Procedia Computer Science

Volume 80, 2016, Pages 744–751

ICCS 2016. The International Conference on Computational Science

Novel Heuristic Algorithm for Large-scale Complex Optimization

Honghao Qiu[1*] and Yehong Liu[2†]

[1]*University of California, Berkeley, U.S.A*

[2]*The University of Hong Kong, Hong Kong, China*

`jimyau@berkeley.edu, liuyh@hku.hk`

Abstract

Research in finance and lots of other areas often encounter large-scale complex optimization problems that are hard to find solutions. Classic heuristic algorithms often have limitations from the objectives that they are trying to mimic, leading to drawbacks such as lacking memory-efficiency, trapping in local optimal solutions, unstable performances, etc. This work considers imitating market competition behavior (MCB) and develops a novel heuristic algorithm accordingly, which combines characteristics of searching-efficiency, memory-efficiency, conflict avoidance, recombination, mutation and elimination mechanism. In searching space, the MCB algorithm updates solution dots according to the inertia and gravity rule, avoids falling into local optimal solution by introducing new enterprises while ruling out of the old enterprises at each iteration, and recombines velocity vector to speed up solution searching efficiency. This algorithm is capable of solving large-scale complex optimization model of large input dimension, including Over Lapping Generation Models, and can be easily applied to solve for other complex financial models. As a sample case, MCB algorithm is applied to a hybrid investment optimization model on R&D, riskless and risky assets over a continuous time period.

Keywords: Large-Scale Complex Optimization, Heuristic Algorithm, Market Behavior, Investment Decision

1 Introduction

Complex modeling technique is widely used in applications of finance, operations research and other areas. As the complexity of computation soars up, many intelligent heuristic algorithms are developed to improve efficiency. J Holland came up with Genetic Algorithm in his work Adaptation in Natural and Artificial Systems (J Holland, 1988) that can approach optimal solutions. Later in 1995, Particle Swarm Optimization algorithm (RC Eberhart, 1995) was put forward as another heuristic algorithm to achieve high computation efficiency by group's mutual imitations. Intelligent algorithm research was divided to deterministic and nondeterministic algorithms after 1990s, and Simulated

[*] Created the first draft and algorithm part of this document
[†] Created the financial case in this document

doi:10.1016/j.procs.2016.05.364

Annealing algorithm (WL Goffe, 1994) as a popular probability based nondeterministic algorithm helped in solving problems in complex solution searching space. Later heuristic algorithms focuses on solving specific problems, for example, the harmony search algorithm (ZW Geem, 2001) for traveling salesman problem. More recent heuristic algorithms works on fundamental computer system control such as assigning Internet files (R Kolisch, 2015) and system state prediction (L Dong, 2015).

However, the above works have their limitations respectively due to the restrictions of objects that they are trying to imitate. Some intelligent algorithms like GA do not have the memory of past searching history, while some other algorithms like PSO often stuck in local optimal solutions (JW Zhuo, 2011). And some nondeterministic algorithms such as SA are not efficient and fast enough in situations where the solution searching space is large and complex. If we apply Monte Carlo Method in such situations aiming to speed up searching process, we lose stability of the searching results. Recent works often try to combine these methods and create hybrid algorithms such as hybrid genetic algorithm, while many of them are till restricted by the original objects that their algorithms are trying to imitate.

This paper focuses on finding another imitation object to build a new intelligent algorithm that could be capable of solving large-scale complex optimization problems with efficiency, stability and accuracy. We find out the Market Competition Behavior is a object that have comprehensive favorable characteristics to imitate in intelligent algorithm, and this algorithm can achieve efficiency, stability and accuracy that we want in complex searching space while avoiding local optimal solutions. We then use a novel investment decision case from financial engineering research to test and prove this algorithm.

2 The Market Competition Behavior (MCB) Algorithm

2.1 The Idea

The objective of Market Competition Behavior is to make decisions within restrictions based on past information over a series of time to develop optimal product of best return (in a certain industry), which is very similar to the process of finding optimal solution in a complex system. Market players compete under certain rules, steadily and effectively develop their products toward the optimal point. Different market competition behaviors provide comprehensive characteristics for our intelligent algorithm to imitate and aggregates to overcome past algorithms' drawbacks.

2.2 Main Features for Imitation

1. Dispersion: As many players are competing in a market and constantly updating products over time, we assume a dispersion of players in the market at the very beginning.
2. Identification: Identification can be categorized into individual identification and group identification. If one player in the market develops a successful product, then very likely it will capture some features leading to success in market and develop product with similar features in the next release. This is individual identification. At the same time, other players in this market will also notice and learn from those successful features and try to incorporate those features into their next product release, which is group identification.
3. Memory: The historical best selling product of the firm and historical best selling product of the market provides guidance for players in the market and allows them to develop better product faster. In other words, they keep their memory of historical product release and never produce products that are worse than before, which improves efficiency of developing toward the best product.

4. Vector-Recombination: The directions (vectors) of product development can recombine with each other across players in the market during competition. For example, one company may find out some features that are attractive to customers while another company may find out some other features, later it is likely that a company would try to combine features from both sides in their product development, which is the effect of vector-recombination. This improves the players' possibility of finding optimal product.
5. Elimination and Renewal: Once in a while some competitors are ruled out from the market due to bad performance, while some other new companies join the market trying to develop some new-featured products. The market characteristics by nature improve the overall performance of the players' performance (through elimination) and prevent current players from being stuck in local optimal solution instead of global optimal solution (through renewal).
6. Conflict Avoidance: The players would not collapse into each other and develop same products in market competition, which avoids repeated searching and unnecessary conflicts.

2.3 Model and Flowchart for Algorithm Development

In Market Competition Behavior (MCB) model each company is considered as a point in m dimensional searching space Ω, all n companies make up a point set denoted as $\{n\}$. The ith company's point is $P_i = (X_1, X_2, \dots, X_m)$. These points are randomized within the constraints in the searching space in initialization and updated later. Our algorithm set 100 points as default initialization size while this should be proportional to the size of the searching space. The total iteration time for points' update is set at 100 by default.

The objective optimization function is $f = F(X_1, X_2, \dots, X_m)$, which can be considered as return of each company in specific time. Each company point memorize the historical best return point it has found, which is local best point $P_{localbest} = (X_{lb1}, X_{lb2}, \dots, X_{lbm})$. And the best return point which has been found among all the historical products across all companies is called global best point and denoted as $P_{globalbest} = (X_{gb1}, X_{gb2}, \dots, X_{gbm})$.

The update for the point set is subject to influences from three forces: 1) Inertia: the inertia from past movement take into consideration by multiplying a inertia multiplier w for the velocity, we use $w = 0.6$; 2) Gravity force from local best: each time when one point is being updated, it tends to be attracted by the historical best return it has found before, we call this local best gravity force F_l; 3) Gravity force from global best: for all the points (companies) in the searching space (market), they are attracted by the historical best return point that has been ever found, and this force is denoted as F_g. Both gravity forces are inversely proportional to the Euclidean distance between each point and the historical best point, this feature allows points to quickly cluster to optimal point. Assume each point has 'mass' of 1, the update rule can be deducted as follows:

$$P_{new} = P_{old} + V_{new} \times dt,$$

$$V_{new} = w \times V_{old} + a \times dt,$$

$$a = \frac{F}{m} = \frac{F_l + F_g}{1} = c_l \times (P_{localbest} - P_{old}) + c_g \times (P_{globalbest} - P_{old}),$$

Combining the above equations together, we get:

$$P_{new} = P_{old} + \{w \times V_{old} + [c_l \times (P_{localbest} - P_{old}) + c_g \times (P_{globalbest} - P_{old})] \times dt\} \times dt.$$

Above is the position update trajectory function. Here dt is the update interim ($dt = 1$ as default value), while multiplier c_l and c_g are the noise for gravity force, meaning other external factors that may influence direction and distance for point (product) update such as confidence level, management change or new information added in the market. c_l and c_g are set as Normal Random Numbers.

As for the restriction functions, the k constrains are $g_1(P), g_2(P), \dots, g_k(P)$ and the constraint set is denoted as $\{g\}$. For initialization and each update, all the new points added should satisfy $\{g\}$, otherwise it will be renewed until meeting this requirement. Especially, the allowable maximum velocity V_{max} is 10% to 20% proportional to the size of searching space.

To imitate the feature of elimination and renewal, each time when we update the point set, ¼ companies points' with lowest objective function value will be ruled out and replaced by new points reinitialized in searching space. This feature helps avoid being stuck in local optimal solution.

As for vector-recombination, each time we update the point set, its velocity as a vector will be the recombination of new velocity of this point and the velocity of current global best point if it returns a higher objective value than merely applying the its own new velocity. Basing on repeated simulation, we find this feature improves searching efficiency by approximately 20%.

The simplified flowchart of MCB intelligent algorithm is shown below in Figure 1.

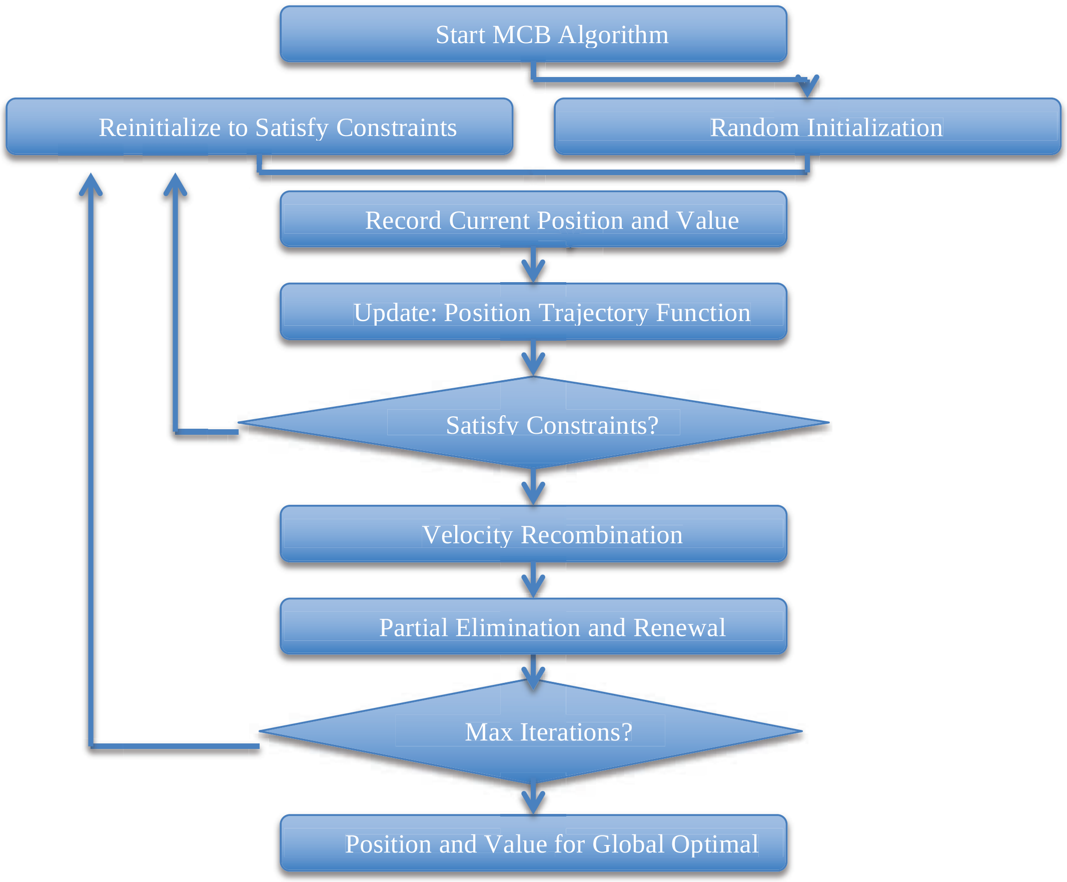
Start MCB Algorithm
Reinitialize to Satisfy Constraints
Random Initialization
Record Current Position and Value
Update: Position Trajectory Function
Satisfy Constraints?
Velocity Recombination
Partial Elimination and Renewal
Max Iterations?
Position and Value for Global Optimal

Figure 1: Flowchart of MCB Algorithm

3 Algorithm Application in Financial Investment Decision

3.1 Continuous Time R&D, Riskless and Risky Assets Hybrid Investment Model

In this paper we consider a hybrid investment decision across R&D, riskless and risky assets over a continuous time. Portfolio analysis with R&D project is a challenging work. On one hand, the value of R&D project can be estimated in many different ways and so is full of uncertainty. On the other hand, it needs to be considered together with risky and risk-free assets. Here we want to find out the best asset allocation strategy for different targets from $t = 1,2, \dots, T$ that leads to optimal return. At each time spot t, the allocable capital is: $A_t, t = 1,2, \dots, T; A_0$ as starting capital.

Assume m R&D projects to be considered. For R&D project, the market value (Economic Net Present Value) of each project can be calculated by adding its Real Option Value (ROV) and Net Present Value (NPV), which is: $ENPV = ROV + NPV$. We use B-S formula (F Black, 1973) to calculate ROV for each R&D project respectively: $ROV_{jt} = S_{jt}N(d_1) - C_j e^{-r_j(T-t)}N(d_2)$, where:

$$d_1 = \frac{\ln\left(\frac{S_{jt}}{C_j}\right) + (r_f + \frac{\sigma_j^2}{2})(T-t)}{\sigma_j\sqrt{T-t}}, d_2 = \frac{\ln\left(\frac{S_{jt}}{C_j}\right) + (r_f - \frac{\sigma_j^2}{2})(T-t)}{\sigma_j\sqrt{T-t}} = d_1 - \sigma_j\sqrt{T-t},$$

S_{jt} is the jth R&D project's NPV at time t, C_j is the capital investment (cost) for jth R&D project. r_f is the risk-free interest rate, σ_j is the standard variation of jth R&D project's market price. $N(.)$ is the standard normal distribution function.

Assume there are n risky assets to be considered, investment in risky assets at time t is denoted as Q_t. $w_t = (w_{1t}, w_{2t}, \dots, w_{nt})^T$ and $X_t = (X_{1t}, X_{2t}, \dots, X_{nt})^T$ denotes investment weigh and expected return rate on different assets at time t. Therefore, the overall expected return from t to $t+1$ is: $Q_t {w_t}^T X_t$. Weighs should be non-negative (assume shorting is not allowed) and add up to 1 for each time t: $w_t \geq 0$, $e^T w_t = 1, e = (1,1, \dots ,1)^T$.

Investment in risk-free asset over $t = 1, \dots T$ is: $RF = (RF_1, RF_2, \dots, RF_T)$. Thus, the return from risk-free asset from t to $t+1$ is: $RF_t r_f$, where r_f is the risk-free return rate.

As for the investment decision for jth R&D project at time t, we use binary indicator I_{jt} to quantify decision (1 means investing while 0 means not investing). The overall R&D investment at time t is: $Y_t = \sum_{j=1}^{m} C_j I_{jt}$. Since we only make investment decision once for each potential R&D project, there is: $\sum_{t=1}^{T} I_{jt} \leq 1, j = 1, \dots m$.

The above investments on three targets are subjected to budget limit: $Q_t + Y_t + RF_t \leq A_t, A_t \geq 0$ for each time t.

Available capital at time t is determined by: $A_t = A_{t-1} + Q_t {w_t}^T X_t + RF_t r_f - Y_t$.

Through deduction we have: $A_t = A_0 + \sum_{i=1}^{t} Q_i {w_i}^T X_i + \sum_{i=1}^{t} RF_i r_f - \sum_{i=1}^{t} Y_i$. The objective optimization function is the total available capital at exit time T:

$$\max\,[E(A_T + \sum_{t=1}^{T}\sum_{j=1}^{m} I_{jt}S_{jt} + \sum_{t=1}^{T}\sum_{j=1}^{m} I_{jt}ROV_{jt})]$$

In this case, r_f is set at current risk-free interest rate, capital requirement for each R&D project is a constant cost C, risky asset return rate and NPV for R&D follows VAR process.

3.2 Algorithm Implementation

This is a typical complex optimization model in financial investment: it is an overlapping generation model with large input parameters, many constraints and high computation complexity. Here we apply MCB intelligent algorithm to this model as a testing case.

In this model, we set investment periods $T = 10,$ number of risky assets and R&D projects $n = m = 5,$ lag length for VAR is 6, market variation is 0.1, risk-free interest rate at 0.08, cost for each R&D project is C=50, initial capital $A_0 = 1000$. Based on these parameters, we can randomly initiate X_t, S_{jt} (NPV), and ROV_{jt}.

The MCB algorithm initialize 100 points upon initialization, each point's P_i is a vector with the length of $(m + 1 + n)\times T = 110$: the first $m\times T$ stands for binary variable I, the next T stands for investment in risky asset Q, and the last $n\times T$ stands for weigh w. The maximum velocity allowed for this three parameters are set at 0.5, 100 and 0.3 respectively. Each time when we receive a new point's position P_i, I, Q, w is set, then we can take turns to calculate $RF(1), A(1), RF(2), A(2),$ ... Once all RF and A are calculated, we can put them into objective function and get the objective value. Each time upon position update, the following 5 constraints are checked to see if satisfied: $A \geq 0$; $Q \geq 0$; $f \geq A_0$; $A_0 \geq Q(1)$; $Q + Y + RF \leq A(\forall t)$.

3.3 Testing Results

According to 100 times' MCB algorithm running output, the optimal investment return of 1000 initial capital after 10 periods is approximately 2420. The corresponding optimal investment strategy is:

1) Invest the 4th R&D project at period 8, invest the 2nd R&D project at period 9, and invest the 1st, 3rd, and 5th R&D project at period 10;

2) Invest 1000、782、607、451、600、524、62、29、515 and 427 in risky assets over the 10 investment periods;

3) The weigh on each risky asset over 10 periods is:

(0.15,0.26,0.18,0.26,0.15), (0.20,0.17,0.24,0.14,0.25), (0.20,0.10,0.27,0.17,0.26),
(0.22,0.25,0.14,0.30,0.09), (0.30,0.17,0.26,0.11,0.16), (0.28,0.08,0.04,0.30,0.30),
(0.21,0.19,0.19,0.12,0.29), (0.27,0.23,0.14,0.19,0.17), (0.26,0.26,0.06,0.21,0.21),
(0.04,0.23,0.26,0.18,0.29).

Our MCB intelligent algorithm successfully solve for this complex optimization problem within 20 seconds, twice as effective as Simulated Annealing algorithm with similar optimal output. The starting point (optimal value at 2360), is the optimal value that Monte Carlo algorithm can find, while the MCB algorithm successfully find a higher optimal value at 2420, which is approximately 3% increase of investment return. Also, Monte Carlo method is high unstable in terms of the optimal value that it find in large-scale complex optimization model, while MCB algorithm almost always find the exact global optimal solution at around 2420. Figure 2 shows an one-time running result of MCB algorithm, as we can see, as iterations increase, the MCB algorithm steadily find better solution, and when it finds out a local optimal solution around Iteration No.24, it successfully get rid of local optimal and approaches the global optimal in the end.

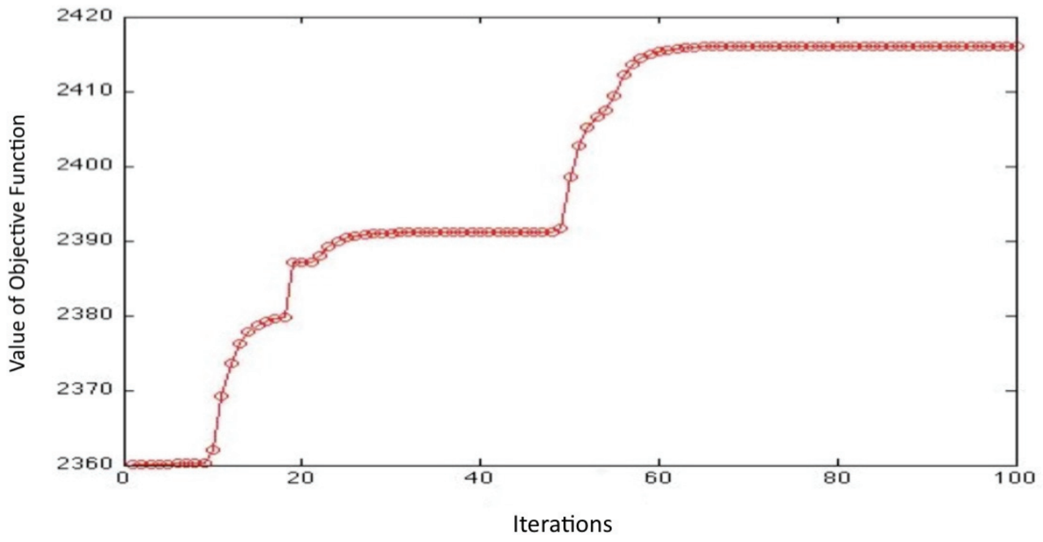

Figure 2: Sample Outcome of Running MCB Algorithm on Financial Optimization Case

4 Conclusions

This paper develops a new intelligent algorithm that can cope with large-scale complex optimization problems. By imitating market competition behavior, it embraces the strengths of current heuristic algorithms such as efficiency, accuracy, and stability. On the other hands, it overcomes the problems of finding local optimal solutions and lacking memory in searching process. In the financial investment case, the test result proves its effectiveness. Further, we can cross-test MCB algorithm with classic optimization algorithms and improve it by considering other competition forces in the market such as substitutes. It can also be combined with other heuristic algorithms in different phases of searching process to optimize speed and accuracy.

References

Ariadji, T., Haryadi, F., Rau, I. T., Aziz, P. A., & Rinaldy Dasilfa. (2014). A novel tool for designing well placements by combination of modified genetic algorithm and artificial neural network. *Journal of Petroleum Science and Engineering [J]*, .

Beaujon, G. J. (2001). "Balancing and optimizing a portfolio of R&D projects.". *Naval Research Logistics (NRL) 48.1* , 18-40.

Black, F. a. (1973). "The pricing of options and corporate liabilities.". *The journal of political economy*, 637-654.

Dong, L. e. (2015). "Predictive event-triggered control based on heuristic dynamic programming for nonlinear continuous-time systems.". *2015 International Joint Conference on IEEE*.

Eberhart, R. C. (1995). "A new optimizer using particle swarm theory.". *Proceedings of the sixth international symposium on micro machine and human science.*, Vol. 1.

Fang, Y. L. (2008). "A mixed R&D projects and securities portfolio selection model.". *European Journal of Operational Research 185.2*, 700-715. .

Geem, Z. W. (2001). "A new heuristic optimization algorithm: harmony search." . *Simulation 76.2*, 60-68.

Goffe, W. L. (1994). "Global optimization of statistical functions with simulated annealing." . *Journal of Econometrics 60.1*, 65-99.

Goldberg, D. E. (1988). "Genetic algorithms and machine learning.". *Machine learning 3.2*, 95-99.

Jinwu Zhuo, W. Y.-s. (2011). *"Application of MATLAB in mathematical modeling [M].".* Beijing: Beijing University of Aeronautics and Astronautics Press 4.

Kirkpatrick, S. C. (1983). "Optimization by simulated annealing.". *science 220.4598* , 671-680.

Niknam, T., & Amiri, B. (2009). An efficient hybrid approach based on PSO, ACO and k -means for cluster analysis. *Applied Soft Computing Journal* , [J] .

Stummer, C. a. (2001). "Interactive R&D portfolio selection considering multiple objectives, project interdependencies, and time: A three-phase approach.". *Management of Engineering and Technology*, .

Taillard, E. (1990). "Some efficient heuristic methods for the flow shop sequencing problem." . *European journal of Operational research 47.1*, 65-74.

Trelea, I. C. (2003). "The particle swarm optimization algorithm: convergence analysis and parameter selection." . *Information processing letters 85.6* , 317-325.