

Appendix B. The pseudocodes for all algorithms.

Table B1 The pseudocode of GA+ALNS combined with Q-learning.

Algorithm 1 The GA+ALNS algorithm combined with Q-learning

Input: Algorithm parameters

Output: Best solution schemes and objective value;

```
1  Set  $iter \leftarrow 0$ ,  $FITNESS \leftarrow 0$ ,  $POP \leftarrow \emptyset$ ,  $BESTsol \leftarrow \emptyset$ ,  $GENERATIONS, Q\_table \leftarrow 0$ 
2  Initialize  $POP$  // initialize population
3  Initialize Q-learning hyperparameters ( $\alpha, \gamma, \epsilon$ );
4  Initialize ALNS operator weights; // Initialize weights for destroy and repair operators
5  while  $iter < GENERATIONS$  do
6      Calculate  $FITNESS$  // calculate the fitness of each individual in the population
7      Sort  $POP$  by  $FITNESS$  // sort the population by fitness in descending order
8      for  $i \leftarrow 0$  to  $POPSIZE$  do
9          Select to create  $POPpar$  // by tournament selection
10     end for
11     // Get state features of the population ( $Diversity, Average Fitness$ )
12      $State \leftarrow GET\_STATE\_FEATURES(POP)$ ;
13     for  $i \leftarrow 0$  to  $POPSIZE$  do
14         // Select crossover strategy using Q-learning's  $\epsilon - greedy$  policy
15          $Crossover \leftarrow SELECT\_CROSSOVER\_STRATEGY(State, Q\_table, \epsilon)$ 
16     end for
17     for  $i \leftarrow 0$  to  $POPSIZE$  do
18         // Select mutation strategy using Q-learning's  $\epsilon - greedy$  policy
19          $Mutation \leftarrow SELECT\_MUTATION\_STRATEGY(State, Q\_table, \epsilon)$ 
20     end for
21     // Generate offspring via genetic operations
22      $offspring \leftarrow GENETIC\_OPERATIONS(POP, Crossover, Mutation)$ 
23     // Evaluate fitness of offspring
24     Evaluate  $FITNESS$  of each individual in offspring;
25     // Update Q-table based on the new state and reward
26      $Next\_State \leftarrow GET\_STATE\_FEATURES(offspring)$ ;
27      $Reward \leftarrow CALCULATE\_REWARD(POP, offspring)$ 
28     Update  $Q\_table(Q\_table, State, Crossover, Mutation, Next\_State, Reward)$ 
29     Select destroy operation based on the weights of ALNS destroy operators
30     Select repair operation based on the weights of ALNS repair operators
31      $Individual \leftarrow repair(destroy(Individual))$  // get new solution
32     Evaluate  $FITNESS$  of each individual in offspring;
33     if  $offspring$ 's best  $FITNESS < BESTsol$ 's  $FITNESS$ 
34          $BESTsol \leftarrow offspring$ 's best individual
35     end if
36     Update the weights of destroy operators and repair operators based on scores
37      $POP \leftarrow SELECT\_NEW\_POPULATION(POP, offspring)$ 
38      $iter \leftarrow iter + 1$ 
39 end while
```
