CSEC 18th - Past Year Paper Solution *2017-2018 Sem2*
CZ4046 – Intelligent Agents

By:

Agent1: SOH GUO HAO, STEFFANO	Agent2: Rain Chua Qin Lei
SOHG0007@e.ntu.edu.sg	CHUA0740@e.ntu.edu.sg

1. (a)

Expert systems can be seen a system with 'expertise' about some (abstract) domain of discourse, for example, a system to detect blood diseases. One example of an expert system is MYCIN, it knows about blood diseases in humans, it has knowledge about blood diseases in the form of rules. A doctor can obtain expert advice about blood diseases by giving MYCIN facts, questions, and posing queries.

An agent is autonomous, smart and active. Agents perform tasks or achieve objectives based on rewards, they are able to make decisions which is in favour of obtaining the maximal rewards for themselves. Agents are selfish and will only perform actions or make decisions that yields maximal reward for them in view of achieving a certain objective.

The main differences between Agents and Expert Systems are that agents are proactive, agents will perform tasks proactively to achieve its goals, whereas an expert system will not, an expert system will only listen to requests given by the user. An Expert system only has knowledge about its specific domain, it is not aware of its surroundings, as compared to Agents who are intelligent, and are aware of what is happening around it. Expert systems seldom perform actions to achieve its goal, its goal is mainly to satisfy the end-user's request, Agents on the other hand are proactive and will take necessary actions which yields maximal gain to them, Agents may also disobey a user if the action requested by the user does not yield sufficient reward for the agent.

(b)

"Intelligence", by definition is the ability to make decisions which yields maximal gain to the decision maker. With the increase of processing power and data storage within a computer, "Intelligence" has led to the emergence of the field of multi-agent systems. One example of intelligence would be that computers can now analyse data and provide a spectrum of outcomes / decisions available and of which yields the corresponding utility of each decision, meaning computers, given a scenario can now analyse and come up with different scenarios/decisions which can help in decision making. The tasks/decisions automated and delegating to computers are getting more and more complex.

(c)

The first type of utility function is to associate a utility value to every possible state available in an environment. The objective of the agent is to bring out the sequence of states that maximizes the utility value. Mathematically we can see the association below:

$$u : E \rightarrow \mathbb{R}$$

The Utility (u) is assigned when we map a specific Environment state (E) to a Real Number (R).

When we take this approach, several doubts may surface when we start to make decisions on which runs to take, what should we use as a basis of comparison between runs? Should we use Max utility? Min Utility? Average Utility? It is also difficult to get a long-term view when assigning utilities to individual states as every action taken may affect the utility of the states, one possible solution is to introduce a discount factor for the later states.

One algorithm that utilizes this concept is value iteration, which uses the maximum utility of state sequences to decide which actions to take. The Algorithm is :

Start out with every state(s) ,U(s) = 0

Iterate until convergence

During the i-th iteration, update the utility of each state according to this rule:

$$U_{i+1}(s) \leftarrow R(s) + \gamma \max_{a \in A(s)} \sum_{s'} P(s' \mid s, a) U_i(s')$$

The Second type of utility function is to assign utilities to individual runs, rather than states. With the similar objective to determine run which maximizes the utility gain. The Association is:

$$u : R \rightarrow \mathbb{R}$$

The Utility (u) is assigned when we map a specific Run (r) to a real Number (R). Such an approach takes an inherently long-term view. It also incorporates the probabilities of different states emerging. Some challenges are that, It is difficult to formulate the real numbers based on jus the runs itself and it is also difficult to formulate task in these terms.

One Algorithm which utilizes this approach is Policy Iteration, which uses the maximum utility of runs to decide which actions to take. The Algorithm is:

Start with some initial policy π^0 and alternate between the following steps:

Policy evaluation: calculate $U\pi^i(s)$ for every state s

$$U(s) = R(s) + \gamma \max_{a \in A(s)} \sum_{s'} P(s' \mid s, a) U(s')$$

Policy improvement: calculate a new policy πi+1 based on the updated utilities

$$\pi^{i+1}(s) = \arg\max_{a \in A(s)} \sum_{s'} P(s' \mid s, a) U^{\pi_i}(s')$$

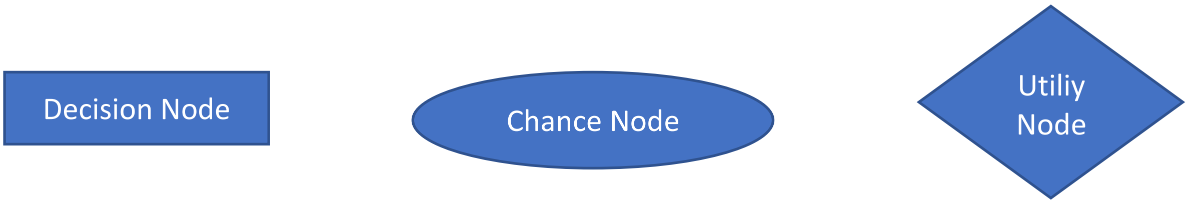
(d)

Achievement tasks are tasks that requires an agent to achieve a certain state. An achievement task is specified by a set of {G}, Goal states, an agent is successful as long as it achieved a state within set {G}, we do not consider which is optimal, the main goal is to achieve a state within the set {G}.

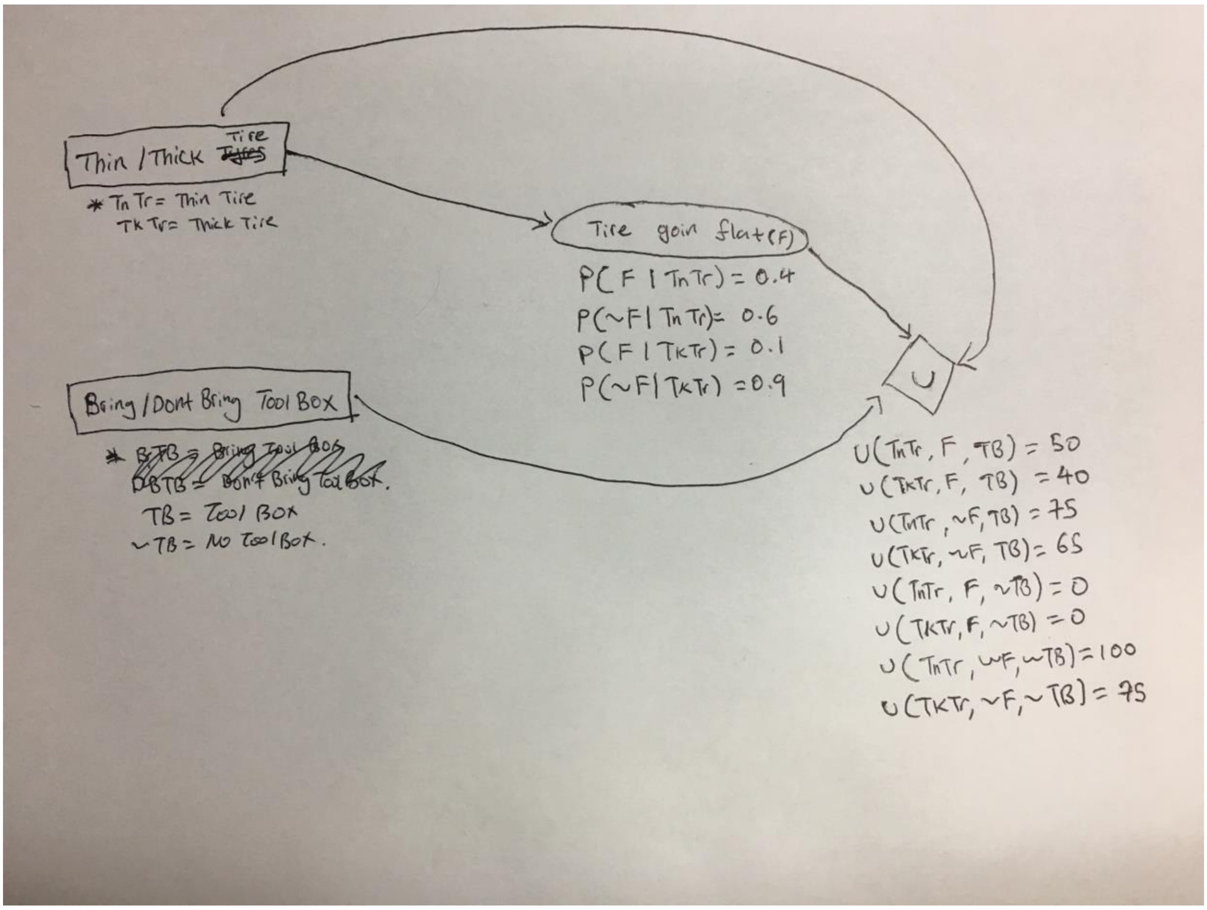
Maintenance tasks are tasks that requires an agent to avoid a certain state. A Maintenance task is specified by a set of {B}, bad states, an agent is successful as long as it manages to avoid any states within {B}. Agent is successful if it does not perform any action which will result is any states that exists in {B}.

2. (a)

The Three types of nodes which may exist in a decision network are:



The Decision Network:



(b)

To find the optimal decision, we first need to calculate the expected utility of all possible scenarios:

EU(TnTr,TB)	EU(TnTr,~TB)	EU(TkTr,TB)	EU(TkTr,~TB)

EU(TnTr,TB):

(P(F|TnTr) * U(TnTr,F,TB)) + (P(~F|TnTr) * U(TnTr,~F,TB))
(0.4 * 50) + (0.6 * 75)
= 65

EU(TnTr, ~TB):

(P(F|TnTr) * U(TnTr,F,~TB)) + (P(~F|TnTr) * U(TnTr,~F,~TB))
(0.4 * 0) + (0.6 * 100)
= 60

EU(TkTr,TB):

(P(F|TkTr) * U(TkTr,F,TB)) + (P(~F|TkTr) * U(TkTr,~F,TB))
(0.1 * 40) + (0.9 * 75)
= 71.5

EU(TkTr, ~TB):

(P(F|TkTr) * U(TkTr,F,~TB)) + (P(~F|TkTr) * U(TkTr,~F,~TB))
(0.1 * 0) + (0.9 * 75)
= 67.5

Optimal Decision is to use the thick tire and bring the tool box.

3. (a)

With the Constructionist perspective, we can use a high level artificial Language for agent co-operation. This allows a secure mean of communication as only agents will be able to understand the language, enemies may require some time to decipher the language even when they get hold of the message. Agents can use this language to co-operate and work amongst themselves and provide an integration of UAS and UGS. As resources may be limited during times of conflicts, we can adopt the operating in a "satisficing" mode concept, so that agents are able to perform their best with the limited resources available. As agents are bound to get damaged, we must ensure there is an organizational relation among agents so that we are able to re-supply and scale agent numbers. Agents must also be able to adapt to the changes in the environment as the war environment is ever changing.

(b)

Below are the 5 steps of contract net which I will adopt to aid me in the organization of the annual celebration.

Recognition: Firstly, I will need to recognize that I am not able to complete this task alone, or that I do not gain a positive reward if I were to complete this task alone.

Announcement: I will send out an announcement to the members of the organization, the announcement should contain the various task's specification, such as a description of the task, weather if there are any constraints such as deadline or quality and lastly the meta-task information, which is when to reply.

Bidding: Members of the organization will then decide if they are able to help me with it and also if the rewards for helping is in their favour. If members decide to help me, they are to submit a tender.

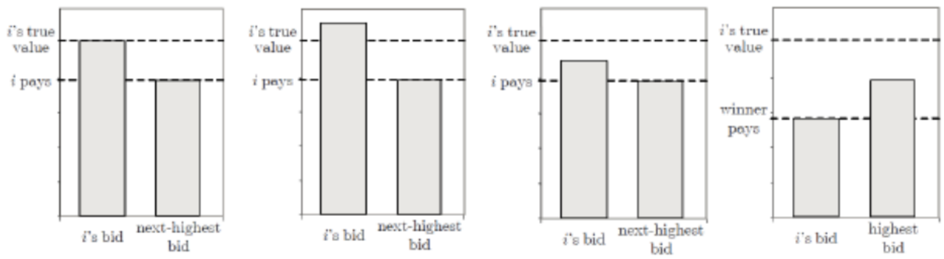
Awarding: after receiving the tenders, I would decide who is more suitable for the job and which assignment will yield the highest utility for myself, after which I will send a contract to the members which I select.

Expediting: Members who are awarded a contract to start working on their job, another contract net may happen if members deemed that they require help on their sub-tasks.

(c)

Suppose you bid more than your valuation, you may win the bid, If you do, you may end up paying more than you think the good is worth. Suppose you bid less than your valuation, you stand less chance of winning the good. So there is no point in bidding above or below your valuation.

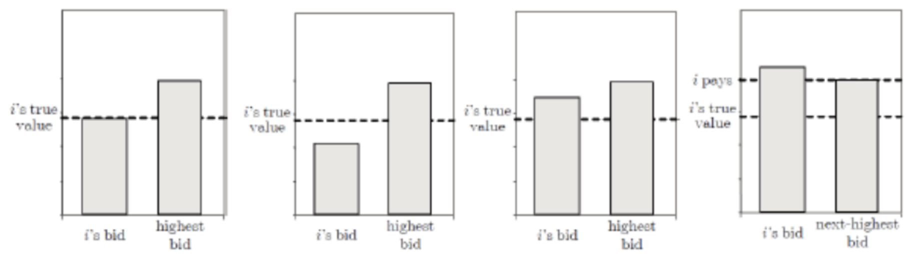
Second-Price proof 1 (Bidding honestly, i would win the auction)



Bidding honestly, assume i is the winner, If i bids higher, he will still win and still pay the same amount ,If i bids lower, he will:

- Either still win and still pay the same amount
- Or lose and get utility of zero

Second-Price proof 2 (Bidding honestly, i would lose the auction)



Bidding honestly, assume i is not the winner. If i bids lower, he will still lose and still pay nothing If i bids higher, he will

- Either still lose and still pay nothing
- Or win and pay more than his valuation

(d)

if (x1,x2,….,xn) is in the core and P1 is dummy -> x1 = 0 (prove)

since (x1,x2,..,xn) is in the core, that means that there are no sub coalition that can reasonably object to it… so…

(x1,x2,….,xn) >= (x2,x3,..,xn)

& (x1,x2,….,xn) >= (x1,x3,..,xn)

& ….

& (x1,x2,…xn) >= (x1,x2,…,xn-1).

Since P1 is a dummy, which means he cannot help or harm the coalition, this in terms reflect that…

(x1,x2,….,xn) = (x2,x3,…xn)

Meaning…
(V(x1) + V(x2) +…+V(xn)) = (V(x2) + V(x3) + … + V(xn))

to satisfy the above, x1 needs to be 0.

4. (a)

Matrix 1 - > No Dominant Strategy for Player A & B
Matrix 2 -> No Dominant Strategy for player A & B

(b)

Matrix 1 - > No Nash Equilibrum
Matrix 2 -> (A:Middle, B:Right)

(A:Middle, B:Right): in this equilibrium, if A stays with playing middle, B Does not Gain more pay off if he changes from playing right to left. And reflectively, if B stays with playing right, A does not get an increase of pay off if he changes from playing Middle to either playing up or down.

(c)

Matrix 1 -> (A:up,B:Left)
(A:down,B:left)

Matrix 2 -> (A:up, B:right)
(A:Middle, B:right)
(A:down, B:left)

For all states listed above, we are not able to find any other states that increases the Payoff of one player without decreasing the payoff of another. Thus Yielding the Praetor optimal property.

(d)

Matrix 1 -> (A:down, B:left)

matrix 2 -> (A:down, B:left)
(A:Middle, B:right)
(A:up,B:right)

For the states listed above, the sum of total payoff of both players A and B are the Highest amongst the different states within the payoff matrix thus, maximizing social Welfare.

For reporting of errors and errata, please visit pypdiscuss.appspot.com
Thank you and all the best for your exams! ☺