Geometric Surrogate-Based Optimisation for Permutation-Based Problems

Alberto Moraglio[1], Yong-Hyuk Kim[2], and Yourim Yoon[3]

[1] School of Computing,University of Kent, Canterbury, UK
albmor@gmail.com
[2] Department of Computer Science and Engineering
Kwangwoon University, Seoul, Korea
yhdfly@kw.ac.kr
[3] School of Computer Science and Engineering
Seoul National University, Seoul, Korea
yryoon@soar.snu.ac.kr

Abstract. Surrogate models (SMs) are often used to tackle continuous optimisation problems with expensive objective functions. Recent work has shown that Radial Basis Function Networks can be used as SMs for combinatorial spaces based in principle on *any arbitrarily complex underlying solution representation* by extending their natural geometric interpretation from continuous to general metric spaces. This approach allows us to use SMs with the most natural representation for the problem at hand, yet without ad-hoc adaptation of the SM to the specific representation. In this paper, we illustrate how the framework applies to combinatorial problems using the permutation representation and report promising initial experimental results on the quadratic assignment problem.

1 Introduction

Many important real-world optimisation problems have objective functions which are prohibitively expensive to evaluate. For example, most engineering design problems are of this type (see e.g., [4]). They require experiments and/or simulations to evaluate to what extent the design objective has been met as a function of parameters controlling the design. Optimisation methods based on surrogate models have been successfully employed to tackle expensive objective functions. For a survey on surrogate model based optimisation methods refer to [2]. A surrogate model is a mathematical model that approximates as precisely as possible the expensive objective function of the problem at hand, and that is computationally much cheaper to evaluate. The objective function is considered unknown. The surrogate model is built solely from available known values of the expensive objective function evaluated on a set of solutions. We refer to the pair (solution, known objective function value) as data-point.

The traditional procedure of surrogate model based optimisation (SMBO) [2] is outlined in Algorithm 1. Note that the role of the evolutionary algorithm in the

Algorithm 1 Surrogate Model Based Optimisation

1: Sample uniformly at random a small set of candidate solutions and evaluate them using the expensive objective function (initial set of data-points)
2: **while** limit number of expensive function evaluations not reached **do**
3: Construct a new surrogate model using all data-points available
4: Determine the optimum of the surrogate model by search, e.g., using an evolutionary algorithm (this is feasible as the model is cheap to evaluate)
5: Evaluate the solution which optimises the surrogate model in the problem with the expensive objective function (additional data-point available)
6: **end while**
7: Return the best solution found (the best in the set of data-points)

SMBO procedure is to infer the location of a promising solution of the problem using the surrogate model, and it is not directly applied to the original problem with the expensive objective function. This is feasible because the computational cost of a complete run of the evolutionary algorithm on the surrogate model is negligible (in the order of few seconds) with regard to the cost of evaluating a solution using the expensive objective function of the problem (in the order of minutes, hours or even days depending on the problem).

SMBOs are naturally suited to continuous optimization or to discrete problems when solutions are vectors of integers, as there are many statistical techniques that can be used to build a surrogate model of the fitness landscape from data-points based on these representations. However, in many optimization problems, natural solution representations are not real or integer vectors, but they can be permutations, variable-length sequences, trees, graphs or any arbitrarily complex structures. The choice of an adequate representation for the problem at hand is often critical to the success of the search algorithm, whichever particular search algorithm is to be used.

Permutations and related representations are natural representations for solutions of many important combinatorial optimisation problems such as scheduling problems. In real-world problems, candidate scheduling solutions may need to be tested in a complex setting by running a computationally expensive simulation of, for example, an entire production process, to be evaluated. This makes SMBOs very relevant to this type of problems. However, although permutations can be regarded as a special type of vectors, they generally cannot be effectively treated as vectors as the relevant information they encode about the problem at hand is in the order of the elements, rather than in the absolute values at each location in the vector. This makes traditional SMBO approaches unsuited to permutations.

There are studies focusing on specific real-world applications with expensive objective functions which are inherently combinatorial problems with structured solutions (e.g., graphs) which are, however, approached by encoding solutions in vectorial form to use standard surrogate models (e.g., [5]). Could we adapt the surrogate model to the natural data structure of the search problem instead? Is

there a systematic and rigorous way to adapt a surrogate model to the target representation which does not require us to rethink the surrogate model, or make ad-hoc adaptation to the model, for any target representation considered however complex it is? In very recent work [3], the authors proposed a generalisation of a well-known class of surrogate models – Radial Basis Function Networks (RBFNs) [1] – which answers in the affirmative the questions above. In this paper, we illustrate how that framework can be straightforwardly applied to permutation-based problems, and report initial experimental results.

The reminder of the paper is organised as follows. In Section 2, we report the generalised model of RBFNs, and its specialisation to permutations. In Section 3, we present experiments testing the SMBO specialised to permutations to the well-known quadratic assignment problem, which we will consider having costly objective function[4]. Finally, in Section 4, we present conclusions and future work.

2 Generalised Radial Basis Function Networks

A radial basis function (RBF) is a real-valued function $\phi : \mathbf{R}^n \rightarrow \mathbf{R}$ whose value depends only on the distance from some point $\mathbf{c}$, called a *center*, so that $\phi(\mathbf{x}) = \phi(\|\mathbf{x} - \mathbf{c}\|)$. The point $\mathbf{c}$ is a parameter of the function. The norm is usually Euclidean, so $\|\mathbf{x} - \mathbf{c}\|$ is the Euclidean distance between $\mathbf{c}$ and $\mathbf{x}$. The most frequently used types of radial basis functions are Gaussian functions of the form:

$$\phi(\mathbf{x}) = \exp(-\beta\|\mathbf{x} - \mathbf{c}\|^2)$$

where $\beta > 0$ is the width parameter. Radial basis functions are used to build function approximations of the form:

$$y(\mathbf{x}) = w_0 + \sum_{i=1}^{N} w_i \, \phi(\|\mathbf{x} - \mathbf{c}_i\|).$$

See Figure 1 for an example of function obtained as a weighted sum of Gaussian functions on the real line.

In an RBFN there are three types of parameters that need to be determined to optimise the fit between $y(\mathbf{x})$ and the data: the weights w_i, the centers $\mathbf{c}_i$, and the RBF width parameters β_i. A widely applied procedure to fit RBFNs to the data consists of choosing the centers $\mathbf{c}_i$ to coincide with the known points $\mathbf{x}_i$, and choosing the widths β_i according to some heuristic based on the distance to nearest neighbors of the center $\mathbf{c}_i$ or to the maximum distance between the

[4] The aim of the present paper is to show that the generalised SMBO provides meaningful results when applied to well-studied permutation problems. This preliminary step, before attacking real-world problems with black-box expensive objective functions, is necessary because the transition from the continuous space to permutation spaces is a huge leap, as the latter present a completely different geometry from the traditional geometry which may affect in completely unpredictable ways the behaviour of the algorithm.

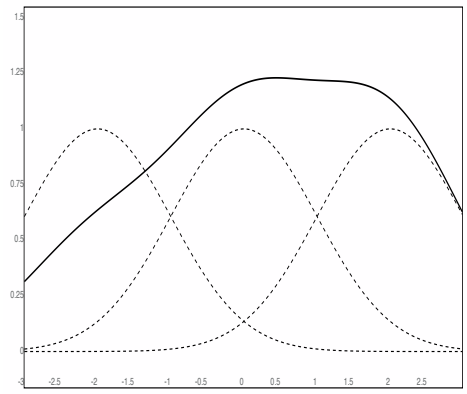

Fig. 1. Example of a function (solid line) obtained as a weighted sum of three Gaussian functions (dashed lines) on the real line.

chosen centers. The bias w_0 is set to the average of the function values b_i at the known data-points (i.e., function values of the points in the training set). The weights w_i can be determined by solving the system of N simultaneous linear equations in w_i obtained by requiring that the unknown function interpolates exactly the known data-points, which can be solved using simple linear algebra, involving a matrix inversion (see [1] for details).

In very recent work [3], the authors used a geometric methodology to generalise RBFNs to any solution representation. The gist of the idea is that all aspects of RBFNs that allow us to use them as surrogate models, i.e., model definition and representation, training, querying and searching of RBFNs can be naturally generalized from Euclidean spaces to general metric spaces, simply by replacing the Euclidean distance with a generic metric. The generalized model applies to any underlying solution representation once a distance function rooted on that representation is provided (e.g., Swap distance on permutations). In particular, this method can be used *as it is* to learn in principle any function mapping directly complex structured representations to reals without introducing any arbitrary ad-hoc adaptation to the RBFNs. There is no special requirement of pre-processing the target representation and shoehorn it in a vector of features[5].

[5] This way of generalizing RBFNs to structured representations is related to kernel methods in Machine Learning. However, in those methods the types of distances to be used between objects are required to be implicitly embedded in a vector space (i.e., positive-definite kernels), which makes designing these distances complicated, whereas this restriction is not present in the proposed approach (see [3]).

3 Experiments

Experiments have been carried out on three standard quadratic assignment problem (QAP) instances (kra32, tho30 and nug30, where the number in the name indicates the instance size), and on two instances of a unimodal problem on permutations of size 30, in which the fitness of a permutation (to minimise) is given by its distance to an arbitrary but fixed permutation. The last problem can be seen as a generalisation of the OneMax problem for binary strings[6]. The two instances of the unimodal problem are obtained by using two different distance functions on permutations, the Hamming distance (unih30) and the Swap distance (unis30). The reason we included the unimodal problem in the test-bed is because we wanted to test the SMBO under controlled conditions on a problem whose topographic features of the fitness landscape are explicit and completely transparent. We will consider the problems in the test-bed as having costly objective functions, and leave as future work testing the SMBO on real-world problems with expensive objective functions. Furthermore, using a larger test-bed and testing the scalability of SMBO with respect to instance size would be desirable. However, we found that it would take an excessive amount of time, as the surrogate model is searched every time it is used to suggest a solution to test in the expensive objective function. We will also consider a larger test-bed and a scalability analysis in future work.

As distance functions between permutations as base for the instantiation of the SMBO, we have used the Hamming distance and the Swap distance. We will refer to these algorithms as $SMBO_H$ and $SMBO_S$, respectively. Clearly, the choice of a distance well suited to the problem at hand is crucial to obtain a surrogate model able to make meaningful predictions and guide appropriately the search of the SMBO. In this paper, we limit ourselves to experiment with these two distances. In future work, we will investigate other distances and other problems in the attempt to find out a rule to select *a priori* a good distance for a given type of problem.

We use a standard surrogate model based optimisation algorithm (see Algorithm 1). The surrogate model used is an RBFN model which is fitted to the available data-points using the simplified learning procedure presented in the previous section. The centers $\mathbf{c}_i$ of the radial basis functions are chosen to coincide with all available data-points. For the $SMBO_H$, the widths of the radial basis functions are assigned to the same value $\beta = \frac{1}{2D^2}$ where D is the maximum distance between all centers. With this setting of β, each radial basis function is "spread on the space" to cover all other centers so that each known function value at a center can potentially contribute significantly to the prediction of the function value of any point in space, and not only locally to function values of

[6] This is because in the OneMax problem the fitness of a solution to maximise is the number of ones it has. This is equivalent to a problem in which the fitness of a solution to minimise is given by the Hamming distance from the solution to the string with all bits set to one. For the symmetry of the Hamming space, this problem is in turn equivalent to any problem in which the string with all bits set to one is replaced with a fixed by arbitrary target string.

points near the given center (i.e., we force the surrogate model to be a global approximating function). From preliminary trials, we found that this setting for β did not work well for $SMBO_S$, and found that $\beta = \frac{1}{D/5}$ produced better results. Admittedly, the parameter β affects greatly the accuracy of the predictions of the surrogate model. So, either this parameter is tuned or, more interestingly, it could be 'learnt' from the sampled data-point. The value of the bias term w_0 is set to the average function value of the known data-points, i.e., the average of vector b_i. In this way, the predicted function value of a point which is out of reach of the influence of all centers is by default set to the average of their function values. The coefficients w_i of the radial basis functions in the linear model are determined by least squares minimisation as described in the previous section.

The other settings of surrogate model based optimisation are as follows. For all problems, the number of total available expensive function evaluations is set to $n = 100$. Clearly, this setting is just a term of reference, as for different problems one may have a different number of solutions available with regard to the problem size. For both $SMBO_H$ and $SMBO_S$, we set the size of the initial sample of data-points to 10, and the number of sample points suggested by the surrogate model to $n-10 = 90$. To search the surrogate model we use a memetic algorithm on permutations with truncation selection, cycle crossover, swap mutation with mutation rate 0.01, and local search based on the 2-opt neighbourhood. The population size and the number of generations are both set to 20, and 10 new offspring are generated each generation, which provide the genetic algorithm (GA) with enough trials to locate a good solution of the surrogate model[7]. The solution with the best predicted objective value is evaluated with the expensive objective function, provided it has not been sampled before. If that is the case, the second best is sampled provided has not been sampled before. Otherwise the third best is sampled, and so on.

To have a reference for the performance of the SMBO algorithm, we compared it with a standard GA and with random search (RS) applied directly on the problem with the expensive objective function. The reason we included random search in the set of algorithms is because, whereas it is safe to assume that in practice GA is better than random search in normal circumstances, with small samples random search can do relatively well. We gave all algorithms in the comparison exactly the same number of expensive objective functions, which is n trials, and report the best solution found. The GA used has a population of 10 individuals runs 18 generations, and it generates 5 new individuals each generation. It uses truncation selection, swap mutation with probability of 0.01 and cycle crossover. For any of the problem in the test-bed and each algorithm in the comparison, we did 50 independent runs of each experiment.

Table 1 reports the results of the comparison. These results consistently rank the $SMBO_H$ as the most effective algorithm, followed by the GA, then by the $SMBO_S$ and finally followed by the RS. An initial conclusion that can be drawn

[7] Ideally, we would like to search the surrogate model more extensively to locate its optimum. However, this was unfeasible as it required excessive time to complete the experiments. We found that the proposed setting is an acceptable trade-off.

Table 1. Results for Random Search (RS), Genetic Algorithm (GA), $SMBO_H$ and $SMBO_S$ on QAP instances (kra32, tho30, nug30) and unimodal instances (unih30, unis30). Best, average and standard deviation over 50 independent runs of the best fitness found by each algorithm are reported.

Algorithm	**Best**	**Average**	**SD**
kra32			
RS	24156	25008.036	434.752
GA	23440	24625.032	586.598
$SMBO_H$	**22590**	**24094.320**	616.714
$SMBO_S$	23848	24833.920	452.219
tho30			
RS	190256	196662.822	3211.198
GA	**180274**	194389.912	4414.218
$SMBO_H$	180860	**193415.440**	4629.032
$SMBO_S$	186172	195231.920	3534.208
nug30			
RS	7350	7618.000	84.598
GA	7296	7558.360	126.650
$SMBO_H$	**7276**	**7500.240**	97.363
$SMBO_S$	7328	7563.880	94.384
unih30			
RS	24	25.796	0.784
GA	22	25.044	1.263
$SMBO_H$	**17**	**20.840**	1.554
$SMBO_S$	21	25.180	0.953
unis30			
RS	19	21.657	0.871
GA	17	20.906	1.294
$SMBO_H$	**15**	**18.480**	1.825
$SMBO_S$	19	21.040	0.937

is that the surrogate based on the Hamming distance really helps in locating better solutions given the same amount of expensive fitness evaluations, both on the QAP and on the unimodal instances. A rather surprising result is that the surrogate model based on the swap distance does not seem to be very effective, as it is better than random search but worse than the GA. This is surprising because the Hamming distance and the Swap distance are related distances. It is also noteworthy that the $SMBO_H$ is better than the $SMBO_S$ on the unimodal landscape under the Swap distance (unis30), which we introduced in the test-bed of problems because intuitively this can be thought as the easiest landscape to optimise for the $SMBO_S$.

To have a better picture of the working mechanism of the SMBO algorithms we report the following analysis. Firstly, to make sure that the distances chosen as the basis for the SMBO are suited to the problem at hand, we did a static

Table 2. Correlation between predicted fitness and real fitness on a test set of randomly sampled solutions after the surrogate models have been trained on 50 randomly sampled data-points. The table reports out of 50 independent experiments the number of times a significantly positive correlation (larger than 0.15) and a significantly negative correlation (less than −0.15) have occurred. The bracketed number is the average correlation over the 50 experiments.

Instance	**Hamming Model**		**Swap Model**	
	pos (ave)	neg	pos (ave)	neg
kra32	46 (0.222)	0	34 (0.198)	0
tho30	12 (0.180)	0	7 (0.159)	0
nug30	32 (0.184)	0	21 (0.186)	0
unih30	50 (0.771)	0	50 (0.431)	0
unis30	47 (0.240)	0	39 (0.185)	0

Table 3. Fitness-distance correlation with respect to Hamming distance and Swap distance (positive values mean positive correlation).

Instance	**Hamming Distance**	**Swap Distance**
kra32	−0.111	0.109
tho30	0.456	0.515
nug30	0.380	0.435
unih30	1.000	1.000
unis30	1.000	1.000

analysis of the predictive power of the surrogate models when they are used in isolation from the SMBO. This analysis is presented in Table 2. By looking at the counts of the occurrences of significantly positive correlations and at the average correlation, it is evident that on all instances the Hamming model produces better prediction than the Swap model. It is also noteworthy that both models are never deceptive as the count for the negative correlations is zero. It is also interesting to look at the fitness-distance correlation analysis with respect to both Hamming distance and Swap distance (see Table 3). This analysis shows that the Swap distance is more suited than the Hamming distance to the QAP instances, as it obtains higher correlation. However, this is not reflected in the performance. So, the static prediction power of a model is a better indicator of which distance to use with the SMBO.

Apart from choosing a distance that is able to lead to meaningful predictions, there are other aspects of the underlying surrogate model which may affect the performance of the associated SMBO. One is as follows. In theory, we would like to locate the optimum of the surrogate model and propose its optimum to be sampled in the expensive objective function. However, in practice, to save computational time the surrogate model is searched with a GA which does not guarantee to find the optimum or even a good solution. In this regard, it is

Table 4. Fitness-distance correlation on the surrogate models after training them with 100 randomly sampled data-points (positive values mean positive correlation).

Instance	**Hamming Model**	**Swap Model**
kra32	0.879	0.795
tho30	0.821	0.830
nug30	0.877	0.713
unih30	0.860	0.932
unis30	0.868	0.790

Table 5. Number of solutions obtained by optimising the surrogate model by the SMBO algorithm (90 sequential optimisations) whose predicted fitness matches or improves on the fitness of the best known solution so far. Average of 50 runs.

Instance	$SMBO_H$		$SMBO_S$	
	♯ better	♯ equal	♯ better	♯ equal
kra32	22.28	0	0	0
tho30	28.16	0	0.04	0
nug30	26.70	0	0.02	0
unih30	4.28	0	0.28	0
unis30	19.64	0	0.04	0

interesting to know how difficult the surrogate model it could be for the GA. In Table 4, we present a fitness-distance correlation on the surrogate models after training them with 100 randomly sampled data-points. As all correlation values are extremely high, we can conclude that the surrogate model is likely to be very easy to search with a GA, which therefore is likely to locate its optimum or a very good solution most of the times.

There is a further aspect of the surrogate model that may have an impact of the performance of the associated SMBO. The choice of the distance and the model parameter β affects the topography of the model. In particular, these choices affect the property of the model of being extrapolative, which is, of having its global optimum with better fitness than the fitness of the data points used to train it. A model which is both strongly predictive and extrapolative is an ideal candidate to be used with an SMBO because its optimum, whose predicted fitness improves over the best fitness of the sampled points so far, it is likely to have real fitness which improves over the best fitness known so far. Table 5 shows that in this respect $SMBO_H$ and $SMBO_S$ are very different in that $SMBO_H$ can extrapolate much more often than $SMBO_S$. Together with the different predictive powers of the surrogate models based on Hamming distance and on Swap distance, this provides an explanation as to why the $SMBO_H$ performs much better than $SMBO_S$. Admittedly, it is still eluding us exactly what particular property is which the Hamming distance has and that the Swap distance has

not that makes the former distance more suitable to be used effectively with an SMBO. We will investigate this further in future work.

4 Conclusions and Future Work

There are many potentially interesting applications of a surrogate model based optimisation framework that can naturally encompass more complex representations beyond the traditional vector-based representation. A direct approach to representations greatly enlarges the scope of SMBO to complex representations which cannot be naturally mapped to vectors of features.

In very recent work [3], a conceptually simple, formal, general and systematic approach to adapt an SMBO algorithm based on RBFNs to *any* target representation was put forward. The new framework was obtained using a geometric generalisation methodology which requires to write the original continuous algorithm only in terms of Euclidean distances between candidate solutions, which then can be generalised by replacing the Euclidean distance function with a generic metric. Then the formal algorithm obtained can be formally specified to any target representation by employing as underlying metric a distance rooted on the target representation (e.g., edit distance). RBFNs can be naturally generalised to encompass any representations because both the approximating model and the learning of the parameter of the model can be cast completely in a representation-independent way, and rely only on distance relations between training instances and query instances.

In this paper, as a preliminary experimental validation of the framework, we have considered the permutation representation endowed with the Hamming distance and with the Swap distance, and tested the SMBO on the QAP and on unimodal problems, obtaining consistently that with the same budget of expensive function evaluations, the SMBO with Hamming distance performs best in a comparison with other search algorithms. This shows that this framework has potential to work well on other more complex representation associated with combinatorial spaces. Surprisingly, the SMBO based on the Swap distance does not work as well as the SMBO based on the Hamming distance. We have presented an analysis in the attempt to elucidate the causes of the different performance. Further investigation is required to pin-point the structural difference between the Hamming distance and the Swap distance that gives rise to the difference in performance.

Much work remains to be done. Firstly, we will do more systematic experiments with well-studied permutation-based problems and consider other distances for permutations (e.g., we will consider the traveling salesman problem and a distance based on the 2-opt move as a basis of the surrogate model). We will also test the framework on standard problems for more complex representations, such as variable-length sequences and trees. Then, we will test how the system performs on a number of challenging real-world problems.

References

1. L. C. Jain. *Radial Basis Function Networks*. Springer, 2001.
2. D. R. Jones. A taxonomy of global optimization methods based on response surfaces. *Journal of Global Optimization*, 21(4):345–383, 2001.
3. A. Moraglio and A. Kattan. Geometric generalisation of surrogate model based optimisation to combinatorial spaces. In *Proceedings of the 11th European Conference on Evolutionary Computation in Combinatorial Optimisation*, 2011. (to appear).
4. S. Tong and B. Gregory. Turbine preliminary design using artificial intelligence and numerical optimization techniques. *Journal of Turbomachinery*, 114:1–10, 1992.
5. I. Voutchkov, A. Keane, A. Bhaskar, and T. M. Olsen. Weld sequence optimization: the use of surrogate models for solving sequential combinatorial problems. *Computer Methods in Applied Mechanics and Engineering*, 194:3535–3551, 2005.