Features for Exploiting Black-Box Optimization Problem Structure

Tinus Abell[1], Yuri Malitsky[2*], and Kevin Tierney[1**]

[1] IT University of Copenhagen, Copenhagen, Denmark
{tmab,kevt}@itu.dk
[2] Cork Constraint Computation Centre, Cork, Ireland
y.malitsky@4c.ucc.ie

Abstract. Black-box optimization (BBO) problems arise in numerous scientific and engineering applications and are characterized by computationally intensive objective functions, which severely limit the number of evaluations that can be performed. We present a robust set of features that analyze the fitness landscape of BBO problems and show how an algorithm portfolio approach can exploit these general, problem independent features and outperform the utilization of any single minimization search strategy. We test our methodology on data from the GECCO Workshop on BBO Benchmarking 2012, which contains 21 state-of-the-art solvers run on 24 well-established functions.

1 Introduction

This paper tackles the challenge of crafting a set of features that can capture the structure of black-box optimization (BBO) problem fitness landscapes for use in portfolio algorithms. BBO problems involve the minimization of an objective function $f(x_1, \ldots, x_n)$, subject to the constraints $l_i \leq x_i \leq u_i$, over the variables $x_i \in \mathbb{R}, \forall 1 \leq i \leq n$. These types of problems are found throughout the scientific and engineering fields, but are difficult to solve due to their oftentimes expensive objective functions. This complexity can arise when the objective involves difficult to compute expressions or that are too complicated to be defined by a simple mathematical expression. Even though BBO algorithms do not guarantee the discovery of the optimal solution, they are an effective tool for finding approximate solutions. However, different BBO algorithms vary greatly in performance across a set of problems. Thus, deciding which solver to apply to a particular problem is a difficult task.

Portfolio algorithms, like Instance Specific Algorithm Configuration (ISAC) [11], which uses a clustering approach to identify groups of similar instances, provide a way to automatically choose a solver for a particular instance using offline learning. However, a set of features that consolidate the relevant attributes of a BBO instance into a vector is required to use such methods on BBO problems. The only way to generate these features for BBO problems is by evaluating

* Yuri Malitsky is partially supported by the EU FET grant ICON (project 284715).
** Kevin Tierney is supported by the Danish Council for Strategic Research as part of the ENERPLAN project.

expensive queries to the black box, which contrasts with most non-black-box problems, e.g. SAT or the set covering problem, where many features can be quickly inferred from the problem definition itself.

In this paper, we propose a novel set of features that are fast to compute and are descriptive enough of the instance structure to allow a portfolio algorithm like ISAC to accurately cluster and tune for the benchmark. These features are based on well-known fitness landscape measures and are learned through sampling the black box. These features allow anybody to take advantage of the recent advances in the ISAC framework in order to more efficiently solve their BBO problems. This paper is a short version of [1].

Related work There has been extensive research studying the structure of BBO problems, and copious measures have been proposed for determining the hardness of local search problems by sampling their fitness landscape [9], such as the search space diameter, optimal solution density/distribution [6], fitness-distance correlation (FDC) [10], the correlation length [17, 20], epistasis measures [15], information analysis [18], modality and neutrality measures [16], and fitness-distance analysis [14]. Difficulty measures for BBO problems in particular were studied by [8], which concluded that in the worst case building predictive difficulty measures for BBO problems is not possible to do in polynomial time[3]. Most recently, Watson introduced several cost models for combinatorial landscapes in order to try to understand why certain algorithms perform well on certain landscapes [19].

In [13], the authors identify six "low-level feature classes" to classify BBO problems into groups. In [4], algorithm selection for BBO problems is considered with a focus on minimizing the cost of incorrect algorithm selections, unlike our approach, which minimizes a score based on the penalized expected runtime. Our approach also differs from online methods [5] and reactive techniques [3] that attempt to guide algorithms based on information from previously explored states because ISAC performs all of its work offline.

2 BBO Dataset & Solver Portfolio

We evaluate the effectiveness and robustness of our features on a dataset from the GECCO 2012 Workshop on Black-Box Optimization Benchmarking (BBOB) [2]. The dataset contains the number of evaluations required to find a particular objective value within some precision on one of 24 continuous, noise-free, optimization functions from [7] in 6 different dimension settings for 27 solvers. The solvers are all run on the data 15 times, each time with a different target value set as the artificial global optimum. Note that the BBOB documentation refers to each of these target values as an "instance". To avoid confusion with the instances that ISAC uses to train and test on, we will only refer to BBOB targets. Removing 7 instances from the dataset for which no solver was able to find a solution, the dataset consists of 1289 instances.

We use the 21 solvers of the BBOB dataset with full solution data for all instances. This portfolio consists of a diverse set of continuous optimizers, including 10 covariance matrix adaptation (CMA) variants, 8 differential evolution

[3] Our results do not contradict this, as we are not predicting the hardness of instances.

Problem definition features
1. Solver accuracy
2. Number of dimensions
Hill climbing features
3-4. Average distance between optima (average, std. dev.)
5-6. Distance between best optima and other optima (average, std. dev.)
7. Percent of optima that are the best optimum
Random point features
8-9. Distance to local optimum (average, std. dev.)
10. Fitness-distance correlation (FDC)

Table 1: BBO problem features.

(DE) variants, an ant colony optimization (ACO) algorithm, a genetic algorithm (GA), and a particle swarm optimization (PSO) algorithm[4].

3 Features

Computing features for BBO problems is difficult because evaluating the objective function of a BBO problem is expensive, and there is scarce information about a problem instance in its definition, other than the number of dimensions and the desired solver accuracy. In the absence of any structure in the problem definition, we have to sample the fitness landscape. However, such sampling is expensive, and on our dataset performing more than 600 objective evaluations removes all benefits of using a portfolio approach. We therefore introduce a set of 10 features that are based on well-studied aspects of search landscapes in the literature [19]. Our features are drawn from three information sources: the problem definition, hill climbs, and random points.

Table 1 summarizes our features. The *problem definition features* contain the desired accuracy of the continuous variables (Feature 1), and the number of dimensions that the problem has (Feature 2), which, together, describe the size of the problem. The *hill climbing features* are based off of a number of hill climbs that are initiated from random points and continued until a local optimum or a fixed number of evaluations is reached. We then calculate the average and standard deviation of the distance between optima (Features 3 and 4), which describes the density of optima in the landscape. Using the best optimum found, we then compute the average and standard deviation of the distance between the optima and the best optimum (Features 5 and 6), using the nearest to each non-best optimum for these features if multiple optima qualify as the best. Feature 7 describes what percentage of the optima are equal to the best optimum, giving a picture of how spread out the optima are throughout the landscape. The *random point features* 8 and 9 contain the average and standard deviation of the distance of each random point to the nearest optimum, which describes the distribution of local optima around the landscape. Feature 10 computes the fitness-distance correlation, a measure of how effectively the fitness value at a particular point can guide the search to a global optimum [10]. In feature 10, we compute an approximation to the FDC.

4 Numerical Results

In this section we describe the results of using our features, in full and in various combinations, to train a portfolio solver using the ISAC method on the BBOB

[4] Full details about the algorithms are available in [2].

2012 dataset. We measure the performance of each solver using a penalized score that takes into account the relative performance of each solver on an instance. We do not directly use the expected running time (ERT) value because the amount of evaluations can vary greatly between instances, and too much focus would be placed on instances where a large number of evaluations is required. The penalized score of solver s on an instance i is given by:

$$score(s,i) = \frac{PERT(s,i) - best(i)}{worst(i) - best(i)}$$

where $PERT(s,i)$ is the penalized ERT defined by

$$PERT(s,i) = \begin{cases} ERT(s,i) & \text{if } ERT(s,i) < \infty \\ worst(i) \cdot 10 & \text{otherwise,} \end{cases}$$

$best(i)$ refers to the lowest ERT score on instance i, and $worst(i)$ refers to the highest non-infinity ERT score on the instance. The penalized ERT therefore returns ten times the worst ERT on an instance for solvers that were unable to find the global optimum. We are forced to use a penalized measure because if a solver cannot solve a particular instance, it becomes impossible to calculate its performance over the entire dataset.

4.1 ISAC Results

Table 2 shows the results of training and testing ISAC on the BBOB 2012 dataset. For each entry in the table, we run a 10-fold cross validation using features from each of the 15 BBOB target values. The scores of each of the cross-validation folds are accumulated for each instance, and the entries in the table are the average and standard deviation across all instances in the dataset. We compare our results against the best single solver (BSS) on the dataset, MVDE [12], which is the best performing solver across all instances.

We train using several subsets of our features; only feature 1 (F_1), only feature 2 (F_2), and only features 1 and 2 ($F_{1,2}$). We then train using all features (All), and only landscape features (LSF), i.e., features 3 through 10. All* and LSF* include the evaluations necessary to compute the features, whereas all other entries do not include the feature computation in the results. We used several different settings of the number of hill climbs and maximum hill climb length based on our feature robustness experiments in [1]: 10 hill climbs of maximum length 10, 50 hill climbs of maximum length 20, and 200 hill climbs of maximum length 400. The closer a score is to 0 (the score of the virtual best solver) the better the performance of an approach.

Based on results for F_1, F_2 and $F_{1,2}$, the easy to compute BBO features alone are only able to give ISAC some information about the dataset, and that a landscape analysis is justified. On the other hand, F_2 outperforms BSS. In fact, F_2 performs equally well to All and LSF for cluster 100 with 10 hill climbs of length 10 and for 50 hill climbs of length 20. In addition, F_2 significantly outperforms All on cluster size 50, where it is clear that it overfits the training data. This is a clear indication that 10 hill climbs of length 10, or 50 hill climbs of length 20, do not provide enough information to train ISAC to be competitive with simply using the number of dimensions of a problem.

κ		10/10				50/20				200/400			
		Test		Train		Test		Train		Test		Train	
		⊘	σ	⊘	σ	⊘	σ	⊘	σ	⊘	σ	⊘	σ
50	BSS	2.23	5.29	2.23	5.29	2.23	5.29	2.23	5.29	2.23	5.29	2.23	5.29
	F_1	2474.63	3×10^5	2.04	5.08	2474.66	3×10^5	2.04	5.08	2474.64	3×10^5	2.04	5.08
	F_2	1.24	4.02	1.24	4.02	1.24	4.02	1.24	4.02	1.24	4.02	1.24	4.02
	$F_{1,2}$	189.11	6743.81	1.27	4.07	189.10	6743.81	1.27	4.07	189.10	6743.81	1.26	4.07
	All	51.32	1801.27	1.21	3.96	96.15	3105.76	0.79	2.94	13.41	452.79	0.82	3.30
	All*	51.42	1801.33	1.32	4.05	97.15	3110.46	1.82	9.90	95.25	1161.92	83.12	760.37
	LSF	1.25	4.01	1.24	4.00	88.18	3137.52	0.82	3.03	**0.53**	2.73	0.55	2.75
	LSF*	1.35	4.09	1.34	4.08	89.18	3142.23	1.85	9.93	99.44	1323.68	82.86	760.40
100	BSS	2.23	5.29	2.23	5.29	2.23	5.29	2.23	5.29	2.23	5.29	2.23	5.29
	F_1	2474.63	3×10^5	2.04	5.08	2474.66	3×10^5	2.04	5.08	2474.64	3×10^5	2.04	5.08
	F_2	1.24	4.02	1.24	4.02	1.24	4.02	1.24	4.02	1.24	4.02	1.24	4.02
	$F_{1,2}$	189.11	6743.81	1.27	4.07	189.11	6743.81	1.27	4.07	189.10	6743.81	1.27	4.07
	All	1.25	4.02	1.24	4.00	1.25	4.03	1.23	4.00	1.16	3.86	1.12	3.80
	All*	1.35	4.10	1.34	4.08	2.28	10.21	2.26	10.20	83.46	760.60	83.43	760.34
	LSF	1.25	4.02	1.24	4.01	1.22	3.99	1.19	3.93	1.20	3.85	1.15	4.00
	LSF*	1.35	4.10	1.34	4.09	2.25	10.19	2.22	10.17	97.31	1223.98	83.45	760.34

Table 2: The average and standard deviation of the scores across all instances for various minimum cluster sizes, numbers of hill climbs and hill climb lengths for the best single solver and ISAC using various features.

The fact that LSF* is able to match the performance of F_2 on 10 hill climbs of length 10 for both cluster size 50 and 100 an important accomplishment. With so little information learned about the landscape, the fact that ISAC can learn such an effective model indicates that our features are indeed effective.

Once we move up to 200 hill climbs of length 400, LSF significantly outperforms F_2, and even outperforms All, which suffers from overfitting. In fact, LSF is able to cut the total score to under a fourth of BSS's score, and to one half of F_2's score, indicating that the fitness landscape can indeed be used for a portfolio. In addition, LSF has a lower standard deviation than BSS. LSF's score on the training set of 0.53 and 0.55 on the test set are surprisingly close to the virtual best solver, which has a score of zero, indicating that ISAC is able to exploit the landscape features to nearly always choose the best or second best solver for each instance. On the downside, 200 hill climbs of length 400 requires too many evaluations to be used in a competitive portfolio, and All* needs 50 times the evaluations of BSS.

5 Conclusion and Future Work

We introduced a set of features based on accepted and well-studied properties and measures of fitness landscapes to categorize BBO problems for use in algorithm portfolios, like ISAC, that can greatly improve the ability of practitioners to solve BBO problems. We experimentally validated our features within the ISAC framework, showing that ISAC is able to exploit problem structure learned during feature computation to choose the fastest solver for an unseen instance. The success of the features we introduced clearly indicates that selecting algorithms from a portfolio based on the landscape structure is possible. For future work, we plan to use these features to analyze what types of landscapes fit best

to which solvers, which could influence solver development, allowing solvers to more specifically target problems they solve well.

References

1. T. Abell, Y. Malitsky, and K. Tierney. Fitness landscape based features for exploiting black-box optimization problem structure. Technical Report TR-2012-163, IT University of Copenhagen, 2012.
2. A. Auger, N. Hansen, V. Heidrich-Meisner, O. Mersmann, P. Posik, and M. Preuss. GECCO 2012 Workshop on Black-Box Optimization Benchmarking (BBOB). `http://coco.gforge.inria.fr/doku.php?id=bbob-2012`, 2012.
3. R. Battiti and M. Brunato. Reactive search optimization: learning while optimizing. *Handbook of Metaheuristics*, pages 543–571, 2010.
4. B. Bischl, O. Mersmann, H. Trautmann, and M. Preuß. Algorithm selection based on exploratory landscape analysis and cost-sensitive learning. In *GECCO'12*, pages 313–320, New York, NY, USA, 2012. ACM.
5. J. Boyan and A.W. Moore. Learning evaluation functions to improve optimization by local search. *The Journal of Machine Learning Research*, 1:77–112, 2001.
6. C. Brooks and E. Durfee. Using Landscape Theory to Measure Learning Difficulty for Adaptive Agents. In Alonso, E., et. al., editor, *Adaptive Agents and Multi-Agent Systems*, volume 2636 of *LNCS*, pages 561–561. Springer, 2003.
7. S. Finck, N. Hansen, R. Ros, and A. Auger. Real-parameter black-box optimization benchmarking 2010: Presentation of the noisy functions. Technical report, 2009/21, Research Center PPE, 2010.
8. J. He, C. Reeves, C. Witt, and X. Yao. A note on problem difficulty measures in black-box optimization: Classification, realizations and predictability. *Evolutionary Computation*, 15:435–443, December 2007.
9. H.H. Hoos and T. Stützle. *Stochastic Local Search: Foundations & Applications*. Morgan Kaufmann Publishers Inc., 2004.
10. T. Jones and S. Forrest. Fitness Distance Correlation as a Measure of Problem Difficulty for Genetic Algorithms. In *ICGA-95*, pages 184–192, 1995.
11. S. Kadioglu, Y. Malitsky, M. Sellmann, and K. Tierney. ISAC–Instance-Specific Algorithm Configuration. In *ECAI-10*, volume 215 of *FAIA*, pages 751–756, 2010.
12. V.V. Melo. Benchmarking the multi-view differential evolution on the noiseless bbob-2012 function testbed. In *GECCO'12*, pages 183–188. ACM, 2012.
13. O. Mersmann, B. Bischl, H. Trautmann, M. Preuß, C. Weihs, and G. Rudolph. Exploratory landscape analysis. In *GECCO'11*, pages 829–836. ACM, 2011.
14. P. Merz and B. Freisleben. Fitness Landscapes, Memetic Algorithms, and Greedy Operators for Graph Bipartitioning. *Evolutionary Computation*, 8:61–91, 2000.
15. B. Naudts and L. Kallel. A comparison of predictive measures of problem difficulty in evolutionary algorithms. *IEEE Trans. On Evo. Comp.*, 4(1):1 – 15, 2000.
16. T. Smith, P. Husbands, P. Layzell, and M. O'Shea. Fitness Landscapes and Evolvability. *Evolutionary Computation*, 10(1):1–34, 2002.
17. P.F. Stadler and W. Schnabl. The landscape of the traveling salesman problem. *Physics Letters A*, 161(4):337 – 344, 1992.
18. V.K. Vassilev, T.C. Fogarty, and J.F. Miller. Information Characteristics and the Structure of Landscapes. *Evolutionary Computation*, 8:31–60, March 2000.
19. J. Watson. An Introduction to Fitness Landscape Analysis and Cost Models for Local Search. In M. Gendreau and J. Potvin, editors, *Handbook of Metaheuristics*, volume 146, pages 599–623. Springer, 2010.
20. E. Weinberger. Correlated and uncorrelated fitness landscapes and how to tell the difference. *Biological Cybernetics*, 63:325–336, 1990.