A New Evolutionary Algorithm with Structure Mutation for the Maximum Balanced Biclique Problem

Bo Yuan, Bin Li, *Member, IEEE*, Huanhuan Chen, *Member, IEEE*, and Xin Yao, *Fellow, IEEE*

***Abstract*—The maximum balanced biclique problem (MBBP), an NP-hard combinatorial optimization problem, has been attracting more attention in recent years. Existing node-deletion-based algorithms usually fail to find high-quality solutions due to their easy stagnation in local optima, especially when the scale of the problem grows large. In this paper, a new algorithm for the MBBP, evolutionary algorithm with structure mutation (EA/SM), is proposed. In the EA/SM framework, local search complemented with a repair-assisted restart process is adopted. A new mutation operator, SM, is proposed to enhance the exploration during the local search process. The SM can change the structure of solutions dynamically while keeping their size (fitness) and the feasibility unchanged. It implements a kind of large mutation in the structure space of MBBP to help the algorithm escape from local optima. An MBBP-specific local search operator is designed to improve the quality of solutions efficiently; besides, a new repair-assisted restart process is introduced, in which the Marchiori's heuristic repair is modified to repair every new solution reinitialized by an estimation of distribution algorithm (EDA)-like process. The proposed algorithm is evaluated on a large set of benchmark graphs with various scales and densities. Experimental results show that: 1) EA/SM produces significantly better results than the state-of-the-art heuristic algorithms; 2) it also outperforms a repair-based EDA and a repair-based genetic algorithm on all benchmark graphs; and 3) the advantages of EA/SM are mainly due to the introduction of the new SM operator and the new repair-assisted restart process.**

***Index Terms*—Estimation of distribution algorithm, evolutionary algorithms, local search, maximum balanced biclique problem (MBBP), structure mutation.**

I. Introduction

A BIPARTITE graph $G(U, V, E)$ is a graph whose nodes can be divided into two disjoint sets U and V such that every edge connects one node in U to one in V, and E denotes the edges of the graph. A balanced bipartite graph $G(U, V, E)$ is a bipartite graph in which the both disjoint node sets are of the same cardinality, having $|U| = |V| = n$. A biclique $B(U^b, V^b, E^b)$ is a subgraph of G that for each pair of nodes $u \in U^b$ and $v \in V^b$, having $(u, v) \in E^b$. If $|U^b| = |V^b| = k$, we call B a balanced biclique of size k. The maximum balanced biclique problem (MBBP) is defined as finding a balanced biclique with the largest size k in a balanced bipartite graph, that is the cardinality of U^b or V^b ($|U^b| = |V^b|$) should be maximized.

The MBBP (also referred to as the maximum balanced complete bipartite subgraph) was stated to be NP-hard in [1] (problem GT24), while the other nice NP-hardness proofs for this problem were presented in [2] and [3]. It is one of the few problems for which we still have neither a hardness of approximation result, nor a good approximation algorithm. A theoretical estimation of the boundary of k was given in [4], that is $k(n) \leq k \leq 2k(n)$ with high probability (for sufficiently large n), where $k(n) = \log n/\log(1/p)$, $0 < p < 1$ is the probability that a particular edge exists in G. Recently, the MBBP has been proven to be even hard to approximate within a factor of 2^{θ}, $\theta = (\log n)^{\delta}$ for some $\delta > 0$ [5]. So it seems to be impossible to have an exact polynomial time algorithm for the MBBP, and approximation of large biclique is also hard.

Meanwhile, the MBBP is closely related to many important real-life applications, such as biclustering of gene expression data in computational biology [6], PLA-folding in VLSI theory [7], and newly emerged applications in nanoelectronic system design [8]–[11], where the MBBP problem is used to model a promising defect-tolerant design flow that extracts the maximum defect-free subcrossbar (balanced biclique) within the original defective nano-crossbar (balanced bipartite graph).

Although the MBBP has been attracting more and more attention in recent years, few effective algorithms have been presented for it due to its NP-hardness. Instead of finding the maximum balanced biclique in G, the problem can be

Manuscript received January 17, 2014; revised May 13, 2014 and July 23, 2014; accepted July 24, 2014. Date of publication August 14, 2014; date of current version April 13, 2015. This work was supported in part by the National Natural Science Foundation of China under Grant 61071024, Grant 61331015, Grant 61329302, Grant 61175065, Grant 61203292, and Grant 61311130140, in part by the European Union 7th Framework Program under Grant 247619 and Grant 257906, in part by the National Natural Science Foundation of Anhui Province under Grant 1108085J16, and in part by EPSRC under Grant EP/I010297/1. The work of X. Yao was supported by a Royal Society Wolfson Research Merit Award. The work of H. Chen was supported by the One Thousand Young Talents Program. This paper was recommended by Associate Editor K. C. Tan.

B. Yuan is with the USTC-Birmingham Joint Research Institute in Intelligent Computation and Its Applications, University of Science and Technology of China, Hefei 230027, China (e-mail: yuanbo@ustc.edu.cn).

H. Chen and X. Yao are with the USTC-Birmingham Joint Research Institute in Intelligent Computation and Its Applications, University of Science and Technology of China, Hefei 230027, China, and also with the Centre of Excellence for Research in Computational Intelligence and Applications, School of Computer Science, University of Birmingham, Birmingham B15 2TT, U.K. (e-mail: hchen@ustc.edu.cn; x.yao@cs.bham.ac.uk).

B. Li is with the USTC-Birmingham Joint Research Institute in Intelligent Computation and Its Applications, University of Science and Technology of China, Hefei 230027, China, and also with the CAS Key Laboratory of Technology in Geo-spatial Information Processing and Application System, University of Science and Technology of China, Hefei 230027, China (e-mail: binli@ustc.edu.cn).

Digital Object Identifier 10.1109/TCYB.2014.2343966

transformed into a decision (yes or no) problem: does G have a biclique of size $k \times k$? Since it has been proven that it would also take exponential time for an exact algorithm [11] to solve such decision problem, the time for graphs of size $n = 64$ or larger would be completely intractable. If the constraint of being balanced was removed, the MBBP problem would be solvable in polynomial time [12]. However, the results obtained in this way were poor [9], because most of the bicliques generated by the polynomial time algorithm without the balance constraint were highly unbalanced. Given a bipartite graph with $2n$ nodes and m edges, the size of the largest (maximum) balanced biclique that is theoretically guaranteed to exist is $\log 2n/\log(8en^2/m)$ [13]. An efficient algorithm was proposed in [13] to approach such a theoretical size. It can only find a subgraph of logarithmic size even in a dense graph, e.g., the graph models of crossbar nano-architectures [8].

Recently, a class of node-deletion-based heuristic algorithms get a lot of attention [8], [9], [14], [15], which convert the MBBP into the maximum balanced independent set problem in the complement graph, which is usually sparse because the original bipartite subgraph is dense, e.g., the graph models of crossbar nano-architectures. To guarantee the outputted biclique to be almost balanced ($-1 \leq |U^b| - |V^b| \leq 1$), the algorithms remove the nodes alternately from U and V. The heuristic in [8] was to remove nodes with the maximum degree. The heuristic in [9] (the most effective algorithm so far) was to remove the node in one partition that was adjacent to the maximum number of minimum degree nodes in the other partition. Therefore in brief, to achieve the maximum balanced biclique, the heuristic in [8] tried to reduce the number of edges in the graph, while the heuristic in [9] tried to reduce the degree of the least degree nodes in a partition. The heuristics in [14] and [15] tried to improve the performance of the node-deletion-based algorithms by combining the two heuristics in [8] and [9], which could obtain results as good as that of [9] while with much lower time complexity. The heuristic in [14] was to remove the node with the maximum degree in one partition that was adjacent to the given minimum degree node in the other partition, and it was speeded up significantly in [15] by considerably reducing the number of major loops. So far, no node-deletion-based heuristic algorithm is effective enough for the MBBP, new techniques have to be designed and investigated.

It has been shown by many works that the hybrids of evolutionary algorithms (EAs) and problem-specific local search operators are effective methods for combinatorial optimization problems [16]. Most of the previous hybrid EAs adopted the paradigm in which EAs played the role of the global search, while local search operators were introduced to enhance the efficiency of the hybrid algorithms. In [17], a different paradigm was proposed with the algorithm R-EVO, in which fine-grained local search strategy was complemented by an EA (EDA). The difference lies in that the local search operator provides the most contribution to the quality improvement of solutions, while the EDA plays a complementary role that reinitializes the solutions when the local search process stagnates. Experimental results on the maximum clique problems (MCPs) [18], another NP-hard problem, showed that such paradigm could produce significantly better solutions than the conventional EAs [19], [20]. For other complex combinatorial optimization problems with more constraints, e.g., the MBBP, the effectiveness of the paradigm still need to be improved to meet the harsh challenges brought by real-world applications.

By going through the procedure of R-EVO [17], it can be seen that the most computation resource is consumed by the local search process, which is the main force for convergence. No technique is considered to keep, to some degree, the algorithm's capability of exploration during the search process. The exploration is performed via a reinitialization process only when the local search process is stagnated. We try to improve the performance of the algorithm by introducing better exploration techniques to complement the local search process. In more details, a problem-specific mutation operator, structure mutation (SM), is proposed to enhance the exploration of the algorithm during the search process. Besides, a new local search operator utilizing the domain knowledge is presented to accelerate the quality improvement of solutions, and a repair-assisted restart process is implemented to utilize the good knowledge gained from previous local search process. Consequently, an effective evolutionary algorithm with structure mutation (EA/SM) for the MBBP is proposed.

The main contributions of this paper can be summarized as follows.

1) *Fast Local Search:* The fast local search operator considers the given current solution (biclique) as a start point and modifies the solution step-by-step through alternating between expansion and corrosion on the biclique. Specifically, the local search operator adds/drops one node on/from the current solution while keeping the solution feasible.
2) *Structure Mutation:* Solutions got by the fast local search will be reprocessed by the SM, which changes the structure of the current biclique dynamically while keeping the size (fitness value) and feasibility of the biclique unchanged. The SM is expected to implement a kind of large mutation in the solution space, which increases the possibility of finding a new search area for the following local search process, therefore enhances the capability of EA/SM to jump out of local optima.
3) *Repair-Assisted Restart:*
    a) *Probability-Model-Based Sampling:* Restart process will be invoked only when the local search process cannot improve the quality of the current solution for a number of iterations. To obtain good efficiency, each new solution is reinitialized by sampling the search space according to a probability model P updated with better solutions in the previous local search process.
    b) *Modified Marchiori's Heuristic Repair:* Since the reinitialized solution is prone to be infeasible, to repair it effectively, Marchiori's heuristic repair operator [19] (proposed for the MCP [18]) is modified and applied on every reinitialized solution to generate a feasible solution.

Algorithm 1 EA/SM (G)

$G(U, V, E)$	Bipartite graph
$B(U^b, V^b)$	Current biclique
P	Probability model
N	Population size
A	Restart parameter
t	Iteration counter
t_b	Time of last improvement
t_r	Time of last restart
k	Size of balanced biclique
$kbest$	Best size obtained

```
$B_i \leftarrow zeros(1, 2n)$; $tb_i \leftarrow 0$; $tr_i \leftarrow 0$; $kbest_i \leftarrow 0$; $i = 1, 2, \dots, N$
$p_i \leftarrow 0.5$; $i = 1, 2, \dots, 2n$
$t \leftarrow 0$;
**Repeat**
    $t \leftarrow t+1$;
    $i = 1: N$
        $B_i \leftarrow$ **Fast Local Search** ($B_i$);
        $B_i \leftarrow$ **Structure Mutation** ($B_i$);
        **If** $k_i > kbest_i$
            $kbest_i = k_i$; $tb_i = t$;
        **end**
        **If** $t$-max$\{tb_i, tr_i\} > A$ // Restart
            $B_i \leftarrow$ Sampling ($P$);
            **If** $B_i$ is infeasible
                $B_i \leftarrow$ **Heuristic Repair** ($B_i$);
            **end**
            $tr_i = t$;
        **end**
    **end**
    **Update** $P$ according to (5);
**until** maximum iteration reached
```

The rest of this paper is organized as follows. The problem formulation is given in detail in Section II. Section III introduces the framework of EA/SM for the MBBP and its main components. Experimental studies and comparisons are given in Section IV. Section V concludes the paper.

II. Problem Formulation

Refer to the description of the MBBP in Section I, we advance formulized definition of the MBBP for the first time. Given an balanced bipartite graph $G(U, V, E)$ as shown in Fig. 1, a subgraph $B(U^b, V^b, E^b)$ of G (the dark dots) can be represented by a $2n$-D 0-1 vector, $B = \{b_1, b_2, \dots, b_n; b_{n+1}, b_{n+2}, \dots, b_{2n}\} \in \{0, 1\}^{2n}$, where $b_i = 1$ if and only if node i is in B, $b_i = 0$ otherwise, the first n values represent the nodes in U, while the last n values represent the nodes in V. Then, the MBBP can be defined as follows:

$$\max : k(B) = \min\left(\mid U^b \mid, \mid V^b \mid\right) \tag{1}$$

$$\text{s.t.: } \mid E^b \mid = \mid U^b \mid \cdot \mid V^b \mid \tag{2}$$

$$\text{where } \mid U^b \mid = \Sigma b_i, 1 \le i \le n \quad \text{and} \quad \mid V^b \mid = \Sigma b_j, n+1 \le j \le 2n. \tag{3}$$

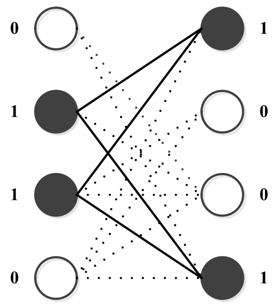


Fig. 1. Balanced bipartite graph $G(U, V, E)$ and its subgraph $B(U^b, V^b, E^b)$, where $B = \{0, 1, 1, 0; 1, 0, 0, 1\}$.

The object function (1) is to maximize the smaller cardinalities of the two node set (U^b and V^b). The constraint (2) is used to ensure that the subgraph B to be a biclique (or a complete bipartite subgraph), that is, for every $u \in U^b$ and $v \in V^b$, the edge $(u, v) \in E^b$. It seems that a balanced biclique could possibly be obtained without considering balance constraint. But including the balance constraint in the quality evaluation of solutions definitely may help to improve the performance of the EAs. In order to obtain better performance, all the effective algorithms for the MBBP generally keep the constraint of being balanced at the same time

$$\text{s.t.: } -1 \le \mid U^b \mid - \mid V^b \le 1. \tag{4}$$

The constraint (4) is used to ensure that the subgraph B is almost balanced. In this paper, both constraints (2) and (4) are considered simultaneously.

III. EA/SM for MBBP

A. Representation and Fitness

Refer to the problem formulation at the beginning of Section II, a subgraph B of G can be encoded as a $2n$-D 0-1 string $B = \{b_1, b_2, \dots, b_{2n}\} \in \{0, 1\}^{2n}$. The fitness of B is defined as its size k [objective function (1)] if B represents an almost balanced biclique [(2) and (4) are satisfied simultaneously]. Since each initialized solution will be repaired, there is no need to define fitness values for infeasible solutions.

B. EA/SM Algorithm

The framework of EA/SM algorithm is similar to that of R-EVO [17], where a fine-grained local search strategy is complemented by an EDA-like restart process. R-EVO was originally designed for the MCP, so it cannot be applied directly on the MBBP. New effective MBBP-specific operators need to be designed to tackle the MBBP. Besides, by investigating the procedure of R-EVO, one can find that there is still much room for improvement. Examples are as follows.

1) In R-EVO, the exploration capability of the algorithm is not seriously considered enough. During the local search process, separated greedy local searches of individuals take all the computational resource. The global search is conducted only when the local search process stagnates and the algorithm enters the restart process. An EDA-like restart process is used to reinitialize the solution

Algorithm 2 Fast Local Search (B)

```
$G(U, V, E)$    Bipartite graph
$B(U^b, V^b)$   Current biclique
$UPA$           Set of nodes in $U$ connected to all nodes in $V^b$
$VPA$           Set of nodes in $V$ connected to all nodes in $U^b$

// Add a node
If $UPA \neq \phi$ and $VPA \neq \phi$
If $|U^b| < |V^b|$
pick $k \in UPA$; return $B \cup \{k\}$; end
If $|U^b| > |V^b|$
pick $k \in VPA$; return $B \cup \{k\}$; end
If $|U^b| = |V^b|$
pick $k \in UPA$ or $k \in VPA$; return $B \cup \{k\}$; end
end
If $UPA \neq \phi$ and $VPA = \phi$ and $|U^b| \leq |V^b|$
pick $k \in UPA$; return $B \cup \{k\}$; end
If $UPA = \phi$ and $VPA \neq \phi$ and $|U^b| \geq |V^b|$
pick $k \in VPA$; return $B \cup \{\mathrm{k}\}$; end
// or Drop a node
If $|U^b| \neq \phi$ and $|V^b| \neq \phi$
If $|U^b| > |V^b|$
pick $k \in U^b$; return $B/\{k\}$; end
If$|U^b| < |V^b|$
pick $k \in V^b$; return $B/\{k\}$; end
If $|U^b| = |V^b|$
pick $k \in U^b$ or $k \in V^b$; return $B/\{k\}$; end
end
If $|U^b| \neq \phi$ and $|V^b| = \phi$
pick $k \in U^b$; return $B/\{k\}$; end
If $|U^b| = \phi$ and $|V^b| \neq \phi$
pick $k \in V^b$; return $B/\{k\}$; end
```

according to a probability model. Since the probability model is trained with the better solutions obtained in the local search process, it is reasonable to expect that the probability model would converge quickly when the local search is effective. Therefore, good exploration ability of the algorithm could not be expected in the restart process.

2) In R-EVO, in the EDA-like restart process, each new solution is generated in a greedy and tough fashion, where candidate nodes at every step are selected not only with probability proportional to the probability model describing the promising area of solution space, but also checking the feasibility of the solution. That means if the feasibility checking of one node generated according the probability model is negative, the node has to be dropped to meet the feasibility constraint. This way of new solution reinitialization would limit the efficiency and effectiveness of the EDA-like process dramatically.

In EA/SM, a well-crafted structure mutation operator is proposed to accompany the local search operator to increase the exploration capability of the algorithm during the search process; a MBBP-specific fast local search operator is presented to accelerate the solution quality improvement (toward suboptima); a new repair-assisted restart process is proposed, in which, each new solution is firstly generated by sampling the search space according to the knowledge (probability model) gained from previous search process, then it is repaired in whole if it violates the constraints. A modified Marchiori's heuristic repair [19] is presented to repair the new reinitialized solutions to be feasible ones. The pseudo-code of EA/SM for MBBP is shown in Algorithm 1.

At the initialization stage, step 1–3, a pool of solutions B is initialized with all variables set to be 0, and the probability model P is initialized with all variables set to be 0.5, which expresses a uniform distribution of sampling probability in the solution space.

Each iteration consists of two processes: local search process and restart process. In the local search process, a fast local search is conducted on current solution B_i, which adds or drops nodes one by one from the current biclique represented by B_i. SM is conducted consequently, which changes the adjacency attributes of the nodes in the current biclique represented by B_i.

When the quality of B_i cannot be improved for A iterations (step 12), the restart process is invoked. First, new solution is sampled according to the probability model P, then, the modified Marchiori's heuristic repair operator is applied to the newly generated solution to guarantee the reinitialized one is feasible. At the end of each iteration, the probability model P is updated [according to the formula (5) on Section III-F].

C. Fast Local Search

The work on R-EVO [17] has shown that a simplification of the original heuristic local search complemented by a global search is effective on achieving state-of-the-art results of the MCP, where the reactive local search (RLS) heuristic [21], [22] is simplified. On the other hand, the heuristics which are excessively strong and frequent are often time-consuming, and may even weaken the stochastic nature of EAs.

With above understanding, a fast local search operator tailored for the MBBP is presented in EA/SM, which works in an alternative manner between expansion and corrosion phases. In each phase, the operator adds/drops one node to/from the current solution while ensuring its feasibility. The pseudo-code of the fast local search is presented in Algorithm 2. It can be seen in Algorithm 2 that the fast local search works via alternating between expansion and corrosion phases. It selects a node k from UPA or VPA randomly and adds it (node k) to B while keeping B almost balanced (expansion phase), and then return B. If no such node is available, it removes a node from U^b or V^b randomly while keeping B almost balanced (corrosion phase), and then return B. In order to keep the obtained biclique balanced, we add one node to the partition whose cardinality is less than that of the other partition, or drop one node from the partition whose cardinality is more than that of the other partition.

D. SM

It has been testified by quite a number of instances that introducing large mutation operators into EAs will effectively improve the performance of EAs on complex optimization problems. Such large mutation can generate new solutions far

Algorithm 3 Structure Mutation (B)

$G(U, V, E)$	Bipartite graph
$B(U^b, V^b)$	Current biclique
UPA	Set of nodes in U connected to all nodes in V^b
VPA	Set of nodes in V connected to all nodes in U^b

```
If $UPA \neq \phi$ and $VPA \neq \phi$
If $|U^b| \neq \phi$
pick $k \in UPA$; $B \leftarrow \{k\}$; pick $k \in U^b$; $B \leftarrow B/\{k\}$; end
Update $VPA$;
If $VPA \neq \phi$ and $|V^b| \neq \phi$
pick $k \in VPA$; $B \leftarrow \{k\}$; pick $k \in V^b$; $B \leftarrow B/\{k\}$; end
end
If $UPA \neq \phi$ and $VPA = \phi$ and $|U^b| \neq \phi$
pick $k \in UPA$; $B \leftarrow \{k\}$; pick $k \in U^b$; $B \leftarrow B/\{k\}$; end
If $UPA = \phi$ and $VPA \neq \phi$ and $|V^b| \neq \phi$
pick $k \in VPA$; $B \leftarrow \{k\}$; pick $k \in V^b$; $B \leftarrow B/\{k\}$; end
return $B$
```

away from the current one in the solution space with considerable probability. Successful examples include the three points swap-based mutation and four stock remove and insert-based mutation for cutting stock problems with and without contiguity [23], merge-split operator for capacitated arc routing problems [24], etc. It has been observed that EAs with large mutation operators are less likely to be trapped in local optima. Original idea of large mutation (referred to as extended neighborhoods) can be traced back to [25] and [26].

In this paper, a new mutation operator, named SM, is proposed to implement the idea of large mutation, it is expected to make large jump in the solution space of the MBBP to enhance the exploration of the algorithm during the search process. The proposed SM has three important characteristics: 1) it can adjust the search direction of the following local search process by changing UPA or/and VPA of the current solution, since the fast local search process starts with a random node selected from UPA or VPA (expansion phase in Algorithm 2); 2) it does not change the fitness value (size) of the solution, thus no extra fitness evaluation is needed after the mutation; and 3) compared to random mutation operators, the feasibility of the solution can be guaranteed [constraints (2) and (4) are satisfied simultaneously], thus no repair operation is needed after the mutation. With characteristics 2) and 3), the SM can be conducted with a very low computational cost.

The pseudo-code of the SM is presented in Algorithm 3. It is an exchange process that replaces the nodes in U^b by nodes from *UPA* and then replaces the nodes in V^b by nodes from VPA, where UPA represents the set of nodes in U connected to all nodes in V^b, while VPA represents the set of nodes in U connected to all nodes in V^b. It looks like that the SM operator can be executed on the current solution more than once, but in practice the cardinality of VPA or UPA is very limited, especially in the later phase of the search, thus, in EA/SM, each solution is manipulated by the SM only once per iteration.

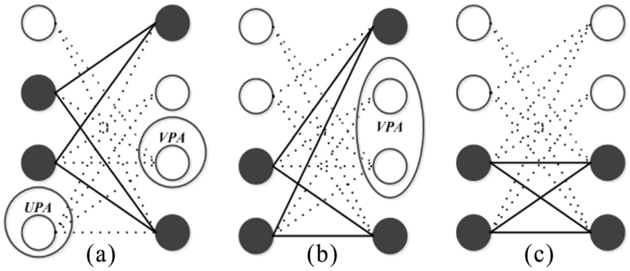


Fig. 2. Example to illustrate the structure mutation procedure. (a) Initial state. (b) Intermediate state. (c) Final state.

Fig. 2 is an example to illustrate the SM procedure for the MBBP, where the dark dots represent the current biclique [Fig. 2(a)]. First, a random node from U^b is replaced by a random node from UPA, then VPA is updated [Fig. 2(b)]; second, if VPA is not empty, a random node from V^b is replaced by a random node from VPA [Fig. 2(c)]. The size of the resulting biclique B [Fig. 2(c)] is unchanged, while its structure is different from the previous one dramatically [Fig. 2(a)]. The difference lies in that all nodes in B [Fig. 2(c)] have new adjacency attributes in G. The SM can increase the possibility of finding a new search area for the following local search process, therefore enhances the capability of EA/SM to jump out of local optima. Furthermore, the new promising search space will be memorized by the probability model for the following EDA-based restart process, which will also enhance the contribution of the restart process for global optimization.

E. Repair-Assisted Restart

One new solution will be generated when the restart process is invoked by a certain trigger condition. In R-EVO [17], the reinitialized solution is built in a greedy fashion, where candidate node at every step is selected firstly with probability proportional to model values in P. Then, the feasibility is checked. Only the node that can legally expand the solution will be accepted into the current solution under built, which causes the size of the reinitialized solution prone to be very small. It can be perceived that the greedy restart process does not take full advantage of the statistical knowledge learned from in previous search process.

In EA/SM, a new repair-assisted restart process is presented, where each new solution is reinitialized in whole by sampling the search space according to the probability model P with the statistical knowledge of possible promising area in solution space extracted during the previous search process. Then, the feasibility of the whole solution is checked. If the new solution violates the constraints, it is repaired by the modified Marchiori's heuristic repair operator (Section III-G) to guarantee the feasibility of the solution.

F. Probability Model for Restarts

Estimation of distribution algorithms (EDAs) [27]–[34] have been investigated in the framework of evolutionary computation for more than ten years. The feature of EDAs is to model promising areas in the search space in a probabilistic manner, then guide the exploration of the algorithm with the probabilistic model [35]. In this paper, the EDA-like technique is adopted to learn the knowledge/model of promising area

Algorithm 4 Heuristic Repair (B)

```
$G(U, V, E)$   Bipartite graph
$B(U^b, V^b)$  Current solution (likely not a biclique)
$\alpha$       A very small value $\alpha \in (0, 1)$

1:  // Extraction (to be a small almost balanced biclique)
2:  $C(U^c, V^c) \leftarrow B$;
3:  While $C \neq \phi$
4:    If $|U^b| < |V^b|$
5:      pick $k \in U^c$; $C \leftarrow C/\{k\}$; end
6:    If $|U^b| > |V^b|$
7:      pick $k \in V^c$; $C \leftarrow C/\{k\}$; end
8:    if $|U^b| = |V^b|$
9:      pick $k \in U^c$ or $k \in V^c$; $C \leftarrow C/\{k\}$; end
10:   // Flip a coin with head probability $\alpha$
11:   If the head turns up
12:     $B \leftarrow B/\{k\}$;
13:   else
14:     remove from $B$ and $C$ all the nodes in $B$ that are
        not connected to $k$; end
15: end // Extraction
16: // Extension (to be a larger almost balanced biclique)
17: $C(U^c, V^c) \leftarrow G/B$
18: While $C \neq \phi$
19:   If $|U^b| < |V^b|$
20:     pick $k \in U^c$; $C \leftarrow C/\{k\}$; end
21:   If $|U^b| > |V^b|$
22:     pick $k \in V^c$; $C \leftarrow C/\{k\}$; end
23:   If $|U^b| = |V^b|$
24:     pick $k \in U^c$ or $k \in V^c$; $C \leftarrow C/\{k\}$; end
25:   If $k$ is connected to all nodes in $B$
26:     $B \leftarrow B \cup \{k\}$; end
27: end // Extension
28: return $B$
```

during the local search process, and then reinitialize the searching population according to the knowledge/model.

The probability model adopted in this paper is a simple univariate model, $P = \{p_1, p_2, \ldots, p_{2n}\} \in [0, 1]$, each of whose variable represents the probability that the correspondent binary variable b in the solution B should take the value 0 or 1. At the beginning of the algorithm, the probability model is set to be uniform distribution, $p_i = 0.5$ for $i = 1, 2, \ldots, 2n$. That means each binary variable has the same probability of taking value 0 or 1. At the end of each generation t in EA/SM, the best M solutions (binary strings) are selected from the current population to update the probability model P. In order to reduce noise, only solutions providing balanced bicliques whose size is comparable (within a tolerance depth Δ) with the largest one are taken into account as suggested in [17]. Let the M selected solutions are $C^j = \{{c_1}^j, {c_2}^j, \ldots, {c_{2n}}^j\}$, $j = 1, 2, \ldots, M$. The probability model P is updated in the same way as that in the PBIL algorithm [27]

$$p_i = (1 - \lambda)p_i + \lambda \sum_{j=1}^{M} c_i^j / M. \quad (5)$$

For $i = 1, 2, \ldots, 2n, \lambda \in (0, 1]$ is the learning rate. The bigger λ is the more contribution of the M selected solutions to the updated probability model will be.

If the algorithm cannot improve the current biclique for A iterations, the proposed repair-assisted restart process is invoked. The same restart counter mechanism as RLS [21] is used. The reinitialized individual, $B = \{b_1, b_2, \ldots, b_{2n}\}$, $i = 1, 2, \ldots, 2n$, is generated one-by-one by sampling the search space according to the probability model P. Since the reinitialized individual may be infeasible, a repair operation is conducted consequently to check and repair the solution.

G. Modified Marchiori's Heuristic Repair

The Marchiori's heuristic repair operator [19] has been shown to be effective in EAs for the MCP, but it is not designed for the MBBP, and cannot be adopted directly in EA/SM. To utilize the merits of Marchiori's heuristic repair operator, it is modified for the MBBP in this paper. The pseudo-code of the modified Marchiori's heuristic repair is presented in Algorithm 4. It has two phases: phase 1 extracts the illegal solution to be a small almost balanced biclique (lines 1–15); phase 2 extends the small biclique obtained in phase 1 to be a larger one (lines 16–27).

The time complexity of the SM operator is of the same order of magnitude as that of the fast local search operator; the two operators only perform one or two node additions or removals. In contrast, the time complexity of the heuristic repair operator is pretty high; the operator requires a sequence of node additions and removals to repair an initialized solution. Since the restart process is invoked only when the quality of solutions cannot be improved in the local search process for A iterations, the interval of restarts should be at least A iterations and the total execution number of restart is very limited. Therefore, the heuristic repair operator is affordable in EA/SM. Experimental results in Section IV also show that the runtime consumed by the heuristic repair operator is very limited.

IV. Experimental Studies

In this section, the performance of EA/SM is experimentally investigated and compared with the state-of-the-art heuristic algorithm [9]. The effectiveness of the SM operator and the repair-based restart process is verified. The performance of EA/SM is also compared with those of two other repair-based EAs: a repair-based EDA and a repair-based GA.

A large number (30) of benchmark testing instances (graphs) are randomly generated as done in previous works [8]–[11], [14] and [15]. The benchmark graphs have different sizes, n = 250, 500, and different probability that a particular edge exists in G, p = 95%, 90%, 85% with uniform edge distribution (which is most common in nano-electronics). For ease of recording and comparison, the following naming rule is introduced, nA_pB_C, where A is the size of the graph, B is the probability that a particular edge exists in G and C is the sequence order in the benchmark set with the same attributes (both n and p). For example, graph n250_p90%_3 means that the graph is of size 250, the probability that a particular edge exists is 90% and it is the third graph in the benchmark set with the same attributes $n = 250$ and $p = 90\%$. (All the benchmark graphs used in the simulation in this paper and EA/SM source codes with supporting documents are available at: http://staff.ustc.edu.cn/∼yuanbo).

As suggested in [19], the parameter α in the repair operator should be very small, α value is set to be 0.001 as [20] recommended for the MCP. The other parameters of EA/SM are set as suggested in [17], [19], and [20]: population size $N = 10$; tolerance depth $\Delta = 2$; learning rate $\lambda = 0.7$; restart parameter $A = 10 \cdot kbest$. These values work well among other test values in our experiments. A rigorous theoretical analysis on how these parameters affect the results is very hard and out of the scope of this paper, but we can comment that: 1) in order to reduce the noise in the updated probability model, tolerance depth Δ shouldn't be too large; 2) the bigger λ is, the more contribution of the M selected solutions to the updated probability model will be, so the probability model would converge quickly; and 3) periodic restarts are needed to assure that the search is not confined in a limited portion of the search space, but frequent restarts are time-consuming and may even weaken the learning ability of the PBIL method.

The algorithm stops after 200 000 calls (maximum times of fitness evaluation $F_{\max} = 200\,000$) of the local search and SM operators when the algorithm is very likely to have converged to a good solution, and help us get sufficient data in the evolutionary process. All the experiments were performed on two quadruple nucleuses 2.33 GHz Intel Xeon processors platform with 12 G memory. However, all tested algorithms were implemented as monolithic processes and no CPU core parallelism has been exploited.

A. Comparisons With State-of-the-Art Heuristic Algorithm

The heuristic algorithms in [8], [9], [14], and [15] are three representative deterministic algorithms for the MBBP, whose performances have been testified successfully. Among these three algorithms, the algorithm in [9] is the most effective in terms of the quality of solutions. It consistently outperforms the algorithm in [8] on the quality of best solutions on all test instances, and such advantages are directly proportional to the crossbar sizes n. The algorithms in [14] and [15] aims to reduce the runtime of the algorithm in [9], while the quality of solutions got by them is slightly lower. Since one of the main advantages of meta-heuristic algorithms over heuristic algorithms is their effectiveness (quality of solutions), the heuristic algorithm in [9] is chosen as a comparing algorithm on the effectiveness in this paper. The heuristic algorithm was reimplemented and the correctness of the implementation has been testified by comparing the simulation results with the original paper's [9] on hundreds randomly generated graphs. It is notable that the heuristic algorithm always starts with an empty initial solution and it is a deterministic algorithm, so it is impossible to make any random initial condition and the same results will be obtained in multiple runs. Therefore, for each benchmark graph, the heuristic algorithm needs to be run only once, while EA/SM is to be run independently for 30 times to get the statistical results as the measure of performance. The following terms of the results are recorded in the experiments.

Algorithm 5 Repair-Based EDA (G)

```
$p_i \leftarrow 0.5$; $i = 1, 2, \ldots, 2n$
$B_i \leftarrow$ Sampling ($P$); $i = 1, 2, \ldots, N$
$B_i \leftarrow$ Heuristic Repair ($B_i$); $i = 1, 2, \ldots, N$
**Repeat**
**Update** $P$ according to (5);
$B_i \leftarrow$ Sampling ($P$); $i = 1, 2, \ldots, N$
$B_i \leftarrow$ Heuristic Repair ($B_i$); $i = 1, 2, \ldots, N$
**until** maximum iteration reached
```

Algorithm 6 Repair-Based GA (G)

```
$p_i \leftarrow 0.5$; $i = 1, 2, \ldots, 2n$
$B_i \leftarrow$ Sampling ($P$); $i = 1, 2, \ldots, N$
$B_i \leftarrow$ Heuristic Repair ($B_i$); $i = 1, 2, \ldots, N$
**Repeat**
$B_i$' $\leftarrow$ Selection Crossover and Mutation ($B$); $i = 1, 2, \ldots, N$
$B_i$' $\leftarrow$ Heuristic Repair ($B_i$'); $i = 1, 2, \ldots, N$
**until** maximum iteration reached
```

1) *Heuristic:* The size obtained by the state-of-the-art heuristic algorithm [9].
2) *Avg:* The average size of the balanced biclique obtained by each EA in 30 runs.
3) *Std:* The standard deviation of the sizes of the balanced biclique obtained by each EA in 30 runs.
4) *Best:* The size of the largest balanced biclique obtained by each EA in 30 runs.
5) *Ftime:* The average times of fitness evaluation that each EA needed to obtain a balanced biclique as large as the Heuristic in 30 runs.
6) *Rtime:* The average times of restart process of each EA in 30 runs.
7) *Time:* The average runtime (in seconds) of each EA in 30 runs.

Table I provides the comparison results between the state-of-the-art heuristic algorithm in [9] and the proposed EA/SM on the MBBP (statistical test results will be presented in a separate table). It can be seen that EA/SM produces significantly better results (lager k) than the Heuristic [9] on all benchmark graphs. In particular, EA/SM can achieve improvements of 5.12%–21.48% with very small standard deviations (≤ 1), and the improvements are more than 10% in most cases (21/30). In addition, EA/SM can obtain a balanced biclique as large as the heuristic in very limited times of fitness evaluation; the value *Ftime* of EA/SM is 0.89%–27.11% of the $F_{\max}$ and less than 10% of the $F_{\max}$ in most cases (26/30). The invoking times of restart process are also very limited in EA/SM; the average period between restarts is 698.6–3072.2 times of fitness evaluation. Given fixed n, as p decreases, the problem becomes more difficult. The following results are observed: 1) the sizes k obtained by both the heuristic algorithm [9] and EA/SM decrease sharply according with the boundary theory [4]; 2) the values of *Ftime* decrease, indicating EA/SM prone to converge more quickly; 3) the values of *Rtime* increase, indicating the restart process is invoked more frequently

TABLE I
COMPARISON RESULTS BETWEEN THE STATE-OF-THE-ART HEURISTIC ALGORITHM [9] AND EA/SM ON THE MBBP (STATISTICAL TEST RESULTS ARE PRESENTED IN TABLE IV)

Graph No.	Graph name	Heuristic	EA/SM				
			Avg±Std	Best	Ftime	Rtime	time(s)
1	*n*250_*p*95%_1	62	67.9±0.3	68	10661.5	111.1	52.94
2	*n*250_*p*95%_2	58	65.3±0.7	66	5715.9	118.5	52.37
3	*n*250_*p*95%_3	66	69.4±0.5	70	24422.5	108.6	53.49
4	*n*250_*p*95%_4	62	67.1±0.6	68	11698.2	114	53.02
5	*n*250_*p*95%_5	61	67.1±0.3	68	7724.7	114.8	53.42
6	*n*250_*p*90%_1	38	43.8±0.4	44	4637.9	197.1	48.31
7	*n*250_*p*90%_2	39	43.9±0.3	44	7604.9	195.9	48.61
8	*n*250_*p*90%_3	37	43.3±0.7	44	3119.1	200.9	48.42
9	*n*250_*p*90%_4	40	44±0.3	45	9524.7	197.5	48.53
10	*n*250_*p*90%_5	40	43.9±0.6	45	12386.3	195.5	48.35
11	*n*250_*p*85%_1	29	32.5±0.5	33	7031.5	282.8	47.07
12	*n*250_*p*85%_2	29	32.8±0.4	33	6055.6	280.2	47.04
13	*n*250_*p*85%_3	27	32.8±0.4	34	1772.3	279.3	47.22
14	*n*250_*p*85%_4	30	32.7±0.5	33	15971.1	281	47
15	*n*250_*p*85%_5	29	32.1±0.4	33	9198	286.3	47.16
16	*n*500_*p*95%_1	82	90.3±0.7	91	18535.1	65.1	100.2
17	*n*500_*p*95%_2	81	88.1±0.8	89	24272.6	68	98.48
18	*n*500_*p*95%_3	81	87.3±0.8	89	20728	73.8	99.49
19	*n*500_*p*95%_4	79	86.2±0.8	88	17700.5	75.4	99.37
20	*n*500_*p*95%_5	84	88.3±0.8	90	54221.5	68.9	99.12
21	*n*500_*p*90%_1	48	53.8±0.9	56	11507.6	149.6	85.41
22	*n*500_*p*90%_2	48	53.9±0.7	56	10055.1	151.1	85.36
23	*n*500_*p*90%_3	48	53.1±0.7	54	13037.7	153.4	85.49
24	*n*500_*p*90%_4	48	53.6±0.9	55	12025.8	150.9	85.45
25	*n*500_*p*90%_5	48	53.3±0.8	55	10726.27	155.2	85.57
26	*n*500_*p*85%_1	34	37.9±0.8	40	10296.9	240.3	81.86
27	*n*500_*p*85%_2	35	39.4±0.9	41	16263.2	228	81.65
28	*n*500_*p*85%_3	34	38.2±0.7	40	7822.7	238	81.4
29	*n*500_*p*85%_4	34	38.3±0.8	40	8162.9	237	81.77
30	*n*500_*p*85%_5	34	38.9±1	41	8493.5	232.9	81.54

in EA/SM; and 4) the values of time decrease, indicating EA/SM becomes less time-consuming for sparser benchmark graphs.

B. Effectiveness of Main Contributions: Structure Mutation and New Repair-Assisted Restart

In EA/SM, the structure mutation is proposed to improve the exploration capability of the algorithm during the local search process to escape from local optima, while the new repair-assisted restart process is introduced to make full use of the statistical information extracted from the good solutions during the local search process with probability model. A very natural question is whether the proposed SM operator and the new repair-assisted restart process have any positive contribution to the performance of the algorithm. To answer this question, the performance of EA/SM are compared with three other EAs.

1) Evolutionary algorithm with greedy restart (EA/GR), which has the same framework as EA/SM, but no SM and repair operator is adopted. EA/GR adopts the same restart process as that present in [17].
2) Evolutionary algorithm with the new repair-assisted restart (EA/RR), which has the same framework as EA/SM, and the new repair-assisted restart process is adopted after local search process, but no SM is adopted.
3) Evolutionary algorithm with structure mutation and greedy restart (EA/SM/GR), which has the same framework as EA/SM, and the SM is adopted in local search process, but no repair operator is adopted. EA/SM/GR adopts the same restart process as that present in [17]. All the parameters of EA/GR, EA/RR, and EA/SM/GR are set as the same as EA/SM. The three algorithms are run independently for 30 times on each benchmark graph.

Table II provides the experiment results of the four algorithms on the MBBP (statistical test results will be presented in a separate table). It can be seen that the performance of ER/GR is much weaker than those of other EAs. EA/GR even cannot obtain a biclique as large as that got by the heuristic algorithm [9] in most case, so the values of *Ftime* are unfilled in Table II. In particular, it is observed that the restart process in EA/GR is invoked more frequently (2.5–4.2 times) than EA/SM to help algorithm jump out of the local optima, which results in much longer runtimes. It can also be observed in Table II that the performances of both EA/RR and EA/SM/GR are weaker than that of EA/SM, while they both outperform the heuristic algorithm significantly.

TABLE II
EXPERIMENT RESULTS OF EA/GR, EA/RR, AND EA/SM/GR ON THE MBBP (STATISTICAL TEST RESULTS ARE PRESENTED IN TABLE IV)

No.	EA/GR				EA/RR					EA/SM/GR				
	Avg±Std	Best	Rtime	time(s)	Avg±Std	Best	Ftime	Rtime	time(s)	Avg±Std	Best	Ftime	Rtime	time(s)
1	58.6±1.2	61	344.3	67.04	66±1.2	68	32330.4	292.3	61.11	66.2±0.8	68	10958.1	126.7	55.32
2	56.4±0.9	58	354.1	67.33	63.5±1.1	66	18901.6	305.9	60.68	63.5±0.8	65	5602.4	134.2	55.31
3	60.3±1.2	64	332.2	67.37	68±1	70	60004.9	283.6	61.43	68.1±0.6	69	26750.9	123.6	55.85
4	58.2±1	61	343.6	67.61	65.2±0.8	67	37244.5	296.3	61.35	65.3±0.6	67	10958	130.2	55.9
5	58.6±1.1	61	342.9	67.55	65.7±1	67	19369.1	295.2	61.45	65.7±0.7	67	7482.5	129.2	55.94
6	37.8±0.6	39	538.4	69.03	42.5±0.8	44	12673	463.1	58.13	42±0.6	44	3856.6	220.9	52.95
7	37.7±0.8	39	539.3	69.1	42.5±0.9	45	17003.9	464.6	58.14	42.4±0.8	44	7081.7	219.3	52.91
8	37.8±0.8	40	541.1	69.22	41.8±0.8	43	10181.5	471.6	58	41.8±0.6	43	2951.5	223.5	53.08
9	38.3±0.8	40	533.9	69.11	43±0.7	44	23211.2	458.8	58.1	42.7±0.6	44	10753	216.6	52.99
10	37.7±0.6	39	542.9	69.18	42.8±1.1	44	24344.5	462.1	57.96	42.5±0.6	44	15504.7	218.3	52.71
11	28.5±0.6	30	723.8	73.99	31.5±0.8	33	17249.4	631.1	57.77	31.1±0.5	32	7388.87	309.2	53.94
12	28.8±0.7	30	716.3	73.74	31.8±0.6	33	16300.4	625.2	57.8	31.7±0.6	33	5759.9	305.8	53.81
13	29±0.7	31	714.3	73.45	32.1±0.6	33	6128.5	617.7	57.66	31.8±0.6	33	1920.1	303	53.67
14	28.5±0.7	30	725.1	73.93	31.7±0.7	33	35974.2	628.5	57.9	31.5±0.6	33	36959.1	308.8	53.92
15	28.4±0.7	30	728.1	74.12	31.3±0.7	32	23165.5	635.4	57.83	30.9±0.4	32	8364.3	312.7	54.16
16	73.9±1.6	78	270.7	121.85	87.2±1.2	89	55402.1	219.7	114.67	87.4±1.1	90	16542.1	87.6	103.99
17	71.9±0.9	73	276.6	121.13	84.7±1.2	87	58543.7	225.5	112.94	85.4±0.9	88	22465	89.5	102.26
18	72.1±1.2	76	275.7	121.48	84.9±1	87	67425.8	225.6	113.05	85.4±1	88	20285.2	90	102.81
19	71.2±1	73	279.9	122.21	83.9±0.8	86	49378.1	228.9	113.46	83.6±0.9	86	18286.9	92.3	103.17
20	72.5±1.2	75	274.8	121.5	85.4±1.1	88	108671.6	225.5	113.24	85.5±0.8	87	45220.8	90	102.94
21	44.7±0.8	46	453.3	121.45	51.6±1.1	54	36911	381.9	103	51.1±0.9	53	12006.9	175.4	92.96
22	45±1.1	47	452.5	121.18	51.6±1.1	54	30994.5	381.3	102.92	51.1±0.8	53	12096.3	175.2	92.93
23	44.6±0.7	46	455.2	121.42	51.7±0.9	53	38187.9	382.6	102.97	50.9±0.7	52	15009	177.6	93.05
24	44.9±1	47	454.7	121.16	51.7±0.9	54	31365.8	380.6	102.84	50.8±0.7	52	11777.9	177.4	93.16
25	44.9±0.9	47	452.3	121.14	51.4±0.8	52	35223.9	384.3	103.06	50.8±0.7	53	12086.7	177	93.2
26	32.8±0.8	35	626.4	131.61	37±0.7	38	26742.6	539.3	101.72	36.3±0.7	38	10292.2	262.2	94.58
27	32.8±0.6	34	622.3	130.94	37.6±0.8	39	35369.9	530.9	101.78	36.7±0.6	38	23007.2	259	94.23
28	32.8±0.6	34	624.3	131.17	37.4±0.9	39	26330.3	532.7	101.28	36.4±0.6	37	7982.1	260.6	94.24
29	33.1±0.7	34	621	130.77	37.3±0.9	39	31258.7	536	101.83	36.6±0.6	38	7676.3	258.9	94.26
30	33.1±0.7	35	622.3	130.56	37.1±0.8	38	26763.9	538.3	101.65	36.4±0.6	38	7783.9	259.7	94.15

Without of SM, the performance of EA/RR can be summarized as follows: 1) the values of *Ftime* are 2–3.8 times more than that of EA/SM; 2) the values of *Rtime* are 2.2–3.4 times more than that of EA/SM; and 3) the runtimes are 13.6%–24.7% longer than that of EA/SM. Making use of greedy restart process [17] in place of the proposed new repair-assisted restart process, the performance of EA/SM/GR can be summarized as follows: 1) the values of *Rtime* are 8.5%–34.6% more than that of EA/SM and 2) the runtimes are 3.3%–15.8% longer than that of EA/SM, although the greedy restart process itself is slightly faster than the new repair-assisted restart process.

C. Comparisons With Repair-Based EAs

For comprehensively understanding the advantage of EA/SM, it is also compared with two other repair-based EAs, which are constructed by incorporating the modified Marchiori's heuristic repair operator into an EDA and a GA, respectively, as shown in Algorithms 5 and 6. In the two repaired-based EAs, every solution generated by evolutionary operators is to be repaired by the heuristic repair operator as did in [19] and [20]. The procedure of repair-based EDA is easy to understand, while the repair-based GA obeys the definition of basic GA which uses roulette wheel selection, single-point crossover and one-bit mutation. Since GAs are prone to be trapped in local optima for combinatorial optimization problems with strict constraints, so no elitist strategy is used in repair-based GA for the MBBP. The population size and learning rate adopted in the repair-based EDA are set as the same as EA/SM. The population size, crossover probability, and mutation probability adopted in the repair-based GA are 40, 0.8, and 0.1, respectively. As mentioned before, the heuristic repair operator is more time-consuming than both local search and SM. In our simulation experiments conducted in this paper, it is observed that the local search and SM operators are approximately two orders of magnitude faster than the heuristic repair operator. Therefore, a direct comparison between EA/SM and repair-based EAs is unfair. In order to solve this problem, a fixed and large $F_{\max}$ is set as the termination condition for the three algorithms to guarantee them converging to optimum or suboptimum solutions before termination. Based on this setting, the repair-based EAs stop after 4000 calls of the heuristic repair operator while EA/SM still stops after 200 000 calls of the local search and SM operators. In spite of above setting, EA/SM is still much faster than the repair-based EAs, especially for large *n*. The two repair-based EAs are run independently for 30 times on each benchmark graph.

Table III provides the comparison results between EA/SM and the other repair-based EAs (statistical test results will be presented in a separate table). It can be seen that the performance of both repair-based EDA and repair-based GA are weaker

TABLE III
COMPARISON RESULTS BETWEEN EA/SM, REPAIR-BASED EDA, AND REPAIR-BASED GA ON THE MBBP (STATISTICAL TEST RESULTS ARE PRESENTED IN TABLE IV)

No.	EA/SM			Repaire-based EDA			Repaire-based GA		
	Avg±Std	Best	time(s)	Avg±Std	Best	time(s)	Avg±Std	Best	time(s)
1	67.9±0.3	68	52.94	64.2±1.1	67	77.72	66.2±0.8	68	78.54
2	65.3±0.7	66	52.37	61.7±1.3	64	76.43	63.3±0.6	65	77.53
3	69.4±0.5	70	53.49	65.7±1	68	78.15	67.7±0.7	69	78.67
4	67.1±0.6	68	53.02	63.4±1	65	78.31	64.9±0.8	67	78.05
5	67.1±0.3	68	53.42	63.4±1	65	78.86	65.4±0.7	67	78.57
6	43.8±0.4	44	48.31	40.9±0.9	43	69.86	42.7±0.5	44	70.88
7	43.9±0.3	44	48.61	41.2±0.9	44	69.22	43±0.8	44	70.69
8	43.3±0.7	44	48.42	40.4±0.9	42	69.3	41.9±0.7	43	70.46
9	44±0.3	45	48.53	41.3±1	44	69.33	42.8±0.7	44	70.7
10	43.9±0.6	45	48.35	41.2±1	43	69.47	42.9±0.8	44	70.79
11	32.5±0.5	33	47.07	30.3±0.9	32	64.67	31.3±0.5	32	66.42
12	32.8±0.4	33	47.04	30.8±0.7	32	65.78	31.7±0.5	33	66.52
13	32.8±0.4	34	47.22	30.8±0.9	32	65.3	32.1±0.6	33	66.5
14	32.7±0.5	33	47	30.4±0.7	32	65.47	31.5±0.6	32	66.1
15	32.1±0.4	33	47.16	30.3±0.8	32	65.32	30.8±0.7	32	66.07
16	90.3±0.7	91	100.2	83±1.8	87	203.97	88.37±1.1	90	210.5
17	88.1±0.8	89	98.48	81.2±1.6	84	203.95	86.2±1.1	88	209.14
18	87.3±0.8	89	99.49	81.4±1.2	83	204.22	85.9±0.9	88	209.25
19	86.2±0.8	88	99.37	80.7±1.2	83	203	84.3±0.9	86	208.39
20	88.3±0.8	90	99.12	81.7±1.7	85	205.14	85.9±1.1	89	209.04
21	53.8±0.9	56	85.41	49.3±1.4	51	179.27	51.8±1.2	54	186.29
22	53.9±0.7	56	85.36	50.1±1	52	180.13	51.6±1.4	54	186.33
23	53.1±0.7	54	85.49	49.6±1.3	52	179.62	49.9±1.9	54	186.17
24	53.6±0.9	55	85.45	49.7±1.3	52	181.81	51.1±1.4	53	186.13
25	53.3±0.8	55	85.57	49.6±1.3	52	178.39	49.6±1.8	53	186.39
26	37.9±0.8	40	81.86	35.6±0.8	37	168.37	32.1±0.5	33	176.37
27	39.4±0.9	41	81.65	36.5±0.9	38	167.48	33±0.7	34	176.47
28	38.2±0.7	40	81.4	35.6±0.8	37	168.87	32.4±0.5	34	175.96
29	38.3±0.8	40	81.77	36.1±0.8	37	167.71	32.7±0.6	34	176.2
30	38.9±1	41	81.54	35.8±0.9	38	168.86	32.5±0.6	34	176.87

TABLE IV
STATISTICAL SIGNIFICANCE WIN–LOSS–TIE SUMMARY OF TWO-TAILED *t*-TEST BASED ON 30 BENCHMARK GRAPHS

Benchmark Graphs	EA/SM	EA/GR	EA/RR	EA/SM/GR	Repair-based EDA	Repair-based GA
EA/SM	\	30-0-0	30-0-0	30-0-0	30-0-0	30-0-0
EA/GR	0-30-0	\	0-30-0	0-30-0	0-30-0	3-25-2
EA/RR	0-30-0	30-0-0	\	12-1-17	30-0-0	8-4-18
EA/SM/GR	0-30-0	30-0-0	1-12-17	\	29-0-1	9-8-13
Repair-based EDA	0-30-0	30-0-0	0-30-0	0-29-1	\	5-23-2
Repair-based GA	0-30-0	25-3-2	4-8-18	8-9-13	23-5-2	\

than that of EA/SM, while both of them outperform the heuristic algorithm [9] in most case. As mentioned above, the heuristic repair operator is very time-consuming and its runtime increases sharply with n, therefore the runtimes of the repair-based EAs are much longer than that of EA/SM, especially for large-scale benchmark graphs ($n = 500$). It can also be observed that the performance of repair-based EDA is slightly weaker than that of repair-based GA in most case. A reasonable explanation is that EDA is prone to premature convergence by its nature, and the convergence is further accelerated by invoking the heuristic repair operator frequently. The performance of repair-based GA becomes poor as p decreases, especially for $n500\_p85\%\_\mathrm{C}$ problems, A reasonable explanation is that is more difficult to extract a large biclique from a sparser graph, and the basic GA cannot jump out of the local optima for these hard problems although no elitist strategy is used.

D. Statistical Comparisons on Single Benchmark Graph

In order to compare the EAs in this paper in a sound statistical context, statistical tests are performed for paired EAs, e.g., EA/SM versus EA/GR and EA/RR versus the repair-based EAs, on each single benchmark graph. Statistical tests are carried out on the size k obtained by each EA and provide the win–loss–tie summary. A two-tailed t-test is conducted with a null hypothesis stating that there is no difference between two algorithms in comparison. The null hypothesis was rejected if the p-value is smaller than the significance level $\alpha = 0.05$. Table IV gives the win–loss–tie summary that comparing the

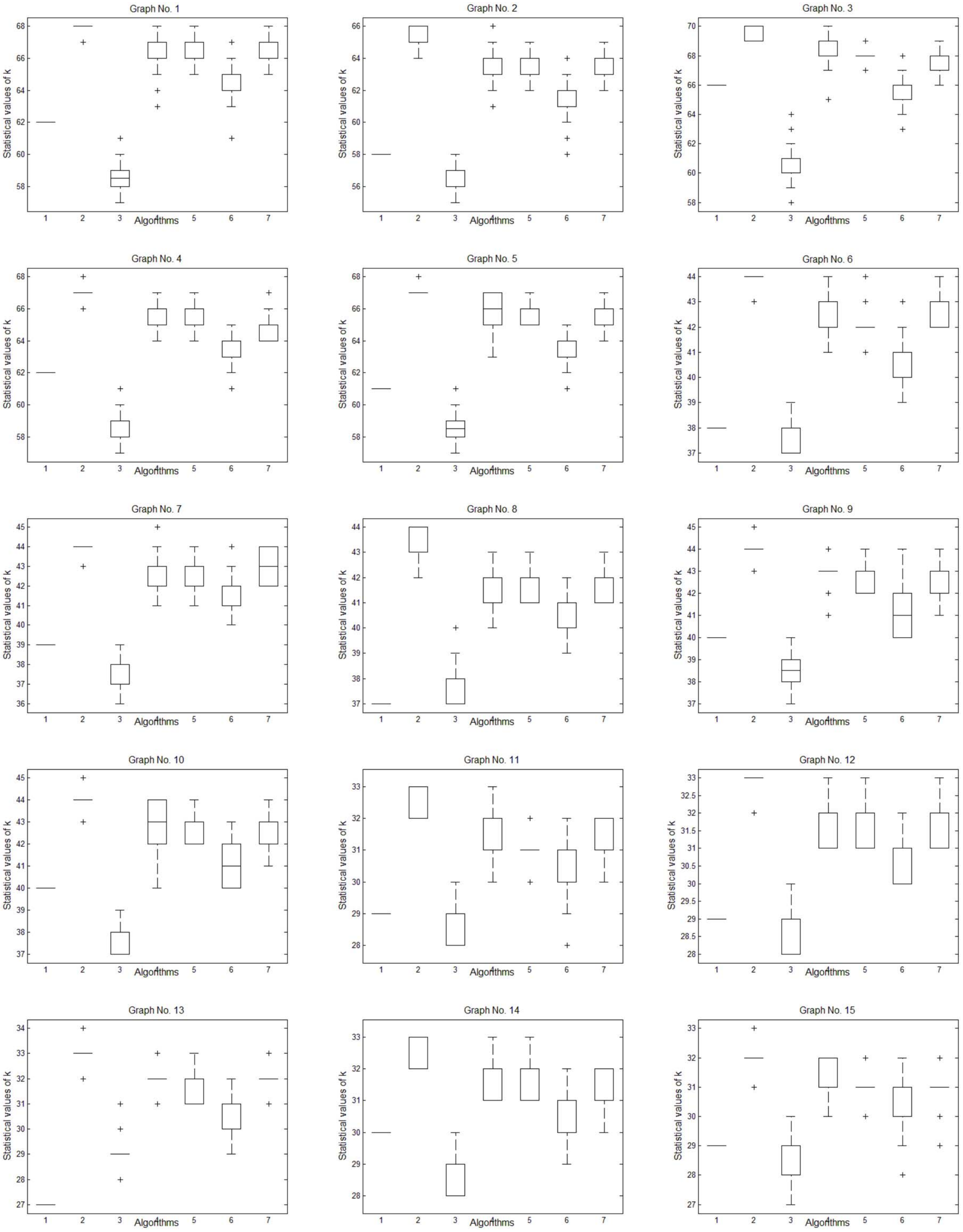


Fig. 3. Values of k on benchmark graphs with size n = 250 (Graph No. 1–15) obtained by 1) Heuristic; 2) EA/SM; 3) EA/GR; 4) EA/RR; 5) EA/SM/GR; 6) repair-based EDA; and 7) repair-based GA, respectively. Box plots are used to illustrate the distribution of these values. In each box, the central mark is the median, the edges of the box are the 25th and 75th percentiles, the whiskers extend to the most extreme data points the algorithm considers to be not outliers, and the outliers are plotted by + individually.

EAs in the 1st column with the EAs in the 1st row based on 30 benchmark graphs. For example, the 2nd row compares EA/SM with other five EAs, EA/SM gets 30-0-0 (30 wins, 0 loss, 0 ties) in each paired comparison. Another example, row 4 column 5 compares EA/RR with EA/SM/GR, EA/RR gets 12-1-17 (12 wins, 1 loss, 17 ties) in the comparison. The significance tests show that EA/SM is statistically better than other EAs on all benchmark graphs (30 wins).

Visualized comparisons of all the algorithms in this paper are displayed by box-plot in Figs. 3 and 4, which show the

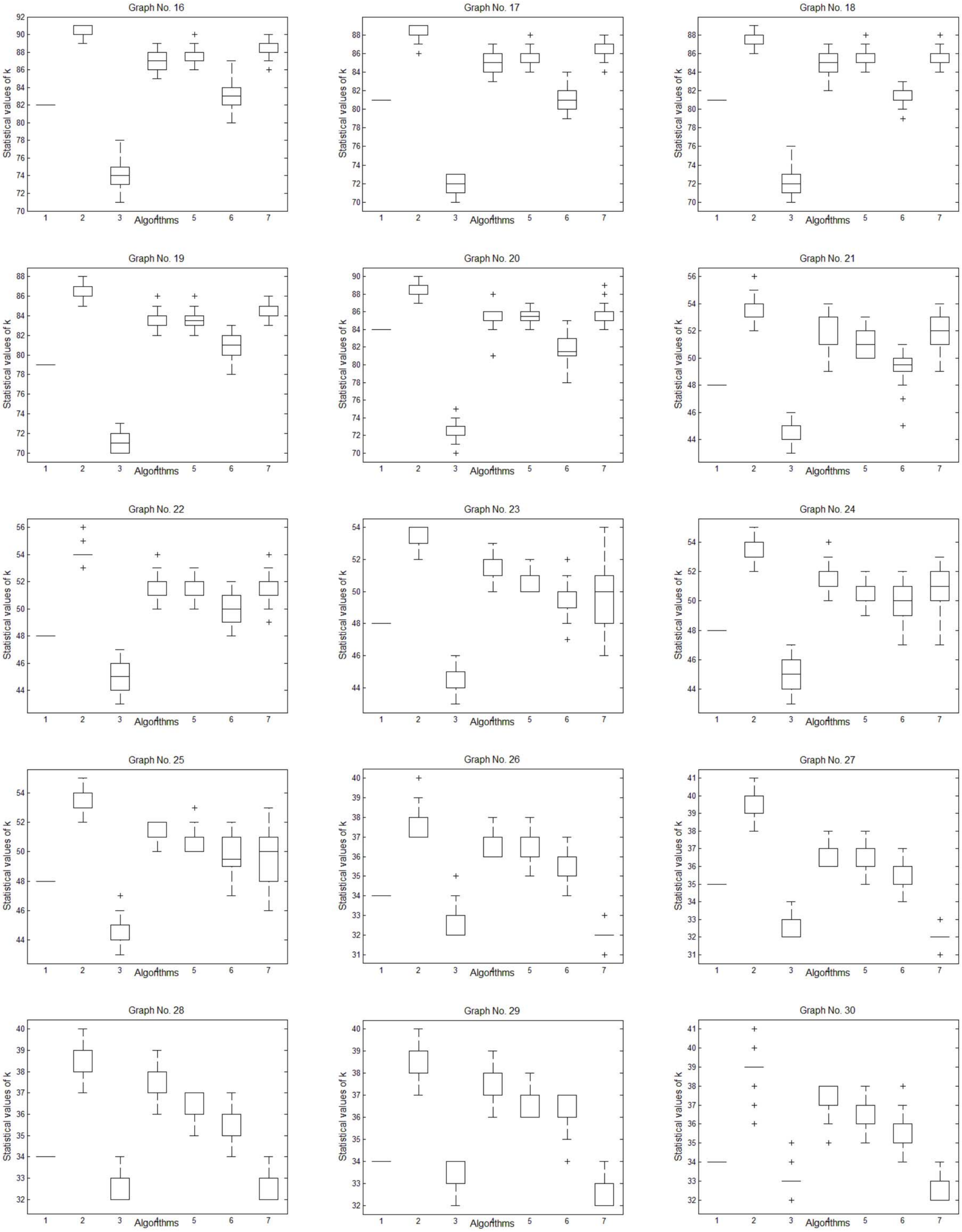


Fig. 4. Values of k on benchmark graphs with size n = 500 (Graph No. 16–30) obtained by 1) Heuristic; 2) EA/SM; 3) EA/GR; 4) EA/RR; 5) EA/SM/GR; 6) repair-based EDA; and 7) repair-based GA, respectively. Box plots are used to illustrate the distribution of these values. In each box, the central mark is the median, the edges of the box are the 25th and 75th percentiles, the whiskers extend to the most extreme data points the algorithm considers to be not outliers, and the outliers are plotted by + individually.

distribution of the values of k on 30 benchmark graphs with size n = 250 and n = 500, respectively.

V. Conclusion

This paper proposes an EA with SM and a new repair-assisted restart scheme for the MBBP. Experimental investigation on a large set of benchmark graphs shows that: 1) effective heuristics (local search and SM) coupled with an EA (EDA) can produce significantly better results than the state-of-the-art algorithm for the MBBP; 2) for complex combinatorial optimization problems, such as the MBBP, the SM, designed for enhancing the exploration capability of the algorithm, can effectively help to avoid stagnation in local optima; and 3) the new repair-assisted restart process can take full advantage of the statistical knowledge learned from good solutions obtained in previous search process, therefore can generate more promising reinitialized solutions for the following local search process.

The proposed SM operator is not limited to the MBBP. It can easily be generalized to other similar problems, such as the MCP [18], the maximum edge biclique problem [36], and the maximum edge-weighted clique problem [37]. We believe that the idea, i.e., efficient local search accompanied by a large mutation, is also applicable to other hard optimization problems.

REFERENCES

[1] M. S. Garey and D. S. Johnson, *Computers and Intractibility: A Guide to NP-Completeness*. New York, NY, USA: Freeman & Co., 1979.

[2] D. S. Johnson, "The NP-completeness column: An ongoing guide," *J. Algorithms*, vol. 8, no. 3, pp. 438–448, 1987.

[3] N. Alon, R. A. Duke, H. Lefmann, V. Rodl, and R. Yuster, "The algorithmic aspects of the regularity lemma," *J. Algorithms*, vol. 16, no. 1, pp. 80–109, 1994.

[4] M. Dawande, P. Keskinocak, J. Swaminathan, and S. Tayur, "On biclique and multi-partite clique problems," *J. Algorithms,* vol. 41, no. 2, pp. 388–403, 2001.

[5] U. Feige and S. Kogan, "Hardness of approximation of the balanced complete bipartite subgraph problem," Dept. Comput. Sci. Appl. Math., Weizmann Inst. Sci., Rehovot, Israel, Tech. Rep. MCS04-04, 2004.

[6] Y. Cheng and G. M. Church, "Biclustering of expression data," in *Proc. Int. Conf. Intell. Syst. Mol. Biol.*, 2000, pp. 93–103.

[7] S. S. Ravi and E. L. Lloyd, "The complexity of near-optimal programmable logic array folding," *SIAM J. Comput.*, vol. 17, no. 4, pp. 696–710, 1988.

[8] M. B. Tahoori, "Application-independent defect tolerance of reconfigurable nanoarchitectures," *ACM J. Emerg. Technol. Comput. Syst.,* vol. 2, no. 3, pp. 197–218, Jul. 2005.

[9] A. Al-Yamani, S. Ramsundar, and D. K. Pradhan, "A defect tolerance scheme for nanotechnology circuits," *IEEE Trans. Circuits Syst. I, Reg. Papers,* vol. 54, no. 11, pp. 2402–2409, Nov. 2007.

[10] M. B. Tahoori, "Low-overhead defect tolerance in crossbar nanoarchitectures," *ACM J. Emerg. Technol. Comput. Syst.,* vol. 5, no. 2, Jul. 2009, Art. ID 11.

[11] M. B. Tahoori, "Defects, yield, and design in sublithographic nano-electronics," in *Proc. 20th IEEE Int. Symp. Defect Fault Tolerance VLSI Syst.,* 2005, pp. 3–11.

[12] M. Yannakakis, "Node-deletion problems on bipartite graphs," *SIAM J. Comput.*, vol. 10, no. 2, pp. 310–327, May 1981.

[13] D. Mubayi and G. Turan, "Finding bipartite subgraphs efficiently," *Inf. Process. Lett.,* vol. 110, no. 5, pp. 174–177, Feb. 2010.

[14] B. Yuan and B. Li, "A low time complexity defect-tolerance algorithm for nanoelectronic crossbar," in *Proc. Int. Conf. Inf. Sci. Technol.*, Nanjing, China, Mar. 2011, pp. 143–148.

[15] B. Yuan and B. Li, "A fast extraction algorithm for defect-free subcrossbar in nanoelectronic crossbar," *ACM J. Emerg. Technol. Comput. Syst.,* vol. 10, no. 3, Apr. 2014, Art. ID 25.

[16] X. Chen, Y. S. Ong, M. H. Lim, and K. C. Tan, "A multi-facet survey on memetic computation," *IEEE Trans. Evol. Comput.*, vol. 15, no. 5, pp. 591–607, Oct. 2011.

[17] M. Brunato and R. Battiti, "R-EVO: A reactive evolutionary algorithm for the maximum clique problem," *IEEE Trans. Evol. Comput.*, vol. 15, no. 6, pp. 770–782, Dec. 2011.

[18] I. M. Bomze, M. Budinich, P. M. Paradalos, and M. Pelillo, "The maximum clique problem," in *Handbook of Combinatorial Optimization*, vol. 4, D.-Z. Du and P. M. Paradalos, Eds. Norwell, MA, USA: Kluwer Academic, 1999.

[19] E. Marchiori, "Genetic, iterated and multistart local search for the maximum clique problem," in *Applications of Evolutionary Computing* (Lecture Notes in Computer Science), vol. 2279. Berlin, Germany: Springer-Verlag, 2002, pp. 112–121.

[20] Q. Zhang, J. Sun, and E. Tsang, "An evolutionary algorithm with guided mutation for the maximum clique problem," *IEEE Trans. Evol. Comput.*, vol. 9, no. 2, pp. 192–200, Apr. 2005.

[21] R. Battiti and M. Protasi, "Reactive local search for the maximum clique problem," *Algorithmica*, vol. 29, no. 4, pp. 610–637, 2001.

[22] R. Battiti and A. A. Bertossi, "Greedy, prohibition, and reactive heuristics for graph partitioning," *IEEE Trans. Comput.*, vol. 48, no. 4, pp. 361–385, Apr. 1999.

[23] K.-H. Liang, X. Yao, C. S. Newton, and D. Hoffman, "A new evolutionary approach to cutting stock problems with and without contiguity," *Comput. Oper. Res.*, vol. 29, no. 12, pp. 1641–1659, Oct. 2002.

[24] K. Tang, Y. Mei, and X. Yao, "Memetic algorithm with extended neighborhood search for capacitated arc routing problems," *IEEE Trans. Evol. Comput.*, vol. 13, no. 5, pp. 1151–1166, Oct. 2009.

[25] X. Yao, "Simulated annealing with extended neighbourhood," *Int. J. Comput. Math.,* vol. 40, nos. 3–4, pp. 169–189, 1991.

[26] X. Yao, "Dynamic neighbourhood size in simulated annealing," in *Proc. Int. Joint Conf. Neural Netw.*, 1992, pp. 411–416.

[27] S. Baluja, "Population-based incremental learning: A method for integrating genetic search based function optimization and competitive learning," School Comput. Sci., Carnegie Mellon Univ., Pittsburgh, PA, USA, Tech. Rep. CMU-CS-94-163, 1994.

[28] S. Baluja and R. Caruana, "Removing the genetics from the standard genetic algorithm," School Comput. Sci., Carnegie Mellon Univ., Pittsburgh, PA, USA, Tech. Rep. CMU-CS-95-141, 1995.

[29] H. Mühlenbein, "The equation for response to selection and its use for prediction," *Evol. Comput.*, vol. 5, no. 3, pp. 303–346, 1998.

[30] H. Müehlenbein, T. Mahnig, and A. O. Rodriguez, "Schemata, distributions, and graphical models in evolutionary optimization," *J. Heuristics*, vol. 5, no. 2, pp. 215–247, 1999.

[31] M. Pelikan, D. Goldberg, and E. Cantu-Paz, "BOA: The Bayesian optimization algorithm," in *Proc. Genet. Evol. Comput. Conf.*, vol. 1. 1999, pp. 525–532.

[32] M. Zlochin, M. Birattari, N. Meuleau, and M. Dorigo, "Model-based search for combinatorial optimization," *Ann. Oper. Res.*, vol. 131, nos. 1–4, pp. 373–395, 2004.

[33] B. T. Zhang, "A Bayesian framework for evolutionary computation," in *Proc. Congr. Evol. Comput.*, 1999, pp. 722–228.

[34] P. Larranaga, *Estimation of Distribution Algorithms: A New Tool for Evolutionary Computation*. Norwell, MA, USA: Kluwer Academic, 2001.

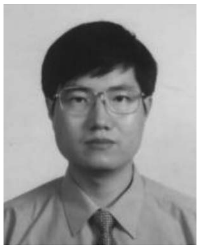
[35] M. Pelikan, D. Goldberg, and F. Lobo, "A survey of optimization by building and using probabilistic models," *Comput. Optim. Appl.,* vol. 21, no. 1, pp. 5–20, 2002.

[36] R. Peeters, "The maximum edge biclique problem is NP-complete," *Discrete Appl. Math.*, vol. 131, no. 3, pp. 651–654, 2003.

[37] E. M. Macambira and C. C. De Souza, "The edge-weighted clique problem: Valid inequalities, facets and polyhedral computations," *Eur. J. Oper. Res.*, vol. 123, no. 2, pp. 346–371, Jun. 2000.

Bo Yuan received the B.Sc. and the Ph.D. degrees in electronic information science and technology from the University of Science and Technology of China (USTC), Hefei, China, in 2009 and 2014, respectively.

He is currently a Post-Doctoral Fellow with the USTC-Birmingham Joint Research Institute in Intelligent Computation and Its Applications, USTC. From 2012 to 2013, he was a Visiting Scholar at the Centre of Excellence for Research in Computational Intelligence and Applications, University of Birmingham, Birmingham, U.K. His current research interests include evolutionary computation, electronic design automation, and machine learning.



Bin Li (M'07) received the B.Sc. degree from the Hefei University of Technology, Hefei, China, in 1992, the M.Sc. degree from the Institute of Plasma Physics, China Academy of Science, Hefei, in 1995, and the Ph.D. degree from the University of Science and Technology of China (USTC), Hefei, in 2001.

He is currently a Professor with the School of Information Science and Technology, USTC. He has authored and co-authored over 40 refereed publications. His current research interests include evolutionary computation, pattern recognition, and real-world applications.

Dr. Li is the Founding Chair of the IEEE CIS Hefei Chapter, a Counselor of the IEEE USTC Student Branch, a Senior Member of the Chinese Institute of Electronics (CIE), and a member of the Technical Committee of Electronic Circuits and Systems Section of CIE.

Huanhuan Chen (M’09) received the B.Sc. degree from the University of Science and Technology of China (USTC), Hefei, China, in 2004, and the Ph.D. degree in computer science from the University of Birmingham, Birmingham, U.K., in 2008.

He is currently a Full Professor with USTC-Birmingham Joint Research Institute of Intelligent Computation and Its Applications, School of Computer Science and Technology, USTC. His current research interests include statistical machine learning, data mining, fault diagnosis, and evolutionary computation.

Dr. Chen was the recipient of the 2011 IEEE Computational Intelligence Society Outstanding Ph.D. Dissertation Award (the only winner) and the 2009 CPHC/British Computer Society Distinguished Dissertations Award (the runner up). His work “Probabilistic Classification Vector Machines” on Bayesian machine learning has been awarded the IEEE TRANSACTIONS ON NEURAL NETWORKS Outstanding Paper Award (bestowed in 2011 and only one paper in 2009).

Xin Yao (F’03) is a Chair (Professor) of Computer Science and the Director of the Centre of Excellence for Research in Computational Intelligence and Applications, University of Birmingham, Birmingham, U.K. He has been invited to present over 70 keynote and plenary speeches at international conferences. He has published over 400 refereed publications in international journals and conferences. His current research interests include evolutionary computation and ensemble learning.

Prof. Yao was the recipient of the 2001 IEEE Donald G. Fink Prize Paper Award, the 2010 IEEE TRANSACTIONS ON EVOLUTIONARY COMPUTATION Outstanding Paper Award, the 2010 BT Gordon Radley Award for Best Author of Innovation (Finalist), the 2011 IEEE TRANSACTIONS ON NEURAL NETWORKS Outstanding Paper Award, and several other best paper awards at conferences. He was also the recipient of the Prestigious Royal Society Wolfson Research Merit Award in 2012 and was selected to receive the 2013 IEEE CIS Evolutionary Computation Pioneer Award. He is a Distinguished Lecturer of the IEEE Computational Intelligence Society. He was also the Editor-in-Chief of IEEE TRANSACTIONS ON EVOLUTIONARY COMPUTATION from 2003 to 2008.