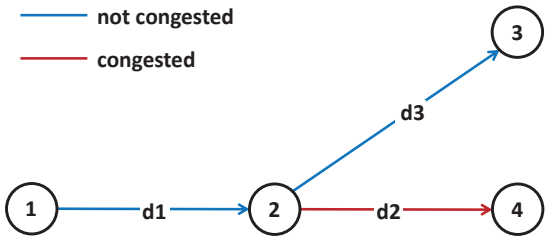
APPENDIX A
A FEASIBLE SOLUTION FOR REFERENCE

In subsection V-A of the submitted paper, we have proposed Algorithm 1 for constructing the state table. In this part, we will discuss how to use the metrics and the state table provided to accomplish the ACO part of our state table based dynamic ACO-A* algorithm. In section III of the submitted paper, we have shown that the route planning problem we investigated is a typical NP-Complete problem. Fortunately, ant colony optimization (ACO) is an efficient intelligent algorithm to solve NP-hard problems. Intuitively, we can adopt ACO to obtain a set of feasible solutions (denoted by $\{L_r\}$) for searching the paths from the source vertex V_s to the destination vertex V_d, where V_s and V_d belong to $G(V, E, F, S)$. The best solution $L_r^* \in \{L_r\}$ can be considered as a local optimal and feasible solution.

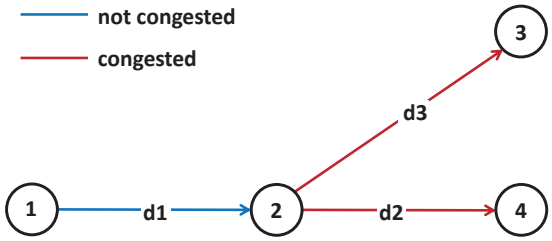
The advantage of ACO is the ability to find feasible solutions and preserve high diversity. However, the disadvantage of ACO is that it will be inclined to involve into a local best optimized solution. Thus, L_r^* produced by ACO can only be used as a reference value. In the next subsection, we will use another solution to produce the global optimal value. To solve the problem more efficiently, we will use the value of L_r^* to reduce the searching space and accelerate the solving process.

A. Background of Ant Colony Optimization

As for our problem, using ACO to obtain the value of L_r^* is not a trivial task. That is because the state table of $G(V, E, F, S)$ is dynamically changing over time. If we use the original ACO to solve the problem, we have to face the following dilemmas:



(a) Congested situation



(b) Dead alley

Fig. 1: The congested and dead alley situations.

- *Cannot avoid congested roads.*
  In Fig.1(a), we assume that the length of the d_1, d_2, d_3 as 1, 1, 2 and when people arrive at v_2, the road state of edge e_{24} is congested. However, traditional routing algorithms cannot avoid this congested road since e_{24} possesses a smaller length. Thus we will obtain the wrong route finally.
- *Meet the "dead alley".*
  In Fig.1(b), we assume that when people arrive at v_2, e_{24} and e_{23} are both congested, and there is no way to choose. Hence, roads like e_{12} should be avoided.

In order to manage the dilemmas mentioned, we optimize several basic concepts to implement the proposed *intelligent ant colony optimization*.

1) *Argument Initialization.* Before routing, we equally set the pheromone level on each road as τ_0 and put m ants on the start vertex.

2) *Choose The Next Road.* Based on preliminary studies [1], the ant at vertex v_i selects road e_{ij} following:

$$p_{ij}^k = \begin{cases} \frac{(\tau_{ij})^\alpha (\eta_{ij})^\beta}{\sum_{x \subseteq N_i} (\tau_{ix})^\alpha (\eta_{ix})^\beta}, & j \in N_i \\ 0, & j \notin N_i \end{cases} \tag{1}$$

where p_{ij}^k is the probability of ant k choosing road e_{ij}, τ_{ij} is the pheromone after the ant cross e_{ij} and α, β are coefficients representing the effects of pheromone and distance, respectively. And with the objective to find the shortest path, we define $\eta_{ij} = 1/(e_{ij}.dist)$.
Importantly, we set N_i as vertices set connected to the vertex v_i except for those whose $s_{ij}(t) = 0$ to settle the problem in Fig.1(a). Moreover, if ant k is confronted with the Fig.1(b) situation, it will stand on the current road and warn others to address this problem.

3) *Pheromone Update.* The update includes both increases and decreases. For decreases, after selecting one road, the pheromone will evaporate with a percentage $\varphi (0 < \varphi < 1)$ and thus reduce the p_{ij}^k:

$$\tau_{ij} \leftarrow (1 - \varphi)\tau_{ij} \tag{2}$$

Then for increases, if one ant(assume ant F) arrives the destination, we will update the whole path pheromone by:

$$\tau_{ij} \leftarrow \tau_{ij} + \Delta\tau_{ij}^F \qquad \Delta\tau_{ij}^F = Q/L^F \tag{3}$$

where Q is a constant representing the reward pheromone for reaching the destination, and L^F is the path length of ant F. With this method, the result will converge faster. After the circulations finish, we will set the shortest path as our approximate reference value.

B. State Table Based Ant Colony Optimization Algorithm

According to our optimization of the ACO, we can design our state table based ant colony optimization algorithm. To solve the navigation problem, we are supposed to ensure this problem is always solvable. Only in this case is the algorithm guaranteed to be workable. To demonstrate it, we have Lemma 1.

Lemma 1. *No matter how s∈ S change, there is at least one path from V_s to V_d .*

Proof. In contrast to $G(V, E, F)$, $G(V, E, F, S)$ containing the road state change information S for all $e \in E$. First, based on the social property of normal reality maps, according to Eq.(4)(in the submitted paper), only when all w_{ij} of the route are infinite at any given moment does it lead to no path, which means the states below the QoS all the time. That violates the social attributes. Furthermore, even in the most extreme case where some roads are always congested, there still exist paths from source V_s to destination V_d in $G(V, E, F)$(according to the first lemma of the submitted paper). Thus, in this unusual case, the path from V_s to V_d can still be acquired even though not all routes can satisfy the QoS requirements. □

Depending on Lemma 1, we can make certain that there is always a solution to this problem. Hence, the main part of the Intelligent ACO algorithm is proposed as the following algorithm, where $phero[j]$ means the pheromone on the road e_{ij}. The following algorithm is often iterated multiple times to obtain the reference value $L_r^* \in \{L_r\}$.

The framework of the ACO algorithm can be described as follows:

Step 1 (Initialization): Initially, set the parameters about the vertices and time. Also, initialize the road pheromone (line 1 and above initialization).

Step 2 (Pheromone update): Pheromone update generally consists of two parts, decreases for pheromone evaporating as Eq.(2) (line 13) and increases for reaching the destination as Eq.(3) (line 21). However, to solve the dilemmas mentioned above, the optimized pheromone update also contains two unique scenarios (line 6-line 16). To avoid congested roads, we update the pheromones according to the corresponding road states during a specific period (line 6-line 9). Meanwhile, to solve the "dead alley", we set the pheromone value to 0 to effectively avoid choosing relevant roads (line 15-line 16).

Step 3 (Road selection): The ant k will choose road according to Eq.(1), a parameterized probability distribution (line 11).

Step 4 (Solution computation): The heuristic algorithm is repeatedly executed to construct the best-performing solution in the solution space (line 19-line 20).

We can significantly reduce the search space and accelerate the solving process with reference value L_r as an optimized strategy. In the next section, L_r will be combined with the dynamic retroactive A* algorithm to achieve better performance than typical algorithms in meeting multiple needs and efficiently solving problems.

Algorithm 1 Intelligent ACO algorithm

Input: The Graph $G(V, E, F)$,The Start Time t_0,The Start Vertex V_s and The Destinantion Vertex V_d
Ant Number m, The Current Time T_c. The Current Vertex V_c =V_s , The Previous Vertex V_p = V_s

Output: The Approximate Shortest Path Length L_r

Initialize the pheromone as τ_0
for k$\subseteq[1, m]$ **do**
  $T_c \longleftarrow t_0$,i $\longleftarrow V_c$
  for all j connected to i **do**
    $\delta t \leftarrow e_{ij}/v_u$
    if $e_{ij} \in N_i$ at time period $[T_c, T_c + \delta t]$ **then**
      phero $[j] \Leftarrow [(\tau_{ij})]^{\alpha} [\eta_{ij}]^{\beta}$
    else
      phero $[j] \longleftarrow 0$
    end if
    Then the ant k chooses the next road e_{ij} according to Eq.(1), update the route L_k.
    V_c = j,V_p = i,$T_c \longleftarrow \delta t + T_c$
    Update the current road pheromone by Eq.(2)
  end for
  if N_i is None **then**
    $\tau_{V_p i} = 0$.Announce the next ant k' to move
  end if
  if V_c= V_p **then**
    if L_k<L_r **then**
      $L_r \Leftarrow L_k$
      Update the whole path(which arrived destination) pheromone by Eq.(3)
    end if
  else
    continue to search
  end if
end for
return L_r

REFERENCES

[1] S. Ebadinezhad, "Deaco: Adopting dynamic evaporation strategy to enhance aco algorithm for the traveling salesman problem," *Engineering Applications of Artificial Intelligence*, vol. 92, p. 103649, 2020.