

## ACS & 방향벡터 알고리즘을 이용한 비 대화형 멀티에이전트 전략에 관한 연구

김 현\*, 윤 석현\*\*, 정 태충\*\*\*

### A research on non-interactive multi agents by ACS & Direction vector algorithm

Hyun Kim\*, SeokHyun Yoon\*\*, TaeChoong Chung\*\*\*

#### 요 약

본 논문에서는 비대화형 에이전트(독립에이전트)를 바탕으로 멀티 에이전트 연구의 대표적 실험 모델인 먹이추적문제(preysuit problem)의 해결에 대한 전략을 제안하고 있다. 먹이추적문제는 가상 격자로 이루어진 공간 내에서 4개의 멀티 에이전트가 1개의 먹이(목표)를 포획하는 실험이다. 이것은 오래전부터 대화형 에이전트, 비대화형 에이전트로 구분되어 연구 되어왔으며 우리는 비대화형 에이전트를 이용하여 문제의 새로운 해법을 찾고자 하였다. 그리고 기존의 제한된 환경과는 전혀 다른 순환구조형 격자 공간에서 ACS를 이용한 방향 벡터 알고리즘을 통해 비대화형 전략의 새로운 해법을 제안할 수 있었다. 에이전트들은 개미의 습성을 응용한 ACS를 이용하여 학습을 하고 목표인 먹이는 에이전트의 학습 속도를 증가시키는 환경변수를 이용하여 이동방향과 경로를 결정하게 된다. 이전에 제기되었던 에이전트간의 정보교환(대화형 에이전트)방식에서 벗어나 비 정보교환(비대화형 에이전트)방식을 새롭게 적용하여 이를 해결할 수 있는 해법을 찾을 수 있었다는 것에 기존의 다른 멀티에이전트 연구와는 차별성이 있다.

#### Abstract

In this paper, We suggest new strategies on non-interactive agents applied in a prey pursuit problem of multi agent research. The structure of the prey pursuit problem by grid space(Four agent & one prey). That is allied agents captured over one prey. That problem has long been known in interactive, non-interactive of multi agent research. We trying hard to find its own solution from non-interactive agent method on not in the same original environment(circular environment). We used ACS applied Direction vector to learning and decide on a direction. Exchange of information between agents have been previously presented (an interactive agent) out

• 제1저자 : 김 현      • 교신저자 : 윤석현

• 투고일 : 2010. 10. 26, 심사일 : 2010. 11. 03, 게재확정일 : 2010. 11. 09.

\* 경희대학교 인공지능 박사 졸업예정    \*\* 정강문화산업대학 컴퓨터정보과 교수    \*\*\* 경희대학교 컴퓨터 공학과 교수

of the way information exchange ratio (non-interactive agents), applied the new method. Can also solve the problem was to find a solution. This is quite distinct from the other existing multi agent studies, that doesn't apply interactive agents but independent agent to find a solution.

▶ Keyword : multi agent, ACS, learning, prey pursuit problem, circular environment, direction vector

## I. 서론

다양하고 복잡한 현실세계의 동적 환경에서 인간의 요구와 문제 해결을 위해 에이전트 시스템이 개발 되어졌고, 이것은 초기에 단일 에이전트였지만 시간과 작업의 효율성을 위해 그리고 복잡한 문제의 해결 방안을 위해 멀티 에이전트 시스템으로 발전하게 되었다. 멀티 에이전트 시스템은 협동을 통한 작업을 수행하는데 크게 비 대화형과 대화형 시스템으로 구분된다[1,3,14,15]. 멀티 에이전트 연구를 위한 대표적인 실험 모델은 먹이추적문제인데 이것은 현실세계를 표방하는 격자 공간(grid)내에 4개의 에이전트가 1개의 먹이를 포획하는 실험으로서 복잡한 현실 세계를 표현하기 위해 MBenda에 의해 제안되었다[2,3].

먹이는 에이전트들이 도달하고자 하는 목표이며, 각 에이전트는 최소의 비용을 통해 효율적으로 목표 획득을 하는 것이 연구의 핵심이라 할 수 있겠다.

그러나 기존 실험 환경은  $n \times n$  크기의 제한된 격자 구조로 현실성이 결여되어 있었으며, 대화형 에이전트라는 기준으로 먹이를 포획하는 제한적 연구목표를 찾고자 하였다[4].

이런 이유로 본 논문에서는 동적환경을 표방한 순환구조형 격자 공간이라는 새로운 실험 환경 속에서 비대화형 에이전트들이 학습을 통해 실험의 목표인 먹이를 포획할 수 있는 알고리즘을 제시하고자 한다. 본 논문에서는 오래전부터 생물학의 개미습성을 응용하여 제안된 대표적 학습방법인 ACS(Ant Colony System)를 이용하였고 독립에이전트들은 학습을 통해 목표를 획득하는 새로운 해법을 제안하였다. 그리고 본 논문에서는 독립 에이전트를 기준으로 하였는데 이는 대화형 에이전트의 경우 통신을 위한 별도의 프로세싱 과정과 추가적 리소스로 인해 효율성이 떨어진다고 판단되었기에 독립적인 에이전트 기반의 실험을 진행하게 되었다. 본 논문의 구성은 다음과 같다. 서론에서는 연구의 배경과 소개를, 본론에서는 관련연구에 대한 설명과 논문에서 제안하는 새로운 해법을 그리고 결론에서 제안한 해법을 통해 결과와 향후 연구에 대한 비전을 제시하였다.

## II. 관련 연구 및 제안

### 2.1 먹이 추적 문제

먹이 추적 문제는 복잡한 현실세계에서의 에이전트의 활용을 실험하는 모델로 사용되어지고 있다[1,2]. 먹이 추적 문제는 (그림1)과 같이 격자(grid)환경에서 4개의 에이전트가 1개의 먹이를 전략에 따라 4방향에서 포획하는 실험이다. 에이전트와 먹이의 이동 조건(속도, 방향)은 동일하며 단, 먹이가 우선 움직임을 가진다. 이 실험을 통해 멀티 에이전트들이 목표인 먹이를 포획하는 것에 대한 다양한 연구들이 진행되었지만 에이전트간의 충돌(리소스 중복 사용, 데이터 중복 등)과 불안정하고, 불규칙한 포획(랜덤한 위치 배정으로 발생하는 문제들)등의 문제점으로 인해 다양한 전략과 해법들이 소개되었다[6,9].

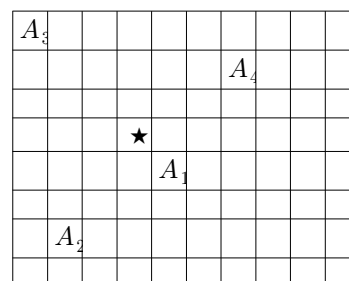


그림 1. 먹이 추적 문제  
Fig 1. Prey pursuit problem

### 2.2 순환구조형 격자공간

기존 먹이추적 문제의 실험환경은  $n \times n$  크기의 제한적 환경이었다. 이는 그림 2와 같은 불완전 포획이라는 문제가 발생한다[12].

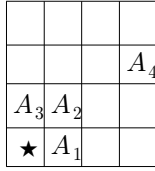


그림 2. 불완전 포획  
Fig 2. An incomplete capture

또한 현실성이 결여되어 현실세계를 표방한 동적실험환경이 필요하였고 이런 이유로 인해 순환구조형 격자공간이 제한되었다[5].

기존의 실험은 그림 1의 형태로  $30 \times 30$ 의 격자 공간상에 4개의 에이전트가 먹이를 포획하는 실험이었다. 하지만 본 논문에서는 제한된 공간의 실험환경은 현실성이 떨어지며 상태 행동에 대한 평가 시간이 길어진다는 것에 착안하여 보다 현실적이고 일반화된 환경을 제시하며 사각형의 그리드 구조를 순환구조라는 동적환경을 기반으로 한 연구를 하였다.

순환구조형 격자공간이란 현실세계와 비슷한 무한 공간 상태를 표현하기 위해 구현한 연속된 격자 공간이다.

(0,0)-(0,10)의 격자 공간에서 (0,10)다음은 오른쪽이 (0,11)이 되어야 하지만 순환시켜서 다시 (0,0)이 되게 하는 것이다. 예를 들어 (3,3)의 상하좌우는 (2,3), (4,3), (3,2), (3,4)이지만 (0,0)의 상하좌우는 (9,0), (1,0), (0,9), (0,1)이 되는 것이다. 이런 경우 에이전트들과 먹이의 속도가 같을 때에도 새로운 에이전트 전략으로 먹이포획이 가능할 수 있게 된다. 본 실험환경은 무한거리 모델인 메비우스띠를 모델로 하여 순환구조형 격자 공간이라 명하였다[5].

## 2.3 ACS(Ant colony system)

ACS는 오래전부터 세일즈맨의 경로 해결 문제(TSP)와 조합형 최적화문제들에 적용되어 다양한 알고리즘을 통해 다뤄져왔다. TSP 해결을 위해 제시된 ACS의 개미(Agent)는 목적지까지 지정된 경로를 따라 이동하게 되며 이때 발생한 페르몬의 양을 바탕으로 학습을 하게 된다[7].

논문에서는 ACS알고리즘을 적용하여 기존의 문제를 해결할 수 있는 새로운 알고리즘을 제시하였다. ASC는 본래 생물학에서 연구되어 왔던 개미집단의 습성을 기초로 하고 있다. 개미들은 생활의 기본이 되는 음식물을 찾아 탐험하고 발견된 음식물을 자신의 등지로 이동시키는데 페르몬 분비를 통해 길을 찾게 된다. 이때 페르몬은 시간이 지날수록 산도(Acidity)가 약해지는 단점이 있다. 때문에 많은 개미들, 혹은 초기에 이동한 개미가 먹이를 찾은 후 등지로 돌아올 때 새롭게 페르

몬을 분비하며 갱신함으로, 산도가 높아지게 되고, 산도가 약한 다른 경로를 무시하고 높은 산도의 경로가 이동경로로 정해지게 된다[8,10].

## III. 실험 및 결과

### 3.1 먹이의 회피 능력 부여를 통한 에이전트의 학습능력 향상

기존 먹이추적문제에서 먹이의 역할은 에이전트의 목표대상이었다. 에이전트에겐 추적과 포획, 협동의 능력이 주어졌지만 먹이는 무작위로 이동만을 반복하였다. 본 논문에서는 에이전트에게 학습능력을 주었고 학습능력의 향상을 위해 먹이에게 에이전트를 회피할 수 있는 가중치를 주었다. 먹이와 에이전트는 순환구조형 공간에 랜덤하게 위치되며 먹이가 먼저 에이전트를 회피하여 도망치게 되고 에이전트들은 먹이를 포획하기 위해 움직이게 된다. 이때 에이전트들은 움직임에 따라 먹이에게 근접하는 경우 강화 학습의 기본룰에 따라 보상 값(추후 행동을 결정하는 중요한 요소)을 받거나 먹이와 멀어지는 경우, 그리고 에이전트간의 충돌 등의 먹이포획에 나쁜 영향을 미치는 결과에 따라 페널티 값을 부여 받는 등의 과정 속에서 차츰 학습을 하게 되고 결국 최적의 이동경로를 찾게 되는 것이다.

에이전트들과 먹이는 거리와 방향성을 고려하여 만들어진 새로운 알고리즘을 통해 이동하게 되며 이들의 다음상태를 결정짓는 중요한 값은 행동에서 얻게 되는 환경변수이다.

본 논문에서는 독립된 에이전트의 학습을 통한 먹이포획 실험을 위해 다음과 같은 정의가 필요하다.

정의 1. 실험의 정의  
Define 1. Define of research

- |                                                                                                                                                                                                                                                                                                                                                          |
|----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| <p><b>정의 1.</b> 실험의 초기상태는 랜덤하게 배치된 각 에이전트들은 목표인 먹이와 에이전트 서로간의 위치 값을 가지고 있다.</p> <p><b>정의 2</b> 각 에이전트들은 독립적으로 실험이 시작된 후 서로 통신을 하지 못한다. 오직 먹이만을 파악할 수 있는 능력만이 주어진다.</p> <p><b>정의 3.</b> 에이전트간 충돌 시 실험은 다시 시작되며 학습을 위해 페널티 값이 주어진다.</p> <p><b>정의 4.</b> 모든 에이전트들이 먹이에게 근접할수록 보상 값을 받게 되고 먹이를 더 이상 이동하지 못하도록 포획 시 목표점을 획득한 것으로 여기고 최종 보상 값을 다시 부여하게 된다.</p> |
|----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|

에이전트는 서로의 상태 정보 값을 확인하고 먹이 쪽으로 이동한다. 먹이는 에이전트를 피하기 위해 에이전트와의 공간

관계를 계산하고 에이전트들이 없는 상태 공간 쪽으로 이동한다. 여기서 에이전트와 먹이의 이동속도가 같고 먹이가 에이전트보다 한 단계 먼저 이동할 때 포획이 상당히 어렵다는 것을 알 수 있다. 이런 이유로 에이전트들은 먹이 포획을 위해 효율적으로 쫓아 가야하는 것을 판단하는 함수를 만들고, 먹이는 근접상태의 공간을 파악할 수 있는 능력을 주어 도망할 수 있는 정책이 필요하여 방향벡터 알고리즘을 제안하게 되었다. 방향벡터의 기본 성질을 이용하여 먹이와의 거리 계산과 다른 에이전트들과의 상관관계를 반영하는 벡터 값을 생성하게 된다.

실험을 위해 에이전트는 기존의 연구들과 동일하게  $A_i \geq 4$ 로 설정하였다. 최소 에이전트 값을 4로 해야만 완전하게 먹이를 포획할 수 있기 때문이다.

격자공간에서 먹이가 이동하는 것을 4방향으로 한다면 그림 3과 같이 표현할 수 있다.

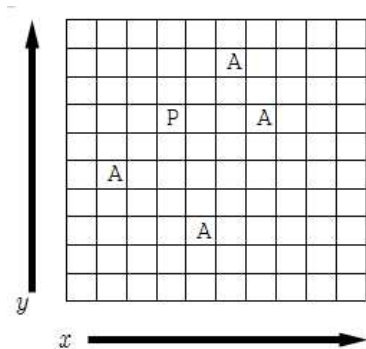


그림 3. 이동의 방향  
Fig 3. Direction of move

먹이  $P$ 를  $x$ 축으로  $\alpha$ 만큼,  $y$ 축으로  $\beta$ 만큼 이동시키는 함수는 수식(1)을 수식(2)에 적용한다.

$$\begin{cases} x' = x + \alpha, y' = y + \beta (P \text{가 이동할 위치를 } P(x', y') \text{ 할때}) \\ \int : (x, y) \rightarrow (x + \alpha, y + \beta) \text{ 이다.} \end{cases} \quad (1)$$

첫 번째로 먹이는 에이전트를 회피하기 위해 수식(1)에 의해 움직인다.  $\overrightarrow{EP}$ (vector of escape prey)는 모든 에이전트와의 방향과 거리 값을 고려하여 에이전트가 없는 곳으로 이동하게 된다.

이것은 에이전트 이동의 반대방향을 선택해야 하기 때문에 수식(3)에 역수를 취해준다.

$$\begin{aligned} \overrightarrow{EP} &= \overrightarrow{A_1P} + \overrightarrow{A_2P} + \overrightarrow{A_3P} + \overrightarrow{A_4P} \quad \dots\dots\dots (2) \\ &= \sum_i \overrightarrow{A_iP} \\ \rightarrow |\overrightarrow{A_iP}| &= \frac{1}{(Px - A_ix)^2 + (Py - A_iy)^2} \end{aligned}$$

각 에이전트들은 먹이와 마찬가지로 수식(1)을 따라 수식(3)에 의해 먹이로 이동하게 된다.

수식(3)은 에이전트( $A_i$ )가 먹이( $P$ )쪽으로 이동하는 방향벡터로 먹이와 에이전트와의 거리를 반영한 것이다. 각 에이전트와 먹이간의 방향 벡터  $\overrightarrow{VP}$ 의 크기는 먹이와 각 에이전트간의 거리에 반비례하고 가중치  $\alpha$  값을 주게 되는데 이것은 에이전트가 먹이와 가까울수록 먹이 방향으로 가려고 하는 에이전트의 성향을 ACS를 이용한 학습 값을 적용하고 이에 그 이동 벡터 값이 커져 에이전트의 이동을 결정하게 되기 때문이다.

$$\begin{aligned} \overrightarrow{VP} &= a \cdot \overrightarrow{A_iP} \quad \dots\dots\dots (3) \\ \rightarrow |\overrightarrow{A_iP}| &= ((Px - A_ix)^2 + (Py - A_iy)^2)^{-1} \end{aligned}$$

기존의 연구들에서 가장 많이 제기되었던 문제는 에이전트간의 충돌 문제였다. 이것을 해결하고자 본 논문에서는 에이전트와 다른 에이전트들 간의 상관관계를 고려한 벡터 함수를 만들었다. 실험 초기에 에이전트들은 각자의 상태를 확인할 수 있기 때문에 먹이를 향하지만 각자의 거리를 유지할 수 있는 함수를 만들게 된다. 수식(3)과 같이 정의할 수 있다.

다른 에이전트들과 거리가 가까워질수록 반대방향으로 값이 커지는 벡터  $\overrightarrow{VA}$ 를 생성하게 된다. 이것은 에이전트간의 일정한 거리를 유지하며 먹이에게로 접근할 수 있는 함수 값이 되며 이 값은 에이전트사이의 모든 방향 벡터의 합으로 만들어진다.

$$\begin{aligned} \overrightarrow{VA} &= \sum_{i \in J(A)} \overrightarrow{R_i} \quad \dots\dots\dots (4) \\ \rightarrow |\overrightarrow{R_i}| &= \left( \frac{1}{(R_ix - Ax)^2 + (R_iy - Ay)^2} \right) \end{aligned}$$

$J(A)$ 는 에이전트( $A$ )의 이웃 에이전트의 집합,  $R$ 은 에이전트( $A$ )와 이웃 에이전트,  $\overrightarrow{R}$ 은 에이전트( $A$ )와 에이전

트( $R$ )의 방향 벡터이다. 이것은 에이전트들 간의 거리가 가까울수록 반대방향으로 커진다. 그리고 에이전트와 다른 에이전트와의 벡터 값은 두 에이전트 거리의 제곱에 반비례하는 값을 갖게 하면 되는데, 이것은 에이전트와 에이전트 사이를 멀어지게 하는 작용을 한다. 수식(4)를 통해 에이전트는 다른 에이전트들과 다른 방향으로 가려고 하는 함수가 만들어져 에이전트간의 충돌문제를 해결하며 다른 에이전트들과는 먹이를 잡는데 있어 공동작전을 효과적으로 펼 수 있게 된다.

에이전트의 이동방향을 결정짓는 함수는 수식(5)으로 생성된다. 에이전트는  $\overrightarrow{VP}$ 와  $\overrightarrow{VA}$ 를 합친 쪽으로 이동하면 먹이 쪽으로 움직이게 된다. 이렇게 만들어진 에이전트의 이동 함수를  $\overrightarrow{VM}$ 이라 한다.

$$\overrightarrow{VM} = \overrightarrow{VP} + \overrightarrow{VA} \quad \dots\dots\dots (5)$$

### 3.2 에이전트의 학습

그런데  $\overrightarrow{VM}$ 을 구할 때 단순히 합하지 않고  $\alpha(\overrightarrow{VP}) + \beta(\overrightarrow{VA})$  형태로 변형시킨다.  $\alpha, \beta$  값은 학습 값이며 에이전트의 다음 상태를 결정짓는 중요 환경변수 값이다. 에이전트들은 이동을 통해 먹이로 근접이동하고 먹이를 포획하였을 때 학습을 하게 되는데 이 값들은 ACS 갱신 식을 통해 생성하여 상태 전이 값으로 적용하게 된다.

$$A_i(x,y) = \begin{cases} \frac{[\tau(x,y)] \cdot [\eta(x,y)]^\beta}{\sum_{u \in J_i(l)} [\tau(x,y)] \cdot [\eta(x,y)]^\beta} & j \in J_i(l) \\ 0 & otherwise \end{cases} \quad \dots\dots\dots (6)$$

수식(6)은 에이전트  $i$ 가 현재 위치에서  $P$ 로 이동할 확률을 표현하는 식으로 상태전이 규칙(state transition rule)으로 부른다.[7,8]

$\tau(x,y)$ 는 에이전트와  $P$ 사이에 분포된 페로몬의 양을 나타내고,  $\eta = 1/\delta$ 은  $\delta(x,y)$ ( $A_i$ 와  $P$ 의 거리)의 역수이고,  $J_i(l)$ 는 에이전트가 방문할 수 있는 남아있는 노드들의 집합이다.  $\beta$ 는 페로몬과 다른 에이전트와  $P$ 의 거리의 상대적 중요도를 결정하는 환경 변수이다.

$$\eta(\beta > 0)$$

$$c = \begin{cases} \arg \max_{j \in J(l)} [\tau(x,y)] \cdot [\eta(x,y)] & q \leq q_0 \\ C & otherwise \end{cases} \quad \dots\dots\dots (7)$$

$q$ 는  $[0,1]$ 사이에 분포된 무작위 파라미터이고,  $q_0$ 는 ( $0 \leq q \leq 1$ )사이 값을 가지는 인자,  $C$ 는 식(7)의 확률분포에 의해 따라 선택된 무작위 파라미터이다. 이것은 에이전트들이  $P$ 를 포획했을 때 생기는 페로몬이며 이는 간선 길이의 연산만으로 다음 위치를 선택하는 것을 벗어나 확률 분포를 이용해서 위치를 선택하는 과정이 추가됨으로써 국부 최적에 빠지기 쉬운 한계에서 어느 정도 벗어나고자 하고 있다.

$q < q_0$ 인 경우 에이전트는 탐색 행동(exploitation)을 취하게 된다. 이것은 장시간동안 축적된 정보(long-term)를 최선의 선택으로 하여 분비된 페로몬 정보와 단시간 생성된 정보(short-term)로 이동 거리와 관련된 휴리스틱 값을 사용하게 되며 확률 값은 페로몬의 양이 많고 이동한 간선의 거리 값이 작을수록 큰 값이 되어 최적이 된다. 반대로  $q > q_0$ 인 경우에는 에이전트는 편향된 탐색을 수행하기 위해 확률 분포(S)를 사용하여 길이가 짧고 많은 양의 페로몬을 가진 간선 선택을 선호하게 된다. 이때 확률 값이 큰 노드가 선택될 가능성이 높다.

본 논문에서 에이전트들은 수식(7)의 지역 갱신 규칙을 적용시켜 페로몬의 양을 갱신시킨다.

$$\tau(x,y) \leftarrow (1-\alpha) \cdot \tau(x,y) + \alpha \cdot \Delta\tau(x,y) \quad \dots\dots\dots (8)$$

수식(8)의  $\alpha$ ( $0 < \alpha < 1$ )는 페로몬의 증발을 막는 지연 파라미터이고

$$\Delta\tau(x,y) = \begin{cases} F & \text{if } (x,y) \in \text{global best choice by ant} \\ 0 & otherwise \end{cases}$$

의  $F$ 는 현재까지의 전역 최적 경로를 선택한 것이다.  $\tau(i,j)$ 는 페로몬 양으로, 최적의 경로면  $F$ 를 그렇지 않으면 0으로 설정하며, 지연 파라미터  $a$ 에 의해 페로몬 양은 감소하게 된다.

(먹이 쪽으로 이동하는 것이 방향의 다양성보다 중요하므로 일반적으로  $\alpha > \beta$ 로 설정한다.  $\alpha$ 값은 에이전트들의 먹이포획에 횟수 누적에 따른 확률 값을 통해 만들어진

다.[11,12] 이는 강화학습의 실험분야로 잘 알려진 ACS를 통해  $\alpha$ 값을 획득하며 알고리즘은 다음과 같으며 이는 수식(9)로 나타낼 수 있다.

```

 $\alpha$  value from ACS
{
  Initialize  $Q(s, a)$  arbitrarily for all  $s, a$ ;
  for {
    Move  $p()$ ; //  $p$ 는 도망친다.
    For all  $A_i$  {
      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  $s_{t+1}, \gamma_{t+1}$ ; //다음 상태와
        강화 값을 구한다.
      }
      Choose  $a_{t+1}$ ; // 다음 상태를 선택
      Update_Qvalue( $s_t, a_t$ ) // 현재 상태의 Q값을
      갱신
      Move_Agent();
    }
  } Until (Prey can't move)
}

```

알고리즘 1. ACS를 이용한 이동 알고리즘  
Algorithm 1. Move algorithm of take the ACS.

$$\alpha = \max(Q(s_t, a)), a \in A(s_t)$$

$$\overrightarrow{VM} = \alpha * \overrightarrow{VP} + \beta * \overrightarrow{VA} \dots\dots\dots (9)$$

수식(9)를 통해 생성된  $\overrightarrow{VM}$  벡터 함수로 에이전트의 이동 방향을, 벡터의 크기는 현재 좌표의 평가 값이 된다. 따라서 생성된  $\overrightarrow{VM}$ 을 가진 에이전트는 생성된 벡터 방향과 가장 가까운 셀(좌표)로 이동하는 것을 시도할 수 있다. 이동 후 보들과 현재 위치 각각에 대해서  $\overrightarrow{VM}$ 을 구해서 제일 큰  $\overrightarrow{VM}$ 의 위치로 이동하게 된다. 여기서 고려해야 할 사항은

먹이나 에이전트는 반드시 움직이지 않아도 된다는 것이다. 제자리에 머물러 있는 것도 하나의 전략으로  $\overrightarrow{VM}$ 을 생성할 수 있다. 즉 생성된  $\overrightarrow{VM}$ 을 통해 에이전트가 이동할 방향이 현재 위치보다 나쁘다면 움직일 필요가 없는 전략인 것이다.

에이전트와 달리 먹이는 각 에이전트에서 먹이 쪽 방향으로 거리의 제곱에 반비례하는 벡터 함수를 생성하고 이것들을 합한 방향으로 이동을 하면 에이전트들을 따돌릴 수 있게 된다.

### 3.3 실험의 평가 및 결과

본 논문에서 제안한 알고리즘의 성능 평가는 순환구조형 격자 공간에서 먹이 포획의 성공률(success rate), 각 에이전트의 상태 전이 수(number of transitions)를 기준으로 하였고, 멀티 에이전트의 연구목표에 최대한 근접하기 위한 실험으로 먹이의 완전포획(100%포획)과 에이전트간의 충돌 방지를 동시에 할 수 있는 전략을 적용하여 기존의 연구들에서 제안된 알고리즘[12]과 비교하여 보았다. 본 실험 환경은 현실성을 고려하여 불완전 포획이라는 요소를 없애고 초기 먹이와 에이전트의 위치는 무작위로 주어지며 먹이의 완전포획을 기준으로 에이전트의 학습 값  $\alpha$ 를 생성하였다.(상태전이 1000회 기준)

표 1은 ACS를 통해 획득한  $\alpha$ 의 증가에 따른 방향벡터 알고리즘의 효율성을 보여주고 있다.  $\alpha$ 의 증가함에 따라 먹이를 포획률도 증가하는데 이것은 ACS의 학습결과를 의미하기도 한다. 결과적으로 학습이 진행됨에 따라 에이전트의 상태 전이횟수도 감소함을 알 수 있었다.

표 1.  $\alpha$  값에 따른 먹이 포획 결과

Table 1. been has a result in the  $\alpha$

| (a) | (b) | 50×50(n × n) 순환형 공간 |                  |
|-----|-----|---------------------|------------------|
|     |     | Capture probability | State transition |
| 0.1 | 0.1 | 3%                  | 921              |
| 0.2 | 0.1 | 12%                 | 872              |
| 0.3 | 0.1 | 36%                 | 758              |
| 0.4 | 0.1 | 58%                 | 683              |
| 0.5 | 0.1 | 64%                 | 611              |
| 0.6 | 0.1 | 72%                 | 598              |
| 0.7 | 0.1 | 89%                 | 453              |
| 0.8 | 0.1 | 92%                 | 401              |
| 0.9 | 0.1 | 100%                | 350              |
| 1   | 0.1 | 100%                | 361              |

표 2는 기존의 먹이추적에 관한 대표 전략인 '강화학습제어 전략'[12], '분산제어 전략'[13]을 순환형 격자공간에서 실험한 결과를 보여준다. 새로운 환경의 특성에 맞는 새로운 전략의 필요성을 알 수 있었다.

표 2. 새로운 환경에서의 기존의 알고리즘 평가  
Table 2. A comparison with the before strategy in new environment

|           | 순환형격자 공간( $50 \times 50$ ) |       |
|-----------|----------------------------|-------|
| 분산제어전략    | 먹이포획                       | 26.2% |
|           | 에이전트 충돌                    | 21.4% |
| 강화학습 제어전략 | 먹이포획                       | 43%   |
|           | 에이전트 충돌                    | 36.1% |

그림 4는 에이전트의 상태전이가 증가할수록 에이전트간의 충돌문제가 해결되고 있는 것과 효과적으로 먹이 포획이 이루어짐을 확인할 수 있다.(100% → 충돌 없음)

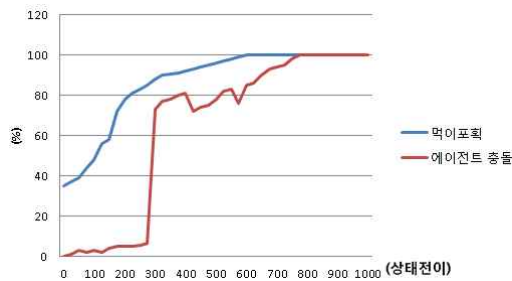


그림 4. ACS적용 방향벡터 알고리즘의 학습효과  
Fig. 4. A result is an application learning of new strategy

#### IV. 결 론

본 논문에서는 비대화형 에이전트들의 목표획득에 관련한 연구를 위해 ACS를 이용한 방향벡터 알고리즘을 제안하고 이를 멀티에이전트 연구의 대표적 실험 환경인 먹이추적문제에 적용하여 보았다. 제한된 격자 공간 내에서 기존의 실험은 현실 세계와는 많이 동떨어져 있다. 이런 이유로 본 논문에서는 현실 세계를 유사하게 표현하고 있는 순환구조(circular)형 격자 공간이라는 새로운 실험 환경에서 ACS를 이용한 학습을 통해 에이전트간의 충돌 방지와 효율적인 먹이 포획을 입증할 수 있었으며 에이전트 목표획득의 효율성이 증가함을 알 수 있었다.

기존의 에이전트간의 정보교환(대화형 에이전트)방식에서 벗어나 비 정보교환(비대화형 에이전트)방식을 새롭게 적용하여 이를 해결할 수 있는 해법을 찾을 수 있었다는 것에 기존의 다른 멀티에이전트 연구와는 차별성이 있다.

마지막으로 본 논문에서 제안한 알고리즘을 실생활에 응용할 수 있는 사례를 찾아보고 ACS의 효율성을 증가시킨 Ant-Q와 같은 다양한 학습방법을 이용한 실험이 필요하리라 생각된다.

#### 참고문헌

- [1] Peter Stone, Manuela Veloso, Multiagent coordination with learning classifier systems, In Proceeding of the AAAI 99 Workshop on Negotiation, pp. 44-49, 1999.
- [2] M. Benda, V. Jagannathan and R. Dodhiawala, 'On optimal cooperation of knowledge source-an empirical investigation,' Technical Report BCS-G2010-28, Boeing Advanced Technology Center, Boeing Computing Services, Seattle, Washington, July, 1986
- [3] 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
- [4] Ran Levy and Jeffrey S. Rosenschein, "A game theoretic approach to the pursuit problem", In Working Papers of the 11th International Workshop on Distributed Artificial Intelligence, February 1992.
- [5] 김현, 이승관, 정태충 '순환형 격자공간에서 방향벡터를 이용한 먹이추적문제 해결에 관한 연구' 제22회 한국정보처리학회 추계학술발표대회 11권 2호, 2005.
- [6] S. Cammarata, D. McArthur, And &, Steeb, "Strategies of Cooperation in Distributed Problem Solving", Proceedings of Eighth International Joint Conference on Artificial Intelligence, Karlsruhe West Germany, August 1993.
- [7] T. Fuku, A. Namatame, and T. Kaizouji. Collective eciency in two-sided matching. In P. Mathieu, B. Beauls, and O. Brandouy, editors, Artificial

- Economics: Agent-Based Methods in Finance, Game Theory and Their Applications, pages 115(126.Springer-Verlag, Berlin, Germany, 2006
- [8] A. Colomi, M. Dorigo, and V. Maniezzo, An Investigation of Some Properties of An Ant Algorithm. Proceedings of the Parallel Problem Solving from Nature Conference (PPSN 92), 1992, R. Mäanner and B. Manderick (Eds.), Elsevier Publishing, pp. 509 - 520.
- [9] 한상우, 김종원 '다자간 협업 환경에서 에이전트 기반 서비스 합성' 전자공학회 논문집, 논문 2008-45CI-5-10, 2008
- [10] M. Dorigo, V. Maniezzo, and A.Colomi. The Ant System: Optimization by A Colony of Cooperating Agents. IEEE Transactions on Systems, Man, and Cybernetics - Part B, vol. 26, No. 2, pp. 29 - 41, 1996.
- [11] V. Maniezzo, A.Colomi, and M.Dorigo, The Ant System Applied To The Quadratic Assignment Problem, Tech. Rep. IRIDIA/94-28, 1994, Université Libre de Bruxelles, Belgium.
- [12] 이형일, 김병찬 '강화학습을 이용한 다중 에이전트 제어 전략 정보처리학회지 10-B권 3호 pp.249-256, 2003
- [13] Stephens, L.M and Merxm, M.B, "The effect of agent control strategy on the performance of a DAI pursuit problem", In Proceedings of the 1990 Distributed AI Workshop.
- [14] 김광중, 고현, 김영자, 이연식 '이동 에이전트의 효율적 이주를 위한 최적 경로 탐색' 한국컴퓨터정보학회논문지 11-3권, pp.117-124, 2006년 7월
- [15] 명순희, '협동에이전트를 이용한 정보검색' 한국컴퓨터정보학회논문지, 제 5권, 제 2호, 43-49쪽, 2000년. 6월.

## 저 자 소 개



### 김 현

2005 : 경희대학교 인공지능 석사  
2011 : 경희대학교 인공지능 박사  
출업예정  
관심분야 : 멀티에이전트, 개인화 강  
화학습, 스마트 TV



### 윤 석 현

1979 : 인하대학교 전자 공학사  
1982 : 연세대학교 전자공학 석사  
2000 : 국민대학교 전자공학 박사  
1981~ : 동양공업전문대학 교수  
1996~현재 :  
청강문화산업대학 컴퓨터정보과 교수  
관심분야 : 프로그래밍 언어, 멀티미  
다이서비스, AIM 네트워크



### 정 태 충

1987 : KAIST 전산학과 공학박사  
1987 : KIST 시스템공학센터  
선임 연구원  
1988 ~ : 현재 경희대학교 컴퓨터  
공학과 교수  
관심분야 : 정음로마자 표기법, 인공  
지능, 인터넷 보안