APPENDIX A
PERFORMANCE WITH DIFFERENT METRICS

To further investigate the effectiveness of our navigation algorithm, we will compare our algorithm with the newly proposed single source shortest path (sssp) algorithm based on Dijkstra's algorithm using different metrics.

Fig. 1: Complex Structure Map

The complex structure map shown in Fig.1 is adopted as a research scenario due to its complex environment containing various multi-metrics cases. In our experimental conditions, the road states are divided into four gradients, numbered 0, 1, 2, and 3, from low to high, indicating that QoS is not generally satisfied, fairly satisfied, and very satisfied, respectively. The experiments are developed under the following settings:

1. Two types of test cases. The experiments are executed on two types of test cases, one with the starting-end vertex pair as the variable and the other with the time period, i.e., population density, as the variable.

2. Two evaluation metrics. There exist two metrics to evaluate the QoS performance. One is to examine the percentage of roads with a status above 0 in the total number of roads, and another is to determine the respective percentage of status for each gradient.

The comparison results of relevant experiments are presented in Fig.2.

According to the experiments data shown in Fig.2, we can analyze the results and obtain relevant conclusions:

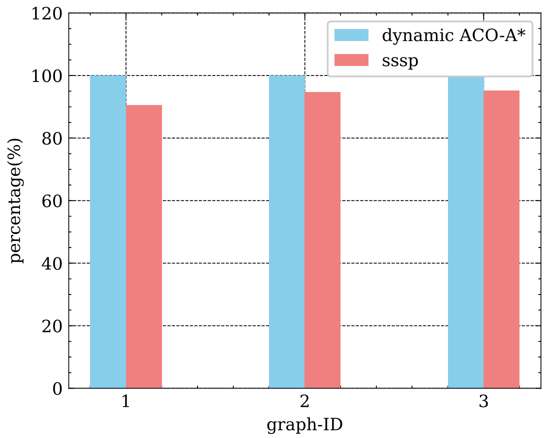
1. The start-end pairs variable: First, in the QoS-satisfying case shown in Fig.2(a), our dynamic ACO-A* algorithm can ensure that the QoS quality above users' criterion throughout the whole route process (ID 1-3), while there always exist some routes below the requirement when navigating with sssp algorithm. Then in the QoS-level case presented in Fig.2(b), the QoS level in our routes is relatively stable (dynamic ACO-A* 1-3) compared with that in sssp solving routes. Besides, the main body of our routes is concentrated in levels 2 and 3, which possess the higher quality. Thus it can be analyzed that the percentage distribution of QoS levels approximates a normal distribution. Therefore, the QoS-aware paths obtained by our algorithm are planned routes with a mean value μ of about level 2, which can generally meet the network quality requirements of users.

2. The time period variable: In the QoS-satisfying case described in Fig.2(c), due to changes in population density, some roads are always congested during rush hours (ID 2-3). However, our algorithm can still ensure more routes meet the demand in contrast with the sssp algorithm. As for the QoS-level case shown in Fig.2(d), its comparison effect is similar to the case depicted in Fig.2(b), and our algorithm still performs better whether some roads are permanently blocked (dynamic ACO-A* 2-3) or not (dynamic ACO-A* 1). Thus, for cases like Graph ID 2 and 3, the paths obtained by our algorithm can still be considered as approximately normally distributed with a mean value μ around level 2, but with larger variance values τ compared to the cases like Graph ID 1.

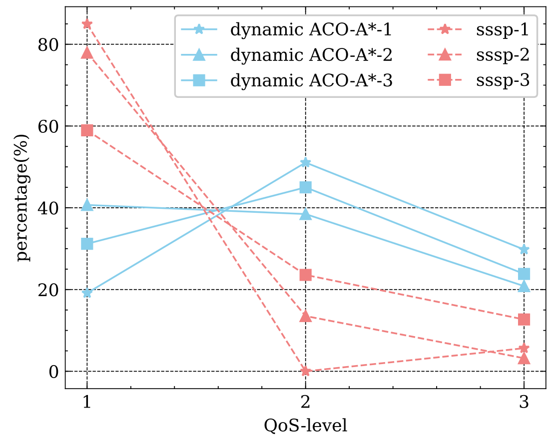
APPENDIX B
EFFICIENCY PERFORMANCE ANALYSIS

To investigate the time efficiency of the proposed algorithm, we measure the algorithm on the rather more complex map shown in Fig.1, where the time consumption situations are more complicated and deservedly more convincing. All algorithms were implemented in C++ and run on a Windows 10 platform with a AMD Ryzen 5 4600U with Radeon Graphics processor (2.1GHz) and 16384MB RAM.

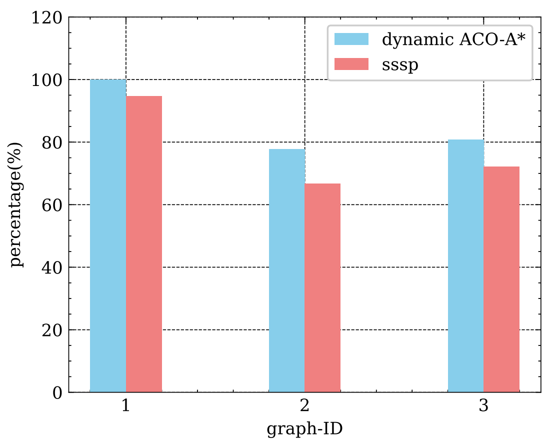
In the complex structure map, we execute the experiments with several different starting-end vertex pairs and different time ranges as variables, aiming at testing our algorithm in various scenarios. These input data are processed by the primary dynamic A* algorithm and the dynamic ACO-A* algorithm. With the same optimal solution but certainly a gap in time consumption,



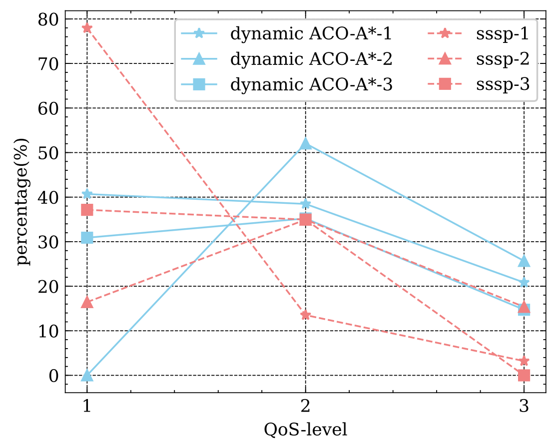
(a) QoS-satisfying comparison(start-end pairs)



(b) QoS-level comparison(start-end pairs)



(c) QoS-satisfying comparison(time period)



(d) QoS-level comparison(time period)

Fig. 2: Different QoS metrics results comparison.

the comparison of the primary and the optimized can sufficiently demonstrate the optimization effects. The comparison results are presented in Fig.3.

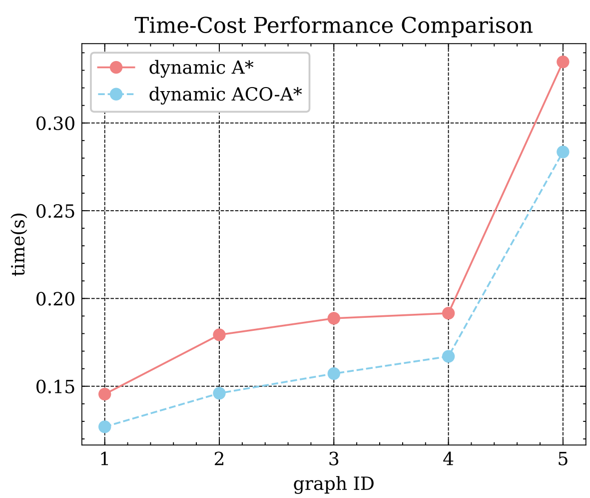


Fig. 3: Efficiency Comparison

In the experiment, the size of the input graph increases. As a result, we can see that the running time also increases. From the comparison of the results, we can see that with the help of the ACO reference values, the performance is improved by a factor of 1 to 1.5 in terms of time consumption.

In summary, the dynamic ACO-A* algorithm can well balance the QoS requirement and obtain the relatively shortest path with low cost in time consumption.