A New Technique for DSMGA-II

Author
Address
Address
Address
email

Author
Address
Address
Address
email

Author
Address
Address
Address
email

Author
Address
Address
Address
email

ABSTRACT

DSMGA-II is a novel genetic algorithm, capable of solving optimization problems via exploiting subproblem structures. In terms of number of function evaluations (NFE), it outperforms LT-GOMEA and hBOA on multiple trap problems, NK-landscape problems with multiple degrees of overlapping, 2D Ising spin-glass problems, and MAX-SAT problems. In this paper, we aim to improve the accuracy of model building and the efficiency of model selection by proposing some new techniques for DSMGA-II operators. Experiment results show that our new techniques can reduce up to 20% of NFE compared with the original DSMGA-II.

CCS Concepts

•**Machine learning approaches** → **Maximum entropy modeling; Genetic algorithms;** *Mixture models;*

Keywords

Genetic Algorithm; Estimation-of-Distribution Algorithm; Linkage Learning; Model Building

1. INTRODUCTION

Dependency structure matrix genetic algorithm-II (DSMGA-II) is a novel model building GA proposed by Hsu and Yu in 2015 [4]. Based on the dependency structure matrix (DSM), a new linkage model, called the incremental linkage set (ILS), is adopted in DSMGA-II to provide potential models for mixing. Restricted mixing and back mixing are the two major operators of DSMGA-II. They are the keys to significantly reduce the number of function evaluations (NFE) compared with other optimal mixing operators. Experiment results show that DSMGA-II requires less NFE than some cutting-edge evolutionary algorithms, such as LT-GOMEA [1] and hBOA [6], on various benchmark problems.

However, there is still room for improvement with model building in DSMGA-II. In this paper, we consider building a two-edged approximation maximum-weight connected subgraph (AMWCS) as a more effective linkage model. We also demonstrate how early-stopping methods enable more efficient model selection during mixing, resulting in less NFE.

This paper is organized in six parts. Section 2 introduces the original DSMGA-II schema and the testing problems. Section 3 describes the mechanism of early stopping technique. Section 4 introduces the 2-edged linkage model in detail. Section 5 shows the experiment results and section 6 gives the summary and conclusion.

GECCO'17, July 15-19, 2017, Berlin, Germany

ACM ISBN TBA.

DOI: 10.1145/1235

2. DSMGA-II

In this Section, we first introduce the framework of DSMGA-II and the concept of incremental linkage set. After that, we give details of restricted mixing and back mixing, which are the kernel operators of DSMGA-II.

2.1 Framework of DSMGA-II

DSMGA-II consists of four major steps: population initialization, linkage model building, restricted mixing and back mixing.

First, DSMGA-II randomly initializes a population. Then, in order to enhance the quality of pairwise linkage model and reduce the noise in the population, DSMGA-II performs a bit-flipping greedy hill climbing (GHC) on each chromosome. For each randomly initialized chromosome, GHC randomly flips each bit in the chromosome and evaluates its fitness. If the fitness improves, the chromosome accepts the change. Otherwise, the flipped bit is restored. Pairwise linkage detection was adopted in a later version of LTGA [7] and DSMGA [11] due to its resistance to sampling noise. Performing GHC before linkage model building can further ensure the pairwise linkage information between genes, by ruling out trivial cases that can be solved without linkage information.

After initializing the population, only the selected chromosomes are used to build the linkage model. The chromosomes are chosen by a tournament selection withãĂĂselection pressure as 2, suggested in [13]. DSMGA-II adopts mutual information as pairwise linkage measure and stores

Algorithm 1: DSMGA-II

```
$P$: population, $S$: selected population,
$s$: selection pressure, $R$: constant,
$DSM$: dependency structure matrix, $M$: mask
input  : $l$: problem size, $p$: population size
output: $P_{best}$
$P \leftarrow$ PopulationInitialization($l$, $p$)
$P \leftarrow$ GHC($P$)
while not ShouldTerminate do
    $S \leftarrow$ TournamentSelection($P$, $s$)
    $DSM \leftarrow$ UpdateMatrix($S$)
    for $k \leftarrow 1$ to $R$ do
        $I \leftarrow$ random permutation from 1 to $p$
        for $i \in I$ do
            $P_i$, $M \leftarrow$ RestrictedMixing($P_i$)
            if $M \neq \emptyset$ then
                $P \leftarrow$ BackMixing($P_i$, $M$)
return best instance in $P$
```

the linkage information in a DSM. The DSM is updated only once in each generation in order to prevent overfitting from frequent model building. Notice that tournament selection is only performed to choose chromosomes for model building instead of updating the population. The following steps of restricted mixing and back mixing proceed with the original population.

Before mixing, DSMGA-II builds the ILS with the linkage information stored in DSM. The ILS is a set of models indicating possible subproblem structures for restricted mixing. In restricted mixing, the receiver tries to flip the bits within the model. If the fitness does not decrease, the receiver becomes the donor in back mixing and the new pattern is tried on all chromosomes in the population. The population is randomly shuffled before restricted mixing so that each chromosome takes turns acting as the receiver in restricted mixing and the donor in back mixing. The pseudo code of DSMGA-II is given in Algorithm 1. The detailed implementations of the ILS and the mixing operators are described in section 2.2 and 2.3.

2.2 Incremental Linkage Set

The DSM is an adjacent matrix representing the dependency between two variables, where each entry stores the pairwise information between two bits. In DSMGA-II, pairwise dependencies are measured by mutual information [5]. Formally, the mutual information of two random variable X and Y can be defined as:

$$I(X;Y) = \sum_{x \in X} \sum_{y \in Y} p(x,y) log \frac{p(x,y)}{p(x)p(y)}$$

where x and y are the outcomes of X and Y. In pairwise information, the random variables X and Y follows the (binomial) Bernoulli distribution with support {0, 1}. p(x) represents the portion of a bit having value 1 in the population. Therefore, the linkage measure can be further derived as:

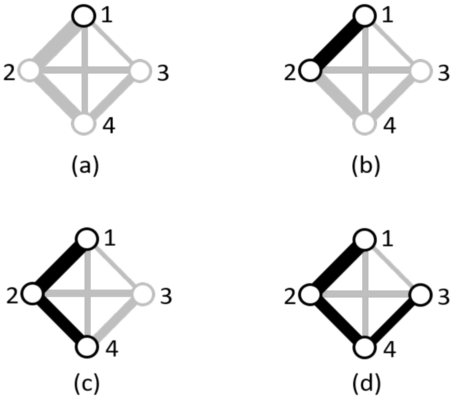

Figure 1: (a) to (d) shows the AMWCS in each iteration. The nodes represent variables and the gray edges represent the dependency measure between pairs. The larger the measure is, the wider the edge is. Black edges represent the determined AMWCS.

$$\begin{aligned} I(X;Y) = P_{00} log \frac{P_{00}}{P_{0*}P_{*0}} + P_{11} log \frac{P_{11}}{P_{1*}P_{*1}} \\ + P_{01} log \frac{P_{01}}{P_{0*}P_{*1}} + P_{10} log \frac{P_{10}}{P_{1*}P_{*0}} \end{aligned} \quad (1)$$

If X and Y are independent, then I(X; Y) equals to 0. Otherwise, the mutual information indicates how much the corresponding bits differ from most of the chromosomes in current population. The value of mutual information between two bits corresponds to the probability of two bits being in the same building-block. Therefore, DSMGA-II constructs building-blocks by clustering the DSM. Instead of using traditional hierarchical clustering algorithms proposed in [9] or the average linkage clustering technique proposed in [10], DSMGA-II adopts a specific subgraph called approximation maximum-weight connected subgraph (AMWCS). DSMGA-II constructs AMWCS from a certain bit, and iteratively adds one single bit into the graph. For example, consider a graph $G = (V, E), V = \{v1, v2, v3, v4\}$, the AMWCS $G_s = (V_s, E_s), V_s = \{v1, v2\}$, and the candidate nodes $N = \{v3, v4\}$. The goal is to find a node from the candidate set with a greedy approach that maximizes the weight of AMWCS after insertion. The equation can be defined as :

$$index = argmax_{i \in 3,4} \sum_{j=1}^{2} I(v_i; v_j) \quad (2)$$

The ILS is an AMWCS constructed with a randomly chosen start node. For example, consider a problem with problem size 4, and its linkage model, i.e. the DSM, shown in Figure 1. The first step of constructing ILS is to choose a start node. Here, the index 1 was randomly chosen from $\{1, 2, 3, 4\}$. Then, the rest of nodes are inserted into AMWCS following the sequence $\{1, 2, 4, 3\}$ that gives the maximum weight. After 3 iterations, the ILS is constructed as an ordered set $L = \langle\{1\}, \{1, 2\}, \{1, 2, 4\}, \{1, 2, 4, 3\}\rangle$. Each component of the ILS is called a mask, which is a potential building-block in the mixing stage.

Algorithm 2: Restricted Mixing

```
$V$: candidate vertices set, $V_S$: $AMWCS$ vertices set,
$ILS$: incremental linkage set, $f$: evaluation function,
$P$: population, $l$: problem size,
$T$: trial solution, $M$: mask,
$R_M$: pattern of $R$ extracted by $M$,
$R_M{}'$: complement pattern of $R_M$
input  : $R$: receiver
output: $R$: reciever, $M$: mask

$V \leftarrow \{1, 2, ..., l\}$
$V_S \leftarrow \{v \mid$ a random vertex $v \in V\}$
remove $v$ from $V$
while $|V| \geq l/2$ do
    $ILS \leftarrow ILS \cup V_S$
    $V_S \leftarrow V_S \cup \{v \mid$ the nearest vertex $v \in V\}$
    remove $v$ from $V$

for $i \leftarrow 1$ to $|ILS|$ do
    $M \leftarrow ILS_i$
    if $R_M{}' \subset P$ then
        $T \leftarrow R$
        $T_M \leftarrow R_M{}'$
        if $f(T) \geq f(R)$ $and$ $T \notin P$ then
            $R \leftarrow T$
            return ($R$, $M$)

return ($R$, $\emptyset$)
```

2.3 Restricted Mixing and Back Mixing

2.3.1 Optimal Mixing

Unlike canonical genetic algorithms that generate offsprings by recombining parent solutions, DSMGA-II extends the the idea of optimal mixing (OM) [10] with two new mixing operators: restricted mixing and back mixing. OM evaluates a chromosome during recombination. With the information of fitness before and after mixing, the chromosome only accepts the change if its fitness improves. Thus, a noise-free decision-making can be achieved with a much smaller population size [2]. Given overlapping building blocks, OM is also capable of solving problems with overlapping structures efficiently, since it acts like building-block wise local search.

2.3.2 Restricted Mixing

In in each iteration of restricted mixing, a receiver is randomly picked from the population, and the building-blocks are provided by a mask which is chosen from the ILS. Each mask is a set of indexes that indicates which bits should be flipped together during mixing operations. All masks in the ILS must go through a supply check to make sure that the complement pattern of the receiver exists in the population. This way, flipping the bits is equivalent to recombining the receiver with the complement pattern. If a mask failed the supply check, all masks with larger size behind it in the ordered ILS are discarded. After supply check, the receiver starts with the smallest subset in the mask, and flips the bits within the subset. If the fitness decrease after recombination, the receiver continues to try bit-flipping with the next (smallest) building-block. Otherwise, the pattern is accepted and restricted mixing terminates. The receiver then becomes the donor of the new pattern for rest the population in back mixing.

Algorithm 3: Back Mixing

```
$P$: population, $f$: evaluation function,
$T$: trial solution, $E$: set of candidate solutions
input  : $D$: donor, $M$: mask
output: $P$: population

$improved \leftarrow false$
for $j \leftarrow 1$ to $|P|$ do
    $T \leftarrow P_j$
    for $m \in M$ do
        $T_m \leftarrow D_m$
    if $f(T) \geq f(P_j)$ then
        $P_j \leftarrow T$
        $improved \leftarrow true$
    else
        if $f(T) = f(P_j)$ then
            $E \leftarrow E \cup \{T\}$

if $not$ $improved$ then
    accept all solutions in $E$
return $P$
```

The idea behind restricted mixing is building-block supply [2]. We believe all the optimal subsolution fragments should exist in the current population, which had been initialized with a sufficient population size. Therefore, given the correct building block with a proper receiver, restricted mixing conducts optimal mixing between the receiver and the chromosome with the complementary optimal pattern. The pseudo-code for restricted mixing is provided in Algorithm 2.

2.3.3 Back Mixing

In back mixing, every chromosome in the population is mixed with the pattern accepted in restricted mixing. Chromosomes are set to accept the new pattern only with strict fitness improvement by default. However, if no chromosome fitness improves in the whole population, then the mixing which results in equal fitness is also allowed. The back mixing acceptance criteria is set differently from restricted mixing in order to tackle real-world problems with plateaus and basins. Many operators, such as the forced improvement (FI) [1] have been developed to deal with multiple equal-quality solutions. Strict mixing improvement criteria often causes numerous evaluations to jump out of the plateaus. On the other hand, allowing all chromosomes to accept the patterns with equal fitness results in a strong drifting effect. Back mixing handles the diversity issue with a default strict-improvement criteria. When no improvement occurs, it switch to equal-acceptance criteria to reduce unnecessary evaluations on plateaus. The empirical experiment results suggest that back mixing is able to deal with both plateaus and diversity issues. The pseudo-code for restricted mixing is provided in Algorithm 3.

3. MODEL SELECTION

3.1 Early Stopping

Break prevents possible cross over and limit attempts for useless function call. Demo by table, showing how many attempt is useless after isInP and how often does cross competition occur

Parameters	Continue	Break
Success rate	1,000	1000
Cross competition occurrence	1,000	1000
NFE mean	1,000	1000
NFE variance	1,000	1000

Table 1: Success rate comparison

4. THE NEW LINKAGE MODEL

This section introduces the concept and the flow of AMWCS construction. In the original AMWCS, nodes are connected with only one linkage. We call this technique the 1-edge AMWCS. In this section, we first provide a new linkage graph called the 2-edge AMWCS which gives more precise building-block models for different chromosomes.. Then we describe supply bound, the theoretical support behind this measure.

4.1 2-edge AMWCS

In the scheme of 2-edge AMWCS, the dependency measurement between two bits is different from the 1-edge graph. The linkage measure equation is divided into two parts as following :

$$\begin{aligned} I(00 \cup 11) &= P_{00} log \frac{P_{00}}{P_{0*}P_{*0}} + P_{11} log \frac{P_{11}}{P_{1*}P_{*1}} \\ I(01 \cup 01) &= P_{01} log \frac{P_{01}}{P_{0*}P_{*1}} + P_{10} log \frac{P_{10}}{P_{1*}P_{*0}} \end{aligned} \quad (3)$$

The reason for dividing the mutual information into two groups is to protect the building-block pattern. For example, if we divide the linkage measure into two parts, $I(01 \cup 00)$ and $I(10 \cup 11)$. For building-blocks with linkage $I(01 \cup 00)$, the pattern 01 might be flipped to 10 and the pattern 00 might be flipped to 11 during mixing. The resulting patterns contradict with the linkage assumption and destroy the schema between these two bits. Therefore, the linkage measure is divided into $I(0 \cup 11)$ and $I(01 \cup 10)$, for these patterns are the complements of each other and cannot be interrupted during mixing.

In the 2-edge AMWCS scheme, the equation for finding a node that maximize the weight of AMWCS is the same as the equation in the 1-edge scheme. However, unlike the 1-edge AMWCS that view all patterns of a building-block equally, the allele of receivers are taken into account in the 2-edge linkage model. The rule of edge selection in the 2-edge AMWCS is as follows :

$I(X;Y) = I(00 \cup 11)$, if reciever's allel pattern is 00 or 11

$I(X;Y) = I(01 \cup 10)$, if reciever's allel pattern is 01 or 10

For instance, consider a problem with length 4, allele $\{1, 0, 1, 1\}$ and the index 2 was randomly chosen from $\{1, 2, 3, 4\}$ at first illustrated in Figure 2. Although there is a strong edge(solid line) between node 2 and 3, the edge is disconnected due to the rule constraint, so node 1 is picked in this iteration because of the thickest dotted line. After 2 iterations, the inserted sequence of nodes into AMWCS is $Q = 2, 1, 4, 3$, and the incremental linkage sets is $L = \langle\{2\}; \{2, 1\}; \{2, 1, 4\}; \{2, 1, 4, 3\}\rangle$.

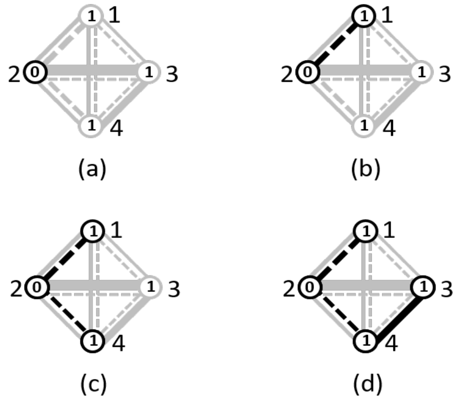

Figure 2: (a) to (d) shows the AMWCS in each iteration. The nodes represent variables, the values in the variables represent the alleles of the nodes, the solid lines represent the 00-11 edge, the dotted lines represent the 01-10 edge, and the gray edges represent the dependency measure between pairs. The larger the measure is, the wider the edge is. Black nodes and edges represent the determined AMWCS.

The advantage of 2-edge AMWCS is exactly that the scheme considers the receiver's allele in the mixing stage. For instance, consider an optimal subsolution 111, and 1-edge AMWCS detect the strong linkage among these three bits, then build the ILS as $\langle\{1\}, \{1, 2\}, \{1, 2, 3\}...\rangle$. Although the mask $\{1, 2, 3\}$ can flip the pattern 000 to 111, however, if the receiver's allele pattern is 010, no mask above can flip it to 111. So even 1-edge AMWCS already detected the correct model, it does not flip efficiently if the scheme not to take the receiver's allele into consideration. 2-edge AMWCS can handle this problem with one of its property: The receiver's allele is inclined to become the same specific pattern which account for the higher rate of the population, and we believe that the high ratio pattern implies the optimal subpattern in average. The reason for that is one of the characteristic of the mutual information : If we divided the mutual information into two parts, then the part with the high-ratio pattern will be positive, and the other part usually be negative. Under this circumstance, for the example above, if the pattern 111 gets higher ratio in the population, the $I(00 \cup 11)$ among these nodes are positive, and the $I(01 \cup 10)$ among them are negative. So if the receiver's pattern is 010, and AMWCS start with node 1. Because the edge between node 1 and node 2 is negative, the AMWCS will not add the node 2 into the candidate set in this iteration, then the ILS might be $\langle\{1\}, \{1, 3\}, ...\}\rangle$ rather than $\langle\{1\}, \{1, 2\}, \{1, 2, 3\}, ...\rangle$. And the mask $\{1, 3\}$ can flip the pattern 010 to the pattern 111 which we want.

4.2 Supply bound

Although 2-edge AMWCS performs more efficiently in linkage building, there is still a problem in restricted mixing. In the restricted mixing stage, receiver keeps trying the masks until the fitness gets better or there is no supply. The problem is that the bit-pattern grouped together are complement to the other, so if the AMWCS add the node into the candidate set, it means the pattern and the complement pattern account for high ratio of the population, and it leads higher probability to pass the supply check. But in the early stage, because the information for model building is not clear and not accurate yet, the restricted mixing usually fails. Due to the much longer length of supply, 2-edge AMWCS wastes much more NFEs for the restricted mixing fail in the early stage. The experiment(Table 2) shows that 1-edge AMWCS performs accurately in finding the length of model, and 2-edge AMWCS gives rise to about double size supply.

Parameters	Continue	Break
Success rate	1,000	1000
Cross competition occurrence	1,000	1000
NFE mean	1,000	1000
NFE variance	1,000	1000

Table 2: Success rate comparison

To handle the problem of 2-edge AMWCS supply overfitting, we decide to adopt 1-edge AMWCS for the supply check. The concept is that 2-edge AMWCS performs more accurately in linkage building, and 1-edge AMWCS is good at model size finding. Then why not we take advantage of these techniques? The pseudocode of the restricted mixing in the 2-edge AMWCS is given in algorithm 4.

```
Algorithm 4: Restricted Mixing
$V$: candidate vertices set, $V_S$: $AMWCS$ vertices set,
$ILS$: incremental linkage set, $f$: evaluation function,
$P$: population, $l$: problem size,
$T$: trial solution, $M$: mask,
$R_M$: pattern of $R$ extracted by $M$,
$R_M{}'$: complement pattern of $R_M$
input  : $R$: receiver
output: $R$: reciever, $M$: mask
$V \leftarrow \{1, 2, ..., l\}$
$V_S \leftarrow \{v \mid$ a random vertex $v \in V\}$
remove $v$ from $V$
while $|V| \geq l/2$ do
    $ILS \leftarrow ILS \cup V_S$
    $V_S \leftarrow V_S \cup \{v \mid$ the nearest vertex $v \in V\}$
    remove $v$ from $V$
for $i \leftarrow 1$ to $|ILS|$ do
    $M \leftarrow ILS_i$
    if $R_M{}' \subset P$ then
        $T \leftarrow R$
        $T_M \leftarrow R_M{}'$
        if $T \in P$ then
            return ($R$, $\emptyset$)
        if $f(T) \geq f(R)$ then
            $R \leftarrow T$
            return ($R$, $M$)
return ($R$, $\emptyset$)
```

5. EXPERIMENT RESULTS

In this section, we first describe the six benchmark problems. After that, the experiment setup is given. Finally, we illustrate the experiment results of the original DSMGA-II, the improved DSMGA-II, LT-GOMEA and hBOA.

5.1 Test Problems

Six types of linkage benchmark problems are used in this paper, including four classical linkage-underlying problems and two real-world problems. These benchmark problems each covers different aspects and characteristics of real-world problems. Detail descriptions of the six benchmark problems are as follows.

5.1.1 *Concatenated trap*

The concatenated trap is composed of m additively separable trap function, each with k variables [3]. It is well known that in order to solve trap problems, the underlying structure must be detected and preserved during mixing [8]. The fitness function is described as follows:

$$f^{trap}_{m,k}(x) = \sum_{i=1}^{m} f^{trap}_{k}\left(\sum_{j=i\cdot k-k+1}^{i\cdot k} x_j\right)$$

where

$$f^{trap}_{k}(u) = \begin{cases} 1, & \text{if } u = k \\ \frac{k-1-u}{k}, & \text{otherwise} \end{cases}$$

5.1.2 *Cyclic trap*

The cyclic trap consists of overlapping trap functions with wraparound [12]. The fitness function is described as follows:

$$f^{cyclic}_{m,k}(x) = \sum_{i=1}^{m} f^{trap}_{k}\left(\sum_{j=i\cdot(k-1)-k+2}^{i\cdot(k-1)+1} x_j\right)$$

where

$$f^{cyclic}_{m,k}(x) = \sum_{i=1}^{m} f^{trap}_{k}\left(\sum_{j=i\cdot(k-1)-k+2}^{i\cdot(k-1)+1} x_j\right)$$

and

$$x_j = x_{j-l}, \text{if } l < j \leq 2l$$

Given a 12-bit cyclic trap with the size of subfunction $k = 5$, the fitness of identifying one correct subsolution, e.g. 111110000000 with $fitness = 2.2$, is lower than having all incorrect subsolutions, e.g. 000000000000 with $fitness = 2.4$. Therefore, the linkage model must not only identify the structure of subsolutions, but also take different scenarios of subsolution combinations into account for recombination to succeed.

5.1.3 *Folded trap*

We use the bipolar deceptive function with the subsolution size k = 6. Each subsolution contains two global optima and lots of local optima. It is one of the multiple variants of NK-landscape problems described in [3]. The performance depends heavily on the ability to reduce unnecessary exploration of plateaus, due to the symmetric characteristics of the trap function. The fitness function is described as follows:

$$f^{folded}_{m,k=6}(x) = \sum_{i=1}^{m} f^{folded}_{k=6}\left(\sum_{j=i\cdot k-k+1}^{i\cdot k} x_j\right)$$

where

$$f_{k=6}^{folded}(u) = \begin{cases} 1, & \text{if } |u = 3| = 3 \\ 0.8, & \text{if } |u = 3| = 0 \\ 0.4, & \text{if } |u = 3| = 1 \\ 0, & \text{if } |u = 3| = 2 \end{cases}$$

5.1.4 *NK-landscape*

The NK-landscape functions are composed of overlapped, randomly generated sub-functions [8]. There are three parameters controlling the fitness function: l is the problem size, k is the number of neighbors of one gene, and so is the step size, i.e. the offset of two adjacent sub-functions. The fitness function is given as follows:

$$f_{l,k,s}^{NK}(x) = \sum_{i=0}^{(l-k-1)/s} f_{k,i}^{subNK}(x_{i\cdot s+1}, x_{i\cdot s+2}, ...x_{i\cdot s+k+1})$$

5.1.5 *Ising spin-glass*

The Ising spin-glass gives a set of variables in one of the two states $\{+1, -1\}$. For each pair of neighboring spins i and j, there exists a coupling constant Jij. The goal is to find a combination of states that minimizes the fitness function described as follows:

$$f_n^{spin}(x) = -\sum_{i,j=0}^{n} x_i x_j J_{ij}$$

5.1.6 *MAX-SAT*

The Maximum Satisfiability problem (MAX-SAT) consists of a series of *logicaland* clauses. Each clause contains a series of *logicalor* variables. It is a classical NP-complete problem. The fitness function is described as follows:

$$F = \bigwedge_{i=1}^{m} \left(\bigvee_{j=1}^{k_i} l_{ij} \right)$$

where m is the number of clauses, k_i is the number of literals in the i-th clause and l_{ij} is the j-th literal in the i-th clause. Here, we use the Uniform Random-3-SAT instances from SATLIB [1] with all satisfiable clauses.

5.2 Experiment Setup

Since DSMGA-II is based on the idea of OM, it is necessary to compare DSMGA-II with GOMEAs.

Describe sweep, 10 hits and the average of 100 instances.

5.3 Results

6. CONCLUSIONS

Our new techniques constructs a more precise linkage model that gives building-blocks customized for each receiving chromosome during mixing. We also show that early-stopping technique can reduce the probability of cross-competition, allowing DSMGA-II to solve the benchmark problems with a smaller population and less NFE. The improved DSMGA-II decrease NFE by up to 20% compared with the original

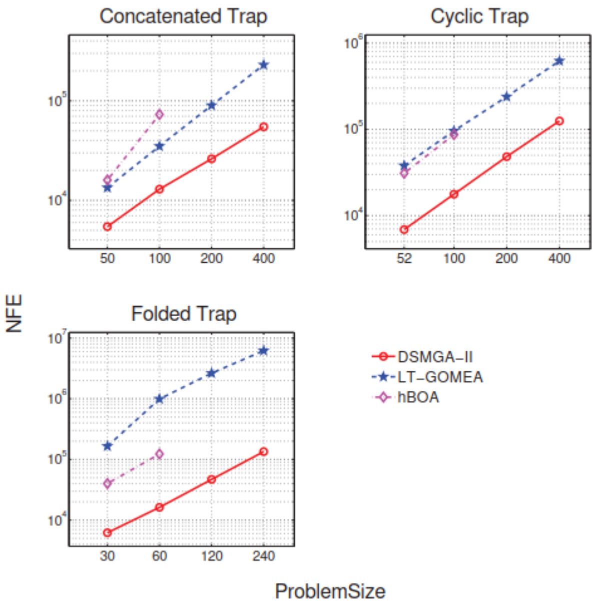

Figure 3: Scalability of DSMGA-II, LT-GOMEA and hBOA on the problems of deceptive variants.

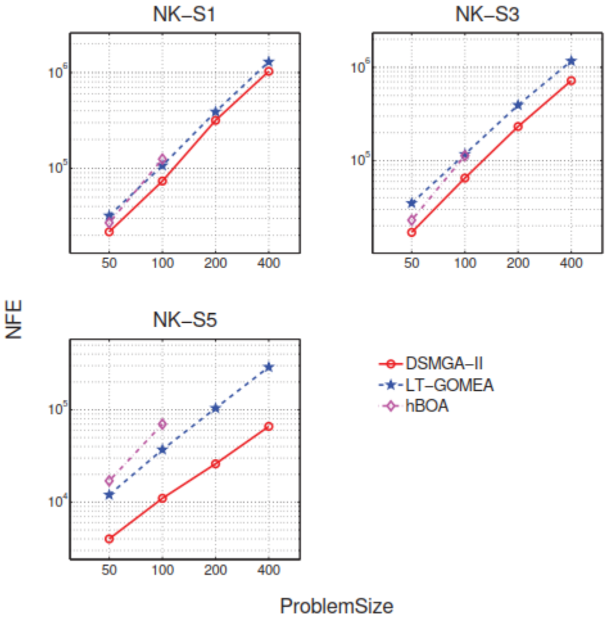

Figure 4: Scalability of DSMGA-II, LT-GOMEA and hBOA on NK-landscape problems with various degrees of overlapping.

[1] http://homepages.cwi.nl/∼bosman/source_code.php

Problems	Original	Improved	Ratio
Concatenated trap,*l=400*	1,000	1000	1%
Cyclic trap, $l = 400$	1,000	1000	1%
Folded trap, $l = 480$	1,000	1000	1%
NK-S1, $l = 400$	1,000	1000	1%
NK-S3, $l = 400$	1,000	1000	1%
NK-S5, $l = 400$	1,000	1000	1%
Ising spin-glass, $l = 400$	1,000	1000	1%
MAX-SAT, $l = 200$	1,000	1000	1%

Table 3: Required NFE of DSMGA-II for the largest test problems

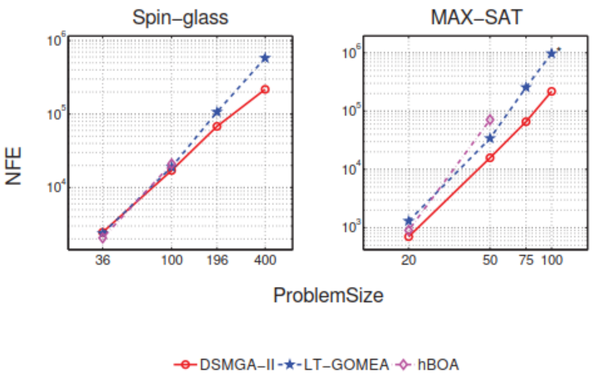

Figure 5: Scalability of DSMGA-II, LT-GOMEA and hBOA on Spin-glass and MAX-SAT (*LT-GOMEA fails to reach the global optima for two instances with $l = 100$ on MAX-SAT).

version. It also outperforms LT-GOMEA and hBOA on multiple benchmark problems.

7. ACKNOWLEDGMENTS

The authors would like to thank the support by Ministry of Science and Technology in Taiwan under Grant No. MOST 103-2221-E-002-177-MY2-1.

8. REFERENCES

[1] P. A. Bosman and D. Thierens. Linkage neighbors, optimal mixing and forced improvements in genetic algorithms. In *Proceedings of the 14th annual conference on Genetic and evolutionary computation*, pages 585–592. ACM, 2012.

[2] D. E. Goldberg, K. Deb, and J. H. Clark. Genetic algorithms, noise, and the sizing of populations. *COMPLEX SYSTEMS*, 6:333–333, 1992.

[3] D. E. Goldberg, K. Deb, and J. Horn. Massive multimodality, deception, and genetic algorithms. *Urbana*, 51:61801, 1992.

[4] S.-H. Hsu and T.-L. Yu. Optimization by pairwise linkage detection, incremental linkage set, and restricted/back mixing: Dsmga-ii. In *Proceedings of the 2015 Annual Conference on Genetic and Evolutionary Computation*, pages 519–526. ACM, 2015.

[5] S. Kullback and R. A. Leibler. On information and sufficiency. *The annals of mathematical statistics*, 22(1):79–86, 1951.

[6] M. Pelikan and D. E. Goldberg. Hierarchical boa solves ising spin glasses and maxsat. In *Genetic and Evolutionary Computation Conference*, pages 1271–1282. Springer, 2003.

[7] M. Pelikan, M. W. Hauschild, and D. Thierens. Pairwise and problem-specific distance metrics in the linkage tree genetic algorithm. In *Proceedings of the 13th annual conference on Genetic and evolutionary computation*, pages 1005–1012. ACM, 2011.

[8] M. Pelikan, K. Sastry, D. E. Goldberg, M. V. Butz, and M. Hauschild. Performance of evolutionary algorithms on nk landscapes with nearest neighbor interactions and tunable overlap. In *Proceedings of the 11th Annual conference on Genetic and evolutionary computation*, pages 851–858. ACM, 2009.

[9] D. Thierens. The linkage tree genetic algorithm. In *International Conference on Parallel Problem Solving from Nature*, pages 264–273. Springer, 2010.

[10] D. Thierens and P. A. Bosman. Optimal mixing evolutionary algorithms. In *Proceedings of the 13th annual conference on Genetic and evolutionary computation*, pages 617–624. ACM, 2011.

[11] T.-L. Yu, D. E. Goldberg, A. Yassine, and Y.-P. Chen. Genetic algorithm design inspired by organizational theory: Pilot study of a dependency structure matrix driven genetic algorithm. In *Genetic and Evolutionary Computation Conference*, pages 1620–1621. Springer, 2003.

[12] T.-L. Yu, K. Sastry, and D. E. Goldberg. Linkage learning, overlapping building blocks, and systematic strategy for scalable recombination. In *Proceedings of the 7th annual conference on Genetic and evolutionary computation*, pages 1217–1224. ACM, 2005.

[13] T.-L. Yu, K. Sastry, D. E. Goldberg, and M. Pelikan. Population sizing for entropy-based model building in discrete estimation of distribution algorithms. In *Proceedings of the 9th annual conference on Genetic and evolutionary computation*, pages 601–608. ACM, 2007.