Article ID: 1006-8775(2010) 03-0280-12

EXPERIMENTS TO TRACK STORMS USING MODERN OPTIMIZATION ALGORITHMS

LI Nan (李 南)[1, 2], WEI Ming (魏 鸣)[1, 3]

(1. Key Laboratory of Meteorological Disaster of Ministry of Education, Nanjing University of Information Science & Technology, Nanjing 210044 China; 2. School of Atmospheric Physics, Nanjing University of Information Science & Technology, Nanjing 210044 China; 3. State Key Laboratory of Severe Weather, Chinese Academy of Meteorological Sciences, Beijing 100081 China)

Abstract: Storm identification and tracking based on weather radar data are essential to nowcasting and severe weather warning. A new two-dimensional storm identification method simultaneously seeking in two directions is proposed, and identification results are used to discuss storm tracking algorithms. Three modern optimization algorithms (simulated annealing algorithm, genetic algorithm and ant colony algorithm) are tested to match storms in successive time intervals. Preliminary results indicate that the simulated annealing algorithm and ant colony algorithm are effective and have intuitionally adjustable parameters, whereas the genetic algorithm is unsatisfactorily constrained by the mode of genetic operations. Experiments provide not only the feasibility and characteristics of storm tracking with modern optimization algorithms, but also references for studies and applications in relevant fields.

Key words: raw weather radar data; storm identification and tracking; optimization algorithms

CLC number: P457.9 **Document code:** A **doi:** 10.3969/j.issn.1006-8775.2010.03.010

1 INTRODUCTION

With the development of weather radar and computer technology, storm identification and tracking using radar systems are becoming an important technique for severe weather nowcasting. Using weather radar data, the storm identification, tracking and nowcasting is to detect storms already in existence, calculate their physical parameters, and match and track them based on continuous radar images to establish corresponding relations, and extrapolate their evolution and motion.

Storm identification can be roughly classified into two categories: a cross-correlation method and a centroid method. With reflectivity data, the cross-correlation method calculates the motion vectors of subregions to obtain optimal correlation between subregions (Rinehart and Garvey[1]; Li et al.[2]). Due to the focus on regional scales, this method is difficult to obtain detailed storm information. By contrast, the centroid method (Crane[3]; Rosenfeld[4]; Handwerker[5]) can provide structure and evolution characteristics of storms. As a result, the centroid method draws more attention.

After one or several reflectivity thresholds are determined, the commonly used centroid method takes three steps to perform three-dimensional identification: (1) contiguous sequences of points are searched and recorded along radials to constitute storm segments if reflectivity meets the threshold; (2) storm segments are searched and recorded to constitute storm components if continuity requirement is satisfied along neighboring radials on the same PPI scan; (3) storm components are searched and recorded to constitute storms on the PPI scans of different elevations. Currently, influential algorithms TITAN (thunderstorm identification, tracking, analysis and nowcasting) (Dixon and Wiener[6]) and SCIT (storm cell identification and tracking) (Johnson et al.[7]) are based on the steps described above. Its defect lies in that it needs to seek one-dimensional storm segments in radials and then compose them to two-dimensional storm components in azimuths based on specified continuity requirement, probably distorting shapes and boundaries of storms to cause identification error.

After storms are identified in successive time intervals, they are matched and tracked for extrapolation forecast. The centroid tracking method assumes that the centroid can represent the storm location; thus, motion vectors can be obtained with corresponding distance changes through continuous radar echoes. Therefore, the centroid tracking method can provide detailed information about the motion and evolution of storms. Various echo parameters can be used for storm matching, including shape, area, centroid, azimuth, delay time, intensity distribution, moment vector, etc. Area

Received date: 2010-01-29; **revised date:** 2010-06-18

Foundation item: Natural Science Foundation of Jiangsu province (BK2009415); Research Fund for the Doctoral Program of Higher Education of China (20093228110002); Key University Science Research Project of Jiangsu Province (201037B); National 863 Project (2007AA061901); Project of State Key Laboratory of Severe Weather of Chinese Academy of Meteorological Sciences (2010LASW-A01); College Graduate Student Research and Innovation Program of Jiangsu province (CX09B_227Z); Project 2009Y0006

Biography: LI Nan, Ph.D. candidate, primarily undertaking research on radar meteorology and disastrous weather forecast.

E-mail for corresponding author: shangjineh@163.com

and centroid location are the two parameters usually used to match storms (Xiao et al.[8]).

This study mainly discusses modern optimization algorithms on storm matching and tracking widely used in various fields. First, a simple and feasible two-dimensional storm identification method was designed to overcome the defect of the commonly used centroid identification method. The new method was able to identify storm cells in three successive time intervals based on data from a next-generation Doppler weather radar (NEXTRAD) in Nanjing. Then, storm matching was conducted using the general Hungarian method and manually to obtain results for reference. Afterwards, the three modern optimization algorithms described above were applied to match the storms. The new storm identification method is described in section 2. Storm matching and tracking is explained in section 3. The three modern optimization algorithms are briefly introduced in section 4. Storm matching experiments on the three optimization algorithms are carried out in section 5. A case that cannot be solved with the Hungarian method is presented in section 6 for further discussion of the three optimization algorithms, and conclusions are given at the end.

2 THE NEW STORM IDENTIFICATION METHOD

To overcome the defect of the storm identification method used in TITAN and conduct a test on the algorithms, a new storm identification method, able to search contiguous points simultaneously in two dimensions, was designed. The basic idea of this method is simple. First, a threshold is determined in advance. Then points with reflectivity exceeding the threshold are identified to serve as storm points. Afterwards, contiguous storm points constituting a storm are sought group by group until all storms have been searched. Contiguous points are defined as the eight points around a reference point on a two-dimensional plane.

Before searching, all contiguous storm points are located. In the first round of search, a storm point serves as the initial reference point to search for all other storm points in the second round; the initial reference point is excluded in this round to avoid repeated search. The storm points contiguous to the initial reference point in the second round serve as the reference points in the third round; after search, each reference point and its contiguous points are eliminated from the round of search by the next reference point to avoid repeated search. This procedure is repeated one after another until no storm points can be sought contiguous to the reference points in a search round. Then the storm points that have been sought in these rounds constitute a storm. Other storms are searched in the same way until all storm points have been sought. Figure 1 is a schematic diagram of the method of searching contiguous points. After all storm cells are identified, those with area less than a preset threshold are eliminated for noise filtering.

With the new method, storms in three successive time intervals (at 0949 UTC, short for Coordinated Universal Time, 0955 UTC and 1001 UTC on 17 May 2006, and represented by t_1, t_2 and t_3, respectively) were identified in terms of the reflectivity data of the lowest angle of elevation or PPI collected by the NEXTRAD in Nanjing. The reflectivity and area thresholds were set to be 40 dBZ and 10 km^2, respectively. Figure 2 shows the results identified with the new method and storm serial numbers. The total number of storms was 13, 14 and 10 for t_1, t_2 and t_3, respectively.

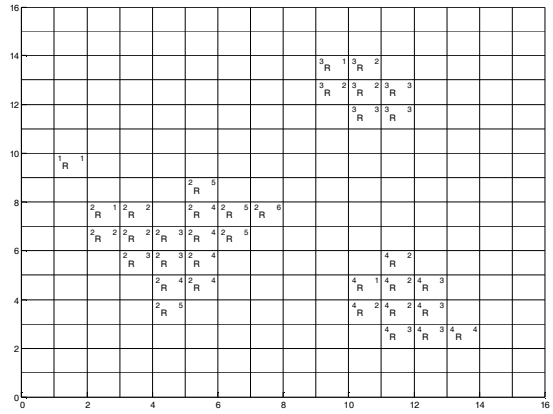

Fig. 1. Schematic diagram of the contiguous points searching method. R represents a storm point with reflectivity exceeding the threshold. The numeral at the top left corner of a box represents the serial number of the storm to which a storm point belongs. The numeral at the top right corner represents the serial number of the round in which a storm point is being searched.

3 STORM MATCHING AND TRACKING

When storms at successive points of time are identified, they need to be matched with each other to determine corresponding storms. Hence, storm matching between two successive points of time is to decide which logically possible path is most likely the true one. If matching is done in successive time intervals, storms can be tracked through their entire life cycles. Storm matching is conducted based on two main assumptions: 1) A correct match connects storms by shorter rather than longer paths; 2) A correct match joins storms of similar characteristics.

It is supposed that there are n_1 storms at t_1 and storm i at t_1 is in a state $s_{1i}=(x_{1i}, y_{1i}, a_{1i})$, and there are n_2 storms at t_2 and storm j at t_2 is in a state $s_{2j}=(x_{2j}, y_{2j}, a_{2j})$. The cost function C_{ij} (in unit of distance) can be defined as $C_{ij}=w_1 \cdot d_p+w_2 \cdot d_a$; the difference in position between the centroid of storm i at t_1 and that of storm j at t_2 is $d_p=[(x_{2j}-x_{1i})^2+(y_{2j}-y_{1i})^2]^{1/2}$, the difference in area (also in unit of distance, because of the square root) between storm i and storm j is $d_a=\left|(a_{2j})^{1/2}-(a_{1i})^{1/2}\right|$, and w_1 and w_2 are weight factors (both set at 1.0 here). In this way, the problem of determining the true set of storm paths may be posed and solved as one of optimization. The optimal set of paths is to find the match that minimizes the objective function $Q=\Sigma C_{ij}$, where i refers to the start point of a path (storm i at t_1) and j the corresponding end point (storm j at t_2), and the summation is performed over all possible sets of storm paths. It is assumed that the optimal set of paths and the true set are the same.

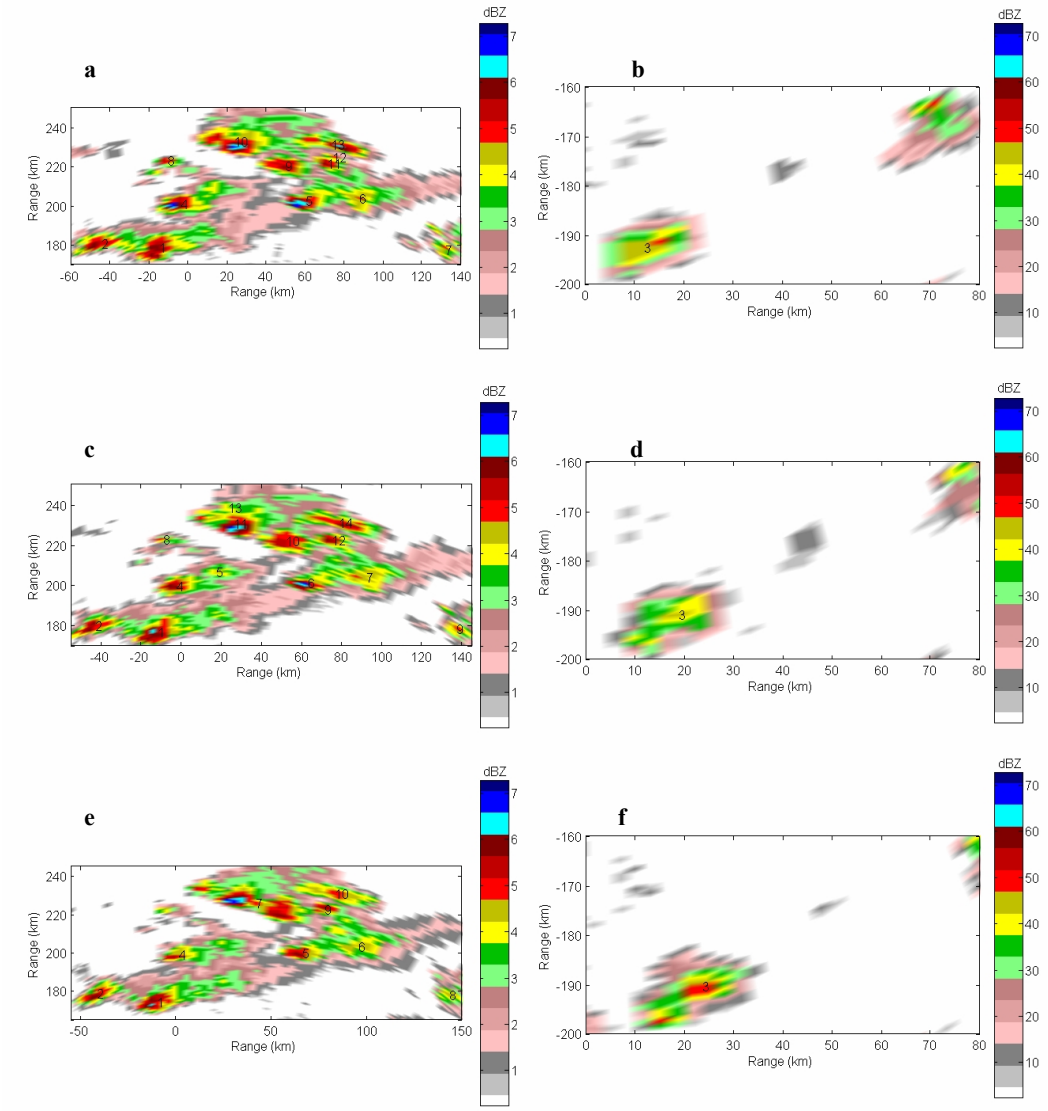

Fig. 2. Results for t_1 (a, b), t_2 (c, d) and t_3 (e, f) at 0.5° angle of elevation or PPI with the new identification method. Identified storms are each assigned with an initial serial number.

In fact, storm matching is equivalent to the assignment of tasks widely used in everyday life. A common assignment problem can be described as follows (workers and tasks are taken for example). There are m tasks that must be accomplished by n workers (unequal quantity between workers and tasks is allowed in this study with $n \geqslant m$). Each task must be accomplished by only one worker and each worker can accomplish one task at most. Each worker performs better at some tasks than at others and obviously the costs at which he accomplishes different tasks are different. The goal is how to assign the workers to the tasks in order to minimize the total cost for accomplishing all of the tasks.

A solution to the assignment problem is just a permutation selecting m tasks from n workers and namely, worker N ($1 \leq N \leq n$) is assigned to accomplish task M ($1 \leq M \leq m$). The feasible solution number of the assignment problem is P_n^m. Therefore, permutation form $S=(P_1, P_2, \cdots, P_n)$ is adopted for storm matching results in this article. If storm j at t_2 is matched with storm i at t_1, $P_i=j$. If no storm at t_2 matches with storm i at t_1 (storms at t_1 are more than those at t_2), $P_i=0$.

If the cost at which worker i accomplishes task j is C_{ij} ($i=1,2,\cdots, n; j=1,2,\cdots, m$), the matrix

$$C = (C_{ij})_{n\times m} = \begin{bmatrix} C_{11}C_{21}\ldots C_{1m} \\ C_{21}C_{22}\ldots C_{2m} \\ \vdots \\ C_{n1}C_{n2}\ldots C_{nm} \end{bmatrix} \quad (1)$$

is termed the cost matrix, in which elements in row i represent the costs at which worker i accomplishes the tasks and elements in column j represent the costs of task j accomplished by workers. 0–1 variables are introduced. If worker i is assigned to accomplish task j, $x_{ij}=1$; otherwise $x_{ij}=0$. The mathematic model to minimize the total cost for the assignment problem can be written as (Li[9])

$$\min z = \sum_{i=1}^{n}\sum_{j=1}^{m} C_{ij}x_{ij}$$

$$s.t.\begin{cases}\sum_{i=1}^{n} x_{ij}=1, i=1,2,\ldots,n;\\ \sum_{j=1}^{m} x_{ij}=1, j=1,2,\ldots,m;\\ x_{ij}\in\{0,1\}, i=1,2,\ldots,n, j=1,2,\ldots,m.\end{cases} \quad (2)$$

It means that given an $n\times m$ matrix C_{ij}, it is to find an $n\times m$ matrix X_{ij} such that the following holds: 1) In any given row or column, X_{ij} has exactly one non-zero element and that element has the value of 1; 2) The sum of $C_{ij}X_{ij}$ over all i, j is a minimum. Generally, the Hungarian method is implemented for the assignment problem to seek matrix X_{ij} (Lawler[10]; Roberts[11]). Detailed operations of the Hungarian method can be referred to in corresponding references.

Based on the identification results of t_1, t_2 and t_3, storms are matched between t_1 and t_2 (Case 1), and between t_2 and t_3 (Case 2). Merger or split storms are excluded before matching; storms 11 and 12 at t_1 merged to storm 12 at t_2, and storms 10 and 11 at t_2 merge to storm 7 at t_3 (Fig. 2). The rest of the storms are assigned with new serial numbers according to their previous numbers. At t_1, for example, the numbers of the first ten storms remain the same whereas storm 13 is given a new number 11 to fill up the vacancy. The cost matrices of the two cases are B_1 and B_2 (see Appendix).

Because there are more storms at t_1 than at t_2, for the purpose of conforming to the defined concept of the assignment problem and in favor of the uniform calculation and comparison, t_1 and t_2 exchange their orders in storm matching in Case 1 (the cost matrix just needs a transposition). Matching results given by the Hungarian method are: 1, 2, 3, 4, 0, 5, 6, 8, 7, 9, 10, 0, 11 (Case 1), and 1, 2, 3, 4, 0, 5, 6, 0, 7, 8, 0, 9 (Case 2). The corresponding total cost values are 52.083 and 42.082, respectively. Manual analysis agrees with the matching results (Fig. 2).

4 BRIEF INTRODUCTION TO MODERN OPTIMIZATION ALGORITHMS

When the assignment problem becomes rather complex (for instance, the order of the cost matrix is quite large) or for some particular calculation processes (Chu[12]; Gu et al.[13]), the Hungarian method may not resolve the problem effectively. More and more studies use the heuristic algorithms—widely discussed and applied since the 1980s—to solve a large amount of actual assignment problems. These algorithms mainly include the simulated annealing algorithm, genetic algorithm and ant colony algorithm. Through comparing with the solution to the optimization problem based on some natural phenomena in the objective world, common characters are identified to establish corresponding algorithms and optimal solutions are achieved by iteration. The three algorithms are briefly introduced as follows (Xing and Xie[14]).

4.1 *Simulated annealing algorithm*

The simulated annealing algorithm was invented by Metropolis in 1953 and applied successfully in the optimization problem by Kirkpatrick in 1983.

Annealing denotes a physical process in which a metal object is heated up by increasing the temperature to a certain value at which all molecules randomly arrange themselves in the phase space. With decreasing temperature, these molecules gradually keep themselves in different states. Statistical mechanics indicates that at temperature T, the probability of being in a state r for a molecule is given by the Boltzmann distribution

$$P\{\bar{E}=E(r)\}=\frac{1}{Z(T)}\exp(-\frac{E(r)}{k_BT}),$$ where $E(r)$ represents the energy in state r, $k_B>0$ is the Boltzmann constant, $\bar{E}$ is a random variable denoting the molecule energy, and $Z(T)$ is a normalization factor.

It can be seen from the Boltzmann distribution that the probability of a molecule being in a state with lower energy is larger than that with higher energy. When temperature is sufficiently high, the probability of a molecule being in each state is almost the same and closes to the mean value. As temperature decreases, the Boltzmann distribution concentrates on states with the lowest energy; as temperature approaches zero, only the states with minimum energy have a non-zero probability of occurrence.

Simulated annealing can be analogous to the optimization problem so that the solution to this problem can be corresponding to the state with the lowest energy in the annealing process, which has the maximum probability for molecules at the minimum temperature.

There are an internal and an external loop in the simulated annealing algorithm. The internal loop denotes a random searching among some states at the same temperature, and the external loop includes temperature decrease and terminating conditions. An intuitive understanding of the simulated annealing is that the searching is randomly switched from one state to another at a certain temperature. The times the state is searched with obey a probability distribution; when temperature is sufficiently low, an optimal solution is achieved with the probability of 1. Each iterative step reflects equilibrium between the concentration and diffusion policies. When a given successive iterative solution is better, the concentration policy is used and this solution is made the new solution; when the successive iterative solution is not better, the diffusion policy is used and this solution becomes the new solution subject to a certain probability.

4.2 *Genetic algorithm*

The genetic algorithm was developed by Professor Holland from the University of Michigan at the beginning of the 1970s. An initial achievement was the publication of *Adaptation in Natural and Artificial Systems* by Holland.

The genetic algorithm takes up the concept of survival of the fittest in biological evolution. In other words, a population more suitable to the natural environment generates more progenies. The genetic algorithm uses some biological evolution characteristics for references. (1) Evolution takes place at solution codes which are called chromosomes. All properties of the optimization problem are studied through these codes. (2) Natural selection determines that the chromosomes have the capacity of generating more progenies than the others through an artificial fitness function according to the aim of the optimization problem. (3) When crossover is carried out for chromosomes, the recombination of genes makes the children keep the characteristics of their parents. (4) Random mutation can cause the children to be different from their parents.

The biological evolution can be analogous to the optimization problem so that the solution of the optimization

problem can be corresponding to individuals having the largest number after evolutions of the original population.

The genetic algorithm primarily includes steps as follows. First, coding is needed for the optimization problem. A solution code is called a chromosome and the components constituting the chromosome are called genes. Coding is used to indicate the solution pattern and to help the calculation in the genetic algorithm. Second, a fitness function is constructed, which basically depends on the cost function of the optimization problem. The chromosomes surviving or dying out are determined by the probability distribution calculated from the fitness function under the natural selection rule. The surviving chromosomes make up of the population which can generate individuals of the next generation. Thirdly, crossover is performed. The recombination of genetic genes produces the next generation through the crossover of codes. The emergence of a new generation is a reproduction process, creating new solutions. The last one is the mutation. Gene mutation takes place through the production of a new solution and it can change codes of some solutions, bringing more ergodicity.

4.3 *Ant colony algorithm*

The ant colony heuristic algorithm was proposed by Marco Dorigo in 1992 in his PhD thesis based on which a fine optimization algorithm has been gradually developed. This algorithm is a distributed intelligence simulation algorithm, with its basic idea imitating the social conduct of ants during communication depending on the pheromone.

In the real world, ants in a colony communicate with each other through the pheromone medium. Ants wander randomly, and upon finding food they return to their own colony while laying down some trails of chemical substances called pheromone. The pheromone can be sensed by other ants in the same colony, and serves as a signal affecting the latecomers. Therefore, ants are more likely to follow the paths with pheromone rather than those without pheromone, and they reinforce the original pheromone iteratively. As a result, the more ants pass a path, the more probably ants will select it. Over a given period of time, the shorter the path, the more ants will visit it, accumulating more pheromone. In the subsequent period of time, the probability for this path to be visited by other ants is larger than any other paths. This positive feedback eventually results in all the ants following a single path.

The social conduct in an ant colony can be analogous to the optimization problem so that the solution of the optimization problem can be corresponding to the final path selected by all the ants through the pheromone.

Imitating the behavior of ants, the paths of artificial ants in the ant colony algorithm are determined by two categories of parameters. One is the pheromone value or the pheromone trail, which is the memory information of ants. The other is the visibility value or the priori value, which is the cost of paths. The update of the pheromone is implemented by two operations. One is the evaporation; it is to reduce the pheromone as in the natural world. Pheromone evaporation also has the advantage of avoiding a fast convergence to a local optimal solution and extending the search range. The other is the enhancement. The pheromone on a path with a high appraisal value is enhanced to make the optimal solution converge stably.

The movement of an ant is executed by a random decision rule that uses the information stored in corresponding nodes to calculate the probability of subsequent nodes the ant could reach. According to the probability distribution, the ant moves a step by which the solution yielded by the ant colony gradually approaches the optimal solution. When an ant finds a solution or in the process of finding a solution, it can evaluate the optimization degree of this solution (or parts of this solution) and save the evaluated information in the corresponding pheromone trail which can conduct the ant in future seeking.

5 TRACKING STORMS WITH MODERN OPTIMIZATION ALGORITHMS

Based on the cost matrices of the storm identification results for Case 1 and Case 2, the simulated annealing algorithm, genetic algorithm and ant colony algorithm are used to track storms. For illustration and convenience, all three algorithms adopt explicit decimal coding for the permutation solution (S=(P_1, P_2,⋯, P_n)) of the assignment problem. It is the most common coding for assignment problems based on which operators are implemented.

5.1 *Storm matching with simulated annealing algorithm*

5.1.1 ALGORITHM STEPS

The following steps are referred to in Duan and Chen[15] and Wu and Dong[16].

Step 1. Original temperature t_0 and original solution i_0 are given. Some permutations representing different states of molecules are randomly created which constitute the solution space D. $t_0=K\delta$ is adopted for the original temperature, where K is a sufficiently large number and $\delta=\max\{f(s)|s\in D\}-\min\{f(s)|\ s\in D\}$. $\max\{f(s)|s\in D\}$ and $\min\{f(s)|\ s\in D\}$ are the maximum and minimum cost value in the solution space, respectively. The solution with the minimum cost value is selected as the original solution i_0, representing the state with the lowest energy at the original temperature.

Step 2. If the terminating condition of the internal loop is satisfied at temperature t_k, go to Step 3; otherwise a new permutation is created by exchanging the location of two randomly selected elements in the permutation. This process is repeated to obtain a neighborhood $L(i)$ constituted by several new solutions. A permutation j is selected randomly from $L(i)$ and the cost difference between i and j, $\Delta f_{ij}=f(j)-f(i)$, is calculated. If $\Delta f_{ij}<0$, i is replaced by j; otherwise if $\exp(-\Delta f_{ij}/t_k)>r$ (r is a random variable distributed between 0 and 1), i is replaced by j. Step 2 is repeated. Given n workers and m tasks, the number of newly created solutions would not exceed C_n^2. Therefore, iteration times $n(n-1)/2$ is set to be the terminating condition of the internal loop.

Step 3. Temperature t_k is decreased gradually. When it is below the terminating temperature t_e during the cooling process, the algorithm is terminated; otherwise go back to Step 2. $t_{k+1}=\alpha t_k$ is chosen as the cooling function, where α is the cooling coefficient.

5.1.2 ALGORITHM EXPERIMENTS

Because the probability of the Boltzmann distribution decreases exponentially as temperature decreases, it is increasingly less probable for a molecule to stay in an unstable state and increasingly easy for it to stay in a stable state. Therefore, the temperature decrease itself is a converging process. When temperature approaches 0, the probability

approaches 1 for the lowest energy states and the corresponding solution is close to the optimal. Experiments on the two cases indicate that results converge to the optimal solution more quickly as the terminating temperature decreases and the cooling coefficient increases. The iteration process has a larger neighborhood space for searching as the cooling coefficient increases, and the molecule is much easier to be in the most stable energy state as the terminating temperature decreases. However, both of them could remarkably affect the calculation time. When a small cooling coefficient and a big terminating temperature are given, temperature decreases to the terminating point quickly, yielding a short calculation time. When a big cooling coefficient and a small terminating temperature are given, temperature decreases slowly, yielding a relatively long calculation time. Because there would not be many storms in each time interval, an appropriate cooling coefficient and terminating temperature can be acquired to achieve a desirable result.

The terminating temperature is set to 5, 1 and 0.1, and the cooling coefficient is set to 0.2, 0.5 and 0.8, respectively. Each scheme is tested 10 times to determine mean total costs, total cost standard deviations, appearing times of correct solution and average calculation time.

Table 1. Calculation results for Case 1

Terminating temperature	Cooling coefficient	Mean total cost	Total cost standard deviation	Appearing times of correct solution	Average calculation time/s
5	0.2	231.51	60.195	0	0.036
	0.5	103.83	44.321	3	0.078
	0.8	64.445	21.66	7	0.22
1	0.2	201.43	172.45	0	0.041
	0.5	54.7	8.277	9	0.091
	0.8	52.083	0	10	0.269
0.1	0.2	61.634	21.763	8	0.053
	0.5	53.451	4.328	9	0.111
	0.8	52.083	0	10	0.336

Table 2. Calculation results for Case 2. Parameter settings and experiment items are the same as in Table 1.

Terminating temperature	Cooling coefficient	Mean total cost	Total cost standard deviation	Correct solution appearing times	Average calculation time/s
5	0.2	196.05	89.633	0	0.03
	0.5	66.784	23.405	3	0.066
	0.8	45.192	6.556	8	0.189
1	0.2	95.545	64.391	4	0.036
	0.5	47.217	16.24	9	0.078
	0.8	42.082	0	10	0.224
0.1	0.2	62.779	32.284	6	0.044
	0.5	42.082	0	10	0.094
	0.8	42.082	0	10	0.281

5.2 *Storm matching with genetic algorithm*

5.2.1 ALGORITHM STEPS

The following steps are referred to in Li et al.[17], Zhang and Xiu[18] and Wu[19].

Step 1. N permutations are created randomly. They are the first generation $P(1)$ having N chromosomes.

Step 2. The fitness value f_i is calculated for each chromosome $p_i(g)$ in generation $P(g)$. The fitness function is an index to measure the chromosome fitness, and serves as a basis in genetic operations. $f_i=k*C_{\max}-c_i$ is adopted as the fitness function, where c_i is the cost of an individual, $C_{\max}$ is the maximum cost in current population and k is a constant (set to be 1.1 here). It can be seen that an individual with a small cost value has a big fitness value and thus the probability with which it is preserved is big.

Step 3. If the terminating condition is satisfied, the algorithm is terminated; otherwise the roulette wheel probability is calculated for each individual. A maximum generation number G is taken as the terminating condition in this article. The roulette wheel probability has the form of $p_i = f_i / \sum_{i=1}^{N} f_i$, where p_i is the selecting probability of individual i, f_i is the fitness value of i, and $\sum_{i=1}^{N} f_i$ is the accumulated fitness value of generation $P(g)$.

Step 4. In order to keep the genetic information of good individuals, N_D individuals in generation $P(g)$ are selected according to the roulette wheel probability of fitness values to be copied into the next generation. $DP(g+1)$ for the next generation is obtained after the copy operation.

Step 5. N_C individuals in generation $P(g)$ are selected according to the roulette wheel probability of fitness values, and crossover is executed for each pair of individuals. A place for crossover is chosen randomly and then the sections of the gene are exchanged between the front and back of the crossover place for the two individuals. For instance, parent A is 501243 and parent B is 251304. The crossover place is chosen to be 4 so that the two children are child a 501204 and child b 251314, respectively. It can be seen that crossover can lead to irrational individuals. Therefore, additional adjustment is needed. Repeated elements are replaced with those who have not appeared. The two children would be 501234 and 531204 or 251340 and 251043 after the adjustment. $CP(t+1)$ for the next generation is obtained after the crossover operation.

Step 6. N_M individuals in generation $P(g)$ are selected according to the roulette wheel probability of fitness values. Two randomly selected elements in an individual are exchanged for mutation. $MP(g+1)$ is obtained for the next generation after the mutation operation. $N_D+N_C+N_M=N$, and the DP $(g+1)$, $CP(g+1)$ and $MP(g+1)$ compose the next generation. Go back to Step 2.

Studies have proved that genetic operators like crossover and mutation could destroy the gene part of high fitness values and generate bad individuals, leading to a local optimal solution instead of global optimal solution (Rudolph[20]). Nevertheless, genetic algorithms with the elitist strategy (De Jong[21]) can converge probabilistically to the global optimal solution (Eiben et al.[22]). The elitist strategy adopted in the article is as follows (Wang et al.[23]): (1) Two individuals with respectively maximum and minimum fitness values are identified; (2) If the fitness value of the optimal individual in the current generation has been less than that of the global optimal individual up to the present, the optimal individual serves as the global optimal; (3) The worst individual in the current generation is replaced with the global optimal.

5.2.2 ALGORITHM EXPERIMENTS

The population size is set to 40, 80 and 120, respectively. The maximum generation number is set to 50, 100 and 200, respectively. The copy, crossover and mutation ratios are set to ① (0.1, 0.8, 0.1), ② (0.2, 0.75, 0.05) and ③ (0.4, 0.5, 0.1), respectively. Each scheme is tested 10 times, achieving mean total costs, standard deviations total cost, appearing times of correct solution and average calculation time.

Table 3. Calculation results of Case 1

Population size	Maximum generation number	Ratio Scheme	Mean total cost	Total cost standard deviation	Correct solution appearing times	Average calculation time/s
40	50	①	173.81	85.995	1	0.503
		②	196.65	60.798	0	0.475
		③	221.99	56.332	0	0.331
	100	①	83	31.832	3	0.995
		②	102.12	23.188	1	0.934
		③	121.94	57.167	3	0.65
	200	①	59.463	16.362	8	1.939
		②	63.756	19.974	7	1.806
		③	54.819	5.77	8	1.302
80	50	①	162.08	68.292	0	1.023
		②	170.95	48.287	0	0.967
		③	152.43	72.102	1	0.683
	100	①	122.01	36.433	1	2.016
		②	91.121	37.116	3	1.87
		③	102.48	31.628	2	1.352
	200	①	71.521	38.233	7	4.008
		②	55.107	9.564	9	3.7
		③	67.997	32.454	7	2.639
120	50	①	160.83	45.865	0	1.555
		②	129.07	59.35	1	1.455
		③	174.74	47.427	0	1.034
	100	①	92.306	37.434	4	3.061
		②	88.62	42.188	4	2.873
		③	98.018	26.225	1	2.05
	200	①	65.328	22.125	7	6.033
		②	52.083	0	10	5.65
		③	60.55	18.74	8	4.063

Table 4. Calculation results of Case 2. Parameter settings and experiment items are the same as in Table 3.

Population size	Maximum generation number	Ratio Scheme	Mean total cost	Total cost standard deviation	Correct solution appearing times	Average calculation time/s
40	50	①	92.173	45.199	2	0.498
		②	139.99	50.27	0	0.472
		③	129.36	69.417	1	0.334
	100	①	73.398	33.744	3	1
		②	85.039	36.886	3	0.914
		③	80.966	38.484	3	0.656
	200	①	42.082	0	10	1.958
		②	45.601	7.48	8	1.825
		③	53.024	18.608	6	1.277
80	50	①	108.46	55.422	2	1.03
		②	116.72	39.747	0	0.983
		③	121.16	35.833	0	0.681
	100	①	59.646	28.788	6	2.019
		②	57.322	16.89	5	1.872
		③	74.563	30.38	4	1.4
	200	①	44.046	6.21	9	4.164
		②	42.082	0	10	3.702
		③	42.082	0	10	2.619
120	50	①	68.748	34.507	5	1.556
		②	98.137	24.444	1	1.442
		③	140.61	50.301	1	1.027
	100	①	42.082	0	10	3.069
		②	55.601	31.518	8	2.866
		③	42.082	0	10	2.027
	200	①	44.046	6.21	9	6.045
		②	42.082	0	10	5.578
		③	42.082	0	10	4.044

It can be seen that the maximum generation number has an important impact on the results. When it is relatively small, mean total costs and appearing times of correct solutions are both unfavorable in spite of the population size. Results become desirable as the maximum generation number increases. It means that the convergence process for the optimal solution is slow and needs multiple generations. In general, the genetic algorithm is unsatisfactory for storm tracking since it cannot achieve the optimal solution and wastes computation resources. That is caused by the characteristics of genetic operations, especially the crossover is playing a vital role in searching solution spaces. If the two parents are relatively excellent solutions with small costs, the children can be quite different from the parents due to the

breakdown of good gene segments and the adjustment of irrational solutions. It causes the iteration to deviate from the correct searching direction and causes the algorithm randomness to decline. Although the elitist strategy is included and the revised algorithm has remarkable improvement over the original one (figures and tables are omitted), results are still barely satisfactory.

Different relative ratios of the copy and crossover have different effects on the results. Large copy ratios and small crossover ratios can keep the good gene segments of individuals, but they also reduce the neighborhood for searching and lead to a local optimal solution. Large crossover ratios and small copy ratios can enlarge the neighborhood for searching, but lead to unstable solutions. Additionally, large crossover ratios can increase remarkably the calculation time. Therefore, proper ratios are needed to acquire a desirable result. Of the three ratio schemes the best results for the two cases are concluded in Table 5. It can be seen that for a given population size, the ratio scheme with the best results always changes along the direction at which the crossover ratio is, in turn, large (①), small (③) and moderate (②), as the maximum generation number increases. This phenomenon can be explained as follows. When the population size and the maximum generation number are relatively small, a large crossover ratio is needed to accomplish the neighborhood searching due to few iteration times. As a result, it is the neighborhood searching that is most beneficial to the optimal solution so that Scheme ①—with a large crossover ratio—is the best. When the population size and the maximum generation number are moderate, a large crossover ratio could cause the solution to be unstable and away from the optimal solution. In this case it is the steady convergence that is most beneficial to the optimal solution so that Scheme ③—with a small crossover ratio and a large copy ratio—is the best. When the population size and the maximum generation number are relatively large, a large crossover ratio could cause the solution to be away from the optimal solution while a small crossover ratio could cause the solution to sink into local optima. Therefore, Scheme ②—with a moderate crossover ratio—is the best. Results also depend on cost matrices. In Case 1, when the population size and the maximum generation number increase to 80 and 100, respectively, Scheme ② becomes the best. By contrast, the solution space of the cost matrix in Case 2 is more complicated and needs more iterations to search. As a result, when the population size and the maximum generation number increase to 120 and 200, respectively, Scheme ② becomes the best.

Table 5. Best ratio schemes for different population sizes and maximum generation numbers in the two cases

Case 1			Case 2		
Population size	Maximum generation number	Best ratio scheme	Population size	Maximum generation number	Best ratio scheme
40	50	①	40	50	①
	100	①		100	①
	200	③		200	①
80	50	③	80	50	①
	100	②		100	①
	200	②		200	③
120	50	②	120	50	①
	100	②		100	③
	200	②		200	②

5.3 *Storm matching with ant colony algorithm*

5.3.1 ALGORITHM STEPS

The following steps are referred to in Yin et al.[24].

Step 1. Information is initialized. Each component in the cost matrix serves as a node given a two-dimensional coordinate (i, j) representing that storm i at t_1 is matched with storm j at t_2. A random permutation is created and its cost f_0 is calculated. Meanwhile, each node is attached with an equal amount of pheromone $h_0=1/f_0$. The tabu node assemble is initialized to be empty.

Step 2. If the maximum search round is achieved, the calculation is terminated; otherwise ants select node v_{ij} from the assemble of remaining nodes according to the following policy

$$(i,j)=\begin{cases}\arg\max\limits_{v\in V_r}\{h(v)^{\alpha}/c(v)^{\beta}\}, q\le q_0\\ p\limits_{v\in V_r}\{h(v)^{\alpha}/c(v)^{\beta}\}, q>q_0\end{cases}. \quad (3)$$

When an ant chooses a node, not only this node but also nodes with the same row and column coordinates are listed in the tabu node assemble to ensure a one-one correspondence for storm matching and to reduce searching times for ants. V_r represents remaining nodes excluding those in the tabu assemble. $h(v)$ is the pheromone value at node v, and $c(v)$ is the cost of node v. $\arg\max\limits_{v\in V_r}\{h(v)^{\alpha}/c(v)^{\beta}\}$ and $p\limits_{v\in V_r}\{h(v)^{\alpha}/c(v)^{\beta}\}$ are respectively the subscript of the node whose $h(v)^{\alpha}/c(v)^{\beta}$ value is the maximum and the node selected by roulette wheel in V_r, where α and β are two parameters determining the relative impact between the pheromone and node cost and they are both set to 1 here. q is a random variable equally distributed between 0 and 1. q_0 is a threshold parameter for q and it indicates the probability of an ant choosing the current best node.

Step 3. Pheromone is locally updated. When an ant chooses node v_{ij}, the pheromone at v_{ij} is updated through $h_{ij}=(1-\xi)\,h_{ij}+\xi\,h_0$, where $0<\xi<1$ and ξ is the local update coefficient. The update reflects an integrated effect of evaporation and enhancement, avoiding quick concentration to local optimal regions while in favor of stable convergence to the optimal solution.

Step 4. Pheromone is globally updated. When all ants achieve local optimal paths, a global optimal path is singled out from these paths. The pheromone is globally updated according to $h_{ij}=(1-\rho)h_{ij}+\rho\Delta h_{ij}^{op}, \forall(i,j)\in W^{op}$, where ρ is the global update coefficient and $0<\rho\leqslant 1$, $\Delta h_{ij}^{op}=1/C^{op}$ is the pheromone increment, C^{op} is the total cost of the current global optimal path, and W^{op} is the current optimal path. The pheromone update is only executed on nodes constituting W^{op} rather than all nodes. Go back to Step 2.

5.3.2 ALGORITHM EXPERIMENTS

It can be seen that when the threshold and global update coefficient are large and the local update coefficient is small, results are desirable. A small local update coefficient makes

the original pheromone influence decrease quickly and keeps more pheromone laid down by ants. A large global update coefficient gives additional pheromone to nodes on the optimal path searched in each round and reinforces the positive feedback. A large threshold makes ants more likely to select nodes with more pheromone and less cost. All of these three parameters can accelerate the convergence to achieve the optimal solution. Therefore, when the ant colony algorithm is applied to storm tracking, a large threshold parameter, a large global update coefficient and a small local update coefficient as well as adequate ants and search rounds can lead to a desirable result with a short calculation time.

Table 6. Calculation results of Case 1

Threshold parameter	Global update coefficient	Local update coefficient	Mean total cost	Total cost standard deviation	Correct solution appearing times	Average calculation time /s
		0.2	74.735	24.347	4	0.155
	0.2	0.5	86.681	28.296	2	0.153
		0.8	76.079	21.6	2	0.155
		0.2	54.819	5.77	8	0.158
0.3	0.5	0.5	64.447	12.1	4	0.158
		0.8	88.127	15.351	0	0.158
		0.2	52.082	0	10	0.158
	0.8	0.5	61.988	15.456	6	0.155
		0.8	66.206	14.888	4	0.152
		0.2	60.054	11.101	6	0.155
	0.2	0.5	61.401	13.687	6	0.156
		0.8	61.83	14.636	6	0.152
		0.2	53.451	4.328	9	0.155
0.6	0.5	0.5	61.83	15.053	6	0.153
		0.8	61.422	10.85	5	0.153
		0.2	56.634	14.393	9	0.156
	0.8	0.5	56.475	10.05	8	0.155
		0.8	68.88	18.066	5	0.156
		0.2	53.451	4.328	9	0.153
	0.2	0.5	52.082	0	10	0.156
		0.8	53.451	4.328	9	0.152
		0.2	53.451	4.328	9	0.155
0.9	0.5	0.5	53.451	4.328	9	0.155
		0.8	59.934	12.643	7	0.153
		0.2	52.082	0	10	0.155
	0.8	0.5	63.366	14.63	6	0.153
		0.8	57.556	7.067	6	0.153

The number of ants and search round are both set to 10. The threshold parameter is set to 0.3, 0.6 and 0.9, respectively. The global update coefficient is set to 0.2, 0.5, and 0.8, and the local update coefficient is set to 0.2, 0.5, and 0.8, respectively. Each scheme is tested 10 times, achieving mean total costs, total cost standard deviations, appearing times of correct solution and average calculation time.

Table 7. Calculation results of Case 2. Parameter settings and experiment items are the same as in Table 6.

Threshold parameter	Global update coefficient	Local update coefficient	Mean total cost	Total cost standard deviation	Correct solution appearing times	Average calculation time /s
		0.2	74.735	24.347	4	0.155
	0.2	0.5	86.681	28.296	2	0.153
		0.8	76.079	21.6	2	0.155
		0.2	54.819	5.77	8	0.158
0.3	0.5	0.5	64.447	12.1	4	0.158
		0.8	88.127	15.351	0	0.158
		0.2	52.082	0	10	0.158
	0.8	0.5	61.988	15.456	6	0.155
		0.8	66.206	14.888	4	0.152
		0.2	60.054	11.101	6	0.155
	0.2	0.5	61.401	13.687	6	0.156
		0.8	61.83	14.636	6	0.152
		0.2	53.451	4.328	9	0.155
0.6	0.5	0.5	61.83	15.053	6	0.153
		0.8	61.422	10.85	5	0.153
		0.2	56.634	14.393	9	0.156
	0.8	0.5	56.475	10.05	8	0.155
		0.8	68.88	18.066	5	0.156
		0.2	53.451	4.328	9	0.153
	0.2	0.5	52.082	0	10	0.156
		0.8	53.451	4.328	9	0.152
		0.2	53.451	4.328	9	0.155
0.9	0.5	0.5	53.451	4.328	9	0.155
		0.8	59.934	12.643	7	0.153
		0.2	52.082	0	10	0.155
	0.8	0.5	63.366	14.63	6	0.153
		0.8	57.556	7.067	6	0.153

6 COMPARATIVE EXPERIMENT ON MODERN OPTIMIZATION ALGORITHMS

Because the order of the cost matrix corresponding to the number of identified storms between successive time intervals may not be very large in actual situations, the three modern optimization algorithms could not show their advantages in just two storm matching cases. In order to further verify their

effect on the assignment problems and highlight the characteristics, a case of assignment problem which cannot be solved by the Hungarian method was presented here to test their capability. An improved Hungarian method still did not achieve the actual optimal solution of the assignment problem (Gu et al.[13]; Yin et al.[24]). The cost matrix of this assignment problem is B_3 (see the Appendix) and its minimum cost is 26. Experiments on this case are implemented 50 times for the simulated annealing algorithm, genetic algorithm and ant colony algorithm, respectively.

The parameters in the simulated annealing algorithm are set as: cooling coefficient α=0.8, terminating temperature t_e=0.1. Figure 3 shows the convergence process of a best global optimal solution among the experiments. It can be seen that the fluctuation of the global optimal solution is remarkable and usually tends to be worse (while the cost is increased). However, the convergence becomes gradually faster and approaches the optimal solution as temperature decreases to the terminating point. Because the temperature decrease is a convergent process according to the Boltzmann distribution, the probability with which a molecule stays in the most stable state becomes gradually larger in the cooling process. In addition, adjustable parameters are intuitional, and therefore the simulated annealing algorithm is convenient and reliable for the assignment problem.

The parameters in the genetic algorithm are set as: population size N=80, maximum generation number G=100, and ratio scheme ② (0.2, 0.75, 0.05) for copy, crossover and mutation. Figure 4 shows the convergence process of a best global optimal solution among the experiments. It can be seen that although tending to approach the optimal solution as the generation number increases, results stay around a local optimal solution with little variation and cannot continue to approach the actual optimal solution. As for the algorithm, when a local optimal solution is close to the global optimal, the successive genetic operation can possibly cause the next generation to pass the neighborhood of the global optimal solution and not to converge. Despite the fact that the elitist strategy is adopted to stabilize the algorithm, it is difficult to guarantee that the results approaches the actual optimal solution gradually; instead they fluctuate around a local optimal solution with slow convergence, with a relatively unsatisfactory result.

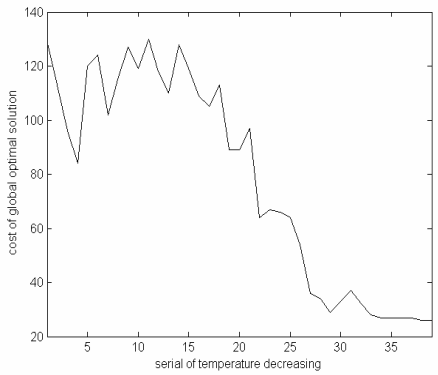

Fig. 3. Convergence process of a best global optimal solution for the experiments on the simulated annealing algorithm

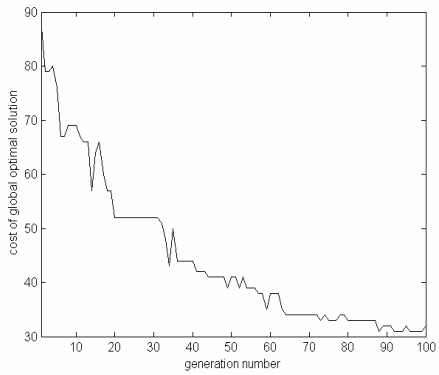

Fig. 4. Same as Fig. 4 except for the genetic algorithm

The parameters in the ant colony algorithm are set as: number of ants N=10, number of search round T=30, threshold parameter q_0=0.9, global update coefficient ρ=0.8, local update coefficient ξ=0.2. Figure 5 shows the convergence process of a best global optimal solution among the experiments. It can be seen that the result shows a gradual convergence to the optimal solution in the form of steps. Ants seek nodes which have large pheromone values and small costs during a path selecting process. The relative ratio of the two factors depends on the pheromone update. Ants seek new paths only when the accumulated pheromone reaches a certain degree, which can be regarded as a convergence from the quantitative to qualitative change. This convergence form makes results stably approach the optimal solution without the appearance of fluctuations. Therefore, the algorithm has a quick convergence and a desirable result.

Table 8 shows a comparison of the three algorithms. Similar to the experiments on storm matching, the simulated annealing algorithm and ant colony algorithm both give desirable results that global optimal solutions are close to the actual optimal solution with a short calculation time while the genetic algorithm gives a relatively poor result. The cooling process in the simulated annealing algorithm and the pheromone update in the ant colony algorithm both have clear convergence directions while genetic operations in the genetic algorithm are only optimizations to solutions already achieved. Both the simulated annealing algorithm and the ant colony algorithm transform the solution in only a few places to search solution space while the genetic algorithm transforms the solution considerably through the crossover operation. As a result, the capability of the genetic algorithm on the assignment problem is relatively poor given common coding and operators. Techniques have been developed to improve the genetic algorithm but the performance is still constrained (Han et al.[25]).

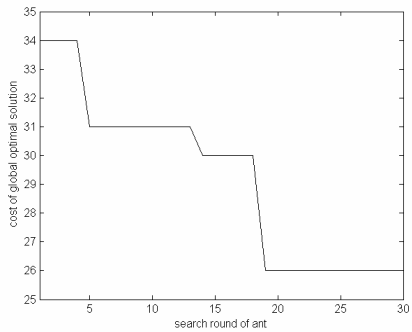

Fig. 5. Convergence process of a best global optimal solution among the experiments on the ant colony algorithm

For large-scale and complex assignment problems, there

is no need to seek optimal solutions which cannot possibly be determined. In this situation, one or several relatively desirable solutions costing short calculation time may satisfy the actual needs. The cost value and calculation time are both factors that should be considered and therefore modern optimization algorithms are suitable. The capability of parallel computation and massive search enables modern optimization algorithms to have diverse solutions and assemble optimal or near-optimal solutions. By contrast, it is difficult for the Hungarian method to attain this goal.

Table 8. Comparative results of the three algorithms on Case 3

	Mean total cost	Total cost standard deviation	Mnimum cost	Correct solution appearing times	Average calculation time /s
Simulated annealing algorithm	28.26	1.291	26	5	0.877
Genetic algorithm	36.54	3.587	32	0	1.944
Ant colony algorithm	27.82	1.305	26	9	0.645

7 CONCLUSIONS

A new, simple and feasible two-dimensional storm identification method was proposed in this article. Compared with the previous methods, the new method searches directly in two dimensions for storm segments as a whole rather than combines storm components identified in radials. Based on the storm identification results in three successive time intervals at the lowest elevation PPI using data collected by a NEXTRAD at Nanjing, three modern optimization algorithms, i.e., the simulated annealing algorithm, genetic algorithm and ant colony algorithm, were tested for storm matching. Experiment results on the two cases showed that the simulated annealing algorithm and ant colony algorithm are reliable and corresponding parameters can be set intuitionally and efficiently, yielding desirable solutions whereas it is difficult for the genetic algorithm to achieve optimal solutions and have low convergence rates due to the characteristics of genetic operations. Meanwhile, a case of the assignment problem which could not be solved by the Hungarian method was used to further test the modern optimization algorithms and highlight their characteristics. This article aims to introduce new ideas and methods, and to offer some preliminary analysis and references. Future work is to needed to study the application of these algorithms in storm tracking and other relevant fields with more effective data, and new coding methods and effective operators are to be proposed to improve the algorithms according to the characteristics of relative problems.

Appendix: Corresponding cost matrices of the three cases in this article

B_1 (Case 1)=
[4.4735, 32.028, 380.35, 27.569, 52.11,84.664, 112.81, 54.592, 159.79, 80.733, 67.103, 80.833, 108.88;
34.919, 3.2289, 383.61, 46.684, 73.87, 110.41, 138.81, 62.315, 186.8, 106.01, 90.062, 96.239, 133.18;
376.81, 379.94, 8.6914, 396.83, 400.97, 400.81, 408.87, 418.36, 391.7, 422.67, 432.1, 433.53, 434.16;
27.444, 46.038, 399.21, 2.5642, 28.737, 67.83, 96.213, 30.113, 147.88, 60.447, 45.533, 53.548, 87.678;
77.748, 107.39, 402.61, 63.729, 50.313, 4.799, 32.815, 80.518, 86.358, 21.992, 47.507, 59.695, 33.779;
106.7, 133.23, 405.25, 91.326, 75.106, 25.526, 5.7685, 104.4, 58.972, 43.341, 72.823, 78.801, 31.144;
151.34, 179.31, 386.52, 141.3, 119.66, 77.49, 53.123, 150.59, 6.0694, 98.685, 128, 126.9, 81.911;
54.395, 58.909, 415.16, 30.352, 32.977, 82.133, 109.99, 2.7326, 156.05, 69.887, 47.348, 38.299, 95.999;
76.982, 102.62, 419.29, 56.82, 42.104, 24.925, 46.203, 66.828, 102.54, 3.6756, 29.219, 39.806, 30.819;
69.654, 91.545, 435.58, 47.877, 38.981, 57.282, 80.344, 47.216, 136.71, 34.54, 5.0482, 18.733, 59.929;
102.43, 128.89, 432.67, 83.073, 68.374, 33.41, 34.089, 91.157, 88.498, 24.258, 50.593, 58.751, 5.0312];

B_2 (Case 2)=
[4.3194, 32.888, 376.39, 30.608, 84.255, 114.52, 162.35, 106.43, 110;
35.445, 1.9358, 380.49, 49.885, 110.95, 141.22, 190.32, 130.67, 138.35;
373.42, 379.56, 5.5868, 394.01, 398.82, 406.07, 390.97, 420.65, 434.21;
27.822, 45.08, 396.07, 6.147, 69.079, 99.496, 152.09, 86.68, 92.645;
49.92, 70.629, 397.8, 21.696, 54.829, 83.012, 131.1, 65.562, 77.901;
79.411, 107.19, 398.11, 63.412, 3.4745, 33.646, 87.83, 29.145, 37.993;
107.23, 136.5, 405.37, 92.819, 26.812, 5.2737, 62.373, 26.897, 29.934;
56.843, 60.86, 415.95, 30.913, 85.288, 112.24, 161.81, 90.954, 101.47;
152.94, 182.35, 386.81, 139.57, 78.152, 51.532, 6.4062, 75.452, 81.494;
101.46, 126.13, 418.53, 77.453, 26.509, 29.558, 83.744, 3.5635, 16.101;
80.372, 94.097, 430.2, 49.653, 64.213, 86.041, 137.2, 60.652, 69.117;
105.03, 132.48, 431.14, 86.481, 35.245, 35.221, 89.663, 12.475, 5.3751];

B_3 (Case 3)=
[4 2 6 2 9 9 8 5 6 8 6 9 7 1 9 9 9 5 2 8 4 2;5 9 6 3 8 9 6 4 6 6 6 6 1 2 2 9 6 4 1 5 3 2;
1 6 2 8 4 3 1 2 9 2 7 4 6 9 3 6 9 4 8 3 4 9;6 4 9 5 4 4 4 1 2 1 3 3 5 6 5 5 1 8 4 2 8 6;
7 7 2 6 1 3 1 4 8 7 2 3 1 5 9 8 9 5 1 7 3 6;9 6 2 8 5 4 8 4 8 1 7 5 4 2 1 9 9 4 8 4 5 2;
3 1 9 8 7 3 9 5 3 6 7 1 2 5 7 1 9 4 2 4 5 3;2 2 5 8 9 8 5 4 7 8 9 6 7 9 1 9 7 6 1 5 4 1;
5 3 5 2 9 4 8 9 8 4 3 4 4 1 8 5 1 4 1 4 2 5;8 8 2 6 1 2 3 2 9 8 3 3 9 4 7 7 1 6 7 7 3 5;
4 1 6 2 8 1 7 5 7 6 3 3 5 1 7 1 7 6 4 4 2 6;5 4 6 4 1 7 3 9 1 8 2 9 9 5 9 4 5 2 3 9 1 5;
8 7 9 7 9 5 5 1 5 9 2 4 1 9 6 6 9 3 3 9 2 2;2 9 7 7 8 3 2 3 3 5 8 9 9 5 6 8 9 2 8 4 7 8;
7 2 2 3 5 4 6 6 2 1 7 6 8 8 2 2 2 8 9 3 8 7;8 5 2 5 6 4 2 1 1 2 2 3 4 6 3 6 4 8 6 2 7 7;
7 5 4 5 1 7 4 2 1 9 2 7 1 6 6 3 7 9 2 8 1 4;7 2 2 5 6 6 7 9 6 4 5 6 4 3 7 5 3 4 2 6 2 7;
4 4 6 2 6 2 1 9 8 4 7 8 8 4 9 5 5 3 4 2 4 6;6 4 6 1 3 7 8 5 2 4 7 6 6 9 3 1 8 8 6 1 5 1;

5 9 9 9 2 1 2 6 2 8 4 1 2 2 2 2 3 9 1 8 9 8;4 2 8 9 7 9 7 5 8 7 1 1 7 9 7 1 4 7 5 6 1 5].

REFERENCES:

[1] RINEHART R E, GARVEY E T. Three-dimensional storm motion detection by conventional weather radar [J]. Nature, 1978, 273: 287-289.

[2] LI L, SCHMID W, JOSS J. Nowcasting of motion and growth of precipitation with radar over a complex orography [J]. J. Appl. Meteor., 1995, 34(6): 1 286-1 300.

[3] CRANE R K. Automatic cell detection and tracking. [J]. IEEE Transact. on Geosci. Electron., 1979, 17(4): 250-262.

[4] ROSENFELD D. Objective method for analysis and tracking of convective cells as seen by radar [J]. J. Atmos. Ocean. Technol., 1987, 4(3): 422-434.

[5] HANDWERKER J. Cell tracking with TRACE3D—A new algorithm [J]. Atmos. Res., 2002, 61(1): 15-34.

[6] DIXON M J, WIENER G. TITAN: Thunderstorm identification, tracking, analysis and nowcasting-a radar based methodology [J]. J. Atmos. Ocean. Technol., 1993, 10(6): 785-797.

[7] JOHNSON J T, MACKEEN P L, WITT A, et al. The Storm cell identification and tracking algorithm: An enhanced WSR-88D algorithm [J]. Weather and Forecasting, 1998, 13(2): 263-276.

[8] XIAO Yan-jiao, TANG Da-zhang, LI Zhong-hua, et al. Storm automatic identification, tracking and forcasting [J]. J. Nanjing Inst. Meteor., 1998, 21(2): 223-229.

[9] LI Wen-hui. Application of Hungary algorithm in assignment problem of train crew [J]. J. Lanzhou Commun. Univ. (Nat. Sci.), 2007, 26(3): 55-57.

[10] LAWLER E L. Combinatorial Optimization: Networks and Matroids [M]. Oxford: Oxford University Press, 1995: 384.

[11] ROBERTS F S. Applied Combinatorics [M]. Prentice Hall, 1984: 565-568.

[12] CHU Yan-zheng. Improvement of Hungarian method in solving designation problems [J]. J. Chongqing Inst. Technol. Manage., 1998, 12(4): 76-77.

[13] GU Da-quan, ZUO Li, HOU Tai-ping, et al. Existing problem and improvement of “Hungary Arithmetic” [J]. Microcom. Develop., 2003, 13(4): 76-78.

[14] XING Wen-xun, XIE Jin-xing. Modern Optimization Algorithms [M], Beijing: Tsinghua University Press, 2005: 247.

[15] DUAN Cha-li, CHEN Bo. Simulated annealing algorithm to solve assignment problem under VB [J]. Comput. Knowl. Technol., 2008, 4(8): 2 153-2 155.

[16] WU Yan-qun, DONG Peng. A general simulated annealing algorithm for solving large scale asymmetrical assignment problem [J]. J. Lanzhou Commun. Univ., 2008, 27(4): 149-155.

[17] LI Yan, CHEN Zu-an, XU Yue-fei, et al. Research on genetic algorithm for assignment problem and its realization [J]. J. Xi’an Univ. Technol., 1996, 12(4): 271-276.

[18] ZHANG Quan, XIU Hong-wei. Application of the genetic algorithm in the assignment problem [J]. J. Shenyang Architect. Civil Eng. Inst., 1997, 13(1): 25-28.

[19] WU Hai-jun. Solving assignment problem based on genetic algorithm [J]. Comput. Study, 2005, 6: 23-24.

[20] RUDOLPH G. Convergence Properties of Canonical Genetic Algorithms [J]. IEEE Trans. on Neural Network, 1994, 5(1): 96-101.

[21] DE JONG K A. An Analysis of the Behavior of a Class of Genetic Adaptive Systems [D]. Theses for Doctoral Degree, Ann Arbor: University of Michigan, 1975: 1-266.

[22] EIBEN A E, AARTS E H, VAN HEE K M. Global convergence of genetic algorithms: A Markov chain analysis [C]//Parallel Problem Solving from Nature. Berlin: Springer, 1991: 3-12.

[23] WANG Lei, SHEN Ting-zhi, ZHAO Yang. An improved adaptive genetic algorithm [J]. Syst. Eng. Electron., 24(5): 75-78.

[24] YIN Ren-kun, WU Yang, ZHANG Jing-wei. Research and application of the ant colony algorithm in the assignment problem [J]. Comput. Eng. Sci., 2008, 30(4): 43-112.

[25] HAN Jian-feng, LI Min-qiang, KOU Ji-song. Encoding schemes of array problem in genetic algorithms [J]. Comput. Eng. Sci., 2002, 12: 29-32.

Citation: LI Nan and WEI Ming. Experiments to track storms using modern optimization algorithms. *J. Trop. Meteor.*, 2010, 16(3): 280-291.