OPTIMAL PORTFOLIO SELECTION BY CVAR BASED SHARPE RATIO — GENETIC ALGORITHM APPROACH —

Shan LIN and Masamitsu OHNISHI

Received July 20, 2006; revised August 10, 2006

Abstract. We address an optimal portfolio selection problem of maximizing so we call CVaR (Conditional Value-at-Risk) based Sharpe ratio of return rate of portfolio, which is defined as the ratio of the expected excess return to CVaR. The Sharpe ratio defined as the ratio of expected excess return to standard deviation, the most common traditional performance measure, takes standard deviation as a risk measure, however, its has been received a lot of criticisms. In our CVaR based Sharpe ratio, the standard deviation is replaced by CVaR, which is a remarkable coherent risk measure which overcomes essential defects of standard deviation. Although our new performance measure is expected to enlarge the applicable area of practical investment problems for which the original Sharpe ratio is not suitable, however, we should device effective computational methods to solve optimal portfolio selection problems with very large number of investment opportunities.

In order to deal with rather complicated non-concave objective function, which comes from the introduction of CVaR, in this paper, we propose a Genetic Algorithm (GA) approach. The paper briefly reviews the literature in the area of application of Genetic Algorithms to financial problems, and then details the development of Genetic Algorithm for portfolio selection. In order to evaluate CVaR for each portfolio, by utilizing the results of Rockafellar and Uryasev (2000), we introduce an auxiliary decision variable to obtain a tractable concave maximization problem. Furthermore, if we estimate or approximate required expected values by sampling methods or historical data, we can reduce this concave maximization problem to an LP (Linear Programming) problem. Therefore, our problem could be solved by GA which incorporates LP for evaluating values of CVaR. Numerical experiments from real Japanese financial data are conducted to test our approach to conclude that, by suitable choice of probability parameter of three evolution operation, we could effectively solve optimal selection problems with practical sizes.

1 Introduction The measure of improving the investment performance, which is called Sharpe Ratio, was presented by W.Sharpe[1952]. Sharpe ratio is the ratio of the average excess return to the variance of the return of portfolio. Using the risk factor-variance, the risk of portfolio investment can be measured by calculating the variance of the return of portfolio. However, variance can describe the risk completely only in the normal world. It means that risk can be presented by variance in the case of the expect return of portfolio following the normal distribution. Value-at-Risk(VaR), a widely used performance measure, can solve the problem: what is the maximum loss with a specified confidence level? In most cases, approaches to calculate VaR rely on linear approximation of risks and assume the joint normal (or log-normal) distribution of the underlying market parameters. Although VaR is a very popular measure of risk, it has undesirable mathematical characteristics such as lack of subadditivity [1] and convexity [Artzner et al.1997,1999]. VaR is coherent only when it is based on the standard deviation of normal distributions (VaR is proportional to the standard deviation for a normal distribution). Furthermore, VaR is difficult to optimize when it is calculated from scenarios. In that case, VaR is a non-convex, non-smooth function of positions, and it has multiple local extreme [2].

2000 *Mathematics Subject Classification.* JEL classification: G11; G12; D81.

Key words and phrases. Optimal Portfolio Selection; Sharpe Ratio; CVaR(Conditional Value-at-Risk); GA(Genetic Algorithms); LP(Linear Programming).

[1]For example, VaR associated with a combination of two portfolios can be deemed greater than the sum of the risks of the individual portfolios.

[2]Mauser and Rosen[1999] and McKay and Keeger[1996] showed that VaR can be ill-behaved as a function of portfolio positions and can exhibit multiple local extreme, which can be a major handicap in trying to determine an optimal mix of positions or even the VaR of a particular mix.

As an alternative measure, Conditional Value-at-Risk(CVaR), being also called mean excess loss, mean shortfall, or tail VaR, is commonly considered to be more consistent measure of risk than VaR [3]. Pflug[2000] proved that CVaR is a coherent risk measure with the following properties : transition-equivalent, positively homogeneous, convex, monotonic w.r.t. stochastic dominance of order 1, and monotonic w.r.t. monotonic of order 2. Because of the coherent of CVaR, some researcher use CVaR to solve the optimization problem and other problems in the field of financing engineering. For example, a simple description of the approach for minimization of CVaR and optimization problems with CVaR constrains were addressed in the review paper by Uryasev[2000]. Although CVaR has not become a standard in the finance industry, CVaR is increasingly used in the insurance industry [4]. Bucay and Rosen[1999] used CVaR in credit risk evaluations. Andersson and Uryasev[1999] had research on the application of CVaR methodology to evaluate credit risk. The conditional expectation constrains and integrated chance constrains were described by Prekopa[1995].

Rockafellar R.T.and S.Uryasev[2000] presented that minimizing CVaR of a portfolio is closely related to minimizing VaR, simultaneous VaR can be calculated by minimizing of CVaR. And they also pointed that similar to the Markowitz mean-variance approach, CVaR can be used in return-risk analysis. They calculated a portfolio with a specified return and minimal CVaR. Alternatively, it was tried to constrain CVaR and to find a portfolio with maximal return by Palmquist,J., Uryasev,S., and P.Krokhmal[1999], rather than constraining the variance, specify several CVaR constraints simultaneously with various confidence levels. At the same time, numerical algorithms to solve the minimization problem of CVaR are able to make use of special mathematical features in the portfolio and can readily be combined with analytical or simulation-based methods. The uncertainty is modelled by scenarios and a finite family of scenarios is selected as an approximation. The problem can even be reduced to one linear programming [5].

However, in order to create a powerful risk management tool, we should face risk and return, simultaneously. Risk and return are always a couple of conflict. Many former researches, such as Rockafellar R.T. and S.Uryasev[2000], Palmquist, J.,Uryasev,S., and P.Krokhmal[1999], and so on, focused on this problem but most of them emphasized one side, in other words, they omitted the mutual effect of the two items [6]. If we would like to investigate the mutual effect of them, a new method must be proposed. The new objective function must include both risk and return.

Then this new objective function should be written just a little bit the same as the Sharpe Ratio, but different from that, this is the ratio of the average excess return to CVaR. In this function, numerator is return and denominator is risk. Maximum objective function means there is rather large return while fairly small risk is obtained. This optimization considers both return and risk, of which changes will effect the final result. Both return and risk are function of independent variable x_i, [7]which implies that between them there are clear relationships. Because both numerator and denominator consist of x_i, this objective function is not linear function of them. So solution of this function cannot rely on linear theory. New method must be introduced.

Because of large numbers of the unknown, it is rather difficult to establish a series of equations to obtain theoretical solution of optimization problem. For this reason, to establish computational method to achieve proper solution without any limitation of the unknown is always magnetic challenge.

Here, we present a joint computational method to deal with the non-linear objective function. The joint computational method is that linear programming and genetic algorithms are be used together. The capital markets have numerous areas with potential applications for soft computing techniques. Given this potential and the impetus on the technologies during the last decade, a number of studies have focused on capital market application and in many cases have showed better performance than competing approaches. Among these the one that was found to be of greatest use is the application of

[3] Artzner at al.[1997)] and Embrechts et al.[1999].

[4] see Embrechts et al.[1997].

[5] Zenios[1996] and Ziemba and Mulvey[1998]

[6] In the paper of Rockafellar R.T. and S.Uryasev [2000], it is obviously that return is fixed and at the same time only optimizes risk.

[7] x_i is the weight of one instrument in a portfolio

genetic algorithms in portfolio optimization problems.

The substantial analysis in this paper by using the joint computational method will be conducted. We calculated the optimal investment weight and the optimal number of securities in risky investment by using the computational method. The process of simulation will be given in detail in the section 6 of our paper.

The paper is organized as follows. In Section 2, we explain what is downside risk measures-VaR and CVaR; Sample approach will be presented in Section 3; we introduce the joint computational method to deal with the optimization problem in section 4; in section 5, substantial analysis will be given; in section 6, empirical analysis will be given, we will give a conclusion.

2 VaR and CVaR Let $\widetilde{r}$ denote a random variable denoting a return rate of an asset or a portfolio of assets. Value at Risk (VaR) of $\widetilde{r}$ with confidence level $\beta \in [0,1]$, denoted $\mathrm{VaR}_\beta[\widetilde{r}]$, is defined as the negative of $(1-\beta)$–quantile of $\widetilde{r}$;

$$\begin{aligned}\mathrm{VaR}_\beta[\widetilde{r}] &:= -\inf\{r \in \mathbb{R} : \mathbb{P}(\widetilde{r} \le r) \ge 1-\beta\} \\ &= \sup\{u \in \mathbb{R} : \mathbb{P}(-\widetilde{r} \ge u) \ge 1-\beta\}. \end{aligned} \tag{1}$$

Conditional Value at Risk (CVaR) of $\widetilde{r}$ with confidence level $\beta \in [0,1]$, denoted $\mathrm{CVaR}_\beta[\widetilde{r}]$, is then defined as follows:

$$\mathrm{CVaR}_\beta[\widetilde{r}] := \frac{1}{1-\beta}\int_0^{1-\beta} \mathrm{VaR}_\alpha[\widetilde{r}]\mathrm{d}\alpha. \tag{2}$$

It can be shown that

$$\mathrm{CVaR}_\beta[\widetilde{r}] = \frac{1}{1-\beta}\mathbb{E}\left[-\widetilde{r}; -\widetilde{r} \ge \mathrm{VaR}_\beta[\widetilde{r}]\right] - \mathrm{VaR}_\beta[\widetilde{r}]\left\{\mathbb{P}\left(-\widetilde{r} \ge \mathrm{VaR}_\beta[\widetilde{r}]\right) - (1-\beta)\right\}, \tag{3}$$

where $\mathbb{E}[Y;A]$ denotes the partial expectation of a random variable Y on an event A; that is $\mathbb{E}[Y;a]; = \mathbb{E}[Y 1_A]$. Although this expression is somewhat complex, if

$$\mathbb{P}\left(-\widetilde{r} \ge \mathrm{VaR}_\beta[\widetilde{r}]\right) = 1-\beta, \tag{4}$$

then the second term vanishes and it becomes

$$\mathrm{CVaR}_\beta[\widetilde{r}] = \frac{1}{1-\beta}\mathbb{E}\left[-\widetilde{r}; -\widetilde{r} \ge \mathrm{VaR}_\beta[\widetilde{r}]\right] = \mathbb{E}\left[-\widetilde{r}| -\widetilde{r} \ge \mathrm{VaR}_\beta[\widetilde{r}]\right]. \tag{5}$$

Average Value at Risk (AVaR), Expected Shortfall (ES), Tail Conditional Expectation (TCE), and others are similar concepts, not few researchers prefer one of these terms to CVaR, but these become identical when the above condition holds (whose sufficient condition is the continuity of cumulative distribution function (cdf) of $\widetilde{r}$).

A very useful characterization is obtained by Pflug (2000), Uryasev (2000), Rockafellar and Uryasev (2000, 2001). Let us introduce a function:

$$F_\beta(a;\widetilde{r}) := a + \frac{1}{1-\beta}\mathbb{E}\left[(-\widetilde{r}-a)^+\right], \quad a \in \mathbb{R}, \tag{6}$$

then the following theorem holds (for a real number $c \in \mathbb{R}$, $(c)^+ := \max\{c, 0\}$ is the positive part of c).

Theorem 1

(1) $\mathrm{CVaR}_\beta[\widetilde{r}]$ coincides with the minimum of function $F_\beta(\cdot;\widetilde{r})$:

$$\mathrm{CVaR}_\beta[\widetilde{r}] = \min\{F_\beta(a;\widetilde{r}) : a \in \mathbb{R}\}. \tag{7}$$

(2) The minimum of function $F_\beta(\cdot;\widetilde{r})$ is attained at when the variable is equal to $\mathrm{VaR}_\beta[\widetilde{r}]$:

$$\min\{F_\beta(a;\widetilde{r}) : a \in \mathbb{R}\} = F_\beta\left(\mathrm{VaR}_\beta[\widetilde{r}(\boldsymbol{x})];\widetilde{r}\right). \tag{8}$$

(3) $F_\beta(a;\widetilde{r})$ is convex both in $a \in \mathbb{R}$ and $\widetilde{r}$.

This is particularly useful when we must consider the minimization of $\mathrm{CVaR}_\beta[\widetilde{r}(\boldsymbol{x})]$ of return rate

$$\widetilde{r}(\boldsymbol{x}) := \widetilde{\boldsymbol{r}}^\top \boldsymbol{x} = \sum_{i=1}^{n} \widetilde{r}_i x_i \tag{9}$$

of portfolio $\boldsymbol{x}$ over a convex constraint set $X \subset \mathbb{R}^n$. According to the definition of $\mathrm{CVaR}_\beta[\widetilde{r}(\boldsymbol{x})]$, for every evaluation of objective function at $\boldsymbol{x} \in X$, we must evaluate the values in the order:

$$(1) \quad \mathrm{VaR}_\beta[\widetilde{r}(\boldsymbol{x})] \quad \Longrightarrow \quad (2) \quad \mathrm{CVaR}_\beta[\widetilde{r}(\boldsymbol{x})], \tag{10}$$

but these are tremendous tasks. The following theorem implies that the evaluation and minimization of $\mathrm{CVaR}_\beta[\widetilde{r}(\boldsymbol{x})]$ can be done by the joint minimization of function $F(a;\widetilde{r}(\boldsymbol{x}))$ with respect to the original decision variable $\boldsymbol{x} \in X$ and an auxiliary variable $a \in \mathbb{R}$.

Theorem 2

(1)

$$\min\{\mathrm{CVaR}_\beta[\widetilde{r}(\boldsymbol{x})] : \boldsymbol{x} \in X\} = \min\{F_\beta(a;\widetilde{r}(\boldsymbol{x})) : a \in \mathbb{R};\ \boldsymbol{x} \in X\}. \tag{11}$$

(2) For $\boldsymbol{x}^* \in X$,

$$\min\{\mathrm{CVaR}_\beta[\widetilde{r}(\boldsymbol{x})] : \boldsymbol{x} \in X\} = \mathrm{CVaR}_\beta[\widetilde{r}(\boldsymbol{x}^*)] \tag{12}$$

if and only if

$$\min\{F_\beta(a;\widetilde{r}(\boldsymbol{x})) : a \in \mathbb{R};\ \boldsymbol{x} \in X\} = F_\beta\left(\mathrm{VaR}_\beta[\widetilde{r}(\boldsymbol{x}^*)];\widetilde{r}(\boldsymbol{x}^*)\right). \tag{13}$$

(3) $F_\beta(a;\widetilde{r}(\boldsymbol{x}))$ is convex both in $a \in \mathbb{R}$ and $\boldsymbol{x} \in X$.

Accordingly, the original convex programming problem with $n+1$ decision variables;

$$\left| \begin{array}{ll} \text{Minimize} & \mathrm{CVaR}_\beta[\widetilde{r}(\boldsymbol{x})] \\ \text{subject to} & \boldsymbol{x} \in X, \end{array} \right. \tag{14}$$

could be reduced to the following convex programming problem with $n+1$ decision variables;

$$\left| \begin{array}{ll} \text{Minimize} & F_\beta(a;\widetilde{r}(\boldsymbol{x})) := a + \dfrac{1}{1-\beta}\mathbb{E}\left[(-\widetilde{r}(\boldsymbol{x}) - a)^+\right] \\ \text{subject to} & a \in \mathbb{R}; \\ & \boldsymbol{x} \in X, \end{array} \right. \tag{15}$$

which is more tractable than the original problem.

3 Algorithms for Optimization of Excess Return-CVaR Genetic Algorithm is a problem-solving technique that evolves solution an nature does, rather than looking for solutions in a more principled manner. This makes it an apt procedure that can be used for portfolio selection problem. Portfolio optimization and selection is a complex task because of the following major problems:

- The availability of a wide range and variety of opportunities to choose from. These include various securities among of debt, equity, option, stock, etc;
- Proportion to be invested in each of selected assets. The problem of choosing a major component of the total investment so as to minimize the risk of the investment and maximize the return of that at the same time.
- Timing of transaction based on the market movements. This leads to a complex decision that the investor has to take as to when he should buy and when should he sell;
- The constraints and objectives of the investors, like growth, regular income, tax to be paid, liquidity, etc;
- In addition to the above four problems, stocks markets are more volatile, have less depth and are less transparent which would turn the investment decision process into herculean task.

In this paper, we employ measure of risk and return which are given by the excess expect return and CVaR(Conditional Value-at-Risk) of the portfolio respectively. The "Risk-Adjusted Return" is the measure that we improved to evaluate the worth of each portfolio. The objective we made will be taken as the fitness function for dynamic portfolio selection and will be dealt with the following method.

3.1 Genetic Algorithms Genetic Algorithms (GA) are search algorithms, inspired by the biological evolution that mimics the operation in natural genetics, to search for the optimal solution in a search space which is defined as a region containing infinite number of points where each point represents a feasible solution, which is marked by its fitness value. One mainly method are designed to efficiently search for attractive solutions to large and complex problems. The search proceeds in survival-of-the-fitness fashion by gradually manipulating a population of potential problem solutions until the superior ones dominate the population. That is done by selecting solution with a higher fitness value in every iteration. This is motivated by a hope that the new population will be better than the old one.

The steps in genetic algorithm are detailed below:

- Initialize a population
- Evaluate each chromosome
- Apply elitist selection; carry on the best individuals to the next generation
- Selection chromosomes for reproduction
- Apply crossover operator for reproduction at crossover probability rate P_c
- Apply mutation operator for reproduction at mutation probability rate P_m
- Apply inversion operator for reproduction at inversion probability rate P_i
- Evaluate the new chromosomes
- If the termination condition is satisfied, retain the best solution; if not, carry the elite to the next generation.

Holland's Algorithm has six steps naming–Creation, Evaluation, Selection, Crossover and Mutation and Inversion. Each chromosomes fitness value is evaluated at this step by using an objective function. This mathematical function maps a particular solution to a single number that is the measure of the solution's worth.

This evaluation process is repeated in each iteration (generation) whereby each individual string in the current population is evaluated using this measure of fitness. The probability of an element being selected to remain in the new population is directly proportional to the fitness of the string of the structure. New off-springs are generated from the current population (parents) to pass them on to the next generation population. This process is called Selection. There is a constant improvement in the fitness over the generations. Evolution provides a powerful and effective recipe for solving problems and creating strategies in an unpredictable environment. Fitness landscapes demonstrate how evolutionary search creates robustness and adaptability through constant experimentation, parallel search , and mix of adaptive walks and long jumps.

Crossover operator is used to generate two new strings (off springs) for the next generation by combining the cross-selections of the two individual strings (parents) of the current generation. The operator generates a cut-point depending on the fitness of the strings. Based on the position of the cut-point, the part of the string occurring before the cut-point in the first parent and that part occurring after the cut-point in the second parent are combined to form the new offspring. Hence, it also serves as a sieve to eliminate low fitness structures. In GAs, the role of mutation operator is to emulate the behavior of nature by introducing diversity into the population. Mutation makes random changes to the chromosomes. The number of mutations in a generation is controlled by a Mutation Rate Parameter, which is defined as the ratio of new individuals produced in the generation, by mutation, to the total size of the population. Crossover of the solution generally improves the solution but gets stuck at the local optima. But to reach a global optimum value, mutations have to be incorporated.

3.2 Sampling Approach Let

$$\boldsymbol{d}^1 = (d_1^1, \cdots, d_n^1)^\top, \cdots, \boldsymbol{d}^m = (d_1^m, \cdots, d_n^m)^\top \in \mathbb{R}^n \tag{16}$$

be a sample of data with size $m \in \mathbb{Z}_{++}$, which are drawn from the population of random vector

$$\widetilde{\boldsymbol{r}} = (\widetilde{r}_1, \cdots, \widetilde{r}_n)^\top. \tag{17}$$

Then, natural unbiased estimator of

$$\overline{\boldsymbol{r}} = (\overline{r}_1, \cdots, \overline{r}_n)^\top \in \mathbb{R}^n \tag{18}$$

is given by

$$\widehat{\overline{\boldsymbol{r}}} := \frac{1}{m} \sum_{j=1}^{m} \boldsymbol{d}^j, \tag{19}$$

or,

$$\widehat{\overline{r}}_i := \frac{1}{m} \sum_{j=1}^{m} d_i^j, \quad i = 1, \cdots, n. \tag{20}$$

Furthermore, a natural unbiased estimator of mean

$$\overline{r}(\boldsymbol{x}) = \overline{\boldsymbol{r}}^\top \boldsymbol{x}, \quad \boldsymbol{x} \in X \tag{21}$$

of random return rate $\widetilde{r}(\boldsymbol{x}) = \widetilde{\boldsymbol{r}}^\top \boldsymbol{x}$ of portfolio $\boldsymbol{x} \in X$ is given by

$$\widehat{\overline{r}(\boldsymbol{x})} := \left\{ \frac{1}{m} \sum_{j=1}^{m} \boldsymbol{d}^j \right\}^\top \boldsymbol{x} = \frac{1}{m} \sum_{j=1}^{m} {\boldsymbol{d}^j}^\top \boldsymbol{x}, \quad \boldsymbol{x} \in X. \tag{22}$$

On the other hand, in order to estimate

$$\mathrm{CVaR}_\beta(\boldsymbol{x}) = \mathrm{CVaR}_\beta[\widetilde{r}(\boldsymbol{x})] = \min\{F_\beta(a;\widetilde{r}(\boldsymbol{x})) : a \in \mathbb{R}\}, \quad \boldsymbol{x} \in X, \tag{23}$$

we use the final representation to obtain

$$\widehat{\mathrm{CVaR}_\beta}(\boldsymbol{x}) := \min\left\{F_\beta\widehat{(a;\widetilde{r}(\boldsymbol{x}))} : a \in \mathbb{R}\right\}, \quad \boldsymbol{x} \in X, \tag{24}$$

where

$$F_\beta\widehat{(a;\widetilde{r}(\boldsymbol{x}))} := a + \frac{1}{1-\beta}\left[\frac{1}{m}\sum_{i=1}^{m}\left(-\boldsymbol{d}^{j\top}\boldsymbol{x} - a\right)^+\right] \quad a \in \mathbb{R};\ \boldsymbol{x} \in X. \tag{25}$$

Accordingly, the objective function of $\mathrm{Q}(z)$ to be maximized, is now estimated as

$$\begin{aligned}
(\overline{r}(\boldsymbol{x}) - \widehat{r_f)\,/F_\beta}(a;\widetilde{r}(\boldsymbol{x}))) &= \left(\widehat{\overline{r}(\boldsymbol{x})} - r_f\right)/F_\beta\widehat{(a;\widetilde{r}(\boldsymbol{x}))}, \\
&= \left(\frac{1}{m}\sum_{j=1}^{m}\boldsymbol{d}^{j\top}\boldsymbol{x} - r_f\right)/\min\left(a + \frac{1}{1-\beta}\left[\frac{1}{m}\sum_{i=1}^{m}\left(-\boldsymbol{d}^{j\top}\boldsymbol{x} - a\right)^+\right]\right) \\
&= \left(\frac{1}{m}\sum_{j=1}^{m}\boldsymbol{d}^{j\top}\boldsymbol{x} - r_f\right)/\min\left(a + \frac{1}{(1-\beta)m}\sum_{i=1}^{m}u_j\right), \\
&\quad a \in \mathbb{R};\ \boldsymbol{x} \in X,
\end{aligned} \tag{26}$$

where we introduced auxiliary variables $\boldsymbol{u} = (u_1, \cdots, u_m)^\top \in \mathbb{R}^m$:

$$u_j := \left(-\boldsymbol{d}^{j\top}\boldsymbol{x} - a\right)^+, \quad j = 1, \cdots, m. \tag{27}$$

Collecting the above results, through a sampling method, we can approximate the problem $\mathrm{Q}(z)$ by the following max-min problem with $n+m+1$ decision variables $a \in \mathbb{R}$; $\boldsymbol{x} = (x_1, \cdots, x_n)^\top \in \mathbb{R}^n$; $\boldsymbol{u} = (u_1, \cdots, u_m)^\top \in \mathbb{R}^m$:

$$\widehat{[\mathrm{Q}(z)]}\ \left|\ \begin{array}{ll}
\text{Maximize} & \left(\frac{1}{m}\sum_{j=1}^{m}\boldsymbol{d}^{j\top}\boldsymbol{x} - r_f\right)/\min\left(a + \frac{1}{(1-\beta)m}\sum_{i=1}^{m}u_j\right) \\
\text{subject to} & a \in \mathbb{R}; \\
& \boldsymbol{x} \in X; \\
& u_j \ge -\boldsymbol{d}^{j\top}\boldsymbol{x} - a;\ u_j \ge 0, \quad j = 1, \cdots, m.
\end{array}\right. \tag{28}$$

The above minimization problem will be solved by Linear Programming(LP), but the whole max-min problem will be solve by the joint computational method.

3.3 the Algorithm of the Joint Computational Method Simulation Algorithm is showed at following:

Algorithm 1 (Joint computational method)

Step 0: (Initialization)

generate an initial $x_i, i = 1, \ldots, 28$ by GA and calculate α based on (28),(29) and (30) by LP solver; Generate initial group at random: $X(0) := (x_1, x_2, ..., x_n)$. The initial group is called population, and the population consist of the number of individual, which is called string. The number of string is population size. The different objective functions are, the size of population based on objective function are different. The population size of our simulation under the advanced Sharpe Ratio is 200, which means there are 200 strings. We use the 200 strings to express $x_i, x = 1, ..., 28$.

step 1:

get a group of $x_i, i = 1, \ldots, 28$ and α;

step 2:

calculate the initial ratio of the excess average return to CVaR;

step 3:

save the solution of the initial ratio;

step 4:

repeat from step.0 to step.3;

step 5.

get the optimal $x_i, i = 1, \ldots, 28$ and find out the optimal α at the same time;

step 6.

output the optimal solutions of $x_i, i = 1, \ldots, 28.$ and α.

4 Substantial Analysis In this section, we try to apply the new performance measure, CVaR–based Sharpe ratio, to a (virtual) risky investment on stock indexes constructed of stocks traded at the Tokyo Stock Exchange. In the empirical analysis, we use joint computational methods (Genetic Algorithm and Linear Programming)approach to derive an optimal portfolio of maximizing the CVaR–based Sharpe ratio. The optimal weights x_i^*, $i = 1, \cdots, n$, and the optimal number K of risky securities included in the optimal investment are found out. At the same time, we can show that the CVaR–based Sharpe ration, i.e., the ratio of mean excess return to CVaR is useful for improving and evaluating the performance of portfolios. The empirical analysis can also prove how effectively Genetic algorithm(GA) works.

We construct an optimal portfolio consisting of risky assets, which maximizes the CVaR–based Sharpe ratio, by using the NIKKEI indexes, and prove the usefulness of the new measure. The monthly stock return data of various types of industries from January 1995 to December 2003 are used. We look these 28 types of industry indexes as 28 kinds of risky securities. At first, we give 9–year average return rates in Table 1 calculated by 9–year monthly returns' data. We also define $\hat{\hat{r}}_i$ in order, $i = 1, \cdots, 28$, for example, the mean excess return of building industry is r_3, and the weight of the building industry in the portfolio investment is x_3. The joint computational calculations are carried out based on Algorithm by using LP solver of MATLAB 6.5 and Visual C computer language on a 2.60 GHz Pentium 4 machine.

The "aff" in Table 1 means the industry of agriculture, forestry and fisheries. The Flow Chart of he Genetic algorithm model is given in Figure 1. The steps in the flowchart are explianed below:

- Initialize the population-The program randomly generates initialize the population according with the objective function.
- Evaluate each individual or chromosome-each individual is evaluated for fitness based on the objective function and restriction.
- Apply elitist selection-Based on the fitness, the good genes are selected for crossover, mutation and inversion.
- Two chromosomes having highest fitness are selected for reproduction; crossover mutation and inversion are performed on the selection process.
- The newly generated chromosomes through crossover mutation and inversion are agian evaluated for fitness as in step 2 and the process is repeated till termination criteria-number of generations is satisfied or no further improvement is achieved.
- The program displays the best solutions in order of fitness finally achieved.

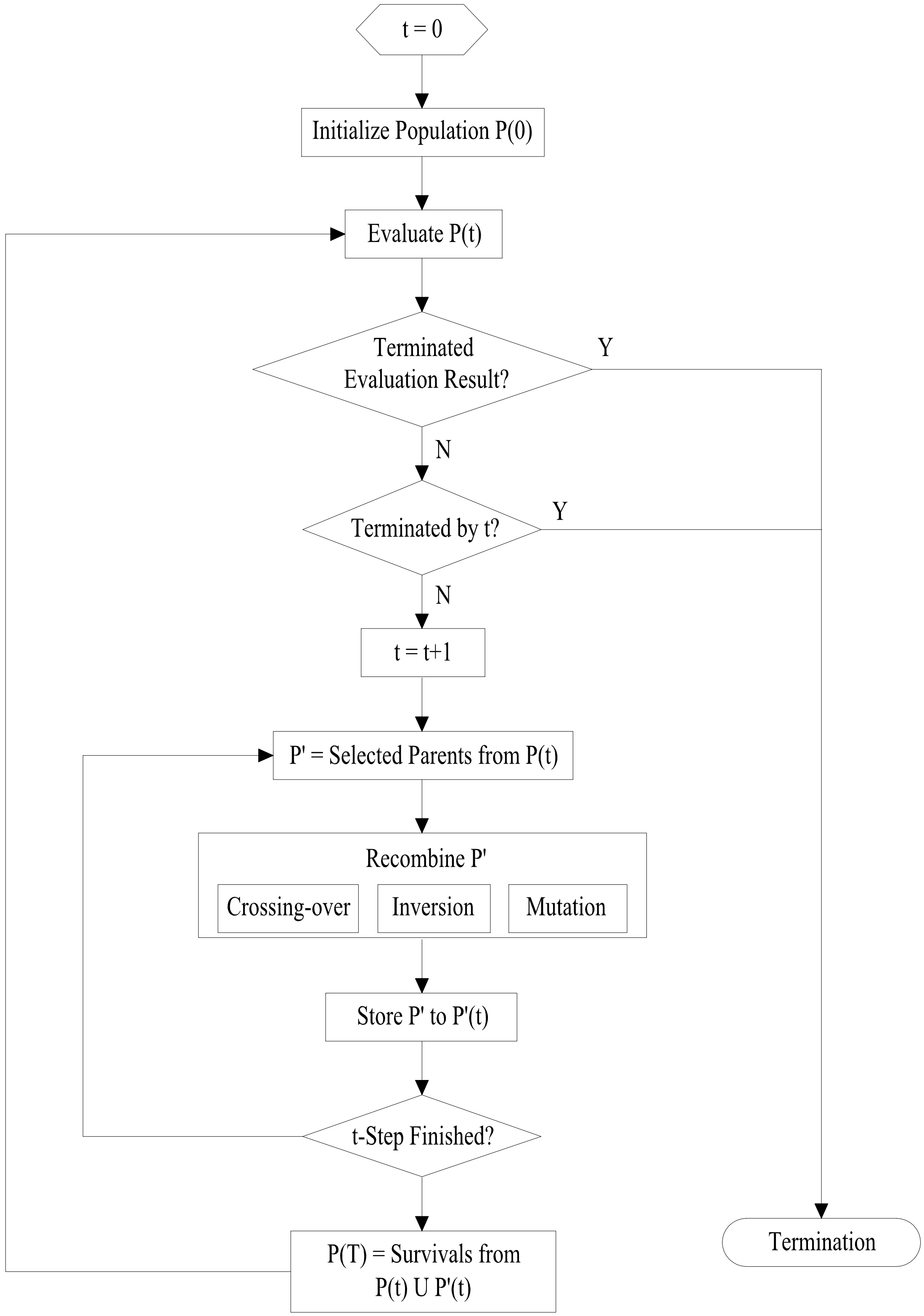

Figure 1: Flow chart of GA.

Table 1: Expected Rate of Return on Each Industry's from Jan. 1995 to Dec. 2003

type of industry		type of industry	r_i
aff (r_1)	−0.756	mining (r_2)	−0.519
building (r_3)	−0.537	grocery (r_4)	−0.14
fiber manufacture (r_5)	−0.27	valve.paper (r_6)	−0.19
medicament (r_7)	0.288	oil.coal (r_8)	−0.07
rubble (r_9)	0.22	glass.soil.stone (r_{10})	−0.08
steel (r_{11})	−0.20	nonferrous metal (r_{12})	−0.16
hardware (r_{13})	−0.21	machinery (r_{14})	−0.11
electric manufacture (r_{15})	0.35	transport application (r_{16})	0.51
precision instrument (r_{17})	0.675	others instrument (r_{18})	0.06
commerce (r_{19})	−0.05	finance.insurance (r_{20})	−0.56
real estate (r_{21})	0.17	transport (r_{22})	−0.06
shipping (r_{23})	0.38	airlift (r_{24})	−1.21
warehouse (r_{25})	−0.24	IT (r_{26})	0.26
electricity.gas (r_{27})	0.08	service (r_{28})	−0.21

Then, for the $\beta-$ values 0.99, 0.95, 0.90, we calculate the $\beta-$VaR and $\beta-$CVaR of the optimal portfolio x^* [8]. The outcomes will be showed from Table 2 to Table 4. At the same time we obtain the optimal number of securities in risky investment and the time of the outcome in different times of circulating genetic process with different β value at Table 2, Table 3 and Table 4[9].

The probability of Inversion is 90/200; the probability of Crossover is 100/200; and the probability of Mutation is 10/200. All of these dates should be set before run the GA programming and all these settings influence the times of rotating the GA process and the speed of convergence. Here, we give all the settings by experience and test. The flow chart of programming will be showed at figure 1.

At all tables, t means the times of the iterations process of evaluate P(T) .

From the table 2, we can find that the times of the iterations process of evaluate P(T) is increasing, the outcomes of $\beta-$VaR, $\beta-$CVaR, the maximum values become to converge. One basic implication of modern portfolio theory is that investors hold well-diversified portfolios. However, there is empirical evidence that individual investors typically hold only a small number of securities. Szego(1980) who emphasizes the point that the returns and risk of a large size portfolio tends to conceal significant singularities or near-singularities, so that enlarging the portfolio beyond the limited diversification size may be superfluous. Our outcome just prove that.

From table 2 to table 4, we can find that the more the times of iterations are, the more convergent outcome will be, not only the maximum value, but also the optimal value of x, and if the times of iterations are increased to 350 times or more, the maximum value and the optimal value of x are stable. With different β-values, the outcomes about x^*, the maximum value, $\beta-$CVaR and $\beta-$VaR are different. As the definitions ensure that the $\beta-$VaR is never larger than the $\beta-$CVaR, we get the result that portfolios with low CVaR must have low VaR as well.

No direct relationship but indirect relationships between the times of iterations and the initial setting including the possibility of the process of mutation, crossover and inversion can be got. That is, with higher the possibility we set and smaller the iterations is. It will cause the outcome convergent faster, but our calculation time turns to be longer. Consequently, it is possible that errors happen before we get the outcome.

When the possibility of crossover P_c is too small, it is difficult to satisfy forward searching, when P_c is too large, it is easy to destroy the structure of the fitness solutions. Generally, the possibility of crossover

[8] x^* means the optimal solution by maximizing the ratio of excess return to CVaR, it is a vector

[9] The three calculations are followed the optimal possibility setting, that is, the probability of Inversion is 90/200; the probability of Crossover is 100/200; and the probability of Mutation is 10/200.

Table 2: the Maximization of R-CVaR, x^*, VaR_β and $CVaR_\beta$ Calculated by Genetic Algorithm(β=0.99, R_f=0.0005, P_c=100/200, P_m=10/200, P_i=90/200.)

x^*	t=50	t=100	t=150	t=200	t=250	t=300	t=350	t=400	t=450	t=500
x_1	0.010230	0.003175	0.005690	0	0	0	0	0	0	0
x_2	0.017903	0.001587	0.001422	0.002710	0	0	0	0	0	0
x_3	0.003836	0	0	0.002710	0.002865	0	0	0	0	0
x_4	0.016624	0.003175	0.004267	0.005420	0.005731	0	0	0	0	0
x_5	0.029412	0.006349	0	0.002710	0	0	0	0	0	0
x_6	0.014066	0.003175	0.002845	0	0	0	0	0	0	0
x_7	0.084399	0.174603	0.160740	0.056911	0.028653	0.007407	0	0	0	0
x_8	0.005115	0	0.005690	0.002710	0	0	0	0	0	0
x_9	0.035806	0.034921	0.018492	0.008130	0.020057	0.007407	0	0	0	0
x_{10}	0.003836	0.003175	0.005690	0	0.002865	0	0	0	0	0
x_{11}	0	0	0.002845	0.002710	0	0	0	0	0	0
x_{12}	0.001279	0.001587	0.004267	0.002710	0.002865	0.007407	0	0	0	0
x_{13}	0	0	0.002845	0.005420	0.002865	0	0	0	0	0
x_{14}	0.023018	0	0.004267	0.002710	0.005731	0	0	0	0	0
x_{15}	0.130435	0.174603	0.180654	0.062331	0.037249	0.007407	0	0	0	0
x_{16}	0.147059	0.174603	0.174964	0.338753	0.361032	0.007407	0	0	0	0
x_{17}	0.159847	0.187302	0.176387	0.341463	0.363897	0.940741	0.8768	0.8768	0.8768	0.8768
x_{18}	0.005115	0	0.007112	0	0	0	0	0	0	0
x_{19}	0.007673	0.020635	0.021337	0.013550	0	0.007407	0	0	0	0
x_{20}	0.010230	0.004762	0.001422	0.002710	0	0	0	0	0	0
x_{21}	0.007673	0.009524	0.002845	0.002710	0.002865	0	0.1121	0.1121	0.1121	0.1121
x_{22}	0	0.011111	0.007112	0	0	0	0	0	0	0
x_{23}	0.001279	0.001587	0.021337	0	0.002865	0	0.0111	0.0111	0.0111	0.0111
x_{24}	0.001279	0.006349	0	0	0		0	0	0	0
x_{25}	0.028133	0.014286	0.004267	0.005420	0.002865	0	0	0	0	0
x_{26}	0.162404	0.161905	0.180654	0.127371	0.151862	0	0	0	0	0
x_{27}	0.020460	0	0.001422	0.005420	0.002865	0	0	0	0	0
x_{28}	0.072890	0.004762	0.001422	0.005420	0.002865	0	0	0	0	0
max	0.0098	0.01121	0.01435	0.01525	0.01764	0.02121	0.02475	0.02475	0.02475	0.02475
$CVaR_\beta$	13.2389	15.7681	17.2172	19.5232	21.2827	22.2691	24.8458	24.8458	24.8458	24.8458
VaR_β	5.1061	6.2712	6.8976	7.9812	8.7652	9.6523	10.8443	10.8443	10.8443	10.8443
Time(s)	3	5	7	10	12	14	16	19	21	23
k	25	21	25	21	17	7	3	3	3	3

Table 3: the Maximization of R-CVaR, x^*, VaR_β and $CVaR_\beta$ Calculated by Genetic Algorithm(β=0.95, R_f=0.0005, P_c=100/200, P_m=10/200, P_i=90/200.)

x^*	t=50	t=100	t=150	t=200	t=250	t=300	t=350	t=400	t=450	t=500
x_1	0.026226	0.003686	0.003984	0	0	0	0	0	0	0
x_2	0.002281	0.002457	0.003984	0.002475	0	0	0	0	0	0
x_3	0.004561	0.002457	0	0	0	0.1348	0	0	0	0
x_4	0.023945	0.007371	0.003984	0	0.006623	0	0	0	0	0
x_5	0.027366	0.020885	0.001992	0.004950	0	0	0	0	0	0
x_6	0.010262	0	0.001992	0.002475	0	0	0	0	0	0
x_7	0.117446	0.141278	0.025896	0.056931	0.013245	0.1197	0	0	0	0
x_8	0.020525	0.002457	0.011952	0	0	0	0	0	0	0
x_9	0.002281	0.055283	0.039841	0.037129	0.013245	0.0525	0	0	0	0
x_{10}	0.013683	0.001229	0.005976	0.002475	0	0	0	0	0	0
x_{11}	0.004561	0.013514	0.001992	0.002475	0	0	0	0	0	0
x_{12}	0.023945	0.045455	0.019920	0.012376	0.006623	0	0	0	0	0
x_{13}	0.014823	0.019656	0.005976	0	0	0	0	0	0	0
x_{14}	0.033067	0.004914	0.001992	0.007426	0	0	0	0	0	0
x_{15}	0.043330	0.114251	0.093626	0.004950	0.039735	0.007634	0	0	0	0
x_{16}	0.136830	0.142506	0.241036	0.257426	0.019868	0.007634	0.0846	0.0846	0.0846	0.0846
x_{17}	0.144812	0.156020	0.249004	0.311881	0.827815	0.1351	0.8606	0.8606	0.8606	0.8606
x_{18}	0.012543	0.030713	0.007968	0	0.006623	0	0	0	0	0
x_{19}	0.126568	0.045455	0.015936	0.002475	0	0	0	0	0	0
x_{20}	0.007982	0.003686	0.003984	0	0	0	0	0	0	0
x_{21}	0.009122	0	0.005976	0	0	0	0.0207	0.0207	0.0207	0.0207
x_{22}	0.017104	0.018428	0.009960	0	0	0	0	0	0	0
x_{23}	0.003421	0.008600	0.001992	0.002475	0.006623	0	0.0341	0.0341	0.0341	0.0341
x_{24}	0.010262	0.001229	0.001992	0	0	0	0	0	0	0
x_{25}	0.017104	0	0	0.002475	0.013245	0	0	0	0	0
x_{26}	0.143672	0.141278	0.223108	0.274752	0.046358	0.0022901	0	0	0	0
x_{27}	0.002281	0.014742	0.013944	0.012376	0	0.4598	0	0	0	0
x_{28}	0	0.002457	0.001992	0.002475	0	0.0981	0	0	0	0
max	0.015	0.018	0.021	0.027	0.031	0.036	0.04	0.04	0.04	0.04
$CVaR_\beta$	10.5676	11.2978	12.6762	13.1121	13.6591	14.0231	14.5398	14.5398	14.5398	14.5398
VaR_β	5.3527	6.0121	6.5476	7.1082	7.6543	8.1312	8.9232	8.9232	8.9232	8.9232
Time(s)	3	5	6	8	10	13	16	19	22	25
k	27	25	26	18	11	9	4	4	4	4

Table 4: the Maximization of R-CVaR, x^*, VaR_β and $CVaR_\beta$ Calculated by Genetic Algorithm(β=0.90, R_f=0.0005, P_c=100/200, P_m=10/200, P_i=90/200.)

x^*	t=50	t=100	t=150	t=200	t=250	t=300	t=350	t=400	t=450	t=500
x_1	0.004655	0.001403	0	0	0.003759	0	0	0	0	0
x_2	0.013966	0.007013	0.006231	0	0	0	0	0	0	0
x_3	0.001862	0.007013	0.004673	0.002849	0	0	0	0	0	0
x_4	0.005587	0.001403	0.001558	0	0.0011278	0	0	0	0	0
x_5	0.015829	0.009818	0.003115	0	0.003759	0	0	0	0	0
x_6	0.009311	0.007013	0.003115	0.002849	0.003759	0	0	0	0	0
x_7	0.107076	0.126227	0.137072	0.042735	0.026316	0.012048	0	0	0	0
x_8	0.002793	0.002805	0	0	0.007519	0	0	0	0	0
x_9	0.115456	0.071529	0.028037	0.005698	0.037594	0.018072	0	0	0	0
x_{10}	0.004655	0.007013	0.007788	0.008547	0	0	0	0	0	0
x_{11}	0.013966	0.005610	0.003115	0	0	0	0	0	0	0
x_{12}	0.0034451	0.005610	0.010903	0.014245	0.018797	0	0	0	0	0
x_{13}	0.014898	0.004208	0.003115	0	0.003759	0	0	0	0	0
x_{14}	0.003724	0.028050	0.004673	0	0	0.012048	0	0	0	0
x_{15}	0.086592	0.130435	0.190031	0.042735	0.041353	0.090361	0	0	0	0
x_{16}	0.108007	0.157083	0.193146	0.270655	0.172932	0	0.1556	0.1556	0.1556	0.1556
x_{17}	0.118250	0.164095	0.188474	0.353276	0.481203	0.759036	0.7905	0.7905	0.7905	0.7905
x_{18}	0.023277	0.009818	0.001558	0.011396	0	0	0	0	0	0
x_{19}	0.094041	0.037868	0.017134	0	0	0	0	0	0	0
x_{20}	0.010242	0.005610	0.001558	0	0	0	0	0	0	0
x_{21}	0.014898	0.004208	0	0	0	0	0.0097	0.0097	0.0097	0.0097
x_{22}	0.013966	0.019635	0.007788	0.002849	0	0	0	0	0	0
x_{23}	0.006518	0.037868	0.007788	0.002849	0	0	0.0441	0.0441	0.0441	0.0441
x_{24}	0.000931	0	0	0	0	0	0	0	0	0
x_{25}	0	0.001403	0.001558	0.005698	0.003759	0	0	0	0	0
x_{26}	0.092179	0.145863	0.172897	0.225071	0.176692	0.030120	0	0	0	0
x_{27}	0.028864	0.001403	0	0.002849	0	0	0	0	0	0
x_{28}	0.054004	0	0.004673	0.005698	0.007518	0	0	0	0	0
max	0.027	0.031	0.036	0.041	0.045	0.05	0.055	0.055	0.055	0.055
$CVaR_\beta$	8.5438	9.2871	9.7612	10.0212	10.5601	11.0652	11.4025	11.4025	11.4025	11.4025
VaR_β	5.2451	5.7871	6.2791	6.8721	7.3217	7.8212	8.0086	8.0086	8.0086	8.0086
Time(s)	3	5	8	10	13	16	18	20	22	25
k	27	26	23	16	15	6	4	4	4	4

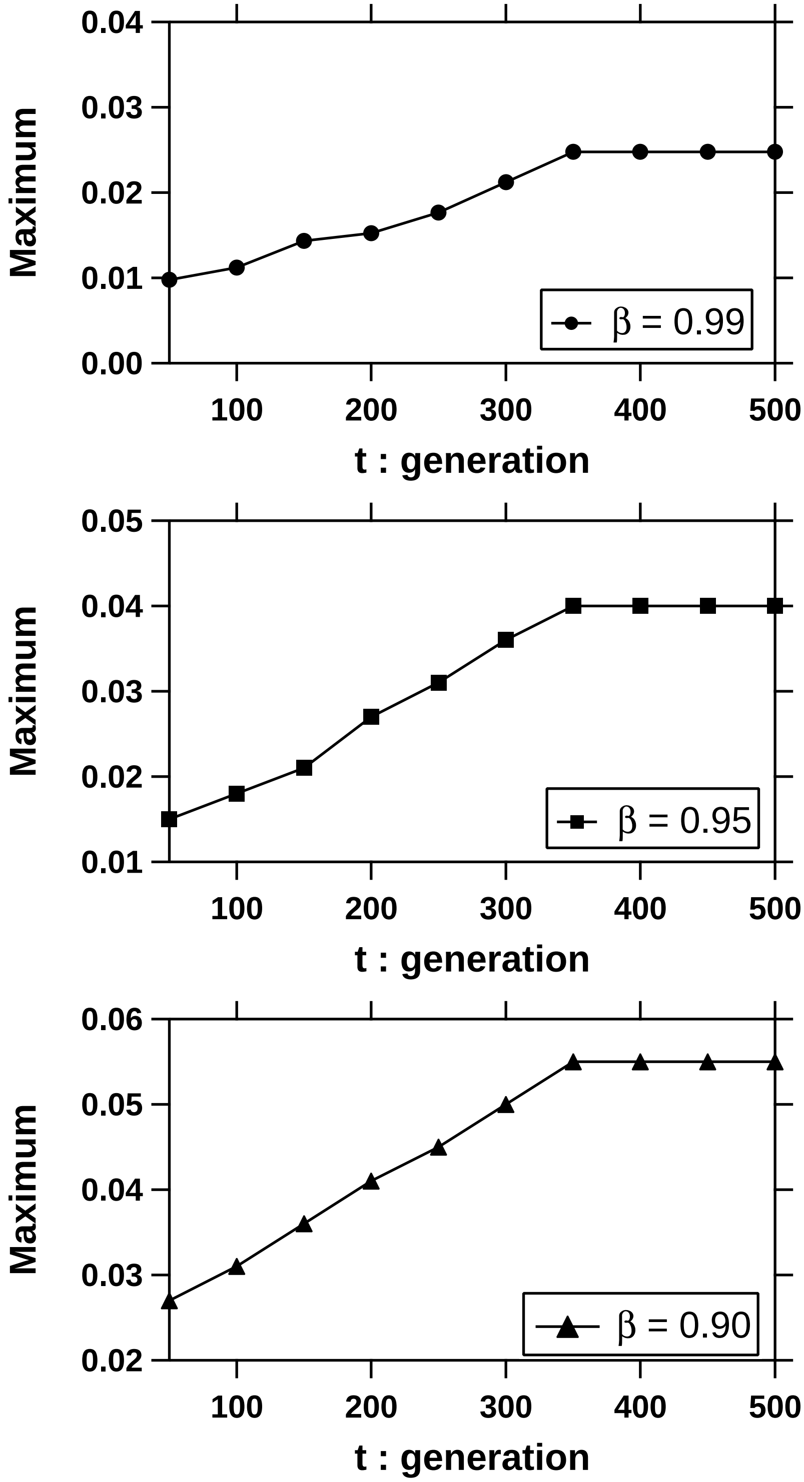

Figure 2: the Maximization under Different β

P_c is from $\frac{50}{200}$ to $\frac{150}{200}$.[10] The outcomes under the different P_c but the same P_i and P_m are showed at table 5 and table 6.

Table 5: the Maximization of R-CVaR, x^*, VaR_β and $CVaR_\beta$ Calculated by Genetic Algorithm(β=0.99, R_f=0.0005, P_c=120/200, P_m=10/200, P_i=90/200.)

x^*	t=50	t=100	t=150	t=200	t=250	t=300	t=350	t=400	t=450	t=500
x_1	0.010230	0.003175	0.005690	0	0	0	0	0	0	0
x_2	0.017903	0.001587	0.001422	0.002710	0	0	0	0	0	0
x_3	0.003836	0	0	0.002710	0.002865	0	0	0	0	0
x_4	0.016624	0.003175	0.004267	0.005420	0.005731	0	0	0	0	0
x_5	0.029412	0.006349	0	0.002710	0	0	0	0	0	0
x_6	0.014066	0.003175	0.002845	0	0	0	0	0	0	0
x_7	0.084399	0.174603	0.160740	0.056911	0.028653	0.007407	0	0	0	0
x_8	0.005115	0	0.005690	0.002710	0	0	0	0	0	0
x_9	0.035806	0.034921	0.018492	0.008130	0.020057	0.007407	0	0	0	0
x_{10}	0.003836	0.003175	0.005690	0	0.002865	0	0	0	0	0
x_{11}	0	0	0.002845	0.002710	0	0	0	0	0	0
x_{12}	0.001279	0.001587	0.004267	0.002710	0.002865	0.007407	0	0	0	0
x_{13}	0	0	0.002845	0.005420	0.002865	0	0	0	0	0
x_{14}	0.023018	0	0.004267	0.002710	0.005731	0	0	0	0	0
x_{15}	0.130435	0.174603	0.180654	0.062331	0.037249	0.007407	0	0	0	0
x_{16}	0.147059	0.174603	0.174964	0.338753	0.361032	0.007407	0	0	0	0
x_{17}	0.159847	0.187302	0.176387	0.341463	0.363897	0.940741	0.8768	0.8768	0.8768	0.8768
x_{18}	0.005115	0	0.007112	0	0	0	0	0	0	0
x_{19}	0.007673	0.020635	0.021337	0.013550	0	0.007407	0	0	0	0
x_{20}	0.010230	0.004762	0.001422	0.002710	0	0	0	0	0	0
x_{21}	0.007673	0.009524	0.002845	0.002710	0.002865	0	0.1121	0.1121	0.1121	0.1121
x_{22}	0	0.011111	0.007112	0	0	0	0	0	0	0
x_{23}	0.001279	0.001587	0.021337	0	0.002865	0	0.0111	0.0111	0.0111	0.0111
x_{24}	0.001279	0.006349	0	0	0		0	0	0	0
x_{25}	0.028133	0.014286	0.004267	0.005420	0.002865	0	0	0	0	0
x_{26}	0.162404	0.161905	0.180654	0.127371	0.151862	0	0	0	0	0
x_{27}	0.020460	0	0.001422	0.005420	0.002865	0	0	0	0	0
x_{28}	0.072890	0.004762	0.001422	0.005420	0.002865	0	0	0	0	0
max	0.0098	0.01121	0.01435	0.01525	0.01764	0.02121	0.02475	0.02475	0.02475	0.02475
$CVaR_\beta$	13.2389	15.7681	17.2172	19.5232	21.2827	22.2691	24.8458	24.8458	24.8458	24.8458
VaR_β	5.1061	6.2712	6.8976	7.9812	8.7652	9.6523	10.8443	10.8443	10.8443	10.8443
Time(s)	7	9	12	15	18	20	23	26	28	30
k	25	21	25	21	17	7	3	3	3	3

From table 5 and table 6, we can find that the outcomes under higher P_c are almost changeless comparing to those under $P_c = \frac{100}{200}$. But with the higher P_c, the time of iteration is longer than before. The initial setting about $P_c = \frac{100}{200}$ is optimal on this problem. Some crossover searching process become to be superfluous. Decreasing the setting of P_c will cause that the outcomes we got are not optimal solutions in the global region. the outcomes are regional solutions.

Mutation does not do any help to the solution, but it can assume that evolution of populations are go on. Because if all individuals are identical, new individual will not be generated by crossover and inversion, and just by mutation, in other words, mutation realize the global optimum. Therefore, when the possibility of mutation P_m is too small, it is difficult to form the structure consisting of new generations. When P_m is too large, Genetic Algorithm turns to simple searching at random. When P_m is from $\frac{2}{200}$ to $\frac{40}{200}$ generally, we can get the outcomes by GA. The outcomes under the different P_m but the same P_i and P_c are showed at table 7 and table 8.

At table 7 and table 8, we can see that the outcomes will not be satisfied if we decease P_m, it just make the time of searching longer. Because the process of searching can not find optimal solutions without

[10] The scale of P_c is fitting for the general case. But it will change under different objective function.

Table 6: the Maximization of R-CVaR, x^*, VaR_β and $CVaR_\beta$ Calculated by Genetic Algorithm(β=0.99, R_f=0.0005, P_c=70/200, P_m=10/200, P_i=90/200.)

x^*	t=50	t=100	t=150	t=200	t=250	t=300	t=350	t=400	t=450	t=500
x_1	0.014190	0.14563	0.002469	0.005545	0.000867	0	0	0	0	0
x_2	0.007513	0.002427	0.005761	0.015712	0	0	0	0	0	0
x_3	0	0.020227	0.009877	0.008318	0	0	0	0	0	0
x_4	0.046745	0.042071	0.076543	0.015712	0	0	0	0	0	0
x_5	0.018364	0.001618	0.005761	0.009242	0	0	0	0	0	0
x_6	0.004174	0.012136	0.004115	0.000924	0	0	0	0	0	0
x_7	0.090150	0.101133	0.069136	0.085028	0.021664	0.007407	0.007407	0	0	0
x_8	0.004174	0.017799	0.026337	0.001848	0.000867	0	0	0	0	0
x_9	0.012521	0.08770	0.050206	0.049908	0.046794	0.00748	0.007407	0	0	0
x_{10}	0.030885	0.038835	0.027160	0.006470	0.001733	0	0	0	0	0
x_{11}	0.005843	0.011327	0.028807	0	0.009532	0	0	0	0	0
x_{12}	0.065943	0.021036	0.009053	0.083179	0.096187	0.022101	0.007407	0	0	0
x_{13}	0.030885	0.021845	0.004938	0.015712	0	0	0	0	0	0
x_{14}	0.002504	0.000809	0.17284	0.004621	0.005199	0	0	0	0	0
x_{15}	0.086811	0.102751	0.083128	0.103512	0.110052	0.0025406	0.003453	0.0054353	0.006453	0.007453
x_{16}	0.089316	0.097087	0.094650	0.112754	0.105719	0.031521	0.0163815	0.024815	0.022372	0.023481
x_{17}	0.102671	0.101942	0.102881	0.108133	0.933334	0.903375	0.936521	0.944312	0.945506	0.954161
x_{18}	0.035058	0.022654	0.072428	0.051756	0.100520	0	0	0	0	0
x_{19}	0.000835	0.101942	0.097119	0.040665	0.102253	0.0127431	0.036972	0.011272	0.012348	0
x_{20}	0.015860	0.006472	0.022222	0.000924	0.010399	0	0	0	0	0
x_{21}	0.000835	0.002427	0.034568	0.027726	0.006066	0	0	0	0	0
x_{22}	0.035893	0.009709	0.011523	0.053604	0.009532	0	0	0	0	0
x_{23}	0.018364	0.025081	0.022222	0.003697	0.075390	0	0	0	0	0
x_{24}	0.001669	0.004045	0	0.009242	0.005199	0	0	0	0	0
x_{25}	0.013356	0.006472	0	0.000924	0.013865	0	0	0	0	0
x_{26}	0.186979	0.099515	0.041975	0.116451	0.101386	0.001511	0.0126513	0.014166	0.0133212	0.014815
x_{27}	0.101002	0.099515	0.041975	0.116451	0.101386	0.011321	0	0	0	0
x_{28}	0	0.028317	0.079835	0.066543	0.010399	0	0	0	0	0
max	0.0074	0.0093	0.0126	0.0143	0.0161	0.01845	0.0202	0.0202	0.0202	0.0202
$CVaR_\beta$	20.4511	20.8765	20.3219	20.7821	21.0911	21.4512	21.8981	21.8981	21.8981	21.8981
VaR_β	7.8773	8.2071	8.8611	9.1482	9.5491	9.9623	10.3212	10.3212	10.3212	10.3212
Time(s)	4	6	8	10	12	15	17	19	21	24
k	26	28	26	27	22	9	8	5	5	4

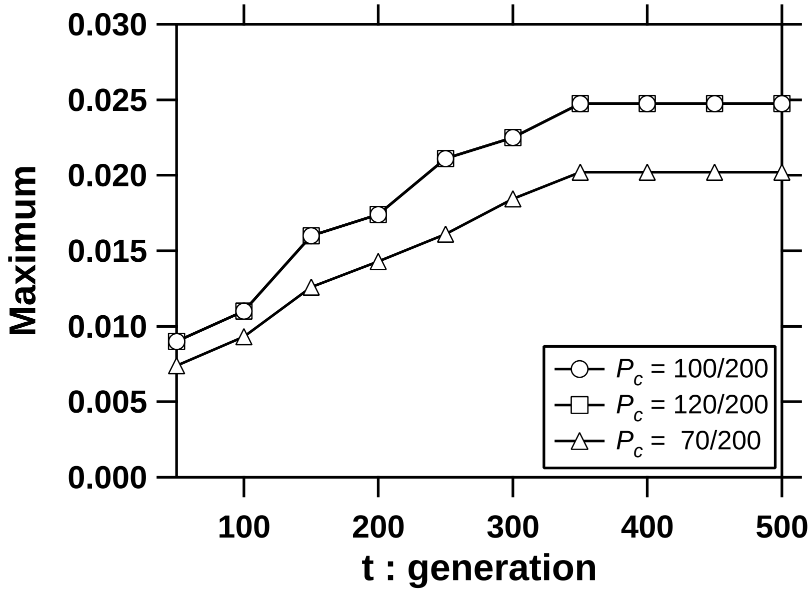

Figure 3: the Maximization under Different P_c

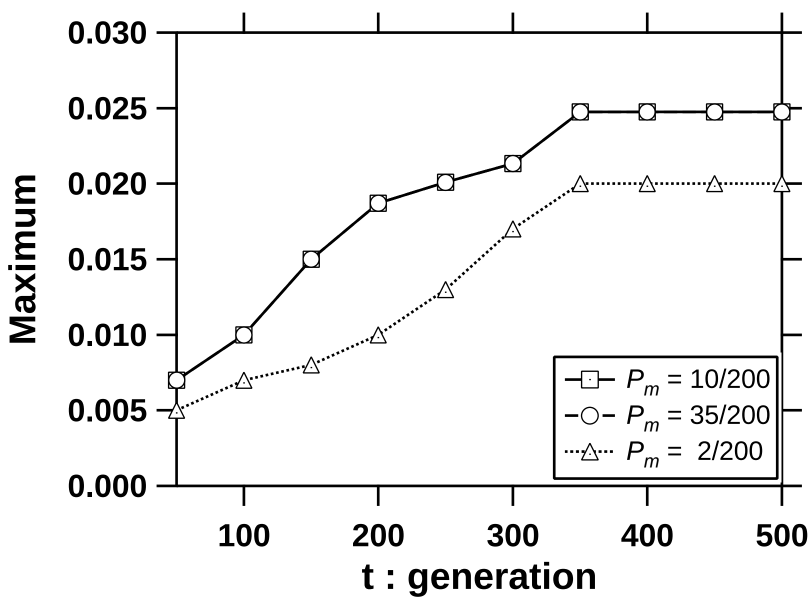

Figure 4: the Maximization under Different P_m

Table 7: the Maximization of R-CVaR, x^*, VaR_β and $CVaR_\beta$ Calculated by Genetic Algorithm (β=0.99, R_f=0.0005, P_m=35/200, P_c=100/200, P_i=10/200.)

x^*	t=50	t=100	t=150	t=200	t=250	t=300	t=350	t=400	t=450	t=500
x_1	0.010230	0.003175	0.005690	0	0	0	0	0	0	0
x_2	0.017903	0.001587	0.001422	0.002710	0	0	0	0	0	0
x_3	0.003836	0	0	0.002710	0.002865	0	0	0	0	0
x_4	0.016624	0.003175	0.004267	0.005420	0.005731	0	0	0	0	0
x_5	0.029412	0.006349	0	0.002710	0	0	0	0	0	0
x_6	0.014066	0.003175	0.002845	0	0	0	0	0	0	0
x_7	0.084399	0.174603	0.160740	0.056911	0.028653	0.007407	0	0	0	0
x_8	0.005115	0	0.005690	0.002710	0	0	0	0	0	0
x_9	0.035806	0.034921	0.018492	0.008130	0.020057	0.007407	0	0	0	0
x_{10}	0.003836	0.003175	0.005690	0	0.002865	0	0	0	0	0
x_{11}	0	0	0.002845	0.002710	0	0	0	0	0	0
x_{12}	0.001279	0.001587	0.004267	0.002710	0.002865	0.007407	0	0	0	0
x_{13}	0	0	0.002845	0.005420	0.002865	0	0	0	0	0
x_{14}	0.023018	0	0.004267	0.002710	0.005731	0	0	0	0	0
x_{15}	0.130435	0.174603	0.180654	0.062331	0.037249	0.007407	0	0	0	0
x_{16}	0.147059	0.174603	0.174964	0.338753	0.361032	0.007407	0	0	0	0
x_{17}	0.159847	0.187302	0.176387	0.341463	0.363897	0.940741	0.8768	0.8768	0.8768	0.8768
x_{18}	0.005115	0	0.007112	0	0	0	0	0	0	0
x_{19}	0.007673	0.020635	0.021337	0.013550	0	0.007407	0	0	0	0
x_{20}	0.010230	0.004762	0.001422	0.002710	0	0	0	0	0	0
x_{21}	0.007673	0.009524	0.002845	0.002710	0.002865	0	0.1121	0.1121	0.1121	0.1121
x_{22}	0	0.011111	0.007112	0	0	0	0	0	0	0
x_{23}	0.001279	0.001587	0.021337	0	0.002865	0	0.0111	0.0111	0.0111	0.0111
x_{24}	0.001279	0.006349	0	0	0		0	0	0	0
x_{25}	0.028133	0.014286	0.004267	0.005420	0.002865	0	0	0	0	0
x_{26}	0.162404	0.161905	0.180654	0.127371	0.151862	0	0	0	0	0
x_{27}	0.020460	0	0.001422	0.005420	0.002865	0	0	0	0	0
x_{28}	0.072890	0.004762	0.001422	0.005420	0.002865	0	0	0	0	0
max	0.0098	0.01121	0.01435	0.01525	0.01764	0.02121	0.02475	0.02475	0.02475	0.02475
$CVaR_\beta$	13.2389	15.7681	17.2172	19.5232	21.2827	22.2691	24.8458	24.8458	24.8458	24.8458
VaR_β	5.1061	6.2712	6.8976	7.9812	8.7652	9.6523	10.8443	10.8443	10.8443	10.8443
Time(s)	3	5	7	10	12	14	16	19	21	23
k	25	21	25	21	17	7	3	3	3	3

Table 8: the Maximization of R-CVaR, x^*, VaR_β and $CVaR_\beta$ Calculated by Genetic Algorithm (β=0.99, R_f=0.0005, P_m=2/200, P_c=100/200, P_i=90/200.)

x^*	t=50	t=100	t=150	t=200	t=250	t=300	t=350	t=400	t=450	t=500
x_1	0.014190	0.14563	0.002469	0.005545	0.000867	0	0	0	0	0
x_2	0.007513	0.002427	0.005761	0.015712	0	0	0	0	0	0
x_3	0	0.002427	0	0	0	0	0	0	0	0
x_4	0.046745	0.042071	0.076543	0.015712	0	0	0	0	0	0
x_5	0.018364	0.001618	0.005761	0.009242	0	0	0	0	0	0
x_6	0.004174	0.012136	0.004115	0.000924	0	0	0	0	0	0
x_7	0.090150	0.101133	0.074897	0.085028	0.021664	0.007407	00.007407	0	0	0
x_8	0.004174	0.017799	0.026337	0.001848	0.000867	0	0	0	0	0
x_9	0.012521	0.068770	0.050206	0.049908	0.046794	0.00848	0	0	0	0
x_{10}	0.030885	0.038835	0.027160	0.006470	0.001733	0	0	0	0	0
x_{11}	0.005843	0.011327	0.028807	0	0.009532	0	0	0	0	0
x_{12}	0.065943	0.021036	0.009053	0.083179	0.096187	0	0	0	0	0
x_{13}	0.030885	0.021845	0.004938	0.015712	0	0	0	0	0	0
x_{14}	0.002504	0.000809	0.017284	0.004621	0.005199	0	0	0	0	0
x_{15}	0.086811	0.102751	0.083128	0.103512	0.110052	0.0015406	0.001453	0.0014353	0.001453	0.001453
x_{16}	0.089316	0.097087	0.094650	0.112754	0.105719	0.033521	0.0163815	0.014815	0.013818	0.013818
x_{17}	0.102671	0.101942	0.102881	0.108133	0.933334	0.923375	0.936521	0.954312	0.968917	0.968917
x_{18}	0.035058	0.022654	0.072428	0.051756	0.100520	0	0	0	0	0
x_{19}	0.000835	0.101942	0.097119	0.040665	0.102253	0.0127431	0.036972	0	0	0
x_{20}	0.015860	0.006472	0.022222	0.000924	0.106586	0	0	0	0	0
x_{21}	0.000835	0.002427	0.034568	0.027726	0	0	0	0	0	0
x_{22}	0.035893	0.009709	0.011523	0.053604	0	0	0	0	0	0
x_{23}	0.018364	0.025081	0.022222	0.003697	0	0	0	0	0	0
x_{24}	0.001669	0.004045	0	0.009242	0	0	0	0	0	0
x_{25}	0.013356	0.006472	0	0.000924	0.013865	0	0	0	0	0
x_{26}	0.101002	0.099515	0.041975	0.116451	0.101386	0.012832	0.0126513	0.013415	0.015812	0.015812
x_{27}	0.078464	0.016990	0.079835	0.116451	0.101386	0	0	0	0	0
x_{28}	0.085977	0.028317	0	0.066543	0.010399	0	0	0	0	0
max	0.005	0.007	0.008	0.01	0.013	0.017	0.02	0.02	0.02	0.02
$CVaR_\beta$	18.5691	19.0212	19.5421	19.9791	20.4312	20.8978	21.4322	21.4322	21.4322	21.4322
VaR_β	8.4322	8.4321	9.0112	9.4111	9.6754	10.0073	10.2123	10.2123	10.2123	10.2123
Time(s)	1	3	4	5	6	7	10	12	15	18
k	27	28	24	26	17	7	6	4	4	4

generating good genes. But if we increase P_m, the time of iteration will be longer, but the speed of convergence do not become quickly absolutely. It is because that the high P_m will destroy the good genes and the good gene structure.

Inversion operator on the one hand can assure that the new generation of individuals attain the characteristics of being feasible solution. On the other hand, can improve the search ability on solution space. If the possibility of inversion P_i is too small, it will difficult to generate good generations, when P_i is too large, it will lose some good genes. We set P_i that $P_i = \frac{140}{200}$ and $P_i = \frac{40}{200}$ separately under the same P_m and P_c. The outcomes are showed at table 9 and table 10.

Table 9: the Maximization of R-CVaR, x^*, VaR_β and $CVaR_\beta$ Calculated by Genetic Algorithm(β=0.99, R_f=0.0005, P_i=140/200, P_c=100/200, P_m=10/200.)

x^*	t=50	t=100	t=150	t=200	t=250	t=300	t=350	t=400	t=450	t=500
x_1	0.010230	0.003175	0.005690	0	0	0	0	0	0	0
x_2	0.017903	0.001587	0.001422	0.002710	0	0	0	0	0	0
x_3	0.003836	0	0	0.002710	0.002865	0	0	0	0	0
x_4	0.016624	0.003175	0.004267	0.005420	0.005731	0	0	0	0	0
x_5	0.029412	0.006349	0	0.002710	0	0	0	0	0	0
x_6	0.014066	0.003175	0.002845	0	0	0	0	0	0	0
x_7	0.084399	0.174603	0.160740	0.056911	0.028653	0.007407	0	0	0	0
x_8	0.005115	0	0.005690	0.002710	0	0	0	0	0	0
x_9	0.035806	0.034921	0.018492	0.008130	0.020057	0.007407	0	0	0	0
x_{10}	0.003836	0.003175	0.005690	0	0.002865	0	0	0	0	0
x_{11}	0	0	0.002845	0.002710	0	0	0	0	0	0
x_{12}	0.001279	0.001587	0.004267	0.002710	0.002865	0.007407	0	0	0	0
x_{13}	0	0	0.002845	0.005420	0.002865	0	0	0	0	0
x_{14}	0.023018	0	0.004267	0.002710	0.005731	0	0	0	0	0
x_{15}	0.130435	0.174603	0.180654	0.062331	0.037249	0.007407	0	0	0	0
x_{16}	0.147059	0.174603	0.174964	0.338753	0.361032	0.007407	0	0	0	0
x_{17}	0.159847	0.187302	0.176387	0.341463	0.363897	0.940741	0.8768	0.8768	0.8768	0.8768
x_{18}	0.005115	0	0.007112	0	0	0	0	0	0	0
x_{19}	0.007673	0.020635	0.021337	0.013550	0	0.007407	0	0	0	0
x_{20}	0.010230	0.004762	0.001422	0.002710	0	0	0	0	0	0
x_{21}	0.007673	0.009524	0.002845	0.002710	0.002865	0	0.1121	0.1121	0.1121	0.1121
x_{22}	0	0.011111	0.007112	0	0	0	0	0	0	0
x_{23}	0.001279	0.001587	0.021337	0	0.002865	0	0.0111	0.0111	0.0111	0.0111
x_{24}	0.001279	0.006349	0	0	0		0	0	0	0
x_{25}	0.028133	0.014286	0.004267	0.005420	0.002865	0	0	0	0	0
x_{26}	0.162404	0.161905	0.180654	0.127371	0.151862	0	0	0	0	0
x_{27}	0.020460	0	0.001422	0.005420	0.002865	0	0	0	0	0
x_{28}	0.072890	0.004762	0.001422	0.005420	0.002865	0	0	0	0	0
max	0.0098	0.01121	0.01435	0.01525	0.01764	0.02121	0.02475	0.02475	0.02475	0.02475
$CVaR_\beta$	13.2389	15.7681	17.2172	19.5232	21.2827	22.2691	24.8458	24.8458	24.8458	24.8458
VaR_β	5.1061	6.2712	6.8976	7.9812	8.7652	9.6523	10.8443	10.8443	10.8443	10.8443
Time(s)	7	9	12	15	18	21	24	27	29	31
k	25	21	25	21	17	7	3	3	3	3

From table 9 and table 10, we can find that the time of iteration will be longer if we increase P_I, but the speed of convergence do not become quickly. It is because that the process will spent more time to find the optimal solutions, and the process of searching will lose some good genes by increasing P_I. We can say that it will waste time and the iteration of searching in increasing P_I. Conversely, the time of iteration will be shorter if we decrease P_I, but the speed of convergence will get slowly, it is possible that we can get the optimal solutions if deceasing P_I too much.

We can find the initial setting is the optimal setting of this problem. In fact, the setting of the possibility of crossover, mutation and inversion will change in different objective problems.

Table 10: the Maximization of R-CVaR, x^*, VaR_β and $CVaR_\beta$ Calculated by Genetic Algorithm(β=0.99, R_f=0.0005, P_i=40/200, P_c=100/200, P_m=10/200.)

x^*	t=50	t=100	t=150	t=200	t=250	t=300	t=350	t=400	t=450	t=500
x_1	0.014190	0.14563	0.002469	0.005545	0.000867	0	0	0	0	0
x_2	0.007513	0.002427	0.005761	0.015712	0	0	0	0	0	0
x_3	0	0.020227	0.009877	0.008318	0	0	0	0	0	0
x_4	0.046745	0.042071	0.076543	0.015712	0	0	0	0	0	0
x_5	0.018364	0.001618	0.005761	0.009242	0.011265	0	0	0	0	0
x_6	0.004174	0.012136	0.004115	0.000924	0.015598	0	0	0	0	0
x_7	0.090150	0.101133	0.069136	0.085028	0.021664	0.007407	0.007407	0.007407	0.007407	0.007407
x_8	0.004174	0.017799	0.026337	0.001848	0.000867	0	0	0	0	0
x_9	0.012521	0.08770	0.050206	0.049908	0.046794	0.007407	0	0.00505	0	0
x_{10}	0.030885	0.038835	0.027160	0.006470	0.001733	0	0	0	0	0
x_{11}	0.005843	0.011327	0.03786	0	0.009532	0	0	0	0	0
x_{12}	0.065943	0.021036	0	0.083179	0.096187	0.007407	0	0	0	0
x_{13}	0.030885	0.021845	0.004938	0.015712	0	0	0	0	0	0
x_{14}	0.002504	0.000809	0.017284	0.004621	0	0	0	0	0	0
x_{15}	0.086811	0.102751	0.083128	0.103512	0.131715	0.007407	0	0	0	0
x_{16}	0.089316	0.097087	0.094650	0.112754	0.105719	0.007407	0.014815	0.012457	0.012457	0.012457
x_{17}	0.102671	0.101942	0.175309	0.108133	0.109185	0.933334	0.955555	0.967679	0.967679	0.967679
x_{18}	0.035058	0.022654	0	0.051756	0.100520	0	0	0	0	0
x_{19}	0.000835	0.101942	0.097119	0.040665	0.102253	0.007407	0.007407	0	0	0
x_{20}	0.015860	0.006472	0.022222	0.000924	0.010399	0	0	0	0	0
x_{21}	0.000835	0.002427	0.034568	0.027726	0.110055	0	0	0	0	0
x_{22}	0.035893	0.009709	0.011523	0.053604	0	0	0	0	0	0
x_{23}	0.018364	0.025081	0.022222	0.003697	0	0	0	0	0	0
x_{24}	0.001669	0.004045	0	0.009242	0	0	0	0	0	0
x_{25}	0.013356	0.006472	0	0.000924	0	0	0	0	0	0
x_{26}	0.101002	0.099515	0.041975	0.116451	0.12565	0.007407	0.014815	0.007407	0.012457	0.012457
x_{27}	0.078464	0.016990	0	0.001848	0	0	0	0	0	0
x_{28}	0.085977	0.028317	0.079835	0.066543	0	0	0	0	0	0
max	0.006	0.007	0.01	0.014	0.017	0.02	0.02215	0.02215	0.02215	0.02215
$CVaR_\beta$	18.8654	19.3313	19.6534	21.1112	21.7509	22.0129	22.6571	22.6571	22.6571	22.6571
VaR_β	7.8765	8.2541	8.7102	9.2342	9.6581	10.0201	10.2322	10.2322	10.2322	10.2322
Time(s)	3	6	8	10	13	16	19	21	23	25
k	27	28	25	20	17	8	5	5	4	4

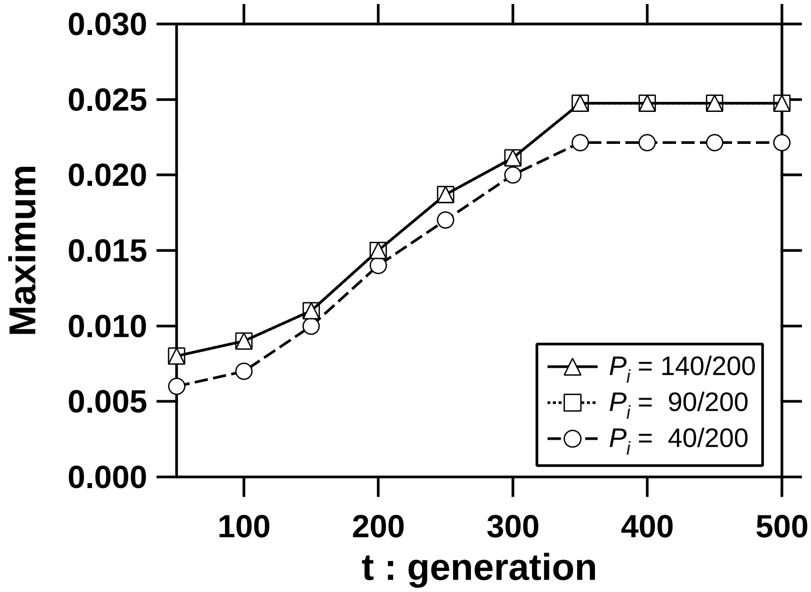

Figure 5: the Maximization under Different P_i

5 Conclusion This paper considered a new approach for simultaneous consideration of return and risk, and defined a new measure to improve performance of portfolio. Different to the famous Sharpe Ratio, the ratio of expect excess return to CVaR can be used in even extra case, which means the distribution of return is not normal. With the acknowledgement of coherence of CVaR, CVaR is widely used to value risk in many fields, consequently it is important to suggest the new measure and take place of the Sharpe Ratio presented by Sharpe. We use the Genetic Algorithm to solve the optimal problem of non-linear programming. Though, the Genetic Algorithm is an important part of a new area of applied research termed Evolutionary Computation. These are actually search processes and naturally useful for discovering optimum solutions. It is rare that the Genetic Algorithm is used in the field of finance, and to solve the problem of optimal portfolio among all kinds of securities. Numerical experiments have been shown that the Genetic Algorithm can be efficiently used to solve the problem of optimal portfolio. At the same time, this program we made is a general program, in other words, each unknown produces stochastically and the program keeps the best answer after each loop. This implies that the answer satisfied fundamental assumption, and it provides a means to maximization under acceptable risk levels. This optimization satisfied common sense that considers both profits and risk without emphasizing particularly on each side.

However, sometimes we must face different clients with their personal opinion. Some clients are conservative. They cannot accept higher risk, even compensate is larger proceeds. They have a limitation of risk. Of course, on the other hand, there are some adventurists who seek the largest profits. Using this program it is easily to satisfy them by changing a few parameters and set some restrictions.

References

Andersson,F., and Uryasev,S. (1999) *Credit risk optimization with conditional value-at-risk criterion*, Research Report 99-9,ISE Department,University of Florida.

Andersson,F., Mausser,H.,Rosen,D and S.Uryasev(1999) *Credit Risk Optimization with Conditional Value-at-Risk*, Mathematical Programming,Series B,December 2000.

Artzner,P., Delbaen F., Eber,J.M and D.Heath(1997) *Thinking Coherently*, Risk,10,November,68-71.

Artner,P., Delbaen F., Eber,J.M and D.Heath(1999) *Coherent Measures of Risk*, Mathematical Finance,9,203-228.

Werner, Dinkelbach(1967) *On Nonlinear Fractional Programming*, Management Science,13,492-498.

Embrechts,P., Kluppelberg,S., and T.Mikosch(1997) *Extremal Events in Finance and Insurance*, Springer Verlag.

Sharpe,William F.(1994) *The Sharpe Ratio*, Journal of Portfolio management,Fall,49-58.

Markowitz, H.M.(1952) *Portfolio Selection*, Journal of finance,7(1),77-91.

Mausser,H. and D.Ro *Efficient Risk/Return Frontiers for Credit Risk*, ALGO Research Quarterly,2(4),35-48.

Pflug, G.(2000) *Some Remarks on the Value-at-Risk and the Conditional Value-at-Risk*, Probabilistic Constrained Optimization:Methodology and Applications.Kluwer Academic Publishers.

Pritsker,M.(1997) *Evaluating Value at Risk Methodologies*, Journal of Financial Services Research,12(2/3),201-242.

Rockafellar,R.T. and S.Uryasev(2000) *Optimization of Conditional Value-at-Risk*, Journal of Risk,2,21-41.

Rockafellar,R.T. and S.Uryasev(2001) *Conditional Value-at-Risk for General Loss Distributions*, Research Report 2001-5.ISE Dept.,University of Florida,April,2001.

Simons,K.(1996) *Value-at-Risk New Approaches to Risk Management*, New England Economic Review,Sept/Oct,3-13.

Stambaugh,F.(1996) *Risk and Value-at-Risk* , European Management Journal,14(6),612-621.

Stublo Beder,T.(1995) *VaR:Seductive but Dangerous*, Financial Analysts Journal,Sep-Oct.,12-24.

Uryasev,S.(2000) *Conditional Value-at-Risk:Optimization Algorithms and Applications*, Financial Engineering News,14,February,2000.

Young,M.R.(1998) *A Minimax Portfolio Selection Rule with Linear Programming Solution*, Management Science,44(5),673-683.

Ziemba,W.T. and J.M.Mulvey(1998) *Worldwide Asset and Liability Modeling*, Cambridge Univ.Pr.

Shan LIN[11], Masamitsu OHNISHI[12]

[11]Graduate School of Economics, Osaka University, 1–7 Machikaneyama–machi, Toyonaka, Osaka 560-0043, Japan; E-mail:linshann@hotmail.com

[12]Graduate School of Economics, Osaka University, 1–7 Machikaneyama–machi, Toyonaka, Osaka. 560-0043, Japan; and Daiwa Securities Chair, Graduate School of Economics, Kyoto University, Yoshida-honmachi, Sakyo-ku, Kyoto. 606-8501, Japan; E-mail: ohnishi@econ.osaka-u.ac.jp