Xi'an Jiaotong-Liverpool University
西交利物浦大学

CPT 302 Individual Project
Coursework/Assignment Submission Form

2021/22 Semester 2
Bachelor's degree – Year 4

Module Code	Module Leader	Module Title
CPT302	**Ka Lok Man**	**Multi-Agent Systems**

Section A: Your Details

To be completed by the student (in English using BLOCK CAPITALS)

Student's Name	Tianlei Shi
Student ID	**1824152**

Section B: Assignment Details

To be completed by the student (in English using BLOCK CAPITALS)

Coursework Assignment Number	**Assignment I**
Coursework (assignment) Title	**Assignment I**
Method of Working	**INDIVIDUAL**
Date and time of submission	**April 22, 2022**

Assignment details can be found in the assignment description.

Section C: Statement of Academic Honesty

To be completed by the student

By submitting this coursework for assessment, you are confirming that you have read and understood the University's policy on plagiarism and collusion and that the submitted work is your own.

(i) I confirm that I have read a copy of the current University's definitions of collusion and plagiarism on coursework and academic honesty, and that I fully understand the meaning of these terms.

(ii) I confirm that the submitted coursework has been created solely by me and that I have not been assisted, nor have copied part or all of somebody else's work, either with their explicit approval or without their knowledge or consent.

(iii) I confirm that this is my own work and that use of material from other sources, including the Internet, has been properly and fully acknowledged and referenced.

(iv) I confirm that the information I have given is correct to the best of my knowledge.

Student's signature	Tianlei Shi	Date	Apr. 22, 2022

If this form is submitted electronically, please type your name in English (BLOCK CAPTIALS)

Q1,

Solution:

1) Possible percepts: Per = {dirt, clean, hole}, which means the environment is dirt/clean/hole.

   Domain predicates describing internal state:
   In(x, y) – agent is at (x, y)
   Near(x, y) – agent can move to (x, y) by only one step
   Dirt(x, y) – there is dirt at (x, y)
   ObserveDirt(x, y) – there is dirt at (x, y) observed by agent
   ObserveHole(x, y) – there is hole at (x, y) observed by agent
   IsAvoid(x, y) – agent should not move to (x, y)
   AllTravel – all positions have been traveled or observed

   Possible actions: Act = { Move(x, y) – agent moves to (x, y)
   Suck – suck dirt
   Observe – observe all adjacent cells
   MoveBack(x, y) – agent move back to origin (x, y)
   Avoid(x, y) – delete (x, y) from locations that agent possible moving to, i.e., make (x, y) become IsAvoid(x, y)}

   Deduction rules:
   In(x, y) $\wedge$ Dirt(x, y) -> Do(Suck)
   In(x, y) $\wedge$ $\neg$ Dirt(x, y) -> Do(Observe)
   Near(x, y) $\wedge$ ObserveHole(x, y) -> Do(Avoid(x, y))
   In(x, y) $\wedge$ $\neg$ Dirt(x, y) $\wedge$ ObserveDirt($\neg$ x, $\neg$ y) $\wedge$ $\neg$ IsAvoid($\neg$ x, $\neg$ y) $\wedge$ $\neg$ AllTravel -> Do(Move($\neg$ x, $\neg$ y))
   In(x, y) $\wedge$ $\neg$ Dirt(x, y) $\wedge$ $\neg$ IsAvoid($\neg$ x, $\neg$ y) $\wedge$ $\neg$ AllTravel -> Do(Move($\neg$ x, $\neg$ y))
   In(x, y) $\wedge$ $\neg$ Dirt(x, y) $\wedge$ AllTravel -> Do(MoveBack($\neg$ x, $\neg$ y))

   In the Vacuum World, the robot R could move to adjacent four cells (up, down, left, and right), and it can also observe adjacent cells to detect whether there is dirt or hole (ObserveHole, and ObserveDirt). Moreover, robot maintains a database, which record cells that have traveled or observed (AllTravel is get from this), and cells that should be avoided (IsAvoid).

2) Clean task start

In(2, a) ∧ ¬ Dirt(2, a) -> Do(Observe)

In(2, a) ∧ ¬ Dirt(2, a) ∧ ¬ IsAvoid(2, b) ∧ ¬ AllTravel -> Do(Move(2, b))

In(2, b) ∧ ¬ Dirt(2, b) -> Do(Observe)

In(2, b) ∧ ¬ Dirt(2, b) ∧ ObserveDirt(1, b) ∧ ¬ IsAvoid(1, b) ∧ ¬ AllTravel -> Do(Move(1, b))

In(1, b) ∧ Dirt(1, b) -> Do(Suck)

In(1, b) ∧ ¬ Dirt(1, b) -> Do(Observe)

In(1, b) ∧ ¬ Dirt(1, b) ∧ ¬ IsAvoid(1, c) ∧ ¬ AllTravel -> Do(Move(1, c))

In(1, c) ∧ ¬ Dirt(1, c) -> Do(Observe)

In(1, c) ∧ ¬ Dirt(1, c) ∧ ¬ IsAvoid(1, d) ∧ ¬ AllTravel -> Do(Move(1, d))

In(1, d) ∧ ¬ Dirt(1, d) -> Do(Observe)

Near(1, e) ∧ ObserveHole(1, e) -> Do(Avoid(1, e))

In(1, d) ∧ ¬ Dirt(1, d) ∧ ObserveDirt(2, d) ∧ ¬ IsAvoid(2, d) ∧ ¬ AllTravel -> Do(Move(2, d))

In(2, d) ∧ Dirt(2, d) -> Do(Suck)

In(2, d) ∧ ¬ Dirt(2, d) -> Do(Observe)

In(2, d) ∧ ¬ Dirt(2, d) ∧ ¬ IsAvoid(3, d) ∧ ¬ AllTravel -> Do(Move(3, d))

In(3, d) ∧ ¬ Dirt(3, d) -> Do(Observe)

In(3, d) ∧ ¬ Dirt(3, d) ∧ ObserveDirt(3, c) ∧ ¬ IsAvoid(3, c) ∧ ¬ AllTravel -> Do(Move(3, c))

In(3, c) ∧ Dirt(3, c) -> Do(Suck)

In(3, c) ∧ ¬ Dirt(3, c) -> Do(Observe)

In(3, c) ∧ ¬ Dirt(3, c) ∧ ¬ IsAvoid(4, c) ∧ ¬ AllTravel -> Do(Move(4, c))

In(4, c) ∧ ¬ Dirt(4, c) -> Do(Observe)

Near(4, b) ∧ ObserveHole(4, b) -> Do(Avoid(4, b))

In(4, c) ∧ ¬ Dirt(4, c) ∧ ¬ IsAvoid(4, d) ∧ ¬ AllTravel -> Do(Move(4, d))

In(4, d) ∧ ¬ Dirt(4, d) -> Do(Observe)

In(4, d) ∧ ¬ Dirt(4, d) ∧ ¬ IsAvoid(5, d) ∧ ¬ AllTravel -> Do(Move(5, d))

In(5, d) ∧ ¬ Dirt(5, d) -> Do(Observe)

Near(5, e) ∧ ObserveHole(5, e) -> Do(Avoid(5, e))

In(5, d) ∧ ¬ Dirt(5, d) ∧ ¬ IsAvoid(5, c) ∧ ¬ AllTravel -> Do(Move(5, c))

In(5, c) ∧ ¬ Dirt(5, c) -> Do(Observe)

In(5, c) ∧ ¬ Dirt(5, c) ∧ ¬ IsAvoid(5, b) ∧ ¬ AllTravel -> Do(Move(5, b))

In(5, b) ∧ ¬ Dirt(5, b) -> Do(Observe)

In(5, b) ∧ ¬ Dirt(5, b) ∧ ObserveDirt(5, a) ∧ ¬ IsAvoid(5, a) ∧ ¬ AllTravel -> Do(Move(5, a))

In(5, a) ∧ Dirt(5, a) -> Do(Suck)

In(5, a) ∧ ¬ Dirt(5, a) -> Do(Observe)

In(5, a) ∧ ¬ Dirt(5, a) ∧ AllTravel -> Do(MoveBack(2, a))

Clean task finish

3) To discuss whether the set of rules is the optimal, we need to establish evaluation criteria. Therefore, we use a utility function to evaluate the run of agent. The utility function encourage agent to clean the most dirt with the least amount of movement, and a higher score indicates stronger agent performance, and poorer agent performance otherwise.

Let u(r) be the utility function, c_d is the number of dirt cleaned by agent, t_d is the number of dirt appeared in Vacuum World, and n_s is the number of step that agent moved (or the number of cell that agent traveled). So,

$$u(r) = \left(\frac{c\_d}{t\_d}\right) + n\_s^{-1}$$

In the example of question 2), our agent moved 14 steps, and cleaned all dirt, so the score is u(r) = 1 + (1 ÷ 14) ≈ 1.07.

In other case, an agent cannot observe, which means the agent only detect dirt in current cell. This agent is easy to fall into a hole, and should travel all cells to clean all dirt. So, its score is u(r) = 1 + (1 ÷ 25) = 1.04.

Additionally, if agent only clean little dirt to save step, for example, agent only clean dirt in (1, b). The agent only moves three step, and clean one dirt. So, its score is u(r) = (1 ÷ 4) + (1 ÷ 3) ≈ 0.58.

Therefore, our rules are the optimal.

Q2,

Solution:

1) Symbolic approach: symbolic reasoning agents use logical formulae (or predicate logic) to represent the environment and action, and use logical deduction (or syntactic manipulation) to perform explicit logical reasoning to decide what to do. Moreover, the problems of symbolic approach are: a) symbolically represent a complex real-world is difficult, b) decision making by theorem proving is complex and may never terminate.

   Reactive approach: reactive agents do not need explicit representations and abstract reasoning, and it make decision by a set of task-accomplishing behaviors, and react to the environment directly. In reactive agent architecture (or subsumption architecture), behaviors arranged into layers, and lower layer can inhibit higher layers, but many behaviors can fire simultaneously. Moreover, the limitations of reactive approach are: a) build agents that contain many layers (> 10) is difficult, b) the relationship between individual behaviors, environment and overall behavior is not understandable.

   Hybrid approach: hybrid agents attempt to combine the best of symbolic and reactive architectures, and it build an agent out of two subsystems, say deliberative and reactive. In hybrid agent architecture, those subsystems are arranged into two hierarchies with interacting layers, say Horizontal layering and vertical layering (including one pass and two pass).

2) a) TouringMachines and InteRRap are both hybrid agent, but TouringMachines consists of three horizontal activity-producing layers, and InteRRap uses a vertically layered two-pass architecture.

   b) TouringMachines and InteRRap both have three layers.
   The layers in TouringMachines are modelling layer, planning layer, and reactive layer. Modelling layer represent entities in the world, and resolve conflicts between agents by generates new goals (goals are posted down to planning layer). Planning layer achieves the agent's proactive behavior, constructs library of predefined plan schemas, and find schemas that matches goal in the library. These two layers are

performed deliberative (develop plans and make decisions), and the last layer performed reactive (reacting to events without complex reasoning). The reactive layer provides immediate response to changes that occur in environment, and which realized as a set of condition action rules. Moreover, there is a control subsystem in TouringMachines, and it responsible for deciding which layer should control the agent at any given time, and this is implemented as a set of control rules.
However, there are different three in InteRRap: behavior-based layer, local planning layer, and cooperative planning layer. In which, behavior-based layer deals with reactive behavior, local planning layer deals with planning to achieve the agent's goal, and cooperative planning layer deals with social interactions, and each layer has an associated knowledge base (to represent the world that appropriate for the layer) additionally.

c) For TouringMachines, the advantages are: i) in horizontal layered architecture, layers are directly connected to the sensory input and action output, and which makes TouringMachines can react fast. ii) each layer acts like an agent, and make decision by control subsystem. This combines the benefits of symbolic and reactive approach, and makes TouringMachines robust and interpretable.
The disadvantages are: i) there is no layer responsible for coordinating or negotiating or in TouringMachines, and which will make TouringMachines hard to cooperate with other agents. ii) the architecture of TouringMachines is relatively simple, and which means that it is hard for TouringMachines to understand and deal with complex or abstract problems. iii) in this sort of architecture, the designer must potentially consider all possible interactions between the layers, and this limits the size of TouringMachines.

For InteRRap, the advantages are: i) the cooperative planning layer in InteRRap can make it social with other agents, and negotiate to complete works. ii) InteRRap uses vertically layered two-pass architecture and associated knowledge base to represent the world, and which make InteRRap can understand and deal with complex or abstract problems.
The disadvantage is: there exists layer interactions and control flow in InteRRap, and this will spend much time. Therefore, InteRRap needs more time to make a decision.

3) For instance, an automated ambulance (agent) needs to travel across the city to pick up patients and take them back to the hospital as quickly as possible, traffic safety should also be guaranteed. This application could be fully applied by TouringMachines but not for InteRRap, because this application needs agents to react quickly to the environment, and TouringMachines fit this requirement perfectly, because the horizontal layered architecture allows it to react almost in real time. However, due to layer interactions and control flow, the vertically layered two-pass architecture of InteRRap spend much time to react, and it can easily lead to traffic accidents.

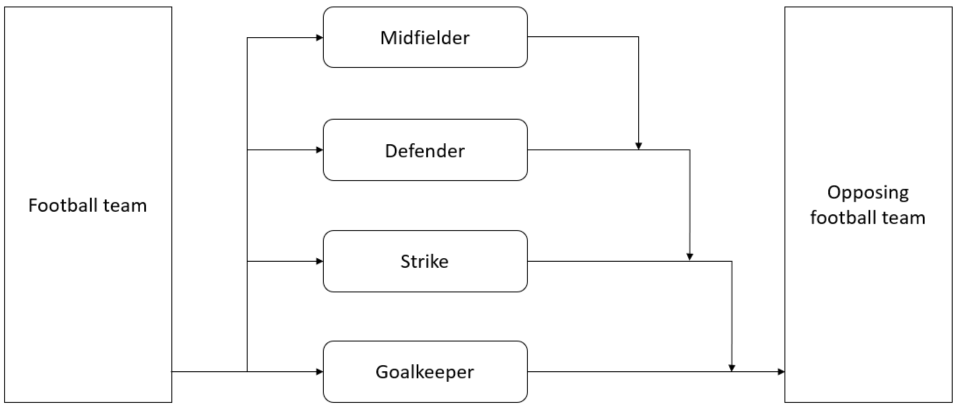
4) For example, a task requires a group of automated forklifts (agents) to load and unload trucks in a loading dock, and agents need to cooperate, coordinate, and negotiate with each other to complete this task. This application could be fully applied by InteRRap but not for TouringMachines, because: a) this application needs agents to social with each other to complete work efficiently, and the cooperation layer in InteRRap can solve social-related problems. However, there is no similar module in TouringMachines, and the modelling layer can only predict conflicts between agents but cannot understand other agent's intentions, which cannot satisfy the demand of application. b) this application needs agents to consider other agents' intentions, and reach consensus by consultation, and this require agents have the ability of coordinate multiple intentions and achieve complex goals. The vertically layered two-pass architecture and associated knowledge base makes InteRRap to understand other agents' intentions and achieve complex goals. But the relatively simple horizontal layered architecture of TouringMachines is hard to accomplish such a complex task.

Q3,

Solution:

A small football play contains two teams, and each team consists of same four players: Goalkeeper, Defender, Midfielder, and Strike. Therefore, the same architecture and rules can apply to both sides.

In the designed subsumption architecture, we take players as modules to construct subsumption hierarchy, with their behaviors that are arranged into layers.



The subsumption architecture is shown as above. In a football play, when the Goalkeeper conduct an action, it means that the opponent is about to goal, and in this moment, other players' actions are no longer important, thus, we put the Goalkeeper on the lowest layer (the first layer). The aim of football play is to score goals, so we put the Strike on the second layer so that the player can try to score untrammeled. Next, the Defender can stop attacks and prevent opposing team from scoring goals, and Defender's role is more important compare to Midfielder, so we put the Defender on the third layer. Finally, we put the Midfielder who can provide link between attack and defense on the last layer. Therefore, the inhibition relationship between player is:

$$Goalkeeper \prec Strike \prec Defender \prec Midfielder$$

What's more, behaviors in each module are discussed in following.

For Goalkeeper:

g0: if ball in the rectangular penalty area and have a right opportunity then stop the ball

g1: if ball is serviced and no another player touches the ball then do not touch ball with hands again

g2: if teammate return a pass then do not catch by hands

g3: if hold the ball nearly or equals to six seconds then pass the ball (or service) g4: if the ball appears around then pass the ball

g5: if true then wait in the rectangular penalty area

The above behaviors are arranged into the hierarchy: $g0 \prec g1 \prec g2 \prec g3 \prec g4 \prec g5$.

For Strike:

s0: if in the opposing half pitch and position nearly or equals to the second last player of opposing team then do not move forward

s1: if hold the ball and have right opportunity to shoot then shoot the goal

s2: if opponent is dribble and have right opportunity then steal the ball

s3: if hold the ball and no threats then dribble toward the opponent's goal

s4: if hold the ball and have right opportunity to pass then pass the ball

s5: if true then move in pitch

The above behaviors are arranged into the hierarchy: $s0 \prec s1 \prec s2 \prec s3 \prec s4 \prec s5$.

For Defender:

d0: if in the opposing half pitch and position nearly or equals to the second last player of opposing team then do not move forward

d1: if opponent is dribble and have right opportunity then stop attack

d2: if hold the ball and have right opportunity to shoot then shoot the goal

d3: if hold the ball and have right opportunity to pass then pass the ball

d4: if hold the ball and no threats then dribble toward the opponent's goal

d5: if true then move in pitch

The above behaviors are arranged into the hierarchy: $d0 \prec d1 \prec d2 \prec d3 \prec d4 \prec d5$.

For Midfielder:

m0: if in the opposing half pitch and position nearly or equals to the second last player of opposing team then do not move forward

m1: if status of attack and defense changes then provide support

m2: if hold the ball and have right opportunity to shoot then shoot the goal

m3: if opponent is dribble and have right opportunity then steal the ball

m4: if hold the ball and no threats then dribble toward the opponent's goal

m5: if hold the ball and have right opportunity to pass then pass the ball

m6: if true then move in pitch

The above behaviors are arranged into the hierarchy: $m0 \prec m1 \prec m2 \prec m3 \prec m4 \prec m5 \prec m6$.

Q4,

Solution:

In this question, we need to design a number of Internet agents that can work in cooperation to perform activities in an e-commerce platform (similar to Taobao), and the activities including plan – develop sales and replenishment plans, arrange – arrange the delivery or refund, buy – conduct replenish, sell – make sales, and negotiate transactions – negotiate with customers or suppliers.

Therefore, we designed five agents, say Manager, Seller, Buyer, Accountant, and Courier, and are shown as below.

Agent Name	Role	Functionality
Manager	manager all agents and develop plans	The objective of Manager is to ensuring all agents can function properly to achieve their objectives, and develop sales and replenishment plans based on inventory and funds. To do this, it needs to cooperate, coordinate, and negotiate with other agents.
Seller	make sales	The objective of Seller is to sale goods to customers according to sales plan. This objective requires it pricing reasonable based on funds, interact with customers, and negotiate transactions. To do this, Seller should cooperate closely with Accountant, Courier, and Manager to obtain funds information, arrange the delivery, and update sales plan.
Buyer	conduct replenish	The objective of Buyer is to conduct replenishment timely according to replenishment plan. This objective requires it quoting reasonable based on funds, interact with suppliers, and negotiate transactions. To do this, Buyer should cooperate closely with Accountant and Manager to obtain funds and goods information, and update replenishment plan.

Accountant	record goods and funds changes	The objective of Accountant is to record the changes of goods and funds faithfully, and manage all funds. Therefore, Accountant should cooperate closely with Seller, Buyer, and Manager to provide and update goods and funds information. Moreover, Accountant also responsible for payment and refund to suppliers and customers.
Courier	arrange the delivery	The objective of Courier is to arrange delivery of goods purchased by customers, and this objective requires it interact with customers. To do this, Courier should work closely with Seller and Accountant, to obtain the information of goods and customers, and update inventory information.

Those agents can perform activities mentioned above.
For plan - develop sales and replenishment plans: This task is completed mainly by Manager, and Manager will complete the task together with Seller, Buyer, and Accountant. Manager will obtain information from Seller (get goods prices), Buyer (get quotes), and Accountant (get funds and in inventory), and negotiate and reach consensus with them to finally develop plans. This task demonstrates the social and proactive behavior of Manager, and the reactive behavior will demonstrate later (in buy and sell).
For arrange – arrange the delivery or refund: This task is completed mainly by Courier, and Courier will complete the task together with Seller and Accountant. Courier will obtain information from Seller (get goods information and address of customer), and inform Accountant to update inventory (reduce inventory of corresponding items) or funds (reduce corresponding number of funds) after shipment or refund. This task demonstrates the social and proactive behavior of Courier, and the reactive behavior of Accountant is demonstrated as well.
For buy – conduct replenish: This task is completed mainly by Buyer, and Buyer will complete the task together with Accountant and Manager. Buyer will obtain information from Accountant (get funds), notify Accountant to pay when the replenishment is finish, and report Accountant and Manager the number of goods bought. This task demonstrates the social and proactive behavior of Buyer, and the reactive behavior of Manager is demonstrated as well.

For sell – make sales: This task is completed mainly by Seller, and Seller will complete the task together with Accountant, Courier, and Manager. Seller will obtain information from Accountant (get funds), and inform Accountant (update/increasing corresponding number of funds), Courier (goods information and address of customer), and Manager (update sakes plan) after deal. This task demonstrates the social and proactive behavior of Seller, and the reactive behavior of Courier is demonstrated as well.
For negotiate transactions – negotiate with customers or suppliers: This task is completed mainly by Seller or Buyer. The Seller or Buyer needs to interact with customers or suppliers, When the transaction cannot be smoothly concluded, or the customers or suppliers has objections. This task demonstrates the reactive behavior of Seller or Buyer.

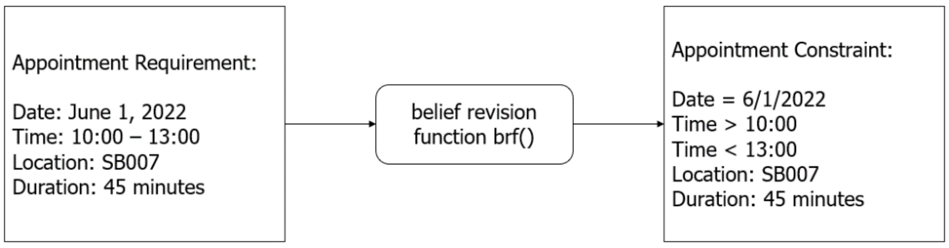
As activities described above, the agents cooperate together to solve problems is much efficiently than the capabilities of any individual agent. Individual agent to solve problems may face the issues such as information shortage, single thread to process large data will take more time, and high coupling. However, agents' cooperation led to more comprehensive and detailed data, low coupling, more efficient and may find better solutions.

Q5,

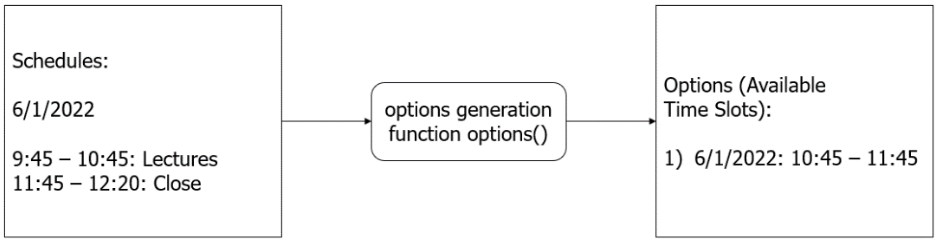
Solution:

To begin with, agent have an initial intention *Intentions*$_0$ = *{appointmentMeeting}*, and an initial belief *beliefs*$_0$ = *{Ø}*.

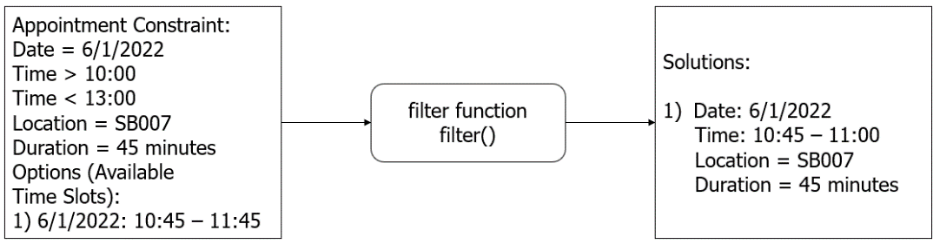
Firstly, agent conducts belief revision function to update beliefs, *brf:* $2^{Bel} \times Per \rightarrow 2^{Bel}$.



Next, agent conducts option generation function to produce desires, *options:* $2^{Bel} \times 2^{Int} \rightarrow 2^{Des}$.



Then, agent conducts filtering function to produce intentions, *filter:* $2^{Bel} \times 2^{Des} \times 2^{Int} \rightarrow 2^{Int}$.



Finally, agent perform action selection function (or means-ends reasoning) to determine an action and carry out to achieve intentions.