2010 International Conference on Technologies and Applications of Artificial Intelligence

An Estimation of Distribution Algorithm Based Portfolio Selection Approach

Rui-Tian xu, Jun Zhang*(corresponding author)*
Dept. of C.S., SUN Yat-sen University
Key Laboratory of Digital Life, Ministry of Education
Key Laboratory of Software Technology, Education
Dept. of Guangdong Province, P.R. China
e-mail: junzhang@ieee.org

Ou Liu and Rui-zhang Huang
Hong Kog Polytechnic University
HongKong SAR, China

***Abstract*—A portfolio selection problem is about finding an optimal scheme to allocate a fixed amount of capital to a set of available assets. The optimal scheme is very helpful for investors in making decisions. However, finding the optimal scheme is difficult and time-consuming especially when the number of assets is large and some actual investment constraints are considered. This paper proposes a new approach based on estimation of distribution algorithms (EDAs) for solving a cardinality constrained portfolio selection (CCPS) problem. The proposed algorithm, termed PBIL-CCPS, hybridizes an EDA called population-based incremental learning (PBIL) algorithm and a continuous PBIL (PBILc) algorithm, to optimize the selection of assets and the allocation of capital respectively. The proposed algorithm adopts an adaptive parameter control strategy and an elitist strategy. The performance of the proposed algorithm is compared with a genetic algorithm and a particle swarm optimization algorithm. The results demonstrate that the proposed algorithm can achieve a satisfactory result for portfolio selection and perform well in searching nondominated portfolios with high expected returns.**

Keywords-estimation of distribution algorithm; population-based incremental learning algorithm; portfolio selection problem

I. INTRODUCTION

Selecting the optimal portfolio is an attractive problem in modern investment research. Investors or investment companies profit from investing a large number of available assets. A promising portfolio is very helpful for investors in making decisions. The portfolio selection (PS) problem is about finding an optimum scheme to allocate a fixed amount of capital to a set of available assets. As early as 1950s, Markowitz [1] proposed a classic mathematical model, termed mean-variance (MV) model, for the problem. Each feasible portfolio in the model has its own expected return and a risk value measured by the variance of the return. The portfolios which have the minimum risk for each level of the expected return form an efficient frontier. In other word, for every level of the desired expected return, the efficient frontier offers the best portfolio with the minimum risk. With the aid of the efficient frontier, investors can choose and decide which portfolio to adopt, according to their own tolerance for risk.

The MV model has advanced the research of investment portfolio on a quantitative basis and has established the modern investment theory. However, the model has no restriction on the number of the invested assets and thus is not very suitable in practical applications. A cardinality constrained mean-variance (CCMV) model [2] fixes the issue and is more suitable for actual investment. In this paper, we focus on the cardinality constrained portfolio selection (CCPS) problem based on the CCMV model.

The CCPS problem can be formulated into a mixed integer quadratic programming problem, which is an NP-hard problem [3]. The computation demanding for solving the problem is high and the problem has the characteristics such as harsh constraints and a non-linear search space. When the number of assets is large, exact algorithm are not efficient to solve the problem. Some researchers introduced computational intelligence (CI) algorithms to solve the problem [2][4][5]. Inspired by the nature, CI algorithms are designed for obtaining an acceptable solution in an acceptable time by imitating natural evolutionary processes [6], animals' group behaviors [7], activities of human thinking [8], or real physical phenomena [9], etc. Some PS approaches based on CI algorithms such as genetic algorithms (GAs) [2] and particle swarm optimization (PSO) algorithms [5] have been developed. Chang et al. [2] proposed three algorithms based on GA, tabu search (TS), and simulated annealing (SA), respectively for solving the PS problem. They reported that the algorithm based on GA generally outperformed the other two algorithms. Fernández et al. [4] and Cura [5] respectively applied a neural network (NN) and a PSO algorithm to solve the problem. Using the same instances as in [2], the two algorithms both performed better than the algorithms in [2] when searching portfolios with low risks.

In this paper, a novel approach based on the estimation of distribution algorithms (EDAs) for the CCPS problem is proposed. EDA is a CI algorithm issued from the improvement of GAs [10][11]. Different from GA, it does not use crossover and mutation operators on the individuals in the population, but adopts an evolutionary mode for searching the best solutions. The evolutionary mode first builds a probabilistic model about the distribution of good individuals in the search space and then samples a new generation of population using the probabilistic model. With the aid of building a probabilistic model about the

978-0-7695-4253-9/10 $26.00 © 2010 IEEE
DOI 10.1109/TAAI.2010.57

distribution of good individuals, EDA manages to acquire the knowledge for approaching the global optimum in the search space step by step. EDA is an effective algorithm especially for complicated high-dimensional optimization problems [12].

At present, there are a large number of EDAs adopting various probabilistic models, such as the Gaussian distribution model [13][14] and the Bayesian network model [15]. However, some of them such as the Bayesian optimization algorithm (BOA) [15] have high computation demands. To reduce the computation time, we adopt a simple EDA, called population-based incremental learning (PBIL) algorithm [10], to solve the CCPS problem. PBIL is easy to implement and costs less in the computation of probabilistic model than other EDAs.

Various EDAs are also available for different optimization domains. In the application of discrete domain, there are PBIL [10], the univariate marginal distribution algorithm (UMDA) [11], mutual information maximizing input clustering (MIMIC) [16], and BOA [15]. For the continuous domain, the available EDAs include the real-encoded PBIL algorithm [17], the continuous PBIL (PBILc) algorithm [13], and the continuous UMDA (UMDAc) algorithm [14]. Considering that the CCPS problem involves both discrete and continuous optimizations, in our work, the proposed algorithm, termed population-based incremental learning-cardinality constrained portfolio selection (PBIL-CCPS), hybridizes PBIL and PBILc to optimize the selection of assets and the allocation of capital respectively. Moreover, PBIL-CCPS incorporates an adaptive parameter control strategy and an elitist strategy in the process of searching the best portfolio. In experiments, PBIL-CCPS is compared with the GA in [2] and the PSO algorithm in [5] on a collection of instances [2]. The results demonstrate that PBIL-CCPS is a competitive algorithm for portfolio selection and performs well in searching portfolios with high expected returns.

The rest of the paper is organized as follows. Section II gives a definition about the CCPS problem. In Section III, we introduce the proposed algorithm in detail. Section IV reports the experiments and the results. At last, a conclusion for the paper is given in Section V.

II. PROBLEM DEFINITION

In this section, we first describe the MV model. Then, the CCMV model, which is an extension of the MV model, is presented for the CCPS problem.

A. MV Model for the PS Problem

Suppose N is the number of available assets, μ_i is the expected return of asset i, σ_{ij} is the covariance between the returns of assets i and j, the decision variable x_i denotes the proportion of capital invested in asset i. The MV model is defined as (1)-(3).

Minimize

$$f(x_1, x_2, ..., x_N) = \lambda\left[\sum_{i=1}^{N}\sum_{j=1}^{N} x_i x_j \sigma_{ij}\right] - (1-\lambda)\left[\sum_{i=1}^{N} x_i \mu_i\right] \quad (1)$$

Subject to

$$\sum_{i=1}^{N} x_i = 1 \quad (2)$$

$$0 \le x_i \le 1, i = 1, 2, ..., N \quad (3)$$

where $\sum_{i=1}^{N} x_i \mu_i$ represents the expected return of the portfolio and $\sum_{i=1}^{N}\sum_{j=1}^{N} x_i x_j \sigma_{ij}$ represents the total risk (variance of the portfolio's return). The value $\lambda \in [0,1]$ is the weighting parameter which represents a tradeoff between the two objectives of minimizing the risk and maximizing the expected return.

Because of the two conflict objectives, the PS problem belongs to the family of multiobjective optimization problems. Therefore, the optimum portfolio can be defined using the Pareto optimality definition. Suppose that D is the domain of feasible portfolios, (r_i,e_i) denotes any portfolio i in D with a risk r_i and an expected return e_i. The portfolio i is optimum (or nondominated) if there does not exit any portfolio $j \in D$ ($r_i \ne r_j$ and $e_i \ne e_j$) satisfies $r_j \le r_i$ and $e_j \ge e_i$. Otherwise, the portfolio i is defined as dominated [4]. Markowitz [1] proposed a concept of the efficient frontier which is formed by the nondominated portfolios. Fig. 1 gives an example of the efficient frontier presented by the black curve, where the horizontal and vertical axes are the risk and the expected return associated with the portfolio respectively. Based on the MV model, the efficient frontier can be obtained by taking different values for λ and solving exactly the corresponding objective function (1).

B. CCMV Model for the CCPS Problem

For the MV model, there is no restriction on the number of invested assets in the portfolio. The CCMV model improves on the issue and is described as (4)-(8).

Minimize $f(z_1, z_2, ..., z_N, x_1, x_2, ..., x_N) =$

$$\lambda\left[\sum_{i=1}^{N}\sum_{j=1}^{N} x_i x_j \sigma_{ij}\right] - (1-\lambda)\left[\sum_{i=1}^{N} x_i \mu_i\right] \quad (4)$$

Subject to

$$\sum_{i=1}^{N} x_i = 1 \quad (5)$$

$$\sum_{i=1}^{N} z_i = K \quad (6)$$

$$\varepsilon_i z_i \le x_i \le \delta_i z_i, i = 1, 2, ..., N \quad (7)$$

$$z_i \in \{0,1\}, i = 1, 2, ..., N \quad (8)$$

where K is the desired number of invested assets in the portfolio, z_i denotes whether asset i is invested and it equals 1 or 0. If z_i equals 1, asset i is chosen to be invested and the proportion of capital x_i lies between ε_i and δ_i, where $0 \le \varepsilon_i \le \delta_i \le 1$. Otherwise, asset i is not invested and x_i equals 0. In this model, z_i and x_i are both the decision variables.

Compared with the MV model, the CCMV model is more useful in the actual investment. However, the CCPS problem is more complicated to be solved. The efficient frontier of CCPS may be quite different from that of the MV model, due to the presence of the cardinality constraint and the bounds for the proportion of capital associated with each invested asset. Moreover, the efficient frontier may be

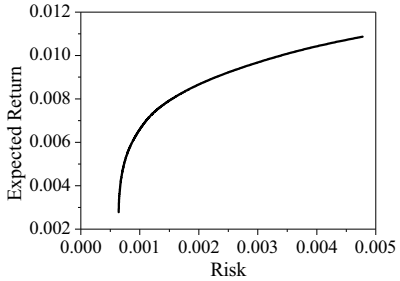

Figure 1. An example of the efficient frontier of the PS problem.

discontinuous [2]. The way of varying λ and solving exactly the corresponding objective function (4) may not be available to obtain the exact efficient frontier any more. On such issue, Chang et al. [2] proposed that an algorithm involving investigating a large number of different solutions could still perform well.

III. PBIL-CCPS FOR CCPS

The CCPS problem can be formulated into a mixed integer quadratic programming problem. It requires optimizing not only the selection of assets but also the allocation of capital. The former is essentially a combinatorial optimization problem in a discrete search space, whereas the latter relates to an optimization in a continuous search space. The proposed PBIL-CCPS algorithm hybridizes PBIL and PBILc to optimize the selection of assets and the allocation of capital respectively. The overall flowchart of PBIL-CCPS is illustrated in Fig. 2.

A. Framework of PBIL-CCPS

In the algorithm, each individual denotes one feasible portfolio and is constructed as a structure containing a selection vector $\vec{z} = (z_1, ..., z_i, ..., z_N)$ and a proportion vector $\vec{x} = (x_1, ..., x_i, ..., x_N)$ where i is the serial number of asset and i=1, 2, ..., N. For $\vec{z}$, each dimention z_i is an integer and equals 0 or 1. If z_i=1, the corresponding asset i is chosen to be invested in the portfolio and the proportion $x_i \in [\varepsilon_i, \delta_i]$. If z_i=0, it denotes that asset i is not invested and x_i is set as 0.

To search the best portfolio, PBIL-CCPS hybridizes PBIL and PBILc. PBIL encodes each individual that denotes a feasible solution as a binary string which consists of 0 and 1, and maintains a real vector, termed a probability vector. The dimension of the vector equals the number of bits in the binary string. Each dimension of the vector denotes the probability of the corresponding bit in the good individual's binary string equaling 1. Like PBIL, PBIL-CCPS maintains a probability vector, whose each dimension describes the probability of the corresponding dimension of $\vec{z}$ equaling 1 in the good individuals.

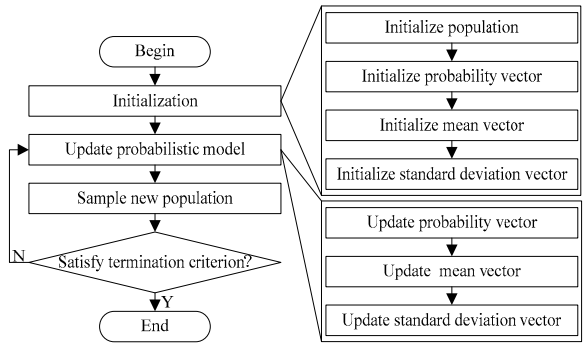

Figure 2. The overall flowchart of the PBIL-CCPS algorithm. The step of initialization consists of initializing the population, the probability vector, the mean vector, and the standard deviation vector. The step of updating the probabilistic model includes updating the probability vector, the mean vector, and the standard deviation vector.

Different from PBIL, PBILc encodes each individual which represents a feasible solution as a real vector. Every dimension of the vector denotes one variable of the problem. The algorithm adopts a Gaussian distribution probabilistic model so that it maintains a mean vector and a standard deviation vector, where each dimension denotes the mean and the standard deviation of the corresponding variable in the good individuals, respectively. Similarly in PBIL-CCPS, each dimension in the mean vector $\vec{X}$ =(X_1,...,X_i,...,X_N) and the standard deviation vector $\vec{\sigma}$ =(σ_1,...,σ_i,...,σ_N) describe the mean and the standard deviation of the corresponding dimension's value in the good individuals' proportion vectors, respectively.

To distinguish the good and bad individuals, the fitness of each individual is evaluated by the fitness evaluation function. While optimizing the objective (4) with a certain value of λ, the fitness evaluation function is defined as (9).

$$f(i) = \lambda^* \left[\sum_{j=1}^{N} \sum_{k=1}^{N} x_{ij} x_{ik} \sigma_{jk} \right] - (1-\lambda^*) \left[\sum_{j=1}^{N} x_{ij} \mu_j \right] \quad (9)$$

where $f(i)$ denotes the fitness of individual i, λ^* is the value of λ, x_{ij} and x_{ik} denotes the j-th and k-th dimensions of the proportion vector in individual i. Obviously, the smaller the fitness is, the better the individual and the corresponding portfolio are. The stepwise procedures of PBIL-CCPS are organized as follows.

Step 1) – Initialization

In the step, the population, probability vector, mean vector, and standard deviation vector are initialized. Firstly, for each individual, randomly choose K assets to be invested. The proportion of capital invested in each chosen asset is initialized to be a random real number between 0 and 1. For the unselected assets, the proportions are initialized to be 0. The portfolio that each constructed individual presents may not satisfy the constraints of the problem. Hence, we need to amend the individual by adjusting the values of the selection vector and the proportion vector. The detail about how to amend an individual will be given in Part B.

Secondly, initialize each dimension of the probability vector to be 0.5.

Finally, initialize the mean vector and the standard deviation vector. Each dimension of the mean vector X_i is initialized as X_i:=(δ_i+ε_i) /2. For the standard deviation vector, each dimension σ_i is initialized as (10).

$$\sigma_i := \sqrt{\sum_{j=1}^{POP} (x_{ji} - X_i)^2 \Big/ POP} \quad (10)$$

where POP is the size of the population, x_{ji} denotes the i-th dimension of the proportion vector in individual j.

Step 2) – Updating the Probabilistic Model

The step updates the probability vector, mean vector, and standard deviation vector according to the current population. The probability vector is updated in the same way as PBIL including the following three steps.

Step A) – Learning from the Best Individual

Each dimension of the probability vector $P(i)$ is updated by learning from the best individual as (11).

$$P(i) := P(i) \times (1 - LR) + best(i) \times LR \quad (11)$$

where $best(i)$ is the i-th dimention of the selection vector in the best individual, LR is the learning rate and $LR \in [0,1]$.

Step B) – Negative Learning

Suppose the i-th dimention of the selection vector in the best individual and the worst individual are $best(i)$ and $worst(i)$, respectively. If $best(i) \neq worst(i)$, the i-th dimention of the selection vector in the optimum may has a greater probability to equal $best(i)$ than $worst(i)$. Therefore, negative learning requires the probability vector to continue learning from the best individual as (12) with a negative learning rate $NEG\_LR \in [0,1]$.

$$P(i) := P(i) \times (1 - NEG\_LR) + best(i) \times NEG\_LR \quad (12)$$

Step C) – Mutation

Each dimension of the probability vector $P(i)$ is altered according to a certain mutation probability PM as (13).

$$P(i) := \begin{cases} P(i) \times (1 - MUT) + MD \times MUT, & rand(0,1) \le PM \\ P(i), & \text{otherwise} \end{cases} \quad (13)$$

where $rand(0,1)$ is a random real number in (0,1], MUT denotes the amount for mutation to affect the probability vector, MD is a random integer equals 0 or 1. Note that, the mutation operates on the probability vector, which differs from that of GA.

The method of updating the mean vector and the standard deviation vector in PBIL-CCPS is similar to that in PBILc. For the mean vector, each dimension X_i is updated as (14) with a learning rate $PLR \in [0,1]$.

$$X_i := (1 - PLR) \times X_i + PLR \times (best_i^{(1)} + best_i^{(2)} - worst_i) \quad (14)$$

where $best_i^{(1)}$, $best_i^{(2)}$, and $worst_i$ denote the i-th dimension of proportion vector in the optimal individual, the suboptimal individual, and the worst individual, respectively.

Analogously, each dimension of the standard deviation vector σ_i is updated as (15).

$$\sigma_i := (1 - PLR) \times \sigma_i + PLR \times \sqrt{\sum_{j=1}^{M} (best_i^{(j)} - \overline{best}_i)^2 \big/ M} \quad (15)$$

where $best_i^{(1)}$, ..., $best_i^{(M)}$ are the i-th dimention of proportion vector in the M best individuals, $\overline{best}_i$ is the mean of $best_i^{(1)}$, ..., $best_i^{(M)}$.

For updating the mean vector and the standard deviation vector, PBIL-CCPS introduces an adaptive parameter control strategy that the value of PLR linearly increases as the number of iterations increases during the optimization process. The strategy is effective for the algorithm to perform sufficient exploration in the search space at the beginning and then for accelerating the convergence speed and improving the accuracy of the solution with a larger learning rate as the evolution continues.

Step 3) – Sampling

Each individual of the new population is constructed with the aid of the probabilistic model. For each individual, firstly construct the selection vector. Supposing that the i-th dimension of the probability vector is $P(i)$, the i-th dimension of the selection vector is set as a random integer which follows the Bernoulli distribution $B(1, P(i))$.

Then construct the proportion vector. If the i-th dimension of the selection vector z_i equals 1, the i-th dimension of the proportion vector x_i is set to be a random real number which follows the Gaussian distribution $N(X_i, \sigma_i^2)$ and satisfies $0 \le x_i \le 1$, where X_i and σ_i denote the i-th dimension of the mean vector and the standard deviation vector, respectively. Otherwise, x_i is set to be 0.

Similar to the initialization process, the constructed individuals must be amended if the representative portfolios do not satisfy the constraints of the problem.

In the step, PBIL-CCPS introduces an elitist strategy. If the best individual in the new sampled population has worse fitness than the global best individual found by the algorithm, then replace R worst individuals by the global best individual. The strategy contributes to ensuring that the global best individual found by the algorithm will not be lost, improving the computational accuracy of the solution, and enhancing the evolutionary level of the new population.

There is a repetition from Step 2 to Step 3 until the fitness evaluation number reaching a predefined value.

B. Amending the Individual

In initializing and sampling the population, each constructed individual must be amended if the representative portfolio does not satisfy the constraints of the problem. The way of amending an individual is similar to that in [5] and consists of two steps.

Step 1) – Adjusting the Number of the Selected Assets

In the first step, the amending operator on the individual ensures that the number of the selected assets equals K. While the number of the selected assets is smaller (larger) than K, the amending operator selects another asset (gives up a selected asset) until the number equals K.

It is useful to exploit the heuristic information in determining which asset to select or give up. In the problem, the asset with a higher expected return and a lower covariance between the returns of itself and other assets, has a greater probability to appear in the best portfolio. Here we define the priority p_i of each asset i to be selected as (16).

$$p_i = \frac{1 + \varphi_i - \min(0, \varphi_1, ..., \varphi_i, ..., \varphi_N)}{1 + \varsigma_i - \min(0, \varsigma_1, ..., \varsigma_i, ..., \varsigma_N)}, i = 1, 2, ..., N \quad (16)$$

where φ_i and ς_i are computed as (17) and (18).

$$\varphi_i = (1 - \lambda) \times \mu_i \quad (17)$$

$$\varsigma_i = \lambda \times \sum_{k=1}^{N} \sigma_{ik} \big/ N \quad (18)$$

If asset i has a higher expected return and a lower covariance, p_i is larger and asset i has a greater priority to be selected. The procedure to ensure the number of the selected assets in individual i equals K is designed as below:

```
Begin
  While ∑_{j=1}^{N} z_ij <K do
    If rand(0,1)>0.5 then j:=randomly choose an unselected asset
    Else j:=choose an unselected asset with a greatest priority
    End If
    z_ij := 1 , x_ij := 0
  End While
  While ∑_{j=1}^{N} z_ij >K do
    If rand(0,1)>0.5 then j:=randomly choose a selected asset
    Else j:=choose a selected asset with a smallest priority
    End If
    z_ij := 0 , x_ij := 0
  End While
End
```

where $rand(0,1)$ is a random real number in (0,1], z_{ij} and x_{ij} are the j-th dimension of the selection vector and the proportion vector in individual i.

Step 2) – Adjusting the Investment Proportions

The procedure to ensure that the proportions of capital invested in the selected assets satisfy the constraints is described as below:

```
Begin
  While true do
    s := Σ_{j=1}^{N} x_ij
    If s = 0 then ∀z_ij = 1, x_ij := 1/K
    Else ∀z_ij = 1, x_ij := x_ij / s
    End If
    Δ := 0, Ψ := 0, H := 0, L := 0
    For j:=1 to N
      If z_ij = 1 then
        h_j := δ_j − x_ij, l_j := x_ij − ε_j
        If h_j < 0 then Δ := Δ + ( x_ij − δ_j )
        Else H:=H + h_j
        End If
        If l_j < 0 then Ψ:= Ψ +( ε_j − x_ij )
        Else L:=L + l_j
        End If
      End If
    End For
    If Δ = 0 and Ψ = 0 then break
    Else
      For j:=1 to N
        If z_ij = 1 then
          If h_j > 0 then x_ij := x_ij + h_j / H × Δ
          Else x_ij := δ_j
          End If
          If l_j > 0 then x_ij := x_ij − l_j / L × Ψ
          Else x_ij := ε_j
          End If
        End If
      End For
    End If
  End While
End
```

The procedure above consists of a repetition whose termination criterion is the proportions satisfying the constraints of the problem. At each iteration, the proportions are firstly adjusted to satisfy (5), and then each proportion associated with each invested asset is amended according to the upper and lower bounds.

IV. Experiments and Comparisons

A. Experiments Description

Experiments take a collection of instances from the OR-Library [18][19], including stocks from different capital market indices around the world, such as the Hang Seng (HS) index in Hong Kong, the Deutscher Aktien IndeX (DAX) in Germany, the Financial Times Stock Exchange (FTSE) index in UK, the Standard & Poor's (S&P) index in USA, and the Nikkei index in Japan. The number of stocks, i.e., the value of N in each instance is 31, 85, 89, 98, and 225 respectively. The detailed data about each stock can be obtained by [20].

In the experiments, PBIL-CCPS is compared with the GA in [2] and the PSO algorithm in [5] which are the two best exiting CI algorithms for the CCPS problem. In the parameter settings for PBIL-CCPS, the number of individuals in the population *POP* is 20, the values of the learning rate *LR* and the negative learning rate *NEG_LR* asscociated with the probability vector are 0.1 and 0.075, respectively. In the mutation on the probability vector, the mutation probability *PM* is 0.02, the amount for mutation to affect the probability vector *MUT* is 0.05. For each value of λ, the learning rate of the mean vector and the standard deviation vector *PLR* is linearly increased in accordance with the number of iterations increases and ranges in [0.05,0.4]. The number M of the best individuals from which the standard deviation vector learns is half of *POP*. And in the elitist strategy, the number R of the worst individuals to be replaced is a quarter of *POP*. The settings of the parameters for PBIL-CCPS are determined based on refering to [21][22] and testing. Whereas, the parameter values of GA and PSO are set as the same as those in [2] and [5] respectively. For each instance, the three algorithms all optimize the objective function (4) by taking 50 different values of λ to obtain the approximative efficient frontier. Each value of λ is set as $\lambda_i=(i-1)/49$, where i=1, 2, ..., 50. For each value of λ, each algorithm terminates when the fitness evaluation number reaches 1000N.

The section consists of two independent experiments. In the first experiment, K=N, ε_i =0, and δ_i =1 (i=1, 2, ..., N), i.e., the number of the invested assets is not constrained and the proportion of capitial associated with each asset is between 0 and 1. As a result, the CCPS problem is same as the PS problem based on the MV model. Whereas in the second experiment, there exits a cardinality constraint that the number of invested assets must equal 10. Moreover, the proportion of capitial associated with each invested asset must lie in [0.01,1].

B. Experiment of PS without Cardinality Constraint

To evaluate the results of the algorithms, we compare them with the standard efficient frontiers of the instances respectively. [20] provides the subset of each instance's standard efficient frontier for the PS problem based on the MV model. The subset includes 2000 distinct portfolios with different combinations of risk and expected return. Based on the portfolios, the approximative efficient frontier can be traced out by the method of the linear interpolation as the standard efficient frontier [2].

In each instance, let the set V consist of 50 obtained portfolios for the 50 objective functions (4) differing from λ. And evaluate V in the way detailedly described in Part C. Table I gives the comparison on the resulting V of each algorithm. For each instance, the three algorithms run 15 trials respectively. The *avgMeanPE* and the *avgMedianPE* respectively denote the average values of the obtained *MeanPE*s and *MedianPE*s of V in 15 trials. In the five instances, the *avgMeanPE* and *avgMedianPE* of PBIL-CCPS are both less than those of GA and PSO except the *avgMedianPE* in HS and the *avgMeanPE* in FTSE. The comparison results demonstrate that evaluating a same number of feasible portfolios, PBIL-CCPS can achieve better solutions.

TABLE I. RESULTS OF PS WITHOUT CARDINALITY CONSTRAINT

Instance		GA	PSO	PBIL-CCPS
HS	*avgMeanPE*[a] (%)	0.0191	0.1422	**0.0003**[b]
	avgMedianPE[a] (%)	0.0166	**1.07×10^{-5}**	1.24×10^{-5}
DAX	*avgMeanPE*(%)	0.0350	1.1044	**0.0023**
	avgMedianPE(%)	0.0124	4.77×10^{-5}	**3.51×10^{-5}**
FTSE	*avgMeanPE*(%)	**0.0109**	1.1430	0.0186
	avgMedianPE(%)	0.0020	0.0084	**2.45×10^{-5}**
S&P	*avgMeanPE*(%)	0.0430	2.0249	**0.0137**
	avgMedianPE(%)	0.0085	0.5133	**2.85×10^{-5}**
Nikkei	*avgMeanPE*(%)	0.3715	8.1781	**0.0606**
	avgMedianPE(%)	0.0068	4.7023	**2.69×10^{-5}**

a. In each instance, the *avgMeanPE* and *avgMedianPE* are respectively the average values of the obtained *MeanPE*s and *MedianPE*s of the algorithm's resulting *V* in 15 trials.
b. The best result among the three algorithms for each instance is bold.

C. Evaluating an Obtained Set of Portfolios

Each obtained portfolio A can be represented by (r,e), where r and e denote the associated risk and expected return. The nearer A is to the standard efficient frontier, the better A is. The closeness between A and the standard efficient frontier is measured by calculating the percentage error of A to the standard efficient frontier like [2].

Let B represent the portfolio with the same expected return as A in the standard efficient frontier, and C represent the one with the same risk as A. Suppose (r_k,e_k) represents any portfolio in the given subset of standard efficient frontier with the risk r_k and the expected return e_k, (r_i,e_i) represents the portfolio satisfies $e_i=\min\{e_k|e_k \geq e\}$ and (r_j,e_j) represents the portfolio satisfies $e_j=\max\{e_k|e_k \leq e\}$ in the subset. If (r_i,e_i) and (r_j,e_j) both exit, the point B (r',e') exits and $e'=e$. Assuming that $s=\sqrt{r}$, $s_i=\sqrt{r_i}$, $s_j=\sqrt{r_j}$, and $s'=\sqrt{r'}$, s' can be acquired by the method of linear interpolation that $s'=s_j+(s_i-s_j)\times(e'-e_j)/(e_i-e_j)$. Note that, e_i-e_j may equal 0. In the case, it is obvious $s'=s_i=s_j$ and $r'=r_i=r_j$. Based on the obtained B, the percentage error in the risk's square root of A to B, termed p_{AB}, is defined to equal $|(s-s')/s'|\times100\%$. p_{AB} will not be calculated if B does not exit.

Similarly, let (r_m,e_m) represent the portfolio satisfies $r_m=\min\{r_k|r_k\geq r\}$ and (r_n,e_n) represent the portfolio satisfies $r_n=\max\{r_k|r_k\leq r\}$. If (r_m,e_m) and (r_n,e_n) both exit, the point C (r'',e'') exits and $r''=r$. Assuming that $s''=\sqrt{r''}=\sqrt{r}$, $s_m=\sqrt{r_m}$, and $s_n=\sqrt{r_n}$, it can be acquired by linear interpolation that $e''=e_n+(e_m-e_n)\times(s''-s_n)/(s_m-s_n)$. In the case that s_m-s_n equals 0, $e''=e_m=e_n$. Then the percentage error in the expected return of A to C, termed p_{AC}, equals $|(e-e'')/e''|\times100\%$. p_{AC} will not be calculated if C does not exit.

Thus, if p_{AB} and p_{AC} are both calculated, the percentage error of A to the standard efficient frontier is the minimum of p_{AB} and p_{AC}. Otherelse, if there is only one calculated, the percentage error will be the calculated one. Or it will not be calculated when neither p_{AB} nor p_{AC} is calculated.

To evaluate an obtained set of portfolios, the mean percentage error *MeanPE* and the median percentage error *MedianPE* are defined as the mean and median of the calculated percentage errors of the portfolios in the set to the standard efficient frontier respectively. The smaller the *MeanPE* and *MedianPE* are, the better the obtained set of portfolios is.

D. Experiment of CCPS

In the experiment, the standard efficient frontier of each instance is unknown. We still use the standard efficient frontiers adopted in the first experiment and evaluate the resulting *V* of each algorithm in the way described by Part C. Note that, for each portfolio in *V*, the obtained percentage error is just an upper bound of the exact percentage error which is unable to obtain [2]. Table II shows the comparison on the average *MeanPE* (*avgMeanPE*) and the average *MedianPE* (*avgMedianPE*) of the resulting *V* in 15 trials.

TABLE II. RESULTS OF CCPS

Instance		GA		PSO		PBIL-CCPS	
		V[a]	***H***[b]	***V***	***H***	***V***	***H***
HS	*avgMeanPE*[c](%)	**1.0993**[e]	0.9518	1.1018	0.8643	1.1026	**0.8472**
	avgMedianPE[c](%)	**1.2181**	1.1845	**1.2181**	1.1243	1.2190	**1.1013**
	Portfolio#[d]	1407		1395		**1540**	
DAX	*avgMeanPE*(%)	3.0479	2.3456	**2.4610**	**1.8493**	2.5163	2.0781
	avgMedianPE(%)	2.5914	2.3098	**2.5544**	**1.7658**	2.5739	2.2783
	Portfolio#	1404		1449		**1933**	
FTSE	*avgMeanPE*(%)	1.1887	0.8737	1.1908	0.8546	**0.9960**	**0.7658**
	avgMedianPE(%)	1.0841	0.6442	1.0841	0.5188	1.0841	**0.4132**
	Portfolio#	1516		1421		**1638**	
S&P	*avgMeanPE*(%)	2.2599	**1.4567**	**2.0530**	1.7289	2.2320	1.6340
	avgMedianPE(%)	1.1972	1.0761	**1.1485**	1.0068	1.1536	**0.8453**
	Portfolio#	1761		1917		**2177**	
Nikkei	*avgMeanPE*(%)	1.7314	0.8637	**0.6101**	**0.4996**	1.0017	0.6451
	avgMedianPE(%)	0.6187	0.6183	0.5907	**0.5335**	**0.5854**	0.5596
	Portfolio#	**2086**		1930		1468	

a. For each algorithm, *V* is the set of the portfolios obtained by solving the objective function (4) with 50 different values for λ.
b. *H* is the set of the portfolios searched during the course of solving the problem.
c. For *V* and *H*, *avgMeanPE* and *avgMedianPE* are respectively the averages of obtained *MeanPE*s and *MedianPE*s in 15 trials.
d. For each algorithm, *Portfolio#* denotes the number of portfolios in the resulting *H* whose *MeanPE* ranks medially in 15 trials.
e. The best result among the three algorithms for each instance is bold.

As stated in Section II, the efficient frontier of CCPS may be discontinuous due to the presence of cardinality constraint and the bounds for the proportion of capital associated with each invested asset. Therefore it is not very appropriate to investigate the effectiveness of PBIL-CCPS solely using the resulting V. Similar to [2], we define a set H. For each value of λ, suppose that $B(\lambda)$ is the current best portfolio found by the algorithm. During the course of evaluating $1000N$ feasible portfolios, a searched portfolio is added to H if it has better fitness than $B(\lambda)$. As a result, the set H will contain a large number of portfolios which are dominated by other portfolios in the set. Therefore, the dominated portfolios must be removed from H. Moreover, in the set H, there will be some equivalent portfolios which share the same risk and expected return. For these equivalent portfolios, only one is reserved in H. The resulting H is evaluated in the same way as evaluating V. For the three algorithms, the comparison on the resulting H is showed in Table II where PBIL-CCPS and PSO achieve better V and H than GA. And for each algorithm, the number *Portfolio#* of portfolios in the resulting H whose *MeanPE* ranks medially in 15 trials, is given in Table II. For each instance, the *Portfolio#* of PBIL-CCPS is larger than those of GA and PSO except Nikkei.

To better investigate the portfolios in the resulting H, in each instance, we choose the resulting H whose *MeanPE* ranks medially in 15 trials for each algorithm, and pool the chosen resulting Hs of the three algorithms to be one set P and remove from P the portfolios which are dominated by other portfolios in P. Table III gives the number of portfolios contributed by each algorithm's resulting H in P. The data show that PBIL-CCPS contributes most portfolios to P except the Nikkei instance. Fig. 3 describes the distribution of portfolios contributed by each algorithm's resulting H in P. The three different kinds of columns correspond to different algorithms. For each instance, the vertical axis lists the intervals of the expected returns, and the horizontal axis of each column in every interval denotes that in P, how many portfolios contributed by the corresponding algorithm's resulting H, have the expected returns lie in the interval. For the DAX, FTSE, and S&P instances, it is obvious that, in the two highest intervals of expected returns, the resulting H of PBIL-CCPS contributes most portfolios to P. The figure demonstrates that PBIL-CCPS performs well in searching the nondominated portfolios with high expected returns.

TABLE III. NUMBER OF PORTFOLIOS IN P[a]

Instance	GA	PSO	PBIL-CCPS
HS	1105	1145	**1292**[b]
DAX	991	1273	**1627**
FTSE	1011	1091	**1303**
S&P	1098	1585	**1736**
Nikkei	1234	**1461**	1218

a. The table lists the number of portfolios contributed by each algorithm's resulting H in P.
b. The best result among the three algorithms for each instance is bold.

V. CONLUSION

In this paper, a novel PS algorithm is proposed to solve the CCPS problem. The algorithm has the features for handling the problems involving both discrete and continuous space by hybridizing a discrete EDA and a continuous EDA. In addition, an adaptive parameter control strategy and an elitist strategy are adopted in the algorithm.

The proposed algorithm is applied to both the PS without cardinality constraint and the CCPS. The results obtained have been compared to those obtained using a GA and a PSO algorithm. The comparisons conclude that the proposed algorithm is a competitive PS algorithm, and it has an advantage in searching the optimum portfolios with high expected returns.

ACKNOWLEDGMENT

This work was supported in part by the National Natural Science Foundation of China No. U0835002; No. 61070004 and No.60673122. by the National High-Technology Research and Development Program ("863" Program) of China No. 2009AA01Z208. Partially supported by an internal grant of the Hong Kong Polytechnic University (Project No. A-PD1X).

REFERENCES

[1] H. Markowitz, "Portfolio selection," Journal of Finance, vol. 7, no. 3, pp. 77-91, 1952.
[2] T.-J. Chang, N. Meade, J. E. Beasley, and Y. M. Sharaiha, "Heuristics for cardinality constrained portfolio optimisation," Computers and Operations Research, vol. 27, no. 13, pp. 1271-1302, 2000.
[3] M. Propato and J. G. Uber, "Booster system design using mixed-integer quadratic programming," Technical Notes, vol. 130, no. 4, pp. 348-352, 2004.
[4] A. Fernández and S. Gómez, "Portfolio selection using neural networks," Computer and Operations Research, vol. 34, no. 4, pp. 1177-1191, 2007.
[5] T. Cura, "Particle swarm optimization approach to portfolio optimization," Nonlinear Analysis: Real World Applications, vol. 10, no. 4, pp. 2396-2406, 2009.
[6] Z. Michalewicz, Genetic Algorithms + Data Structures = Evolution Programs, 3rd ed., New York, USA: Springer, 1996.
[7] R. C. Eberhart and Y.-H. Shi, "Particle swarm optimization: development, applications and resources," Proc. IEEE Conf. Evol. Comput., pp. 81-86, 2001.
[8] S.-H. Huang and H.-C. Zhang, "Artificial neural networks in manufacturing: concepts, applications, and perspectives," IEEE Trans. Components, Packaging and Manufacturing Technology, vol. 17, no. 2, pp. 212-228, 1994.
[9] P. J. M. Laarhoven and E. H. L. Aarts, Simulated Annealing: Theory and Application, Norwell, MA, USA: Kluwer Academic Publishers, 1987.
[10] S. Baluja, "Population-based incremental learning: A method for integrating genetic search based function optimization and competitive learning," Technical Report: CS-94-163, Pittsburgh, PA, USA: Carnegie Mellon University, 1994.
[11] H. Mühlenbein and G. Paaβ, "From recombination of genes to the estimation of distributions I. Binary parameters," Proc. 4th Int. Conf. on Parallel Problem Solving from Nature, London, UK: Springer-Verlag, pp. 178-187, 1996.
[12] R. Santana, P. Larrañaga, and J. A. Lozano, "Protein folding in simplified models with estimation of distribution algorithms," IEEE Trans. Evol. Comput., vol. 12, no. 4, pp. 418-438, 2008.
[13] M. Sebag and A. Ducoulombier, "Extending population-based

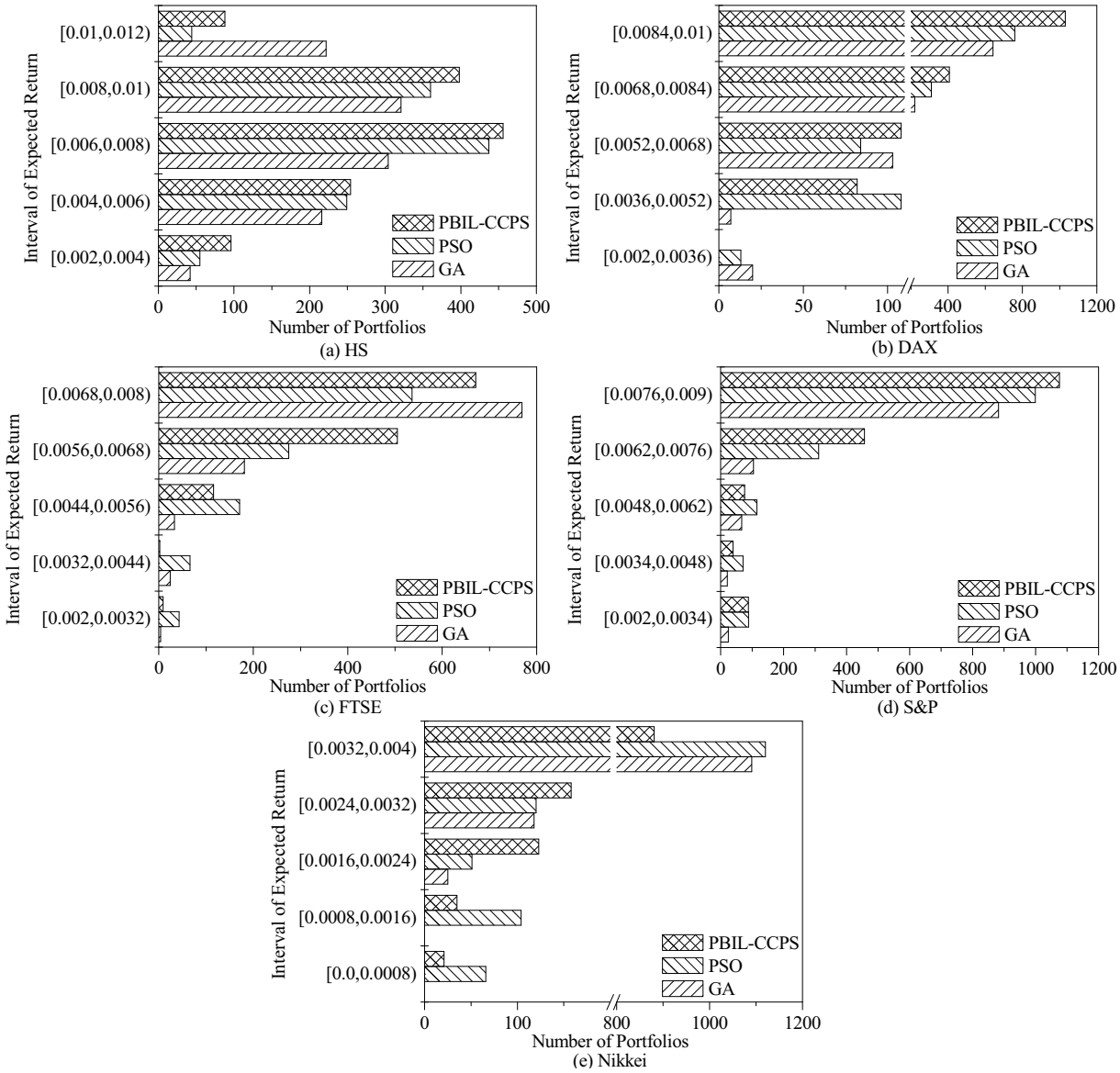

Figure 3. The distribution of portfolios contributed by each algorithm's resulting H in P for each instance. (a), (b), (c), (d), and (e) are for the HS, DAX, FTSE, S&P, and Nikkei instances, respectively. The three different kinds of columns correspond to different algorithms. The vertical axis lists the intervals of the expected returns and the horizontal axis of each column in an interval denotes that in P, how many portfolios contributed by the corresponding algorithm's resulting H, have the expected returns lie in the interval.

incremental learning to continuous search spaces," Proc. 5th Int. Conf. Parallel Problem Solving from Nature, London, UK: Springer-Verlag, pp. 418-427, 1998.

[14] P. Larrañaga, R. Etxeberria, J. A. Lozano and J. M. Peña, "Optimization in continuous domains by learning and simulation of gaussian networks," Proc. Genetic and Evolutionary Computation Conf. Workshop, San Francisco, CA, USA: Morgan Kaufmann, pp. 201-204, 2000.

[15] M. Pelikan，D. E. Goldberg, and E. Cantú-Paz, "BOA: the bayesian optimization algorithm," Proc. Genetic and Evolutionary Computation Conf., San Francisco, CA, USA: Morgan Kaufmann, pp. 525-532, 1999.

[16] J. S. De Bonet, C. L. Isbell, and P. Viola, "MIMIC: finding optima by estimation probability densities," Proc. Advances in Neural Information Processing Systems, Cambridge, MA, USA: MIT Press, pp. 424-431, 1997.

[17] I. Servet, Travé-Massuyès, and D. Stern, "Telephone network traffic overloading diagnosis and evolutionary computation techniques," Proc. 3rd European Conf. Artificial Evolution, London, UK: Springer-Verlag, pp. 137-144, 1997.

[18] J. E. Beasley, "OR-Library: distributing test problems by electronic mail," Journal of the Operational Research Society, vol. 41, no. 11, pp. 1069-72, 1990.

[19] J. E. Beasley, "Obtaining test problems via Internet," Journal of Global Optimization, vol. 8, no. 4, pp. 429-433, 1996.

[20] Available at: http://people.brunel.ac.uk/~mastjjb/jeb/orlib/portinfo.html in July, 2010.

[21] S. Baluja, "An empirical comparison of seven iterative and evolutionary heuristics for static function optimization (extended abstract)," Proc. 11th Int. Conf. Systems Engineering, Las Vegas,

USA: University of Nevada, pp. 692-697, 1996.
[22] K. A. Folly and G. K. Venayagamoorthy, “Effects of learning rate on the performance of the population based incremental learning algorithm,” Proc. Int. Joint Conf. Neural Networks, Piscataway, NJ, USA: IEEE Press, pp. 3477-3484, 2009.