Hidden Markov Modeling of Multi-agent Systems and Its Learning Method

Itsuki Noda[1,2]

[1] Cyber Assist Reseach Center National Institute
of Advanced Industrial Science and Technology
[2] PRESTO, Japan Science and Technology Corporation (JST)

Abstract. Two frameworks of hidden Markov modeling for multi-agent systems and its learning procedure are proposed. Although a couple of variations of HMMs have been proposed to model agents and their interactions, these models have not handled changes of environments, so that it is hard to simulate behaviors of agents that act in dynamic environments like soccer. The proposed frameworks enables HMMs to represent environments directly inside of state transitions. I first propose a model that handles the dynamics of the environments in the same state transition of the agent itself. In this model, the derived learning procedure can segment the environments according to the tasks and behaviors the agent is performing. I also investigate a more structured model in which the dynamics of the environments and agents are treated as separated state transitions and coupled each other. For this model, in order to reduce the number of parameters, I introduce "symmetricity" among agents. Furthermore, I discuss relation between reducing dependency in transitions and assumption of cooperative behaviors among multiple agents.

1 Introduction

When we train an agent or a team of agents (learners) to imitate behaviors of another agent or team (demonstrators), we must determine a framework to model agents or teams. Hidden Markov Model (HMM) is a popular candidate for this purpose. Because the behaviors of intelligent agents are complicated and structured, however, we should apply HMM carefully.

Suppose that we write a code of a reactive soccer agent by hands. We may write the following code for it:

```
while(true) {
  if (⟨is my role a passer ?⟩) {
    if (⟨is a receiver near ?⟩) {
      ⟨kick the ball to the receiver !⟩; ...
      ⟨change my role to receiver !⟩; }
    else if (⟨found dribbling course ?⟩...
  }
  else if (⟨is my role a receiver ?⟩) {
    if (⟨find an open space ?⟩) {
      ⟨move to the space !⟩ }... }
  else ... }
```

G.A. Kaminka, P.U. Lima, and R. Rojas (Eds.): RoboCup 2002, LNAI 2752, pp. 94–110, 2003.

As shown in this example, situations (combinations of states of the environment and the agent) are segmented into finite states like "*is a receiver near?*" and "*found dribbling course?*". These segmentations are vary according to agent's role or intention. In this example, "*is a receiver near?*" is not used when the agent's role is "*receiver*". In the context of the imitation learning, it is important to estimate what kind of segmentation of environment a demonstrator is using. The difficulty of the segmentation of environment is that the segmentation should change according to agent's current intentions that are usually invisible. This means that segmentation of environment should be acquired in the same time of learning intentions in the context of imitation learning.

In addition to it, when agents interact with each other, it is necessary to assume a kind of structure of states in HMM. In the above example, whole situations are classified into states based on roles of agents ("*passer*" and "*receiver*") and conditions of the environment ("*is a receiver near?*" and so on). Because it is difficult to acquire such structure through learning, it is better to use HMM in which states are structured suitably. In this case, we must pay attention the learning performance and reasonabilities of the structure.

In this article, I propose frameworks of HMM that can segment environment interaction between agents effectively through learning. In the following sections, I introduce a integrated HMM of agents and environment for learning of segmentation of environment in Section 2, and framework of HMM to represent interaction of agents in Section 3.

2 HMM for Agents in Dynamic Environment

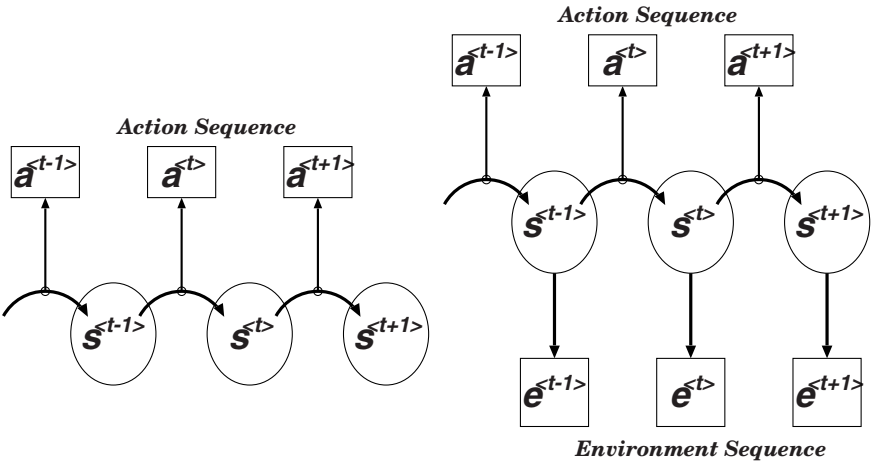
2.1 Agent Model

In general, an autonomous agent is modeled as a Mealy-type HMM (shown in Figure 1(a)), in which the agent's behaviors are decided by the following manner:

- The agent has finite internal states.
- The internal state is transfered in a discrete time step.
- The next state ($s^{\langle t+1 \rangle}$) is determined only by the previous state ($s^{\langle t \rangle}$).
- The agent' action ($a^{\langle t+1 \rangle}$) is selected by the current state transition ($s^{\langle t \rangle} \rightarrow s^{\langle t+1 \rangle}$).

Moreover, in order to represent agent's behavior in a dynamic environment, we need take effect of interaction between the agent and the environment in account. Strait forward implementation of the effect is using input-output-type HMM [BF95,JGS97a], in which the data from the environments are treated as input for the HMM. However, using the such HMM has the following drawbacks:

- If an HMM handles the environment as input, the HMM may not include world model by which agent can predict changes of the environments.
- When the data from the environment consists of continuous values, we need additional mechanisms to handle the continuous values to bridge them to HMM's behaviors.



(a) Agent Model by Mealy-type HMM. (b) Agent Model by Moore-Mealy-type HMM.

Fig. 1. Agent HMM.

In order to overcome these drawbacks, we introduce the following assumption:

– The internal state and the environment has a probabilistic relation.

This means that the environment ($e^{\langle t \rangle}$) can be determined by the internal state ($s^{\langle t \rangle}$) under the probabilistic relation ($Pr(e^{\langle t \rangle}|s^{\langle t \rangle})$). In other words, the changes of the environment can be handled as a Moore-type HMM (Figure 1(b)).

In summary, an agent and its environment can be defined as a combination of Moore- and Mealy-type HMM (MM-HMM) as follows:

$$\text{Agent} = \langle \boldsymbol{S}, \boldsymbol{A}, \boldsymbol{E}, \boldsymbol{P}, \boldsymbol{Q}, \boldsymbol{R}, \boldsymbol{\pi} \rangle ,$$

where $\boldsymbol{S} = \{s_i\}$ is a set of internal states, $\boldsymbol{A} = \{a_i\}$ is a set of action symbols, $\boldsymbol{E} = \{e_i\}$ is a set of environment symbols, $\boldsymbol{P} = \{p_{ij} = Pr(j^{\langle t+1 \rangle}|i^{\langle t \rangle})|i, j \in \boldsymbol{S}, \forall t\}$ is a probability matrix of state transitions, $\boldsymbol{Q} = \{q_{ij}(a) = Pr(a^{\langle t+1 \rangle}|i^{\langle t \rangle}, j^{\langle t+1 \rangle})|i, j \in \boldsymbol{S}, a \in \boldsymbol{A}, \forall t\}$ is a set of probability functions of actions for each transition, $\boldsymbol{R} = \{r_i(e) = Pr(e^{\langle t \rangle}|i^{\langle t \rangle})|i \in \boldsymbol{S}, e \in \boldsymbol{E}, \forall t\}$ is a set of probability functions of environment for each state, $\boldsymbol{\pi} = \{\pi_i = Pr(i^{\langle 0 \rangle})\}$ is a set of probability functions of initial states, and $\langle t \rangle$ on the right shoulder of each variable indicates time t.

2.2 Learning Algorithm

Suppose that a learner can observe a sequence of demonstrator's actions $\{a^{\langle 1 \rangle} \ldots a^{\langle T \rangle}\}$ and changes of an environment $\{e^{\langle 0 \rangle} \ldots e^{\langle T \rangle}\}$. The purpose of the learner is estimate an HMM that can explain the given action and environment sequences most likely. We can derive a learning procedure based on the combination of Baum-Welch algorithms for Moore-type and Mealy-type HMM as follows:

For given a set of sequences $\langle \{a^{\langle 1\rangle} \dots a^{\langle T\rangle}\}, \{e^{\langle 0\rangle} \dots e^{\langle T\rangle}\}\rangle$, the forward ($\alpha^{\langle t\rangle}(j)$) and backward ($beta^{\langle t\rangle}(i)$) probabilities are given by the following recursive formulas.

$$\alpha^{\langle t\rangle}(j) = \begin{cases} \pi_j r_j(e^{\langle 0\rangle}) & ; t = 0 \\ \sum_{i \in \boldsymbol{S}} \alpha^{\langle t-1\rangle}(i) p_{ij} q_{ij}(a^{\langle t\rangle}) r_j(e^{\langle t\rangle}) & ; \textit{otherwise} \end{cases}$$

$$\beta^{\langle t\rangle}(i) = \begin{cases} 1 & ; t = 0 \\ \sum_{j \in \boldsymbol{S}} p_{ij} q_{ij}(a^{\langle t+1\rangle}) r_j(e^{\langle t+1\rangle}) \beta^{\langle t+1\rangle}(j) & ; \textit{otherwise} \end{cases}$$

Using these probabilities, p_{ij}, $q_{ij}(a)$, $r_i(e)$ and π_i are adjusted by the following formulas:

$$p_{ij} \leftarrow \frac{\sum_t \xi^{\langle t\rangle}(i,j)}{\sum_t \gamma^{\langle t-1\rangle}(i)} \qquad q_{ij}(a) \leftarrow \frac{\sum_{t|a^{\langle t\rangle}=a} \xi^{\langle t\rangle}(i,j)}{\sum_t \xi^{\langle t-1\rangle}(i,j)}$$

$$r_i(e) \leftarrow \frac{\sum_{t|e^{\langle t\rangle}=e} \gamma^{\langle t\rangle}(i)}{\sum_t \gamma^{\langle t\rangle}(i)} \qquad \pi_i \leftarrow \gamma^{\langle 0\rangle}(i)$$

where

$$\xi^{\langle t\rangle}(i,j) = \frac{\alpha^{\langle t-1\rangle}(i) p_{ij} q_{ij}(a^{\langle t\rangle}) r_j(e^{\langle t\rangle}) \beta^{\langle t\rangle}(j)}{P(\boldsymbol{a}^{\langle *\rangle}, \boldsymbol{e}^{\langle *\rangle} | \text{Agent})}$$

$$\gamma^{\langle t\rangle}(j) = \frac{\alpha^{\langle t\rangle}(j) \beta^{\langle t\rangle}(j)}{P(\boldsymbol{a}^{\langle *\rangle}, \boldsymbol{e}^{\langle *\rangle} | \text{Agent})}$$

2.3 Segmentation of Environments

When the above learning succeeds, the learner gets a suitable HMM, each of whose states corresponds a combination of internal intentions of the demonstrator and fragments of the environment. In other words, the representation of the intention and the segmentation are mixed in a set of states. While such representation is enough to imitate demonstrator's behavior, it will be still useful to know how the HMM segments the environment.

In the MM-HMM context, the segmentation of the environment means how each data of the environments is mapped to the states of the HMM. Therefore, the segmentation is represented by a probability function $Pr(s|e)$ where e is an environment data and s is an internal state of HMM. This probability can be calculated by the following equation:

$$Pr(s|e) = \frac{Pr(e|s)Pr(s)}{Pr(e)} = \frac{r_s(e)Pr(s)}{Pr(e)} \tag{1}$$

Using this equation, we can illustrate the mapping from the environments to the states.

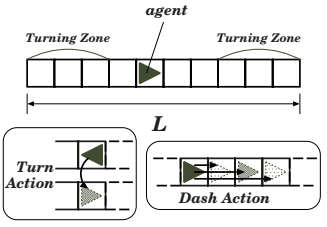


Fig. 2. Setting of Ex.1: Discrete World.

2.4 Experimental Result

In order to show the performance of the proposed model to acquire segmentation of environments, I conducted the following experiments.

Ex.1: Discrete World. In the first experiment, we use a linear block world shown in Figure 2. An agent is moving in the world using *dash* and *turn* actions. When the agent *turn*s, the agent's direction flips between left and right. When the agent *dash*es, the agent's position moves forward. The step size of a dash action varies 1 to 3 randomly.

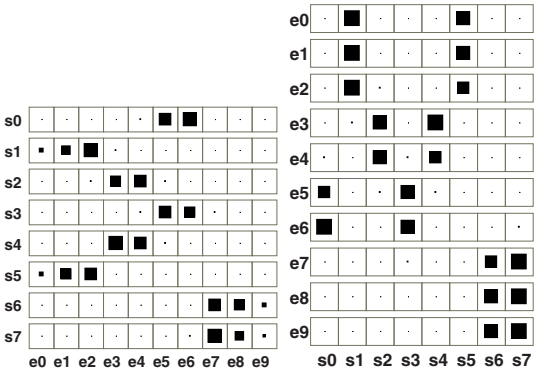
The task of a learner is to acquire the rule of another agent (demonstrator) who behaves as described below. A learner can observe the demonstrator's position (= environment, $\{e^{\langle t \rangle}\}$) and action ($\{a^{\langle t \rangle}\}$) in each time t. We suppose that the demonstrator behaves according to the following rules:

- If the position is in the left(or right) turning zone (margin of the zone is 3) and the direction is left (or right), then *turn*.
- Otherwise, *dash*.

The main point of this experiment is that whether the learner can acquire the correct segmentation by which the turning zone will be represented explicitly in the state transition, because the concept of the turning zone is unobservable to the learner. Also, estimation of the dash step size is also important, because it defines how the world (environment) should be segmented in order to simulate it by an HMM. Note that demonstrator's direction is not observable. This means that the learner needs to acquire the representation of the direction through the learning.

In the experiments, we executed the above learning procedure using different initial parameters 100 times, calculated the average of the likelihood of given example sequences, and select the best one as the result. This is because the learning of general HMMs is not guaranteed to reach the global optimal solution: the adaptation may fall down to a local optimum.

Figure 3 shows the result of the experiment. In this figure, (a) shows possibilities of environment symbols ($e_0 \ldots e_9$) for each state ($s_0 \ldots s_7$). Each box corresponds to $Pr(e_i|s_j)$ whose value is denoted the size of black area in the box. For example, in the s_0 line of boxes, columns of e_5 and e_6 have significant values



(a) $Pr(e|s)$ (b) $Pr(s|e)$

Fig. 3. Result of Ex.1: Discrete World (L=10).

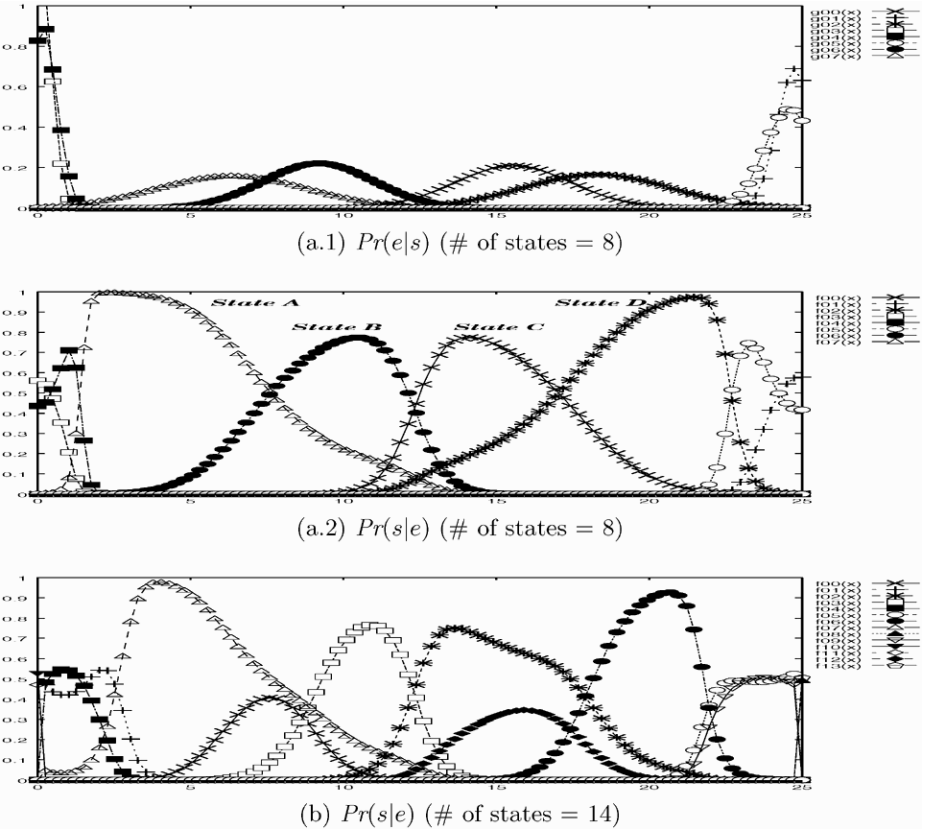
and both values are relatively equal. This can be interpreted that e_5 and e_6 are grouped in the same state s_0, because the environment is estimated e_5 or e_6 equally when the HMM is in state s_0. Similarly, Figure 3-(b) shows possibilities of states for each environment symbol calculated by Eq. 1, whose value $Pr(s_j|e_i)$ is denoted black area of each box. For example, the e_0 line has two columns of s_1 and s_5 who have relatively the same significant possibilities. This can be interpreted that e_0 can correspond to two states, s_1 and s_5, equally.

From these figures, we found that the acquired HMM segments the environment into 4 regions, $\{e_0, e_1, e_2\}$, $\{e_7, e_8, e_9\}$, $\{e_3, e_4\}$, and $\{e_5, e_6\}$. The first two regions correspond the "turning zone" defined inside of the demonstrator. Rest of the two regions have the same length, 2. This value correspond the average step size of dash commands. These results means that the acquired HMM represents both of the rules of agent behavior and dynamics of the environment. In addition to it, each environment symbol corresponds two states. This means that the HMM recognizes the (unobservable) direction of the demonstrator by doubled states for each environment.

Ex.2: Continuous World. The second experiment is a continuous version of the previous experiment, in which the demonstrator's position (e) is a continuous value instead of a discrete symbol. The rule of the demonstrator's behavior is the same as the previous one. The length of the world L is 25.0 and step size of a dash varies 5.0 to 15.0 randomly.

Figure 4 shows the acquired probabilities, $Pr(e|s)$ and $Pr(s|e)$. In these graphs, the probabilities are plotted as follows:

- $Pr(e|s)$ is plotted as a set of probability density functions of environment value, $g_s(e) = Pr(e|s)$, for each state s.
- $Pr(s|e)$ is plotted as a set of changes of probability of each state, $f_s(e) = Pr(s|e)$, according to environment value.



(a.1) $Pr(e|s)$ (# of states = 8)

(a.2) $Pr(s|e)$ (# of states = 8)

(b) $Pr(s|e)$ (# of states = 14)

Fig. 4. Result of Ex.2.

In the figure, (a.1) and (a.2) show the result when the number of HMM's states is 8, and (b) shows the result in the case of 14. From (a.1) and (a.2), we can find that the HMM segments the environment in the similar way as the discrete case.

- Two turning zones at the both ends of the world are segmented clearly.
- There are two corresponding states for the most of the environment value. This means that the HMM represents the direction of movement.

We can also find an additional features from these graphs: There are 4 peaks (State A–D in (a.2)) in the middle of the environment in the graph, and, A and B (or C and D) are relatively overlapped with each other. As same as the first experience, this means that the two states A and B (or C and D) indicate difference of the direction of the movement. On the other hand, while two states share the same part of the environment in the first experiment, peaks of A and B (or C and D) are shifted. As a result, the segment point of the environment in each direction are different. For example, the segment point[1] is at about 13.0 in

[1] A crossing point of probabilities of two states in the graph of Figure 4-(a.2).

the case of the rightward transition (B →D), and it is at about 11.0 in the case of leftward (C →A)[2] This means that the segmentation of the environment is not fixed for every agent's internal state, but varies depend on them. The structure of the segmentation are stable when we use more states in the learning. For example, when we use 14 states to learn the same task, the HMM can acquire the similar segmentation of the environment (Figure 4-(b)).

In order to show that the proposed method can acquire segmentation of environment flexibly, we conducted the following experiment. We introduce an additional *half turning zone* in the middle of the world, where the demonstrator turns in the probability 0.5 . The detailed rule in this turning zone is as follows:

– If the demonstrator faces rightward (leftward)and the position is in the left (right) hand side of the *half turning zone*, then turn in the probability 0.5.

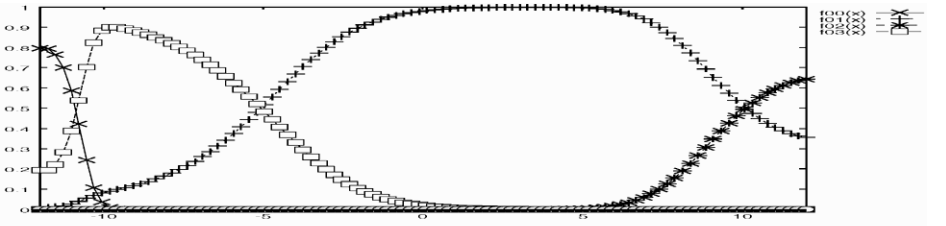
Figure 5 shows how the segmentation of the environment ($Pr(s|e)$) changes according to the various numbers of states. As shown in this result, many states are used to represent turning zones, especially the half turning zone (the middle area of the world). We can see when the number of state increases, the HMM assigns many states to segment the half turning zone. This is because that the conditions to decide demonstrator's behaviors are complicated so that the HMM needs to represent detailed information about the environment.

2.5 Discussion: Environment as Output

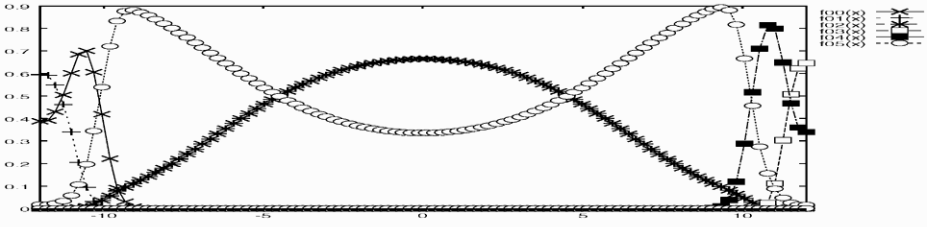
The proposed method looks little bit strange because it handles environment as output from states rather than as input to state transitions. As mentioned above, input-output HMMs seems more reasonable to model relations between agents and environments, in which the environment is treated as input to state-transitions [BF95,JGS97a]. There are the following different points between these two methods:

– When we handle the environment as input, we can apply the HMM for planning. Suppose that initial and goal situations of environment ($e^{\langle 0 \rangle}$ and $e^{\langle T \rangle}$) are given. Then, the planning can be formalized as follows:
  To get the most likely path of state transitions that maximizes the probability $Pr(e^{\langle 0 \rangle}, e^{\langle T \rangle}|\text{Agent})$.
  When the environment is handled as output like the proposed method, we can seek the most likely path simply using well-known algorithm like Viterbi's one. On the other hand, we need an additional simulator or inverse model of environment when the environment is handled as input.
– When we use continuous value for input of HMM, we need to use gradient ascent methods like neural networks to learn the parameters in a cycle, which requires more computation power. On the other hand, in the proposed method, we can apply the one-shot adaptation algorithm derived in Section 2.2.

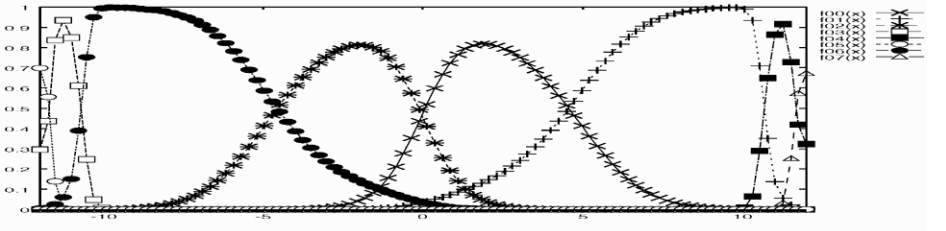
[2] The transitions B →D and C →A are extracted from the probability matrix of state transitions of the trained HMM.



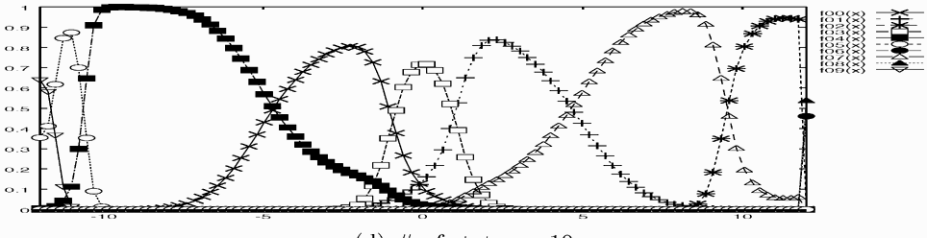
(a) # of states = 4



(b) # of states = 6



(c) # of states = 8



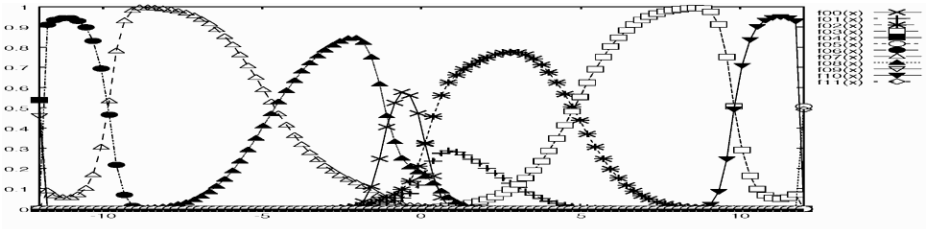
(d) # of states = 10

Fig. 5. Changes of Segmentation ($Pr(s|e)$) by the Number of States (in Ex.2).

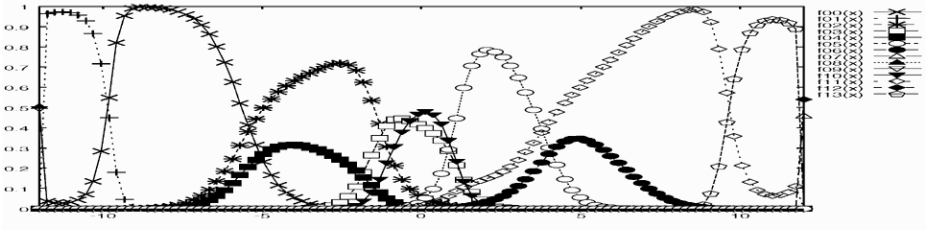
3 Symmetrically Coupled HMM

3.1 Symmetricity Assumption

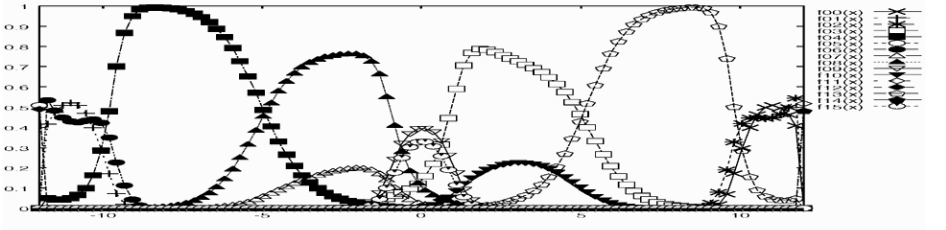
In Section 2, we handle environment and agent's intention by a single HMM. However, the number of states increases exponentially when the agent has more complex intentions. This is significant when HMM handles interactions among agents in multi-agent systems(MAS). In this case, we will face *generalization performance problem*. As the number of states or learning parameters increases,



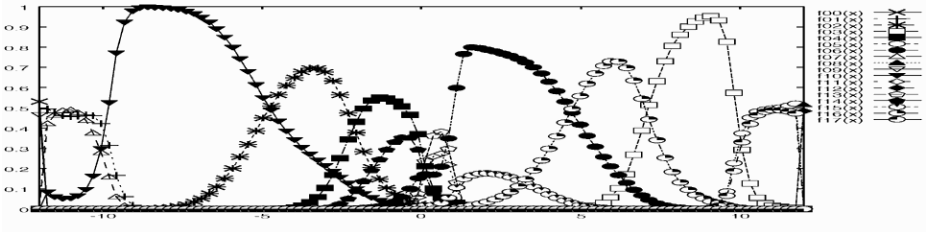
(e) # of states = 12



(f) # of states = 14



(g) # of states = 16

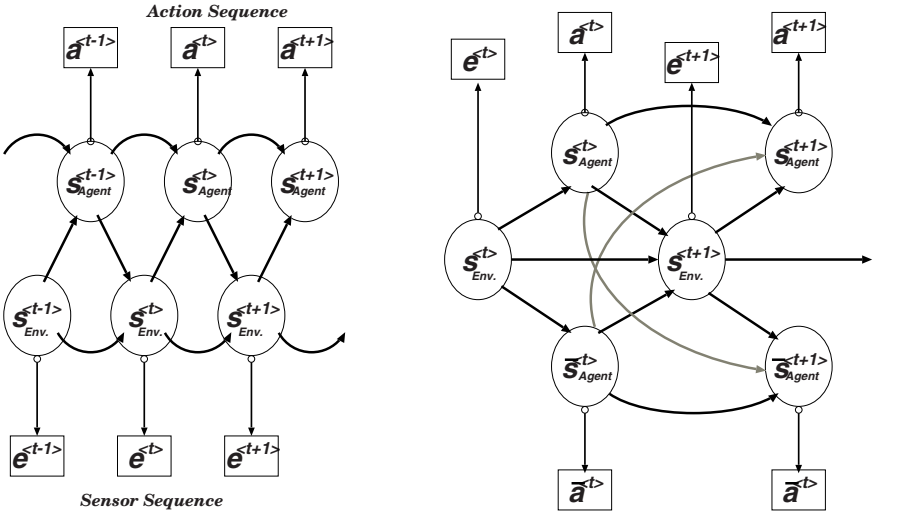


(h) # of states = 18

Fig. 5. (Continued).

the huge number of examples are required to guarantee the generalization performance. In order to avoid this problem, I introduce *symmetricity assumption* among agents as follows:

symmetricity assumption
Agents in a MAS are *symmetric*, that is, every agent has the same rules of behavior. In the HMM context, every agent shares the same state transition rules with each other.



(a) Agent and Environment (b) Two Agents and Environment

Fig. 6. Coupled HMMs of Agents and Environment.

To reflect the above assumption in HMM, first, I divide the internal state into two states, environment state $s_{\mathbf{e}}$ and agent state $s_{\mathbf{a}}$, and form a coupled HMM as shown in Figure 6-(a). In this model, sensor data $e^{\langle t \rangle}$ and action commands $a^{\langle t \rangle}$ are determined by environment states $s_{\mathbf{e}}^{\langle t \rangle}$ and agent states $s_{\mathbf{a}}^{\langle t \rangle}$ respectively. Transitions of both states are determined as follows: The next environment state $s_{\mathbf{e}}^{\langle t+1 \rangle}$ is determined according to the current environment and agent states $\{s_{\mathbf{e}}^{\langle t \rangle}, s_{\mathbf{a}}^{\langle t \rangle}\}$. The next agent state $s_{\mathbf{a}}^{\langle t+1 \rangle}$ is determined according to the current agent and the new environment states $\{s_{\mathbf{a}}^{\langle t \rangle}, s_{\mathbf{e}}^{\langle t+1 \rangle}\}$. Then I introduce the second agent who cooperates with the first agent as shown in Figure 6-(b). In this coupling, both state transitions become affected by the second agent state $\bar{s}_{\mathbf{a}}^{\langle t \rangle}$ in the following manner:

$$Pr(s_{\mathbf{e}}^{\langle t+1 \rangle}|*) = Pr(s_{\mathbf{e}}^{\langle t+1 \rangle}|s_{\mathbf{e}}^{\langle t \rangle}, s_{\mathbf{a}}^{\langle t \rangle}, \bar{s}_{\mathbf{a}}^{\langle t \rangle})$$
$$Pr(s_{\mathbf{a}}^{\langle t+1 \rangle}|*) = Pr(s_{\mathbf{a}}^{\langle t+1 \rangle}|s_{\mathbf{a}}^{\langle t \rangle}, \bar{s}_{\mathbf{a}}^{\langle t \rangle}, s_{\mathbf{e}}^{\langle t+1 \rangle})$$

In order to complete the state transition for Figure 6-(b), we must consider about transitions of the second agent state $\bar{s}_{\mathbf{a}}$. Here, I apply *symmetricity assumption* for the second state transition, that is, the probabilities of state transitions of the second agent are determined by the same one of the first agent. The most naive implementation of this assumption is that the probabilities are described as follows:

$$\begin{aligned} Pr(\bar{s}_{\mathbf{a}}^{\langle t+1 \rangle}|*) &= Pr(\bar{s}_{\mathbf{a}}^{\langle t+1 \rangle}|\bar{s}_{\mathbf{a}}^{\langle t \rangle}, s_{\mathbf{a}}^{\langle t \rangle}, s_{\mathbf{e}}^{\langle t+1 \rangle}) \\ &= Pr(s_{\mathbf{a}}^{\langle t+1 \rangle} = \bar{s}_{\mathbf{a}}^{\langle t+1 \rangle}|s_{\mathbf{a}}^{\langle t \rangle} = \bar{s}_{\mathbf{a}}^{\langle t \rangle}, \bar{s}_{\mathbf{a}}^{\langle t \rangle} = s_{\mathbf{a}}^{\langle t \rangle}, s_{\mathbf{e}}^{\langle t+1 \rangle}) \end{aligned}$$

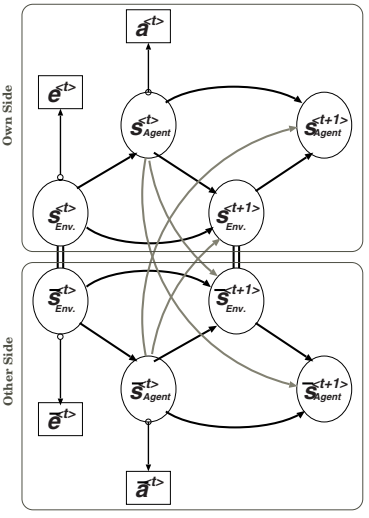


Fig. 7. Symmetrically Coupled HMM. s_{Agent} and s_{Env} correspond to $s_{\mathbf{a}}$ and $s_{\mathbf{e}}$ in the agent definition respectively.

This formulation is valid when both agents share the same environment state. In general, however, two agents may have different environment state inside of them, because the environment state in this formalization is a kind of internal world state that each agent has. Such a situation is not avoidable especially when the sensor data $e^{\langle t \rangle}$ is represented from the viewpoint of each agent. In order to overcome this problem, I propose a *symmetrically coupled HMM* (sCHMM) shown in Figure 7. In this model, the second agent has its own environment state $\bar{s}^{\langle t \rangle}$. Using this, the transition of $\bar{s}_{\mathbf{a}}^{\langle t \rangle}$ are represented as follows:

$$Pr(\bar{s}_{\mathbf{a}}^{\langle t+1 \rangle}|*) = Pr(s_{\mathbf{a}}^{\langle t+1 \rangle} = \bar{s}_{\mathbf{a}}^{\langle t+1 \rangle}|s_{\mathbf{a}}^{\langle t \rangle} = \bar{s}_{\mathbf{a}}^{\langle t \rangle}, \bar{s}_{\mathbf{a}}^{\langle t \rangle} = s_{\mathbf{a}}^{\langle t \rangle}, s_{\mathbf{e}}^{\langle t+1 \rangle} = \bar{s}_{\mathbf{e}}^{\langle t+1 \rangle}),$$

where the transition of the second environment state $\bar{s}^{\langle t \rangle}$ follows:

$$Pr(\bar{s}_{\mathbf{e}}^{\langle t+1 \rangle}|*) = Pr(s_{\mathbf{e}}^{\langle t+1 \rangle} = \bar{s}_{\mathbf{e}}^{\langle t+1 \rangle}|s_{\mathbf{e}}^{\langle t \rangle} = \bar{s}_{\mathbf{e}}^{\langle t \rangle}, s_{\mathbf{a}}^{\langle t \rangle} = \bar{s}_{\mathbf{a}}^{\langle t \rangle}, \bar{s}_{\mathbf{a}}^{\langle t \rangle} = s_{\mathbf{a}}^{\langle t \rangle})$$

3.2 Formalization and Learning Procedure

I summarize the sCHMM agent as the following tuple:

$$\text{Agent} = \langle \boldsymbol{S}_{\mathbf{e}}, \boldsymbol{S}_{\mathbf{a}}, \boldsymbol{E}, \boldsymbol{A}, \boldsymbol{P}_{\mathbf{e}}, \boldsymbol{P}_{\mathbf{a}}, \boldsymbol{Q}_{\mathbf{e}}, \boldsymbol{Q}_{\mathbf{a}}, \boldsymbol{\pi}_{\mathbf{e}}, \boldsymbol{\pi}_{\mathbf{a}} \rangle ,$$

where $\boldsymbol{S}_{\mathbf{a}} = \{s_{\mathbf{a}i}\}$ and $\boldsymbol{S}_{\mathbf{e}} = \{s_{\mathbf{e}i}\}$ are sets of states for agent and environment respectively, $\boldsymbol{E} = \{e_i\}$ is a set of sensor data of environment, and $\boldsymbol{A} = \{a_i\}$ is a set of agent action symbols. $\boldsymbol{P}_{\mathbf{e}} = \{p_{\mathbf{e}ijkl} | i \in \boldsymbol{S}_{\mathbf{e}}, j, k \in \boldsymbol{S}_{\mathbf{a}}, \forall t\}$ and $\boldsymbol{P}_{\mathbf{a}} = \{p_{\mathbf{a}jklm} | j, k \in \boldsymbol{S}_{\mathbf{a}}, m \in \boldsymbol{S}_{\mathbf{e}}, \forall t\}$ are probability tensors of state transitions of environment and agent, $\boldsymbol{Q}_{\mathbf{e}} = \{q_{\mathbf{e}i}(e) | i \in \boldsymbol{S}_{\mathbf{e}}, e \in \boldsymbol{E}, \forall t\}$ and $\boldsymbol{Q}_{\mathbf{a}} = \{q_{\mathbf{a}j}(a) | j \in$

$\boldsymbol{S_a}, a \in \boldsymbol{A}, \forall t\}$ are probability tensors of is observed symbols of environment and actions, and $\boldsymbol{\pi_e} = \{\pi_{\mathbf{e}i} = Pr(s_{\mathbf{e}}^{\langle 0\rangle} = i) | i \in \boldsymbol{S_e}\}$ and $\boldsymbol{\pi_a} = \{\pi_{\mathbf{a}j} = Pr(s_{\mathbf{a}}^{\langle 0\rangle} = j) | j \in \boldsymbol{S_a}\}$ are probability vectors of initial states of environment and agent. Each element of $\boldsymbol{P_e}$, $\boldsymbol{P_a}$, $\boldsymbol{Q_e}$, and $\boldsymbol{Q_a}$ represents the following probability.

$$
\begin{aligned}
p_{\mathbf{e}ijkl} &= Pr(s_{\mathbf{e}}^{\langle t\rangle} = l \mid s_{\mathbf{e}}^{\langle t-1\rangle} = i, s_{\mathbf{a0}}^{\langle t-1\rangle} = j, s_{\mathbf{a1}}^{\langle t-1\rangle} = k) \\
p_{\mathbf{a}jklm} &= Pr(s_{\mathbf{a}}^{\langle t\rangle} = m \mid s_{\mathbf{a0}}^{\langle t-1\rangle} = j, s_{\mathbf{a1}}^{\langle t-1\rangle} = k, s_{\mathbf{e}}^{\langle t\rangle} = l) \\
q_{\mathbf{e}i}(e) &= Pr(e^{\langle t\rangle} = e \mid s_{\mathbf{e}}^{\langle t\rangle} = i) \\
q_{\mathbf{a}j}(a) &= Pr(a^{\langle t\rangle} = a \mid s_{\mathbf{a}}^{\langle t\rangle} = i)
\end{aligned}
$$

We can derive a learning procedure for sCHMM as shown below. Suppose that sequences of sensor information $\{e^{\langle t\rangle}\}$, agent's own actions $\{a^{\langle t\rangle}\}$, and other's actions $\{\bar{a}^{\langle t\rangle}\}$ are observed ($0 \le t < T$). We can calculate agent's own forward and backward probabilities, $\alpha_{lmn}^{\langle t\rangle}$ and $\beta_{ijk}^{\langle t\rangle}$ respectively, as follows:

$$
\alpha_{lmn}^{\langle t\rangle} = \begin{cases} \pi_{\mathbf{e}l}\pi_{\mathbf{a}m}\pi_{\mathbf{a}n} Q_{(lmn)}(W^{\langle 0\rangle}) & ; t = 0 \\ \displaystyle\sum_{(ijk)} \alpha_{ijk}^{\langle t-1\rangle} P_{(ijk)(lmn)} Q_{(lmn)}(W^{\langle t\rangle}) & ; \textit{otherwise} \end{cases}
$$

$$
\beta_{ijk}^{\langle t\rangle} = \begin{cases} 1 & ; t = T-1 \\ \displaystyle\sum_{(lmn)} P_{(ijk)(lmn)} Q_{(lmn)}(W^{\langle t+1\rangle}) \beta_{lmn}^{\langle t+1\rangle} & ; \textit{otherwise} \end{cases},
$$

where

$$
\begin{aligned}
P_{(ijk)(lmn)} &= p_{\mathbf{e}ijkl} \cdot p_{\mathbf{a}jklm} \cdot p_{\mathbf{a}kjln} \\
Q_{(ijk)}(W^{\langle t\rangle}) &= Q_{(ijk)}(e^{\langle t\rangle}, a^{\langle t\rangle}, \bar{a}^{\langle t\rangle}) \\
&= q_{\mathbf{e}i}(e^{\langle t\rangle}) \cdot q_{\mathbf{a}j}(a^{\langle t\rangle}) \cdot q_{\mathbf{a}k}(\bar{a}^{\langle t\rangle})
\end{aligned}
$$

In the same way, other's forward and backward probabilities, $\bar{\alpha}_{lnm}^{\langle t\rangle}$ and $\bar{\beta}_{ikj}^{\langle t\rangle}$ respectively, can be calculated:

$$
\bar{\alpha}_{lnm}^{\langle t\rangle} = \begin{cases} \pi_{\mathbf{e}l}\pi_{\mathbf{a}m}\pi_{\mathbf{a}n} Q_{(lnm)}(\bar{W}^{\langle 0\rangle}) & ; t = 0 \\ \displaystyle\sum_{(ikj)} \bar{\alpha}_{ikj}^{\langle t-1\rangle} P_{(ikj)(lnm)} Q_{(lnm)}(\bar{W}^{\langle t\rangle}) & ; \textit{otherwise} \end{cases}
$$

$$
\bar{\beta}_{ikj}^{\langle t\rangle} = \begin{cases} 1 & ; t = T-1 \\ \displaystyle\sum_{(lnm)} P_{(ikj)(lnm)} Q_{(lnm)}(\bar{W}^{\langle t+1\rangle}) \bar{\beta}_{lnm}^{\langle t+1\rangle} & ; \textit{otherwise} \end{cases},
$$

where $\bar{W}^{\langle t\rangle} = \{\bar{e}^{\langle t\rangle}, \bar{a}^{\langle t\rangle}, a^{\langle t\rangle}\}$, and $\bar{e}^{\langle t\rangle}$ is the sensor data received by the second agent. Using these probabilities, we can adapt transition and output probabilities $p_{\mathbf{e}ijkl}, p_{\mathbf{a}jklm}, q_{\mathbf{e}i}, q_{\mathbf{e}j}$ as follows:

$$
\begin{aligned}
p_{\mathbf{e}ijkl} &\leftarrow \sum_m \sum_n \hat{P}_{(ijk)(lmn)} \\
p_{\mathbf{a}jklm} &\leftarrow \frac{\sum_i \sum_n \hat{P}_{(ijk)(lmn)}}{\sum_i p_{\mathbf{e}ijkl}} \\
q_{\mathbf{e}i}(e) &\leftarrow \sum_j \sum_k \sum_a \sum_{\bar{a}} \hat{Q}_{(ijk)}(e, a, \bar{a}) \\
q_{\mathbf{a}j}(a) &\leftarrow \sum_i \sum_k \sum_e \sum_{\bar{a}} \hat{Q}_{(ijk)}(e, a, \bar{a}),
\end{aligned}
$$

where

$$
\hat{P}_{(ijk)(lmn)} = \frac{\sum_t \xi^{\langle t \rangle}_{(ijk)(lmn)} + \sum_t \bar{\xi}^{\langle t \rangle}_{(ikj)(lnm)}}{\sum_t \gamma^{\langle t-1 \rangle}_{ijk} + \sum_t \bar{\gamma}^{\langle t-1 \rangle}_{ikj}} \tag{2}
$$

$$
\hat{Q}_{(ijk)}(W) = \frac{\sum_{t, W^{\langle t \rangle} = W} \gamma_(ijk)^{\langle t \rangle} + \sum_{t, W^{\langle t \rangle} = W} \bar{\gamma}^{\langle t \rangle}_{ikj}}{\sum_t \gamma_(ijk)^{\langle t \rangle} + \sum_t \bar{\gamma}^{\langle t \rangle}_{ikj}} \tag{3}
$$

$$
\begin{aligned}
\xi^{\langle t \rangle}_{(ijk)(lmn)} &= \alpha^{\langle t-1 \rangle}_{(ijk)} P_{(ijk)(lmn)} Q_{(lmn)}(W^{\langle t \rangle}) \beta^{\langle t \rangle}_{(lmn)} \\
\bar{\xi}^{\langle t \rangle}_{(ikj)(lnm)} &= \bar{\alpha}^{\langle t-1 \rangle}_{(ikj)} P_{(ikj)(lnm)} Q_{(lnm)}(\bar{W}^{\langle t \rangle}) \bar{\beta}^{\langle t \rangle}_{(lnm)} \\
\gamma^{\langle t \rangle}_{(lmn)} &= \alpha^{\langle t \rangle}_{(lmn)} \beta^{\langle t \rangle}_{(lmn)} \\
\bar{\gamma}^{\langle t \rangle}_{(lnm)} &= \bar{\alpha}^{\langle t \rangle}_{(lnm)} \bar{\beta}^{\langle t \rangle}_{(lnm)}
\end{aligned}
$$

3.3 Discussion: The Number of Parameters in the Model

As mentioned before, the number of parameters in HMM is an important factor for generalization performance of learning. In the case of the coupled HMM, especially, the number of parameters increases exponentially. Actually, if we use the model shown in Figure 6-(b) without the symmetricity assumption, the number of parameters in the state transition is

$$
|S_{\mathbf{e}}|^2 |S_{\mathbf{a}}|^N + N |S_{\mathbf{e}}| |S_{\mathbf{a}}|^{N+1}
$$

where N is the number of agents. This is already reduced from $(|S_{\mathbf{e}}| |S_{\mathbf{a}}|^N)^2$, the number of parameters in the case we represent the same model using single HMM. Compared with this, symmetrically coupled HMM has fewer parameters as follows:

$$
|S_{\mathbf{e}}|^2 |S_{\mathbf{a}}|^N + |S_{\mathbf{e}}| |S_{\mathbf{a}}|^{N+1}
$$

In addition to it, the symmetricity assumption can increase the virtual number of examples. Eq. 2 and Eq. 3 mean that the same HMM is trained by using both pairs of $\{e^{\langle t \rangle}, a^{\langle t \rangle}\}$ and $\{\bar{e}^{\langle t \rangle}, \bar{a}^{\langle t \rangle}\}$ for a given observation $\{e^{\langle t \rangle}, a^{\langle t \rangle}, \bar{a}^{\langle t \rangle}\}$. As a result, the generalization performance is improved by the virtually doubled examples.

It is, however, true that an sCHMM still has too many parameters for real applications. Therefore, it is meaningful to introduce additional assumptions to reduce the number of parameter. Fortunately, in the case of cooperative interaction in the MAS, we can pick-up reasonable assumptions as follows:

– "*no explicit communication*" assumption: In the formalization of sCHMM, the transition of the agent state is affected by the previous states of other agents. This corresponds the case that agents use explicit communication with each other in every action cycle. In the case of human cooperative behaviors like soccer, on the other hand, we do not use so much explicit communication, but model others via sensor information instead. In such case, the transition of the agent state can be represented as follows:

$$Pr(s_{\mathbf{a}}^{\langle t+1\rangle}|*) = Pr(s_{\mathbf{a}}^{\langle t+1\rangle}|s_{\mathbf{a}}^{\langle t\rangle}, s_{\mathbf{e}}^{\langle t+1\rangle})$$

In this case, the total number of the parameters is reduced to:

$$|S_{\mathbf{e}}|^2 |S_{\mathbf{a}}|^N + |S_{\mathbf{e}}| |S_{\mathbf{a}}|^2$$

– "*filtering*" assumption: Usually, when we write a code of agent behavior, we classify states systematically. For example, in the code shown in Section 1 states are grouped by agent's roles (agent states) first then branched by world status (environment states) second. This can be represented by the following manner in the transition of HMM:

$$Pr(s_{\mathbf{a}}^{\langle t+1\rangle}|*) = Pr(s_{\mathbf{a}}^{\langle t+1\rangle}|s_{\mathbf{e}}^{\langle t+1\rangle}) \cdot Pr(s_{\mathbf{a}}^{\langle t+1\rangle}|s_{\mathbf{a}}^{\langle t\rangle})$$

In this case, the number of parameters are reduced to:

$$|S_{\mathbf{e}}|^2 |S_{\mathbf{a}}|^N + |S_{\mathbf{e}}| |S_{\mathbf{a}}| + |S_{\mathbf{a}}|^{N+1}$$

– "*shared joint intention*" assumption: During a cooperation of multiple agents each agent believes that all agents share the joint intention. This means that each agent thinks that all other agents will behave as the agent wants. In this case, the transition of environment states can be represented as follows:

$$Pr(s_{\mathbf{e}}^{\langle t+1\rangle}|*) = Pr(s_{\mathbf{e}}^{\langle t+1\rangle}|s_{\mathbf{e}}^{\langle t\rangle}, s_{\mathbf{a}}^{\langle t\rangle})$$

This reduces the number of parameters to:

$$|S_{\mathbf{e}}|^2 |S_{\mathbf{a}}| + |S_{\mathbf{e}}| |S_{\mathbf{a}}|^{N+1}$$

Note that this assumption can not be applied with the "*no explicit communication*" assumption, because the sCHMM is reduced into a simple CHMM like Figure 6-(a) that does not reflect cooperation among agents.

3.4 Related Works

Uther and Veloso [UV98] have been attacked the problem of segmentation of the continuous environment and proposed Continuous U Tree algorithm. Although the algorithm is a powerful tool for segmenting a given continuous data space, it is hard to apply for the purpose to find unobservable features like direction feature in the experiments shown in Section 2.4. Han and Veloso [HV99] showed a framework to recognize behaviors of robots by HMM in which the environment is handled as output. In this work, they did not focused on acquiring state transitions, but method to find an HMM from multiple pre-defined HMMs for seen data.

Brand Et al. [Bra97,hMmfcar96] proposed coupled HMM and its learning method, in which several HMMs are coupled via inter-HMM dependencies. Jordan Et al. [JGS97b,GJ97,JGJS99] proposed factorial HMM and hidden Markov decision trees. Both of works mainly focused on reducing the complexity in EM processes. Even using these HMMs, the complexity of calculation of a naive implementation increase exponentially, so that it is hard to handle the large number of states. They use mean field approximation or N-heads dynamic programming to reduce the cost of the approximation of posterior probabilities. However, they does not focused on symmetricity in agent-interactions and generalization performance problem.

These methods can be applicable to our model. Actually, a naive implementation of learning method derived in the previous section costs $O(TN^4M^2)$, which is too huge for dynamical application like soccer. Above methods will reduce the cost into $O(TN^2M)$, which is reasonable cost for real application.

4 Concluding Remarks

In this article, we proposed two frameworks to learn behaviors of multiple agents in dynamic environment using HMM. The first framework handles agent's environments as output of HMM rather than as input. As the result, the acquired HMM represents suitable segmentation of environment explicitly in the states. The explicit segmentation is expected to leads the following features to the HMM:

- HMM can be used planning of agent's behavior working in a dynamic environment.
- Flexible segmentation can improve generalization performance of the learning.

The second framework is conducted to represent interactions among multiple agents and environments. In order to avoid the explosion of the number of parameters, I introduced symmetricity assumptions among agents, and propose symmetrically coupled HMM (sCHMM) and its learning procedure.

There are the following open issues on the proposed model and method:

– The cost of calculation increase exponentially when structures of agents and environments become complicated. In order to reduce the complexity, several techniques like mean field approximation and N-head dynamic programming should be applied to these models.
– The incremental learning will suit to acquire high-level cooperative behaviors. We may be able to realize the step-by-step learning using dependency of the initial parameters.

References

BF95. Yoshua Bengio and Paolo Frasconi. An input output hmm architecuture. In G. Tesauro, D. Touretzky, and T. Leen, editors, *Advances in Neural Information Processing Systems*, pages 427–434. The MIT Press, 1995.

Bra97. Matthew Brand. Coupled hideen markov models for modeling interacting processes. Perceptual Computing/Learning and Common Sense Technical Report 405, MIT Lab, jun 1997.

GJ97. Zoubin Ghahramani and Michael I. Jordan. Factorial hidden markov models. *Machine Learning*, 29:245–275, 1997.

hMmfcar96. Coupled hidden Markov models for complex action recognition. Matthew brand and nuria oliver and alex pentland. Perceptual Computing/Learning and Common Sense Technical Report 407, MIT Media Lab, 20 1996.

HV99. Kwun Han and Manuela Veloso. Automated robot behavior recognition applied to robotic soccer. In *Proceedings of IJCAI-99 Workshop on Team Behaviors and Plan Recognition*, 1999.

JGJS99. Michael I. Jordan, Zoubin Ghahramani, Tommi Jaakkola, and Lawrence K. Saul. An introduction to variational methods for graphical models. *Machine Learning*, 37(2):183–233, 1999.

JGS97a. Michael I. Jordan, Zoubin Ghahramani, and Lawrence K. Saul. Hidden markov decision trees. In Michael C. Mozer, Michael I. Jordan, and Thomas Petsche, editors, *Advances in Neural Information Processing Systems*, volume 9, page 501. The MIT Press, 1997.

JGS97b. Michael I. Jordan, Zoubin Ghahramani, and Lawrence K. Saul. Hidden markov decision trees. In Michael C. Mozer, Michael I. Jordan, and Thomas Petsche, editors, *Advances in Neural Information Processing Systems 9*, page 501. The MIT Press, 1997.

UV98. William T. B. Uther and Manuela M. Veloso. Tree based discretization for continuous state space reinforcement learning. In *AAAI/IAAI*, pages 769–774, 1998.