5

Conflicting Multi-Objective Compatible Optimization Control

Lihong Xu[1], Qingsong Hu[2], Haigen Hu[1] and Erik Goodman[3]
[1]*Tongji University, 1239 Siping Road, Shanghai 200092,*
[2]*Shanghai Ocean University, 999 Huchenghuan Road, Shanghai 201306,*
[3]*Michigan State University, East Lansing, MI 48824,*
[1,2]*P.R.China*
[3]*USA*

1. Introduction

It is clear that there exist many practical control problems in which the consideration of multiple objectives is typically required, and these objectives may conflict with each other. For example, in many practical control systems, control error often conflicts with energy consumption. In the past ten years, we have been studying the greenhouse environment control problem and have gained a considerable understanding of greenhouse dynamics. In a greenhouse, we must keep the temperature and humidity in certain range that is suitable for the plants. However, we are simultaneously required to minimize energy consumption to reduce the cost. The control means include ventilation, heating and spraying, of which heating and spraying are high-energy-consumption methods. In winter, we can improve the temperature by heating and decrease the humidity by heating and ventilating. With the traditional control strategy, we could maintain the temperature and humidity at a very precise point, but the high energy consumption and expensive cost of this strategy would make the greenhouse unprofitable, which implies that this control strategy would not to be chosen by any users. This type of problem is also widely found in industrial control.

There have existed two main traditional methods to solve the above-mentioned multi-objective problem. One is the trade-off weight method (Masaaki, 1997), which translates this multi-objective problem into a single-objective one by adding a set of trade-off weights (Masaaki, 1997; Rangan & Poolla, 1997; Eisenhart, 2003). The major advantage of this method is that the translated single-objective control problem is very easy to deal with, but the disadvantage is that the control result will be strongly associated with the trade-off weights chosen, and the controlled objectives may not be satisfactory if we provide bad weights. In addition, the selection of weights is very difficult for users and engineers in the practical situation. Another approach is the constraints method, which optimizes the most important control objective and translates the others into system constraints (Scherer, 1995; Scherer et al, 1997; Sznaier et al, 2000). The advantage of this approach is to satisfy all controlled objectives through constraints; however, the constraint bounds are very difficult for users or control engineers to determine suitably in a practical problem. Bounds that are too tight may bar the existence of a feasible solution for the optimization problem, while too loose bounds may make the optimization problem lose practical significance.

Source: New Achievements in Evolutionary Computation, Book edited by: Peter Korosec,
ISBN 978-953-307-053-7, pp. 318, February 2010, INTECH, Croatia, downloaded from SCIYO.COM

Since the traditional multi-objective control method cannot ensure the existence of a feasible controller in advance, we have adopted a multi-objective coordinated control system in the greenhouse. When these objectives conflict with each other, it is impractical to fix all the objectives at some given optimal "points". To ensure the existence of a feasible controller, we are willing to "back off" on our desire that all the controlled objectives be precisely at their optimal values, relaxing these "point" controlled objectives to some suboptimal "intervals" or "regions," more generally—we call them "compatible objective regions". For example, in the greenhouse, we regulate the temperature objective to be in the interval 24-30°C instead of exactly at 28°C, and the humidity objective to 60%-85% instead of exactly 70%. According to the experts, this greenhouse environment is also very suitable for the plants. Then we design a controller by optimizing the energy consumption objective. This compatible control system can obtain better economic benefit than before and has gained considerable attention from users (Wu, 2003).

Based on the successful application, we have generalized a common compatible control theory framework (Xu et al, 2006) from the practical experience obtained with the compatible objective region used in the greenhouse. We call this method "multi-objective compatible control (MOCC)".

Control of urban traffic flow is also a typical complex multi-objective control problem, evidencing conflict between the main roads and support roads in the saturation state. Intensive research has focused on this problem to improve traffic management. MOCC has also been applied to this traffic flow control problem, including a Ph.D. dissertation written on the subject (Chen et al, 2008).

Considering a general discrete-time system, the multi-objective control problem can be abstracted as

$$x(k+1) = f(x(k), u(k), k) \tag{1}$$

where k denotes the time step, $x(k) \in R^n$ denotes the state, and $u(k) \in R^m$ is the control variable. The state and control variables are required to fulfill the following constraints

$$u(k) \in X \ , \ u(k) \in U \tag{2}$$

where X and U are subsets of R^n and R^m, respectively, containing the origin as an interior point. Then the multi-objective control problem considered here consists in minimizing, at any time step k ,

$$\min_{x(k)\in X, u(k)\in U} J_i\{(k+1)\} = h_i(x(k), x(k-1), ..., x(0), u(k), u(k-1), ..., u(0)) \tag{3}$$

$$i = 1,2,...,n, k = 1,2,...$$

subject to the system dynamics (1), and

$$x(k-j) \in X, j = 0,1,...,k \ , \ u(k-j) \in U, j = 0,1,...,k \tag{4}$$

It is difficult to obtain the Pareto solutions by traditional optimization methods. A multi-objective evolutionary algorithm (MOEA) is a robust search and optimization methodology that is able to cope with multiple objectives in parallel without translating the multiple objectives into one (see (Fleming & Purshouse, 2002; Goldberg, 1989), and among MOEA algorithms, especially when used for problems with only two objectives, NSGA-II performs

relatively well in both convergence and computing speed, see (Deb et al, 2002; Jensen, 2003)). It permits a remarkable level of flexibility with regard to performance assessment and design specification.
This paper is organized as follows. The second section is the description of two-layer MOCC with the precise model. The third section is the one-layer MOCC combining offline and online parts. A non-even spread reflecting the preference of the user is proposed in this section. The fourth section is the MOCC application to greenhouse environment control. The fifth section is the conclusion.

2. Multi-objective compatible control strategy with a precise model

In this section, we propose a two-layer MOCC framework suitable for a problem with a precise model.

2.1 System description and two-layer MOCC strategy

We first abstract the theoretical multi-objective control problem. Here the controlled system model can be described as follows:

$$x(k+1) = f(x(k), u(k), k) \tag{5}$$

$$y(k) = Cx(k) \tag{6}$$

where $x(k) \in R^n$ denotes the plant states, $y(k) \in R^n$ and $u(k) \in R^m$ are the control outputs and inputs, subject to constraints $\| u(k) \|_\infty \le a$. The aim of control is to make $y(k)|_{k\to\infty} \to y^*$. Taking two objectives as an example, we aim to design controller

$$\pi^p(x(0)) = \{u(0), u(1), ..., u(p-1)\}$$

and to minimize two conflicting objectives: control error, h_1, and control energy consumption h_2, defined as:

$$\begin{aligned} h_1 &= \sum_{k=1}^{p} (y(k) - y^*)^T (y(k) - y^*) \\ h_2 &= \sum_{k=0}^{p-1} u(k)^T u(k) \end{aligned} \tag{7}$$

Then the multi-objective control problem (5)-(7) can be translated into the following multi-objective optimization problem:

$$\min h_1 = \sum_{k=1}^{p} (y(k) - y^*)^T (y(k) - y^*) \tag{8}$$

$$\min h_2 = \sum_{k=0}^{p-1} u(k)^T u(k) \tag{9}$$

$s.t.$ 1. $\pi^p(x(0)) = \{u(0), u(1), ..., u(p-1)\}$

2. $\| u(k) \|_\infty \le a$
3. $h_1 \le h_1^L$
4. $h_2 \le h_2^L$

In a practical control system, control error and control energy consumption always lie within an acceptable range; here we denote by h_1^L and h_2^L the maximum acceptable values--which are called the practical objective constraint condition L. Note that h_1^L and h_2^L are only the worst acceptable values and not our control objectives. For example, in a greenhouse, the worst acceptable values h_1^L and h_2^L only ensure that the plants survive, but not that they flourish; our aim is to construct a suitable environment for the plants to grow productively, and not only one in which they can survive.

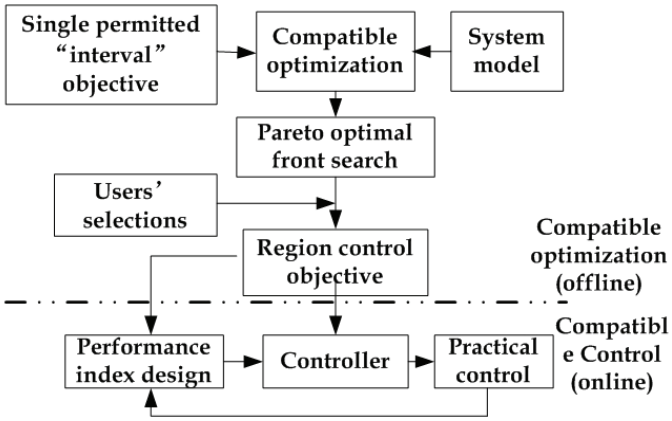


Fig. 1. Two-layer compatible control framework

Next we describe what is meant by a compatible control framework. If the model is precise, the two-layer compatible control framework is as shown in Figure 1. In this section, the uncertainty in the model is all reflected as uncertainty of initial conditions, as will be described more fully in the next subsection. For a two-objective problem, compatible optimization will mean optimization of one of the objectives while maintaining both within an acceptable region of the space identified via the multi-objective search. It differs from the classical method of converting two-objective search to a single-objective search with a constraint on the other, since that approach does not use the Pareto front from the multi-objective search to set the values for the constraints for the measure that is converted from an objective to a constraint.

The first layer is compatible optimization and has the following two requirements:

1. Obtain a compatible (multi-dimensional) controlled objective region
2. The compatible controlled objective region must meet Pareto-optimality and the users' requirements.

The second layer is the compatible control layer and is devoted to satisfy the following requirements:

3. Design a real-time controller to control the system to remain within the (multi-dimensional) objective region determined in the first layer;

4. Optimize further the objective that is most critical to the user to optimize, rather than simply to keep within a specified region.

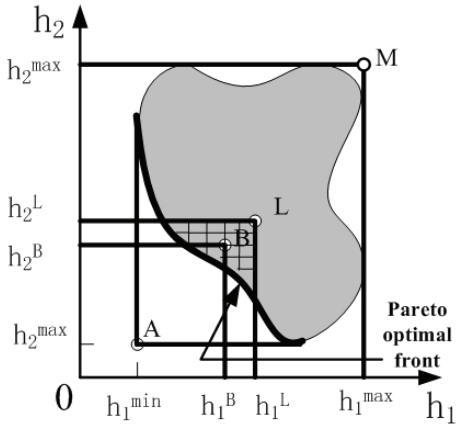


Fig. 2. Space with two conflicting objectives

In Figure 2, the shaded area in rectangle *AM* is the space of objectives in which control solutions exist; the cross-hatched area in rectangle *AL* is the area that meets the constraints of the practical problem; we shall call it the objective region with feasible control solutions; the bold curve is the Pareto optimal front of the objective space with control solutions.

The "optimal point" *A* is not in the subset of the objective space that contains feasible control solutions when the objectives conflict. So *A* is not an optimal point objective that can be attained. This means that there is no controller to realize *A*(*A* is a "Utopia" point). To guarantee the existence of a solution, the point objective *A* will be expanded to a region objective *AB*, where *AB* is a rectangular region (it is an interval for each single objective in *AB*; that is why we have to expand the point objective to an interval objective). To ensure the existence of a solution (i.e., a compatible solution), *B* must be in the region that includes a feasible control solution (the cross-hatched area in Figure 2). Since the selection of *B* would ideally optimize certain of the users' requirements, *B* should be a point on the Pareto optimal front, and included in rectangle *AL* (the bold Pareto front in Figure 2), in order not to be dominated by a better choice of *B*.

To determine the position of *B*, we must have two steps in the first layer algorithm: the first step is to find the Pareto front of the objective space with control solutions--that is, to find multiple, uniformly distributed points on (or approximating) the Pareto front; the second step is, according to the requirements of the users, to select one point *B* on the Pareto front that best defines the users' objective region (that is, the users' desired region for keeping the objectives within). Thus, the compatible objective region is obtained and the first-layer task is finished.

The second layer aims to design a compatible control system to realize the compatible multiple objectives from the first layer. In the controller design, we will not only realize these interval objectives, but also further optimize the objective that users are most concerned to minimize. The discussion above sketches the main ideas of our compatible control methodology.

The detailed algorithm will be introduced in the next section.

2.2 Energy-saving multi-objective compatible control algorithm (Xu, 2006)

Supposing the system model to be precise, an open-loop control method is adopted here. To conveniently illustrate our compatible control algorithm, we use as an example a linear discrete-time system as our controlled model

$$x(k+1) = Ax(k) + Bu(k) \tag{10}$$

$$y(k) = Cx(k) + Du(k) \tag{11}$$

where $x(k) \in R^n$ denotes the plant states, $y(k) \in R^q$ and $u(k) \in R^m$ are the control outputs and inputs, respectively, with constraint $\| u(k) \|_\infty \le a$. Because of different practical situations, the initial conditions may be different. We suppose that the initial conditions lie in some given compact set, *i.e.*, $X_0 = \{(x_1, x_2) \mid x_1 \in [9,11], x_2 \in [9,11]\}$ and the state-space matrices are

$$A = \begin{bmatrix} 0.8 & 0 \\ -0.1 & 0.9 \end{bmatrix}, B = \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix}, C = \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix}, D = 0 .$$

We shall denote this kind of control problem with uncertain initial states as the $I.C.-(X(0))$ problem (namely, control under uncertain initial state).

The control horizon is set $p = 15$ and $y^* = 0$ here. We aim to minimize the following two control performance indexes (control error, h_1 , and energy consumption, h_2).

$$h_1 = \sum_{k=1}^{15} y(k)^T y(k) \tag{12}$$

$$h_2 = \sum_{k=0}^{14} u(k)^T u(k) \tag{13}$$

To reduce computation and ensure control performance, we enforce $u(k) = u(5), k > 5$.

2.2.1 The compatible optimization layer-first layer

The aim of the compatible optimization layer is to find a compatible and relatively optimal objective region. To achieve this, first we should have a method to compare points in the multi-objective space, judging them to be better or worse, or sometimes, neither. It is easy to compare points in a single objective problem. However, it is not so direct in multi--objective problems. In this paper, we adopt Pareto non-domination as the comparison method.

In Figure 3 we assume that every individual i in the population has two attributes: 1) Non-domination rank r_i; 2) Local crowding distance d_i. Here $r(1) = r(3) = r(5) = 1,$, $r(2) = r(6) = 2,\ r(4) = 3$. The next step is to seek a Pareto-optimal set--i.e., a set composed of many non-dominated solutions. In each generation of the GA, popsize offspring and popsize parents are sorted using non-dominated sorting, and the popsize best are retained as parents for the next generation. The non-dominated sorting principle used to select solutions is:

$$i \prec_n j \ \text{ if } (r_i < r_j) \text{ or } ((r_i = r_j) \text{ and } d_i > d_j) \tag{14}$$

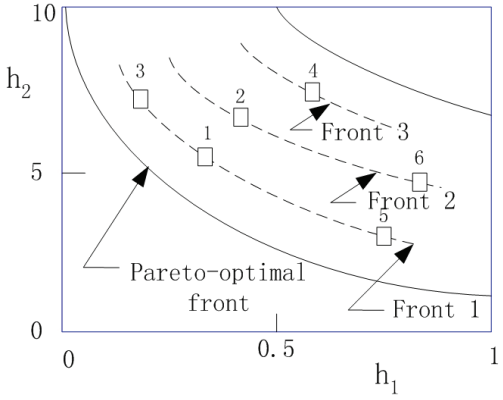


Fig. 3. Three fronts according to non-domination

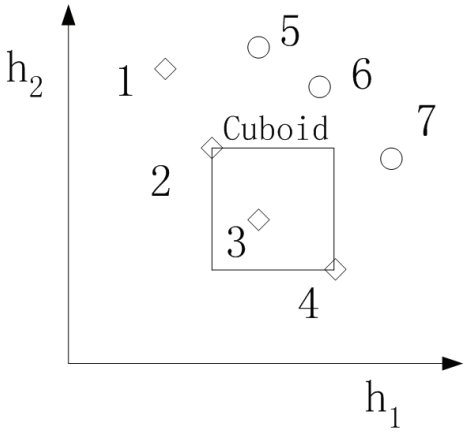


Fig. 4. The crowding distance relationship calculation

Based on the discussion above, and combined with the non-dominated sorting principle of NSGA-II, we propose our MOCC algorithm as follows, to determine the set of control inputs to be applied.

Algorithm 2.1 Robust Multi-Objective Optimization Algorithm

Step 1. Initialize parameters including the population size, $NIND$, the number of generations to calculate, $MAXGEN$, the number of variables $NVAR$, and the binary code length $PRECI$;

Step 2. For variables $u(k), k = 0,1,2,...,14$, create a random initial population $Chrom$ of candidate control vectors and set generation $gen = 0$;

Step 3. $X(0)$ is the initial region. Calculate the maximum h_1 and h_2 values of the population $Chrom$ in the $X(0)$ region;

Step 4. Based on the non-dominated solution definition, separate the population of solutions into consecutively ranked fronts and calculate the individual crowding distances d_i within each front;

Step 5. If $gen \le MAXGEN$, set $Chrom\_f = Chrom$ and store $Chrom\_f$ as the parent population;

Step 6. Selection operation: randomly choose pairs of solutions in the population and subject each pair to a tournament; the total number of tournaments is $NIND$. In each tournament, one solution is selected according to the solution's rank and crowding distance d_i value, and then becomes a breeder in the new population, $Chrom$;

Step 7. Crossover operation: perform crossover on pairs of solutions in population $Chrom$;

Step 8. Mutation operation: mutate the solutions in population $Chrom$; we obtain an offspring population $Chrom$;

Step 9. $X(0)$ is the initial region. Calculate the maximum h_1 and h_2 values of the offspring population $Chrom$ over the $X(0)$ region;

Step 10. Compare the offspring population $Chrom$ and parent population $Chrom\_f$ according to the Pareto non-domination definition and retain $NIND$ Pareto-optimal solutions;

Step 11. Set $gen = gen + 1$; if $gen < MAXGEN$ then return to Step 5; otherwise, stop the loop;

Step 12. Display the Pareto-optimal solutions.

Let $NIND = 80, MAXGEN = 150, NVAR = 12, PRECI = 2$. For the initial region control problem example here, we choose five arbitrary initial states $X(0)$ in the initial state region $X(0)$: [10.9, 10.8], [10.8, 10.9], [10.7, 11], [11, 10], [11, 9.5]; see Figure 5. The computational results of Algorithm 2.1 as a curve are shown in Figure 6.

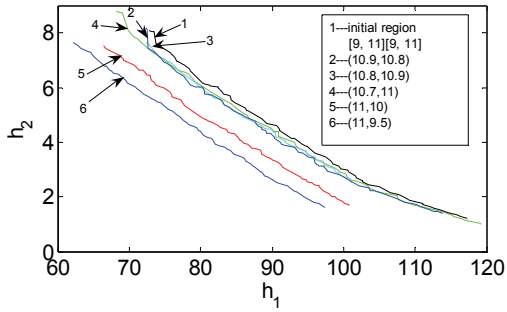


Fig. 5. Pareto fronts for some fixed initial points, namely, [11,10], [11,9.5], [10.7,11], [10.8,10.9] and [10.9,10.8].

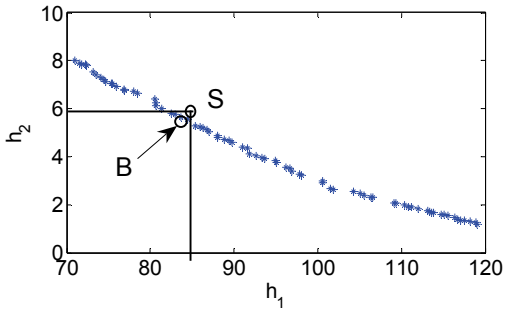


Fig. 6. The upper-right boundary of Pareto band and the user's selection of interval objective

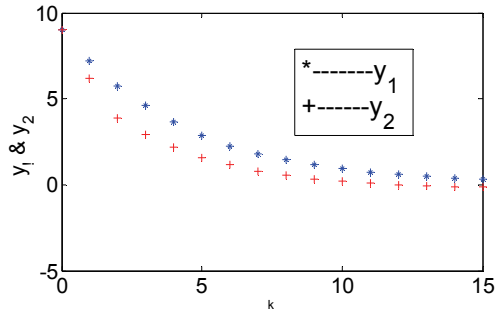


Fig. 7. The control result of the second layer controller

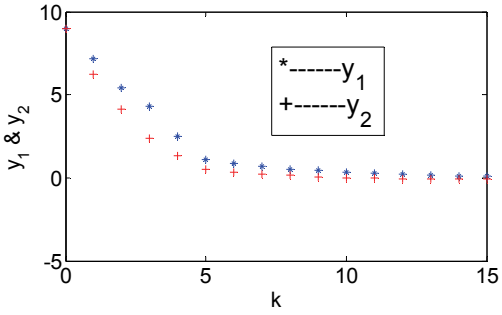


Fig. 8. The control result of the first layer (corresponding to point *B* in Figure 6)

2.2.2 Compatible control system design — the second layer

The second layer aims to design a multi-objective compatible control system. Assume that we now take energy consumption h_2 as the objective that users are most concerned to minimize.

If we normalize h_1 and h_2 in the intervals $h_1 \in [0,800]$ and $h_2 \in [0,8]$ as $\bar{h}_1$ and $\bar{h}_2$, then the normalized constraints are computed to be $\bar{h}_1 \le 0.66$. We make the interval objective $\bar{h}_1 \in [0,0.6556]$ from the first layer into a constraint. We now do constrained, single-objective optimization of $\bar{h}_2$ subject to the $\bar{h}_1$ constraint. The online multi-objective compatible control algorithm is now defined as follows:

Algorithm 2.2 (Robust multi-objective compatible controller design)

Step 1. For an arbitrary given initial condition $x(0) \in X(0)$ and randomly created initial values of variable $u(k), k = 0,1,2,...,14$;

Step 2. Determine the control input $u_{best}(k), k = 0,1,2,...,14$ by minimizing the performance index $\bar{h}_2$ with plant performance constraints $\bar{h}_1 \in [0,0.6556]$ and input constraints $u(k) \le 1, k = 0,1,2,...,14$ by traditional optimal methods with constraints for multiple variables;

Step 3. Implement the control input $u_{best}(k), k = 0,1,2,...,14$.

The system control result for the example with control input $u_{best}(k)$ and $x(0) = [10.8, 10.8]$ is shown as Figure 7.

In order to show that the performance for the primary controlled objective h_2 (i.e., energy consumption) has been improved in the second layer design, the control result of the first layer controller at the same point B is shown in Figure 8, and the difference of objective h_2 between the controllers of the first and second layers is quite apparent in Figure 9.

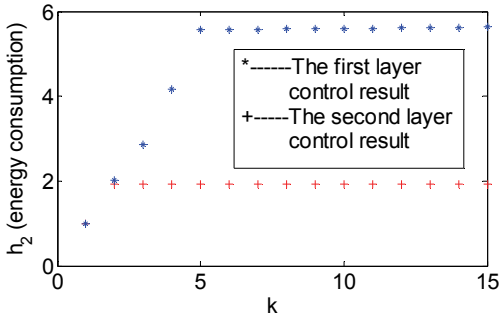


Fig. 9. Comparison of objective h_2 between the controllers of the first and second layers

From Figure 9, compared with the controller obtained by algorithm 2.1 in the first layer at B, the energy consumption h_2 with control input $u_{best}(k)$ obtained by Algorithm 2.2 in the second layer has decreased from 2.338 to 1.5651 (actual energy consumption, not normalized). It indicates that our method not only ensured the robustness of the system but also obviously reduced energy consumption.

3. Iterative MOCC based on preference selection strategy

It is difficult to generate a model that matches the real-world system precisely, so the two-layer method in the previous section is limited in its applicability. Disturbance and model error are usual in control problems. To make the method more usable for real-world problems, an online iterative MOCC algorithm is proposed in this section.

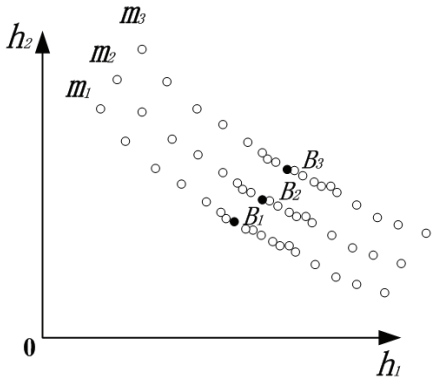


Fig. 10. Control process with non-even staged Pareto front at every control step

The control process can also be explained by figure 10. With the increasing of the time step k, the staged Pareto front will progress from m_3 to m_1 with selecting the control inputs

corresponding with B_3, B_2 and B_1 step by step. Note that staged Pareto front m_3, m_2, or m_1 represents an optimization at every time step k, that means the control input is computed by iteratively solving a suitable constrained optimization problem. Since k increases as the control system operates, the Pareto front of the control objective space is related to k, that is, it differs from the ultimate control problem Pareto front, but will converge to it. When the system is stable after certain steps, the Pareto front will also come to be stable.

The multi-objective control problem (MOCP) is different from the pure MOEA because the state variables are time dependent. This is a big challenge because it means that the time to compute a satisfactory solution at each time step must be small, in order to allow for multiple time steps (essentially, allowing for on-line determination of the control.) Population size in the evolutionary algorithm dramatically affects the computation time, but it is necessary to keep a certain population size to make the control process stable. This variety of control problem is different from pure dynamic multi-objective optimization since the states between neighboring time steps are tightly related, which means the neighboring sets in the time series of solution sets are typically relatively close. This is the foundation for taking the evolved population (set of candidate solutions) calculated for any given step as the initial population for the next step, which can obviously decrease the computing load as well as improve system stability. In this section, based on NSGA-II, a multi-objective iterative compatible control algorithm is presented according to the principles above. The iterative MOCC process is as shown in Figure 11.

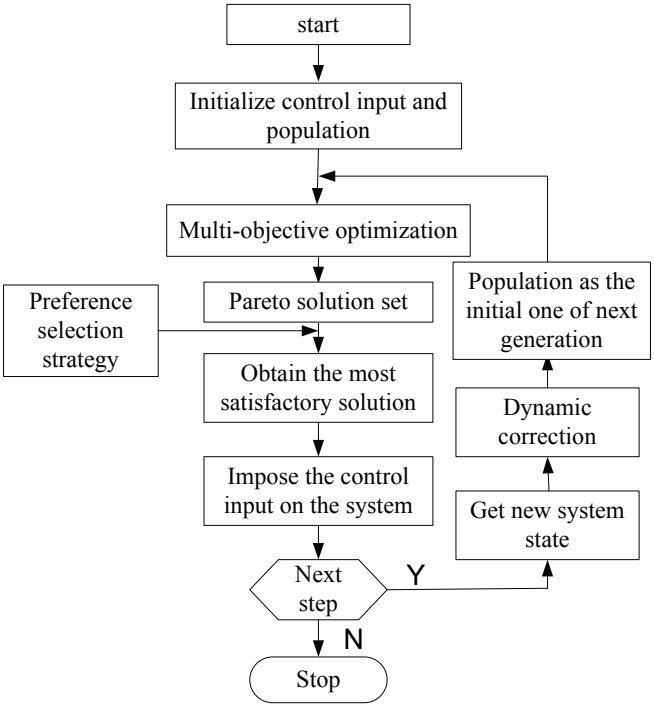


Fig. 11. Iterative MOCC control flow chart with preference selection strategy

3.1 Preference selection strategy and iterative control algorithm (Hu(a) et al, 2009)

Since the control problem is different from the pure optimization problem. For example, it concerns much more about the system stability, even on the sacrifice of losing certain optimal performance. There also exist disturbance and uncertainty in model structure. Therefore, without considering much more about the uniform spread of the solutions on the Pareto front, we take care more about the certain section of the Pareto front. This certain section is named as the optimal control objective area. To realize the non-uniform spread of the staged Pareto front, integrated with the niche technology that maintain the uniform spread of solutions in NSGA-II, an optimal control objective preference selection strategy is proposed which can be explained by the following eq.(15), where the right part is adopted to revise the fitness value of every solution according to its distance to the optimal control objective h_m^C .

$$d_{I_j^m} = d_{I_j^m} + \frac{h_m^{(I_{j+1}^m)} - h_m^{(I_{j-1}^m)}}{f_m^{\max} - f_m^{\min}} \frac{W}{(h_m^{(I_j^m)} - h_m^C)^2} \tag{15}$$

For the convergence of the system, whether or not the system is convergent can be evaluated through state variation, and convergence speed can be improved by selecting suitable individuals from the population. Since the system convergence cannot be judged from one or two steps, certain long step of system state should be tracked to evaluate whether the system is convergent or divergent. Whether a solution in the Pareto front is convergent or divergent is according to its oscillation at this point. The oscillation judgment should be after certain long time from the start of the control process since the system required the time to converge to the objective value. By this method, we can make sure which part in the full Pareto front is convergent. If it is, guide the state to an individual that locates in the nearest non-divergent segment of the Pareto front. The detailed algorithm shown as the flow chart in figure 11 is as follows.

Algorithm 3.1 (Online iterative control process based on preference selection strategy)

Step 1. Initialize parameters including the population size, $NIND$, the number of generations to calculate, $MAXGEN$, the number of variables, $NVAR$ and the binary code length $PRECI$; for variable $u(k), k = 0,1,2,...,14$ (the control vector to be solved for), create a random initial population $Chrom$ and set generation $gen = 0$;

Step 2. Calculate the h_1 and h_2 values of population $Chrom$ based on the initial state $X(0)$; according to the non-dominated sorting relationship, separate the population of solutions into consecutively ranked fronts and calculate the individual crowding distances d_i within each front;

Step 3. Set $Chrom\_f = Chrom$ and store $Chrom\_f$ as the parent population; selection operation: randomly choose pairs of solutions in the population $Chrom$ and subject each pair to a tournament, the total number of tournaments is $NIND$; in each tournament, one solution is selected according to the solution's rank and crowding distance d_i value, and then becomes a breeder in the new population, $Chrom$; crossover operation: perform crossover on the solutions in the population, $Chrom$; mutation operation: mutate the solutions in population $Chrom$; obtain an offspring population $Chrom$; compare the offspring population and parent population

$Chrom\_f$ according to the Pareto non-domination definition and (when applicable) crowding, and retain $NIND$ Pareto-optimal solutions;

Step 4. Set $gen = gen + 1$; if $gen \le MAXGEN$, then return to Step 3, otherwise, stop the loop; selection strategy: according to the user's preference strategy, however it may be algorithmically captured, select the individual in the population that is most satisfactory, and impose its control input on the system, then get the actual state;

Step 5. Keep population $Chrom$ as the initial population of the online control calculation; initialize the online loop counter $MAXGEN\_Online$ and the initial state with the current state, and replace $X(0)$ with the current system state;

Step 6. For as long as the process is to be controlled, repeat Step3 to Step5, only replace $MAXGEN$ with $MAXGEN\_Online$, otherwise, stop the loop.

3.2 Multi-objective control problem example

This subsection intends to introduce a multi-objective control problem example with an oscillating Pareto front segment.

(a) Control system model:

$$\begin{aligned} x_1(k+1) &= -0.2x_1(k)^2 - 0.2x_1(k)^2 + 0.1 - u_1(k)^2 - u_2(k)^2 \\ x_2(k+1) &= 0.3x_1(k) - 0.3x_2(k) + 0.1 + u_1(k) - u_2(k) \end{aligned} \tag{16}$$

(b) Two Objectives:

$$\min\{h_1(k) = (1 + x_1(k))^2, h_2(k) = (1 + x_2(k))^2\} \tag{17}$$

(c) Constraints:

$$u_1, u_2 \in [-0.2, 0.5] \tag{18}$$

(d) Initial point:

$$x_1(0) = 1, x_2(0) = 1 \tag{19}$$

Since the ε-*constra*int method has the capability to determine non-convex Pareto solutions, it is applied in this section to get the ultimate Pareto front of the control problem. One Pareto solution on the ultimate Pareto front will be obtained with one constraint. The Pareto front will be found by calculating with enough different constraints to fill the possible range. The Pareto front of this control problem is as shown in Figure 12.

The feature of this example is that when 0.95<c<1, objective h_2 is oscillating (see Figure 12). Simulation results of the two-objective variation process are as shown in Figure 13 and Figure 14, where c=0.965. Obviously the system is unstable, from Figure 14. It is easy to find that the oscillation in Figure 14 is the swing between A_1 and A_2 in Figure 12.

In this example, the control result is oscillating if the constraint c is located in 0.95-1.0. This subsection will try to design a selection strategy to judge and jump out if the control system is in the oscillating or divergent state, which can be embedded as the selection strategy in the flow chart shown in Figure 10. With Algorithm 3.1, if first h_1 is set as h_1<0.965, the same value as used in the ε-*constra*int method in the example, the algorithm will find that the state is not stable. A nearest non-divergent state will be found. See Figure 12, where a new suitable solution B is selected, and leads the system into a convergent state (see Figure 15 and the oscillating part and evenly distributed part in Figure 16).

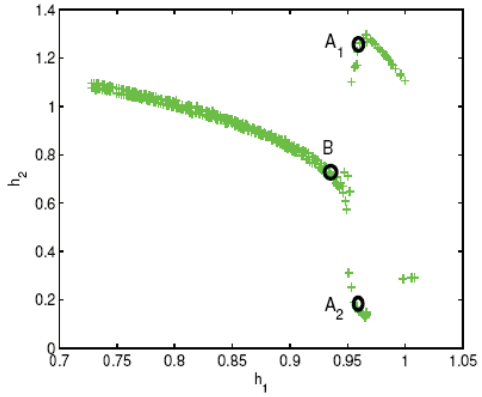


Fig. 12. Ultimate Pareto front of the oscillating multi-objective non-convex control problem example

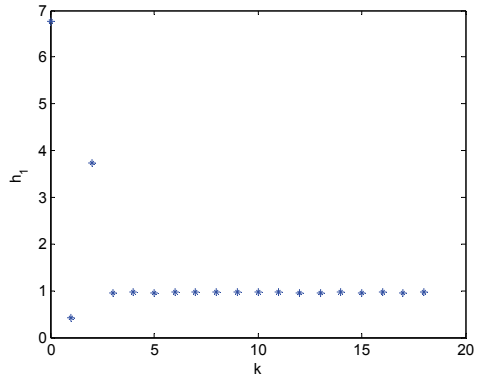


Fig. 13. Variation of h_1 with $\varepsilon - constraint$ method at $c = 0.965$

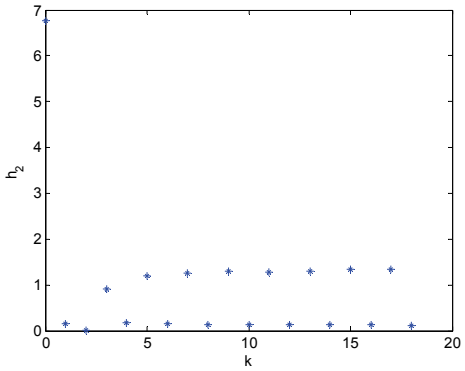


Fig. 14. Variation of h_2 with $\varepsilon - constraint$ method at $c = 0.965$

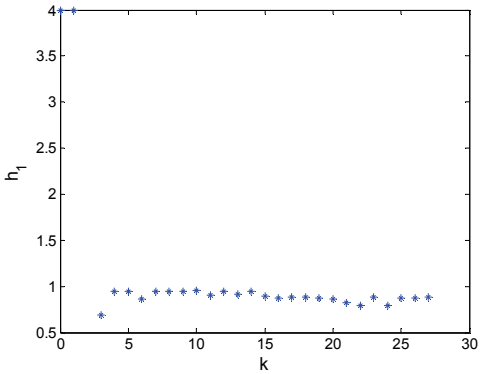


Fig. 15. Variation of h_1 with Algorithm3.1

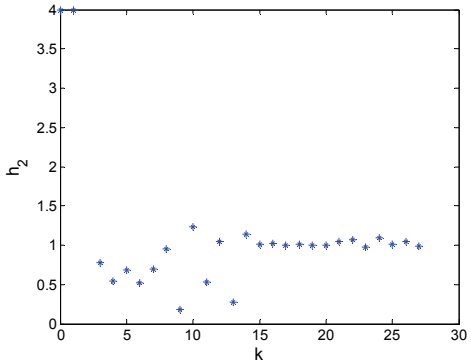


Fig. 16. Variation of h_2 with Algorithm 3.1

4. Application of MOCC to greenhouse environment control

It is well recognized that the greenhouse environment has a great influence on plant growth, production yield, quality, and maintenance processes of the plants. The greenhouse environment differs from a purely physical (non-biological) system, in that the greenhouse system is typically more complex and nonlinear, and the biological system is likely to have significant and numerous effects on its physical surroundings. Greenhouse interior temperature, air humidity and CO_2 concentration are the main control components influencing plant growth and energy usage. These components can be changed through heating, fogging, CO_2 injection, respectively, and ventilation affects all three of these components. Studies and research applications involving environmental control of greenhouses have been performed by many researchers (Pasgianos et al, 2003; Nielsen & Madsen, 1995; Young et al, 2000; Arvanitis et al. 2000; Taylor et al. 2000; Zolnier et al., 2000). These studies and researches are very important to engineering applications in the greenhouse.

There exists a series of dynamic models for greenhouse environments in the literature. The central state variable of greenhouse climate is typically air temperature, with relative

humidity and carbon dioxide concentration also considered. There are many disturbances to the greenhouse climate, which are primarily from solar radiation, outside temperature (conductive heat transfer and ventilation heat transfer) and interactions with occupants (plants), the controlled heating and ventilating equipment, and the floor.
Taking into account these analysis mentioned above, a simple greenhouse heating/cooling/ ventilating model can be obtained from the extant literature as the following differential equations:

$$\frac{dT_{in}(t)}{dt}=\frac{1}{\rho C_p V_T}[Q_{heater}(t)+S_i(t)-\lambda Q_{fog}(t)]-\frac{V_R(t)}{V_T}\cdot[T_{in}(t)-T_{out}(t)]-\frac{UA}{\rho C_p V_T}[T_{in}(t)-T_{out}(t)] \quad (20)$$

$$\frac{dw_{in}(t)}{dt}=\frac{Q_{fog}(t)}{V_H}+\frac{1}{V_H}[E(S_i(t),w_{in}(t))]-\frac{V_R(t)}{V_H}\cdot[w_{in}(t)-w_{out}(t)] \quad (21)$$

$$E(S_i(t),w_{in}(t))]=\alpha\frac{S_i(t)}{\lambda}-\beta_T w_{in}(t)$$

where T_{in}/T_{out} is the inside/outside air temperature(^oC), w_{in}/w_{out} is the inside/outside relative humidity(%), UA is the heat transfer coefficient of enclosure (WK^{-1}), V is the geometric volume of the greenhouse (m^3), ρ is the air density ($1.2kgm^{-3}$), C_p is the specific heat of air ($1006Jkg^{-1}K^{-1}$), Q_{heater} is the heat provided by the greenhouse heater(W), Q_{fog} is the water capacity of the fog system (gH_2Os^{-1}), S_i is the intercepted solar radiant energy (W), λ is the latent heat of vaporization ($2257\,Jg^{-1}$), V_R is the ventilation rate (m^3s^{-1}), $E(S_i,w_{in})$ is the evapotranspiration rate of the plants (gH_2Os^{-1}), which is affected by the given solar radiation, α and β_T are scaling parameters, V_T and V_H are the temperature and humidity of the actively mixing air volumes, respectively. Generally, V_T and V_H are as small as 60%-70% of the geometric volume V of the greenhouse.

4.1 Description of the MOCC algorithm based on energy-saving preference

Classical Multi-objective Control Problem methods commonly pursue a precise point as the control objective, and then optimize its tolerance with the reference value. In greenhouse climate, energy consumption and control precision conflict with each other. Low control deviation tolerance is at the cost of high energy consumption. If the greenhouse climate is controlled to a precise point, energy consumption must be high. In practical greenhouse engineering, plants can grow and flourish in some interval or region of humidity and temperature rather than only at one precise point. So it is unnecessary for the greenhouse climate control problem to pursue low control deviation tolerance. Humidity and temperature setpoints can be enlarged to intervals, which can reduce energy consumption while keeping the greenhouse climate suitable for plants to grow and flourish.

4.2 Control objectives

In greenhouse climate, Q_{heater} , Q_{fog} and V_R are the control inputs. In order to save energy, they should be minimized as much as possible. In practical greenhouse engineering, these three inputs have different power requirements. We set these three power ratios as $\lambda_1:\lambda_2:\lambda_3$ respectively. Then the energy objective function can be described as follows:

$$f_1 = \lambda_1 Q_{heart,\%} + \lambda_2 Q_{\%,\text{fog}} + \lambda_3 V_{R,\%} \quad (0 \le Q_{heart,\%}, Q_{\%,\text{fog}}, V_{R,\%} \le 1) \tag{22}$$

In the greenhouse climate model, although energy saving is an optimization objective, the temperature and humidity must also be kept in a range that promotes healthy plant growth. If the interior temperature and humidity of the greenhouse are unfit for plant growth, energy saving loses its practical significance. So energy consumption must be reduced while maintaining a greenhouse climate suitable for plant growth.

According to the analysis above, temperature and humidity objective functions will be cast as tolerances around a pre-set point, shown as:

$$f_2 = abs(T_{in} - T_{set}) \tag{23}$$

$$f_3 = abs(W_{in} - W_{set}) \tag{24}$$

T_{set}, W_{set} are the optimal values of the temperature and humidity ranges that will serve as the midpoints of the allowable ranges to be determined as suitable for plant growth. In the control process, if the values of (23) and (24) are within the ranges to be determined, then these objectives will be treated as having the same value, and will allow have no effect on the multi-objective optimization, which will consider only energy consumption.

4.3 The preference interval of energy-saving information (Xu et al, 2009)

In the greenhouse climate control problem, the energy-saving preference information is incorporated into the optimization process. In this situation, solutions with lower energy consumption are superior to others. In standard NSGA-II, the definition of crowding distance is identical except for the extreme points. Crowding distance plays a key role in obtaining well-distributed Pareto optimal solutions. In order to obtain dense Pareto optimal solutions distributed in the preference interval, the crowding definition of standard NSGA-II is modified.

First, the special temperature and humidity intervals that are suitable for plant growth are defined. Second, the minimum energy consumption value J within these special intervals can be obtained in each evolutionary generation. Because J is changed in the evolutionary computation process, adaptation is applied in this algorithm and J is updated in each evolutionary generation. Third, in the evolutionary process, J is set as the preference point. According to the preference point, the preference interval $J \le J_1 \le \theta * J$ (here θ is a constant, $1 < \theta$) is defined. Finally, in each evolutionary generation, if the solution lies within the preference interval, its crowding distances is set to n times that of the standard NSGA-II crowding distance. The time step should be chosen appropriately and carefully. If it is too small, the preference information will lose its power and can't direct the optimization, whereas if it is too larger, it can lead to the phenomenon of premature convergence.

Through modification of NSGA-II, dense Pareto optimal solutions with energy-saving preference information are obtained in the neighborhood of minimum value J, and, to some degree because of the reduction of population diversity, the algorithm can quickly converge to Pareto-optimal solutions.

4.4 Simulation results (Hu(b) et al, 2009)

In order to validate the effectiveness of the algorithm with energy-saving preference information, we use the classical greenhouse climate model mentioned above to illustrate it. Because Q_{heater} , Q_{fog} and V_R are normalized in the energy consumption function (22), coefficients $\lambda_1, \lambda_2, \lambda_3$ are percents of maximum power, respectively. In practical greenhouse engineering, the energy consumption of a heater is higher than that of devices generating fog or ventilation, and ventilation has the lowest energy consumption of these three control inputs.

In most scenarios, in summer, the control inputs used in a greenhouse are only fog and ventilation. So in that case, the energy consumption can be presented as:

$$f_1 = \lambda_2 Q_{\%,\text{fog}} + \lambda_3 V_{R,\%} \tag{25}$$

In simulation, we set $\lambda_2 : \lambda_3 = 20:1$ and $T_{set} = 27,\ W_{set} = 0.7$. Then

$$f_2 = abs(T_{in} - 27) \tag{26}$$

$$f_3 = abs(W_{in} - 0.7) \tag{27}$$

In the greenhouse climate model, the parameters of Table 1 are suitable for summer or winter. Their differences are the initial conditions. For the MOCC method, we set 24°C-30 °C and relative humidity 60%-80% as the control intervals. For the classical control method, we set 27 °C and relative humidity 70% as the control point.

Parameters name	Unit expression	values
C_0	$\min W^0 C^{-1}$	-324.67
UA	$W^0 C^{-1}$	29.81
t_v	min	3.41
λ'	W	465
α'	$\text{g}m^{-3}\min^{-1}W^{-1}$	0.0033
$1/V'$	$\text{g}m^{-3}\min^{-1}$	13.3

Table 1. Identified greenhouse model parameters

Operators and parameters	values
$T_{\text{in}}(0) = C_{T_0}(^0C)$	35
$w_{\text{in}}(0) = C_{w_0}(\%)$	35
$T_{\text{out}}(t)(^0C)$	30
$w_{\text{out}}(t)(\%)$	40
$S_i(t)(W/m^2)$	200

Table 2. Initial parameters of greenhouse in summer

Due to the rapidity of change of outdoor climate, we chose 15 minutes as the control step size and operate the control for 1.5 hours. In Figures 17 and 18, control results of the MOCC

method and classical control method (precise point control) in summer are shown. The horizontal axis represents control step and vertical axis is the control result. The total energy consumption is 4.1 for MOCC, and 8 for the precise point control. Because the control point is enlarged to an interval in MOCC, it allows much more room to compromise between control precision and energy consumption. In the greenhouse climate control problem, the requirement for control precision is low, and the control results of MOCC are suitable for plant growth, which allows a large reduction of energy consumption compared to precise point control.

In winter, control inputs are heat, fog and ventilation, and solar radiant energy is weak, so $S_i(t)$ is chosen as 20 rather than 200 in summer. In simulation, the initial conditions are shown in Table 3, and the energy consumption function is described as follows:

$$f_1 = 100Q_{heart,\%} + 20Q_{\%,\mathrm{fog}} + V_{R,\%} \tag{28}$$

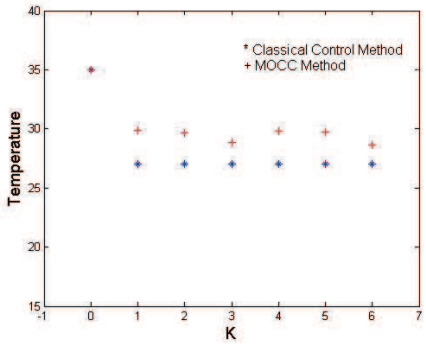


Fig. 17. The control results of Temperature in summer

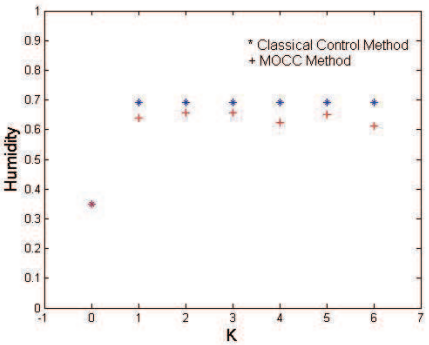


Fig. 18. The control results of Humidity in summer

In Figures 19 and 20, control results of the MOCC method and classical control method (precise point control) in winter are shown. The total energy consumption is 34.8 for MOCC, and 204.5 for precise point control. Comparing the energy consumption of MOCC and the classical control method, the MOCC method has an overwhelming advantage over classical control method with respect to energy saving.

Operators and parameters	values
$T_{in}(0) = C_{T_0}(^0C)$	15
$w_{in}(0) = C_{w_0}(\%)$	40
$T_{out}(t)(^0C)$	-2
$w_{out}(t)(\%)$	20
$S_i(t)(W/m^2)$	20
temperature 'interval' control objective	$24^0C - 30^0C$
Humidity 'interval' control objective	60% – 80%

Table 3. Initial parameters of greenhouse in winter

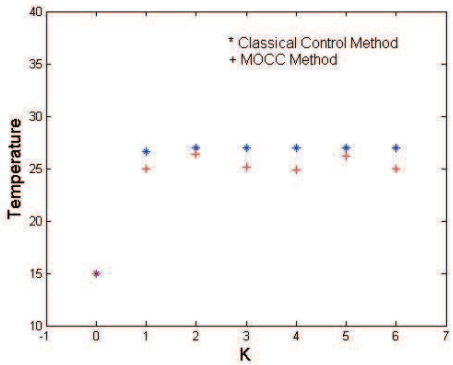


Fig. 19. The control results of Temperature in winter

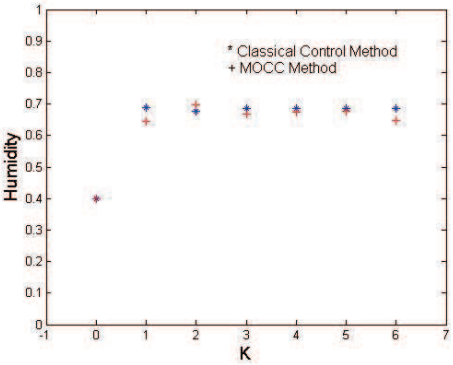


Fig. 20. The control results of Humidity in winter

5. Conclusions

Based on ideas developed in addressing practical greenhouse environmental control, we propose a new multi-objective compatible control method. Several detailed algorithms are proposed to meet the requirements of different kinds of problem: 1) A two-layer MOCC framework is presented for problems with a precise model; 2) To deal with situations

including model error and disturbance in the practical problem, a MOCC combining offline and online parts is proposed; 3) MOCC is applied to practical greenhouse control. The result illustrates the validity of the new strategy. The result of applying MOCC to this problem shows that MOCC can be applied widely.

6. Acknowledgement

This paper is supported by NSF (No. 60674070), 863 Plan (No. 2008AA10Z227) and National SCI. & Tech. Support Plan (No. 2008BADA6B01) Of China.

7. References

Arvanitis, K.G., Paraskevopoulos, P.N. & Vernardos, A.A., (2000). Multirate adaptive temperature control of greenhouses. *Computers and Electronics in Agriculture*. 26, 3, 303-320, 0168-1699

Chen, J. (2008). Multi-objective Control Research on Oversaturated Urban transportation, Ph.D dissertation, (Advisor: Xu, L.), Tongji University, Shanghai, China

Deb, K., Agrawal, S., Pratap, A., & Meyarivan, T.(2002). A fast and elitist multi-objective genetic algorithm: NSGA-II, *IEEE Transactions on Evolutionary Computation*, 6, 182-197, 1089-778X

Eisenhart, K. J. (2003). Multiobjective optimal control problems with endpoint and state constraint, Ph.D. thesis, Western Michigan University

Fleming, P.J., & Purshouse, R.C.(2002). Evolutionary algorithms in control systems engineering: a survey, *Control Engineering Practice*, 10, 1223-1241, 0967-0661

Goldberg, D.E.(1989). Genetic Algorithms in Search, *Optimization and Machine Learning*, Addison-Wesley, Reading, MA

Hu(a), Q, Xu, L & Goodman, D E (2009). Non-even spread NSGA-II and its application to conflicting multi-objective compatible control, *2009 World Summit on Genetic and Evolutionary Computation*, June 12-14, 2009, Shanghai, China

Hu(b), H, Xu, L & Hu, Q (2009). Model-based Compromise control of greenhouse climate using Pareto optimization, *2009 World Summit on Genetic and Evolutionary Computation*, June 12-14, 2009, Shanghai, China

Jensen, M.T.(2003). Reducing the run-time complexity of multi-objective EAs: the NSGA-II and other algorithms, *IEEE Transactions on Evolutionary Computation*, 7, 503-515, 1089-778X

Masaaki, I. (1997). Multiobjective optimal control through linear programming with interval objective function, *Proceedings of the 36th SICE Annual Conference*, July 29-31, 1997,Tokushima, 1185-1188.

Nielsen, B. & Madsen, H.(1995). Identifieation of transfer functions for control of greenhouse air temperature. *Journal of agriculture Engineering Research*., 60, 25–34, 1995.

Pasgianos, G., Arvanitis K., Polycarpou P., & Sigrimis N.(2003). A nonlinear feedback technique for greenhouse environmental control, *Computers and Electronics in Agriculture*, 40:153–177, October 2003, 0168-1699

Rangan, S. & Poolla, K.(1997). Weighted optimization for multiobjective full-information control problems, *System & Control Letter*, 31, 207-213, 0167-6911

Scherer, C. W.(1995). Multiobjective $H_2 - H_\infty$ Control, *IEEE Transaction on Automatic Control*, 40, 1054-1061, 0018-9286

Scherer, C., Gahinet P. & Chilali, M.(1997). Multiobjective output-feedback control via LMI Optimization, *IEEE Transactions on Automatic Control*, 42, 896-911, 0018-9286

Sznaier, M., Rotstein, H., Bu, J. Y. & Sideris, A.(2000). An exact solution to continuous-time mixed $H_2 - H_\infty$ control problems, IEEE Transactions on Automatic Control, 45, 2095-2101, 0018-9286

Taylor, C.J., Chotai, A. & Young, P.C. (2000). State space control system design based on non-minimal state variable feedback: further generalizations and unification results. *International Journal of Control*, 73, 1329-1345, 1755-9340

Wu, J. (2003) Research and application of low-cost greenhouse environment control system, March, 2003, Ph.D. thesis (Advisor: Xu L.) of Tongji University, Shanghai, China, 53-58.

Xu, L., Zou, Z. & Hu, Q.(2006). Two-Layer Optimization Compatible Control for Multi-Objective Control Systems, *IEEE International Conference on Networking, Sensing and Control*, 23-25 April, 2006, 658-663.

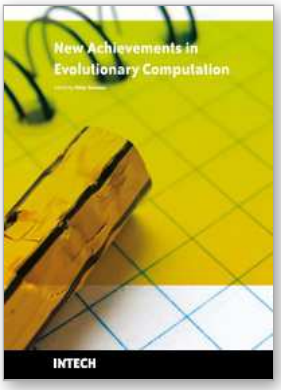
Xu, L. , Hu, Q. & Goodman E. D.(2007). Two Layer Iterative Multi-Objective Compatible Control Algorithm , *46th IEEE CDC (Conference on Decision and Control)*, USA

Xu, L. , Hu, Q. & Goodman E. D.(2007). A Compatible Energy-saving Control Algorithm for a Class of Conflicted Multi-objective Control Problem , *2007 IEEE CEC (Congress on Evolutionary Computation)*, Singapore

Xu, L, Hu, H. & Zhu, B.(2009). Energy-saving control of greenhouse climate based on MOCC strategy, *2009 World Summit on Genetic and Evolutionary Computation*, June 12-14, 2009, Shanghai, China

Young, L. B. D., Price P.C. & Janssens K.(2000). Recent developments in the modelling of imperfectly mixed irspaces. *Computers and electronics in agriculture*, 26, 239–254, 0168-1699

Zolnier, S., Gates, R.S., Buxton, J. & Mach, C., (2000). Psychrometric and ventilation constraints for vapor pressure deficit control. *Computers and electronics in agriculture*. 26, 343-359, 0168-1699



New Achievements in Evolutionary Computation

Edited by Peter Korosec

ISBN 978-953-307-053-7
Hard cover, 318 pages
Publisher InTech
Published online 01, February, 2010
Published in print edition February, 2010

Evolutionary computation has been widely used in computer science for decades. Even though it started as far back as the 1960s with simulated evolution, the subject is still evolving. During this time, new metaheuristic optimization approaches, like evolutionary algorithms, genetic algorithms, swarm intelligence, etc., were being developed and new fields of usage in artificial intelligence, machine learning, combinatorial and numerical optimization, etc., were being explored. However, even with so much work done, novel research into new techniques and new areas of usage is far from over. This book presents some new theoretical as well as practical aspects of evolutionary computation. This book will be of great value to undergraduates, graduate students, researchers in computer science, and anyone else with an interest in learning about the latest developments in evolutionary computation.

InTech Europe
University Campus STeP Ri
Slavka Krautzeka 83/A
51000 Rijeka, Croatia
Phone: +385 (51) 770 447
Fax: +385 (51) 686 166
www.intechopen.com

InTech China
Unit 405, Office Block, Hotel Equatorial Shanghai
No.65, Yan An Road (West), Shanghai, 200040, China
中国上海市延安西路65号上海国际贵都大饭店办公楼405单元
Phone: +86-21-62489820
Fax: +86-21-62489821