文章编号:1006 - 9348(2021)02 - 0301 - 07

# 基于 DP - SAMQ 行为树的智能体决策模型研究

陈妙云,王 雷,丁治强
(中国科学技术大学信息科技学院,安徽 合肥 230031)

**摘要**:在多智能体仿真中使用行为树进行决策具有直观、易扩展等优点,但行为树的设计过程过于复杂,人工调试时效率低下。引入 Q - Learning 来实现行为树的自动设计。为解决传统 Q - Learning 的收敛速度慢的问题,将模拟退火算法中的 Metropolis 准则应用到动作选择策略中,随着学习过程自适应改变次优动作的选择概率以及将动态规划思想应用到 Q 值更新策略。实验结果证明,基于改进的多步 Q - Learning 行为树的智能体决策模型具有更快的收敛速度,并且能够实现行为树的自动设计和优化。

**关键词**:多智能体;行为树;模拟退火;动态规划;用动态规划和模拟退火改进的多步 Q 学习
**中图分类号**:TP391 **文献标识码**:B

# Research on Agent Decision Model Based on Multi - Step Q - Learning Behavior Tree

CHEN Miao - yun, WANG Lei, DING Zhi - qiang
(School of Information Science and Technology, University of Science and Technology of China, Hefei Anhui 230031, China)

**ABSTRACT**: The use of behavior tree for decision - making in multi - agent simulation is intuitive and easy to expand, but the design process of behavior tree is complex and the efficiency of manual debugging is low. The paper introduced Q - Learning to realize the automatic design of behavior tree. In order to solve the problem of slow convergence speed of traditional Q - Learning, a simulated annealing algorithm was used to improve the action selection strategy of multi - step Q - learning, which reduces the probability of non - optimal action selection, and a dynamic programming algorithm was used to update Q value function in reverse order. The experimental results show that the agent based on the improved Q - Learning behavior tree has faster decision - making speed, and can achieve automatic scheduling while reducing the use of conditional nodes, and get more reasonable behavior decision.

**KEYWORDS**: Multi - agent; Behavior tree; Simulated annealing; Dynamic programming; DP - SAMQ

## 1 引言

多智能体仿真是当前仿真的主流研究方法,应用在军事,交通,社会安全娱乐游戏等广泛领域[1]。在多智能体仿真的过程中,智能体需要对不同场景下的不同事件进行决策,能否合理决策是判断仿真成功与否的重要依据。基于有限状态机的决策模型具有决策过程较为僵化的缺点[2],而模糊状态机[3]的优点在于随机性较好,但是其场景适应性差。行为树决策模型是当前智能体研究中重要的行为模型,广泛应用于仿真领域[4]。其优点在于提供了丰富的流程控制方法,可以更加直观的看到状态的改变,还具有扩展性好,易于编辑的特点。但目前基于行为树决策模型的多智能体仿真研究还有如下不足:

- 行为树的设计需要对不同的状态进行判断[5],涉及到大量的条件节点。当人物的行为逻辑较为复杂时,行为树会非常庞大,需要进行大量的调试工作,极大的加大了开发难度;
- 需要对单个物体或人物的行为树进行设计,大大降低了开发效率;
- 不合理的行为树设计会导致仿真时出现异常。

本文通过引入 Q - Learning 算法来解决行为树的上述不足。Q - Learning 具有优秀的自学习和自适应能力[6],可以

基金项目:中科院创新基金(高技术项目 CXJJ - 17 - M139),中科院重大专项课题(KGFZD - 135 - 18 - 027)
收稿日期:2019 - 05 - 27 修回日期:2019 - 08 - 04

用于实现行为树的自动化设计。但是传统的 $\varepsilon$ 贪心 Q – learning[7] 为了避免陷入局部最优的陷阱,需要对次优的动作进行探索,随着智能体学习经验的不断增加会产生大量的无效计算,存在难以收敛的问题。

基于以上问题,本文提出了一种新的人群仿真多智能体决策模型,该模型对传统的 $\varepsilon$ 贪心 Q – learning 算法进行了改进,并引入了行为树,使得模型能够对行为树节点进行自动重排,实现自动化调试。

## 2 多步 Q – Learning

传统 $\varepsilon$ 贪心 Q – Learning 算法[8] 是一种单步时序差分算法,在估计状态动作价值时只考虑下一步的信息,而忽略了未来决策对当前的影响,因此具有低预见性的缺点。$Q(\lambda)$ 学习算法[9] 是一种新的强化学习算法,它在估计状态动作价值时考虑了将来的所有状态奖励,极大的提高了算法的预见能力。但当状态 – 动作空间规模较大时,该算法的计算复杂度过高。基于该缺点,Nachum [10] 提出了一种多步 Q – Learning 算法,它在更新 Q 值时考虑了将来 n 步的信息,极大的降低了算法的计算复杂度,加快收敛;同时又增强了智能体的预见能力,使其决策更加合理。

其算法描述如算法 1 所示:

输入:初始 Q 表,S[m],A[m],E[m],E′[m]
输出:收敛的 Q 值表

For each episode
  $t=1$, 初始化 $s_i$
  For each step of episode
  采取动作 $a_t$, 得到收益 $r_t$, $s_t \rightarrow s_{t+1}$;
  $e_t' = r_t + \gamma \cdot V_t(s_{t+1}) - Q_t(s_t, a_t)$
  $e_t = r_t + \gamma \cdot V_t(s_{t+1}) - V_t(s_t)$
  Update Array $S, A, E, E'$
  *if*$(t < m)$

$$Q_{t+1}(s_t, a_t) = (1-\alpha_t) Q_t(s,a) + \alpha_t \cdot [r_t + \gamma \cdot \max_b Q_t(s_{t+1}, b)] \quad (1)$$

  *else*

$$Q_{t+1}(s_t, a_t) = Q_t(s,a) + \alpha_t \cdot [e_t' + \sum_{i=1}^{k-1} (\lambda\gamma)^i \cdot e_{t+i}] \quad (2)$$

  $t \leftarrow t+1$
  Until $s_i$ is the end state

在算法 1 中,输入数组的长度用 $m$ 来表示,$m$ 个步骤的状态、动作、$e$ 和 $e'$ 分别用数组 $S[m]$,$A[m]$,$E[m]$ 和 $E'[m]$ 来存储。当 $t < m$ 时,选用贪心 *Q – Learning* 的更新策略[11][12] 作为更新策略,如式(1)所示,当 $t \geq m$ 时,更新策略如式(2)所示。数组长度为 $m$,这是因为每次更新只用到当前状态的前 $m$ 个状态。$m$ 值由实验给出,具体将在 5.2 节介绍。算法步骤包括外层循环和内层循环,分别对应为 episode 和 step [13],每一个 step 的执行过程从初始状态到终止状态,每一个 episode 包含多个 step,直到 Q 值表收敛时结束学习。

## 3 改进的多步 Q – Learning 算法设计

多步 Q – Learning 算法的动作选择是基于 $\varepsilon$ 贪心策略,会有一定的概率选择非最优动作,而这概率保持不变会使得在算法学习后期产生许多无效的计算,导致算法收敛速度变慢,性能下降。

除此之外,其值函数更新策略也会降低算法的收敛速度。当采取动作从环境中得到收益之后,算法只是对当前状态 – 动作对的 Q 值进行更新。在学习过程中,该状态动作对会重复进行更新。

针对当前多步 Q – learning 难以收敛的问题,本文使用模拟退火算法中的 Metropolis 准则改进了多步 Q – learning 算法的动作选择策略,使用动态规划改进了值函数的更新策略,提出了一种改进的多步 Q – learning 算法。

### 3.1 动作选择策略的设计

在传统的 Q – Learning 算法中使用 $\varepsilon$ 贪心策略进行动作选择,会以 $\varepsilon$ 的概率选择次优动作。但是当智能体学习到足够多的经验后,$\varepsilon$ 的值仍然不发生改变,导致产生许多无效的计算,降低算法的收敛速度。

基于以上分析,本算法改进了 $\varepsilon$ 贪心动作选择策略,随着学习过程的不断迭代,$\varepsilon$ 值会逐渐降低。在学习初期,智能体经验不足,因此选择较大的 $\varepsilon$ 值对环境进行探索。随着智能体的经验不断增加,$\varepsilon$ 值逐渐减小,智能体更趋于利用而非探索。而模拟退火算法[14] 通过降温策略来改变转移概率,可以逐渐降低选择次优动作的概率。因此本文将 Metropolis 准则应用到动作选择策略中。

Peng [15] 将 Metropolis 准则应用到传统的单步 Q – Learning 算法中,有效的平衡了探索和利用的问题。

本文采用模拟退火动作算法来改进多步 Q – learning 算法中的动作选择策略,动作选择概率如式(3)所示:

$$p(a_p \rightarrow a_r) = \begin{cases} 1, Q(s,a_r) \leq Q(s,a_p) \\ exp[(Q(s,a_p) - Q(s,a_r))/T], Q(s,a_r) > Q(s,a_p) \end{cases} \quad (3)$$

式中,$a_r$ 和 $a_p$ 分别表示随机选择策略和贪婪选择策略对应的动作,当 $Q(s,a_r) \leq Q(s,a_p)$ 时,选择 $a_r$ 动作进行探索,通过探索非最优动作提升算法的性能,当 $Q(s,a_r) > Q(s,a_p)$ 时,以 $\exp[(Q(s,a_p) - Q(s,a_r))/T]$ 的概率探索非最优动作,$T$ 是温度控制参数,即 $T$ 值较大时,探索的概率会比较大,当 $T$ 值趋向于 0 时,不在探索非最优动作。$T$ 值的选取借鉴了模拟退火算法中的温度降温策略,其公式如(4)所示

$$T(N) = T_0 \exp(-AN^{1/M}) \quad (4)$$

$T_0$,$N$,$A$,$M$ 分别表示初始的温度,算法迭代的次数,指定的常数和待反演参数的个数。上式也可表示为:

$$T(N) = T_0 a^{N1/M} \tag{5}$$

其中 $\alpha \in (0.7,1)$，$1/M$ 通常取为 1 或 0.5。在算法的起始阶段，$T$ 一般取较大的值，依据式(3)可得 $p(a_p = a_r)$ 较大，此时选择随机动作的概率较大。随着不断迭代，依据式(4) $T$ 值会逐渐降低，即会进行降温处理，使得 $p(a_p = a_r)$ 变小，则选择非最优动作的概率变小，有利于在 Q－Learning 算法的后期更倾向于选择最优动作，随着迭代的进行，模型不断逼近收敛状态，T 值也逐渐接近 0。此时模型的动作选择策略退化为贪心策略，即不再进行探索，每次动作选择只选最优动作。通过上述分析可以看到，使用模拟退火动作选择策略可以较好的实现 ε 值的自适应变化，加速了 Q－Learning 算法的收敛。

基于 Metropolis 准则的动作选择策略流程图如图 1 所示。

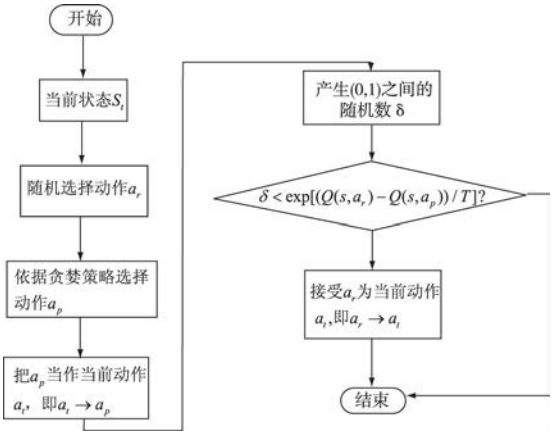


**图 1　基于 Metropolis 准则的动作选择流程**

### 3.2　值函数更新策略

在多步 Q－Learning 算法的学习过程中，智能体根据自身状态以及动作选择策略选择动作，并加以执行，从环境中得到收益。但是每一步只能在 Q 值表中更新当前状态－动作对的 Q 值。当 Q 值表没有收敛到最优时，每个状态下同一个动作会反复执行，这也造成了多步 Q－learning 算法的难以收敛。Shilova [16] 等提出了逆序的思想对传统的单步 Q－learning 算法进行了改进，可以加快收敛。原因在于对学习到的状态动作对进行逆序更新避免了相同状态动作对的重复执行。智能体根据自身状态选择动作加以执行并从环境中得到收益，更新 $Q$ 值。动态规划也是基于逆序的思想，本文中用动态规划来对 $Q$ 值进行更新。

本文算法使用邻接链表来存储到达不同状态前的状态动作对，以达到降低时间复杂度的效果。当智能体采取动作进入到下一状态后，邻接链表进行相应的更新，比如智能体经过$(s_1,a_1)$，$(s_2,a_2)$之后到达状态 $s_3$，那么此时在邻接链表中 $s_3$ 对应的就是$\{(s_1,a_1),(s_2,a_2)\}$。同时对邻接链表的状态进行逆序更新。更新步骤如图 2 所示。智能体根据当前状态选择动作并执行从环境中得到奖励后，根据 $Q$ 值更新公式更新当前状态动作对的 $Q$ 值，判断当前状态动作对的 $Q$ 值是否为该状态下所有动作的最大 $Q$ 值，如果是，就对邻接表中当前状态下的状态动作集的 $Q$ 值进行逆序更新。

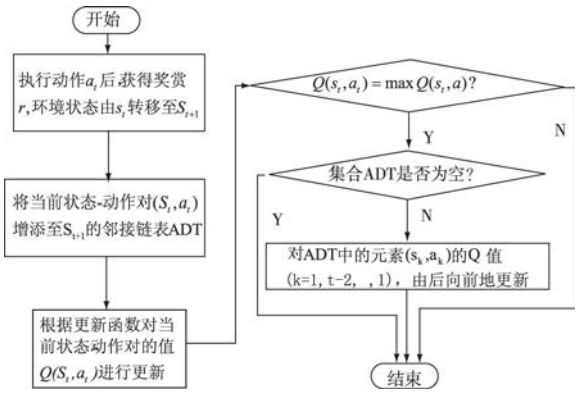


**图 2　动态规划逆序更新 Q 值**

### 3.3　改进的多步 Q－Learning 算法

本算法将 Metropolis 准则和动态规划分别应用到动作选择策略和值更新策略中，提出了改进的多步 Q－Learning 算法。本算法在每一 episode 的开始时初始化参数和 Q 值表，建立邻接链表。在每一 step 中，智能体根据当前状态 $s_t$ 和基于 Metropolis 准则的动作选择策略选择动作 $a_t$ 并且执行，从环境中得到相应的奖励 $r$，转移到 $s_{t+1}$ 状态，然后根据基于动态规划思想的值更新策略对 $Q$ 值表以及邻接链表进行更新。当到达终止状态时结束当前 episode 的学习。重新开始下一 episode 的学习。

本算法流程如图 3 所示。

其中存储状态，动作，$e$ 和 $e'$ 的四个数组分别是 $S[k]$，$A[k]$，$E[k]$ 和 $E'[k]$

## 4　人群仿真系统设计与实现

### 4.1　模型概述

本系统模型描述如下：

Step1：初始化行为树，动作空间，状态空间，状态转移表，奖励表和 Q 值表。

Step2：应用上述介绍的改进的多步 Q－learning 算法学习得到最终收敛的 Q 值表。然后根据不同的动作将收敛的 Q 值表划分成不同的状态允许子表，如图 4 所示。

Step3：将得到的不同动作的状态允许子表替换行为树的条件节点，并输出不同动作下获得的最大 Q 值；

Step4：根据不同动作节点得到的最大 Q 值对行为树进行重排；

Step5：输出重排后的 Q－learning 行为树。

步骤 1 是初始化行为树，其中动作节点代表智能体可以采取的动作，条件节点对智能体所处状态进行判断。状态空

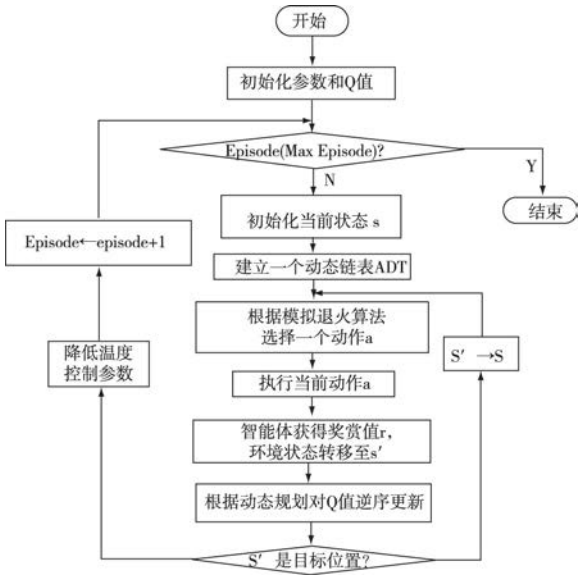


**图 3　改进的多步 Q – Learning 算法**

间,动作空间,状态转移表和奖励表都是人为给定的。

系统输入是一棵普通的行为树,输出是自动重排后的 Q – Learning 行为树。该系统首先应用上述介绍的改进的多步 Q – learning 算法得到最终收敛的 Q 值表。然后对初始行为树进行 dfs 搜索找到所有的动作集合,根据动作集合的动作将 Q 值表分割成不同的状态 Q 值集合,并将其替换掉行为树中的条件节点,并将最大的 Q 值填入动作节点对应的顺序节点中。从而行为树在选择动作的时候就不需要进行复杂的条件判断,只需判断当前状态是否存在动作节点对应的状态允许列表中即可,如果存在,就采取该动作。改造的 Q – Learning 行为树如图 4 所示。得到了含有 Q 值和状态允许列表的行为树后,根据动作节点对应的状态集合中的最大 Q 值对行为树进行重排序,得到的行为树即为系统输出。

以下内容将对 step2,3,4 分别进行具体描述。

### 4.2　行为树改造

应用本文提出的改进的多步 Q – learning 算法学习到收敛的 Q 值表之后,根据不同的动作将得到的 Q 值表分割成状态 – Q 值集合,对每一个集合中的状态根据 Q 值从大到小排序,根据实验情况保留一定比例的高 Q 值状态,命名为状态允许列表,用得到的不同动作对应的状态允许列表替换行为树中的条件节点,如图 4 所示。

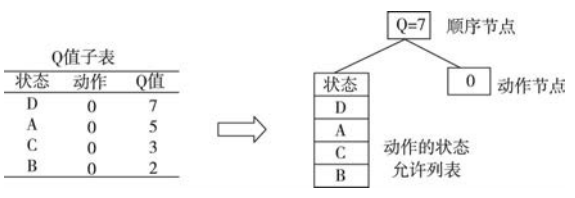


**图 4　将 Q 值信息整合到行为树**

### 4.3　选取最大 Q 值

状态允许列表中去掉 Q 值小于 0 的状态,并保留一定比例的高 Q 值状态。

算法 2 是选取最大 Q 值的伪代码:

```
输入:动作集合 A,状态集合 S,Q 值表
输出:输出动作 – 最大 Q 值
for each a in A
    for each s in S
    q = Q. getQ(s,a)
    if q > 0
      act_states[a]. insert(s)
    sort(act_states[a]) in reverse order
    act_states. remove[size * (1 – x):]
    res[a] = act_states[0]
```

### 4.4　行为树拓扑重排序

得到含有 Q 值的行为树后,根据 Q 值大小对行为树进行重排序。Q 行为树中节点的 Q 值自上而下自左而右依次减小,如算法 3 所示。

算法 3 是行为树重排的伪代码

```
输入:初始 Q – tree
输出:已重排 Q – tree
for each node in bottom layer
    if node. value > fa_node. value
      swap(node, fa_node)
      down(node)
    sort(node. childs)
```

图 5 为具体的重排过程。行为树对应节点的 Q 值从上往下,从左往右依次减少。重排后得到决策更加合理的行为树

## 5　实验设计与仿真分析

本文将改进的多步 Q – Learning 算法命名为 DP – SAMQ (Dynamic Programming Simulated Annealing Multi – step Q – learning)算法。本文的实验环境基于手动搭建的人群仿真场景,场景中部署了一定的监控设备。

在 Unity 3D 中仿照真实人物构造了相应的 3d 模型,并导入相应的运动动画,使其显示更加逼真。

### 5.1　实验设计

实验的场景基于城市中的一座广场,其中重要人物正在视察,安保人员的任务是保护重要人物的安全,嫌疑人的任务是找到攻击重要人物。

本实验选择基于模拟退火的单步 Q – Learning 算法(SAMQ)以及采用 ε 贪心策略的(GQ)算法与本文的 DP – SAMQ 算法进行对比实验。实验中的参数设置如下:折扣因

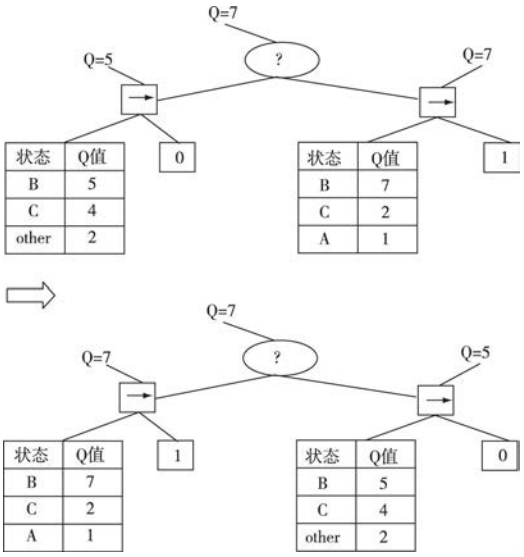


**图 5　根据 Q 值重排行为树**

子 γ 设置为 0.9,权重参数 λ 设置为 0.85,学习率 α 设置为 0.4,初始温度 $T_0$ 设置为 500。

Q – Learning 算法涉及智能体的状态,动作,状态转移,奖励,之后 Q – Learning 算法才能开始具体的学习。所以接下来将详细介绍嫌疑人和安保人员的状态空间,动作空间,状态转移和奖励的设计。

5.1.1　状态空间

安保人员的状态空间:(生命值 lv,与嫌疑人距离 dis_em, 与同伴距离 dis_pt, 是否被攻击 hited)

生命值 lv(0 – 1000):高(500 – 1000),中 (200 – 500),低(0 – 200),死亡(0);

是否被攻击(hited):T/F;

与嫌疑人距离(dis_em):近(0 – 50),中 (50 – 100),远( >100);

与同伴距离(dis_pt):近(0 – 50),中(50 – 100),远( >100);

为了避免维数灾难,对距离和健康值进行了离散化,模糊化处理。

危险分子的状态空间:(生命值 lv,发现重要人物 fl, 被安保人员发现 fd,发现安保人员 fp,与安保人员距离 dis_pl)。

生命值 lv(0 – 1000):高(500 – 1000),中 (200 – 500),低(0 – 200),死亡(0);

发现重要人物(fl): T/F;

是否被安保人员发现(fd): T/F;

发现安保人员(fp): T/F;

与安保人员距离(dis_pl):近(0 – 50),中(50 – 100),远( >100);

5.1.2　行为空间

安保人员的行为空间:(追击/pa,逃跑/ma,巡视/sa,寻找伙伴/fc,寻找嫌疑人/fe)。

危险分子的行为空间:(自首/sr,逃跑/ma,攻击重要人物/al,寻找重要人物/fl,攻击安保人员/ap)

5.1.3　动作选择策略

采用如上介绍的基于 Metropolis 准则的动作选择策略。

5.1.4　奖励表

**表 1　安保人员的奖励表**

| 状态 | 指标 | | | | | |
|---|---|---|---|---|---|---|
| | dis_em | dis_pt | lv | hited | Action | Reward |
| State1 | – | – | 0 | – | – | –10 |
| State2 | – | – | L | T | ma | 5 |
| State3 | – | N/M | L | F | fc | 10 |
| State4 | N/M/F | – | H/M | – | pa | 9 |
| State5 | – | – | H/M | F | sa | 4 |
| State6 | – | – | H/M | T | fe | 3 |

**表 2　危险分子的奖励表**

| 状态 | 指标 | | | | | |
|---|---|---|---|---|---|---|
| | fl | fd | lv | dis_pl | Action | Reward |
| State1 | – | – | 0 | – | – | –10 |
| State2 | – | T | L | N/M | sr | 4 |
| State3 | – | – | L | F | ma | 5 |
| State4 | T | – | H/M | – | al | 10 |
| State5 | F | T | H | N | ap | 2 |
| State6 | F | F | H/M | – | fl | 7 |

除表 1 中给出的状态 – 动作对有奖励值之外,其它状态 – 动作对奖励值均设置为 0。

5.1.5　行为树

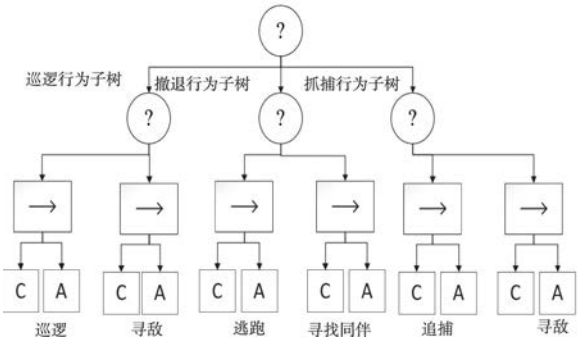


**图 6　安保人员初始行为树**

本实验中安保人员和嫌疑人的行为树设计如上。重排后行为树由多棵子树组成,从上到下从左到右 Q 值逐渐降低,输入状态,输出具有最大 Q 值的动作。嫌疑人初始行为树中存在的不合理地方在于当危险分子的健康值小于 200 且与安保人员的距离大于 100 时,应该优先逃跑而不是继续攻击安保人员。经过 DP – SAMQ 算法重排后的嫌疑人行为树则有较好的表现。

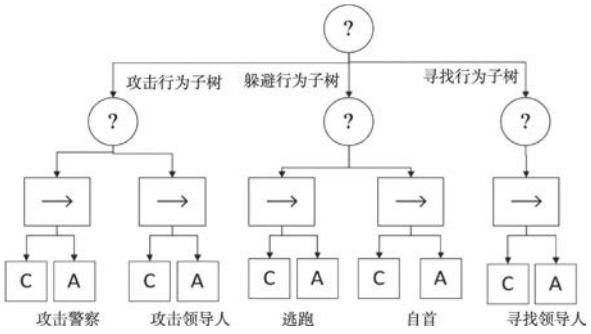


图 7　嫌疑人初始行为树

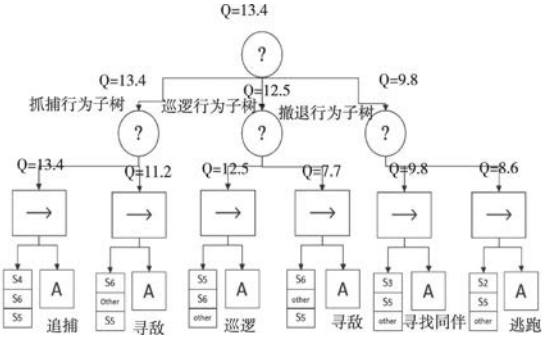


图 8　基于 DP – SAMQ 的行为树模型得到的安保人员的行为树

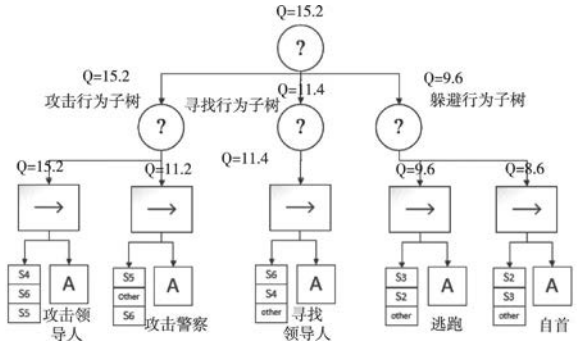


图 9　基于 DP – SAMQ 的行为树模型得到的嫌疑人行为树

接下来的实验需要找到改进后多步 Q – learning 的最优 n 值。本文接下来的实验分成两组。第一组是测试不同 n 值下算法的收敛速度,找到最佳 n 值,第二组验证应用了本文算法的行为树的合理性。

### 5.2　n 值的确定

从算法 1 中可以看出该算法的时间复杂度为 O( Episode _num * step_num)。

n 值决定了智能体的预见能力,n 值越大,预见能力越高,但是计算复杂度同时也加大,n 值越小,虽然计算复杂度降低,但是预见能力也越低。因此需要权衡 n 的取值。

实验过程中采取了不同的 n 值进行实验,分别取了 n = 10,20,. . ,100,在实验中发现当 n > 20 的时候,算法的收敛速度明显下降,因此本文只给出 n 为 10,20 时算法的收敛曲线。每次实验运行 1000 个 Episode。横坐标为 episode_num/40,纵坐标为 40 个 episode 的 steps_num 的平均值。

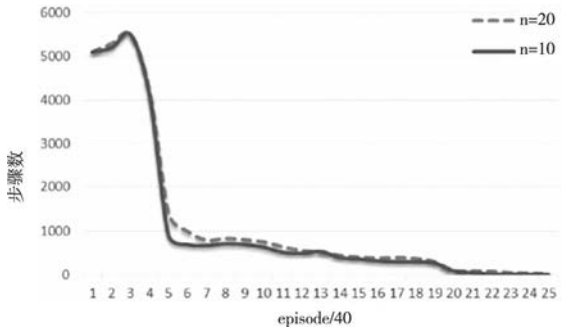


图 10　不同 n 值下 DP – SAMQ 算法的收敛曲线

从图中可以看出,当 n = 10 的时候,算法收敛速度更快,最终 steps_num 收敛为 17,而 n = 20 时,steps_num 最终收敛为 28。因此 n 值最终取 10。

### 5.3　实验一

下图是 GQ/SAQ 与本文算法( DPSAMQ) 的对比效果图,纵坐标为 log( steps_num) ,表示最终算法收敛时的步骤数。横坐标为实验的次数,总共是 100 次实验。

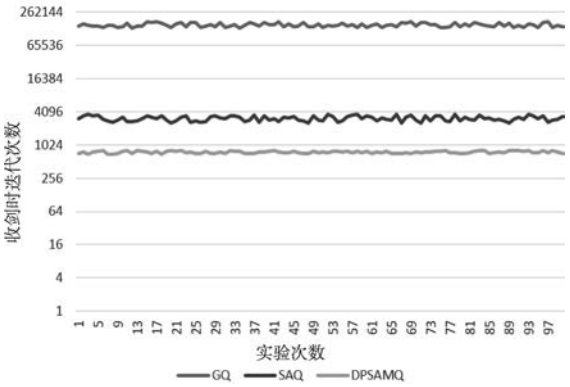


图 11　GQ/SAQ/DPSAMQ 对比实验

从图中可看出本文 DP – SAMQ 算法收敛所需迭代次数相较 GQ 和 SAQ 算法有非常明显的下降,算法性能显著提高。

### 5.4　实验二

从图 12/13 的对比可看出,基于本文算法( DP – SAMQ) 训练出来的嫌疑人在面临安保人员攻击时更倾向于逃跑,这是符合嫌疑人的行为逻辑的。而且行为树在选择动作时是基于 dfs 算法搜索的,从上到下从左到右的动作优先级逐渐降低,因此当嫌疑人被安保人员发现时优先选择逃跑而不是主动攻击。

初始行为树仿真结果如图 12 所示。

可以发现实验中嫌疑人被安保人员发现时选择进一步接近并攻击安保人员。

**图 12　初始行为树仿真结果**

基于 DP－SAMQ 算法学习得到的行为树仿真结果如图 13 所示。

**图 13　基于 DP－SAMQ 的行为树模型仿真结果**

从图中可看出，当嫌疑人被安保人员发现时，优先选择逃跑。从以上对比实验可看出，基于 DP－SAMQ 训练得到的行为树决策模型更加合理，证明了该算法的合理性

## 6　结语

针对目前传统的单步 Q－learning 算法预见性低，难以收敛以及 Q(λ)算法计算复杂度过大等缺点，本文引入了模拟退火算法中 Metropolis 准则对多步 Q－learning 算法中的动作选择策略进行改进，使得算法能够随着学习过程自适应改变次优动作的选择概率。并且本文还引入了动态规划来改进了 Q 值函数的更新策略，极大地加快了算法的收敛；并且本文还针对目前基于行为树的智能体决策模型存在人工开发效率低的缺点，将改进的多步 Q－leaning 算法应用到行为树决策模型中，实现了行为树的自动化设计和优化；通过实验也证明了本文算法（DP－SAMQ）的合理性与有效性。但是本文实验仍是基于单智能体，现实生活中更多是多智能体场景，因此在进一步的工作中会考虑引入多智能体。

**参考文献：**


[1] 张学锋，张成俊，白晨曦等. 基于智能体技术的多重灾难人员疏散感知模型[J]. 系统仿真学报，2016，28(3)：534 － 541.

[2] 李伟，门佳. 一种事件驱动有限状态机的编程实现框架[J]. 计算机与现代化，2014，6：116－119.

[3] G Mohmed，A Lotfi，C Langensiepen，and A Pourabdollah. Clustering－Based Fuzzy Finite State Machine for Human Activity Recognition[C]. in UK Workshop on Computational Intelligence，2018：264－275.

[4] 郝运. 行为树驱动的人工智能决策模式设计与实现[D]. 中国科学院大学(中国科学院沈阳计算技术研究所)，2018.

[5] 徐文胜，武博，蒋坚鸿. 武器装备虚拟维修训练系统行为树设计与实现[J]. 系统仿真学报，2018，(7)：37.

[6] 闫雪飞，李新明，刘东，等. 基于多分辨率的 multi－Agent 武器装备体系作战仿真研究[J]. 系统仿真学报，2017，29(1)：136－143.

[7] A D Tijsma，M M Drugan，and M A Wiering. Comparing exploration strategies for Q－learning in random stochastic mazes[C]. in 2016 IEEE Symposium Series on Computational Intelligence (SSCI)，2016：1 － 8.

[8] Schulman J，Chen X，Abbeel P. Equivalence between policy gradients and soft q－learning[J]. arXiv preprint arXiv：1704.06440，2017.

[9] 闫丰亭，贾金原. DP－Q(λ)：大规模 Web3D 场景中 Multi－agent 实时路径规划算法[J]. 系统仿真学报，2019，(1)：4.

[10] 唐克双，张桁嘉，衣谢博闻. 基于多智能体仿真的交通诱导系统效率评价[J]. 系统仿真学报，2018，30(7)：2630－2639.

[11] Nachum O，Norouzi M，Xu K，等. Bridging the gap between value and policy based reinforcement learning[C]. Advances in Neural Information Processing Systems，2017：2775－2785.

[12] M Hessel et al. Rainbow：Combining improvements in deep reinforcement learning，[C]. in Thirty－Second AAAI Conference on Artificial Intelligence，2018.

[13] Xi Y，Chang L，Mao M，et al. Q－learning algorithm based multi－agent coordinated control method for microgrids[C]. 2015 9th International Conference on Power Electronics and ECCE Asia (ICPE－ECCE Asia)，IEEE，2015：1497 － 1504.

[14] 刘炜，许嘉轩，王沛沛，等. 基于模拟退火算法的列车节能操纵研究[J]. 系统仿真学报，2018，(6)：41.

[15] Peng F，Cui G. Efficient simultaneous synthesis for heat exchanger network with simulated annealing algorithm[J]. Applied Thermal Engineering，Elsevier，2015，78：136－149.

[16] Shilova Y，Kavalerov M，Bezukladnikov I. Full Echo Q－routing with adaptive learning rates：a reinforcement learning approach to network routing[C]. 2016 IEEE NW Russia Young Researchers in Electrical and Electronic Engineering Conference (EIConRusNW)，IEEE，2016：341 － 344.

[17] 洪晔，王宏健，边信黔. 基于分层马尔可夫决策过程的 AUV 全局路径规划研究[J]. 系统仿真学报，2008，(9)：2361－2363，2367.



**[作者简介]**

**陈妙云**(1997－)，女(汉族)，海南文昌人，硕士研究生，主要研究领域为机器学习与多智能体仿真。

**王　雷**(1972－)，男(汉族)，安徽宿州人，副教授，硕士研究生导师，主要研究领域为未来网络以及多智能体仿真。

**丁治强**(1992－)，男(汉族)，广西壮族自治区桂林市人，硕士研究生，主要研究领域为机器学习与多智能体仿真。