We are IntechOpen, the world's leading publisher of Open Access books Built by scientists, for scientists

6,900
Open access books available

185,000
International authors and editors

200M
Downloads

154
Countries delivered to

Our authors are among the

TOP 1%
most cited scientists

12.2%
Contributors from top 500 universities

CLARIVATE ANALYTICS
BOOK CITATION INDEX
INDEXED

WEB OF SCIENCE™

Selection of our books indexed in the Book Citation Index
in Web of Science™ Core Collection (BKCI)

Interested in publishing with us?
Contact book.department@intechopen.com

A Hybrid ACO-GA on Sports Competition Scheduling

Huang Guangdong and Wang Qun
School of Information Technology,
China University of Geosciences (Beijing), Beijing, 100083
China

1. Introduction

During the past 20 years sports competition scheduling (SCS) has received a great deal of attention in the operations research literature. The existed researches on SCS mainly focus on league scheduling and round-robin tournament play including American Football, baseball, chess, cricket, dressage, ice hockey, tennis and even the Olympic Games. Some of the applications use highly problem-specific approaches, others use more general techniques such as goal programming, simulated annealing, subcost-guided search, integer programming, tabu search and local search and so on. Graph theory was used by de Werra to develop schedules with desirable properties, such as the minimum number of breaks in the sequences of home and away games. Schreuder constructed a timetable for the Dutch professional soccer leagues that minimizes the number of schedule breaks. Russell and Leung developed a schedule for the Texas Baseball League that satisfied stadium availability and various timing restrictions with the objective of minimizing travel costs. Nemhauser and Trick created a schedule for the Atlantic Coast Conference men' s basketball season, taking into consideration numerous conflicting requirements and preferences.Most of problems of sports competition scheduling are NP-complete. The computationally burdensome presentation of SCS makes the use of heuristic approaches appropriate.

Swarm Intelligence (SI) is a property exhibited by some mobile systems such as social insect colonies and other animal societies that have collective behavior. Individuals of those systems such as ants, termites an wasps are not generally considered to be individually intelligent however they do exhibit a degree of intelligence, as a group, by interacting with each other and with their environment. Those systems generally consist of some individuals sensing and acting through a common environment to produce a complex global behavior.

They share many appealing and promising features that can be exploited to solve hard problems. Furthermore, they are particularly well suited for distributed optimization, in which the system can be explicitly formulated in terms of computational agents.

One of the most popular swarm inspired methods in computational intelligence areas is the Ant Colony Optimization (ACO) method. Ants are able to evolve their foraging behavior by changing their environment through the evolving pheromone trails. When an ant follows a shorter path it travels trough it more frequently than the others, building up a higher

pheromone count in the process. This attracts more ants to that path, which makes the pheromone level to go up even further and a more number of ants are now drawn into it. Such interactions continue between the ant colony members till an optimum path emerges. The ACO method has become one of the best-known metaheuristic techniques, side to side with tabu search, genetic and evolutionary algorithms, simulated annealing, iterated local search, variable neighborhood search and some other prominent general-purpose search methods. an approach that has originally been developed by Dorigo et al. in the early 1990s, have evolved to a powerful, versatile optimization tool with a still rapidly growing number of publications and numerous applications in diverse areas of operations research, management science and technology.

Genetic algorithms (GA) developed by Holland are heuristic approaches based on evolutionary principles. GA is developed in a way to simulate biological evolutionary process and genetic operations on chromosomes. GA can handle any kind of objective functions and any kind of constraints without much mathematical requirements about the optimization problems. When applying to optimization problems, GA starts with a set of randomly selected chromosomes as the initial population that encodes a set of possible solutions. then the chromosomes are evaluated according to their fitness values using some measures of profit or utility that we want to optimize. Two genetic operators, crossover and mutation, alter the composition of genes to create new chromosomes called offspring. The selection operator is an artificial version of natural selection, a Darwinian survival of the fittest among populations, to create populations from generation to generation, and chromosomes with better fitness values have higher probabilities of being selected in the next generation. After several generations, GA can converge to the best solution. In the past several years, GA has been applied successfully in many fields such as automatic control, programming design, combination optimization, image processing and signal processing.

The population-based approach of the Genetic Algorithms is very efficient for a global search, but they do not use the feedback in the system information enough and the accuracy solution efficiency is low. ACO is a kind of positive feedback mechanism, but it lacks information early, the solution speed is slow. In this chapter, Based on the existed researches of sports competition scheduling (SCS) the author presented a hybrid Ant Colony Optimization and Genetic Algorithms (ACO-GA) to improve the productivity and efficiency of sports competition scheduling (SCS). The ACO-GA procedures of combining ant colony optimization and genetic algorithms were as follows: (1) applying the GA in a specified solution space to generate the activity lists, which provided an initial population for ACO; (2) performing ACO; (3) executing the crossover and mutation operations of GA to generate a new population based on the ACO result; (4) performing ACO and GA alternately and cooperatively in the solution space for n times. In order to test the ACO-GA algorithm, we selected Oliver30 and att48 problems. The results indicated that ACO-GA could find the best solution presented on the TSPLIB95. Finally we converted a sports competition scheduling (SCS) into a TSP model, and then using the ACO-GA algorithm. The result illustrated that the ACO-GA was a good method for solving the SCS. The remaining of this paper is organized as follows. Section 1 introduce the existed researches of sports competition scheduling (SCS), Ant Colony Optimization and Genetic Algorithms. Section 2 converted a sports competition scheduling (SCS) into a TSP model firstly, Secondly we introduce the ACO-GA algorithm, then we selected Oliver30 and att48 problems to test the ACO-GA algorithm. Finally we use the ACO-GA algorithm to solve SCS problem. Sections 3 is conclusion and section 4 is Acknowledgment.

2. A hybrid ACO-GA on Sports Competition Scheduling

In this section we converted a sports competition scheduling (SCS) into a TSP model firstly, Secondly we introduce the ACO-GA algorithm, then we selected Oliver30 and att48 problems to test the ACO-GA algorithm. Finally we use the ACO-GA algorithm to solve SCS problem.

2.1 Modeling of SCS

We consider a situation where there are a number of competitors and sports competitions. Let A_s (s=1,2,…,m) present the competitor and B_t (t=1,2,…,n) present the games. We define C_{ij} as follows:

$$C_{st} = \begin{cases} 1 & \text{competitor } A_s \text{ takes part in sports game } B_t. \\ 0 & \text{otherwise} \end{cases} \tag{1}$$

In this way, we can get a 0-1 matrix C_{st}:

$$(C_{st})_{mn} = \begin{pmatrix} C_{11} & C_{12} & \cdots & C_{1n} \\ C_{21} & C_{22} & \cdots & C_{2n} \\ \vdots & \vdots & & \vdots \\ C_{m1} & C_{m2} & \cdots & C_{mn} \end{pmatrix} \tag{2}$$

For an n-game competition, the complete weighted graph G_n on n vertexes is considered. The vertex i ($i = 1, \ldots, n$) presents the game, and the edge $i-j$ presents the games between games i and j (i,j {1, . . . , n}). The weighted function $D_{ij}(i = 1,2,...,n; j = 1,2,...,n)$ is the competitors who take part in both games i and j.

The objective of SCS is to design a schedule of sports competitions in which the competitors who take part in two sequential games are minimized. For example, we consider a sports competition with three games B_1, B_2, B_3 and three competitors A_1, A_2, A_3. Matrix C_{st} is

$C_{3,3} = \begin{pmatrix} 1 & 0 & 0 \\ 1 & 1 & 0 \\ 0 & 1 & 1 \end{pmatrix}$; the weighted function is $D_{3,3} = \begin{pmatrix} 0 & 1 & 0 \\ 1 & 0 & 1 \\ 0 & 1 & 0 \end{pmatrix}$. Thus the objective is 1, and the

games are scheduled to be B_1, B_3, B_2.

If we combine the first game with the last game, then SCS boils down to a Traveling Salesman Problem (TSP). The TSP is an NP-hard problem in combinatorial optimization studied in operations research and theoretical computer science. Given a list of cities and their pairwise distances, the task is to find a shortest possible tour that visits each city exactly once. Thus, it is likely that the worst case running time for any algorithm for the TSP increases exponentially with the number of cities.

The problem was first formulated as a mathematical problem in 1930 and is one of the most intensively studied problems in optimization. It is used as a benchmark for many optimization methods. Even though the problem is computationally difficult, a large number of heuristics and exact methods are known, so that some instances with tens of thousands of cities can be solved.

The TSP has several applications even in its purest formulation, such as planning, logistics, and the manufacture of microchips. Slightly modified, it appears as a sub-problem in many areas, such as DNA sequencing. In these applications, the concept city represents, for example, customers, soldering points, or DNA fragments, and the concept distance represents travelling times or cost, or a similarity measure between DNA fragments. In many applications, additional constraints such as limited resources or time windows make the problem considerably harder.

TSP can be modeled as an undirected weighted graph, such that cities are the graph's vertices, paths are the graph's edges, and a path's distance is the edge's length. A TSP tour becomes a Hamiltonian cycle, and the optimal TSP tour is the shortest Hamiltonian cycle. Often, the model is a complete graph (i.e., an edge connects each pair of vertices). If no path exists between two cities, adding an arbitrarily long edge will complete the graph without affecting the optimal tour.

2.2 ACO-GA

In this section we introduce the GA algorithm firstly, then we we introduce the ACO algorithm. Finally we introduce the ACO-GA algorithm. The procedures of ACO-GA are as follows. First, GA searches the solution space and generates activity lists to provide the initial population for ACO. Next, ACO is executed, when ACO terminates, the crossover and mutation operations of GA generate new population. ACO and GA search alternately and cooperatively in the solution space and we selected Oliver30 and att48 problems to test the ACO-GA algorithm.

2.2.1 Genetic Algorithm

GA expresses the problem-solving process as the chromosomes' survival of the fittest process. From a population (group of feasible solutions) of the potential solution set of a problem, it is called the original population and is composed of certain individual encoded. In fact, each individual is a chromosome entity with characteristics. The chromosome, namely the gene set is the main carrier of genetic material, whose internal performance (gene) is some kind of gene combination, and it decides the exterior performance (phenotype) of the individual shape.

Therefore, we first need to realize coding—mapping from the phenotype to the gene. After the original population is produced, the "chromosome" generation evolves unceasingly, including selection, crossover and mutation operation. The fine quality is gradually retained and performs to combine, finally restrains to the individual "most adapting to circumstances", thus obtains the problem's the optimal solution or the approximate solution. That GA uses biology evolution and hereditary thought to realize the optimization process distinguishes GA from traditional optimization algorithms. Because it operates on the "chromosome", rather than on problem parameters, GA is not restricted to problem's constraint conditions, like continuity and so on. GA carries on searching only according to the individual's fitness value, no any other information. GA has the ability of global search, connotative parallel search and strong robustness. But its flaw of local search ability is also well-known. GA carries on gradually converging process. In practical application, GA generally converges to a certain feasible solution, which may be not the global optimal point, even not local one. Therefore, it is extremely essential both in theory and practice to introduce other local search algorithm to GA to improve GA's local search ability.

The following algorithm presents the fundamental frame of GA.

Begin
    t=0
    Define heredity parameters
    Produce the initial population denoted by X(t)
    Evaluate the individuals X(t)
    While terminal condition does not satisfy do
      Begin
      Select individuals from X(t), crossover and mutation oprations.
      Reorganize (select, crossover and mutate) X(t) to produce population denoted by C(t).
      t=t+1
      End
End

2.2.2 Ant Colony Optimization

ACO algorithms are a class of algorithms inspired by the observation of real ants. Ants are capable of exploring and exploiting pheromone information, which have been left on the ground when they traversed. They then can choose routes based on the amount of pheromone. While building the solutions, each artificial ant collects pheromone information on the problem characteristics and uses this information to modify the representation of the problem, as seen by the other artificial ants. The larger amount of pheromone is left on a route, the greater is the probability of selecting the route by artificial ants. In ACO, artificial ants find solutions starting from a start node and moving to feasible neighbor nodes in the process of ants' generation and activity. During the process, information collected by artificial ants is stored in the so-called pheromone trails. In the process, artificial ants can release pheromone while building the solution (online step-by-step) or while the solution is built (online delayed). An ant-decision rule, made up of the pheromone and heuristic information, governs artificial ants to search towards neighbor nodes stochastically. Pheromone evaporation is a process of decreasing the intensities of pheromone trails over time. This process is used to avoid locally convergence and to explore more search space. The following algorithm presents the fundamental frame of ACO.

Step 1. Pheromone initialization. Let the initial pheromone trail $\tau_0 = k$, where k is a parameter,

Step 2. Main loop. In the main loop, each of the m ants constructs a sequence of n nodes. This loop is executed for Itemax = s iterations and each iteration has two steps.

Step 2.1. Constructing a node sequence by each ant. A set of artificial ants is initially created. Each ant starts with an empty sequence and then successively appends an unscheduled node to the partial sequence until a feasible solution is constructed (i.e., all nodes are scheduled). In choosing the next node j to be appended at the current position i is determined by p_{ij}^k which is probability of ant k transforming from node i to node j at time t

$$p_{ij}^k = \begin{cases} \dfrac{\tau_{ij}^\alpha(t)\eta_{ij}^\beta(t)}{\sum\limits_{u \in U} \tau_{iu}^\alpha(t)\eta_{iu}^\beta(t)} & j \in U \\[4mm] o & otherwise \end{cases} \qquad (3)$$

Where U is the set of unscheduled jobs and $\tau_{ij}(t)$ is the pheromone trail associated with the assignment node j to position i at time t. The parameter $\eta_{ij}(t)$ is the

heuristic desirability of assigning node j to position i at time i. The parameter α is the relative importance of the trace and β is the parameter which determines the relative importance of the heuristic information.

Step 2.2. Update of pheromone trail. The updating rule is applied after each ant has completed a feasible solution (i.e., an iteration). Following the rule, the pheromone trail is added to the path of the incumbent global best solution, i.e., the best solution found so far. If node j is placed at position i in the global best solution during iteration t, then

$$\tau_{ij}(t+1) = p\tau_{ij}(t) + \Delta\tau_{ij} \tag{4}$$

where $\rho, (0<\rho<1)$ is a parameter representing the evaporation of pheromone. The amount $\Delta\tau_{ij} = \dfrac{1}{wt}$, where wt is the weight tardiness of the global best solution.

2.2.3 ACO-GA

GA and ACO are population-based search algorithms by maintaining a population of structures as key elements in design and implementation of problem solving algorithms. These algorithms are sufficiently complex to provide powerful adaptive search approaches, and usually can be embedded with other approaches to speed up the search performance. GA is very efficient for a global search, but it does not use the feedback in the system information enough and the accuracy solution efficiency is low. ACO is a kind of positive feedback mechanism, but it lacks information early, the solution speed is slow. Combined ant colony optimization (ACO) and genetic algorithm (GA) is an effective method. GA generates the information distributes and ACO seeks the accurate solution. The ACO-GA procedures of combining ant colony optimization and genetic algorithms were as follows: (1) applying the GA in a specified solution space to generate the activity lists, which provided an initial population for ACO; (2) performing ACO; (3) executing the crossover and mutation operations of GA to generate a new population based on the ACO result; (4) performing ACO and GA alternately and cooperatively in the solution space for n times. The ACO-GA is represented by the following steps:

Step 1. GA generates solutions and pheromone is left on the excellent paths.

Step 2. t=0(t is the times of iterative), Each ant starts with an empty sequence and then successively appends an unscheduled node to the partial sequence until a feasible solution is constructed (i.e., all nodes are scheduled). In choosing the next node j to be appended at the current position i is determined by p_{ij}^{k} (You can see formula 1) .

Step 3. Crossover operation is executed on the solutions and new solutions are generated. If the target of the new solution is better than the old one, we accept the new solution, otherwise refuse.

Step 4. Mutation operation is executed on the solutions and new solutions are generated. If the target of the new solution is better the old one, we accept the new solution, otherwise refuse.

Step 5. calculate the path length L_k, (k=1,2,...,m) of every ant and record the best value.

Step 6. If the path length L_k is less than settle length, update pheromone.

Step 7. t=t+1

Step 8. If t is less than the settle times of iterations and the solution has not improvement, return to step 2.

Step 9. print the best solution.

The mutation operations are as fallowing: Selecting a gene in the gene cluster randomly, this gene is exchanged with his foregoing gene and the other is steadiness. For a example the chromosome is c0=[2,3,4,1,5,7,9,8,6],and the choosing gene is 3. So the mutation chromosome is c=[2,4,3,1,5,7,9,8,6].

The crossover operations are as fallowing: Selecting two chromosomes and crossover area randomly, the gene in the crossover area of the first chromosome replaces the gene in the crossover area of the second chromosome. Then the gene of the first chromosome disappeared in the crossover area of the second chromosome are deleted. For a example the first chromosome is c1=[1,2,3,4,5,6,7,8,9] and the second chromosome is c2=[9,8,7,6,5,4,3,2,1]. The crossover area is [6,5,4,3]. So the crossover chromosome is newc=[1,2,6,5,4,3,7,8,9].

2.2.4 Algorithm testing

In order to test this ACO-GA, we choose Oliver30 (best solution is 423.7406) and att48[20] (best solution is 33522) as examples. We compare the results obtained with that from Ant Colony optimization (ACO), GA, simulated annealing ($T = 100000$, $T_0 = 1$, $\alpha = 0.99$) .The results are reported in Table 1[21]. The best loops of Oliver30 and att48 are in Figure 1 and Figure 2. Every algorithm is tested for 20 times.

Algorithm	Oliver30			Att48		
	mean	best	worst	mean	best	worst
Simulate Anneal	438.522	424.692	478.831	35176	34958	40536
GA	483.457	467.684	502.574	38732	38541	42458
ACO	450.035	441.958	499.933	36532	35876	42234
ACO-GA	431.4987	423.7406	447.6865	33798	33524	36821

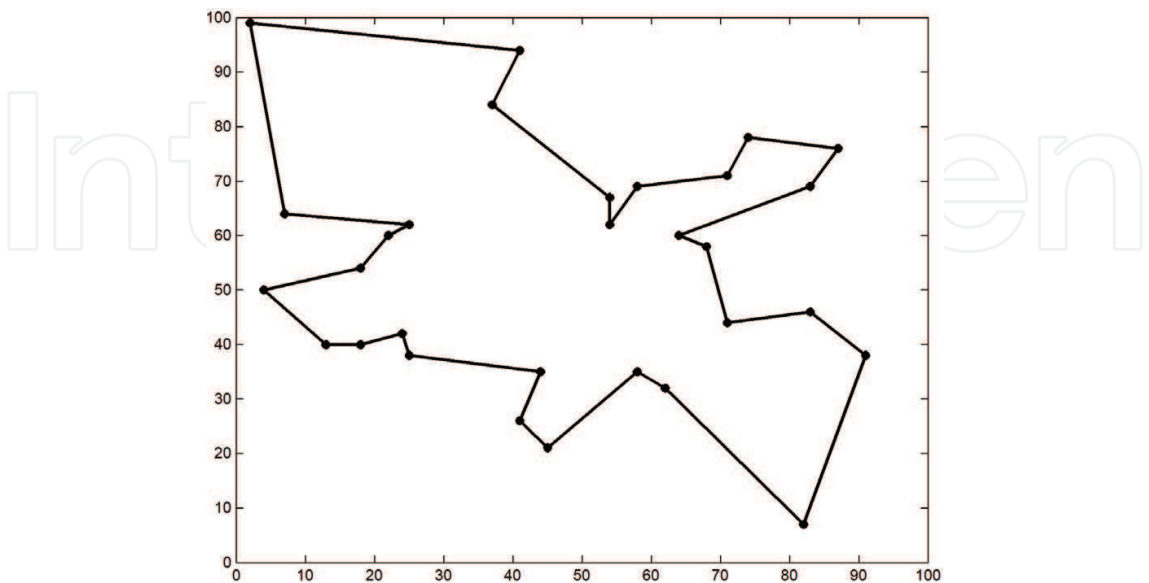Table 1. The results of Oliver30 and Att48 using four algorithms

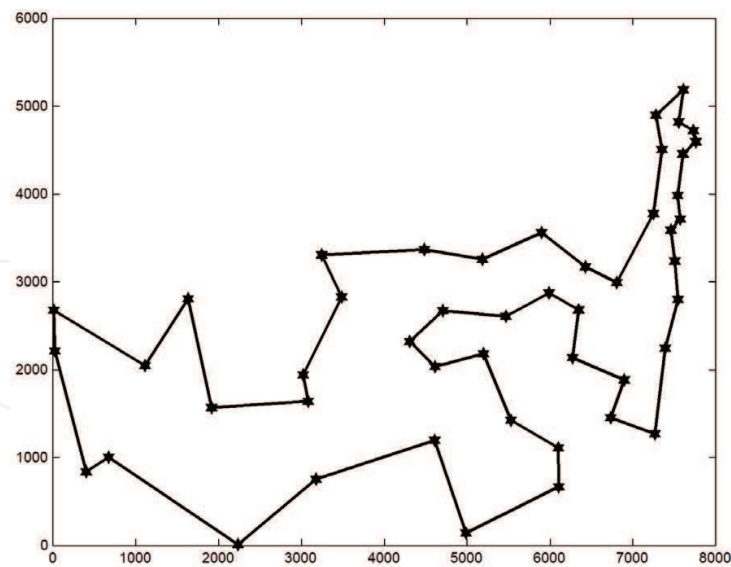Fig. 1. The best loops of Oliver30 using ACO-GA.

Fig. 2. The best loops of Att48 using ACO-GA.

2.3 Sports competitions scheduling with the ACO-GA

The following gives the explanation of several key operations in the scheduling. The data is in table2 and comes from Chinese Undergraduate Mathematical Contest in Modeling on Electrician of 2005.

2.3.1 Modeling of SCS

Let 1 replace # and 0 replace empty, we can get a 0-1 matrix $C_{14\times40}$. For a 14-game sports competition, the complete weighted graph G_{14} on 14 vertexes is considered. The vertex i ($i = 1, \ldots, 14$) represents the game, the edge $i-j$ presents the games between i, j $\{1, \ldots, 14\}$. The weighted function $D_{ij}(i = 1,2,...,14; j = 1,2,...14.)$ is the competitors who take part in both the games i and j. Summing the competitors who take part in both games i and j, we can get the weighted matrix $D_{14\times14}$ as following. The objective of SCS is to create a schedule of sports competitions in which the competitors taking part in two sequential games are minimized.

$$D_{14\times14} = \begin{pmatrix} 0 & 2 & 1 & 2 & 0 & 0 & 1 & 0 & 1 & 2 & 1 & 1 & 1 & 1 \\ 2 & 0 & 1 & 4 & 1 & 0 & 1 & 1 & 1 & 3 & 1 & 0 & 2 & 1 \\ 1 & 1 & 0 & 1 & 0 & 0 & 0 & 3 & 1 & 1 & 0 & 2 & 2 & 1 \\ 2 & 4 & 1 & 0 & 1 & 1 & 2 & 1 & 0 & 2 & 1 & 0 & 1 & 1 \\ 0 & 1 & 0 & 1 & 0 & 2 & 0 & 1 & 1 & 1 & 0 & 1 & 1 & 2 \\ 0 & 0 & 0 & 1 & 2 & 0 & 1 & 2 & 1 & 1 & 1 & 2 & 1 & 2 \\ 1 & 1 & 0 & 2 & 0 & 1 & 0 & 1 & 1 & 1 & 0 & 2 & 2 & 1 \\ 0 & 1 & 3 & 1 & 1 & 2 & 1 & 0 & 1 & 2 & 1 & 4 & 2 & 2 \\ 1 & 1 & 1 & 0 & 1 & 1 & 1 & 1 & 0 & 1 & 1 & 1 & 3 & 1 \\ 2 & 3 & 1 & 2 & 1 & 1 & 1 & 2 & 1 & 0 & 1 & 0 & 0 & 3 \\ 1 & 1 & 0 & 1 & 0 & 1 & 0 & 1 & 1 & 1 & 0 & 3 & 1 & 1 \\ 1 & 0 & 2 & 0 & 1 & 2 & 2 & 4 & 1 & 0 & 3 & 0 & 1 & 0 \\ 1 & 2 & 2 & 1 & 1 & 1 & 2 & 2 & 3 & 0 & 1 & 1 & 0 & 4 \\ 1 & 1 & 1 & 1 & 2 & 2 & 1 & 2 & 1 & 3 & 1 & 0 & 4 & 0 \end{pmatrix} \qquad (5)$$

Competitor \ Game	1	2	3	4	5	6	7	8	9	10	11	12	13	14
1		#	#						#				#	
2								#			#	#		
3		#		#						#				
4			#					#				#		
5											#		#	#
6					#	#								
7												#	#	
8										#				#
9		#		#						#	#			
10	#	#		#			#							
11		#		#									#	#
12								#		#				
13					#					#				#
14			#	#				#						
15			#					#				#		
16									#		#	#		
17						#								#
18							#					#		
19			#							#				
20	#			#										
21									#					#
22		#			#									
23							#					#		
24							#	#					#	#
25	#	#								#				
26					#									#
27						#					#			
28		#						#						
29	#										#	#		
30				#	#									
31							#					#		
32							#			#				
33				#		#								
34	#		#										#	#
35					#	#						#		
36				#			#							
37	#								#	#				
38							#		#		#			#
39					#			#	#				#	
40						#	#			#			#	

Table 2. The Registration Form for a Sports Competition

2.3.2 The result of SCS using ACO-GA

First we use GA to generate 100 solutions and select 30 excellent solutions from the 100 solution. Pheromone is left on the excellent solutions. Then ACO is executed, when ACO terminates, the crossover and mutation operations of GA execute on the solutions generated by ACO and generate new solutions. ACO and GA's crossover and mutation operations search alternately and cooperatively in the solution space. Using ACO-GA, we can get a loop of sports competitions, and then we cut the loop at one edge with the biggest weighted function. By this way we can get 3 game schedules using ACO-GA:

$$(1)\ 9 \rightarrow 4 \rightarrow 13 \rightarrow 10 \rightarrow 12 \rightarrow 14 \rightarrow 2 \rightarrow 6 \rightarrow 3 \rightarrow 7 \rightarrow 11 \rightarrow 5 \rightarrow 1 \rightarrow 8$$

$$(2)\ 13 \rightarrow 10 \rightarrow 12 \rightarrow 14 \rightarrow 2 \rightarrow 6 \rightarrow 3 \rightarrow 7 \rightarrow 11 \rightarrow 5 \rightarrow 1 \rightarrow 8 \rightarrow 9 \rightarrow 4$$

$$(3)\ 2 \rightarrow 6 \rightarrow 3 \rightarrow 7 \rightarrow 11 \rightarrow 5 \rightarrow 1 \rightarrow 8 \rightarrow 9 \rightarrow 4 \rightarrow 13 \rightarrow 10 \rightarrow 12 \rightarrow 14$$

Their objective function is two. That is to say, there are two competitors who take part in two successive games.

3. Conclusions

In this chapter, Based on the existed researches of sports competition scheduling (SCS) the author presented a hybrid Ant Colony Optimization and Genetic Algorithms (ACO-GA) to improve the productivity and efficiency of sports competition scheduling (SCS). The ACO-GA procedures of combining ant colony optimization and genetic algorithms were as follows: (1) applying the GA in a specified solution space to generate the activity lists, which provided an initial population for ACO; (2) performing ACO; (3) executing the crossover and mutation operations of GA to generate a new population based on the ACO result; (4) performing ACO and GA alternately and cooperatively in the solution space for n times. In order to test the ACO-GA algorithm, we selected Oliver30 and att48 problems. The results indicated that ACO-GA could find the best solution presented on the TSPLIB95. Finally we converted a sports competition scheduling (SCS) into a TSP model, and then using the ACO-GA algorithm. The result illustrated that the ACO-GA was a good method for solving the SCS. But it is likely that the worst case running time for ACO-GA algorithm for the SCS increases exponentially with the number of sports game.

4. Acknowledgment

5. References

Urban TL & Russell RA. Scheduling sports competitions on multiple venues. European Journal of Operational Research 2003; 148(2):302–11.

Russell RA & Leung JMY. Devising a cost-effective schedule for a baseball league. Operations Research 1994; 42(4):614–25.

Kiselyov O. Scheduling algorithms and NP-complete problems. Dr. Dobbs Journal 1997; 22(2):107–9.

Armstrong J & Willis RJ. Scheduling the cricket World Cup—a case study. Journal of the Operational Research Society 1993; 44(11):1067–72.

Willis RJ & Terrill BJ. Scheduling the Australian state cricket season using simulated annealing. Journal of the Operational Research Society 1994; 45(3):276–80.

Wright M. Timetabling county cricket fixtures using a form of tabu search. Journal of the Operational Research Society 1994; 45(7):758–70.

Shirland LE. Computerized dressage scheduling. Interfaces 1983; 13(3):75–81.

Fleurent C & Forland JA. Allocating games for the NHL using integer programming. Operations Research 1993; 41(4):649–54.

Costa D. An evolutionary tabu search algorithm and the NHL scheduling problem. INFOR 1995; 33(3):161–78.

Della Croce F & Tadei R, Asioli PS. Scheduling a round robin tennis tournament under courts and players availability constraints. Annals of Operations Research 1999; 92:349–61.

Corominas A. SUCCESS92—a DSS for scheduling the Olympic Games. Interfaces 1989; 19(5):1–12.

M. Dorigo & G.D. Caro, Ant colony optimization: a new meta-heuristic, Proc. 1999 Congress Evol. Comput. 2 (1999) 1470–1477.

M. Dorigo & T. Stu¨tzle, Ant Colony Optimization, The MIT Press, 2004.

Dorigo M, ManiezzoV & Colorni A. The ant system: an autocatalytic optimization process. Technical Report 91–016, Department of Electronics, Politecnico di Milano, Italy; 1991.

Dorigo M, Maniezzo V & Colorni A. The ant system: optimization by a colony of cooperating agents. IEEE Transactions on Systems, Man, and Cybernetics 1996;26:1–13.

Dorigo M & Di Caro G. The ant colony optimization metaheuristic. In: Corne D, Dorigo M, Glover F, editors. New ideas in optimization. New York: McGraw-Hill; 1999. p. 11–32.

Dorigo M & Blum C. Ant colony optimization theory: a survey. Theoretical Computer Science 2005;344:243–78.

Z.-J. Lee, C.-Y. Lee & S.-F. Su, An immunity based ant colony optimization algorithm for solving weapon-target assignment problem, Appl. Soft Comput. 2 (2002) 39–47.

Holland J. Adaptation in natural and artificial systems: an introductory analysis with applications in biology, control, and artificial intelligence. Ann Arbor, MI: University of Michigan Press; 1975.

Garey M. R & Johnson D. S.. Computers and Intractability: A guide to the Theory of NP-Completeness. San Francisco: W.H. Freeman, 1979.

http://www.iwr.uni-heidelberg.de/groups/comopt/software/TSPLIB95/.

Gao S & Yang J.Y. Swarm Intelligence Algorithms and Applications. China Waterpower Press, 2006.

Ching-Jong Liao & Hsiao-Chien Juan. An ant colony optimization for single-machine tardiness scheduling with sequence-dependent setups. Computers & Operations Research 34 (2007) 1899–1909.

Zne-Jung Lee & Shun-Feng Su, Chen-Chia Chuang, Kuan-Hung Liu. Genetic algorithm with ant colony optimization (GA-ACO) for multiple sequence alignment. Applied Soft Computing (2006)

Issmail Ellabib, Paul Calamai & Otman Basir Exchange strategies for multiple Ant Colony System, Information Sciences 177 (2007) 1248–1264

Walter J. Gutjahr, First steps to the runtime complexity analysis of ant colony optimization, Computers & Operations Research

Robert A. Russell & Timothy L. Urban, A constraint programming approach to the multiple-venue, sport-scheduling problem Computers & Operations Research 33 (2006) 1895-1906

De Werra D. Geography, games and graphs. Discrete Applied Mathematics 1980;2(4):327-37.

De Werra D. Minimizing irregularities in sports schedules using graph theory. Discrete Applied Mathematics 1982;4(3):217-26.

Schreuder JAM. Combinatorial aspects of construction of competition Dutch professional football leagues. Discrete Applied Mathematics 1992;35(3):301-12.

Nemhauser GL & Trick MA. Scheduling a major college basketball conference. Operations Research 1998;46(1):1-8.

Ferland JA & Fleurent C. Computer aided scheduling for a sport league. INFOR 1991;29(1):14 – 25.

Costa D. An evolutionary tabu search algorithm and the NHL scheduling problem. INFOR 1995;33(3):161–78.

Bean JC, Birge JR. Reducing travelling costs and player fatigue in the National Basketball Association. Interfaces 1980;10(3):98-102.

Wright MB. Scheduling fixtures for Basketball New Zealand.Working Paper, Lancaster University Management School, 2003.

Cain WO. The computer-aided heuristic approach used to schedule the major league baseball clubs. In: Ladany SP, MacholRE, editors. Optimal strategies in sports. Amsterdam: North-Holland; 1977. p. 32–41.

Armstrong J & Willis RJ. Scheduling the cricket world cup—a case study. Journal of the Operational Research Society1993;44(11):1067–72.

Willis RJ & Terrill BJ. Scheduling the Australian state cricket season using simulated annealing. Journal of the Operational Research Society 1994;45(3):276 – 80.

Wright M. Timetabling county cricket fixtures using a form of tabu search. Journal of the Operational Research Society1994;45(7):758 – 70.

Della Croce F, Tadei R & Asioli PS. Scheduling a round robin tennis tournament under courts and players availability constraints. Annals of Operations Research 1999; 92:349–61.

http://en.wikipedia.org/wiki/Travelling_salesman_problem
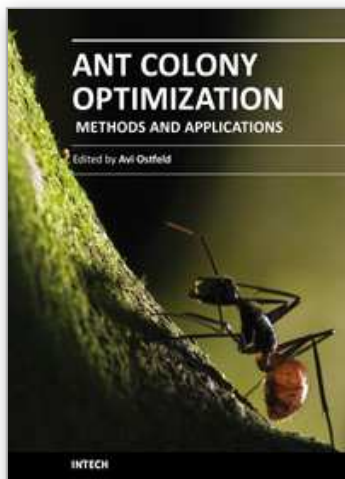
Ant Colony Optimization - Methods and Applications

Edited by Avi Ostfeld

Ants communicate information by leaving pheromone tracks. A moving ant leaves, in varying quantities, some pheromone on the ground to mark its way. While an isolated ant moves essentially at random, an ant encountering a previously laid trail is able to detect it and decide with high probability to follow it, thus reinforcing the track with its own pheromone. The collective behavior that emerges is thus a positive feedback: where the more the ants following a track, the more attractive that track becomes for being followed; thus the probability with which an ant chooses a path increases with the number of ants that previously chose the same path. This elementary ant's behavior inspired the development of ant colony optimization by Marco Dorigo in 1992, constructing a meta-heuristic stochastic combinatorial computational methodology belonging to a family of related meta-heuristic methods such as simulated annealing, Tabu search and genetic algorithms. This book covers in twenty chapters state of the art methods and applications of utilizing ant colony optimization algorithms. New methods and theory such as multi colony ant algorithm based upon a new pheromone arithmetic crossover and a repulsive operator, new findings on ant colony convergence, and a diversity of engineering and science applications from transportation, water resources, electrical and computer science disciplines are presented.

How to reference

In order to correctly reference this scholarly work, feel free to copy and paste the following:

Huang Guangdong and Wang Qun (2011). A Hybrid ACO-GA on Sports Competition Scheduling, Ant Colony Optimization - Methods and Applications, Avi Ostfeld (Ed.), ISBN: 978-953-307-157-2, InTech, Available from: http://www.intechopen.com/books/ant-colony-optimization-methods-and-applications/a-hybrid-aco-ga-on-sports-competition-scheduling

INTECH
open science | open minds