WCCI 2012 IEEE World Congress on Computational Intelligence
June, 10-15, 2012 - Brisbane, Australia
IEEE CEC

# Use of Graph Kernels in Estimation of Distribution Algorithms

Hisashi Handa
Graduate School of Natural Science and Technology Okayama University
Okayama 700–8530, JAPAN
Email: handa@sdc.it.okayama-u.ac.jp

***Abstract*—The graph-related problems, which solutions are represented by graphs, have attracted much attention since there are a large number of application areas in bioinformatics and social science. In this paper, we propose a novel Estimation of Distribution Algorithm which can effectively cope with graphs. The proposed method employs graph kernels in estimation and sampling phases in the EDAs. The preliminary experiments on edge-max problems and edge-min problems elucidate the effectiveness of the proposed method.**

## I. Introduction

Estimation of Distribution Algorithms are promising Evolutionary Algorithms (EDAs) [1]. The EDAs employ the probabilistic models to reproduce offspring from parents, instead of genetic operators such as crossovers and mutations. Both of continuous and discrete problems are examined in EDA studies [1], [2], [3]. They have shown the effectiveness of the EDAs in these problem domains. A few researches, however, are found in their extensions to other problem domain [4].

As computational resources are increasing, graph-related problems, which solutions are represented by graphs, draw increasing attention. Since evolutionary algorithms can cope with various problems because of the flexibility of the EAs, the graph-related problems are a potential application area of the EAs. The graph-related problems are still time-consuming so that only several EA approaches can be found. For instance, Chicano and Alba propose ACOhg (Ant Colony optimization for huge graphs) for model verification [5], [6]. McDermott and O'Reilly employ graphs for music generation [7]. In addition, the notion of graph is incorporated into evolutionary algorithms [8], [9], [10].

In this paper, a novel estimation of distribution algorithm with graph kernels is proposed. By using the graph kernels, we can address the difficulties, explained in section II-A, in which we apply evolutionary algorithms to solve the graph-related problems. The individual representation in the proposed method is a graph so that we expect that the proposed method can apply any sorts of the graph-related problems. In order to estimate the distributions of graphs, we employ a kernel density function with the graph kernels.

The organization of the remainder of the paper is as follows: Section II points out the difficulties in the case of using graphs in evolutionary algorithms and introduces a graph kernel used in this paper, i.e., shortest-path graph-kernel. The general calculation procedure of Estimation of Distribution Algorithms is briefly introduced in section III. The estimation of distribution algorithms with graph kernels are proposed in Section IV. Preliminary experiments on two test functions, edge-max problems and edge-min problems, are examined in Section V.

## II. Graph Kernels

### A. The Difficulties in Graph-Related Problems for EAs

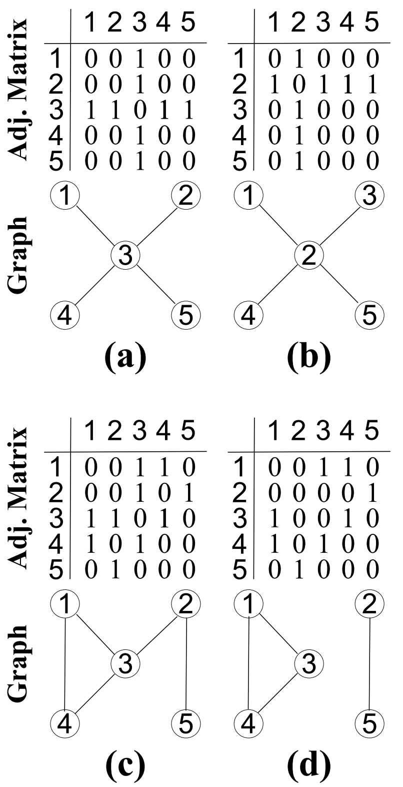
Evolutionary Algorithms have the flexibility which can be applied to any problem domains. Such flexibility is achieved by the characteristics of EAs: EAs do not require the differentiability in problems, but they only require that the fitness between two individuals can be compared. In order to constitute effective EAs for practical problems, however, we need to adopt a balanced approach in a coding method and genetic operations. This difficulty is often caused in graph-related problems.

For instance, suppose that the adjacency matrix is used as the genotype for undirected graph problems. In addition, suppose that, in this problems, we care about only the topology of edges. In Figure 1(a)(b), they are different representations but cause the same topology. Such reflected/rotated solutions in the adjacency matrixes coding yield multi-modal fitness functions even if original problems have the uni-modal landscape.

In Figure 1(c)(d), on the other hand, these adjacency matrixes are quite similar, i.e., the edge between nodes 2 and 3. They yield different graphs: The graph by Figure 1 (d) indicates a non-connected graph. It depends on the nature of original problems. In most cases, these solutions cause different performance. Such difference may be significantly small if the edge between nodes 1 and 4 is removed. That is, graph-related problems tend to have high dependency so that the neighborhood structure is quite rugged. In our humble opinion, these are the reasons why there are fewer studies regarding EAs for graph-related problems than the ones for continuous function optimization problems and combinatorial optimization problems.

### B. Shortest-Path Kernel on Graphs

Kernel methods have attracted much attention in Machine Learning communities. Recently, the notion of kernel functions has been extended to cope with graphs [11]. This paper employs the shortest-path graph kernel proposed by Borgwardt and Keiegel [12]: First, all the pairs-shortest-paths in two



(a)

| | 1 | 2 | 3 | 4 | 5 |
|---|---|---|---|---|---|
| 1 | 0 | 0 | 1 | 0 | 0 |
| 2 | 0 | 0 | 1 | 0 | 0 |
| 3 | 1 | 1 | 0 | 1 | 1 |
| 4 | 0 | 0 | 1 | 0 | 0 |
| 5 | 0 | 0 | 1 | 0 | 0 |

(b)

| | 1 | 2 | 3 | 4 | 5 |
|---|---|---|---|---|---|
| 1 | 0 | 1 | 0 | 0 | 0 |
| 2 | 1 | 0 | 1 | 1 | 1 |
| 3 | 0 | 1 | 0 | 0 | 0 |
| 4 | 0 | 1 | 0 | 0 | 0 |
| 5 | 0 | 1 | 0 | 0 | 0 |

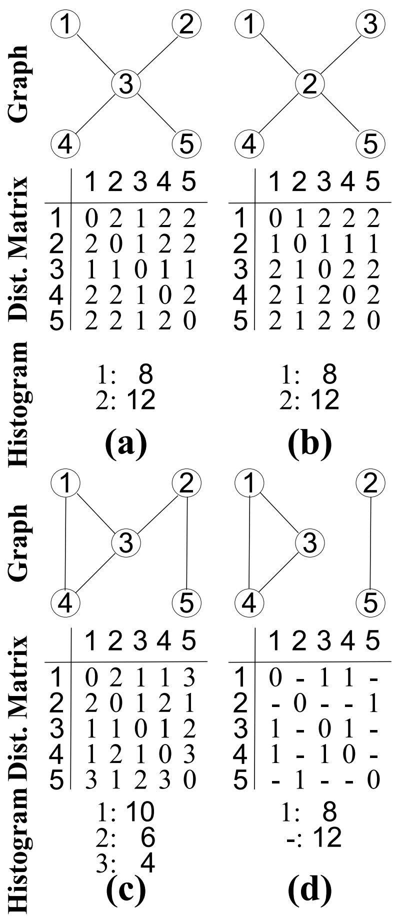
(c)

| | 1 | 2 | 3 | 4 | 5 |
|---|---|---|---|---|---|
| 1 | 0 | 0 | 1 | 1 | 0 |
| 2 | 0 | 0 | 1 | 0 | 1 |
| 3 | 1 | 1 | 0 | 1 | 0 |
| 4 | 1 | 0 | 1 | 0 | 0 |
| 5 | 0 | 1 | 0 | 0 | 0 |

(d)

| | 1 | 2 | 3 | 4 | 5 |
|---|---|---|---|---|---|
| 1 | 0 | 0 | 1 | 1 | 0 |
| 2 | 0 | 0 | 0 | 0 | 1 |
| 3 | 1 | 0 | 0 | 1 | 0 |
| 4 | 1 | 0 | 1 | 0 | 0 |
| 5 | 0 | 1 | 0 | 0 | 0 |

Fig. 1. An Example of the difficulties in Graph-related problems



(a)

| | 1 | 2 | 3 | 4 | 5 |
|---|---|---|---|---|---|
| 1 | 0 | 2 | 1 | 2 | 2 |
| 2 | 2 | 0 | 1 | 2 | 2 |
| 3 | 1 | 1 | 0 | 1 | 1 |
| 4 | 2 | 2 | 1 | 0 | 2 |
| 5 | 2 | 2 | 1 | 2 | 0 |

1: 8
2: 12

(b)

| | 1 | 2 | 3 | 4 | 5 |
|---|---|---|---|---|---|
| 1 | 0 | 1 | 2 | 2 | 2 |
| 2 | 1 | 0 | 1 | 1 | 1 |
| 3 | 2 | 1 | 0 | 2 | 2 |
| 4 | 2 | 1 | 2 | 0 | 2 |
| 5 | 2 | 1 | 2 | 2 | 0 |

1: 8
2: 12

(c)

| | 1 | 2 | 3 | 4 | 5 |
|---|---|---|---|---|---|
| 1 | 0 | 2 | 1 | 1 | 3 |
| 2 | 2 | 0 | 1 | 2 | 1 |
| 3 | 1 | 1 | 0 | 1 | 2 |
| 4 | 1 | 2 | 1 | 0 | 3 |
| 5 | 3 | 1 | 2 | 3 | 0 |

1: 10
2: 6
3: 4

(d)

| | 1 | 2 | 3 | 4 | 5 |
|---|---|---|---|---|---|
| 1 | 0 | - | 1 | 1 | - |
| 2 | - | 0 | - | - | 1 |
| 3 | 1 | - | 0 | 1 | - |
| 4 | 1 | - | 1 | 0 | - |
| 5 | - | 1 | - | - | 0 |

1: 8
-: 12

Fig. 2. Short-path graph kernels

graphs $G$, and $G'$ are calculated by using Floyd-Warshall algorithms. The shortest-path kernel is defined by comparing all pairs of shortest path lengths from $G$, and $G'$:

$$k(G,G') = \sum_{v_i,v_j \in G} \sum_{v'_i,v'_j \in G'} k_{length}(d(v_i,v_j), d(v'_i,v'_j)),$$

where $d(v_i, v_j)$ denotes the path distance of the shortest path between nodes $v_i, v_j$. $k_{length}$ is a kernel that compares the lengths of two shortest paths. In the case of a delta kernel [12],

$$k_{length}(d(v_i,v_j), d(v'_i,v'_j)) = \begin{cases} 1 & \text{if } \ d(v_i,v_j) = d(v'_i,v'_j) \\ 0 & \text{otherwise.} \end{cases}$$

Let $h(G)$ be a histogram of distances for all the shortest paths in a graph $G$. Each bin of $h(G)$ can be written as follows:

$$h(D,G) = |\{v_i, v_j \in G | d(v_i,v_j) = D\}|.$$

In the case of the delta kernel, the shortest-path kernel $k(G, G')$ can be rewritten as follows:

$$k(G,G') = \sum_D h(D,G) \cdot h(D,G'). \tag{1}$$

Figure 2 illustrates how graph kernels work. The graphs in this figure are the same as the ones in Figure 1. “Dist. Matrix” in Figure 2 represents the distance matrixes of shortest-path between corresponding nodes. “Histogram” means the histogram of distances mentioned in the previous paragraph. As depicted in this figure, rotated/reflected solutions indicate the same histograms. Meanwhile, solutions in Figure 2 (c), (d), which are similar in the adjacency matrix representation, show quite different histograms.

## III. Estimation of Distribution Algorithms

Estimation of Distribution Algorithms are a class of evolutionary algorithms which adopt the probabilistic models to reproduce individuals in the next generation, instead of conventional crossover and mutation operations. The probabilistic model is represented by conditional probability distributions for each variable. This probabilistic model is estimated from the genetic information of selected individuals in the current generation. Hence, the pseudo-code of EDAs can be written as Figure 3, where $D_l$, $D^s_{l-1}$, and $p_l(\mathbf{x})$ indicate the set of individuals at $l^{\text{th}}$ generation, the set of selected individuals at $l-1^{\text{th}}$ generation, and estimated probabilistic model at $l^{\text{th}}$ generation, respectively [1]. As described in this figure, the main calculation procedure of the EDAs is that (1) firstly, the $N$ selected individuals are selected from the population in the previous generation. (2) Secondly, the probabilistic model is estimated from the genetic information of the selected individuals. (3) A new population whose size is $M$ is then sampled by using the estimated probabilistic model. (4) Finally, the

```
Procedure Estimation of Distribution Algorithm
begin
  initialize $D_0$
  evaluate $D_0$
  until Stopping criterion is reached
    $D_l^s \leftarrow$ Select $N$ individuals from $D_{l-1}$
    $p_l(\mathbf{x}) \leftarrow$ Estimate the probabilistic model from $D_l^s$
    $D_l \leftarrow$ Sampling $M$ individuals from $p_l(\mathbf{x})$
    evaluate $D_l$
  end
end
```

Fig. 3. Pseudo-code of Estimation of Distribution Algorithms

new population is evaluated. (5) Steps (1)-(4) are iterated until stopping criterion is reached.

## IV. Proposed Method

Figure 4 depicts the procedure of the proposed method. This procedure is almost same as of the conventional EDAs mentioned in the previous section. One of the differences with the conventional EDAs is the use of graphs for the individual representation. Because of this difference, the estimation phase and the sampling phase are also different. Therefore, "individual representation," "estimation," and "sampling" are separately described in the following subsections. As the same as in the conventional EDAs, we assume that fitness is evaluated by user-defined fitness functions. The truncation selection is used as a selection method [1].

### A. Individual Representation

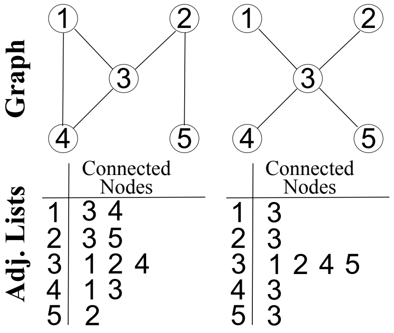
The proposed method employs the direct coding for representing individuals so that a graph indicates an individual. For an implementation issue, we can use either of the adjacency matrix data structure as in Figure 1 or the adjacency list data structure as in Figure 5. The propose method does not manipulate graphs by using crossover and mutation operations, but requires the data structures to add/remove edges and to calculate all the shortest-path distances. Hence, both the data structures can be used. We can use the adjacency matrixes for small/dense problems and the adjacency lists for large/sparse problems. For ease of implementation, we could use public domain softwares of graph libraries.

### B. Estimation of individual distributions

Suppose that $N$ denotes the number of the selected individuals at each generation, and the selected individuals are represented by $G_1, G_2, \ldots, G_N$. A kernel density function $f(G)$ can be defined as follows:

$$f(G) = \frac{1}{N \cdot p_n} \sum_i k(G, G_i) \qquad (2)$$

where $k$ denotes the kernel function in equation (1), and $p_n$ stands for a normalizing parameter which normalizes the magnitude of the kernel function to 1. By using equation (1), equation (2) can be rewritten as follows:



| | Connected Nodes |
|---|---|
| 1 | 3 4 |
| 2 | 3 5 |
| 3 | 1 2 4 |
| 4 | 1 3 |
| 5 | 2 |

| | Connected Nodes |
|---|---|
| 1 | 3 |
| 2 | 3 |
| 3 | 1 2 4 5 |
| 4 | 3 |
| 5 | 3 |

Fig. 5. Adjacency list data structure of graphs

$$\begin{aligned} f(G) &= \frac{1}{N \cdot p_n} \sum_i \sum_D h(D, G) \cdot h(D, G_i) \\ &= \sum_D h(D, G) \cdot \frac{1}{N \cdot p_n} \sum_i h(D, G_i) \end{aligned}$$

Therefore, the probability density estimation for the selected individuals in the proposed method is to sum up $h(D, G_i)$ of all the selected individuals for each possible distance $D$. Hence, the procedure of this is summarized as follows:

1) The distance matrixes of shortest-path for all the selected individuals are calculated.
2) A histogram of distances for all the selected individuals is constituted.

### C. Sampling of new individuals

The following sampling algorithm is used:

1) A graph $G'$ is copied from one of selected individuals.
2) $p_d$ percents edges in the $G'$ are removed.
3) All the shortest-path distances are calculated for the graph $G''$ in 2).
4) Adding edge one by one randomly: Let $G'''$ be the graph adding one edge from the graph $G''$.
5) If $f(G''') > f(G'')$ or with a small probability $p_m$, the addition in 4) is accepted: $G'' \leftarrow G'''$
6) If the number of iterations is less than twice of the number of removed edges in 2), go back to 4).

The reason why we adopt this approach, i.e., delete at first, then adding one by one, is to reduce the computational costs of calculating all the shortest-path distances. If we delete an edge, we need to re-calculate all the shortest-path distances. However, in the case of adding an edge, we can incrementally calculate all the shortest-path distances. The calculation costs for all the shortest-path distances and for this incremental update are $O(v^3)$ and $O(v^2)$.

## V. Experiments

### A. Experimental Results on Edge-max problems

This paper examined two simple problems: Edge-max problems and Edge-min problems. This subsection shows the

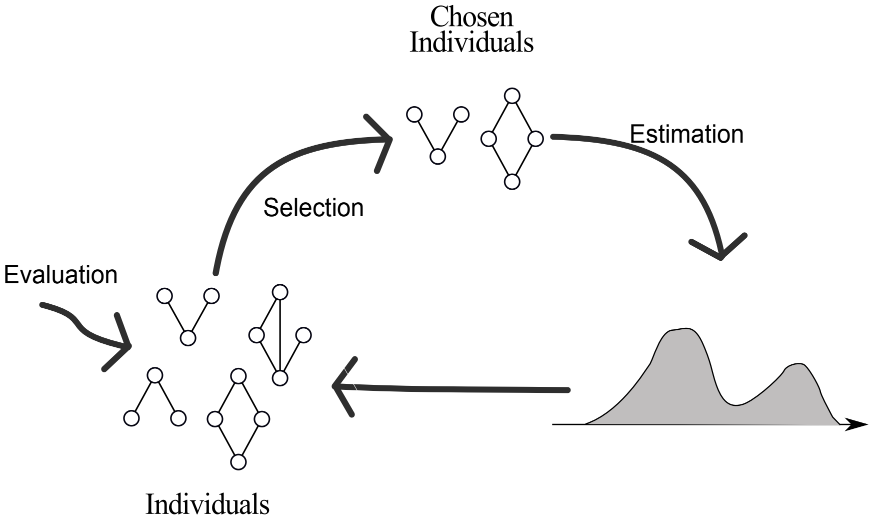


Fig. 4. Procedure of the proposed method

experimental results on the Edge-max problems. Edge-max problems are just like one-max problems used in binary Genetic Algorithms and binary Estimation of Distribution Algorithms. The fitness $f_{emax}$ of the Edge-max problems is defined as follows:

$$f_{emax}(G) = (\text{the number of edges in } G),$$

where $f_{emax}(G)$ should be maximized.

The number of selected individuals $N$ is set to be half of the number of individuals $M$. The truncation selection method, which selects the best $N$ individuals form $M$ individuals, to constitute the selected individuals, is adopted. The number $M$ of individuals in the population vary from 16 to 256.

We examined the Edge-max problems with ether of 10, 20, 30, or 40 nodes. The optimal values for these problems are 45, 190, 435, and 780, respectively. For comparison, one-max problems of 45, 190, 435, and 780 bits are examined. We employed UMDA and MIMIC as conventional EDAs [13], [14].

Figure 6 shows the experimental results of edge max problems: the averaged fitness of the best individuals over 30 runs. We examined two kinds of $p_m$ in subsection IV-C: 0.01 ("Proposed w high prob." in the legend in graphs) and $2/(v(v+1))$ ("Proposed w low prob."). For 10 and 20 nodes problems, the both proposed methods can find out optimal values. For 30 and 40 nodes problems, only the proposed method with high probability achieved to optimal solutions. In the case of UMDA and MIMIC, algorithms with larger population size can solve corresponding one-max problems.

### B. Experimental Results on Edge-min problems

As you may see from step 4) in subsection IV-C, the proposed methods tend to bias to adding edges. Hence, we introduced edge-min problems to confirm this. The fitness $f_{min}$ of the edge-min problems is the same as the one of the edge-max problems. However, this fitness function $f_{min}$ must be minimized. In this subsection, we did not examine UMDA and MIMIC since these algorithms are expected to show the same performance between one-max problems and zero-max problems.

Figure 7 shows the experimental results of edge min problems. The performances of the proposed methods are not optimal one. As mentioned in the head of this subsection, this would be caused by the bias to adding edges.

## VI. Conclusion

This paper proposed a novel Estimation of Distribution Algorithms with Graph Kernels. Section II showed the potential capability of the Graph Kernels in the case of evolutionary searches. Thanks to the Graph Kernels, new individuals in the proposed method can be sampled by referring to the feature space, i.e., histograms of short-path distances. Two simple problems were introduced: the edge-max problems and the edge-min problems. These problems revealed that 1) the sampling method in the proposed method should be improved, and 2) the proposed methods could show satisfiable performance in terms of that the proposed methods were not affected by the population size.

Future works are summarized as follows: As mentioned in the previous paragraph, the sampling method in the proposed method should be improved, where there are no biases to any direction. We should examine Gibbs sampling even if the computational costs in the edge deletions are expensive. In this paper, we only examined simple problems such that adding/removing edges are required. The proposed method can easily extend to adding/removing nodes or nodes are distinguished with labels. The latter case can be found in practical situations such as graph mining from chemical compounds, social network and so on. We can slightly modify the definition of the distance matrixes and histograms of the short-path distances, where distinguished nodes are not treated as the same type.

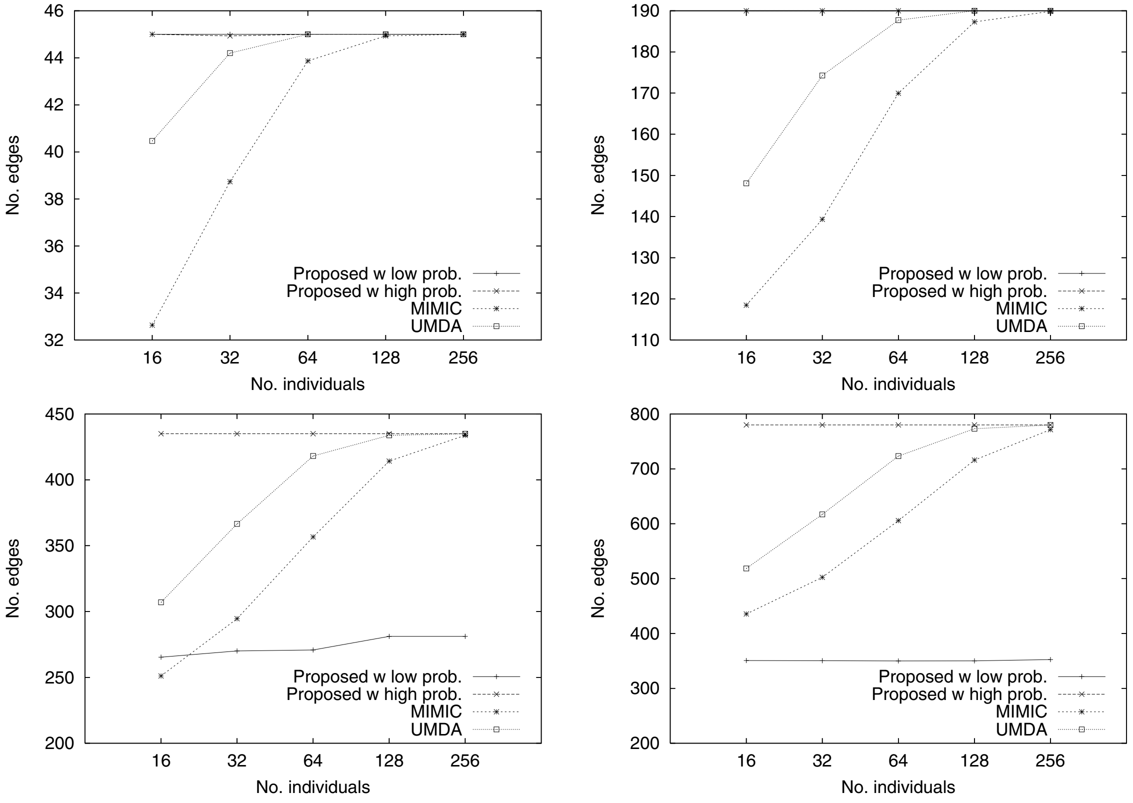


Fig. 6. Experimental results of edge max problems: the average of the best individuals over 30 runs: results of 10 nodes (upper left); 20 nodes (upper right); 30 nodes (lower left); 40 nodes (lower right)

## Acknowledgment

This work was partially supported by the Grant-in-Aid for Scientific Research (B) and the Grant-in-Aid for Young Scientists (B) of MEXT, Japan (21700254, 23700267).

## References

[1] P. Larrañaga, and J.A. Lozano, *Estimation of Distribution Algorithms*, Kluwer Academic Publishers, 2003

[2] M. Pelikan, D. E. Goldberg, and E. Cantú-paz. Boa: The bayesian optimization algorithm. In *Proc. of the 1999 Genetic and Evolutionary Computation Conf.*, pages 525–532, 1999.

[3] P. A. N. Bosman, J. Grahl, and D. Thierens. Enhancing the performance of maximum-likelihood gaussian EDAs using anticipated mean shift. In *Proc. 10th International Conference on Parallel Problem Solving from Nature - PPSN X*, pages 133–143, 2008.

[4] H. Handa, *EDA-RL: estimation of distribution algorithms for reinforcement learning problems,* Proc. GECCO '09: Proceedings of the 11th Annual conference on Genetic and evolutionary computation, pp.405–412, 2009.

[5] E. Alba, and F. Chicano, *ACOhg: Dealing with huge graphs*, Proc. the 2007 ACM Genetic and Evolutionary Conference, pp.10–17, 2007.

[6] F. Chicano, and E. Alba, *Ant colony optimization with partial order reduction for discovering safety property violations in concurrent models*, Information Processing Letters, Vol. 106, No. 6, pp.221–231, 2008.

[7] J. McDermott,U.-M. O'Reilly, *An executable graph representation for evolutionary generative music*, Proc. the 2011 ACM Genetic and Evolutionary Conference, pp.403–412, 2011.

[8] T.E. Lewis, and G.D. Magoulas, *Strategies to minimise the total run time of cyclic graph based genetic programming with GPUs*, Proc. the 2009 ACM Genetic and Evolutionary Conference, pp.1379–1386, 2009.

[9] S. Shirakawa, and T. Nagao, *Graph structured program evolution with automatically defined nodes*, Proc. the 2009 ACM Genetic and Evolutionary Conference, pp.1107–1115, 2009.

[10] S. Mabu, K. Hirasawa, and J.Hu, *A Graph-Based Evolutionary Algorithm: Genetic Network Programming (GNP) and Its Extension Using Reinforcement Learning*, Evolutionary Computation, Vol.15, No.3, pp.369–398, 2007.

[11] H. Kashima, K. Tsuda, and A. Inokuchi, *Marginalized Kernels Between Labeled Graphs*, Proc. 20th International Conference on Machine Learning, pp.321–328, 2003.

[12] K.M. Borgwardt,and H.-P. Kriegel, *Shortest-path kernels on graphs*, Proc. 5th International Conference Data Mining, 2005.

[13] H. Mühlenbein, G. Paaß, G., *From Recombination of genes to the estimation of distributions I. Binary parameters*, Parallel Problem Solving from Nature – PPSN IV, pp.178–187, 1996.

[14] J.S. De Bonet, *et al. MIMIC: Finding optima by estimating probability densities*, Advances in Neural Information Processing Systems 9, pp.424–430, 1996.

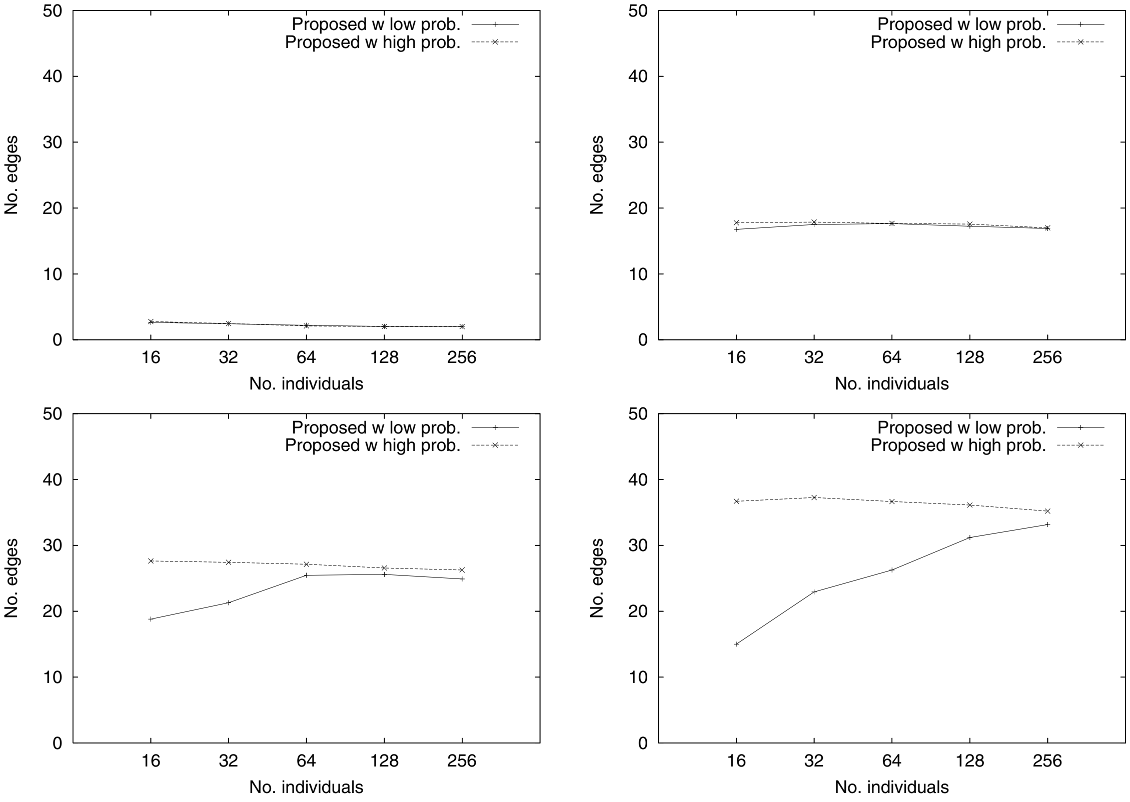


Fig. 7. Experimental results of edge min problems: the average of the best individuals over 30 runs: results of 10 nodes (upper left); 20 nodes (upper right); 30 nodes (lower left); 40 nodes (lower right)