다중에이전트 강화학습을 위한 SOM기반의 상태 일반화

임문택*, 김인철**

SOM-Based State Generalization for Multiagent Reinforcement Learning

Mun-Tack Lim*, In-Cheol Kim**

요 약

다중 에이전트 학습이란 다중 에이전트 환경에서 에이전트간의 조정을 위한 행동전략을 학습하는 것을 말한다. 본 논문에서는 에이전트간의 통신이 불가능한 다중 에이전트 환경에서 각 에이전트들이 서로 독립적으로 대표적인 강화학습법인 Q학습을 전개함으로써 서로 효과적으로 협조할 수 있는 행동전략을 학습하려고 한다. 하지만 단일 에이전트 경우에 비해 보다 큰 상태-행동 공간을 갖는 다중 에이전트환경에서는 강화학습을 통해 효과적으로 최적의 행동 전략에 도달하기 어렵다는 문제점이 있다. 이 문제에 대한 기존의 접근방법은 크게 모듈화 방법과 일반화 방법이 제안되었으나 모두 나름의 제한을 가지고 있다. 본 논문에서는 대표적인 다중 에이전트 학습 문제의 예로서 먹이와 사냥꾼 문제(Prey and Hunters Problem)를 소개하고 이 문제영역을 통해 이와 같은 강화학습의 문제점을 살펴보고, 해결책으로 신경망 SOM을 이용한 일반화 방법인 QSOM 학습법을 제안한다. 이 방법은 기존의 일반화 방법과는 달리 군집화 기능을 제공하는 신경망 SOM을 이용함으로써 명확한 다수의 훈련 예가 없어도 효과적으로 이전에 경험하지 못했던 상태-행동들에 대한 Q값을 예측하고 이용할 수 있다는 장점이 있다. 또한 본 논문에서는 실험을 통해 QSOM 학습법의 일반화 효과와 성능을 평가하였다.

Key words : 다중 에이전트, 강화학습, 신경망, QSOM

1. 서론

일반적으로 다중 에이전트 학습은 다중 에이전트 시스템에서 각 에이전트가 다른 에이전트를 고려하여 자신의 행동전략을 학습하는 것을 말한다. 따라서 다중 에이전트 학습에서 가장 중요한 문제는 다른 에이전트들과 얼마나 효과적으로 행위를 조정(coordinate)할 수 있느냐 것이다. 대부분의 전통적인 분산 인공지능 및 다중 에이전트시스템 연구들에서는 참여하는 에이전트들의 기능과 목적 그리고 환경의 제약 등을 고려하여 설계자가 미리 효과적인 조정방법을 고안하여 이것을 실제 시스템 운영 시에 적용함으로써 에이전트간의 행위조정을 이루어내려 했다. 대표적인 연구들로는 계약 망(contract net), 협상 프로토콜(negotiation protocols), 동맹(coalition formation) 등에 관한 연구들이 있다. 하지만 다중 에이전트 학습에서는 이러한 조정 지식 혹은 조정 방법을 참여하는 에이전트들이 스스로 경험을 통해 찾아내고 이것을 적용할 수 있도록 하는 것을 목적으로 한다.

* 경기대학교 전자계산학과
** 경기대학교 전자계산학과

이런 면에서 다중 에이전트 학습은 전통적인 분산 인공지능이나 다중 에이전트시스템에 관한 연구와 다르다고 볼 수 있다.

다중 에이전트 학습 방법은 참여 에이전트들이 서로 협조해야 하는 관계에 있는지 아니면 서로 경쟁하거나 대립하는 관계에 있는지에 따라 크게 달라진다. 또한 참여 에이전트들이 기능면에서 동일한지 이질적인지, 에이전트간에 통신이 가능한지, 통신이 가능하다면 빈도와 구조화 정도가 어느 정도인지에 따라서도 크게 달라질 수 있다. 본 논문에서는 다수의 동질 에이전트(homogenous agent)들이 공통의 목표를 위해 서로 협조해야 하는 다중 에이전트 시스템을 가정한다. 또 이러한 에이전트들 간에 서로 직접적인 통신이 불가능하여 각 에이전트들이 단지 관찰에 의해서만 다른 에이전트들과 협조를 이루어내야 하는 환경을 가정한다. 본 논문에서는 이러한 환경에서 각 에이전트들이 각자 독립적인 강화학습을 전개함으로써 서로 효과적으로 협조할 수 있는 행동전략을 학습하려고 한다. 하지만 기존의 강화학습의 경우 에이전트의 상태 공간 정보를 모두 저장할 공간을 필요로 하며, 최적의 행동 전략을 얻기 위해서 가능한 모든 상태와 행동에 대하여 충분한 반복적 경험을 필요로 한다. 더구나 단일 에이전트 강화학습 환경과 비교해 다중

에이전트 강화학습은 고려해야 할 상태공간이 더욱 커져 저장 공간 문제가 발생할 뿐 아니라 최적의 행동전략에 도달하기 위한 학습 시간이 매우 길어져 효과적인 학습이 불가능하다.

본 논문에서는 대표적인 다중 에이전트 학습 문제의 예로서 먹이와 사냥꾼 문제(Prey and Hunters Problem)를 소개한다. 이 문제는 사냥꾼(hunter)에이전트들 사이에 사전에 주어진 지식 없이 협력적 행위를 통해 도망가는 먹이(prey) 에이전트를 포획하려는 문제이다. 그리고 본 논문에서는 이 문제에서 각 사냥꾼 에이전트의 행동전략을 학습하기 위해 독립적인 강화학습을 적용하려고 한다. 그러나 강화학습의 경우 앞서 설명한 바와 같이 에이전트의 실제 경험 없이는 다음상태와 보상값을 예측할 수 없으며, 최적화된 행동 전략을 학습하기 위해서는 가능한 모든 상태와 행동에 대하여 충분한 반복적인 경험이 필요하다. 따라서 먹이와 사냥꾼 문제와 같은 다중 에이전트 환경에서는 강화학습이 효과적으로 진행되기 어렵다. 이에 본 논문에서는 이와 같은 다중 에이전트 강화학습 문제의 해결책으로 신경망 SOM을 이용한 일반화 방법인 QSOM을 제안한다. 이 방법은 일반화를 통해 부분적인 상태 및 행동 쌍에 대한 경험을 바탕으로 가능한 모든 상태 및 행동 쌍에 대한 Q값을 유도할 수 있다. 그리고 이 방법은 기존의 일반화 방법과는 달리 군집화 기능을 제공하는 SOM을 이용함으로써 명확한 다수의 훈련 예가 없어도 이전에 경험하지 못했던 상태-행동들에 대한 Q값을 효과적으로 예측하고 이용할 수 있다는 장점이 있다. 본 논문에서는 실험을 통해 본 논문에서 제시한 QSOM 학습법의 일반화 효과와 성능을 평가한다. 이를 위해 임의전략, 순수 Q-학습법 등과 비교 평가를 시도한다

2. 관련연구

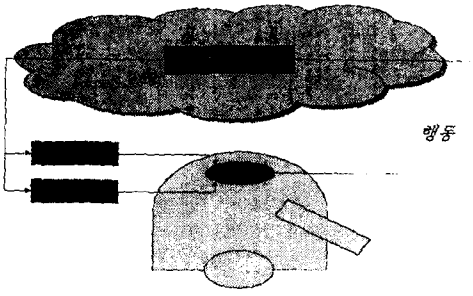
2.1 강화학습

기계학습(machine learning)의 일반적인 목적은 스스로 필요한 지식이나 행동전략을 학습할 수 있는 지능적인 프로그램을 만드는 것이다[14]. 강화학습(reinforcement learning)은 이러한 기계학습 방법의 하나로서, 에이전트가 환경과의 상호작용을 통해 스스로 최적의 행동 전략을 학습하는 방법이다[9,11,17]. 강화학습에서 에이전트는 [그림 1]과 같이 자신이 놓여 있는 환경의 상태를 감지하고 가장 적합하다고 판단되는 행동을 수행하며, 이에 대해 환경은 적절한 보상(reward)이나 벌(penalty)을 제공함과 동시에 새로운 상태로 변화한다. 따라서 에이전트와 환경과의 상호작용은 아래와 같이 정리할 수 있다.

① 에이전트는 환경의 현재상태를 감지한다.
② 에이전트 내부의 행동전략에 따라 적합한 행동을 결정한다.
③ 에이전트는 결정된 행동을 수행한다.
④ 그 결과 에이전트는 환경으로부터 보상을 받는다.
⑤ 환경은 새로운 상태에 놓여진다.
⑥ ①에서 ⑤까지의 과정이 계속 반복된다.

강화학습 에이전트는 이와 같은 환경과의 상호작용 중에 언제나 누적 보상이 최대가 될 수 있는 행동을 선택함으로써 최적의 행동전략에 이를 수 있다. 따라서 강화학습에서 최적의 행동 전략은 [식 1]과 [식 2]와 같이 누적 보상 값을 최대화하는 행동전략을 말한다.

[그림 1] 환경과 에이전트의 상호작용



$$V^{\pi}(s_t) = \gamma^0 r_t + \gamma^1 r_{t+1} + \gamma^2 r_{t+2} + \ldots \qquad [식\ 1]$$

$$\pi^* = \arg\max_{\pi} V^{\pi}(s), (\forall s) \qquad [식\ 2]$$

다음은 강화학습의 특징과 응용분야에 대해 살펴본다. 강화학습은 비 교사학습에 해당된다. 교사학습은 명확한 훈련 예들로부터 사전학습을 하게 된다. 예를 들어 교사학습은 정답이 있는 예상 시험 문제들을 미리 풀어본 뒤 실제 시험을 치르는 수험생의 경우와 같다. 그와 반대로 비교사 학습은 곡예사가 볼을 다루는 기술을 학습하는 것에 비유할 수 있다. 곡예사는 볼을 던지고 받고 하는 일을 반복하면서 볼을 공중에서 유지시킬 수 있는 기술을 학습케 된다. 강화학습 에이전트는 환경이 어떤 상태일 때 어떤 행동을 취하는 것이 최선인지 사전에 주어진 명확한 훈련 예가 없고 단지 실제 수행한 행동에 대한 뒤늦은 보상만 존재할 뿐이다. 따라서 강화학습은 비교사 학습에 해당된다. 그뿐만 아니라 강화학습은 에이전트가 환경에 놓여 실제 행동해봄으로써 비로소 학습할 수 있는 온라인 학습(online learning) 방법이다.

강화학습이 효과적으로 진행되기 위해서는 새로운 행동을 시도해보는 탐색(exploration)과 기존의 경험을 바탕으로 최선의 행동을 선택하는 이용(exploitation)이 균형적으로 수행되어야 한다. 만약 에이전트가 탐색없이 기존 경험에 대한 이용만을 계속한다면 경험해보지 못한 상태와 행동에 대한 새로운 정보를 환경으로부터 얻을 수 없게 되어 학습이 더 이상 진전될 수 없다. 이와는 반대로, 에이전트가 낭비적인 탐색만을 계속한다면 실제 환경이 요구하는 성능의 행위를 보여줄 수 없게 된다. 따

라서 탐색과 이용이 균형을 이루지 못하면 성공적인 학습이 될 수 없다. 강화학습은 다양한 분야에 적용될 수 있다. 하지만 온라인 학습의 한 유형인 강화학습은 그 동안 에이전트가 실제 상황에 놓여 행동할 때만 학습이 가능한 컴퓨터 게임 분야, 로봇 및 자동제어 분야 등에 특히 많이 적용되어 왔다.

2.2 Q학습

Q학습은 앞에서 설명한 강화학습 방법 중 가장 널리 이용되는 대표적인 학습 방법이다[9,21]. Q학습은 상태-행동 쌍에 대한 기대응답(expected response)대한 평가함수를 구하는데 사용된다. 그리고 상태-행동에 대한 평가함수 값을 구하는 방법은 동적 프로그래밍(dynamic programming) 방식을 사용한다. Q학습의 가장 큰 특징은 환경 모델, 즉 보상함수 및 상태 결정함수가 필요 없는 강화학습이라는 점이다. Q학습에서는 상태-행동에 대한 평가함수 Q(s, a) 값을 예측하고 이 Q값을 기초로 행동을 결정한다. 그러나 사전에 정확한 평가함수 Q(s, a)값을 알 수 없기 때문에 에이전트는 경험을 통해 점진적으로 최적의 Q함수 값을 찾아 간다. 최적의 Q함수 값에 수렴하기 위해서는 가능한 모든 상태와 행동에 대한 충분한 반복경험이 필요하다. 따라서 Q학습과정에서는 모든 상태-행동 쌍에 대한 평가함수 Q(s, a) 값을 저장 운영할 상당히 큰 Q표가 필요하다.

$$a^* \leftarrow \arg\max_{a \in action} Q(s,a) \qquad \text{[식 3]}$$

[식 3]은 Q학습에서 현재 상태 s에서 취할 최적의 행동을 결정하는 식을 나타낸다. 이 식에 따라서 에이전트는 현재 상태에서 가능한 행동 중 Q함수 값이 가장 큰 행동을 최적의 행동으로 선택한다. 그리고 이렇게 선택된 행동을 수행하고 나서 환경으로부터 보상값 r이 주어지면 [식 4]에 의해 기존의 Q(s, a)함수 값이 새롭게 갱신된다.

$$Q(s,a) \leftarrow Q(s,a) + \alpha(r + \gamma \max_{a'} Q(s',a') - Q(s,a)) \qquad \text{[식 4]}$$

<표 1>은 Q학습 알고리즘을 나타낸다. 이 알고리즘에서 는 현재상태, 보상 값, 행동, 다음상태, 학습률, 경감률을 나타낸다. 먼저 임의의 Q값으로 Q표를 초기화 한다. 그리고 매 단계 기존의 Q값을 바탕으로 현재상태에 가장 적합한 행동을 선택한다. 이어서 선택된 행동을 수행하고 그 결과 주어지는 보상값과 새로운 상태 변화를 받아들인다. 기존의 Q값과 보상값을 기초로 Q값을 새롭게 갱신하며 이러한 과정은 계속 반복 된다.

Q학습을 비롯한 기존의 많은 강화학습 방법들은 에이전트가 최적의 행동전략에 수렴되기 위해서는 가능한 모든 상태와 행동들에 대해 충분한 반복경험이 있어야 한다고 요구하고 있다. 하지만 일반적으로 이러한 수렴조건을 만족하기 위해서는 각 상태와 행동에 대한 평가 함수값을 저장 운영하기 위해 매우 큰 저장 공간을 확보하고 있어야 하며, 또한 학습에 매우 긴 시간이 필요하게 된다.

<표 1> Q학습 알고리즘

Initialize Q(s,a) arbitrarily
Repeat
  Stat s
  Repeat
    Choose a from s using policy derived from Q
    Take action a ,observe r , s'
    $Q(s,a) \leftarrow Q(s,a) + \alpha[r + \gamma \max_{a} Q(s',a') - Q(s,a)]$
    $s \leftarrow s'$
  until s is terminal

특히 Q학습에서는 한번의 행동 경험을 통해 특정 상태-행동 쌍(s, a)에 대한 평가 함수 Q(s, a)값이 한번만 갱신된다. 따라서 큰 상태-행동 공간을 갖는 복잡한 학습문제의 경우에는 이전에 경험해보지 못했거나 경험이 충분치 않은 상태-행동들이 많아 효과적인 학습을 기대하기 힘들다. 그리고 특히 다중 에이전트 환경에 적용하였을 경우, 단일 에이전트 환경에 비해서 고려해야 할 상태공간이 더욱 커져서 각 상태들에 대한 정보를 저장, 운영할 수 있는 Q표를 확보하는데 문제가 발생하며, 수렴이 지연되는 문제가 발생한다. 따라서 기존의 연구들에서는 이러한 문제점을 해결하기 위해 모듈화 방법과 일반화 방법 등을 제시하고 있다.

2.3 모듈화 방법

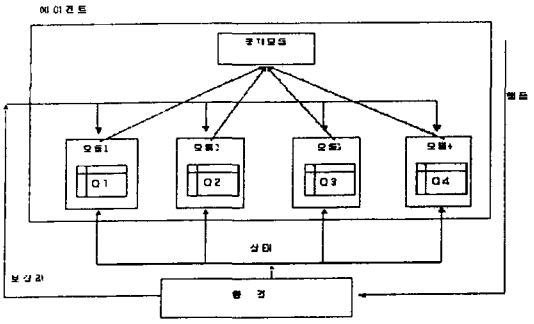
Q학습을 비롯한 대부분의 강화학습에서 문제가 되는 큰 상태-행동 공간문제를 해결하고자 하는 기존의 연구들은 크게 모듈화 방법과 일반화 방법으로 나눌 수 있다. 모듈화 방법에서는 큰 상태-행동 공간을 몇 개의 작은 모듈로 나누어, 모듈별로 별도의 Q학습을 전개한 뒤 각 모듈의 Q학습 결과를 취합하여 행동을 결정하는 방법이다. 모듈화 방법에는 Modular-Q, AMQL 등이 있다.

2.3.1 Modular-Q 학습

Modular-Q 학습은 하나의 큰 학습을 작은 세부 문제들로 분해하고 각 세부 문제에 대해 별도의 Q학습을 전개함으로써 Q학습의 큰 상태공간 문제를 해결해 보려는 학습 방법이다[16,19].

[그림 2]는 대표적인 모듈화 방법인 Modular-Q 학습법을 표현한 것이다. Modular-Q학습을 전개하는 에이전트는 Q학습을 전개하는 n개의 모듈과 그 결과를 취합 및 결합함으로써 행동을 결정하는 1개의 중재 모듈을 가지고 있다.

[그림 2] Modular-Q 학습법



Modular-Q학습과정은 다음과 같다. 먼저 에이전트의 행동선택 단계에서는 현재 상태에서 가능한 각 행동들에 대한 Q값을 각 모듈들이 제공하면, 중재모듈에서 이 값들을 취합하여 [식 5]과 같이 Q값의 합이 최대가 되는 행동을 선택하고 이것을 실행하게 된다. 그 결과 환경으로부터 보상값이 주어지면 이 보상값은 각 모듈에 입력되어 Q함수값을 새롭게 갱신하는데 사용되어진다.

$$a^* \leftarrow \arg\max_{a \in A} \sum_{i=1}^{n} Q_i(s,a) \qquad \text{[식 5]}$$

2.3.2 AMQL 학습

AMQL 학습은 환경변화에 맞추어 에이전트 스스로가 적합한 모듈들의 집합을 만들어 가는 구조인 자동화된 모듈화 학습법(Automatic Modular Q-Learning, AMQL)이다[16,19]. 이 방법은 다음과 같은 3단계를 과정을 통해 학습을 진행한다.

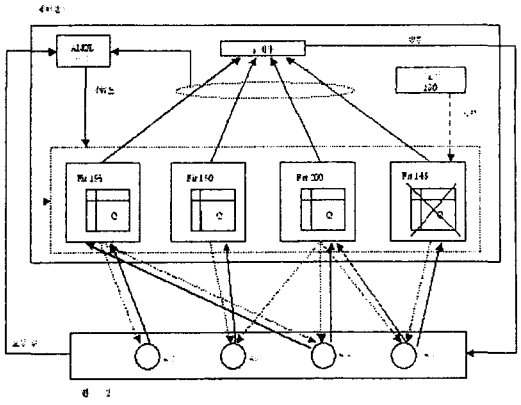
① 요소 선택단계
현재환경 요소들을 임의로 선택하여 각 모듈마다 선택된 요소를 기초로 Q표를 구성한다.

② 적합성 평가단계
각 모듈이 선택한 행동에 따라서 보상 값 및 모듈에 대한 적합성을 평가한다.

③ 모듈 선택단계
평가된 적합성이 기준치(임계치)보다 낮은 모듈들은 삭제하고 새로운 환경요소를 선택하여 이들을 대신할 새로운 모듈을 재 구성한다.

[그림 3]은 AMQL학습을 나타내는 그림으로서, 위에서 설명한 3단계를 통해서 학습이 진행된다.

[그림 3] AMQL 학습법



위에서 열거한 모듈화 방법은 다음과 같이 몇가지 문제점을 가지고 있다. 첫째, 이 방법은 사전에 잘 정의된 모듈들을 필요로 하며, 효과적인 모듈화를 위해서는 많은 영역지식이 필요하다. 둘째, 각 모듈별 학습 결과를 결합하는 중재모듈의 결합방식이 매우 단순하고 고정적이다. 셋째는 결과적으로 모듈화를 통해 각 모듈에서 고려하는 환경요소는 한 두개 정도로 줄어들어 실제 문제의 복잡도에 비해 지나치게 단순화된다는 것이다.

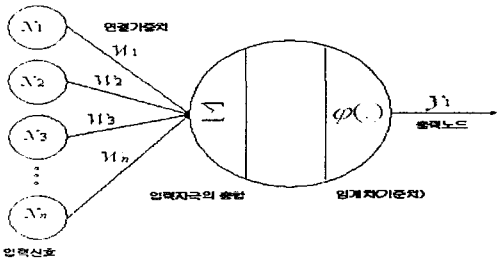
2.4 일반화 방법

일반화(generalization) 방법은 문제공간을 작은 조각으로 나누는 모듈화와는 달리 귀납적 학습(inductive learning) 기법을 적용함으로써 이전에 경험해본 일부 상태-행동 쌍의 Q함수 값을 기초로 나머지 경험하지 못한 상태-행동 쌍에 대한 Q함수 값을 예측하는 방법들을 말한다. 기존의 일반화 방법에는 적용하고자 하는 귀납적 학습기법에 따라 신경망을 이용한 QCON과 SGA학습 그리고 결정트리(decision tree)를 이용한 G-학습 등이 있다 [1,2,6,].

2.4.1 QCON 학습

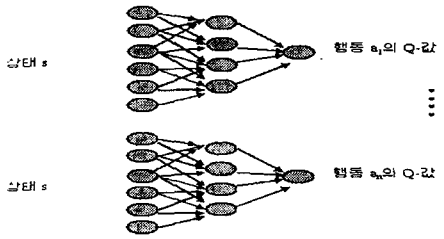
신경망은 인간이나 동물의 뇌의 구조를 모방한 계산 모델로 다수의 뉴런을 집합이 망 구조를 형성하고, 주어진 망을 환경 데이터에 대하여 적응 시키는 학습 알고리즘이다[5]. 신경망은 입력층, 은익층, 출력층의 세 계층으로 이루어져 있으며, 입력과 출력 계층 사이에는 여러 개의 계층이 존재할 수 있다. [그림 4]은 신경망을 구성하는 한 뉴런(neuron) 노드인 퍼셉트론(perceptron)을 나타낸 그림으로 각 입력신호는 서로 다른 연결강도를 지닌 연결선에 의해서 뉴런에 전달되며, 입력 자극의 총합이 일정 수준이 넘으면 출력 노드에서 1 혹은 0을 산출하게 된다.

[그림 4] 퍼셉트론 구조



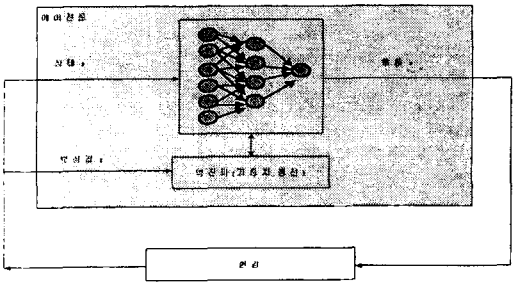
신경망 기술을 이용하면 불완전하고 잡음이 많은 입력의 해석뿐만 아니라 패턴인식, 학습, 분류, 일반화, 추상화 등에 활용할 수 있다. 특히 신경망의 일반화(generalization) 기능을 이용하면 불완전하거나 사전에 알 수 없던 입력에 대해서도 합리적인 출력을 생성할 수 있다. 그리고 뛰어난 적응성(adaptability)을 통해 에이전트간 새로운 환경에 대해 학습하거나 동적 프로그래밍 환경에서 프로그램을 새롭게 갱신하고 유지하는데 사용할 수 있다.

[그림 5] 다층 퍼셉트론



일반화 방법을 적용한 대표적인 강화학습 중 하나인 QCON알고리즘은 [그림 5]과 같이 각 행동마다 상태를 입력으로, Q함수 값을 출력으로 하는 단일 출력의 다층 퍼셉트론(multi-layer perceptron)을 두고 이것을 이용함으로써 경험하지 못한 상태들에 대한 Q함수 값도 예측할 수 있게 하였다[4,5,13].

[그림 6] QCON 학습법



[그림 6]는 전체적인 QCON 학습과정을 나타낸다. 에이전트는 현재 상태에서 취할 행동을 선택하기 위해 현재 상태를 [그림 5]과 같은 각 신경망의 입력으로 제공함으로써 가능한 모든 행동의 Q 함수 값을 예측할 수 있고, 그러면 [식 7]와 같이 이들 중 가장 큰 Q 함수 값을 가지는 행동을 선택한다.

$$a^* = \arg\max_{a \in A} QNET_a(s) \qquad \text{[식 5]}$$

선택된 행동이 실행되고 나서 환경으로부터 새로운 상태와 보상 값 r이 주어지면, 이 보상 값 r은 [식 6]에 의해 새로운 Q함수 값을 구하는데 이용된다. 그리고 이 새로운 Q함수 값과 신경망의 출력으로 제시된 기존의 Q함수 값의 차(difference)를 오차(error)로 삼아 역전파 알고리즘(backpropagation algorithm)을 적용하여 신경망의 가중치를 갱신한다.

$$Q_{a^*} \leftarrow r + \gamma \left(\max_{a \in A} QNET_a(s') \right)$$

$$\Delta Q_{a^*} \leftarrow Q_{a^*} - QNET_{a^*}(s) \qquad \text{[식 6]}$$

그러나 다층 퍼셉트론과 역전파 알고리즘에 기초한 교사학습(supervised learning)을 적용한 QCON은 효과적인 학습을 위해서는 다수의 정확한 훈련 예가 필요하다. 하지만 훈련 예로 사용할 정확한 Q값을 미리 알 수 없으므로 역시 오차를 포함한 추정된 Q값을 실제 훈련에 사용함으로써 오차가 커져 높은 성능을 기대하기 어렵다.

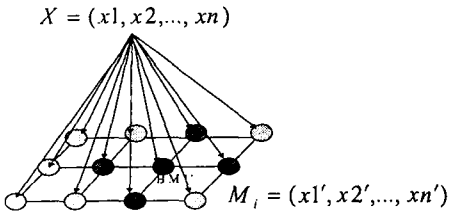
2.5 신경망 SOM

SOM(Self Organizing Map)은 입력 데이터들에 대하여 유사한 것들 끼리 묶어 군집화(clustering)하는데 사용되는 특수한 형태의 신경망이다. SOM은 전체적으로 두 층으로 구성되고, 입력 층을 통해 군집화 하고자 하는 데이터가 들어 오면 입력층과 출력 층 사이에 존재하는 연결강도를 조정함으로써 스스로 군집을 형성하게 된다[7,13].

SOM은 들어오는 입력에 대하여 경쟁을 통해 승리한 출력층 뉴런만이 자신과 자신의 이웃 입력층과의 연결강도를 조정할 수 있도록 하는 Winner-Takes-All 방법을 취하고 있다. 이러한 경쟁을 통하여 점차 신경망은 입력의 구조를 반영하는 자기 조직화 지도(Self Organizing Map)를 완성해 나가게 된다.

자기 조직화 신경망은 군집화 역할을 수행할 뿐만 아니라, 최종적으로 생성된 자기조직 지도 내에 데이터 간의 위상적인 관계까지도 포함함으로써 군집과 군집간의 관계 또한 생성해 준다. 따라서 최종적으로 생성된 지도내의 위상관계를 활용하여 유사한 군집과 비유사 군집을 확인할 수 있다.

[그림 7] 신경망 SOM



[그림 7]은 신경망 SOM의 구조를 나타낸다. 신경망 SOM은 단일 뉴런으로 구성된 입력층과 격자 모양으로 연결된 출력층의 뉴런들로 구성된 신경망으로서, 입력벡터 X들이 주어지면 이들과 가장 잘 매치되는 뉴런(Best Matching Unit, BMU)들에 배정함으로써 유사한 입력들에 대한 군집화를 해준다. SOM에서 각 뉴런은 하나의 클러스터를 나타내며, 뉴런별로 이 클러스터에 속한 구성원들의 속성값에 기초한 가중치 벡터(weight vector)를 유지한다.

각 뉴런의 가중치 벡터와의 비교를 통해 새로운 입력 X가 BMU에 배정이 되면 이 BMU와 이웃한 뉴런들의 가중치는 입력 X쪽에 가깝도록 갱신된다. 이와 같은 SOM의 학습과정은 결국 유사한 입력 데이터들을 몇 개의 클러스터로 군집화함과 동시에 유사한 클러스터들을 인접한 위치에 배치시킴으로써 군집화의 결과를 시각화 하는데 유리한 특징을 가지고 있다.

$$\|X - M_{bmu}\| = \min\{\|X - M_i\|\} \qquad [식\ 7]$$

[식 7]은 BMU(Best Matching Unit)을 계산하는 식으로서, 입력벡터 X와 뉴런의 가중치 벡터 Mi와의 거리가 가장 가까운 뉴런이 BMU로 선택된다.

<표 2> SOM 알고리즘

```
Input : Set of N demansion vector, X
Output : Subsets of input data. ( M subsets )
begin
   Randomly initialize W = ( w1,w2,···,wN ) for each node
   for ( t = 0; unless a stopping condition is reached; Increase t )
      for ( all input data )
         for (i = 0 to M)
            Compute D = | X - W ···|
         endfor
         find the winner j = i such that D( t ) is minimum for over all i
         Updata the winner j ( and its neighbors)
      endfor
   endfor
end
```

<표 2>은 일반적인 SOM학습 알고리즘을 나타내고 있다. 먼저 알고리즘에서 N차원 입력 벡터집합, 입력 데이터에 부분 집합인 출력을 요소로 각 출력 노드를 위한 연결 가중치 벡터들의 초기값을 임의의 값으로 할당한다. 그리고 임의의 연결 가중치를 할당한 후 입력벡터와 유사성을 측정한다. 유사성 측정 방법은 유클리드 거리(Euclidean distance)를 사용한다. 입력벡터와 연결가중치 사이에 유클리드 거리를 구하여 입력벡터와 가장 거리가 작은 연결 가중치를 결정하고, 그에 따른 출력 노드가 결정된다. 결정된 출력 노드의 연결 가중치를 갱신하고, 그 이웃의 노드의 연결 가중치를 갱신하게 된다. 이러한 동작을 반복함으로써 각각의 승자노드의 가중치 벡터는 입력벡터의 방향으로 이동하게 된다. 따라서 각 입력벡터는 자신과 가장 가까이에 있는 가중치 벡터의 출력노드에 해당하는 클러스터에 할당된다.

유사한 입력 데이터들에 대한 군집화 기능을 제공하는 신경망 SOM을 일반화에 이용하는 방법은 간단하다. 서로 다른 상태나 행동을 하나씩 독립적으로 구분하지 않고 유사한 것들끼리 묶어 군집화를 이루고 이 클러스터별로 동일한 출력을 공유하도록 함으로써 일반화를 이룰 수 있다. 이와 같이 신경망 SOM을 일반화에 이용하면 기대할 수 있는 가장 큰 장점은 비교사 학습(unsupervised learning)으로서 정확한 훈련 예가 없어도 높은 성능의 일반화가 가능하다는 점이다. 따라서 본 논문에서는 강화학습을 위한 일반화에 신경망 SOM을 적용한 QSOM 알고리즘을 제안한다.

3. 먹이와 사냥꾼 문제

3.1 문제의 정의

먹이와 사냥꾼 문제(Prey and Hunters Problem)는 m x n 크기의 격자세계(grid world)에서 도망가는 하나의 먹이(prey)와 이를 포위해 잡으려는 다수의 사냥꾼(hunter)들로 이루어진 문제로서, 서로 통신이 불가능한 사냥꾼들이 어떤 행동 전략을 사용하여 보다 효과적인 협조 하에 먹이를 잡을 수 있느냐 하는 다중 에이전트 학습 문제이다 [3,10,12,15,18,20]. 먹이 및 사냥꾼 에이전트는 매 단계마다 동시에 현재 위치에서 5개의 이동 행동, { 동, 서, 남, 북, 제자리 } 중 하나를 선택하여 실행한다. 그리고 먹이 에이전트의 경우 사냥꾼 에이전트들로부터 도망가기 위해 사냥꾼들의 상태 위치로부터 가장 거리가 먼 곳으로 이동하는 행동을 선택하게 된다. 사냥꾼들은 개별적으로 먹이를 잡기 위한 전략 뿐만 아니라 공동 학습 전략을 통해 효율적 포위 전략이 필요하다.

3.2 문제 특징

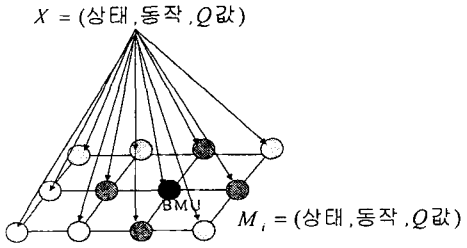
먹이와 사냥꾼 추적 문제는 일반적으로 다음에 특징을 가진다. 첫째로 다수의 동질적(homogeneous) 에이전트가 존재한다. 다수의 사냥꾼 에이전트 및 먹이 에이전트는 기본행동(primitive action) 집합으로 { 동, 서, 남, 북, 제자리 } 가진다. 둘째로 에이전트들은 모두 제한된 인지 능력을 가진다. 에이전트들의 일반적인 격자세계의 크기를 30 x 30라 가정하면 (3, 3)에서 (30,

30)의 제한된 인지범위를 갖는 것으로 가정한다. 셋째로 에이전트 사이에 통신이 불가능하다. 특히 다수의 사냥꾼 에이전트들 사이에 통신이 불가능하다. 넷째로 먹이 에이전트를 포위하기 위한 공동학습 전략이 필요하다. 개별적인 사냥꾼 에이전트의 추적 만으로 먹이 에이전트를 포위하기 불가능하기 때문에 다수에 사냥꾼 에이전트들 간에 포위하기 위한 공동 학습 전략이 필요하다. 마지막으로 커다란 상태 공간을 가진다. 격자세계의 크기를 30 x 30라 가정하면, 하나의 사냥꾼 에이전트가 나타낼 수 있는 상태공간의 크기는 30 x 30이며, 따라서 하나의 먹이 에이전트와 네개의 사냥꾼 에이전트를 포함한 하나의 상태공간(state space)의 크기는 900x900x900x900x900 = 9005 이 된다. 이와 같이 상태공간이 큰 다중 에이전트 학습문제는 단순 Q학습과 같은 단순한 강화학습을 적용할 경우, 제한된 시간 내에 효과적인 행동전략을 학습할 수 없게 되는 단점을 가진다.

4. QSOM 학습법

순수 Q학습이 갖는 큰 상태공간 문제를 해결하기 위해 신경망 SOM을 이용한 확장된 QSOM 학습법을 제시한다. SOM은 신경망의 한 종류로서, 사전에 학습 모델이 필요하지 않고, 입력데이터에 따라 자기 조직화하는 특성을 가진다. 그리고 입력패턴에 따른 위상적 군집화가 가능하다.

[그림 9] QSOM의 입력벡터 및 가중치 벡터

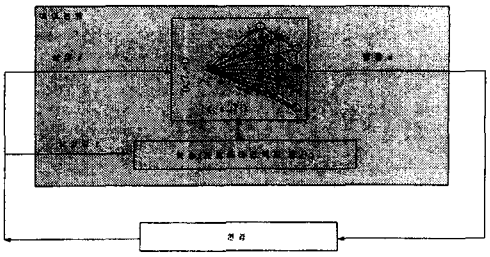


4.1 QSOM 학습법의 정의

[그림 9]는 QSOM 학습에 이용되는 신경망 SOM의 입력벡터와 가중치 벡터들을 나타낸다. 특히 QSOM 학습에서 입력벡터는 , 각 뉴런의 가중치 벡터는 의 3차원 벡터로 표현된다.

QSOM의 전체적인 학습과정은 [그림 10]과 같다. 먼저 현재 상태에서 에이전트가 취할 행동을 결정하기 위해서는 입력벡터 X=(현재상태,■,+1)에 대한 BMU을 선택한 뒤 그 BMU의 가중치벡터 Mi =(상태,동작,Q값)에 기술된 동작에 따라 행동한다.

[그림 10] QSOM 학습법



그 결과 환경으로부터 보상값 r이 주어지면 이번에는 입력벡터 X'=(현재상태, 실행한 동작, ■)에 대한 BMU을 선택하고, 벡터 (현재상태, 실행한 동작, 보상값)를 이용하여 BMU와 이웃 뉴런들의 가중치벡터를 갱신한다.

4.2 QSOM 학습법의 특징

QSOM 학습법은 Q학습의 큰 상태공간 문제를 신경망 SOM을 이용하여 해결하고자 하였다. QSOM 학습법의 큰 특징은 일반화 기능을 가지는 신경망 SOM을 Q표 대신 사용함으로써 실제 경험해보지 못한 상태-행동 쌍에 대한 Q값도 유도할 수 있으며, 이를 행동선택에 바로 이용할 수 있다는 것이다. 이렇게 함으로써 보다 빠른 시간 내에 높은 성능의 행위를 보여줄 수 있을 뿐 아니라 학습시간도 단축할 수 있다. 또한 비 교사 학습기능을 가진 신경망 SOM을 이용함으로써 QCON학습법과는 달리 학습을 위해 상태-행동 쌍에 대한 정확한 Q값을 미리 알아야 할 필요가 없다. 따라서 본 논문에서는 이와 같은 QSOM 학습법을 먹이와 사냥꾼 문제와 같은 다중 에이전트 학습문제에 적용하기를 제안한다.

5. 실험평가

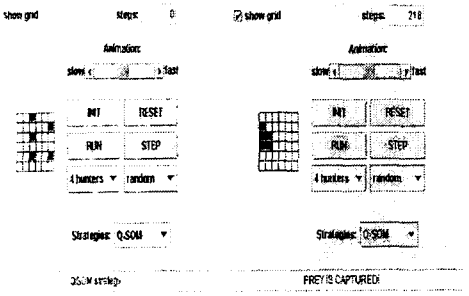
먹이와 사냥꾼 문제 영역의 실험을 통해 본 논문에서 제시한 QSOM 학습방법의 일반화 효과를 평가한다. 그리고 다른 학습법인 순수 Q학습 등과 비교 실험을 통해 QSOM 학습법의 효율을 평가 한다.

5.1 실험방법

실험 방법 및 환경은 다양하게 계획할 수 있으며, 본 논문에서는 아래와 같이 실험환경 및 방법을 결정하여 실험을 하였다.

[그림 11]은 먹이와 사냥꾼 문제의 실험환경으로 6 x 6 크기에 격자세계를 가지고 있으면, 하나의 먹이 에이전트와 네 개의 사냥꾼 에이전트로 구성되어 있다.

[그림 11] 먹이와 사냥꾼 문제의 실험환경



[그림 11]의 왼쪽 그림은 초기상태에서 하나의 먹이와 다수의 사냥꾼 에이전트들의 초기 위치를 오른쪽 그림은 먹이가 사냥꾼들에 포위 당한 상태를 나타내고 있다. 각각 에이전트의 행동으로는 { 동, 서, 남, 북, 제자리 } 다섯 개의 이동 동작을 가지고 있으며, 모든 에이전트는 한번에 한 개의 행동을 선택하여 수행하게 된다.

본 실험에서 먹이 에이전트는 사냥꾼들과의 거리를 계산해서 거리를 넓히기 위한 행동전략을 가지고 행동을 선택한다. 반면에 사냥꾼 에이전트의 경우, 다음과 같은 세 개의 서로 다른 행동전략을 구현하고 이들을 가지고 실험을 전개하였다.

첫 번째 행동 전략은 먹이 에이전트와 다른 사냥꾼 에이전트의 상태를 고려하지 않고 임의의 행동전략(random strategy)을 가진 경우이다.

두 번째 행동 전략은 Q학습 전략(Q Learning strategy)으로 먹이 에이전트뿐만 아니라 다른 사냥꾼 에이전트들의 고려하게 된다. Q학습 전략에서는 Q표를 이용하여 상태-행동 쌍에 따른 평가함수, Q값을 고려하여 행동을 결정한다. Q학습에서 현재 상태는 (먹이위치, 사냥꾼1위치, 사냥꾼2위치, 사냥꾼3위치, 사냥꾼4위치)로 표현된다. 평가함수 Q값 따라 현재 상태에 수행할 행동을 결정한다. 행동을 수행하고 나면 즉각적인 보상과 다음상태가 결정되며, 평가함수 Q값의 갱신에는 현재상태 및 다음상태의 Q값과 보상값이 사용되어 진다.

세 번째 행동 전략으로는 본 논문에서 제시하고 있는 QSOM 학습법을 사용한다. QSOM 학습법에서 현재상태는 (먹이위치, 사냥꾼1위치, 사냥꾼2위치, 사냥꾼3위치, 사냥꾼4위치)로 다섯 개의 복합 상태로 표현되고, 입력벡터는 (상태, 행동, 평가함수 Q값)으로 표현된다. 출력벡터의 경우, 입력벡터를 고려하여 n개 노드의 출력벡터 (상태, 행동, 평가함수 Q값) 가진다. 행동 선택 시에는 입력벡터(현재상태, unknow, +1)에 대응되는 BMU 노드를 결정하고, 선택된 BMU에 기술된 행동을 선택하여 수행한다. 수행한 행동에 대한 보상값이 주어지면 입력벡터 (현재상태, 수행한 행동, 보상값)에 대한 BMU을 찾아내어, 그 BMU과 이웃 노드들의 가중치 벡터를 갱신한다.

실험평가 전에 결정해야 할 학습인자로 Q학습과 QSOM학습에서 사용되는 초기 Q값은 0.0, 학습률 (learning rate)는 0.3과 경감율(discount rate)는 0.9로 정하였다. 그리고 신경망 SOM 크기는 초기 6 x 6 으로 36개의 출력벡터 뉴런으로 정의하였으며, 이동 제한 횟수는 1000이내로 한정 하였다. 그리고 Q학습과 QSOM 학습의 사용되는 보상함수 (reward)는 다음과 같다.

Reward (s) = 10, if 상태 s에서 먹이의 이동 가능한 동작수 = 0

7, if 상태 s에서 먹이의 이동 가능한 동작수 = 1

5, if 상태 s에서 먹이의 이동 가능한 동작수 = 2

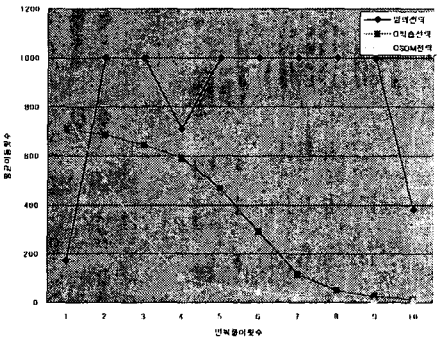
3, if 상태 s에서 먹이의 이동 가능한 동작수 = 3

1, if (현재 먹이까지 거리) - (과거 먹이까지 거리) < 0

0, if 나머지 모든 상태

마지막으로 실험은 서로 다른 먹이와 사냥꾼 문제를 20개를 생성 후, 한가지 동일한 문제에 대하여 동일한 전략으로 연속해서 10회 반복 시행하였다. 그리고 매번 먹이가 포획될 때까지 소요된 평균 이동 횟수와 평균 Q값 변동을 측정하였다.

[그림 12] 평균 이동 횟수 비교

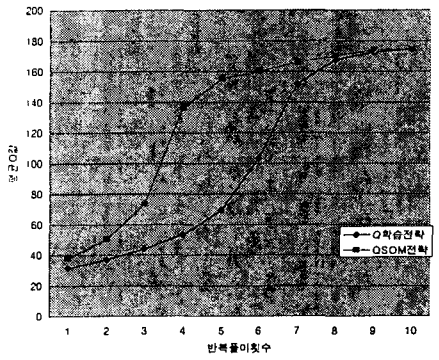


5.2 실험결과

[그림 12]는 먹이와 사냥꾼 문제의 평균 이동 횟수를 각 행동전략 별로 나타낸 그림이다. [그림 12]에서 X축은 반복풀이 횟수를 Y축은 평균이동 횟수를 나타내고 있다. 그리고 비교평가를 통해 첫 번째 임의전략의 경우, 이동 제한 횟수 1000을 넘는 경우가 대부분으로 사냥꾼 에이전트가 먹이 에이전트를 포획하지 못하는 경우에 해당한다. 두 번째로 Q학습전략의 경우, 일정 반복횟수가 되면 평균이동 횟수가 일정한 값을 유지하는 것을 알 수 있다. 세 번째로 QSOM학습전략의 경우, Q학습 전략과 같이 일정 반복횟수에서 평균이동 횟수가 일정한 값을 유지하였다. 그러나 Q학습 전략과 QSOM학습 전략을 비교할 때 QSOM학습 전략은 반복 횟수 초기인 3~4 사이에 평균 이동 횟수가 일정한 값을 유지한 반면에 Q학습 전략은 반복 횟

수 후반인 6~7 사이에서 평균 이동 횟수가 일정한 값을 유지하였다. 이것은 QSOM학습 전략을 사용한 사냥꾼에이전트가 Q학습 전략을 사용한 사냥꾼 에이전트 보다 더 빨리 먹이를 포획하는 결과를 보인 것이다. 따라서 학습이 진행됨에 따라 일반적으로 QSOM학습 전략이 Q학습 전략 보다 성능 속도가 더 빠르고, 더 높은 성능을 나타낸 것을 보여준다.

[그림 13] 평균 Q값 비교



[그림 13]는 먹이와 사냥꾼 문제에 대한 Q학습 전략과 QSOM학습 전략 간의 평균 Q값을 비교한 그림이다. [그림 13]에서 X축은 반복풀이 횟수를 Y축은 평균 Q값을 나타내고 있다. 임의전략을 제외한 Q학습 전략과 QSOM학습 전략의 평균Q값을 비교해 보면, Q학습 전략은 반복풀이 횟수가 6~7 경우에 평균Q값이 일정한 Q값에 도달한다. QSOM학습전략은 반복풀이 횟수가 3~4 경우에 일정한 Q값에 도달한다. 따라서 QSOM학습전략이 Q학습전략에 비해서 더 빨리 최적의 Q값에 수렴한다는 것을 알 수 있다.

따라서 먹이와 사냥꾼 문제 실험을 통해 본 논문에서 제안하고 있는 QSOM학습 방법을 기존에 Q학습 방법을 적용하였을 때 보다 효율적 학습이 가능하다는 가정을 만족할 수 있었다.

6. 결론

지금까지 본 논문에서는 다중 에이전트 환경에서 에이전트간의 협조를 위한 행동전략 학습방법으로 강화학습 방법을 소개하였다. 그리고 강화학습의 대표적인 학습방법 중 하나인 Q학습의 특징과 문제점에 대하여 살펴보았다. 특히 Q학습의 큰 상태-행동공간을 갖는 문제를 극복하기 위한 방법으로 모듈화 방법과 일반화 방법들에 대하여 살펴보았다.

그리고 본 논문에서는 새로운 일반화 방법으로 신경망 SOM을 이용한 QSOM학습 방법을 제안하였다. QSOM학습 방법은 Q표 대신 신경망 SOM을 사용하며, 비 교사 학습으로 정확한 훈련 예 없이도 효과적인 학습이 가능하다는 특징이 있다. 또 유사 상태들에 대한 군집화가 가능하여 군집별로 동일한 상태평가와 행동을 공유할 수 있다.

본 논문에서는 큰 상태공간을 가지는 대표적인 다중 에이전트 학습 문제인 먹이와 사냥꾼 문제의 실험을 통해, 본 논문에서 제안한 QSOM학습방법의 일반화 효과를 입증하였다.

참고문헌

[1] Abul, O., Polat F. and Alhajj R., "Multi-Agent Reinforcement Learning Using Function Approximation," *IEEE Transaction on Systems, Man and Cybernetics*, Vol.30,No.4(2000), 485-497

[2] Abul, O., Polat F. and Alhajj R., "Function Approximation Based Multi-Agent Reinforcement Learning," *Proceedings of IEEE International Conference on Tools with Artificial Intelligence (ICTAI-2000)*, 36-40.

[3] Arai, S., Sycara, K., and Payne, T., "Experience-based Reinforcement Learning to Acquire Effective Behavior in a Multi-agent Domain", *Proceedings of the 6th Pacific Rim International Conference on Artificial Intelligence, Lecture Notes in Artificial Intelligence*, Springer-Verlag, 125-135, 2000.

[4] C. TOUZET, Norbert Giambiasi, Samira Sehad., " Neural Reinforcement Learning for Behavior Synthesis", *Proceedings of the International Conference Computational Engineering in Sytems Applications*(1996), 9-12.

[5] C. TOUZET, "Neural Networks and Q-learning for Robotics," *Proceedings of IJCNN'99 Tutorials*, 1999.

[6] Hajime KIMURA, Shigenobu KOBAYSHI, "Reinforcement Learning for Continuous Action using Stochastic Gradient Ascent", *Proceedings of The 5th International Conference on Intelligent Autonomous Systems,(IAS-5)*, 288-295, 1998.

[7] Helga Ritter, Thomas Martinets, and Klaus Schulten, "Neural Computation and Self-Organizing Maps", *Addison-Wesley*, 1992.

[8] Jacques Ferber, "Multi-Agent Systems - An Introduction to Distributed Artificial Intelligence", *Addison-Wesley*, 1999.

[9] Kaelbling, L. P., Littman, M. L., Moore, A. W., "Reinforcement Learning : a Survey.", *Journal of Artificial Intelligence Research*, 237-285, 1996.

[10] Kuter, U. and Polat F., "Learning Better in Dynamic, Partially Observable Environments," *Proceedings of the Fourteenth Biennial European Conference on Aritificial Intelligence (ECAI), Workshop on Modeling Artificial Societies and Hybrid Organizations*, 50-68, 2000.

[11] Leslie Pack Kaelbling, Michael L. Littman, Andrew W. Moore. "Reinforcement Learning", *Journal of Artificial Intelligence Research*, Vol. 4(1996).

[12] Littman, M. L., Cassandra, A. R., Kaelbing, L. P., "Learning policies for partially observable environments : Scaling up.", Readings in Agents, Morgan Kaufman (eds. M.N. Huhns, M. P. Singh). 495-503, 1995.

[13] Long-Ji Lin. Reinforcement Learning for Robots Using Neural Networks. *PhD thesis, Carnegie Mellon University,* Pittsburgh, PA, 1993.

[14] Mitchell, T. M., "Machine Learning", *McGraw-Hill,* 1997.

[15] Michael L. Littman., "Markov game as a framework for multi agent reinforcement learning.", *Proceeding of the Eleventh International Conference on Machine Learning*, 157-163, 1994.

[16] N. Ono, K. Fukumoto "A Modular Approach to Multi-agent Reinforcement Learning.", *Distributed Artificial Intelligence Meets Machine Learning: Learning in Multi-agent Environments,* Springer-Verlag, 25-39, 1997.

[17] Rishard S. Sutton. "Reinforcement Learning", *MIT Press,* Cambridge, *MA,* 1998.

[18] Sen, S., Sekaran, M., Hale, J., "Learning to coordinate without sharing information", *Readings in Agents,* Morgan Kaufman (eds. M.N. Huhns, M. P. Singh), 509-514, 1994.

[19] Takayuki Kohri, Kei Matsubayashi, Mario Tokoro, "An Adaptive Architecure for Modular Q-Learning", *Proceedings of 5th International Joint Conference on Artificial Intelligence,* 1997.

[20] Tan, M., "Multi-Agent Reinforcement Learning : Independent vs. Cooperative Agents", *Readings in Agent,* Morgan Kaufman (eds. M. N. Huhns, M. P. Singh), 487-494, 1993.

[21] Watkins, C. J. C. H., Dayan, P., "Technical Note: Q-Learning Machine Learning", 279-292, 1992.