2014 IEEE Congress on Evolutionary Computation (CEC)
July 6-11, 2014, Beijing, China

Use Model Building On Discretization Algorithms For Discrete EDAs To Work On Real-Valued Problems

Yi-En Su
Department of Electrical Engineering
National Taiwan University
Email: r97941007@ntu.edu.tw

Tian-Li Yu
Department of Electrical Engineering
National Taiwan University
Email: tianliyu@ntu.edu.tw

***Abstract*—Discretization algorithms have been combined with discrete estimation of distribution algorithms (EDAs) to work on real-valued problems. Existing discretization algorithms, such as the fixed-height histogram (FHH) and the split-on-demand (SoD), utilize merely densities of selected chromosomes to build next-generation population, and therefore have limited exploration. This paper adds the concept of model building to FHH and SoD to solve these problems. The model utilizes a variety of information from selected chromosomes to improve the abilities of FHH and SoD to identify promising regions for future exploration.**

Specifically, a model of expected values of selected chromosomes is combined with FHH and SoD to form expected-value FHH and expected-value SoD. The expected-value-discretization algorithms outperform their original versions on an exploration test function as well as the 25 benchmark functions used in the SoD paper. This paper also introduces a model of differential-expected-value of selected chromosomes. The differential-expected-value FHH and differential-expected-value SoD outperform their expected-value versions when tested on the exploration test function and the 25 benchmark functions.

I. Introduction

A simple genetic algorithm (SGA) independently mixes genes by using the crossover operator. Therefore, SGA is easily trapped at local optima on problems with important linkages [1]. Unlike SGA, estimation of distribution algorithms (EDAs) [2], [3] build probabilistic models to recognize linkages among genes, hence linkages are preserved during recombination, making EDAs have a high chance to solve strongly-interconnected problems [4].

Discretization algorithms transform a continuous region into discrete, so that discrete EDAs can be applied to real-valued problems [5]. Existing discretization algorithms, such as the fixed-height histogram [6] (FHH) and the split-on-demand [7]–[9] (SoD), use only the densities of selected chromosomes to discretize the continuous region.

The purpose of this paper is to utilize the concept of model building to improve the performance of existing discretization algorithms. The rest of this paper is organized as follows. In the next section, this paper reviews the three existing discretization algorithms, the fixed-width histogram (FWH), FHH and SoD. This paper also introduces how discretization algorithms work with EDAs. A representative EDA called the extended compact genetic algorithm [10] (ECGA) is taken as an example to be integrated with discretization algorithms. In the third section, we introduce a sine-waved exploration test function to study the limitations of FHH and SoD. After understanding the limitations, we provide the expected-value and the differential-expected-value model as add-ons for FHH and SoD. The newly formed algorithms outperform their original versions on the sine-waved exploration test function. They are expected-value FHH (ev-FHH), expected-value SoD(ev-SoD), differential-expected-value FHH (dev-FHH) and differential-expected-value SoD (dev-SoD). At the last section, we focus on comparing these newly formed discretization algorithms with other discretization algorithms, their original versions, FHH and SoD. The discretization algorithms are integrated with ECGA to be tested on the 25 benchmark problems [11] used in the SoD paper [7]. Results show that ev-FHH and dev-FHH outperforms FHH, and ev-SoD and dev-SoD outperforms SoD on about 20 out of 25 benchmark problems [11] with 95% confidence.

II. Background

In this section we first introduce three existing discretization algorithms, FWH, FHH and SoD. We also describe how these discretization algorithms are integrated with discrete EDAs. This section also briefly introduces ECGA, a representative discrete EDA, and describes how to integrate a discretization algorithm with ECGA.

A. Discretization Algorithms

Discretization algorithms partition the whole search region into a number of bins. For a χ-array discrete EDA, each bin is denoted by a unique integer from 0 to χ. Discrete EDAs then use these integers to build probabilistic models and then generate the next-generation populations inside each bin.

a) Fixed-Height Histogram: The basic rule behind FHH is that more bins should be formed on the region where selected chromosomes are denser. FHH makes each of its bin contain the same number of selected chromosomes. The term *height* in the algorithm's name refers to the number of selected chromosomes in each bin. For a population of N

978-1-4799-1488-3/14/$31.00 ©2014 IEEE

selected chromosomes, the H-bin FHH divides the region into H bins, and each bin contains $\frac{N}{H}$ selected chromosomes. If we represent the positions of these N selected chromosomes as x_i for $i = 1, 2, ..., n$, and $x_i < x_{i+1}$ for $i = 1, 2, ..., n-1$, then, the bin region of FHH can be represented as $[L_i, L_{i+1})$ for $i = 0, 1, ..., H-1$, where

$$L_i = \begin{cases} \text{lower bound} & \text{if } i = 0 \\ \text{upper bound} & \text{if } i = H \\ \frac{x_j + x_{j+1}}{2}, j = \frac{iN}{H} & \text{otherwise.} \end{cases}$$

The flexibility of FHH's bin width makes FHH have unbounded resolution on seemly promising regions with high densities of selected chromosomes, while FWH only has bounded resolution due to its fixed bin width. FWH reaches its bounded resolution when generating all of its next generation population inside the most promising bin. FHH is also empirically shown to outperform FWH in [6]. FWH is not further investigated in this paper.

b) Split-on-Demand: SoD, like FHH, uses the densities of selected chromosomes as information to divide the region, but with two additional parameters, *split rate* γ and *decrease rate* ϵ. The values of γ and ϵ are restricted between 0 and 1. SoD iteratively, randomly splits the region that contains more selected chromosomes than $\gamma \times N$, where N is the number of selected chromosomes. SoD also decreases the value of γ by a factor ϵ every generation i.e. $\gamma \leftarrow \gamma \times \epsilon$ per generation. The reason for a higher value of γ at the early stage is that it makes SoD implement a more roughly global search at the beginning. The lower value of γ at late stage makes SoD to have a more fine model to for local search. The two additional parameters of SoD, γ and ϵ, are acclaimed to help SoD avoid being trapped at local optima [7].

B. ECGA

ECGA [10] is a discrete EDA based on the idea that good probability distribution model can be viewed as learning linkage between genes. ECGA then uses marginal product models (MPM) to describe the probability distribution. The MPM with the minimal description length is considered to be the most accurate, which is used to generate a population of the next generation. The description lengths of the MPMs includes two parts. The first part is called model complexity, as Equation 1, which quantifies complexity of the model. The second part is called compressed population complexity, as Equation 2, which quantifies accuracy of the model. By summing these two parts, ECGA can find an MPM, which satisfies both simplicity and accuracy. ECGA uses greedy heuristic search to find MPM due to computational complexity.

$$\text{MODELCOMPLEXITY} = \sum_{bb \in BB} \left(2^{|bb|-1}\right) \log_2 n \qquad (1)$$

$$\text{DATACOMPLEXITY} = -n \sum_{x} p(x) \log_2 p(x) \qquad (2)$$

C. Discretization Algorithms Integrated with Discrete EDAs

Discretization algorithms, as FHH and SoD, use different strategies to split each continuous region into a number of bins. Each bin is assigned a different integer from 0 to χ for a χ-array discrete EDA. Then, selected chromosomes are encoded into vectors according to their genes' positions, i.e. every gene of the selected chromosomes is assigned the integer of the bin containing the gene. Discrete EDAs then build joint probabilistic models over these vectors, and use the models to generate the next generation population in the form of vectors. These code-vectors are later transformed back into the continuous domain by uniformly sampling the bin region each integral code represents. Details about discretization algorithms can be refered to [8], [12].

III. Exploration Test Function and Model Building Discretization Algorithms

In this section, we make two speculations about the limitations of FHH and SoD according to their bin-splitting strategies. A sine-waved exploration test function is consequently made to verify these two speculations. We build two kinds of models to make FHH and SoD overcome these limitations. The first model utilizes expected value of selected chromosomes, while the second one utilizes differential expected value of selected chromosomes between each generation. By combining the two models with FHH and SoD, four new model-building discretization algorithms are generated. The newly generated algorithms, ev-FHH, ev-SoD, dev-FHH and dev-SoD outperform the original FHH and SoD on the sine-waved exploration test function.

A. Speculations of FHH and SoD

In FHH and SoD, the only information used to split the explored region is the densities of selected chromosomes. FHH and SoD do not utilize the fitness values of selected chromosomes, let alone other information of selected chromosomes. Therefore, the chromosome with the highest tness is treated the same as the one with the lowest tness after selection. This makes our first speculation that the discretization algorithms recognize no differences between optima of different height but same width, thus, limit their chances to find global optimum. The definition of width of a optimum is in the next subsection.

Next, we speculate that if selected chromosomes around the global optimum does not surpass certain threshold, it is harder for FHH or SoD to find global optimum in latter generations, even though these selected chromosomes have much higher fitness than others. The speculation inferred from the characteristic that FHH and SoD tempt to converge their bins only to the regions with high densities of selected chromosomes. It is understandable that it is hard for an algorithm to find the global optimum, if there are not any selected chromosomes exist around it, but it is inefficient if the algorithm reduces its resolution on the promising region even there exists some selected chromosomes with relatively high fitness to other selected chromosomes.

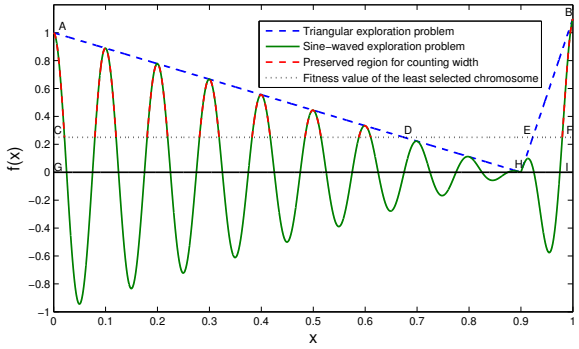


Fig. 1. A triangular exploration test function ($\overline{AH}$ and $\overline{BH}$) and a sine-waved exploration problem (solid green line). The triangular function is composed of two straight lines, $\overline{AH}$ and $\overline{BH}$. Global optimum is fixed at the right boundary, while a local optimum is fixed at the left boundary. In this figure, the ratio of the whole region to the width of global optimum is 10 and the ratio of the peak value of global optimum to the peak value of local optimum is 1.1. We times the triangular function with $\sin(10 \times 2\pi + \frac{\pi}{2})$ to form the sine-waved exploration test function. The dotted line is $y = 2.5$.

Take an H-bin FHH with N selected chromosomes as an example. If the number of selected chromosomes around the global optimum is less than $\frac{H}{N}$, FHH does not take the region of these selected chromosomes as one single bin, instead, FHH combines the region with some other regions together to obtain exactly $\frac{H}{N}$ selected chromosomes in a bin. If the additionally included selected chromosomes are far away from the primary chromosomes, the assigned bin is wide. The resolution around global optimum decreases, thus, making it harder for FHH to have enough selected chromosomes around global optima in the next generation. This situation happens no matter how high the fitnesses of selected chromosomes around the global optimum are.

B. Exploration Test Function

According to our speculations above, the triangular exploration test problem and the sine-waved exploration test problem are proposed for verification. Several points in Figure 1 are defined as follows.

A:$(\mathbf{0}, \mathbf{1})$ The local optimum of both test problems.

B:$(\mathbf{1}, \mathbf{1.1})$ The global optimum of both test problems.

C:$(\mathbf{0}, \mathbf{0.25})$

D:$(\mathbf{0.675}, \mathbf{0.25})$ The intersection of the dotted line and the dashed line with negative slope.

E:$(\mathbf{0.925}, \mathbf{0.25})$ The intersection of the dotted line and the dashed line with positive slope.

F:$(\mathbf{1}, \mathbf{0.25})$

G:$(\mathbf{0}, \mathbf{0})$

H:$(\mathbf{0.9}, \mathbf{0})$

I:$(\mathbf{1}, \mathbf{0})$

First we need to define the width of an optimum, so that we can define two parameters of the exploration test functions. The width of an optimum should be highly related to the distribution of selected chromosomes. The reason is that a discretization algorithm can only observe an optimum through selected chromosomes. For a discretization algorithm with chromosomes uniformly distributed, and the selection pressure as s, we infer that only the first $\frac{1}{s}$ regions with the highest fitnesses have chances to be observed. The definition of width of an optimum is given as below.

Definition 1: The width of an optimum is the projection of its surrounding region which has its fitness the first $\frac{1}{s}$ height, on certain axis. s is the selection pressure of the discretization algorithm.

In the case of the triangular exploration problem in Figure 1, assuming the lowest fitness of the selected chromosomes is the value of the dotted line in Figure 1, the width of the global optimum is $\overline{EF}$, the projection of $\overline{BE}$ on x-axis. The width of the local optimum is $\overline{CD}$, the projection of $\overline{AD}$ on the x-axis.

In our speculation, FHH and SoD tend to reduce their resolution on the global optimum when facing a problem with its global optimum of little width while the global optimum is far away from other local optima. The newly set bin which contains the global optimum is wide, as discussed in subsection 3.1. The triangular exploration test problem and the sine-waved exploration test problem are designed accordingly, as in Figure 1, so we can test the two speculations of FHH and SoD. Two parameters are made to vary the attributes of the problems. The first parameter is the ratio of width (RoW). The second parameter is the ratio of peak value (RoPV). Definitions of RoW and RoPV are given below.

Definition 2: The ratio of width (RoW) of the exploration test problems is the ratio of the width of the whole region to the width of the global optimum.

Definition 3: The ratio of peak value (RoPV) of the exploration test problems is the ratio of the peak value of the global optimum to the peak value of the local optimum.

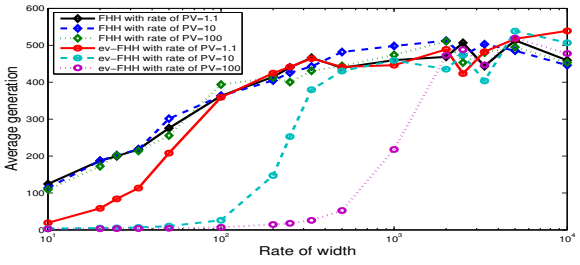
In case of the triangular exploration problem, RoW is $\frac{\overline{AB}}{\overline{AB+CD}}$. The exploration test function is chosen to be triangular, because RoW is independent of the selection pressure of the discretization algorithm in a triangular function. We obtain this result from $\frac{\overline{AB}}{\overline{AB+CD}} = \frac{\overline{FG}}{\overline{EG}}$.

To make the exploration problem closer to a real problem, we times the triangular exploration problem with a sine wave, $\sin(10 \times 2\pi + \frac{\pi}{2})$, to build the sine-waved exploration test problem. The sine-waved exploration test problem is used throughout this paper, as the solid sine-waved line in Figure 1. The RoPV of the sine-waved exploration test problem is the same as the triangular version, but the RoW has changed. The regions to be counted on RoW are highlighted by the red dashed line in Figure 1. Here, we approximate the RoW of the sine-waved exploration test problem the same as the triangular version, $\frac{\overline{FG}}{\overline{EG}}$.

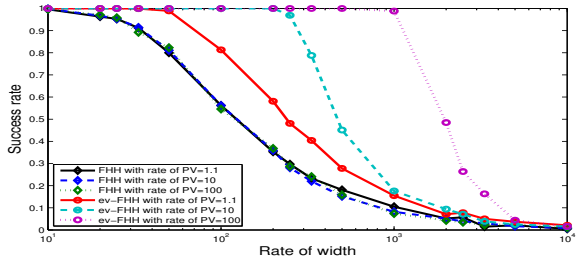
By varying RoW and RoPV, we can change the difficulties of the exploration test function: the larger the RoW, the more difficult for FHH and SoD to identify the global optimum. The larger the RoPV should make it easier for a search algorithm to identify the global optimum. According to our first speculations, FHH and SoD do not improve any of their

performances despite RoPV increases extremely high. FHH and SoD also degrade their performance badly when RoW increases, according to our second speculation.

In this paper, RoW of the exploration test function is ranged from 10 to 10000 while RoPV is ranged from 1.1 to 100 to observe the attribution of the discretization algorithms on the exploration problem. Each test was independently run 1000 times with population 100 and upper bound generation 1000. The bin number is 10 for FHH, while the split rate γ is 0.5, and the decrease rate ϵ is 0.999 for SoD. The values of γ and ϵ are one of the pairs of parameters used in the SoD paper [7] which have its average bin number around 10 under the experimental condition, so the results of FHH and SoD can also be compared. We use truncational selection with selection pressure 2 in the experiments.



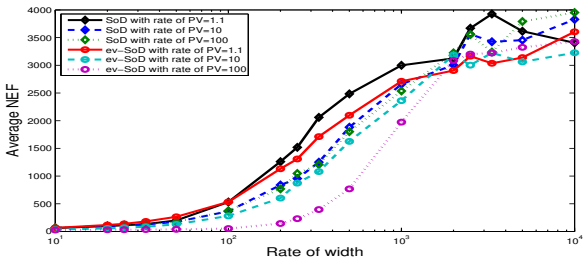
(a)



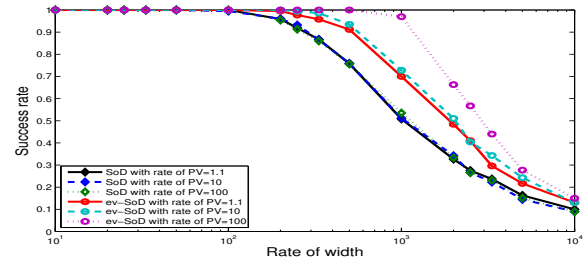
(b)

Fig. 2. Performances comparisons of FHH and ev-FHH on the sine-waved exploration test function with different RoW and RoPV. The diamonds represent the results of FHH, while the circles represent the results of ev-FHH. The solid lines represent the results on the exploration test function with RoPV = 1.1. The dashed lines are on the function with RoPV = 10 and the dotted line with RoPV = 100. According to the figures, FHH can not improve any of its performances when RoPV increases from 1.1 to 100. On the other hand, ev-FHH improves both its average generation and success rate as RoPV increases. Besides, ev-FHH outperforms FHH on the exploration test functions throughout the tested region of different parameters.

The experimental results are shown on Figures 2 and 3. Figure 2(a) shows the average generations taken by FHH or ev-FHH to find the global optimum. Figure 2(b) shows the success rates for FHH and ev-FHH to find the global optimum. Figure 3(a) illustrates the average number of function evaluations (NFEs) taken by SoD or ev-SoD to find the global optimum, Figure 3(b) shows the success rates for SoD and ev-SoD to find the global optimum. According to the results, FHH can not identify any differences between the exploration test functions with RoPV equals 1.1 or 100. SoD



(a)



(b)

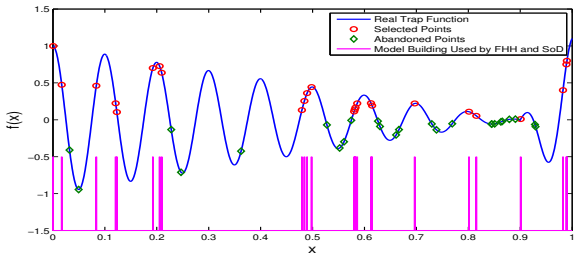
Fig. 3. Performances comparisons of SoD and ev-SoD on the sine-waved exploration test function with different RoW and RoPV. The diamonds represent the results from SoD, while circles represent the results from ev-SoD. Other settings are the same as Figure 2. According to the figures, SoD's average NFE decreases as RoPV increases, while SoD can not improve any of its success rate as RoPV increases. Ev-SoD improves both its average NFE and success rate and outperforms SoD on the exploration test functions with all tested parameters.

consumes fewer average NFE when dealing with higher RoPV but can not improve any of its success rates. These results are consistent with our first speculation that the exploration ability of FHH and SoD are limited when facing certain types of problems as the exploration test problem. Discussion about the second speculation is hold until ev-FHH and ev-SoD are introduced, and their results on the exploration test function can be compared with the results of FHH and SoD.

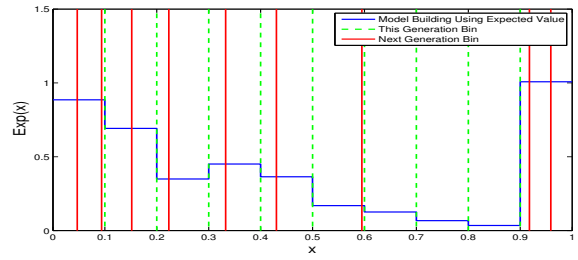
C. Model Building Discretization Algorithms Using expected value Model

The model building discretization algorithms (MBDAs) use models instead of merely the densities of selected chromosomes as the criteria, to separate the region into bins. In MBDAs, the built model is estimates the potential of each region for containing the global optimum. The expected value of selected chromosomes is considered a representative measurement. The expected-value model for discretization algorithms is introduced. The expected-value model calculates the average fitness values of selected chromosomes inside each bin, subtracting the value with the overall-least fitness value of selected chromosomes to eliminate the offset provided by the problems. The model then takes these expected-values as the estimated potential for finding global optima inside each bin, forming the expected value model, $exp(x)$, as the blue solid histogram in Figure 4(b). The expected-value model then can

be used to split the region into bins by different bin-splitting strategies. If we combine the expected-value model with the FHH bin splitting strategy, we obtain ev-FHH; on the other hand, if we combine the model with the SoD bin splitting strategy, we obtain ev-SoD. The only difference between ev-FHH and FHH is that ev-FHH uses the integral value of expected value model, $\int exp(x)$, instead of the number of selected chromosomes, to separate the region into bins. First, ev-FHH calculates the value of $\int_{lb}^{ub} exp(x)$, where lb and ub are the lower and upper bounds of the explored region. Then, ev-FHH divides the search region into bins that the values of $\int exp(x)$ are the same inside each bin, in the case of Figure 4, the red solid bins equally divide the area under the blue solid histogram. As for ev-SoD, $\int exp(x)$ is also used to substitute for the role of number of selected chromosomes in SoD. Ev-SoD iteratively splits the region into two bins until each bin satisfies the condition $\int^{R} exp(x) < \gamma \times \int_{lb}^{ub} exp(x)$, where R denotes the range of the bin. The value of γ decreases by the factor ϵ per generation. The pseudo-code for ev-FHH and ev-SoD is given in Algorithms 1 and 2, respectively.



(a)



(b)

Fig. 4. Using the expected value as a model to proximate the exploration test problem. The population size is 50, and 25 chromosomes (red circless) are selected, while others (green diamonds) are discarded. The pink erected solid lines in the Figure 4(a) represents the model used by FHH and SoD. The blue solid histogram at Figure 4(b) represents the expected value model used by ev-FHH and ev-SoD. The red solid bins are the 10 bins set up to generate the next generation population. According to Algorithm 1, the integral value of the expected value model should be the same inside each next-generation bin, which means the area under the blue solid histogram should be the same in each red solid bin.

FHH and SoD can also be regarded as special cases in MBDA. The model used by FHH and SoD uses the delta functions to represent every selected chromosome, as the pink erected solid lines in Figure 4(a). The heuristic behind implies that the regions with higher densities of selected chromosomes have greater potential to contain the global optimum. FHH and SoD use the same model to estimate the potential of the search region while using different strategies to split the area. In other words, FHH and SoD utilize the same model but use different ways to create the next-generation population.

Algorithm 1: Pseudo code for ev-FHH

Input:
$Bin_i \leftarrow$ a set of bins to divide the searched region
$H \leftarrow$ number of bins
$R_i \leftarrow$ range of Bin_i $i = 1...H$
Output:
R_i $\#Bin_i$'s range used for next generation

$F_{least} \leftarrow$ lowest fitness value of all selected chromosomes
$\bar{F}_i \leftarrow$ averaged fitness of selected chromosomes inside R_i
$M_i \leftarrow \bar{F}_i - F_{least}$
$M(x) \leftarrow$ {if x inside R_i, return M_i}
$Avg \leftarrow \frac{\int_{lb}^{ub} M(x)dx}{H}$
Decide the range of R_i so that $\int^{R_i} M(x) = Avg$ for each R_i
return R_i

Algorithm 2: Pseudo code for ev-SoD

Input:
$\gamma \leftarrow$ decrease rate
$R_i \leftarrow$ range of the i th bin
Output:
R_i #the i th bin range used for next generation

$F_{least} \leftarrow$ smallest fitness value of all selected chromosomes
$\bar{F}_i \leftarrow$ averaged fitness values of selected chromosomes inside R_i
$M_i \leftarrow \bar{F}_i - F_{least}$
$M(x) \leftarrow$ {if x inside R_i, return M_i}
$M_{total} \leftarrow \int_{lb}^{ub} M(x)dx$
$k \leftarrow 0$
$R_0 \leftarrow$ the whole region.
Iteratively do
  for $i = 0$ to k
    If $\int^{R_i} M(x) > \gamma \times M_{total}$
      Split R_i into R_i, R_{k+1}, $k \leftarrow k + 1$
Until $\int^{R_i} M(x) < \gamma \times M_{total}$ for all i
return R_i

D. Performances of MBDA on The Sine-waved Exploration Test Problem

The experimental results of ev-FHH and ev-SoD on the exploration test problem are shown in Figures 2 and 3. Ev-FHH and ev-SoD improve both their average problem-solving generation and success rate when RoPV increases. Both ev-

FHH and ev-SoD outperform their original versions under different values of RoW. In Figure 2, for the exploration test problem with RoPV = 100 and RoW = 1000, ev-FHH has 100% success rate while the success rate of FHH is less than 10%. According to these results, the expected value model does help the discretization algorithms be more adaptive to test problems like exploration test problems.

Now we evaluate the validity of the second speculation mentioned at the beginning of subsection 3.1, by comparing the results of FHH and ev-FHH; SoD and ev-SoD on the exploration test problem. The second speculation states that because FHH and SoD use the density of selected chromosomes as the criterion to split bins, they tend to neglect the narrow optimum, even though the fitness value of the selected chromosomes inside the optimum are extremely high. The newly invented algorithms, ev-FHH and ev-SoD use the expected value of the selected chromosomes instead of the densities of selected chromosomes. For ev-FHH and ev-SoD, there is no difference whether a region has one or a hundred selected chromosomes if they all have the same fitness value. By removing the reason that cause FHH and SoD converge their bins to a wider local optimum, ev-FHH and ev-SoD should have higher chance to find a global optimum of little width. Figures 2 and 3 show that both ev-FHH and ev-SoD increase their success rate on finding a global optimum of less width, comparing to their original versions. This is consistent with our second speculation. The discretization algorithms increase their chance on solving a problem with a global optimum of little width by removing the density-inclination factor.

To improve the performance of discretization algorithms on sine-waved exploration problem further, we introduce another model called differential expected value model. The 'differential' here implies the difference of expected value between generations. The idea of adding the differential part makes discretization algorithms more adaptive to the problems. The heuristic behind is using the 'differential value' to detect if the area inside a bin has already been fully explored. If the area has been fully explored, its obtained expected values in successive generations should be close; otherwise, the obtained expected values may variate radically. By using differential-expected value as a model, we created dev-FHH and dev-SoD. Pseudo codes for dev-FHH and dev-SoD are identical to that for ev-FHH and ev-SoD except the following modification should be added. Following pieces of code should be inserted after the declaration of M_i:

$$E \leftarrow M_i^t$$
$$M_i^t \leftarrow E + (E - M_i^{t-1})$$

and before the return of R_i:

$$E(x) \leftarrow \begin{cases} E_i & \text{if x inside } R_i \\ 0 & \text{otherwise.} \end{cases}$$

$$M_i^t \leftarrow \frac{\int^{R_i} E(x)}{Length(R_i)},$$

where t denotes the generation number, and $M_i = M_i^t$ in Algorithms 1 and 2.

The experimental results of dev-FHH and dev-SoD are shown in Figures 5 and 6. In Figure 6, dev-SoD maintains its success rate over 20% for all tested PV values even when RoW reaches 10000. The differential expected value model seems to help the discretization algorithms find narrow global optimum with higher probability. discretization algorithms using differential expected value model are observed to outperform the version using expected value model on the sine-waved exploration problem, especially in the case of SoD. Dev-SoD outperforms ev-SoD more obviously than dev-FHH to ev-FHH on the problem.

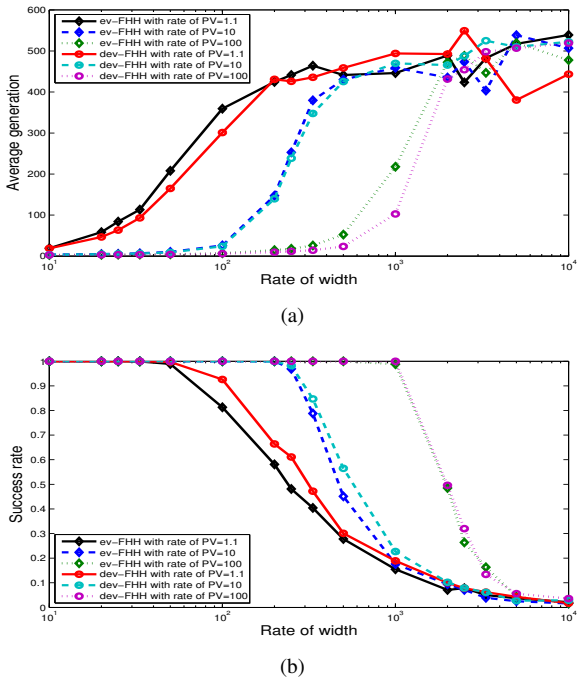


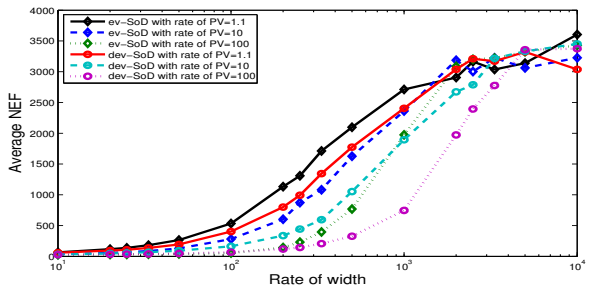
Fig. 5. Performances comparisons of ev-FHH and dev-FHH on the exploration test functions with different rate of width and rate of PV. The diamonds represent the results of ev-FHH. The circles represent the results of dev-FHH, while other settings are the same in Figure 2. According to the figures, dev-FHH barely improves its performance comparing with ev-FHH on the exploration test function.

IV. Performance with ECGA on 25 Benchmark Problems

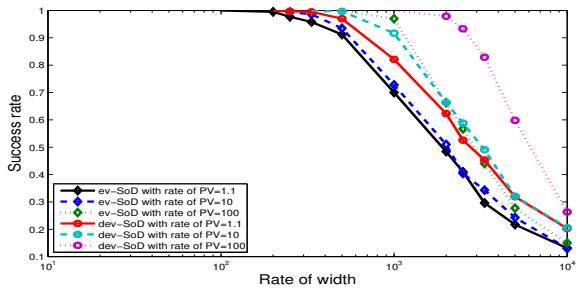
Four newly created discretization algorithms, ev-FHH, dev-FHH, ev-SoD and dev-SoD, are integrated with ECGA to be tested on the 25 benchmark problems [11]. We also integrated other discretization algorithms, FHH and SoD, with ECGA and compared all their results in this section.

A. Integrating ECGA with MBDA

Before integrating the new discretization algorithms with ECGA, there is one difficulty need to be resolved. Different from FHH and SoD, whose bin splitting strategies are based on



(a)



(b)

Fig. 6. Performances comparisons of ev-SoD and dev-SoD on the exploration test function with different rate of width and rate of PV. The diamonds represent the results of ev-SoD. The circles represent the results of dev-SoD, while other settings are the same in Figure 2. Comparing to ev-SoD, dev-SoD improves its performances, especially the success rate, on exploration test functions with rate of width greater than 1000.

the number of chromosomes, other model building discretization algorithms may not guarantee that every newly generated bin region contains at least one selected chromosome. If there are no selected chromosomes in a newly generated bin, the code of this bin is assigned zero probabilities by ECGA, and no code-vectors containing this code is generated by ECGA afterward. This empty-bin-code is ignored in the probability model built by ECGA, and there are no newly generated chromosomes located inside this bin in the next generation. To avoid this issue, we uniformly add virtual chromosomes to the model built by ECGA. The sum of virtual chromosomes should be small, which is 10% of population size in this paper.

Integration of model building discretization algorithms with ECGA can be enumerated as below.

1) Randomly initialize a population with N chromosomes. The discretization algorithm uniformly split the searched region into H bins per dimension.
2) Select the population using a selection strategy, such as tournament selection.
3) Build the model according to the selected chromosomes inside each bin. Generate new bins according to the bin-splitting strategy the discretization algorithm uses.
4) Check each selected chromosome for which bin region it locates inside, and assigned the code of the bin to its correspondent gene. Each chromosome is turned into an M-dimensional code-vector. M is the dimension of the given problem.
5) Use ECGA to model the code-vectors with greedy search of minimum description length of MPM model. The MPM model found with the smallest value of the sum of Equations 1 and 2 are used.
6) Add virtual chromosomes uniformly to the MPM model built by ECGA. The sum of virtual chromosomes is 10% of the population size. Use the MPM model to generate the next population in forms of M-dimensional vectors.
7) Turn each M-dimensional vector back into continuous domain by uniformly sampling a gene inside the bin region which each of its code stands for. If stop criteria isn't met, return to step 2.

B. Performances on 25 Benchmark Problems

Parameters used in the experiments of the discretization algorithms integrated with ECGA follow the SoD paper [7], with population size 250, crossover probability 0.975, chromosome length 10 and tournament size 8. We have two different upper bounds numbers of evaluation functions (NFE), 30000 and 100000, so we can observe the performances of discretization algorithms both in short and long term. The bin numbers of FHH, ev-FHH and dev-FHH are 18 or 25 for NFE=30000, and bin number is set as 25 for NFE=100000. For SoD, ev-SoD and dev-SoD, values of the split rate, γ, and a decrease rate, ϵ, are listed in Table I. Using two different values of parameters of γ and ϵ for SoD also follow the SoD paper [7]. For ev-SoD and dev-SoD, the values of γ are the same with SoD, while the values of ϵ are tuned to make the average bin number of ev-SoD and dev-SoD close to SoD.

Algorithm	NFE=30000			NFE=30000			NFE=100000		
	γ	ϵ	H_{avg}	γ	ϵ	H_{avg}	γ	ϵ	H_{avg}
SoD	0.7	0.99	18.00	0.45	0.988	25.93	0.5	0.998	25.79
ev-SoD	0.7	0.98	17.68	0.45	0.981	25.22	0.5	0.994	25.61
dev-SoD	0.7	0.979	18.08	0.45	0.98	25.84	0.5	0.9935	25.64

TABLE I
PARAMETER SETTINGS FOR VALUE OF γ, ϵ AND ITS CONSEQUAL AVERAGE BIN NUMBER (H_{avg}) PER GENERATION UNDER 250 RUNS FOR THE 25 BENCHMARK PROBLEMS [11].

The performances comparisons between FHH, ev-FHH, dev-FHH/SoD, ev-SoD and dev-SoD integrated with ECGA on 25 benchmark problems [11] are listed in Tables II and III, under 250 independent runs. We use t-test to determine the winner algorithm under different match. The numbers inside the first column denote the problem's serial number detailed in [11]. If the former discretization algorithm outperforms the latter with 95% t-test confidence, the values are marked italic, while the latter outperforms the former, the values are marked bold. The last row of the tables sum up the total number of problems one algorithm win over the other. 4/20 in the first column of the last row means that FHH win over ev-FHH on 4 problems while ev-FHH win over FHH on 20 problems out of 25. Results show that the newly built model building discretization algorithms, ev-FHH, dev-FHH, ev-SoD and dev-SoD outperform the original ones on most of the benchmark

	NFE=30000 Bin=18			NFE=30000 Bin=25			NFE=100000 Bin=25		
	FHH vs ev-FHH	FHH vs dev-FHH	ev-FHH vs dev-FHH	FHH vs ev-FHH	FHH vs dev-FHH	ev-FHH vs dev-FHH	FHH vs ev-FHH	FHH vs dev-FHH	ev-FHH vs dev-FHH
1	**12.17**	**12.17**	0.00	**10.88**	**10.88**	0.00	**11.09**	**11.09**	0.00
2	**23.04**	**23.47**	**11.88**	**21.96**	**22.13**	**7.43**	**26.55**	**26.55**	**10.88**
3	**13.56**	**14.02**	1.42	**12.36**	**12.63**	0.79	**14.23**	**14.38**	1.19
4	**21.07**	**21.30**	**9.69**	**18.46**	**18.54**	**3.87**	**20.02**	**20.02**	**7.52**
5	**16.70**	**16.83**	**2.61**	**17.40**	**17.52**	1.00	**15.75**	**15.75**	**2.03**
6	**5.37**	**5.93**	**3.44**	**5.92**	**6.24**	**2.84**	**7.73**	**7.47**	-1.26
7	**11.54**	**11.54**	0.00	**11.29**	**11.29**	0.00	0.00	0.00	0.00
8	*-7.20*	*-6.55*	0.68	*-4.34*	*-4.48*	-0.14	*-5.46*	*-4.42*	1.08
9	*-46.94*	*-25.19*	**10.21**	*-16.24*	**8.48**	**17.29**	*-2.35*	**10.06**	**3.25**
10	**6.62**	**6.91**	0.12	**2.56**	**3.02**	0.37	1.06	**3.62**	**2.43**
11	**28.41**	**30.49**	**1.97**	**23.07**	**25.52**	**1.71**	**23.99**	**24.53**	0.54
12	**6.40**	**8.29**	**2.26**	**8.93**	**8.73**	-0.43	**5.64**	**6.60**	0.55
13	*-48.69*	*-40.22*	**7.14**	*-25.93*	*-13.94*	**9.71**	*-23.34*	*-11.64*	**11.49**
14	**8.17**	**8.17**	-0.07	**9.54**	**6.59**	*-3.10*	**12.51**	**13.39**	1.00
15	*-3.84*	*-2.24*	1.56	*-2.35*	-0.84	1.48	0.20	1.04	0.84
16	**7.58**	**8.36**	0.80	**7.55**	**7.16**	-0.60	**5.55**	**6.06**	0.57
17	**13.21**	**17.27**	**3.98**	**15.60**	**17.13**	1.51	**12.62**	**13.85**	1.24
18	**7.21**	**8.02**	1.07	**6.62**	**7.23**	0.42	**4.25**	**5.81**	1.48
19	**7.70**	**8.77**	1.46	**8.60**	**7.16**	-1.32	**8.07**	**8.51**	0.38
20	**6.45**	**9.03**	**2.88**	**7.76**	**7.55**	0.36	**7.52**	**6.82**	-0.37
21	**5.63**	**7.34**	1.52	**2.63**	**4.66**	**2.09**	**3.70**	**4.53**	0.71
22	**5.82**	**6.13**	0.37	**6.49**	**6.41**	0.11	**6.36**	**6.26**	-0.01
23	**5.54**	**7.33**	**1.88**	**3.96**	**5.95**	**2.05**	**5.15**	**4.40**	-0.96
24	-0.93	-0.73	0.22	0.33	-0.40	-0.06	1.05	0.22	-0.82
25	**16.90**	**20.27**	**3.25**	**20.08**	**22.59**	**2.38**	**22.62**	**20.19**	-0.85
Sum	4/20	4/20	0/12	4/20	2/21	1/9	3/18	2/20	0/6

TABLE II
FHH, EV-FHH AND DEV-FHH PERFORMANCES COMPARISON.

	NFE=30000 γ=0.7			NFE=30000 γ=0.45			NFE=100000 γ=0.5		
	SoD vs ev-SoD	SoD vs dev-SoD	ev-SoD vs dev-SoD	SoD vs ev-SoD	SoD vs dev-SoD	ev-SoD vs dev-SoD	SoD vs ev-SoD	SoD vs dev-SoD	ev-SoD vs dev-SoD
1	0.00	0.00	0.00	**2.59**	**2.59**	0.00	0.00	0.00	0.00
2	1.66	0.23	-1.03	**8.45**	**8.46**	1.01	0.00	0.00	000
3	**6.62**	**7.66**	1.36	**8.76**	**9.51**	1.36	**2.68**	**3.24**	0.61
4	**4.66**	**6.77**	**2.14**	**9.32**	**9.36**	**1.97**	0.00	0.00	0
5	0.32	1.23	0.86	**11.00**	**11.02**	0.35	**4.49**	**4.49**	**13.46**
6	**5.51**	**5.64**	**1.63**	**6.17**	**6.22**	1.67	**5.90**	**6.47**	0.74
7	0	0	0	**5.36**	**5.35**	0	0.00	0.00	0
8	*-30.27*	*-30.82*	-0.37	*-26.40*	*-28.16*	-1.18	*-38.51*	*-34.75*	**2.24**
9	**32.57**	**32.92**	0.70	**32.29**	**32.75**	1.66	**34.85**	**34.85**	0.00
10	**8.95**	**8.91**	-0.18	**4.82**	**3.74**	-1.05	**7.38**	**6.62**	-0.84
11	**8.68**	**8.28**	0.07	**17.15**	**19.76**	**2.84**	**15.33**	**18.02**	**2.71**
12	-1.31	0.71	**2.15**	**5.75**	**5.57**	-0.41	**4.14**	**4.13**	-0.02
13	0.63	*-4.35*	*-5.29*	**14.80**	**13.32**	-1.52	**16.22**	**16.36**	0.16
14	*-3.78*	*-3.55*	0.11	**3.68**	**1.96**	*-2.00*	**9.62**	**10.28**	0.39
15	**5.36**	**3.96**	-1.25	**3.95**	**3.75**	-0.19	**5.05**	**4.08**	-0.86
16	**8.31**	**7.55**	-0.70	**6.28**	**6.12**	-0.43	**7.65**	**7.70**	0.90
17	**5.45**	**4.11**	-1.41	**4.60**	**6.43**	**2.09**	**8.48**	**7.86**	-0.59
18	-0.36	0.59	0.94	**6.34**	**6.95**	0.39	**3.54**	**4.40**	0.94
19	1.48	0.76	-0.17	**8.23**	**7.84**	-0.40	**4.19**	**3.72**	-0.50
20	0.48	-0.14	-0.62	**7.29**	**6.78**	-0.53	**3.45**	**3.80**	0.40
21	**7.66**	**8.79**	1.24	**9.39**	**10.96**	**1.94**	**17.39**	**18.31**	1.37
22	-0.65	-1.16	-0.56	**6.31**	**5.22**	-0.28	**7.76**	**7.29**	0.86
23	**4.11**	**3.91**	-0.23	**8.30**	**7.55**	-0.91	**11.75**	**12.40**	0.58
24	1.42	-0.82	*-2.02*	-0.69	-0.17	0.58	0.70	0.70	0.00
25	*-5.22*	*-6.05*	-0.95	**4.51**	**1.92**	*-2.37*	1.40	1.58	0.20
Sum	3/11	4/11	2/3	1/23	1/23	2/4	1/18	1/18	0/3

TABLE III
SOD, EV-SOD AND DEV-SOD PERFORMANCES COMPARISON UNDER.

problems, about 20 out of 25. Dev-FHH also outperforms ev-FHH on some benchmark functions. But the performances difference between dev-SoD and ev-SoD on the 25 benchmark functions [11] are barely discernible. This is different from the results we observed on the exploration test function in previous section. The difference is attributed to the different characteristics between the exploration test function and the 25 benchmark functions [11].

V. CONCLUSION

This paper proposed the use of model building on existing discretization algorithms. The built models extract more information from the population and try to estimate more properly the potential of the unexplored region of the problem. We have introduced two models to realize the concept of model building, the expected value model and the differential expected value model. The models were combined with the original bin splitting strategies in FHH and SoD, forming the new discretization algorithms, ev-FHH, dev-FHH, ev-SoD and dev-SoD. Our discretization algorithms outperform the original versions on the invented exploration test function and the 25 benchmark functions [11]. From our experiments, we believe that exploration is crucial for real-valued optimization problems and that exploration should be properly integrated into discretization algorithms to yield better performance when combining with discrete EDAs.

REFERENCES

[1] D. E. Goldberg, *Genetic Algorithms in Search, Optimization, and Machine Learning*. Reading, MA: Addison-Wesley, 1989.

[2] P. Larrañaga and J. A. Lozano, Eds., *Estimation of Distribution Algorithms: A New Tool for Evolutionary Computation*. Boston, MA: Kluwer, 2002.

[3] M. Pelikan, K. Sastry, and E. Cantu-Paz, Eds., *Scalable Optimization via Probabilistic Modeling: From Algorithms to Applications*. Berlin: Springer, 2006.

[4] M. Hauschild and M. Pelikan, "An introduction and survey of estimation of distribution algorithms," pp. 111–128, 2011.

[5] J. Sun, Q. Zhang, and E. Tsang, "De/eda: A new evolutionary algorithm for global optimization," pp. 249–262, 2005.

[6] S. Tsutsui, M. Pelikan, and D. E. Goldberg, "Evolutionary algorithm using marginal histogram models in continuous domain," *Proc. of the 2001 Genetic and Evolutionary Computation Conference Workshop Program*, pp. 230–233, 2001.

[7] Y.-p. Chen and C.-H. Chen, "Enabling the extended compact genetic algorithm for real-parameter optimization by using adaptive discretization," *Evol. Comput.*, vol. 18, pp. 199–228, 2010.

[8] C.-H. Chen, W.-N. Liu, and Y.-p. Chen, "Adaptive discretization for probabilistic model building genetic algorithms," pp. 1103–1110, 2006.

[9] J.-J. Liou and Y.-p. Chen, "Adaptive discretization on multidimensional continuous search spaces," pp. 977–984, 2008.

[10] G. R. Harik, F. G. Lobo, and K. Sastry, "Linkage learning via probabilistic modeling in the extended compact genetic algorithm (ECGA)," in *Scalable Optimization via Probabilistic Modeling*. Berlin: Springer Berlin Heidelberg, 2006, ch. 3, pp. 39–61.

[11] P. N. Suganthan, N. Hansen, J. J. Liang, K. Deb, Y.-p. Chen, A. Auger, and S. Tiwari, "Problem definitions and evaluation criteria for the cec 2005 special session on real-parameter optimization," pp. 1–50, 2005.

[12] E. Cantú-Paz, "Supervised and unsupervised discretization methods for evolutionary algo- rithms," pp. 213–216, 2001.