Solution and Fitness Evolution (SAFE): A Study of Multiobjective Problems

Moshe Sipper
Institute for Biomedical Informatics
University of Pennsylvania
Philadelphia, PA 19104-6021, USA
and *Department of Computer Science*
Ben-Gurion University
Beer Sheva 84105, Israel
sipper@upenn.edu

Jason H. Moore
Institute for Biomedical Informatics
University of Pennsylvania
Philadelphia, PA 19104-6021, USA
jhmoore@upenn.edu

Ryan J. Urbanowicz
Institute for Biomedical Informatics
University of Pennsylvania
Philadelphia, PA 19104-6021, USA
ryanurb@upenn.edu

***Abstract*—We have recently presented SAFE—Solution And Fitness Evolution—a commensalistic coevolutionary algorithm that maintains two coevolving populations: a population of candidate solutions and a population of candidate objective functions. We showed that SAFE was successful at evolving solutions within a robotic maze domain. Herein we present an investigation of SAFE's adaptation and application to multiobjective problems, wherein candidate objective functions explore different weightings of each objective. Though preliminary, the results suggest that SAFE, and the concept of coevolving solutions and objective functions, can identify a similar set of optimal multiobjective solutions without explicitly employing a Pareto front for fitness calculation and parent selection. These findings support our hypothesis that the SAFE algorithm concept can not only solve complex problems, but can adapt to the challenge of problems with multiple objectives.**

***Index Terms*—evolutionary computation, coevolution, novelty search, objective function, multiobjective optimization**

I. Introduction

We have recently highlighted a fundamental problem recognized to confound algorithmic optimization, namely, *conflating* the objective with the objective function [1]. Even when the former is well defined, the latter may not be obvious. We presented an approach to automate the means by which a good objective function may be discovered, through the introduction of SAFE—**S**olution **A**nd **F**itness **E**volution (**SAFE**)—a *commensalistic* coevolutionary algorithm that maintains two coevolving populations: a population of candidate solutions and a population of candidate objective functions [2]. We showed that SAFE successfully evolved solutions within a robotic maze domain, simultaneously with the objective functions needed to measure solution quality during evolution.

In this paper we perform a preliminary investigation into SAFE's efficacy in solving multiobjective problems, targeting the well-known ZDT benchmark suite. With multiobjective problems the evolutionary algorithm is tasked with optimizing more than one objective simultaneously. The objective function—which, as noted above, should not be conflated with the objective(s)—thus needs to perform a "juggling act" that balances the (multiple) objectives in an efficacious manner, producing viable solutions. SAFE, with its inherent ability to devise its own objective functions on the fly (as we shall see below), is thus a good option to try out in the domain of multiobjective optimization.

Ultimately, our goal is to employ this SAFE algorithm concept as a machine learning (ML) algorithm for the modeling and discovery of complex patterns of associations, which manifest themselves as problems with a deceptive fitness landscape. ML has been defined as computer programs (i.e., algorithms) where the performance (P) on some task (T) is improved with experience (E) [3]. However, this makes the assumption that the P metric is both fixed and it has been correctly selected a priori to represent T optimality. This assumption is often violated in real-world problems leading to bias and overfitting.

In the next section we provide a précis of multiobjective optimization. Section III discusses coevolutionary algorithms. Section IV presents the SAFE algorithm and describes how it can be applied to the current domain of interest. Note that, as opposed to many other multiobjective optimizers, SAFE does not rely on measures of the Pareto front (i.e., a Pareto front is not employed to calculate solution fitness, or as a standard for selecting parent solutions to generate offspring solutions). Experiments and results are provided in Section V, followed by concluding remarks in Section VI.

II. Multiobjective Optimization

A multiobjective optimization problem involves two or more objectives all of which need to be optimized. For example, think of buying a car, where the objectives are to optimize both cost and fuel efficiency. Applications of multiobjective optimization abound in numerous domains [4].

Consider the 2-objective ZDT problems, defined in Table I [5], [6]. The two objectives are to minimize both $f_1(\mathbf{x})$ and $f_2(\mathbf{x})$. This classic benchmark suite epitomizes the basic setup of multiobjective optimization. The utility of this suite is

©Copyright 2019 IEEE. Proceedings of 2019 IEEE Congress on Evolutionary Computation (CEC). This work was supported by National Institutes of Health (USA) grants AI116794, LM010098, and LM012601.

that the ground-truth optimal Pareto front can be computed and used to determine and compare multiobjective algorithm performance.

With a multiobjective optimization problem there is usually no single-best solution, but rather the goal is to identify a set of 'non-dominated' solutions that represent optimal tradeoffs between multiple objectives. This set of solutions is also known as the *Pareto optimal set*. Usually, a representative subset will suffice.

Reviewing the state of the art, Zhou et al. [4] noted several types of multiobjective algorithms: 1) those based on decomposition, where the problem is decomposed into a number of scalar objective optimization problems; 2) preference-based algorithms, where the user provides information about preferred solutions; 3) indicator-based algorithms, where the quality of an approximated Pareto front is measured by a scalar indicator such as generational distance or hypervolume, which is used to guide the search; 4) hybrid algorithms, which hybridize different search methods; 5) memetic multiobjective algorithms, which incorporate local search; and 6) coevolutionary methods, which is the category our method fits in.

III. Coevolution

Coevolution refers to the simultaneous evolution of two or more species with coupled fitness [7]. Strongly related to the concept of symbiosis, coevolution can be mutualistic, parasitic, or commensalistic [8]: 1) In mutualism, different species exist in a relationship in which each individual (fitness) benefits from the activity of the other; 2) in parasitism, an organism of one species competes with an organism of a different species; and 3) in commensalism, members of one species gain benefits while those of the other species neither benefit nor are harmed. The idea of coevolution originates (at least) with Darwin—who spoke of "coadaptations of organic beings to each other" [9] (Figure 1).

A cooperative (mutualistic) coevolutionary algorithm involves a number of independently evolving species, which come together to obtain problem solutions. The fitness of an individual depends on its ability to collaborate with individuals from other species [7], [10]–[12].

In a competitive (parasitic) coevolutionary algorithm the fitness of an individual is based on direct competition with individuals of other species, which in turn evolve separately in their own populations. Increased fitness of one of the species implies a reduction in the fitness of the other species [13].

To the best of our knowledge, SAFE is the first coevolutionary algorithm to employ a form of commensalism [2].

IV. SAFE

SAFE is a coevolutionary algorithm that maintains two coevolving populations: a population of candidate solutions and a population of candidate objective functions (Figure 2). The evolution of each population is identical to a standard, single-population evolutionary algorithm—except where fitness computation is concerned. Below we describe the various components of the system, which includes: (1) population composition, (2) initialization, (3) selection, (4) elitism, (5) crossover, (6) mutation, and (7) fitness computation.

Populations. SAFE maintains two coevolving populations. An individual in the *solutions population* is a list of 30 real values, each in the range given in Table I and defined by the respective ZDT problems. .

An individual in the *objective-functions population* is a list of 2 real values $[a, b]$, each in the range $[0, 1]$, whose usage is described under *fitness* below. Population sizes and other parameters are given in Table II (Sipper et al. [14] provide interesting insights into parameters in evolutionary computation).

Initialization. For every evolutionary run: both populations are initialized to random (fixed-length) lists, wherein each component value is in the appropriate range.

Selection. Tournament selection with tournament size 5, i.e., choose 5 individuals at random from the population and return the individual with the best fitness as the selected one.

Elitism. The 2 individuals with the highest fitness in a generation are copied ("cloned") into the next generation unchanged.

Crossover. Standard single-point crossover, i.e., select a random crossover point and swap two parent genomes beyond this point to create two offspring. The crossover rate is the probability with which crossover between two selected parents occurs. (Note that for an objective-function individual, which comprises two values, if crossover occurs, it will always be, ipso facto, at the same position.)

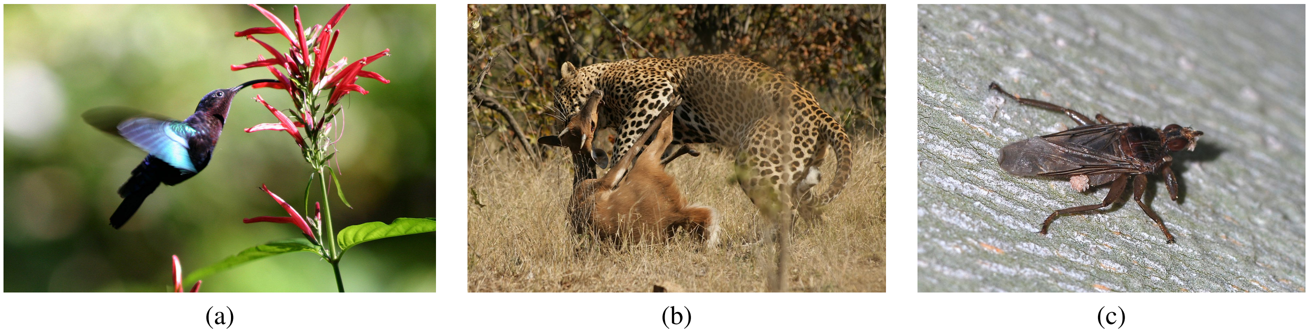
Mutation. Mutation is done with probability 0.4 (per individual in the population) by selecting a random gene (of the 30 or 2, respectively) and replacing it with a new random value in the appropriate range.

Fitness. Fitness computation is where SAFE dynamics come into play. In SAFE, each solution individual, S_i, $i \in \{1, \ldots, n\}$ is scored by every candidate objective-function individual, O_j, in the current population, $j \in \{1, \ldots, m\}$ (Figure 2A). This is in contrast with a standard evolutionary algorithm, where a population of solutions is evolved but the objective function is fixed (Figure 2B).

A candidate SAFE objective-function individual describes a candidate set of weights, balancing the two objectives of the ZDT functions. The best (highest) of all calculated objective function scores (out of the set of all individual objective functions in the objective-function population) is then assigned to the individual solution as its fitness value.

As noted above, an objective-function individual is a pair $[a, b]$; specifically, a determines f_1's weighting and b determines f_2's weighting. $O_j(S_i)$ is the fitness score that objective function O_j assigns to solution S_i. Algorithm 1 details fitness computation of solutions.

As for the objective-functions population, determining the quality of an evolving objective function places us in uncharted waters. Such an individual is not a solution to a problem, but rather the "guide"—or "path"—to a solution. As such, it is not clear what comprises a good measure of success. As in [2], we turned to a commensalistic coevolution strategy, where the



(a) (b) (c)

Fig. 1. Coevolution: (a) cooperative: Purple-throated carib feeding from and pollinating a flower (credit: Charles J Sharp, https://commons.wikimedia.org/wiki/File:Purple-throated_carib_hummingbird_feeding.jpg); (b) competitive: predator and prey—a leopard killing a bushbuck (credit: NJR ZA, https://commons.wikimedia.org/wiki/File:Leopard_kill_-_KNP_-_001.jpg); (c) commensalistic: Phoretic mites attach themselves to a fly for transport (credit: Alvesgaspar, https://en.wikipedia.org/wiki/File:Fly_June_2008-2.jpg).

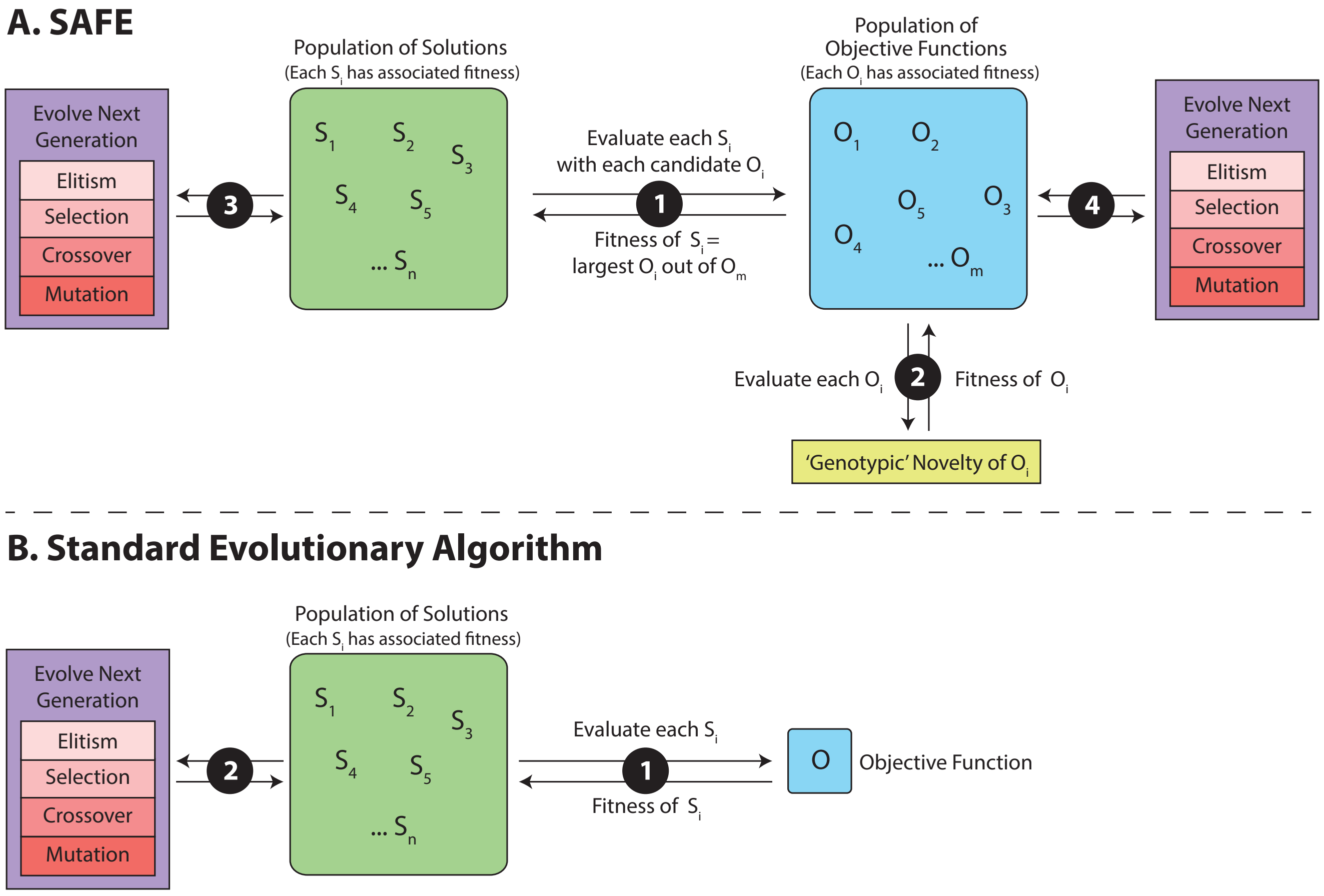

A. SAFE
Evolve Next Generation
Elitism
Selection
Crossover
Mutation
3
Population of Solutions
(Each S_i has associated fitness)
S_1 S_2 S_3 S_4 S_5 ... S_n
Evaluate each S_i with each candidate O_i
1
Fitness of S_i = largest O_i out of O_m
Population of Objective Functions
(Each O_i has associated fitness)
O_1 O_2 O_5 O_3 O_4 ... O_m
4
Evolve Next Generation
Elitism
Selection
Crossover
Mutation
Evaluate each O_i
2
Fitness of O_i
'Genotypic' Novelty of O_i
B. Standard Evolutionary Algorithm
Evolve Next Generation
Elitism
Selection
Crossover
Mutation
2
Population of Solutions
(Each S_i has associated fitness)
S_1 S_2 S_3 S_4 S_5 ... S_n
Evaluate each S_i
1
Fitness of S_i
O
Objective Function

Fig. 2. A single generation of SAFE vs. a single generation of a standard evolutionary algorithm. The numbered circles identify sequential steps in the respective algorithms. The objective function can comprise a single or multiple objectives. Recall, also, that we advocate not conflating objective and objective function.

TABLE I
TWO-OBJECTIVE BENCHMARK ZDT PROBLEMS. THE TWO OBJECTIVES ARE TO MINIMIZE BOTH $f_1(\mathbf{x})$ AND $f_2(\mathbf{x})$. THE DIMENSIONALITY OF THE PROBLEM IS $k = 30$, I.E., SOLUTION VECTOR $\mathbf{x} = x_1, \ldots, x_{30}$

Name	Problem	Parameter domain
ZDT1	$f_1(\mathbf{x}) = x_1$ $g(\mathbf{x}) = 1 + 9/(k-1)\sum_{i=2}^{k} x_i$ $f_2(\mathbf{x}) = 1 - \sqrt{f_1/g}$	$[0, 1]$
ZDT2	as ZDT1, except $f_2(\mathbf{x}) = 1 - (f_1/g)^2$	$[0, 1]$
ZDT3	as ZDT1, except $f_2(\mathbf{x}) = 1 - \sqrt{f_1/g} - (f_1/g)\sin(10\pi f_1)$	$[0, 1]$
ZDT4	as ZDT1, except $g(\mathbf{x}) = 1 + 10(k-1) + \sum_{i=2}^{k}(x_i^2 - 10\cos(4\pi x_i))$	$x_1 \in [0, 1]$, $x_2, \ldots, x_n \in [-5, 5]$

TABLE II
EVOLUTIONARY PARAMETERS

Description	Value
Number of evolutionary runs	50
Maximal number of generations	3000
Size of solutions population	500
Size of objective-functions population	150
Type of selection	Tournament
Tournament size	5
Type of crossover	single-point
Crossover rate	0.8
Probability of mutation (solutions)	0.4
Probability of mutation (objective functions)	0.4
Number of top individuals copied (elitism)	2

Algorithm 1 Compute fitness values of solutions population

```
$n \leftarrow$ size of solutions population
$m \leftarrow$ size of objective-functions population
for $i \leftarrow 1$ to $n$ do
    compute $f_1(S_i)$, $f_2(S_i)$
for $i \leftarrow 1$ to $n$ do
    for $j \leftarrow 1$ to $m$ do
        $a'_j \leftarrow a_j/(a_j + b_j)$
        $b'_j \leftarrow b_j/(a_j + b_j)$
        $O_j(S_i) \leftarrow 1/(a'_j \times f_1(S_i) + b'_j \times f_2(S_i))$
    $solutionFitness_i \leftarrow \max_j O_j$
```

objective functions' fitness does not depend on the population of solutions. Instead, it relies on *genotypic novelty*.

Novelty search was introduced by [15], the key idea being that instead of rewarding closeness to objective, individual solutions would be considered valuable if their behavior diverged from prior solutions. Effectively, novelty search is identical to a standard evolutionary algorithm, with the fitness function replaced by the novelty metric. We seek novel behaviors rather than the objective, hoping that the former will ultimately lead to the latter.

In our case, the novelty metric is based on the objective-function individual's two-valued genome, $[a, b]$. The distance between two objective functions—$[a_1, b_1]$, $[a_2, b_2]$—is simply the Euclidean distance of their genomes, given as: $\sqrt{(a_1 - a_2)^2 + (b_1 - b_2)^2}$. Note that this is a genotypic novelty metric, defined by referring to the genome, as opposed to phenotypic, behavioral novelty, used elsewhere [2].

Each generation, every candidate objective function is compared to its cohorts in the current population of objective functions and to an archive (of size 1000) of past individuals whose behaviors were highly novel when they emerged. The novelty score is the average of the distances to the k (= 15) nearest neighbors, and is used in computing objective-function fitness (Algorithm 2).

Algorithm 2 Compute fitness values of objective-functions population

```
$m \leftarrow$ size of objective-functions population
for $i \leftarrow 1$ to $m$ do
    compute $noveltyScoreObj_i$
    $objectiveFitness_i \leftarrow noveltyScoreObj_i$
```

Every generation each new solution is examined and added to a Pareto front if appropriate, this front being the algorithm's final desired output. Note, however, that while the front is continually updated for tracking and analysis purposes, it is *not* used in any way by SAFE in fitness calculation.

V. EXPERIMENTS AND RESULTS

We performed 50 replicate runs of SAFE for each of the four ZDT problems, ZDT1, ZDT2, ZDT3, ZDT4, recording the evolving Pareto front as evolution progressed. In particular, we were interested in the *inverted generational distance (igd)* of the Pareto fronts evolved.

The igd is one of the leading measures used in recent years to asses the success of a multiobjective optimizer. It measures the distance between the true Pareto front and the front produced by an algorithm [16]. The metric is defined as:

$$igd = \frac{\sum_{v \in TF} d(v, PF)}{|TF|}$$

where TF is the true Pareto front, PF is the front found by the algorithm, and $d(v, PF)$ is the minimum Euclidean distance (measured in objective space) between v and the solutions in

TABLE III
RESULTS OF 50 EVOLUTIONARY REPLICATE RUNS PER EACH PROBLEM. SHOWN IS THE AVERAGE INVERTED GENERATIONAL DISTANCE (igd) AND STANDARD DEVIATION

Problem	Inverted generational distance
ZDT1	2.06E-04 (2.12E-05)
ZDT2	2.65E-04 (3.82E-05)
ZDT3	3.81E-02 (2.48E-02)
ZDT4	1.23E-03 (4.53E-04)

TABLE IV
COMPARING SAFE'S AVERAGE igd RESULTS WITH THE BEST VALUES REPORTED IN [17] AND [22]

Problem	SAFE	Cheng et al. [17]	Han et al. [22]
ZDT1	**2.06E-04**	3.88E-03	2.81E-03
ZDT2	**2.65E-04**	3.85E-03	3.92E-03
ZDT3	3.81E-02	4.82E-03	**4.45E-03**
ZDT4	**1.23E-03**	3.99E-03	3.77E-03

PF; $|TF|$ is the cardinality of set TF. The igd measures both the convergence and spread of the obtained Pareto solutions along the true Pareto front [17].

Table III shows the average inverted generational distance (igd) of the four experiments carried out.

We compared our results with two very recent studies. Cheng et al. [17] presented HTL-MOPSO, a novel hybrid teaching learning based particle swarm optimization algorithm (HTL-PSO) with circular crowded sorting (CCS). They compared their algorithm to: NSGA-II [18], TV-MOPSO [19], MOEA/D [20], and MOTLBO [21].

Han et al. [22] presented an improved multiobjective quantum-behaved particle swarm optimization algorithm based on double search strategy and circular transposon mechanism (MOQPSO-DSCT). They compared their algorithm to: MPSO/D [23], MOQPSO-AG [24], dMOPSO [25], SMPSO [26], NSGA-II [18], and MOEA/D [20].

Table IV compares our average igd results with the best values reported in these two recent works, showing the merits of SAFE. Figure 3 shows four Pareto fronts produced by SAFE, compared with the optimal ground truth Pareto fronts as defined by the ZDT functions.

We find that by allowing objective function weights to evolve in the SAFE framework, driven by 'genotypic novelty', we are able to identify a Pareto optimal solution set that is quite close to the ground-truth optimal solutions across the full span of the front. This behavior makes sense since SAFE evolutionary pressures for objective functions are driven by genotypic novelty, which translates in the ZDT problems to different objective weightings. This diverse set of objective weightings in turn is likely to encourage the evolution of optimal solutions at different locations along the Pareto optimal front.

VI. CONCLUDING REMARKS

Aiming to confront the optimization conflation problem—where the objective is conflated with the objective function—we separated these two entities into two populations, and presented SAFE, a coevolutionary algorithm to evolve the two simultaneously. Our previous work showed promising results in a maze domain [2] and herein we presented preliminary results that show promise in a multiobjective-optimization domain while simultaneously demonstrating that the SAFE concept of coevolving solutions with objective functions is able to identify a range of Pareto optimal, non-dominated solutions without adopting a Pareto front for fitness evaluation or parent selection. This finding interestingly suggests that SAFE can not only adapt to deceptive fitness landscapes but to problems that demand multiple objectives be optimized. It also presents an alternative strategy to generating non-dominated multiobjective solutions outside of the Pareto front.

We realize that this work is preliminary and, hoping it leads to new frontiers, have several avenues we wish to explore.

- The coevolutionary dynamics engendered by SAFE in solving multiobjective problems are likely to be worthy of study in and of themselves.
- We will adapt the SAFE concept to new problem domains, particularly those that target the machine learning modeling of complex pattens of association in biomedical data mining. In general, we plan to identify those problems wherein the SAFE concept is not only competitive but clearly advantageous. It is of particular interest to demonstrate whether SAFE can adapt itself to problems where little prior knowledge exists regarding either the objective or the best path to said objective.
- We did not discuss computational cost herein, given the preliminary investigative nature of this work, and our interest in simply showing that SAFE is a viable algorithm that produces satisfactory results. Of course, computational cost needs to be addressed in the future. Note that comparing multiobjective algorithms is often far from trivial, since many exhibit various "hidden" costs, e.g., hypervolume-based algorithms compute a costly hypervolume measure [27] (e.g., Bezerra et al. [28] recently noted that, "The drawback of indicator-based approaches is the computational complexity of some quality indicators such as the hypervolume, which is exponential in the number of objectives in the worst case").
- We used a simple measure to drive objective-function evolution in this problem domain—genotypic novelty. Other, possibly better measures might be designed.
- Examine the incorporation of alternative, more sophisticated evolutionary algorithm components into the SAFE concept (e.g., selection, elitism, genetic operators, and solution representation).

REFERENCES

[1] M. Sipper, R. J. Urbanowicz, and J. H. Moore, "To know the objective is not (necessarily) to know the objective function," *BioData Mining*, vol. 11, no. 1, Oct 2018.

[2] ——, "Solution and fitness evolution (SAFE): Coevolving solutions and their objective functions," in *European Conference on Genetic Programming (EuroGP 2019)*. Cham: Springer International Publishing, 2019, (to appear).

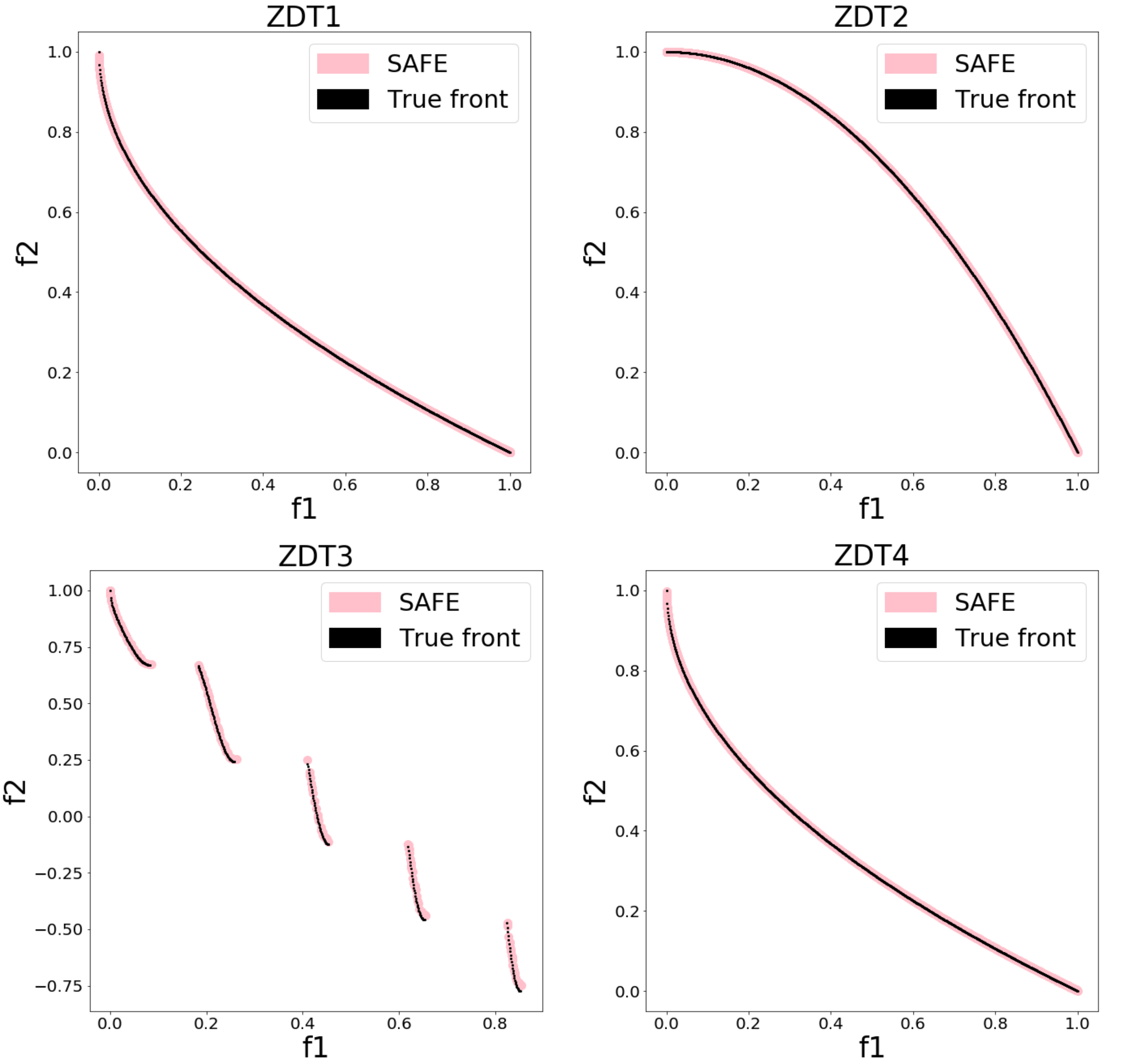


Fig. 3. Four sample solutions produced by SAFE, compared with the true fronts. igd values are: ZDT1: 1.67E-04, ZDT2: 1.60E-04, ZDT3: 5.24E-03, ZDT4: 6.21E-04.

[3] T. M. Mitchell, *Machine Learning*, 1st ed. New York, NY, USA: McGraw-Hill, Inc., 1997.

[4] A. Zhou, B.-Y. Qu, H. Li, S.-Z. Zhao, P. N. Suganthan, and Q. Zhang, “Multiobjective evolutionary algorithms: A survey of the state of the art,” *Swarm and Evolutionary Computation*, vol. 1, no. 1, pp. 32–49, 2011.

[5] E. Zitzler, K. Deb, and L. Thiele, “Comparison of multiobjective evolutionary algorithms: Empirical results,” *Evolutionary computation*, vol. 8, no. 2, pp. 173–195, 2000.

[6] S. Huband, P. Hingston, L. Barone, and L. While, “A review of multiobjective test problems and a scalable test problem toolkit,” *IEEE Transactions on Evolutionary Computation*, vol. 10, no. 5, pp. 477–506, 2006.

[7] C. A. Pena-Reyes and M. Sipper, “Fuzzy CoCo: A cooperative-coevolutionary approach to fuzzy modeling,” *IEEE Transactions on Fuzzy Systems*, vol. 9, no. 5, pp. 727–737, 2001.

[8] Wikipedia, “Symbiosis,” 2018. [Online]. Available: https://en.wikipedia.org/wiki/Symbiosis

[9] C. R. Darwin, *On the Origin of Species by Means of Natural Selection, or the Preservation of Favoured Races in the Struggle for Life*. London: John Murray, 1859.

[10] A. Zaritsky and M. Sipper, “Coevolving solutions to the shortest common superstring problem,” *Biosystems*, vol. 76, no. 1, pp. 209–216, 2004.

[11] M. A. Potter and K. A. De Jong, “Cooperative coevolution: An architecture for evolving coadapted subcomponents,” *Evolutionary Computation*, vol. 8, no. 1, pp. 1–29, 2000.

[12] G. Dick and X. Yao, “Model representation and cooperative coevolution for finite-state machine evolution,” in *2014 IEEE Congress on Evolutionary Computation (CEC)*. Piscataway, NJ: IEEE, 2014, pp. 2700–2707.

[13] W. Hillis, “Co-evolving parasites improve simulated evolution as an optimization procedure,” *Physica D: Nonlinear Phenomena*, vol. 42, no. 1, pp. 228–234, 1990.

[14] M. Sipper, W. Fu, K. Ahuja, and J. H. Moore, “Investigating the

parameter space of evolutionary algorithms," *BioData Mining*, vol. 11, no. 2, pp. 1–14, 2018.

[15] J. Lehman and K. O. Stanley, "Exploiting open-endedness to solve problems through the search for novelty," in *Proceedings of the Eleventh International Conference on Artificial Life (ALIFE)*. Cambridge, MA: MIT Press, 2008.

[16] C. A. C. Coello and M. R. Sierra, "A study of the parallelization of a coevolutionary multi-objective evolutionary algorithm," in *Mexican International Conference on Artificial Intelligence*. Springer, 2004, pp. 688–697.

[17] T. Cheng, M. Chen, P. J. Fleming, Z. Yang, and S. Gan, "A novel hybrid teaching learning based multi-objective particle swarm optimization," *Neurocomputing*, vol. 222, pp. 11–25, 2017.

[18] K. Deb, A. Pratap, S. Agarwal, and T. Meyarivan, "A fast and elitist multiobjective genetic algorithm: NSGA-II," *IEEE Transactions on Evolutionary Computation*, vol. 6, no. 2, pp. 182–197, 2002.

[19] P. K. Tripathi, S. Bandyopadhyay, and S. K. Pal, "Multi-objective particle swarm optimization with time variant inertia and acceleration coefficients," *Information Sciences*, vol. 177, no. 22, pp. 5033–5049, 2007.

[20] Q. Zhang and H. Li, "MOEA/D: A multiobjective evolutionary algorithm based on decomposition," *IEEE Transactions on Evolutionary Computation*, vol. 11, no. 6, pp. 712–731, 2007.

[21] F. Zou, L. Wang, X. Hei, D. Chen, and B. Wang, "Multi-objective optimization using teaching-learning-based optimization algorithm," *Engineering Applications of Artificial Intelligence*, vol. 26, no. 4, pp. 1291–1300, 2013.

[22] F. Han, Y.-W.-T. Sun, and Q.-H. Ling, "An improved multiobjective quantum-behaved particle swarm optimization based on double search strategy and circular transposon mechanism," *Complexity*, vol. 2018, 2018.

[23] C. Dai, Y. Wang, and M. Ye, "A new multi-objective particle swarm optimization algorithm based on decomposition," *Information Sciences*, vol. 325, pp. 541–557, 2015.

[24] Z. Shi and Q. Chen, "Multi-objective optimization algorithm based on quantum-behaved particle swarm and adaptive grid," *Inf Control*, vol. 40, no. 2, pp. 214–220, 2011.

[25] S. Zapotecas Martínez and C. A. Coello Coello, "A multi-objective particle swarm optimizer based on decomposition," in *Proceedings of the 13th Annual Conference on Genetic and Evolutionary Computation*. ACM, 2011, pp. 69–76.

[26] A. J. Nebro, J. J. Durillo, J. Garcia-Nieto, C. C. Coello, F. Luna, and E. Alba, "SMPSO: A new pso-based metaheuristic for multi-objective optimization," in *Computational Intelligence in Multi-Criteria Decision-Making, 2009. MCDM'09. IEEE Symposium on*. IEEE, 2009, pp. 66–73.

[27] L. M. Antonio and C. A. C. Coello, "Coevolutionary multiobjective evolutionary algorithms: Survey of the state-of-the-art," *IEEE Transactions on Evolutionary Computation*, vol. 22, no. 6, pp. 851–865, 2018.

[28] L. C. T. Bezerra, M. Lpez-Ibez, and T. Sttzle, "A large-scale experimental evaluation of high-performing multi- and many-objective evolutionary algorithms," *Evolutionary Computation*, vol. 26, no. 4, pp. 621–656, 2018.