Enhanced Optimization with Composite Objectives and Novelty Pulsation

Hormoz Shahrzad, Babak Hodjat, Camille Dollé, Andrei Denissov, Simon Lau, Donn Goodhew, Justin Dyer, Risto Miikkulainen

Abstract An important benefit of multi-objective search is that it maintains a diverse population of candidates, which helps in deceptive problems in particular. Not all diversity is useful, however: candidates that optimize only one objective while ignoring others are rarely helpful. A recent solution is to replace the original objectives by their linear combinations, thus focusing the search on the most useful tradeoffs between objectives. To compensate for the loss of diversity, this transformation is accompanied by a selection mechanism that favors novelty. This paper improves this approach further by introducing novelty pulsation, i.e. a systematic method to alternate between novelty selection and local optimization. In the highly deceptive problem of discovering minimal sorting networks, it finds state-of-the-art solutions significantly faster than before. In fact, our method so far has established a new world record for the 20-lines sorting network with 91 comparators. In the real-world problem of stock trading, it discovers solutions that generalize significantly better on unseen data. Composite Novelty Pulsation is therefore a promising approach to solving deceptive real-world problems through multi-objective optimization.

1 Introduction

Multi-objective optimization is most commonly used for discovering a Pareto front from which solutions that represent useful tradeoffs between objectives can be selected [5, 8, 9, 10, 15]. Evolutionary methods are a natural fit for such problems because the Pareto front naturally emerges in the population maintained in these methods. Interestingly, multi-objectivity can also improve evolutionary optimization because it encourages populations with more diversity. Even when the focus of

Hormoz Shahrzad
Cognizant Technology Solutions, e-mail: hormoz@cognizant.com

Risto Miikkulainen
The University of Texas at Austin e-mail: risto@cs.utexas.edu

optimization is to find good solutions along a primary performance metric, it is useful to create secondary dimensions that reward solutions that are different in terms of structure, size, cost, consistency, etc. Multi-objective optimization then discovers stepping stones that can be combined to achieve higher fitness along the primary dimension [26]. The stepping stones are useful in particular in problems where the fitness landscape is deceptive, i.e. where the optima are surrounded by inferior solutions [20].

However, not all such diversity is helpful. In particular, candidates that optimize one objective only and ignore the others are less likely to lead to useful tradeoffs, and they are less likely to escape deception. Prior research demonstrated that it is beneficial to replace the objectives with their linear combinations, thus focusing the search in more useful areas of the search space, and make up for the lost diversity by including a novelty metric in parent selection [31]. This paper improves upon this approach by introducing the concept of novelty pulsation: the novelty selection is turned on and off periodically, thereby allowing exploration and exploitation to leverage each other repeatedly.

This idea is tested in two domains. The first one is the highly deceptive domain of sorting networks [17] used in the original work on composite novelty selection [31]. Such networks consist of comparators that map any set of numbers represented in their input lines to a sorted order in their output lines. These networks have to be correct, i.e. sort all possible cases of input. The goal is to discover networks that are as small as possible, i.e. have as few comparators organized in as few sequential layers as possible. While correctness is the primary objective, it is actually not that difficult to achieve, because it is not deceptive. Minimality, on the other hand, is highly deceptive and makes the sorting network design an interesting benchmark problem. The experiments in this paper show that while the original composite novelty selection and its novelty-pulsation-enhanced version both find state-of-the-art networks up to 20 input lines, novelty pulsation finds them significantly faster. It also beat the state of the art for 20-lines network by finding a 91 comparators design, which broke the previous world record of 92 [32].

The second domain is the highly challenging real-world problem of stock trading. The goal is to evolve agents that decide whether to buy, hold, or sell particular stocks over time in order to maximize returns. Compared to original composite novelty method, novelty pulsation finds solutions that generalize significantly better to unseen data. It therefore forms a promising foundation for solving deceptive real-world problems through multi-objective optimization.

2 Background and Related Work

Evolutionary methods for optimizing single-objective and multi-objective problems are reviewed, as well as the idea of using novelty to encourage diversity and the concept of exploration versus exploitation in optimization methods. The domains of minimal sorting networks and automated stock trading are introduced and prior work in them reviewed.

2.1 Single-Objective Optimization

When the optimization problem has a smooth and non-deceptive search space, evolutionary optimization of a single objective is usually convenient and effective. However, we are increasingly faced with problems of more than one objective and with a rugged and deceptive search space. The first approach often is to combine the objectives into a single composite calculation [8]:

$$Composite(O_1,\ O_2,\ldots,O_k) = \sum_{i=1}^{k} \alpha_i O_i^{\beta_i} \tag{1}$$

Where the constant hyper-parameters α_i and β_i determine the relative importance of each objective in the composition. The composite objective can be parameterized in two ways:

1. By folding the objective space, and thereby causing a multitude of solutions to have the same value. Diversity is lost since solutions with different behavior are considered to be equal.
2. By creating a hierarchy in the objective space, and thereby causing some objectives to have more impact than many of the other objectives combined. The search will thus optimize the most important objectives first, which in deceptive domains might result in inefficient search or premature convergence to local optima.

Both of these problems can be avoided by casting the composition explicitly in terms of multi-objective optimization.

2.2 Multi-Objective Optimization

Multi-objective optimization methods construct a Pareto set of solutions [10], and therefore eliminate the issues with objective folding and hierarchy noted in Section 2.1. However, not all diversity in the Pareto space is useful. Candidates that optimize one objective only and ignore the others are less likely to lead to useful tradeoffs, and are less likely to escape deception.

One potential solution is reference-point based multi-objective methods such as NSGA-III [9, 10]. They make it possible to harvest the tradeoffs between many objectives and can therefore be used to select for useful diversity as well, although they are not as clearly suited for escaping deception.

Another problem with purely multi-objective search is crowding. In crowding, objectives that are easier to explore end up with disproportionately dense representation on the Pareto front. NSGA II addresses this problem by using the concept of crowding distance [8], and NSGA III improves upon it using reference points [9, 10]. These methods, while increasing diversity in the fitness space, do not necessarily result in diversity in the behavior space.

An alternative method is to use composite multi-objective axes to focus the search on the area with most useful tradeoffs [31]. Since the axes are not orthogonal, solutions that optimize only one objective will not be on the Pareto front. The focus effect, i.e. the angle between the objectives, can be tuned by varying the coefficients of the composite.

However, focusing the search in this manner has the inevitable side effect of reducing diversity. Therefore, it is important that the search method makes use of whatever diversity exists in the focused space. One way to achieve this goal is to incorporate a preference for novelty into selection.

2.3 Novelty Search

Novelty search [23, 25] is an increasingly popular paradigm that overcomes deception by ranking solutions based on how different they are from others. Novelty is computed in the space of behaviors, i.e., vectors containing semantic information about how a solution performs during evaluation. However, with a large space of possible behaviors, novelty search can become increasingly unfocused, spending most of its resources in regions that will never lead to promising solutions.

Recently, several approaches have been proposed to combine novelty with a more traditional fitness objective [11, 13, 29, 30] to reorient search towards fitness as it explores the behavior space. These approaches have helped scale novelty search to more complex environments, including an array of control [2, 7, 29] and content generation [19, 21, 22] domains.

Many of these approaches combine a fitness objective with a novelty objective in some way, for instance as a weighted sum [6], or as different objectives in a multi-objective search [29]. Another approach is to keep the two kinds of search separate, and make them interact through time. For instance, it is possible to first create a diverse pool of solutions using novelty search, presumably overcoming deception that way, and then find solutions through fitness-based search [18]. A third approach is to run fitness-based search with a large number of objective functions that span the space of solutions, and use novelty search to encourage search to utilize all those functions [7, 28, 30]. A fourth category of approaches is to run novelty search as the primary mechanism, and use fitness to select among the solutions. For instance, it is possible to add local competition through fitness to novelty search [22, 23]. Another version is to accept novel solutions only if they satisfy minimal performance criteria [11, 24]. Some of these approaches have been generalized using the idea of behavior domination to discover stepping stones [26, 27].

In the Composite Novelty method [31], a novelty measure is employed in a fifth way: to select which individuals to reproduce and which to discard. In this manner, it is integrated into the genetic algorithm itself, and its role is to make sure the focused space that the composite multiple objectives define is searched thoroughly.

2.4 Exploration versus Exploitation

Every search algorithm needs to both explore the search space and exploit the known good solutions in it. Exploration is the process of visiting entirely new regions of a search space, whilst exploitation is the process of visiting regions within the neighborhood of previously visited points. In order to be successful, a search algorithm needs to establish a productive synergy between exploration and exploitation [35].

A common problem in evolutionary search is that it gets stuck in local minima, i.e. in unproductive exploitation. A common solution is to kick-start the search process in such cases by temporarily increasing mutation rates. This solution can be utilized more systematically by making such kick-starts periodic, resulting in methods such as in delta coding and burst mutation [33, 34].

This paper incorporates the kick-start idea into novelty selection. By turning novelty selection on and off periodically allows local search (i.e. exploitation) and novelty search (i.e. exploration) to leverage each other, leading to faster search and better generalization. These effects will be demonstrated in the sorting networks and stock trading domains, respectively.

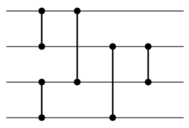

Fig. 1 A Four-Input Sorting Network. This network takes as its input (left) four numbers, and produces output (right) where those number are sorted (large to small, top to bottom). Each comparator (connection between the lines) swaps the numbers on its two lines if they are not in order, otherwise it does nothing. This network has three layers and five comparators, and is the minimal four-input sorting network. Minimal networks are generally not known for large input sizes. Their design space is deceptive which makes network minimization a challenging optimization problem.

2.5 Sorting Networks

A sorting network of n inputs is a fixed layout of comparison-exchange operations (comparators) that sorts all inputs of size n (Fig.1) [17]. Since the same layout can sort any input, it represents an oblivious or data-independent sorting algorithm, that is, the layout of comparisons does not depend on the input data. The resulting fixed communication pattern makes sorting networks desirable in parallel implementations of sorting, such as those in graphics processing units, multi-processor computers, and switching networks [1, 16, 31]. Beyond validity, the main goal in

designing sorting networks is to minimize the number of layers, because it determines how many steps are required in a parallel implementation. A tertiary goal is to minimize the total number of comparators in the networks. Designing such minimal sorting networks is a challenging optimization problem that has been the subject of active research since the 1950s [17]. Although the space of possible networks is infinite, it is relatively easy to test whether a particular network is correct: If it sorts all combinations of zeros and ones correctly, it will sort all inputs correctly [17].

Many of the recent advances in sorting network design are due to evolutionary methods [32]. However, it is still a challenging problem, even for the most powerful evolutionary methods, because it is highly deceptive: Improving upon a current design may require temporarily growing the network, or sorting fewer inputs correctly. Sorting networks are therefore a good domain for testing the power of evolutionary algorithms.

$3,500
$3,000
$2,500
$2,000
$1,500
$1,000
$500
$0
-$500
All
SELL
HOLD
BUY
PBR
1.50%
1.00%
0.50%
0.00%
-0.50%
-1.00%
-1.50%
-2.00%
10 AM
11 AM
12 PM
1 PM
2 PM
3 PM
4 PM

Fig. 2 Stock Trading Agent. The agent observes the time series of stock prices and makes live decisions about whether to buy, hold, or sell a particular stock. The signal is noisy and prone to overfitting; generalization to unseen data is the main challenge in this domain.

2.6 Stock Trading

Stock trading is a natural multi-objective domain where return and risk must be balanced [36, 37]. Candidate solutions, i.e. trading agents, can be represented in several ways. Rule-based strategies, sequence modeling with neural networks and LSTMs

(Long Short-Term Memory), and symbolic regression using Genetic Programming or Grammatical Evolution are common approaches [38, 39]. Frequency of trade, fundamental versus technical indicators, choice of trading instruments, transaction costs, and vocabulary of order types are crucial design decisions in building such agents.

The goal is to extract patterns from historical time-series data on stock prices and utilize those patterns to make optimal trading decisions, i.e. whether to buy, hold, or sell particular stocks (Fig.2) [40, 41]. The main challenge is to trade in a manner that generalizes to previously unseen situations in live trading. The data is extremely noisy and prone to overfitting, and methods that discover robust decisions are needed.

3 Methods

In this section, the genetic representation, the single and multi-objective optimization approaches, the composite objective method, the novelty-based selection method, and the novelty pulsation method are described, using the sorting network domain as an example. These methods were applied to stock trading in an analogous manner.

3.1 Representation

In order to apply various evolutionary optimization techniques to the sorting network problem, a general structured representation was developed. Sorting networks of n lines can be seen as a sequence of two-leg comparators where each leg is connected to a different input line and the first leg is connected to a higher line than the second:$\{(f_1,\, s_1),(f_2,\, s_2),(f_3,\, s_3),\ldots,(f_c,\, s_c)\}$.

The number of layers can be determined from such a sequence by grouping successive comparators together into a layer until the next comparator adds a second connection to one of the lines in the same layer. With this representation, mutation and crossover operators amount to adding and removing a comparator, swapping two comparators, and crossing over the comparator sequences of two parents at a single point.

Domain-specific techniques such as mathematically designing the prefix layers [3, 4] or utilizing certain symmetries [32] were not used.

3.2 Single-Objective Approach

Correctness is part of the definition of a sorting network: Even if a network mishandles only one sample, it will not be useful. The number of layers can be considered the most important size objective because it determines the efficiency of a parallel implementation. A hierarchical composite objective can therefore be defined as:

$$SingleFitness\,(m,\ l,\ c) = 10000\,m + 100\,l + c \tag{2}$$

Where m, l, and c are the number of mistakes (unsorted samples), number of layers, and number of comparators, respectively.

In the experiments in this paper, the solutions will be limited to less than one hundred layers and comparators, and therefore, the fitness will be completely hierarchical (i.e. there is no folding).

3.3 Multi-Objective Approach

In the multi-objective approach, the same dimensions, i.e. the number of mistakes, layers, and comparators $m,\ l,\ c$, are used as three separate objectives. They are optimized by the NSGA-II algorithm [8] with selection percentage set to 10%. Indeed, this approach may discover solutions with just a single layer, or a single comparator, since they qualify for the Pareto front. Therefore, diversity is increased compared to the single-objective method, but this diversity is not necessarily helpful.

3.4 Composite Multi-Objective Approach

In order to construct composite axes, each objective is augmented with sensitivity to the other objectives:

$$Composite_1\,(m,\ l,\ c) = 10000\,m + 100\,l + c \tag{3}$$

$$Composite_2\,(m,\ l) = \alpha_1 m + \alpha_2 l \tag{4}$$

$$Composite_3\,(m,\ c) = \alpha_3 m + \alpha_4 c \tag{5}$$

The primary composite objective (Formula 3), which will replace the mistake axis, is the same hierarchical fitness used in the single-objective approach. It discourages evolution from constructing correct networks that are extremely large. The second objective (Formula 4), with $\alpha_2 = 10$, primarily encourages evolution to look for solutions with a small number of layers. A much smaller cost of mistakes, with $\alpha_1 = 1$, helps prevent useless single-layer networks from appearing in the Pareto front. Similarly, the third objective (Formula 5), with $\alpha_3 = 1$ and $\alpha_4 = 10$, applies the same principle to the number of comparators.

The values for α_1, α_2, α_3, and α_4 were found to work well in this application, but the approach was found not to be very sensitive to them; A broad range will work as long as they establish a primacy relationship between the objectives.

It might seem like we are adding several hyper-parameters which need to be tuned, but we can estimate them in each domain by picking values that push away trivial or useless solution off the Pareto front.

3.5 Novelty Selection Method

In order to measure how novel the solutions are it is first necessary to characterize their behavior. While there are many ways to do this, a concise and computationally efficient approach is to count how many swaps took place on each line in sorting all possible zero-one combinations during the validity check. Such a characterization is a vector that has the same size as the problem, making the distance calculations quite fast. It also represents the true behavior of the network; that is, even if two networks sort the same input cases correctly, they may do it in different ways, and the characterization is likely to capture that difference. Given this behavior characterization, novelty of a solution is then measured by the sum of pairwise distances of its behavior vector to those of all the other individuals in the selection pool:

$$NoveltyScore(x_i) = \sum_{j=1}^{n} d(b(x_i)\,,\, b(x_j)) \qquad (6)$$

The selection method also has another parameter called *selection multiplier* (e.g. set to 2 in these experiments), varying between one and the inverse of the elite fraction (e.g. 1/10, i.e. 10%) used in the NSGA-II multi-objective optimization method. The original selection percentage is multiplied by the selection multiplier to form a broader selection pool. That pool is sorted according to novelty, and the top fraction representing the original selection percentage is used for selection. This way, good solutions that are more novel are included in the pool.

One potential issue is that a cluster of solutions far from the rest may end up having high novelty scores while only one is good enough to keep. Therefore, after the top fraction is selected, the rest of the sorted solutions are added to the selection pool one by one, replacing the solution with the lowest minimum novelty, defined as:

$$MinimumNovelty(x_i) = \underset{\leq j \leq n;\ j \neq i}{Min} \; d(b(x_i)\,,\, b(x_j)) \qquad (7)$$

Note that this method allows tuning novelty selection continuously between two extremes: by setting it to one, the method reduces to the original multi-objective method (i.e. only the elite fraction ends up in the final elitist pool), and setting it to the inverse of the elite fraction reduces it to pure novelty search (i.e. the whole population, sorted by novelty, is the selection pool). In practice, low and midrange values for the multiplier work well, including the value 2 used in these experiments.

3.6 Novelty Pulsation Method

Parent selection is a crucial step in an evolutionary algorithm. In almost all such algorithms, whatever method is used for selection remains unchanged during the course of an evolutionary run. However, when a problem is deceptive or prone to over-fitting, changing the selection method periodically may make the algorithm more robust. It can be used to alternate the search between exploration and exploitation, and thus find a proper balance between them.

In Composite Novelty Pulsation, novelty selection is switched on and off after a certain number of generations. As in delta-coding and burst mutation, once good solutions are found, they are used as a starting point for exploration. Once exploration has generated sufficient diversity, local optimization is performed to find the best possible versions of these diverse points. These two phases leverage each other, which results in faster convergence and more reliable solutions.

Composite Novelty Pulsation adds a new hyper-parameter, P, denoting the number of generations before switching to novelty selection. Preliminary experiments showed that $P = 5$ works well in both sorting network and stock trading domains; however, in principle it is possible to tune this parameter to fit the domain.

4 Experiment

Previous work in the sorting networks domain demonstrated that composite novelty can match the minimal known networks up to 18 input lines with reasonable computational resources [31, 32]. The goal of the sorting network experiments was to achieve the same result faster, i.e. with fewer resources. The experiments were therefore standardized to a single machine (a multi-core desktop).

In the stock market trading domain, the experiments compared generalization by measuring the correlation between seen and unseen data.

4.1 Experimental Setup

In the sorting networks domain, to compare composite novelty with novelty pulsation, experiments were run with the following parameters:

- Eleven network sizes, 8 through 18;

- Ten runs for each configuration (220 runs in total);
- 10% parent selection rate;
- Population size of 1000 for composite novelty selection and 100 for novelty pulsation. These settings were found to be appropriate for each method experimentally.

In the trading domain, to compare composite novelty with novelty pulsation, 10 experiments were run with the following parameters:

- 10 runs on five years of historical data;
- Population size of 500;
- 100 generations;
- 10% parent selection rate;
- Performance of the 10 best individuals from each run compared on the subsequent year of historical data, withheld from all runs.

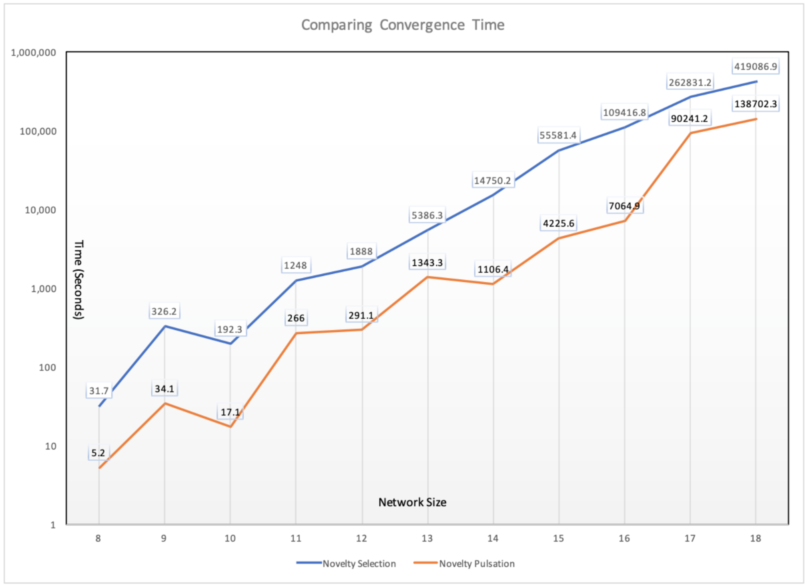

Fig. 3 The average runtime needed to converge to the state of the art over different size networks. Novelty pulsation converges significantly faster at all sizes, demonstrating improved balance of exploration and exploitation.

4.2 Sorting Networks Results

Convergence time of the two methods to minimal solutions for different network sizes is shown in Fig.3. Novelty pulsation shows an order of magnitude faster convergence across the board. All runs resulted in state-of-the-art sorting networks.

An interesting observation is that sorting networks with an even number of lines take proportionately less time to find the state-of-the-art solution than those with odd numbers of lines. This result is likely due to symmetrical characteristics of even-numbered problems. Some methods [32] exploit this very symmetry in order to find state-of-the-art solutions, but this domain-specific information was not used in our implementations. The fact that we have been able to achieve the state-of-the-art without exploiting domain specific characteristics of the problem is itself a significant result.

One of the nice properties of Novelty Pulsation Method is the ability to converge with a very small pool size (like only 30 individuals in case of sorting networks). However, it still took almost two months to break the world record on the 20-lines network running on a single machine (Fig.6). Interestingly, even if it takes the same number of generations for the other methods to get there with a normal pool size of a thousand, those runs will take almost five years to converge!

4.3 Stock Trading Results

Fig.4 and Fig.5 compare generalization of the composite novelty selection and novelty pulsation methods, respectively. Points in Fig.5 are noticeably closer to a diagonal line, which means that better training fitness resulted in better testing fitness, i.e. higher correlation and better generalization. Numerically, the seen-to-unseen correlation for the composite novelty method is 0.69, while for composite novelty pulsation, it is 0.86. In practice, this difference is significant, translating to much improved profitability in live trading.

5 Discussion and Future Work

The results in both sorting network and stock trading domains support the anticipated advantages of the composite novelty pulsation approach. The secondary objectives diversify the search, composite objectives focus it on most useful areas, and pulses of novelty selection allow for both accurate optimization and thorough exploration of those areas. These methods are general and robust: they can be read-

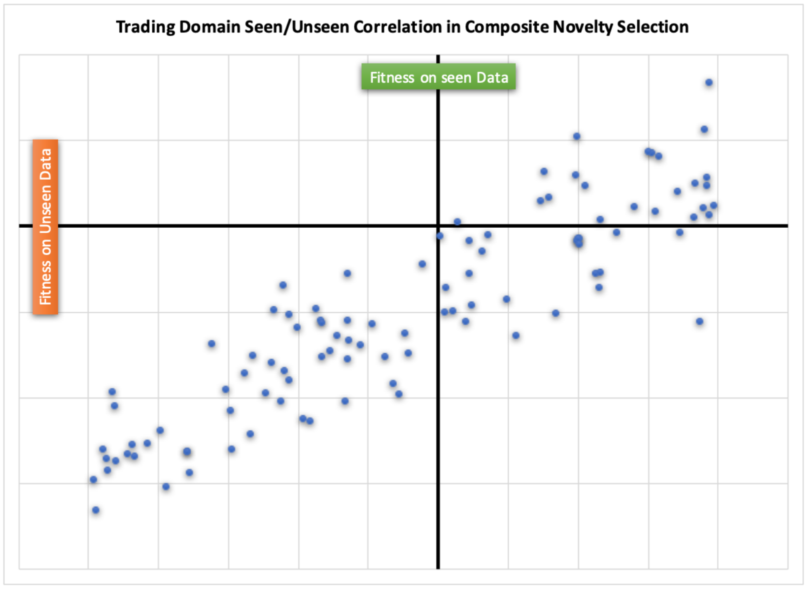

Fig. 4 Generalization from seen (x) to unseen (y) data with the Composite Novelty method. The correlation is 0.69, which is enough to trade but could be improved.

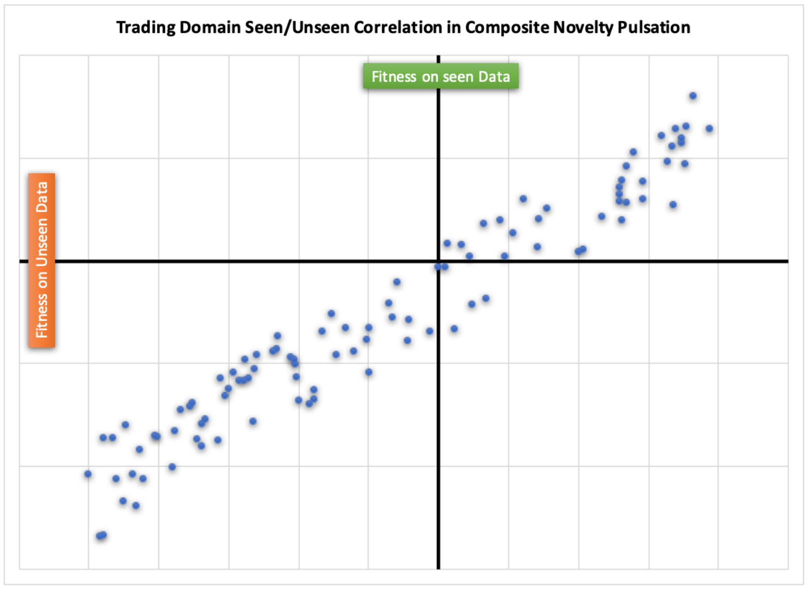

Fig. 5 Generalization from seen (x) to unseen (y) data with Composite Novelty Pulsation method. The correlation is 0.89, which results in significantly improved profitability in live trading.

ily implemented in standard multi-objective search such as NSGA-II and used in combination with many other techniques already developed to improve evolutionary multi-objective optimization.

The sorting network experiments were designed to demonstrate the improvement provided by novelty pulsation over the previous state of the art. Indeed, it found the best known solutions significantly faster. One compelling direction of future work is to use it to optimize sorting networks systematically, with domain-specific techniques integrated into the search, and with significantly more computing power, including distributed evolution [14]. It is likely that given such power, many new minimal networks can be discovered, for networks with even larger number of input lines.

The stock trading experiments was designed to demonstrate that the approach makes a difference in real-world problems. The main challenge in trading is generalization to unseen data, and indeed in this respect novelty pulsation improved generalization significantly.

The method can also be applied in many other domains, in particular those that are deceptive and have natural secondary objectives. For instance, various game strategies from board to video games can be cast in this form, where winning is accompanied by different dimensions of the score. Solutions for many design problems, such as 3D printed objects, need to satisfy a set of functional requirements, but also maximize strength and minimize material. Effective control of robotic systems need to accomplish a goal while minimize energy and wear and tear. Thus, many applications should be amenable to the composite novelty pulsation approach.

Another direction is to extend the method further into discovering effective collections of solutions. For instance, ensembling is a good approach for increasing the performance of machine learning systems. Usually the ensemble is formed from solutions with different initialization or training, with no mechanism to ensure that their differences are useful. In composite novelty pulsation, the Pareto front consists of a diverse set of solutions that span the area of useful tradeoffs. Such collections should make for a powerful ensemble, extending the applicability of the approach further.

6 Conclusion

The composite novelty pulsation method is a promising extension of the composite novelty approach to deceptive problems. Composite objectives focus the search on the most useful tradeoffs and allow escaping deceptive areas. Novelty pulsation leverages the exploration of diversity with local search, finding solutions faster and finding solutions that generalize better. These principles were demonstrated in this paper in the highly deceptive problem of minimizing sorting networks and in the highly noisy domain of stock market trading. Composite novelty pulsation is a general method that can be combined with other advances in population-based search, thus increasing the power and applicability of evolutionary multi-objective optimization.

Appendix

The graph of the new world record for 20-lines sorting network, which moved the previous record of 92 comparators also discovered by evolution [31] down to 91:

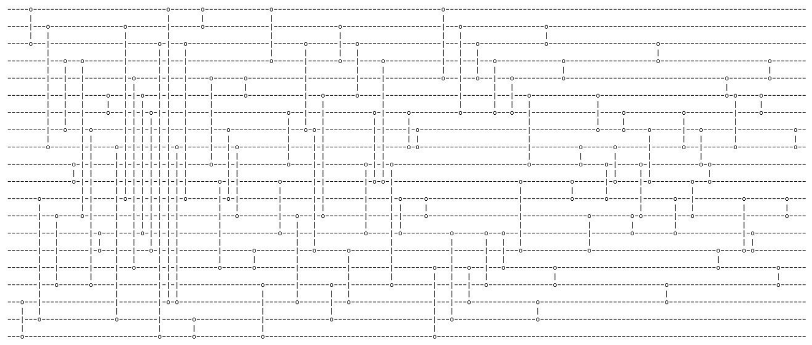

Fig. 6 The new 20-lines sorting network with 91 comparators, discovered by Novelty Pulsation.

References

1. S. W. A. Baddar. 2009. *Finding Better Sorting Networks*. PhD thesis, Kent State University.
2. J. A. Bowren, J. K. Pugh, and K. O. Stanley. 2016. Fully Autonomous Real-Time Autoencoder Augmented Hebbian Learning through the Collection of Novel Experiences. In Proceedings of ALIFE. 382?389.
3. M. Codish, L. Cruz-Filipe, and P. Schneider-Kamp. 2014. The quest for optimal sorting-networks: Efficient generation of two-layer prefixes. In *Symbolic and Numeric Algorithms for Scientific Computing (SYNASC), 2014 16th International Symposium on* (pp. 359-366). IEEE.
4. M. Codish, L. Cruz-Filipe, T. Ehlers, M. Muller, and P. Schneider-Kamp. 2016. Sorting networks: to the end and back again. *Journal of Computer and System Sciences*.
5. C. A. C. Coello, G. B. Lamont, and D. A. Van Veldhuizen. 2007. *Evolutionary algorithms for solving multi-objective problems*. Vol. 5. Springer.
6. G. Cuccu and F Gomez. 2011. When Novelty is Not Enough. In *Evostar*. 234–243.
7. A. Cully, J. Clune, D. Tarapore, and J-B. Mouret. 2015. Robots that can adapt like animals. Nature 521, 7553 (2015), 503–507.
8. K. Deb, A. Pratap, S. Agarwal, and T. A. Meyarivan. 2002. A fast and elitist multiobjective genetic algorithm: NSGA-II. *IEEE Trans. on Evolutionary Computation* 6, 2 (2002), 182–197.

9. K. Deb, and H. Jain. 2014. An Evolutionary Many-Objective Optimization Algorithm Using Reference-Point-Based Nondominated Sorting Approach, Part I: Solving Problems With Box Constraints. In *IEEE Transactions on Evolutionary Computation*, vol. 18, no. 4, 577-601.
10. K. Deb, K. Sindhya, and J. Hakanen. 2016. Multi-objective optimization. In *Decision Sciences: Theory and Practice*. 145–184.
11. J. Gomes, P. Mariano, and A. L. Christensen. 2015. Devising effective novelty search algorithms: A comprehensive empirical study. In Proc. of GECCO. 943-950.
12. J. Gomes, P. Urbano, and A. L. Christensen. 2013. Evolution of swarm robotics systems with novelty search. *Swarm Intelligence*, 7:115–144.
13. F. J. Gomez. 2009. Sustaining diversity using behavioral information distance. In Proc. of GECCO. 113–120.
14. B. Hodjat, H. Shahrzad, and R. Miikkulainen. 2016. Distributed Age-Layered Novelty Search. In Proc. of ALIFE. 131–138.
15. H. Jain, and K. Deb. 2014. An Evolutionary Many-Objective Optimization Algorithm Using Reference-Point Based Nondominated Sorting Approach, Part II: Handling Constraints and Extending to an Adaptive Approach. In *IEEE Transactions on Evolutionary Computation*, vol. 18, no. 4, 602-622.
16. P. Kipfer, M. Segal, and R. Westermann. 2004. Uberflow: A gpu-based particle engine. In HWWS 2004: Proc. of the ACM SIGGRAPH/EUROGRAPHICS, 115–122.
17. D. E. Knuth. 1998. *Art of Computer Programming: Sorting and Searching*, volume 3. Addison-Wesley Professional, 2 edition.
18. P. Krcah, and D. Toropila. 2010. Combination of novelty search and fitness-based search applied to robot body-brain coevolution. In Proc. of 13th Czech-Japan Seminar on Data Analysis and Decision Making in Service Science.
19. J. Lehman, S. Risi, and J. Clune. 2016. Creative Generation of 3D Objects with Deep Learning and Innovation Engines. In Proc. of ICCC. 180–187.
20. J. Lehman, and R. Miikkulainen. 2014. Overcoming deception in evolution of cognitive behaviors. In Proc. of GECCO.
21. J. Lehman and K. O. Stanley. 2012. Beyond open-endedness: Quantifying impressiveness. In Proc. of ALIFE. 75–82.
22. J. Lehman and K. O. Stanley. 2011. Evolving a diversity of virtual creatures through novelty search and local competition. In Proc. of GECCO. 211–218.
23. J. Lehman and K. O. Stanley. 2011. Abandoning objectives: Evolution through the search for novelty alone. Evolutionary Computation 19, 2 (2011), 189–223.
24. J. Lehman and K. O. Stanley. 2010. Efficiently evolving programs through the search for novelty. In Proc. of GECCO. 836–844.
25. J. Lehman and K. O. Stanley. 2008. Exploiting Open-Endedness to Solve Problems Through the Search for Novelty. In Proc. of ALIFE. 329–336.
26. E. Meyerson, and R. Miikkulainen. 2017. Discovering evolutionary stepping stones through behavior domination. In Proc. of GECCO, 139-146. ACM.
27. E. Meyerson, J. Lehman, and R. Miikkulainen. 2016. Learning behavior characterizations for novelty search. In Proc. of GECCO. 149–156.
28. J-B. Mouret and J. Clune. 2015. Illuminating search spaces by mapping elites. CoRR abs/1504.04909 (2015).
29. J-B. Mouret and S. Doncieux. 2012. Encouraging behavioral diversity in evolutionary robotics: An empirical study. Evolutionary Comp. 20, 1 (2012), 91–133.
30. J. K. Pugh, L. B. Soros, P. A. Szerlip, and K. O. Stanley. 2015. Confronting the Challenge of Quality Diversity. In Proc. of GECCO. 967–974.
31. H. Shahrzad, D. Fink, and R. Miikkulainen. 2018. Enhanced Optimization with Composite Objectives and Novelty Selection. In Proc. of ALIFE. 616-622.
32. V. K. Valsalam, and R. Miikkulainen. 2013. Using symmetry and evolutionary search to minimize sorting networks. Journal of Machine Learning Research, 14(Feb):303–331.
33. Gomez, F. and Miikkulainen, R., 1997. Incremental evolution of complex general behavior. Adaptive Behavior, 5(3-4), pp.317-342.

34. Whitley D, Mathias K, Fitzhorn P. Delta coding: An iterative search strategy for genetic algorithms. InICGA 1991 Dec 31 (Vol. 91, pp. 77-84).
35. ?repin?ek, Matej & Liu, Shih-hsi & Mernik, Marjan. (2013). Exploration and Exploitation in Evolutionary Algorithms: A Survey. ACM Computing Surveys. 45. Article 35.
36. Brabazon, A., O'Neill, M. 2006. Biologically Inspired Algorithms for Financial Modelling. Springer.
37. Bradley, R., Brabazon, A., O'Neill, M. 2010. Objective function design in a grammatical evolutionary trading system. In: 2010 IEEE World Congress on Computational Intelligence, pp. 3487-3494. IEEE Press.
38. Allen, F., Karjalainen, R. 1999. Using genetic algorithms to find technical trading rules. Journal of Financial Economics 51, 245-271.
39. Cui, W., Brabazon, A., O'Neill, M. 2011. Adaptive trade execution using a grammatical evolution approach. International Journal of Financial Markets and Derivatives 2(1/2), 4-3.
40. White H. 2000. A reality check for data snooping. Econometrica Sep. 2000 ;68(5):1097-126.
41. Contreras, I., Hidalgo, J.I., Nunez-Letamendia, L., Velasco, J.M. 2017. A meta-grammatical evolutionary process for portfolio selection and trading. Genetic Programming and Evolvable Machines 18(4), 411-431.