vectors into Parsons Puzzles
    3.3.1.2 EvoParsons Architecture - EA, Broker and Learners
    3.3.1.3 User Fatigue, Sparse Interaction Matrix and Selection Policies
  3.3.2 EvoParsons Interaction with Real Human Students
    3.3.2.1 Quantitative Perspective
    3.3.2.2 Qualitative Perspective
  3.3.3 Design Guidelines Validation
3.4 Validations of EvoParsons
Chapter 4: Accelerating Progress in Elementary Coevolution

4.1 Need for Progress Acceleration 107
4.2 Definitions and Formalism 108
4.3 Experiments #9 Relaxing Selection Criteria in P-PHC-P 108
  4.3.1 Problems and Algorithms 109
    4.3.1.1 Base Selection in P-PHC-P 110
    4.3.1.2 Competitive Shared Fitness Based Relaxed Selection 111
  4.3.2 Methods 113
  4.3.3 Results 115
  4.3.4 Relaxation vs Convergence, Exploration and Monotonic Progress 115
  4.3.4.1 Convergence and Quality of the Solutions 115
    4.3.4.2 Relaxation vs Exploration 116
    4.3.4.3 Relaxation vs Monotonic Progress 117
4.4 Experiment #10 Pareto Front Based Relaxed Selection 118
  4.4.1 Problems and Algorithms 119
  4.4.2 Methods 121
  4.4.3 Results 121
4.5 Relaxation Based on Interaction Space of Pareto Coevolution 121
4.6 Experiment #11 - Stepping Stones of Interaction Space 123
  4.6.1 Interactive Domains and Interaction 124
  4.6.2 Interaction Space 124
    4.6.2.1 Properties of Interaction Space 126
    4.6.2.2 Traversal of Interaction Space and Selection Conditions 127
  4.6.3 Methods 127
  4.6.4 Results 128
  4.6.5 Observations on Interaction Space Based Relax Conditions 128
4.7 Experiment #12 - Proposed Relaxed Selection Conditions 129
  4.7.1 Selection Methods 129
  4.7.2 Problems and Algorithms 132
  4.7.3 Results 132
  4.7.4 Exploration vs Diversity in Relaxed Section Conditions 133
4.8 Exp#13: Progress vs Distribution of Candidate Solutions in Pareto Layers 134
  4.8.1 Problems and Algorithms 135
  4.8.2 Methods 135
  4.8.3 Results 135
List of Tables

Table 1.1 Design parameters along with challenges and opportunities for competitive coevolutionary algorithm based system. 33
Table 2.1 Comparison of overall mean best fitness, and dispersion, for P-CHC and P-PHC on focusing game 44
Table 2.2 Experiment #3 - Comparison of overall mean best fitness, and dispersion, between P-CHC and P-PHC on Teacher-Learner problem 49
Table 2.3 Performance of P-CHC and P-PHC-I under pathology (overspecialization), noise and different bounds on two pay off functions defined in Equations 2.1 and 1.1. 57
Table 2.4 Monitoring coevolution of practice problem for trivial and Pareto coevolution - P-CHC, P-PHC-P - under mutation, rand(r) and combined effect 67
Table 3.1 The quantitative performance of evolved puzzles in simulated EvoParsons averaged over 30 independent trials. 86
Table 3.2 Performance measures of P-PHC-P in experiment #1. 97
Table 3.3 CDE dimensions, singletons, extrema and duplicate puzzles for each of the matrices in both experiments. 99
Table 3.4 The four extrema found from CDE analysis on the sub-matrices from Run # 1. 100
Table 4.1 Performance of P-PHC-P, rP-PHC-P and fP-PHC-P for FG and Comp-on-one. 115
Table 4.2 The frequency based performance metrics for Pareto relationship for (parent, child) pair in P-PHC-P under FG. 116
Table 4.3 Significance tests for pairwise comparison, ↑ means significant, ↓ indicates failed to significance, \( p \leq 0.01 \). 117
Table 4.4 Interaction between candidates vs tests 124
Table 4.5 The upward, downward and horizontal movement probability of a child in P-PHC-P solutions under two number games. 128

Table 4.6 Selection conditions for respective methods. 129

Table 4.7 The mean rank of last generational solutions in underlying extracted dimension tree and mean duplicates among them 132

Table 4.8 Pairwise comparisons for Mean Performance Indicator. 133
List of Figures

Figure 1.1 An IEA system where user operates a face montage system [27] [115] to reconstruct a particular face being remembered by the witness. 5
Figure 1.2 Coordinate system for candidates obtained by CDE algorithm 18
Figure 1.3 Summary of number of publications and citations grouped by year using $Q_1$. 31
Figure 1.4 Word cloud formed by index terms of the papers retrieved using $Q_3$. 34
Figure 2.1 Experiment #2 - Mean overall best fitness and dispersion for focusing game with P-CHC and P-PHC under different noise levels 46
Figure 3.1 Distribution of DC, ARR. 87
Figure 3.2 Overall work flow of EvoParsons. 91
Figure 3.3 The distractor in subtraction quiz helps this practice problem to be more competitive i.e., Pareto dominate than its child Practice problem. 100
Figure 4.1 The performance of the candidate solutions of P-PHC-P and r-P-PHC-P in a typical run under FG number game. 118
Figure 4.2 The performance of the candidate solutions of P-PHC-P and r-P-PHC-P in a typical run under $G_{one}$ game. 119
Figure 4.3 Interaction space for $m = 4$ tests. 126
Figure 4.4 Performance indicator for different selection methods under FG game. 134
Figure 4.5 Monotonic progress of base selection. 136
Figure 4.6 Distribution of candidate solutions in Pareto front and last layers calculated from final generations over all the independent trials. 137
Figure 4.7 Different non dominated fronts in two objectives space for base and relaxed selection from an independent trial. 139
Figure 5.1 Structural and relational matrix, their respective damaged and difference matrices for the three number games.

Figure 5.2 MSE vs SSIM for structural and dominance relational matrices shown in first and second column respectively for \( d_r \) from 0% to 100%.

Figure 5.3 Distribution of \( a \sim b \) relations for comp-on-one in different damage ratio.

Figure 5.4 Damage ratio vs mean frequency of \( a \sim b \) for three number games.

Figure 5.5 Structural matrix is repaired using MLP assuming we have access to small number of interactions of a 100 × 100 interaction matrix.

Figure 5.6 Mean test accuracy of MLP for different number games.

Figure 5.7 The average proportions of relations of all four relations from different relational images obtained from interaction matrix in Experiment #1.

Figure 5.8 Distribution of relative moves for 10 puzzles over 61 students.

Figure 6.1 Decision between Pareto and Aggregation solution concepts under coevolutionary pathologies, noise and gene bounds.
Abstract

Evolutionary Algorithms (EA) have been successfully applied to a wide range of optimization and search problems where no mathematical model of the quality of a candidate solution is available. Interactive Evolutionary Algorithms (IEA) and Competitive Coevolutionary Algorithms (CCoEA) go one step further by being able to tackle problems where the only means to evaluate the quality of a candidate solution is via interactions. In a typical IEA, interactions take place between the solution being evolved and human evaluators. In a CCoEA, interactions take place between solutions themselves, without need for human interaction. This dissertation identifies computer-aided learning as an application domain which exemplifies the overlap of both fields. In particular, this work first develops a novel interactive and competitive (co)evolutionary approach to evolve candidate solutions. To do so, we identify viable algorithms, analyze them and author new variants of hill climber algorithms. Then, we design and implement a competitive coevolutionary interaction-based algorithm. The performance of the resulting heuristic is evaluated with respect to its ability to approximate a full Coevolutionary Dimension Extraction (CDE) process. This allows us to ensure that the proposed approach evolves candidate solutions that have pedagogically relevant in an educational application. However, the underlying hill climber algorithm produces some candidate solutions that exhibit the same interaction outcomes against opponent solutions. So, we also propose different approaches to improve the diversity of the solutions being evolved. To this end, we relax the strict acceptance condition in existing hill climbing algorithms relying on Pareto dominance. The proposed variant draw its inspiration from the Non-dominated Sorted Genetic Algorithm (NSGA), commonly used in evolutionary multi-
objectives optimization. We also introduce selection methods based on competitive shared
fitness, and the analysis of the interaction space among solutions. Finally, we study Pareto
dominance relations of coevolutionary interactions by looking at the interaction matrix of
both coevolutionary benchmarks and our educational application. This results in a unique
perspective to understanding both structural and relational dominance in coevolutionary
interactions. This method can be applied in any open-ended problems where the quality
of solutions can not be defined mathematically. It reveals the applicability of CDE, its
sensitivity to dominance relations, and its robustness to noisy outcomes.
Chapter 1: Introduction

1.1 Background

EAs are nature-inspired, population-based, stochastic search and optimization algorithms that leverage genetic-like concepts and operators in order to improve an initially random population of candidate solutions. As a result, the field of Evolutionary Computation (EC) often relies on terminology that has been borrowed from population genetics. Let us start by defining a few terms and concepts that will be used frequently in this paper[41];

- **Individual (also known as, Candidate Solution, Genotype)** - A possible solution to the problem being solved by the algorithm. Candidate solutions can be encoded as fixed-length integer vectors but other encoding are also explored.

- **Trait (aka, Gene)** - A specific feature of a given individual. For candidate solutions encoded as fixed-size integer vectors, a trait could be the particular value found at one of the vector’s positions.

- **Phenotype** - the observable characteristics of a solution. For instance, a fixed-length integer vector may be used to encode an image in PNG format. In such scenario, the vector represents the genotype encoding the candidate solution, while the image itself is its phenotype. If the goal is to evolve images that are favorably selected based on aesthetic criteria by human agents, then the phenotype is what holds the actual desirable features that are recognized during evaluation. The genotype is the encoding that is manipulated by the above-mentioned genetic operators during evolution. The
sketch of a human face in Figure 1.1 is an example of phenotype. The evolutionary algorithm may represent the genotype of this image as a fixed size of integer vector internally.

- **Population** - A set of individuals being evolved by the EA.
- **Fitness** - A measure of the quality of an individual with respect to the problem being solved. Generally, this value is computed by a so-called fitness function during the individual’s evaluation step. In this work, we more specifically consider problems in which individuals are evaluated by interactions with other individual(s) or human evaluator(s).
- **Genetic operators** - Algorithms used to create new individuals based on one or more so-called parent individuals. The two most commonly used operators are mutation (randomly changes a trait of an individual) and crossover (inherits traits from two parents and creates new individuals featuring the combined traits).
- **Selection** - Algorithm used to select individuals so as to allow the best ones to proliferate in the population over time.
- **Parent** - a solution that undergoes genetic operators.
- **Child** - a solution that is produced by applying genetic operators to its parent.
- **Generation** - Step of the EA where all the individuals in the population are evaluated, genetic operators applied, and individuals to be preserved in the next generation are selected.
- **Search Space** - The set of all possible solutions to the problem being considered. To take an example, if we are working with genotypes encoded as 3 integer values in the range $[1 : 10]$, then the complete search space is the set of all possible such vectors.
While the fitness function is computed directly from the solution when applying EAs to optimization problems, there are other applications of EAs in which the evaluation of a solution requires interactions. Evolution through interaction-based fitness is best suited for search or optimization problems in which a formal definition of the fitness of a solution is not available. Interactions may come in two types - phenotype vs. human (e.g., a human evaluator likes or dislikes pictures), and phenotype vs. phenotype (e.g., one phenotype’s fitness is based on how it performs with other phenotypes in the population).

The first type of interaction is the focus of IEAs. These focus on scenarios in which human agents evaluate the phenotypes and select them, or in which human evaluators participate even more actively to the optimization process by identifying traits of an individual that should no longer be modified by the genetic operators.

The second type of interaction is found in Coevolutionary Algorithms (CoEA). These algorithms do not require human evaluation but instead evaluate one phenotype by having them interact with other phenotype(s).

1.1.1 Interactive Evolutionary Algorithms

An IEA applies the same genetic operators, selection mechanisms, and genotype-level encodings than those found in a traditional EA. However, it is used to tackle optimization problems for which the fitness function can not be computed but, instead, requires human evaluators to somehow interact with the phenotype [125]. As such, an IEA may be described as an EA in which the fitness function is replaced by ratings from a human evaluator. There are two main ways an IEA allows a human agent to interact with phenotypes [115, 126, 125].

The first, and most straightforward way, consists in having the evaluator(s) directly assign fitness values to phenotypes. For example, human agent can provide directly a rating in the form of “acceptable” / “not acceptable” binary response. A Likert or numerical scale may be used instead in order to provide more nuanced feedback. Last but not least, the
evaluator may be presented a small set of candidate solutions and asked to identify the best one, or rank them in order of preference. Figure 1.1 shows an example of an IEA searching for a sketch image of a suspect matching a witness’ memory (our human evaluator in this problem) [27, 115]. In this case, a human evaluator provides a numerical or binary value indicating if the shown image is near to the suspect’s image.

The second technique goes further by allowing human evaluators to not only evaluate candidate solutions but also affect the application of genetic operators. In such case, the evaluator also has the ability to direct the evolutionary search process itself to focus on specific traits. Human evaluators thus drive the search by enabling or disabling genetic operators on parts of the genotypes. This boils down to guiding the underlying evolutionary dynamics by freezing (respectively unfreezing) traits that are found to be worth keeping (respectively reconsidering). In effect, this also narrows the evolutionary search process to human-specified areas of the search space. Human evaluator in the first technique now provides if certain parts (e.g., nose, eye, ear) of the shown image is similar to the suspects and direct evolutionary algorithm to lock/unlock that part for its search process.

There are many potential applications for IEAs. For examples, IEAs are applied into education [86, 78], industrial design [74, 89, 88, 53, 27] and musical composition [62, 14]. We refer the interested reader to Takagi et al. [125] for a thorough overview of IEA applications.

1.1.2 Competitive Coevolutionary Algorithms

Samuel’s ground-breaking work [112] in the late 1950’s illustrates the possibilities of leveraging coevolution as a form of machine learning that only relies on self-play to improve its abilities. In his work, the objective was for a computer to learn the game of checkers. Although this work did not use an evolutionary computation technique, as we would define it nowadays, it is recognized as the first attempt to embody founding elements of the coevolutionary algorithms that would follow.
Some of the notable milestones in the field of CoEA from the 1960’s to the late 1990’s include Barricelli’s use of CoEAs to learn strategies for the TacTix game [11], and Axelrod’s coevolutionary strategies to the Iterated Prisoner’s Dilemma (IPD) [4]. Two particularly notable applications of coevolution are those of Hillis [60] and Sims [117]. Hillis coevolved a population of sorting networks against a population of unsorted sequences. His work showed that coevolutionary methods are able to design a sorting network that used 61 comparators - just one more than the smallest sorting network known for input sequences of size 16. Both of the populations used in Hillis’ seminal paper were placed in a competitive relationship; the sorting networks were rewarded for properly sorting input sequences, while input sequences were rewarded for causing the sorting networks to sort them improperly. This competitive coevolutionary set up of two populations has become a standard framework known as “teacher-learner”, “host-parasite” or “test-based” coevolution.
A test-based problem is a co-search or co-optimization problem in which tests are evolved against candidate solutions [103]. The term was first introduced by De Jong and Pollack [34]. Later, “test-based” coevolution was found to be analog to multi-objective optimization by De Jong and Bucci [33] by simply considering that “tests” represent objectives for the solutions being evaluated against them. This relation between coevolution and multi-objectives optimization is implicit in early work on Pareto coevolution [94, 44] and is more explicit in later work [31]. Ficici [45] describes in details the connection between Pareto coevolution and multi-objective optimization.

1.1.3 (Co)evolutionary Approaches in Educational Software

Coevolutionary interactions are used for educational applications previously. The InOutMachine [119] is an educational software that uses evolutionary principles to teach arithmetic skills to students. Such equations start with an input (e.g., 3), apply a rule (e.g., x4) and end with a related output (e.g., 12). Each student is required to solve 10 tutors. The tutors are problem specific; represented by a 6-bit binary strings. The first two bits are for arithmetic operator (+, - or x) and the last four bits indicate numbers between 0 to 15. A student completes five problems generated using a specific tutor, and is then given another set of problems to complete. Once the students have solved all the tutors in a population, the next population is generated.

In the Community of Evolving Learners (CEL), Sklar et al., [120] built a freely available, web-based, learning community in which students engage in multiplayer games. The underlying system is able to provide continuous challenges for all students. In another work, Sklar et al., [120] use an evolutionary approach to implement two keyboard typing games; Keyit and PicKey. In these games, two players are each given 10 words to type and are scored based on typing accuracy and time. Words are then selected and clustered based on
7 features such as length, number of vowels, number of consonants etc. The EA is used to guide selection of words from that 7 dimensional feature space.

In [6], Bader et al., define the “Teacher’s dilemma” (TD); a criterion that motivates peers to challenge each other with problems of appropriate difficulty. This criterion introduces “appropriateness-dominance” and “effort-dominance”. The former is the dominant strategy for teachers that challenges the appropriate difficulty for the learners. Similarly, the latter is the dominant strategy for learners.

Based on the above ideas, the authors built a reciprocal tutorial network for the educational domain of spelling [7] - Spellbee. The application was designed for the students in grade 3-7 and takes the form of an educational activity. In this activity, each player alternates role between “problem-selector” (Teacher role in TD) and “problem-solver” (student role in TD). There are seven rounds in a game. In each round, a player first selects a word for their partner to spell. Each word has its point value based on TD teacher-matrix and TD student-matrix. After both players select words, the word-challenges are exchanged. After spelling the word, the student first gets feedback on his/her accuracy, then that of his/her partner. This concludes the current round and the game enters into next round.

Although it is not an evolutionary approach per se, a reciprocal tutorial network is similar to “teacher-learner” coevolutionary framework. Its goal goes beyond simply having learners improve and challenges get harder, but also motivates peers to appropriately challenge one another in order to reduce the “motivational gap” between an activity’s educational objectives.

1.2 Candidate Solution Evaluation

In section 5.1, we mention two ways of candidate solution evaluation in IEA and CCoEAs. This section describes them in detail.
1.2.1 Evaluation in Interactive Evolutionary Algorithms

An IEA replaces the fitness computation by interactions with human evaluators, thus allowing the latter to guide the selection process. In other variants, the human evaluator is also empowered to disable genetic operators on specific parts of the genotype, thus preventing the EA from potentially further altering traits that they deem worth keeping. Conversely, genetic operators may be later re-enabled on these same portions of the genotype if the human evaluator determines that there is a need to reconsider the corresponding traits. In such an IEA, the evaluators also exert control over the genetic recombination occurring at the genotypic level; as such, they become an even more integral part of the exploration process.

In the field of Interactive Evolutionary Multi Objective Optimization (EMOO) [17], the algorithm identifies multiple trade-off solutions which are then reviewed by an expert in order to determine which is the best. This is because, the definition of the “best” among multiple trade-off solutions may not exist under the considered problem or it may change from time to time, based on the subjective needs of the target system. In this case, the human evaluator acts as a “selector”.

1.2.2 Evaluation in Competitive Coevolutionary Algorithms

The phenotypes in CCoEA are evaluated by interacting with other phenotypes. In terms of multi-objective optimization, the interaction between “teachers and learners” happens in such a way that each learner serves as an objective for a teacher. So, a teacher tries to optimize its performance against all the learners with which it interacts. This coevolutionary dynamics is often referred as “Pareto coevolution” [44].

Interestingly, this scenario has direct counterparts in educational settings. For example, “practice-problems” can be designed in a way such that students can grasp all or
some of their learning objectives. This phenomenon is described by Watson in his work on minimal substrate of coevolution [141]. In this work, Watson discusses how the setup of two populations can be done in such a way that one population beats (or wins) against the other population for some time, or all the time. This phenomenon is referred to as “loss of gradient” in the literature [45]. Other challenges that have been repeatedly investigated in the Coevolutionary computation literature, also have direct educational counterparts; e.g., focusing on improving on a subset of learning objectives and ignoring others. This is phenomenon often found in Coevolutionary interactions and referred as “focusing” or “overspecialization” [45]. These phenomena are referred to as coevolutionary pathologies and make it often frustrating to understand coevolutionary interaction dynamics. These same pathologies may also be observed when looking at the interaction between practice-problems and students. These analogies between test-based coevolutionary computation and educational “practice-problems vs students” interactions make it particularly relevant to further explore whether techniques from CCoEA may be leveraged to evolve a population of “practice-problems” that interact with actual students.

1.3 Challenges and Opportunities in Candidate Evaluation of Competitive Co-evolutionary Algorithm

In this section, we describe the challenges and opportunities in CCoEA. There are several challenges commonly known as coevolutionary pathologies found in the literature. Be we focus on mostly the crucial ones that are required to design an interactive coevolutionary algorithm based applications. Example of such challenges are “loss of gradient”, “overspecialization” and “intransitivity”. However, such challenges are best described and analyzed with the help of coevolutionary benchmark problems such as number games. Also, we need a good understanding of subjective and objective fitness of candidate solutions to demonstrate
the challenges under the lenses of number games. We then describe the challenges in CCoEA with the help of number games.

1.3.1 Number Games

Number games are examples of symmetric games where each game role shares the same strategy. They are minimal application domains that have been used as benchmarks to illustrate well-known issues occurring in coevolutionary systems. They were first introduced by Watson et al. [141]. Since then, many researchers proposed variants and used them to characterize specific coevolutionary dynamics [20, 34, 47, 30]. In these number games, each individual in the population is an \( n \)-tuple of natural numbers. The number game is defined by the value of \( n \), along with how we choose to compare individuals.

Let us start with an example of a very simple “greater than” number game originally introduced in [141]. In this number game, each individual is represented by a single scalar value. We evaluate each individual \( a \) by having it interact with all other individuals from population \( S \); Evaluation of the candidate solution \( a \) in this particular number game is defined as follows;

\[
f(a, S) = \sum_{i=1}^{\vert S \vert} \text{score}(a, S_i)
\]

In the above, \( S_i \) is the individual at index \( i \) in population \( S \), and \( \text{score}(a, b) = 1 \) if \( a > b \), 0 otherwise. Please note that \( S \) may be composed of individuals from the same population than \( a \), or consists of individuals from another population. There are several other number games [21, 36] - Focusing, Intransitive, CompOnOne - that we describe in Section 1.3.3.

1.3.2 Objective and Subjective Fitness

When considering number games in coevolution, two types of fitness definitions need to be distinguished; objective and subjective.
An objective fitness is a function that returns the quality value of an individual. Such a fitness allows to monitor whether the overall quality of a population is improving during coevolution. On the other hand, a subjective fitness is the outcome from a particular interaction and is therefore context sensitive.

Let us consider \( a \) and \( b \), any two individuals, that we want to differentiate based on the “greater than” number game. Let us also denote the subjective and objective fitness respectively by \( P_{subj}(a,b) \) and \( P_{obj}(a,b) \). We define \( P_{subj}(a,b) = P_{obj}(f(a,S_a), f(b,S_b)) \), where

\[
P_{obj}(x,y) = \begin{cases} 
  \text{a “wins”} & \text{if } x > y \\
  \text{a “looses”} & \text{if } x < y \\
  \text{“draws”} & \text{otherwise}
\end{cases}
\]

At first sight, the “greater than” number game seems to be trivial i.e., maximizing the objective fitness seems to be the optimization process. However, that is not always true because we may get a subjective fitness that differs from the objective fitness, depending on how we choose \( S \) i.e., context sensitivity of the selecting individuals from \( S \). For example, \( P_{obj}(5,6) = 5 \text{ “looses”} \) but \( P_{subj}(5,6) = 5 \text{ “wins”} \), if \( S_5 = 4 \) and \( S_6 = 7 \).

1.3.3 Coevolutionary Pathologies

The outcome of a coevolutionary interaction is context sensitive. For example, one candidate solution may perform very well against an arbitrary set of tests, yet exhibit mediocre performance against all other tests. These types of contextual performance variations open the door to the manifestation of coevolutionary pathologies such as “loss of gradient”, “focusing”, “cycling” etc. [45, 110, 77, 111, 102, 46]. These pathologies, in turn, represent the most common challenges occurring when applying coevolutionary algorithms.
This section will therefore use the previously introduced concept of number games to illustrate how such coevolutionary pathologies might occur.

### 1.3.3.1 Loss of Gradient

The set of candidate (test) solutions may evolve in such a way that the set of tests (candidates) always perform poorly against the candidates (tests). As a result, the coevolutionary search process becomes unable to further improve tests (candidates). This phenomenon is commonly referred to as *loss of gradient* in the coevolutionary computation literature.

This pathology can be illustrated by the “greater than” game. Consider $a = 10$ and $b = 11$, so $P_{obj}(a, b)$ returns $b$. If we choose $S_a = \{4, 5, 6\}$ and $S_b = \{8, 1, 2\}$ then $P_{subj}(a, b) = P_{obj}(f(a, S_a), f(b, S_b)) = P_{obj}(3, 3) = \text{"draw"}$. However, if we choose $S_a = S_b = \{20, 30, 31\}$ or $x \in S_a \mid x > a$ and $y \in S_b \mid y > b$ then $P_{subj}(a, b) = \text{"draw"}$. In the former, $a$ and $b$ both win against $S_a$ and $S_b$ respectively. For latter, they loose respectively. $a$ and $b$ either wins or looses against interacting opponents. So, the gradient of the evolution gets lost. Loss of gradient may occur occasionally during coevolution, however, if it is sustained, it prevents the improvement of either populations.

This pathology can also be found in educational settings. The “practice-problems”, for a specific course, can be designed in such a way that the students either fail or pass all of them. In such scenario, “practice-problems” are not usable to determine the comparative performance of the students.

### 1.3.3.2 Overspecialization, Also Known as Focusing

Another example of well-studied pathology is *overspecialization*. When it occurs, the quality of candidate solutions improve only with respect to a few, or even a single, test rather than improving against all tests.
This phenomenon has a direct counterpart in the “practice-problem vs student” interaction. Suppose students are given a practice problem where three skills are tested. Some students might further improve the skill that they are already comfortable with, at the detriment of the others.

Overspecialization can be demonstrated using number games such as the ones described in [21]. The Focusing Game \((FG)\) determines the outcome of the interaction of an individual \((i_1, j_1)\) with another individual \((i_2, j_2)\). An individual is a two dimensional tuple of natural numbers. For example, \((i_1, j_1)\) is an individual whose first and second dimensions are \(i_1\) and \(j_1\), respectively. The Focusing Game’s pay-off function is defined as follows;

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

First, we identify the dimension with the highest value in the second individual. Then, this value is compared with the value found in the corresponding dimension of the first individual. The individual with the highest value in this dimension wins.

Another number game that produces overspecialization is CompOnOne [34]. In this variant, we first identify the largest dimension of the test. We then compare a candidate solution to that test, using that same dimension.

\[
G_{\text{one}}(C, T) = \begin{cases} 
    1 & \text{if } C_m \geq T_m \\
    0 & \text{otherwise}
\end{cases}
\]

where, \(m = \max_i T_i\) (1.1)

In the above, \(C\) is a candidate, \(T\) is a test, and \(x_i\) denotes the value of individual \(x\) in dimension \(i\).
1.3.3.3 Cycling, Also Known as Intransitivity

Cycling occurs when a coevolutionary algorithm revisits previously discovered solutions due to the fact that the coevolutionary dynamics does not hold the transitive relation.

Let us illustrate a transitive relation involving three integers \(a, b\) and \(c\) as follows; if \(a > b\) and \(b > c\) then \(a > c\). However, we may set up a number game where interactions among tuples of numbers do not satisfy such transitive relation. The interactions may produce scenarios where \((a, b) > (b, c)\) and \((b, c) > (c, a)\) does not necessarily imply that \((a, b) > (c, a)\). The following Intransitive Game (IG) illustrates this type of problem;

\[
IG((i_1, j_1), (i_2, j_2)) = \begin{cases} 
    j_1 > j_2 & \text{if } |i_1 - i_2| > |j_1 - j_2| \\
    i_1 > i_2 & \text{otherwise}
\end{cases}
\]

First, we identify the dimension in which both individuals feature the closest values. Then, we compare these individuals with respect to that dimension only; the individual with the highest value wins. For instance, let us compute the payoff for individual \((1, 6)\) when interacting with individual \((4, 5)\). Both individuals are the closest in their second dimension since \(|1 - 4| > |6 - 5|\). Since \(6 > 5\), individual \((1, 6)\) wins this interaction and receives a score of 1.

This number game can be used to illustrate a cycling behavior. Let us consider three individuals - \((1, 6), (4, 5),\) and \((2, 4)\) - interacting with each other. We see that \((1, 6) > (4, 5)\) and \((4, 5) > (2, 4)\) but \((2, 4) > (1, 6)\) which is a cycle. In a more informal manner, intransitivity can also be illustrated with the rock-scissors-paper game [21] in which rock beats scissors, scissors beats paper, but paper beats rock.

The above challenges are key factors to understand coevolutionary dynamics. However, some established techniques (solution concepts, coevolutionary dimension extraction, archiving, etc.) are already available in coevolutionary literature to understand and improve
the dynamics of coevolution. The next subsection explores these techniques as opportunities to better understand CCoEA dynamics.

1.3.4 Opportunities in Competitive Coevolutionary Algorithm

Though there are frustrating challenges in CCoEA, there are several opportunities that can be helpful to tackle such challenges. In this section, we will describe some of these opportunities.

1.3.4.1 Solution Concepts

The term “solution concepts” was first introduced in game theory [98]. It has then been used in the context of coevolutionary computation by Ficici [46]. A solution concept is a formalism that specifies which elements of a search space are solutions to a search problem. The coevolutionary computation literature has identified several relevant solution concepts [104, 22, 30, 45]; Pareto optimal set, simultaneous maximization of all outcomes, maximization of expected utility, Nash equilibrium, Pareto optimal equivalence set, aggregation etc.

Let us assume that $C$ and $T$ respectively denote the set of all possible candidate solutions and the set of all possible tests. The payoff matrix $G$ describes the performance of all candidate solutions against all tests, where $G_{i,j}$ is the payoff earned by candidate $i$ when interacting with test $j$. Based on the above, we may now formally define the following commonly used solution concepts;

- Aggregation - This solution concept includes candidate solutions $x$ that are better than other candidate solutions $y$ such as $\sum_{i=1}^{\|T\|} G(x, T_i) > \sum_{i=1}^{\|T\|} G(y, T_i)$.

- Pareto optimal solutions - Pareto dominance is the central concept in Evolutionary Multi-objective Optimization (EMO) [45]. A candidate solution $x$ Pareto dominates another candidate solution $y$, with respect to the set of tests $T$, denoted as $x \succ_T y$, where
iff: $\forall z \in T : G_{x,z} \geq G_{y,z} \land \exists w \in T : G_{x,w} > G_{y,w}$. The solution concept for Pareto optimal set consists of all the candidate solutions that are not dominated by any other solution.

- Mutually non-dominating solutions - Candidates $x$ and $y$ are mutually non-dominating, denoted as $x \sim_T y$, iff: $\exists w, v \in T : G_{x,w} > G_{y,w} \land G_{x,v} < G_{y,v}$. $x$ and $y$ can also be referred to as non-comparable solutions since none is better or worse than the other.

- Maximization of expected utility - This solution concept includes the candidates that maximize the expected score against a randomly selected test. The set of this solution concept is defined as $x \in C | \forall y' \in C : E(G(x,t)) \geq E(G(y',t))$, where $E$ is the expectation operator, and $t$ is randomly drawn from $T$.

- Pareto Fronts - We can define the members of the Pareto front are those candidate solutions in $C$ that are non-dominated. Formally, $F^0(C, T) \equiv C - D^0(C, T)$ where, $D^0(C, T) \equiv \{ c \in C : \exists c_x \in C, c_x \succ c \}$. Hence, the Pareto optimal candidates are those candidates in $C$ that are not dominated, as with the previous "Pareto Optimal" solution concept listed above. The set $F^0(C, T)$ is known as the Pareto front.

However, we can go one step further as there may be several such fronts defined on a set of candidate solutions. The candidate solutions can be ranked based on their front number if we layer them based on the Pareto dominance relation. We can recursively examine $D^0(C, T)$ to locate candidates that would be non-dominated if $C$ does not contain the candidates in $F^0(C, T)$. Similarly, we can do that for $F^N(C, T)$ [45].

$$F^N(C, T) \equiv F^0(D^{N-1}(C, T), T)$$
$$D^N(C, T) \equiv D^0(D^{N-1}(C, T), T)$$
• Pareto Equivalent - Two candidates $x$ and $y$ are Pareto equivalent with respect to $T$, denoted $x \equiv y$ iff: $\forall T_j \in T : G_{x,j} = G_{y,j}$.

Being able to choose an appropriate solution concept is central to the design of efficient coevolutionary algorithms. More specifically, it is related to being able to guarantee monotonic progress during the convergence [46]. However, even when an appropriate solution concept is adopted, Ficici showed that monotonic progress may not be achieved [45]. On the other hand, when designing a coevolutionary algorithm, it is wise to choose a solution concept that has proven to be better than others in terms of avoiding specific pathologies. Bucci et al., [21] showed that the Pareto dominance solution concept is better at avoiding overspecialization than, for instance, the aggregation solution concept.

Although the Pareto-dominance solution concept is better at avoiding this particular pathology, test-based Pareto coevolution requires optimizing the performance of candidates against all the tests. This is because the tests are treated as objectives from the perspective of the evaluation of candidate solutions. In this regard, test-based Pareto coevolution is similar to many objective optimization[45]. However, not all the tests are equally as important in determining the worth of candidate solutions. So, we need to select the representative tests to evaluate the candidate solutions. This leads to the idea of coevolving a set of ideal solutions.

1.3.4.2 Ideal Evaluation

Coevolution evolves the set of tests used for evaluation, but has often been plagued by the impact of inaccurate evaluation of candidate solutions [34]. The term “ideal evaluation” was first introduced by De Jong [34]. His work introduced the idea that the candidate solutions can be evaluated against a set of representative tests, instead of all the tests, without losing anything. This set of representative tests elicits the underlying structure of the problem that the algorithm is trying to solve.
1.3.4.3 Coevolutionary Dimension Extraction (CDE)

In [33, 23], the authors developed automatic extraction methods to determine the underlying objectives from coevolutionary interactions. This technique is commonly referred to as CDE. Its roots are established both in coevolution and co-optimization theory [19]. The CDE algorithm takes a matrix, representing interactions between all candidates vs. all tests, and extracts a vector-space-like coordinate system featuring potentially multiple dimensions as illustrated in Figure 1.2.

![Coordinate system for candidates obtained by CDE algorithm](image)

**Figure 1.2:** Coordinate system for candidates obtained by CDE algorithm

Each dimension is a linearly ordered subset of the set of all candidate solutions. A candidate that is located further along a given axis is “no worse than” those preceding it. For instance, candidate $PP_4$ in *Dimension 1* Pareto dominates candidate $PP_6$. Similarly $PP_6$ Pareto dominates $PP_2$.

A problem with applying the CDE algorithm is that it requires a full interaction matrix featuring all possible interactions [23]. A possible workaround has been proposed
in the form of combining the benefits of both aggregation and Pareto-optimality [81]. It is however unclear whether the authors handle sparse interaction matrix and how.

1.3.4.4 Archiving

Monotonic improvement may be guaranteed in a coevolutionary algorithm by using either of the following two approaches. The first approach consists in striving towards accurate, or ideal, evaluation. The second approach consists in maintaining an archive of solutions [31, 32, 67]. It is important to consider that an archive-based algorithm requires extra memory in comparison to its non-archive-based counterparts. For this reason, it is relevant to keep exploring possible algorithms that can achieve good performance without relying on archives.

The authors in [31, 67, 32] showed that bounded and incremental Pareto coevolutionary archive algorithms can maintain steady progress in coevolution. Examples of such no-archive methods can be found in [21, 48, 8].

1.3.5 Discussion on Challenges and Opportunities in CCoEA

There are significant challenges in establishing a fruitful coevolutionary dynamics that avoids pathologies. The literature also identifies ways to understand and track the performance of CCoEA. For example, the underlying solution concept can be chosen at the beginning of the algorithm development. Choosing an existing or proposing a new solution concept for an algorithm also depends on several factors. Aggregation or maximization of expected utility can be chosen for the algorithm that solves single objective problems. On the other hand, Pareto Optimal or Nash equilibrium can be a good choice of solution concept for the algorithms that solve multi or many objective problems.

The choice of solution concept also depends on the evaluator of candidate solutions, cost and time required for a single evaluation. In case of human evaluator, aggregation based
solution concept seems a better fit. However, Pareto optimiality based solution concept can also be chosen. In such scenario, minimum number of candidate evaluation needs to be fixed at the beginning of algorithm design.

The choice of solution concept has also direct consequence of algorithmic performance of avoiding certain coevolutionary pathologies. For example, an algorithm that implements aggregation as its solution concept does not avoid overspecialization pathology better than that implements Pareto optimality.

The monotonic progress of an algorithm is another important aspect that a successful CCoEA should exhibit. An algorithm is said to be shown monotonic performance, if the evolved solutions in previous generation are at least equal or better than previous as the generation increases. Competitive coevolutionary interaction can be intransitive i.e., a previously visited solution can be revisited. Though some solution concepts guarantee monotonic progress, choosing the best applicable solution concept for CCoEA may not guarantee the monotonic progress. In such scenarios, maintaining an archive of the previous best candidate solutions helps prevent losing them at the cost of extra memory.

In addition, choosing a solution concept that is very strict in its filtering of best performer prohibit the exploration of CCoEA’s search space. For example, Pareto dominance is an example of very strict solution concept by definition. A candidate should perform better (greater) against at least one test than its opponent candidate in same test. When two candidates optimize against many tests, the chance of dominating one candidate against other becomes too low. A CCoEA implementing Pareto dominance as a solution concept thus exploits the dominant solutions. So, relaxing the strict condition (e.g., greater than equal instead of greater in dominance relation of two candidates) of the solution concept improves overall performance of the algorithm with sacrificing the non-decreasing performance.

In Pareto coevolution, candidates are tested against a set of tests. Each candidate acts as an objective against tests. So, tests want to maximize the performance against
candidates. Similarly, tests act as an objective set for candidates. In such optimization algorithms, a single candidate evaluation needs interactions with many tests. Typically, the objective size is as large as the population size. Allowing so many interactions for a single candidate evaluation is impractical in applications especially when human evaluator is engaged.

However, the underlying objectives says that we don’t need all the interactions to guide the coevolutionary search towards ideal evaluation. Instead, a set of objectives exists which guarantee the ideal evaluation. In this regards, CDE can automatically extract such ideal set of objectives (candidates/tests) and converge to the exact dimensions of benchmark problems (e.g., dimension of number games) in certain conditions. However, CDE has some limitations - requires a full binary interaction matrix between candidate and tests - that prohibits CDE to apply it into applications where solutions are evaluated by human evaluator.

1.4 Challenges and Opportunities in Candidate Evaluation for Interactive Evolutionary Algorithm

The evaluation of candidate solution in IEA needs human evaluator. This is because the fitness of the solution is subjective. As a result, IEA-based applications are designed in such a way that the quality of candidate solution evaluation is minimally affected by user fatigue.

1.4.1 User Fatigue

Human fatigue phenomenon is common to all human-machine interaction system. User fatigue has been repeatedly identified in the literature as one of the prominent challenges especially when using an IEA [123] [125] [126]. The quality of candidate solution evaluation becomes unreliable if users are given similar solutions again and again. Doing so will cause
users disinterest to use the system or provide candidate evaluations that are not helpful to
direct IEA into better solution space. So, we should reduce user fatigue while evaluating
candidate solutions in IEA. This challenge is even more prominent in educational applica-
tions e.g., “practice-problems vs learner” interactions. In such applications, student’s solve
practice problems by interacting with the candidate solutions (practice-problems) evolved
by the algorithm. Lack of proper user fatigue mitigation schemes in such learning appli-
cations may result loosing its attraction to the students. For example, students may find
the candidate solutions too easy or too difficult if the underlying algorithm does not select
practice-problems that is under student’s learning skills.

1.4.2 Techniques to Reduce User Fatigue

A significant number of different user-mitigation schemes have been proposed so far. Some schemes use statistical ranking and selection mechanism while others propose surrogate
model, some involve the evaluators more closely to the EC search, while others propose a
better selection algorithm etc. We categorize the literature based on those proposed schemes
and continue our discussion in the following subsections.

1.4.2.1 Selecting Better Solutions

Selective pressure is fundamental to both exploitation and convergence properties of
IEAs. Its role in indirectly mitigating user fatigue by accelerating convergence has been,
however, less studied.

Branke et al.[16] proposes three selection procedures; the Indifference Zone (IZ),
the expected Value of Information Procedure (VIP) and the Optimal Computing Budget
Allocation (OCBA). The IZ [13, 75] approach tries to guarantee that Probability of Correct
Selection (PCS) will produce the best individual which is at least $\delta^\ast$ better than the other.
However, the IZ is statistically conservative and assumes that the cost of simulating each individual is the same. So, the VIP allows a computing budget constraints and permits the cost of each individual’s simulation to be different. Lastly, OCBA [29] allocates an additional $\tau$ replications to an individual but no replications to other individuals which lead to an Estimated Approximate PCS ($EAPCS$).

### 1.4.2.2 Statistical Ranking of Solutions

In [113], the authors integrate new statistical ranking and selection procedures. These focus the sampling of representative individuals, and also distribute samples in a way that efficiently reduces uncertainty. Their approach is a variant of OCBA which evaluates the fitness function $n_0$ times per individual during the first stage of sampling. In each subsequent sequential stage, $\tau$ additional evaluations are given to one individual, and none to others. The algorithm stops sampling when there is sufficient evidence for correct selection.

EXPLORE-m [71] selects a subset of size $m$ with highest mean but requires as few samples as possible from $n$ real-valued random variables. The authors provide three algorithms; direct, incremental and halving. These algorithms are the extensions of EXPLORE-1 that uses $k$-armed bandits to identify the best of $n$ random variables. In the $k$-armed bandit problem, there are $k$ alternative arms each with a stochastic reward whose probability distribution is initially unknown [87]. The objective is to select a policy for choosing the arm with maximum expected rewards.

With the $direct$ algorithm, bandit arms are sampled a fixed number of times such that with high probability $m$ arms with the highest empirical averages are all $(\epsilon, m)$ optimal.

The $incremental$ algorithm proceeds through $m$ rounds. At the beginning of round $l$, $S_l$ is the set of arms that have been selected, and $R_l$ is the set of arms remaining. During round $l$, and $(\epsilon, 1)$-optimal arm in $R_l$ is selected with high probability.
While the incremental algorithm selects an arm every round, the **halving algorithm** eliminates multiple arms every round, based on their inferior empirical averages. Arms are sampled enough in each round such that at least $m (\epsilon, m)$-optimal arms are likely to survive elimination.

### 1.4.2.3 Evaluation of Solutions by Human Evaluator

The statistical ranking and selection algorithms select and rank the best individuals with highest mean using minimum sampling. Those algorithms reduce search spaces by discarding the individuals whose mean is below a threshold. There are also other algorithms which reduce the search space by allowing human evaluators to fix some genes so that they are left unchanged for upcoming generation. This allows human experts to guide the evolutionary search by preserving traits that are judged to be beneficial. Takagi et al., [124], implement this IEA variant in two ways; the online knowledge embedding, and the visualized IEA.

In the **knowledge embedding** approach, when a user perceives that s/he finds an image similar to his/her target image, s/he can select and fix that image. The IEA then prunes the search space by fixing the genes that encode that phenotypical feature. However, this method is applicable when genes have a one-to-one correspondence with the phenotypical features being fixed. In other terms, when fixing a parameter will fix a particular phenotype for target outcome. The **visualized IEC** combines different advantages of EC and human search techniques. It introduces a $2D$ visualization of an $nD$ gene space based on EC operators which is better than human search technique. The role of human as an evaluator in visualized IEC helps EC search because human have an excellent capacity to grasp an entire distribution of individuals in the $2D$ space.

Nakasu et al., [93] also propose a novel interactive system where users can take part to draw facial caricatures reflecting their target facial impression. In their system, face’s
important feature points and line of caricatures are used to provide their users a variety of expression. There are total 300 feature points which are connected by spline curves. The visibility of these curves are also controlled by 126 parameters which are treated as “gene” for their interactive system.

Users are presented with a collection of ready-made caricatures as the initial generation. They can also create new caricatures and add them to the set of dominant gene. However, users can get more accurate facial impression if they evaluate more generation.

Picbreeder [114] evolves aesthetically pleasing images via an online community allowing users to share their images, and even continue evolving other users’ images. This approach has been termed Collaborative Interactive Evolution (CIE) [122]. To do so, Picbreeder users either start from a random image of their choices or begin evolving from an already published image. In a single evolution session, the collection of images evolved throughout the generations is called series. The last image of a series is defined as representative image or genome for that series. While Picbreeder retains every image in each series for further analysis, users browsing the site only see the representative images. The process of evolving from a representative image is called branching. When branching, a representative spawns the first generation of the new branched series. So, Picbreeder implements branching and thus allows long chains of content to grow. This reduces the work of each individual user.

1.4.2.4 Limiting User Intervention

Although fixing some traits can help reduce the size of the search space, and thus indirectly help mitigate user fatigue, evaluating a large number of individuals in every generation will negatively affect the reliability of evaluations, due to the already discussed user fatigue phenomenon. Industries like software and manufacturing limit user participation in every $n^{th}$ generations while developing UI and simulating machine design respectively.
The human evaluator is also restricted to select a limited number of individuals in those specific generations. In Kamalian et al. [72], human participation is more in a supervisory role. In this role, human are allowed to input their expert knowledge and visual perception of a design when desired. It provides evolution of machine design not in every generation but only every $n^{th}$ ($n = 10$ in their experimental setup) generation. A user requires shorter time (45 minutes) for one run of the IEC than its previous implementation (60 minutes) because a user evaluates each individual in each generation in the previous IEC synthesis.

Pallez et al., [99] use a non intrusive eye-tracking system like Tobii$TM$. They applied this system to the interactive one-max problem [1], maximizing $\sum_{i=1}^{l} x_i$, $x_i \in \{0, 1\}$ for a binary string $x$ of length $l$. Their system calculates total time and frequency a user spending on an individual before selecting it for next generation. The system also measures pupil diameter of user’s eye which is inversely proportional to user’s concentration [68].

The method has several advantages which help mitigate user fatigue by measuring time duration and the frequency of “stare at” to an individual, analyzing cognitive activity, making the system non-intrusive i.e., user’s could forget that s/he is being observed. For example, when transition numbers between individuals decreases or total watch time for a generation is decreased, there is a chance that user is bored. However, the eye tracker need to be calibrated to the user and user does not have full freedom of head movement.

Quiroz et al., [105, 106] develop a grid-based widget layout. They also encode aesthetic properties, e.g., color, of those widget. Button, label, text box, slider, drop down list etc. are example of widgets. Quiroz defines two chromosomes; characteristics chromosome, that encodes color of each widget in a standard bit string, and layout chromosome, that encodes the position of the widgets in the grid. All the widgets are identified by an integer (ID) greater than zero; empty cells in the grid are identified by zero. The user interface layouts are displayed for evaluation by the user (a user interfaces designer) who is asked, in every $t$
generations, to choose two layouts: the one considered the best and the one considered the worst.

### 1.4.2.5 Surrogate Model

Surrogate user models are meant to mitigate user fatigue by applying machine learning techniques to model the user’s preferences, as we observe the selections they make while using the system. Such models can then be used to approximate users’ selections, and thus spare them having to consider too many specimen.

The Approximate User (TAU) in [61] builds two user modeling approaches - one is for human evaluation i.e., selecting the better of two trigonometric shapes, and the other one is for driving a part of EC search i.e., predicting fitness score of a shape. A classifier takes features of two shapes as input, builds a training algorithm which then uses these features to predict the shape that would be selected by that user. The classifier acts as a user model for evaluation. A user preference relationship graph on individual is built which eventually determines the fitness score of each individual. TAU was compared with basic IEA and found 75 times more reliable at achieving optimal results and 2.75 times faster than a more traditional IEA.

In [121], Sun et al. propose two surrogate models respectively built with a classifier and a regressor to recognize the fuzzy uncertainties of a user cognition. The models create two training datasets; input samples - the individuals evaluated by the user, and output samples - individuals’ fuzzy fitness assigned by the user. An individual’s fuzzy fitness is a kind of imprecise fitness that can well reflect user’s cognition on the evaluated object.

When assigning a fitness, a user gives an appropriate linguistic value and its center according to his/her preference, such as “about 20”. A Support Vector Classification (SVC) is used to estimate the width and a Support Vector Regression (SVR) to center [132]. These
two surrogate models are simultaneously applied to the subsequent generations with enlarged population size so as to alleviate user fatigue and enhance the search ability of the algorithm.

1.4.2.6 Various Schemes of Mitigating User Fatigue

In [143], the authors propose a novel way to strike a new exploration vs exploitation trade-off in evolutionary methods. Their approach relies on Temporal Difference (TD) [12] reinforcement learning methods’ selection mechanism to choose action and use that action to choose policies for evaluation. This method evaluates three selection mechanisms and test their performance on two reinforcement learning tasks such as mountain car and server job scheduling.

The first selection algorithm, $\epsilon$-greedy selection [139], selects a policy randomly with $\epsilon$ probability whereas it exploits by selecting the best policy discovered so far in the current generation with $(1 - \epsilon)$ probability. The ranking of selected policy depends on the score of each policy which is just the average reward per episode the policy has received so far.

The second selection algorithm is based on Softmax [12] and distributes evaluations proportionally to each individual’s estimated fitness. At the beginning of each generation, each individual is evaluated for one episode, to initialize its fitness. Then the remaining episodes are allocated for an individual $p$ with a probability which is proportion to its fitness $f(p)$. Softmax selection can abandon a poorly performing policy quicker than $\epsilon$-greedy selection because it varies the selection probability of a policy as a graded function of estimated value.

The third selection algorithm uses interval estimation [70] to compute confidence intervals for the fitness of each policy, and always evaluates the policy with the highest upper bound.

Miler et al. [91] model convergence characteristics of several selection schemes, e.g., proportionate selection, tournament selection, $\mu - \lambda$ selection. They extend selection inten-
sity to accurately predict the selection pressure, a degree to which the better individuals are favored. This pressure drives the GA to improve population fitness over succeeding generations. Experimental results show that their predicted model is almost same as experimental average.

Gong et al., [52] divide the population into several clusters. The users only evaluate one individual of each cluster, referred to the cluster representative. The fitness of other individuals are estimated by their representative’s fitness. Because the user only evaluates cluster representative, the number of individuals being evaluated by the user also changes along with the evolution. The computer estimates other individuals’ fitness in a cluster and expresses it with an interval. The midpoint of the fitness is equal to the center’s fitness, and the width of the fitness is related with the similarity between the estimated individuals and their center.

Wang et al., [138] introduce the idea of absolute scale which improves prediction accuracy of the user’s subjective evaluation in IEC. Thus, it accelerates convergence and reduces user’s fatigue. An absolute scale [137, 136] is the average difference in evaluation between two neighboring generations. The key definition is user’s rating scale on individual is relative within each generations i.e., best individual in $i^{th}$ generation may be worse than the worst individual in the $(i + j)^{th}$ generation. These generational difference is called noise. So, the algorithm can learn and predict user’s evaluation characteristics from this noise.

1.5 Competitive (Co)Evolutionary Algorithms and Their Applications: Analysis from the IEEE Xplore Meta Data Repository

In our target applications of “practice-problem vs students” interaction, coevolutionary pathologies are prevalent as described in section 5.1. As a result, we are more interested to investigate methods from state of the art that resolve the challenges often found in co-evolutinary interactions. To do so, we extract meta data from the IEEE Xplore repository
using their available meta data API. It provides a set of query parameters to retrieve specific information of publications. For example, we can query using article number, author’s name, year of publications, index terms, publication title and many more. In addition, a free text search parameter is also available. A client can provide customized keywords as query parameter in order to retrieve publications with meta data, using free text search parameter. Using this special parameter, the repository can be queried to retrieve articles containing query keywords in the abstract and the document text of the publication. In this analysis, we use the following keywords as free text search parameter.

- First one is $Q_1$ (Generalized Query). “(coevolution* or co-evolution* or co-evolutionary algorithm or coevolutionary algorithm).” It retrieves publications with their title, abstract and all meta data that contain any word from coevolution, coevolutionary, co-evolution, co-evolutionary, co-evolution, algorithm, co-evolutionary, algorithm. $Q_1$ returned 1651 papers in total. Figure 1.3 summarizes the number of publications and citations of these papers, by year. However, these papers include both cooperative and competitive coevolutionary algorithms. Since $Q_1$ lumps together both types of coevolutionary algorithms, we refine our next query to only focus on competitive co-evolution.

- Second query is $Q_2$ (Specific Query). “((competitive AND coevolution) OR (competitive AND co-evolution) OR (competitive AND co-evolution))”. This query returned only those papers that must have two keywords (e.g., competitive coevolution, competitive co-evolution). Interestingly, $Q_2$ returns 40 papers only. $Q_2$ did not include any terms related to the challenges and opportunities in competitive coevolutionary algorithms that we previously described in Section 1.3. Our next query therefore integrated more specific keywords meant to include publications related to the various known pathologies and solution concepts.
• \( Q_3 \) (More Specific Query): “((competitive AND coevolution) OR (competitive AND co-evolution) OR (competitive AND co-evolution)) AND (intransitive OR cycle OR focusing OR pathology OR solution-concept OR pareto OR subjective+fitness OR objective+fitness OR arms-race OR forgetting) NOT (cooperative OR co-operative)”.

This query is supposed to return papers that are found in \( Q_2 \) but also focus on coevolutionary pathologies or solution concepts, while still excluding cooperative algorithms. This query resulted in 24 papers.

Figure 1.3: Summary of number of publications and citations grouped by year using \( Q_1 \).
As $Q_3$ is more specific to the challenges and opportunities of competitive coevolutionary algorithms, we paid attention to those 24 papers specially. Our first step was to perform a terms frequency analysis. We consider “index_terms” attribute which is basically the union of “ieee_terms” and “author_terms”. Figure 1.4 shows the keywords that most frequently occurred in those 24 papers. It seems that “Game”, “sorting”, “Evolutionary computation”, “Genetic Algorithm”, “model”, “Optimization” are the most prominent words found in those papers.

We then proceeded to review the papers in order to address the following questions:

1. What is the proposed contribution of the paper?

2. Is the work applied to a realistic application where candidates are evaluated by human expert or artificial agent?

3. Is there any use of relevant solution concepts, archive or CDE?

4. What solution concept(s) the proposed algorithms use to evolve?

5. More importantly, does the paper propose strategies to mitigate specific co-evolutionary pathologies?

6. Does the proposed method fall into single, multi or many objective optimization?

The above questions are very important to researchers interested in competitive co-evolutionary algorithm as answering them is fundamental to both understanding these algorithms and developing effective variants for new problems.

Examples of fundamental characteristics to consider when designing a coevolutionary algorithm include; choosing an appropriate solution concept, determining if an archive will be maintained, guaranteeing the monotonic progress of the evolution, deploying techniques to
avoid known coevolutionary pathologies... While not exhaustive, the above list encompasses fundamentals design decisions that any researcher interested in applying coevolutionary algorithms to an interactive domain will have to face. Table 1.1 summarizes answers of most of the above questions by thoroughly investigating the works retrieved by $Q_3$.

**Table 1.1:** Design parameters along with challenges and opportunities for competitive coevolutionary algorithm based system.

<table>
<thead>
<tr>
<th>Solution Concept</th>
<th>Archive</th>
<th>Pathology</th>
<th>Solution Evaluator</th>
<th>#Objectives</th>
<th>Study</th>
</tr>
</thead>
<tbody>
<tr>
<td>Pareto Dominance</td>
<td>No</td>
<td>NA</td>
<td>multi</td>
<td>82</td>
<td></td>
</tr>
<tr>
<td></td>
<td>Yes</td>
<td>NA</td>
<td>NA</td>
<td>128</td>
<td></td>
</tr>
<tr>
<td></td>
<td>No</td>
<td>NA</td>
<td>multi</td>
<td>135</td>
<td></td>
</tr>
<tr>
<td></td>
<td>No</td>
<td>Yes</td>
<td>multi</td>
<td>146</td>
<td></td>
</tr>
<tr>
<td></td>
<td>Yes</td>
<td>NA</td>
<td>NA</td>
<td>85</td>
<td></td>
</tr>
<tr>
<td></td>
<td>No</td>
<td>NA</td>
<td>multi</td>
<td>84</td>
<td></td>
</tr>
<tr>
<td></td>
<td>No</td>
<td>NA</td>
<td>NA</td>
<td>147</td>
<td></td>
</tr>
<tr>
<td></td>
<td>Yes</td>
<td>NA</td>
<td>multi</td>
<td>127</td>
<td></td>
</tr>
<tr>
<td></td>
<td>No</td>
<td>Yes</td>
<td>multi</td>
<td>9</td>
<td></td>
</tr>
<tr>
<td></td>
<td>Yes</td>
<td>NA</td>
<td>multi</td>
<td>109</td>
<td></td>
</tr>
<tr>
<td></td>
<td>No</td>
<td>NA</td>
<td>multi</td>
<td>116</td>
<td></td>
</tr>
<tr>
<td></td>
<td>Yes</td>
<td>NA</td>
<td>single</td>
<td>111</td>
<td></td>
</tr>
<tr>
<td></td>
<td>Yes</td>
<td>Yes</td>
<td>single</td>
<td>129</td>
<td></td>
</tr>
<tr>
<td></td>
<td>No</td>
<td>NA</td>
<td>multi</td>
<td>66</td>
<td></td>
</tr>
<tr>
<td></td>
<td>No</td>
<td>Yes</td>
<td>NA</td>
<td>79</td>
<td></td>
</tr>
<tr>
<td></td>
<td>No</td>
<td>NA</td>
<td>single</td>
<td>133</td>
<td></td>
</tr>
<tr>
<td></td>
<td>No</td>
<td>Yes</td>
<td>single</td>
<td>96</td>
<td></td>
</tr>
<tr>
<td></td>
<td>Yes</td>
<td>NA</td>
<td>single</td>
<td>109</td>
<td></td>
</tr>
</tbody>
</table>

It is important to avoid coevolutionary pathologies when designing a coevolutionary algorithm. Otherwise, we will get evolved solutions, for example, that are already visited during the evolution. It is surprising to notice that very few papers are actually focused on mitigating the well-known coevolutionary pathologies.

It seems that aggregation-based work are more interested to use “archive” solutions to avoid pathologies i.e., maintaining monotonic progress. Though some of the Pareto dominance based studies maintain archive, directions for detection or avoidance of pathology are
rarely found except in [111]. We also note that, all of the studies evaluate the solutions with other candidate/test solutions.

Please note that, [3] is a survey paper on coevolutionary algorithm and multi-objective optimization. It describes all types of cooperative and competitive coevolutionary algorithms and pathologies in general. Ebner et al., [40] finds the necessary conditions for arms race using fitness landscape decomposition where they model only population average instead of population individuals found in usual evolutionary model.

The remaining three papers are not related to competitive coevolutionary algorithms. Instead, they contain the word “competitive” to manifest that their results are competitive than other state of the art. Therefore, we exclude those three papers while summarizing.

Figure 1.4: Word cloud formed by index terms of the papers retrieved using Q3.
1.6 Discussions

In this chapter, we discussed the challenges found in IEA and CCoEA and also showed the possible opportunities for designing efficient interactive competitive coevolutionary algorithms. We also perform a thorough analysis of state of the art competitive coevolutionary research works to understand their methods to overcome such pathologies. We also discuss which specific opportunities from coevolutionary literature they use or propose any new opportunities. However, there is no gold standard to detect, understand, and resolve the pathologies found in coevolutionary interactions. Some of the pathologies, e.g., overspecialization, may be observed when using carefully designed synthetic benchmarks.

However, the study of their presence and detection in real applications remains an open research area. In addition, several user fatigue mitigation schemes were explored. It is interesting to explore what are the design guidelines for a system that would leverage coevolution to evolve candidate solutions that require human evaluations. Such system would present the challenge of needing to handle coevolutionary pathologies, mitigate user fatigue. Some of the following points to ponder on developing interactive competitive coevolutionary algorithm

- We need to measure the progress in Coevolution. The tracking of progress in coevolution is not the same as that of traditional optimization where solutions are evolved targeting a static/dynamic minimization/maximization function. This is because the fitness is context sensitive in coevolution. The Coevolutionary progress can be measured for benchmark problems e.g., number games. For applications such as “practice-problem vs student” interactions, the progress need to be measured both in quantitatively and qualitatively.

- We also need to consider the role of solution concepts to detect pathology. Pareto dominance based solution concept is proved to tackle overspecialization in coevolu-
tion better than aggregation solution concept. In addition, Pareto dominance also prohibits coevolutionary search to fall into cycling. Avoidance of certain pathologies e.g., overspecialization and cycling can be shown in a coevolutionary interactions of number games. In case of “practice-problem vs student” interaction, detecting the presence of coevolutionary pathology is an open question that requires further investigation. Though designing/choosing solution concept depends on other criteria also (such as, do we study the problem in terms of mono/multi/many objective optimization), considering the role of solution concept in avoidance of pathology should be studied significantly.

• We need techniques to mitigate user fatigue. As soon as we combine CCoEA techniques with fitness evaluations based on interactions with actual students, it becomes imperative, as with most IEAs, to mitigate efficiently the user fatigue phenomenon. While developing or choosing mitigation techniques for any educational application, we need minimum number of student evaluation for each practice problem to accelerate the evolution.
Chapter 2: Elementary Coevolution

2.1 Coevolutionary Interaction Dynamics

This chapter was published in IEEE ICTAI conference \(^1\) and AAAI FLAIRS conference \(^2\). IEEE and AAAI permissions are included in Appendix A.

EAs are meta-heuristics that have been successfully applied to a wide range of search and optimization problems for which provably efficient algorithms are not available. The absence of a formal model of the underlying optimization problem often makes these algorithms one of the few available options. As such, they are interesting as novel approaches to generate practice problems.

The first relevant such research niche is that of CEA. Interestingly, the “pathological dynamics” observed in such coevolutionary systems are often analogous to those observed in educational settings in which a population of learners interacts with a set of practice problems. To intuitively illustrate this relation, one may consider the loss of gradient phenomenon occurring when a population significantly outperforms its coevolutionary counterpart. Such a dynamic finds a natural counterpart in the classroom when instructors’ hand-designed assignments fall outside the difficulty level that learners are able to tackle with tutoring, but not alone. Recent theoretical advances in CEA theory open new opportunities to improve our understanding of the dynamic occurring when adapting a set of practice problems.

\(^1\) Alessio Gaspar, ATM Golam Bari, Amruth N. Kumar, R. Paul Wiegand, Anthony Bucci, Jennifer L. Albert. "Evolutionary Practice Problem Generations: Design Guidelines". In the proceedings of International Conference on Tools in Artificial Intelligence, 2016, Nov 6-9, San Jose, California, USA.

alongside a population of learners. This, in turn, is essential in gaining insights into efficient coevolutionary solutions to generate practice problems.

The second relevant research niche is that of IEA. In order to be evaluated, evolved practice problems have to be first worked on by learners. IEA are characterized by applications where candidate solutions being evolved are evaluated via their interaction with human agents. As such, many of the problems e.g. the user fatigue phenomenon, documented in the IEA literature, are directly relevant to our target application.

We therefore propose to identify problems occurring in both CEA and IEA that are relevant to pedagogical issues involved in evolving practice problems for learners. The suitability of state-of-the-art algorithms from both research communities’ literature are then evaluated with respect to a suite of benchmark problems. These problems are designed to start with simple environments used to establish theoretical properties, after which we progressively add characteristics that are expected to be found when evolving practice problems for learners. Such an approach is preferable over directly experimenting with real students for two reasons. From an ethical perspective, students should not be exposed to tools without first accruing evidence of the tools’ potential. From a pragmatic perspective, a single experience with a group of student would take a semester as we would have to wait for the relevant courses to need our tool. As we experimentally evaluate our algorithms on these approximations of our target problem, we discuss relevant guidelines regarding the design of suitable evolutionary approaches.

In this chapter, we investigate the interaction dynamics of elementary coevolution; more specifically focus on the effect of noise, gene bounds and presence of pathology’s in the interaction. This helps us to develop properties and propose variant of elementary coevolutionary algorithm. Based on the properties of variant of elementary coevolution, we eventually propose propose a minimal design guidelines for interactive coevolutionary algorithm that are relevant to evolve practice problem for real human learners.
2.2 Experiment#1 - Elementary Coevolution

2.2.1 Problems

Our first experiment uses a benchmark coevolutionary problem; FG. Despite its simplicity, this number game has been shown to be sufficient to hinder the emergence of effective arms-race dynamics in coevolutionary computation by encouraging individuals to overspecialize. Overspecialization occurs when individuals improve by maximizing a single dimension. While highly competitive, such individuals miss the optimal strategy which consists of maximizing both dimensions. As such, this simple problem enabled us to investigate the suitability of various algorithms for problems in which overspecialization is likely to occur.

2.2.2 Algorithms

To address the above problem, we considered two algorithms defined in [21]. Population-based Coevolutionary Hill Climber (P-CHC) is a one-population based coevolutionary algorithm. It is also called “trivial coevolution” because it acts as a test-bed to understand the interaction dynamics of coevolutionary algorithms. Algorithm 2.1 describes P-CHC.

Every individual is mutated to produce a single child. This offspring is then compared to the parent based on its subjective fitness and only the best is kept. The subjective fitness is the sum of the scores resulting from interactions of an individual with every other in the population. Wins/losses result respectively in scores of 1/0.

Population-based Pareto Hill Climber (P-PHC) is a coevolutionary algorithm that uses two populations; one for tests and one for candidates. Algorithm 2.2 describes P-PHC. When evaluating individuals, each candidate interacts with each test and vice-versa. As with P-CHC, individuals are mutated then compared with their parent. However, this time, we rely on the concept of Pareto dominance for candidate evolution and informativeness for
Algorithm 2.1 P-CHC - Population-based Co-evolutionary Hill Climber Algorithm

P-CHC (Payoff Function PF)

1: \textit{popsize} ← 100
2: \textit{population} \( P \) ← \{
3: \textbf{for} \ \textit{popsize} \ \textbf{times} \ \textbf{do}
4: \quad \textit{P} ← \textit{P} \cup \{ \textit{new random individual} \}
5: \textbf{end for}
6: \textbf{repeat}
7: \quad \textbf{for each} \ \textit{individual} \ \textit{p} ∈ \textit{P} \ \textbf{do}
8: \quad \quad \textit{c} ← \textit{Mutate}(\textit{p})
9: \quad \quad \textit{f}_p ← \textit{f}_c ← 0 \quad \triangleright \text{initial fitness}
10: \quad \quad \textbf{for} \ \textit{j} = 1 \ \textbf{to} \ \textit{popsize} \ \textbf{do}
11: \quad \quad \quad \textit{f}_p ← \textit{f}_p + \text{PF}(\textit{p}, \textit{P}_j)
12: \quad \quad \quad \textit{f}_c ← \textit{f}_c + \text{PF}(\textit{c}, \textit{P}_j)
13: \quad \quad \textbf{end for}
14: \quad \quad \textbf{if} \ \textit{f}_c > \textit{f}_p \ \textbf{then}
15: \quad \quad \quad \textit{P} ← \textit{P} \setminus \{\textit{p}\}
16: \quad \quad \quad \textit{P} ← \textit{P} \cup \{\textit{c}\}
17: \quad \quad \textbf{end if}
18: \quad \textbf{end for}
19: \textbf{until} \ \text{Stopping Criterion Met}

test evolution. For candidates, only the non-dominated individual is kept. Other aspects are identical to P-CHC.

In both algorithms, a generation is processed by considering each individual in the first population as a parent and then by generating a mutated offspring (child) from it. Both child and parent then each interact with all the individuals from the other population. These interactions are recorded in so-called outcomes vectors, one for the parent and one for the child. We then consider all individuals in the second population and repeat the same process. P-CHC and P-PHC differ in how the parent’s and child’s outcome vectors are compared in order to determine who will be kept in the next generation and who will be discarded.

- P-CHC assesses the fitness of an individual (\textit{ind}) by summing the values in its outcome vector:

\[
f(\text{ind}) = \sum_{s_c \in S} \delta(s_i, \text{ind}),
\]
where $s_i \neq \text{ind}$ is the other individuals from $S$, $\delta(x,y)$ denotes an interaction outcome between two individuals — $x$ and $y$, and $f(\text{ind})$ is the fitness of $\text{ind}$. A child, $c$, is said to be strictly better than its parent, $p$, and thus replaces it in the next generation iff $f(c) > f(p)$. Both candidates and tests are evolved based on $f(p)$.

- P-PHC does not aggregate the values of the outcome vector. Instead, each outcome is treated as an objective in the sense of multi-objectives optimization. The outcome vectors are therefore compared using the concept of Pareto dominance. The outcome vector of a child ($\vec{x}_c$) Pareto-dominates that of a parent ($\vec{x}_p$), which is denoted by $\vec{x}_c \succ \vec{x}_p$, if the following conditions are met: 1) $f_i(\vec{x}_c) \geq f_i(\vec{x}_p)$ for all $i$ in $\vec{f}$, and 2) there is at least one $i$ such that $f_i(\vec{x}_c) > f_i(\vec{x}_p)$. A child replaces its parent in the next generation if it is strictly better, i.e., its outcome vector Pareto-dominates the parent’s. The candidates are thus evolving based on the concept of Pareto dominance, whereas the tests evolve based on an informativeness measure described in the following subsection.

2.2.3 Methods

As in [21], we used objective fitness in order to measure coevolutionary improvement, or lack thereof. To this end, we defined the objective fitness of any given genotype as being the sum of the two integer values used as genes. In the original study, the authors plotted this sum against the generation number, i.e. evolutionary time, for a single run of 500 generations. Our experiment replicated and validated their results by averaging each generation’s measurement over 50 trials. We computed the mean of the best individual’s objective fitness in each generation, over all generations and trials. Table 2.1 summarizes results obtained with both algorithms on the FG game. All results were compared using a Welch’s two-sample, one-sided $t$-test at a 99% confidence level. In order to get an idea of focusing in our
Algorithm 2.2 P-PHC - Population-based Pareto Hill Climber Algorithm

1: \( \text{popsize} \leftarrow 50 \)  \\ 2: \( P_C \leftarrow \{\} \quad \triangleright \text{Population of Candidates} \)  \\ 3: \( P_T \leftarrow \{\} \quad \triangleright \text{Population of Tests} \)  \\ 4: for \( \text{popsize times} \) do  \\ 5: \( P_C \leftarrow P_C \cup \{\text{new random individual}\} \)  \\ 6: \( P_T \leftarrow P_T \cup \{\text{new random individual}\} \)  \\ 7: end for  \\ 8: repeat  \\ 9: for each candidate \( \text{parent} \in P_C \) do  \\ 10: \( \text{child} \leftarrow \text{Mutate(} \text{parent} \text{)} \)  \\ 11: \( \text{Vector} \text{ } V_{\text{parent}} \leftarrow \square \)  \\ 12: \( \text{Vector} \text{ } V_{\text{child}} \leftarrow \square \)  \\ 13: for \( i \leftarrow 1 \) to \( \text{popsize} \) do  \\ 14: \( V_{\text{parent}}[i] \leftarrow PF(\text{parent}, P_T[i]) \)  \\ 15: \( V_{\text{child}}[i] \leftarrow PF(\text{child}, P_T[i]) \)  \\ 16: end for  \\ 17: if \( V_{\text{child}} \succ V_{\text{parent}} \) then  \\ 18: \( P_C \leftarrow P_C \setminus \{\text{parent}\} \)  \\ 19: \( P_C \leftarrow P_C \cup \{\text{child}\} \)  \\ 20: end if  \\ 21: for each \( \text{parent} \in P_T \) do  \\ 22: \( \text{child} \leftarrow \text{Mutate(} \text{parent} \text{)} \)  \\ 23: \( F_{\text{child}} \leftarrow 0 \)  \\ 24: \( F_{\text{parent}} \leftarrow 0 \)  \\ 25: for \( (c_1, c_2) \in P_C \times P_C \) do  \\ 26: if \( PF(c_1, \text{parent}) = PF(c_2, \text{parent}) \) then  \\ 27: \( F_{\text{parent}} \leftarrow F_{\text{parent}} + 1 \)  \\ 28: end if  \\ 29: if \( PF(c_1, \text{child}) = PF(c_2, \text{child}) \) then  \\ 30: \( F_{\text{child}} \leftarrow F_{\text{child}} + 1 \)  \\ 31: end if  \\ 32: end for  \\ 33: if \( F_{\text{child}} > F_{\text{parent}} \) then  \\ 34: \( P_T \leftarrow P_T \setminus \{\text{parent}\} \)  \\ 35: \( P_T \leftarrow P_T \cup \{\text{child}\} \)  \\ 36: end if  \\ 37: end for  \\ 38: end for  \\ 39: until Stopping Criterion Met

42
experiments, we computed the mean of the difference between both dimensions in all the individuals from the last generation.

Please note that,

- The stopping criterion is met after a predetermined number of generations (500) elapsed.

- Both algorithms use 100 individuals; P-PHC works with two populations of 50 candidates and 50 tests, while P-CHC uses a single population.

- As a way of introducing mutation, we add +1 or -1 to both dimension with independent equal probability.

- Each candidate solution is a two-dimensional integer vector with values bound in [1, 50]

- These two algorithms do not use any crossover.

- The implementations of all algorithms used in this study are available from our project’s repository\(^3\). We hope that this will facilitate replication studies and encourage other teams to derive their own variants.

As a result of applying both algorithms to the focusing number game, this experiment allowed us to validate previous theoretical results stressing the potential benefits of leveraging the concept of Pareto dominance when comparing candidate solutions in a coevolutionary scenario. Given the previously discussed role played by overspecialization in practice problem generation tasks, this experiment was a natural starting point in identifying relevant guidelines as to how to tune evolutionary techniques to overcome what is one of the defining hindrances to efficient coevolution.

\(^3\)Implementations of algorithms, experiments’ configuration files, and data analysis scripts available in the project’s repository at https://sourceforge.net/projects/evotutoring/
2.2.4 Results

Based on this “dispersion” metric, Table 2.1 also reveals that, as expected, P-CHC generated individuals that are overspecialized; i.e. improved only one of their two dimensions and thus have a higher dispersion measurement. Instead, P-PHC evolved both candidates and tests that featured balanced improvements on both of their dimensions. These results, once again, confirmed on repeated trials, the theoretical expectations expressed in [21].

Together, both series of experiments provided us with a more complete picture: while P-PHC appeared to improve solutions slower than P-CHC, it actually avoided overspecialization. This difference is essential as, once it has fully overspecialized on one dimension, an algorithm such as P-CHC is unlikely to find its way back to the global optimum.

These results therefore provide confidence that theoretically grounded, coevolutionary-based approaches to practice problems generation will be able to address overspecialization occurring in actual student-problem interactions.

Table 2.1: Comparison of overall mean best fitness, and dispersion, for P-CHC and P-PHC on focusing game

<table>
<thead>
<tr>
<th>Algorithm</th>
<th>Individual</th>
<th>Mean best fitness</th>
<th>Dispersion</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td>(\mu)</td>
<td>(\sigma)</td>
</tr>
<tr>
<td>P-CHC</td>
<td>candidate</td>
<td>156.76</td>
<td>83.06</td>
</tr>
<tr>
<td>P-PHC</td>
<td>candidate</td>
<td>98.8</td>
<td>43.45</td>
</tr>
<tr>
<td></td>
<td>test</td>
<td>95.67</td>
<td>42.74</td>
</tr>
</tbody>
</table>

2.3 Experiment#2 - Noisy Elementary Coevolution

2.3.1 Problems

In order to bring our benchmark problems one step closer to the real expectations of evolving practice problems, we introduced noise in the focusing game by inverting its binary outcome with fixed probability \(P_{noise}\).
2.3.2 Algorithms

By varying $P_{\text{noise}}$, we were able to establish how resilient both P-CHC and P-PHC algorithms were. This also allowed us to validate whether relying on Pareto dominance to compare candidate solutions, in non-deterministic scenarios, provided the same benefits predicted by theory in deterministic ones.

2.3.3 Methods

For this experiment, we used the same performance metrics used in Section 2.2.3. The only difference is that we add noise in the focusing game.

2.3.4 Results

Figure 2.1 plots both previously defined metrics (mean overall best fitness and dispersion) for P-PHC and P-CHC. This time, these algorithms are applied to successive noisy versions of the FG game featuring noise levels $P_{\text{noise}}$ ranging from 0% to 100%. We used Tukey multiple comparison of means to ensure that results were statistically significant. All of them are significant at the 99% level up to $P_{\text{noise}} = 90\%$, with the exception of P-PHC-candidate vs. P-PHC-test for $P_{\text{noise}} = 10\%, 20\%$ and $30\%$.

Both algorithms showed a decrease in performance (both in terms of overall mean best fitness and dispersion) as $P_{\text{noise}}$ increased. This was expected since increasing noise makes it more difficult to converge toward appropriate solutions. However, Figure 2.1 also shows that P-PHC responded to increased noise levels better than P-CHC by preserving a higher dispersion. This again increases our confidence that sound coevolutionary-based approaches to generate practice problem will not be particularly sensitive to the inherent noise in student-problem interactions.
Figure 2.1: Experiment #2 - Mean overall best fitness and dispersion for focusing game with P-CHC and P-PHC under different noise levels

2.4 Experiment #3 - Noisy Teacher-Learner Coevolution

2.4.1 Problems

Our third experiment investigated a simple stochastic model of teacher-learner co-evolution. We adopted the same 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 $[1..10]$. Let us note the sum of the genes for a learner as $S_L$, and the sum of the genes for a practice problem as $S_P$.

$S_L$ was interpreted as a measure of the learner’s skill level, expressed as a number of attempts they would take to solve an arbitrary problem. Similarly, $S_P$ was interpreted as a difficulty level for the problem.

Based on these, we defined the outcome of the interaction of a given practice problem with a specific learner, as the number of attempts taken by that learner to solve it;

$$N = \left( \frac{1}{2} S_L (1 + S_P / 40) \right) + \text{rand}(2)$$

where rand(2) returns a random integer in $[0 : 1]$ capturing the variability of students performance. This value was then used to compute the learner and practice problem finesses; $f_P = N$ and $f_L = -N$

While clearly not meant to capture the complex nature of human learners’ performance, this minimal model was sufficient to establish a relevant coevolutionary interaction between our two populations while integrating integer-valued genes. It is interesting to note that, while simplistic, this format actually mimics the genotypes we are currently evolving in our proof of concept implementation which evolves Parsons puzzles, a new type of practice problem for novice programmers.

### 2.4.2 Algorithms

Unlike the focusing game, this model opens the possibility for learners to not necessarily focus on improving their informativeness. It is indeed reasonable to expect that, instead, real students will only focus on improving their skills while practicing. 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 versions, the practice problems' outcome vectors are this time 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 number of attempts, induced by a given learner between each pairs of practice problems.

2.4.3 Methods

All the experimental parameters are same as described in Section 2.2.3 except the followings;

- We add a random integer in the range of \([0 : 1]\) with pay-off function
- We increase genotype size from 2 to 4.

2.4.4 Results

As in our previous experiments, we used objective finesses to track improvements in quality. For both practice problems and learners, we simply summed all genes. Given how practice 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 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 prob-
lems. For this reason, such results are omitted in table 2.2 which focuses solely on tracking Practice Problems improvement.

**Table 2.2**: Experiment #3 - Comparison of overall mean best fitness, and dispersion, between P-CHC and P-PHC on Teacher-Learner problem

<table>
<thead>
<tr>
<th>Algorithm</th>
<th>Mean best fitness</th>
<th>Dispersion</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>$\mu$</td>
<td>$\sigma$</td>
</tr>
<tr>
<td>P-CHC</td>
<td>37.82</td>
<td>0.36</td>
</tr>
<tr>
<td>P-PHC-I</td>
<td>36.68</td>
<td>0.99</td>
</tr>
<tr>
<td>P-PHC-P</td>
<td>34.54</td>
<td>1.91</td>
</tr>
</tbody>
</table>

We performed a Kruskal-Wallis test to ensure of the statistical significance of the results; $\chi^2 = 86.175$ for mean best fitness and $\chi^2 = 833.48$ for dispersion, both significant at $p < 0.01$.

The Informativeness-based variant, P-PHC-I, led to the coevolution of better practice problems when compared to the performance-driven one, P-PHC-P, as suggested by both the mean best fitness and the dispersion of the last generation’s individuals. This is expected since the selective pressure in the learners’ population favored those who were most informative with respect to evolving practice problems.

When comparing P-CHC to P-PHC-I, the dispersion confirms our previous experimental results, suggesting that P-PHC-I prevents overspecialization dynamics. However, the fact that mean best fitness is higher for P-PHC led us to visually inspect the convergence curves plotting the mean best fitness over 50 trials at each generation. These revealed that P-CHC starts improving the best fitness earlier in the run. However, by the end of the run, it fails to finding individuals which fitness is as high as those found by P-PHC-I. As such, the latter converges to better quality solutions within the allotted resources. To confirm these observations, we averaged the fitness of the best individual of the last generation of each of the 50 trials. The mean obtained for P-CHC was 38.4 and the mean for P-PHC-I was 39.12 with a difference confirmed to be statistically significant at $p < 0.001$ with a t-test.
Overall, these findings suggest that recent breakthroughs in coevolutionary computation theory [19] are reasonably applicable to our target application; i.e., competitive co-evolution in a teacher-learner scenario may be improved by prioritizing informativeness over performance in one of the populations. This also brings forth an interesting characteristic of educational applications in general; while coevolutionary theories and simulations agree on the benefits of informativeness over performance, real students are unlikely to strive to improve anything but their performance when practicing with a tutoring system.

2.5 Guidelines for Evolving Practice Problems Based on Elementary Coevolution

We summarize below design guidelines suggested by experimenting on a simplified model of our target application of evolving practice problem.

Experiment #2’s results on the sensitivity to noise levels suggest that, both traditional and Pareto coevolutionary techniques suffer from increasing noise levels. However, Pareto-based approaches feature small differences when we increase from $P_{\text{noise}} = 10\%$ to $P_{\text{noise}} = 50\%$, both in terms of both overall best fitness and dispersion. This suggests an ability to cope with noise in evaluations which is essential in our target application since we expect students’ performance to be affected by external factors.

In line with coevolutionary learning literature [19], Experiment #3 illustrated the potential benefits of prioritizing interactions with informative students in order to efficiently evolve practice problems. However, it is unlikely that real students will strive to improve their informativeness rather than their performance. Therefore, the target application might have to only use a subset of students’ interactions when evaluating practice problems; i.e., interactions with high informativeness ones.

In terms of user fatigue mitigation, the above finding suggests that, in order to coevolve practice problems, we will need to further explore the possibility of pairing them with
learners based on both pedagogical consideration, e.g. appropriate difficulty levels, but also informativeness considerations. We expect these two criteria to require a multi-objective form of evaluation as none is trivially predominant over the other. As a result, evolving practice problems might require an interesting trade-off in terms of how many, and which interactions between our two coevolving populations, should be leveraged. To the best of our knowledge, such a problem has not yet been investigated in the coevolutionary algorithms or interactive evolutionary algorithms literature.

In the subsequent sections we will focus on investigating more complex models of the target application, then leverage resulting guidelines in our proof of concept implementation in order to validate our findings with real students.

2.6 Noisy Teacher-Learner Coevolution Revisited

Preliminary work in investigating the challenges posed by practice problem generation to EA [48] 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. Comp-on-one[34]). 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.

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. [63, 28], or data-mine educational data, e.g. [108]. 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. [118] and established the foundations for the game-theoretic study of coevolutionary learning involving human learners [5]. Even more recently, theoretical results explaining pathological coevolutionary dynamics [18] have helped gain insights about the difficulties encountered in an introductory programming course [145].

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 [48]:

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 [83] 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.
2.7 Experiment#4 - Three Properties of Noisy Teacher-Learner Coevolution

2.7.1 Problems

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 coevolution that was used in [48].

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. The latter is meant to have an EA variant evolve specific practice problems for novice programmers, known as Parsons puzzles [100]. 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 distractors, 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 distractors. 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.

\[
N = S_L + S_P + \text{rand}(r)
\]  

(2.1)

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 finesses 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.

2.7.2 Algorithms

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

2.7.3 Methods

All the experimental parameters are same as described in Section 2.4.3 except we choose a very simplistic fitness function to maintain arms-race between practice problem and learners.
2.7.4 Results

As in our previous experiments, we used objective finesses 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 2.3 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 [48].

2.7.5 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...
Table 2.3: Performance of P-CHC and P-PHC-I under pathology (overspecialization), noise and different bounds on two pay off functions defined in Equations 2.1 and 1.1.

<table>
<thead>
<tr>
<th>Property</th>
<th>Fitness</th>
<th>Bounded</th>
<th>Mean Objective Fitness</th>
<th>Mean Dispersion</th>
</tr>
</thead>
<tbody>
<tr>
<td>Pathology</td>
<td>Eq 1.1</td>
<td>[1, 10]</td>
<td>35.99 35.41</td>
<td>7.13 7.10</td>
</tr>
<tr>
<td></td>
<td></td>
<td>[1, 100]</td>
<td>67.60 45.52</td>
<td>31.42 7.51</td>
</tr>
<tr>
<td></td>
<td>Eq 2.1a</td>
<td>[1, 10]</td>
<td>38.50 38.47</td>
<td>3.66 3.71</td>
</tr>
<tr>
<td></td>
<td></td>
<td>[1, 100]</td>
<td>82.67 81.70</td>
<td>12.05 11.86</td>
</tr>
<tr>
<td>Noise</td>
<td>Eq 1.1a</td>
<td>[1, 10]</td>
<td>36.75 35.18</td>
<td>7.20 6.89</td>
</tr>
<tr>
<td></td>
<td></td>
<td>[1, 100]</td>
<td>63.94 38.60</td>
<td>20.13 7.65</td>
</tr>
<tr>
<td></td>
<td>Eq 2.1</td>
<td>[1, 10]</td>
<td>38.72 38.47</td>
<td>1.60 3.72</td>
</tr>
<tr>
<td></td>
<td></td>
<td>[1, 100]</td>
<td>82.19 83.50</td>
<td>15.16 11.96</td>
</tr>
</tbody>
</table>

The results presented in row “RL #8” from Table 2.3 suggests that our target application can benefit from Pareto coevolution when $N_G = 100$. Equation 2.1 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 2.3, 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.

2.7.6 Analysis of the Interaction in Experiment #4

We propose to revisit previous section’s findings in order to identify the dynamics responsible for the results detailed in Table 2.3. To this end, we identify and quantify three properties that explain the algorithms’ behavior for trivial and Pareto-based coevolution of practice problems.
2.7.6.1 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 2.1). 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;

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

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

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, $P_{S_c > S_p}^\mu$, “+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 $P_{S_c > S_p}^\mu$ is $P(X > N/2)$.

$$P_{S_c > S_p}^\mu = P(X = \lceil N/2 + 0.5 \rceil) + \ldots + P(X = N)$$

$$= \left( \sum_{i = \lceil N/2 + 0.5 \rceil}^{N} \binom{N}{i} \right) / 2^N$$

58
To satisfy $S_c > S_p$, the frequency of “+1” mutation operation need to exceed than that of “-1” operation. So, this is basically a count of different order-less arrangement of “+1” out of $2^N$ possible combination of “+1” and “-1” operations. 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 $S_p = S_c$. ; $P^\mu_S = 0$.

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

2.7.6.2 Property 2 - rand(r) Effect

As mentioned earlier, the probability that $F_c > F_p$ is also affected by the $\text{rand}(r)$ term found in Equation 2.1. 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 $P(Y \leq y) = \sum_{y=1}^{r} P(Y = y) = \frac{1}{r}$.
Since \( p_2(y) \) and \( F_2(y) \) are constants and dependent on \( r \) only, we denote them \( p_2 \) and \( F_2 \) respectively.

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

Proof:

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

\[
= \frac{r - 1}{r^2} + \frac{r - 2}{r^2} + ... + \frac{r - x}{r^2}
\]

\[
= \frac{1}{r^2}(r - 1 + r - 2 + r - 3 + ... + r - x)
\]

\[
= \frac{1}{r^2}(xr - \frac{x(x + 1)}{2})
\]

\[
x \frac{r}{r} - \frac{x(x + 1)}{2r^2}
\]

\[
F_2 = \frac{1}{r^2} + \frac{1}{r^2} + ... + \frac{1}{r^2}
\]

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

\[
= \frac{1}{r} + \frac{r - 1}{r} - \frac{(r - 1)(r - 1 + 1)}{2r^2}
\]

\[
= \frac{1}{r} + \frac{r^2 - r}{2r^2}
\]

\[
= \frac{1}{r} + \frac{r^2 - r}{2r^2}
\]

\[
= \frac{1}{2} + \frac{1}{2r}
\]

\( \text{rand}(r) \) 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 \( p_a = p_b \), \( \frac{r^2 - x}{2} \) pairs follow \( p_a > p_b \) and rest of the \( \frac{r^2 - x}{2} \) are obliged by \( p_a < p_b \).
As we did with the $\delta$ values for the Mutation Effect, we define $\delta_{\text{rand}}^{\text{min}} = 1$ and $\delta_{\text{rand}}^{\text{max}} = r - 1$. In addition, $\delta_{\text{rand}}^{\text{rand}}$ corresponds to $p_a = p_b$. So, when $S_c > S_p$ the difference is bounded by $\delta_{\text{rand}}^{\mu} + \delta_{\text{rand}}^{\text{rand}} = 2 + 0 = 2 \leq \delta_{\text{max}}^{\mu} + \delta_{\text{max}}^{\text{rand}} = 4 + (r - 1) = 3 + r = 5$.

### 2.7.6.3 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 $\text{rand}(r)$ Effect. For instance, if $S_c > S_p$ by $\delta$ 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 $\text{rand}(r)$ 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 ?? 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 $1 \leq k_1 \leq r - 1$;

$$P_{F_c > F_p} \geq 0.3125 \times P_{R_c \geq R_p}^{\text{rand}} + 0.6875 \times F_1(k_1) \quad (2.2)$$
Similarly, for \(k_1, k_2 \in \delta\):

\[
P_{F_c=F_p} = \sum_{i=1}^{r} \frac{1}{r^2} - \frac{k_1 + k_2}{r^2}
\]  

(2.3)

and

\[
P_{F_c<F_p} = 1 - (P_{F_c>F_p} + P_{F_c=F_p})
\]  

(2.4)

*Proof*: We can derive the following equation for \(P_{F_c>F_p}\).

\[
P_{F_c>F_p} = 0.3125 \times F_1(k) + 0.3125 \times F_1(k_1)
\]

\[
+ 0.3125 \times F_2(r) + 0.375 \times F_1(k_1)
\]

\[
+ 0.3125 \times \frac{r^2 - r}{2r^2} - F_2(k_2), \text{ where}
\]

\[
1 \leq k < \delta, 1 \leq k_1 \leq r - 1, 1 \leq k_2 \leq \delta,
\]

\[
\geq 0.3125 \times \left(\frac{k}{r} - \frac{k(k + 1)}{2r^2}\right)
\]

\[
+ 0.6875 \times \frac{k}{r} - \frac{k(k + 1)}{2r^2}
\]

\[
+ 0.3125 \times \frac{1}{r} + 0.3125 \times \left(\frac{r^2 - r}{2r^2} - \frac{k}{r} + \frac{k(k + 1)}{2r^2}\right)
\]

\[
= 0.3125 \times \left(\frac{1}{r} + \frac{r^2 - r}{2r^2}\right) + 0.6875 \times F_1(k_1)
\]

\[
= 0.3125 \times P_{R_c > R_p}^{rand} + 0.6875 \times F_1(k_1)
\]

\[
= 0.15625 + 0.15625 \times \frac{1}{r} + 0.6875 \times F_1(k_1)
\]
To satisfy $F_c = F_p$, the three conditions listed in can be summarized as follows for $k_3 = p1(k) \times k + p2 \times k2$.

\[
P_{F_c=F_p} = 0.375 \times \frac{1}{r} + 0.3125 \times \frac{r - k_1}{r^2} + 0.3125 \times \frac{r - k_2}{r^2}, \text{where } k_1, k_2 \in \delta
\]

\[
= 0.375 \times \frac{1}{r} + 0.3125 \times \left( \frac{2r - (k_1 + k_2)}{r^2} \right)
\]

\[
= 0.375 \times \frac{1}{r} + 0.625 \times \frac{1}{r} - 0.3125 \times \frac{k_1 + k_2}{r^2}
\]

\[
= 0.6875 \times \frac{1}{r} + 0.3125 \times \left( \frac{1}{r} - \frac{k_1 + k_2}{r^2} \right)
\]

\[
= \frac{1}{r} - 0.3125 \times \frac{k_1 + k_2}{r^2}
\]

\[
= F_2(r) - 0.3125 \times \frac{k_1 + k_2}{r^2}
\]

\[
= F_2 - 0.3125 \times \frac{k_3}{r^2}
\]

\[
= \sum_{i=1}^{r} \frac{1}{r^2} - 0.3125 \times \frac{k_3}{r^2}
\]

$P_{F_c=F_p} = \sum_{i=1}^{r} \frac{1}{r^2} - 0.3125 \times \frac{k_3}{r^2}$ The two properties, Mutation and rand(r) effect, contribute to the number of attempts that a parent $p$ or a child $c$ take when interacting with a learner. The third one, Combined effect, basically measures the total outcome of the two independent events caused by the previous two effects in that interaction. The first event produced by Mutation effect is performed for a single time within a generation for a specific $(p, c)$ pair. However, the second event generated by rand(r) effect will be initiated number of learner(s) times the pair is interacting. So, it is task of the Combined effect to quantify the value of those two independent events, when $(p, c)$ interacts with learner(s). In short, these three properties tell us the probabilistic quantification of number of attempts of practice problems when they interact with learner(s).
2.7.7 Experimental Validation of the Three Properties

So, we can monitor the behavior of practice problems and learners interaction using those properties. As we can easily compute the probabilities of all possible ways of win/loss/draw of a single interaction of the pair \((p, c)\) it is time to validate those theoretical results. To do so, we perform experiments on trivial and Pareto coevolution that uses Equation 2.1 as pay-off function and explained whether the experimental results meet the theoretical expectation.

2.7.7.1 Experimental Parameters

To monitor the above three properties of the algorithms, we define the following metrics and run experiments for different population size \(M = \{1, 2, 4, 8, 16, 32, 64, 128\}\). All the experimental parameters are same as in Section 2.2.3 except the following,

- population size is chosen from \(M\).
- minimum and maximum genome size is changed to large positive integer such that no gene can be negative due to behavior of \(+1/−1\) mutation operator.
- mutation operation is unbounded i.e. it does not limit gene values if exceeds maximum or reaches minimum.

2.7.7.2 Performance Metrics

Three new metrics \(\%gt, \%lt\) and \(\%eq\) are, originally motivated by the probabilities described in those three properties, for trivial coevolution and their corresponding metrics \(\%ds, \%dt\) and \(\%non\) are for Pareto coevolution introduced. They are the determinant to measure the total probabilities of combined effect for three conditional relationships between \(F_c^p\) and \(F_p^p\).
• $(\% ds)$ is how much in percent $c$ averagely dominate $p$ per generation $(G)$;

$$ds = \frac{\sum_{i=1}^{[G \times M]} f_{c\succ p}^i}{[M \times G]}$$

where $f_{\text{condition}}^g$ is the result, $g = \{\text{win}, \text{loss}, \text{draw}\}$, in $i^{th}$ interaction satisfying the constraint in $\text{condition}$.

• In a similar fashion, we define $\% dt$ and $\% non$. The conditions for them are $c \prec p$ and $c \simeq p$ respectively.

• $(\% gt)$ is Same as we did for $\% ds$ but the condition is changed from $c \succ p$ to $c > p$

• Similarly, $(\% lt)$ and $(\% eq)$ are defined. The conditions for them are $c < p$ and $c = p$ respectively.

### 2.7.7.3 $(M, r = 2)$ Analysis

To understand the evolution of practice problem based on pay off function defined in Equation 2.1 for a specific $r = 2$, we run experiments for different $M$ values and monitor how theoretical expectation derived in respective properties deviates from experimental results.

$(M, r) = (1, 2)$ row of Table 2.4 confirms that the calculated probabilities from Table ?? and experimental results are almost same. If $c$ wins, it does by $\delta$ in $\text{Mutation Effect}$ and can get either 0 or 1 in $\text{rand}(r)$ Effect. So, a $c$ win in $\text{Combined Effect}$ happens with $S_c^P = S_p^P + A$. Note that, if $c$ wins in mutation then it is the ultimate winner in that interaction, no matter what the result in $\text{rand}(r = 2)$ because $\delta > \{\delta_{\text{rand}}, \delta_{\text{rand}}\}$. The same explanation is applicable when a $c$ loss happens in mutation. The draw occurrence happens only when a $c$ draws on both mutation and random effect because winning by $\delta$ in $\text{Mutation Effect}$ can’t be neutralized in $\text{rand}(r)Effect$ and vice versa.
To understand, let us examine the three behavioral probabilities of the algorithms in Combined Effect. Firstly, $0.3125 \times P_{R_c \geq R_p}^{rand}$ in $\sum F_c^P > F_p^P$ captures the probabilities of cases where $c$ wins in Mutation Effect and either draws or wins in rand(r) Effect whereas the other term $0.6875 \times F_1(k_1)$ gives the probability of winning/loosing by $\delta$ in Mutation Effect but loosing/wining by $\delta - 1$ or $\delta + 1$ respectively in rand(r) Effect. Here, $P_{R_c \geq R_p}^{rand} = \frac{1}{2} + \frac{1}{2 \times 2} = 0.75$ and $F_1(k_1) = F_1(1) = 0.25$. Hence $P_{R_c \geq R_p}^{rand} \geq 0.40625$. Secondly, $F_2$ in $P_{R_c = R_p}^{rand}$ represents the draw in rand(r) Effect no matter what the decision is in Mutation Effect. That’s why, the second term $0.3125 \times \frac{k_3}{r}$ subtracts the other cases where a child $c$ does not win or loss by same $\delta$ in the mutation and random effect. In case of $r = 2$, $k_1 = k_2 = \delta_{min} = 2$ and so $P_{R_c = R_p}^{rand} = 0.1875$. Lastly, $\sum F_c^P < F_p^P = 1 - (0.40625 + 0.1875) = 0.40625$.

However, when we increase $M$ it also increases number of interactions because each $c$ and $p$ interacts with same set of learners to build their corresponding fitness vectors. The elements of the vectors are added or Pareto compared based on which algorithm, trivial or Pareto coevolution, we analysis. In such scenario, the algorithms need to consider several cases to be summation of fitness vectors between $c$ and $p$ becomes equal for trivial coevolution or to be non dominated for Pareto coevolution.

- $F_c > F_p$ by $A$ in one interaction need to be neutralized by another interaction.
- $F_c = F_p$ indicates that draw events happen in both mutation and random effect.
- $F_c > F_p$ for some $x$ interactions need to be neutralized by another $y$ interactions.

The probability that, for a given interaction, the outcome ($c$ win by $z$) of Combined Effect is neutralized by the outcome ($c$ loss by $z$) of another interaction is $0.11$. This is because $c$ can win, lose or get a draw by $y \in A \cup \{0\}$. On top of that, this $0.125$ probability is also dependent on $\sum F_c > F_p$ if it is win neutralization or $\sum F_c < F_p$ for loss and $\sum F_c = F_p$ for draw. If there is a total of $M > 1$ interactions, then it requires that the result of $M/2$ independent
Table 2.4: Monitoring coevolution of practice problem for trivial and Pareto coevolution - P-CHC, P-PHC-P - under mutation, rand(r) and combined effect

<table>
<thead>
<tr>
<th>$(M,r)$</th>
<th>P-CHC</th>
<th>P-PHC-P</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>$%gt$</td>
<td>$%lt$</td>
</tr>
<tr>
<td>(1, 2)</td>
<td>40.63</td>
<td>40.56</td>
</tr>
<tr>
<td>(2, 2)</td>
<td>43.22</td>
<td>42.40</td>
</tr>
<tr>
<td>(4, 2)</td>
<td>45.00</td>
<td>44.82</td>
</tr>
<tr>
<td>(8, 2)</td>
<td>46.18</td>
<td>46.33</td>
</tr>
<tr>
<td>(16, 2)</td>
<td>47.29</td>
<td>47.50</td>
</tr>
<tr>
<td>(32, 2)</td>
<td>48.13</td>
<td>48.15</td>
</tr>
<tr>
<td>(64, 2)</td>
<td>48.54</td>
<td>48.84</td>
</tr>
<tr>
<td>(128, 2)</td>
<td>49.13</td>
<td>49.03</td>
</tr>
</tbody>
</table>

unique outcomes need to cancel the result of rest of the $M/2$ independent unique outcomes to validate only the case of $F_c = F_p$ let alone the other two cases mentioned above. As a result, “$\%eq$” decreases when $M$ increases which leads both “$\%gt$” and “$\%lt$” to increase.

On the other hand, when fitness vectors are Pareto compared, it is sufficient to flip the win of $c$ by a value in one interaction by the loss by same value in another interaction within same generation. If we combine the probability of the three cases above, it can be seen that this flipping possibility increases as $M$ increases. As a result, the value of $\%non$ increases for Pareto-based coevolution but the other two other metrics decrease.

2.7.7.4 Implications Regarding Design Guidelines

The analysis of Exp#4 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 EMOO where problems with more than about five objectives are much more difficult to tackle by state of the art algorithms [56]. 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[80].
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.

2.8 Experiment#5 - Genes' Bounds

2.8.1 Problems

We use $G_{one}(C, T)$ where C and T represent practice problem and learner respectively.

2.8.2 Algorithms

We measure performance of P-CHC, P-PHC-P and P-PHC-I, for two different payoff functions defined in Equations 2.1 and 1.1 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. [18] or [34]. The latter does not limit the values taken by a gene. disengagements etc.

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.

2.8.3 Results

Table 2.3, “RL #1-4” show the performance of P-CHC and P-PHC-I under two payoff functions. Let us label Equation 2.1a as the version of Equation 2.1 in which the $rand(r)$
The term is discarded. The result for Comp-on-one indicates that when we use a low bound value for genes, we do not need P-PHC abilities to overcome overspecialization.

### 2.8.4 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 [18] 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.

### 2.9 Experiment #6 - Noise and Genes’ Bounds

#### 2.9.1 Problems and Algorithms

In this experiment, we use both Equation 2.1 and a noisy version of Equation 1.1 in which the outcome is flipped (+1 by −1 or vice 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 $N_G = 100$.

#### 2.9.2 Results

Table 2.3, “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 over-
specialization for the noisy Compare-on-one. It is also better when genes can take more values.

2.9.3 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.

2.10 Guidelines for Evolving Practice Problem by Noisy Teacher-Learner Co-evolution

This three experiments allowed us to extend previous work focused on identifying design guidelines for leveraging evolutionary algorithms to generate practice problems. Experiment #4 confirmed previous results [48] 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 [48] 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 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 #5 and #6 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 P-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.

### 2.11 Minimal Guidelines for Interactive Noisy Teacher-Learner CCoEA

Based on the literature reviews on Chapter 1 and the experimental results obtained in Experiment #1 - #6, we propose minimal design guidelines to evolve candidate solutions that require human evaluations. We also discussed the challenges found in IEA and CCoEA and also showed the possible opportunities for designing an interactive competitive coevolutionary algorithm.

It is interesting to explore what are the design guidelines for a system that would leverage coevolution to evolve candidate solutions that require human evaluations. Such
system would present the challenge of needing to handle coevolutionary pathologies, mitigate user fatigue, but also be resilient to the inherently noisy interaction outcomes.

### 2.11.1 Measure the Progress in Coevolution

It is very important that such system progresses to better solutions over time. However, the tracking of progress in coevolution is not the same as that of traditional optimization where solutions are evolved targeting a static/dynamic minimization/maximization function. This is because the fitness is context sensitive in coevolution. For example, a predator’s win is regarded as a prey’s defeat. There are basically two types of fitness; objective - an external measure of tracking progress of interaction dynamics - and subjective - outcome obtained as a result of interaction. In a number game, sum of genes is an example of objective fitness whereas $+1/ -1$ i.e., interaction outcome is regarded as subjective fitness. We can measure the progress of coevolutionary setup using number games like benchmark problems by objective fitness. But how can we measure such progress when solutions interact with human evaluator, and more importantly genotype and phenotype of candidate solutions are not same? We don’t have any objective measures in such situations. In this case, we need to measure and validate the progress based on only subjective fitness and qualitatively, if possible.

### 2.11.2 Design Solution Concepts

Another important aspect is to study the pros and cons of different solution concepts available for competitive coevolution. For example, aggregation solution concept provides equal weights to each objective whereas Pareto optimal solution concept respects the weight of each objectives. However, most of the solutions become Pareto non dominated when number of objectives increases. Design of solution concept or choosing the right from existing one also require to consider cost, time incurred into an evaluation.
2.11.3 Consider the Role of Solution Concepts

Pareto dominance based solution concept is proved to tackle *overspecialization* in coevolution [21] better than aggregation. In addition, this solution concept also prohibits coevolutionary search to fall into cycling.

Avoidance of certain pathologies like *overspecialization* and *cycling* can be shown in a coevolutionary setup for number games. In case of such application, detecting the presence of coevolutionary pathology is an open question that requires further investigation. Though designing/choosing solution concept depends on other criteria also (such as, do we study the problem in terms of mono/multi/many objective optimization), considering the role of solution concept in avoidance of pathology should be studied significantly.

2.11.4 Determine Minimum Number of Evaluations

Determining the number of unique evaluations for every single solution is a critical step regardless the way (i.e., mono/multi/many objective optimization) we study the problem. The system should be observed under a coevolutionary setup to see the effect of minimum number of evaluations. As mentioned earlier, test-based coevolution requires as many evaluations as population size. However, this huge number of evaluations for a single candidate solution is in-feasible for such applications due to user fatigue, noise and also arises ethical concerns. But a minimum number of evaluations need to be determined.

2.11.5 Develop the Outcome of an Interaction

In the case of number games, interaction outcome can be designed to target exhibiting certain pathology. Similarly, we need an outcome measurement when an evaluator evaluates a solution. The interaction outcome is application dependent. For example, in case of practice-
problem vs student interaction number of actions required to solve a practice problem, time etc. can be used to generate the outcome.

2.11.6 Develop Techniques to Mitigate User Fatigue

As soon as we combine CCoEA techniques with fitness evaluations based on interactions with actual students, it becomes imperative, as with most IEAs, to mitigate efficiently the user fatigue phenomenon. While developing or choosing mitigation techniques for any educational application, we need minimum number of student evaluation for each practice problem to accelerate the evolution. As with many population-based algorithms, it is essential to balance the need for exploration of new practice problems and exploitation of those that show promising evaluations already.

2.11.7 Focus on Noisy Evaluations

Account for noise in the evaluation process as it relies on human interactions. In case of practice-problem evolution by student interaction, student may try randomly all possible way to solve a practice problem mimicking random search. We need to be careful about noisy evaluation when validating the evolution.

2.11.8 Validate the Design Guidelines

The validation of design guideline is required to add or modify the guideline directions incrementally. To do so, we need the evaluation of practice problems from real students. Analyzing student’s interaction data and log will provide an insight about validation of each guidelines. In such applications, some phenomena like pathologies may not be measured directly. However, deploying the system with best possible design guideline is a wise move towards evolving the practice problems by student’s interaction.
2.11.9 Validate the Evolution

The validation of evolved solution needs to be justified both quantitatively and qualitatively. As the fitness of the solutions is subjective, it is important to consider evaluator’s cognition, bias etc. while interpreting evolved solutions qualitatively. In the case of quantitative, we can perform a CDE analysis on the interaction matrix. This analysis will provide insights into the underlying difficulties of the evolved practice problems. In addition, the evolved solutions need to be reviewed manually. Suppose, we evolve 10 practice-problem after a semester then how shall we know that the evolved practice problems are informative, harder or challenging than the problems in the beginning of the semester? Some sorts of qualitative measure and also feedback from the evaluators may help in this regard.
Chapter 3: EvoParsons: Evolving Parsons Puzzles Based on Elementary Coevolution

3.1 Evolutionary Parsons Puzzle

This chapter was published in Journal of GPEM 1. The permission is included in Appendix A.

Parsons puzzles are a relatively new type of practice problem aimed at helping novice programmers [100]. Each puzzle is made of a plain English description of a program’s requirements, accompanied by its implementation. This “reference solution” is then scrambled, usually by isolating each line of code into a “fragment” and shuffling them. At the same time, a few of these fragments are selected and transformed into so-called “distractor fragments”. Each distractor fragment features syntactical or logical errors; e.g., forgetting a ‘;’ after statement or transforming a ‘<’ into ‘<=’ in a conditional expression. The requirements are then presented to the student, along with the shuffled list of both original and distractor fragments. The learner is expected to identify the valid fragments and re-order them so as to reconstitute the original solution. Since their inception, several independent studies have repeatedly illustrated the benefits of Parsons puzzles; e.g., [39, 64, 57, 73, 42, 92].


---

sequences to be sorted. This scenario has since been referred to as the teacher-learner framework. The random input sequences act as “teachers” to the sorting networks that play the role of “learners”, progressively improving their ability to sort any input sequence. Interestingly, the so-called “coevolutionary pathologies” [46] that have been observed and analyzed in such applications are analogous to phenomena observed in educational settings. For instance, the loss of gradient between a population of sorting networks and one of testing sequences [59] finds a natural counterpart in the classroom when instructors’ hand-designed assignments are too difficult or trivial for the students as a whole. In pedagogical terms, we would consider that such assignments fall outside the students’ zone of proximal development [134]. Similarly, it is notoriously difficult to measure objective progress in any non-trivial coevolutionary dynamics. A candidate solution A might emerge as superior in its population, only to be superseded, a few generations later, by a solution B that will, later again, be outperformed by a solution C. However, it is well-known that this does not prevent a re-occurrence of solution A at a later point as it might prove better than C. Such cycles can occur easily in non-trivial coevolutionary dynamics. While the algorithm reports finding better and better solutions every few generations, an external observer with an objective way to measure the quality of solutions might notice that we are simply rediscovering previous solutions.

In our specific application, we will show that EvoParsons can be categorized as an instance of test-based Pareto coevolution[140]. This allows us to leverage recent theoretical advances in coevolutionary computation theory to extract and analyze the underlying dimensions of the problem vs. student interaction matrix [23, 145, 24, 65] and use them to estimate the relevance of the evolved practice problems.

In addition to the above-mentioned coevolutionary aspects, evolving practice problems for a student population also entails challenges commonly found in IEAs[125]. The fitness of the evolved practice problems is derived from their interaction with students.
Such interactions are particularly time-consuming, thus driving the need to make the best out of each of them in order to assess the quality of our Parsons puzzles. Furthermore, human-based evaluations are inherently noisy and subject to the well-studied user fatigue phenomenon [123, 137, 138]. As such, it is essential for us to also investigate selection policies for dispatching Parsons puzzles to students that minimize unnecessary evaluations.

In this context, the goal of this work is therefore to apply recent advances in coevolutionary computation and interactive evolutionary algorithm techniques to automatically generate pedagogically-sound Parsons puzzles for a population of novice programmers. We propose to do so in two steps.

First, we identify guidelines necessary to successfully apply coevolutionary algorithms to generate Parsons puzzles. To do so, we study the coevolutionary dynamics and pathologies [101] that coevolutionary interactions exhibit. This leads us to identify design guidelines for the EvoParsons system, based on factors such as algorithm, noise, genes encoding, and coevolutionary pathologies. Previous works [48] [49] already identified appropriate algorithms and estimated their performance under noisy evaluations and coevolutionary pathologies. In this study, we summarize and extend these results by proposing new metrics to evaluate these guidelines. The metrics are derived from the usage of the aforementioned dimension extraction techniques.

Second, we leverage the above guidelines to implement the EvoParsons system, use it on actual students, analyze the resulting interaction matrix, and investigate the relevance of the evolved Parsons puzzles in terms of the interaction matrix’s underlying dimensions. Although this preliminary experimental evaluation on real students is not designed to measure direct pedagogical impact on learners, it assists educators in automatically identifying relevant practice problems. Indeed, the distractors embodied in evolved Parsons puzzles represent the misconceptions with which students struggled [25], and identifying such misconceptions is the first step toward building Concept Inventories (CI).
Concept Inventories have been developed for digital logic [58], discrete mathematics [2], operating systems [142], algorithms and data structures [131], and algorithms analysis [43]. With regard to introductory programming, a number of studies have refined topics by their importance, difficulty, and potential misconceptions. In [50], authors identified hard concepts in programming, then ranked them by importance and difficulty. They later added to this work with a Delphi study for three different introductory computing topics [51]. In [69] and [26], possible student misconceptions were identified and leveraged. While these studies demonstrated significant impact, current methods for building concept inventories lack consensus, have difficulties identifying appropriate distractors, and are overall resource-intensive to apply. There is therefore an interest in relying instead on data-driven approaches that would, as is the case in EvoParsons, rely on student vs. Parsons puzzle interaction logs in order to identify such concepts. Such approaches have the potential to contribute to both automate (at least partially) the concept inventory design process, and increase its objectivity.

3.2 Experiment #7 - Evaluating Evolved Practice Problems by Simulation

The previous section demonstrates what coevolutionary techniques can be successfully applied to a teacher-learner scenario. This particular scenario was designed to integrate, into a simulation, some of the general characteristics that we will encounter when evolving Parsons puzzles for actual students. We now turn our attention to the expectations we have for the kind of Parsons puzzles that such a system should be able to produce.

To this end, we use simulations again, but this time we also conduct a separate analysis on the practice problems vs. learners interactions. Please note that practice problems evolve by optimizing their performance against the learners. Therefore, learners are treated as objectives in the same sense as intended in the multi-objectives and many-objectives optimization literature[17]. This allows us to leverage the CDE described in the next subsection.
Such an analysis results in a coordinate system that identifies the dimensions of the under-
lying interaction space—i.e., the “types” of practice problems—and ranks them along each
dimension using Pareto dominance.

Such information sets the “gold standard” in terms of available information regarding
the interactions that took place. This is due to the fact that CDE removes redundant
solutions and groups the underlying interaction space into hierarchy of solutions based on
their performance against opponents. This representation can also be used to guide the
coevolutionary search to ideal evaluation of candidate solutions [33]. However, as we will
discuss below, CDE analyses may be difficult to conduct in some real world scenarios. We
therefore propose to validate the outcomes of applying P-PHC-P by estimating how much the
practice problems found in its last generation reflect the information that would be provided
by a full CDE analysis. This is done with the understanding that, unlike the CDE analysis,
P-PHC-P will be practically applicable when dealing with actual students.

3.2.1 Problems and Algorithms

We use the same simplified Coevolutionary Teacher Learner problem with bounded
genes that we previously defined in Section 2.4.3. We also use P-PHC-P to coevolve practice
problems and learners, as previously defined. As we do so, the algorithm explores a subset of
the interaction space between all possible encoded learners and all possible encoded practice
problems.

3.2.1.1 Evaluating Evolved Practice Problems on CDE Metrics

A CDE analysis provides us with a terse, yet exhaustive, view of the information
contained in the full interaction space. In comparison, the P-PHC-P heuristically searches
through a space approximating the full interaction space since it only considers a subset of
all possible practice problems and learners genotypes.
While this makes P-PHC-P practically usable when we switch to actual students instead of simulations, it is desirable for its population to converge toward practice problems that overlap those that would be identified by CDE. More specifically, its last generation should contain practice problems representing the various underlying dimensions of the interaction space that are also identified by a CDE analysis. Similarly, each of these representatives should rank high in their respective dimensions in order to capture all the information they offer.

Establishing the presence of such individuals in P-PHC-P’s last generation thus provides us with a quantitative measure of the quality of information obtained by simply co-evolving practice problems, as opposed to running a CDE analysis on the full interaction space. Such a measure is of particular interest if we also take into consideration how much of the interaction space P-PHC-P sampled, in comparison to CDE’s usage of the entire available information.

3.2.2 Methods

All experimental parameters are kept the same as described in the previous section for our minimal practice problem - learner model with the exception that we do not consider the effect of external noise. The reasoning behind this decision is that, in our target application, the noise is embedded into fitness function. The purpose of adding noise was only to establish which algorithm would be best to use in a noisy environment (P-PHC-P).

We define several quantitative metrics allowing us to establish how well P-PHC-P performs:

- Our first concern is to determine how many of the dimensions that are identified by a CDE analysis are also found by P-PHC-P. Such a dimension is found by P-PHC-P if it has at least one representative in the last generation. To this end, we define Dimension
Coverage (DC),

\[ DC = \frac{k}{n}, \]

where \( n \) is the total number of dimensions found by CDE and \( k \) is the number of unique CDE dimensions represented in P-PHC-P’s last generation. Here, a dimension can be thought as a concept. Each individual in that dimension represents a hierarchy of practice problems that distinguish student’s performance.

- Once we have established DC, the next step is to measure how close the representatives from P-PHC-P’s last generation are to the extrema revealed on each dimension by CDE. By definition, CDE ranks practice problems on each dimension based on Pareto-dominance. We are interested in the CDE ranking that would be assigned to the representative of each dimension found by P-PHC-P as a way to establish whether we converge toward the most informative practice problems for each dimension. We therefore define Average Rank of Representatives (ARR),

\[ ARR = \frac{\sum_{k=1}^{k} \max P(D_k, p)}{|D_k|}, \]

where \( P(D_k, p) \) returns the rank assigned by CDE along dimension \( D_k \) to practice problem \( p \), \( p \) is an individual from the last generation of P-PHC-P, \( |D_k| \) is the length of dimension \( D_k \) denoting number of unique puzzle in the dimension, and \( D_k \) represents only those dimensions where at least one \( p \) from P-PHC-P’s last generation is found. The extrema practice problems, in each dimension, are the most interesting, informative, and difficult. Investigating concepts and distractors embedded into those practice problems has potential to support the automated definition of concept inventories.

- A composite practice problem is one in which position in a CDE coordinate system may be expressed as a linear combination of the positions of existing practice problems that
are already on some CDE dimensions. In terms of CDE analysis, such practice problems are discarded as they are redundant with the already constructed coordinate system. We are interested in whether P-PHC-P keeps such redundant practice problems or eliminates them. Redundancy can be defined as the number of practice problems in P-PHC-P’s last generation that would have been discarded by CDE. We define this metric as Redundancy (R).

\[ R = \frac{n_r}{N}, \]

where \( n_r \) is the number of redundant practice problems and \( N \) is P-PHC-P’s population size. Let a practice problem A fail group of learners \( G_1 \) who struggle with \textit{loops} (for instance). Another practice problem B fails a group of learners \( G_2 \) who are not good at using \textit{conditional} statements. Then the practice problem \( C \), that fails both \( G_1 \) and \( G_2 \), is redundant because it does not differentiate the students who are struggling to master each of the topics (same as two dimensions in Figure ??).

A high redundancy means that available population slots have been wastefully dedicated to practice problems that overlap each other in terms of informativeness about students’ performance.

- In some cases, practice problems may be discarded during a CDE analysis because they feature the exact same outcome vector than another one that has been already integrated in the CDE coordinate system. We therefore also measure the occurrence of such practice problems in the last generation as follows (Duplication (D))

\[ D = \frac{n_d}{N}, \]

where \( n_d \) is the total number of duplicate practice problems found and \( N \) the population size for P-PHC-P. Duplicate practice problems in a dimension are non-comparable
in terms of their difficulties, distractors and inherent context. If the evolutionary algorithm converges to a final population featuring many duplicate practice problems, then there is a lack of efficient diversity preservation during convergence. Such a system may converge to local optima i.e., evolve some mediocre practice problems. In addition, Duplicates and Redundant practice problems waste population slots that could be used to sample more dimensions, or better representatives in each of the already represented dimensions. However, redundant practice problems are an even bigger issue than duplicate ones.

- We know that the full CDE analysis benefits from complete information on every single possible student vs. puzzle interaction. On the other hand, P-PHC-P only explores a limited subset of these interactions. Therefore, all other measurements proposed in this section only make sense if we also consider the difference in resources consumed by each approach. To this end, we define Uncovered Search Space (USS),

\[ USS = 1 - \frac{x}{T}, \]

i.e., the complement of the ratio of the number of unique puzzles \(x\) explored by P-PHC-P over the total number of practice problems \(T\) in the search space.

Please note that all of the above metrics are defined in \([0,1]\). For DC, ARR and USS, higher values indicate better performance of the algorithm. Since R and D measure different forms of “waste of population slots”, smaller values indicate better performance of the algorithm.

3.2.3 Results

The evolved puzzles that are found in the last generation of P-PHC-P are the result of a coevolutionary search that considered only a limited subset of the full underlying search
space. On the other hand, a CDE analysis benefits from exhaustive information regarding the interactions of all possible puzzles with all possible learners. As a result, our comparison aims at establishing whether P-PHC-P offers a reasonable approximation of the information found in a full CDE analysis, while using an amount of resources that makes it practically applicable.

As we use the metrics defined in the previous subsection to guide our comparison, it is worth noting that since CDE does not ignore any part of the search space it follows that the values of USS, ARR and DC for CDE are respectively 0, 1 and 1. Also, when two or more practice problems are located at the same position in any dimension (i.e., feature identical outcome vectors, aka duplicates), a CDE analysis discards all but one of them. This removal of duplicates allows CDE to identify unique and informative practice problems as underlying objectives. However, this also means that some or all of the practice problems found in the last generation of P-PHC-P may have been discarded by a CDE analysis and therefore not present in the resulting coordinate system. In the case of duplicates, either practice problem holds the same information and thus CDE drops one arbitrarily. As such, when measuring DC and ARR for P-PHC-P, we take into consideration all duplicates of any given representative practice problem.

Lastly, it is also worth mentioning that the above performance metrics depend on how many unique individuals are found in the last generation of P-PHC-P. The smaller the genome length, and bound of genes, the more likely we will find fewer unique puzzles in P-PHC-P’s last generation. In this simulation, we set the genome length to 3 and each gene has value in [1, 10]. As a result, the number of unique individuals in the last generation of P-PHC-P reduces to 2 on average where the population size is set to 50. Table 3.1 shows the performance of both algorithms.

Please note that, the DC of P-PHC-P is very small compared to CDE. It is to be expected based on the fact that there were only two unique individuals, on average, in the
Table 3.1: The quantitative performance of evolved puzzles in simulated EvoParsons averaged over 30 independent trials. Due to CDE’s privilege to access into entire search space of interaction between puzzles and learners, its DC, ARR and USS are 1.0, 1.0 and 0 respectively. Performance of P-PHC-P’s evolved puzzles are measured against CDE’s performance where P-PHC-P has limited access.

<table>
<thead>
<tr>
<th>Algorithm</th>
<th>Performance Metrics</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>DC</td>
</tr>
<tr>
<td>CDE</td>
<td>1.0</td>
</tr>
<tr>
<td>P-PHC-P</td>
<td>0.0034</td>
</tr>
</tbody>
</table>

last generation. We find a total of 552 dimensions in the CDE analysis. Out of these, P-PHC-P covers only two \((1.89 = 552 \times 0.003433)\) dimensions on average. Also, while none of the individuals found in P-PHC-P’s last generation is an extrema of one of these dimensions, on average they “climb” 63% of the discovered dimensions. At a first glance, this average seems low. However, inspection of these “climbs” in every independent coevolutionary run suggests that in 50% of the runs P-PHC-P reaches the peak of the dimension. Although P-PHC-P performance in terms of DC should be further improved, it is reasonable given the extremely low number of unique individuals in the last generation and given the extremely limited portion of the entire search space that was explored \((3.5E - 4 = 1.0 - 0.99965)\).

In addition to considering the overall values, we also look at how the values of several metrics change during evolution. More specifically, we consider DC, ARR, and the number of unique individuals in the population. Figure 3.1 shows the values for these metrics in a single run.

3.2.4 Implications Regarding Design Guidelines

The values measured with the various quantitative metrics presented in the previous subsection provided us with further insights regarding the applicability of our approach to actual students.
First, it would be unrealistic to consider computing the full interaction matrix for CDE when dealing with practice problems presented to actual students. User fatigue alone would preclude having each student interact with each possible practice problem. If it were reasonable to do so, the main implication would be that the space of all possible puzzles is small enough to be exhaustively searched, thus rendering the use of meta-heuristics unnecessary.

Second, our experiment illustrated the potential for P-PHC-P to find practice problems that would be located at the extrema of the dimensions revealed by a full CDE analysis. This is interesting, especially given the fact that P-PHC-P is much more economical than a

Figure 3.1: Left: Distribution of DC, ARR. Mean of DC and ARR is 0.007 and 0.53 respectively. Right: Change of unique individuals after each generation. Mean and standard deviation of unique individuals is 3.72 and 8.83 respectively. This figure and respective data represent a single run of P-PHC-P in our simplistic teacher-learner model to show the change of performance metrics during evolution. All the experimental parameters are same as before.
full CDE analysis in terms of the number of interactions needed to identify these practice problems. This makes P-PHC-P a worthwhile algorithm to consider as a way to approximate CDE’s thoroughness, at a fraction of its cost, in order to mitigate user fatigue when evolving practice problems for actual students.

However, our experiment also stressed P-PHC-P limits in terms of how many of the CDE dimensions are represented in its last generation. This result was expected in so far that, while P-PHC-P is designed to “climb” within any dimension, it is not designed to maintain a diverse coverage of multiple dimensions in its population.

We also observed that the number of unique individuals decreases over time. A first possible explanation is that the strict acceptance condition in Pareto dominance is biased to promote parent practice problems to the next generation in most of the evaluation. As evolutionary time increases, almost all of the practice problems reach to its maximum gene values and dominates its child. Given the previously outlined benefits, we see these results as an acceptable trade-off, for now, and plan on investigating improvements to address the above-discussed limitations.

### 3.3 Experiment #8 - Evaluating Evolved Puzzles in EvoParsons

Simulations have helped us so far to sketch design guidelines, as well as to validate the performance of P-PHC-P based on relying on CDE analyses as a gold standard. In this section, we leverage these guidelines to implement a proof of concept EvoParsons software that coevolves Parsons puzzles by interacting with actual students.

#### 3.3.1 Problems and Algorithms

In order to evolve Parsons puzzles with P-PHC-P, we must first define how to encode them as genotypes.
3.3.1.1 Mapping Integer Vectors into Parsons Puzzles

We opted to use fixed-length integer vectors in which the first value, defined in $[1 : N_p]$, represents the index of a program in our *programs library* (see Figure 3.2). Each such program is composed of requirements, describing in plain English what the program does, along with a complete and correct implementation of a reference solution to the underlying problem. This reference solution is broken down into code fragments, as expected for Parsons puzzles. In this experiment, we consider each line of code in the solution as a separate fragment but different schemes are possible. Similarly, the remaining genes, each defined in $[1 : N_t]$, represent the indexes of distractors stored in our *distractors library*. Each distractor is implemented as a regular expression that matches certain fragments and transforms them so as to introduce a specific type of bug.

The process of mapping a given genotype to its corresponding Parsons puzzle is as follows:

1. We retrieve the program referenced by the first gene’s value from the programs library.

2. For all remaining genes:

   (a) We retrieve the regular expression referenced by the next gene from the distractors library.

   (b) We try to match its regular expression to every fragment of the program.

   (c) For those fragments where a match was found, we apply the transform specified in the regular expression to the first match in the fragment.
This results in a new distractor fragment that features the bug modeled by that particular regular expression.

3. The distractor fragments resulting from the above iterations are then shuffled, along with the valid fragments, to form the Parsons puzzle.

Let us take, as example, the genotype [5, 4, 3, 10]. The above-described mapping process starts by retrieving program number 5 from the programs library and distractors 4, 3, 10 from the distractors library. For the sake of this example, we will assume that program 5 simply displays “Welcome to Java”. Similarly, we will assume that the distractors 4, 3, 10 respectively remove the semicolon after a statement, capitalize the “class” keyword to “Class”, and replace occurrences of “void” by “char”.

```java
public class Welcome {
    public static void main(String[] args) {
        System.out.println("Welcome to Java!");
    }
}
```

After applying the above distractors in program 5, and shuffling the valid and invalid line of codes, we get the following Parsons puzzle:

```java
public class Welcome {
    public static void main(String[] args) {
        System.out.println("Welcome to Java!");
    }
}
public Class Welcome {
    public static void main(String[] args) {
    public class Welcome {
        System.out.println("Welcome to Java!");
    public static char main(String[] args) {
```
As mentioned in the above description, all these steps are implemented inside a component of EvoParsons called the “Broker”.

Please note that, the program and distractor libraries are written for Java languages only. I also develop C++ program and distractor libraries to extend EvoParsons’ scope in other programming language. To this end, the Summer 2019 internship experience as C++ developer intern at MathWorks help me to extend the scope in C++ programming language.

Figure 3.2 provides an overview of the complete architecture of EvoParsons.

![Figure 3.2: Overall work flow of EvoParsons. Evolutionary Algorithm (EA) sends integer vector i.e., genotypes of puzzle to the broker. Then Broker uses its libraries to create puzzles, send to the students on demand. As soon as students evaluation is obtained, fitness data is sent back to the Evolutionary algorithm via broker.](http://epplets.org/)

3.3.1.2 EvoParsons Architecture - EA, Broker and Learners

The Broker receives genotypes from an Evolutionary Algorithm working on the previously defined genotypes. After expressing genotypes into Parsons puzzles, by using the library of programs and the library of distractors, the Broker waits for requests for Parsons
puzzles from the student-facing components. Students are able to start a user interface from the epplets.org website by downloading a Java Web Start application. This application will then request an initial Parsons puzzle, then an additional one each time the user completed a puzzle. After students have complete each Parsons puzzle, the client returns evaluation data to the Broker that captures the student-puzzle interaction; e.g., time spent by student on Parsons puzzle, number of actions taken to solve it... This evaluation data is then archived and used to compute the fitness of that puzzle. When available, that fitness information is then returned to the Evolutionary Algorithm.

The matrix storing the outcomes of the interactions of registered users with specific Parsons puzzles is also managed by the Broker. The Broker uses it to not only assess Pareto dominance between a parent and child puzzle, but also to dispatch puzzles to students, so as to avoid sending the same puzzle several times to any one student. Other factors, described in the next subsections, can be easily leveraged to craft more elaborate dispatch policies.

This implementation is also distributed, in so far that the Evolutionary Algorithm, the Broker and the Epplets server components may be deployed on different physical or virtual servers. The Java Remote Method Invocation interfaces linking these components allow to easily replace both the front end and back end of the system. In order to ensure the technical feasibility of such exchanges, we ran preliminary calibration tests with two different implementations of evolutionary algorithms. The first one relied on the well-established ECJ framework (see next subsection for details). The other leveraged an “in-house” framework used in previous research (see “Jade” project\(^2\)). These tests were solely used to assess the technical feasibility of swapping various components in the system, thus providing reassurances on the flexibility of the system to be adapted by other researchers. In the next section, we will only discuss the ECJ implementation since it is the one that was used during experiments involving actual students.

\(^2\)Available at https://sourceforge.net/projects/jade/
3.3.1.3 User Fatigue, Sparse Interaction Matrix and Selection Policies

The Broker dispatches Parsons puzzles to students according to one of two available dispatching policies; The first one is an $\epsilon$-greedy policy that aims at assigning to already promising Parsons puzzles, more opportunities to be evaluated by students. This policy is inspired by the application of Reinforcement Learning techniques to Evolutionary Algorithms when dealing with noisy fitness landscapes. In such environments, each candidate solution may benefit from being evaluated multiple times. More specifically, we implemented White-son’s $\epsilon$-greedy policy that allows to select the current best puzzle for further evaluation with probability $\epsilon = 0.85$. The reader is referred to [144] for a detailed description and evaluation of this technique. In our application, such a policy helps mitigate user fatique. Each Parsons puzzle needs indeed to be evaluated by interacting with multiple students. Therefore, we start by guaranteeing a minimal number of interactions before to consider that it has been properly evaluated (minimum of 2 evaluations). After this, we use the $\epsilon$-greedy policy to determine whether this Parsons puzzle will receive additional evaluations.

Alternatively, a sparse distribution policy is used to prioritize dispatching of puzzles from the current generation that have only been evaluated so far by few students. Since evaluation data is not sent back to the EA until a given Parsons puzzle has been evaluated by a minimum number of students, this policy helps ensure, and even speed up, completion of a generation. Let us consider a scenario whereby the Parsons puzzle for a given child genotype needs to be evaluated by at least two different students in order to complete its outcome vector so that it may then be Pareto-compared with the Parsons puzzles of its parent genotype. Let us also assume that the system has three puzzles — $p_1$, $p_2$, $p_3$ — and three students — $s_1$, $s_2$ and $s_3$ — who are logged in and working. $s_1$, $s_2$ and $s_3$ request puzzles from the broker and $p_1$, $p_2$ and $p_3$ are allocated to them, respectively. When the numbers of students and puzzles are larger, this allocation policy minimizes the number of
evaluations for each Parsons puzzle in the current generation, more so than if the puzzles are distributed to $s_1$ first, then $s_2$ and so on.

When dispatching a Parsons puzzle, the broker relies on the “sparse distribution” policy with probability 30% and on the $\epsilon$—greedy policy the rest of the time. The $\epsilon$—greedy policy helps balancing the exploration vs. exploitation trade-off, whereas the “sparse distribution policy” reduces the sparsity of the interaction matrix (a dense interaction matrix being more convenient for CDE analysis).

From the implementation perspective, we leveraged time-established, software components;

- Amruth Kumar’s latest extension to the Problet tutoring system, Epplets, available at http://epplets.org/, which allows students to interact with Parsons puzzles and receive automated feedback.

- Sean Luke’s ECJ Java framework, available at https://cs.gmu.edu/~eclab/projects/ecj/, which provides implementations of many EA variants and that we extend to also implement P-PHC-C.

We extend both components so as to allow them to inter-operate via the Broker, and communicate with the latter using Remote Method Invocation (RMI) Java technology.

### 3.3.2 EvoParsons Interaction with Real Human Students

The software specified in the previous section was used during Spring 2017 with Information Technology students enrolled in an on-line introductory programming course at the University of South Florida (USF, COP2512 Programming Fundamentals for IT). The course is meant as a first introduction to programming for sophomores and is a state-
mandated prerequisite for the USF BS in Information Technology program. We ran two experiments over the course of the semester.

*Run #1* was conducted at the beginning of the semester, after students were exposed to basic Java concepts: data types, selection and iteration. In the course timeline, this means that it took place after module [203] (see previously referenced website for details). 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 from module [201], selection from module [202] and iteration from module [203]. P-PHC-P ran for a total of six generations as students worked on their assignments and explored 79 unique genotypes.

*Run #2* was conducted at the end of the semester, after module [305]. At that point in time, students had been exposed to more Java and were therefore more experienced. We used the 10 puzzles evolved by P-PHC-P during the first experiment and required participating students to work on all of them. CDE was then applied to that full interaction matrix as a way of visualizing the underlying interactions between puzzles and students. This also helped us evaluate the quality of the evolved puzzles found in the last generation, 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.

---

3 The course material is freely available at [http://cereal.forest.usf.edu/edu/COP2512/](http://cereal.forest.usf.edu/edu/COP2512/) so that the reader may have access to all details regarding the material to which students were exposed, prior to using our software.
3.3.2.1 Quantitative Perspective

Five puzzles from the last generation of Run #1 were not solved by any student. Similarly, 19 students just logged into the system but never solved any puzzle. As a result, the usable interaction matrix for Experiment #1, which was expected to be originally of size $107 \times 79$, was actually reduced to $88 \times 74$. Since this matrix was also sparse, we could not apply CDE directly to it. Therefore, we first extracted the largest sub-matrices of dimension $x \times y$, where $x$ stands for the number of students and $y$ for the number of Parsons puzzles with which these $x$ students interacted. Starting from $x \times y = 5 \times 5$, 99 such sub-matrices were found, the largest being of dimension $x \times y = 6 \times 10$.

These sub-matrices consisted of puzzles that appeared at any generation during the evolutionary process. Among these 99 matrices, we focused on those in which at least 75% of the Parsons puzzles were from the last generation (as these represent the end product of the evolutionary process). We found a total of 3 sub-matrices that met this criterion.

Table 3.2 shows the performance of P-PHC-P in both experiments. Although the Parsons puzzles covered all the CDE dimensions, and even found their extrema, most of the discovered dimensions in each interaction sub-matrix were singleton because of the high waste of P-PHC-P’s population slots due to the presence of duplicate and redundant puzzles. However, the result for Experiment #1 is inconclusive because the size of student population is limited to $\{5, 6\}$ whereas the total participating students were 79.

We required participating students to solve the 10 Parsons puzzles evolved during Run #1 in order to receive full participation credit. This introduces Run #2. However, we granted partial credit to those who interacted with fewer puzzles. As a result, we initially obtained an interaction matrix of dimensions $x \times y = 79 \times 10$ (as 79 students participated). After eliminating students who did not interact with all 10 Parsons puzzles, the dimensions were reduced to $61 \times 10$. The CDE analysis of this $M_{61 \times 10}$ matrix revealed no duplicate or
redundant puzzle; however, each dimension was a singleton — meaning each dimension consisted of a single puzzle. See the “Threats to Validity” section for a discussion of implications of this observation (Section 3.4.2).

When applying CDE to the particular matrix $M_{61 \times 10}$, we found that the DC and ARR metrics are not instructive since all 10 puzzles were taken into account. When applying CDE in Experiment #2 data, we use the same threshold parameter of win/loss (> 1.2 times move than a puzzle length considered as “loss”) as set in Experiment #1. Though adjusting this parameter may reveal some duplicates or redundant puzzles, CDE metrics are meaningful when analyzing non-coevolutionary interactions. Table 3.3 shows the number of total dimensions($\#D$) and singletons($\#S$) among them. It also provides the ID for extrema ($Ex-ID$) and duplicates ($d-ID$) for the matrix considered in Table 3.2.

**Table 3.2:** Performance measures of P-PHC-P in Experiments #1. % denotes the ratio of Parsons puzzle from P-PHC-P’s last generation. P-PHC-P does not produce any redundant puzzles in its last generation. Also, it covers all the dimension and their respective extrema puzzle. However, CDE wastes its population slot by accommodating duplicate puzzles.

<table>
<thead>
<tr>
<th>Matrices</th>
<th>%</th>
<th>D</th>
</tr>
</thead>
<tbody>
<tr>
<td>$M_{5 \times 10}$</td>
<td>0.9</td>
<td>0.44</td>
</tr>
<tr>
<td>$M_{6 \times 8}^+$</td>
<td>0.8</td>
<td>0.375</td>
</tr>
<tr>
<td>$M_{6 \times 8}$</td>
<td>0.8</td>
<td>0.125</td>
</tr>
</tbody>
</table>

The ID set of extrema is \{1, 5, 8, 9\}. These four puzzles might be worth investigating further from a qualitative perspective since they are extrema in their respective CDE dimensions and thus more informative than others.

### 3.3.2.2 Qualitative Perspective

In this section, we discuss the pedagogical relevance of the Parsons puzzles that were evolved by P-PHC-P. To this end, we leverage existing Computing Education research litera-
ture and, more specifically, work dedicated to defining a Concept Inventory for introductory programming courses.

Concept Inventories are usually created using multiple-choice assessments designed to measure students’ understanding of core concepts of a course or discipline [130]. They are often used with pretest-posttest experimental research designs as a way to quantify the learning resulting from the use of a specific pedagogical intervention, e.g., usage of Peer Instruction techniques instead of traditional lecturing.

Although CIs have had great impact on pedagogical techniques in other disciplines such as Physics, they are not currently widely used, nor often validated, for computer science topics. However, recent works [25] [26] on developing CI for introductory programming has identified students’ misconception on several programming topics. This was achieved by interviewing both students and instructors as well as analyzing errors committed by students in exams. These works resulted in multiple-choice questions covering the categories of misconceptions along with suggested distractors based on students’ misconceptions on those topics.

We found that four of the eight topics mentioned in [26] were represented in our programs and distractors libraries based on the modules that the students worked through by the time they were assigned to work with our software system. The exception are the topics of methods and recursions, which were not yet covered in the course at the time of Experiment #1. Furthermore, pointers and structures are out of the scope of the Java programming language. As a result, four of those topics are irrelevant here.

However, our system does not require semi-structured interviews with instructors. As such, it does not rely on their perspective on the misconceptions they believe to be common for students. This significantly differentiates our work from the above-mentioned approaches [26, 25]. Instead, the EA starts with randomly assembled programs and distractors, then evolves based on students’ interactions with these Parsons puzzles. In addition, the CDE
analysis of the underlying interaction matrices helps to classify the puzzles and misconception automatically. It also produces a hierarchical visualization of those misconceptions.

We find a total of 23 distractors, present in the Parsons puzzles found in the last generation, causing syntax or semantic error. They are categorized in Table 3.4. The coevolutionary algorithm considers a total of 19 different programs during its run. Out of these, 9 programs were represented in the very first generation of 10 random Parsons puzzles. Two of the extrema (Table 3.3)—ID: 1, 5—defeat all of their children from first to the final generation whereas the remaining two—ID: 8, 9—evolve from second generation defeating two other Java programs of the first generation and more from the next generation.

Our subsequent discussion starts with pedagogical relevance of the extrema along with their generational evolution during the coevolutionary run. The goal is to review the selective decisions made by P-PHC-P during the evolutionary process, from a pedagogical perspective.

**Table 3.3:** CDE dimensions, singletons, extrema and duplicate puzzles for each of the matrices in both experiments.

<table>
<thead>
<tr>
<th>Matrices</th>
<th>#D</th>
<th>#S</th>
<th>Ex-ID</th>
<th>d-ID</th>
</tr>
</thead>
<tbody>
<tr>
<td>$M_{5 \times 10}$</td>
<td>4</td>
<td>3</td>
<td>9</td>
<td>(1, 9), (3, 4, 5, 7, 8)</td>
</tr>
<tr>
<td>$M_{6 \times 8}^\uparrow$</td>
<td>3</td>
<td>2</td>
<td>8</td>
<td>(1, 8), (3, 4, 7)</td>
</tr>
<tr>
<td>$M_{6 \times 8}$</td>
<td>4</td>
<td>2</td>
<td>5, 1</td>
<td>(4, 6)</td>
</tr>
<tr>
<td>$M_{61 \times 10}$</td>
<td>10</td>
<td>10</td>
<td>NA</td>
<td>NA</td>
</tr>
</tbody>
</table>

Table 3.4 describes the four extrema we find from Experiment #1. They deal with the topics of control statement (*if else*), variable declaration and initialization, and use of logical operator. All, except the “Subtraction Quiz” Parsons puzzle, also deal with loops. Table 3.4 lists the distractors for each of these extrema.

We then examined the log over all the generations of the selection decisions between parents and their children. This allowed us to trace the origin of the extrema in terms of successive mutations and selections steps.
Table 3.4: The four extrema found from CDE analysis on the sub-matrices from Run # 1.

<table>
<thead>
<tr>
<th>ID</th>
<th>Parsons Puzzles</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>Use of do while loop</td>
<td>Add all the numbers given by a user using a do while loop until s/he inputs zero</td>
</tr>
<tr>
<td>5</td>
<td>Subtraction quiz</td>
<td>Prompts for the result of subtraction of two already initialized numbers and verifies whether the user provides the correct result</td>
</tr>
<tr>
<td>8</td>
<td>GCD calculator</td>
<td>Finds greatest common divisor between two integers</td>
</tr>
<tr>
<td>9</td>
<td>Guess game</td>
<td>Prompts the user to guess an already initialized random number until the user guesses it correctly</td>
</tr>
</tbody>
</table>

Figure 3.3 shows the hierarchy of distractors between extrema and their respective child practice problem. “Usage of do while loop” is contextually harder than its distractor. We exclude this extrema in Figure 3.3.

Figure 3.3: The distractor in Subtraction Quiz helps this practice problem to be more competitive i.e., Pareto dominate than its child Practice problem. The same description applies for GCD Calculator and Guess game.
3.3.3 Design Guidelines Validation

In this subsection, we review the design guidelines, previously detailed in sections 3.2.4 and 3.2.4, then discuss how they fared when applied to actual students. In both the preliminary simulations and the application to real students, the Parsons puzzles evolved by P-PHC-P are, on average, ranked high in the dimensions that would be obtained by a CDE analysis. We examined detailed traces of successive mutations and selection operations, especially for the four extrema Parsons puzzles from the last generation. Based on our findings, we conclude that P-PHC-P is indeed capable of evolving Parsons puzzles that feature increased informativeness, when compared to their counterparts from earlier generations. In terms of pedagogy, this evolution means that P-PHC-P evolves Parsons puzzles that feature interesting distractors.

The coverage of all CDE dimensions, as well as the absence of composite Parsons puzzles may be interpreted as a symptom of P-PHC-P’s ability to maintain sufficiently diverse individuals in its population, during convergence. However, the presence of duplicate puzzles suggests the need for further improvement.

We cannot really say anything about coevolutionary pathologies, e.g., we have not shown that the problem exhibits overspecialization or intransitivity.

As far as user fatigue is concerned, we rely on selection policies deployed into the Broker software component. Although we have not proposed any metric to quantify user fatigue, we analyzed the time and number of moves distributions for the 10 puzzles evolved by P-PHC-P. This analysis gave us an idea about the effectiveness of the selection policies that we used. This, in turn, helped understanding the severity of the underlying user fatigue phenomenon.

We found that only two out of ten puzzles require more than double relative moves to solve by the students. The rest of them are solved in the range of [1.3, 1.66] relative moves.
which indicates that student were engaged to solved most of the puzzles in a viable number of moves. It supports the effectiveness of the selection policy, i.e., less user fatigue during evaluation.

Selecting a coevolutionary algorithm already known to handle such commonly occurring pathologies remains a sound design guideline in the absence of more detailed analysis of the coevolutionary dynamics.

The design guidelines discussed in section 3.2.4 also covered the gene’s bounds values, noise levels, and the number of interactions. In EvoParsons, the gene bounds are imposed by the number of programs and distractors available in their respective libraries. As already discussed, the noise is inherent to the evaluations. However, the distributions of “time” and “number of moves” required to solve the evolved puzzles does not show many outliers.

For the number of interactions, we evaluated each pair of puzzles with at least two unique students during a generation. This helped expedite the evolution of Parsons puzzles.

As we applied the above design guidelines to the implementation of EvoParsons, experimental results did not reveal any hindrance preventing our system from evolving pedagogically sound puzzles (as evidenced by their relation to a CDE coordinate system). Future work will focus on further improving the quality of the evolved Parsons puzzles, and will leverage the present results as baseline for fair comparison.

The qualitative analysis of evolved puzzles, with respect to the dimensions of CDE, indicates that the interaction matrix obtained from EvoParsons can be used to generate a concept tree. Each concept is a hierarchy of Parsons puzzles, based on their difficulties and informativeness with respect to their evaluators.
3.4 Validations of EvoParsons

3.4.1 Summary of Findings

This work examined the potential of leveraging coevolutionary techniques to automatically evolve Parsons puzzles. We started by considering coevolutionary number games and progressively moved towards our target EvoParsons application: evolving Parsons puzzles for actual students. To this end, we chose a simple Pareto-based hill climber algorithm embodying Pareto-based coevolutionary principles. We replicated experiments of well-studied number games to confirm its ability to circumvent known coevolutionary pathologies that are also relevant to educational scenarios.

Second, we introduced a new variant of coevolutionary number games featuring noisy interactions. This allowed us to assess the suitability of the algorithm beyond traditional benchmarks and on simulations capturing an essential aspect of algorithms interacting with human evaluators. We used this to introduce a synthetic problem capturing more aspects of the genetic encoding to be used to represent Parsons puzzles in our target application.

All along, our contribution consisted of identifying design guidelines to adapt the chosen algorithm to this succession of simulated problems that each integrated more of the features we expected to find in our target application. These design guidelines were then leveraged to implement a Pareto-based coevolutionary algorithm that evolved Parsons puzzles for actual students enrolled in an introductory programming course taught at USF during the spring 2017 semester. Preliminary results suggested that the Pareto-based coevolutionary approach in general, and our design guidelines in particular, show great promise as a viable solution to the problem of evolving practice problems for novice programmers.

We supplemented the above results by taking a closer look at the ten Parsons puzzles that were previously coevolved. 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. Results also suggested that the Pareto coevolution we used in this work may be further improved. More particularly, we are considering 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.

3.4.2 Threats to Validity

Although the two samples used in our experiment with actual students were taken from the same underlying student population (students enrolled in COP2512 during Spring 2017 at USF) the following potential biases should be considered when reviewing results.

First, the two experiments were performed several weeks apart. As such, it is reasonable to expect that a significant portion of students improved their skills over that time. Similarly, some of the students who were struggling may have dropped the course. As a result, students in the second sample might perform better than the students in the first sample.

Second, the 10 Parsons puzzles used in our second experiment discussed in section 3.3.2 were evolved during the first experiment. As such, some of the students may have been exposed to the same Parsons puzzles during both experiments, which would lead us to expect improved performance as well.

In order to establish whether or not the students performance was indeed biased, we considered the relative number of moves students took to solve each of those 10 puzzles in both experiments. We found that students took fewer moves on average in the second exper-
iment. Two out of the 10 puzzles showed significant improvement ($p < 0.01$ for Palindrome Detector and $0.01 < p < 0.05$ for Multiplication Table). The differences for other puzzles failed to prove statistically significant.

As such, we have a method that allows us to measure such bias and therefore determine its impact on interpretations of results. However, in this work, this approach did not reach statistical significance for most of the puzzles for the final experiments involving students. Therefore, we plan to repeat the experiment in the near future with a larger student sample. Such an experiment will allow us to validate our preliminary findings, including estimating the potential impact of the threats to validity we discussed in this section.

3.4.3 Improvement of EvoParsons

P-PHC-P may be further improved by enabling it to reduce the number of duplicate individuals found in the last generation. We hypothesize that doing so may help increase both dimension coverage and average rank of representatives.

The two inherent requirements of CDE analysis — i.e., binary interaction outcome and full interaction matrix — have already been shown to limit its applicability to EvoParsons. As we have seen in section 3.3.2, when every single evaluation is costly (e.g., time consuming, involving human agents, inducing user fatigue...) it is unrealistic to expect a full interaction matrix. Similarly, such interactions may result in performance measurements that can only be boiled down to binary outcomes by fixing arbitrary thresholds; i.e., if students perform below the threshold, then the outcome is a loss. It would be preferable to remove such extraneous parameters from the system, especially in the absence of any mathematical model allowing us to formally study their impact on coevolutionary dynamics. Therefore, we plan on also extending CDE to handle both non-binary outcomes and sparse interaction matrices.
We used EvoParsons in several courses. The evolved distractors and student’s activities helped us to understand which distractors posed problems to our students. However, multi-year or multi-semester studies would provide better insights about the distractors that cause students to struggle the most, and those that best capture differences in students’ performance. We therefore intend to conduct such follow-up, larger scale, studies as part of a wider dissemination effort for EvoParsons.
Chapter 4: Accelerating Progress in Elementary Coevolution

4.1 Need for Progress Acceleration

This chapter was published in IEEE SSCI conference \(^1\), IEEE CEC conference \(^2\) and IEEE ICCIT conference \(^3\). IEEE permissions are included in Appendix A.

In P-PHC, the candidate solutions are evolved based on Pareto dominance. The child candidate solutions are in strict acceptance condition and hence forwarded to the next generation if Pareto dominate the parents. Otherwise, the child is discarded. Please note that, (parent, child) pair becomes non-comparable most of the time when the number of interacting individuals is as large as the population size in a test based coevolutionary interaction. Hence, this study does not discard the child immediately if it is non-comparable with its parent. In this chapter, first we seek the answer if relaxing the strict acceptance condition of P-PHC accelerates the convergence. Then we propose some selection criteria to relax the strict condition. Finally, we measure the convergence and diversity under both strict and relaxed selection criterion of P-PHC-P candidate solutions. Experiment #9 and #10 propose two relaxed selection based variant of P-PHC-P, Experiment #11 and #12 investigates four other selection criterion based on upward, lateral and forward direction of the search inside interaction space. While developing such criterion, we also inject some diversity based


techniques to improve the diversity of the variants. Finally, we measure the distribution of layered individuals in the population in Experiment #13 and performance of the relaxed variants, in Experiment #14, under multi-objective optimization performance metrics.

4.2 Definitions and Formalism

• This is a method of rewarding unusual individuals; e.g. a new innovation in the search space or an old niche that recently became important [109]. The fitness, $D_1$ (Competitive Shared Fitness), assigned to a candidate solution that defeats tests with the set of indexes $X$ is $\sum_{j \in X} \frac{1}{N_j}$, where $N_j$ is the total number of candidates in the population defeating test $j$. Please note that the non-comparable candidate solutions have different count for $N_j$. Therefore, rewarding the solutions whose $N_j$ is lesser helps retain unusual individuals in the population.

Algorithm 4.1 calculates the shared fitness of an individual $\text{indiv}$. Interaction outcomes of $\text{indiv}$ against all the individuals of other population are stored in outcome vector $V_{\text{indiv}}$. The algorithm also requires interaction matrix of all individuals ($V_{\text{tmp}}$) against other population. Then the number of individuals from other populations that are defeated by $\text{indiv}$ are summed and used to calculate its shared fitness. In this study, interaction outcome is either 1 or 0 to indicate "win" or "lose" respectively.

• Two candidates $x$ and $y$ are equivalent, $D_2$ (Pareto Equivalence Set), with respect to $T$, denoted $x \equiv y$ iff: $\forall T_j \in T : G_{x,j} = G_{y,j}$. As the solutions are equivalent, one of them can be discarded due to representing waste in the population slot.

4.3 Experiments # 9 Relaxing Selection Criteria in P-PHC-P

First we investigate the impact of competitive shared fitness [109] between a non-comparable ($\text{parent, child}$) pair rather than just discarding the $\text{child}$. Second, we do fast
Algorithm 4.1 Competitive Shared Fitness

SharedFit \((V_{\text{indiv}}, V_{\text{imp}})\)

\[
X \leftarrow \{\}
\]

\textbf{for} \(i = 1\) \textbf{to} length\(\left(V_{\text{indiv}}\right)\) \textbf{do}

\textbf{if} \(V_{\text{indiv}}[i] > 0\) \textbf{then}

\quad \(X \leftarrow X \cup i\) \quad \triangleright \text{Defeated index of opponents}

\textbf{end if}

\textbf{end for}

\textbf{fitness} \leftarrow 0

\textbf{for each} defeated index \(i \in X\) \textbf{do}

\quad \(N \leftarrow 0\)

\quad \textbf{for each row} \(r \in V_{\text{imp}}\) \textbf{do}

\quad \quad \(N \leftarrow N + V_{\text{imp}}[r, i]\)

\quad \textbf{end for}

\quad fitness \leftarrow fitness + 1/N

\textbf{end for}

\textbf{return} fitness

non dominated sort [38] on the temporary population consisting of all the \((\text{parent}, \text{child})\) pairs. Then we build the next generational population based on the Pareto fronts. In case of considering individuals from the last front, our selection criteria is based on aggregation of the interaction outcomes rather than crowding distance used in [38]. The idea of promoting competitive non-comparable children helps preserving diversity in the population and hence are expected to accelerate the progress. In addition, maintaining Pareto front helps discarding inferior solutions regardless of being parent or child.

4.3.1 Problems and Algorithms

We use FG and \(G_{\text{one}}\) game. Before going to the subsequent discussions, we introduce some notations to clearly understand the algorithms that are used.

- \(p, pop\) refers to the population.

- \(indiv, child, parent\) indicate an individual that is a member of population.

- \(V_X\) implies interaction outcome vector for \(X\), \(X = \{indiv, child, parent\}\).
• $oP$ denotes Opponent Population.

• $\text{popsize}$ is for Population Size.

• $\square$ for initialized value of an interaction outcome vector.

• $F_i$ is the $i^{th}$ Pareto Front. Please note that $i = 0$ indicates the front of superior individuals.

4.3.1.1 Base Selection in P-PHC-P

The selection between parent and child in P-PHC-P is based on Pareto optimality as described in Algorithm 4.3. The replacement of a parent by its child is done by Algorithm 4.2.

Algorithm 4.2 Replacement of parent by child
BaseReplacement ($parent, child, p$)

1: $p \leftarrow p - \{parent\}$
2: $p \leftarrow p \cup \{child\}$

Algorithm 4.3 Selection between parent and child in P-PHC-P
BaseSelection ($V_{parent}, V_{child}, parent, child, p$)

1: if $V_{child} \succ V_{parent}$ then
2: BaseReplacement ($parent, child, p$)
3: return true
4: end if
5: return false

Algorithm 4.3 implements the deterministic selection policy between parent and child to decide which one is kept in the next generation. Because the offspring only replaces its parent, if the former is Pareto optimal, P-PHC-P guarantees monotonic improvement. However, two limitations stem from this strictly acceptance condition policy.

Firstly, the progress of candidate solutions or tests stalls most of the coevolutionary time because parent and child become mutually non-dominated due to the large set of
objectives. So, parents move to the next generation most of the time. Our analysis shows that this count of mutual non-domination increases rapidly when the number of opponents increases more than four [49]. Another analysis in number games like FG shows that the addition of two probabilities - parents dominates child and both of them are mutually non-dominated - is larger than 0.98 and the results are statistically significant ($p < 0.01$).

Secondly, the algorithm is susceptible to maintain diverse individuals in the population because of the presence of mutually non-dominated parents in the population. As a result, P-PHC-P may have difficulty to maintain ideal objective sets [33] of its underlying problem structure.

These observations motivated us to explore minimalist variations of P-PHC-P which would provide better candidate solutions in terms of accelerating progress and also discovering set of solutions that approximates the ideal evaluation.

4.3.1.2 Competitive Shared Fitness Based Relaxed Selection

The objective of this first variant is to give a chance to the non-comparable offspring to be kept in the next generation if the child satisfies all the conditions stated in the following three steps.

- **Step #1:** Calculate *Objective Tolerance* ($\tau$) - A measure that represents at best in how many objectives a child can be an inferior performer than its non-dominated parent. This metric is represented as the percentage of population size and we set the fraction low (5%).

  Let’s $O_p$ be the number of objectives (out of $m (= \text{popsize})$) a parent ”wins”, the same for child is $O_c$. Then $\tau = O_p - O_c < 0.05 \times m$. Please note that the mutual non-domination may also occur when $O_c \geq O_p$. The requirement of $\tau$ in this case is meaningless. We introduce this tolerance parameter only for the case when $O_p > O_c$. 

111
$O_p$ and $O_c$ are calculated using Algorithm 4.4. Please note that Algorithm 4.4 takes interaction outcome vector ($V_{indiv}$) of any individual and returns the number of “win”s.

- Step #2: Calculate Shared Fitness - If $\tau < 0.05 \times m$ or $O_c > O_p$, calculate the shared fitness of parent ($\delta_{parent}$) and child $\delta_{child}$ using Algorithm 4.1.

- Step #3: If $\delta_{child} > \delta_{parent}$ then we replace parent by its child using Algorithm 4.2.

Our rP-PHC-P variant is described in Algorithm 4.6 which basically calls Algorithm 4.5 (line 32) for each pair of (parent, child) in the population. In case of parent Pareto dominates child, parent is not replaced as it is originally done in P-PHC-P’s strict acceptance condition. Hence, forwarding child meant to enable rP-PHC-P to consider alternative, and equally valuable, candidate solutions when P-PHC-P progress stalls due to the strictly better than Pareto optimal concept. Doing so has benefits of conditional exploration of the search space instead of exploiting parent solutions most of the time.

Our rP-PHC-P variant defined in Algorithm 4.6 leverages competitive fitness sharing [109] between non-dominated parent and child. The use of the shared fitness as a differentiating criteria between mutually non-dominated parent and its child helps to explore the search space for unusual but important solutions and hence adding such candidates to the solution concept accelerate the progress. In case of Pareto equivalent set, shared fitness for

<table>
<thead>
<tr>
<th>Algorithm 4.4 Aggregate Fitness</th>
</tr>
</thead>
<tbody>
<tr>
<td>AggregateFit ($V_{indiv}$)</td>
</tr>
<tr>
<td>fitness ← 0</td>
</tr>
<tr>
<td>for $i = 1$ to length($V_{indiv}$) do</td>
</tr>
<tr>
<td>fitness ← fitness + $V_{indiv}[i]$</td>
</tr>
<tr>
<td>end for</td>
</tr>
<tr>
<td>return fitness</td>
</tr>
</tbody>
</table>

parent and child remains the same. However, we can’t select child in case of its equivalence with parent; because doing so can forward a less competent child instead of a strong parent, if the opponents are too strong or too easy due to the loss of fitness gradient [141].
Algorithm 4.5 Selection between mutually non-dominated pair

\[ \text{RelaxedSelection} \ (V_{\text{parent}}, V_{\text{child}}, V_{\text{tmp}}, \text{parent}, \text{child}, p) \]

\[
\text{status} \leftarrow \text{BaseSelection} \ (V_{\text{parent}}, V_{\text{child}}, \text{parent}, \text{child}, p)
\]

\[
\text{if status} = \text{false AND NOT } V_{\text{parent}} \equiv V_{\text{child}} \text{ then}
\]

\[
O_p \leftarrow \text{AggregateFit} \ (V_{\text{parent}})
\]

\[
O_c \leftarrow \text{AggregateFit} \ (V_{\text{child}})
\]

\[
\text{if } \tau < 0.05 \times m \text{ OR } O_c \geq O_p \text{ then}
\]

\[
\delta_{\text{parent}} \leftarrow \text{SharedFit} \ (V_{\text{parent}}, V_{\text{tmp}})
\]

\[
\delta_{\text{child}} \leftarrow \text{SharedFit} \ (V_{\text{child}}, V_{\text{tmp}})
\]

\[
\text{if } \delta_{\text{child}} > \delta_{\text{parent}} \text{ then}
\]

\[
\text{BaseReplacement} \ (\text{parent}, \text{child}, p)
\]

\[
\text{end if}
\]

\[
\text{end if}
\]

4.3.2 Methods

We are interested in measuring the ability of the algorithm to maintain their progress, avoid overspecialization, and determining the “waste” in population slot with respect to the cardinality in Pareto Equivalence Set found from the last generation of the algorithms. The catch for waste is that the algorithm producing less waste is expected to keep more individuals that have unique interaction outcomes and hence can be treated as an indicator of maintaining ideal solution set better than the algorithm that produces more waste.

Along with the metric Mean Objective Fitness (\(\mu_{\text{obj}}\)) in Section 2.2.3 to avoid stall in progress and overspecialization (\(\mu_{\text{disp}}\)) in we also measure the waste of the algorithms in their population slot. To this end, the number of Pareto-equivalent solutions in the last generation of the algorithms can be accounted as an indicator of this wastage.

This metric (Waste ratio (\(W_r\))) borrows the concept discussed in definition \(D_2\). Formally, 

\[
W_r = \frac{M}{N}
\]

where \(M\) and \(N\) represents the number of total Pareto equivalent candidate individual and the population size respectively. The better algorithm will keep less Pareto equivalent solutions in its population slot. This metric is in the range of \([0, 1]\) where smaller values indicate better performance.
Algorithm 4.6 Population-based Pareto Hill Climber based on Algorithm 4.5

rP-PHC-P (Payoff Function PF (FG, Gone))

1: popsize ← 50
2: \( P_C \) ← {} \( \triangleright \) Population of Candidates
3: \( P_T \) ← {} \( \triangleright \) Population of Tests
4: for popsize times do
5: \( P_C \) ← \( P_C \cup \{ \text{new random individuals} \} \)
6: \( P_T \) ← \( P_T \cup \{ \text{new random individuals} \} \)
7: end for
8: pop ← \([P_C, P_T]\) \( \triangleright \) A list of populations
9: repeat \( \triangleright \) Evaluating each population
10: for each population \( p \in \text{pop} \) do
11: if \( p = P_C \) then
12: \( oP \) ← \( P_T \) \( \triangleright \) Is candidate current pop?
13: else
14: \( oP \) ← \( P_C \) \( \triangleright \) Opponent is candidate
15: end if
16: \( V_{tmp} \) ← [] \( \triangleright \) List of vectors,size=2 * popsize
17: \( p_{tmp} \) ← [] \( \triangleright \) List of individuals,size=2 * popsize
18: for each parent ∈ \( p \) do
19: \( \text{child} \) ← Mutate(parent) \( \triangleright \) Outcomes vector between parent and \( \forall \text{ind} \in oP \)
20: \( V_{parent} \) ← [] \( \triangleright \) dimension = 1 \( \times \) popsize
21: \( V_{child} \) ← [] \( \triangleright \) Outcomes vector between child and \( \forall \text{ind} \in oP \)
22: for \( i \) ← 1 to popsize do
23: \( V_{parent}[i] \) ← PF(parent,\( oP[i] \)) \( \triangleright \) dimension = 1 \( \times \) popsize
24: \( V_{child}[i] \) ← PF(child,\( oP[i] \))
25: end for
26: \( V_{tmp}[\text{parent}] \) ← \( V_{parent} \)
27: \( V_{tmp}[\text{child}] \) ← \( V_{child} \)
28: \( p_{tmp}[\text{parent}] \) ← parent
29: \( p_{tmp}[\text{child}] \) ← child
30: end for
31: for \( i = 1 \) to length(\( V_{tmp} \)) do
32: RelaxedSelection \( (V_i, V_{i+1}, V_{tmp}[i], p_{tmp}[i+1], p) \)
33: end for
34: end for
35: until Stopping Criterion Met
Please note that, the algorithm whose $W_r$ is less is expected to keep more solutions in its population slot that can be representative of minimal objective set [20] toward ideal evaluation of coevolution than the algorithm whose $W_r$ is more. Hence, $W_r$ can be treated as, not all the time, an approximation of the cardinality of minimal objective set.

4.3.3 Results

Table 4.1 shows the performance of the algorithms for the number games. The experimental results suggest that $r$-P-PHC-P performs better for all the metrics - $\mu_{obj}, \mu_{disp}, W_r$ - than P-PHC-P ($p < 0.01$).

Table 4.1: Performance of P-PHC-P, $r$P-PHC-P and $f$P-PHC-P for FG and Comp-on-one.

<table>
<thead>
<tr>
<th>Algorithm</th>
<th>Problem</th>
<th>Performance Metrics</th>
<th>$\mu_{obj}$</th>
<th>$\mu_{disp}$</th>
<th>$W_r$</th>
</tr>
</thead>
<tbody>
<tr>
<td>P-PHC-P</td>
<td>FG</td>
<td>$r$P-PHC-P</td>
<td>96.48</td>
<td>2.27</td>
<td>0.93</td>
</tr>
<tr>
<td>$r$P-PHC-P</td>
<td></td>
<td>FG</td>
<td>98.51</td>
<td>1.03</td>
<td>0.90</td>
</tr>
<tr>
<td>$f$P-PHC-P</td>
<td></td>
<td>FG</td>
<td>97.65</td>
<td>2.02</td>
<td>0.93</td>
</tr>
<tr>
<td>P-PHC-P</td>
<td>$G_{one}$</td>
<td>$r$P-PHC-P</td>
<td>96.73</td>
<td>2.36</td>
<td>0.93</td>
</tr>
<tr>
<td>$r$P-PHC-P</td>
<td></td>
<td>$G_{one}$</td>
<td>98.26</td>
<td>1.53</td>
<td>0.90</td>
</tr>
<tr>
<td>$f$P-PHC-P</td>
<td></td>
<td>$G_{one}$</td>
<td>97.86</td>
<td>1.54</td>
<td>0.93</td>
</tr>
</tbody>
</table>

4.3.4 Relaxation vs Convergence, Exploration and Monotonic Progress

4.3.4.1 Convergence and Quality of the Solutions

The promotion of non-comparable child seems helpful to reduce duplicate solutions from the population. This helps the algorithm to reduce the pathological effect better and also maintain better progress.

The selection pressure in $r$P-PHC-P is based on both Pareto optimal set and shared fitness for mutually non-dominated (parent, child). On the other hand, it is only Pareto
optimality for P-PHC-P. It seems that the selection pressure in rP-PHC-P has benefits of keeping unusual child who beats more test(s) that are less defeated than its parent.

4.3.4.2 Relaxation vs Exploration

Relaxing strict acceptance condition in rP-PHC-P opens the door for exploration. To empirically measure how much exploration can be, we use the three metrics defined in Section 2.7.7.2 that tell us the portion averagely a parent dominates its child (ds), a parent is dominated by its child (dt) and both of them are mutually non-dominated (non). Then we observe how those metrics vary with increasing objective size m.

Table 4.2 shows the three metrics for P-PHC-P under FG. Please note that the results are statistically significant for most of the successive pair shown in Table 4.3. non decreases with the increment of m while the other two metrics increases. However, only a very small portion (1.4%) of time child becomes strictly better than its parent. So, P-PHC-P has very small chances to explore the search space because 98.6% times child fails to replace its parent. Hence, P-PHC-P’s progress stalls.

Table 4.2: The frequency based performance metrics for Pareto relationship for (parent, child) pair in P-PHC-P under FG.

<table>
<thead>
<tr>
<th>m</th>
<th>FG</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>ds</td>
</tr>
<tr>
<td>2</td>
<td>0.022</td>
</tr>
<tr>
<td>4</td>
<td>0.047</td>
</tr>
<tr>
<td>6</td>
<td>0.069</td>
</tr>
<tr>
<td>8</td>
<td>0.089</td>
</tr>
<tr>
<td>10</td>
<td>0.116</td>
</tr>
<tr>
<td>12</td>
<td>0.130</td>
</tr>
<tr>
<td>16</td>
<td>0.167</td>
</tr>
<tr>
<td>32</td>
<td>0.276</td>
</tr>
<tr>
<td>50</td>
<td>0.371</td>
</tr>
</tbody>
</table>
Table 4.3: Significance tests for pairwise comparison, ↑ means significant, ↓ indicates failed to significance, $p \leq 0.01$

<table>
<thead>
<tr>
<th>$m$</th>
<th>$p$ in FG</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>$ds$</td>
</tr>
<tr>
<td>(2 vs 4)</td>
<td>↑</td>
</tr>
<tr>
<td>(4 vs 6)</td>
<td>↑</td>
</tr>
<tr>
<td>(6 vs 8)</td>
<td>↑</td>
</tr>
<tr>
<td>(8 vs 10)</td>
<td>↑</td>
</tr>
<tr>
<td>(10 vs 12)</td>
<td>↑</td>
</tr>
<tr>
<td>(12 vs 16)</td>
<td>↑</td>
</tr>
<tr>
<td>(16 vs 32)</td>
<td>↑</td>
</tr>
<tr>
<td>(32 vs 50)</td>
<td>↑</td>
</tr>
</tbody>
</table>

In case of relaxing the strict acceptance condition, $rP$-PHC-P benefits exploring search space more ($\leq 61.5\%$) than P-PHC-P. So, relaxing the condition opens the door for exploration.

4.3.4.3 Relaxation vs Monotonic Progress

$rP$-PHC-P is neither gathering all its previous knowledge as described in [45] nor the objective size ($m$) varies from generation to generation. Instead, $rP$-PHC-P has fixed size objectives and it does not store the knowledge of previous generation. However, the promotion of non-dominated child may upset the monotonic progress only when the best parent gets replaced by its inferior child that is non-dominated but in the inferior Pareto front. But that probability seems very low because there may have other parents who are equivalent to the current best.

As we have not studied the probable chance of replacing best(s) parent(s) by respective non-dominated inferior child, we are not claiming that relaxing the conditions in $rP$-PHC-P maintains the monotonic progress. However, the empirical analysis shows that if we promote mutually non-dominated child from same front or the superior front then
monotonic progress is maintained. Figure 4.1 and Figure 4.2 show the performance of a typical run, in terms of objective fitness vs evolutionary time, for FG and Comp-on-one game respectively. The performance curves have no indication of upsetting the progress.

**Figure 4.1:** The performance of the candidate solutions of P-PHC-P and r-P-PHC-P in a typical run under FG number game. Please note that the maximum achievable objective fitness is 100.

### 4.4 Experiment #10 Pareto Front Based Relaxed Selection

Let’s consider two pairs \((p_1, c_1)\) and \((p_2, c_2)\) where \(p_1 \succ c_1\) and \(p_2 \succ c_2\). In case of P-PHC-P, \(\{p_1, p_2\}\) creates the population of next generation regardless the Pareto relationship between \(c_1\) and \(p_2\). However, if \(c_1 \succ p_2\) then \(p_1 \succ c_1 \succ p_2 \succ c_2\) and hence it is unwise to keep \(p_2\) in next generation. But if we maintain Pareto layers then \(\{p_1\}\) creates \(F^0\) which opens the door of keeping \(\{p_1\}\) first and then considering \(\{c_1\}\) or any one in \(\{p_2, c_1\}\) assuming they
are mutually non-dominated based on the designed specific (e.g. diversity) criteria. This motivates us to propose fP-PHC-P.

4.4.1 Problems and Algorithms

We use FG and $G_{one}$ game in this experiment. In fP-PHC-P, Pareto fronts from a temporary population of $2 \times popsize$ is created. The temporary population can be created using rP-PHC-P (up to line 30). Then Algorithm 4.7 is called to generate fronts and updating the population from next generation. We apply fast non dominated sort [38] on the temporary population and fill up the population slot front by front. The only difference is that the remaining population slot is filled up based on non-ascending aggregated fitness

![COMPARE-ON-ONE Game](image)

**Figure 4.2:** The performance of the candidate solutions of P-PHC-P and r-P-PHC-P in a typical run under $G_{one}$ game. Please note, the maximum achievable objective fitness is 100.
done by *ReverseSort* instead of the crowding distance concept employed in [38]. *ReverseSort* takes the front individuals and their aggregated fitness. Then it sorts them in descending order of aggregated subjective fitness.

Please note that, it may happen that all the individuals in the population may fall into a single front (front count = 0) or the number of individuals in $F^0$ is equal to or larger than population size. we use Algorithm 4.3 in both cases.

**Algorithm 4.7** Population-based Pareto Hill Climber based Pareto Front

\[
\text{P-PHC-P} \quad (V_{tmp}, p_{tmp}, p, \text{popsize})
\]

1: $F \leftarrow \text{fast} - \text{non} - \text{dominated} - \text{sort}(p_{tmp})$
2: if $F$.size = 0 then \(\triangleright A \text{ single layer}
3: \text{for each} (\text{parent, child}) \text{ pair in } p_{tmp} \text{ do}
4: \quad \text{BaseSelection} (V_{\text{parent}}, V_{\text{child}}, \text{parent, child}, p)
5: \text{end for}
6: \text{else}
7: \quad pt_{t+1} \leftarrow \{\}, i \leftarrow 0, \text{firstTime} \leftarrow \text{true}
8: \text{while } |pt_{t+1}| + |Fi| \leq \text{popsize} \text{ do} \(\triangleright || \text{ denotes size}
9: \quad \text{firstTime} \leftarrow \text{false}
10: \quad pt_{t+1} \leftarrow pt_{t+1} \cup Fi
11: \quad i \leftarrow i + 1
12: \text{end while}
13: \text{if firstTime = true then}
14: \quad \text{for each} (\text{parent, child}) \text{ pair in } p_{tmp} \text{ do}
15: \quad \text{BaseSelection} (V_{\text{parent}}, V_{\text{child}}, \text{parent, child}, p)
16: \text{end for}
17: \text{else}
18: \quad v \leftarrow []
19: \quad \text{for each} \text{ member } m \in Fi \text{ do}
20: \quad \quad v[m] \leftarrow \text{AggregateFit}(V_m)
21: \text{end for}
22: \quad Fi \leftarrow \text{ReverseSort}(Fi, v)
23: \quad pt_{t+1} \leftarrow pt_{t+1} \cup Fi[1 : \text{popsize} - |pt_{t+1}]]$
24: \text{end if}
25: \quad p \leftarrow pt_{t+1}
26: \text{end if}
4.4.2 Methods

We use the same performance metrics and experimental parameters available in Experiment #7 of Section 3.2.

4.4.3 Results

All the experimental parameters, problems and methods are the same as defined before. Table 4.1 shows the performance of \( f_{P-PHC-P} \). The results shown in Table 4.1 are statistically significant \( (p < 0.01) \) and indicate that \( f_{P-PHC-P} \) outperforms P-PHC-P based on the metrics \( \mu_{obj} \) and \( \mu_{disp} \). However, \( W_r \) does not show significantly better result.

Though maintaining front helps the algorithm to improve progress and also avoidance of pathology, \( f_{P-PHC-P} \) is unable to place the Pareto Equivalent solutions at lower fronts. This may be due to applying the strict acceptance condition when it produces only one Pareto layer.

4.5 Relaxation Based on Interaction Space of Pareto Coevolution

Relaxing the strict acceptance condition for mutually non-comparable (parent, child) pair and discarding inferior individuals by introducing Pareto fronts as a form of relaxation improve the progress of the algorithms. This also leads the algorithms to outperform in case of avoiding coevolutionary pathology like overspecialization. The inclusion of relaxed condition based on competitive shared fitness and also by means of Pareto fronts help the algorithms explore the search space while maintaining the same effect of exploitation by following strict acceptance condition also.

Pareto coevolutionary algorithms apply a Pareto-based solution concept. In general there are two salient representational concepts in Pareto coevolution: an archive, which represents the current best state of a solution set and a population, which represents the
current state of the search. In some algorithms, these are separate structures [44], but in algorithms such as P-PHC these two concepts are combined into a single representation: the population is the archive. Algorithms that combine these structures are useful in certain application domains. For instance, in the educational applications in which our team is interested, it would be impractical to have a separate set of potentially optimal students, kept “on the side” as a permanent archive to be periodically updated.

Indeed, when using software-based coevolution practice problems, students “come and go”, using the system as needed for their practice rather than abiding by rules ensuring proper evaluation of the practice problems. Unfortunately, conflating an archive and the population creates some search challenges when using particularly strict solution concepts, such as the Pareto solution concept. The algorithm should select individuals in the population for future generations to meet certain search needs: maintenance of diversity, pressure to make progress toward increasingly optimal solutions, etc. The algorithm should accept individuals into the archive based on the solution concept. In the case of purely Pareto-based approaches, we must accept individuals into the archive solely based on the idea that the archive has only non-dominated individuals but without preference among those.

However, in most search scenarios, there can be situations in which a vast number of candidates exist that are non-dominated. This is particularly problematic given that, in most cases, the population size is fixed, while an archive is typically permitted to grow. These two challenges, taken together, can create situations in which there is no selective pressure for the algorithm to make progress.

In recent work, new selection methods were introduced that relax the Pareto acceptance criteria for coevolutionary algorithms that combine the populations and the candidate/test archives as a means of increasing selective pressure [9]. One such method leveraged shared fitness among individuals with similar outcome vectors to enforce diversity within the population. Another method leveraged the NSGA-II selection model [37] in which non-
dominated individuals are assigned rank 1, then the individuals that are non-dominated after rank 1 individual removal are assigned rank 2, etc. The assigned rank may then be used to select individuals. These methods appeared to improve convergence of a coevolutionary system on well-known number games problems.

Nevertheless, these relaxations have issues that need to be addressed. First, even including maximum utility in the acceptance criteria can be insufficient since in many cases there are still large numbers of non-dominated individuals with the same number of successful outcomes against fixed-population size sets. Second, shared fitness approaches can introduce the artifact that non-dominated children in many circumstances can be less likely to be accepted than parents, creating new barriers against progress. Further, these methods introduce new parameters that must be set by a designer a priori, and there is little insight for how to do this effectively.

We investigate relaxed selection methods in more detail. We lay out CDE based performance metrics to compare results across different selection methods by heuristically measuring the exploratory and exploitative power of an algorithm empirically. We also introduce new methods to address issues of coevolutionary algorithms that combine population and archive by explicitly incorporating the probability of lateral, regressive, or progressive movement of individuals within the interaction space implied by the pre-orders induced by the comparisons seen so far. These movements are formalized and measured empirically on benchmark number game problems. We show that a hill climber style coevolutionary algorithm applying these selection methods is able to gain both exploratory and exploitative power over comparable algorithms that implement a stricter acceptance condition.

4.6 Experiment #11 - Stepping Stones of Interaction Space

In this section, we propose to adapt the interactive domains defined in [103] for test-based Pareto coevolution.
4.6.1 Interactive Domains and Interaction

The interactive domain for candidate and test roles can be defined as \( p : C \times T \rightarrow R \), where \( C \) and \( T \) are respectively the set of all possible candidates and the set of all possible tests. A tuple \((c, t) \in C \times T\) represents an interaction between candidate \( c \in C \) and test \( t \in T \). The outcome of such interaction is noted \( p(c, t) \) and is defined in an ordered set \( R \). For instance, we may define \( C = \{c_1, c_2, c_3, c_4, c_5\} \), \( T = \{t_1, t_2, t_3, t_4\} \), and \( R = \{1, 0\} \). Based on these sets, the outcomes of interactions may be defined as a matrix such as the one in Table 4.4.

**Table 4.4: Interaction between candidates vs tests**

<table>
<thead>
<tr>
<th></th>
<th>( t_1 )</th>
<th>( t_2 )</th>
<th>( t_3 )</th>
<th>( t_4 )</th>
</tr>
</thead>
<tbody>
<tr>
<td>( c_1 )</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>( c_2 )</td>
<td>0</td>
<td>1</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>( c_3 )</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>( c_4 )</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>1</td>
</tr>
<tr>
<td>( c_5 )</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
</tbody>
</table>

In this example, \( p(c_1, t_3) = 1 \) means that, when candidate \( c_1 \) interacts with test \( t_3 \), \( c_1 \) wins. Similarly, \( p(c_5, t_4) = 0 \) indicates that \( c_5 \) loses against \( t_4 \).

4.6.2 Interaction Space

We define the set of all interaction outcomes \( S = [01] \{m\} \) where 0 and 1 respectively correspond to “loss” and “win” outcomes. The notation \( \{m\} \) represents the number of interactions. For instance, \( S = [01] \{2\} \) corresponds to the set of all strings of length 2, each representing the two interaction outcomes; \( S = [01] \{2\} = \{00, 01, 10, 11\} \). Figure 4.3 illustrates such an interaction space for \( m = 4 \).

We may visualize all possible interaction outcome vectors of such a space by organizing them based on the number of “wins” they feature. Figure 4.3 shows such a visualization.
where the level $h$ represents the number of “wins” in the outcome vectors belonging to that level, and $N$ represents the number of outcome combinations in that level. Please note that, the interaction outcome vector at the highest level Pareto dominates all lower-level outcome vectors.

Let us assume that a candidate $c \in C$ interacts with all the tests. The size of each population is $m$. A given P-PHC-P individual, through the process of generating children and potentially being replaced by them, can be seen as following a path through the interaction space. There may be different such paths leading to the interaction outcome vector $1111$ that characterizes a candidate solution that wins against all tests. For instance, as an individual of the population is replaced by consecutive children, the corresponding interaction outcomes vector might follow the path: $1000 \rightarrow 1100 \rightarrow 1110 \rightarrow 1111$ or $1000 \rightarrow 1010 \rightarrow 1110 \rightarrow 1111$ or $1011 \rightarrow 1111$. We are thus interested in evolving candidate solutions along paths leading to the optimal solutions.

However, this evolution might also follow a path such as $1000 \rightarrow 1100 \rightarrow 1110 \rightarrow 1110 \rightarrow 1110 \rightarrow 1110 \ldots$. In this lineage, a parent, whose interaction outcome is $1110$, limits the possibilities of exploration inside the interaction space. Indeed, in P-PHC-P, the use of Pareto dominance as the only way for a child to be selected over its parent, would keep the candidate solution with outcome vector $1110$ in the population until its child features the outcome vector $1111$. Such an example illustrates that the probability of making progress can therefore be seriously limited. However, if we relax the selection condition, such that the search can proceed in other directions (e.g., $1110 \rightarrow 0111$ or $1110 \rightarrow 1011$) then the algorithm may avoid stalling altogether and reach the interaction outcomes vector $1111$; e.g., $1000 \rightarrow 1100 \rightarrow 1110 \rightarrow 0111 \rightarrow 1111$.

This idea is reminiscent of the role played by neutral networks in evolutionary dynamics. As was hypothesized by Kimura[76] in his neutral theory of molecular evolution, evolutionary dynamics may not be fully explained in terms of fitness gradients. Replacing a
genotype by another of equal fitness may have benefits in terms of population diversity that make such neutral transitions actually relevant.

4.6.2.1 Properties of Interaction Space

When visualizing the interaction space, as discussed in the previous section, the following observations come to mind;

- Let $m$ be the maximal level of the interaction space, then interaction outcomes vectors at a given level $h \in [0..m]$ features $h$ “wins” over the $m$ opponents. In section 4.7, we will use $h$ to determine whether a child is considered for replacement of its parent in the next generation.

- Let us assume that a parent creates a child and that both of them interact against same set of $m$ opponents. The relation between the level of the child’s interaction outcome vector $h_{V_{child}}$ and that of its parent $h_{V_{parent}}$ is $h_{V_{child}} \succ^m h_{V_{parent}}$, $h_{V_{child}} \sim^m h_{V_{parent}}$ or $h_{V_{parent}} \succ^m h_{V_{child}}$. The Pareto selection condition (as used in P-PHC-P admits a
parent to the next generation if $h_{\text{parent}} \succeq^{m} h_{\text{child}}$ or $h_{\text{parent}} \sim^{m} h_{\text{child}}$. The child replaces the parent in the next generation only if $h_{\text{child}} \succeq^{m} h_{\text{parent}}$. In [49], we derived the probabilities of such movements—upward, downward, horizontal—for P-PHC-P candidates when applied to a simulated practice-problem vs student interaction. In this work, we refer to these probabilities as $P_u$, $P_d$, $P_h$, respectively, and measure them for P-PHC-P.

4.6.2.2 Traversal of Interaction Space and Selection Conditions

The traversal of individuals into any one of the three directions depends on the selection criterion. Suppose that the interaction outcome of a parent candidate solution is 1100 (as shown in Figure 4.3, at $h = 2$). By only replacing a parent by its child if the later Pareto dominate the former, upward movement is only possible if the child’s interaction outcomes vector is one of $\{1110, 1101, 1111\}$. Let us now consider the following two selection conditions;

- This selection condition allows a child to traverse in both upward or horizontal directions in the interaction space. This first condition, $SC_1$, triggers if a) child Pareto dominates parent’s interaction outcome vector b) child and parent interaction outcome vectors both are mutually non-dominated.

- This condition ($SC_2$) is similar but also allows downward movement. A parent might be replaced by either a strictly better, a non-comparable, or an inferior child.

4.6.3 Methods

We measure the probabilities of downward, upward and horizontal movement for P-PHC-P when applied to FG and Comp-on-one.
4.6.4 Results

Table 4.5 shows the three probabilities $P_u$, $P_d$, and $P_h$ for P-PHC-P candidate solutions. The observed variations are similar among the two games. This is due to the fact that the comparison between the parent’s interaction vector vs that of its child is the same (i.e. based Pareto-dominance). This seems to take precedence over the specifics of the number game being used. Please note that the probabilities are measured for increasing population size, $m = \{2, 8, 16, 32, 50\}$. Lower values of $P_u$ indicate more frequent selection of the parent over the child.

Table 4.5: The upward, downward and horizontal movement probability of a child in P-PHC-P solutions under two number games.

<table>
<thead>
<tr>
<th>m</th>
<th>FG</th>
<th>Comp-on-one</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>$P_d$</td>
<td>$P_h$</td>
</tr>
<tr>
<td>2</td>
<td>0.022</td>
<td>0.971</td>
</tr>
<tr>
<td>8</td>
<td>0.089</td>
<td>0.902</td>
</tr>
<tr>
<td>16</td>
<td>0.167</td>
<td>0.824</td>
</tr>
<tr>
<td>32</td>
<td>0.276</td>
<td>0.711</td>
</tr>
<tr>
<td>50</td>
<td>0.371</td>
<td>0.615</td>
</tr>
</tbody>
</table>

4.6.5 Observations on Interaction Space Based Relax Conditions

The results provided in Table 4.5 illustrate the potential benefits of considering alternate selection conditions between parent and child in algorithms such as P-PHC-P. Most of the time, a selection condition solely based on strict Pareto dominance will lead P-PHC-P to select parents over children, thus stalling progress. Using the previously defined $SC_1$ and $SC_2$ allows the algorithm to reduce stalling.

These selection conditions thus enable a stronger exploration of the search space, while still exploiting the best candidate solutions in each of the (parent, child) pair. Please note that $SC_1$ satisfies the “strict acceptance condition” implied in Pareto dominance and
forbids downward movement. This selection criteria based algorithms may not maintain steady progress in some cases.

Although the performance of the SC2 based algorithm is improved, it does not guarantee steady improvement without (temporary) loss of quality. For instance, steady progress may be jeopardized when selecting a non-dominated child belonging to a lower level than its parent. Examples of such scenarios include;

- All the parents get replaced by their inferior children that are non-dominated because of downward movement but it has a low probability of occurrence.
- The best parents are replaced by their inferior children and there are no upward movements to compensate for the corresponding downward movements.

### 4.7 Experiment # 12 - Proposed Relaxed Selection Conditions

Section 4.6 discussed the need to relax the strict acceptance condition used in P-PHC-P. We now leverage the two proposed selection conditions SC1 and SC2 to implement P-PHC-P variants that we then experimentally evaluate.

#### 4.7.1 Selection Methods

**Table 4.6:** selection conditions for respective methods. If Ctrue then *parent* goes to the next generation regardless of the methods.

<table>
<thead>
<tr>
<th>Methods</th>
<th>Selection Condition for</th>
<th>Who is selected?</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td><strong>parent (pa)</strong></td>
<td><strong>child (ch)</strong></td>
</tr>
<tr>
<td>FRS</td>
<td>$C_{FRS} : V_{pa} \sim V_{ch}$</td>
<td>pa or ch</td>
</tr>
<tr>
<td>UHS</td>
<td>$C_{UHS} : V_{pa} \sim V_{ch}$</td>
<td>pa or ch</td>
</tr>
<tr>
<td>UHS-RI</td>
<td>$h(ch) \geq h(pa)$</td>
<td>pa or RI</td>
</tr>
<tr>
<td>UHS-QI</td>
<td></td>
<td>pa or QI</td>
</tr>
</tbody>
</table>
We use the P-PHC-P base selection method as reference and define below variant selection methods.

Full Relaxed Selection (FRS) is based on selection condition $SC_2$, as detailed in Algorithm 4.8. It selects a parent when it Pareto dominates its child, thus guaranteeing steady progress. We name this “Universal condition” and define it as $C_U$ in Table 4.6. However, FRS also considers scenarios where parent and child are mutually non-dominated. In this case, FRS can select a child either from the same level, or a lower level, of the interaction space than its parent. This is termed as $C_{FRS}$ (see Table 4.6). “Full” in FRS indicates that this method can select a child from any level - higher, lower or same - than its parent.

Upward Horizontal Selection (UHS) is based, along with the two variants defined below, on $SC_1$. We coined the name “Upward Horizontal” for it because a parent may be replaced by a non-dominated child from either the same level in the interaction space than its parent, or from an upper level. This constraint is defined as $C_{UHS}$ in Table 4.6. This selection scheme considers replacing a parent by its child when; a) parent Pareto dominates child, b) a non-dominated child is in the same level than its parent, or c) a non-dominated child is in a higher level than its parent. FRS behaves as UHS if a) there is no mutually non-dominated lower level child that will be selected to replace its parent in the next generation, or b) there is such a child but we replace its parent by a random individual. Algorithm 4.8 describes both FRS and UHS selection methods by changing the “condition” accordingly.

Algorithm 4.8 Generic-I (FRS, UHS)

<table>
<thead>
<tr>
<th>Generic-I $(V_{parent}, V_{child}, parent, child, p)$</th>
</tr>
</thead>
<tbody>
<tr>
<td>1: if Condition then ⊃ for FRS, Condition = $C_U$ OR $C_{FRS}$, for UHS, Condition = $C_U$ OR $C_{UHS}$</td>
</tr>
<tr>
<td>2: BaseReplacement $(parent, child, p)$</td>
</tr>
<tr>
<td>3: return true</td>
</tr>
<tr>
<td>4: end if</td>
</tr>
<tr>
<td>5: return false</td>
</tr>
</tbody>
</table>
The next two selections methods, UHS-RI and UHS-QI, are variants of UHS. As with the basic version of P-PHC-P, a parent is selected over its child if it Pareto dominates the latter. However, if the parent and child outcome vectors are non-dominated, then these selection methods replace the parent by a completely new individual. Depending on the variant, we refer to this new individual as a random immigrant (RI) or a quantum individual (QI) (see Table 4.6). While FRS and UHS promote either parent or child to the next generation, UHS-RI and UHS-QI either keep the parent or replace it, respectively, by the new RI or QI. The motivation is to favor exploration of the search space by considering new individuals when the algorithm’s progress would otherwise stall.

**UHS Random Immigrant** (UHS-RI) generates this new individual by simply randomizing it. The technique is inspired by Grefenstette’s work on random immigrants [54] that was applied to dynamical optimization problems. Algorithm 4.9 creates a random immigrant, it is called by Algorithm 4.11 to replace parents when “Condition \( C_{UHS} \)” is satisfied. Please note that, if \( C_U \) is satisfied, the parent is selected instead.

**UHS Quantum Individual** (UHS-QI) uses a technique inspired by [90] to generate a new individual that is randomized within the neighborhood of the current best candidate solution from the current population. This so-called Quantum Individual is therefore generated inside a ball centered on the current best solution. Let us define \( p_g \), the global best solution, and radius \( r \). Algorithm 4.10 creates a quantum individual within \( r \) and is called by Algorithm 4.11 to replace a parent when appropriate. Please note that, a Pareto dominant parent is always selected. We therefore aim at directing exploration toward high fitness areas of the search space. This additional exploitation dynamics may be adjusted by tuning \( r \).

**Algorithm 4.9** Create a Random Immigrant

getRandomImmigrant ()

1: Generate \( X \leftarrow x_i \) where, \( 1 \leq i \leq d \) \( x_i \in [\text{min, max}] \) \( d \) : dimension of the problem

2: return \( X \)
Algorithm 4.10  Create a Quantum Individual

getQuantumIndividual ()

1:  Generate \( x_i \in [\text{min}, \text{max}] \) for \( 1 \leq i \leq d \)

2:  Compute the distance of the point to the \((\text{min}, \text{min})\)  
    \( \text{dist} \leftarrow \sqrt{\sum_{i=1}^{d}(x_i - \text{min})^2} \)

3:  Determine the radius \( r \leftarrow [\text{min}, \text{max}] \)

4:  return \( X \)

4.7.2 Problems and Algorithms

We use FG game and use those various selections in P-PHC-P instead of its BaseSelection.

4.7.3 Results

Table 4.7 shows the performance of the selection methods used in P-PHC-P algorithm. Please note that all the selection methods cover both dimensions (so \( DC = 1 \) for all the methods). We used the Kruskal-Wallis post-hoc test for multiple comparisons with Bonferroni \( p \) value adjusted method. It seems that no method is better than Base Selection for ARR. However, UHS-RI has less duplicates in its final generation than Base Selection.

<table>
<thead>
<tr>
<th>Selection Methods</th>
<th>ARR</th>
<th>D</th>
</tr>
</thead>
<tbody>
<tr>
<td>Base Selection</td>
<td>0.1098</td>
<td>0.8538</td>
</tr>
<tr>
<td>FRS</td>
<td>0.1149</td>
<td>0.9655</td>
</tr>
<tr>
<td>UHS</td>
<td>0.1116</td>
<td>0.9669</td>
</tr>
<tr>
<td>UHS-RI</td>
<td>0.0600</td>
<td>0.5020</td>
</tr>
<tr>
<td>UHS-QI</td>
<td>0.0895</td>
<td>0.9227</td>
</tr>
</tbody>
</table>

Figure 4.4 tracks the progress of different selection methods. To do so, we average the sum of genes of the best individuals in each generation over the trials. We call this average “Mean Performance Indicator”. Note that this is an ad-hoc rough, linear estimator of distance to the optimum for these particular numbers games. In a multi-objective setting,
Algorithm 4.11 Generic-II (UHS-RI, UHS-QI)

Generic-II ($V_{parent}, V_{child}, parent, child, p$)

1: type ← \{RI, QI\}
2: if $C_U$ then
3:  BaseReplacement ($parent, child, p$)
4:  return true
5: else if $C_{UHS}$ AND type = QI then
6:  parent ← getQuantumIndividual()
7:  return true
8: else if $C_{UHS}$ AND type = RI then
9:  parent ← getRandomImmigrant()
10: return true
11: end if
12: return false

such a measure should not be construed as a true estimator of overall algorithm performance. High values indicate that the method is able to move in a direction that closes the distance to the global optima. Figure 4.4 shows that UHSelection, FRSelection and UHSelection-QI are better than Base Selection. The same \textit{p value adjustment} method was used for Performance Indicator also (see Table 4.8).

\begin{table}[h]
\begin{center}
\begin{tabular}{|c|c|c|c|c|}
\hline
Methods & BaseSelection & FRSelection & UHS & UHS-QI \\
\hline
FRS & < 0.01 & - & - & - \\
\hline
UHS & < 0.01 & 1.0 & - & - \\
\hline
UHS-QI & < 0.01 & < 0.01 & < 0.01 & - \\
\hline
UHS-RI & < 0.01 & < 0.01 & < 0.01 & - \\
\hline
\end{tabular}
\end{center}
\caption{Pairwise comparisons for Mean Performance Indicator. $0.01 \leq p \leq 0.05$ indicates statistical significance between pairs.}
\end{table}

4.7.4 Exploration vs Diversity in Relaxed Section Conditions

Relaxing the strict selection condition between parent and child in the P-PHC-P algorithm improved its ability to progress toward better solutions. Even more importantly, it did so by improving its ability to navigate the underlying search space. All the variants...
of the Base Selection algorithm that we discussed in so far provide a trade-off between exploitation (i.e., selecting based on strict Pareto dominance) and exploration (i.e., selecting a non-dominating child or replacing it altogether by a randomly generated candidate). This trade-off is the direct result of relaxing the initially strict selection condition.

4.8 Exp#13: Progress vs Distribution of Candidate Solutions in Pareto Layers

In this experiment, we analyze the progress of selection methods with respect to objective fitness. We also compare the distribution of candidate solutions in different Pareto layers.
4.8.1 Problems and Algorithms

We use three games FG, IG and Comp-on-one in this experiments. The P-PHC-P algorithm is used to implement both the base and relaxed selection schemes.

4.8.2 Methods

We measure objective fitness, sum of genes of the best candidate solutions in every generations for each selection method. The fitness is averaged over the independent trials for every generation to get a “generational” average fitness. This average fitness is plotted against generation for each of the three number games.

We also measure the distribution of number of Pareto layers, candidate distribution in Pareto front and that of the last layer (where inferior solutions exist). To do so, we store the interaction outcome of each candidate vs all the tests at the end of every independent trial. These are the outcomes for final generations i.e., from evolved candidate solutions in a trial. These distribution shows overall dispersion of the evolved solutions in the search space.

4.8.3 Results

Figure 4.5 shows the progress of P-PHC-P for both base and relaxed selections. The progress in Focusing game for base selection does not accelerate as it is in relaxed selection. The performance in Comp-on-one for relaxed selection is also better than base selection. In case of intransitive game, the progress in base is better than that of relaxed selection.

Figure 4.6 shows the distribution of candidate solutions in the Pareto front and the last layer. The distribution of number of Pareto layers in each selections are also shown. Maximum possible number of Pareto layer is 100, the size of population when every single candidate solution creates a layer maintaining a strict order with respect to Pareto optimality.
Figure 4.5: Monotonic progress of base selection. Relaxed selection shows steady improvements in three different number games.

The relaxed selection scheme allows for more Pareto layers to be represented in the population than the base selection one. The front in relaxed selection has less candidate solutions while the last layer has similar number of solutions.

The relaxation of strict acceptance condition allows for faster convergence while still maintaining steady progress. This is due to the solution concept - Pareto dominance - deployed into P-PHC-P algorithm. In addition, relaxed selection takes the advantages of aggregation when a (parent, child) pair for a candidate solution is Pareto Equivalent or Pareto non-comparable.

Relaxed selection explores the search space more than that of base selection due to the advantages of aggregation. The increased number of Pareto layers represented in the population when using the relaxed selection scheme can be explained by an increase in the algorithm’s ability to explore the search space. As a result, candidate solutions visits new region of search space which helps relaxed selection escape from local maxima.
4.9 Exp#14: Diversity and Convergence in Relaxed Pareto Coevolution

In previous work, we already compared population diversity when using the base selection scheme vs. the relaxed selections one [10, 9]. However, we only relied on a metric based on coevolutionary dimension extraction [23] as the basis for comparison. In this experiment, we analyze the diversity in terms of the concept of Hyper volume that is commonly used in the EMOO literature [107].

4.9.1 Problems and Algorithms

We use FG, and two variants of the P-PHC-P algorithm; one implementing the base selection scheme, and another implementing the relaxed selection scheme.
4.9.2 Methods

We use the same interaction matrix, $M_{100 \times 100}$, obtained at the end of each trial in Experiment #13. In both version of selections, the underlying algorithm is evolving candidate solutions by maximizing them against all the test solutions. So, a single row in $M_{100 \times 100}$ is treated as the 100 objective values against 100 tests for that specific candidate solution. However, we calculate the hyper volume for $m \in \{2, 4, 5\}$ objectives instead of $m = 100$ objectives. The reduction of $m$ is done by aggregation. Let us consider a candidate solution $C_1$, the first objective is given by $O_1 = \sum_{i=1}^{i=50} f(T_i)$ where, $T_i$ is the interaction outcome when $C_1$ interacts with $T_i$. Hence, the optimal outcome is to “win” against all the 100 tests and achieving $O_1 = 50$, $O_2 = 50$ for $m = 2$. Similarly, the objectives $O_{1...m}$ for $m = 4$ and $m = 5$ are defined where $i = 25$ and $i = 20$ respectively. This reduction technique is adopted from [146]. After this, $M_{100 \times 100}$ converts into $M_{100 \times m}$. We compute hyper volume for each of $M_{100 \times m}$ under both selection mechanisms. We use the pygmo library [15] to compute the hyper volumes and visualize the approximation set. Please note that all the experimental parameters are kept the same as in Experiment #13.

4.9.3 Results

Relaxed selection is found better than that of base selection. For illustration purposes, Figure 4.7 shows the Pareto layers for a random trial. In the case of relaxed selection, the approximated front is very close to the true Pareto front [50, 50]. On the other hand, relaxed selection does not reach to true Pareto front. The approximate set is far from true fronts.

Relaxed selection improves accuracy and diversity better than that of base selection. Both the hyper volume and visualization of non-dominated fronts prove the superiority of relaxed selection. The reasons for improving accuracy and diversity for converging populations in relaxed selection based Pareto coevolution are investigated. When using the relaxed
Figure 4.7: Different non dominated fronts in two objectives space for base and relaxed selection from an independent trial. Reference point is \([51, 50]\). Base selection is far away from true Pareto front.

selection scheme in P-PHC-P, we observe an increase in the hyper volume of the population. This also changes the distribution of Pareto layer in converging populations so that solutions are distributed to the new region of search space. We conjecture that this may be an essential factor in preventing this variant of P-PHC-P to get trapped in local optima.

Do we really need to keep solutions far away from true Pareto front because it is strictly better than its offspring? This study suggests that we should avoid that type of strictness because, keeping such solutions is a wastage of population slot. That is why, relaxed selection has higher hyper volume and better progress than base selection. Along with avoiding above extreme strictness, this study also suggests to consider the solutions
starting from approximated front regardless of being parent and child. Though NSGA-II [38] style based relaxed selection was considered previously, its monotonicity and progress need to be investigated further. In a separate line of thought, archive can be maintained in both population. Doing so will help to get rid of base selection and facilitate comparing an old parent solution with any other solution.
Chapter 5: Dominance Relations: Synthetic to EvoParson’s Interaction

5.1 Dominance Relations in Interaction Space

In test-based Pareto coevolution, the dominance relationship between two candidates is determined by comparing their respective interaction outcomes against the same set of tests. When candidate learners are compared across a vector of many test problems, the result can be more complicated than when compared against a single problem. A typical way to compare such candidates is to use the notion of Pareto dominance: A candidate dominates another candidate if it performs at least as well on all tests and better on at least one. Between two candidates \( A \) and \( B \), there are four possible dominance relations: \( A \) is not comparable to \( B \) (better on some, worse on others), \( A \) dominates \( B \), \( A \) is dominated by \( B \), and the two candidates perform identically. Different coevolutionary problems will have different ratios of such relationships, depending on the structural and relational properties of the problem.

We propose to take a closer look at the relative proportions of occurrence of each of these relationships. In the number games commonly studied in the coevolutionary computation literature\cite{141, 21, 35, 65} such proportions directly result from the definition of the interaction function.

However, in some complex applications, the interaction functions produces uncertain outcomes or may not be able to produce an outcome at all for some candidate-problem interactions. Moreover, in some cases a deterministic and complete interaction function cannot be formulated. It is unclear how coevolutionary systems should handle such cases;
however, a good first step is to begin to understand how sensitive the underlying structure of a game is to uncertain and missing outcomes.

Of particular interest to our team is the scenario in which human students try to solve practice problems. In this particular educational application, practice problems are evolved so as to differentiate students based on their performance. Such a space is inherently noisy, and interactive coevolutionary methods in general will have to deal with such noise.

In this chapter, we first focus on problems in which interactions between candidates and tests can be captured in a binary interaction matrix. We will refer to such matrix as the “structural matrix” \((M_{m \times n})\). \(M_{m \times n}\) is obtained by having \(n\) candidates interact against \(m\) tests under a given number game. In addition, we also define a “dominance relational matrix” corresponding to each structural matrix. The dominance relations especially non-comparable relation between two tests vary drastically based on interaction function and also if any single alteration of interaction outcome happens. So, study of dominance relations between two tests may reveal interesting findings about underlying interaction dynamics. It also determines the applicability of CDE algorithm under a certain level of alteration of interaction outcome. To this end, dominance relational matrix stores the Pareto dominance outcome for the tests.

We use the dominance relational matrix of a given problem to visualize the effects of damaging the information contained in its structural matrix by converting a “win” to a “loss” or vice-versa. The motivation behind such an analysis is that such damage, or missing information in the structural matrix is a common occurrence in realistic applications such as “EvoParsons”.

Second, we train a MLP [55] to approximate the interaction function of different coevolutionary number games. The idea is to use partial available information to repair a damaged structural matrix. We use the distributions of each type of dominance relation as they occur in the dominance relational matrix after repairing the corresponding damaged
structural matrix with the above machine learning model as one means of determining the effectiveness of the repair.

Third, we leverage the insights gained in the above-discussed steps to investigate both the structural and dominance relational matrices that were obtained when using a more realistic, EvoParsons system.

In EvoParsons interaction, a full interaction matrix is not feasible that we find in coevolutionary number games. However, the interaction matrix contains some small submatrices which are dense i.e., contain outcome for all the interactions. To this end, we extract all possible dense sub matrices from the interaction matrix obtained from EvoParsons’s evaluation by real human students conducted in an Introductory programming course at University of South Florida.

Then we analyze the distribution of dominance relations of each of the sub matrices. Such analysis is also extended into full interaction matrices of EvoParsons evaluation obtained from different experimental settings. The result is a deeper insight into the effects of noisy outcome spaces on coevolutionary systems, some potential means of mitigating these effects, and feedback about when such mitigation is feasible.

We also validate EvoParsons’s efficacy and conclude our discussions with future research directions.

5.2 Experiment #15 The Effect of Flipping Interaction on Dominance Relations

This experiment investigates the effects of altering the interaction outcomes (i.e., flipping a “win” into a “loss”, or vice versa) inside the interaction matrices obtained from different number games. More specifically, we focus on the effect that such modifications have on the underlying dominance relations.
5.2.1 Problems

For this experiment, we chose to use the three number games defined in Section 1.3; FG, IG, and Comp-on-one.

5.2.2 Methods

Our intent is to measure the impact that flipping interaction outcomes has on the underlying dominance relations. To this end, we propose to start by randomly altering a certain number of outcomes in the structural matrix. We then observe the resulting changes of the relations in dominance relational matrix. Observations are based on both visualizations of the changes, and metrics commonly used when working with differences between images. Formally, the structural matrix can be defined as;

\[
S_{c \times t} = \begin{cases} 
1 & \text{if } c \text{ "wins" against } t \\
0 & \text{otherwise}
\end{cases}
\]

In the above, \(S_{c \times t}\) is a matrix of candidates by tests and each cell indicates if c “wins” or “loss” against \(t\).

Similarly, we can also define the relational matrix \((R_{t \times t})\), \(x = \{1, 2, \ldots, t\}\), as a test by test matrix. Please note that \(R\) can similarly be defined for candidates. Each cell indicates any one of the four possible relations between any pair of tests \((t_1, t_2)\); \(t_1 \succ t_2\), \(t_2 \succ t_1\), \(t_1 = t_2\), and \(t_1 \sim t_2\). We encode these relations in the matrix with respective values.
1, −1, 0 and −2. $R_{t_x \times t_y}$ can therefore be defined as follows;

$$R_{t_x \times t_y} = \begin{cases} 
1 & \text{if } t_x > t_y \\
-1 & \text{if } t_y > t_x \\
0 & \text{if } t_x = t_y \\
-2 & \text{if } t_x \sim t_y 
\end{cases}$$

For our purpose, it is relevant to investigate the effect of flipping interaction outcomes in $S_{cxt}$, in terms of their impact on $R_{t_x \times t_y}$. This observation is important because $R_{t_x \times t_y}$ can have different impacts for the different problems (e.g., number games, EvoParson’s interaction) we are dealing with. This also helps us to determine the applicability of CDE in flipped interactions of different problems.

With that in mind, we analyze the relations in $R_{t_x \times t_y}$, especially “non-comparable”, change with different amount of flipping applied to in $S_{cxt}$. To introduce noise, we flip binary outcomes in $R_{t_x \times t_y}$ uniformly at random i.i.d. with different levels of probability. We then measure how each of the four dominance relations is affected as we increase the proportion of outcomes being flipped.

Flipping a given proportion of the outcomes in the interaction matrix of a number game is analogous to situations where a digital image is damaged by randomly modifying pixels’ values. This motivated us to adopt two common digital image quality assessment metrics: the Structural Similarity Index (SSIM) [148], and the most widely used Mean Squared Error (MSE). These metrics have potential to help us better understand the impact of introducing noise in $S_{cxt}$, on the dominance relations depicted in $R_{t_x \times t_y}$.
SSIM: Suppose $x$ and $y$ are two non-negative image signals;

$$SSIM(x, y) = \frac{(2\mu_x\mu_y + C_1)(2\sigma_{xy} + C_2)}{\left(\mu_x^2 + \mu_y^2 + C_1\right) \left(\sigma_x^2 + \sigma_y^2 + C_2\right)}$$

where $\mu_x$ and $\mu_y$ are mean intensity of image signal $x$ and $y$ respectively. $\sigma_x$ and $\sigma_y$ are respective standard deviation. Lastly, $C_1$ and $C_2$ are two constants.

MSE: The mean squared error is the average of the squares of the errors between two pixels $x$ and $y$;

$$MSE = \frac{1}{MN} \sum_{n=1}^{M} \sum_{m=1}^{N} [x(n, m) - y(n, m)]^2$$

where each image has $M \times N$ pixels.

In the following, we refer to the proportion of outcomes flipped as the “damage ratio” being applied to the interaction matrix. During the experiment, we considered a range of values for this damage ratio $d_r = \{0.0, 0.01, 0.03, 0.05, 0.07, 0.09, 0.10, 0.15, 0.20, 1.0\}$, where 0.0 and 1.0 indicate that $S_{cxt}$ is, respectively, the original and fully damaged matrix. The random flipping for a specific value of $d_r$ was repeated over 1,000 independent trials.

Please also note that;

- We choose problem dimension, $D = 2$.
- Problem size varies depending on the analysis we do. To visually observe the original structural matrix or dominance relations matrix, their flipped version and difference, we use a small $M_{9x9}$ (problem size = 9). On the other hand, the problem size is larger (100) when we measure difference between image quality matrices and analyze the distribution of relations in dominance relational matrix.
5.2.3 Results

5.2.3.1 Visualizing Structural vs Dominance Relational Matrix

For the purpose of visualization of original structural matrix/relational matrix and their corresponding counterpart flipped matrix, we plot them in Figure 5.1 for a single sample of small matrix \((S_{c=9} \times t=9)\). The left half focuses on the structural matrices for three number games (each on their own row), while the right half shows the corresponding relational matrix. In each half, the first column shows respectively the original structural matrix and its corresponding relational matrix. The second column shows the result of flipping a portion of the outcomes in the structural matrix, and how it affected the corresponding relational matrix. The third column captures the differences between the matrices in the first and second column, to make it easier to visualize the impact.

In case of relational matrix, the change between original and relational matrix can be easily observed. Though there are differences in Comp-on-one and FG, IG does not show as much changes in flipped matrix compared to others.

To understand the effect of damage quantitatively, we use the previously discussed metrics; SSIM and MSE. As would be expected, when increasing the damage ratio in the structural matrix, MSE increases and SSIM decreases, thus indicating increased differences (see Figure 5.2). On the other hand, the changes in errors and similarity indexes for relational matrix are not as straightforward; they remain fixed up to high damage ratio. Interestingly, the error decreases when the original structural matrix is fully damaged.

It is important to keep in mind that the error and similarity index measured in the dominance relational matrices vary based on underlying dynamics of number games. For instance, both metrics feature lesser values in the case of the intransitive game as opposed to the comp-on-one and focusing games.
5.2.3.2 Distribution of Dominance Relations

As discussed earlier, damage applied by flipping interaction outcomes in a structural matrix, impacts dominance relations among solutions. Figure 5.3 shows the proportion of occurrences of the $a \sim b$ relation in the relational matrix for a comp-on-one number game, under different damage ratio. We increase problem size from $9 \times 9$ to $100 \times 100$ because we want to minimize the overlapping of random damage. We can observe that the number of non-comparable tests in relational damage increases rapidly even for small damage in the structural matrix.

Figure 5.4 summarizes mean frequency of $a \sim b$ dominance relations for $d_r = [0.0\ldots0.20]$ for three different number games. As we increase $d_r$, more tests become non-comparable.

![Figure 5.1:](image)

Figure 5.1: Structural and relational matrix, their respective damaged and difference matrices for the three number games.
Please note that, the mean number of $a \sim b$ reaches approximately the highest possible value for smallest $d_r = 0.15$.

This suggests that even a small amount of damage in a structural matrix may have a great impact in the dominance relational matrix in terms of the proportion of $a \sim b$ relations. The mean number of non-comparable relations increases rapidly and reaches its maximum possible value even for a small amount of damage in structural matrix.

As would be expected, both MSE and SSIM have linear relations with the damage ratio. No such relation is found for relational damage. Interestingly, the error decreases when we apply the large damage (e.g., greater than 90%) to the structural matrix. Although each number game features different MSE and SSIM measurements for their dominance relational matrix, the mean number of non-comparable relations reaches its maximum for $d_r = 0.15$

Figure 5.2: MSE vs SSIM for structural and dominance relational matrices shown in first and second column respectively for $d_r$ from 0% to 100%.
for all of them. We find that the non-comparable relation is very sensitive to small amount of damage.

We found that even a small amount of damage applied to the structural matrix has a significant impact on the underlying relational information, especially in terms of the proportion of non-comparable relation. The proportion of occurrences of this relation in the relational matrix doubles even after a very small amount of damage (e.g., 3%) in the structural matrix. This proportion reaches its maximum after only 9% of damage (see Figure 5.4).

**Figure 5.3:** Distribution of \( a \sim b \) relations for comp-on-one in different damage ratio. Interaction matrix is obtained from 100 candidates interact against each of the 100 tests. Median number of \( a \sim b \) reaches approximately maximum possible values for a small damage of 0.15.
Figure 5.4: Damage ratio vs mean frequency of $a \sim b$ for three number games. There are total 10,000 possible relations for all four types of dominance relations.

Any single change in the same objective value can change the dominance relation between two tests from “one dominates other” to mutually non-dominant. In addition, the probability of such a transition increases when the solutions are trying to optimize against many objectives.

This suggests that, applying CDE even to a slightly damaged structural image may result in radically different results than applied to the undamaged image. In extreme cases, it may uncover little structure in the space at all (e.g., all tests become singleton dimensions). This hinders CDE’s applicability to interaction matrices where even only a small amount of damage may be present.
Also, we found that this increase in the proportion of non-comparable relations in the relation matrices, as the amount of damage increases, is independent of the particularities of the underlying number game. Independently of the number game being considered, a small amount of damage in the structural matrix is enough to get most of the elements in the relational matrix non-comparable.

Therefore, it is reasonable to conclude that, when dealing with an interaction matrix in which an unknown level of damage is present (e.g., noise), we would require a technique with very high accuracy to repair the damaged structural matrix.

5.3 Experiment #16 - Learning Underlying Interaction Function of Number Games

The previous experiment showed that, even small amount of damage in structural the matrix increases frequency of $a \sim b$ relations rapidly. However, we can repair the damaged structural matrix and investigate whether we are able or not to preserve the original proportions of, for instance, non-comparable relations. Theoretically, the method needs to be 100% accurate to restore the original structural image. However, any method that can even restore approximately 97% to 99% interaction of structural image may be applicable for repairing the damaged image. This is because the previous experiment shows that the frequency of non-comparable relations goes high even for small amount like 3% of damage in structural image. To do so, this experiment trains an MLP to approximate the interaction function of coevolutionary number games. The motivation is to approximate interaction function of different number games and also observe the relations in repaired image.

5.3.1 Problems

In this experiment, we focus on the four number games defined in section 1.3.
5.3.2 Methods

We represent each interaction, \( C \times T \rightarrow R \), as an ordered pair of \((x, r)\). The training set contains \( N \) such interactions:

\[
X = \{x^i, r^i\}_{i=1}^N
\]

\( x \) and \( r \) are defined as follows:

\[
x = [C, T]
\]

and

\[
r = \begin{cases} 
1 & \text{if } c \text{ "wins" against } t \\
0 & \text{otherwise}
\end{cases}
\]

Here, \( C \) and \( T \) are candidates and tests both represented by tuples of \( n \) integer values, \( |C| = |T| = N \). So, each sample has \( 2n \) features and \( r \) represents the corresponding label or class of that sample. We train MLP with different amounts of training data \((t_r)\) randomly selected from \( N \) samples. Then the MLP is used to approximate the interaction outcome of respective \( 1 - t_r \) data. While training the MLP, we choose different amounts of training data \((t_r = \{0.05, 0.1, 0.2, 0.3, 0.4, 0.5, 0.6, 0.7, 0.8, 0.9\})\) randomly selected from \( N \). We then evaluate the accuracy of the MLP on the remaining data \((1 - t_r)\). We used 1,000 independent trials in our experiment. The dimension of the problem is \( n \), the problem size to \( 100 \times 100 \). The trained MLP featured one hidden layer containing 100 neurons.

5.3.3 Results

Figure 5.6 shows the mean MLP accuracy for each of the number games. The data not used as training set is used to test the resulting MLP. Mean test accuracy of MLP is equal or higher than 99.0% for all number games. The mean test accuracy also increases
when we have more known data points. The proportions of occurrences in the dominance relational matrices of all four dominance relations are also measured for each number game. $a \sim b$ relation does not change rapidly as observed in Figure 5.5. Please note that, mean value of $a \succ b$ and $b \succ a$ are same. As a result, their curves overlap.

![Figure 5.5: Structural matrix is repaired using MLP assuming we have access to small number of interactions of a $100 \times 100$ interaction matrix. MLP is trained by that accessible data and infer rest of them as unknown or inaccessible. Mean values of all four relations is shown for each number games.](image)

The MLP learns the interaction outcome function of each of the number games with a very high accuracy equal or greater than 99.0%. As a result, the mean value of the proportion of dominance relations occurring in the dominance relational matrix remains closer to the original proportions.

The dominance relations also depend on the nature of number games. For example, MLP learns intransitive function with accuracy higher than 99.0% but still its mean value of $a \sim b$ relations remain high after repair.
Our results suggest that the interaction functions of a wide range of coevolutionary number games can be learned effectively (i.e., high accuracy, small percentage of total number of interactions available) by a simple MLP model. It seems that problems that can lead to cycling pathologies (exhibited by intransitive game) may be more vulnerable and sensitive to damage than problems leading to overspecialization (shown by FG or Comp-on-one game).

However, a simple MLP may be trained on small amounts of interactions to extrapolate the missing interactions. This, in turn, opens the possibility of repairing interaction matrices in problems such as EvoParsons in order to apply algorithms like CDE to evaluate

Figure 5.6: Mean test accuracy of MLP for different number games. %known data represents the amount of training data ($t_r$) out of 10,000 data point. Corresponding rest of the data points, ($1 - t_r$), are used as test data.
the coevolutionary progress achieved. It also opens the door to further mitigate user fatigue by leveraging surrogate fitness techniques to simulate student-problem interactions.

5.4 Experiment #17 - Dominance Relations in EvoParsons Interactions

This section analyzes the dominance relations from interaction matrices obtained in the context of our EvoParsons educational application.

5.4.1 Problems

When coevolving a population of genotypes encoding practice problems with a population of learners, there is no mathematical formulation of the underlying problem available. EvoParsons is an educational tool that evolves Parsons puzzles based on Pareto coevolution. The practice problems are evaluated via their interaction with actual students. Figure 3.2 illustrates the overall architecture of the EvoParsons system.

5.4.2 Algorithms

We use P-PHC-P\textsuperscript{48} to evolve Parsons puzzles. The parent and child puzzles are compared based on Pareto dominance in order to determine which one is selected for next generation. Please note that the student population co-adapted to EvoParsons’ population of Parsons puzzles.

5.4.3 Methods

We ran two regional experiments as described in 3.3.2.
5.4.4 Results

5.4.4.1 Run #1

The size of the interaction matrix from Experiment #1 is $M_{\text{student} \times \text{puzzle}} = 88 \times 74$. Since $M_{\text{student} \times \text{puzzle}}$ is sparse, we extract the largest sub-matrices of dimension $x \times y$, where $x$ is for student count and $y$ represents the number of parsons puzzles with which these $x$ students interacted. Starting from $x \times y = 5 \times 5$, 100 such sub-matrices were found, the largest being of dimension $x \times y = 6 \times 10$. Of them, there are 38, 32, 18, 9 and 1 sub matrices where puzzle size was 5, 6, 7, 8, 9 and 10 respectively. Figure 5.7 shows the proportions of relations of three dominance relational of each of the sub matrices found.

The number of students that interacted with those puzzles are in the range of $[5, 10]$. The dimension of relational images for those sub matrices are $5 \times 5$, $6 \times 6$, $7 \times 7$, $8 \times 8$, $9 \times 9$ and $10 \times 10$ respectively.

The proportions of relations occurring in each dominance relational matrix under a specific dimension, are averaged over the number of matrices. For example, we find nine sub matrices of $M_{8 \times x}$, where $x \in [5, 10]$. These nine sub matrices results in nine $8 \times 8$ dominance relational matrices for puzzles. All the four relations are averaged over these nine sub matrices. There exists a total of 64 dominance relation between puzzles for an $8 \times 8$ sub matrix. Figure 5.7 shows that the non-comparable relations does not converge towards its maximum values. For example, approximately 55 out of 81 relations are non-comparable for a $9 \times 9$ relational image.

Also, each $m[i, j] \in \mathbb{R}$ in $M_{88 \times 74}$ is an interaction outcome that is a relative move requires to solve puzzle $j$ by the student $i$. To measure the dominance relations, we convert each of the sub matrices into binary. To do so, we define “win” for a student if $m[i, j] \leq \mu$ where $\mu$ is the average of relative moves to solve puzzle $j$ by all the students. Otherwise, student $i$ looses against puzzle $j$ if $m[i, j] > \mu$. 

157
5.4.4.2 Run #2

In this experiment, 61 students solved each of the 10 puzzles. At first, we see the relative move distributions for each of those puzzles. The median of relative moves are in [1.1, 1.7]. There are some outliers for each of the puzzles (see Figure 5.8).

Then we convert this $M_{61 \times 10}$ interaction matrix into a binary outcomes matrix. This time, we not only consider $m[i, j] \leq \mu$ for “win” but also extend the range from $-1$ standard deviation to $+1$ standard deviation. More specifically, “win” and “loss” are defined as follows;

$$
\text{outcome} = \begin{cases} 
\text{“win”} & \text{if } m[i, j] \leq \mu + fr \ast \text{std} \\
\text{and } m[i, j] \geq \mu + fr \ast \text{std} \\
\text{“loss”} & \text{otherwise}
\end{cases}
$$

Figure 5.7: The average proportions of relations of all four relations from different relational images obtained from interaction matrix in Experiment #1.
where \( fr = \{0.0, 0.01, 0.02, \ldots, 0.8, 0.81, \ldots, 0.99, 1.0\} \) and \( std \) is the standard deviation of relative moves for a specific puzzle. This results in a total of 100 binary interaction matrices where “win” and “loss” are defined for various interval of relative number of moves. We then calculate the proportions of relations of all four relations for each of those dominance relational matrices obtained from the corresponding \( M_{61 \times 10} \) binary interaction matrix. We average the count of each relations over the total number of relational image.

While the distribution for \( a\)-dominates-\( b \) and \( b\)-dominates-\( a \) are same, \( a\)-non-comp-\( b \) relations are different than the previous two dominance relations. The mean proportion of relations of \( a\)-non-comp-\( b \) relations reaches nearly the maximum possible values as found in the dominance relational matrix analysis for intransitive game in Section 5.2.

**Figure 5.8:** Left: Distribution of relative moves for 10 puzzles over 61 students. Right: Distribution of all the four relations (1 = \( a\)-dominates-\( b \), 2 = \( b\)-dominates-\( a \), 3 = \( a\)-equals-\( b \) and 4 = \( a\)-non-comp-\( b \)).
The number of objectives against which two solutions are Pareto compared plays a role in their dominance relations. These pair of competing solutions try to optimize against many (as large as population size in Pareto coevolution) objectives. So the number of non-comparable relations are high in the dominance relational matrix.

Distributions of all four dominance relations also depend on the underlying payoff functions of interactions. Interactions in EvoParsons can’t be mathematically formulated. We can study the dominance relations and compare with that of coevolutionary number games. Experimental results show that the change of proportion of relations of non-comparable relations in smaller sized relational matrix is much less than that of larger relational image. In case of larger relational image, the distribution of non-comparable relations is similar to intransitive number games.

5.5 Findings in Experiments #15 - #17

In this work, we showed that Pareto dominance relations change based on the underlying interaction function. Coevolutionary number games producing cycling pathologies are more sensitive to changing the nature of the interactions than those exhibiting overspecialization pathologies. So study of dominance relations in coevolutionary number games mostly depends on the inherent properties of the interaction dynamics for which the game is originally designed.

Pareto coevolution being a form of many objective optimization can be demonstrated to study the artificial interaction dynamics and their dominance relations. When applying coevolutionary optimization in EvoParsons, we need a size of many objectives (student) against whom each pair of puzzles optimize (Run #1). This is because Parsons puzzles that are evaluated by large number of objectives are mostly non-comparable (Run #2) Many objectives also have side effects in applying CDE to underlying full interaction matrix. The
more non-comparable solutions, the more susceptible degeneration of the interaction space structure in CDE analysis.

This study is the first to examine the dominance relations of coevolutionary number games. It also measures the effect of noisy interaction in the distribution of dominance relations. The observation of dominance relations in number games help to understand the underlying interactions of the problem as a proxy for situations in which the problem itself can not be mathematically formulated. In such cases, there is hope that methods to repair uncertain or missing outcomes may be feasible.
Chapter 6: Conclusions

We conclude this dissertation by summarizing its three contributions then discuss limitations as well as possibilities for future research on this topic.

6.1 Contributions

- Authoring Underlying Algorithms for Coevolutionary Interactive Systems: We develop variants of population based hill climber algorithms and compare their performance under different design criteria (coevolutionary pathology, noise, gene bounds). These criterion are minimal requirements to develop coevolutionary interactive system. The choice of appropriate solution concepts for the underlying algorithms for such interactive systems can be summarized in Figure 6.1. We find that Pareto solution concept is better to use than aggregation when evolving candidate solutions for our educational application.

The pedagogical analysis of our target system’s interaction matrix under CDE algorithm shows that the evolved solutions present in the last generation cover all the dimensions of the corresponding CDE’s coordinate system. The evolved solutions also hold a high rank on these dimensions. However, we also found some of these solutions feature the same interaction outcomes than others. The presence of such duplicates indicates that the underlying algorithm can still be further improved by better managing the diversity in its population.
Figure 6.1: Decision between Pareto and Aggregation solution concepts under coevolutionary pathologies, noise and gene bounds. Pareto dominance solution concept is a better choice in several situations than aggregation.

- Improving Diversity for Variants of Coevolutionary Hill Climbers: We relax the “strict acceptance condition” of Pareto dominance while comparing interaction outcomes between a parent and its corresponding child candidate solution. Relaxing such conditions promotes child candidates that helps the coevolutionary search to explore new region of the search space. We propose three different methods of such relax condition based coevolutionary variant algorithms.

  - NSGA-based relaxed selection: Instead of accepting the Pareto optimal between parent and child solution, we maintain an archive of population of size 2N (the original population size is N) and select the N solutions staring from the Pareto front. This enables non-comparable child solutions also to be included in the next generations.
– Competitive shared fitness based relaxed selection: This method rewards unusual solutions that exhibits a new feature in search space. In this method, we promote non-comparable child solutions in the next generation if they hold interesting features e.g., winning against a particular opponent whereas the other solutions in the population loose against that opponent.

– Introducing random and quantum individuals in the population: Instead of promoting a non-comparable child solution, we introduce new random and quantum solutions and replace the child with such newly introducing solutions.

• Methods for Repairing Interaction Matrix: CDE analysis requires full interaction matrix to create the coordinate system. Building such interaction matrix is feasible for coevolutionary benchmark problems. However, it is impossible to build such full interaction matrix when solutions are evaluated by human agent. In this final contribution, we analyze the effects of flipping the interaction outcome of benchmark problems and observe the changes in the frequencies of four dominance relations under the Pareto dominance solution concept. This unique approach reveals that even a small amount of damage, as low as 15% significantly affects proportion of non-comparable relations in the number games being investigated. We train a multilayer perceptron to understand the pay-off functions of coevolutionary benchmarks and use it to repair the damaged interaction matrices. The respective relational matrices still show high frequencies of non-comparable solutions. Interestingly, the non-comparable dominance relations in our target system’s sub-matrices also exhibits higher frequencies.

6.2 Future Works

• The qualitative analysis of candidate solutions evolved by the underlying algorithm of our target system shows the presence of misconceptions commonly found in the
concept inventories of computer programming. Currently, our system's programs and
distractors are written only for Java programming. We can extend the scope for other
computer programming language; even for other disciplines.

- The sparsity of interaction matrix is high (89%) because we need to balance between
the speed of evolutionary process and the number of interactions. Though some of the
selection methods are introduced to increase the local density of interaction matrix, we
need approaches e.g., missing value imputation methods to reduce the global sparsity.

- The CDE analysis are done offline so far i.e., we analyze the interaction matrix under
CDE after the end of evolutionary process. However, we can apply CDE online if the
generated interaction matrix shows less sparsity.
References


[114] Jimmy Secretan, Nicholas Beato, David B D Ambrosio, Adelein Rodriguez, Adam Campbell, and Kenneth O Stanley. Picbreeder: evolving pictures collaboratively on-


[129] Tse Guan Tan, Jason Teo, and Hui Keng Lau. Augmenting spea2 with k-random competitive coevolution for enhanced evolutionary multi-objective optimization. In


Appendix A: Copyright Permissions

The permission below is for the use of material in 2.1 to 2.5.

Thesis / Dissertation Reuse

The IEEE does not require individuals working on a thesis to obtain a formal reuse license, however, you may print out this statement to be used as a permission grant:

Requirements to be followed when using any portion (e.g., figure, graph, table, or textual material) of an IEEE copyrighted paper in a thesis:

1) In the case of textual material (e.g., using short quotes or referring to the work within these papers) users must give full credit to the original source (author, paper, publication) followed by the IEEE copyright line © 2011 IEEE.
2) In the case of illustrations or tabular material, we require that the copyright line © [Year of original publication] IEEE appear prominently with each reprinted figure and/or table.
3) If a substantial portion of the original paper is to be used, and if you are not the senior author, also obtain the senior author’s approval.

Requirements to be followed when using an entire IEEE copyrighted paper in a thesis:

1) The following IEEE copyright/credit notice should be placed prominently in the references: © [year of original publication] IEEE. Reprinted, with permission, from [author names, paper title, IEEE publication title, and month/year of publication].
2) Only the accepted version of an IEEE copyrighted paper can be used when posting the paper or your thesis online.
3) In placing the thesis on the author’s university website, please display the following message in a prominent place on the website: In reference to IEEE copyrighted material which is used with permission in this thesis, the IEEE does not endorse any of [university/educational entity’s name goes here]’s products or services. Internal or personal use of this material is permitted. If interested in reprinting/republishing IEEE copyrighted material for advertising or promotional purposes or for creating new collective works for resale or redistribution, please go to http://www.ieee.org/publications_standards/publications/rights/rights_link.html to learn how to obtain a License from RightsLink.

If applicable, University Microfilms and/or ProQuest Library, or the Archives of Canada may supply single copies of the dissertation.
The permission below is for the use of material in 2.6 to 2.11.
Currently, I am in need to include this publication into my Ph.D. dissertation. Would you please let me know the process of getting the permission? I have provided the conference information as follows; Please let me know if you have any questions.

Conference title: 30th International Conference of Florida Artificial Intelligence Research Society
Paper Title: Evolutionary Practice Problems Generation: More Design Guidelines
Authors name: Alessio Gaspar, ATM Golam Bari, R. Paul Wiegand, Anthony Bucci, Jennifer Albert, Amruth N. Kumar.

Carol Hamilton <hamilton@aaai.org> Thu, Oct 3, 2019 at 3:28 PM
To: bari@mail.usf.edu
Cc: Carol Hamilton <hamilton@aaai.org>, AAAI Press <publications19@aaai.org>

Dear Golam,

In the Returned Rights section of the AAAI copyright form, authors are specifically granted back the right to use their own papers for noncommercial uses, such as inclusion in their dissertations or the right to deposit their own papers in their institutional repositories, provided there is proper attribution. The published version is not available for posting outside the AAAI Digital Library.

No further permission is required.

Best regards,

Carol Hamilton

ATM Golam Bari <bari@mail.usf.edu> Thu, Oct 3, 2019 at 3:35 PM
To: Carol Hamilton <hamilton@aaai.org>
Cc: Carol Hamilton <hamilton@aaai.org>, AAAI Press <publications19@aaai.org>

Thanks for the prompt response.
So, it means I can use the published paper in AAAI in my dissertation assuming I also attach the following copyright form in my dissertation?

[Quoted text hidden]

AAAI Press Copyright Form.pdf
38K

AAAI Press <publications19@aaai.org> Thu, Oct 3, 2019 at 4:14 PM
To: ATM Golam Bari <bari@mail.usf.edu>

Dear Golam,

https://mail.google.com/mail/u/0?ik=a0d33f2159&search=all&permthid=thread-f%3A1646401132153829247&aimpl=mail-f%3A164640113215...
You don’t need to attach the copyright form to your dissertation. What you need to do is, after the title page (or wherever the copyright notice appears), reproduce the following text:


I am assuming you reproduced the whole paper, if you only reproduced portions, then put:

The excerpts appearing in this dissertation have been reproduced with permission.

Regards,

Mike Hamilton
AAAI Press

Intelligence Research Society

ATM Golam Bari <bari@mail.usf.edu> Thu, Oct 3, 2019 at 4:16 PM
ton: AAAI Press <publications19@aaai.org>

Thank you for your reply, I will do that.

[Quoted text hidden]
The permission below is for the use of material in Chapter 3.

Hi,

One of my journals (Genetic Programming and Evolvable Machines) is published under SpringerLink. Currently, I am in need to include this publication into my Ph.D. dissertation. Would you please let me know the process of getting the permission?

Please find the journal information below;

Title:
EvoParsons: design, implementation and preliminary evaluation of evolutionary Parsons puzzle


Journal name: Genetic Programming and Evolvable Machines;

Dear Golam,

Thank you for your Springer Nature permissions query. Author retains the right to use his/her article for his/her further scientific career by including the final published journal article in other publications such as dissertations and postdoctoral qualifications provided acknowledgement is given to the original source of publication.

You do not need permission to reuse your article in a thesis/dissertation.

Best regards,

Paloma Hammond

Rights Assistant

SpringerNature
The permission below is for the use of material in 4.1 to 4.4.
The permission below is for the use of material in 4.5 to 4.7.
The permission below is for the use of material in 4.8 to 4.9

Thesis / Dissertation Reuse

The IEEE does not require individuals working on a thesis to obtain a formal reuse license, however, you may print out this statement to be used as a permission grant:

Requirements to be followed when using any portion (e.g., figure, graph, table, or textual material) of an IEEE copyrighted paper in a thesis:

1) In the case of textual material (e.g., using short quotes or referring to the work within these papers) users must give full credit to the original source (author, paper, publication) followed by the IEEE copyright line © 2011 IEEE. 
2) In the case of illustrations or tabular material, we require that the copyright line © [Year of original publication] IEEE appear prominently with each reprinted figure and/or table. 
3) If a substantial portion of the original paper is to be used, and if you are not the senior author, also obtain the senior author’s approval.

Requirements to be followed when using an entire IEEE copyrighted paper in a thesis:

1) The following IEEE copyright/credit notice should be placed prominently in the references: © [year of original publication] IEEE. Reprinted, with permission, from [author names, paper title, IEEE publication title, and month/year of publication] 
2) Only the accepted version of an IEEE copyrighted paper can be used when posting the paper or your thesis on-line. 
3) In placing the thesis on the author's university website, please display the following message in a prominent place on the website: In reference to IEEE copyrighted material which is used with permission in this thesis, the IEEE does not endorse any of [university/educational entity’s name goes here]’s products or services. Internal or personal use of this material is permitted. If interested in reprinting/republishing IEEE copyrighted material for advertising or promotional purposes or for creating new collective works for resale or redistribution, please go to http://www.ieee.org/publications_standards/publications/rights/rights_link.html to learn how to obtain a License from RightsLink.

If applicable, University Microfilms and/or ProQuest Library, or the Archives of Canada may supply single copies of the dissertation.
Appendix B: Institutional Review Board Authorization

The IRB approval below is for the use of material in 3.1.

ATM Golam Bari <bari@mail.usf.edu>

eIRB: Study Approved
1 message
rsch-arc@usf.edu <rsch-arc@usf.edu>
Reply-To: rsch-arc@usf.edu
To: bari@mail.usf.edu

Wed, Oct 31, 2018 at 1:41 PM

IRB Study Approved
To: Atm Golam Bari
RE: Student Experiences with a New Parsons Puzzles Software
PI: Alessio Gaspar
Link: Pro00037332

You are receiving this notification because the above listed study has received Approval by the IRB. To ensure compliance with IRB requirements, please review your approval letter by navigating to the project workspace by clicking the Link above.

DO NOT REPLY: To ensure a timely response, please direct correspondence to Research Integrity & Compliance either through your project's workspace or the contact information below.

Research Integrity & Compliance
University of South Florida - Research and Innovation
ARC Help Desk (eIRB, eIACUC, eCOI): (813) 974-2980
Email: rsch-arc@usf.edu
Mail: 12901 Bruce B. Downs Blvd, MDC 35, Tampa, FL 33612-4799

Template: _000 - IRB Study: Approved