Tank Behaviour Decision Based on Behaviour Tree and SA-QL

Ruifeng Liu* and Mengfan Tian
China Aerospace Academy of Systems Science and Engineering, room 708, Fucheng Road, 16th sector, Beijing, China
Email: liu_ruifeng163@163.com

Abstract. With the wide application of artificial intelligence technology in the military field, unmanned intelligent equipment will play an increasingly important role in the future equipment combat system, and autonomous decision-making ability will directly affect the combat performance of equipment. Therefore, the research on decision-making ability of unmanned intelligent equipment has important practical significance. This paper introduces the principle of behavior tree and SA-QL algorithm, proposes a modeling method of autonomous behavior decision based on behavior tree and SA-QL, and applies it to the construction of tank behavior decision model. By constructing the tank behavior tree model, the autonomous behavior decision is realized, and the tank model has the ability of continuous learning and evolution by using Q learning. Simulation results show that the built tank model can learn new decision instructions according to environmental changes and make behavioral decisions autonomously, which also proves the effectiveness of the proposed modeling method.

1. Introduction
With the rapid development of artificial intelligence technology and the proposal of new operational concepts, unmanned intelligent equipment will gradually become the backbone of future operations. At present, an important factor affecting the rapid development of unmanned intelligent equipment is the autonomous decision-making ability of unmanned intelligent equipment. The research on the decision-making ability of unmanned intelligent equipment has been the focus of research in various countries. The ideal autonomous decision-making method of intelligent equipment should have two characteristics: one is intelligence, the other is learning. At present, the commonly used methods include neural network, deep learning, and genetic algorithm and so on, which have achieved good results. However, its model construction method is relatively complex, and the learnability needs to be further improved.

2. Behavior Tree Configuration
2.1. Behavior Tree
The behavior tree is an inverted tree structure, composed of nodes and directed edges, which is defined as a binary BT=⟨N,E⟩, among

\[ N = G \cup D \cup A \cup C \cup R, \ E \in N \ast N \]  

(1)
N is a finite set of nodes, including composite node G, decorative node D, action node A, condition node C, root node R. E is the set of directed edges, \( \forall \{N_i, N_j\} \in E \) and \( N_i, N_j \in N \) where \( N_i \) is the parent node of \( N_j \)[1]. The composite nodes include: selection node, sequence node, parallel node, random node, condition node and action node belong to leaf node [2].

Behavior tree is an improvement of finite state machine and hierarchical finite state machine. It describes the switch between a group of finite tasks in a modular way [3]. Its essence is a piece of logic code, which can be converted into various programming language codes through behavior Tree Editor [4]. The behavior tree diagram is shown in figure 1.

![Figure 1. Behavior tree diagram.](image)

### 2.2. Tank State Space
In the course of battle, the state of tank will change with the change of battle environment. Because of the need of research, the state space of tank is set as four kinds: whether to find the target, whether to fight, the number of ammunition and the armor protection state. In the determination of state space, some state changes are continuous, so it needs to be discretized. Otherwise, with the increase of the complexity of the behavior tree, the number of States will continue to increase, and the continuous state combination will show exponential explosion, so it is impossible to carry out actual learning.

- Target found: is findtarget (yes, no).
- Whether to fight: is attack (yes, no).
- Ammunition quantity: ammo (none, low, high).
- Armor protection status (none, low, medium, high).

Among them, ammunition quantity none indicates no ammunition, low indicates insufficient ammunition, and high indicates sufficient ammunition. In the armor protection state, none indicates that the tank has been destroyed, low indicates that the tank armor is low, medium indicates that the armor is moderate, and high indicates that the armor is high [5].

### 2.3. Tank Behavior Space
In the behavior tree, each action node is a kind of behavior. Suppose there are four kinds of behavior spaces in the tank behavior space:

- **Patrol**: the tank patrols the designated route to search for the enemy.
- **Attack**: tanks patrol or are attacked and find the enemy to attack.
- **Avoid**: when the tank is attacked by the enemy, avoid the fire attack.
- **Retreat**: the tank withdraws from the battlefield when its ammunition quantity and armor protection are poor and its combat ability is lost.
2.4. Behavior Tree Configuration

2.4.1. Initialization Behavior Tree. Initialization behavior tree provides input for simulation environment construction and behavior tree reset, that is, determining what behavior is performed in what state. In this study, the behavior tree of initialization tank can be set in the behavior space, and the occurrence conditions of tank behavior are shown in Table 1.

**Table 1. Conditions of tank behaviour.**

| Behavioral  | Conditions                          |
|-------------|-------------------------------------|
| Patrol      | Not fighting, armor medium or high  |
| Attack      | Target found, battle, ammo, armor medium or high |
| Avoid       | Battle, in armor                    |
| Withdraw from the battlefield | Combat, no ammo or low armor |

2.4.2. Reconstruct Behavior Tree. The SA-QLAL algorithm is used to learn, and the convergent Q-value table is calculated. The corresponding states of each action in the Q-value table are sorted according to the Q-value size to get the state permission list of each action. After checking the corresponding status, the Condition node before the behavior node is replaced by the final status allow list for status query.

2.4.3. Sorting Behavior Tree. To rearrange the behavior tree, we first need to get the maximum Q value, and then rearrange the behavior tree according to the size of the parent node Q value. The method to obtain the Q value of the parent node is to select the maximum Q value of the state in the state allowed list contained in the parent node as the Q value of the parent node.

The sorting principle of the parent node is to prioritize the Q value of the lower level. The parent nodes of the same level are sorted according to the Q value size, and the parent nodes are sorted according to the way from left to right from large to small. When the sorting is moved, the connected child nodes under the parent node should move with the parent node. Loop through the sorting until the parent node of the root node connection is sorted.

3. Learning Process Based on SA-QL

3.1. Behavior Selection Strategy

Q-learning algorithm often adopts the -greedy strategy for action selection, but with the increase of learning times, non optimal actions are still selected by probability, which will increase the number of learning iterations and slow down the convergence speed, resulting in the decline of Q-learning algorithm performance [6]. Therefore, in this study, the Metropolis criterion of simulated annealing algorithm is selected as the action selection strategy of Q-learning algorithm. By dynamically adjusting the probability of non optimal action selection, the agent can break through the constraints of artificial experience and effectively improve the performance of the algorithm [7].

If the Metropolis criterion of simulated annealing algorithm is used as the behavior selection strategy, the probability of selecting the non optimal action is as shown in equation (2).

\[
P(a_t = a_r) = \begin{cases} 
1, & Q(s, a_r) ≥ Q(s, a_p) \\
\exp \left( \frac{Q(s, a_r) - Q(s, a_p)}{T} \right), & Q(s, a_r) < Q(s, a_p)
\end{cases}
\] (2)

With the learning iteration of the Q-learning algorithm, the temperature will continue to drop under the control of the cooling strategy. When the Q-learning algorithm is close to the convergence state, the
temperature \( T \) is also close to 0, and the probability of selecting the non optimal action is close to 0, which is more scientific and reasonable than the common green strategy [8].

In the simulated annealing algorithm, the cooling strategy will dynamically reduce the temperature with the increase of the number of iterations to change the transfer probability until the termination condition is reached. In this study, index cooling strategy is used for cooling control, such as equation (3).

\[
T_k = a^k T_0 = T_0 e^{\ln a} k \in [0.5, 0.99]
\]  

(3)

3.2. Algorithm Flow
The algorithm flow of SA-QL is as follows:
(1) Initialize Q value table, reward value \( R \), state space \( s \), behavior space \( a \);
(2) \( T = 0 \), initialization status;
(3) According to the greedy strategy, the action is selected randomly, and the execution action is determined according to the dynamic greedy strategy;
(4) Execute the action, and update the Q value table according to the reward value \( R \) and Bellman equation according to the action update status;
(5) Determine whether it is a target state. If it is a target state, perform step (6). Otherwise, \( t = t + 1 \). Repeat step (3) and step (4);
(6) Determine whether the eposide reaches the maximum eposide value. If it reaches the maximum eposide, the learning will end. Otherwise, according to the dynamic cooling strategy, reduce the temperature \( T \), end the current eposide, and eposide + 1;
(7) Repeat steps (2) to (6).

4. Simulation
Assume that \( \gamma = 0.9, a = 0.5, T_0 = 100, \) episode = 1000, \( a = 0.95 \). After learning from the reward table to obtain the convergent Q value table, conduct the behavior tree configuration, build the simulation environment in Unity, implement the code implementation of the corresponding nodes of the behavior tree, and use the Behavior Designer plug-in for visual editing to obtain the tank behavior tree. The running simulation diagram is shown in figure 2. The tank entity can perform the corresponding behavior according to the change of state, and the execution logic conforms to the learning result, which proves the effectiveness of this method.

![Figure 2. Simulation operation diagram.](image)

5. Conclusion
Aiming at the problem of autonomous decision-making of unmanned intelligent equipment, this paper proposes a method of tank behavior decision-making based on behavior tree and SA-QL. Based on the complexity, intelligence and learning of three common problems in autonomous decision-making
modeling of unmanned intelligent equipment, this paper combines the modelling of unmanned intelligent equipment with behavior tree and SA-QL algorithm, and gives a solution, which has achieved good results. This method can obtain intelligent decision-making behavior through training, and can realize self-evolution through subsequent training, and can be applied to the development of unmanned intelligent equipment.

References
[1] Fu Y C 2016 Research on CGF Behavior Modeling Based on Behavior Tree (National University of Defense Science and Technology).
[2] Qiu X 2017 Implementation of Non-player Role Artificial Intelligence in Role-Playing Games (South China University of Technology).
[3] Xu W 2019 Application of multi-sensor information fusion based on FSM in the programming of wheeled mobile robot Computer and Digital Engineering 47 2478-81.
[4] Shoulson A, Garcia F M, Jones M, Mead R and Badler N I 2011 Parameterizing behavior trees Motion in Games 144-155.
[5] Liu R F, Wang J S, Zhang H L and Tian M F 2019 Research progress and application of behavior tree technology 2019 6th International Conference on Behavioral, Economic and Socio-Cultural Computing (Beijing: IEEE).
[6] Fang J, Yan W J, Deng X Y, et al. 2017 CGF air combat behavior decision based on Q-Learning and behavior tree Computer and Modernization (5) 37-39.
[7] Tu H and Liu H X 2017 An improved Q learning algorithm and its application in behavior tree Computer Application and Software 12 241-245.
[8] Jia B B and Liu J Y 2019 Estimation of finite isometric constant based on simulated annealing algorithm Automation Technology and Application 38 9-11+31.