Conference Presentation

A Surrogate Model Assisted Evolutionary Algorithm for Computationally Expensive Design Optimization Problems with Discrete Variables

Liu, B., Sun, N., Zhang, Q., and Gielen, G.

A Surrogate Model Assisted Evolutionary Algorithm for Computationally Expensive Design Optimization Problems with Discrete Variables

Bo Liu
Department of Computing, Glyndwr University, UK
Email: b.liu@glyndwr.ac.uk

Nan Sun
Department of Engineering, Glyndwr University, UK
Email: nan.b.sun@gmail.com

Qingfu Zhang
Department of Computer Science, City University of Hong Kong, Kowloon, Hong Kong SAR
Email: qingfu.zhang@cityu.edu.hk

Georges Gielen
ESAT-MICAS, Katholieke Universiteit Leuven, Belgium
Email: Georges.Gielen@esat.kuleuven.be

Vic Grout
Department of Computing, Glyndwr University, UK
Email: v.grout@glyndwr.ac.uk

***Abstract*—Real-world computationally expensive design optimization problems with discrete variables pose challenges to surrogate-based optimization methods in terms of both efficiency and search ability. In this paper, a new method is introduced, called surrogate model-aware differential evolution with neighbourhood exploration, which has two phases. The first phase adopts a surrogate-based optimization method based on efficient surrogate model-aware search framework, the goal of which is to reach at least the neighbourhood of the global optimum. In the second phase, a neighbourhood exploration method for discrete variables is developed and collaborates with the first phase to further improve the obtained solutions. Empirical studies on various benchmark problems and a real-world network-on-chip design optimization problem show the combined advantages in terms of efficiency and search ability: when only a very limited number of exact evaluations are allowed, the proposed method is not slower than one of the most efficient methods for the targeted problem; when more evaluations are allowed, the proposed method can obtain results with comparable quality compared to standard differential evolution, but it requires only 1% to 30% of exact function evaluations.**

1. Introduction

Many real-world design optimization problems require computationally expensive simulations to evaluate candidate solutions. To address these problems, using computationally cheap surrogate models to replace expensive exact evaluations is a routine approach. For design optimization problems, the objective functions are often continuous but the design variables are or include discrete variables in many real-world applications (e.g., mixed-integer problems). Far less research has been carried out for expensive design optimization problems with discrete variables (EDOD) than that focusing on expensive continuous optimization. Discrete variables include numerical, categorical and Boolean variables, etc. Categorical and Boolean variables involve combinatorial space, where the distance measure between candidate solutions is a main research concentration [1], [2] and is not the goal of this research. Instead, we focus on discrete numerical variables, which are sometimes unavoidable in product design and manufacturing. For example, [3], [4] show that most discrete variables in electronic circuit and system design (e.g., number of fingers of transistors, design grids) are discrete numerical variables and other categories of discrete variables are often not involved. In this paper, we try to obtain combined advantages on efficiency and solution quality for complex EDOD. In particular, the following two requirements must be met at the same time: (1) The allowed number of exact evaluations can be very limited due to expensive evaluations; (2) High search ability is required because of both discontinuous landscapes and problem complexity. This poses significant challenges to surrogate-based non-population optimization algorithms (SBNOAs) and surrogate model-assisted evolutionary algorithms (SAEAs) [5], [6], [7].

For SBNOAs, successful methods that handle discrete numerical variables include [5], [8], [9], etc. The generation of possible promising candidate solutions is based on a random sampling strategy [5], solving an auxiliary optimization problem [9], or traditional heuristic search methods [8] such as branch and bound. A reasonably optimal solution can often be obtained within a limited number of exact evaluations. However, because matured population-based search engines are often not adopted for the search, obtaining highly optimized results (e.g., comparable to standard EA) for complex problems and / or problems with dozens of variables is often a challenge even when more exact evaluations are allowed for many SBNOAs. Many SAEA methods, on the other hand, can often obtain highly optimized solutions, but may

have difficulty to obtain a great efficiency improvement due to the nature of standard EAs (see Section 3 for details). [10] develops a radial basis function (RBF)-assisted genetic algorithm for mixed continuous and discrete optimization and requires 30% to 80% exact evaluations compared to a standard genetic algorithm obtaining comparable results. The surrogate model-aware evolutionary search (SMAS) [11] framework uses a population-based search but with a new algorithm structure different from standard EA. It inherits the high search ability of EA to a large extent, but avoids the more necessary function evaluations of SAEAs with a standard EA structure. Comparisons show a substantial speed improvement compared to several state-of-the-art SAEAs using 20-30-dimensional continuous optimization problems [11]. However, pilot experiments show that SMAS can be trapped in local optima for EDOD.

To address the above challenges, a new method, called two-phase surrogate model-aware differential evolution (DE) with neighbourhood exploration (SMDN), is proposed. SMDN has two main innovations for achieving the combined high search ability and high efficiency: (1) Proposing a two-phase approach rather than traditional one phase SAEA or memetic SBO: The first phase adopts an improved SMAS framework-based SAEA for handling discrete variables, the goal of which is to reach at least the neighbourhood of the global optimum efficiently even when being trapped in local optima. A neighbourhood exploration method is used in the second phase and is collaborating with SMAS to jump out of local optima and shorten the distance to the global optimum. (2) Proposing a neighbourhood exploration method for discrete numerical variable optimization: unlike local search in memetic algorithms, the neighbourhood exploration has an opposite goal. It must have the ability to find potentially good candidates with discrete numerical variables, to improve diversity and to make use of surrogate modeling in the neighbourhood. SMDN aims to:

- obtain comparable efficiency with state-of-the-art SBNOAs using a very limited number of exact evaluations;
- provide highly optimized results comparable to standard EAs when slightly more exact evaluations are allowed, including problems with both discontinuous and rugged landscapes;
- use much fewer exact evaluations than standard EAs;
- although concentrating on fundamental problem (unconstrained EDOD), it should be straightforward to be applied to more general problems, such as mixed-integer problems, and has a good compatibility with existing successful techniques, e.g., constraint handling, hybrid surrogate models [12].

The remainder of this paper is organized as follows. Section 2 briefly introduces the basic techniques. Section 3 presents the SMDN method. Section 4 presents results of the empirical study, including benchmark problems and a real-world electronic engineering problem. Concluding remarks are presented in Section 5.

2. Basic Techniques

2.1. Gaussian Process Surrogate Modeling

In literature, Gaussian process (GP) [13] and RBF surrogate modeling are widely used for the targeted problem. GP modeling is adopted in SMDN because GP modeling can provide an estimate of the model uncertainty for each predicted point and several presceening methods utilizing the prediction uncertainty show good performance [14]. To model an unknown function $y = f(x), x \in R^d$, GP modeling assumes that $f(x)$ at any point x is a Gaussian random variable $N(\mu, \sigma^2)$, where μ and σ are two constants independent of x. For any x, $f(x)$ is a sample of $\mu + \epsilon(x)$, where $\epsilon(x) \sim N(0, \sigma^2)$. By maximizing the likelihood function that $f(x) = y^i$ at $x = x^i$ $(i = 1, \ldots, K)$ (where $x^1, \ldots, x^K \in R^d$ and their f-function values $y^1, \ldots, y^K$ are K training data points) and best linear unbiased prediction:

$$\hat{f}(x) = \hat{\mu} + r^T C^{-1}(y - \mathbf{1}\hat{\mu}) \tag{1}$$

the mean squared error is:

$$s^2(x) = \hat{\sigma}^2[1 - r^T C^{-1} r + \frac{(1 - \mathbf{1}^T C^{-1} r)^2}{\mathbf{1}^T C^{-1} r}] \tag{2}$$

where $r = (c(x, x^1), \ldots, c(x, x^K))^T$. C is a $K \times K$ matrix whose (i, j)-element is $c(x^i, x^j)$. $c(x^i, x^j)$ is the correlation function between x^i and x^j [13]. $y = (y^1, \ldots, y^K)^T$ and $\mathbf{1}$ is a K-dimensional column vector of ones. Detailed implementations are in [11].

For a minimization problem, given the predictive distribution $N(\hat{f}(x), s^2(x))$ for $f(x)$, a lower confidence bound (LCB) prescreening of $f(x)$ is used as in [11] to promote explorative global search:

$$\begin{aligned} &f_{lcb}(x) = \hat{f}(x) - \omega s(x) \\ &\omega \in [0, 3] \end{aligned} \tag{3}$$

where ω is a constant. More details are in [15].

2.2. Individual Solution-based Training Data Selection

There are different simple empirical methods to select proper training data points from a database for surrogate modeling. A revised individual solution-based training data selection (ISS) method [16] is used in SMDN. Given a database D, and a population of candidate solutions U to be predicted, ISS for discrete variables works as follows:

(1): Round vectors in D and U to the nearest allowed discrete values.

(2): For each solution in U, take the nearest $s_1 \times d$ solutions in D (based on Euclidean distance) as temporary training data points.

(3): Combine all the temporary training data points and remove the duplicate ones.

2.3. Differential Evolution

DE is an effective and popular global optimization algorithm. Besides continuous optimization, [17] shows its good ability to handle mixed-discrete optimization problems by a quantization method: the floating numbers are used in the DE operators and they are rounded to the nearest allowed values only for function evaluations. DE with this quantization method is used as the search engine in SMDN.

Suppose that P is a population and the best individual in P is x^{best}. Let $x = (x_1, \ldots, x_d) \in R^d$ be an individual solution in P. To generate a child solution $u = (u_1, \ldots, u_d)$ for x, the DE/current-to-best/1 strategy works as follows.

A donor vector is first produced by mutation:

$$v_i = x^i + F \cdot (x^{best} - x^i) + F \cdot (x^{r_1} - x^{r_2}) \qquad (4)$$

where x^{r_1} and x^{r_2} are two different solutions randomly selected from P and also different from x^{best} and x^i. $F \in (0, 2]$ is the scaling factor [17]. Then the following crossover operator is applied to produce the child u:

(1) Randomly select a variable index $j_{rand} \in \{1, \ldots, d\}$,

(2) For each $j = 1$ to d, generate a uniformly distributed random number $rand$ from $(0, 1)$ and set:

$$u_j = \begin{cases} v_j, & \text{if } (rand \leq CR) | j = j_{rand} \\ x_j, & \text{otherwise} \end{cases} \qquad (5)$$

where $CR \in [0, 1]$ is a constant called the crossover rate.

3. The SMDN Algorithm

3.1. The SMDN Framework

The SMDN algorithm works as follows.

The SMDN Algorithm

Step 1: Use Latin Hypercube sampling [13] to sample α candidate solutions from $[a, b]^d$, where a and b are the lower and upper bounds of the decision variables, respectively. Perform exact evaluations (including quantization) to these points and let them form the initial database D.

Step 2: If a preset stopping criterion is met (e.g., a certain maximum number of exact evaluations), output the best solution in D; otherwise find the current optimization phase (Phase 1 / Phase 2). Go to Step 3.

Step 3: Select from D the λ best candidate solutions in terms of exact evaluation to form a population P.

Step 4: Adaptively update the crossover rate. Apply the DE current-to-best/1 mutation (4) and crossover (5) operations on P to generate λ child solutions.

Step 5: Use ISS to select training data to construct a GP surrogate model for the λ child solutions.

Step 6: Prescreen the λ child solutions generated in Step 4 by using the GP model with LCB prescreening. Select the estimated best child solution x_{be} based on the LCB values of each candidate solution.

Step 7: (1) If x_{be} can be found in D, (i) in Phase 1, go to Step 7.1; (ii) in Phase 2, go to step 7.2. (2) Otherwise, perform an exact evaluation to x_{be}.

Step 7.1: Use the proposed perturbation method (Section 3.3) to generate a new candidate solution and to perform exact evaluation to it.

Step 7.2: Use the proposed neighbourhood exploration method (Section 3.3).

Step 8: Update D by adding the new solutions and their exact function values. Go back to Step 2.

In SMDN, Phase 1 is used from the beginning until the best solution found so far does not improve after TH exact evaluations. At this point, we assume that: (1) The search will not be effective in future iterations when only using the strategy from Phase 1. (2) The obtained solution is not far from the global optimum due to the effective Phase 1 search. Our pilot experiments verify that these two assumptions happen in almost all of the test cases. In Phase 2, for the revisited (good) solutions (x_{be} which is already included in D), instead of only using the proposed perturbation method, an iterative neighbourhood exploration operation is carried out. Note that in Phase 2, the neighbourhood exploration is only applied to revisited solutions. The reason is that SMAS with DE-based search is also vital in Phase 2 when appropriate solutions from neighbourhood exploration are added to the population.

3.2. Phase 1: The SMAS-based SAEA

The key idea of SMAS [11] is inherited in SMDN. In each iteration, the λ current *best* candidate solutions form the parent population (it is reasonable to assume that the search focuses on the promising subregion) and the prescreened best candidate in the child population is selected to replace the worst one in the parent population. Hence, only at most one candidate is changed in the parent population in each iteration; so the best candidate in the child solutions in several consecutive iterations may be quite near to each other, which will then be evaluated and are used as training data points. Due to this, the training data points describing the current promising region can be much denser compared to those generated by a standard EA population updating mechanism, which may spread in different subregions of the decision space and where there may not be sufficient training data points around the candidate solutions to be prescreened. Therefore, SMAS can obtain comparable high quality solutions in much fewer exact evaluations than several state-of-the-art SAEAs with the standard EA structure [11].

Because a higher search ability is required for EDOD, four improvements are done to the standard SMAS-based

SAEA [11]. The DE/current-to-best/1 mutation strategy and the ISS method are used to replace the DE/best/1 and the training data selection method using the latest available data. The discussion and experimental verifications are in our earlier work [16] and are not repeated here.

The crossover rate CR is adaptively adjusted in SMDN. CR is usually the most sensitive parameter in standard DE and good values for CR generally fall into a small range for some problems [18]. [18] proposes an effective self-adaptive method to select CR. Because a standard EA structure is not used and only one mutation strategy is used in SMAS, a method is proposed based on similar ideas. CR is generated from $N(CR_m, 0.1)$ for each pair of candidate solutions. In the first L iterations, the initial CR_m is used. Then, the following procedure will be used.

The Self-Adaptive Generation of CR

$CR_m = median(CRM)$
FOR $i = 1$ to λ
$CR_i = Normrnd(CR_m, 0.1)$
IF $CR_i < 0, CR_i = 0$; IF $CR_i > 1, CR_i = 1$;
END
$CRM = CRM \cup CR_b$

where the function $Normrnd$ generates a Gaussian distributed random number, and CR_b refers to the CR value used for generating x_{be} (see Step 6), which will be added to the memory of successful CR values, CRM. It is reasonable to assume that most of the CR values used for generating x_{be} are appropriate for the given problem.

In Phase 1, when x_{be} can be found from D, a new candidate solution is generated by the proposed perturbation method to explore the neighbourhood, as will be introduced in the next subsection.

3.3. Phase 2: SMAS+Neighbourhood Exploration

Experimental results show that for complex problems with both discontinuous and (very) rugged landscapes, there is a high probability that the result is trapped in local optima near the global optimum when only using Phase 1 (Section 3.2). A neighbourhood exploration method is therefore proposed. Different from local search, the neighbourhood exploration aims to *jump out of* local optima, to introduce diversity which cannot be obtained by Phase 1 and also to directly improve the solution. Starting from a revisited x_{be}, neighbourhood exploration works as follows.

The Neighbourhood Exploration Method for Discrete Numerical Variables

(1): Generate a new candidate solution x_{ne} using the perturbation method below.

(2): If the number of maximum iterations TN is met, go to Step 5; otherwise go to Step 3.

(3): Construct a surrogate model using the nearest $s_2 \times d$ points to x_{ne}. Estimate the value of x_{ne} using the GP model with LCB prescreening.

(4): If improvement based on prescreening is shown compared to x_{be}, evaluate x_{ne} using an exact evaluation. Update the database D. If $f(x_{ne})$ is better than $f(x_{be})$, replace x_{be} by x_{ne} and go back to Step 2.

(5): Generate an opposite point of the (final updated) x_{be} in the neighbourhood and evaluate it using an exact evaluation. Update the database D.

The Perturbation Method

(1): Generate the number of units of perturbation by ceil(Normrnd(0,1)).

(2): Generate the percentage of selection for perturbation for each variable $x_1, \ldots, x_d$, the sum of which is 100%. x_i with the same value in P have two times the rate of selection than those with different values in P.

(3): Select a single variable x_i to be perturbed by roulette wheel selection with the above percentage.

(4): Perturb x_i by the number of units from Step 1 in a random direction.

Here the function $ceil(Normrnd(0, 1))$ means to get the nearest integer larger than $Normrnd(0, 1)$. Hence, in about 95% rate, one or two units will be used. In Step 2, if the x_i have the same value in P, x_i cannot be changed using DE mutation (4); we therefore assign a higher rate of perturbation to it.

Method to generate an opposite point

(1): Sort the database D in ascending order (considering a minimization problem). Save the first $D_{j,i} \neq x_i, j = 1, \ldots, n_D, i = 1, \ldots, d$ into DF_i.

(2): $LB_i = min(x_i, DF_i)$, $UB_i = max(x_i, DF_i)$, $i = 1, \ldots, d$.

(3): The neighbourhood opposite point $xo_i = LB_i + UB_i - x_i$, $i = 1, \ldots, d$.

where n_D is the number of points in D. Some traces of opposite DE [19] can be observed. But rather than using the original space, we define a subregion using currently being visited good solutions as the search space and explore the opposite side of it for diversity enhancement. Our pilot experiments show that this method performs especially well for problems with rugged landscapes.

4. Experimental Studies

4.1. Parameter Settings

Most of the parameters in SMDN are inherited from (or similar to) SAEAs based on SMAS with DE search and ISS. We set $\alpha = 5 \times d$, $\lambda = 5 \times d$, $F = 0.8$, $\omega = 2$, $s_1 = 0.5$ and $s_2 = 5$ in our experiments. They are not sensitive when following the setting rules, which are verified by [16], [20]. More details of the reasons behind the parameter setting rules are provided in [16], [20]. The newly added parameters are CRm, L, TN and TH. To avoid a wrong judgement of

Table 1. TEST INSTANCE F1-F9 USED IN THE EXPERIMENTAL STUDIES

Problem	Function	Range	Unit	Global Opt.	Property
F1	Problem 8.7 [21]	$[-100, 100]^4$	1	0	multimodal
F2	Problem 8.3 [21]	$[-100, 100]^5$	1	-737	rugged
F3	nvs09 [22]	$[3, 9]^{10}$	1	-43.13	unimodal
F4	Rastrigin [23]	$[-30, 30]^{10}$	0.5	0	very rugged
F5	Ellipsoid [23]	$[-30, 30]^{15}$	1	0	unimodal
F6	Rosenbrock [23]	$[-30, 30]^{15}$	1	0	narrow valley, multimodal
F7	Step [23]	$[-30, 30]^{20}$	1	0	unimodal
F8	Ackley [23]	$[-30, 30]^{20}$	0.5	0	rugged
F9	Griewank [23]	$[-600, 600]^{20}$	1	0	rugged

Table 2. STATISTICS OF THE BEST FUNCTION VALUES OBTAINED BY PURE SMAS FOR F1-F9

Problem	best	worst	average	median	std	sr
F1	0	6	0.3	0	1.34	95%
F2	-737	-727	-733.35	-732	3.62	45%
F3	-43.13	-43.13	-43.13	-43.13	0	100%
F4	1	15	8.30	8	5.12	0
F5	0	0	0	0	0	100%
F6	0	320	84.53	14	99.44	10%
F7	0	1	0.05	0	0.22	95%
F8	0	0	0	0	0	100%
F9	0.8	0.99	0.96	0.96	0.04	0

Phase 1 or Phase 2 when the best solution found so far does not improve, TH may not be small, but it is not necessary to make it very large. For problems with less than 10 variables, we set $TH = 80$, while for problems around 20 variables (or more), we set $TH = 150$. To balance the effectiveness of neighbourhood exploration and the possible number of exact evaluations, an empirical setting is $TN = 50$. CRm and L are related to the self-adaptive crossover rate generation. L is not sensitive but cannot be too large or too small: We set L to 50. Experiments [11], [16] show that $CR = 0.8$ often works well for many unimodal and multimodal problems using SMAS, but may not work well for some problems with rugged landscapes. Hence, we set initial CRm to 0.8. The above settings are used to **all** the test problems of various kinds to verify the robustness of SMDN.

4.2. Test Problems

Nine benchmark problems from [21], [22], [23] (see Table 1) and a real-world network-on-chip (NoC) design optimization problem are used to test SMDN. All these problems have continuous objective functions but with discrete numerical variables, which are the targeted EDOD problem. The experiments are run on an Intel core i7 3.0GHz computer in the MATLAB environment under the Linux system. It can be seen that besides discontinuous landscapes, the problems themselves include unimodal / multimodal (only with a few local optima) / (very) rugged (numerous local optima) landscapes with dimensions from 4 to 20. We use these problems to mimic real-world design optimization problems with different complexity. Note that the 15-dimensional Rosenbrock function has a very narrow valley with a local optimum located in it. Although the general landscape of the continuous Griewank function is smoother when the dimensionality is higher, the discontinuous landscape adds additional difficulty. For F1-F3, F5, F7 and F9, we set 1000 evaluations. For F4, F6 and F8, we set 2000 evaluations. In all the experiments, they converge (reach the global optimum or get stuck in a local optimum) far before the assigned maximum number of exact evaluations, except F9, for which the objective function value is improving in every 100 exact evaluations. 20 runs are carried out for F1-F9.

4.3. SMDN Performance and Comparisons

An SMAS-based SAEA is tested first to show the performance of SMAS for EDOD with numerical discrete variables. Only DE/current-to-best/1 mutation and ISS are used besides the basic SMAS from [11]. It can be considered as a part of Phase 1 of SMDN. The results are shown in Table 2, where sr refers to the success rate of reaching the global optimum. It can be seen that when only SMAS is applied: (1) for unimodal problems, the performance often has 100% success, (2) for multimodal problems without rugged landscapes, the performance is good, (3) for problems with (very) rugged surfaces, the success rate of SMAS is often low.

We then investigate the efficiency of the above SAEA using a very limited computing budget. We use three unconstrained mixed-integer global optimization problems from [5] (6 problems are used in [5] and a few of them are similar. 3 of them with different types are used for comparison in this paper) and the comparison is shown in Table 3. It is easy to extend the above SAEA for solving mixed-integer optimization by not rounding the continuous variables to the nearest integers in surrogate modeling. The average values are compared for both methods. It can be seen that when using a very restricted computing budget, the efficiency of SMAS is comparable to SO-MI [5], which is one of the most efficient methods designed for very expensive mixed-integer optimization.

The performance of SMDN is shown in Table 4. Compared to Table 2, a substantial improvement in terms of solution quality can be observed for multimodal problems and especially for every problem with (very) rugged landscapes. For example, for F6, the 10% success rate of pure SMAS increases to 95% when SMDN is used. The improvements verify the effectiveness of the two phase approach and the neighbourhood exploration phase. The efficiency of SMDN is equal to or better than pure SMAS. This is straightforward because: (1) for problems which can be well

Table 3. COMPARISON OF SMAS AND SO-MI USING T10,T11,T13 FROM [5]

Method-Problem	100 eval.	200 eval.	300 eval.	500 eval.
SMAS-T10	-491.20	-527.40	-528.09	-528.09
SOMI-T10	-386.33	-420.91	-432.59	N.A.
SMAS-T11	-35.08	-42.57	-42.93	-43.07
SOMI-T11	-42.92	-42.99	-42.99	N.A.
SMAS-T13	-4.42	-7.76	-9.61	-11.37
SOMI-T13	-4.89	-8.48	-9.63	N.A.

Table 4. STATISTICS OF THE BEST FUNCTION VALUES OBTAINED BY SMDN FOR F1-F9

Problem	best	worst	average	median	std	sr
F1	0	0	0	0	0	100%
F2	-737	-737	-737	-737	0	100%
F3	-43.13	-43.13	-43.13	-43.13	0	100%
F4	0	4	1.25	1	1.16	30%
F5	0	0	0	0	0	100%
F6	0	4	0.2	0	0.89	95%
F7	0	0	0	0	0	100%
F8	0	0	0	0	0	100%
F9	0.76	0.97	0.87	0.85	0.08	0

handled by SMAS, Phase 1 of SMDN has the same or better performance in terms of efficiency for problems which are sensitive to the CR value, (2) for problems for which SMAS gets stuck in local optima (i.e., the result is difficult to be improved even when more exact evaluations are used), Phase 2 provides significant help.

To verify the efficiency of SMDN, SMDN is compared with standard DE [17] with the same mutation strategy and common parameters. 30 runs are carried out for standard DE. The number of exact evaluations is 100,000. The results are shown in Table 5. It can be observed that the success rate of SMDN and standard DE are comparable except for F6 (where SMDN is better) and for F9 (where standard DE is better).

The median of both SMDN and standard DE are used. The results are shown in Table 6. $SMDN_M$ is the median of the converged values by SMDN. $SMDN_{FE}$ exact evaluations are used to reach $SMDN_M$, which are less than the assigned maximum number of exact evaluations. DE_{FE} is the number of exact evaluations used to obtain

Table 5. STATISTICS OF THE BEST FUNCTION VALUES OBTAINED BY STANDARD DE FOR F1-F9

Problem	best	worst	average	median	std	sr
F1	0	0	0	0	0	100%
F2	-737	-732	-736.75	-737	1.12	95%
F3	-43.13	-43.13	-43.13	-43.13	0	100%
F4	0	4	0.9	1	1.07	45%
F5	0	0	0	0	0	100%
F6	0	201	34.65	9	56.32	40%
F7	0	0	0	0	0	100%
F8	0	0	0	0	0	100%
F9	0	0.2	0.05	0.01	0.06	50%

Table 6. COMPARISON BETWEEN SMDN AND STANDARD DE

Problem	$SMDN_M$	$SMDN_{FE}$	DE_{FE}	speedup
F1	0	270	860	3.2
F2	-737	489	1675	3.4
F3	-43.13	129	1700	13.2
F4	1	1830	173700	94.9
F5	0	478	9600	20.1
F6	0	1213	N.A.	Inf
F7	0	762	19800	26.0
F8	0	1105	28200	25.5
F9	0.86	998	33500	33.6

$SMDN_M$ by standard DE. For F6, the median of standard DE is 9, which is much worse than that of SMDN. For other problems, it can be seen that SMDN consumes about 1% to 30% of the number of exact evaluations of standard DE to get comparable high quality results. Note that the speed enhancement increases with increasing dimensionality and complexity of the function landscape. This shows the ability of SMDN to obtain highly optimized results using a limited computing budget, even for very complex problems.

4.4. Network on Chip (NoC) Design Optimization

Due to the dramatic increase of integrated intellectual property (IP) cores in System-on-Chip (SoC), Network-on-Chip (NoC), serving as the underlying communication structure, is attracting more and more attention in recent years. NoC consists of a network constructed of multiple point-to-point data channels (links) interconnected by routers. The routers are connected to a set of distributed IPs. Energy consumption and delay are the main design criteria of an NoC. To improve the design quality, the number of virtual surface wave channels and the number of global SWI arbiter grant period as well as all the locations of the master nodes should be optimized. To obtain the performance of a candidate design, computationally expensive simulation is often necessary and explicit analytical formulations are not available. NDPAD [24] based on SMAS has been proposed for NoC design optimization and achieves good results. In this paper, a 8×8 NoC is selected as an example. The problem is defined as (6):

$$\begin{aligned} &\text{minimize } E(N_c, S_p, X_1, Y_1, \ldots, X_5, Y_5) \\ &\text{s.t. } AD(N_c, S_p, X_1, Y_1, \ldots, X_5, Y_5) \leq D_{MAX} \end{aligned} \tag{6}$$

where E is the energy consumption, AD is the average delay, $N_c \in [1, 16]$ is the number of virtual surface wave channels, $S_p \in [1, 12]$ is the number of global SWI arbiter grant period and $(X_i, Y_i) \in [1, 8]^2$ are the locations of the master nodes, which are not overlapping. The calculation of E and AD is based on SystemC software simulations. All the 12 design variables are integers. The designer's empirical suggestion is that $D_{MAX} \in [21, 21.5] cycles$ to get a good performance and the smaller the better in this range as long as a feasible solution can be obtained.

NDPAD, SMDN and tournament selection-based DE (SBDE) are compared. Following the same method as in

[24], SMDN is revised to handle constrained optimization problems by using a revised tournament selection-based method to rank the generated candidate solutions considering their level of constraint satisfaction. SBDE uses a constraint satisfaction considering tournament selection method [25] to replace the selection operator in DE, which is widely used in many electronic design optimization problems [3]. The parameter setting of NDPAD and SBDE follows [24]. The maximum running time of SMDN, NDPAD and SBDE is set to 15 hours, 48 hours and 3 weeks, respectively.

When using $D_{MAX} = 21.5cycles$, NDPAD and SMDN get very similar results both in terms of solution quality and efficiency. When using $D_{MAX} = 21.3cycles$, NDPAD cannot obtain feasible design solutions when the diversity of P is very low over three runs. Hence, the ability of NDPAD to get a highly optimized design near the border of constraints is questioned, which is needed for high-performance design optimization. The median design solution optimized by SMDN over three runs has a performance of $E = 83.17mJ$ and $AD = 21.22cycles$ using about 13 hours of CPU time. The three results are very similar. SBDE obtains $E = 83.13mJ$ and $AD = 21.30cycles$ (the first solution that is better than the median of the SMDN results) using 12 days while the final solution over three weeks' time is $E = 83.13mJ$ and $AD = 21.24cycles$.

5. Conclusions

In this paper, the SMDN method for expensive design optimization problems with discrete numerical variables has been presented. According to the experiments, SMDN simultaneously achieves the following goals: (1) a comparable efficiency as state-of-the-art very efficient SBNOAs, (2) the ability to obtain a comparable solution quality as standard DE even for complex problems, (3) it only requires 1% to 30% of the number of exact evaluations compared to standard DE and (4) it is easy to extend to more general problems and is compatible with existing successful techniques. Hence, SMDN combines the advantages of SBNOAs and standard EA-based SAEAs. The performance of SMDN is achieved by the key ideas of using the proposed two-phase search, where Phase 1 uses an improved SMAS for EDOD problem to find the neighbourhood of the global optimum and Phase 2 uses a new neighbourhood exploration method for discrete variables and collaborates with SMAS for further improvement of the solution towards global optimum. Future works include more applications in the ICT area and developing methods for discrete expensive optimization with multiple objectives and complex constraints.

References

[1] R. Li, M. T. Emmerich, J. Eggermont, E. G. Bovenkamp, T. Back, J. Dijkstra, J. Reiber, Metamodel-assisted mixed integer evolution strategies and their application to intravascular ultrasound image analysis, in: IEEE Congress on Evolutionary Computation, IEEE, 2008, pp. 2764–2771.

[2] M. Zaefferer, J. Stork, T. Bartz-Beielstein, Distance measures for permutations in combinatorial efficient global optimization, in: Parallel Problem Solving from Nature–PPSN XIII, Springer, 2014, pp. 373–383.

[3] B. Liu, G. Gielen, F. Fernández, Automated design of analog and high-frequency circuits: A computational intelligence approach, Springer, 2013.

[4] M. Barros, M. F. Barros, J. Guilherme, N. Horta, Analog circuits and systems optimization based on evolutionary computation techniques, Springer, 2010.

[5] J. Müller, C. A. Shoemaker, R. Piché, So-mi: A surrogate model algorithm for computationally expensive nonlinear mixed-integer black-box global optimization problems, Computers & Operations Research 40 (5) (2013) 1383–1400.

[6] L. Zhuang, K. Tang, Y. Jin, Metamodel assisted mixed-integer evolution strategies based on kendall rank correlation coefficient, in: Intelligent Data Engineering and Automated Learning, Springer, 2013, pp. 366–375.

[7] M. Holena, D. Linke, L. Bajer, Surrogate modeling in the evolutionary optimization of catalytic materials, in: the 14th annual conference on Genetic and evolutionary computation, ACM, 2012, pp. 1095–1102.

[8] K. Rashid, S. Ambani, E. Cetinkaya, An adaptive multiquadric radial basis function method for expensive black-box mixed-integer nonlinear constrained optimization, Engineering Optimization 45 (2) (2013) 185–206.

[9] K. Holmström, N.-H. Quttineh, M. M. Edvall, An adaptive radial basis algorithm (arbf) for expensive black-box mixed-integer constrained global optimization, Optimization and Engineering 9 (4) (2008) 311–339.

[10] L. Bajer, M. Holena, Rbf-based surrogate model for evolutionary optimization, in: ITAT, 2012, pp. 3–8.

[11] B. Liu, Q. Zhang, G. Gielen, A gaussian process surrogate model assisted evolutionary algorith for medium scale expensive optimization problems, IEEE Transactions on Evolutionary Computation 18 (2) (2014) 180–192.

[12] M. N. Le, Y. S. Ong, S. Menzel, Y. Jin, B. Sendhoff, Evolution by adapting surrogates, Evolutionary computation 21 (2) (2013) 313–340.

[13] T. J. Santner, B. J. Williams, W. I. Notz, The design and analysis of computer experiments, Springer, 2003.

[14] B. Liu, Q. Zhang, F. V. Fernández, G. Gielen, Self-adaptive lower confidence bound: A new general and effective prescreening method for gaussian process surrogate model assisted evolutionary algorithms, in: IEEE Congress on Evolutionary Computation (CEC), IEEE, 2012, pp. 1–6.

[15] M. Emmerich, K. Giannakoglou, B. Naujoks, Single-and multiobjective evolutionary optimization assisted by Gaussian random field metamodels, IEEE Transactions on Evolutionary Computation 10 (4) (2006) 421–439.

[16] B. Liu, Q. Chen, Q. Zhang, G. Gielen, V. Grout, Behavioral study of the surrogate model-aware evolutionary search framework, in: IEEE Congress on Evolutionary Computation, IEEE, 2014, pp. 715–722.

[17] K. Price, R. Storn, J. Lampinen, Differential evolution: a practical approach to global optimization, Springer-Verlag New York Inc, 2005.

[18] A. K. Qin, V. L. Huang, P. N. Suganthan, Differential evolution algorithm with strategy adaptation for global numerical optimization, IEEE Transactions on Evolutionary Computation 13 (2) (2009) 398–417.

[19] S. Rahnamayan, H. R. Tizhoosh, M. M. Salama, Opposition-based differential evolution, IEEE Transactions on Evolutionary Computation 12 (1) (2008) 64–79.

[20] B. Liu, D. Zhao, P. Reynaert, G. G. Gielen, Gaspad: A general and efficient mm-wave integrated circuit synthesis method based on surrogate model assisted evolutionary algorithm, IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems 33 (2) (2014) 169–182.

[21] K. E. Parsopoulos, M. N. Vrahatis, Recent approaches to global optimization problems through particle swarm optimization, Natural computing 1 (2-3) (2002) 235–306.

[22] M. R. Bussieck, A. S. Drud, A. Meeraus, Minlplib - a collection of test models for mixed-integer nonlinear programming, INFORMS Journal on Computing 15 (1) (2003) 114–119.

[23] M. Jamil, X.-S. Yang, A literature survey of benchmark functions for global optimisation problems, International Journal of Mathematical Modelling and Numerical Optimisation 4 (2) (2013) 150–194.

[24] M. Wu, A. Karkar, B. Liu, A. Yakovlev, G. Gielen, V. Grout, Network on chip optimization based on surrogate model assisted evolutionary algorithms, in: IEEE Congress on Evolutionary Computation, IEEE, 2014, pp. 3266–3271.

[25] K. Deb, An efficient constraint handling method for genetic algorithms, Computer methods in applied mechanics and engineering 186 (2) (2000) 311–338.