강화학습을 이용한 다중 에이전트 제어 전략

이 형 일[†] · 김 병 천[††]

요 약

다중 에이전트 시스템에서 가장 중요한 문제는 여러 에이전트가 서로 효율적인 협동(coordination)을 통해서 목표(goal)를 성취하는 것과 다른 에이전트들과의 충돌(collision) 을 방지하는 것이다. 본 논문에서는 먹이 추적 문제의 목표를 효율적으로 성취하기 위해 새로운 전략 방법을 제안한다. 제안된 제어 전략은 다중 에이전트를 제어하기 위해 강화 학습을 이용하였고, 에이전트들간의 거리관계와 공간 관계를 고려하였다.

Multagent Control Strategy Using Reinforcement Learning

Hyong-Ill Lee[†] · Byung-Cheon Kim[††]

ABSTRACT

The most important problems in the multi-agent system are to accomplish a goal through the efficient coordination of several agents and to prevent collision with other agents. In this paper, we propose a new control strategy for succeeding the goal of the prey pursuit problem efficiently. Our control method uses reinforcement learning to control the multi-agent system and consider the distance as well as the space relationship between the agents in the state space of the prey pursuit problem.

키워드 : 강화학습(Reinforcement Learning), 다중에이전트(Multiagent), 먹이 추적 문제(Pursuit Problem), 제어전략(Control Strategy)

1. 서 론

M. L Minsky에 의해 소개된 강화학습(reinforcement learning)은 심리학(psycolgy) 분야에서 동물의 학습을 연구하는 과정에서 기원하였다[1, 2]. 강화학습은 동적 프로그래밍과 교사 학습을 혼합한 형태의 학습 방법으로서 학습을 수행하는 에이전트(agent)는 에이전트의 외부에 존재하는 환경(environment)과 시행-착오(trial-and-error)를 통해 상호작용(interaction)하면서 학습한다[3, 4]. 즉, 학습을 수행하는 에이전트는 학습을 수행하는 동안 주어진 환경에서 취할 수 있는 행동(action)을 시도(trial)하며, 외부 환경으로부터 에이전트가 선택한 행동에 대한 평가로서 스칼라(scalar) 형의 강화값을 받아 강화(reinforce)된다. 그러므로 강화 학습은 동적 환경(dynamic environment)에서도 효율적인 학습이 가능하기 때문에 Cart-pole 균형 문제[5], 교통신호 제어 문제[6], 엘리베이터 운행 제어[7], 그리고 로봇 이동[8]등과 같은 동적 환경에서 널리 이용되고 있다. 강화 학습을 위해 널리 이용되고 있는 학습 방법은 C. J. C. H. Watkins가 제안한 Q-학습[9]과 $Q(\lambda)$-학습[10]이 있다. 그러나 이들 강화 학습은 에이전트의 행동에 따른 강화 값이 의도된 목표에 도달할 때에만 주어지기 때문에 최적해에 매우 느리게 수렴한다. 또한 하나의 에이전트만을 위한 학습 방법이기 때문에 다중 에이전트 환경에서는 효율적으로 학습을 수행할 수 없는 단점이 있다.

본 논문에서는 다중 에이전트 환경에서 효율적으로 학습을 수행할 수 있는 강화학습 방법을 제안하였다. 제안된 방법은 다중 에이전트 환경하에서 에이전트들의 공동 목표에 보다 빠르게 도달할 수 있으며, 다중 에이전트 환경하에서 문제점 즉, 에이전트들간의 충돌 문제를 해결하였다.

본 논문의 구성은 다음과 같다. 2장에서는 관련 연구로서 강화 학습을 위해 널리 이용되고 있는 Q-학습과 다중 에이전트의 성능 평가를 위해 널리 이용되고 있는 먹이 추적 문제(pursuit problem)에 대하여 소개하고, 3장에서는 본 논문에서 제안한 다중 에이전트 제어 전략에 대하여 설명하였다. 그리고 4장에서는 제안된 방법을 먹이 추적 문제에 적용하여 성능을 분석하였고, 5장에서는 결론 및 향후 연구 방향을 제안하였다.

† 종신회원 : 김포대학 소프트웨어제작과 교수
†† 종신회원 : 한경대학교 웹정보공학과 교수
논문접수 : 2003년 3월 3일, 심사완료 : 2003년 5월 26일

2. 관련 연구

2.1 강화 학습

동적 환경에서 학습을 수행하기 위해 Q-학습이 널리 이용되고 있다. Q-학습은 통계적 동적 프로그래밍(stochastic dynamic programming)에 근거한 학습 방법으로서 학습을 수행하는 에이전트가 현재 상태(s_t)에서 어떤 행동(a_t)을 수행하였을 때 외부 환경으로부터 받는 강화 값(reinforcement value)에 대한 근사 값(approximation value)을 (상태-행동) 쌍에 대한 Q-함수, $Q(s_t, a_t)$에 할당한다. 그리고 나서 다음 상태(s_{t+1})의 (상태-행동) 쌍에 대한 Q-함수 $Q(s_{t+1}, a_{t+1})$가 최대가 되는 행동(a_{t+1})을 선택하여 현재 상태의 (상태-행동) 쌍에 대한 Q-함수 값과의 차(difference)를 이용하여 식 (2.1)과 같이 갱신하면서 학습을 수행한다.

$$Q(s_t, a_t) = (1-\alpha) \cdot Q(s_t, a_t) + \alpha \cdot \delta_t \tag{2.1}$$

식 (2.1)에서 α는 학습률(learning rate)이고, δ_t는 현재 상태에서 선택한 (상태-행동)에 대한 TD-오류(error) 즉, 현재 상태의 $Q(s_t, a_t)$ 값과 다음 상태의 $Q(s_{t+1}, a_{t+1})$ 값에 대한 차(difference)를 의미하며, 식 (2.2)와 같이 계산된다.

$$\delta_t = r_{t+1} + \gamma\{\max_{a \in A(s_t)} Q(s_{t+1}, a_{t+1}) - Q(s_t, a_t)\} \tag{2.2}$$

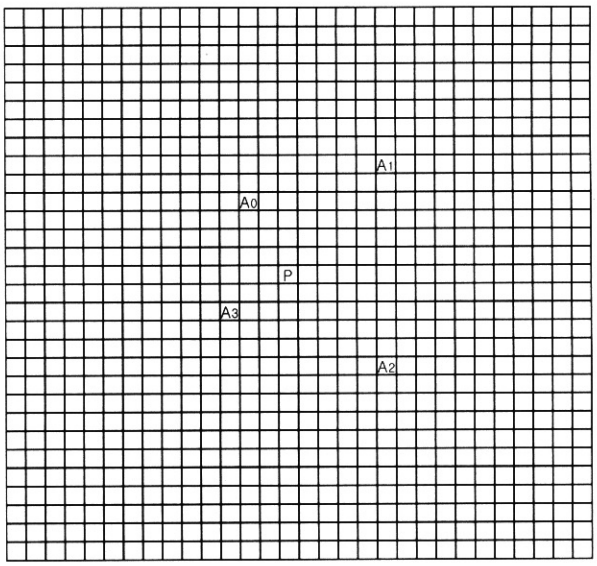
식 (2.2)에서 r_{t+1}은 강화 값이고, $\gamma(0 < \gamma < 1)$은 할인율(discount rate)이다. 그리고 $A(s_t)$는 학습을 수행하는 에이전트가 현재 상태(s_t)에서 선택 가능한 행동들의 집합을 의미한다. 현재 상태에서 에이전트가 어떤 행동을 선택할 것인가는 식 (2.3)과 같은 볼쯔만 확률 분포에 따라 선택하는 방법이 널리 이용된다.

$$p(\bar{a}|s) = \frac{e^{\frac{Q(s_t, \bar{a})}{T}}}{\sum_{a \in A(s_t)} e^{\frac{Q(s_t, a)}{T}}} \tag{2.3}$$

식 (2.3)에서 T는 행동 선택의 임의성(randomness)정도를 제어하는 온도 변수(temporal variable)이다.

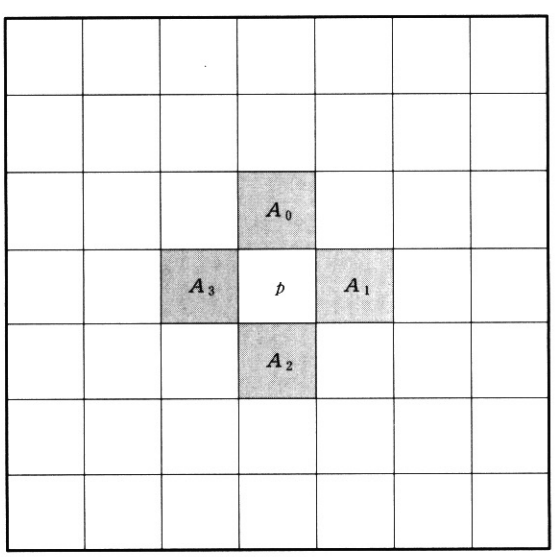
2.2 먹이 추적 문제

먹이 추적 문제는 분산 인공지능에서 잘 알려진 실험 환경으로 M. Benda[11]에 의해 처음 소개되었으며, 최근 복잡한 실세계의 전체적인 면을 표현할 수는 없지만 실세계의 개념을 구체화 해나가는 모델로 널리 이용되고 있다[12].



(그림 2.1) 먹이 추적 문제

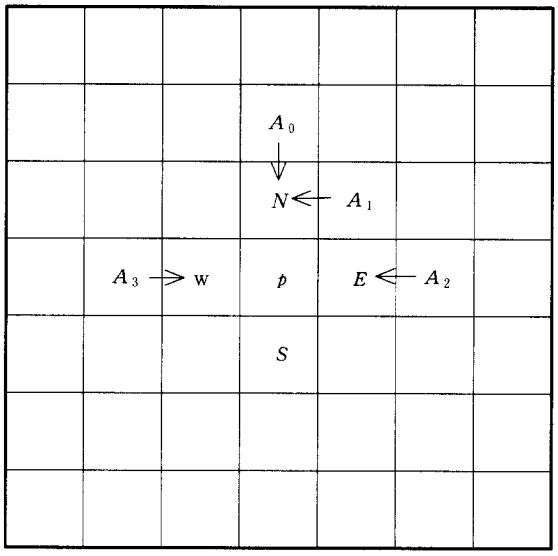
먹이 추적 문제는 (그림 2.1)과 같이 30×30 격자(grid) 환경에서 4개의 에이전트(A_i)들과 1개의 먹이(prey)로 구성되어 있으며, 에이전트들은 제어 전략에 따라 동, 서, 남, 북 즉, $A = N, S, E, W$ 중에서 한 방향으로 이동하며, 먹이는 $p = N, S, E, W, s$ 중에서 임의의 방향으로 이동한다. 여기에서 s는 이동하지 않고 제자리에 있음을 의미한다. 매시간 에이전트들과 먹이는 동일한 속도로 움직이며, 먹이 추적 문제에서 각 에이전트들의 공동 목표는 (그림 2.2)와 같이 먹이가 더 이상 움직일 수 없도록 포획하는 것이다.



(그림 2.2) 포획 상태

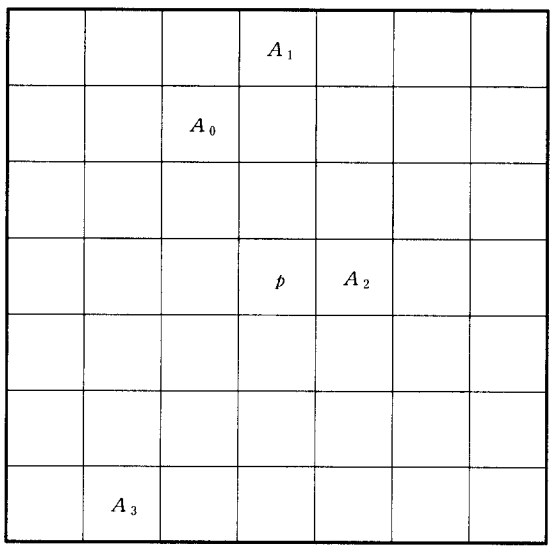
먹이 추적 문제에서 에이전트들간의 충돌 문제는 (그림 2.3)과 같이 먹이 p를 포획하기 위한 위치로 이동하기 위해 에이전트 A_0와 A_1이 동시에 N 위치로 이동할 경우 발생한

다. 이러한 충돌 현상은 에이전트들간의 협동이 원활하게 이루어지지 않으며, 에이전트들의 공동 목표에 빠르게 수렴할 수 없다[13, 14].



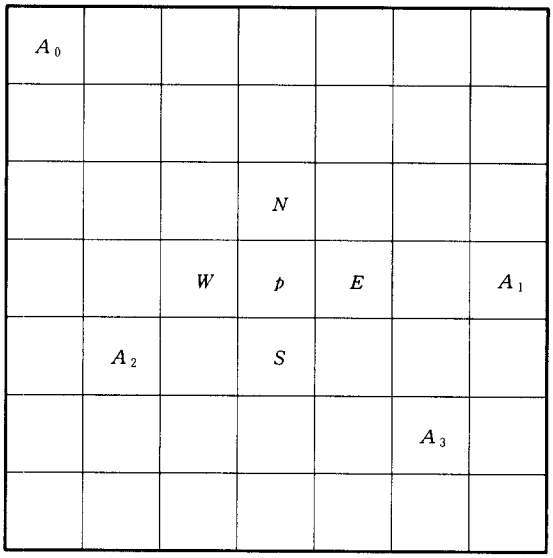
(그림 2.3) 충돌 현상

먹이 추적 문제에서 먹이를 효율적으로 포획하기 위해 Stephen과 Merx가 제안한 지역 제어 전략과 분산 제어 전략[15]이 널리 이용되고 있다. 지역 제어 전략은 비 대화형 제어 전략으로서 각 에이전트는 전역 목표인 먹이를 포획하기 위해 포획 위치(capture position)인 지역 목표를 먼저 설정하고 나서 이동한다. 지역 제어 전략은 (그림 2.4)와 같이 에이전트 A_2가 포획 위치를 선점하면 다른 에이전트들은 먹이의 위치를 인지하고, 먹이를 포획하기 위한 위치로 이동한다. 지역 제어 전략은 반드시 하나 이상의 에이전트가 먹이의 잡는 위치를 차지하여야만 각 에이전트간의 협동이 가능하다. 그리고 지역 제어 전략은 수렴 단계에 이르기까지 많은 상태 전이와 에이전트들 간에 포획 위치 중복에 따른 충돌 현상이 발생하는 문제점이 있다.



(그림 2.4) 지역제어 전략

분산 제어 전략은 대화형 제어 전략으로서 (그림 2.5)와 같은 상황에서 각 에이전트는 먹이 p를 포획하기 위한 포획 위치 N, S, E, W에 이르는 거리 정보를 <표 1>과 같이 생성한다. 그리고나서 각 에이전트는 먹이와 가장 멀리 있는 에이전트에게 가장 가까운 포획위치를 선정하도록 할당하고, 할당된 위치와 가장 가까운 거리를 갖는 상태로 이동한다. 예를 들어 <표 1>에서 먹이와 가장 멀리 떨어진 에이전트부터 가장 가까운 에이전트까지 즉, A_0는 포획 위치를 N로 선택하고, A_3는 S, A_1은 E, A_2는 W로 포획 위치를 선택하고 포획 위치와 가장 가까은 다음 상태로 이동한다.



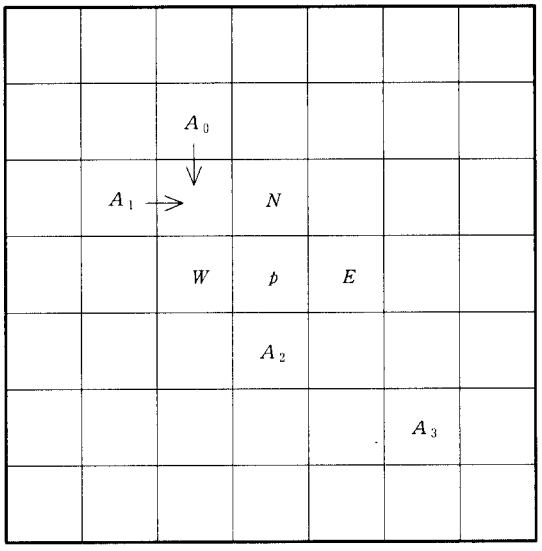
(그림 2.5) 분산제어 전략

〈표 1〉 각 에이전트의 잡는 위치의 거리 표

에이전트 \ 포획위치	N	S	E	W	p
A_0	***3.60***	5.00	5.00	3.60	***4.24***
A_1	3.16	3.16	***2.00***	4.00	3.00
A_2	2.82	2.00	3.16	***1.41***	2.23
A_3	4.47	***2.82***	3.16	4.24	3.60

분산 제어 전략은 지역 제어 전략의 잡는 위치의 중복에 따를 충돌 현상을 극복하기 위해 제안된 전략으로 충돌 현상을 감소 시켰으며 먹이를 중심으로 에이전트간의 협동을 잘 나타낸 전략이다. 그러나 (그림 2.6)과 같이 분산 제어 전략에 따라 먹이와 제일 먼 거리에 있는 에이전트 A_3가 먼저 E 지역을 포획 위치로 선정하고 나서 이동한다. 그리고 나서

A_0 와 A_1은 N 와 W지역이 같은 거리에 있기 때문에 임의의 한 곳을 목표로 정한다. 이 과정에서 A_0는 W, A_1은 N를 목표로 하여 화살표 방향으로 이동할 경우 충돌 현상이 발생한다. 이와 같은 충돌 문제는 다중 에이전트 시스템에서 중요한 문제로 제기되고 있다.

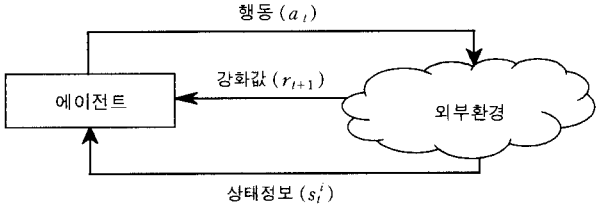


(그림 2.6) 분산제어 전략의 충돌 문제

3. 강화 학습을 이용한 제어 전략

본 논문에서 제안한 강화 학습을 이용한 다중 에이전트 제어 전략은 먹이를 포획하기 위하여 4개의 에이전트가 서로 협동과 조정을 통하여 충돌을 방지하고, 보다 빠르게 먹이를 포획하기 위한 대화형 전략 즉, 분산 제어 전략이라 할 수 있다.

다중 에이전트의 상태공간은 $\langle S, a, h, r\rangle$로 정의한다. 여기에서 S는 상태 공간, a는 에이전트가 선택할 수 있는 행동 (a_t)들의 유한 집합으로서 $a_i = N, S, E, W$을 의미한다. h는 각 에이전트가 현재 상태에서 다음 상태로 전이하기 위한 상태전이 함수 그리고 r은 에이전트가 선택한 행동이 얼마나 좋은가에 대한 정도를 나타내는 강화 값으로서 이들의 관계는 (그림 3.1)과 같다.



(그림 3.1) 에이전트와 상태 공간과 관계

(그림 3.1)에서 처럼 각 에이전트 (A_i)는 먹이를 포획하기 위해 상태 공간으로부터 에이전트가 선택한 행동 (a_t)에 대한 보상으로 강화 값 (r_{t+1})과 다른 에이전트들에 대한 상태 정보 (s_t^i)를 입력받아 상태 전이 함수 (h)에 의해 최적의 행동을 선택하여 다음 상태 (s_{t+1}^i)로 이동한다. 각 에이전트는 현재 상태 (s_t^i)에서 선택 가능한 모든 (상태-행동) 쌍에 대한 평가 값을 식 (3.1)과 같이 갱신한다.

$$\begin{aligned} &\text{For all } (s_t^i, a_t) \quad a \in A(s_t) \\ &Q(s_t, a_t) = (1-\alpha)\, Q(s_t, a_t) \\ &\quad + \alpha[r_{t+1} + \gamma\{\max Q(s_{t+1}, a_{t+1}) - Q(s_t, a_t)\}] \end{aligned} \tag{3.1}$$

식 (3.1)에서 α는 학습율, γ는 할인율(discount factor) 그리고 r은 에이전트가 선택한 행동에 대한 평가를 나타내는 강화 값이다. 강화 값 (r_{t+1})은 식 (3.2)와 같이 계산된다.

$$r_{t+1} = \frac{DR(s_{t+1}^i, p) + SR(A^j, A^i, A^k)}{2} \tag{3.2}$$

식 (3.2)에서 $DR(s_{t+1}^i, p)$는 에이전트 (A^i)가 선택한 다음 상태 (s_{t+1})와 먹이와의 거리 관계를 나타내며, 에이전트가 먹이를 추적할 때 먹이와 가까운 상태로 이동하기 위한 것으로서 식 (3.3)과 같이 계산된다.

$$DR(s_{t+1}^i) = \frac{1}{\sqrt{(s_{t+1}^i.x - p.x)^2 + (s_{t+1}^i.y - p.y)^2}} \tag{3.3}$$

식 (3.2)에서 $SR(A^j, A^i, A^k)$은 에이전트 (A^i)와 다른 인접 에이전트들 (A^j, A^k)과의 공간적 관계를 나타내며, 에이전트가 먹이를 추적할 때 먹이와 에이전트 그리고 인접 에이전트들과 충돌을 방지하면서 추적하는 효과가 있으며 식 (3.4)와 같이 계산된다.

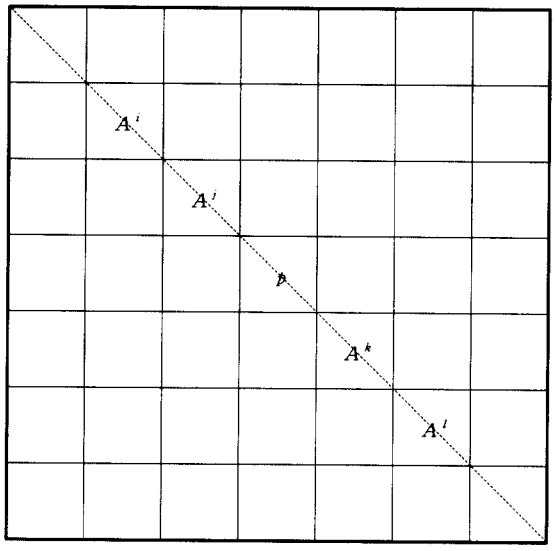
$$SR(A^j, A^i, A^k) = 1 - \frac{max(\theta_j, \theta_k)}{2\pi} \tag{3.4}$$

식 (3.4)에서 에이전트 (A^i)와의 인접 에이전트 A^j와 A^k는 (그림 3.2)와 같이 에이전트 (A^i)와 먹이 p와 이루는 직선상에 존재하는 경우와 직선상에 존재하지 않고 산재된 경우가 있다.

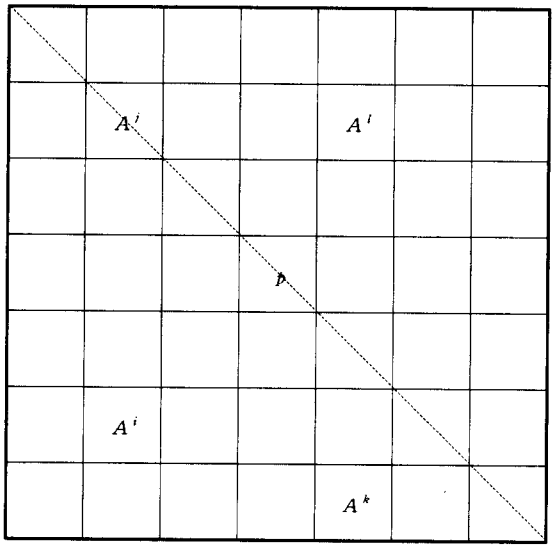
(그림 3.2)(a)와 같이 직선상에 존재하는 경우 식 (3.5)와 같이 (A^i)의 두 인접 에이전트 A^j와 A^k를 선택한다.

$$\begin{aligned} &A^j = A^{((i+1)\%n)} \\ &A^k = A^{(i+(n-1)\%n)} \\ &\text{여기서, } n\text{은 에이전트 수} \end{aligned} \tag{3.5}$$

에이전트가 (그림 3.2)(b)와 같이 산재되어 있는 경우 (그림 3.3)과 같이 에어전트 A^j와 먹이 p 그리고 다른 에이전트와의 공간적 관계를 이용하여 인접 에이전트를 선택한다.



(a) 직선상에 존재하는 경우



(b) 산재된 경우

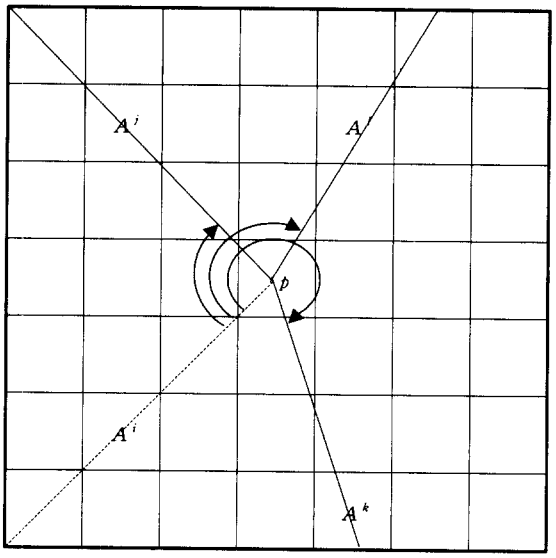
(그림 3.2) 인접 에이전트

(그림 3.3)에서 에이전트 A^i와 먹이 p를 중심으로 시계 방향으로 다른 에이전트들과 이루는 각을 구하여 다음과 같이 A^i와 인접한 두 에이전트 A^j와 A^k를 선택한다.

- A^j : 에이전트 A^i와 최소 각을 이루는 에이전트 ($\min(\angle A^i, p, A^j)$)
- A^k : 에이전트 A^i와 최대 각을 이루는 에이전트 ($\max(\angle A^i, p, A^k)$)

에이전트 A^i와 인접한 두 에이전트 A^j와 A^k가 선택되면 식(3.4)의 θ_j와 θ_k는 식 (3.6)과 같이 정의된다.

$$\theta_j = \angle A^i p A^j, \quad \theta_k = 2\pi - (\angle A^i p A^k) \tag{3.6}$$



(그림 3.3) 인접 에이전트와의 공간적 관계

에이전트 A^i가 선택할 수 있는 모든 (상태-행동) 쌍에 대한 강화 값이 계산되면, 에이전트 A^i는 식 (3.7)과 같은 상태 전이 함수 h에 의해 가장 큰 $Q(s_t, a)$ 값을 갖는 행동을 선택하여 먹이를 추적 하며, 먹이 추적 알고리즘은 (그림 3.4)과 같다.

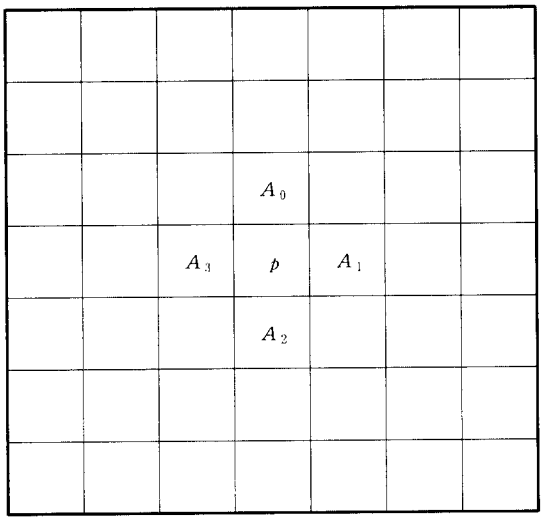
$$h = max(Q(s_t, a)),\ a \in A(s_t) \tag{3.7}$$

```
Pursuit_Problem( )
{
   Initialize Q(s, a) arbitrarily for all s, a ;
   Repeat {
      Move_Prey( ) ; // 먹이를 임의의 방향으로 이동한다.
      For all agents {
        Observe current state s_t ;
        // 에이전트가 선택할 수 있는 모든 다음 상태에 대해
        For all actions {
          Selection an action a_t at state s_t ;
          Observe s_{t+1}, r_{t+1} ; // 다음 상태와 강화 값을 구한다.
        }
        Choose a_{t+1} ;            // 다음 상태를 선택한다.
        Update_Qvalue (s_t, a_t)    // 현재 상태의 Q-값을 갱신한다.
        Move_Agent( ) ;             // 에이전트를 다음 상태로 이동
                                       한다.
      }
   } Until (Prey can't move)
}
```

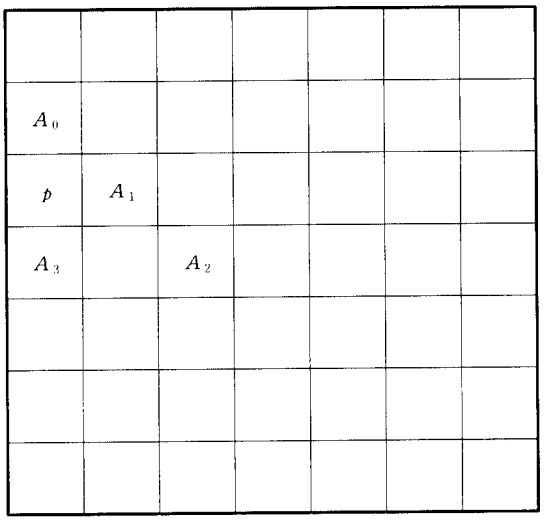
(그림 3.5) 먹이 추적 알고리즘

4. 실험 및 분석

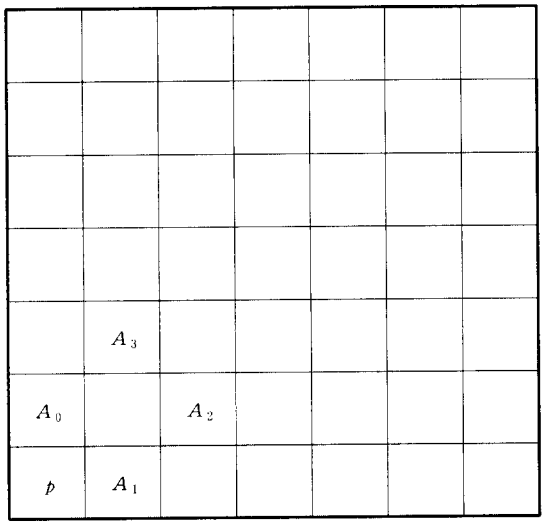
본 논문에서 제안된 강화 학습을 이용한 다중 에이전트 제어 전략과 Stephan과 Merx가 제안한 지역 제어 전략과 분산 제어 전략의 성능 평가를 위해 (그림 2.1)과 같은 30×30 격자 환경을 이용하였다. (그림 2.1)에서 에이전트들은 초기에 각각 (0, 0), (0, 29), (29, 0) 그리고 (29, 29)에서 제어 전략에



(a) 완전 포획



(b) 불안전 포획



(c) 탈출

(그림 4.1) 먹이 포획 상태

따라 N, S, E, S 방향으로 먹이를 포획하기 위해 이동하며, 먹이는 (14, 14)에서 N, S, E, W, s 방향 중 임의의 방향으로 이동한다. 에이전트들과 먹이의 이동 속도는 동일하며, 먹이 포획은 (그림 4.1)과 같이 4개의 에이전트가 먹이를 움직이지 못하도록 포획한 완전 포획(capture), 3개의 에이전트가 먹이를 움직이지 못하도록 포획한 불완전 포획(stalemate) 그리고 일정 횟수 만큼 시도하였으나 먹이를 포획하지 못하였거나 2개의 에이전트가 먹이를 움직이지 못하도록 한 탈출(escape)로 구분한다.

일반적으로 먹이 추적 문제에서 성능 평가 기준은 먹이 포획의 성공률(success rate), 각 에이전트의 상태 전이 수(number of transitions) 그리고 에이전트들 간의 충돌 횟수(number of collisions)를 기준으로 한다.

본 논문에서 제안한 강화 학습을 이용한 제어 전략과 Stephan과 Max가 제안한 지역 제어 전략 그리고 분산 제어 전략을 각 100회씩 실행한 후의 포획 결과와 포획 과정에서 발생한 충돌 횟 수는 <표 2>와 같다.

〈표 2〉 먹이 포획 결과

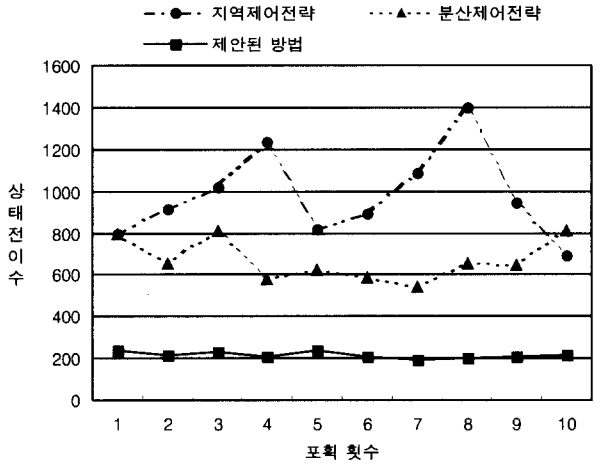
포획 상태 / 제어 전략	완전 포획	불완전 포획	탈출	충돌 횟수	
				최소	최대
지역 제어 전략	68	24	8	49	138
분산 제어 전략	82	13	5	5	98
제안된 제어 전략	85	15	0	0	

<표 2>에 나타난 것처럼 100회 반복 실행하는 동안 지역 제어 전략은 68회 완전 포획을 하였고, 8번을 포획하지 못하였다. 그리고 에이전트들 간의 충돌 횟수는 49회에서 138회까지 발생하였으며, 분산 제어 전략은 82회 완전 포획을 하였고, 5번을 포획하지 못하였다. 그리고 에이전트들간의 충돌 횟수는 5회에서 98회까지 발생하여 지역 제어 전략보다 분산 제어 전략이 다중 에이전트에 더 효율적임을 알 수 있었다. 그러나 본 논문에서 제안한 제어 전략은 85회 완전 포획을 하였고, 에이전트들 간의 충돌 횟수는 최대 12번 밖에 발생하지 않아 에이전트와 먹이와의 거리 관계 뿐만아니라 다른 에이전트들간의 공간적 관계를 고려하는 것이 더욱 효율적임을 알 수 있었다. 또한 지역 제어 전략과 분산제어 전략 그리고 본 논문에서 제안한 제어 전략이 먹이를 포획하기 위한 상태 전이 수는 <표 3>과 같다.

〈표 3〉 상태 전이 수

상태전이 / 제어 전략	상태 전이 수		
	최 소	최 대	평 균
지역제어전략	161	2443	978
분산제어전략	129	2317	666
제안된 제어전략	123	750	233

<표 3>에서 나타난 것처럼 먹이 추적 문제를 100회 실행하는 동안 지역 제어 전략은 상태 전이가 평균 978회, 분산 제어 전략은 평균 656회 그리고 본 논문에서 제안된 제어 전략은 233회 발생하였다. 그러므로 본 논문에서 제안된 제어 전략이 지역 제어 전략이나 분산 제어 전략보다 빠르게 먹이를 포획할 수 있음을 알 수 있었고, 100회 실험하는 동안 먹이를 10회 포획하는 동안 에이전트들의 평균 상태 전이 수는 (그림 4.2)와 같다.



(그림 4.2) 평균 상태 전이 수

(그림 4.2)에서 나타난 것처럼 지역 제어 전략과 분산 제어 전략은 먹이의 위치에 따라 상태 전이 수가 많은 변화를 일으켰으나 본 논문에서 제안된 방법은 먹이의 위치와 상관 없이 매우 빠르고 안정적으로 포획할 수 있음을 알 수 있다.

5. 결론 및 향후 연구방향

먹이 추적 문제는 다중 에이전트의 성능 평가를 위해 자주 이용되고 있으며, 다양한 실세계의 개념을 구체화하기 위해 널리 이용되고 있다. 실험을 통해 본 논문에서 제안된 제어 전략은 지역 제어 전략과 분산 제어 전략보다 매우 빠르게 먹이를 포획할 수 있고, 에이전트들 간의 충돌 현상이 거의 없음을 알 수 있었다. 그러므로 다중 에어전트를 제어하기 위해서는 에이전트들과 목표와의 관계만을 고려하는 것보다 에이전트들 간의 공간적 관계도 같이 고려하는 것이 더 효율적인 제어 전략임을 알 수 있었다.

본 논문에서 제안된 방법은 목표와 다른 에이전트들의 상태 정보를 알고 있는 대화형 제어 전략이므로 목표와 다른 에이전트들의 상태 정보를 알지 못한 상태에서는 적용이 불가능하다. 그러므로 목표와 다른 에이전트들의 상태 정보를 알지 못하는 상태에서 에이전트들의 공동 목표에 빠르게 수렴할 수 있는 비 대화형 다중 에이전트 시스템의 개발이 필요하다.

참 고 문 헌

[1] M. L. Minsky, Theory of Neural-Analoy Reinforcement Systems and Application to th Brain-Model Problem, Ph.D. Thesis, Princeton University, Princetion, 1954.

[2] M. L. Minsky, "Step towards aritificial intelligence," In Proceedings of the Institute of Radio Engineers, 49, pp.8-30, 1961.

[3] A. G. Barto, D. A. White and D. A. Sofge, "Reinforcement Learning and adaptive critic methods," Handbook of Intelligent Control, pp.469-491, 1992.

[4] A. W. Moore and C. G. Atkeson, "Prioritized sweeping : Reinforcement Learning with less data and less real time," Machine Leraning, 13, pp.103-130, 1993.

[5] C. W. Anderson, "Leraning to control an inverted pendulum using neural network," IEEE Control Systems Magazine, 9, pp.31-37, 1989.

[6] F. S. Ho, "Traffic flow modeling and control using artificial neural networks," IEEE Control Systems, 16(5), pp.16-26, 1996.

[7] R. H. Crites and A. G. Barto, "Improving Elevator Performance Using Reinforcement Learning," Advances in Neural Information Processing Systems, 8, MIT Press, Cambridge MA, 1996.

[8] S. P. Singh, "Transfer of Leraning by Composing Solutions of Elemental Sequential Tasks," Machine Leraning, 8, pp. 323-339, 1992.

[9] C. J. C. H. Watkins, "Technical note : Q-leraning," Machine Leraning, 8, pp.279-292.

[10] R. S. Sutton, A. G. Barto, "Reinforcement Learning : An Introduction," MIT press, 1988.

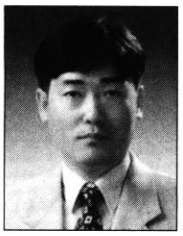
[11] M. Benda, V. Jagannathan and R. Dodhiawala, "On optimal cooperation of knowledge source-an empirical inverstigation," Technical Report BCS-G2010-28, Boeing Advanced Technology Center, Boeing Computing Services, Seattle, Washington, July, 1986.

[12] Peter Stone and Manuela Veloso, "Multiagent System : A Survey from a Machine Learning," Technical Report CMU-CS-97-193, The University of Carnegie Mellon, December, 1997.

[13] Sandip Sen, Mahendra Sekaran and John Hale, "Learning to coordinate without sharing information," National Conference on Aritificial Intelligence, pp.426-431, July, 1994.

[14] Tomas Haynes and Sandip Sen, "Evloving behavioral strategies in predators and prey," Adaptation and Learning in Multiagent System, Springer Verlag, Berlin, pp.113-126, 1996.

[15] L. M. Stephens and M. B. Merx, "The effect of agent control strategy on the performance of a DAI pursuit problem," In Proceeding of the 1990 Distributed AI Workshop, October, 1990.



이 형 일

e-mail : hilee@kimpo.ac.kr

1985년 명지대학교 전자계산학과(학사)

1994년 명지대학교 대학원 전자계산과 (석사)

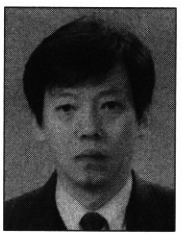
2000년 명지대학교 대학원 컴퓨터공학과 (박사)

1985년~1990년 (주)쌍용정보통신

1990년~1996년 (주)시에치노컨설팅

1997년~현재 김포대학 컴퓨터계열 조교수

관심분야 : 기계학습, 에이전트시스템, 정보검색 등



김 병 천

e-mail : bckim@hnu.hankyong.ac.kr

1988년 한남대학교 전자계산학과(학사)

1990년 숭실대학교 대학원 전자계산학과 (석사)

1999년 명지대학교 대학원 컴퓨터공학과 (박사)

1991년~현재 한경대학교 웹정보공학과 부교수

관심분야 : 기계학습, 다중객체최적화, 에이전트 시스템 등