FP6-IST-002020

COGNIRON

The Cognitive Robot Companion

Integrated Project

Information Society Technologies Priority

D6.2.1
Report on Paradigms for Decisional Interaction

Due date of deliverable: 31/12/2004
Actual submission date: 31/12/2004

Start date of project: January 1st, 2004 **Duration:** 48 months

Organisation name of lead contractor for this deliverable:
LAAS-CNRS

Revision: Final
Dissemination Level: PU

Executive Summary

The documents reports on our preparatory work for the development of a framework for human-robot collaborative planning and associated topics: study of relevant representations and algorithmic issues.

The main topic is to investigate paradigms for human-robot shared decision in the context of the interactive robot companion.

We have studied how the robot can elaborate plans that will allow it to select and perform its tasks while taking into account explicitly the constraints imposed by the presence of humans and their needs and preferences.

We have considered a scheme where the robot plans for itself and for the human in order

- to assess the feasibility of the task (at a certain level) before performing it
- to share the load between the robot and the human
- to explain/illustrate a possible course of actions

Relation to the Key Experiments

The work developed here will conduct to the development of concepts and algorithms that will be implemented and illustrated in the Key Experiments: paradigms for human-robot decisional interactions and a planner that will allow to robot to select and perform it high level tasks in a socially acceptable way.

1 Introduction

This workpackage focuses on human-robot decisional interaction. We have investigated how a situation or a request from a human will be transformed into a goal and how goals satisfaction and the associated tasks elaboration and achievement can be seen as a process that is conducted collaboratively by the human and the robot.

We add complementary requirements that should allow robot intentions to be “legible” at different levels of abstraction and also allow the human to influence the robot decisional processes.

We have tried to derive such schemes through a study of the task-oriented human-robot interaction processes induced by the scenarios of the key experiments.

The concept of teamwork as developed by [5] and further elaborated in various ways constitutes for us a relevant framework in which decisional human-robot issues can expressed.

We have studied how high level robot decisional activities and mainly planning skills should be developed in order to allow it to act as a companion. We concentrated on a planner that is able to take into account “social constraints”: plans that take into account human preferences, that are acceptable and easily legible in terms of intention.

Another aspect is the elaboration of a framework for sharing decisions between the robot and the human and more generally for collaborative problem solving. This is of course linked to dialogue but also will also have consequences on the robot high-level decisional processes.

2 High-level planning abilities in the HRI context

Most papers on HRI address various ways of performing teleoperation or path planning aid. Only a limited number of papers consider the robot and the human as actors (or agents) who can collaborate to achieve common goals. The idea explored in the current report is that it is possible to have the robot make choices at a high abstraction level to synthesize plans which are acceptable for the human. So our works deals with the consideration of the human in the robot decisional process, and more especially on how the robot can take into account the human in its planning process. We will see how the robot can represent the human as constraints for the planning process and we will illustrate our first studies with an example.

It is important to note that in this perspective, and in a first step, the plan is entirely made by the robot. This is not opposed to a process where the human may take a key role in the global decisional process. Indeed, in a second phase, we will also consider situations in which the human can influence the planning process with advice/orders. If we want the robot to make acceptable plans for the human, it is necessary that the robot has a model of the human. This model must specify constraints related to the human, we assume in this report that this model has been acquired/learned by the robot living near the human.

2.1 Principles

The robot planning process must obey to some rules if we want the human to accept the plans. Various criteria can be defined to state that a plan elaborated by the robot is a good plan: actions feasibility, realization difficulty of actions, pleasure[1] in actions realization, relationship between the human and the robot. For instance, it is necessary that the plan does not put the human in a position of a “slave” in front of the robot. Other key criteria are legibility of actions with respect to human conventions, coherence of the actions (the robot actions and their sequencing must seem ”logic” for the human), the need to communicate at some specific steps, etc.

In our approach, we would like to transform such considerations as constraints on the planning process. The planning process takes in input two elements: an initial set of facts and a domain definition in which are defined all possible actions which can affect the facts data base.

In order to investigate further and to test our ideas, we have made some preliminary design choices in order to define a context in which we can express high-level planning related to COGNIRON .

In this context, the human model is synthesized in these two elements: the domain definition contains the actions the human can perform together with their “costs” in various contexts. The robot is represented in a similar way: a set of possible actions and an associated set of costs, the difference between a human model an a robot model comes from the interpretation of costs. A cost for a robot action symbolizes the difficulty the robot has to realize the action whereas a cost for a human action represents a mix between realization difficulty and realization pleasure/convenience. This allows to elaborate basic metrics for the planner.

The planning process is assumed to use these costs and to try to synthesize plans which have the least cost. The idea is to make plans according to the least human effort without forgetting the pleasure that a human can have to achieve by himself some actions even if they can be performed by the robot.

We have tested our approach on different planners developed in recent AI planning competitions. As we need to optimize the total cost of plans we cannot use a planner based on local heuristics and we must use a planner which considers explicitly the notion of costs in its planning process. After several tentative investigations, our choice is the HTN (Hierarchical task Networks) planner SHOP2[1] mainly because it permits to specify costs for each action and it can produce plans with the least total cost. It is important to note that as SHOP2 does not explicitly manage time, in the actual state of our works we will not consider it. In this first investigation, our work is limited to STRIPS-like domains.

2.2 Formalization attempt

We can try to make a first formalization of our ”actors” model, the actors can be of two types: human or robot. If we note A_i the set of actions that an actor a_i can achieve and C_i the set of associated costs we can write:

$$M_{a_i} = < A_i, C_i > \text{ where } M_{a_i} \text{ is the model of the actor } a_i$$

[1] We mean here by ’pleasure’, the interest, the preference or the benefit that a human may have for an action or a situation

It is clear that if we want our model to be realistic the actions costs must have the possibility to be depend on the context, so in future we will note the costs set by C_i^{ctxt} to exhibit this property. So we can try a first attempt of our actors ”team” by writing:

$$M =< M_{a_1}, ..., M_{a_n} > \text{ where n is the actors number and } M_{a_i} =< A_i, C_i^{ctxt} >$$

However this model needs to be enriched because it cannot permit to consider all the aspects which make a plan to be good. The costs permit us to represent the difficulty and the pleasure an actor has in an action realization, to use costs depending on the context permit us a better approximation of the difficulty and the pleasure.

Although this approach permits to consider two important aspects linked to the acceptability of a plan by a human, it is not sufficient because some other criteria like social rules are not considered. With the model presented above, it is clear that the actions are chosen according to their cost in a given context but we cannot ”detect” actions series and/or situations which may be unacceptable for the human. To do this we define two sets P_{states} and $P_{sequences}$ as follows:

- $P_{states} =< S_{SR_1}, C_{SR_1}^{ctxt} >$ where S_{SR_1} is a set of social rules which describe the ”undesired states” of the world and $C_{SR_1}^{ctxt}$ the set of associated costs which can depend on the context.
- $P_{sequences} =< S_{SR_2}, CO_{SR_2}, C_{SR_2}^{ctxt} >$ where S_{SR_2} is a set of social rules which describe the ”undesired sequences of actions”, CO_{SR_2} the set of associated conditions and $C_{SR_2}^{ctxt}$ the set of associated costs which can depend on the context.

Thus when an undesirable situation (i.e. a state of the world at a given moment) or chain of actions (i.e. a combination of actions that make the plan unacceptable with respect to social rules) is detected we apply a penalty on the total cost of the plan by increasing it with a suitable quantity.

So our model becomes:

$M =< M_{a_1}, ..., M_{a_n}, P_{states}, P_{sequences} >$ where n is the actors number, $M_{a_i} =< A_i, C_i^{ctxt} >$, P_{states} and $P_{sequences}$ are the sets of penalties defined above.

2.3 Illustration with an example

In order to illustrate the abilities of our approach we are going to consider a simple example. This example takes place in an apartment. We consider only two rooms: the living-room and the kitchen. There are two things in the kitchen: a sandwich and a mop. There are two actors present in the living-room at the beginning: ’human1’ and ’robot1’. Each actor has two ”hands” to take an object: the left or right hand/claw. Their common goals are to clean the living-room with the mop and to make the human have the sandwich.

It is important to remind that in this report we are interested by the decisional aspects of HRI and not by the geometric level. This means that we are concerned by action planning, role allocation and eventually coordination between actors.

It is also clear that, in order to allow role allocation, dialogue about plans, human-robot 'negotiation', we have to choose a level of abstraction where we can define a common language of 'generic actions' i.e. actions that can be performed/understood by the robot and the human.

Each actor can perform the following actions:

Action	**Predicate syntax**	**Syntax meaning**
Move	(move ?name ?place1 ?place2)	The actor ?name moves from ?place1 to ?place2.
Take	(take ?name ?thing ?side ?place)	The actor ?name takes the object ?thing in his ?side hand in the room ?place.
Drop	(drop ?name ?thing ?side ?place)	The actor ?name drops the object ?thing from his ?side hand in the room ?place.
Give	(give ?name1 ?side1 ?name2 ?side2 ?thing)	The actor ?name1 gives the object ?thing from his ?side1 hand to ?side2 hand of ?name2.
Mop use	(mop_use ?name ?mop ?place)	The actor ?name uses ?mop in the room ?place.

The different costs of the actions are:

Action	**Robot1 costs**	**Human1 costs**
Move	1	$2 * (6 - HUMAN\_MOOD)$
Take	1	$1 * (6 - HUMAN\_MOOD)$
Drop	1.5	$1 * (6 - HUMAN\_MOOD)$
Give	6.6	$1 * (6 - HUMAN\_MOOD)$
Mop use	22.7	$1 * (6 - HUMAN\_MOOD)$

The HUMAN_MOOD parameter symbolizes the mood of the human. For the sake of simplicity it is measured here on 1 to 5 scale in which a score of 1 represents a bad mood so the human is not really cooperative and in the opposite way a score of 5 means that the human is in a good mood and is ready to participate to the common goals achievement. More elaborate models and cost estimation functions can be used. The human mood represents a combination between two aspects: the inclination part is relatively fixed in time, it depends on age, preferences, etc. whereas the contextual mood part is very unstable in time, it represents the human mental state at a given moment depending on stress level, social or activity context, etc...

We are now going to present plans obtained in different situations. We remind that we are in a STRIPS-like task planning domain so all the actions are made in a sequential way.

2.3.1 Situation 1: the human is cooperative

In this first situation, the human is in a good mood (we have HUMAN_MOOD = 5) and there are not any penalties of any type that is there are no undesirable world states or actions sequences (i.e $P_{states} = \emptyset$ and $P_{sequences} = \emptyset$). A plan obtained is:

```
(move human1 livingroom kitchen)
(take human1 sandwich right kitchen)
(take human1 mop left kitchen)
(move human1 kitchen livingroom)
(mop_use human1 mop livingroom)
```

The total cost of this plan is 7.0. All the actions are done by the human because he has pleasure to do these actions and he can achieve them more efficiently than the robot.

2.3.2 Situation 2: the human is not already cooperative

In this situation, the human is in bad mood (we have HUMAN_MOOD = 1) and there are no penalties anymore. The obtained plan is:

```
(move robot1 livingroom kitchen)
(take robot1 sandwich right kitchen)
(take robot1 mop left kitchen)
(move robot1 kitchen livingroom)
(drop robot1 sandwich right livingroom)
(drop robot1 mop left livingroom)
(take human1 sandwich right livingroom)
(take human1 mop left livingroom)
(mop_use human1 mop livingroom)
```

The total cost of this plan is 22.0. As the human does not really want to be involved in the goals achievement a big part of the plan will be executed by the robot. The total cost is more important because the robot achieves its actions with bigger costs than the human in situation 1. However it is important to note that it is cheaper (in terms of costs) for the robot to drop the objects and leave the human take them.

2.3.3 Situation 3: insertion of a first type of social rule

We are now in a situation similar to the situation 2 except that we want to prevent the fact that it is cheaper for the robot to drop the objects and make the human take them instead of giving them directly to the human. To prevent this we add a penalty in $P_{sequences}$:

- S_{SR_2} = $\{\{$(drop ?name1 ?thing ?side1 ?place) (take ?name2 ?thing ?side2 ?place)$\}\}$
- CO_{SR_2} = $\{\{$(and (robot ?name1) (human ?name2))$\}\}$
- $C^{ctxt}_{SR_2}$ = * $\{30\}$

This means that when the action ”take” follows the action ”drop”, if these actions concern the same ?thing in the same ?place and if the action drop is performed by a robot and the action take by a human, we have to add 30 to the total cost of the plan. To see how we have made this in SHOP2 see Annex 1. The set P_{states} is already empty. A plan obtained is:

```
(move robot1 livingroom kitchen)
(take robot1 sandwich right kitchen)
(take robot1 mop left kitchen)
(move robot1 kitchen livingroom)
(give robot1 right human1 left sandwich)
(give robot1 left human1 left mop)
(mop_use human1 mop livingroom)
```

The total cost of this plan is 22.2, as we can observe in the plan above the goals are achieved in a similar way to the situation 2 but the robot uses action 'give' instead of a combination 'drop-take'. The plan created in this situation seems to be more acceptable than the plan created in the situation 2, however we can see that the plan implies that during a moment the robot will have the sandwich and the mop simultaneously. The person might consider this as undesirable because he fears the robot to make a contact between food and a cleaning object.

2.3.4 Situation 4: insertion of a second type of social rule

To prevent the situation where the robot will have simultaneously food in a hand and a cleaning object in the other, we add a penalty in the set P_{states} as follow:

- S_{SR_1} = {{(robot ?name) and (have ?name sandwich) and (have ?name mop)}}
- $C_{SR_1}^{ctxt}$ = {20}

This means that when in the data base the facts '(have ?name sandwich)' and '(have ?name mop)' coexist, we have to add 20 to the total cost of the plan. One can see in Annex 1 how we have implemented this in SHOP2[1]. The set $P_{sequences}$ is identical as in situation 3. The obtained plan is:

```
(move robot1 livingroom kitchen)
(take robot1 sandwich right kitchen)
(move robot1 kitchen livingroom)
(give robot1 right human1 left sandwich)
(move robot1 livingroom kitchen)
(take robot1 mop right kitchen)
(move robot1 kitchen livingroom)
(give robot1 right human1 right mop)
(mop_use human1 mop livingroom)
```

The total cost of this plan is 24.2. We can see that the robot "prefers" to make two trips instead of infringing the social rule that we have just introduced.

2.3.5 Situation 5: domain modification

We now assume that there is a problem with the kitchen tiling that makes it impossible for the robot to go in the kitchen. We always have a human in a bad mood and the two social rules introduced in the situation 3 and 4. The obtained plan is:

```
(move human1 livingroom kitchen)
(take human1 sandwich right kitchen)
(take human1 mop left kitchen)
(move human1 kitchen livingroom)
(mop_use human1 mop livingroom)
```

The total cost of this plan is 35.0. With this situation we can see that our approach permits to consider relevance and feasibility: the human is not really motivated in the goals achievement but is constrained to do it because the robot is incapable to do it itself.

2.4 Discussion

This study have confirmed, for us, the relevance of this level and of the types of considerations that should be taken into account when building robot plans in this context. This should be the basis for task planning but also, as we mentioned, role allocation, dialogue about plans, human-robot 'negotiation',

The simple example that we have presented shows the capacities of our approach: it is possible to make plans taking in account different criteria in order to have the plan accepted by the human. It is possible to define relatively easily social rules (to be considered at this level) by describing world states or actions sequences in a given context.

However the current planning systems present limitations that we have to consider. The first one is computation time. As our approach requires to get plans that minimize some cost criteria, it is necessary to explore large amounts of potential plans.

The second limitation is the role of the human in the planning process. We have shown that we make plans according to models of actors and social rules but the human cannot influence the planning process itself. We would like to provide means that may allow the human to influence the planning process by giving (on-line) some advice or by adding new constraints that the robot didn't perceive.

3 Future Work

In the next phase of our work we will consider temporal constraints in scenarios. Indeed, it is necessary to consider explicitly the time related issues for plan specification as well plan realization. The consideration of time will introduce new criteria of acceptance. For instance, it will allow to express the necessity to elaborate plans where some situations cannot last more than a maximum duration. The converse may also happen.

Another issue will consider the role of the human in the planning process, it is an important point because the human can add constraints which influence in an important way ”solution plans”. Another important point concerning the role of the human in the planning process loop is about advice because they can permit to reduce the size of the search space and so accelerate considerably the speed of the plan elaboration. There are a some interesting studies about human influence in planning process. We can cite the work of Myers about the Advisable Planning (see [7], [8] and [9]), the contribution of Cohen and Levesque on the joint intentions (see [3], [4] and [5]) or the work of Grosz about the Collaborative Planning (see [10] and [11]).

According to our first results and the two points above, we will develop a planner in order to satisfy our design criteria. This planner use the notion of actions costs as it is in the center of our approach, but also time and various constraints. Another key issue will be about the feature of detecting given world states, sequences of actions under particular conditions and more generally ”chronicles” like it is considered in IxTeT (see [2]). The points of interest will be:

- Models: how will we represent the world, the human and the tasks? The HTN formalism seems to be a good way.
- Metrics: how will we express plan quality differently from a total cost?
- Constraints: what types of constraints will we need?
- Planning algorithm: finally, we have to devise a planning algorithm according to all the other choices and able to provide access to its current search state (cf. next point).

Concerning human preferences and social acceptance of robot behavior, we plan to apply recommendations that will be drawn from the user studies that UH conducts in the framework of Cogniron.

Besides, we would like the human to act on the plan elaboration process. This will have consequences on the planner specification especially on two points: the architecture and the knowledge representation. The first point is essential. The architecture of our planner will have to permit to the human to act on the plan creation in a natural way. For instance, Allen and Ferguson have proposed a hybrid architecture for collaborative planning in [13]. Their system is composed of several modules: the central component is a HTN Task Decomposer which permits to decompose goals into tasks and which produces ”Abstract Plans”, another module is in charge of temporal planning and the last module transforms abstract plans into explicit plans prior to their execution. Another interesting approach is the Mixed-Initiative Planning which has already been used in some specific planners as in [15]. The paradigm of the Mixed-Initiative planning is specifically dedicated to the problem where several participants cooperate to create plans. A fair amount of work has been done in the domain of path planning aid and we will study in its relevance an possible transposition to our framework.

Knowledge representation is also very important. We have to define what data will be available to the human and consequently how to extract them from the internal planning system. The HTN formalism seems to be a good candidate because it permits to show to the human a clear hierarchical decomposition of the tasks. Moreover it is easy for the human to make some specifications like ”I will achieve this part of the task” because he just has to specify the associated branch in the HTN. Another plan representation which seems to have good capacities is the representation based on arguments (see [14]). It is specifically designed for the Mixed-Initiative Planning but the concepts can be adapted in particular for the robot when it has to justify its choices.

4 References

References

[1] D. Nau, T.-C. Au, O. Ilghami, U. Kuter, W. Murdock, D. Wu, and F.Yaman. *SHOP2: An HTN Planning System*. JAIR, volume 20, pp. 379-404, 2003.

[2] S. Lemai. *IXTET-EXEC: planning, plan repair and execution control with time and resource management.* Thesis, Institut National Polytechnique, Toulouse, June 2004.

[3] P. R. Cohen, and H. J. Levesque. *Intention is choice with commitment.* Artificial Intelligence, volume 42, pages 213-261, 1990.

[4] H. J. Levesque, P. R. Cohen, and J. H. T. Nunes. *On acting together.* In Proceedings of the Eighth National Conference on Artificial Intelligence (AAAI-90), pages 94-99, 1990.

[5] P. R. Cohen, and H. J. Levesque. *Teamwork.* Nous, 25(4): 487-512, 1991.

[6] N. R. Jennings. *Specification and Implementation of a Belief- Desire-Joint-Intention Architecture for Collaborative Problem Solving.* In Int. Journal of Intelligent and Cooperative Information Systems, 2(3),pp 289-318, 1993.

[7] K. Myers. *Strategic advice for hierarchical planners.* In Principles of Knowledge Representation and Reasoning: Proceedings of the Fifth International Conference (KR ’96) (L. C. Aiello, J. Doyle, and S. C. Shapiro, eds.), Morgan Kaufmann Publishers, 1996.

[8] K. Myers. *Domain Metatheories: Enabling User-Centric Planning.* In Proceedings of the AAAI Workshop on Representational Issues for Real-World Planning Systems, 2000.

[9] K. Myers. *Planning with Conflicting Advice.* In Proceedings of the Fifth International Conference on AI Planning Systems (AIPS 2000), 2000.

[10] B. Grosz, and S. Kraus. *The Evolution of SharedPlans.* In Foundations and Theories of Rational Agencies, A. Rao and M. Wooldridge, eds. pp. 227-262, 1999.

[11] B. Grosz, and S. Kraus. *Collaborative Plans for Complex Group Action.* In Artificial Intelligence. 86(2), pp. 269-357, 1996.

[12] C. Breazeal, G. Hoffman, A. Lockerd. *Teaching and Working with Robots as a Collaboration.* In Proceedings of the Third International Joint Conference on Autonomous Agents and Multiagent Systems (AAMAS), 2004.

[13] J. Allen, G. Ferguson. *Human-Machine Collaborative Planning*. In Proceedings of the Third International NASA Workshop on Planning and Scheduling for Space, 2002.

[14] G. Ferguson, and J. Allen. *Arguing About Plans: Plan Representation and Reasoning for Mixed-Initiative Planning*. In Proceedings of the Second Conference on Artificial Intelligence Planning Systems (AIPS 1994), 1994.

[15] G. Ferguson, J. Allen, and Brad Miller. *TRAINS-95: Towards a Mixed-Initiative Planning Assistant.* In Proceedings of the Third Conference on Artificial Intelligence Planning Systems (AIPS 1996), 1996.

4.1 Applicable documents

Cogniron proposal.

Key experiments: D7.1

4.2 Reference documents

Deliverable D.6.1.1

Annex

Examples of social rules specifications

We provide here some details on how actions and complementary knowledge are specified in SHOP2. We illustrate this explanation with two examples and we also show how we specify social rules of the two types.

In SHOP2 the description of a planning domain includes operators and methods. Operators are similar to those of classical planning, they have an "add list" and a "delete list" to respectively add/delete facts in the facts data base when they are used. Whenever an operator is used in the current plan, its cost is evaluated and added to the plan cost. Methods permit to describe how to decompose a task into a set of sub-tasks. Methods specify preconditions to verify and a tasks list composed of operators and/or methods. In this way it is clear that methods can not directly modify data base; they need to invoke the operators. A method can have several tasks lists. In this case the planner checks the associated preconditions lists in the order and when all the preconditions of a list are true it is the corresponding task list which is used. If there are several methods with the same preconditions the planner considers this as a "OR" and chooses one of the methods. For more details see [1].

The operator TAKE without social rule:

```
;; action TAKE : ?name takes ?thing
;; basic operator
(:operator (!take ?name ?thing ?side ?place)
              ;; precondition
              ((take_cost ?name ?cost))
              ;; delete list
              ((in ?thing ?place))
              ;; add list
              ((have ?name ?thing ?side))
              ;; cost
              ?cost
)

;; "basic" method to call the basic operator
(:method (take ?name ?thing)
           ;; preconditions
           (:first (actor ?name) (thing ?thing) (side ?side) (place ?place)
                     (at ?name ?place) (in ?thing ?place)
                     (forall (?t) (thing ?t) (not (have ?name ?t ?side))) )
           ;; task list
           (:ordered (!take ?name ?thing ?side ?place))
)
```

To introduce a social rule in P_{states} we have modified the "basic method" as follows (the operator (!!apply_penalty ?pen) is a blank operator, it does not modify the facts basis but it has a cost of ?pen):

```
;; "basic" method to call the basic operator
(:method (take ?name ?thing)
           ;; preconditions: idem + ?name possesses a cleaning_object and food
           (:first (actor ?name) (thing ?thing) (side ?side) (place ?place)
                     (at ?name ?place) (in ?thing ?place)
                     (forall (?t) (thing ?t) (not (have ?name ?t ?side)))
                     ;; expression of the social rule
                     (robot ?name) (or ((food ?thing) (cleaning_object ?thing2)
                                              (possess ?name ?thing2))
                                             ((cleaning_object ?thing) (food ?thing2)
```

```
                                        (possess ?name ?thing2)))
                    (penalty_have_food_and_cleaning_object ?pen) )
          ;; task list
          (:ordered (!take ?name ?thing ?side ?place)
                       (:immediate !!apply_penalty ?pen))

          ;; preconditions: idem
          (:first (actor ?name) (thing ?thing) (side ?side) (place ?place)
                    (at ?name ?place) (in ?thing ?place)
                    (forall (?t) (thing ?t) (not (have ?name ?t ?side))))
          ;; task list
          (:ordered (!take ?name ?thing ?side ?place))
)
```

We define the task (reach_place_with_free_hands ?name ?side ?place) which permits to the actor ?name to be in the room ?place with the ?side hand free. So we can write the task ”obtain_object_from_someone” without social rule:

```
;; action OBTAIN_OBJECT_FROM_SOMEONE : ?name obtains ?thing from ?name2
;; (3 methods: ?name has already ?thing or "?name2 gives ?thing to ?name"
;; or "?name2 drops ?thing and ?name takes it")
(:method (obtain_object_from_someone ?name ?name2 ?thing)
          ;; precondition : ?name is different from ?name2, ?name has ?thing
          ((actor ?name) (actor ?name2) (thing ?thing) (possess ?name ?thing))
          ;; task list : empty
          ()
)

(:method (obtain_object_from_someone ?name ?name2 ?thing)
          ;; precondition : ?name2 (different from ?name) has ?thing
          ((actor ?name) (actor ?name2) (thing ?thing) (place ?name2_place)
           (side ?side1) (side ?side2) (have ?name2 ?thing ?side2)
           (different ?name ?name2) (at ?name2 ?name2_place))
          ;; task list
          (:ordered (reach_place_with_free_hands ?name ?side1 ?name2_place)
                       (give ?name2 ?side2 ?name ?side1 ?thing))
)

(:method (obtain_object_from_someone ?name ?name2 ?thing)
          ;; precondition : ?name2 (different from ?name) has ?thing
          ((actor ?name) (actor ?name2) (thing ?thing) (place ?name2_place)
           (side ?side1) (different ?name ?name2) (possess ?name2 ?thing)
           (at ?name2 ?name2_place))
          ;; task list : reach_place_with_free_hands ?name ?name2_place,
          ;; ?name2 drops ?thing and ?name takes it
          (:ordered (reach_place_with_free_hands ?name ?side1 ?name2_place)
                       (drop ?name2 ?thing) (take ?name ?thing))
)
;; end action OBTAIN_OBJECT_FROM_SOMEONE
```

To introduce a social rule in $P_{sequences}$ we have modified the third method of ”obtain_object_from_someone” as follows:

```
(:method (obtain_object_from_someone ?name ?name2 ?thing)
          ;; precondition : ?name2 (different from ?name) has ?thing,
          ;; ?name is a human and ?name2 is a robot
          ((actor ?name) (actor ?name2) (thing ?thing) (place ?name2_place)
           (side ?side1) (possess ?name2 ?thing) (different ?name ?name2)
           (at ?name2 ?name2_place)
           (human ?name) (robot ?name2) (penalty_drop_take ?pen))
          ;; task list : idem + penalty
          (:ordered (reach_place_with_free_hands ?name ?side1 ?name2_place)
                       (drop ?name2 ?thing) (take ?name ?thing)
```

```
                    (:immediate !!apply_penalty ?pen))
           ;; precondition : ?name2 (different from ?name) has ?thing
           ((actor ?name) (actor ?name2) (thing ?thing) (place ?name2_place)
            (side ?side1) (possess ?name2 ?thing) (different ?name ?name2)
            (at ?name2 ?name2_place))
           ;; task list : idem
           (:ordered (reach_place_with_free_hands ?name ?side1 ?name2_place)
                    (drop ?name2 ?thing) (take ?name ?thing))
)
```