Springer

Journal of
Mechanical
Science and
Technology

Journal of Mechanical Science and Technology 22 (2008) 628~638

www.springerlink.com/content/1738-494x

Concept optimization for mechanical product by using ant colony system

Ruifeng Bo*, Ruiqin Li and Hongxia Pan

Department of Mechanical Engineering, North University of China, Taiyuan, 030051, PR China

(Manuscript Received March 2, 2007; Revised October 18, 2007; Accepted October 18, 2007)

Abstract

The aim of conceptual design is to generate the best design candidate. Concept solving in conceptual design can be viewed as a problem of combinatorial optimization, in which there exists a "combinational explosion" phenomenon when using the traditional morphological matrix method to tackle it. In this research, a concept optimization problem is studied based on an Ant Colony System (ACS). By analyzing the similarity between concept solving and Traveling Salesman Problem (TSP), concept solving is transformed into a problem of optimal path in combinatorial optimization, where the dynamic programming based solution space model and the longest path based optimization model are developed. Then, the ant algorithm to resolve TSP is adopted to implement concept optimization according to the positive feedback searching mechanism of ACS, and some improvements are made incorporating crossover and mutation operators of a genetic algorithm (GA), to obtain the optimal scheme rapidly and effectively. Finally, a conceptual design case of press is given to demonstrate the feasibility and rationality of this proposed approach. The employment of ACS enables concept solving to be implemented with an algorithm and thus possesses better operability, which offers a promising way to solve the "combinatorial explosion" problem in conceptual design.

Keywords: Ant colony system; Concept optimization; Conceptual design; Traveling salesman problem; Combinational explosion

1. Introduction

The development of a mechanical product undergoes a sequence of processes including conceptual design, embodiment design, detailed design, production process planning, manufacturing, assembly, and so on [1]. Conceptual design is the first step in the overall process of product development, in which the best concept satisfying design requirements is required to be developed for further refinement in the latter design stages using a certain method or tool, which is called concept solving. The concept solving problem has a profound impact on the success of product design, because a poor concept solution can rarely be compensated for at later design stages and incurs great expense of redesign cost. For a mechanical product, its performance, quality, cost, and market to time are mainly embodied through the design concept. It has been recognized that nearly 75% of product life-cycle cost is committed by the end of conceptual design [2].

At present, in the field of conceptual design, approaches of principle synthesis are widely used to tackle the problem of concept solving. Zou [3] introduced morphology matrix incorporating a function carrier database to generate a number of mechanism systems on the basis of function analysis. Hoeltzel et al. [4] suggested knowledge–based approaches for the creative design of a mechanism, in which a morphology matrix was used to implement the creative synthesis of the mechanism. Ermer et al. [5] utilized bond-graphs and function-based models to develop a CAD system for conceptual design in engineering. Yao and Huang [6] proposed a "three steps" structure model of design catalogues for conceptual design of a mechanical transmission and applied it in the intelligent design system. Ng and Leng [7] investigated the

*Corresponding author. Tel.: +86 351 3922106
E-mail address: brfcxh@163.com
DOI 10.1007/s12206-007-1047-0

feasibility of automating the conceptual design of a micro-air vehicle on a personal computer system. The proposed design methodology uses GA as the search engine in the design process. Sun and Kalenchuk [8] proposed a method for design candidate evaluation and identification using neural network-based fuzzy reasoning, in which a modeling relation between design specifications and customer requirements was built by using a multi-layer feed-forward neural network, and the satisfactory conceptual design candidate is retrieved by fuzzy reasoning from requirements. Shu and Hao [9] proposed a CAD system model based on GA to improve design efficiency and accuracy for mechanical products. Huang and Bo [10] adopted GA to tackle the concept optimization problem in conceptual design of a mechanical product. Huang and Bo [11] employed GA incorporating fuzzy neural network as a crucial technology to solve two problems of concept generation and evaluation.

Most of the above approaches have used a kind of tool such as morphology matrix, bond graph, or design catalogues incorporating an expert system to deal with the concept generation, in which concepts are usually combined and enumerated through function analysis. However, when the solution space possesses a large dimension, a large number of generated concepts will inevitably result in "combination explosion" phenomenon. Thus it is difficult to evaluate these candidates one by one to obtain the best concept using a regular algorithm. GA is a simulated evolutionary algorithm that is able to resolve combinatorial optimization problems. But it is apt to update the good individuals entirely and cannot exploit the characteristics of the concept solution space, which will result in a low convergent rate or local convergence. Currently, there is a necessity to search for an effective method or tool to tackle concept solving in conceptual design. How to find the optimal solution by minimizing the searching space and controlling the searching process adaptively according to certain rules is the key of conceptual solving.

The Ant Colony system (ACS) is a novel heuristic simulated evolutionary algorithm developed by Dorigo and Maniezzo [12]. ACS simulates the population behavior of ant colonies in nature when they are foraging for food and finding the most efficient routes from their nests to food sources. By means of the positive feedback searching mechanism of ant colonies, ACS has been successfully applied to a great many combinatorial optimization problems that are NP-hard, such as the Traveling Salesman Problem (TSP) [13], Flow-shop scheduling [14], and so on. Concept solving in conceptual design is a problem of combinational optimization. At the same time, ACS can search the whole solution space through a mode of structuring a combinational solution, considering the partial and global performance of the solution, which is consistent with the combinational principle of concept solution in concept design. Thus, theoretically speaking, it is feasible to apply ACS to the process of concept solving.

ACS is a new method to solve combinational optimization. But in order to make ACS become a practical tool to solve concept solving, the associated researches on concept optimization are required to be conducted to meet the applied requirements of ACS. The optimization model of ACS is founded on the basis of solving TSP problems. In this paper, through analyzing the similarity between concept solving and TSP problems, concept solving is transformed into a problem of optimal path in combinatorial optimization required by ACS. Then an improved ACS-based concept solving method is proposed to tackle it, in which not only the compatible degree among the function carriers combined is considered, but also the synthetical evaluation value of the function carriers combined. Thus, the whole optimization of concept solution is ensured by using this proposed method.

The remainder of the paper is organized as follows. Section 2 gives an overview of ACS, including its principle and basic algorithm. Section 3 investigates the essence of concept solving, and gives the concept solving process model based on function analysis. Section 4 presents the ACS mode of concept solving problem, in which the dynamic programming based solution space model and the longest path based optimization model are developed. Section 5 proposes an improved ACS algorithm, by giving the necessary principles and its steps to define an ACS for concept optimization. Section 6 provides a case study to show the effectiveness of the developed method. Section 7 summarizes this research.

2. Ant colony system

2.1 Ant colony system principle

ACS is a multi-agent system in which the behavior of each agent (ant) is inspired by the foraging behavior of real ants. In particular, ACS simulates the be-

havior of ant colonies in nature when they are foraging for food and looking for a short path from their nest to food source. The way in which a short path is built by the colony is based on the ants' ability to deposit and sniff a chemical substance called a *pheromone*. In fact, when an ant is going from the nest to the food source and vice versa, it deposits a small amount of pheromones on the ground. The amount of pheromone on a path is used to guide the colony during the search, so that when an ant is located in a branch and has to make a decision, it makes a probabilistic decision biased by the amount of pheromone deposited on the different branches. In this way, at the beginning of the process, all the paths have the same probability of being chosen (because there is no pheromone at all), but with the continuous action of the colony, the shortest paths are more frequently visited, receiving a higher amount of pheromone and thereby becoming more attractive for the subsequent ants. By contrast, the longest paths are less visited; together with the pheromone evaporation process, this makes these paths less attractive for subsequent ants. This process will last until all the ants select the shortest path. Hence, the final solution (path) emerges from the collaboration among all the members in the colony.

2.2 Basic ant colony algorithm

ACS algorithms were initially used to solve problems related to path searching in graphs, mainly the TSP [12-14]. In order to easily understand the optimization mechanism of the proposed algorithm, all descriptions in the following section are based on the Traveling Salesman Problem with n cities.

According to the features of TSP, the solution space of TSP can be transformed into a searching topological graph of a network defined as $G=(N,E)$, where N is the set of nodes and E is the set of oriented arcs between any two nodes. A path connecting an initiative node and a terminal node through a series of interim nodes by oriented arcs corresponds to a feasible solution candidate. At the beginning of the algorithm, ants are generated at time point t and are distributed randomly on the nodes in the graph. Suppose M is the number of ants in an ant colony; k denotes the kth ant where $k=1, 2, \cdots, M$. tabu(k), which records the nodes that ant k has visited up till now, is introduced to ensure that any node cannot be visited more than once by the same ant in a cycle. The operational mechanism of a basic ant colony algorithm is based on the combination of positive feedback principle and a certain heuristic search technique. The transition probability of ant k migrating from node i to node j at time point t is decided by the formula:

$$P_{ij}^{k}(t)=\begin{cases}\tau_{ij}^{\alpha}(t)\cdot\eta_{ij}^{\beta}(t)\Big/\sum\tau_{is}^{\alpha}(t)\cdot\eta_{is}^{\beta}(t) & j,s\notin \text{tabu}(k)\\ 0 & \text{otherwise}\end{cases} \quad (1)$$

where $\tau_{ij}(t)$ is the amount of pheromone trail on the edge ij at time t. When $t=0$, every edge will be given a rather small initial pheromone value C, and this value is frequently updated in the consecutive step s. η_{ij} is the local heuristic information which represents the expectation of migrating from node i to node j, where $\eta_{ij}=1/d_{ij}$ and d_{ij} is the Euclidean distance between node i and node j. α and β are two parameters that control the relative weight of pheromone trail and heuristic value when probability P_{ij} is calculated.

Each of the ants has finished a journey covering every node and M feasible solutions emerge, which is called a cycle. Then, the pheromone on the edge is updated according to the quality of acquired routes. When the ants finish a cycle, the pheromone concentration of the edges they have passed is updated. The updating formula is as follows:

$$\tau_{ij}(t+n)=\rho\tau_{ij}(t)+\Delta\tau_{ij} \quad (2)$$

where ρ is the evaporating coefficient, $\rho\in(0,1)$. $\Delta\tau_{ij}$ is the pheromone increment left on the edge ij in this cycle., which is determined by the following formula:

$$\Delta\tau_{ij}=\sum_{k=1}^{M}\Delta\tau_{ij}^{k} \quad (3)$$

where $\Delta\tau_{ij}^{k}$ is the pheromone increment left on the edge ij by the kth ant in this cycle and can be computed by

$$\Delta\tau_{ij}^{k}=\begin{cases}Q\big/L_k & \text{if } k\text{th ant uses path } ij \text{ in its tour}\\ 0 & \text{othwise}\end{cases} \quad (4)$$

where Q is the heuristically defined constant. L_k is the length of the tour performed by ant k. When $t=0$, $\Delta\tau_{ij}^{k}=0$, $\tau_{ij}=C$.The values of Q, α, β,

C, ρ, and M can be determined by using the method proposed by Zhan and Xu [15].

3. Concept solving process model based on function analysis

Concept solving in conceptual design is founded on the approach of function analysis. According to the related principle of function analysis [16], the solving process model is established in Fig. 1. The fundamental process is "Function determination→Function decomposition→Function solving→Function synthesis→Function evaluation." The explanation of this developed model is as follows.

The first step is to implement a map from the design requirements domain to the function domain, including acquiring the general function, decomposing the general function into function units, and establishing the function-structure diagram. This is an analytical process and is achieved by human-machine interaction mode. Then function unit solutions corresponding to each function unit in function-structure diagram are searched employing the function carrier base developed. Through different combinations of function carriers achieving different function units, design concepts are generated by using a morphology matrix and constructing a concept solution space. Thus, the map from the function domain to the structure domain is completed, which is a synthetical process. Since there exists a many-to-many relationship between function and function carrier, the problem of combination, generation, evaluation, and decision for concepts is put forward as a key problem of conceptual solving. In essence, the concept solving problem is a concept optimization process and its aim is to acquire the best concept. The intellectualization and algorithmization of concept design mainly concentrates on the two stages of synthesis-optimization.

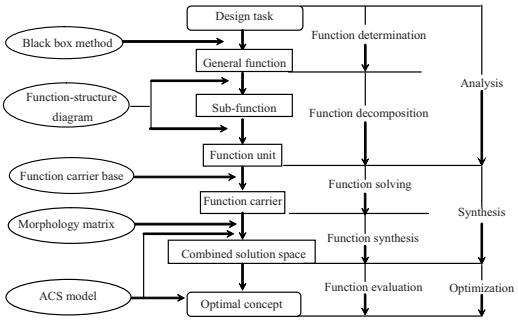


Fig. 1. Concept solving process model.

In this research, ACS is adopted to implement the combinational optimization process.

4. ACS mode for concept solving problem

4.1 Dynamic programming based solution space model

The traditional concept optimization is usually achieved by manual operation, which belongs to a static optimization mode. The best concept solution is acquired by a combination of function unit solutions, each of which is the best solution of the corresponding function unit. However, this method is not correct on certain conditions and the generated solution may be a locally optimal solution. As a method to deal with multistage decision problems, dynamic programming is proposed to tackle the involved problem on the whole, in which the problem is divided into a certain number of stages, and a decision is made in each of the stages [17]. Here, we use the idea of dynamic programming for reference to represent the concept solution space and transform the concept solving problem into the required TSP mode of ACS.

Firstly, according to the flow direction of signal, power, and substance, function units in function-structure diagram developed are arranged in a sequence with a specific correlative relation and corresponding order. Here, in order to state the method simply, a serial mode is used to connect all the function units, each of which is viewed as a certain stage of dynamic programming, as shown in Fig.2. In this figure, S_0 is a supposed initial state to represent a TSP problem and does not signify a function carrier. Except for S_0, all the elements in every column represented with S_{ij} are the function carriers to achieve the function unit, called states of the corresponding stage. To select a state in a certain stage is decision making, in which not only partial performance (i.e., carrier performance) is required to be considered, but also the whole combinational performance (i.e., the consistency among the carriers). Concept solving of conceptual design is an apparent multistage decision problem, and furthermore, to solve TSP using ACS is also a multistage decision problem. Under ACS, a city (node) that an ant has not visited needs to be selected according to transition probability in every stage. Whereas, ants start off from a certain city and finally return to the starting city, which constitutes a closed route; the relative definitions for concept solv-

ing are given as follows.

Definition 1: The oriented line connecting two states (function carriers) selected from two adjacent stages (function units), respectively, is called the distance between two states. The value of the distance is obtained by a quantitative evaluation for the carrier the oriented line points at, and the consistency between two carriers is required to be considered at the same time.

Definition 2: All the states, each of which is arbitrarily selected in every stage from stage 0 to stage m, constitute a decision-making set, which is called a route. It is obvious a route represents a combinational concept solution. The length of this route is the total evaluation value of concept solution, which can also be viewed as a sum of all the distances between two adjacent stages in a route.

As the above definitions demonstrate, a different route will be generated if we make a different decision in every stage, and thus all the generated routes form the searching space of concept optimization. Fig. 2(a) denotes the initial searching status and the number of solutions is $n_1 \times n_3 \times \cdots \times n_m$ .The size of a circle signifies the magnitude of the selecting probability, which is calculated by Eq. (1). In Fig. 2(a), the selecting probabilities of all states are equivalent, and thus all the circles have the same size. The purpose of employing ACS is just to change the size of these circles continuously and find the best concept solution as an optimization result. Fig. 2(b) represents the change of searching space after certain numbers of iterations, in which the large circle in every stage represents the function carrier with a large selecting probability. Here the possible concept solution is a combination of "$S_{12} - S_{21} - S_{3n_3} - \cdots - S_{m2}$".

4.2 Determination of optimization model

In the TSP model of ACS, each feasible solution is represented with a route an ant has visited. On the other hand, the aim of concept solving is to acquire the best concept satisfying the design objectives, i.e., find an optimal route from stage 0 to stage m, to ensure the maximum of total evaluation value of concept solution. Hence, the optimal route actually is equivalent to the longest path of the TSP model. The constraint conditions include the following: (1) The function carrier must satisfy the required requirements in design requirements. (2) The adjacent function carriers must satisfy the consistency condition. (3) The function carrier must be a solution belonging to the corresponding function unit solution set: $S_{ij} \in S_i$. For a concept solution, these three constraints must be simultaneously satisfied. Here, a penalty item is employed to represent the effect of constraint on the objective value. The optimization model can be described as

$$\max \sum_{i=1}^{m} E_{(i-1)i} \tag{5}$$

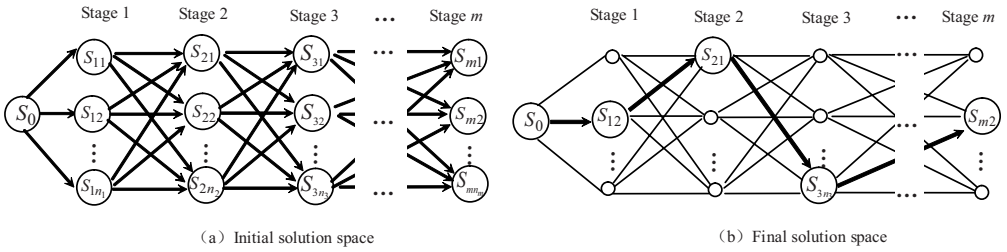
where $\sum_{i=1}^{m} E_{(i-1)i}$ is the length of a route--the total evaluation value of concept; $E_{(i-1)i}$ is the distance between two adjacent two states, as calculated by

$$E_{(i-1)i} = W_i \cdot e(S_{ij}) + K \cdot C_{(i-1)i} - P \tag{6}$$

where W_i is the weight of ith stage, $\sum_{i=1}^{m} W_i = 1$; $e(S_{ij})$ is the evaluation value of state S_{ij} selected from ith stage, as calculated by

$$e(S_{ij}) = \sum_{k=1}^{n} f_{ij}(V_k) \cdot w_k \tag{7}$$

where V_k is kth evaluation indicator of S_{ij} , $1 \le k \le n$; n is the number of evaluation indicator; w_k is the weight of V_k , $\sum_{k=1} w_k = 1$, $f_{ij}(V_k)$ is the



（a）Initial solution space （b）Final solution space

Stage 1~Stage m: Function unit code; S_0 :Initial stage; $S_{11} \sim S_{mn_m}$: Carrier code

Fig. 2. Representation of solution space.

evaluation value of S_{ij} towards V_k; P is the penalty factor: if the concept solution satisfies three constraint conditions, then $P=0$, otherwise, $P=R$, where R is a large real number defined as 100 here; K denotes a conversion coefficient to convert compatible degree into evaluation value of carrier; $C_{(i-1)i}$ represents the compatible degree between two adjacent carriers, which can be acquired by

$$C_{(i-1)i}=\sum_{k=1}^{N}\alpha_k c(\gamma_k) \tag{8}$$

where $c(\gamma_k)$ is the compatible degree of two carriers towards kth compatible unit γ_k, $1\le k\le N$, N is the number of compatible units. In the conceptual design of a mechanical system, the compatible degree is usually obtained by comparing the input feature of one carrier with the output feature of the other carrier. α_k is the weight of γ_k. It is important to note that S_0 does not denote a function carrier and thus $C_{01}=0$.

5. Applying ACS to concept solving

5.1 Fundamental principle and steps

From the concept solving process model shown in Fig. 1, we can see ACS is mainly applied in the latter stages of conceptual design. Above all, the function-structure diagram of a mechanical product can be acquired by using function analysis, and function units are arranged in a serial sequence according to the flow direction of signal, power, and substance. Then function-unit solutions are searched by employing the function carrier base developed, and thus the correspondence relations between function unit and function carriers are formed, which can be signified with a morphology matrix. The next step is a combinational optimization process, which can be implemented with ACS.

According to the solution space model in Fig. 2, suppose there are M ants in this system. First, M initial concept solutions can be generated by stochastic combination using a morphological matrix. Regarding these M concepts as an initial solution set, there are M states altogether for the ith $(1\le i\le m)$ stage, which constitute a candidate group of this stage. Considering m stages of concept solution as m nodes, there are M lines connecting the $(i-1)$th node to ith node, which represents the M candidate value of ith candidate group. In these lines, the jth line is denoted with ij and its pheromone amount at time t is denoted with τ_{ij}^{t}. Each ant starts off from the initial stage S_0, makes q decision to select m lines in turn according to a certain strategy, and finally arrives at the mth node. The route each ant has visited denotes a concept solution. The m lines constituting a route denote its m function unit solutions. To avoid the local optimization, the operators of crossover and mutation in GA are applied to M generated candidates of every stage. The M candidates generated after crossover and mutation are employed as the corresponding candidates of the new generation solutions. After M concept solutions are generated, the pheromone amount on every edge is modified according the length of the route (concept solution). After an iteration, the candidate value of every stage is updated by selecting M candidates with more pheromone amount from the last candidates and the generated candidates. This iteration is repeated till the termination condition is satisfied. The termination condition can be established by specifying a maximum iteration number.

The fundamental steps of this algorithm are described as follows [18].

(1) Initialization. Generate M initial concept solutions randomly, calculate their objective values, generate the candidate group of each stage, and calculate the pheromone amount of each candidate in candidate group according to the candidates' value and objective values.

(2) Iteration process. While not satisfy[fso1] the termination condition *do*

① for i=1 to m do (loop for m stages)

for k=1 to M do (loop for M ants)

According to $P_{ij}^{k}(t)$, select the value of kth ant in candidate group of ith stage as the initial value of this stage

end for k

Apply crossover and mutation to the M initial values of ith stage selected, generate the new M values of ith stage, and constitute the new candidate group

end for i

② Modify the pheromone amount of each candidate in each candidate group according to the length of route;

③ Select the values with more pheromone amount in candidate group as the new candidate group

end for *while*

5.2 Improvement on ACS

It is the shortest path problem to be solved when using ACS to deal with TSP, whereas a concept solving problem actually is a problem of longest path since the aim of concept solving is to acquire the best concept satisfying the design objectives. Thus, some improvements are required to be implemented on the basic ACS, which are illuminated by the following.

(1) When calculating the transition probability $P_{ij}^k(t)$, two methods can be adopted here. One is to use Eq. (1), but the local heuristic factor η_{ij} ought to be the distance between two adjacent carriers, i.e., $\eta_{ij} = E_{(i-1)i}$, not the reciprocal of $E_{(i-1)i}$. The other is to use the following formula.

$$P_{ij}^k(t)=\tau_{ij}(t)\Big/\sum_{r=1}^{n_i}\tau_{ir}(t) \tag{9}$$

where $\tau_{ir}(t)$ is the pheromone amount of the rth carrier in the ith stage candidate group, $1 \le r \le n_i$, n_i is the number of the carriers in this candidate group.

(2) When calculating the pheromone increment $\Delta\tau_{ij}^k$, the corresponding modification is applied to Eq. (4):

$$\Delta\tau_{ij}^k=\begin{cases} Q\cdot f_k & \text{if } k\text{th ant uses path } ij \text{ in its tour} \\ 0 & \text{othwise} \end{cases} \tag{10}$$

where f_k is the objective value corresponding to the concept solution, and can be obtained by

$$f_k=\sum_{i=1}^{m}E_{(i-1)i}^k \tag{11}$$

(3) In order to ensure the diversity of the optional path, crossover and mutation in GA are applied to the M initial values of ith stage. Here, the adaptive crossover and mutation schemes proposed by Srinivas and Patnaik [19] are employed to change the probabilities of two operations adaptively in terms of the solution's quality. The crossover probability P_c and the mutation probability P_m are determined by

$$P_c=\begin{cases} \dfrac{k_1(f_{\max}-f')}{f_{\max}-f_{avg}} & f'\ge f_{avg} \\ k_2 & f'<f_{avg} \end{cases} \tag{12}$$

$$P_m=\begin{cases} \dfrac{k_3(f_{\max}-f)}{f_{\max}-f_{avg}} & f\ge f_{avg} \\ k_4 & f<f_{avg} \end{cases} \tag{13}$$

where $f_{\max}$ is the maximal fitness value in the current population, f_{avg} is the average fitness value in the current population, f' is the larger fitness value of two individuals implementing crossover, f is the fitness value of the individual implementing mutation, and k_1, k_2, k_3, and k_4 are user-specified constants, $0 < k_1, k_2, k_3, k_4 < 1$.

Under this approach, the changeable probabilities of crossover and mutation are adopted to implement optimization. When the fitness values of individuals tend to the same value or the evolution tends to a local optimum, the values of P_m and P_c will be increased to avoid local convergence. When the fitness values of individuals tend to decentralization, the values of P_m and P_c will be reduced to preserve better individuals. Furthermore, for the individual whose fitness value is higher than the average value, the values of P_m and P_c will be smaller to protect it from being destroyed. For the individual whose fitness value is smaller than the average value, the values of P_m and P_c will be larger to ensure it being eliminated as possible. However, the above approach suffers from a major drawback. When the fitness value of an individual is close to the maximal fitness value, the values of P_c and P_m will be very small. When it is equal to the maximal fitness value, the values of P_c and P_m will be 0. Apparently, it is unfavorable for the good individuals at the early evolution stage and is apt to result in a local optimum. For the sake of rapid convergence and avoiding a local optimum, some improvements on the original schemes are made by amending the computing method of P_c and P_m. In this paper, P_c and P_m are determined by

$$P_c=\begin{cases} P_{c1}-\dfrac{(P_{c1}-P_{c2})(f'-f_{avg})}{f_{\max}-f_{avg}} & f'\ge f_{avg} \\ P_{c1} & f'<f_{avg} \end{cases} \tag{14}$$

$$P_m=\begin{cases} P_{m1}-\dfrac{(P_{m1}-P_{m2})(f-f_{avg})}{f_{\max}-f_{avg}} & f\ge f_{avg} \\ P_{m1} & f<f_{avg} \end{cases} \tag{15}$$

where P_{c1}, P_{c2}, P_{m1}, and P_{m2} are user-specified constants, $0 < P_{c1}, P_{c2}, P_{m1}, P_{m2} < 1$, $P_{c1} > P_{c2}$, $P_{m1} > P_{m2}$. Apparently, according to Eq. (14) and Eq. (15), when the fitness value of an individual equals the maximal fitness value, the values of P_c and P_m will not be equal to 0 and rise to P_{c2} and P_{m2} respectively. Thus,

the local convergence at early stage can be effectively avoided and the diversity of the population can be ensured.

Compared with basic ACS, the employment of adaptive crossover and mutation schemes can ensure the diversity of the solutions and avoid a local optimum. At the same time, it can protect some better solutions as possible and improve the convergence speed. Apparently, by determining P_m and P_c of individuals adaptively, it strikes a balance between two incompatible goals, global optimum and rapid convergence.

It is necessary to note if the new generated carrier by crossover or mutation is not a function unit solution in a solution set derived from the carrier base, then the combinational concept including this carrier is not a feasible solution. A penalty function can be applied to eliminate this solution finally.

6. A case study

A concept design problem for a mechanical press serves as an example to illustrate the solution process of the best concept by using this proposed approach.

This mechanical press is mainly used to produce the body part of a car and is required by a metal forming machinery factory. The design needs includes quantitative data and qualitative requirements. The quantitative data is provided as follows. The required pressure is 800 KN, the stroke length of pressure is 6 mm, the stroke length of the punch varies from 70 mm to 130 mm, the number of stokes varies from 30 to 50 per minute, the maximal die filling height is 500 mm, and the press weight is less than 10 ton, etc. The major design requirement is to improve its working performance, simplify its structure as possible, shorten its transmission chain, reduce its noise, and advance its transmission efficiency. Moreover, design requirement also includes being convenient for manufacture and design.

6.1 Function decomposition and establishment of morphology matrix

According to the proposed concept solving process model in Fig. 1, through function determination, the general function of a press can be obtained firstly, i.e., making the metal in the die generate a permanent deformation by employing a press working method. Then seven main function units are acquired by function decomposition, structuring the function-structure diagram of the press, as shown in Fig. 3. According to the direction of transmission chain of a press, these function units are approximately arranged in a serial sequence: drive → deceleration 1→ start → accumulation energy and balance → brake → deceleration 2 → reciprocating motion. As noted before, since the compatible degree between two adjacent carriers with connecting relation is considered when calculating it, these function units must be arranged according to the direction of transmission chain, otherwise an incorrect optimization result could be obtained.

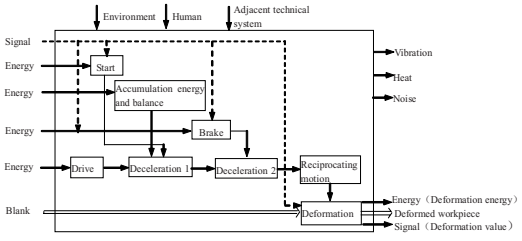


Fig. 3. Function-structure diagram of press.

Table 1. Corresponding knowledge of function carriers.

Corresponding knowledge		Function carrier			
		Sliding pin clutch	Rotating key clutch	Disk-type friction clutch	Quadrat friction clutch
Performance indicators	Joining time	Fair	Fair	Somewhat short	Somewhat Short
	Joining stabilization	Poor	Somewhat poor	Good	Somewhat Good
	Structural complexity	Somewhat simple	Simple	Complex	Fair
	Manufacturability	Easy	Easy	Difficult	Somewhat difficult
	Reliability	Poor	Somewhat poor	Fair	Fair
	Adjustable performance	Poor	Poor	Somewhat good	Good
	Energy consumption	Low	Low	High	Somewhat high
	Working life	Somewhat short	Short	Fair	Fair
	Cost	Somewhat low	Low	High	Somewhat high

Table 2. Morphology matrix of the mechanical press.

Function unit	Function carrier			
	1	2	3	4
A (Drive)	A_1 (Enclosed Squirrel-cage motor)	A_2 (Squirrel-cage motor with high slip ration)	A_3 (Wound-rotor motor)	A_4 (Salient motor)
B (Deceleration 1)	B_1 (Wide V-belt transmission)	B_2 (Narrow V-belt transmission)	B_3 (Common V-belt transmission)	B_4 (Joined V-belt transmission)
C (Accumulation energy and balance)	C_1(Disk-type flywheel)	C_2 (Web-type flywheel)		
D (Start)	D_1 (Sliding pin clutch)	D_2 (Rotating key clutch)	D_3 (Disk-type friction clutch)	D_4 (Quadrat friction clutch)
E (Brake)	E_1 (Disk-type brake)	E_2 (Block brake)	E_3 (Belt brake)	
F (Deceleration 2)	F_1 (Cylindrical gear drive)	F_2 (Herringbone gear drive)	F_3 (Worm drive)	F_4 (Bevel gear drive)
G (Reciprocating motion)	G_1 (Crank-block mechanism)	G_2 (Rotating guiding rod mechanism)	G_3 (crank-elbow bars mechanism)	G_4 (six rods mechanism)

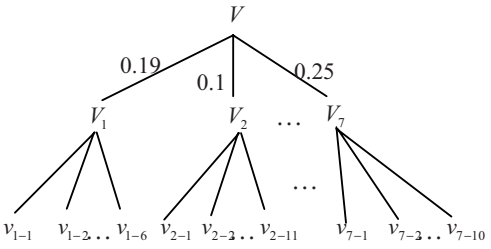


Fig. 4. Evaluation objective tree.

The next step is to search the solutions to each function unit to develop the morphology matrix. The corresponding knowledge base of the function carrier is required to be built using an object oriented method [9, 10], including its title, function, performance indicators, the input/output features denoting consistency, and so on, which are usually expressed with linguistic terms. Take the function unit of *start* as an example; its function carriers and corresponding knowledge (the input/output features are omitted) are summarized with linguistic expressions in Table 1. Generally speaking, a function carrier achieving a function unit is more than one. Thus function carriers achieving each corresponding function unit are listed by using the morphology matrix of Table 2.

6.2 Applying the improved ACS to concept optimization

From the provided morphology matrix, we can see the number of combinational concepts N is 6144, viz. $N = 4\times4\times2\times4\times3\times4\times4 = 6144$. Apparently, it is impossible to acquire the best concept by evaluating these concepts one by one. Here the improved ACS (IACS) is employed to acquire the satisfactory concepts. First evaluation objective tree is established in Fig.4, including two levels of sub-objectives and their weights. In this figure, V denotes total objective, $V_1 \sim V_7$ representing 7 function units constitute 1st level sub-objective, the performance indicators included in each of function units denote the 2nd level sub-objective, e.g. $v_{1-1} \sim v_{1-6}$ are 2nd level sub-objectives of V_1. The numerical values in this figure are the corresponding weights, which represent the importance of different function units. By analyzing design requirements and working conditions of press, it can be known that the functional requirement is the most important of all, power requirement takes the second place, and the other five function units possess close importance. Here, the weight of 7 function units is, in order, 0.19, 0.1, 0.12, 0.11, 0.11, 0.12, 0.25, which is determined by using the pair-wise comparison method [20].

The precondition of applying ACS is to calculate the distance between two adjacent carriers, including calculating the carrier's evaluation value and compatible degree. When calculating evaluation value, linguistic expressions measuring carrier's performance indicators ought to be quantified. Take the performance indicator of *reliability* in Table 1 as an example; its linguistic expression includes 5-level lingual evaluation--poor, somewhat poor, fair, somewhat good, and good--which can be represented with 0, 0.25, 0.5, 0.75, and 1, respectively. According to Eq. (7) we can calculate a carrier's evaluation value. When calculating compatible degree by using Eq. (8)

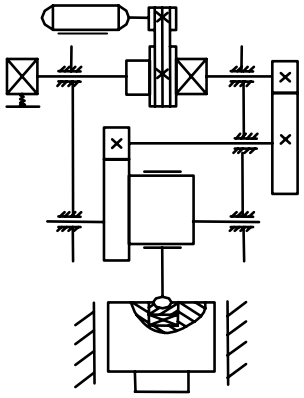


Fig. 5. The sketch of the press.

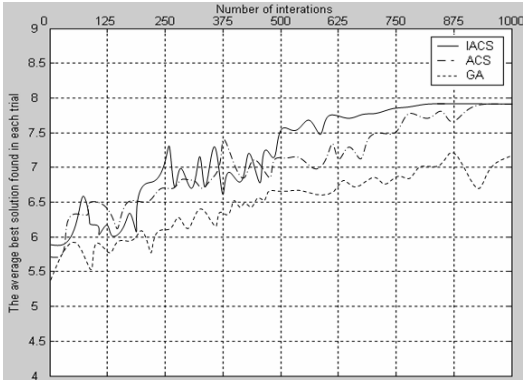


Fig. 6. The evolutionary processes of the best solution.

the output feature of one carrier is required to be compared with the input feature of the other carrier. Considering consistency is less important than the design objective in total evaluation, let $k = 0.45$ and $P = 100$. Then using Eq. (6) we can acquire the distance two adjacent carriers.

After determining all the distances, we can use IACS to implement concept optimization. First, the related parameters ought to be given. Here, according to the method proposed by Zhan and Xu [15], let $M = 15$, $\alpha = 1$, $\beta = 1.5$, $\rho = 0.6$, $Q = 1$, $P_{c1} = 0.9$, $P_{c2} = 0.6$, $P_{m1} = 0.1$,and $P_{m2} = 0.003$ .Then we can employ the improved algorithm in Section 5 to implement programming. The termination condition can be established by specifying the maximum iteration number as 1000.

Finally, the best path obtained by running the program is $A_2 \to B_2 \to C_2 \to D_3 \to E_1 \to F_2 \to G_1$. The corresponding concept solution is as follows.

Drive: Squirrel-cage motor with high slip ration;

Deceleration 1: Narrow V-belt transmission;

Start: Disk-type friction clutch;

Accumulation energy and balance: Web-type flywheel;

Brake: Disk-type brake;

Deceleration 2: Herringbone gear transmission;

Reciprocating motion: Crank-block mechanism.

The sketch of the press concept is shown in Fig. 5, which is consistent with the concept employed in the current new press.

6.3 Algorithm comparison study

In the current field of conceptual design, another commonly used optimization algorithm is GA [10]. In order to explain the advantage of the proposed algorithm with IACS, we employed GA and ACS to implement the concept optimization aiming at the same case. The evolutionary processes of the best solution using three algorithms are shown in Fig.6. From this figure, we can see the acquired best solution using ACS is the same as that using IACS. However, the rate of approaching the best solution with IACS is faster than that with ACS, and when arriving at the best solution, the calculation result can keep relative stabilization around this solution with IACS. Compared with IACS and ACS, GA possesses a lower convergent speed and poorer stabilization. Moreover, when the iteration number arrives at 1000, the acquired best solution using GA is poorer than that using IACS and ACS.

7. Conclusions

Concept solving in conceptual design is a combinational optimization problem in essence. There exists a "combinational explosion" phenomenon when using the traditional morphological matrix method to tackle it. ACS is a new simulated evolutionary algorithm to solve combinational optimization problems. By analyzing the similarity between concept solving and traveling salesman problem, concept solving is transformed to meet the applied requirements of ACS. Then an improved ACS-based concept solving method is proposed to implement concept optimization incorporating the positive feedback searching mechanism of ACS and crossover and mutation operation of GA. It is verified that this method enables concept solving to be implemented with better operability, which offers a promising way to solve combinatorial explosion problems in conceptual design.

Acknowledgments

This research was supported by the Shanxi Youth Science Foundation of China under contract number 2007021028 and the National Natural Science Funds of China under contract number 50575214.

References

[1] N. Singh, Systems Approach to Computer-integrated Design and Manufacturing, John Wiley & Sons, New York, USA, (1995).

[2] J. L. Nevins and D. E. Whitney, Concurrent Design of Products and Processes, McGraw-Hill, New York, USA, (1989).

[3] H. J. Zou, Y. L. Tian, W. Z. Guo, et al., Methodology for conceptual design of mechanism system, *Journal of Shanghai Jiaotong University*, 37 (5) (2003) 668-673.

[4] D. A. Hoeltzel and W. H. Chieng, Knowledge-based approaches for the creative synthesis of mechanisms, *Computer Aided Design*, 22 (1) (1990) 57-67.

[5] G. Ermer and R. Rosenberg, Steps toward integrating function-based models and bond-graphs for conceptual design in engineering, *Automated Modeling for Design, ASME*, 47 (11) (1993) 47-62.

[6] Z. Y. Yao and C.Y. Huang, Study on the structure catalogues for conceptual design of mechanical transmission, *China Mechanical Engineering*, 9 (6) (1998) 53-55.

[7] T. T. H. Ng and G. S. B. Leng, Application of genetic algorithms to conceptual design of a micro-air vehicle, *Engineering Applications of Artificial Intelligence*, 15 (5) (2002) 439-445.

[8] J. Sun and D. K. Kalenchuk, Design candidate identification using neural network based fuzzy reasoning, *Robotics and Computer-Integrated Manufacturing*, 16 (5) (2000) 382-396.

[9] Q. L. Shu and B. Hao, An automatic conceptual design system for mechanical products, *China Mechanical Engineering*, 13 (19) (2002) 1676-1678.

[10] H. Z. Huang, R. F. Bo and X. F. Fan, Concept optimization for mechanical product using genetic algorithm, *Journal of Mechanical Science and Technology*, 19 (5) (2005) 1072-1079.

[11] H. Z. Huang, R .F. Bo, W. Chen, An integrated computational intelligence approach to product concept generation and evaluation, *Mechanism and Machine Theory*, 41 (5) (2006) 567-583.

[12] M. Dorigo, V. Maniezzo and A. Colorni, The Ant system: Optimization by a colony of cooperating agents, *IEEE Transactions on Systems, Man, and Cybernetics-Part B,* 26 (1) (1996) 1-13.

[13] M. Dorigo and L. M. Gambardella, Ant colony system: a cooperative learning approach to the traveling salesman problem, *IEEE Transactions on Evolutionary Computing*, 1 (1) (1997) 53-66.

[14] L. M. Gambardella and M. Dorigo, Solving symmetric and asymmetric TSP by ant colonies, Proc. IEEE Int Conf Evol Comp, Piscataway, New Jersey, USA. (1996) 622-627.

[15] S. C. Zhan, J. Xu and J. Wu, The optimal selection on the parameters of the ant colony algorithm, *Bulletin of Science and Technology*, 19 (5) (2003) 381-386.

[16] G. Pahl and W. Beitz, Engineering Design, The Design Council, London, UK, 1984.

[17] Y. Q. Hu, Operational research foundation and application, Harbin Institute of Technology Press, Harbin, CHN, (1998).

[18] J. Chen, J. Shen and L.Qin, A method for solving optimization problem in continuous space by using Ant Colony Algorithm, *Journal of software*, 13 (12) (2002) 2317-2322.

[19] M. Srinivas and L. M. Patnaik, Adaptive probabilities of crossover and mutation in genetic algorithms, *IEEE Trans on Systems, Man and Cybernetics*, 24 (6) (1994) 656-667.

[20] H. J. Zou, Conceptual Design for Mechanical System, Machine Press, Beijing, CHN, (2003).