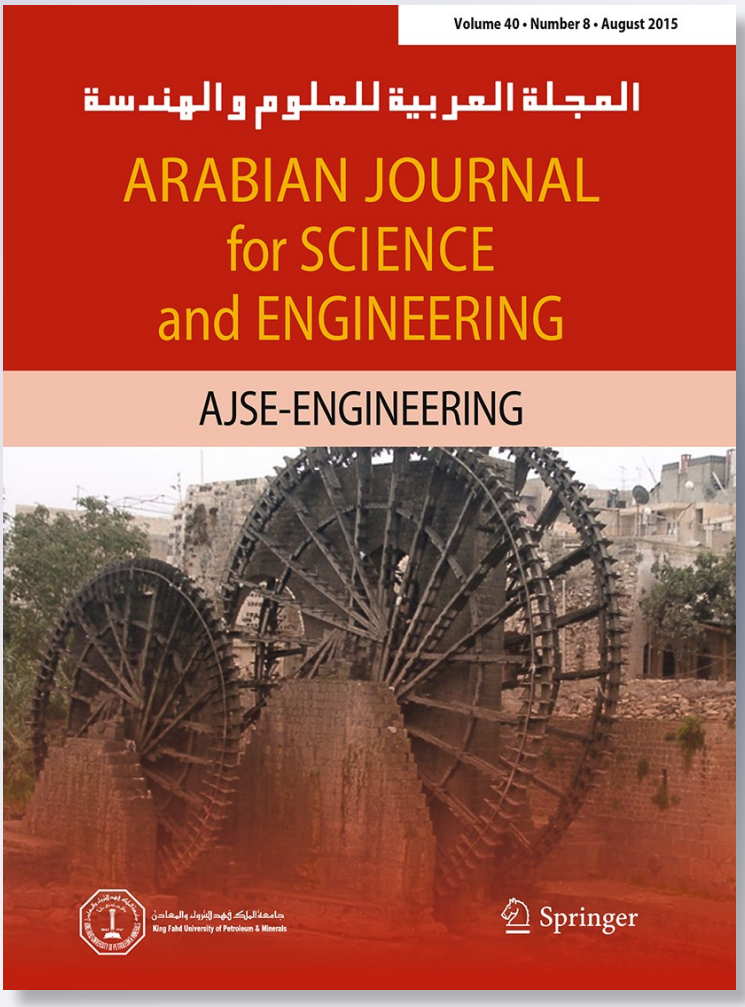
Solving Multi-objective Portfolio Optimization Problem for Saudi Arabia Stock Market Using Hybrid Clonal Selection and Particle Swarm Optimization

Sara A. Bin Shalan & Mourad Ykhlef

Arabian Journal for Science and Engineering

ISSN 1319-8025
Volume 40
Number 8

Arab J Sci Eng (2015) 40:2407-2421
DOI 10.1007/s13369-015-1744-4



Springer

Arab J Sci Eng (2015) 40:2407–2421
DOI 10.1007/s13369-015-1744-4

RESEARCH ARTICLE - COMPUTER ENGINEERING AND COMPUTER SCIENCE

Solving Multi-objective Portfolio Optimization Problem for Saudi Arabia Stock Market Using Hybrid Clonal Selection and Particle Swarm Optimization

Sara A. Bin Shalan[1] · **Mourad Ykhlef**[2]

Received: 31 December 2014 / Accepted: 7 June 2015 / Published online: 27 June 2015

Abstract The portfolio is a group of assets held by an institution or a private individual. Each asset has an investment share of the total investment. The investor tries to distribute the investment to these different assets. The main issue in portfolio optimization is the allocation of different assets for maximum return and minimum risk within a suitable time. These two objectives lead to the multi-objective portfolio optimization problem, which must be solved. Several previous studies have addressed this issue. In this article, we propose a new intelligence hybrid evolutionary algorithm that combines clonal selection with particle swarm optimization to optimize the portfolio's return and risk. We then show the results of the proposed solution through experiments that are conducted using stocks in Kingdom of Saudi Arabia stock exchange market (*Tadawul*). Moreover, we compare our hybrid algorithm, clonal selection and particle swarm optimization-based solution.

Keywords Clonal selection algorithm · Hybrid evolutionary algorithm · Portfolio optimization · Particle swarm optimization algorithm

✉ Mourad Ykhlef
ykhlef@ksu.edu.sa

Sara A. Bin Shalan
saalshalan@ksu.edu.sa

[1] Department of Natural and Engineering Sciences, College of Applied Studies and Community Service, King Saud University, Riyadh, Saudi Arabia

[2] Department of Information Systems, College of Computer and Information Sciences, King Saud University, Riyadh, Saudi Arabia

1 Introduction

The portfolio is "A grouping of financial assets such as stocks, funds or bonds and cash equivalents, as well as their mutual, exchange-traded and closed-fund counterparts. Portfolios are held directly by investors and/or managed by financial professionals" [1]. Each asset in a portfolio has an investment share of the total investment. The investment share is the percent of money invested in the portfolio. Therefore, the investor tries to distribute the investment to these different assets in order to ensure the maximum return and minimum risk. The number of people who participate in the stock exchange market has rapidly increased, which has led to the needs to develop a tool for investors to manage their investment portfolios. Needless to say, the incorrect allocation of investment may lead to big losses. However, if we distribute the investment correctly, the portfolio will contain less risk and accrue high returns. These two objectives, which must be addressed concurrently, lead to the multi-objective portfolio optimization problem. To solve this problem, several models have been developed by using a variety of computational techniques.

In this article, we propose a new intelligence hybrid evolutionary algorithm for portfolio optimization, the portfolio optimization based on clonal selection and particle swarm (POCSPS). Our new algorithm combines clonal selection (CS) with particle swarm optimization (PSO) [2] to optimize the portfolio's return and reduce its risk. The POCSPS will help finance practitioners, researchers and investors in selecting portfolios. We show the results of the proposed solution through experiments that are conducted using the stocks in Kingdom of Saudi Arabia stock exchange market (Tadawul) [3]. The experiments are based on historical data of Saudi stock prices in the 5 years from 2009 to 2013. Moreover, we compare CS [4], a PSO-based solution, and the hybridiza-

tion algorithm, which combines CS and PSO in an optimized portfolio.

Moreover, to our knowledge, this research is the first to apply the portfolio optimization process to the Saudi Arabia stock exchange market, as well as the first application based on the Mean-CVaR [5] portfolio optimization model. In addition, to our knowledge, no previous study has combined clonal selection and particle swarm in the field of portfolio optimization.

Related Work. Intelligent system techniques are computational techniques that were developed by researchers to solve the problem of optimal portfolio selection. Previous researchers developed several intelligent systems, two of which are the particle swarm optimization (PSO) [6,7] and the artificial immune system (AIS) [8]. In addition, many approaches were developed based on hybridization algorithms. In these approaches, more than one evolutionary algorithm is combined. In [2], the authors introduced an algorithm based on the clonal selection algorithm and small population-based particle swarm optimization, which complied with four well-known benchmark functions.

In this study, our goal is to develop and implement our algorithm, portfolio optimization based on clonal selection and particle swarm (POCSPS), to solve the multi-objective portfolio optimization problem. This research will help investors and others interested in the Saudi Arabia stock exchange market. To our knowledge, no previous study has examined this issue, and this is the first attempt to apply a hybridization algorithm that combines clonal selection and particle swarm algorithms to portfolio optimization. Our study offers tools that simplify portfolio creation and management in a timely manner under the assumption that the portfolio returns are subject to heavy tails.

Most previous studies on portfolio optimization, such as [6,9–12] and [13], adopted Markowitz's [14] mean–variance method to solve the optimal portfolio problem. However, following [5], the model adopted in the present study is preferable to the Markowitz model in two ways. First, the Markowitz model suffers from two problems. It supposes the normal distribution of return, but some researchers, such as [15], found that the assets were heavy-tailed and distribution was skewed. In addition, Markowitz's mean–variance model introduces variance in order to describe portfolio risk, which considers the case of a sub-bit value, and does not take into account the limits of part of the loss. Second, the risk described as conditional value at risk (CVaR) is higher than value at risk (VaR), which is useful for investors to their increase their awareness of risk prevention, thus decreasing the investment risk.

Moreover, the hybridization of clonal selection and small population particle swarm algorithms [2] can gather two features, in our algorithm, make it superior to previous studies like [6–8] and [9]. The concept of small population helps to find the optimum solution with less memory requirement, and the concept of clonal selection increases the exploration capability.

The reminder of this article is organized as follows: The portfolio optimization model is described in Sect. 2, and the clonal selection algorithm and particle swarm are defined in Sect. 3. Our approach of applying a hybridization algorithm to the portfolio is described in Sects. 4 and 5. The experimental results and the results of the comparison among the previous clonal selection-based algorithm [4], particle swarm optimization and our hybrid-based algorithm are reported in Sect. 6. The conclusion is provided in Sect. 7.

2 Portfolio Optimization Model

In 1952, Markowitz initiated modern portfolio theory [14]. His work laid the cornerstone for solving the multi-objective portfolio optimization problem. Under the assumption that portfolio returns are subject to heavy tailing, the authors of [5] proposed the optimal portfolio model, which maximizes returns and minimizes risk. In this model, risk is expressed as conditional value at-risk CVaR.

In the following, we explain the calculation of the expected return, the expected risk, and the objective function of a portfolio according to [5].

To calculate the expected return on a portfolio, first we need to calculate the daily yield $y_{i,j}$ for each stock in the period (from j = second day in the period to the last day of the period):

$$y_{i,j} = \frac{p_{i,j} - p_{i,j-1}}{p_{i,j-1}} \tag{1}$$

where $p_{i,j}$ is the previous day's closing price, and $p_{i,j-1}$ is the next day's closing price. Then take the mean of $y_{i,j}$ represented as m.

Second, let $m^T = (m_1, m_2 \ldots m_n)$ are n assets' mean return vector, and $w^T = (w_1, w_2 \ldots w_n)$ is their weight of the investment share.

The weight w has two constraints:

$$(1) \quad \sum_{i=1}^{n} w_i = 1$$

$$(2) \quad 0 \leq w_i \leq 1$$

From the mean return and investment share of each stock in the portfolio, the investor's expected portfolio return, which is hoped to be maximized, is:

$$\text{Max } E(R) = w^T m \tag{2}$$

After calculating the expected return, we calculate the expected risk on a portfolio. First, we need to calculate the portfolio loss function:

$$f(w, y) = -w^T y.$$

where $w^T = (w_1, w_2 \ldots w_n)$ are the asset positions (weights) in the optimal portfolio and $y^T = (y_1, y_2 \ldots y_j)$ is the daily yield of all portfolio assets in the period $j = 1, 2, \ldots J$. Let a be the degree of belief of the objective function, t trading days, b the degree of belief of the CVaR; $u_k = [f(w, y_k) - a]^+$ is the portfolio loss in kth day, and $[x]^+ = \max\{x, 0\}$. From the above, the risk described as CVaR, which was intended to be minimized, is

$$\text{Min } a + \frac{1}{t(1-b)} \sum_{k=1}^{t} u_k \tag{3}$$

CVaR has two constraints:

$$(1) \quad w^T y + a + u_k \geq 0, \quad u_k \geq 0$$

$$(2) \quad \sum_{i=1}^{n} w_i = 1, \quad 0 \leq w_i \leq 1$$

The results of formulas (2) and (3) define the multi-objective optimization problem. To solve it, we use the linear weighted sum method [16] to formulate the objective function, where α_1, α_2 is the investor preference. For example, if the investors are concerned more about income than risk, α_1 is higher than α_2:

$$\text{Max } \alpha_1(w^T m) - \alpha_2 \left(a + \frac{1}{t(1-b)} \sum_{k=1}^{t} u_k \right) \tag{4}$$

Subject to

$$(1) \quad w^T y + a + u_k \geq 0, \quad u_k \geq 0$$

$$(2) \quad \sum_{i=1}^{n} w_i = 1, \quad 0 \leq w_i \leq 1$$

3 Clonal Selection Algorithm and Particle Swarm Algorithm

In this section, we present the descriptions of the two evolutionary algorithms that we use in our work: clonal selection algorithm (CS) and particle swarm optimization (PSO). Both CS and PSO are used to solve optimization problems in polynomial time, but they do not guarantee the optimal solution.

3.1 Clonal Selection Algorithm

Clonal selection algorithms (CSA) are a class of immune algorithms (IA) that are inspired by the clonal selection principle [17]. According to [18], "The clonal selection principle is used to explain the basic features of an adaptive immune response to an antigenic stimulus." When antigens attack the human body, the immune system activates antibodies and begins the cloning process to create a large quantity of antibodies that bind powerfully to a specific antigen. The mutation process of the cloned antibodies is in reverse proportion to their affinity for antigens, the highest affinity trial, the lowest mutation rates and vice versa [8].

The CSA was introduced by Castro and Zuben [18]. It can be defined as an evolutionary algorithm that aims to find a satisfactory solution to solving optimization problems within a reasonable time of execution by using techniques inspired by evolutionary biology, such as cloning and hypermutation.

CSA is composed of two populations: a set of antibodies and a set of antigens. The antibodies that are able to bind successfully to an antigen will be reproduced and maintained as memory cells. The reproduction process uses two clonal selection operators to generate the children of the existing population, which are the clone operator and the hypermutation operator. The process of clone operation duplicates the production of a large quantity of antibodies, and it is applied to each antibody in the current population. Every cloned antibody is then submitted to a maturation process that is inversely proportional to the affinity for antigens.

Portfolio optimization based on clonal selection (POCS) [4] is used in optimizing the portfolio. We applied a common CSA, the CLONALG [17] to the portfolio.

POCS has only one population: antibodies. The antigen is represented by the given objective function. In our algorithm, the antibody population is the set of current nominee solutions, and the antibody represents the portfolio.

In the following, we describe some aspects of our algorithm POCS, which we used for portfolio optimization.

3.1.1 Antibody

In POCS, the antibody (Ab) represents a portfolio with a weight and mean return for each stock, and the expected return, expected risk and fitness function value of the portfolio. The size of the antibody in POCS is determined by the number of stocks in the portfolio, and each weight corresponds to one stock.

3.1.2 Clonal Selection Operator

POCS uses two clonal selection operators to generate the children of the existing population: the cloning operator and hypermutation. The mutation rate formula [18] is

Fig. 1 Pseudo code for POCS algorithm

Input:

Population size: N

Maximum number of generations: Ngen

Maximum number of cloning: Nclo

Number of random antibodies to insert at the end of each generation: d

Antibody represents a portfolio with weight and mean return for each stock, expected return, expected risk and fitness function value for the portfolio.

Output:
Antibody population P(t)

Algorithm:

1. P(0) ← Generate random Ab population of size N
2. For t = 1 to Ngen
3. For each Ab ∈ P(t)
 3.1. Calculate fitness function value (Ab)
4. For each Ab∈ P(t)
 4.1. For j = 1 to Nclo
  4.1.1.P(clo) ← clone(Ab)
5. For each Ab ∈ P(clo)
 5.1. P(hyp) ← hyper mutate (Ab)
6. For each Ab ∈ P(hyp)
 6.1. Calculate fitness function value (Ab)
7. For each Ab ∈ P(t)
 7.1. Ab* ← Select highest fitness function value from P(hyp)
 7.2. If (Ab* > Ab)
  7.2.1. Replace (Ab , Ab*)
8. Abd ← Generate d random Ab
 8.1. Replace (d lowest fitness function value from P(t), Abd)
9. Go to step 2

$$\text{Mutation rate} = \frac{1}{\text{Mutation factor}} * \frac{}{\text{Exp}\,(-1 * \text{Normalized fitness})} \quad (5)$$

3.1.3 Fitness Function

We use Eq. (4) to calculate the fitness function. We suppose that $b = 0.5$. According to our experience, $a = 0.05$.

3.1.4 Pseudo code for POCS

The pseudo code of POCS is shown in Fig. 1. First, population size, maximum number of generations, maximum number of cloning and number of random antibodies to insert at the end of each generation are taken as inputs. Then POCS algorithm works as follows:

1. Randomly generate the Ab population. Every Ab represents a portfolio. Every portfolio has M stocks, and each stock has a random weight (investment allocation).
2. Loop generations from $t = 1$ to Ngen.
3. Determine the fitness function value of all the Abs in $P(t)$ by using Eq. (4).
4. All Abs are cloned (reproduced) independently, generating a repertoire of clones.
5. The repertoire is submitted to the affinity maturation process, which is inversely proportional to the fitness function value, generating a population of matured clones: the higher the fitness function value, the smaller the mutation rate, according to Eq. (5).
6. Determine the fitness function value of the matured clones by using Eq. (4).
7. From this set of mature clones, for each Ab reselect the one with the highest fitness function value. If the fitness function value of this Ab is larger than its respective memory Ab, then it will replace this memory Ab.
8. Finally, replace the Abs with the lowest fitness function value by new random individuals.
9. Go to step 2.

3.2 Particle Swarm Optimization Algorithm

The particle swarm optimization (PSO) was introduced by Kennedy and Eberhart [19]. The PSO is an evolutionary algorithm that aims to find an effective solution to optimization problems. The particles in the swarm work together by exchanging information about the places they have visited and what they discovered in them. Each particle has a position and velocity in a search space and has a neighborhood that is associated with it. The particle moves and remembers

the best position that it has visited and the fitness of the neighborhood. It uses the position of the best fitness to update the velocity.

In each iteration from $i = 1 \ldots$ max population, a particle has to move to a new position by adjusting its velocity according to the following:

$$v_{i+1} = [w * v_i + c1 * \text{rand}() * (p_i - x_i) + c2 * \text{rand}() * (p_g - x_i)] \tag{6}$$

A new position is the old position plus a new velocity:

$$x_{i+1} = [x_i + v_{i+1}] \tag{7}$$

p_i denotes the personal best position:

$$p_{i+1} = \begin{cases} p_i & \text{if } p_i \geq x_{i+1} \\ x_{i+1} & \text{if } p_i < x_{i+1} \end{cases} \tag{8}$$

p_g denotes the best position found by the particles in its neighborhood:

$$p_g = \begin{cases} p_g & \text{if } p_g \geq p_i \\ p_i & \text{if } p_g < p_i \end{cases} \tag{9}$$

where $c1$ and $c2$ are two positive constants as the learning factors, rand() is a random number between 0 and 1, and w is the inertia weight. The most common setting values of $c1$ and $c2$ are $c1 = c2 = 2.0$. Inertia weight w has values between 0 and 1, which are used to eliminate the necessity of checking if v is inside its range.

In the following, we describe some aspects of our algorithm portfolio optimization based on the particle swarm optimization (POPSO) that we used for portfolio optimization.

3.2.1 Particle

We know that the particle (Ps) represents a solution. Thus, the Ps in POPSO represents a portfolio with the mean return for each stock, expected return, expected risk, fitness function value for the portfolio, best fitness and best position, as well as the velocity and position (weight for each stock) of the particle. The dimensions of the Ps in POPSO are determined by number of stocks in the portfolio, where each weight corresponds to one stock.

Table 1 illustrates the particle structure.

3.2.2 Initialization

Initially, many particles are randomly generated by assigning random values to the velocity and position of the particle, that is, the random assignment of weight to each stock in the particle (portfolio) to form the random position. The particle position value is between 0 and 1, and the velocity value is between −1 and 1. The best position (pbest) and global best position (gbest) are initialized by 0.

Moreover, the inertia weight (w) in Eq. (6) has the following value [8]:

$$w = \text{lower bound} + (\text{upper bound} - \text{lower bound}) * \frac{\text{generation}}{\text{generation size}}$$

The upper and lower values are 0 and 1, respectively.

3.2.3 Fitness Function

The quality of the particle or the portfolio is calculated by measuring the fitness function value of the solution. The fitness function is used to measure the performance of the particles: larger values of fitness mean a better solution. We use Eq. (4) to calculate the fitness function, and we suppose that $b = 0.5$. According to our experience, $a = 0.05$.

3.2.4 Pseudo Code of POPSO

The pseudo code of POPSO is shown in Fig. 2. First, the population size and maximum number of generations are taken as inputs. The POPSO algorithm then works as follows:

1. Randomly generate the Ps population as in Sect. 3.2.1. Every Ps represents a portfolio. Every portfolio has M stocks, and each stock has a random weight (investment allocation). The weight represents the particle's position.
2. Loop generations from $t = 1$ to Ngen.
3. Determine the fitness function value of all the Ps in $P(t)$ by using Eq. (4). Then update the best fitness and pbest if the current Ps fitness is higher than the best-use Eq. (8). Update the global fitness and gbest if the best fitness is higher than global fitness by using Eq. (9).
4. For each Ps in the population, update its velocity and position, according to Eqs. (6) and (7) in Sect. 3.2.
5. Go to step 2.

Table 1 Particle structure

Velocity	Position (weight)	Mean return	Expected return	Expected risk	Fitness function value	Best fitness	Best position

Fig. 2 Pseudo code for POPSO algorithm

Input:

population size: N

Maximum number of generations: Ngen

Particle as shown in Table 1 represents a portfolio with mean return for each stock, expected return, expected risk, fitness function value for the portfolio, best fitness and best position. Also, the velocity and position (weight for the stock) of the particle.

Output:

Particle population P(t)

Algorithm:

1. P(0) ← Generate random particle (Ps) population of size N
2. For t = 1 to Ngen
3. For each Ps ∈ P(t)
   3.1. Calculate fitness function value (Ps)
   3.2. If(fitness(Ps) > best fitness(Ps))
      3.2.1. Best fitness(Ps) = fitness(Ps)
      3.2.2. Pbest(Ps) = position(Ps)
   3.3. If(best fitness(Ps) > global fitness)
      3.3.1. Global fitness = best fitness(Ps)
      3.3.2. Gbest = pbest(Ps)
4. For each Ps ∈ P(t)
   4.1. Update velocity and position
5. Go to step 2

4 Portfolio Optimization Using Hybrid Clonal Selection and Particle Swarm

In this section, we describe our portfolio optimization based on the POCSPS algorithm to optimize the portfolio. POCSPS applies a hybrid algorithm composed of clonal selection and small population particle swarm [2] to the portfolio.

We first present our particle structure, and then we initialize our algorithm. We then describe the clonal selection operators and define the fitness function. Finally, we describe the termination of our algorithm.

4.1 Particle

We know that the particle (Ps) represents a solution. Thus, the Ps in portfolio optimization based on POCSPS represents a portfolio with the mean return for each stock, expected return, expected risk, fitness function value of the portfolio, best fitness and best position, as well as the velocity and position (weight for each stock) of the particle. The Ps dimensions in POCSPS are determined by the number of stocks in the portfolio, where each weight corresponds to one stock.

Table 1 illustrates the particle structure.

4.2 Initialization

Initially, many particles are randomly generated by assigning random values to the velocity and position of the particle, that is, the random assignment of weight to each stock in the particle (portfolio) to form a random position. The particle position has a value between 0 and 1, and the velocity value is between -1 and 1. The best position (pbest) and global best position (gbest) are initialized by 0.

The inertia weight (w) in Eq. (6) has the following value [8]:

$$w = \text{Lower bound} + (\text{Upper bound} - \text{Lower bound}) * \frac{\text{Generation}}{\text{Generation size}}$$

The upper and lower values are 0 and 1, respectively.

4.3 Clonal Selection Operators

After selecting the best particles, we apply the clonal selection algorithm that uses clonal selection operators to generate the children of the existing population.

This section describes two operators in the clonal selection algorithm, which we used in POCSPS: cloning and hypermutation. The cloning operator is applied to each particle in the current population by copying each particle without changing it. The number of copies is calculated [2] using the following equation:

$$\text{Number of clone} = \text{Round}\left(\text{Cloning index} * \frac{\text{Total number of particles to be cloned}}{\text{Particle rank according to its fitness}}\right) \quad (10)$$

In the hypermutation operator, every cloned particle is submitted to the maturation process that is inversely proportional to the fitness function value; the higher the fitness function value, the smaller the mutation rate. The mutation formula [20] is

$$C^* = C + \text{Exp}(-\text{Fitness}) * \text{rand} * C + \text{Exp}(-\text{Fitness}) * \text{rand} * (C - \text{gbest}) \quad (11)$$

where C^* = the mutated version of the clone C.

4.4 Fitness Function

The quality of the particle or the portfolio is calculated by measuring the fitness function value of the solution. The fitness function is used to measure the performance of the particles: the bigger the value of the fitness, the better the solution. We use Eq. (4) to calculate the fitness function, and we suppose that $b = 0.5$. According to our experience, $a = 0.05$.

4.5 Termination

The POCSPS algorithm performs the updating process on the best fitness, the best local position (pbest), the global fitness and best global position (gbest) under some conditions. Then each Ps in a population of size N updates its velocity and position if iteration $\neq I$. However, if iteration $= I$, the algorithm selects higher $N/2$ particles. It then creates a number of clones from each particle and mutates each clone. Next, it determines the fitness function value of the matured clone. Lastly, POCSPS generates the $N/2$ number of new random Ps. From the best of these Ps, mutated clones and $N/2$ random Ps, it selects N best Ps, based on their fitness, which are carried over to the next I iterations.

This process is repeated until a termination condition is reached. The maximum generation number could be used as the termination condition. Finally, the best particle of the last iteration is considered the best solution.

5 POCSPS Algorithm for Optimizing the Portfolio

This section presents the algorithm structure of POCSPS, which we produce in addition to the description of its pseudo code.

5.1 Algorithm Structure

POCSPS has one population: particles. The particle population is the set of current nominee solutions, and the Ps represents the portfolio. The flowchart of POCSPS is shown in Fig. 3.

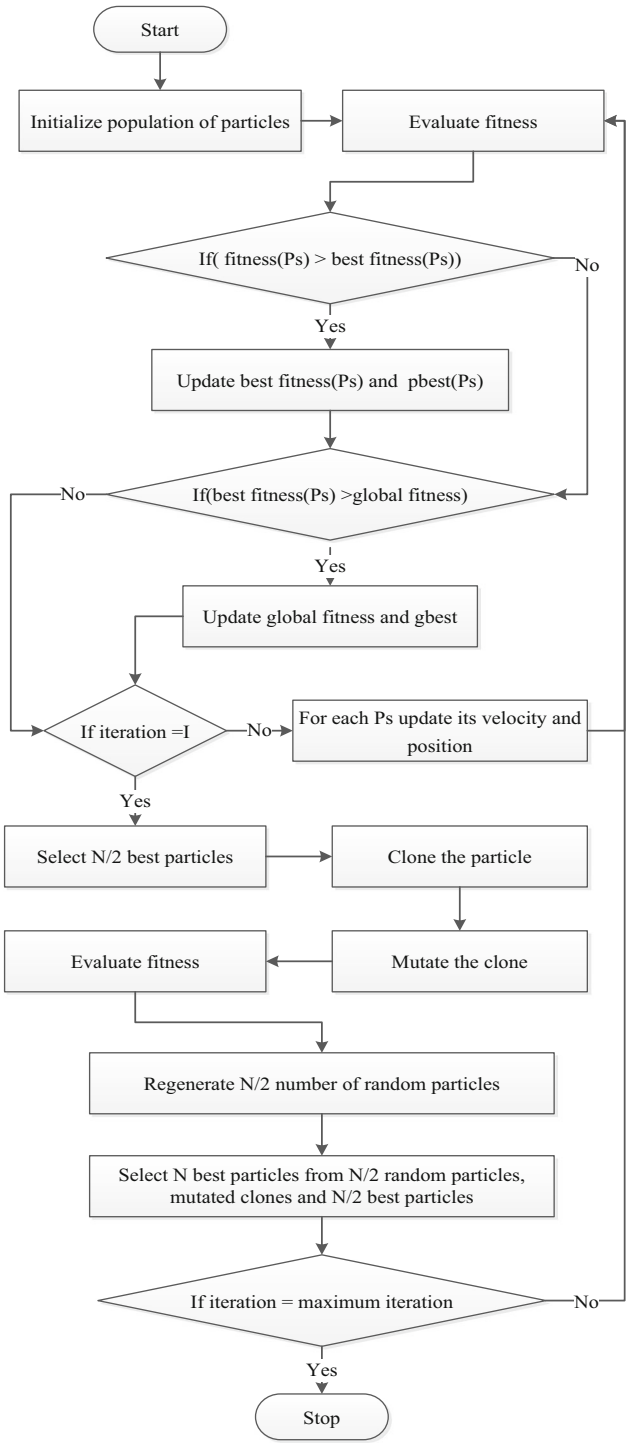


Fig. 3 POCSPS flowchart

First, the population of size N is initialized randomly, and then the algorithm is looped for a predefined maximum number of generations. In the first step, the algorithm determines the fitness function value of each Ps in the population. Next, it updates the best fitness and the best local position (pbest) and determines whether the current Ps fitness is higher than

Input:

population size: N

Maximum number of generations: Ngen

Iteration: I

Particle as shown in Table 1 represents a portfolio with mean return for each stock, expected return, expected risk, fitness function value for the portfolio, best fitness and best position. Also, the velocity and position (weight for the stock) of the particle.

Output:

Particle population P(t)

Algorithm:

1. P(0) ← Generate random particle (Ps) population of size N
2. For t = 1 to Ngen
3. For each Ps ∈ P(t)
   - 3.1. Calculate fitness function value (Ps)
   - 3.2. If(fitness(Ps) > best fitness(Ps))
     - 3.2.1. Best fitness(Ps) = fitness(Ps)
     - 3.2.2. Pbest(Ps) = position(Ps)
   - 3.3. If(best fitness(Ps) > global fitness)
     - 3.3.1. Global fitness = best fitness(Ps)
     - 3.3.2. Gbest = pbest(Ps)
4. If (t = I)
   - 4.1. Select N/2 best Ps
   - 4.2. Create Nclo number of clones from each Ps
   - 4.3. Hyper mutate each clone
   - 4.4. Calculate fitness
   - 4.5. Generate N/2 random Ps
   - 4.6. From N/2 best Ps , Nclo mutated clones and N/2 random Ps, select N best Ps
5. Else
   - 5.1. For each Ps ∈ P(t)
     - 5.1.1. Update velocity and position
6. Go to step 2

Fig. 4 Pseudo code for POCSPS algorithm

the best. The algorithm then updates the global fitness and the best global position (gbest) and determines whether the best local fitness is higher than the global fitness. It then updates the velocity and position of each Ps in the population according to Eqs. (6) and (7) if the iteration $\neq I$. However, if the iteration $= I$, the algorithm selects the highest $N/2$ particles according to their fitness and creates a number of clones from each particle. Each clone then mutates so that it is inversely proportional to the fitness function value: the higher the fitness function value, the smaller the mutation rate. Next, the algorithm determines the fitness function value of the matured clone. Finally, POCSPS generates $N/2$ number of new random Ps. From the $N/2$ best Ps, mutated clones and $N/2$ random Ps, it selects N best Ps based on their fitness, and these are carried over to the next I iterations.

5.2 Pseudo code for POCSPS

In this section, we introduce the pseudo code for POCSPS.

The pseudo code for POCSPS is shown in Fig. 4. First, population size, maximum number of generations and number of iterations (I) are taken as inputs. The POCSPS algorithm then works as follows:

1. Randomly generate Ps population as described in Sect. 4. Every Ps represents a portfolio. Every portfolio has M stocks, and each stock has a random weight (investment allocation). The weight represents the particle's position.
2. Loop generations from $t = 1$ to Ngen.
3. Determine the fitness function value of all the Ps in $P(t)$ by using Eq. (4). Then update the best fitness and pbest if the current Ps fitness is higher than the best using Eq. (8). Then update the global fitness and gbest to determine whether the best fitness is higher than the global fitness by using Eq. (9).
4. If iteration $= I$, select the highest $N/2$ particles according to their fitness and then create N_{clo} number of clones from each particle, where N_{clo} is calculated according to Eq. (10). Then mutate each clone so that it is inversely proportional to the fitness function value: the higher the fitness function value, the smaller the mutation rate. The mutation formula is calculated according to Eq. (11).

Company Code	Date	Price
1010	2009/01/03	21.75
1010	2009/01/04	21.65
1010	2009/01/05	21.80
1010	2009/01/06	21.80
1010	2009/01/07	22.00
1010	2009/01/10	21.50
1010	2009/01/11	21.75
1010	2009/01/12	21.40
1010	2009/01/13	21.40
1010	2009/01/14	21.10
1010	2009/01/17	20.45
1010	2009/01/18	20.40
1010	2009/01/19	20.40
1010	2009/01/20	19.90
1010	2009/01/21	19.85
1010	2009/01/24	20.65

Fig. 5 Example of part from the dataset

Company Code	Yesterday	Today	Day Yield
1010	2009/01/03	2009/01/04	-0.00460
1010	2009/01/04	2009/01/05	0.00690
1010	2009/01/05	2009/01/06	0.00000
1010	2009/01/06	2009/01/07	0.00920
1010	2009/01/07	2009/01/10	-0.02270
1010	2009/01/10	2009/01/11	0.01160
1010	2009/01/11	2009/01/12	-0.01610
1010	2009/01/12	2009/01/13	0.00000
1010	2009/01/13	2009/01/14	-0.01400
1010	2009/01/14	2009/01/17	-0.03080
1010	2009/01/17	2009/01/18	-0.00240
1010	2009/01/18	2009/01/19	0.00000
1010	2009/01/19	2009/01/20	-0.02450
1010	2009/01/20	2009/01/21	-0.00250
1010	2009/01/21	2009/01/24	0.04030

Fig. 6 Day yield in each day

Next, determine the fitness function value of the matured clone by using Eq. (4). Generate $N/2$ number of new random Ps. Select N best Ps from $N/2$ best Ps, N_{clo} mutated clones and $N/2$ random Ps, according to their fitness.

5. If iteration $\neq I$, update the velocity and position of each Ps in the population, according to Eqs. (6) and (7).
6. Go to step 2.

6 Results and Discussion

This section reports the results of the experiments conducted to solve the portfolio optimization problem, which applied the POCS, POPSO and POCSPS algorithms.

All experiments were performed on a 2.90-GHz Intel(R) Core(TM) PC machine with 8 GB RAM and running Microsoft Windows7. The algorithm was written with C# in the Visual Studio.Net 2008 environment. The dataset is a daily adjusted close price of assets in the Saudi Arabia stock exchange market (*Tadawul*) [3] from January 1, 2009, to December 30, 2013, or 1,246 trading days.

We imported the dataset used in this study from historical data of stocks in *Tadawul* [21] using a code that we developed for this purpose. In our experiments, we used the adjusted close price, which was calculated directly from *Tadawul* (Fig. 5). The dataset contained the stocks of 118 companies, all of which had 1246 trading days January 1, 2009, to December 30, 2013. From the adjusted close prices during this period, we calculated the daily yield of each stock by using Eq. (1) (Fig. 6). We also calculated the mean return for each company's stock by taking the average of all daily yields. The results showed the portfolios that had the best fitness function with the investment share for each stock in the portfolio.

We applied the algorithm to nine different stock companies in different market sectors:

- Tabuk Cement Co. (C1)
- Nama Chemicals Co. (C2)
- Al Rajhi Bank (C3)
- Saudi Electricity Company (C4)
- Saudi Re for Cooperative Reinsurance Company (C5)
- Saudi Kayan Petrochemical Company (C6)
- Basic Chemical Industries Co. (C7)
- Saudi Arabia Fertilizers Co. (C8)
- Saudi Basic Industries Corp. (C9)

The mean return (m) in Eq. (4) for each stock is illustrated in Table 2.

6.1 POCSPS

This section reports the results of the experiments conducted to solve the portfolio optimization problem by applying the POCSPS algorithm.

In these experiments, four parameters were determined: population size, number of generations, the cloning index and iteration.

We conducted three experiments. The first and second experiments tested the speed and fitness of POCSPS. The third experiment was conducted to show the change in portfolio allocation according to investor preferences.

In the first experiment, the population was 10 and the generations were [4, 6, 10…100]. In the second experiment, the generation was 10 and the populations were [4, 6, 10…100]. In the third experiment, the population was 6, the generations were 100, and α_1, α_2 of [0.1… 0.9] represented investor preferences. In all experiments, the cloning index was 1 and the iterations were 5.

The results of the first and second experiments were related to increase the population size and the number of gen-

Table 2 Mean return for each stock throw 5 years

1. Tabuk Cement Co	0.0004	2. Saudi Electricity Company	0.0004	3. Basic Chemical Industries Co.	0.0006
4. Nama Chemicals Co.	0.0006	5. Saudi Re for Cooperative Reinsurance Company	0.0005	6. Saudi Arabia Fertilizers Co.	0.0007
7. Al Rajhi Bank	0.0003	8. Saudi Kayan Petrochemical Company	0.0005	9. Saudi Basic Industries Corp	0.0007

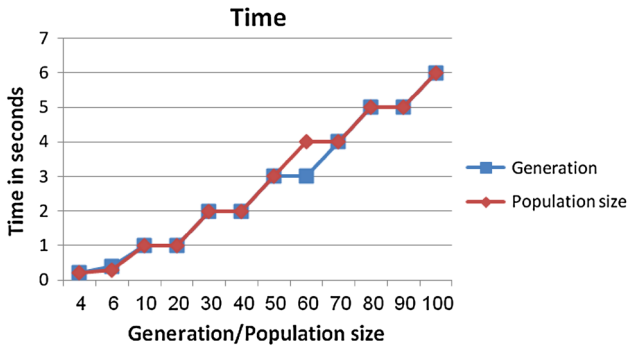


Fig. 7 The effect on time by altering generation/population size

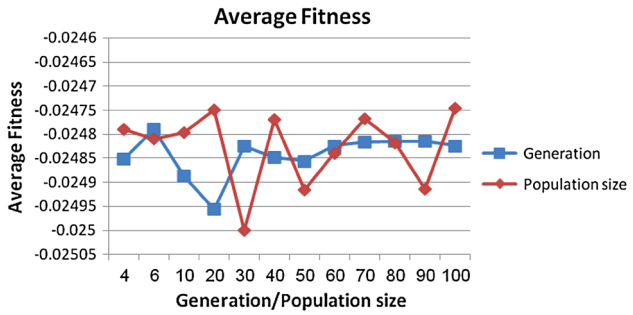


Fig. 8 The effect on average fitness by altering generation/population size

erations. Figure 7 shows the results of the first experiment regarding the time, in seconds, that was spent by the POCSPS to increase the number of generations and population size. Figure 8 shows the results of the second experiment regarding the average fitness found by POCSPS when the population size and number of generations were increased. The fitness value in Eq. (4) is calculated as follows:

$$\text{Max } 0.5(w^T m) - 0.5\left(a + \frac{1}{t(1-b)}\sum_{k=1}^{t} u_k\right)$$

The mean return (m) of each stock is illustrated in Table 2.

The results of the experiments showed that time increased according to the increase in generation and population. The comparison of the results of the two experiments is shown in Fig. 7. As the figure shows, POCSPS took a similar amount of time to increase the population size. The same situation occurred when it determined the average of fitness (Fig. 8). Therefore, when the population increased, the average of fitness increased more than the generation increased. However, the results showed no improvement in small population sizes, such as 4, 6 or 10, but only in medium population sizes, such as 20.

From the results shown in Figs. 7 and 8, we conclude that although the execution time was the same, increasing the

Table 3 Sample shows the best way of distributing the investment to portfolio stocks according to different investor's preferences

Investor preferences	$\alpha_1 = 0.1$ $\alpha_2 = 0.9$	$\alpha_1 = 0.2$ $\alpha_2 = 0.8$	$\alpha_1 = 0.3$ $\alpha_2 = 0.7$	$\alpha_1 = 0.4$ $\alpha_2 = 0.6$	$\alpha_1 = 0.5$ $\alpha_2 = 0.5$	$\alpha_1 = 0.6$ $\alpha_2 = 0.4$	$\alpha_1 = 0.7$ $\alpha_2 = 0.3$	$\alpha_1 = 0.8$ $\alpha_2 = 0.2$	$\alpha_1 = 0.9$ $\alpha_2 = 0.1$
Expected return %	0.05	0.05	0.05	0.05	0.05	0.05	0.05	0.05	0.06
Risk	0.0501	0.0501	0.0501	0.0501	0.0501	0.0500	0.0501	0.0501	0.0501
Fitness	−0.0450	−0.0399	−0.0349	−0.0298	−0.0248	−0.0197	−0.0147	−0.0096	−0.0045
1. C1	05.46	16.54	0.27	12.66	04.97	0.03	02.25	02.52	12.01
2. C2	09.07	11.42	15.29	01.59	02.23	15.91	05.18	09.84	08.59
3. C3	18.78	12.29	14.96	01.62	13.85	10.64	18.73	10.39	01.75
4. C4	09.11	22.26	20.28	16.98	13.49	27.45	17.32	16.34	12.45
5. C5	14.33	13.96	14.89	18.54	06.56	04.97	13.42	13.15	12.12
6. C6	06.81	06.58	02.48	04.77	01.59	01.12	09.27	04.97	08.20
7. C7	04.89	02.44	18.05	25.34	29.17	20.38	05.02	18.51	19.61
8. C8	18.90	01.92	13.50	12.67	18.64	15.11	13.31	07.08	11.05
9. C9	12.65	12.59	0.28	05.83	09.51	04.38	15.50	17.20	14.22

Table 4 The best way of distributing the investment to all stocks

Company Name	% of portfolio	Company name	% of portfolio	Company name	% of portfolio
1. Riyad Bank	0.22	2. Bank AlJazira	0.28	3. The Saudi Investment Bank	0.2
4. Saudi Hollandi Bank	0.02	5. Banque Saudi Fransi	01.1	6. The Saudi British Bank	0.96
7. Arab National Bank	0.15	8. Samba Financial Group	01.07	9. Al Rajhi Bank	0.66
10. Bank AlBilad	0.27	11. Alinma Bank	01.50	12. Advanced Petrochemical Company	01.98
13. Methanol Chemicals Company	0.07	14. Alujain Corporation	0.82	15. Nama Chemicals Co.	0.35
16. National Industrialization Co.	0.10	17. Rabigh Refining and Petrochemical Co.	01.07	18. Sahara Petrochemical Co.	0.74
19. Saudi Arabia Fertilizers Co.	01.33	20. Saudi Basic Industries Corp	01.73	21. Saudi Industrial Investment Group	01.30
22. Saudi International Petrochemical Co.	0.67	23. Saudi Kayan Petrochemical Company	0.27	24. Yanbu National Petrochemical Company	1.06
25. Arabian Cement Co.	02.29	26. Eastern Province Cement Co.	01.54	27. Saudi Cement Company	0.97
28. Southern Province Cement Co.	0.54	29. Tabuk Cement Co.	0.68	30. The Qassim Cement Co.	1.12
31. Yamamah Saudi Cement Co.	1.47	32. Yanbu Cement Co.	0.84	33. Abdullah Al Othaim Markets Company	1.62
34. Aldrees Petroleum & Transport Services Co.	0.52	35. Alkhaleej Training and Education Company	1.53	36. Fawaz Abdulaziz AlHokair Company	0.34
37. Fitaihi Holding Group	0.72	38. Jarir Marketing Co.	0.44	39. National Agriculture Marketing Co.	1.86
40. Saudi Automotive Services Co.	0.86	41. National Gas & Industrialization Co.	1.91	42. Saudi Electricity Company	0.69
43. Al-Jouf Agriculture Development Co.	1.17	44. Almarai Company	0.72	45. Ash-Sharqiyah Development Company	0.61
46. Food Products Co.	0.90	47. Halwani Bros	0.69	48. Jazan Development Co.	0.36
49. National Agriculture Development Co.	1.15	50. Qassim Agriculture Co.	0.56	51. Saudi Fisheries Co.	0.30
52. Saudia Dairy and Foodstuff Co.	0.96	53. Savola Group	1.46	54. Tabuk Agriculture Development Co.	0.76
55. Mobile Telecommunications Company Saudi Arabia	0.32	56. Saudi Telecom	0.72	57. Al Sagr Co-operative Insurance Co	0.45
58. Al-Ahlia Insurance Company	0.70	59. AlAhli Takaful Company	0.34	60. Allianz Saudi Fransi Cooperative Insurance Company	0.36
61. Arabia Insurance Cooperative Company	0.36	62. Arabian Shield Cooperative Insurance Company	1.08	63. Bupa Arabia for Cooperative Insurance	0.42

Table 4 continued

Company Name	% of portfolio	Company name	% of portfolio	Company name	% of portfolio
64. Gulf Union Cooperative Insurance Company	0.42	65. Malath Cooperative Insurance and Reinsurance Company	0.10	66. SABB Takaful	0.13
67. Salama Cooperative Insurance Co.	0.90	68. Sanad Insurance and Reinsurance Cooperative Company	1.53	69. Saudi Arabian Cooperative Insurance Company	0.70
70. Saudi Indian Company for Co-operative Insurance	0.18	71. Saudi Re for Cooperative Reinsurance Company	1.32	72. Saudi United Cooperative Insurance Company	1.00
73. The Company for Cooperative Insurance	1.34	74. The Mediterranean & Gulf Insurance & Reinsurance Co	0.6	75. Trade Union Cooperative Insurance Company	1.00
76. Al-Ahsa Development Co.	0.33	77. Aseer Trading, Tourism & Manufacturing Co.	0.25	78. Kingdom Holding Company	0.56
79. Saudi Advanced Industries Co.	0.23	80. Saudi Arabia Refineries Co.	0.77	87. Saudi Industrial Services Co.	0.11
82. Alabdullatif Industrial Investment CO.	0.78	83. Astra Industrial Group	1.33	84. Basic Chemical Industries Co.	2.22
85. Filing & Packing Materials Manufacturing Co.	1.11	86. National Metal Manufacturing and Casting Co.	1.02	87. Saudi Arabian Mining Company	0.32
88. Saudi Chemical Company	1.03	89. Saudi Industrial Export Co	0.06	90. Saudi Paper Manufacturing Co.	0.23
91. Saudi Pharmaceutical Indust.& Med. Appliances Corp.	1.10	92. The National Co. for Glass Industries	0.90	93. Al-Babtain Power & Telecommunication CO	1.22
94. Arabian Pipes Company	1.47	95. Middle East Specialized Cables Co.	0.73	96. National Gypsum Company	1.54
97. Red Sea Housing Services Company	0.33	98. Saudi Arabian Amiantit Co.	1.58	99. Saudi Cable Company	1.62
100. Saudi Ceramic Co.	1.38	101. Saudi Industrial Development Co.	1.11	102. Saudi vitrified clay pipes co.	0.87
103. Zamil Industrial Investment Co.	0.14	104. Arriyadh Development Co.	0.38	105. Dar Alarkan Real Estate Development Company	1.51
106. Emaar The Economic City	0.45	107. Jabal Omar Development Company	0.11	108. Makkah Construction and Development Co.	1.26
109. Saudi Real Estate Co.	1.52	110. Taiba Holding Co.	1.61	111. Saudi Public Transport Co.	0.68
112. Saudi Transport and Investment Company	0.57	113. United International Transportation Company Ltd.	0.16	114. Saudi Printing & Packaging Company	1.44
115. Saudi Research and Marketing Group	1.97	116. Tihama Advertising & Public Relations Co.	0.69	117. Saudi Hotels & Resort Areas Co.	0.69
118. Tourism Enterprise Co.	1.12				

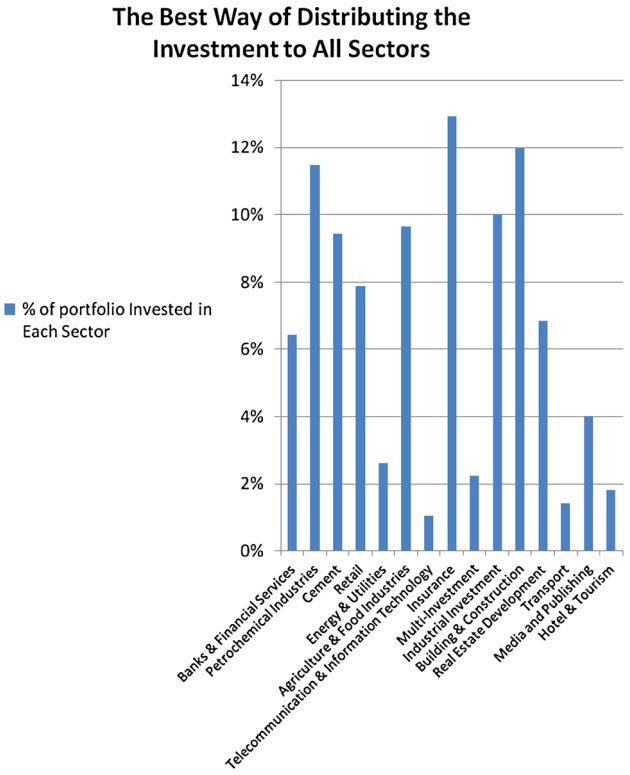


Fig. 9 Percent of investment in each sector

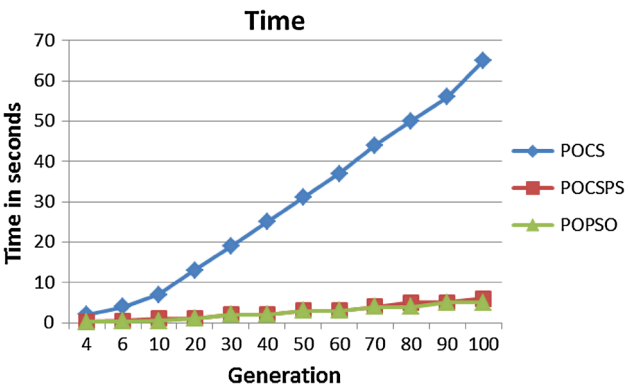


Fig. 10 Time related to generation

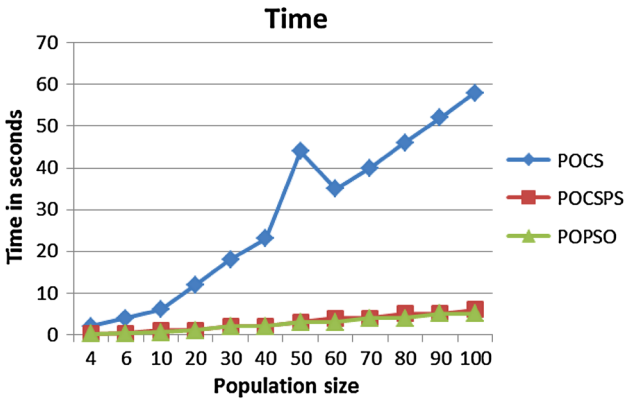


Fig. 11 Time related to population size

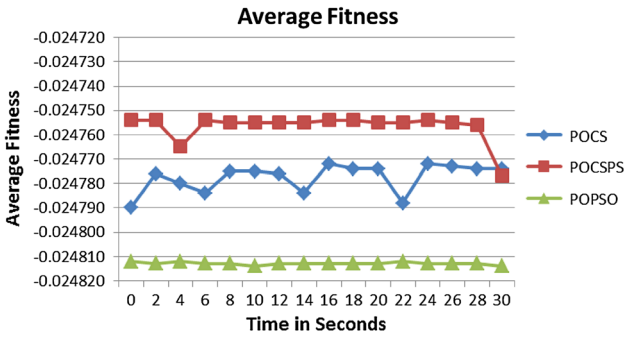


Fig. 12 Average fitness related to time

population yielded better results than increasing the number of generations did, particularly in average fitness.

We also note that the POCSPS algorithm is very fast; the execution time to compile stock prices that were collected every day for 5 years was only 6 s.

Moreover, we know that the investment allocation to different stocks in the portfolio affects both the return of the portfolio and the degree of risk. Table 3 shows the best distribution of the investment to portfolio stocks, which is determined by altering α_1, α_2 values in Eq. (4) according to the investor's preference.

Table 4 and Fig. 9, respectively, show the best way to distribute the investment to all stocks and sectors in *Tadawul*. Only stocks that were available during the period from January 1, 2009, to December 30, 2013, for which all data were available, were used in the test, that is, 118 stocks. From the results, we can conclude that the insurance sector was the best for investing, followed by the building and construction sector.

6.2 Comparison of POCS, POPSO and POCSPS

This section compares the results of the experiments conducted to compare portfolio optimization based on the POCS algorithm [4], the POPSO algorithm and the POCSPS algorithm, which were described in Sect. 3. These results are shown in Figs. 10, 11 and 12 below.

In the experiments, four parameters were determined in POCS: the number of generations, the population size, the number of clones and the mutation factor. Two parameters were determined in POPSO: the number of generations and the population size. Four parameters were determined in POCSPS: the number of generations, the population size, the cloning index and the iteration.

Three experiments were conducted to compare the three algorithms. The first and second experiments compared the

speed of the algorithms, and the third experiment compared the fitness of POCSPS with POCS and POPSO. The results showed that POCSPS was the superior algorithm.

In the first experiment, the population was 10, and the number of generations was 4, 6, 10...100. In the second experiment, the generation was 10 and the population was 4, 6, 10...100. In the third experiment, the population was 10 and the generations occurred in 30 s. In the POCS algorithm, the clone number was 20 and the mutation factor was 80. In the POCSPS algorithm, the iteration was 5 and the cloning index was 1.

Figure 10 shows the results of the first experiment regarding the relation of the POCSPS, POCS and POPSO algorithms increasing the number of generations. Figure 10 also shows the time, in seconds, taken by both algorithms. As the figure shows, POCSPS and POPSO took less time than POCS did. Figure 11 shows the results of the second experiment regarding the relation of the POCSPS, POCS and POPSO algorithms to increasing the population size. The figure shows the time, in seconds, taken by each algorithm. As the figure shows, POCSPS and POPSO took less time than POCS did. Figure 12 shows the results of the third experiment. POCSPS had better average fitness than POCS and POPSO did.

The comparison of the results of the two experiments shown in Figs. 10 and 11, indicates that increasing the number of generations slowed the POCS algorithm more than increasing the population size did. However, in the POCSPS and POPSO algorithms, increasing the number of generations and the size of population took almost the same amount of time to execute.

According to the results of the experiments, the POCSPS algorithm showed better performance than POCS and POPSO did, particularly in average fitness. Moreover, the POCSPS and POPSO algorithms spent less time in execution than the POCS algorithm did. Moreover, increasing the number of generations took more time than increasing the population size did, particularly in POCS.

In brief, the results indicated that the POCSPS algorithm is superior to the POCS and POPSO algorithms. However, although the POCSPS algorithm was the fastest, it is important to choose the right generation and population size.

In our opinion, in portfolio management, it is better to use the POCSPS algorithm with a moderate population size and a high number of generations. This algorithm yields better average fitness, and its execution time is short. This algorithm will potentially guarantee an effective portfolio within a short time.

7 Conclusion

The authors applied a hybridized algorithm POCSPS that combined clonal selection and particle swarm optimization to solve the portfolio optimization problem and determine the best portfolio allocation. Then, they made comparison between it, the clonal selection algorithm POCS and particle swarm optimization POPSO. The experiments were conducted using company stocks in the Kingdom of Saudi Arabia stock exchange market (*Tadawul*). They applied the algorithms to nine different company stocks in different market sectors. The comparison showed that the hybrid POCSPS algorithm performed the best, and the fitness function value increased, particularly in a medium population size. The results showed that POCSPS was more effective and faster than either POCS or POPSO, thus providing a potential solution to the portfolio optimization problem.

Acknowledgments This work was supported by Deanship of Scientific Research and Research Centre of College of Computer and Information Sciences, King Saud University. The authors are grateful for this support.

References

1. http://www.investopedia.com/terms/p/portfolio.asp#axzz2AtSKOowZ
2. Mitra, P.; Venayagamoorthy, G.K.: Empirical study of a hybrid algorithm based on clonal selection and small population-based PSO. In: 2008 IEEE Swarm Intelligence Symposium, St. Louis, MO, USA (2008)
3. http://www.tadawul.com.sa/wps/portal/!ut/p/c1/04_SB8K8xLLM9MSSzPy8xBz9CP0os3g_A-ewIE8TIwP3gDBTA08Tn2Cj4AAvY_dQA_3gxCL9gmxHRQB0Zc_U/
4. Ykhlef, M.; Bin Shalan, S.: Solving portfolio optimization problem by clonal selection algorithm. In: The 2nd International Conference on Smart Media and Application, Kota Kinabalu (2013)
5. Yu, X.; Tan, Y.; Liu, L.; Huang, W.: The optimal portfolio model based on mean-Cvar with linear weighted sum method. In: 2012 Fifth International Joint Conference on Computational Sciences and Optimization, Harbin (2012)
6. Talebi, A.; Molaei, M.A.; Sheikh, M.J.: Performance investigation and comparison of two evolutionary algorithms in portfolio optimization: Genetic and particle swarm optimization. In: 2010 2nd IEEE International Conference on Information and Financial Engineering (ICIFE), Chongqing (2010)
7. Eshlaghy, A.T.; Abdolahi, A; Moghadasi, M.; Maatofi, A.: Using genetic and particle swarm algorithms to select and optimize portfolios of companies admitted to Tehran stock exchange. Res. J. Int. Stud. **20**, 95 (2011)
8. Abbas, A.; Haider, S.: Comparison of AIS and PSO for constrained portfolio optimization. In: 2009 International Conference on Information and Financial Engineering, Singapore (2009)
9. Hoklie, Z.L.R.: Resolving multi objective stock portfolio optimization problem using genetic algorithm. In: 2010 The 2nd International Conference on Computer and Automation Engineering, Singapore (2010)
10. Chen, A.H.L.; Yun-Chia, L.; Chia-Chien, L.: An artificial bee colony algorithm for the cardinality-constrained portfolio optimization problems. In: 2012 IEEE Congress on Evolutionary Computation. Brisbane, QLD (2012)
11. Adewumi, A.; Moodley, A.: Comparative results of heuristics for portfolio selection problem. In: 2012 IEEE Conference on Computational Intelligence for Financial Engineering & Economics. New York, NY (2012)

12. Soam, V.; Palafox, L.; Iba, H.: Multi-objective portfolio optimization and rebalancing using genetic algorithms with local search. In: 2012 IEEE Congress on Evolutionary Computation. Brisbane, QLD (2012)
13. Levy, M.; Ritov, Y.: Portfolio optimization with many assets: the importance of short-selling. University of California at Los Angeles, Anderson Graduate School of Management (2001)
14. Markowitz, H.: Portfolio selection. J. Finance **7**(1), 77–91 (1952)
15. Heyde, C.C.; Kou, S.G.: On the controversy over tail weight of distributions. Oper. Res. Lett. **32**, 399–408 (2004)
16. Chen, J.; Li, L.: The portfolio model based on geometric Brownian motion. Pract. Underst. Math. **38**, 24–26 (2008)
17. Cutello, V.; Narzisi, G.; Nicosia, G.; Pavone, M.: Clonal selection algorithms: a comparative case study using effective mutation potentials. In: 4th International Conference on Artificial Immune Systems. Banff, Alberta (2005)
18. de Castro, L.N.; Von Zuben, F.J.: Learning and optimization using the clonal selection principle. IEEE Trans. Evol. Comput. **6**, 239–251 (2002)
19. Kennedy, J.; Eberhart, R.C.: Particle swarm optimization. In: Proceedings of IEEE International Conference on Neural Networkings. Perth, WA (1995)
20. Ramaswamy, S.A.P.; Venayagamoorthy, G.K.; Balakrishnan, S.N.: Optimal control of class of non-linear plants using artificial immune systems: application of the clonal selection. In: IEEE 22nd International Symposium on Intelligent Control. Singapore (2007)
21. http://www.tadawul.com.sa/wps/portal/!ut/p/c1/04_SB8K8xLLM9MSSzPy8xBz9CP0os3g_A-ewIE8TIwMLj2AXA0_vQGNzY18g18cQKB-JJO8eEGZq4GniE2wUHOBlbOBpREB3cGKRvp9Hfm6qfkFuRDkAgpcLJw!!/dl2/d1/L2dJQSEvUUt3QS9ZQnB3LzZfTjBDVlJJNDIwR05QOTBJSzZFSUlEUjBHRzA!/