Jurnal Teknologi

Full paper

Power Consumption Optimization of a Building Using Multiobjective Particle Swarm Optimization

Ahmad Faiz Ab Rahman[a], Hazlina Selamat[a*], Fatimah Sham Ismail[a], Nurulaqilla Khamis[b]

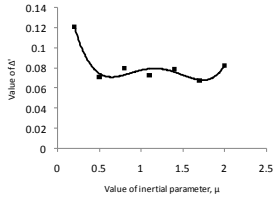
[a]*Centre for Artificial Intelligence & Robotics (CAIRO), Faculty of Electrical Engineering, Universiti Teknologi Malaysia, 81310 UTM Johor Bahru, Johor, Malaysia*
[b]*Malaysia-Japan International Institute of Technology, Universiti Teknologi Malaysia, Jalan Semarak, 51400 UTM KL, Kuala Lumpur, Malaysia*

*Corresponding author: hazlina@fke.utm.my

Article history

Received :15 June 2014
Received in revised form :
15 September 2014
Accepted :15 October 2014

Graphical abstract



Abstract

This paper discusses the development of a building energy optimization algorithm by using multiobjective Particle Swarm Optimization for a building. Particle Swarm Optimization is a well known algorithm that is proven to be effective in many complex optimization problems. Multiobjective PSO is developed by utilizing non-dominated sorting algorithm in tandem with majority-based selection algorithm. The optimizer is written by using MATLAB alongside its GUI interface. Results are then analyzed by using the Binh and Korn benchmark test and natural distance performance metrics. From the results, the optimizer is capable to minimize up to 42 percent of energy consumption and lowering the electricity bills up to 43 percent, while still maintaining comfort at more than 95 percent as well. With this, building owner can save energy with a low-cost and simple solution.

Keywords: Multiobjective optimization; particle swarm optimization; building energy optimization; pareto front; benchmark test; performance metrics

1.0 INTRODUCTION

The current advancement of technology has allowed the installation of different kinds on electrical appliances designed to improve the quality of our everyday lives. Among them are electrical appliances developed in order to maintain the comfort inside our homes and buildings, such as lamps, fans, air-conditioning (A/C) systems and Heating, Ventilation and Air Conditioning (HVAC) systems. However, HVAC system is one of the highest power consumers, causing a significant increase in power consumption. In addition, the increasing number of buildings, offices, factories, etc would mean that these systems will be widely installed, causing the electricity bills to rise as well as giving more loads to the power supplier in the future.

In order to reduce the power load caused by HVAC systems, many solutions have been desired by past researchers to solve the problem. In this case, perhaps the simplest way to reduce the load of these electrical systems is by operating these electrical devices efficiently, where there is no excessive energy wasted. Even though many technologies have been developed in the past years in order to assist reduction of load inside a building, such as smart building system [1-4], or intelligent controllers [6, 8, 11], these system are known to be expensive and hard to implement due to the diverse nature of the building itself. To solve these problems, simple energy optimization scheme need to be developed with low cost and complexity. One of the possible ways is by using a system which do not need any pre-installation into the building itself; it can just assist building owners to save energy by giving suggestions or tips to reduce energy by setting the already installed system in more effective way.

To find the best settings for our needs, optimization algorithm is needed which can successfully find the best sets of suggestions to achieve its objective. Optimization is a technique where the algorithm selecting the best solution from a set of candidates in accordance to a certain criteria [12-14]. In most cases, the optimization algorithm will choose the solution where criteria such as maximum or minimum value are achieved, where it can be implemented for better power saving without compromising comfort [12-14].

In this research, new software is developed which consists of a Graphical User Interface (GUI) and an optimizer algorithm. This software system is developed using multiobjective Particle Swarm Optimization (PSO) algorithm as its basis. PSO is a well-known metaheuristic algorithm that has been proven to be effective in solving complex optimization problems [12-14].

In the next section, the discussion will be focused on the basic concept regarding the model that expresses the relationships between environmental variables with power consumption and comfort level, where the optimizer would need to operate with, as well as the basic ideas regarding optimization. Subsequently, the discussion will continue on the methodology of this research before continuing on results, discussion and final conclusion can be made on this research.

2.0 ENVIRONMENTAL VARIABLES, POWER AND COMFORT MODEL

2.1 Environmental Variables and Objective Variables

For the optimizer to be able to help solving the problem, investigations on the characteristics of the indoor environment have to be done first. It is clear that this is a problem with two objectives. The first one would be minimization of power consumption inside the building, while the second one would be the maximization of building occupant comfort. These two objectives will ensure that the optimizer will achieve the desired objective of this research.

At this point it is clear that this is a multiobjective optimization problem which both power consumption, denoted as P, and comfort level, denoted as C, will be chosen as an indicator, better known as objective variables. For both objective variables to act as an indicator, these variables must be affected by other variables which tell conditions of the environments inside the buildings. These variables, commonly known as environmental variables, are groups of variables that tell about the conditions inside the buildings. This includes the temperatures, brightness, air quality, etc.

2.2 Comfort Model

For the evaluation of occupants comfort, C, most past researchers agreed that comfort evaluation can be divided into three main components, which are thermal comfort, visual comfort and indoor air quality [1-4]. Each component has certain variable which is chosen as the indicator of the comfort. However, for the sake of simplicity, this research only focuses on two components, that is the thermal comfort and visual comfort, since indoor air quality (IAQ) is considered to be hard to control in a closed building environments without any additional device installation that would add more costs.

Thermal comfort, denoted as C_T, as its name suggests, is comfort measurements based on the degree of hotness or coldness inside a building. Even though there are better indicators that precisely point out the level of thermal comfort by past researchers [5-8], indoor air temperature, denoted as T, in degree celcius (°C) as an indicator for thermal comfort is chosen for simplifying the thermal comfort measurement [1-4]. Visual comfort, denoted as C_L, on the other hand, deals with the comfort measurements based on the brightness or darkness inside a building [9-10]. For this component, illumination level, denoted as L, in lux, is used as the indicator [1-4][9-10]. Both air temperature and illumination are the environmental variables considered in this research.

For comfort evaluation, the indicator used here is considered as a unitless index, where the value is ranging from 0 (most uncomfortable) to 1 (most comfortable) [1-4]. The computation of the comfort indicator is given by Equations (1) to (5):

$$C = C_T + C_L \ where \ 0 \leq C \leq 1 \tag{1}$$

$$C = \delta_T \left(1 - \left(\frac{e_T}{T_{set}}\right)^2\right) + \delta_L \left(1 - \left(\frac{e_T}{T_{set}}\right)^2\right) \tag{2}$$

$$\delta_T + \delta_L = 1 \tag{3}$$

$$e_T = T - T_{set} \tag{4}$$

$$e_L = L - L_{set} \tag{5}$$

C = Comfort level
C_T = Thermal comfort
C_L = Visual comfort
δ_T = Weighting for thermal comfort
δ_L = Weighting for visual comfort
T_{set} = Set temperature where the building occupant set as most comfortable
L_{set} = Set illumination where the building occupant set as most comfortable
e_T = Error between current temperature and set temperature
e_L = Error between current illumination level and set illumination level
T = Current indoor air temperature
L = Current illumination level

From the equations above, both weightings for thermal and visual comfort, δ_T and δ_L were set to be 0.5 in order to treat both types of comfort as equally important. In addition, the desired indoor air temperature where the occupants fell as most comfortable, T_{set}, is fixed at 24°C, and the desired illumination level where the occupants fell as most comfortable, L_{set}, is fixed at 300lux. We also fix the value of outdoor air temperature, T_{out}, to 32°C. Since both T_{set} and L_{set} are highly dependent on user preferences, the comfort level is a value that is highly dependent on personal preferences and may be different from one person to another.

2.3 Power Consumption Model

Apart from comfort, the environmental variables also affect another objective variable, which is the power consumption, P. For power consumption computation, the overall power consumption we can found by adding the power consumption due to air conditioning unit or HVAC systems, P_{AC} [11], which is related to indoor air temperature, T and power consumption caused by lighting systems, P_L, given by Equations (6) to (9):

$$P = P_{AC} + P_L \tag{6}$$

$$P = q_{AC} \frac{\rho v c_{air} \Delta T}{COP \times t} + q_{lamps} \frac{LA}{\eta_{lamps}} \tag{7}$$

$$COP = \frac{T}{T_{out} - T} \tag{8}$$

$$\Delta T = T_{out} - T \tag{9}$$

P = Power consumption
P_{AC} = Power consumption due to air conditioning or HVAC units
P_L = Power consumption due to lighting units
q_{AC} = Quantity of air conditioning or HVAC units
q_{lamps} = Quantity of lighting units
ρ = Air density, fixed at 1.184kgm^{-3} at 25°C
V = Volume of air inside the building
c_{air} = Specific heat density of air, fixed at 1012Jkg^{-1}°C^{-1} at 25°C

ΔT = Changes in indoor air temperature at t time interval
t = Time interval, in seconds
COP = Coefficient of performance
T_{out} = Outdoor air temperature
A = Area of illuminated surface
η_{lamps} = Luminous efficacy of the lamps, fixed value depending on the lamp type
T = Current indoor air temperature
L = Current illumination level

2.4 Multiobjective Particle Swarm Optimizer

Optimization is a technique where a selection of best solution is done among solution candidates in accordance to the criteria being tested. Designed to imitate the social behavior of animals such as fishes and birds, the particle swarm optimizer is widely adopted in various engineering applications due to its simple but still powerful as an optimizer [12-14].

Basically, the optimization algorithm starts by producing a set of randomized values, which is known as solution set, or sometimes known as particles. This candidate solution contains values that are constrained in a certain range of value known as solution space. For each solution or particles, they will be given two values known as particle position and particle velocity, where the particle or solution will move in accordance to these two values. The candidate solution or particle position and velocity value will be adjusted in small increments or decrements for each iteration, in accordance to the Equations (10) and (11) [12-14].

$$v_{(i)}(t+1) = \mu v_{(i)}(t) + \alpha_1 r_1 \left(p_{best(i)}(t) - l_{(i)}(t)\right) + \alpha_2 r_2 \left(g_{best(i)}(t) - l_{(i)}(t)\right) \quad (10)$$

$$x_{(i)}(t+1) = x_{(i)}(t) + v_{(i)}(t+1) \quad (11)$$

$v_{(i)}$ = Velocity of i-th particle
$v_{(i)}$ = Position of i-th particle
$p_{best(i)}$ = Particle best position for i-th particle
$g_{best(i)}$ = Particle best position
t = Number of iteration
α_1, α_2 = Acceleration parameters
μ = Inertia parameter

A function called the 'objective' or 'cost' function is used for evaluation of the quality of the candidate solutions in the solution space mentioned earlier. In most cases, the optimizer usually targets the maximum or minimum value of cost function as the desired solution. In this case, since there are two cost functions to be considered, the concept of Pareto front is introduced [15-16]. Comfort function in Equation (2) is set as first cost function, while the power consumption function in Equation (7) will act as the second cost function. This is important because in multiobjective optimization problems, it is quite seldom to encounter situations where the objectives do not conflict between each other; most of the time, one objective is sacrificed if improvements on the other objective to be made [15-16]. To solve these problems, the solution will only be updated if the new solution is better than the old ones in at least one objective while as good as old solution in other objectives. After a few iterations, the group of best candidate solutions, which there is no other solution is better will be produced, producing a curve where they will be considered as Pareto optimal fronts. This is the solution that will be chosen as final result of the optimization.

3.0 METHODOLOGY

The Multiobjective Particle Swarm Optimizer (MOPSO) was developed by using MATLAB. There are two main components of the program developed here; the first component is the Graphical User Interface (GUI) program as shown in Figure 1, while the other one is the optimizer code itself.

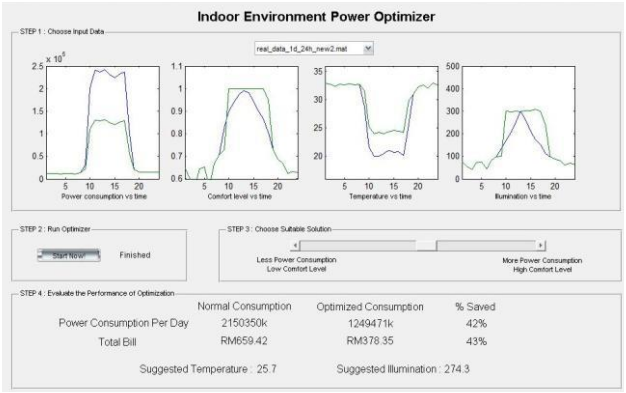


Figure 1 Snapshot of GUI program for optimizer

3.1 Multiobjective Majority Non-Dominated Sorting Algorithm

There are several approaches for developing a multiobjective optimization algorithm is available, such as no-preference methods, priori method, posteriori method and interactive methods. However, posteriori method is chosen in this research due to two main reasons. First, posteriori method gives a whole range of possible best solution, known as Pareto optimal fronts, leaving the human user will make a decision on which solution among the Pareto optimal fronts is preferable, thus giving flexibility to the user to make choices and comparing between the ranges of results produces by the fronts. The second reason of this approach is that many evolutionary algorithm such as the Non-Dominated Sorting Algorithm II (NSGA-II) and Strength Pareto Evolutionary Algorithm 2 (SPEA-2) is classified under this category, where it focused on the ranking of the Pareto fronts. Because of this reason, it is reasonable to develop a multiobjective PSO algorithm based on posteriori approach, due to the nature of the evolutionary algorithm, which usually produces a set of solutions, allowing computation of the entire Pareto front.

For this research, the multiobjective optimizer is developed utilizing the concept of non-dominated sorting algorithm and majority-based selection algorithm. Multiobjective majority comparator is developed with the idea of 'voting'. While comparing in between two candidate solutions by using two cost functions, the better solution will be given a 'vote' for each cost function. A solution that is getting at least a vote will be considered non-dominated by the other solution. Solutions that do not getting any vote for each cost function evaluation are considered to be dominated, and will not be chosen by the comparator. This will solve the problem on how to evaluate solutions based on both cost functions at the same time.

On the other hand, the non-dominated sorting algorithm is an algorithm where the optimizer evaluates each solution and stores the best candidate inside a repository. For every iteration, the solution in the repository will be introduced with newer solution, compared among each other and discarding any dominated solution so that only the best solution prevails. The best solution will later be used to produce Pareto fronts, where it shows the points of all best solution inside the repository.

Aside from all the modifications mentioned before, the other parts of the PSO algorithm are still similar with the original ones, which are shown in Figure 2. The PSO algorithm itself was developed to produce 1000 candidate solutions or particles in a two-dimensional solution space of two environmental variables. The particles then will be moved across solution space by using velocity and position update equation in Equations (10) and (11) and checked by using both cost functions of Equation (2) and Equation (7) by using multiobjective majority comparator algorithm. Multiobjective majority comparator algorithm will choose the better solution by comparing which solution is better, whether in at least one objective, or better in both objective. The better ones for that particular particle will be set as new pbest values. All pbest values will be added into a repository where a non-dominated sorting algorithm will be done among every solution inside the repository. This process will leave only the best solution available inside the repository. This process is repeated until the number of iteration is reached. Final results stored in the repository is collectively known as Pareto optimal solution and plotted as a Pareto front.

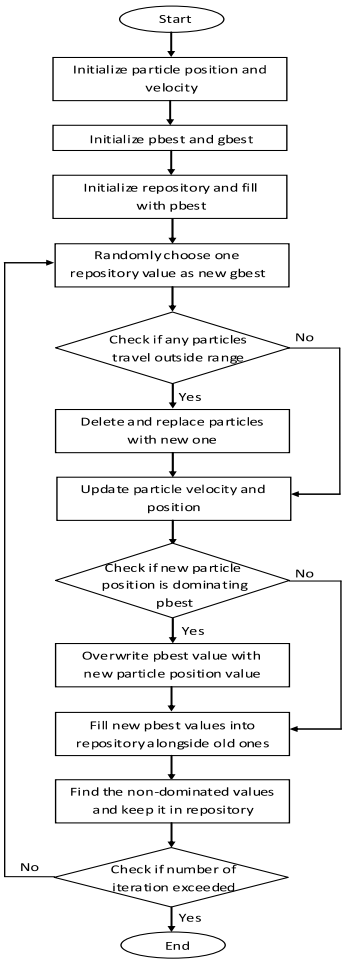


Figure 2 Flow chart of MOPSO optimization algorithm

3.2 Benchmark Test and Performance Metrics

The analysis of Pareto front is the next step of this research. This step is important because the quality of Pareto front produced by the optimizer may sometimes hard to evaluate visually. In order to start the analysis, two requirements are needed. The first would be the benchmark test, which is widely used in optimization testing [17-18]. The benchmark test is done by replacing cost function with benchmark test equation, which is designed to be a complex mathematical problem for optimizer to solve [17-18].

In this research Binh and Korn test is used due to its similarity between the number of variables inside the test function involved and the shape of Pareto front produced with the case study of this research. The Binh and Korn benchmark test function is shown below in Equations (12) and (13), where the optimizer needs to minimize them in order to produce better, minimal result:

$$f_1(x,y) = 4x^2 + 4y^2 \tag{12}$$

$$f_2(x,y) = (x-5)^2 + (y-5)^2 \tag{13}$$

where the search domain are $0 \leq x \leq 5$, $0 \leq y \leq 3$ where the constraints are in Equation (14)–Equation (15)

$$(x-5)^2 + y^2 \leq 25 \tag{14}$$

$$(x-8)^2 + (y-3)^2 \geq 7.7 \tag{15}$$

Another requirement for analyzing Pareto fronts would be the performance metrics of the fronts. Natural distance performance metrics, Δ' test is chosen as the performance metrics as it is the most compatible performance metrics for the analysis of the Pareto fronts in this research [18]. The equation of the performance metrics is shown in Equation (16), where the lower values indicate the better Pareto fronts:

$$\Delta'(S) = \sum_{i=1}^{|s|-1} \frac{|d_i - \bar{d}|}{|s| - 1} \tag{16}$$

d_i = Euclidean distance between pareto optimal solution
$\bar{d}$ = Average Euclidean distance
$|S|$ = Number of pareto optimal solution

From the values of performance metrics calculated in the analysis mentioned before, a conclusion can be made on what are the best values of PSO parameters that will produce the best Pareto optimal solution. From this analysis alone, the best values for number of iteration done, acceleration parameters, α_1 and α_2, as well as inertial parameter, μ, can be deducted and proven.

■4.0 RESULTS AND DISCUSSIONS

After the MOPSO algorithm is completed and operating well, an analysis to deduce the best values for parameters including number of iterations need to be done, values of acceleration parameters, α_1 and α_2, as well as inertial parameter, μ, can be started.

The first analysis is done on numbers of iterations need to be done, where all other parameters are set as constant, that is with 1000 particles, acceleration parameters, α_1 and α_2 of 1.0, as well as inertial parameter, μ of 0.5. From the graph in Figure 3, the best configuration for number of iterations needed to produce a

good result is 500 iterations, as minimal values of Δ' are better. The same step is repeated on analysis on both acceleration parameters analysis, where iteration value is fixed on 500 iterations while inertial value is at 0.5, which shown 1.0 is the best value, as shown in Figure 4. Lastly, it is clear from the graph in Figure 5 that the best value for inertial parameter as both iteration values and acceleration parameters, α_1 and α_2, are fixed at 500 and 1.0 is 0.5.

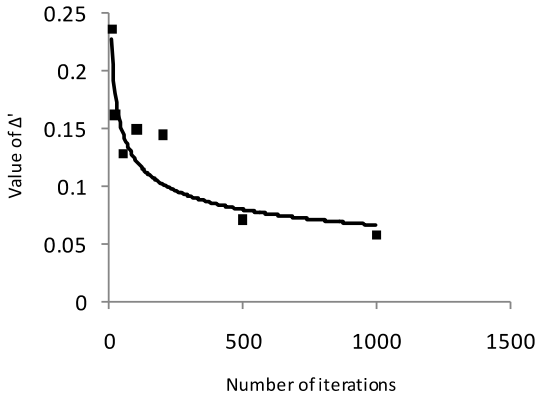


Figure 3 Graph of performance metrics Δ′ values versus number of iterations

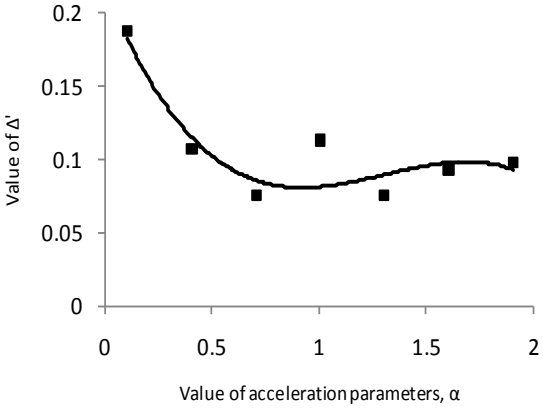


Figure 4 Graph of performance metrics Δ′ values versus value of acceleration parameters, α

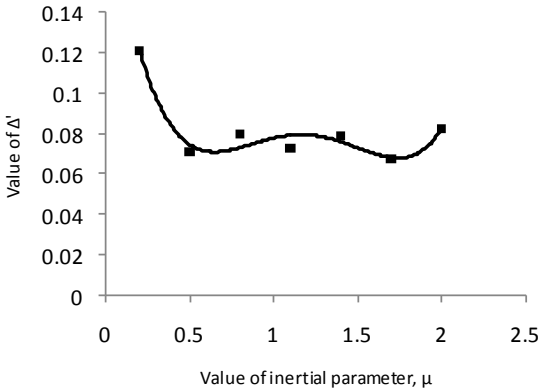


Figure 5 Graph of performance metrics Δ′ values versus value of inertial parameter, μ

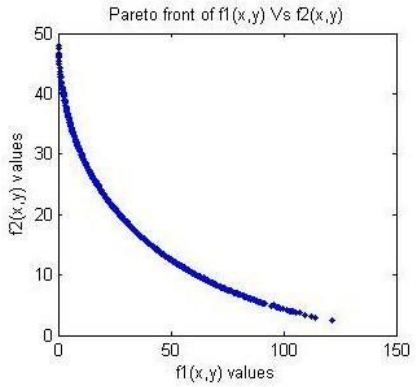


Figure 6 Pareto Front of power consumption and comfort optimization problem produced by MOPSO algorithm

Table 1 Comparison between normal and optimized power consumption in KWH and bill costs

	Normal Consumption	Optimized Consumption	Percentage Saved
Power Consumption (kWh)	2150350	1279471	42%
Total Bill (RM)	659.42	378.35	43%

From the conclusion that has been made in benchmark analysis, we use the same configurations in order to implement the MOPSO algorithm with the energy and comfort optimization problem by changing the cost functions and running it once again by using GUI program. By using energy comfort data collected for P02 building in Electrical Engineering Faculty, Universiti Teknologi Malaysia, a Pareto front is produced, which is shown in Figure 6. The optimizer also managed to give suggestions which would reduce energy consumption in kWh, by up to 42 percent and reduce bill costs by 43 percent, as shown in Table 1.

Other important results need to be highlighted can be seen from the Pareto fronts shown in Figure 6. As mentioned before, a user can freely make decisions among the sets of results derived from the Pareto front, which results will satisfy the needs. As an example, if a user chooses power minimization around 42% as mentioned in Table I, the optimizer still capable to produce results, which comfort value that is more than 95%. These results proved that the algorithm has successfully gives solution that not only manages to minimize considerable level of power consumption, but also maintaining comfort level at a very acceptable level as well.

5.0 CONCLUSIONS

From this research, it can be concluded that multiobjective particle swarm optimization (MOPSO) algorithm can assist building owners to reduce power consumption while still maintaining comfort in a simple but effective approach that has been highlighted in this paper. However, more improvements need to be done, such as using a better, more accurate comfort evaluation model need to be done because the model currently use is simplified and easily affected by different personal preferences which may affect its efficiency. In addition, the effects of ever changing weather are also not considered in this research, which makes the power and comfort evaluation model to be more complex.

Acknowledgement

The author would like to give acknowledgement to the Ministry of Education. This work is funded by Research University Grant VOT 00G19 of Universiti Teknologi Malaysia.

References

[1] Yang, R. and Wang, L. 2013. Development of Multi-agent System for Building Energy and Comfort Management Based on Occupant Behaviours. *Energy and Buildings*. 56: 1–7.

[2] Wang, Z., Wang, L., Dounis, A. I. and Yang, R. 2012. Multi-agent Control System with Information Fusion Based Comfort Model for Smart Buildings. *Applied Energy*. 99: 247–254

[3] Wang, Z., Dounis, A. I., Wang, L. and Yang, R. 2011. An Information Fusion Based Multi-agent Control System for Indoor Energy and Comfort Management in Smart and Green Buildings. *Power and Energy Society General Meeting*. 1–8.

[4] Wang, Z., Yang, R. and Wang, L. 2010. Multi-agent Control System with Intelligent Optimization for Smart and Energy-efficient Buildings. In *IECON 2010-36th Annual Conference on IEEE Industrial Electronics Society*. 1144–1149

[5] Griego, D., Krarti, M. and Hernández-Guerrero, A. 2012. Optimization of Energy Efficiency and Thermal Comfort Measures for Residential Buildings in Salamanca, Mexico. *Energy and buildings*. 4: 540–549.

[6] Ferreira, P. M., Ruano, A. E., Silva, S. and Conceição, E. Z. E. 2012. Neural Networks Based Predictive Control for Thermal Comfort and Energy Savings in Public Buildings. *Energy and Buildings*. 55: 238–251.

[7] Daum, D., Haldi, F. and Morel, N. 2011. A Personalized Measure of Thermal Comfort for Building Controls. *Building and Environment*. 46(1): 3–11.

[8] Freire, R. Z., Oliveira, G. H. and Mendes, N. 2008. Predictive Controllers for Thermal Comfort Optimization and Energy Savings. *Energy and buildings*. 40(7): 1353–1365.

[9] Yun, G. Y., Kong, H. J., Kim, H. and Kim, J. T. 2012. A Field Survey of Visual Comfort and Lighting Energy Consumption in Open Plan Offices. *Energy and Buildings*. 46: 146–151.

[10] Turns, S. R. 2006. *Thermodynamics: Concepts and Applications*. Cambridge University Press.

[11] Kolokotsa, D., Tsiavos, D., Stavrakakis, G. S., Kalaitzakis, K. and Antonidakis, E. 2001. Advanced Fuzzy Logic Controllers Design and Evaluation for Buildings' Occupants Thermal–Visual Comfort and Indoor Air Quality Satisfaction. *Energy and buildings*. 33(6): 531–543.

[12] Poli, R., Kennedy, J. and Blackwell, T. 2007. Particle Swarm Optimization. *Swarm Intelligence*. 1(1): 33–57.

[13] Shi, Y. and Eberhart, R. C. 1999. Empirical Study of Particle Swarm Optimization. In *Evolutionary Computation, 1999. CEC 99*. 3: 1–6.

[14] Bratton, D. and Kennedy, J. 2007. Defining a Standard for Particle Swarm Optimization. In *Swarm Intelligence Symposium. SIS 2007. IEEE*. 120–127.

[15] Agrawal, S., Dashora, Y., Tiwari, M. K. and Son, Y. J. 2008. Interactive Particle Swarm: A Pareto-adaptive Metaheuristic to Multiobjective Optimization. *Systems, Man and Cybernetics, Part A: Systems and Humans, IEEE Transactions on*. 38(2): 258–277.

[16] Zhongkai, L. and Zhencai, Z. 2010. DSMOPSO: A Distance Sorting Based Multiobjective Particle Swarm Optimization Algorithm. In *Natural Computation (ICNC), 2010 Sixth International IEEE Conference*. 5: 2749–2753.

[17] Li, X. 2003. A Non-dominated Sorting Particle Swarm Optimizer for Multiobjective Optimization. In *Genetic and Evolutionary Computation—GECCO 2003*. 37–48). Springer Berlin Heidelberg.

[18] Okabe, T., Jin, Y. and Sendhoff, B. 2003. A Critical Survey of Performance Indices for Multi-objective Optimisation. In *Evolutionary Computation, 2003. CEC'03. The 2003 Congress*. 2: 878–885.