MOPSO for BIM: A Multi-Objective Optimization Tool Using Particle Swarm Optimization Algorithm on a BIM-based Visual Programming Platform

Zhen Han [1] , Ning Cao [1] , Gang Liu [1] , Wei Yan [2]

[1] Tianjin University
{dkhertz\lutteuse\lglgmike}@163.com
[2] Texas A&M University
wyan@tamu.edu

Abstract. With the increasing applications of computational methods in the field of design optimization, intelligent metaheuristic algorithms are playing a more important role in building performance optimization. To enable the integration of optimization algorithms with Building Information Modeling (BIM), this research implemented the Particle Swarm Optimization (PSO) algorithm on Revit + Dynamo, which is a parametric BIM platform. A Multi-Objective PSO (MOPSO) Solver has been developed in Dynamo using MATLAB and C# programming languages. The methodology of the research and the validation studies are presented in the paper. The validation studies prove the effectiveness of the MOPSO Solver for both standard optimization test functions and an optimization example of a simplified building design.

Keywords: Particle Swarm Optimization, BIM, multi-objective optimization, visual programming.

1 Introduction

Design optimization has become an important design method supported by the advancement of computing technologies. The optimization technology is based on mathematics and computational sciences, as well as an understanding of the design problems. In real life, almost all decision-making problems need to deal with the conflicts among multiple objectives under different constraints at the same time. Multi-objective optimization provides decision makers with multiple solutions to specific problems, and provides architect and designers with design alternatives as decision-making basis [1].

There has been extensive research on multi-objective optimization in the field of architectural design using optimization algorithms. From traditional optimization algorithms (such as gradient-based search) to intelligent metaheuristic algorithms, the

processes and methods of optimization are different. For example, Pilla et al. [2] took building energy consumption and costs as the objective functions to optimize the use of energy-saving technologies in various regions on LINDO (a software package for linear programming, integer programming, nonlinear programming, stochastic programming and global optimization), and the optimization results significantly reduce the costs of energy consumption. Yang and Lin [3] optimized the materials of envelope structure of a green building by using NSGA-II in order to minimize the cost of building envelope, energy consumption of the envelope and maximize the opening rate of windows. Evangelos-Nikolaos and Madias et al. [4] took energy efficiency, sufficient illumination and maximum illumination uniformity as the objective functions and optimized the luminance of the lamps with the help of the global optimization toolbox in MATLAB and Relux light simulation platform to verify the feasibility and flexibility of the artificial lighting optimization model. Tezer T. et al [5] studied Genetic Algorithm (GA), Particle Swarm Optimization (PSO), Evolutionary Algorithm (EA) methods for optimizing independent hybrid renewable energy systems. The acquisition, maintenance, replacement, operation and maintenance costs of hybrid system components are analyzed in order to optimize the objectives that include system costs and system reliability, and to determine the effectiveness of the multi-objective optimization methods. Delgarm et al. [6] used the artificial bee colony algorithm to solve the multi-objective optimization of building energy-saving performance and indoor thermal comfort. Taking the room orientation, the window size, air conditioning setting temperatures, the glass and wall material performance as decision variables, they conducted the performance simulations on the platform of EnergyPlus and jEPlus, and optimization on MATLAB.

With the increasing applications of various optimization algorithms in the AEC community, especially in the field of building performance optimization, more intelligent metaheuristic algorithms gradually assert the dominance over traditional algorithms due to being flexible with the conditions of decision variables, objective functions and constraints, and the ability to deal with large-scale and complex optimization problems [7]. There is a need to integrate intelligent metaheuristic algorithms with Building Information Modeling (BIM) to facilitate a streamlined building modeling, simulation, and optimization process.

Considering the above need and problem, this paper presents the research and implementation of one of the most important metaheuristic optimization algorithms - the Particle Swarm Optimization (PSO) algorithm - on the BIM authoring tool Revit and its visual programming companion Dynamo, and the published package: MOPSO in Dynamo. This is an significant addition to the existing optimization packages in BIM, such as the Optimo Genetic Algorithm tool in Dynamo [8].

2 PSO Algorithm

In 1995, Kennedy and Eberhart [9] proposed a parallel stochastic algorithm, particle swarm optimization algorithm, based on the bird swarm intelligence behavior. PSO belongs to the metaheuristic algorithms, and is a typical swarm intelligence

algorithm based on animal social behavior. The PSO algorithm's major steps are as follows in pseudo code:

```
Initialize particles (solution vectors) randomly and their velocities  = 0
Loop until stopping criteria met: maximum iterations or minimum error {
        For each particle {
                Calculate fitness value
                If the fitness value is better than particle's personal best (pBest) {
                        Set pBest = current fitness value
                }
                If pBest is better than global best (gBest) {
                        Set gBest = pBest
                }
        }
        For each particle{
                Set particle Velocity = weighted sum (previous velocity, difference
                        between particle's current vector and pBest's, and difference
                        between the particle's current vector and gBest's)
                Update particle using Velocity
        }
}
```

Visually, PSO can be illustrated in Fig. 1. with the meanings of the symbols explained in Table 1.

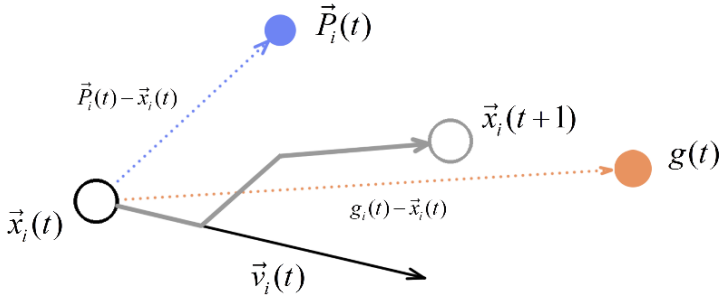

Fig. 1. Schematic diagram of particle swarm optimization algorithm

Table 1. Explanation of symbols

Symbols	Explanation
i	Particle i
t	Time t
$x_i(t)$	The location of particle i at time t
$p_i(t)$	The best location particle i has found before time t (pBest)
$g(t)$	The best location population have found before time t (gBest)
$x_i(t+1)$	The location of particle i at time $t+1$

So, the location and velocity of particle *i* at time *t+1* can be calculated as followings:

$$v_i(t+1) = wv_i(t) + r_1C_1(p_i(t) - x_i(t)) + r_2C_2(g(t) - x_i(t)) \quad (1)$$

$$x_i(t+1) = x_i(t) + v_i(t+1) \quad (2)$$

where w is inertia factor; r1 and r2 are random variables between 0 and 1; C1 is Cognitive Component and C2 is Social Component. Multi objective particle swarm optimization (MOPSO) is a method based on PSO to solve multi-objective optimization problems. The PSO algorithm and evolutionary algorithms have many similarities, such as swarm search (with populations and individuals) and information exchange in the search process, which suggests that there are many similarities between MOPSO and multi-objective evolutionary algorithms. At the same time, there are also some differences between MOPSO and multi-objective evolutionary algorithm: there are multiple best positions in the group at the same time, there are also many optimal positions of particles in the iterative process, so gBest and pBest also need to adopt certain strategies to choose.

3 MOPSO Solver

While PSO has been studied for building design optimization (Rapone and Saro 2012 [10]), this research developed an integrated PSO and BIM system through the components (classes) of MOPSO, created as a new package in Dynamo, shown in Fig. 2. The components include five main visual programming nodes as demonstrated in Fig. 3: four user input nodes with which users can input the objective functions, the parameters of the PSO algorithm, the constraints, and non-linear constraints, and one solver node which can solve the problem and output the Pareto Optimal Front into an Excel file. These nodes are explained below.

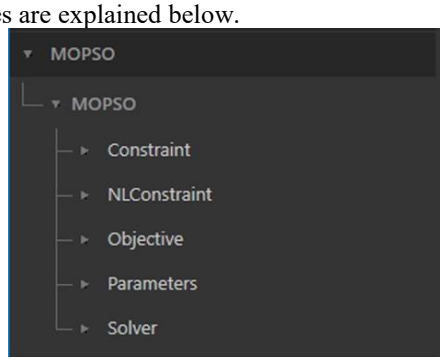

Fig. 2. The components (classes) of MOPSO

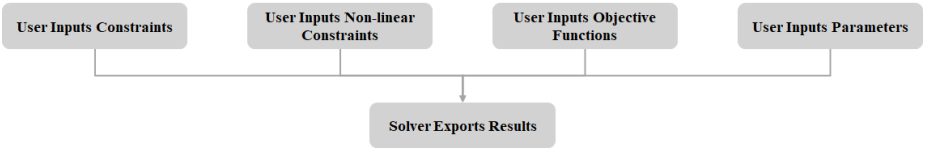

Fig. 3. MOPSO Solver Structure

• Constraint Node:

This node has 6 inputs and 1 output, which are used to create linear constraints of objective functions and define lower and upper bounds of decision variables. As the node diagram shown in Fig. 4, *lb* and *ub* stand for the lower and upper bounds of decision variables; *Aeq* and *beq* stand for the coefficient and constant of decision variables in the linear equality constraints; *A* and *b* stand for the coefficient and constant of decision variables in the linear inequality constraints. For example, there is a multi-objective optimization problem described as formula (5), whose two decision variables, *x* and *y*, subject to the conditions described as formula (3):

$$\text{s.t.}\begin{cases} 0.1 \le x_1 \le 1 \\ 0 \le x_2 \le 5 \\ g_1(x,y) = -9x_1 - x_2 \le -6 \\ g_2(x,y) = -9x_1 + x_2 \le -1 \end{cases} \tag{3}$$

According to formula (3): the coefficient of decision variables in the linear inequality constraint is -9, -1, -9, 1. So, we can input a one-dimensional float array, {-9, -1, -9, 1}, to A. And there is no linear equality constraint of decision variables, so we can give an empty array to *Aeq* and *beq*. The final result is shown in Fig. 4.

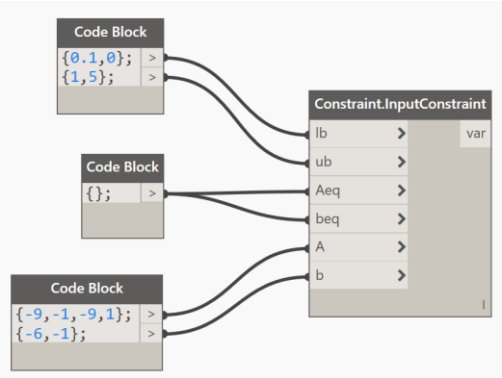

Fig. 4. Constraint Node

• NLConstraint Node:

This node has 3 inputs and 1 output, which are used to create nonlinear constraints on objective functions. As the node diagram shown in Fig. 5, *NLfunc* and *NLc* stand for the nonlinear constraints of decision variables; *nNLfun* stands for the number of the nonlinear constraints. For example, there is a multi-objective optimization problem which has two nonlinear constraints satisfying the conditions described as formula (4):

$$\text{s.t.}\begin{cases} g_1(x_1,x_2) = (x_1-5)^2 + {x_2}^2 \le 25 \\ g_2(x_1,x_2) = -((x_1-8)^2 + (x_2+3)^2) \le -7.7 \end{cases} \tag{4}$$

So, we can input a string array and a float array to *NLfunc* and *NLc*. The final result is shown in Fig. 5.

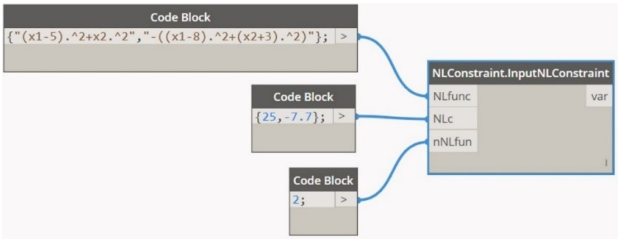

Fig. 5. NLConstraint Node

• Objective Node:

This node has 3 inputs and 1 output, which are used to create the objective function. As the node diagram shown in Fig. 6, *func* stands for the objection functions of the multi-objective optimization problem; *nVar* and *nFun* stand for the numbers of the decision variables and objective functions, respectively.

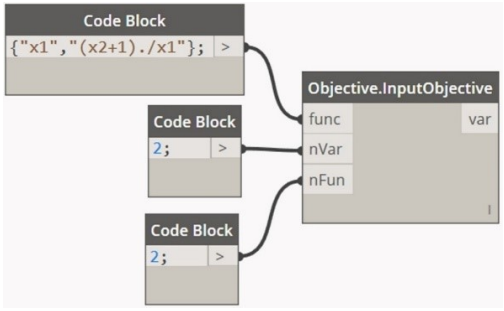

Fig. 6. Objective Node

• Parameters Node:

This node has 3 inputs and 1 output, which are used to set parameters related to population and individuals. As the node diagram shown in Fig. 7, for the input of this node, the default values can be used except for the three variables *Np* (population size), *Nr* (size of external memory with elitism strategy), and *maxgen* (generation number).

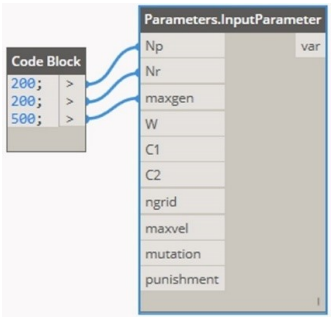

Fig. 7. Parameters Node

• Solver Node:

Figure 8 shows the final node - the solver of MOPSO, which has 4 inputs and 1 output. After accepting the output parameters of the preceding four nodes, this can output the result, Pareto Optimal Front, into an Excel spreadsheet.

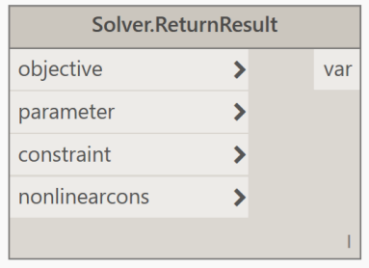

Fig. 8. Solver Node

The Multi-Objective Particle Swarm Optimization (MOPSO) algorithm is realized by programming using MATLAB and C#. The system employs the powerful scientific calculation capacity of MATLAB R2016a to create the PSO algorithm core and its testing. Additionally, based on the Microsoft Visual Studio 2017 IDE, this work accomplishes the encapsulation, transformation and transmission of the variables, enabling the solver core to read the input data from Dynamo, and exporting the calculation results and major data into an Excel file. The schematic view of MOPSO in Dynamo is shown in Fig. 9.

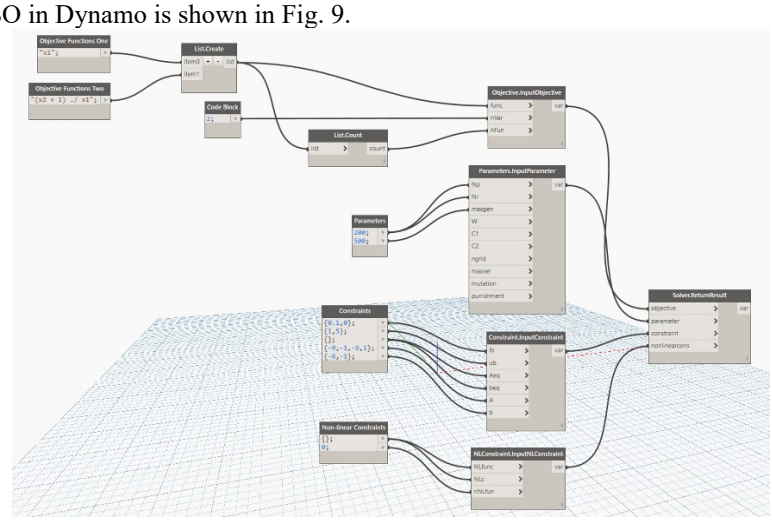

Fig. 9. The schematic view of MOPSO in Dynamo

Fig 9 demonstrates a sample multi-objective optimization test case – a modified version of the standard test case - Constr-Ex. The test case objective functions and constraints are shown as formula (5) and formula (6). Running the developed program, an Excel file containing the values of variables and objective functions are generated. According to the data in the file, the Pareto Front can be plotted by data processing software (MATLAB, Tableau etc.) for users to view. Fig. 10. shows the Pareto Front obtained after the running of MOPSO in Dynamo for this case study.

$$Minimize = \begin{cases} f_1(x_1, x_2) = x_1 \\ f_2(x_1, x_2) = \dfrac{1 + x_2}{x_1} \end{cases} \tag{5}$$

$$\text{s.t.} \begin{cases} 0.1 \le x_1 \le 1 \\ 0 \le x_2 \le 5 \\ g_1(x, y) = -9x_1 - x_2 \le -6 \\ g_2(x, y) = -9x_1 + x_2 \le -1 \end{cases} \tag{6}$$

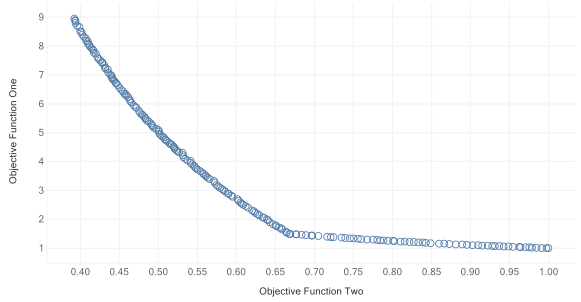

Fig. 10. Pareto Front for a test case

4 Validation Study

Test functions are very useful and commonly used to evaluate the characteristics of optimization algorithms. In this paper, three test functions [11] are used to prove the validity of the MOPSO solver. Table 2 shows the details of the three test functions and Fig. 11. compares the standard Pareto Front with the Pareto Front obtained by the MOPSO algorithm coded in this research.

Table 2 Multi-objective optimization test functions used in this study

Name	Functions	Constraints	Search Domain
Schaffer function N.1	$Minimize = \begin{cases} f_1(x) = x^2 \\ f_2(x) = (x-2)^2 \end{cases}$	none	$-10 \le x \le 10$
Poloni function	$Minimize = \begin{cases} f_1(x,y) = [1 + (A_1 - B_1(x,y))^2 + (A_2 - B_2(x,y))^2 \\ f_2(x,y) = (x+3)^2 + (y+1)^2 \end{cases}$ $where = \begin{cases} A_1 = 0.5\sin(1) - 2\cos(1) + \sin(2) - 1.5\cos(2) \\ A_2 = 1.5\sin(1) - \cos(1) + 2\sin(2) - 0.5\cos(2) \\ B_1(x,y) = 0.5\sin(x) - 2\cos(x) + \sin(y) - 1.5\cos(y) \\ B_2(x,y) = 1.5\sin(x) - \cos(x) + 2\sin(y) - 0.5\cos(y) \end{cases}$	none	$-\pi \le x, y \le \pi$
Constr-Ex function	$Minimize = \begin{cases} f_1(x,y) = x \\ f_2(x,y) = \dfrac{1+y}{x} \end{cases}$	$\text{s.t.} = \begin{cases} g_1(x,y) = 9x + y \ge 6 \\ g_2(x,y) = 9x - y \ge 1 \end{cases}$	$0.1 \le x \le 1$ $0 \le y \le 5$

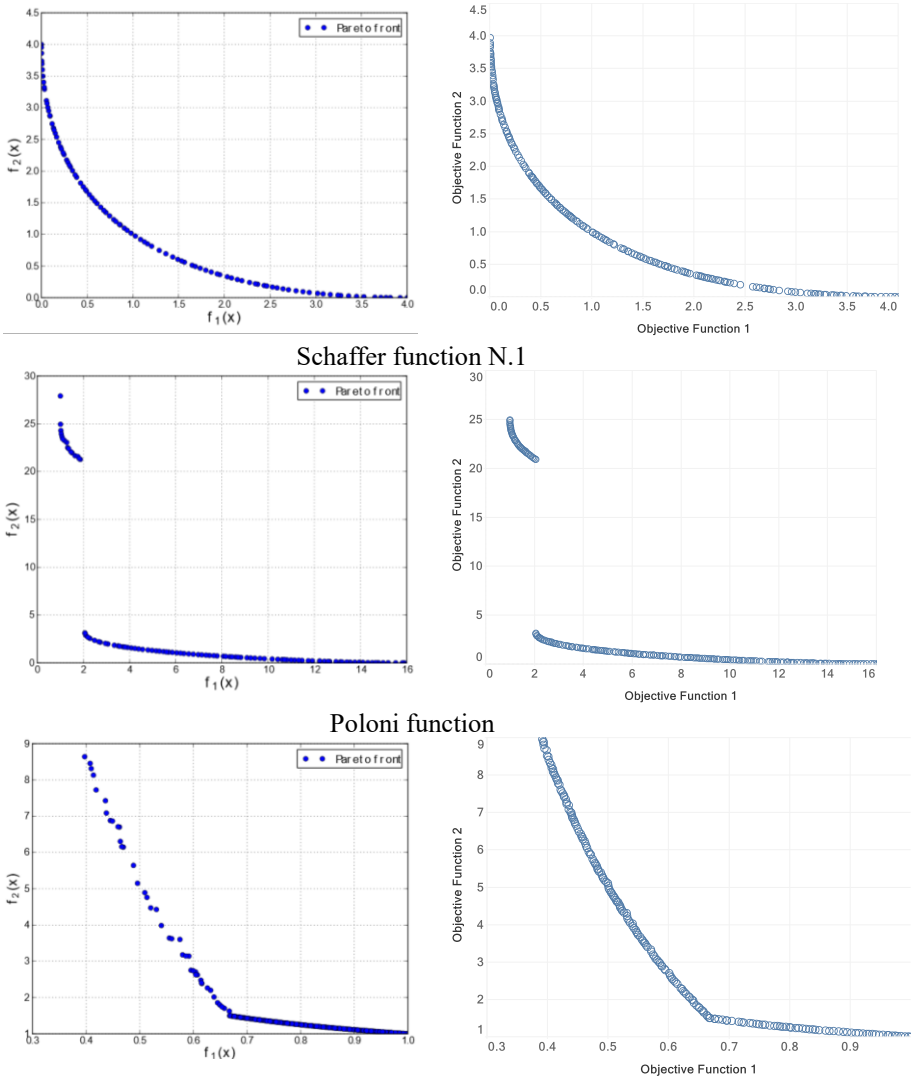

Fig. 11. Standard Pareto Fronts (Left) vs. the Pareto Fronts plotted by MOPSO (Right)

The multi-objective optimization test function in the first row in Fig. 11. is Schaffer function N.1 [11]. The curve plotted by MOPSO is a sunken curve and it is consistent with the shape of the standard Pareto Front. By comparing the key data points in the graph, all two curves pass through Point (0, 4), Point (1, 1) and Point (4, 0). And two other problems have also been satisfactorily solved. Therefore, it can be concluded that the MOPSO Solver can solve the multi-objective problems effectively and accurately.

5 Introduction

The purpose of the MOPSO development on Dynamo - a BIM-based Visual Programming platform – is to facilitate multiple-objective building design optimization. Among the building design objectives, building energy consumption is a major concern during the design process. In this case study, a specific AEC optimization task will be presented. There is an office building [12] whose standard level plane is shown in the lower part of Fig. 12. This paper chooses the typical office in the south of the building in order to find out the influence of the opening, depth and the window-wall rate of the office on the solar radiation heat gain of the room. As the upper part of Fig 12 shows, a simplified office model was built using Revit.

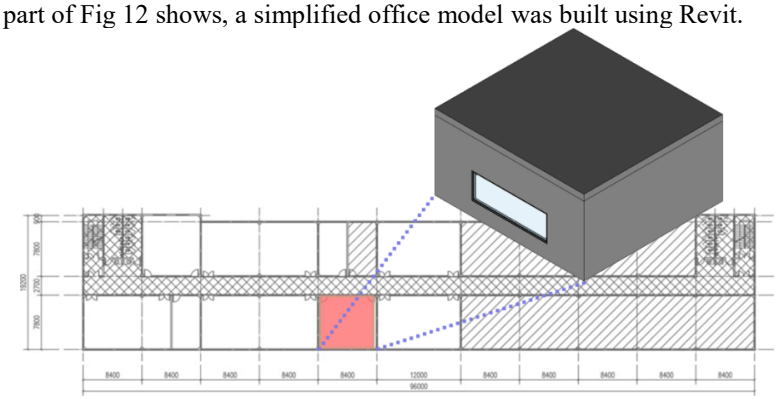

Fig. 12. A simplified building model as a case study for MOPSO

This paper calculates the radiation in Tianjin using the .epw weather file provided by the EnergyPlus official website [13]. The study only considers the radiation of the roof and the south façade. In summer and winter, the radiation of roof and south façade of a building in Tianjin is shown in Table 3.

Table 3 Solar radiation in Tianjin

Surface and Season	Radiation (KWh/m^2)
roof in summer	397.38
south façade in summer	136.61
roof in winter	165.25
south façade in winter	217.07

Let *a* be the width of the office, *b* be the depth of the office and *r* be the window-wall ratio. The heat gain (HG) from radiation in summer and winter can be calculated with formula (7) and (8):

$$HG_{summer} = 397.38 \times \alpha_o ab + 136.61 \times (\alpha_o ah - \alpha_o ahr + a_g ahr) \quad (7)$$

$$HG_{winter} = 165.25 \times \alpha_o ab + 212.07 \times (\alpha_o ah - \alpha_o ahr + a_g ahr) \quad (8)$$

where $\alpha_0 = 0.8$ is the absorptivity of the opaque enclosure structure, $\alpha_g = 0.7$ is the absorptivity of the transparent enclosure structure and $\mathrm{h} = 4.2m$ is the height of the office.

According to the "Standard for daylighting design of buildings" in China (GB 50033-2013), the required value of daylight factor C_{av} must be bigger than 3.0% and the window wall ratio must be smaller than 0.7. C_{av} can be described as formula (9):

$$C_{av} = \frac{A_c \tau \theta}{A_z (1 - \rho^2)} \tag{9}$$

where $\tau = 0.6$ is the total transmittance of window; $\rho = 0.5$ is the weighted average of the surface reflectance ratio of indoor materials; $\theta = 90^\circ$ is angle of vertical visible sky calculated from the center of the window. And A_c, A_z stand for the area of window and the area of the indoor surface which can be described as formula (10) and formula (11):

$$A_c = 4.2ar \tag{10}$$

$$A_z = 4.2a(1-r) + 4.2a + 4.2b \times 2 + 2ab \tag{11}$$

So, the multi-objective optimization problem can be modeled in Table 4.

Table 4 Information of the multi-objective function

Functions	Constraints	Search Domain
$Minimize = \begin{cases} HG_{summer} \\ -HG_{winter} \end{cases}$	$C_{av} = \frac{A_c \tau \theta}{A_z (1 - \rho^2)} \geq 3\%$	$0 \leq r \leq 0.7$ $6 \leq a \leq 8.4$ $7 \leq b \leq 8$

By using the MOPSO Solver, we can get the result shown in Fig. 13. With the help of the Pareto Front, architects can choose the building parameters to optimize their designs.

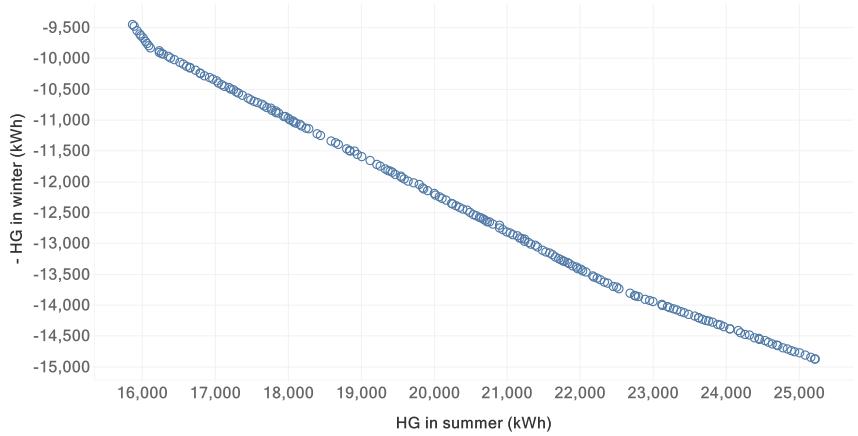

Fig. 13. Pareto Front of the study case

6 Conclusions

The MOPSO Solver developed in this research is a multi-objective optimization engine using the MOPSO algorithm. It works with Autodesk Revit and its visual programming platform Dynamo. The tool can assist designers to improve their design optimization while using Building Information Models. When optimizing the standard test functions, the MOPSO Solver achieves correct results, and when optimizing the simplified building design test case, the program runs smoothly and the result is reasonable. These tests reveal the great potentials of the MOPSO for applications in more complex design optimization problems in the future, after additional building design projects are tested and improvements of the system are made. The contribution of this research lies in the advancement and demonstration of multi-objective optimization for architectural design utilizing a parametric BIM environment. Currently, MOPSO Solver can run in a computer environment with Dynamo Sandbox and MATLAB R2018a, and a new version of the tool that can run in the more general Revit/Dynamo environment is being developed as part of a more comprehensive BIM-based optimization tool development of future work.

Acknowledgement

This research is supported by the National Natural Science Foundation of China (Grant No. 51628803).

References

1. Machairas V, Tsangrassoulis A, Axarli K. Algorithms for optimization of building design: A review. Renewable & Sustainable Energy Reviews, 2014, 31(2):101-112.
2. Pilla L D, Desogus G, Mura S, et al. Optimizing the distribution of Italian building energy retrofit incentives with Linear Programming. Energy & Buildings, 2016, 112:21~27.
3. Yang M D, Lin M D, Lin Y H, et al. Multi-objective optimization design of green building envelope material using a non-dominated sorting genetic algorithm. Applied Thermal Engineering, 2016.
4. Madias E N D, Kontaxis P A, Topalis F V. Application of multi-objective genetic algorithms to interior lighting optimization. Energy & Buildings, 2016, 125:66~74.
5. Tezer T, Yaman R, Yaman G. Evaluation of approaches used for optimization of stand-alone hybrid renewable energy systems. Renewable & Sustainable Energy Reviews, 2017, 73:840-853.
6. Delgarm N, Sajadi B, Delgarm S, et al. Multi-objective optimization of building energy performance and indoor thermal comfort: A new method using artificial bee colony (ABC). Energy & Buildings, 2016, 131(11):42-53.
7. Evins R. A review of computational optimisation methods applied to sustainable building design. Renewable & Sustainable Energy Reviews, 2013, 22(8):230~245.
8. Rahmani Asl, M., Zarrinmehr, S., Bergin, M., and Yan, W. , BPOpt: A framework for BIM-based performance optimization, Energy and Buildings, Elsevier, 2015.

9. Kennedy J, Eberhart R. Particle swarm optimization. IEEE International Conference on Neural Networks, 1995. Proceedings. IEEE, 2002(4):1942~1948.
10. Rapone G, Saro O. Optimisation of curtain wall façades for office buildings by means of PSO algorithm. Energy & Buildings, 2012, 45(45):189~196.
11. Deb, Kalyanmoy (2002) Multiobjective optimization using evolutionary algorithms (Repr. ed.). Chichester [u.a.]: Wiley. ISBN 0-471-87339-X.
12. Rahmani Asl, M., Stoupine, A., Zarrinmehr, S., and Yan, W., Optimo: A BIM-based Multi-Objective Optimization Tool Utilizing Visual Programing for High Performance Building Design, Proceedings of the Conference of Education and Research in Computer Aided Architectural Design in Europe (eCAADe), 2015, Vienna, Austria.
13. https://energyplus.net/weather