Human-computer co-operative co-evolutionary method and its application to a satellite module layout design problem

J-Z. Huo, H-F. Teng and W. Sun
School of Mechanical Engineering
Dalian University of Technology
Dalian, China

J. Chen
Department of Naval Architecture and Ocean Engineering
Dalian University of Technology
Dalian, China

ABSTRACT

The layout design of a satellite module is a complex mechanical layout problem. Its main difficulties lie in combinatorial explosion of computational complexity, engineering complexity, and applicability in engineering practice. Inspired by the human-computer co-operation ideas, a human–computer co-operative co-evolutionary method for optimising layout design of a satellite module is developed. This method constructs the diversity reference set by using the diversity intelligence solutions (DIs) that are created by using the combinatorial operators of differential evolution (DE) and the blend crossover operator (BLX-a). During the co-evolution process of the presented method, the AIs, the DIs and the algorithm solutions are expressed by unified encoding strings and incorporated together to create new co-operative solutions. An instance of a satellite module layout design is presented to demonstrate the feasibility and effectiveness of the proposed method. Compared with the co-evolutionary approach and the all-at-once optimisation approaches, computational results show that the proposed method not only can produce better solutions, but also can better balance the conflicting objectives on the trade-off issues.

NOMENCLATURE

ADS	automated design synthesis
AIs	artificial intelligence solutions
a_i	lengths of the longer edge of Obj_i
b_i	lengths of the shorter edge of Obj_i
BLX	blend crossover operator
CCEF	co-operative co-evolutionary framework
CCGA	co-operative co-evolutionary genetic algorithm
C	the large constant
$C_1, C_2, ..., C_k$	the normalising coefficients of each sub-objective function
CCEF	name co-operative co-evolutionary framework
$C_i = \{P_i, R_i\}$	the permissible location area of the ith object
C_{max1}	the big positive numbers to ensure that the value of the $(C_{max1} - T_p)$ is larger than zero
C_{max2}	the big positive numbers to ensure that the value of the $(C_{max2} - NObj)$ is larger than zero
$CR \in [0,1]$	a cross factor
D	the feasible region of variable X
DE	differential evolution

Paper No. 3417. Manuscript received 12 March 2009, revised 1 October 2009, second revision 23 October 2009, accepted 26 October 2009.

https://doi.org/10.1017/S0001924000003663 Published online by Cambridge University Press

DIs diversity intelligence solutions
D^i_j a segment that is between the centre of the *i*th object and the *j*th direction of the *i*th object
d_{ij} the distance between the *i*th object and the *j*th object
Dmin the minimise diverse disparity value between a new AI and each AI of the populationGALib genetic algorithm library
D_t thickness of bearing plate
$F \in [0,2]$ a zoom factor
$F_1, ..., F_k$ the sub-objective set
H_1 distance between the fourth bearing base and the plane *xoy*
H_2 distance between the second bearing base and the plane *xoy*
HAKBLDM human-algorithm-knowledge-based layout designing method
HCCGA human-computer co-operative co-evolutionary genetic algorithm
h_i height of Obj_i
H_t height of standing column
$\mathbf{I_E}$ vector – the inertia matrix of whole system with the deployed solar panel
$\mathbf{I_F}$ vector – the inertia matrix of whole system with the folded solar panel
I_{xx}, I_{yy}, I_{zz} the moments of inertia of the whole module with respect to inertial co-ordinate system
$J(D^i_j, obj_k)$ a judgmental mark that judges whether the segment D^i_j crosses with the object obj_k
m_0 the mass of Obj_0
MDO multi-disciplinary design optimisation
n total number of objects
NObj the number of the accessibility directions of all objects
OBJ vector – a set of objects
$\mathbf{Obj_i}$ vector – the *i*th object
Obj_0 the left fixed part of the module
obj,(*X*) the location area of the *i*th object
$O_e(x_e, y_e, z_e)$ the expected position of O_m
$O_m(x_m, y_m, z_m)$ the real centroid of the whole system
p_c the crossover probability
$\mathbf{p_i}$ vector – position of a reference point of Obj_i
p_m the mutation probability
q the accessibility directions of each object
$r_1 \in \{1,2,...n\}$ a random integer
R radius of bearing plate
R_C radius of standing column
r_i radius of Obj_i
T_G the sum of weight-distance product of the gathering objects
th a diverse threshold value
T_P the sum of weight-distance product of the disturbing objects
$\mathbf{U^1}$ vector – a state vector that indicates the whole system state
VLSI very large scale integrated circuits
w_{ij} the gathering weight between the *i*th object and the *j*th object
X vector – a layout scheme of a satellite module
$\vec{x}_{r1}$ a solution vector

Greek characters

$\alpha \in [0,1]$ a uniform random number
α_i angle between the longer edge of its bottom and the reference plane *xoz*
$\Delta I_{xx}, \Delta I_{yy}, \Delta I_{zz}$ the engineering expected value of the inertia of the whole module
$\delta\theta_x, \delta\theta_y, \delta\theta_z$ the allowable errors of θ_x, θ_y and θ_z
ΔV_{jk} the overlapping volume between Obj_j and Obj_k
$\delta x_e, \delta y_e, \delta z_e$ the allowable error of O_m
$\lambda_1, \lambda_2, ..., \lambda_k$, the weight factors of each objective function
$\lambda_{ij} \in [0,10]$ the disturbing weight between the *i*th object and the *j*th object
$\theta_x, \theta_y, \theta_z$ the angles between the principal axis of inertia of the module and the reference axis *ox*, *oy*, oz along each direction
Ψ_i vector – the orientation vector with respect to a global co-ordinate system

1.0 INTRODUCTION

The spatial layout problem of a satellite module concerns the placement of components in an available space such that a set of objectives can be optimised until satisfying optional spatial or performance constraints. It belongs to a combinatorial optimisation problem in mathematics, a complex layout conceptual design in mechanics and a human-computer co-operative problem in artificial intelligence. Its main difficulties lie in computing complexity, engineering complexity and applicability in engineering practice[1]. In 1997, Braun[2] adopted a multilevel collaborative approach in the solution of launch vehicle design. This approach performed the layout design of each subspace iteratively. In this approach, a problem was decomposed into a user-defined number of subspace optimisation problems that were driven toward interdisciplinary compatibility and the appropriate solution by a system-level co-ordination process. Kamaran and Maziar[3] proposed a iterative layout algorithm for the arrangement of functional subassemblies of the satellite ZS3-SAT. Based on modular decomposition and a reference layout scheme, they iteratively added subassemblies to a satellite by using a composite design method and human–computer interactive techniques. In 2004, Grignon and Fadel[4] presented a genetic algorithm (GA) based configuration design optimisation method. This method presented the designer with multiple solutions to the engineering configuration design problem using a Pareto genetic algorithm working on a population of sets instead of a population of individuals. It results in a more informed decision on the trade-off issues between the various objectives. This method had been used to solve a satellite configuration design problem with complex objectives and constraints of up to seven components. The above-mentioned research shows that the stochastic algorithms are superior in solving the complex three-dimensional component layout problems. Ferebee[5] used a software package ADS (Automated Design Synthesis) to solve a satellite module layout problem with the constraints of the inertia angle and the electromagnetic compatibility among components. Tanner[6] adopted human–computer interactive technologies and a constraint reasoning method to solve the instruments stand layout design of an American space platform. In 1995, Szykman and Cagan[7] made use of a simulated annealing algorithm to solve the three dimensional constraint layout and routing problems that involve multiple objectives and multiple constraints.

The studies directly related to this article are as follows: In 2001, Qian *et al*[8] proposed a human-computer interactive genetic algorithm to solve a simplified satellite module layout problem. The artificial solutions and the algorithm solutions were expressed by uniform encoding strings and co-operated together to create new co-operative solutions, and took part in the genetic operations together, which realised the combination of humans and algorithm in the level of algorithm's genes. In 2008, Liu and Teng[9] presented a human-algorithm-knowledge based human–computer co-operative layout designing method to solve a satellite module layout problem. In the method, humans provided artificial layout schemes (artificial solutions), relevant layout diagrams afforded knowledge layout schemes (knowledge solutions), and an evolutionary algorithm produced layout schemes (algorithm solutions). Based on the

https://doi.org/10.1017/S0001924000003663 Published online by Cambridge University Press

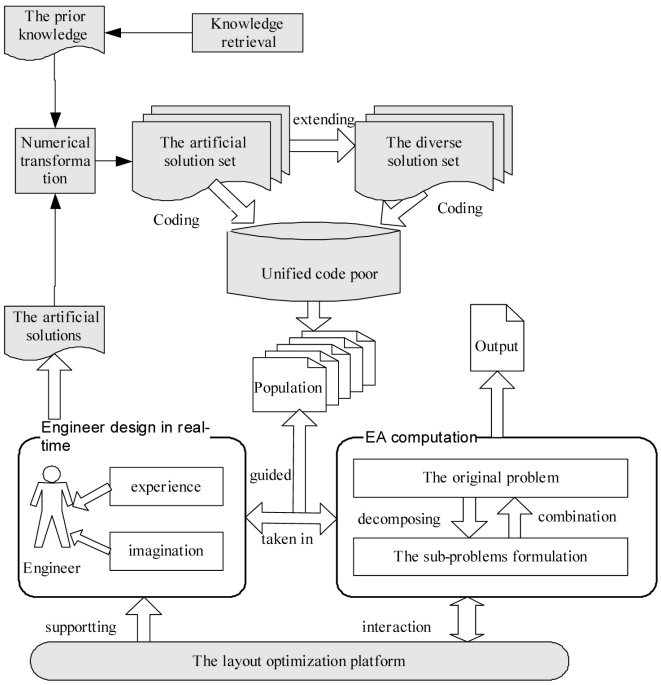


Figure 1. Framework of the human–computer co-operative co-evolutionary method.

human-computer co-operative genetic algorithm, all these solutions were expressed by uniform encoding strings and put together to take part in the genetic operations[8]. In 2005, based on the CCEF, Shi[21,23] studied a co-operative co-evolutionary differential evolution (CCDE) and used CCDE to solve a satellite module layout problem with performance constraints, in which a whole satellite module layout problem is decomposed into several sub-layout problems according to the multi-sub-physical structures of a satellite module. In 2009, Huo and Teng[20] presented a co-evolutionary genetic method with heuristic rules for the optimal layout design of a satellite module, in which a final whole layout scheme was obtained by using CCGA with a heuristic combinatorial-rotation method.

The previous researches can be concluded that two methods can be used to reduce the engineering complexity of the medium complexity layout problems. One is the divide and conquer methods (such as the multidisciplinary design optimisation, the co-evolutionary algorithm, etc.) which has been used widely to solve the VLSI problems[10] and the satellite module layout problems[20-24]. The other is the human-computer co-operation methods[8,9,11], which can combine human's special abilities (visible sensation, perceptivity and system controlling ability) with computer-based capabilities.

In this paper, the layout design of a satellite module exists of two key issues. Firstly, the satellite module layout problem contains more multiple complex constraints, such as accessibility requirement, the gathering requirement and the disturbing requirement. Secondly, it is necessary to study a human-computer co-operation method to solve this problem efficiently.

Based on the Refs 9 and 21, in this study, a human-computer co-operative co-evolutionary method is developed to solve a satellite module layout problem. In this method, the diversity reference set is constructed by the DIs using the combinatorial operators of DE and the BLX-α to improve the integration of the human experience and the computer. During the co-evolution of the sub-problems, the AIs, the DIs and the algorithm solutions are expressed by unified encoding strings and co-operated together to create new co-operative solutions. In Section 4.4, a satellite module layout design was presented and solved by the proposed method.

2.0 FIVE ISSUES IN THE HUMAN-COMPUTER CO-OPERATIVE CO-EVOLUTIONARY DESIGN METHOD

The framework of the human–computer co-operative co-evolutionary design method was shown in Fig. 1. There are five issues in this method for a satellite module layout problem, which are described hereafter.

EA selecting in CCEF algorithm. In 1994, Potter[12] proposed a co-operative co-evolutionary framework (CCEF) that adopted multiple cooperating subpopulations to co-evolve subcomponents of the solution. Based on the CCEF, GA was tested in solving high-dimensional function optimisation problems. In Ref. 21, CCDE was used to solve a satellite module layout problem. This study adopts CCGA because of its high computation efficiency;

The extending and diversifying of the 'human intelligence', i.e., to use the proposed DIs to extend and exploit human abilities. The DIs is constructed based on the proposed AIs. Engineers can construct some AIs by real time on-line design and translation of the prior knowledge, but it is not adequate comparing with the size of the algorithm's solutions. The evolutionary algorithm is a population-based algorithm. If the AIs are directly added into the population of the algorithm, it might result in some problems like:

1. The AIs might be drowned into the algorithm's population if the fitness values of the algorithm solutions are better than those of the AIs;
2. If the fitness values of the AIs are better than those of the algorithm's solutions, the AIs might replace the potential superior algorithm solutions, thus make the algorithm fall into a local search. This process is a 'replacement' instead of an 'integration' in essence. In order to combine the algorithm's solutions with the AIs naturally and efficiently, it is necessary to extend the artificial solutions' effect to produce several DIs.

The integration of the algorithm's solutions, the AIs and the DIs. Many researches have been done for the 'Man-Machine Synergy' or the 'Human–Computer Co-operation'. One of them is the human-computer evolutionary algorithms[13-15]. Based on the co-operation method of three kinds of solutions[9] and the CCDE method[23] for the satellite module layout problem, in this study, the algorithm's solutions can be obtained by the evolution of the CCGA. The proposed AIs can be designed based on engineers' intelligence and experience. This study constructed the DIs proposed by using the diversity methods based on the AIs to improve the integration of the human and the computer. During the co-evolution process, all these solutions are expressed by unified encoding strings and added into the population of the algorithm according to their fitness values and their diverse values. Thus the human intelligence and the algorithm are integrated as an intelligent system on the level of the algorithm gene. The relationship of the three kinds of solutions is shown in Fig. 2.

The human-computer interface design. This part is excluded in this paper (see Ref. 16. The human–computer interface of Ref. 16 can be used to design the interface of human-computer co-operative algorithm of Ref. 8.)

3.0 HUMAN-COMPUTER CO-OPERATIVE CO-EVOLUTIONARY METHOD (HCCGA)

3.1 The construction and updating of the AIs set

The flowchart of the construction and updating of the AIs set is illustrated as Fig. 3. The AIs set consists of the solutions designed by engineers during the real time on-line process and the solutions

https://doi.org/10.1017/S0001924000003663 Published online by Cambridge University Press

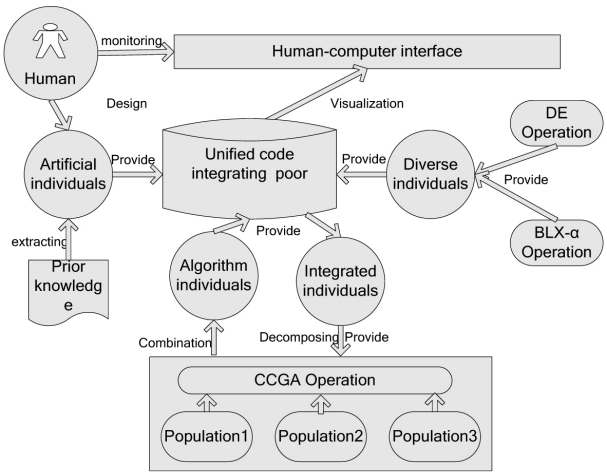


Figure 2. Relationship of the three kinds of solutions.

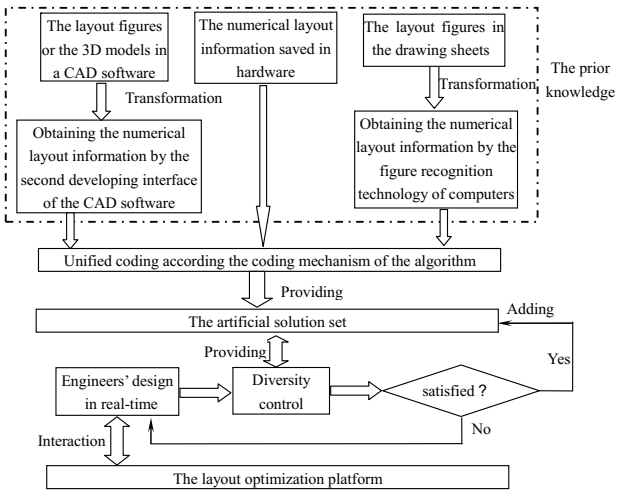


Figure 3. Flowchart of the artificial solution set construction and updating.

translated by the prior knowledge. The AIs translated by the prior knowledge are predetermined in the optimisation process. The on-line AIs can be updated and extended continuously. In order to keep the diversity of the AIs set, this study adopted a population-based updating method which is described as follows: first, set a diverse threshold *th*, compute all the diverse disparity values between a new AIs and each AIs of the population, and search for the minimum diverse disparity value *Dmin*. If *Dmin* > *th*, this new artificial solution is added into the artificial solution set. If not, engineers can keep on modifying this new AIs or designing a new AIs.

The diverse disparity value *Dmin*(*x*, *y*) of two AIs can be calculated by Equation (1).

$$Dmin(\mathbf{x},\mathbf{y}) = \sqrt{\sum_{i=1}^{n}(x_i - y_i)^2} \quad \forall i = 1,\cdots,n \qquad \ldots (1)$$

where $\mathbf{x},\mathbf{y}$ denotes the solution vector of the population, $\mathbf{x}_i,\mathbf{y}_i$ denotes the variables of the solution vectors $\mathbf{x},\mathbf{y}$.

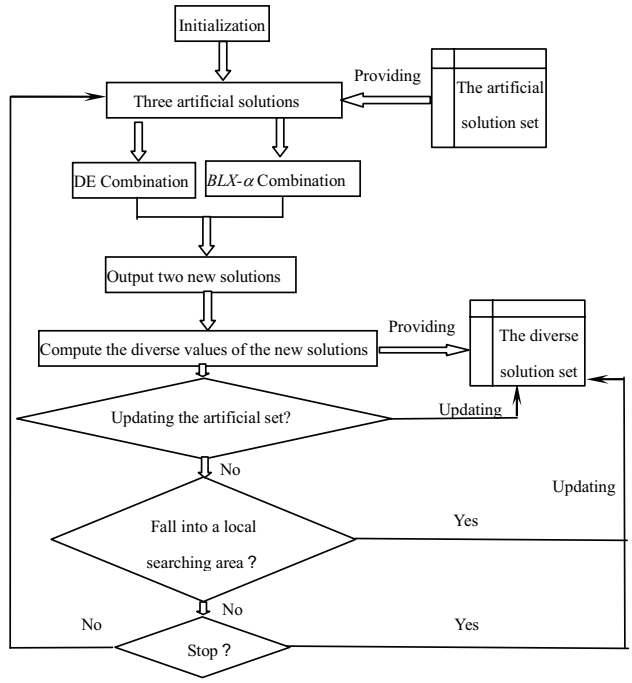


Figure 4. Flowchart of the diverse solution set construction and updating.

3.2 The construction and updating of the DIs set

The construction and updating process of the DIs set is illustrated in Fig. 4. This study adopted combinatorial operators to construct the DIs. Currently, the combinatorial operators of EA mainly consist of the average combinatorial operator, the linear combinatorial operator, the combinatorial operator of DE[17] and of the BLX-α [18]. Among all these combinatorial operators, the combinatorial operator of DE and the BLX-α are comparatively more efficient in keeping the diversity of the population[18-19]. This study made use of the combinatorial operators of DE and the BLX-α to construct the DIs set according to the AIs set. The construction process of the DIs set can be described as follows:

1. Select several AIs from the AIs set, and use the combinatorial operators of DE and the BLX-α to produce new DIs;
2. Judge whether the diverse values of the DIs are larger than the minimum value of the DIs set. If yes, add the new DIs into the DIs set;
3. Repeat Steps 1 and 2 until the AIs set reach its maximum size.

The combinatorial operator of DE[17] can be described as follows: suppose that three solutions vectors, $\vec{\mathbf{x}}_{r1}$, $\vec{\mathbf{x}}_{r2}$ and $\vec{\mathbf{x}}_{r3}$ are randomly selected from the current population, then the difference vector $\vec{\mathbf{v}}_i$ can be calculated by Equation (2).

$$\vec{\mathbf{v}}_i = \vec{\mathbf{x}}_{r1} + F(\vec{\mathbf{x}}_{r2} - \vec{\mathbf{x}}_{r3}) \quad i = 1,2,...,N \qquad \ldots (2)$$

where $r1, r2, r3 \in \{1,2,...,n\}$ are random integers, and $F \in [0,2]$ is a zoom factor.

Suppose that $\vec{\mathbf{v}}_i = [v_{1i}, v_{2i}, ..., v_{ni}]$, and the solution vectors selected from the parent and the current populations are $\vec{x}_i = [x_{1i}, x_{2i}, ..., x_{ni}]$ and $\vec{\breve{x}}_i = [\breve{x}_{1i}, \breve{x}_{2i}, ..., \breve{x}_{ni}]$, respectively. After the combinatorial operation, the *j*th variable of the $\vec{\breve{x}}_i$ can be determined by Equation (3).

https://doi.org/10.1017/S0001924000003663 Published online by Cambridge University Press

```
Begin
    Setting the integration number Nc ;
    Selecting an artificial solution AIi and a diverse solution DIi from the artificial solution set and
    the diverse solution set respectively;
    For (each algorithm subpopulation)
        count=0
        Do
        {
            Adding the solution AIi into the algorithm population and Substituting a algorithm
        solution with the poor fitness value ;
            Adding the solution DIi into the algorithm population and Substituting a algorithm
        solution with the poor diverse value ;
            Count++ ;
        }While ( count != Nc )
    End For
End
```

Figure 5. Flowchart of the integration process of three kinds of solutions.

$$x'_{ji} = \begin{cases} v_{ji}, & \text{if } randb(j) \le CR \text{ or } j = randr(i) \\ x_{ji}, & \text{if } randb(j) > CR \text{ and } j \ne randr(i) \end{cases} \quad \ldots (3)$$

where $j \in [0,1]$; n is the number of the variables; $randb(j) \in [0,1]$ is a uniform random number; $randb(i) \in [1,2, ..., n]$ is the number of the selected solution, and $CR \in [0,1]$ is a cross factor.

The combinatorial operator of the BLX-α can be described as follows: supposed that $x = (x_1,\ldots..., x_n)$ and $y = (y_1,\ldots..., y_n)$ are random solution vectors of the population, $z = (z_1,\ldots..., z_n)$ is a new solution vector after the BLX-α operation, then the new solution vector **z** can be determined by Equation (4).

$$\begin{aligned} z_i &= Rand(C_{min} - I*\alpha, C_{max} - I*\alpha) \\ C_{min} &= min(x_i, y_i) \\ C_{max} &= max(x_i, y_i) \\ I &= C_{max} - C_{min} \end{aligned} \quad \ldots (4)$$

where $\alpha \in [0,1]$ is a uniform parameter. Usually, $\alpha = 0{\cdot}5$.

During the optimisation, there is a need to update the diverse solution set to continuously produce the DIs. The DIs set should be updated under the following three circumstances:

1. If the AIs set needs to be updated, the DIs set should be updated too;
2. If a new AIs is produced, use the combinatorial operators to produce a new DIs, then add this new DIs into the DIs set;
3. If the algorithm falls into a local searching area, produce several new algorithm solutions randomly. Combine those algorithm solutions with the AIs to construct new DIs, and then add new DIs into the DIs set.

3.3 The integration of the algorithm solutions with the AIs and the DIs

How to integrate the algorithm solutions with the AIs and the DIs efficiently is very important. Current integration methods mainly consist of two categories:

1. The 'replacement' mode, which replaces several algorithm solutions with the AIs randomly during the optimisation process[8].
2. The 'cross' mode, which crosses the AIs with the algorithm solutions to produce new solutions, and add those new solutions into the population of the algorithm.

The above-mentioned integration methods may improve the searching ability of the algorithm to some extent, but several problems may exist in performing them and which should be further studied. The problems are:

1. The AIs may be drowned in the algorithm population if the best algorithm solution is superior to that of the AIs;
2. The AIs may dominate the algorithm and make the algorithm fall into local search area if the best solution of the AIs is far more superior to that of the algorithm;
3. The two integration modes only work well for a short time and cannot perform a sustainable integration during the whole optimisation process;
4. Only few AIs integrate with the algorithm solutions, which is insufficient, and would be an 'unfair' integration. Population *vs* population co-operation and competition is expected between the AIs and the algorithm solutions.

The integration steps of the three kinds of solution are shown in Fig. 5. In order to avoid the AIs drowning into the algorithm's population and improve the efficiency of the integration of human and computer, this study constructed the AIs set to produce more AIs to co-operate or compete with the algorithm's population. The AIs and the DIs are added into the algorithm's population based on their fitness values or their diverse values instead of being added into the algorithm's population randomly to run their course. During the co-evolutionary process, the AIs set and the DIs set are updated continuously to perform a sustainable integration.

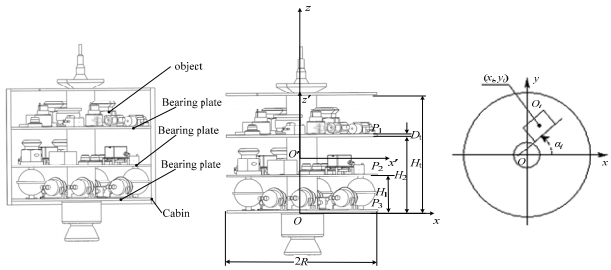


Figure 6. Schematic layout pattern of the simplified satellite module.

4.0 LAYOUT DESIGN INSTANCE OF A SATELLITE MODULE

4.1 Problem statement

In this study, the simplified satellite module is constructed based on the module of Ref. 24 as shown in Fig. 6. A total number of n objects are to be located within a cylindrical satellite module. The upper surfaces of three bearing plates attached on a standing column in the module are used to fix all objects. The basic problem is to design an optimal scheme until satisfying the following eight technical requirements[24]:

1. All objects should be contained within the module, with no overlapping among the objects and no clashing between the module wall and each object;
2. The inertia value of the whole module should be as small as possible;
3. The position error of the centroid of the whole system under the folded or deployed solar panel should be as small as possible;

https://doi.org/10.1017/S0001924000003663 Published online by Cambridge University Press

4. The equilibrium error of the system with the folded or deployed solar panel should be permissible, and the smaller the better;
5. The interference among objects should be as small as possible, the mutually interfered objects should be parted, and the farther away from each other the better;
6. Some objects should be located in the special area of the bearing plates;
7. All objects should have accessibility for the conveniences of installation and maintenance;
8. Some objects are expected the closer to each other the better. The seventh and eighth technical requirements come from Ref. 4.

Note that: the satellite module problem has the following special layout requirements:

1. the constraints or objectives to be satisfied or optimised are more complex, such as the accessibility requirement, the gathering requirement and the disturbing requirement;
2. some of the constraints or objectives cannot be precisely formulated, such as the accessibility requirement.

4.2 Mathematical optimisation model

Suppose that a set of objects to be located is $OBJ = \{Obj_1, Obj_2, \ldots, Obj_n\}$, where n = total number of objects. All objects are simplified as cuboids or cylinders in this work and regarded as rigid bodies with uniform mass distribution. They will be placed upright on the bases with serial numbers 1–3 as shown in Fig. 6. The ith object is denoted by Obj_i (p_i, Ψ_i, d_i), where $p_i = (x_i, y_i, z_i)^T \in R^3$ denotes the position of a reference point (the centroid of the object in this study) of Obj_i in the co-ordinate system $oxyz$; $\Psi_i = (\alpha_i, \beta_i, \chi_i)^T \in R^3$ indicates the orientation with respect to a global co-ordinate system and $di \in R^3$ = its dimensions. For a cuboid object, $d_i = (a_i, b_i, h_i)^T$, where a_i, b_i = lengths of the longer and shorter edges of its bottom, respectively, and h_i = its height. For a cylinder, $d_i = (r_i, 0, h_i)^T$, where r_i = radius of its bottom and h_i = its height. For convenience, the notation Obj_0 is used to denote the left fixed part of the module, including the shell, the bearing plates, and the standing column and so on. Generally, the masses and dimensions of all objects are given in advance, so p_i and $\Psi_i (i = 1, 2, ..., n)$ are variables to be manipulated in the following procedure. Thus, a general layout scheme of a satellite module can be formulated as

$$\boldsymbol{X} = \bigcup_{j=1}^{n} \{x_j, y_j, z_j, \alpha_j, \beta_j, \chi_j\} \quad \ldots (5)$$

In this study, all objects are to be located on three bearing surfaces. Each bearing surface is paralleled with the xoy plane of a global co-ordinate system, so the angles β_i, χ_i of Obj_i are zero. For a cuboid object, α_i = the angle between the longer edge of its bottom and the reference plane xoz and follows the right hand rule around axis z (with $\alpha_i \in (0, \pi)$). For a cylinder, $\alpha_i = 0$. Then a layout scheme of a satellite module in this study can be formulated as

$$\boldsymbol{X} = \bigcup_{j=1}^{n} \{x_j, y_j, z_j, \alpha_j\} \quad \ldots (6)$$

The mathematical optimisation model of whole layout problem can be formulated as follows:

Find a layout scheme $X \in R^{4n}$, such that

$$\begin{aligned} &\min_{X \in \boldsymbol{D}} ((I_{xx} - \Delta I_{xx})^2, (I_{yy} - \Delta I_{yy})^2, (I_{zz} - \Delta I_{zz})^2, T_G) \\ &\max_{X \in \boldsymbol{D}} (T_P, NObj) \end{aligned} \quad \ldots (7)$$

Subject to:

the overlapping constraints:

$$g_1(\boldsymbol{X}) = \sum_{j=0}^{n-1} \sum_{k=j+1}^{n} \Delta V_{jk} \le 0 \quad \ldots (8)$$

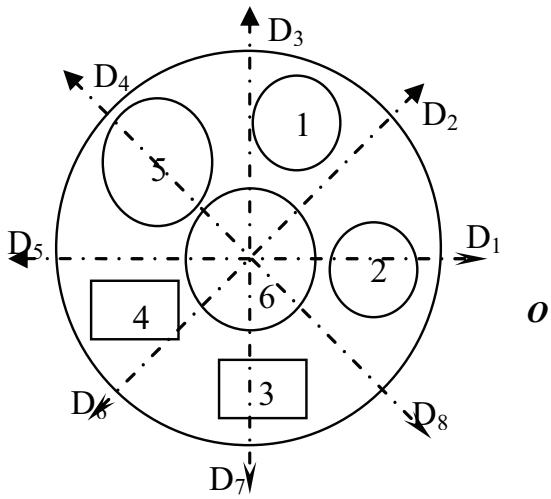


Figure 7. Discretisation description of the accessibility directions of an object.

the static balance constraints:

$$\begin{aligned} g_2(\boldsymbol{X}, \boldsymbol{U}^I) &= |x_m - x_e| - \delta x_e \le 0 \\ g_3(\boldsymbol{X}, \boldsymbol{U}^I) &= |y_m - y_e| - \delta y_e \le 0 \\ g_4(\boldsymbol{X}, \boldsymbol{U}^I) &= |z_m - z_e| - \delta z_e \le 0 \end{aligned} \quad \ldots (9)$$

the dynamic balance constraints:

$$\begin{aligned} g_5(\boldsymbol{X}, \boldsymbol{U}^I) &= |\theta_x| - \delta\theta_x \le 0 \\ g_6(\boldsymbol{X}, \boldsymbol{U}^I) &= |\theta_y| - \delta\theta_y \le 0 \\ g_7(\boldsymbol{X}, \boldsymbol{U}^I) &= |\theta_z| - \delta\theta_z \le 0 \end{aligned} \quad \ldots (10)$$

the special location constraints:

$$g_8(\boldsymbol{X}) = \{obj_i(\boldsymbol{X}) \in C_i\} \quad \ldots (11)$$

where $\boldsymbol{D}$ denotes the feasible region of variable X; $\mathbf{U^I} = \{\mathbf{I_E}, \mathbf{I_F}\}$ is a state vector that indicates the whole system state; $\mathbf{I_E}$ = the inertia matrix of whole system with the deployed solar panel; $\mathbf{I_F}$ = the inertia matrix of whole system with the folded solar panel; I_{xx}, I_{yy}, I_{zz} = the moments of inertia of the whole module with respect to inertial co-ordinate system $o_m x_m y_m z_m$; $\Delta I_{xx}, \Delta I_{yy}, \Delta I_{zz}$ = the engineering expected value of the inertia of the whole module; T_G = the sum of weight-distance product of the gathering objects which can be calculated by Equation (12); T_P = the sum of weight-distance product of the disturbing objects which can be calculated by Equation (13); d_{ij} = the distance between the ith object and the jth object; w_{ij} = the gathering weight between the ith object and the jth object. The larger this value is, the closer the objects should be; $\lambda_{ij} \in [0,10]$ = the disturbing weight between the ith object and the jth object. The larger this value is, the further away from each other the objects should be; C = the large constant; $Obj_i(X)$ indicates the location area of the ith object; $C_i = \{P_i, R_i\}$ indicates the permissible location area of the ith object which can be described as a circle location area with a radius R_i and a centre P_i; $NObj$ is the number of the accessibility directions of all objects. If the ith object satisfies the accessibility requirement, then the Equation (14) should be satisfied. In Equation (14), q is the accessibility direction of each object that can be determined by humans' experience; D^i_j indicates a segment that is between the centre of the ith object and the jth direction of the ith object. As shown in Fig. 7, the sixth object has eight accessibility directions (D_1~D_8). If the sixth circle object satisfies the accessibility requirement, then the sixth circle object should have at least one accessibility direction which means at least one segment among the segment set (D_1~D_8) cannot cross with other objects. Based on the above-mentioned

https://doi.org/10.1017/S0001924000003663 Published online by Cambridge University Press

analysis, the directions D_2, D_3, D_5, D_8 of the sixth circle object satisfy the accessibility requirement; $J(D^i_j, obj_k)$ indicates a judgmental mark that judge whether the segment D^i_j crosses with the object obj_k; ΔV_{jk} is the overlapping volume between Obj_j and Obj_k, $O_m(x_m, y_m, z_m)$ = the real centroid of the whole system and $O_e(x_e, y_e, z_e)$ = the expected position O_m; $(\delta x_e, \delta y_e, \delta z_e)$ = the allowable error; $(\theta_x, \theta_y, \theta_z)$ = the angles between the principal axis of inertia of the module and the reference axis ox, oy, oz along each direction, and $(\delta\theta_x, \delta\theta_y, \delta\theta_z)$ are the allowable errors of θ_x, θ_y, and θ_z, respectively.

$$T_G = \sum_{i=1}^{g-1} \sum_{j=i+1}^{g} w_{ij} d_{ij} \quad \ldots (12)$$

$$T_P = C - \sum_{i=1}^{P-1} \sum_{j=i+1}^{P} \lambda_{ij} d_{ij} \quad \ldots (13)$$

$$\forall k \in \{1, \cdots, n\} : (\bigcup_{j=1}^{q} \sum_{k=1, k\neq i}^{n} J(D^i_j, obj_k)) \neq 0 \quad \ldots (14)$$

In this paper, the Equations (5)~(10) in the mathematical model are used for Ref. 24.

4.3 Strategies for handling the constraints and the objectives

In this study, based on the CCEF, A GA is used to solve each layout sub-problem. GA needs scalar fitness information to work. Currently, based on the idea of aggregating functions, the methods of combining objectives into a single objective to construct the fitness function have been widely studied, which mainly include the weighted sum approach, and the goal programming method, etc. The strength of those methods lies in their efficiency and their easiness to be implemented in engineering practices. The weighted sum approach is used here to transfer the constraint mathematical optimisation model into an unconstrained mathematical optimisation model. The individual X_i of each subpopulation cannot be evaluated independently, because X_i is not a complete individual of the whole problem. Some co-operative solutions X_{ci} are obtained from other sub-problems. The individual X_i and co-operative solutions X_{ci} are combined into a complete individual X of the whole layout problem. Then Equation (15) is used to evaluate the whole individual X, and finally the fitness value of the whole individual X is regarded as the fitness value of the individual X_i of the subpopulation.

$$\begin{aligned} F(\boldsymbol{X_i}) &= \lambda_1 C_1 F_1(\boldsymbol{X}) + \lambda_2 C_2 F_2(\boldsymbol{X}) + \cdots + \lambda_k C_k F_k(\boldsymbol{X}) \\ \boldsymbol{X} &= \{\boldsymbol{X_{c1}}, \cdots, \boldsymbol{X_i}, \cdots, \boldsymbol{X_{cn}}\} \end{aligned} \quad \ldots (15)$$

where λ_1, λ_2, ..., λ_k are the weight factors of the objective functions that reflect the importance of the objectives; F_1, F_2 ..., F_k are the sub-objective set; C_1, C_2 ..., C_k are the normalising coefficients of the sub-objective functions.

In this study, according to the mathematical optimisation model of the satellite module layout problem, Equation (15) can be transformed into:

$$\begin{aligned} F(\boldsymbol{X_i}) = {} & \lambda_1 C_1 (I_{xx} - \Delta I_{xx})^2 + \lambda_2 C_2 (I_{yy} - \Delta I_{yy})^2 + \lambda_3 C_3 (I_{zz} - \Delta I_{zz})^2 + \lambda_4 C_4 T_G \\ & + \lambda_5 C_5 (C_{\max 1} - T_P) + \lambda_6 C_6 (C_{\max 2} - NObj) \\ & + \lambda_7 C_7 g_1(\boldsymbol{X}) + \lambda_8 C_8 (g_2(\boldsymbol{X}) + g_3(\boldsymbol{X}) + g_4(\boldsymbol{X})) \\ & + \lambda_9 C_9 (g_5(\boldsymbol{X}) + g_6(\boldsymbol{X}) + g_7(\boldsymbol{X})) + \lambda_{10} C_{10} g_8(\boldsymbol{X}) \end{aligned} \quad \ldots (16)$$

where C_{max1} and C_{max2} are large positive numbers to ensure that the values of the $(C_{max1} - T_P)$ and the $(C_{max2} - NOb_j)$ are larger than zero.

4.4 Numerical experiments

The data of this satellite module problem are given reference to Ref. 24, the number of objects and the bearing stand are decreased, but two performance constraints are added. Forty five objects need to be located on three bearing surfaces in the satellite module shown in Fig. 6. Twenty objects are cuboids and the rest are cylinders. The module is structured with the following parameters: R = 600mm, R_C = 120mm, H_1 = 300mm, H_2 = 630mm, H_t = 945·25mm, and D_t = 18·75mm. The mass of Obj_0 (including the shell, four bearing bases and the standing column) is m_0 = 776·53kg. The Obj_0 centres at C_0 (–0·978, 0·0085, 411·32)mm in co-ordinate system $oxyz$ with an inertia matrix $\mathbf{I_0}$ at the solar paddle deployed state, and centres at C_1(–0·9, 0·008, 409·96)mm in co-ordinate system $oxyz$ with an inertia matrix $\mathbf{I_1}$ at the solar paddle folded state. The technical requirements for a final layout scheme of the whole module are given as Equation (8)–(11) with parameters of x_c = 0, y_c = 0, z_c = 420, $x_c = \delta y_c = \delta z_c$ = 3mm; $\delta\theta_x = \delta\theta_y = \delta\theta_z$ = 0·03rad; each object has eight permissible accessibility directions. The basic problem is to optimise the objective function formulated by Equation (7) until satisfying the technical constraints given by Equation (8)–(11).

$$\mathbf{I_0} = \begin{pmatrix} I_x & I_{xy} & I_{xz} \\ I_{yx} & I_y & I_{yz} \\ I_{zx} & I_{zy} & I_z \end{pmatrix} = \begin{pmatrix} 324\cdot455 & -0\cdot004 & 0\cdot3776 \\ -0\cdot004 & 489\cdot738 & -0\cdot003 \\ 0\cdot3776 & -0\cdot003 & 366\cdot234 \end{pmatrix} Kg \cdot m^2 \quad \ldots (17)$$

The data of the objects are shown in Table 1. The assigned scheme of the objects in is shown in Table 2. The disturbing weights among the objects are shown in Table 3. The group weights among the objects are shown in Table 4. The dimensions and locations of the special objects are shown in Table 5.

The numerical experiments were run on an AT-compatible PC, with 1,700MHz Intel processor with 512Mb memory. Four methods were used to solve the satellite layout problem and each of them randomly ran 50 times. The four methods are a Genetic algorithm library Galib2.45 (GALib), a co-operative co-evolutionary genetic algorithm (CCGA), a human-algorithm-knowledge based layout designing method (HAKBLDM)[9] and the proposed HCCGA. The detailed parameters set for each method are shown in Table 6. Based on the formulation of Equation (17), an inertia matrix of the solar paddle folded state can be calculated. Note: the problem was solved successfully when the constraints were all satisfied. The sum of the inertia moment was smaller than 1,130Kg·mm^2. The value of T_G was smaller than 24,000,the value of T_P was greater than 28,000 and the accessible directions of all the objects were greater than or equal to 41.

4.5 Experimental results and analysis

The compared experimental results of the four methods are reported in Table 7. The optimal locations of the objects of the layout schemes solved by the four methods are listed in Table 8. The compared statistic data and the qualitative comparison results of the four methods are listed in Table 9 and Table 10, respectively. The layout patterns obtained by GALib, CCGA, HAKBLDM and HCCGA are illustrated as Figs 8, 9, 10 and 11, respectively. The 3D layout pattern of the whole module by HCCGA is simulated on the CAD platform Pro/Engineer and illustrated as Fig. 12. The fitness value profiles with respect to the function evaluations of the four methods are illustrated as Fig. 13.

Comparing the GALib and the CCGA on the objectives, the constraints, the success probability, and the runtime (Table 7):

- The sum of the lowest value and the average value of the inertia obtained by the proposed HCCGA are lower than those of the GALib, and higher than those of the CCGA;
- The best value and the average value of T_G obtained by the HCCGA are higher than those of the GALib, and lower than those of the CCGA;
- The best value and the average value of T_P obtained by the GALib are highest. It means that the optimal layout scheme obtained by GALib can make all disturbing objects far away from each other;

https://doi.org/10.1017/S0001924000003663 Published online by Cambridge University Press

Table 1
Data of the objects

Obj No	Mass (kg)	Dimension(mm) a_i	b_i	h_i	Obj No	Mass (kg)	Dimension(mm) $a_i(r_i)$	b_i	h_i	Obj No	Mass (kg)	Dimension(mm) r_i	h_i
1	28·13	250·00	150·00	250·00	16	5·40	150·00	100·00	120·00	31	8·48	75·00	160·00
2	28·13	250·00	150·00	250·00	17	5·40	150·00	100·00	120·00	32	8·48	75·00	160·00
3	28·13	250·00	150·00	250·00	18	5·40	150·00	100·00	120·00	33	7·95	75·00	150·00
4	28·13	250·00	150·00	250·00	19	5·40	150·00	100·00	120·00	34	7·95	75·00	150·00
5	28·13	250·00	150·00	250·00	20	23·56	150·00	100·00	250·00	35	7·95	75·00	150·00
6	28·13	250.00	150·00	250·00	21	23·56	100·00		250·00	36	7·95	75·00	150·00
7	28·13	250·00	150·00	250·00	22	23·56	100·00		250·00	37	7·95	75·00	150·00
8	28·13	250·00	150·00	250·00	23	23·56	100·00		250·00	38	7·95	75.00	150·00
9	19·20	200·00	160·00	200·00	24	23·56	100·00		250·00	39	5·09	60·00	150·00
10	19·20	200·00	160·00	200·00	25	23·56	100·00		250·00	40	5·09	60·00	150·00
11	19·20	200·00	160·00	200·00	26	23·56	100·00		250·00	41	5·09	60·00	150·00
12	15·36	160·00	120·00	250·00	27	23·56	100·00		250·00	42	5·09	60·00	150·00
13	15·36	160·00	120·00	250·00	28	18·85	100·00		200·00	43	5·09	60·00	150·00
14	8·64	160·00	120·00	150·00	29	18·85	100·00		200·00	44	5·09	60·00	150·00
15	8·64	160·00	120·00	150·00	30	18·85	100·00		200·00	45	5·09	60·00	150·00

Table 2
Assign scheme of objects

Obj No	Located number	Obj No.	Located number	Obj No	Located number	Obj No	Located number	Obj No	Located number
1	3	11	2	21	3	31	1	41	1
2	3	12	1	22	2	32	3	42	1
3	3	13	1	23	1	33	3	43	3
4	2	14	1	24	3	34	3	44	2
5	2	15	3	25	1	35	2	45	2
6	2	16	1	26	1	36	2		
7	3	17	1	27	1	37	2		
8	3	18	2	28	1	38	3		
9	1	19	3	29	3	39	3		
10	2	20	2	30	1	40	2		

Table 3
The disturbing weights among the objects

Obj No.	3	6	13	15	17	21	30	33	37
3	0	0	0	5	0	7	0	8	0
6	0	0	0	0	0	0	0	0	4
13	0	0	0	0	8	0	3	0	0
15	5	0	0	0	0	6	0	9	0
17	0	0	8	0	0	0	2	0	0
21	7	0	0	6	0	0	0	4	0
30	0	0	3	0	2	0	0	0	0
33	8	0	0	9	0	4	0	0	0
37	0	4	0	0	0	0	0	0	0

Table 4
The group weights among the objects

Obj No.	8	11	16	21	22	26	30	32	40
8	0	0	0	4	0	0	0	6	0
11	0	0	0	0	5	0	0	0	8
16	0	0	0	0	0	3	9	0	0
21	4	0	0	0	0	0	0	4	0
22	0	5	0	0	0	0	0	0	7
26	0	0	3	0	0	0	5	0	0
30	0	0	9	0	0	5	0	0	0
32	6	0	0	4	0	0	0	0	0
40	0	8	0	0	7	0	0	0	0

https://doi.org/10.1017/S0001924000003663 Published online by Cambridge University Press

Table 5
Dimensions and locations of the special objects

Obj No	Location (mm)		Dimension (mm)		Located number	Obj No	Location (mm)		Dimension (mm)	Located number
	x	y	$a_i(r_i)$	b_i			x	y	r_i	
2	0·000	450·000	300·000	200·000	3	34	–250·000	0·000	120·000	3
9	0·000	–450·000	300·000	200·000	1	35	0·000	–250·000	120·000	2
27	250·000	0·000	120·000		1	43	0·000	–250·000	120·000	3
28	0·000	250·000	120·000		1	44	0·000	250·000	120·000	2

Table 6
Parameter setting of the four methods

Methods	GALib	CCGA	HCCGA	HAKBLDM
Parameters Setting	Real code, the maximum generations is 10,000, the population size is 600, two-point crossover, uniform mutation operation, linear ranking selection; mutation probability $P_m = 0{\cdot}1$, cross probability $P_c = 0{\cdot}9$.	Real code, the maximum generations is 10,000, each subpopulation size is 200, two-point crossover, Gaussian mutation operation, linear ranking selection, the sub problem size is 3, $P_m = 0{\cdot}1$, $P_c = 0{\cdot}9$.	The algorithm parameter setting is same to CCGA, the artificial solutions size is 15, the diverse solutions size is 15, the large constant C = 50,000.	The algorithm parameter setting is same as GALib, the artificial solutions size is 15.

Table 7
The obtained technical indices by the four methods

Methods		GALib	CCGA	HAKBLDM	HCCGA
The inertia moment of the folded solar panel (Kg·mm^2)	$I_{x'}$	388·322	373·401	380·265	385·986
	$I_{y'}$	408·358	417·574	413·754	407·439
	$I_{z'}$	331·606	325·902	329·824	328·752
The inertia angle of the folded solar panel (rad)	$\theta_{x'}$	0·002	0·001	0·02	0·017
	$\theta_{y'}$	0·018	0·005	0·007	0·002
	$\theta_{z'}$	0·01	0·000	0·005	0·006
The centroid of the folded solar panel (mm)	δx_c	0·004	0·6	0·46	0·653
	δy_c	0·07	0·001	0·43	0·407
The object area beyond the fixed area (mm^3)		0·000	0·000	0·000	0·000
Tg	T_G	20,925·8	21,329·2	16,057·38	20,981·0
TP	T_P	41,227·1	28,018·60	31,935·65	35,017·1
The accessibility directions of all objects		41	42	45	45
DV(mm^3)	ΔV(mm^3)	0·000	0·000	0·000	0·000
the running time (s)		1,024·030	831·500	1,108	1200·360
The success probability (%)		25	48	90	95

Note: the programme GALib is obtained from the free source code on the internet. The programme CCGA, HAKBLDM and HCCGA are developed and run by the first author.

https://doi.org/10.1017/S0001924000003663 Published online by Cambridge University Press

Table 8
The optimal locations of the objects obtained by the four methods

Obj. No.	GALib		CCGA		HAKBLDM		HCCGA	
	x(mm)	*y*(mm)	*x*(mm)	*y*(mm)	*x*(mm)	*y*(mm)	*x*(mm)	*y*(mm)
1	–371·87	–179·61	207·10	–158·22	340·98	143·38	248·42	–60·29
2	0·92	425·00	2·65	441·78	–3·70	427·13	–11·77	431·74
3	411·59	–144·28	245·00	17·25	343·35	–20·09	397·17	119·07
4	–253·53	171·97	–228·99	142·56	233·32	168·85	258·49	–208·35
5	133·19	195·33	–245·56	–21·65	342·33	–283·86	–287.73	–97.32
6	389·22	231·98	–209·86	–218·59	–244·54	–135·89	–14·50	379·49
7	–23·41	–419·42	31·29	–351·84	–262·14	–303·30	–227·13	–186·06
8	–270·62	341·63	–441·90	63·04	–291·73	336·39	–359·99	261·54
9	–41·28	–430·16	–5·16	–436·27	–45·89	–429·18	–12·22	–451·92
10	79·77	395·07	288·13	81·45	253·64	–105·90	225·98	-26·33
11	–130·38	–392·01	271·96	–154·53	–307·47	128·77	-267·08	335·74
12	292·06	–380·59	145·70	–194·05	295·73	184·77	-189·04	274·03
13	–288·54	–350·21	180·91	413·70	–456·78	100·35	-185·19	122·78
14	420·05	76·73	–499·19	–18·36	493·74	–55·10	195·12	401·84
15	385·48	–320·43	479·09	114·99	–155·60	156·84	192·05	–436·45
16	201·70	–154·69	221·45	–349·07	112·71	455·01	448·13	168·85
17	141·73	477·51	461·33	104·27	447·83	-203·84	–170·67	437·16
18	216·45	53·84	–322·81	308·38	–244·97	–434·78	135·76	–370·05
19	–140·89	152·45	–405·80	–97·29	77·44	–408·26	37·11	215·10
20	375·19	94·11	–498·28	103·27	290·44	31·71	–154·23	–237·83
21	–331·03	158·12	273·63	241·08	–438·08	–123·82	–477·03	–83·25
22	–217·00	–43·00	488·34	–41·17	–483·97	–89·08	–299·19	–392·70
23	–28·04	–224·20	–108·94	–197·59	–261·96	–202·66	111·60	–204·67
24	374·41	77·29	479·54	-91·91	399·61	–230·34	413·63	–248·97
25	–256·92	256·17	230·52	224·53	19·73	–227·71	–102·02	–215·40
26	–425·12	131·32	–223·36	140·22	–240·90	15·16	–350·51	7·30
27	230·35	2·48	246·45	10·17	232·88	15·29	250·49	7·26
28	3·35	230·37	–9·52	244·73	0·98	229·69	12·31	235·35
29	179·47	–269·86	–137·94	195·56	100·12	223·50	360·79	319·54
30	289·11	235·22	–301·10	-65·32	–150·13	411·61	266·10	214·70
31	–290·27	–78·49	402·61	–138·36	327·01	324·72	299·00	–183·07
32	–266·98	–369·92	–417·78	–270·44	–385·97	177·27	–143·87	210·83
33	–480·00	39·87	–236·02	–176·35	194·07	–189·04	–354·57	–368·63
34	–286·58	–21·62	–221·71	–0·60	–231·42	–23·64	–216·34	16·85
35	1·79	–205·67	27·08	–213·45	–0·54	–220·46	9·64	–230·46
36	317·38	–242·39	–475·13	–197·27	–203·91	–291·13	205.82	198·36
37	237·02	–393·64	379.30	265·38	–100·94	329·25	472·21	–187·20
38	–160·48	–122·95	–416·86	228·11	–121·47	–481·91	141·47	–222.27
39	214·47	378·82	–235·29	–367·85	320·52	407·83	–92·49	–466·04
40	–151·57	–235·90	451·19	–265·48	–137·74	140·35	–215·64	153·39
41	–147·89	120·27	–84·70	436·27	502·26	139·89	448·92	–178·67
42	137·01	–327·01	–458·63	–182·04	–185·31	200·34	–396·05	314·48
43	–28·40	–213·58	–8·73	–198·03	46·43	–236·04	19·41	–304·07
44	–59·31	249·05	–7·58	222·94	13·07	197·56	16·95	224·69
45	–329·04	–340·29	102·49	–402·56	154·35	–487·90	347·31	303·57

- The HCCGA can produce the best solutions with the highest accessibility directions that can make the objects easily to be modified or assembled;
- The proposed HCCGA needs the longest runtime, and CCGA cost the shortest runtime;
- The HCCGA keeps a higher success probability, and the GALib has the lowest success probability.

Compared with the HAKBLDM (Table 7), the proposed HCCGA can produce better solutions with better values of objectives and constraints, and can keep a higher success probability. But it needs a longer runtime.

The above-mentioned quantitative comparison of the computational performance shows that the HCCGA can guide the evolutionary algorithm more efficiently, and can solve the layout problem more in a stable way by adding the AIs and the DIs into the algorithm's population during the whole optimisation. It also can be concluded that some complex nonlinear objectives or constraints (such as the inertia moment and the inertia angle requirements) cannot be fully solved only by engineers. Solving those objectives or constraints efficiently needs the computational capability and the logical reasoning capability of the algorithm, and the values of those objectives or constraints of the optimal solution obtained by the CCGA can reach a higher precision.

As shown in Figs 8-11 the word 'grouping' means the locations of

https://doi.org/10.1017/S0001924000003663 Published online by Cambridge University Press

Table 9
Comparison of the statistic data of the four methods

Methods		GALib	CCGA	HAKBLDM	HCCGA
In the folded solar panel	$I_x + I_y + I_z$ (Kg·m^2)	(1,138·571, 28·216, 1,103·861,1,145·472)	(1,118·265, 25·456, 1,114·264, 1,123·564)	(1,125·236, 30·265, 1,120·357, 1,135·648)	(1121·264, 35·456, 1118·268, 1130·462)
	$\lvert\theta_x\rvert + \lvert\theta_y\rvert + \lvert\theta_z\rvert$ (rad)	(0·090, 0·181, 0·001, 0·652)	(0·001, 0·025, 0·000, 0·043)	(0·0072, 0·083, 0·000, 0·053)	(0·006, 0·064, 0·001, 0·265)
	$\delta x_c + \delta y_c$ (mm)	(0·007, 0·070, 0·000, 0·044)	(0·131, 4·265, 0·000, 3·756)	(1·631, 10·325, 0·05, 2·354)	(1·032, 8·436, 0·036, 2·860)
The runtime (s)		(1,040·950, 196·430, 993·500, 1,118·750)	(831·043, 54·428, 821·594, 852·953)	(1218·340, 215·189, 1,015·6, 1549·36)	(1226·350, 105·169, 925·360, 1389·345)
The collision area in xOy plane (mm^2)		(101·406, 2,904·130, 0·000, 2130·480)	(10·933, 44·7405, 0·000, 19·832)	(0·566, 10·39, 0·000, 40·125)	(2·536, 4·369, 0·000, 10·369)
The object area beyond the fixed area (mm^3)		(24·879, 495·008, 0·000, 423·233)	(1·603, 20·914, 0·000, 9·790)	(3·254, 2·39, 0·000, 13·215)	(1·569, 1·369, 0·000, 3·269)
T_G		(26,130·800, 28,270·400, 12,269·900, 34,290·300)	(22,605·700, 7,879·630, 21,008·300, 25,310·700)	(2,0216·150, 8,576·330, 17,953·369, 25,663·321)	(20, 361·560, 6,536·370, 19,653·250, 28,693·210)
T_P		(31,151·500, 32,378·200, 21,987·100, 43,146·700)	(28,358·700, 4,767·540, 27,283·400, 29,983·700)	(30,154·350, 15276·600, 28,363·456, 40134·569)	(32,653·260, 10,236·600, 27,563·360, 42,138·970)
The accessibility directions of all objects		(40·5, 8·65, 39, 42)	(41, 5.4762, 40, 42)	(45, 0, 45, 45)	(45, 0, 45, 45)

Table 10
Qualitative comparison of the results of four methods

Methods	The inertia moment	T_G	T_P	The accessibility directions	The runtime	The success probability
GALib	worst	average	best	worst	average	worst
CCGA	best	worst	worst	average	best	average
HAKBLDM	average	best	average	best	average	average
HCCGA	average	average	average	best	average	best

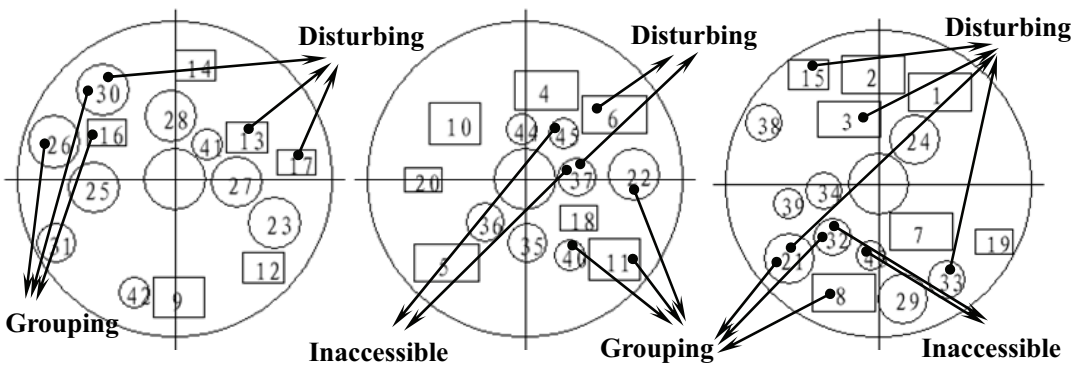


Figure 8. The best layout pattern obtained by the GALib.

https://doi.org/10.1017/S0001924000003663 Published online by Cambridge University Press

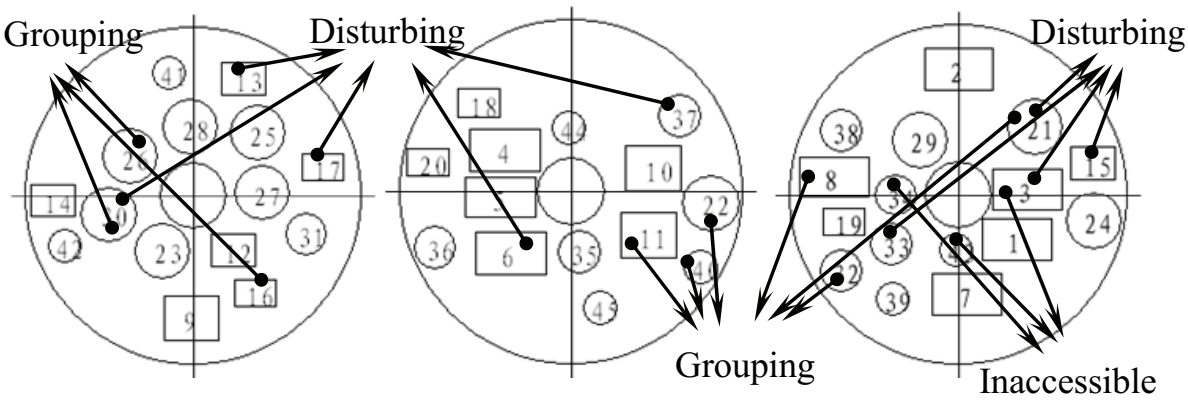


Figure 9. The best layout pattern obtained by the CCGA.

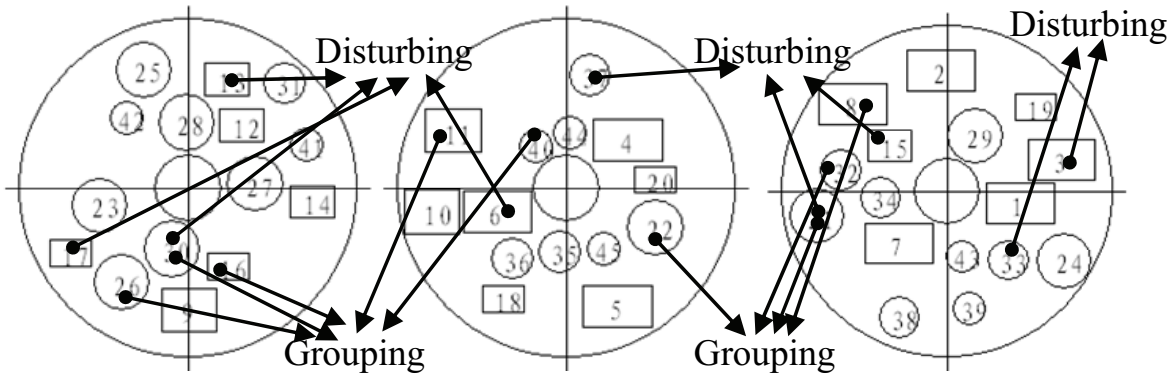


Figure 10. The best layout pattern obtained by the HAKBLDM.

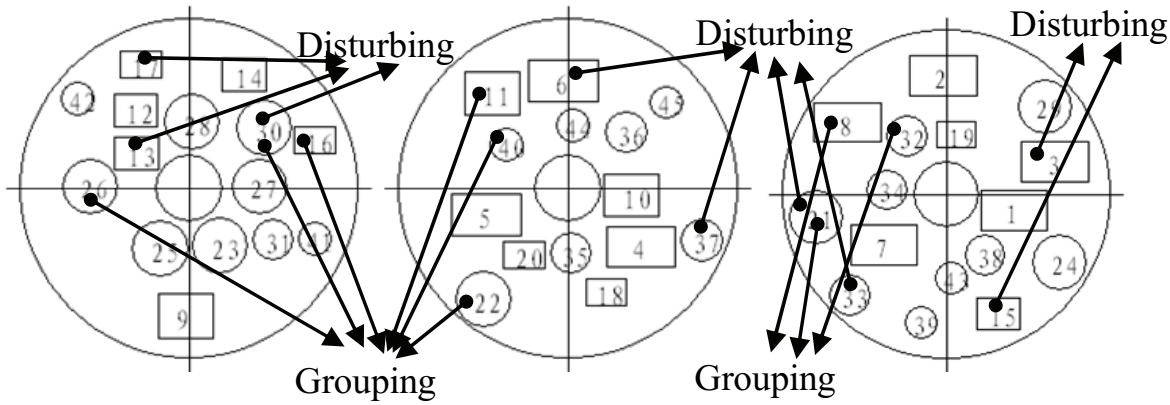


Figure 11. The best layout pattern obtained by the HCCGA.

https://doi.org/10.1017/S0001924000003663 Published online by Cambridge University Press

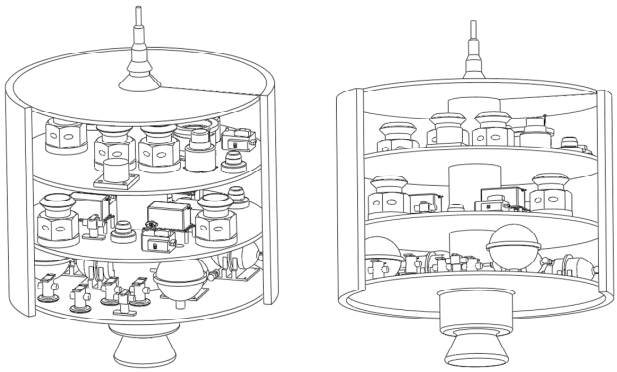
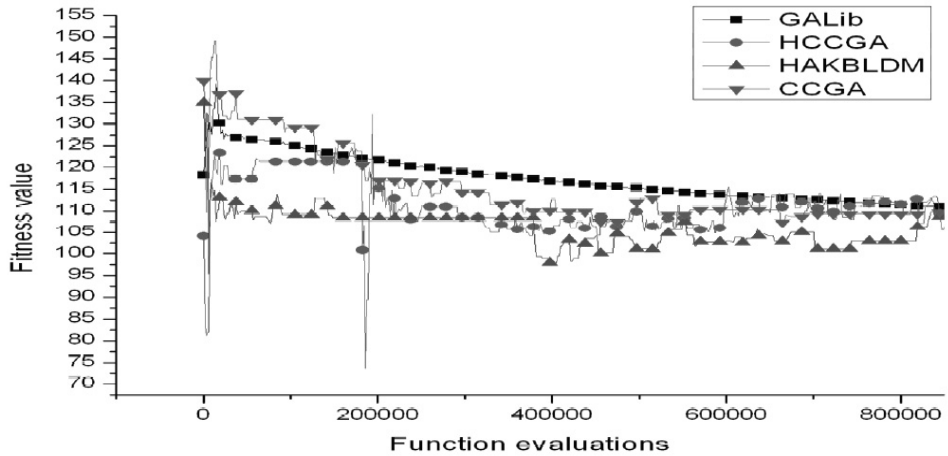


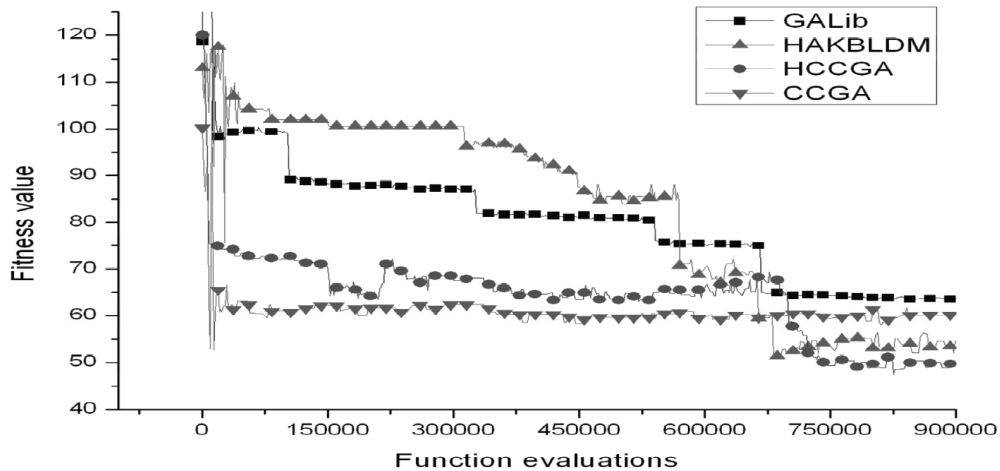
Figure 12. The 3D layout pattern of the whole module obtained by the HCCGA.

the objectives are expected to be the nearer the better; the word 'disturbing' means the locations of the objectives that are expected to be the father away from each other the better; and the word 'inaccessible' denotes the objectives that have no accessibility which should be avoided since this would make the installation and maintenance of these equipments very hard. Comparing the above-mentioned four best layout patterns, the best layout pattern obtained by the HCCGA can tackle some objectives or constraints which cannot be formulated precisely. For example, for the accessibility direction requirements, the HCCGA can obtain many good schemes in which the objects' layout pattern is easy to be modified or assembled.

As shown in Fig. 13 of the fitness curves, the HCCGA can keep a sustained oscillation state during the whole optimisation process, which can produce many different potential layout pattern schemes, and make the algorithm skip out from local convergence area, and then improve the searching stability of the algorithm. During the optimisation process, adding the AIs into the population can guide



(a)The average fitness curve of the four methods



(b) The best fitness curve of the four methods

Figure 13. The fitness value profiles with respect to function evaluations of the four methods.

https://doi.org/10.1017/S0001924000003663 Published online by Cambridge University Press

the algorithm in accordance with the engineer's intention. To avoid the algorithm falling into a local searching area, adding some DIs into the population in real-time can provide many potential solutions to make the algorithm keep the sustained searching ability. It can be seen that the HAKBLDM can also keep the sustained oscillation state during the whole optimisation process. In the HAKBLDM, the co-operation process of crossing several AIs with the algorithm solutions or replacing the algorithm solutions randomly with the AIs during the optimisation can also provide many potential solutions and improve the searching stability of the algorithm.

As shown in Table 10, the four methods are compared based on six aspects. Each aspect has three evaluation grades: 'best', 'average' and 'worst', respectively. 1. for the GALib, one aspect (T_P) is evaluated as the 'best' grade, two aspects (the T_G and the runtime) are evaluated as the 'average' grade, and the remaining three aspects (the inertia moment, the accessibility direction and the success probability) are evaluated as the 'worst' grade. 2. for the CCGA, two aspects (the inertia moment and the runtime) are evaluated as the 'best' grade, three aspects (the T_G, the accessibility direction and the success probability) are evaluated as the 'average' grade, the remaining one aspect (T_P) are evaluated as the 'worst' grade. 3. for the HAKBLDM, two aspects (the T_G and the accessibility direction) are evaluated as the 'best' grade, four aspects (the T_P, the inertia moment, the success probability and the runtime) are evaluated as the 'average' grade, and no aspect is evaluated as the 'worst' grade. 4. for the HCCGA, two aspects (the accessibility direction and the success probability) are evaluated as the 'best' grade, four aspects (the T_P, the T_G and the inertia moment, the runtime) are evaluated as the 'average' grade, and no aspect is evaluated as the 'worst' grade.

It can be concluded that the HAKBLDM and the HCCGA can both obtain better solutions than the other two algorithms. The HCCGA is superior to the HAKBLDM in success probability, because the CCGA is employed in the HCCGA, and GA is adopted in the HAKBLDM. The CCGA is more efficient than GA in solving the complex layout problems. Furthermore, the HCCGA adopts the diversity extended mechanism to provide many potential solutions, so the HCCGA is superior to the HAKBLDM in the computational precision and the success probability.

For the satellite module layout problem, the objectives and the constraints are in conflict with each other, and some of them even cannot be formulated precisely. This makes it impossible for the problem to be fully solved only by the algorithm. Engineers' experience and intuition can better balance those conflicting objectives and constraints, especially for some visual objectives and constraints, such as the overlapping constraints, the grouping requirements and the disturbing requirements of the objects. As shown in Table 7 and Table 10, the HCCGA and the HAKBLDM can both better balance those objectives and constraints efficiently than the other two algorithms.

Based on the data in Table 10, the CCGA and the HCCGA show a complementary relation. The HCCGA can tackle some objectives or constraints efficiently which are hard to be solved by the CCGA, such as the disturbing requirements, the accessibility direction requirements and the computational stability of the algorithm, and vice versa. The above-mentioned experiments show that humans and algorithm are complementary with each other, especially in solving the complex layout problems. This complementary advantage is very obvious. It is also shown that only when humans integrate with computer efficiently, can their advantages be fully utilised and can the humans' burden of solving the complex layout problems be reduced greatly.

5.0 CONCLUSIONS

The modern engineers' design practices show that it is an effective way to study the integration of 'human intelligence' with 'computer algorithm' to solve complex engineering layout problems. Based on the previous research works of satellite module layout design of references[4,9,21,24], a human–computer co-operative co-evolutionary method is developed to solve a satellite module layout problem. In this method, the diversity reference set is constructed by the DIs using the combinatorial operators of DE and the BLX-α to improve the integration of the human experience and the computer. During the co-evolution of the sub-problems, the AIs, the DIs and the algorithm solutions are expressed by unified encoding strings and co-operated together to create new co-operative solutions. Experimental results show that:

1. Compared with the integration of the AIs and the algorithm solutions, adding the DIs into the population of the human–computer co-operative method can exploit the advantages of human intelligence and computer algorithm and solve the complex layout problem of the satellite module more efficiently.
2. Compared with the CCGA and the GALib, the HCCGA can better balance the multiple objectives or constraints, especially for some visual and unmodeled objectives, and can keep a more stable success probability.

In this study, only a preliminary research has been made. The proposed method only demonstrates its efficiency on the finite satellite module layout problem. It is too soon to draw general conclusions. There are several questions that have not been fully solved like:

1. General principles of the interactive optimisation and the essence of the engineering optimisation problem should be further studied to further explore the advantages of the integration of human intelligence and computer. Further work should be focused on solving other complex engineering problems to obtain a general conclusion to guide the engineering design.
2. Commercial 3D-CAD software can aid engineers to design on multi-levels, and it may be used as an efficient tool to tackle the practical engineering problems and can integrate the visibility and perceptivity of engineers in the optimisation computation more efficiently.

ACKNOWLEDGEMENTS

This work was supported by the National Natural Science Foundation of China (Granted No. 50575031,60674078), the Major State Basic Research Development Program of China (973 Program) (Granted No.2007CB714006), the National Key Technology R&D Program (Granted No.2007BAF09B01), the Liaoning Key Scientific and Technological Project (Granted No.2008220017), and the China Postdoctoral Science Foundation (Granted No.20080431136).

REFERENCES

1. SUN, Z.G., TENG, H.F. and LIU, Z. W. Several key problems in automatic layout design of spacecraft modules, *Chinese J Prog in Natural Science*, 2003, **13**, (11), pp 801-808.
2. BRAUN, R.D., MOORE, A.A. and KROO, I.M. Collaborative approach to launch vehicle design, *J Spacecraft and Rockets*, 1997, **34**, (4), pp 478-486.
3. KAMRAN, D., MAZIAR, A. and HOSSEIN, S.F. 'FARAGAM' algorithm in satellite layout, 2001, Proceedings of Sixth Asia-Pacific Conference on Multilateral Co-operation in Space Technology and Application. Beijing, China, pp 120-127.
4. GRIGNON, P.M. and FADEL, G.M. A GA-based configuration design optimization method, *J Mechanical Design*, 2004, **126**, pp 6-15.
5. FEREBEE, M.J. and ALLEN, C.L. Optimization of payload placement on an arbitrary spacecraft, *J Spacecraft and Rocket*s, 1991, **28**, (5), pp 612-614.

https://doi.org/10.1017/S0001924000003663 Published online by Cambridge University Press

6. TANNER, S. and FENNEL, R. The placement of equipment in the space station freedom using constraint based reasoning, 1991, Proceedings AAAI Conference on Innovative Applications of AI, Anaheim, USA, pp 51-71.
7. CAGAN, J., DEGENTESH, D. and YIN, S. A simulated annealing-based algorithm using hierarchical models for general three-dimensional component layout, *Computer-Aided Design*, 1998, **30**, (10), pp 781-790.
8. QIAN, Z.Q., TENG, H.F. and SUN, Z.G. Human-computer interactive genetic algorithm and its application to constrained layout optimization, *Chinese Journal of Computers* (in Chinese), 2001, **24**, (5), pp 299-302.
9. LIU, Z.W. and TENG, H.F. Human-computer co-operative layout design method and its application, *Computers and Industrial Engineering*, 2008, **55**, pp 735-757.
10. SUN, W.J. and SCCHEN, C. Efficient and effective placement for very large circuits, 1995, IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems, **14**, (3), pp 349-359.
11. LENAT, D.B. and FEIGENBAUM, E.A. On the thresholds of knowledge, 1988, Proceedings of International Workshop on Artificial Intelligence for Industrial Applications, Oakland, pp 291-300.
12. POTTER, M.A and DE JONG, K.A. A co-operative coevolutionary approach to function optimization, 1994, Proceedings of the Third Conference on Parallel Problem Solving from Nature. Jerusalem, Israel, pp 249-257.
13. SUNG-BAE, C. Towards creative evolutionary systems with interactive algorithm, *Applied Intelligence*, 2002, **16**, (1), pp 129-138.
14. KOSORUKOFF, A. Human based genetic algorithm, 2001, IEEE International Conference on Systems, Man and Cybernetics, Tucson, pp 3464-3469.
15. MURTHY, S., AKKIRAJU, R. and RACHLIN, J. *ET AL*. Agent-based co-operative scheduling, 1997, Proceedings of AAAI Workshop on Constraints and Agents, USA, pp 112-117.
16. LIU, J. and TENG, H. F. Human-genetic algorithm co-operation and its interface, 2002, Proceedings of Fifth Asia Pacific Conference on Computer Human Interaction (APCHI), Beijing, China, pp. 378-387.
17. BABU, B.V. and MUNAWAR, S.A. Differential Evolution for the Optimal Design of Heat Exchangers, 2000, Proceedings of All India Seminar on Chemical Engineering Progress on Resource Development: A Vision 2010 and Beyond.
18. LOZANO, M., HERRERA, F. and VERDEGAY, J.L. Tackling real-coded genetic algorithms: Operators and tools for the behavioral analysis, *Artificial Intelligence Reviews*, 1998, **12**, (4), pp 265–319.
19. NOMURA, T., SHIMOHARA, K. An analysis of two-parent recombinations for real-valued chromosomes in an infinite population, *Evolutionary Computation J*, 2001, **9**, (3), pp 283-308.
20. HUO, J. Z. and TENG, H. F., Optimal layout design of a satellite module using a coevolutionary method with heuristic rules, *J Aerospace Eng*, 2009, **22**, (2), pp 101-111.
21. SHI, Y.J. The Co-operative Co-Evolutionary Differential Evolution and Its Applications for Complex Layout Optimization, 2005, PhD dissertation (in Chinese), Dalian University of Technology, China.
22. SUN, Z.G. Sequential and Physical Decomposition Methods for a Layout Design Problem of Spacecrafts, 2005, PhD dissertation (in Chinese), Dalian University of Technology, China.
23. SHI, Y.J., TENG, H.F. and LI, Z.Q. Co-operative co-evolutionary differential evolution for function optimization, *Lecture Notes in Computer Science*, 2005, **3611**, pp 1080-1088.
24. SUN, Z.G., TENG, H.F. and LIU, D.Q. Optimal layout design of a satellite module, *Engineering Optimization*, 2003, **35**, (6), pp 513-529

https://doi.org/10.1017/S0001924000003663 Published online by Cambridge University Press