Research of Ant Colony Algorithm with Elite Strategy in Process Route

FengYun Huang, ShiQiu Jiang
School of Mechanical and Electrical Engineering, Wuhan University of Technology, Wuhan, China
Email: 2747222643@qq.com

Abstract. The basic ant colony algorithm is slow in convergence and easy to fall into the local optimal solution. In this paper, it is proposed to apply the ant colony algorithm with elite strategy to the sequencing of the processing technology routes. With the least number of process equipment transformations and the shortest path of the tool, automatic ordering of the parts processing is realized. Simulation experiments show that the ant colony algorithm with elite strategy converges to the optimal solution after 39 iterations, while the basic ant colony algorithm converges after 71 iterations. Comparing the time-consuming of two algorithms, the ant colony algorithm with elite strategy improved by 42.7%. It illustrates that the ant colony algorithm with elite strategy has better application in sequencing process routes, which is beneficial for shortening the process planning time and increasing the process design efficiency.

1 Introduction
For many manufacturing companies, high-efficiency, low-cost, and high-quality production methods will have a significant impact on the survival, competition and development of enterprises. Therefore, efficiently formulating and generating optimal, low-cost and short-time processing routes are the focal points of scholars' research. Many scholars have been studying the decision-making and optimization of parts process routes to obtain low-cost, high-efficiency parts process routes that meet the actual production requirements. Zheng Yongqian[1] and others used genetic algorithms to select and sort the processing schemes, and verified that this method can improve the optimization ability of the processing technology; Pan Yuling[2] and others used the improved genetic algorithm to avoid the "premature" disadvantage of a single genetic algorithm; LI Xinyu[3] provided a blueprint for efficient manufacturing system. This paper proposes a novel algorithm hybridizing the genetic algorithm with strong global searching ability and variable neighborhood search with strong local searching ability for the IPPS problem; XY Li[4] proposed a flexible process optimization method based on genetic programming, and compared it to genetic algorithms which proved that the algorithm is adaptable and superior; Lan Xuan PHUNG[5] proposed a clustering algorithm applied to the CAPP process planning to generate the optimal operation sequence, which can effectively shorten the operation time; G. Nallakumarsamy[6] used simulated annealing technology to generate a feasible sequence of operations on the basis of the cost priority matrix and the reward and penalty matrix, and proved the feasibility and robustness of the algorithm; F. Zhang[7] proposed a new computer-aided process planning model. This method considers multiple decision-making activities and illustrates its performance through examples. Milica Petrovića[8] provided a new algorithm for optimization of flexible process plans based on utilization of particle swarm optimization algorithm and chaos theory, experimental results show that the developed method outperforms genetic algorithm, simulated annealing, hybrid GA-SA and generic PSO based approach; Su Yuliang[9] proposed an edge selection strategy based GA to solve the premature convergence problem
may occur facing some complicated PCOSPs; Zhaohui Deng\[10\] proposed a multi-objective machining process route optimization model based on the genetic algorithms (GA), and the minimum processing time and the optimal carbon efficiency were set as the optimization objectives. Yujia Wu\[11\] formulated the priority matrix according to the sorting rules of the working steps, which makes the processing process more standardized and convenient. However, from the existing literature, many researches mainly optimizing the cost of machine conversion, tool conversion and fixture conversion when optimizing the process route, and consider the tool path less.

2 Introduction to related algorithms

2.1 Basic ant colony algorithm

Ant colony algorithm is one of many bionic algorithms. It simulates the behavior of ants foraging. In the process of foraging, ants will release pheromones on their paths. Ants have high probability to choose the path with high pheromone concentration and release a certain amount of pheromones to form a positive feedback mechanism, which makes the search results converge to approximate the optimal solution\[12\].

The most important thing in the ant colony algorithm is to determine the transfer rule of ants transferring from one point to another point, and the update rule of pheromone on the path. The state transition rule of ant \(k\) at time \(t\) in the basic ant colony algorithm is:

\[
P_{ij}^k(t) = \frac{\tau_{ij}^k(t)\eta_{ij}^k(t)}{\sum_{u \in N^i_k} \tau_{ui}^k(t)\eta_{ui}^k(t)} \quad j \in N^i_k
\]

In the formula, \(P_{ij}^k\): the probability of ant \(k\) transferring from point \(i\) to point \(j\); \(\tau_{ij}\): pheromone intensity on path \(l_{ij}\); \(\eta_{ij}\): heuristic information on path \(l_{ij}\); \(\alpha\): heuristic factor; \(\beta\): expected heuristic factor; \(N^i_k\): The point that Ant \(k\) is allowed to visit next, \(N^i_k = \{0,1,...,u,...,n - 1\}\), \(n\) is the number of points. The heuristic information \(\eta_{ij}\) generally takes the reciprocal of the distance between points, \(\eta_{ij} = 1/d_{ij}\), the greater the distance, the smaller the probability of being selected.

After completing a cycle, the ants need to adjust the pheromone on the path. The pheromone update principle of the general ant colony algorithm is:

\[
\tau_{ij}(t + 1) = (1 - \rho)\tau_{ij}(t) + \Delta\tau_{ij}(t, t + 1)
\]

\[
\Delta\tau_{ij}(t, t + 1) = \sum_{k=1}^{m} \Delta\tau_{ij}^k(t, t + 1)
\]

In the formula, \(\Delta\tau_{ij}^k\): the amount of pheromone released by ant \(k\) on path \(l_{ij}\); \(\rho\): pheromone volatilization factor; \(\Delta\tau_{ij}(t, t + 1)\) represents the amount of pheromone left on the path \(l_{ij}\) of ant \(k\) at time \((t, t + 1)\); \(\Delta\tau_{ij}(t, t + 1)\) is the pheromone increment on the path \(l_{ij}\) in this cycle.

2.2 Ant colony algorithm with elite strategy

Although the ant colony algorithm has a strong capability of global searching, it has a long search time and is prone to stagnation. Therefore, many improved algorithms have been proposed, and the ant colony algorithm with elite strategy is one of them. The core of the ant colony algorithm with elite strategy is to give the optimal solution obtained so far after each cycle with an extra amount of pheromone, so that in the next cycle the ants will be more concentrated near the optimal solution. In the ant colony algorithm with elite strategy, the updating principle of pheromone is different from the basic ant colony algorithm, as follows:

\[
\tau_{ij}(t + 1) = \rho\tau_{ij}(t) + \Delta\tau_{ij} + \Delta\tau_{ij}^e
\]

\[
\Delta\tau_{ij} = \sum_{k=1}^{m} \Delta\tau_{ij}^k
\]
\[ \Delta \tau_{ij}^k = \begin{cases} \frac{Q}{L_k} & \text{The kth ant passes the path J again in this cycle} \\ 0 & \text{others} \end{cases} \] (6)

\[ \Delta \tau_{ij}^* = \begin{cases} \frac{egQ}{L^*} & \text{Path l_{ij} is part of the optimal solution found} \\ 0 & \text{others} \end{cases} \] (7)

In the formula, \( \Delta \tau_{ij}^* \) represents the pheromone increment caused by elite ants on the path after walking through the path \( l_{ij} \); \( e \) represents the number of elite ants; \( L^* \) represents the path length of the optimal solution found. The ant colony algorithm with elite strategy enables ants to find a better solution earlier in the algorithm process, and increase the efficiency of the algorithm.

3 Implementation of ant colony algorithm with elite strategy in process routing

Step1: Initialize the ant colony algorithm: initialize the number of iterations, and set the iteration counter \( NC = 0 \), the maximum number of iterations \( NC_{\text{max}} \); initialize the pheromone on each path, set \( \tau_{ij}(0) = C \) (\( C \) is a small constant), \( \Delta \tau_{ij} = 0 \); set the taboo table of each ant to \( Tabu_k (k = 1, 2, \ldots, m) \); initialize the priority matrix of the processing primitives;

Step2: Detect the priority matrix, determine the initial processing primitives according to the constraints of the priority matrix, place all ants randomly on these processing primitives, and update the ant's taboo table;

Step3: Ant \( k \) starts to find the optimal path;

Step4: According to the taboo table and the priority processing matrix, the ants determine the processing primitives that can be transferred. Then calculate the probability of the ants transferring to these processing primitives according to the state transition rules, and select the processing primitive \( u_j \) with the highest probability to transfer. Add \( u_j \) to the taboo table of ants \( Tabu_k \); set all the elements of column \( j \) corresponding to \( u_j \) in the priority matrix to 0, and update the priority matrix;

Step5: \( j = j + 1; \) if \( j < N \), go to Step4;

Step6: \( k = k + 1; \) if \( k < m \), go to Step3;

Step7: Calculate the path length after each ant traverses the processing primitives, and compare with the current optimal path, if it is less than the optimal path, update the current optimal path;

Step8: According to the pheromone update rule with elite strategy ant colony algorithm, update the pheromone on the path;

Step9: \( NC = NC + 1; \) if \( NC < NC_{\text{max}} \), clear the taboo table \( Tabu_k \) of all ants, initialize the priority matrix of the processing primitives, go to Step2 otherwise, output the optimal path.

4 Results and analysis of two algorithms

In order to compare the results of the two algorithms, we now take the driving gear part of the automobile final reducer as an example. The two-dimensional diagram of this part is as shows in Figure 1.
According to the characteristic surface information, based on the constraints of the processing primitives, the information table of the processing primitives is obtained as shown in Table 1:

**Table 1. Processing primitive information table.**

| $u_i$ | Number | $u_i.S_i$ | $u_i.M_i$ | $u_i.T_i$ | $u_i.C_i$ | $u_i.f_i$ | $u_i.r_i$ |
|------|--------|----------|-----------|-----------|-----------|-----------|-----------|
| $u_1$ | $F_{T3}$ | 1 | 1 | 1 | 1 | $F_{T1}$, $F_{T2}$, $F_{T13}$ (16.5,0,0) |
| $u_2$ | $F_{T3}$ | 2 | 2 | 2 | 2 | $F_{T5}$, $F_{T6}$, $F_{T9}$ (16.5,0,0) |
| $u_3$ | $F_{T3}$ | 3 | 2 | 3 | 2 | $F_{T5}$, $F_{T6}$, $F_{T9}$ (16.5,0,0) |
| $u_4$ | $F_{T3}$ | 4 | 3 | 4 | 3 | $F_{T6}$, $F_{T9}$ (16.5,0,0) |
| $u_5$ | $F_{T4}$ | 1 | 1 | 1 | 1 | $F_{T1}$, $F_{T2}$, $F_{T13}$ (34.5,0,0) |
| $u_6$ | $F_{T5}$ | 5 | 1 | 1 | 1 | $F_{T1}$, $F_{T2}$, $F_{T13}$ (36.0,0,0) |
| $u_7$ | $F_{T5}$ | 6 | 1 | 5 | 1 | $F_{T1}$, $F_{T2}$, $F_{T13}$ (49.0,0,0) |
| $u_8$ | $F_{T6}$ | 7 | 1 | 1 | 1 | $F_{T1}$, $F_{T3}$, $F_{T13}$ (49.0,0,0) |
| $u_9$ | $F_{T7}$ | 6 | 1 | 1 | 1 | $F_{T1}$, $F_{T2}$, $F_{T13}$ (67.5,0,0) |
| $u_{10}$ | $F_{T8}$ | 6 | 1 | 1 | 1 | $F_{T1}$, $F_{T2}$, $F_{T13}$ (75.5,0,0) |
| $u_{11}$ | $F_{T9}$ | 6 | 1 | 5 | 1 | $F_{T1}$, $F_{T2}$, $F_{T13}$ (85.5,0,0) |
| $u_{12}$ | $F_{T9}$ | 7 | 1 | 1 | 1 | $F_{T1}$, $F_{T3}$, $F_{T13}$ (85.5,0,0) |
| $u_{13}$ | $F_{T10}$ | 6 | 1 | 1 | 1 | $F_{T1}$, $F_{T2}$, $F_{T13}$ (105.0,0,0) |
| $u_{14}$ | $F_{T10}$ | 8 | 4 | 6 | 1 | $F_{T1}$, $F_{T13}$ (105.0,0,0) |
| $u_{15}$ | $F_{T11}$ | 6 | 1 | 1 | 1 | $F_{T1}$, $F_{T2}$, $F_{T13}$ (180.0,0,0) |
| $u_{16}$ | $F_{T11}$ | 9 | 4 | 7 | 1 | $F_{T1}$, $F_{T13}$ (180.0,0,0) |
| $u_{17}$ | $F_{T13}$ | 10 | 5 | 8 | 4 | $F_{T2}$, $F_{T6}$, $F_{T9}$ (180.0,8) |

In the above table, $u_i.S_i$: the machining method used for machining primitives, 1 means fine turning cone, 2 means rough milling gear, 3 means fine milling gear, 4 means grinding gear, 5 means fine turning plane, 6 means fine turning the outer cylindrical surface, 7 means fine turning after heating, 8 means splining, 9 means rolling thread, 10 means milling bevel; $u_i.M_i$: the machine tool used for machining primitives, 1 means external cylindrical lathe, 2 means Gleason gear milling machine, 3 means special grinding machine, 4 means horizontal gear rubbing machine, 5 means horizontal milling machine; $u_i.T_i$: the tool used for machining primitives, 1 means fine turning outer cylindrical turning tool, 2 means rough milling gear tool, 3 means fine milling gear tool, 4 means grinding sand, 5 means fine cylindrical turning tool for bear, 6 means tool for rolling threads, 7 means tool for rolling splines, 8 means disc milling tool; $u_i.C_i$: the fixture used for machining primitives, 1 means the special center chuck, 2 means the fixture for special milling gear, 3 means the fixture for special grinding gear, 4 means the fixture for special vertical milling; $u_i.f_i$: the clamping surface in machining primitive machining; $u_i.r_i$: the center coordinates of the surface feature to which the processing primitive belongs.

In the processing, if the machine tools, tools, fixtures, etc. of the two processing elements matches, they need to be arranged processing together to reduce the change of process equipment. Therefore, clustering constraint relationships between the processing elements need to be taken into sufficient consideration when defining "distance". At the same time, it is also necessary to consider the path of the tool. When there are multiple processing primitives with the same process equipment, the spatial distance between the processing primitives needs to be considered. The processing primitive with a shorter spatial distance is preferentially processed first to reduce the moving distance of the tool, instead of randomly selecting these machining primitives. For example, when processing shaft parts, the tool will process the surface of the part in the same direction in one process instead of reciprocating. The distance between processing primitives can be defined as:
The values of the weighting coefficient $\omega$ in the formula are: $\omega_s$: 0.3, machine tool; $\omega_M$: 0.3, tool; $\omega_T$: 0.15, fixture; $\omega_C$: 0.15, clamping surface; $\omega_f$: 0.1, $\text{sim}(a, b)$ is a similarity judgment function, $d(u_i, r_i, u_j, r_j)$ is the distance function between the processing primitives, $\max[\text{sim}(u_i - C_i, u_j - C_j), \text{sim}(u_i - f_i, u_j - f_j)]$ means to consider the transformation of both the fixture or the clamping surface of the processing primitive, that is, take the maximum value.

\[
\begin{aligned}
\text{sim}(a, b) &= \begin{cases} 
1 & a \neq b \\
0 & a = b 
\end{cases} \\
d(u_i, r_i, u_j, r_j) &= 0.001 \sqrt{(x_i - x_j)^2 + (y_i - y_j)^2 + (z_i - z_j)^2}
\end{aligned}
\]

In this paper, the goal is to minimize the number of process equipment changes and the shortest tool path, and realize the automatic sequencing of parts processing procedures. According to the flow of ant colony algorithm and algorithm implementation, based on Matlab programming, the basic ant colony algorithm and the ant colony algorithm with elite strategy are used to realize the process sequencing of processing primitives. According to the research results of Xu Hongmei\cite{13}, the value of the number of ants $m$ should be $[0.6n, 0.9n]$, $n$ is the number of cities in the TSP, so this paper takes $m = 15$ and the number of elite ants $e = 4$. When the heuristic factor $\alpha \in [1.0, 2.0]$ and the expected heuristic factor $\beta \in [3.0, 6.0]$, the performance of the ant colony algorithm is better. In this paper, $\alpha = 2$ and $\beta = 3$. The larger the value of the pheromone constant $Q$, the worse the global search ability of the algorithm, and the computational power is very unstable. In this paper, $Q = 1$. The pheromone volatilization factor $\rho$ is too small, which is not conducive to the accumulation of pheromone. It is value too large, and it is easy to fall into the local optimal solution. Generally, $\rho$ is $[0.3, 0.9]$, and $\rho = 0.4$ in this paper. NC_max = 100.

The processing primitive results obtained by using two algorithms are shown in figure 2 and figure 3. From the figure, it can be seen that the results obtained by the two algorithms are consistent, and the results meet the actual processing results. The results are as follows:

\[
\begin{align*}
&u_1 \rightarrow u_5 \rightarrow u_6 \rightarrow u_9 \rightarrow u_{10} \rightarrow u_{13} \rightarrow u_{15} \rightarrow u_7 \rightarrow u_{11} \rightarrow u_{14} \rightarrow u_{16} \rightarrow u_{17} \rightarrow u_2 \rightarrow u_3 \rightarrow u_0 \rightarrow u_{12} \rightarrow u_4 \\
\text{ACO Shortest Route} &= \\
1 & 6 & 8 & 6 & 9 & 10 & 13 & 16 & 7 & 11 & 14 & 16 & 17 & 2 & 3 & 8 & 12 & 4
\end{align*}
\]

\textbf{Figure 2.} Sorting results of basic ant colony algorithm.

\[
\begin{align*}
&u_1 \rightarrow u_5 \rightarrow u_6 \rightarrow u_9 \rightarrow u_{10} \rightarrow u_{13} \rightarrow u_{15} \rightarrow u_7 \rightarrow u_{11} \rightarrow u_{14} \rightarrow u_{16} \rightarrow u_{17} \rightarrow u_2 \rightarrow u_3 \rightarrow u_0 \rightarrow u_{12} \rightarrow u_4 \\
\text{A Selite Shortest Route} &= \\
1 & 6 & 8 & 6 & 9 & 10 & 13 & 16 & 7 & 11 & 14 & 16 & 17 & 2 & 3 & 8 & 12 & 4
\end{align*}
\]

\textbf{Figure 3.} Sorting results of ant colony algorithm with elite strategy

The iteration curves of the two algorithms are shown in figure 4 and figure 5:
It can be seen from the figure 4 and figure 5, during the seventh iteration of the basic ant colony algorithm, some ants found the optimal solution with the shortest distance of 7.412; after the 71st iteration, the ants concentrated on the optimal solution and the algorithm converged. In the ant colony algorithm with elite strategy, during the sixth iteration, some ants found the optimal solution with the shortest distance of 7.412; after the 39th iteration, the ants concentrated on the optimal solution and the algorithm converged. Comparing the time consumption of the two algorithms, the basic ant colony algorithm takes 25.825s, and the elite ant colony algorithm takes 14.793s.

5 Conclusion
Using basic ant colony algorithm, the convergence speed is slow, and the ability to find the optimal solution is weak. Compared with the basic ant colony algorithm, the ant colony algorithm with elite strategy algorithm performs better in convergence speed and the ability to find the optimal solution. In addition, the ant colony algorithm with elite strategy takes less time to make processing element sequencing decisions, and is more suitable for solving processing element sequencing problems, which is beneficial for shortening process planning time and increasing process design efficiency.
6 References

[1] Yongqian Z. Optimization of Process Selection and Sequencing Based on Genetic Algorithm[J]. China Mechanical Engineering, 23: 59-65, 2012.

[2] Yu-ling P. Optimization of Process Route for Complicated Box Parts Based on Hybrid Genetic Algorithm[J]. Modular Machine Tool & Automatic Manufacturing Technique, 2(2): 103-107, 2019.

[3] Li X., Gao L., Pan Q., et al. An Effective Hybrid Genetic Algorithm and Variable Neighborhood Search for Integrated Process Planning and Scheduling in a Packaging Machine Workshop[J]. IEEE Transactions on Systems, Man, and Cybernetics: Systems, 49(10): 1933-1945, 2019.

[4] Li X.Y., Shao X.Y., Gao L. Optimization of flexible process planning by genetic programming[J]. The International Journal of Advanced Manufacturing Technology, 38(1): 143-153, 2008.

[5] Phung L.X., Tran D.V., Hoang S.V., et al. Effective method of operation sequence optimization in CAPP based on modified clustering algorithm[J]. Journal of Advanced Mechanical Design, Systems, and Manufacturing, 11(1), 2017.

[6] Nallakumarsamy G., Srinivasan P.S.S., Venkatesh Raja K., et al. Optimization of operation sequencing in CAPP using simulated annealing technique (SAT)[J]. The International Journal of Advanced Manufacturing Technology, 54(5): 721-728, 2011.

[7] Zhang F., Zhang Y.F., Nee A.Y.C. Using genetic algorithms in process planning for job shop machining[J]. IEEE Transactions on Evolutionary Computation, 1(4): 278-289, 1997.

[8] Petrović M., Mitić M., Vuković N., et al. Chaotic particle swarm optimization algorithm for flexible process planning[J]. The International Journal of Advanced Manufacturing Technology, 85(9): 2535-2555, 2016.

[9] Su Y., Chu X., Chen D., et al. A genetic algorithm for operation sequencing in CAPP using edge selection based encoding strategy[J]. Journal of Intelligent Manufacturing, 29(2): 313-332, 2018.

[10] Deng Z., Lv L., Huang W., et al. A High Efficiency and Low Carbon Oriented Machining Process Route Optimization Model and Its Application[J]. International Journal of Precision Engineering and Manufacturing-Green Technology, 6(1): 23-41, 2019.

[11] Wu Y.-j., Cao Y., Wang Q.-f. Assembly sequence planning method based on particle swarm algorithm[J]. Cluster Computing, 22(1): 835-846, 2019.

[12] Songtao W. Improved genetic algorithm based on optimized genetic algorithm ant colony algorithm AGV path planning[J]. Automation Application, 3(3): 47-49, 2017.

[13] Hong-me X. The research on the parameters of the ant colony algorithm[J]. Journal of ShanDong University of Technology(Natural of Science Edition), 22(1): 7-11, 2008.