Chapter 5

When to take decisions during Shared Plans elaboration and execution

Contents

5.1 Motivation

When performing a Joint Action and more particularly when executing Shared Plans, several choices have to be made. Some of them are implicit, while others require a negotiation or an adaptation between the Joint Action participants. To be a good partner when performing Joint Action with humans, the robot should be able to identify which decisions are implicit and correctly communicate about the other ones. Indeed, a robot which communicates about each detail of a Shared Plan would quickly become too "chatty" while a robot which does not communicate can be confusing.

Let's take for example a robot helping a human to build a flat-pack table: the legs of the table need to be assembled with a hammer, the tray with a screwdriver and finally someone needs to put the tray on the legs. The robot is equipped with several tools including a screwdriver but no hammer and the human has only one hammer. The human and the robot are both able to put the tray on the legs. It is common sense that the robot should assemble the tray while the human assemble the legs. However, a decision needs to be taken concerning who will put the tray on the legs. In this scenario, we would like the robot to assemble the tray without asking the human and negotiate or adapt its behavior to put the tray at the last moment.

The work presented in this chapter consists in finding which decisions are implicit or not, when the decisions should be taken and how to take them. We identify three types of decisions to be taken during Shared Plans elaboration and execution:

- **Which action to perform in which order:** this is one of the biggest concern during Shared Plan elaboration. We do not focus our work in this challenge. Indeed, we use HATP, a human-aware HTN planner which has been demonstrated to be well suited to human-robot joint action [Lallement 2014], to deal with this issue.

- **Who will perform which action:** sometimes this decision can be trivial when only one agent is able to perform an action. However, in other situations, the robot should be able to decide who will perform an action by negotiating or adapting its behavior to the human one.

- **With which object:** for practical reason, the robot reasons on objects by attributing them a unique id. However, for the purpose of an action, two objects can be semantically identical. When there is a choice in which object to use for an action, the robot should be able to adapt to the human behavior in order to avoid potential conflicts.

This work has been the subject of a publication at the ICSR 2017 conference [Devin 2017].

5.2 Background

When the robot needs to achieve a joint goal, several works allow it to compute plans which take into account the human ([Cirillo 2010, Lallement 2014]). They allow the robot to reduce resource conflicts [Chakraborti 2016], take divergent beliefs into account ([Warnier 2012, Talamadupula 2014]) or promote stigmergic collaboration for agents in co-habitation [Chakraborti 2015].

The relevance of using a Shared Plan in human-robot interaction has been studied by [Lallée 2013]. They suggest that the joint plan should be fully communicated in order to sustain effective collaboration. Moreover, in [Gombolay 2015], it is shown that subjects prefer letting the robot plan when the task is too complex, prioritizing

efficiency. In more simple tasks, a robot proactively helping the human is preferred to one waiting before proposing help [Baraglia 2016].

If the robot decides to share the plan, several studies have been reported on how to communicate about the plan. Some researchers studied how a system could acquire knowledge on plan decomposition from a user [Mohseni-Kabir 2015] and how dialog can be used to teach new collaborative plans to the robot and to modify these plans [Petit 2013]. In [Sorce 2015], the system is able to learn a plan from a user and transmit it to another user. [Allen 2002] presents a computer agent able to construct a plan in collaboration with a user. Finally, in [Milliez 2016], Milliez et al. present a system where the robot shares the plan with a level of details which depends on the expertise of the user. In our work, we try to get rid of the entire shared plan verbalization by taking the right decision at the right time in order to come up with a robot which communicates at the right time.

Several contributions have been done to allow more adaptability during human-robot Shared Plan execution. [Chien 2000] proposes a method to plan only a few steps in advance and then plan the actions further in an iterative way. This allows the plan to incorporate execution feedback such as early or late execution of actions and over-use or under-use of resources. Chaski, a task-level executive that uses insights from human-human teaming to make human-robot teaming more natural and fluid, is presented in [Shah 2011]. The system chooses and schedules the robot's actions by taking into account the human partner and acts to minimize the human's idle time. A system which mixes plan recognition and adaptation is described in [Levine 2014]. It computes all possibilities for the plan and chooses an action based on the choice of the human and causal links. [Hoffman 2007] proposes an adaptive action selection mechanism for a robotic teammate, making anticipatory decisions based on the confidence of their validity and their relative risk. [Karpas 2015] presents Pike, an online executive that unifies intent recognition and plan adaptation for temporally flexible plans with choice. Finally, the work presented here is based on SHARY [Clodic 2009] which was extended in [Fiore 2014], a supervisor allowing to execute human-aware shared plans taking into account joint actions aspects like reactive action execution.

In the cooperative multi-robot literature, task allocation and cooperative activity achievement has been thoroughly investigated [Gerkey 2004]. Auction has been used very successfully for distributed multi-robot in various contexts ([Gerkey 2002, Botelho 1999]). Several works on teamwork and cooperative task achievement take into account explicit constraints to facilitate the activity of the other robots or agents activity. [Tambe 1997] presents STEAM, an architecture based on the *joint intention* theory. It integrates concepts such as team synchronisation and monitoring of joint intention in order to improve flexibility and reusability. A plan manager which provides the services needed to build and execute plans in a multirobot context is presented in [Joyeux 2009]. It provides tools for safe concurrent execution and modification of plans, and handles distributed plan supervision without permanent robot-to-robot communication. While these contributions have inspired work on human-robot collaboration, it is however important to exhibit

some differences between the two fields. Indeed, the human and the robot are not equal in any aspect. The robot is here to help the human and facilitate his activity.

In AI, the goal reasoning domains deals with some problems similar to Shared Plan management [Molineaux 2010, Roberts 2016]. The role of goal reasoning is to survey the current goals of a robot, check that they remain feasible and relevant and establish new goals if needed. Moreover, part of the goal reasoning function is sometimes linked to the plan management as it is in charge of deciding when and how to generate plans (but it is not producing the plan) and checking for unexpected events.

5.3 Assumptions

The work presented in this chapter deals with the needed decision during Shared Plan elaboration and execution. The idea here is to focus on decisions concerning the action allocation and instantiation. To do so, we make several assumptions:

Single human: the work presented in this chapter has been designed for a robot interacting with a single human. However, all the data structures and main principles are compatible with multi-humans set-up.

Commitment: we do not focus in this work on issues related to commitment. Consequently, we consider here that the joint goal has already been established. We also consider that the human will not abort the goal unless he knows or infers that the goal is not achievable any more.

Shared Plan: we put the focus here on the issues related to action allocation and instantiation. In order to decide which action to execute in which order, we use HATP, a human-aware HTN planner which has been demonstrated to be well suited to human-robot Joint Action [Lallement 2014]. We are focusing in this work about medium complexity Shared Plans where the human might want to decide for his own actions.

Humans perception: we make the assumption here that a human will see and understand an action of the robot when he is present and looking at it. We also assume that when he is present, the human is able to hear and understand the information verbalized by the robot.

Robot capacities: we consider that the robot is able to perform simple high level actions like Pick, Place or Drop. We also assume that the robot is able to ask to the human if he wants to perform an action and to understand a basic answer (yes/no type). The robot is able to detect and localize objects and agents and to recognize simple high level actions performed by the human like Pick, Place or Drop. Let us also note that the ways the robot achieves actions (e.g. human-aware motion planning and execution) and recognizes human's actions are outside of the scope of this chapter.

Communication: the focus of this work is more on "what to communicate" rather than on "how to communicate". Here we use the basic dialogue module described in Chapter **??** to communicate with the human but more complex communication mechanisms can be envisioned.

5.4 Main principles

We will present in this section the main principles we use for Shared Plans management. Three algorithms are used to allow the robot to elaborate and execute Shared Plans. They interact through the Shared Plan data SP and several signals (noted S_X where X is the name of the signal). These three algorithm run constantly and in parallel.They allow respectively to maintain the state of the shared plan, to choose actions for the robot, and to monitor the human.

Alg. 1 allows the robot to elaborate a Shared Plan when needed, to maintain the current Shared Plan and to manage the human mental states.[1]

When the robot has a goal g_R to achieve and no current plan or when a signal is received to compute a new plan ($S\_needReplan$), the robot computes a Shared Plan to perform the goal based on the current world state WS (see Sec. 5.5):

$$SP \leftarrow PLAN(g_R, WS)$$

When an action a from the plan is performed by an agent a signal is received ($S\_needUpdate$) and the supervisor updates the plan (see Sec. 5.6.1):

$$SP \leftarrow UPDATE\_PLAN(SP, a)$$

When an action from A^X has been allocated ($S\_actionAllocated$), the robot looks for the consequences of this allocation in the plan (see Sec. 5.6.3):

$$SP \leftarrow EVALUATE\_PLAN(SP, WS)$$

The robot also constantly checks if the goal is reached (the objectives of the goal are in the current World State). Finally, each time a change occurs in $WS(H)$ or TS, the robot estimates the human mental states as described in the previous chapter (see Chapter **??**):

$$MS(H) \leftarrow ESTIMATE\_MS(MS(H), TS)$$

If there is a conflict between the knowledge of the robot and the human mental state, the robot tries to solve it (see Chapter **??**):

$$SOLVE\_DB(MS(H), TS)$$

[1]The terms used in the algorithms are reminded in Appendix **??**.

Algorithm 1 Shared Plan management

while g_R **do**
  if $!SP \parallel S\_needReplan$ **then**
    $SP \leftarrow PLAN(g_R, WS)$
  end if
  if S_needUpdate **then**
    $SP \leftarrow UPDATE\_PLAN(SP, WS)$
  end if
  if S_actionAllocated **then**
    $SP \leftarrow EVALUATE\_PLAN(SP, WS)$
  end if
  if $Obj_g \in WS$ ▷ *The goal is achieved*
  then
    $g_R \leftarrow \emptyset$
    $SP \leftarrow \emptyset$
  end if
  if $< Human, isPresent, true > \in WS$ & $(WS(H) \neq WS(H)_{t-1} \parallel TS \neq TS_{t-1})$ **then**
    $MS(H) \leftarrow ESTIMATE\_MS(MS(H), TS)$
    if $MS(H) \neq TS$ ▷ *Divergent belief*
    then
      $SOLVE\_DB(MS(H), TS)$
    end if
  end if
end while

Running in parallel with the first algorithm, Alg. 2 allows the robot to decide when to act and which action to perform.

When the robot has a Shared Plan SP, it looks for the actions of this plan which need to and can be executed (see Sec. 5.6.1):

$$A_{next} \leftarrow GET\_NEXT\_ACTIONS(SP, WS)$$

If there is no action in A_{next} nor in progress, it means that the plan is blocked, so the robot looks for another Shared Plan. If there are actions to do, the robot looks if there is an action it can execute (actions allocated to it or not allocated yet). If there is none, the robot waits for the human to perform an action (A_{next} contains only actions from A^H_{next}):

$$actionExecuted \leftarrow WAIT\_ACTION(A^H_{next}, t)$$

If after a time t, the human did not execute any action, the robot looks for another plan. If there are actions the robot can execute, the robot selects an action a (see Sec. 5.6.2):

$$a \leftarrow SELECT\_ACTION\_TODO(A_{next})$$

If the selected action is not yet allocated, the robot first tries to allocate it (see Sec. 5.6.3):

$$actor \leftarrow ALLOCATE\_ACTION(SP, a, WS, Prefs)$$

If the action is allocated to the robot (after selection or allocation), the robot executes it:

$$success \leftarrow EXECUTE(a)$$

It will first instantiate the action if needed and then launch its execution (see Sec. 5.6.4). If the action succeeds, the robot updates the plan, else it looks for another plan.

Algorithm 2 Robot action decision

```
while SP do
  A_next ← GET_NEXT_ACTIONS(SP, WS)
  if A_next = ∅                                   ▷ No more feasible actions
   then
    S_needReplan
  else if {A^R_next ∪ A^X_next} = ∅               ▷ No actions for the robot
   then
    actionExecuted ← WAIT_ACTION(A^H_next, t)
    if !actionExecuted then
      S_needReplan
    end if
  else
    a ← SELECT_ACTION_TODO(A_next)
    if a ∈ A^X_next then
      actor ← ALLOCATE_ACTION(SP, a, WS, Prefs)
      S_actionAllocated
    end if
    if a ∈ A^R_next || (a ∈ A^X_next & actor = robot) then
      success ← EXECUTE(a)
      if success then
        S_needUpdate
      else
        S_needReplan
      end if
    else if (a ∈ A^X_next & actor = human) then
      a → A^H_next
    end if
  end if
end while
```

In parallel to the other execution loops, the robot is constantly monitoring human activities (Alg. 3).

When the human performs an action, the robot first looks if the action is conflicting to the one it is performing (for example, if the human picks an object the robot was going to pick), and if it is the case, the robot stops its actions. Then, if the human performs an expected action with respect to the plan and already allocated to him, the robot updates the plan accordingly. If the action executed by the human is expected with respect to the plan but was not yet allocated, the robot looks for the consequences of this actions in the plan (see Sec. 5.6.3). If the human performs an unexpected action with respect to the plan (action not in A_{next} or in A^R_{next}), the robot waits the end of the human action and then looks for a new plan from the new situation induced by the human action.

Algorithm 3 Human monitoring

```
while < Human, isPresent, true > ∈ WS do
  if ∃ a ∈ A^H_cur then
    if a ∈ A^R_cur then
      S_stop
    end if
    WAIT_END_ACTION(a)
    if a ∈ A^H_next then
      S_needUpdate
    else if a ∈ A^X_now then
      S_actionAllocated
    else
      S_needReplan                          ▷ Unexpected action
    end if
  end if
end while
```

The next sections will define more precisely the operators we just defined.

5.5 Shared Plans elaboration

The first step of this work is to be able to compute Shared Plans that are flexible enough to postpone part of the decisions to execution time. This step correspond to the operator:

$$PLAN(g_R, WS)$$

As stated before, the human-aware HTN task planner HATP is used in this work to compute Shared Plans taking into account a number of social rules for both the robot and its human partner [Lallement 2014]. However, in order to obtain flexible plans with HATP, a number of issues have to be considered.

First, when HATP returns a plan, it returns only one, which is assumed to be the best plan it has found given the situation and the associated costs. However,

this plan is not always the only one possible (even at constant cost or computing time). Indeed, in such case, HATP makes some choices that could be preferably done on-line. For example, it can happen that one action can be done by several agents at the same cost. In a collaborative setting and more particularly when the human is concerned, it could be interesting to let the agents decide at execution time (or whenever it is interesting) who will do what. To handle this, we have adapted HATP by inserting what we call the *X agent*. The capabilities of the *X agent* correspond to the intersection of the capabilities of the human and the robot with a lower cost. Consequently, it will be chosen by the planner instead of the human or the robot whenever it is possible. If HATP returns a plan containing an action to be done by the *X agent*, it means that this action could either be performed by the human or the robot. The decision concerning who will finally do this action is postponed. We will see in Sec. 5.6.3 how *X agent* actions will be finally allocated to the human or the robot.

As inputs to a planner such as HATP, we give a set of objects that are present in the environment and on which it will be able to apply its operators. Basically, each object is tagged and is unique. That means that if we have the same object twice, they will be uniquely tagged (e.g. two identical red cubes will be tagged as RED_CUBE_1 and RED_CUBE_2). When two similar objects can be used in a same way during a task, the planner will choose either one or the other. In a collaborative setting, it could be counter intuitive since even if there is no distinction between the two objects at planning time, there can be one during execution. To handle this, we have adapted HATP by inserting the notion of *similar* objects which aims to group interchangeable objects under a common name: two *similar* objects will have the same role in the task.

Finally, rather than in other contexts where we used HATP, we do not consider here that an agent is only capable to perform one action at a time. This allows the human to choose the order of his action when there is no impact in the global plan.

5.6 Shared Plans execution

We will now see in more detail how the robot executes the flexible Shared Plans obtained.

5.6.1 Plan maintaining

First, the robot needs to be able to follow the Shared Plan execution and to determine which actions need to be executed and which actions need to be left for later.

As said before, the actions composing a Shared Plan can be decomposed as:

$$A_p =< A_{prev}, A_{cur}, A_{next}, A_{later} >$$

By default, when a Shared Plan is computed by the robot, all actions are put

in A_{later}. When the robot performs an action or detects an action execution from a human, the executed action goes in A_{cur} and, at the end of the execution, the action goes in A_{prev} with a label equal either to:

- DONE if the execution has been successful,
- FAILED if the execution has not been successful,
- ABORTED if the robot had to stop the execution for an external reason.

An action will be put in A_{next} if all previous actions in the plan are DONE (based on causal links) and its preconditions are checked:

$$a \in A_{next} \Leftrightarrow Precs_a \in WS \ \& \ (\forall l \in L_p \mid next_l = id_a,$$
$$\exists \ ap \in A_{prev} \mid (id_{ap} = prev_l \ \& \ label_{ap} = DONE))$$

The $UPDATE\_PLAN$ operator updates the state of each action of the plan and the $GET\_NEXT\_ACTIONS$ operator returns the actions in A_{next}.

5.6.2 Action selection

Once the robot knows which actions need to be executed, it needs to choose one from the set of actions it can execute. To do so, it uses the $SELECT\_ACTION\_TODO$ operator which returns the action with the highest priority:

$$\underset{a \in \{A^R_{next} \cup A^X_{next}\}}{\operatorname{argmax}} \ priority(a)$$

Priorities used: In our case, we have chosen to give higher priority to the actions allocated to the robot compared to those not yet allocated. In general for this work, we have made the choice to postpone as much as possible the decisions to be made by the robot. Indeed, this is done in order to give as much latitude as possible to the human, which allows him to take the initiative until the last possible moment. In the current implementation of our system, the priorities of the different actions of the robot are the same, so the robot will simply select one. However, there is a possibility to later integrate costs as, for example, select the action the farthest of what the human is currently doing. Concerning the priorities of not allocated actions, we still follow the principle to postpone as much as possible the robot decision. To do so, we put a higher priority on what we call *analogous* actions. Two actions are considered *analogous* when they have exactly the same decomposition (same action name and same parameters). Indeed, as there is there is an action that has to be achieved several times, putting a higher priority on *analogous* actions allows to robot to execute one (and advance in the plan) while letting to the human the possibility to perform the other one.

5.6.3 Action allocation

Once a not yet allocated action is selected, the robot needs to decide if it should execute it or not ($ALLOCATE\_ACTION$ operator). To do so, the robot first looks for the possible actors of this action: agents which verify the preconditions of the action and which are not already busy. Note that even if the action was not allocated by HATP it is possible that there is only one possible actor. In this case, the robot automatically allocates the action to this agent. For example if the human is currently busy and there is a not allocated action to perform, the robot will execute it. If there is more than one possible actor for the action, the robot follows the algorithm 4.

Algorithm 4 Action allocation: $SP \leftarrow ALLOCATE\_ACTION(SP, a, WS, Prefs)$

Require: $a \in A^X_{next}$
  if $cost(a, R) << cost(a, H)$ **then**
    $actor \leftarrow robot$
  else if $cost(a, H) << cost(a, R)$ **then**
    $actor \leftarrow human$
  else if mode = negotiation **then**
    $answer \leftarrow ASK(a, H)$
    if answer = yes **then**
      $actor \leftarrow human$
    else
      $actor \leftarrow robot$
    end if
  else
    $actionPerformed \leftarrow WAIT\_ACTION(a, t)$ ▷ *adaptation mode*
    if $actionPerformed$ **then**
      $actor \leftarrow human$
    else
      $actor \leftarrow robot$
    end if
  end if
  return $actor$

First, the robot compares the estimated cost for the human and for itself to perform the action. If it considers it significantly more costly for the human to perform the action, it will allocate the action to itself. Then, we have developed two possible modes for the robot. In the first mode, called **negotiation** mode, the robot directly asks its human partner if he wants to perform the action and then allocates the action according to his answer. In the other mode, called **adaptation** mode, the robot waits a certain amount of time, and, if the human does not take the initiative to perform the action, it executes it.

Allocating an action to an agent can lead to other actions being automatically allocated. For this reason, after each allocation of an action, a new plan is built

taking into account the possible allocations of the actions remaining in A_X.

Costs used for action selection ($cost(a, R)$ and $cost(a, H)$): In the current implementation, we use a cost concerning the *analogous* actions (see description in the previous subsection). These *analogous* actions will have a lower cost for the robot to execute them leading the robot to automatically execute one of them. Indeed, as there is several time the same action to perform, the robot can execute one while letting to the human the possibility to perform the other(s).

Other costs based on human preferences or on the context can be considered. For example, we can imagine having a list of actions the human likes to perform and another he dislikes. The robot can then allocate the actions following these preferences.

5.6.4 Action execution

Once the robot has decided to execute an action ($EXECUTE$ operator), it needs to be able to deal with the *similar* objects introduced before. To do so, we keep the principle that the robot waits until the last moment to take a decision. For example, if the robot and the human have to pick objects and place them in one out of several similar placements, the robot will first pick an object and only after choose a placement to place it. Then, when the robot has to choose an object, it will choose the one it considers the less costly.

Finally, if the human approaches an object which is involved in the current robot action (e.g. if he places an object in a placement the robot has chosen), the robot first halts its action. Then, the robot looks if it can find another *similar* object. If it finds one, it continues its action with this object. If not, it waits for the human to retreat from the object, and if the human actions did not lead to a new plan it continues its action if possible.

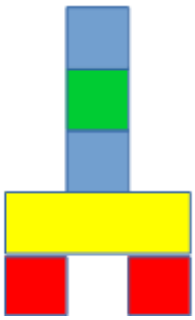
Costs used for objects selection: Here we choose to put a lower cost on objects accessible only by the robot (we still want to let the maximum choices to the human). Then, we use a simple cost based on distance. In our cost, we get the distances between the agents hands (here we have chosen the right hand) and objects. For objects accessible only by the robot the costs will be proportional to the distance between the robot hand and the objects, leading the robot to choose the closest one. Concerning objects accessible also by the human, the costs will be inversely proportional to the distance between the human hand and the objects, leading the robot to choose the farthest object from the human to minimize the efforts for the human to reach the objects left.

5.7 Results

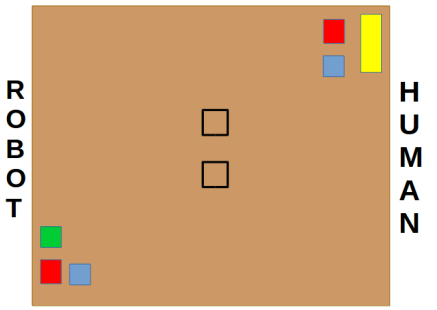
5.7.1 Task

To illustrate the work done in this chapter, we use a task adapted to the manipulation abilities of a PR2 robot and inspired from the one in [Clodic 2014]. A human

and a robot have to build a blocks construction as represented in Fig. 5.1(a). At the beginning of the task, the robot and the human have several colored blocks they can access as in a set up like the one illustrated in Fig. 5.1(b). Two identical placements are set on the table to indicate where to put the two red cubes.



(a) Goal of the task (side view)



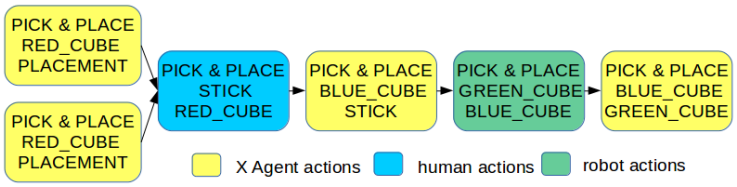
(b) One possible initial set-up (top view)

Figure 5.1: Description of the blocks building task. The human and the robot have to build the stack together. We assume that the robot and the human know where all the available blocks are. We would like the robot to adapt as much as possible to the human actions and decisions while avoiding useless or tiresome verbal interactions

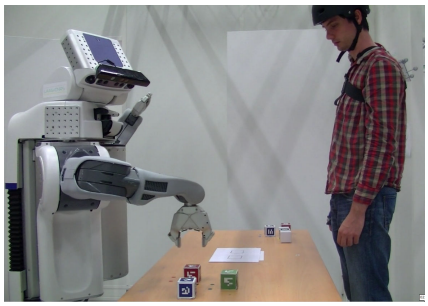
5.7.2 Illustrative example

We will first present one possible scenario of the task described earlier which illustrates well the benefits of this work.

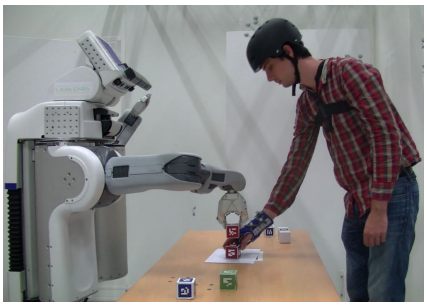
The presented scenario starts with the set-up in Fig. 5.1(b). The plan produced by HATP for this set-up can be found in Fig. 5.2(a). As this plan starts with two *analogous* actions for the *X agent* (place a red cube into a placement), the robot selects one and starts to execute it. So, the robot picks the red cube (Fig. 5.2(b)) and, at the same time, looks for the consequences of its choice in the plan. As both agents own only one red cube, in the new plan computed by the robot (Fig. 5.2(d)) the human needs to place the second red cube. After picking its red cube the robot starts to place it on the placement to its right. However, the human picks his red cube and places it in the very same placement (Fig. 5.2(c)). So, the robot stops its movement and adapts by placing its cube in the other placement (Fig. 5.2(e)). Then, the human places the stick on the red cubes. In this scenario, we have chosen to set the robot into the **negotiation** mode. As the next action is allocated to the *X agent*, the robot asks the human if he wants to do it (*"Do you want to place the blue cube?"*). The human answers yes, leading the robot to compute the new plan in Fig. 5.2(f) where the human have to place the first blue cube and the robot the second one. Finally, the human and the robot perform their last actions and achieve the goal.



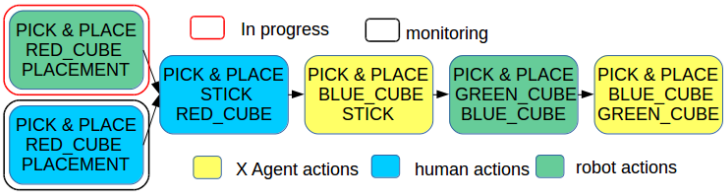
(a) Initial plan



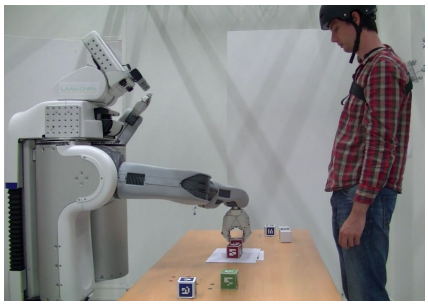
(b) The robot chooses to put the red cube in the placement to its right



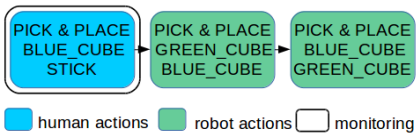
(c) The human places his cube in the placement the robot chose



(d) Second computed plan



(e) The robot adapts by changing its placement choice



(f) Third computed plan

Figure 5.2: The human and the robot build a blocks construction together. The robot adapts its behavior to the human actions.

5.7.3 Quantitative results

In order to evaluate our system, we run it in simulation using the blocks building scenario. Different set-ups were used as initial state of the task: we randomized the number of cubes of each color in the environment and their position (accessible by the robot or the human). The robot was confronted to a simulated human with

different types of behaviors. This simulated human performs all actions that are feasible only by him and answers robot questions. When confronted to an *X agent* action, he:

- chooses to perform it with 50% chance **(50%-case)**,
- systematically chooses to perform it **(hurry-case)**,
- systematically chooses not to perform it **(lazy-case)**.

Then, we settled two different human behaviors:

- **the "kind" human (case=K)** who adapts his behavior to what the robot verbalizes (ie does an action if the robot asks him and stops an action if the robot says it will perform it)
- **the "stubborn" human (case=S)** who does not react nor comply to robot verbalization (he will not change his decision whatever the robot says).

We compared 4 different modes:

- using the original system, called Reference System (RS), with all decisions and instantiations performed at planning time:
  - **RS- mode:** the robot verbalizes nothing (unless it is strictly necessary)
  - **RS-all mode:** the robot informs the human when he has to perform an action and when it will act,
- using the proposed system, called New System (NS):
  - **NS-N:** the robot uses the **Negotiation** mode previously defined when a decision need to be made concerning *X agent* action,
  - **NS-A:** the robot uses the **Adaptation** mode.

We measured:

- *the number of verbal interactions* between the human and the robot (either an information given by the robot or question asked), in Tab. 5.1.
- *the number of human/robot incompatible decisions*: either both decide to perform the same action (and the robot stops its own action to avoid the conflict) or both decide not to perform the action (the robot first asks the human to perform the action after a predefined time and, if after another period the human has still not executed the action, the robot looks for a new plan where it can proceed), in Tab. 5.1.

We also measured execution time but no significant difference was found between the different conditions. Indeed, this criterion is not pertinent here since, as all actions concern the same stack, they need to be performed one after the other. Consequently, there is no significant difference time between the different options.

	RS-none	RS-all	NS-N	NS-A
50%-K	0.6 (0.52)	0.0 (0.0)	**0.0 (0.0)**	**0.0 (0.0)**
hurry-K	0.3 (0.48)			
lazy-K	0.9 (0.32)			
50%-S	0.5 (0.53)	0.6 (0.52)		
hurry-S	0.3 (0.48)	0.3 (0.48)		
lazy-S	0.9 (0.32)	0.9 (0.32)		

Table 5.1: **Number of incompatible decisions** between the human and the robot (i.e. either both agents decide to perform the same action or both decide not to perform a given action). Results for the reference system (RS) and the proposed system (NS-N for the negotiation mode and NS-A for the adaptation mode). The numbers correspond to means in 10 runs and their associated standard deviations.

	RS-none	RS-all	Neg	Adapt
50%-K	0.4 (0.52)	6.0 (0.0)	**1.2 (0.0)**	**0.0 (0.0)**
hurry-K	0.0 (0.0)			
lazy-K	0.9 (0.32)			
50%-S	0.4 (0.52)	6.4 (0.52)		
hurry-S	0.0 (0.0)	6.0 (0.0)		
lazy-S	0.9 (0.32)	6.9 (0.32)		

Table 5.2: **Number of verbal interactions**: question asked by the robot in the negotiation mode or an information given with the reference system. Results for the reference system (RS) and the proposed system (NS-N for the negotiation mode and NS-A for the adaptation mode). The numbers correspond to means in 10 runs and their associated standard deviations.

Reference System performance: The verbalizations in the RS-none mode corresponds to the case where the human and the robot both choose not to execute the action: the robot tries to solve the conflict by asking the human to execute the action. Because the robot does not inform about its decisions, the number of verbal interactions is low in this mode. However, due to the same reason, there is several incompatible decisions in each conditions. The RS-all mode avoids incompatible decisions with the "kind" human. However, the number of verbal interaction is high (6 as the number of actions to execute in the task and so to verbalize). With the "stubborn" human, even if the robot informs the human, incompatible decisions remains. The number of verbal interaction also increases as, when the human does not want to perform the action, as it is stubborn, the robot needs to compute a new plan where it executes the action, and so inform about the new action.

New System performance: We can see that the robot is able to avoid conflicts in all cases without being too talkative (or without being talkative at all for the adaptation mode). Moreover, the efficiency of the system is not degraded with the "stubborn" human: the system allows the human to execute the actions he

wants without an increase of verbal interaction. Finally, here the adaptation mode performs better than the negotiation one since the human is simulated and always performs his actions in time. However, in a real context, the negotiation mode would certainly have the benefit to ensure the absence of conflicts even if the robot is a little more talkative. Moreover, a human would surely be more comfortable with a robot which directly asks when (and only when) there is a decision to take compared to a robot which has unnecessary waiting time. Such measure of "satisfaction" cannot be easily simulated and further experiments will be done with real humans.

5.8 Conclusion

In this chapter we have shown how we enable the robot to compute and execute more flexible Shared Plans. In these new plans, the needed decisions on who will execute an action and with which objects are let to the execution. A number of the presented algorithms involve cost estimation in order to decide between options. In the current system, simple costs are used but could be easily replaced by more elaborate ones. For instance, a finer estimation of action costs based on geometric reasoning and human efforts would allow better informed choice for action or object selection. Another interesting issue would be to integrate the estimation of accumulated costs of all actions remaining in the plan.

The benefits of this work have been demonstrated with an illustrative example and simulation results. We will show in Chapter **??** more complete simulation results which also include the work of the previous chapter as well as results with the system running in a real situation with real humans.

Bibliography

[Allen 2002] James Allen and George Ferguson. *Human-machine collaborative planning.* In Third International NASA Workshop on Planning and Scheduling for Space, 2002. (Cited in page 3.)

[Baraglia 2016] Jimmy Baraglia, Maya Cakmak, Yukie Nagai, Rajesh Rao and Minoru Asada. *Initiative in robot assistance during collaborative task execution.* In HRI, ACM/IEEE, 2016. (Cited in page 3.)

[Botelho 1999] Sylvia C Botelho and Rachid Alami. *M+: a scheme for multi-robot cooperation through negotiated task allocation and achievement.* In Robotics and Automation, IEEE, 1999. (Cited in page 3.)

[Chakraborti 2015] Tathagata Chakraborti, Gordon Briggs, Kartik Talamadupula, Matthias Scheutz, David Smith and Subbarao Kambhampati. *Planning for serendipity-altruism in human-robot cohabitation.* In ICAPS Workshop on Planning and Robotics (PlanRob), 2015. (Cited in page 2.)

[Chakraborti 2016] Tathagata Chakraborti, Yu Zhang, David E Smith and Subbarao Kambhampati. *Planning with resource conflicts in human-robot cohabitation.* In AAMAS, 2016. (Cited in page 2.)

[Chien 2000] Steve A Chien, Russell Knight, Andre Stechert, Rob Sherwood and Gregg Rabideau. *Using Iterative Repair to Improve the Responsiveness of Planning and Scheduling.* In AIPS, 2000. (Cited in page 3.)

[Cirillo 2010] Marcello Cirillo, Lars Karlsson and Alessandro Saffiotti. *Human-aware task planning: an application to mobile robots.* IST, 2010. (Cited in page 2.)

[Clodic 2009] Aurélie Clodic, Hung Cao, Samir Alili, Vincent Montreuil, Rachid Alami and Raja Chatila. *Shary: a supervision system adapted to human-robot interaction.* In Experimental Robotics, pages 229–238. Springer, 2009. (Cited in page 3.)

[Clodic 2014] Aurélie Clodic, Rachid Alami and Raja Chatila. *Key Elements for Human-Robot Joint Action.* Frontiers in AI and Applications, 2014. (Cited in page 12.)

[Devin 2017] Sandra Devin, Aurélie Clodic and Rachid Alami. *About Decisions During Human-Robot Shared Plan Achievement: Who Should Act and How?* In ICSR'17. to appear, 2017. (Cited in page 2.)

[Fiore 2014] Michelangelo Fiore, Aurélie Clodic and Rachid Alami. *On planning and task achievement modalities for human-robot collaboration.* In ISER, 2014. (Cited in page 3.)

[Gerkey 2002] Brian P Gerkey and Maja J Mataric. *Sold!: Auction methods for multirobot coordination.* IEEE Trans. on Robotics and Automation, 2002. (Cited in page 3.)

[Gerkey 2004] Brian P Gerkey and Maja J Matarić. *A formal analysis and taxonomy of task allocation in multi-robot systems.* The IJRResearch, 2004. (Cited in page 3.)

[Gombolay 2015] Matthew C Gombolay, Reymundo A Gutierrez, Shanelle G Clarke, Giancarlo F Sturla and Julie A Shah. *Decision-making authority, team efficiency and human worker satisfaction in mixed human–robot teams.* Autonomous Robots, 2015. (Cited in page 2.)

[Hoffman 2007] Guy Hoffman and Cynthia Breazeal. *Effects of anticipatory action on human-robot teamwork efficiency, fluency, and perception of team.* In HRI, ACM/IEEE, 2007. (Cited in page 3.)

[Joyeux 2009] Sylvain Joyeux, Rachid Alami, Simon Lacroix and Roland Philippsen. *A plan manager for multi-robot systems.* IJRR, 2009. (Cited in page 3.)

[Karpas 2015] Erez Karpas, Steven James Levine, Peng Yu and Brian C Williams. *Robust Execution of Plans for Human-Robot Teams.* In ICAPS, pages 342–346, 2015. (Cited in page 3.)

[Lallée 2013] Stéphane Lallée, Katharina Hamann, Jasmin Steinwender, Felix Warneken, Uriel Martienz, Hector Barron-Gonzales, Ugo Pattacini, Ilaria Gori, Marc Petit, Giorgio Metta*et al. Cooperative human robot interaction systems: IV. Communication of shared plans with Naïve humans using gaze and speech.* In IROS, IEEE/RSJ, 2013. (Cited in page 2.)

[Lallement 2014] Raphaël Lallement, Lavindra de Silva and Rachid Alami. *HATP: An HTN Planner for Robotics.* CoRR, 2014. (Cited in pages 2, 4, and 8.)

[Levine 2014] Steven James Levine and Brian Charles Williams. *Concurrent Plan Recognition and Execution for Human-Robot Teams.* In ICAPS, 2014. (Cited in page 3.)

[Milliez 2016] Grégoire Milliez, Raphaël Lallement, Michelangelo Fiore and Rachid Alami. *Using human knowledge awareness to adapt collaborative plan generation, explanation and monitoring.* In HRI ACM/IEEE, 2016. (Cited in page 3.)

[Mohseni-Kabir 2015] A. Mohseni-Kabir, C. Rich, S. Chernova, C. L. Sidner and D. Miller. *Interactive Hierarchical Task Learning from a Single Demonstration.* In HRI, ACM/IEEE, 2015. (Cited in page 3.)

[Molineaux 2010] Matt Molineaux, Matthew Klenk and David W Aha. *Goal-driven autonomy in a Navy strategy simulation.* Technical Report, DTIC Document, 2010. (Cited in page 4.)

[Petit 2013] Marc Petit, Stéphane Lallée, J-D Boucher, Grégoire Pointeau, Pierrick Cheminade, Dimitri Ognibene, Eris Chinellato, Ugo Pattacini, Ilaria Gori, Uriel Martinez-Hernandez*et al. The coordinating role of language in real-time multimodal learning of cooperative tasks.* Autonomous Mental Development, IEEE, 2013. (Cited in page 3.)

[Roberts 2016] Mark Roberts, Ron Alford, Vikas Shivashankar, Michael Leece, Shubham Gupta and David W Aha. *Goal reasoning, planning, and acting with actorsim, the actor simulator.* In ICAPS Workshop on Planning and Robotics (PlanRob), 2016. (Cited in page 4.)

[Shah 2011] Julie Shah, James Wiken, Brian Williams and Cynthia Breazeal. *Improved human-robot team performance using chaski, a human-inspired plan execution system.* In Proceedings of the 6th international conference on Human-robot interaction, pages 29–36. ACM, 2011. (Cited in page 3.)

[Sorce 2015] Marwin Sorce, Grégoire Pointeau, Maxime Petit, Anne-Laure Mealier, Guillaume Gibert and Peter Ford Dominey. *Proof of concept for a user-centered system for sharing cooperative plan knowledge over extended periods and crew changes in space-flight operations.* In RO-MAN, IEEE, 2015. (Cited in page 3.)

[Talamadupula 2014] Kartik Talamadupula, Gordon Briggs, Tathagata Chakraborti, Matthias Scheutz and Subbarao Kambhampati. *Coordination in human-robot teams using mental modeling and plan recognition.* In IROS, IEEE/RSJ, 2014. (Cited in page 2.)

[Tambe 1997] Milind Tambe. *Agent Architectures for Flexible, Practical Teamwork.* In 14th National Conf. on AI, 1997. (Cited in page 3.)

[Warnier 2012] Mathieu Warnier, Julien Guitton, Séverin Lemaignan and Rachid Alami. *When the robot puts itself in your shoes. managing and exploiting human and robot beliefs.* In RO-MAN, 2012 IEEE, pages 948–954. IEEE, 2012. (Cited in page 2.)