Hybrid Latin-Hyper-Cube-Hill-Climbing Method for Optimizing: Experimental Testing

Calista Elysia[1], Michelle Hartanto[2], Ditdit Nugeraha Utama[3]
Computer Science Department, BINUS Graduate Program – Master of Computer Science
Bina Nusantara University, Jakarta, Indonesia 11480

***Abstract*—A noticeable objective of this work is to experiment and test an optimization problem through comparing hill-climbing method with a hybrid method combining hill-climbing and Latin-hyper-cube. These two methods are going to be tested operating the same data-set in order to get the comparison result for both methods. The result shows that the hybrid model has a better performance than hill-climbing. Based on the number of global optimum value occurrence, the hybrid model outperformed 7.6% better than hill-climbing, and produced more stable average global optimum value. However, the model has a little longer running time due to a genuine characteristic of the model itself.**

***Keywords*—Hill-Climbing; Latin-Hyper-Cube; Optimization**

I. Introduction

Optimization, also known as mathematical programming, is a process or action to obtain the highest achievable performance under the given constraints. The final goal of all optimization is to minimize the effort required or to maximize the desired benefit. Several methods for optimization have been developed are hill-climbing (HC) [1], simulated-annealing (SA) [2], generic algorithm [3], particle swarm optimization [4], ant colony optimization [5], and others.

Many researchers have performed various studies in the scope of optimization. Author in [6] conducted a study to improve the sampling algorithm in the Bayesian network by applying the Latin-hyper-cube (LHC) sample. This study showed a better result compared to applying simple random sampling. Author in [7] discussed the Quasi-Newton methods which are an iterative optimization method. There is also a study that used the hill-climbing optimization method which was found to effectively detect grids on microarray images taken from databases from GEO and Stanford genomic laboratories [8]. Also, [9] constructed an optimization model using simulated-annealing and hill-climbing then compared them. The result said that hill-climbing consumed the shortest running time. Author in [10] applied genetic algorithm to optimize the control (process) parameters. Finally, [11] explained the success of simulated-annealing method for global optimization problems by studying the ideal version of the algorithm.

The objective of this study is to scientifically experiment an optimization problem by comparing HC method with a hybrid method combining LHC and HC (L2HC). While HC is going to use a random starting point in any position, the hybrid method is divided starting point into several identified clusters. This two methods are going to be tested using the same data set in order to get the comparison result for both methods. Here, the proposed hybrid method conjoining two methods LHC and HC is a novel-configuration in optimizing. It is an optimization method by dividing search-area via several definitive-clusters, and conducting searching alternatives in each cluster.

The organization of this paper is as follows. Section II presents a literature overview about optimization, HC, and LHC. The related works and experimental method is going to be delivered in Section III and Section IV, respectively. In Section V, we present experimental result and analysis. Finally, Section VI concludes the paper.

II. Literature Overview

A. Optimization

Optimization derived from Latin words ‘optimus’ which has the ‘best’ meaning, it can be interpreted as a process of finding conditions that provide optimal value from an objective function [12]. The optimal value can be a minimum value or a maximum value of the objective function in accordance with the existing problems (Fig 1).

The combination method L2HC will be discussed in this experimental paper. The HC method itself is included in the method of mathematical programming or modern (non-traditional) optimization techniques that are very useful for finding the optimal value of an objective function of several variables that are in certain constraints. They are heuristic methods and are often faster than exact methods, especially in the case of non-linear objective functions with many variables. While the LHC method is included in the statistical sampling method which is a method for analyzing experimental data and developing empirical models to get the most accurate representation of the physical situation [13].

B. Hill Climbing

HC is a method that aims to find local maximum or minimum values through simple iterations that continue to move towards increasing values if looking for maximum values or decreasing values if looking for minimum values from an objective function to finding the nearest peak value or closest valley point [12]. This method is very simple and has been successfully applied to various optimization problems. The success of this method is due to the fact that choosing a heuristic that more accurately predicts the actual solution that produces more opportunities to obtain the optimal solution [1][8][14][15]. Pseudocode of HC method is displayed in Code 1. Searching for optimum value with HC method is greedy and the search starts from a random starting point. This causes the

optimum results can be the local optimum, except for the random value of the luckiest starting point [16].

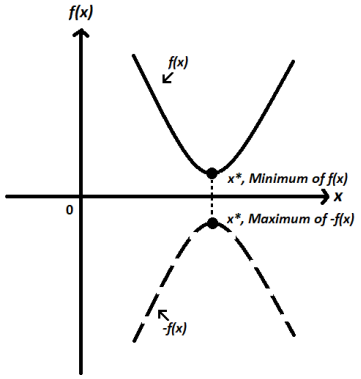

Fig. 1. Illustration of Optimization for Maximum and Minimum Value [13].

There are several parameters needed for the HC method, some of them are mandatory; such as, $cVal$, $bestNeighbor$, $bestVal$, which respectively represents the current value, neighbor's best value, and best value (local or global optimum). Based on the pseudocode, it can be comprehended that the iteration process is going to be terminated when the local optimum value has been obtained. That is when there is no value from the $bestNeighbor$ variable that is better than the value of the $cVal$ variable.

Code 1. Pseudocode of Hill Climbing Method [12]

```
Procedure HillClimbing()
Begin
   <...variables definition...>
   //randomizing new parameter combination
   cPos <-- random()
   cVal <-- objFunction(cPos)
   //looping until local optimum is found
   While(search is not terminated)
      //finding the best neighbor
      bestNeighbor <-- getBestNeighbor()
      //finding local optimum
      If(cVal>=bestNeighbor)
         Local optimum is found
         Search is terminated
      Else //move to next position
         cVal <-- bestNeighbor
         cPos <-- bestNPos
      End if
   End while
```

C. Latin-Hyper-Cube

LHC was proposed by [17], which is a method for selecting samples from populations. LHC has been proven to be able to reduce variance [18] and produce better analysis results compared to simple random sampling method [19]. In this method, how many intervals of equal probability so input variable value can be divided into several cubes according to the intervals are firstly determined. Each parameter combination is randomly selected and is determined how many combinations of parameters can be in the same group. After getting a combination of parameters randomly, check which group of parameters the combination is. If it exceeds the maximum number of parameter combinations in that group, the value must be randomized again [12].

Code 2 shows a pseudocode for the LHC method when implemented as an optimization method. Here, randomizing a position from the population is required, and the clusters checked then by using a function named $checkCluster()$. It will return a Boolean value and refers to equal status ($eqStatus$) variable, returned 'true' if it exceeds the maximum number in that group, and 'false' if it able. This random process will stop when $eqStatus$ is 'false'.

Code 2. Pseudocode of Latin-Hyper-Cube Method [12]

```
Procedure LatinHypercubeSampling()
Begin
   <...variables definition...>
   <...parameters cluster making...>
   bestVal <-- 0
   While(loopCount)
      //randomizing new parameter combination and
      //checking the equality
      eqStatus <-- true
      While(eqStatus=true)
         //randomizing new parameter
         cPos <-- random()
         //checking cluster equality status
         eqStatus<--checkCluster(cPos)
      End while
      cVal <-- objFunction(cPos)
      //checking the best value
      If(cVal > bestVal)
         bestVal <-- cVal
         bestPos <-- cPos
      End if
      loopCount++
   End while
End
```

III. Related Works

Numerous studies exploring about LHC or HC method have been already conducted by many researchers. Author in [20] adopted LHC sampling to design a sophisticated gas turbine. This sampling method is applied recursively to identify the most important input parameters. Author in [21] proposed a new method named 'IDLHCSA' for history matching to get reliable forecast. 'IDLHCSA' combines iterative discrete Latin-hyper-cube (IDLHC) to find good matched models with SA method.

Furthermore, [22] examined the performance of HC method for mesh router node placement in wireless mesh network. The result shows that connectivity and user coverage are achieved well. Author in [23] applied smart HC using the ideas of LHC sampling to find an optimal configuration for web application server. This proposed method can learn from previous searches and more efficient than traditional heuristic methods. Also, [24] operated HC method to prove about statement of 'optimization has a better solution when it closer to the local optimum value' is wrong. Besides the local optimum value, number of steps to reach the local optimum also needs to be considered.

Particular in hybrid model, [25], in discussing a communication behavior for social dynamical systems, studied a hybrid opinion network containing of continuous valued and discrete valued agents. The agents talked about were copier, voter, and averager agents. Here, the communication topologies were modeled. The study concluded that the voters'

existence has dissimilar impact on the evolution and consensus value of negotiation process.

Additionally, [26] also investigated resilient consensus problem in hybrid multi-agent system. The hybrid multi agent system itself consists of continuous time and discrete time dynamical agents. Author in [26] successfully constructed a hybrid censoring strategy to reach resilient consensus. The consensus here defined as compromise between cooperative agents and Byzantine agents (as uncooperative agents).

IV. Research Methodology

This experiment involved five stages (Fig 2); i.e. preliminary study, construct model, data preparation, experiment, and evaluation. In the fisrt stage, optimization, HC, and LHC methods were learned deeply. Then, the model was constructed using class diagram. For the next two stages, python 3.6 is functioned methodically for generating the data set and test the optimization method. We performed this experiment by using 7th generation of Intel core i7 processor, 8GB RAM, and operating system Windows 10 64-bit as the hardware and software specifications. With the help of interpolate.lagrange from scipy library and linspace from numpy, data are generated (Fig. 3) and the equation is provided in □1). It is going to produce interpolated values randomly. The last stage is to perform an evaluation by analyzing the experiment result.

In addition, while doing the experiment stage, each method randomizes 100 starting point for one process. It means that HC will do 100 iterations while hybrid L2HC will do 20 iterations with 5 starting point for each iteration. This process is done 10 times in this work so the total iteration obtained is 1000 iterations.

$$f(x) = -6{,}317\text{e}-44x^{13} - 2{,}611\text{e}-39x^{12} + 2{,}727\text{e}-34x^{11} - 7{,}965\text{e}-30x^{10} + 1{,}241\text{e}-25x^{9} - 1{,}196\text{e}-21x^{8} + 7{,}563\text{e}-18x^{7} - 3{,}197\text{e}-14x^{6} + 8{,}987\text{e}-11x^{5} - 1{,}631\text{e}-7x^{4} + 0{,}000179x^{3} - 0{,}1038x^{2} + 24{,}48x + 923 \quad (1)$$

Fig. 2. Experiment Stages.

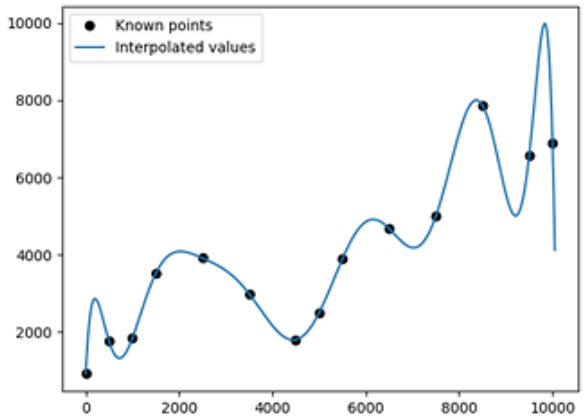

Fig. 3. Dataset Operated in Experiment.

V. Result and Analysis

Schematically, the constructed model is configured by Fig 4. It consists of three classes, where class L2HC (the class symbolizes the proposed hybrid model) is consisting two classes LHC and HC. All attributes defined in a main class, where they practically belong to and able to be equipped by both classes LHC and HC.

The pseudocode of L2HC is provided in Code 3. HC has seven parameters that are initiated and defined (section B). The first is *currentPosition* which defines the initial position that will be used in the optimization search and *currentValue* is the value of that *currentPosition*.

Then, bestNeighborPosition defines the position of the best neighbor around currentPosition and bestNeighborValue is the value of that bestNeighborPosition. LocalOptimum is found when there is no neighbor that have better value than current value, and terminatedLoop is a Boolean which indicates when the search must stop. There are also operations that are used here, they are definingOF() where the objective function for optimization is defined, randomizingPosition() for initial positioning, findingBestNeighbor() to find the best neighbour value, movingToNext(), and findingBest() to find the best value.

The idea of hybrid L2HC method is applying LHC method to HC method. This method is done by dividing the data into several clusters using clustering operation and running a HC method in each cluster. It must be ensured that the process is run only once on each cluster, this is checked by the *checkCluster*() function. Can be concluded here that the other parameters used in L2HC model are *clusterNumber*, *loopCount*, and *eqStatus*.

The challenging work in developing the proposed model is to merge the part of clustering algorithm with the part of HC searching. At this point, the searching part was inserted in the clustering looping fragment (see Code 3). It technically affects that the proposed model operates two types of looping block for both checking the clusters and searching the optimal value.

Practically, it was dissimilar with [25] and [26] for catching consensus rate, here we constructed a hybrid model via combining two types of methods for obtaining new optimization value for heuristic optimization problem.

After running 10 times process, the number of occurrence of HC method to get the global optimum value was 73 and hybrid L2HC got 85 times global optimum value from 1,000 iterations. From the obtained global optimum, 53.8% is produced by the hybrid method, outperforming HC method around 7.6%. Details of number occurrence for each process can be seen in Fig 5.

The average of local optimum obtained from each iteration is calculated and presented in Fig 6. We can see 8 out of 10 processes show hybrid L2HC method is better than HC. Also, the average obtained by hybrid L2HC method is much more stable. In other hands, HC produces an up and down average value. This can be happened because LHC will divide population into several clusters and randomize starting point

for each cluster. So the starting point for each iteration in the process more or less will be in the similar position. Therefore, it will lead to a similar local optimum value.

Code 3. Pseudocode of the Hybrid L2HC Method

```
Procedure LHS_HC()
Begin
   initialize iteration
   initialize numCluster
   while(it<iteration)
      <...variables definition...>
      <...parameters cluster making...>
      While(loopCount<numCluster)
         eqStatus <-- true
         While(eqStatus = true)
            cPos <-- random()
            eqStatus <-- checkCluster(cPos)
         End While
         search <-- true
         cVal <-- objFunction(cPos)
         While(search is not terminated)
            bestNeighbor <--
getBestNeigh(cPos,
            totalData)
            If(cVal>=bestNeighbor)
               opt = cVal
               //Local optimum is found
               Search is terminated
            Else //move to next position
               cVal <-- bestNeighbor
               cPos <-- bestNPos
            End if
         End While
         loopCount++
      End while
      it++
   End while
```

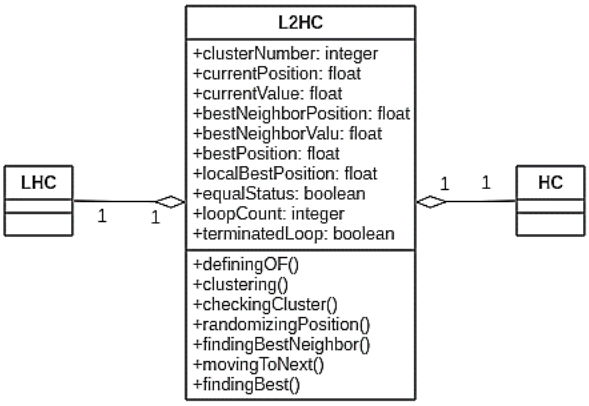

Fig. 4. Constructed Model.

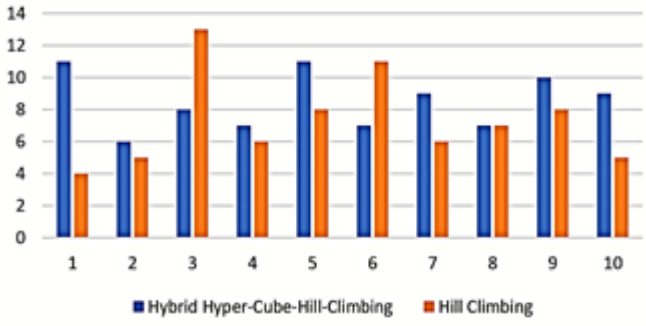

Fig. 5. Comparison of Global Optimum Occurrence.

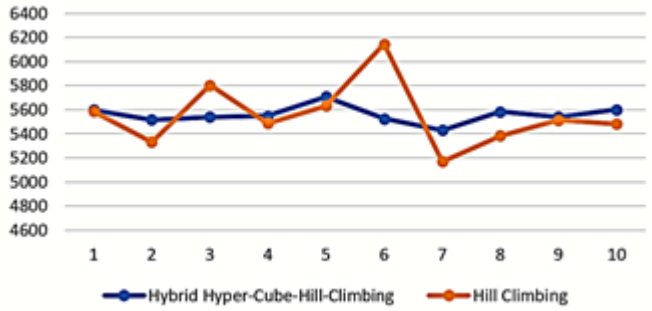

Fig. 6. Average Comparison of Optimum Value.

TABLE. I. AVERAGE OF RUNNING TIME FOR EACH PROCESS

	Hill Climbing	*Hyper-Cube-Hill-Climbing*
Running Time	0,001718695	0,002031107
	0,001718657	0,001718674
	0,002031164	0,002968359
	0,002031169	0,002031209
	0,001562223	0,002187619
	0,001562517	0,002187636
	0,002187026	0,001874907
	0,00203126	0,002030938
	0,001718521	0,001718628
	0,002031105	0,001562419
Average	**0,001859234**	**0,002031150**

Furthermore, average of running time is also calculated and displayed in Table I. Here, we got a fact about the average running time of 10 times process with hybrid L2HC method is more worse than HC. Hybrid L2HC need more time to randomizing the starting point. It will always randomize if starting point in one iteration has the same cluster.

VI. CONCLUSION

Experiments using two optimization model, namely, HC and hybrid L2HC, were done in this work. The conclusion revealed that hybrid L2HC has a better performance than HC itself. Based on the number of global optimum value occurrence, hybrid L2HC model outperformed 7.6% better than HC, and from the average comparison of the global optimum value, hybrid L2HC model is more stable and has a greater number. However, hybrid L2HC has a little longer running time. This happened because when randomizing the starting position, it is going to check whether the cluster where the position is located has been initialized or not, while hill climbing does not do the cluster checking first.

The future study for this work is regarding the effectiveness of the model. Although both models are good for optimization problems, it is found in this work that the effectiveness of both HC and hybrid L2HC are still need to be improved. Their effectiveness respectively are only 7.3% and 8.5%. Considering this value, hopefully further research can improve the model.

ACKNOWLEDGMENT

We would like to thank BINUS University who has supported our works, particularly BINUS graduate program, Master of Computer Science.

REFERENCES

[1] P. R. Norvig and S. A. Intelligence, A modern approach. Prentice Hall, 2002.

[2] N. Metropolis, A. W. Rosenbluth, M. N. Rosenbluth, A. H. Teller, and E. Teller, "Equation of state calculations by fast computing machines," J. Chem. Phys., vol. 21, no. 6, pp. 1087–1092, 1953.

[3] J. H. Holland, "Genetic algorithms," Sci. Am., vol. 267, no. 1, pp. 66–73, 1992.

[4] R. Eberhart and J. Kennedy, "A new optimizer using particle swarm theory," in MHS'95. Proceedings of the Sixth International Symposium on Micro Machine and Human Science, 1995, pp. 39–43.

[5] A. Colorni, M. Dorigo, and V. Maniezzo, "Distributed optimization by ant colonies," in Proceedings of the first European conference on artificial life, 1992, vol. 142, pp. 134–142.

[6] J. Cheng and M. J. Druzdzel, "Latin hypercube sampling in Bayesian networks.," in FLAIRS Conference, 2000, pp. 287–292.

[7] K. Bryan, "Quasi-Newton Methods," Rose-Hulman Inst. Technol., 2004.

[8] L. Rueda and V. Vidyadharan, "A hill-climbing approach for automatic gridding of cDNA microarray images," IEEE/ACM Trans. Comput. Biol. Bioinforma., vol. 3, no. 1, p. 72, 2006.

[9] D. N. Utama, N. Ani, and M. M. Iqbal, "An optimal generic model for multi-parameters and big data optimizing: A laboratory experimental study," in Journal of Physics: Conference Series, 2018, vol. 978, no. 1, p. 12045.

[10] R. Malhotra, N. Singh, and Y. Singh, "Genetic algorithms: Concepts, design for optimization of process controllers," Comput. Inf. Sci., vol. 4, no. 2, p. 39, 2011.

[11] H. E. Romeijn and R. L. Smith, "Simulated annealing and adaptive search in global optimization," Probab. Eng. Informational Sci., vol. 8, no. 4, pp. 571–590, 1994.

[12] D. N. Utama, "The Optimization of the 3-d Structure of Plants, Using Functional-Structural Plant Models. Case Study of Rice(Oryza sativa L.) in Indonesia," Georg-August University, 2015.

[13] S. S. Rao, Engineering optimization: theory and practice. John Wiley & Sons, 2009.

[14] A. Cawsey, The essence of artificial intelligence. Prentice Hall PTR, 1997.

[15] K. A. Sullivan and S. H. Jacobson, "A convergence analysis of generalized hill climbing algorithms," IEEE Trans. Automat. Contr., vol. 46, no. 8, pp. 1288–1293, 2001.

[16] J. Boyan and A. W. Moore, "Learning evaluation functions to improve optimization by local search," J. Mach. Learn. Res., vol. 1, no. Nov, pp. 77–112, 2000.

[17] M. D. McKay, R. J. Beckman, and W. J. Conover, "Comparison of three methods for selecting values of input variables in the analysis of output from a computer code," Technometrics, vol. 21, no. 2, pp. 239–245, 1979.

[18] M. Stein, "Large sample properties of simulations using latin hypercube sampling," Technometrics, vol. 29, no. 2, pp. 143–151, 1987.

[19] J. C. Helton and F. J. Davis, "Latin hypercube sampling and the propagation of uncertainty in analyses of complex systems," Reliab. Eng. Syst. Saf., vol. 81, no. 1, pp. 23–69, 2003.

[20] A. M. Briones, D. L. Burrus, J. P. Sykes, B. A. Rankin, and A. W. Caswell, "Automated Design Optimization of a Small-Scale High-Swirl Cavity-Stabilized Combustor," J. Eng. Gas Turbines Power, vol. 140, no. 12, p. 121509, 2018.

[21] C. Maschio and D. J. Schiozer, "A new methodology for history matching combining iterative discrete Latin Hypercube with multi-start simulated annealing," J. Pet. Sci. Eng., vol. 169, pp. 560–577, 2018.

[22] A. Xhafa, E. Spaho, D. Elmazi, and M. Takizawa, "A Study on Performance of Hill Climbing for Router Placement in Wireless Mesh Networks," in 2015 10th International Conference on Broadband and Wireless Computing, Communication and Applications (BWCCA), 2015, pp. 460–465.

[23] B. Xi, Z. Liu, M. Raghavachari, C. H. Xia, and L. Zhang, "A smart hill-climbing algorithm for application server configuration," in Proceedings of the 13th international conference on World Wide Web, 2004, pp. 287–296.

[24] L. Hernando, A. Mendiburu, and J. A. Lozano, "Hill-Climbing Algorithm: Let's Go for a Walk Before Finding the Optimum," in 2018 IEEE Congress on Evolutionary Computation (CEC), 2018, pp. 1–7.

[25] Y. Shang, "Hybrid consensus for averager-copier-voter networks with non-rational agents," Chaos, Solitons & Fractals, vol. 110, pp. 244-251, 2018.

[26] Y. Shang, "Consensus of hybrid multi-agent systems with malicious nodes," IEEE Transactions on Circuits and Systems II: Express Briefs, pp. 1-1, 2019.