Chapter 3
cBDI: Extended BDI agent for Human-Agent Collaboration

"Gettin' good players is easy. Gettin' 'em to play together is the hard part"

Casey Stengel

Contents

3.1 Introduction

Human–machine collaboration necessitates active involvement of both the human and the machine in accomplishment of a task. To support collaboration, the machine should have a set of processes that enable it to understand and perform task solving in ways similar to that of humans [21]. It was Ferguson's view that an *agent* can serve as the basis for such machines [185]. In this chapter, work is

presented which centers on how a traditional agent architecture can be extended to effectively collaborate with a human in the context of human–machine system.

The traditional architecture chosen here is the Belief–Desire–Intention agent. Within agent literature BDI paradigm is widely used to achieve human-like intelligence [186]. BDI can act as standalone substitute for human and human decision–making behaviour. This chapter presents how a traditional BDI agent architecture can be extended so as to enable collaboration with a human. This chapter first offers a justification of using the BDI architecture as a starting point for the human–machine teaming application and then presents why BDI needs extension to collaborate with human. Description of the extended BDI architecture, which is christened as cBDI agent, is presented. Formal structure of cBDI architecture and human–cBDI agent collaboration is presented. An example wherein a cBDI agent inhabits the Block-world domain is presented and the cBDI architecture is demonstrated to be adequate for collaborative task.

3.2 Why a BDI agent architecture?

As discussed in section 2.2.3.1 of Chapter 2, the Belief-Desire-Intention (BDI) architecture is one of the most well known and well studied software agents' architecture.

- It is based on a folk psychology–Bratman's theory of human reasoning [17]. This characteristic allows the BDI agent to imitate the human decision-making process and to be easily understood by the real human.
- BDI is a relatively mature framework.
- BDI has been successfully used in several practical applications [83], [187].
- Another attractive aspect of the BDI architecture is that it has been rigorously formalized [80], [188].
- The BDI agent can be integrated with any other agent-based system and also can be integrated with complex systems [101].

Above reasons make BDI agent to be one of the best options for agent implementation. However, adoption of the BDI architecture for human–machine teaming applications is largely motivated by the following remark by Urlings *et al.* [186]:

> *"Agents that implement BDI architecture have been suggested as of particular interest for human–agent teaming applications. They have in*

> *common with other agents the capability to provide a range of heterogeneous expertise and functionality. But their syntax is well aligned with the principles of cognitive teaming and they are well suited to provide a situation awareness capability to a human–machine team"*
>
> Urlings *et al.* [186, Page 109]

Urlings's statement highlights that BDI agent is aligned with *the principles of cognitive teaming* as well as *situation awareness* capability for a human-machine teaming. Further, several research efforts as mentioned in section 2.2.3.2, Chapter 2 contribute to the idea that extending BDI like agent for human agent collaboration holds promise.

We agree with [185]'s view that just because *as needed* a basic architecture should not be changed fundamentally. It was [186], where they mentioned that to accommodate human, BDI architecture needs some refinement or extension. From the mentioned fact of [186], the extension of BDI architecture for collaboration (with human) appears reasonably valid.

3.3 Why BDI architecture needs extension?

It is well known that collaborative agents have roots in the BDI logic [69]. Then why is it that BDI architecture needs to be extended for human-agent collaboration? This section tries to answer and provide justification for such an extension.

Even though BDI paradigm is used to achieve human-like intelligence, the paradigm does not emulate human-like qualities such as *coordination* and *learning* [186] or *collaborative planning*. The BDI agent is able to act on its individual *standard task planning*, which is insufficient for the agent to interact with a human in a collaborative context. To be a collaborative agent, BDI agent must have a concept of shared tasks. The agent to be a part of a human–agent team need to plan and perform activities jointly with the human. To perform jointly some of the cognitive elements need to be shared. Therein lies the motivation to extend the basic BDI agent model to include the mechanisms necessary for collaboration. Extension to basic BDI is based on the idea that agent knowledge of *human centric strategies* for a task is particularly crucial for agent to collaborate with human. To collaborate, cBDI agent need to follow a strategy that has the same intent as the human in the system. Extension to the basic BDI agent is undertaken with the two aims:

1. The first aim is to allow the agent to include *human centric strategy*: Knowledge of human strategy to solve a task plays a central role in the extended version of the classic BDI architecture.

2. The second aim is to allow the agent to execute human centric plan: To devise how a given goal can be accomplished, the agent has to produce a plan taking explicit account of human team mate.

The next section describes the proposed extension of the BDI architecture for supporting collaboration.

3.4 cBDI agent: Modules and Functions

Previous section discussed main concern over extension to classical Belief-Desire-Intention (BDI) model. BDI agent while performing collaborative task with human, needs to plan its actions by taking in to account what actions the human would perform (where human actions are likely to be based on the human's strategy). There is need of a specific mechanism within the BDI agent to maintain human centric strategy instances. Thus, the desirable features for such an architecture are:
(a) a set of human centric strategy.
(b) the mechanism to maintain strategy.
(c) ability to perceive the human's actions and internal representation of human intent.
(d) ability to plan agent's actions taking into account the various human strategies.

The cBDI architecture combines these concerns into the BDI architecture. cBDI plan its actions by taking in to account what actions the human would perform, by maintaining actions that are based on the human's strategy. The collaborative BDI agent is a BDI agent together with the strategic state planner module.

Figure 3-1 depicts extended version of the classic BDI architecture. The architecture is termed as *cBDI*, c here stands for collaboration.

Definition 1. cBDI agent: The cBDI agent $\mathcal{C}$ is given by a 6–tuple

$$< \mathcal{B}, \mathcal{D}, \mathcal{I}, A^c, m, \Pi >$$

where
$\mathcal{B}$ denotes set of all possible beliefs.
$\mathcal{D}$ denotes set of all possible desires.
$\mathcal{I}$ denotes set of all possible intentions.
A^c denotes all communicative action of the human. [1]; A^c: set of human actions.

[1]Human intent can be through either communication or performance of actions. In accordance with Lesh *et al.* [99] human and agent maintains mutual understanding by performance of actions

m is a set of agent modes.
Π denotes the the strategic state planner.

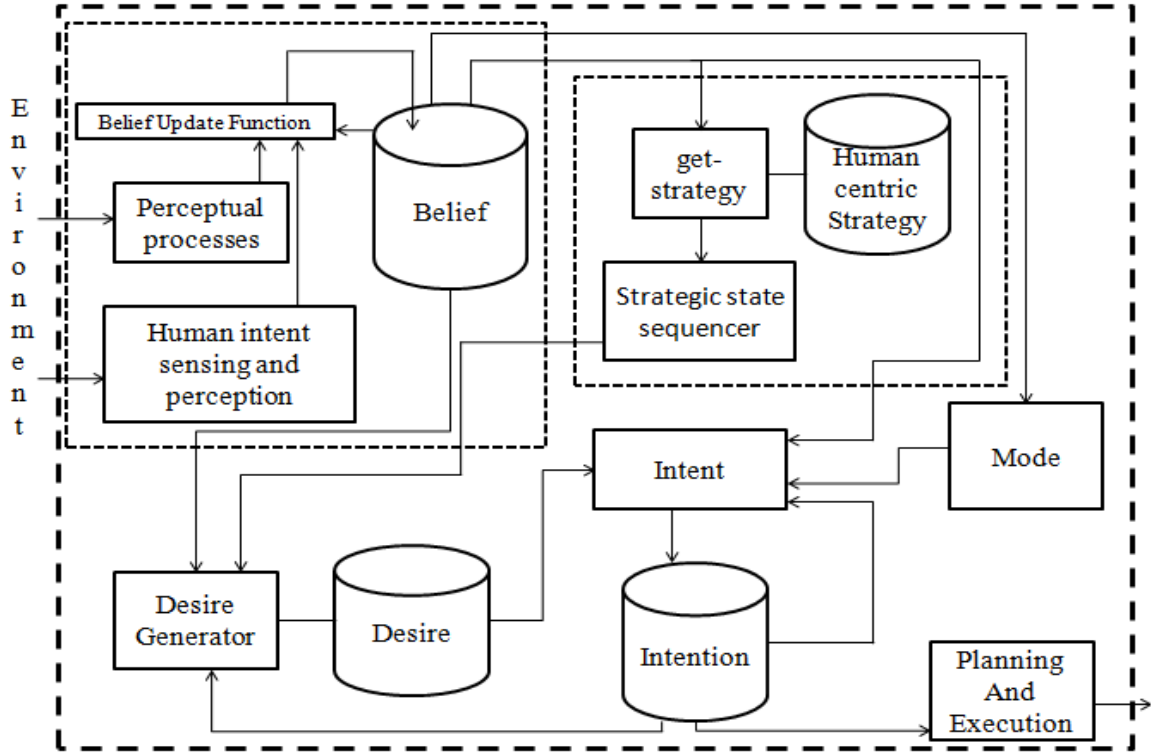

Figure 3-1: The cBDI Agent:

The following subsections present a description of important components of the underlying cognitive agent.

3.4.1 Beliefs

The Belief module maintains a set of beliefs $\mathcal{B}$, where each belief b $\in \mathcal{B}$ can describe either localizing information about self; set of executable action of agent; knowledge of human intent; capacity of human. The original BDI model uses only perception to acquire its beliefs. In cBDI architecture acquiring beliefs have been enhanced to *human intent sensing and perception module*; as to collaborate, there must be certain beliefs about the human abilities as well as knowledge about human intent. Further there is pro-prioception through a set of perceptual processes.

Definition 2 (Beliefs)**.** Belief is a 4-tuple $< \mathcal{B}_A, \mathcal{B}_S, \mathcal{B}_I, \mathcal{B}_h >$ where $\mathcal{B}_A$ denotes *Assumed belief*, $\mathcal{B}_S$ denotes *Basic belief*, $\mathcal{B}_I$ denotes *Interaction belief*, $\mathcal{B}_h$ denotes belief of human actions.

- Assumed belief $\mathcal{B}_A$ is agent's belief of its own capabilities (C_a) and a set of actions A_c. $\mathcal{B}_A$ are held *a-priori* held.

- Basic belief $\mathcal{B}_S$ is agent's belief about its surrounding environment. *Basic beliefs* includes map of the surrounding environment, the agent's position in the environment and the agent's orientation.
- Interaction belief $\mathcal{B}_I$ is agent's belief of human capacity (C_h).
- $\mathcal{B}_h$ is belief of human actions.

The Belief module manages two key processes: creating new beliefs, merge these new beliefs into existing beliefs.

For first of these processes, belief update functions are defined:
There are three belief update functions:

- self-aware($self_{aware}$)
- $Interaction$
- $human_{intent}$

Definition 3. self-aware($self_{aware}$): It is a function that translates information about the environment into agent's *Basic beliefs* ($\mathcal{B}_S$).

$$self_{aware} : W \rightarrow \mathcal{B}_S$$

where
Environment W, is directed graph defined as W=$< V, E >$ where
V is a set of vertices, and
E is a set of edges, E $\subseteq VXV$.

Graphs are well suited to model environment [147],[189]. We also use graph to represent the agent's environment. Where we conceptualized that the cBDI agent moves through its environment in such a way that at any given instant the agent is in exactly one location. Agent's transition from one location to other location takes place only along the edges. At each instant agent can move from one location to another. Then the cBDI agent's environment can be represented as the directed graph with nodes of the graph corresponding to distinct locations and the directed edge of the graph correspond to possible transitions for each location.

Definition 4. *Interaction*: It is a function that generates $\mathcal{B}_I$ through human *interaction.*

$$Interaction : M_{int} \rightarrow \mathcal{B}_I$$

where M_{int} is the set of human interaction and is formed by {{G}, {C_h}}
where,

G is task objective as defined in [190]
C_h denotes human capacity.

Definition 5. $human_{intent}$: *Human intent* function ($human_{intent}$) is designed to gather information about the human's actions. The $human_{intent}$ function translates human action to generate $\mathcal{B}_h$ through communicative action (A^c).

$$human_{intent} = A^c \rightarrow \mathcal{B}_h$$

Every action of A_c, upon execution, causes changes in the belief about the environment. The STRIP style structure [191] is applied to represent the agent actions. Each action consists of two STRIPS-style operator elements: the *Preconditions (precond)* list and the *Postconditions (postcond)* list.

Definition 6. [Action] An agent action $a_c \in A_c$ is a tuple $< precond, postcond >$, where *precond*is set representing precondition for a_c, *postcond* is a set representing consequence of executing a_c.

For the second key process, the Belief module manages to merge new beliefs into existing ones. We label this process Updating of belief $U_{\mathcal{B}}$. Updating of belief takes into consideration the current action, and revises the current beliefs based on previous beliefs, the set of belief obtained from the environment, communicative action of human and the set of belief obtained through human interaction.

Definition 7. Updating of belief $U_{\mathcal{B}}$: Updating of belief $U_{\mathcal{B}}$ is defined as

$$U_{\mathcal{B}} : (A_c \cup A^c) \times (\mathcal{B}_s \cup \mathcal{B}_I \cup \mathcal{B}_h \cup \mathcal{B}) \rightarrow \mathcal{B}$$

In sum, a complete Belief module update cycle proceeds as follows:

1. begin with current belief set
2. receive agent's surrounding environment from the perceptual process
3. create incoming belief set $\mathcal{B}_S$
4. receive goal information and human capacity from the human intent sensing and perception module
5. create incoming belief set $\mathcal{B}_I$
6. receive human action from the human intent sensing and perception module
7. create incoming belief set $\mathcal{B}_h$
8. Updating of belief $U_{\mathcal{B}}$:
   $\mathcal{B} \leftarrow U_{\mathcal{B}}(A_c \cup A^c, \mathcal{B}_s, \mathcal{B}_I, \mathcal{B}_h, \mathcal{B})$

3.4.2 Strategic planner

The "Strategic planner" is responsible for maintaining a *human-centric strategy* for a task objective. Belief specifies task objectives to the agent. Strategic planner module has an internal structure that facilitates derivation of a set of "adopted" goals for a given task objective; goals are based on agent's knowledge of human-centric strategies for that task objective. Adopted goals are a consistent, feasible set of tasks. A schema of the entities which forms Strategic planner module is presented in Figure 3-2.

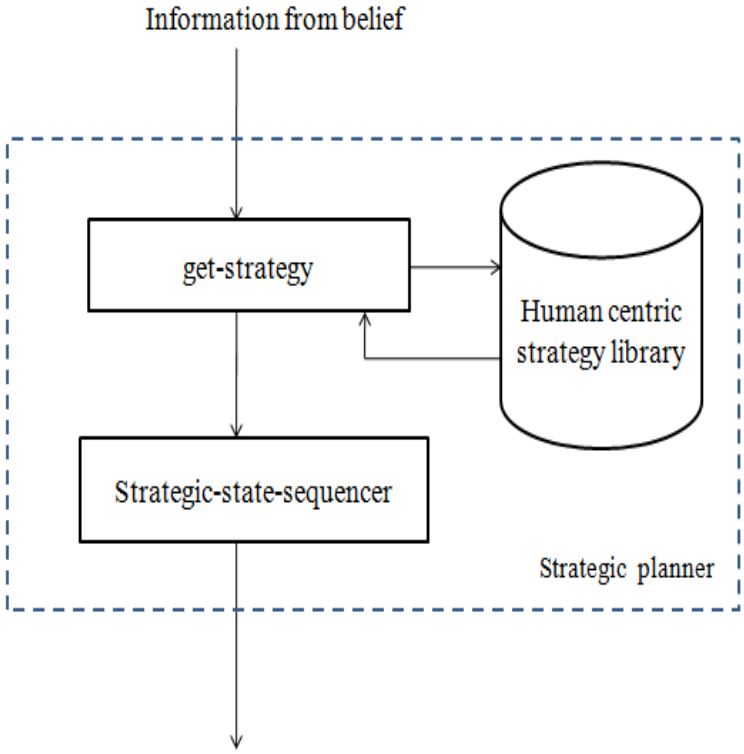

Figure 3-2: The Strategic planner module

- Human centric strategy library: As mentioned, knowledge of human strategies for task plays a central role in the cBDI agent architecture. It is accomplished by a component called human centric strategy library–makes the cBDI agent to know set of human strategies that a human can execute for a given task. Human centric strategy library is Domain Specific.
- $get - strategy$: This function is responsible for extracting adopted goal, which are derived based on agent's belief set, agent's knowledge of human centric strategy.

Definition 8. Strategic planner is defined through the 6–tuple:

$$\Pi =< \mathcal{B}, G, Hp, get - strategy, \psi, \gamma >$$

where

- $\mathcal{B}$ set of beliefs
- G is task objective
- Hp is agent strategy repository, a library of human centric strategies of task
- $get - strategy$ function returns the derived strategy Hp^i that has the maximum percentage of observed states (of belief $\mathcal{B}$) in the strategy from the repository (in Hp)
- ψ denotes adopted goal
- γ is a function that assess relevance of each adopted goal.

Human centric strategy library contains set of strategies that are expressed as graphs

$$Hp^i :< V, E >$$

where
V is set of nodes
E is a set of edges, $E \subseteq V \times V$

The Strategic planner controls two key processes: Derive a set of adopted goal through $get - strategy$, and check the relevance of each adopted goal; the $get - strategy$ create and sequences agent's set of adopted goals.

For the second process, we define the function $strategic - state - sequencer$ which create and sequences agent's set of adopted goals.

$$strategic - state - sequencer : \mathcal{B} \times Hp^i \rightarrow \psi$$

where
ψ represented as

$$\psi =< g, \prec >$$

where,

- $g =\{g_1, ...g_n\}$
- $\prec$ is a order constraint on g of the form $(g_i \prec g_j)$

Function γ assess relevance of each $g \in \psi$. γ filter returns a value that reflects the relevance of a adopted goal for a set of belief.

$$\gamma : g \times \mathcal{B} \rightarrow \{True, False\}$$

The following is the cycle of Strategic planner:

1. begin with a belief set $\mathcal{B}$
2. $Hp^i \leftarrow get - strategy(\text{Hp}, \mathcal{B})$
3. $\Psi \leftarrow strategic - state - sequencer(\mathcal{B}, Hp^i)$
4. for each $g \in \psi$
5. current belief set $\mathcal{B}$
6. decide relevance of g for current belief set
   $\gamma(\text{g}, \mathcal{B}) \leftarrow \{True, False\}$
7. **if**true **then** $execute(\pi)$ **else** update g

Structure of the strategy retrieval

To collaborate, cBDI agent is attempting to follow a strategy as the human in the system. The strategy retrieved from Human centric strategy library dictate what kind of composition the agent will produce for a specific domain. An overview of the strategy retrieval process is shown in Figure 3-3. An observation for cBDI agent is a communicative action of human; a set of actions corresponds to a strategy state. Structure of the strategy retrieval is inspired by the work of [192].

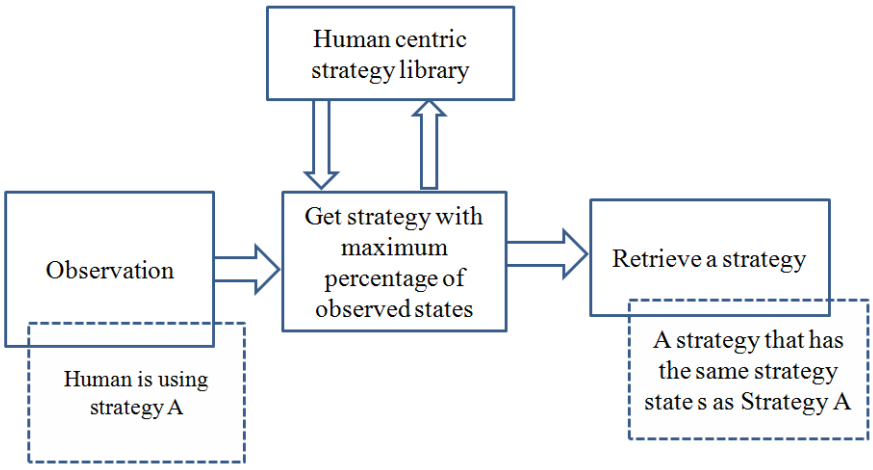

Figure 3-3: Structure of the strategy retrieval process

From the communicative action of human, agent makes observations about strategy state of human in the environment. Instead of trying to find an exact match between the observations about human intent and a strategy in the human

centric strategy library, the agent is attempting to find a strategy that is similar to the (strategy) states that are being observed.

A Strategy is considered similar to observations on two criterion:

- The number of observation that appears in a particular strategy. A strategy is called similar where 90% of the observations occurs. A similar strategy gets a higher score.
- The number of states in the strategy that has not yet been observed.

Consider the following example to bring home the above retrieval process.

Example:

There are 10 states in the system. They are denoted by the numbers 1 through 10. A strategy is represented as a list of these states, for example$\{1, 2, 3, 4, 7\}$. In this example, cBDI agent's strategy library is shown in Table 3.1. The observation

Table 3.1: Example of Strategy Library

Strategy Name	List of states
Strategy A	1, 2, 3
Strategy B	3, 4, 6, 7, 8
Strategy C	8, 9, 10

made by cBDI agent are found in Table 3.2

Table 3.2: Observation made by cBDI agent

Time instant	Observation
t_1	1
t_2	7
t_3	3

Here, strategy A and B both include two out of the three states. So strategy A and B are more similar to the observation. Strategy C has a less score compared to scores of strategy A and B. Strategy A is chosen as current strategy for cBDI as there is only one state in Strategy A that has not yet been observed, while there are three states in plan B that have not been observed. Strategy A is preferred when retrieving the strategy from the library.

3.4.3 Desires

Desires is the module responsible for the inducement of cBDI agent to reach certain states. Such a state, once it appears, can be described as a set of beliefs that hold true.

In the cBDI agent, selection of a desire occurs in response to a particular belief state of the agent and *current adopted goal*. *Desire generator* function is responsible for selecting one desire state from the set of possible desires. Desire generator function described here is similar with the one in classical BDI agent; however, we have added an *adopted goal*. Desires describes *goal states* which agent attempts to make true!

Definition 9. [*Desire generator*] *Desire generator* function ($\mathcal{D}g$) generates agent desire. $\mathcal{D}g$ is a mapping computing the relevance of a desire in the current situation.

$$\mathcal{D}g : \mathcal{B} \times \mathcal{I} \times g \rightarrow \mathcal{D}$$

where,

- $\mathcal{D}$ is set of desires.
- g is agent's current adopted goal, $g \in \psi$; where ψ is adopted goal.
- $\mathcal{I}$ is agent intention.

3.4.4 Intentions

Intention is the module that describes what has to be done and link the actions to the corresponding desires and belief. An intention is basically the commitment of cBDI agent to actively follow a desire. This also includes a specific action that was selected in order to fulfill the desire. An intention is created that uses one or more actions to achieve a desire. The agent selects one *desire* (which becomes an *intention*) via the *intent function*.

Definition 10. [*Intent*] *Intent* is agent intention generator function that determines the agent intention:

$$Intent : \mathcal{B} \times \mathcal{D} \times \mathcal{I} \times value \rightarrow \mathcal{I}$$

The *Intent* function described here is similar with the one in classical BDI agent; however, here we have added agent's behaviour state *value*. *value* reflects the agent's mode. The *value* influences agent's intention.
Mode function (*mode*) generates agent's behaviour states.

Definition 11. [*mode*]: It is a function that determines the agent's current mode based on the human capacity.

$$mode : C_h \rightarrow value$$

where
C_h denotes human capacity
$value$=$\{0, 1\}$ reflects the agent's mode during collaboration.
where,
0 signifies human is cut-off; cBDI is active. 1 signifies human is active; cBDI is inert.

The agent's *plan* generates plans to achieve its *intention* (the standard BDI interpretation).

Definition 12. The agent *plan* to generate a ordered sequence of actions to satisfy agent's intention

$$plan : \mathcal{I} \times A_c \rightarrow \pi$$

where, π denotes a Plan.

Definition 13. The *execute* function is to execute π, defined as

$$execute : \pi \rightarrow W$$

3.5 Formal model of human–cBDI agent collaboration

To collaborate, as in [193, 194], cBDI agent form team with human by adopting joint persistent goal. Here, the interpretation of joint persistent goal is the task objective. To collaborate with human, there must be some belief of cBDI agent that is mutual.

cBDI agent–human collaboration here is seen as cBDI agent–human collaborative planning. cBDI agent collaboration with human (HAC) can be described by cBDI agent's execution of set of plan with a human H in an environment having common task object G.

Definition 14. [Human–cBDI agent collaboration (HAC)]
Human–cBDI agent collaboration (HAC) is a tuple:

$$< \Delta, G, MB, Collb >$$

where:
Δ is a pair of $< H, \mathcal{C} >$ H is human and $\mathcal{C}$ is cBDI agent
G is a common task objectives

MB is set of mutual beliefs and $MB \subset \mathcal{B}$
$Collb$ is collaborative plan for G

Mutual Beliefs: $\mathcal{C}$ has beliefs about the current state of the world and human. This is referred as mutual beliefs.

Definition 15. [Collaborative plan]
Collaborative plan $Collb$ is a set of plans of cBDI agent to support the human in accomplishment of task object G. Collaborative plan for pair Δ is a 2–tuple

$$Collb =< W_\pi, \mathcal{A} >$$

- W_π=$\{\pi_1, \pi_2, ...\}$ is a set of sub-plans

- $\mathcal{A}$ is a set of of actions for each $\pi \in W_\pi$

$$\mathcal{A} : A_c \cup A^c$$

3.6 The control loop

The following control loop is for cBDI architecture. The main control loop starts with the process of observing the world. The control loop of cBDI architecture includes following abstract steps:

1. Observe the environment.
2. The agent generates belief about goal assigned by the human and human capacity
3. The agent generates belief about human action.
4. Agent current belief of human capability calculates a *value* using function *mode*.
5. Generate set of ordered adapted goals using belief candidates together with *human strategy library*.
6. Calculates relevance of each adapted goal for current belief.
7. If relevance is True, from the beliefs and intentions and adapted goal, the agent generates desires.

Algorithm 1: cBDI Architecture: Pseudo-code of cBDI agent's main control loop

```
1  B ← B_A;%comment: Initial belief%
2  D ← D_0;%comment: Initial desire%
3  I ← I_0;% comment: Initial intention%
4  while true do
5      B_S ← self_aware(W);
6      B_I ← interaction(G, C_h);
7      B_h ← human_intent(A^c);
8      B ← (B_A ∪ B_S ∪ B_I ∪ B_h);
9      while ¬G do
10         value ← mode(C_h);
11         Hp^i ← get − strategy(B, Hp);
12         Ψ ← strategic − state − sequencer(B, Hp^i);
13         while g ∈ ψ is not empty do
14             comment: γ(g, B) check relevance of current g in current B
15             if γ(g, B) ≡ True then
16                 D ← Dg(B, I, g);
17                 I ← Intent(B, D, I, value);
18                 π ← plan(I, B);
19                 execute(π);
20                 update B;
21                 new_C_h ← check(B, C_h);
22                 if new_C_h ≠ C_h then
23                     value ← mode(C_h);
24                 update g
25             else
26                 update g;
27                 update B;
28                 new_C_h ← check(B, C_h);
29                 if new_C_h ≠ C_h then
30                     value ← mode(C_h)
31             if no g ∈ ψ with untried action then
32                 return failure
```

8. Under the influence of the value, the agent chooses intentions based on current beliefs, desires and intentions.

9. For current intention, and set of actions, the agent selects sets of action using the functional plan.

10. Executes each of the action.

11. If relevance is False, agent updates its adopted goal and then updates its belief. From the beliefs and value of C_h, agent then generates new_C_h.

12. Select a new adopted goal until it's not empty.

13. If relevance is False, agent update its adopted goal and then updates its belief. From the beliefs and value of C_h, agent then generates new_C_h. If new_C_h is not equal to previous C_h, agent calculate *value* using function *mode*. Otherwise, if new_C_h is equal to previous C_h, agent continually updates its belief. this Continues until the goal is not reached.

14. Goto to step1

3.7 An illustrative example

This section presents an illustrative example, describing how the cBDI agent collaborates its action with human. The example is taken from the HRI 2015 Workshop "Towards a Framework for Joint Action". The aim of this example is to show human-cBDI agent collaboration in achieving block stacking events.

3.7.1 Scenario: Block stacking problem

The scenario described here is proposed by the workshop organizers at http://fja.sciencesconf.org/resource/page/id/1. *In the scenario, a human (H) and a robot (A) are asked to build a pile with four cubes in a specified order with a triangle (cone) on top. Each agent has a number of cubes accessible in front of him and would participate in the task by placing its cubes on the pile. One of the agents should place a cone at the top of the pile at the end. Figure 3-4 shows initial state of the world. There are cube1, cube3 and a cone in front of H. Those are accessible only to H. There are two cubes, cube 2, cube 4 and a cone in front of A. These are accessible only to A. The pile P is a position on the table, which is accessible for both A and H.*

Actions available for each agent are the following (with object = cube or triangle):

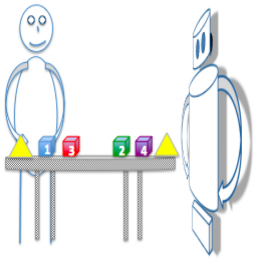

Figure 3-4: The initial scenario

- take an object on the table
- take an object from the pile
- put an object on the pile
- give an object to the other agent
- support the pile

Figure 3-5 shows two possible goal states where either Human can take his side of the cone and put it on the pile or agent *A* can take his side of the cone from the table and put it on the pile.

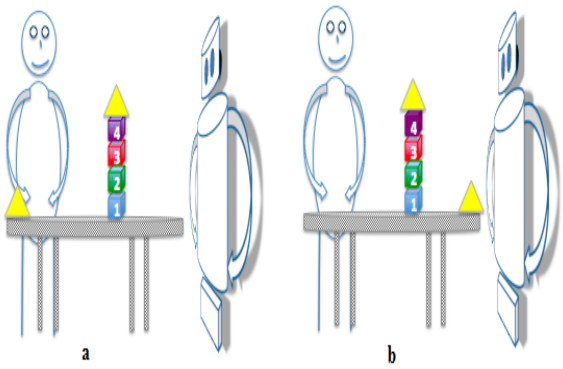

Figure 3-5: The expected final state.

Each agent is able to infer the state of the world so it knows:

- where each object is
- if an object is reachable for itself
- if an object is reachable for the other one

Both agents cannot perform the task alone. Both the agents have an incentive to collaborate in this task; as neither H nor A can reach all the objects required for the pile.

- The agent begins its activity by gathering information from the world.
- Agent A has full visibility of the environment.
- Agent A perceive the action of H as communicative action.
- The pile is common location and reachable for both.
- Agent A updates its belief for every action.
- Agent A recognizes following pre–defined human strategies to stack blocks and cone.
  1. Strictly Ascending order,
  2. Ascending order
  3. Even numbers
  4. Odd numbers
  5. Strictly descending order

3.7.2 cBDI agent in action

Let's focus on the agent A, and see how to apply the cBDI architecture to this agent in detail. Considering first the main components:

- Belief set of agent stores information about the Block world.

$$\mathcal{B} = \{R(obj, loc), A_c \cup A^c, G\}$$

where

$$obj \in \{cone, cube_i\}$$

$$i \in \{1, 2, 3, 4\}$$

R is a relation

$$R \in \{on, near\}$$

$$loc \in \{hand_H, hand_A, T, P, L_H, L_A\}$$

Here,
$hand_H$ belongs to H, $hand_A$ belongs to A.
T=Table
Pile, P; the location that is reachable for both human and agent
L_H location accessible only to H.

L_A location accessible only to A.
G is *task objective*. Agent A received G from human H.
Other obvious predicates are assumed / used as required. In this example G is of the form

$$G = ((stack - ascending - order - cone - on - top) \equiv true)$$

- A_c is set of actions of A; A^c is set of actions of H.

  $Action : Take_k(obj, T)$; $k \in \{H, A\}$

  $precond : on(obj, T) \wedge hand_k(empty)$
  $postcond : \neg on(obj, T) \wedge\neg\ hand_k(empty)$

  $Action : Take_k(obj, P)$

  $precond : on(obj, P) \wedge\ hand_k(empty)$
  $postcond : \neg on(obj, P) \wedge \neg\ hand_k(empty)$

  $Action : put_k(obj, P)$

  $precond : \neg on(obj, P) \wedge \neg hand_k(empty)$
  $postcond : on(obj, P) \wedge hand_k(empty)$

  $Action : give(hand_x, obj, hand_y)$; $x \neq y$

  $precond : hold(object, hand_x) \wedge hand_y(empty)$
  $postcond : \neg hold(object, hand_x) \wedge \neg\ hand_y(empty)$

  $Action : support_k(P, hand_k)$

  $precond : hand_k(empty)$
  $postcond : hold(P, hand_k)$

- Hp is human strategy library
  A human strategy in $Hp^i \in Hp$ represented as

  $$\{name, body\}$$

  where
  name denotes name of the strategy

$$\text{name}\in \{strictly-ascending-order, ascending-order, even-order, odd-order, strictly-descending-order, descending-order, random-order\}$$

body denotes description of the strategy.

- The Strategic planner derives adopted goal ψ for a given task objective G in Block world.

  Adopted goal ψ is represented as

  $$\psi =< g, \prec >$$

  where,

  - g=$\{state-1, state-2, state-3, state-4, state-cone\}$
  - $\prec$ is a order constraint on g of the form $(state-i \prec state-j)$

- Agent's current desires are stored in desire set. Assume six desires are used for this agent A, and these desires are given by the set

  $$\mathcal{D}_m = \{observe, maintain-state-1, maintain-state-2, maintain-state-3, maintain-state-4, maintain-state-cone\}$$

  The Desire for agent A can be represented as :

  $$\mathcal{D} = \{\mathcal{D}_m\}$$

  Example, *maintain-state-1* represents the desire to stack cube(1). Initially, $\mathcal{D}_0$= *observe*.

- Intention is the agent's current plan. For intention we are going to use simple set

  $$\mathcal{I}_m = \{observe_a, reachstate\}$$

The Intention set for agent A can be represented as :

$$\mathcal{I} = \{\mathcal{I}_k | \mathcal{I} \in \mathcal{I}_m, k \in \{-, +\}\}$$

Initially, $\mathcal{I}_0 = \{observe_a\}$.

The main function for agent

We now consider the main function for agent A. We consider the belief functions:

- $self_{aware}$ gets the belief about location and surrounding environment.

  Example 1. Let us consider world (W) like the one shown in Figure 3-4, Agent obtain a belief candidate as

$$\begin{aligned}\mathcal{B}_S = \{(&on(cube_1, T), near(cube_1, L_H), on(cube_3, T),\\ &near(cube_3, L_H), on(cube_2, T), near(cube_2, L_A),\\ &on(cube_4, T), near(cube_4, L_A), near(cone, L_H), near(cone, L_A)), -, -\}\end{aligned}$$

- *Interaction* obtains the belief through interaction with agent A. The human interaction M_{int} is in the form of

$$\{G, C_h\}$$

  G denotes *task objective*
  C_h denotes agent H (human) capacity

  Example 2. Suppose

$$C_h = \{proactive, incorrect\}$$

$$M_{int} = \{stack - ascending - order - cone - on - top, proactive\}$$

  then *interaction* obtains the belief candidate as

$$\mathcal{B}_I = \{-, -, (((ascending - order - cone - on - top) \equiv true), proactive)\}$$

which means that agent A obtained a belief that H requested to create stack in *ascending order with cone on top* and H has *proactive* abilities for actions.

- $human_{intent}$ is agent function that generates $\mathcal{B}_h$ through communicative action.

  Example 3. If agent H executed an action $put_H(cube_3, P)$
  $human_{intent}$ obtains the belief candidate as

  $$\mathcal{B}_h = \{-, put_H(cube_3, P), -\}$$

  which means that A obtained a belief that agent H executed action $put_H(cube_3, P)$

- Updating of belief $U_{\mathcal{B}}$ function considers current action status and revises the current beliefs based on previous beliefs and the set of belief candidates from $\mathcal{B}_S$, $\mathcal{B}_I$ and $\mathcal{B}_h$:

  Example 4. Suppose, for an instance current $\mathcal{B}$ is

  $$\begin{aligned}\mathcal{B} = \{&(on(cube_1, T), near(cube_1, L_H), on(cube_3, T),\\ &near(cube_3, L_H), on(cube_2, T), near(cube_2, L_A),\\ &on(cube_4, T), near(cube_4, L_A), near(cone, L_H), near(cone, L_A)), -,\\ &(((ascending - order - cone - on - top) \equiv true), proactive)\}\end{aligned}$$

  H executed an action

  $$Take_H(cube_3, T)$$

  $human_{intent}$ obtains the belief candidate as

  $$\mathcal{B}_h = \{-, Take_H(cube_3, T), -\}$$

  updated belief $U_{\mathcal{B}}$ obtains the belief as

  $$\begin{aligned}U_{\mathcal{B}} = \{&(on(cube_1, hand_H), on(cube_3, T),\\ &near(cube_3, L_H), on(cube_2, T), near(cube_2, L_A),\\ &on(cube_4, T), near(cube_4, L_A), near(cone, L_H), near(cone, L_A)), Take_H(cube_3, T),\\ &(((ascending - order - cone - on - top) \equiv true), proactive)\}\end{aligned}$$

Example 5. Suppose, for an instance current $\mathcal{B}$ is

$$\begin{aligned}\mathcal{B} = \{&(on(cube_3, cube_1), on(cube_1, P), on(cube_2, T),\\ &near(cube_2, L_A), on(cube_4, T), near(cube_4, L_A),\\ &near(cone, L_H), near(cone, L_A)),\\ &put_H(cube_3, P),\\ &(((ascending - order - cone - on - top) \equiv true), proactive)\}\end{aligned}$$

then *interaction* obtains the belief candidate as

$$\mathcal{B}_I = \{-, -, (((ascending - order - cone - on - top) \equiv true), incorrect)\}$$

updated belief $U_{\mathcal{B}}$ obtains the belief as

$$\begin{aligned}U_{\mathcal{B}} = \{&(on(cube_3, cube_1), on(cube_1, P), on(cube_2, T),\\ &near(cube_2, L_A), on(cube_4, T), near(cube_4, L_A),\\ &near(cone, L_H), near(cone, L_A)),\\ &put_H(cube_3, P),\\ &(((ascending - order - cone - on - top) \equiv true), incorrect)\}\end{aligned}$$

- *mode* function returns a *value*.

  Example, Suppose for a situation

  $$C_h = (incorrect)$$

  then *mode* function returns *value*=0.

- $get - strategy$ function returns the derived strategy Hp^i that has the maximum percentage of observed states (of belief $\mathcal{B}$) in the strategy from the repository (in Hp).
  Suppose, in an initial instance $\mathcal{B}$ is

$$\mathcal{B} = \{(on(cube_1, T), near(cube_1, L_H), on(cube_3, T), near(cube_3, L_H), on(cube_2, T), near(cube_2, L_A), on(cube_4, T), near(cube_4, L_A), near(cone, L_H), near(cone, L_A)), -, (((ascending - order - cone - on - top) \equiv true), proactive)\}$$

$get - strategy$ which is defined as:

$$get - strategy(\mathcal{B}, G, Hp) = Hp^i$$

derive Hp^i as

$$\{strictly - ascending - order, body\}$$

body denotes description of "strictly-ascending-order"

- $strategic - state - sequencer$ function sequences states of the strategy as agent's set of adopted goals. $strategic - state - sequencer$ which is defined as:

$$strategic - state - sequencer(\mathcal{B}, Hp^i) = \psi$$

where

$$\psi =< state - 1 \prec state - 2 \prec state - 3 \prec state - 4 \prec state - cone >$$

- γ assess relevance of each $g \in \psi$. γ filter returns a value that reflects the relevance of a adopted goal for $\mathcal{B}$.

$$\gamma : g \times \mathcal{B} \rightarrow \{True, False\}$$

Suppose, $\mathcal{B}$ is

$$\mathcal{B} = \{(on(cube_3, hand_H), on(cube_2, cube_1), near(cube_1, P), on(cube_4, T), near(cube_4, L_A), near(cone, L_H), near(cone, L_A)), Take_H(cube_3, T), (((ascending - order - cone - on - top) \equiv true), proactive)\}$$

If agent's adopted goal is

$$g = \{state - 3\}$$

as g is not relevant for current $\mathcal{B}$ then

$$\gamma(state-3,\mathcal{B}) = False$$

- The desire generator function generates new desires based on the agent's current beliefs and intentions, current adopted goal.

  Example 6.

$$\begin{aligned}\mathcal{B} = \{(&on(cube_2, cube_1), on(cube_1, P), on(cube_3, cube_2),\\ &on(cube_4, T), near(cube_4, L_A),\\ &near(cone, L_H), near(cone, L_A)),\\ &put_H(cube_3, P),\\ &(((ascending-order-cone-on-top) \equiv true), proactive)\}\end{aligned}$$

  If agent's adopted goal is

$$g = \{state-4\}$$

  and there is intention

$$\mathcal{I} = \{reachstate_+\}$$

  The *desire generator* function $\mathcal{D}g$ which is defined as

$$\mathcal{D}g(\mathcal{B},\mathcal{I},g) = \mathcal{D}$$

  then generate the desire as

$$\mathcal{D} = \{maintain-state-4\}$$

  If agent's adopted goal is

$$g = \{state-cone\}$$

  and there is intention

$$\mathcal{I} = \{reachstate_+\}$$

  The *desire generator* function then generate the desire as

$$\mathcal{D} = \{maintain-state-cone\}$$

- The intent function makes a decision on the intention.

Example 7. Suppose,

$$\begin{aligned}\mathcal{B} = \{&(on(cube_2, cube_1), on(cube_1, P), on(cube_3, cube_2),\\ &on(cube_4, T), near(cube_4, L_A),\\ &near(cone, L_H), near(cone, L_A)),\\ &put_H(cube_3, P),\\ &(((ascending - order - cone - on - top) \equiv true), proactive)\}\end{aligned}$$

If agent's current desire state is

$$\mathcal{D} = \{maintain - state - 4\}$$

and there is intention

$$\mathcal{I} = \{reachstate\_\}$$

and there is value

$$value = \{1\}$$

then the agent generate the Intention as

$$\mathcal{I} = \{reachstate_+\}$$

Again if agent's current desire state is

$$\mathcal{D} = \{maintain - state - 4\}$$

and there is intention

$$\mathcal{I} = \{reachstate\_\}$$

and there is value

$$value = \{0\}$$

then the agent generate the Intention as

$$\mathcal{I} = \{reachstate\_\}$$

- The plan generates a sequence of actions based on the intentions and current belief.

Suppose current intention

$$\mathcal{I} = \{reachstate_{+}\}$$

then it generate a sequence of actions:
π=$\{(Take_A(cone, T), give(hand_A, cone, hand_H)\}$
Suppose, current intention

$$\mathcal{I} = \{reachstate_{-}\}$$

then it generate a sequence of actions as:
π=$\{(Take_A(cone, T), put_A(cone, P)\}$

- *execute* function executes current π.

The above example shows an cBDI agent for the Block stacking world. The next example is to show how the cBDI architecture can solve the Block stacking problem

Example 8. Suppose, the human, in a situation, take $cube_1$ and put on the pile P. After that, he took $cube_3$ and put on the pile. The following trace of the loop gives behavior of the cBDI agent under such situation.

26. $g = \{state - 2\}$

27. update $\mathcal{B}$

$$\begin{aligned}\mathcal{B} = \{&(on(cube_1, hand_H), on(cube_3, T), near(cube_3, L_H),\\ &on(cube_2, T), near(cube_2, L_A), on(cube_4, T), near(cube_4, L_A),\\ &near(cone, L_H), near(cone, L_A)), Take_H(cube_1, T),\\ &(((ascending - order - cone - on - top) \equiv true), proactive)\}\end{aligned}$$

27. update $\mathcal{B}$

$$\begin{aligned}\mathcal{B} = \{&(on(cube_1, P), on(cube_3, T), near(cube_3, L_H),\\ &on(cube_2, T), near(cube_2, L_A), on(cube_4, T), near(cube_4, L_A),\\ &near(cone, L_H), near(cone, L_A)),\\ &put_H(cube_1, P), (((ascending - order - cone - on - top) \equiv true), proactive)\}\end{aligned}$$

27. update $\mathcal{B}$

$$\mathcal{B} = \{(on(cube_1, P), on(cube_3, hand_H) \\ on(cube_2, T), near(cube_2, L_A), on(cube_4, T), near(cube_4, L_A), \\ near(cone, L_H), near(cone, L_A)), \\ Take_H(cube_3, T), (((ascending{-}order{-}cone{-}on{-}top) \equiv true), proactive)\}$$

27. update $\mathcal{B}$

$$\mathcal{B} = \{(on(cube_1, P), on(cube_3, cube_1) \\ on(cube_2, T), near(cube_2, L_A), on(cube_4, T), near(cube_4, L_A), \\ near(cone, L_H), near(cone, L_A)), put_H(cube_3, P), \\ (((ascending - order - cone - on - top) \equiv true), proactive)\}$$

28. %check for C_h %
check($\mathcal{B}$, proactive)=new_C_h
new_C_h= $incorrect$;

29. $new\_C_h \neq C_h$ is true;

30. $value$ =0;

13. $g \in \Psi$ *is not empty* =True

14. % check relevance of current g in current $\mathcal{B}$%

15. $$\gamma(state - 2, \mathcal{B}) = True;$$

16. $$\mathcal{D}g(\mathcal{B}, \{observe_a\}, state - 2) = \{maintain - state - 2\}$$

17. $$intent(\mathcal{B}, \{maintain - state - 2\}, \{observe_a\}, 0) = \{reachstate\_\}$$

18. $\pi =< Take_A(cube_3, P), give(hand_A, cube_3, hand_H), Take_A(cube_2, T), put_A(cube_2, P) >$

19. $execute(Take_A(cube_3, P))$

20.

$$\mathcal{B} = \{(on(cube_3, hand_A), on(cube_1, P)$$
$$on(cube_2, T), near(cube_2, L_A), on(cube_4, T), near(cube_4, L_A),$$
$$near(cone, L_H), near(cone, L_A)), Take_A(cube_3, P),$$
$$(((ascending - order - cone - on - top) \equiv true), incorrect)\}$$

19. $execute(give(hand_A, cube_3, hand_H))$

20.

$$\mathcal{B} = \{(on(cube_3, hand_H), on(cube_1, P)$$
$$on(cube_2, T), near(cube_2, L_A), on(cube_4, T), near(cube_4, L_A),$$
$$near(cone, L_H), near(cone, L_A)), give(hand_A, cube_3, hand_H),$$
$$(((ascending - order - cone - on - top) \equiv true), incorrect)\}$$

19. $execute(Take_A(cube_2, T))$

20.

$$\mathcal{B} = \{(on(cube_3, hand_H), on(cube_1, P)$$
$$on(cube_2, hand_A), on(cube_4, T), near(cube_4, L_A),$$
$$near(cone, L_H), near(cone, L_A)), Take_A(cube_2, T),$$
$$(((ascending - order - cone - on - top) \equiv true), incorrect)\}$$

19. $execute(put_A(cube_2, P))$

20.

$$\mathcal{B} = \{(on(cube_3, hand_H), on(cube_1, P)$$
$$on(cube_2, cube_1), on(cube_4, T), near(cube_4, L_A),$$
$$near(cone, L_H), near(cone, L_A)), put_A(cube_2, P),$$
$$(((ascending - order - cone - on - top) \equiv true), incorrect)\}$$

The number here represent the line number of the main control loop.

3.8 Summary

In this chapter, we have presented extended BDI based agent architecture; to manage collaboration. We have identified why BDI based architecture need extension for human agent collaboration and described the relevant features. Described the importance of strategic planner module for providing support for building decisional agent. We have presented the formal structure of extended BDI based agent architecture–cBDI agent. We have also formalized human and cBDI agent collaboration to achieve joint activity task.