Poseidon Team Description Paper
robocup 2006 Bremen

Nasrin Mostafazadeh,Saba Ardeshiri,Sepideh Movaghati Shadi Hariri,Zeinab Jahanzad,Alireza Fathi,Majid Valipour

Tehran Farzanegan highschool

Abstract. The aim of this paper is to technically describe Poseidon Team's contributions to rescue agent development. each section clearly defines mathematical model of main problems in its area and discuss Poseidon's approach to solve them while explaining existing solutions and their pros and cons. also each section is followed by a future work part which illustrates Poseidon road map toward Bremen 2006.

1 Introduction

The Rescue Simulation league has been introduced to advance research on the multi-agent systems and AI fields.This goal has been achieved by simulating a large urban disaster,and modeling heterogeneous agents' actions in this environment.

As a rescue simulation team we have tried to improve available solutions for each agent's problems.
Our contributions include new ideas for communication which have also influenced our team structure as well,better solutions for police force and ambulance team agents and using data mining algorithms to model different simulators' behaviors.

In the following sections,we start our discussion be setting out our communication and world modeling approach,then we proceed to explain other team's solutions in our fields of interest for each agent and compare ours with theirs.The final section will illustrate the future works and road map to Bremen 2006 competitions.

2 Communication Infrastructure

We are calling our communication an infrastructure because we have attempted to separate communication from decision making section for each agent.Currently we have a transparent communication layer between our base and main agents code (*Fig. 1*).Developing communication as an independent layer has helped us to be able to modify and alter the communication or decision making layer without any concern about the other one.

In the following subsection we are going to explain more about our communication which is completely distributed but also has capabilities to be used for implementing centralized agents too.

2.1 Communication strategy

If messages among agents were not constrained by size and quantity, it would have been best for each agent to send all its information in each cycle to other agents in order to have a distributed decision making strategy. But in reality not only there are limitations on both size and number of messages,but also there are limits on the process and network capacities. Therefore we have to send the most important data and try to avoid redundancy.

The upper bound limit for the number of messages a platoon agent can hear in one cycle, is 4. Also this bound for station agent is 2N, in which N is the number of platoon agents of its type. The number of messages an agent can tell is equal to the number of messages it can hear. The other restriction is that an agent can choose between the messages just by reading each message's header which contains only the ID of the sender.

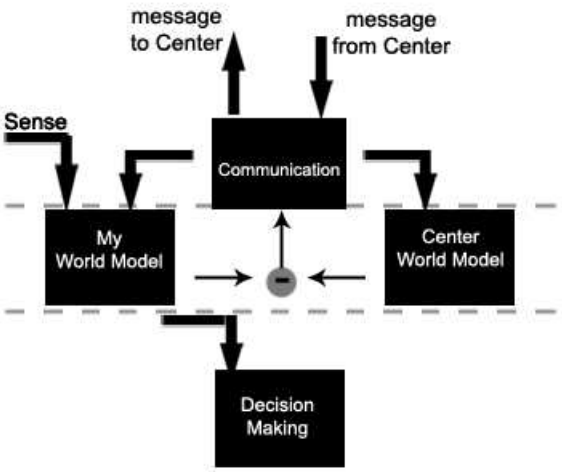

Fig. 1. Architecture of our agents' code

It has been experimentally verified that the size of the sensed package in each cycle for a platoon agent is on average nearly equal to the size of one message packet so each agent is able to send all its sensed information to the center in a packet.In the other words, if each agent tells a message which contains its recently sensed information in each cycle then there will be N messages available for the station agent and also platoon agents of the same type to hear in each cycle.

Usually N is a number greater than 4, then it is not possible for platoon agents to hear all these messages. But station agent has the capability to hear 2N messages in each cycle so it can hear N messages and it is able to hear N more messages too.Let's have a sample environment including **P** PoliceForce agents,**A** AmbulanceTeam agents and **F** FireBrigade agents. We define:

$$\text{MIN} = \text{minimum(F,P,A)}$$

Each station agent will use N messages to send its information to its platoon agents and MIN messages to send information to the other station agents. For example the PoliceOffice will send P messages to its agents and $\frac{MIN}{2}$ messages to the AmbulanceCenter and $\frac{MIN}{2}$ messages to the FireStation (*Fig. 2*).

But the main problem is how to send the most useful information in these N messages to platoon agents since in each cycle they can just hear the maximum of 4 messages. To solve this problem there are different ideas which we discuss each one with its cons and pros in the next paragraphs:

- **Idea1:** The station tells 4 messages in each cycle and all agents listen to these 4 messages. The station should consider what the best messages are for all agents. Information not known by the majority of agents would have a higher priority.

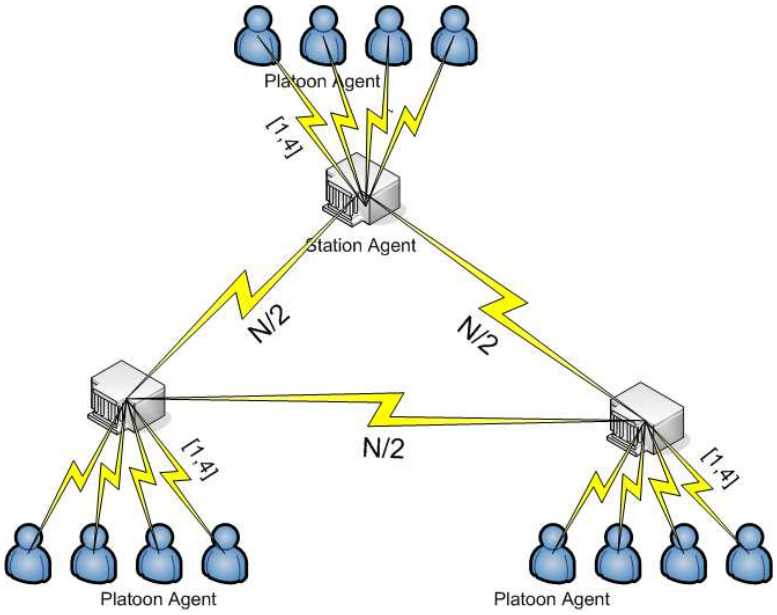

Fig. 2. Scheme of communication structure for Agents

- **Idea2:** There are N messages available for N platoon agents.The easiest solution is one message for each agent. In this case the station agent,while considering what each agent knows up to now, packs a message for each agent and sends these messages in a sequence sorted by agents' ids. This strategy is easy to implement but isn't considered efficient.
- **Idea3:** Although each platoon agent has the capability of hearing 4 messages it just hears one message in each cycle in the previous strategy. To use all of their capacity we divide platoon agents into groups of four. For each group station agent, sends four messages. This is better than the previous strategy because there are some duplicated information which would be eliminated when putting up the messages.
- **Idea4:** Although the third idea will solve the stated problem of *"no using maximum capacity "* it can be like the second idea in the worst case. The worst case happens when information in the packets are mutually exclusive. So we offer another idea which we think is the best.

  let's assume that we have some information units, some messages and some agents (*Fig.* **??**). Each message is owned by 4 agents. Some Information Units are useless for some agents because they have known them before, some Information units are more important and critical because they don't know about them at all. It is equal to consider that each agent has a priority function which assigns a priority to each information unit.

This problem has two main concerns. The first concern is which IUs[1] should be included in which packet or packets of message and the second is which agents own each message. Finally we need each agent to be assigned some packets of messages which will hear in the coming cycle.
There is a dynamic programming solution for this problem but it is too time consuming that we prefer using auction method instead. In our auction solution each IU has a value and each agent is going to buy it. Agents with the maximum offer will win. Then one message will be assigned to each 4 agents using greedy algorithm. Of course this is not the best solution but it is inevitable considering our process and time.

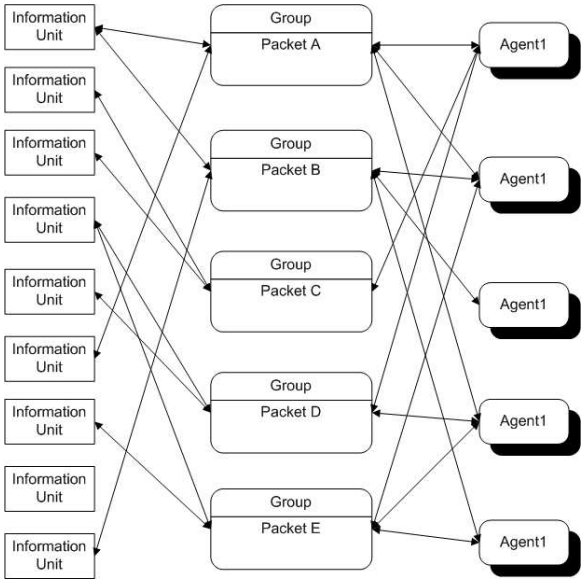

Fig. 3. Implementation of Idea4

Future Works: Our current agents are using idea2 because of its ease of implementation. We plan to implement idea4 for the Bremen 2006.

3 Injured civilian selection problem

Ambulance team agent can be considered as the most important agent in a normal simulation run because of the ranking method of the competition which gives number of alive civilians a great weight in the final ranking(due to importance of human life in real world).

[1] Information unit

For an ambulance team agent, selecting best injured civilian among available ones is the most important decision to make. Because rescuing a buried agent will take time and in the meantime the other injured civilians would get closer to death.

We believe that only three parameters affect our decision.These parameters are death time of civilians, rescue time of civilians and possible move time between two consequent selected civilians.

So the problem can be defined in the following form:

- **Goal:**maximizing number of alive civilians after 300 cycles.
- **Inputs:** civilians' *death time, rescue time, move time* between two civilians
- **Output:** the best possible civilian to rescue.

obviously our incomplete knowledge about state of all civilians at the beginning of the simulation would directly affect the validity of our decision.

3.1 Available solutions

Due to complexity of the stated problem there is not any exact polynomial solution for it. In order to solve the problem some methods of simplification have been employed by different teams, one of which assumes that all ambulance team agents are working together. This simple assumption leads to a dynamic algorithm for which its output is a specific sequence of civilians.Rescuing the civilians of this sequence in the exact order will result to minimization of number of dead civilians.

One of the best implementation of this algorithm belongs to **s.o.s**[1] which relays on a centralized decision making scheme and uses a three dimension table(civilian,time,move time) to select the mentioned sequence of civilians.

In the **Osaka 2005** competition it has been practically proved by several teams that this simple assumption will dramatically change the result of the dynamic algorithm from the ideal possible set of rescue targets which leads to maximum score.

Another major drawback of the *working together assumption* is its essential need to coordinate and eventually the unbearable dependency to the center.

3.2 Poseidon's approach

As we discussed in the previous section, there isn't any good solution for the problem when we don't assume that all ambulance teams are working together. To solve the selection problem we decide to transform it to a similar problem which is a modified version of **Impossibles**'s idea. we have tried to change the goal of the selection algorithm and have defined a new criteria to optimize using our proposed algorithm.

We found out that almost in every situation there are some valuable civilians to rescue even at the end of the simulation. So we changed our goal to maximize the number of "valuable act rescues" instead of maximizing the number of rescued civilians. We believe that maximizing the former will maximize the latter.

valuable act rescue: an act rescue published by an ambulance team agent which will prevent a civilian from death.

In order to maximize this criterion we should assign a civilian only the number of ambulances which it needs to stay alive until getting to refuge. As we defined, helping a civilian who will die if not being helped is valuable, independent from the other parameters.

For example helping a civilian who will die in next coming 100 cycles and only needs one ambulance to be saved, is as valuable as helping a civilian who will die in next coming 20 cycles and needs 5 ambulances to be saved.

Here is pseudo-code of our proposed algorithm to maximize our defined parameter:

```
for i = 1 to ‘‘number of civilians’’
n:=numberOfNeededAgent(civilian(i));//This function will calculate
// number of needed agents using DT and RT of civilian and numberOfAgents
// which has working on it before
  if(n>=1)//means that this civilian still need someone to survive
    PriorityQueue[n].add(civilian(i));//this Queue will sort civilians
    // considering their deathTime
  end
end
for j = 1 to ‘‘number of agents’’
  for k = 1 to j
    list.add(PriorityQueue[j].front());
  end
  PriorityQueue[j].pop();
end
mySelection = list.get(myNumberBetweenAmbulanceTeamAgents);
```

This algorithm has practically been proved to be useful enough specially when agents are working distributed and there isn't any central decision making unit.

To prevent assigning more than needed number of agents to a civilian [This is in conflict with our goal],it is important to have enough information about other ambulances' positions which will be provided by the communication layer.

3.3 Death time estimation

One of the vital parameters which influences ambulance team agents' selection result is the estimation of a civilian death time. To avoid being dependent on a specific simulator formula, we tried to model the behavior of simulator using data mining methods[2] by a black box approach (*Fig. 4*).

In order to acquire information about the misc simulator's behavior in different situations, it was changed in the way that it logged some basic data about civilians.The logs have been analyzed and used to approximate the simulator

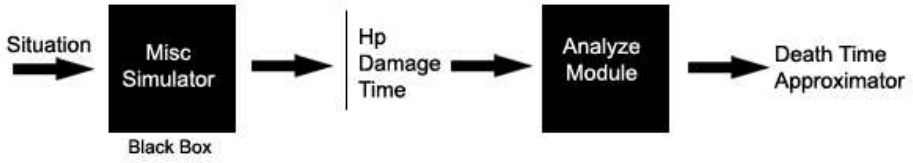

Fig. 4. Our estimation procedure

functionality. We devised our simple approximation method however there are some advanced and standard methods for approximation which we will discuss them in future works section.

The *Fig. 5* is a sample demonstration of our death time estimation function, generated using stated method.The x axis shows the passage of time and the value of y is *estimated death time* using only *hp* and *damage* of the civilian in that time.

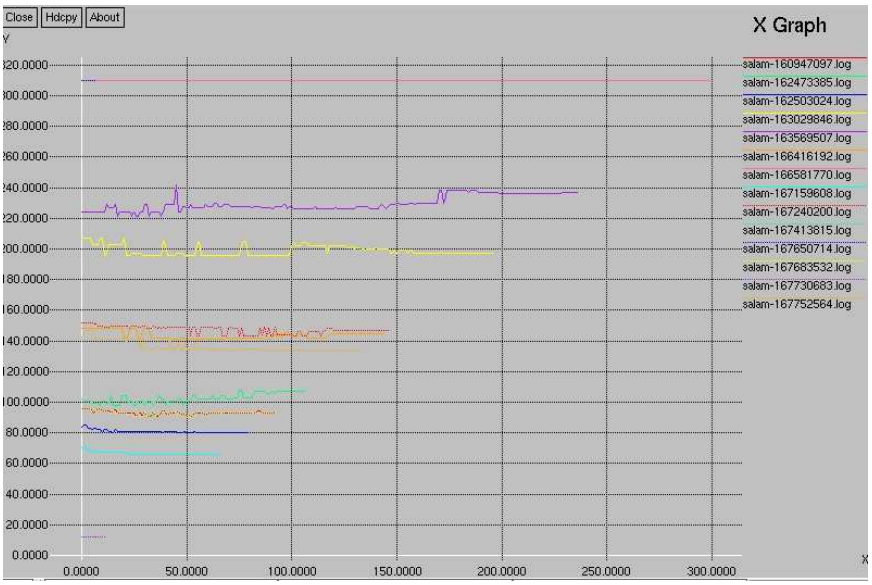

Fig. 5. Our estimation procedure

Future Works: As we mentioned we are currently using a simple approximation method to model the behavior of misc simulator,we have planned to use a fuzzy modeling technique like *takagi-sugeno modeling*[3] or a neuro-fuzzy modeling system like ANFIS[4] in order to have more general and accurate model of the misc simulator.

4 Police Force agent

The blockades are the most important reason for malfunction in cooperation between agents and their action commands. So it is very critical to clear the blockades as soon as possible. As a result clearing the roads started from the beginning of simulation process and we are going to describe our team's algorithm in detail here and the latter will start after the city is a connected graph.

4.1 Reducing blockade's negative effect

To solve the problem We have modeled the city as a graph in which Nodes are the graph's vertices and Roads are the graph's edges. We do believe that there should be at least a path from each vertex of the graph to other vertices to make agents able to move easily within the city. There are different algorithms available to show wether a graph is connected or not. But the main problem is that there are roads that are more worthy to be opened at early cycles of simulation. These roads are the ones near the refuges, the ones near the circumference of fire zones and the roads near the buildings containing buried civilians.

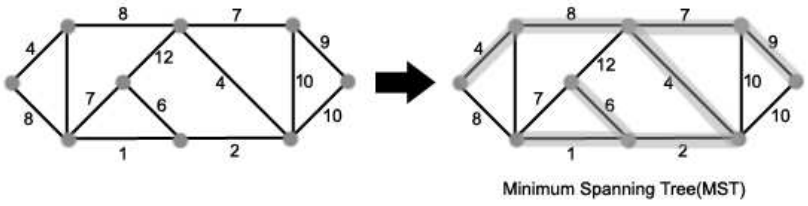

Fig. 6. MST algorithm used by police force

We have given each road a weight by considering the mentioned parameters and additionally regards the time needed to open each road. The more important the road the lower the weight. Then we use a minimum spanning tree algorithm (*Fig. 6*) to choose a tree of roads with the minimum weight. This tree makes the city connected.

Our PoliceForce agents choose the roads that have the best priority first. But as a result of our robust communication, the problem of choosing the same road each cycle by different PoliceForces arises. This causes traffic problems and is completely inefficient. To solve this problem, we have defined a zone for each PoliceForce. Each PoliceForce tries to clear the roads in its zone with a higher priority and roads in its neighboring zones with a lower priority (*Fig. 7*).

After having the city connected PoliceForces will start searching for buried civilians in the buildings and thanks to our communication, none of the buiding would be searched twice.

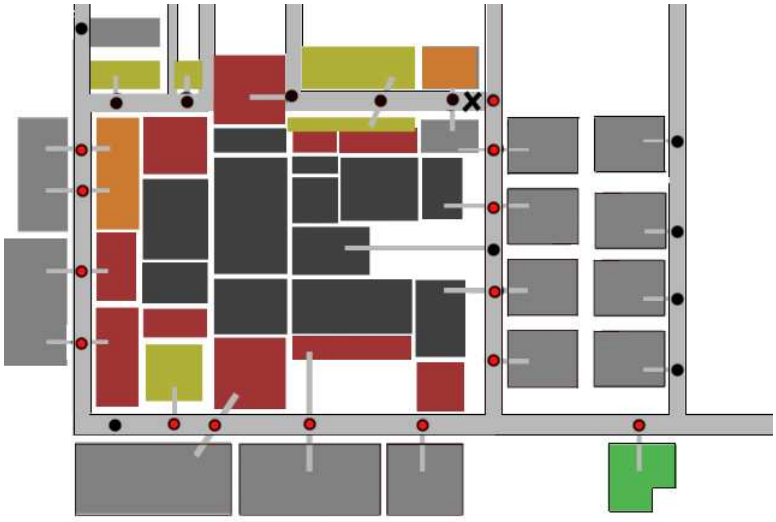

Fig. 7. example of high priority nodes

Finally it should be noted that PoliceAgents are using different weights for path finding to the other agents that makes them pass from the roads that are blocked or have not seen yet.

5 Fire Brigade Agents

Fire Agents are perhaps the most complex agents in our world model because of their job's unpredictability, uncertainty, distribution and sometimes unavailability of enough information about it. Co-operation between Fire Agents is very important and will strongly affect the result in all possible situation. FireBrigades can't extinguish a fire site solely but will be able to do so with the hwlp from others.So it is vital to have this golden cooperation between agents. In this section, we are going tp explain how we have implemented this in Poseidon's FireBrigades.

Our FireBrigades are deciding completely distributed and there are quite enough convincing arguments behind it. Firstly, FireStation information about simulation status are older than agents and this would make the center's decision much more errornous for the agents which are in the states different from what the station thinks they are.

Secondly, Centralized decision making should be able to predict the time for each agent for reaching a fire zone, while existence of blockades especially in former cycles of simulation defects this estimation. clearly, this estimation won't be precise unless having a map-situation dependent way of computation.And it

is in contrast with our strategy to have a code with as little as possible pre-computation and eventually as much as possible situation independence.Also distributed decision making is more compatible with our communication routine.

5.1 Fire fighting strategy

We have divided the simulation into two sections: *First Attack* which is also first phase of fire extinguishing and *Fire Control* which starts afterward and remains to the end of simulation (*Fig. 8*). First Attack is when agents do not know enough about the world and cannot make good distributed decisions. In these cycles each FireBrigade agent plans for himself without considering others. Each FireBrigade tries to extinguish the first building which it sees in the city. When agents have the same information about the ignited buildings it means that we have reached a state that agents can start distributed planning. Then the FirstAttack ends and agents start to extinguish buildings using cooperation.

In these cycles each agent chooses between fire zones the best to be extinguished. Our agents choose this best fire zone considering some parameters like building area of each zone, number of buildings in each zone and sum of buildings' burning potential. It should be noted that for switching between these two strategies,only weight of effective factors in our fire priority assigning function would change.

5.2 Buiding selection

It is interesting that the zone chosen by the agents is always the same because they have the same information about the world. Now in each zone each agent has to choose a building to extinguish. Our building selection module consists of two major parts. The first part assigns a priority to each ignited building in the zone and the second part decides which building to extinguish considering how easy is extinguishing this building for this FireBrigade.

In the first part, just the properties of buildings are regarded. We are using a parameterized learning method[5] to find effectiveness weight of each property, to achieve best result. Our selected parameters for learning are:

- Area of the building
- Number of floors
- Type of the building
- Cycles passed from building's ignition time
- Area of buildings around which are not extinguished
- Area of the smallest building around which is not extinguished
- Area of buildings around which have fieryness between 1 and 3
- Distance from the burning buildings
- Distance from unburned buildings

Our FireBrigades are not doing well enough currently but we are certain that they will improve day by day.

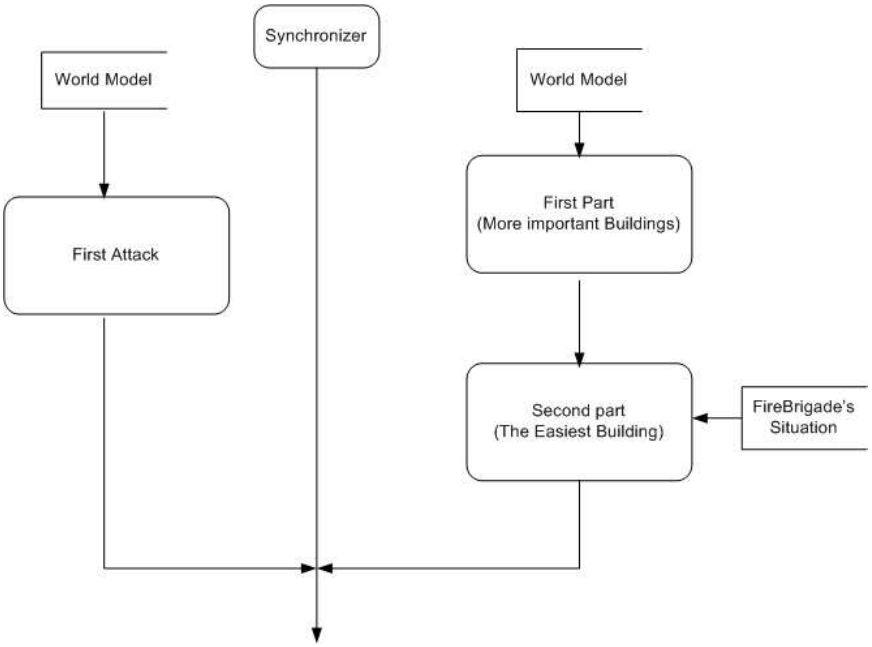

Fig. 8. Fire Brigade decision making data flow

In the second part, each FireBrigade decides which building is easiest and more important to extinguish. We consider parameters such as the buildings distance from the FireBrigade, Number of buildings behind available to be used for extinguishing and the amount of water the FireBrigade has.

Future Works: 1-Currently our FireBrigades do not consider number of civilians who will die because of each fire zone in their zone selection but we are going to implement it for Bremen. 2-We are going to compute the optimal number of FireBrigades to be dispatched for extinguishing of reignited fire zones.

References

1. Amirpour et al: SOS 2004:An Attemp Toward a multi-agent rescue team(2004)
2. Witten, I; Frank, E: Data Mining: Practical Machine Learning Tools and Techniques with Java Implementations Morgan Kaufmann (1999)
3. Takagi, T; Sugeno, M:
   Fuzzy identification of systems and its applications to modeling and control IEEE Transactions on Systems, Man, and Cybernetics. Vol.**15**,Jan.-Feb.116–132, (1985).
4. Jang, J. S. R.:
   ANFIS: Adaptive-Network-Based Fuzzy Inference System IEEE TRANSACTIONS ON SYSTEMS MAN AND CYBERNETICS , Vol.**23**; pages 665,(1993)
5. Sutton, RS., Barto, AG.:
   Reinforcement learning: an introduction Cambridge University Press (1999)