Profiles of Behaviour for Logic-Based Agents

F. Sadri and F. Toni

Department of Computing, Imperial College London,
{fs,ft}@doc.ic.ac.uk

Abstract. In an earlier paper [6] we presented a declarative approach for agent control. In that work we described how control can be specified in terms of *cycle theories*, which define declaratively the possible alternative behaviours of agents, depending on their internal state and (their perception of) the external environment in which they are situated. This form of control has been adopted for logic-based *KGP agents* [8, 2]. In this paper we show how using this form of control specification we can specify different *profiles* of agents, how they would vary the *behaviour* of agents and what advantages they have with respect to factors in the application and in the environment, such as time-criticality.

1 Introduction

In an earlier paper [6] we described how to specify control of agents via cycle theories. The approach is based on representing and reasoning with preferences and allows flexible control of the operations of agents. This takes the control beyond a fixed one-size-fits-all approach and allows the operations of the agents to be chosen dynamically given the circumstances of the environment, the state of the agent and its preferences.
Cycle theories have been adopted as the means of control in the KGP agent model [8, 2] developed within the SOCS project [1]. KGP is a modular logic-based model developed to cater for the challenges of open global computing environments. It relies upon a collection of capabilities utilised within transitions controlled by cycle theories. All the components are defined within computational logic, some using abductive logic programming and others using logic programming with preferences. The capabilities are designed to provide functionalities such as planning, reactivity, temporal reasoning and goal decision, all of which have been envisaged useful, maybe even necessary, for coping and adapting in a dynamic open environment. The KGP model has been implemented within the PROSOCS platform [12].
The behaviour of KGP agents can be seen as the sequence of transitions or operations they perform, and this sequence is determined by the agents' cycle theories. Thus by varying the cycle theory one can vary the behaviour of the agent. We have explored a number of such variations resulting in different profiles of behaviour. In an earlier paper [6] we briefly mentioned three, the focussed, careful and impatient profiles. In this paper we detail the first two. We characterise them formally, show how to design cycle theories that achieve them and discuss their advantages depending on the features of the environment and application domains. Other profiles are described in [1].

[1] `http://lia.deis.unibo.it/Research/Projects/SOCS/`

The motivation for this work is threefold: 1) to explore the degree of heterogeneity that can be achieved by varying cycle theories; 2) to explore the advantages of different profiles of behaviour with respect to different parameters such as the dynamic nature of the environment and the time-critical nature of applications; 3) to explore how such analysis can provide guidelines for implementers who use the PROSOCS platform.

Environments and circumstances in which agents have to function can vary. Some environments can be fairly static and predictable, while others can be highly dynamic and unpredictable. Agents may or may not have strict deadlines for their activities, and agents' resources may be limited, thus constraining what they can do, or they may have few resource restrictions. What interests us in this paper is to explore what profiles of behaviour would be advantageous in what type of environment and under what circumstances. Moreover, we would like to explore how to define such profiles by varying the control strategies of agents defined via cycle theories.

The paper is organised as follows. In Section 2 we present two examples to motivate the careful and focussed profiles. In Sections 3 and 4 we describe the necessary background to our work. In Section 5 we describe the careful and the focussed profiles in detail, show the characteristics of cycle theories that provably achieve these profiles, and discuss the pros of the two behaviour profiles. In Section 6 we conclude.

2 Motivating Examples

In this section we motivate, in the context of concrete examples, the two profiles we will study and formally define later, in Section 5.

2.1 Careful profile

Intuitively an agent endowed with this profile frequently re-examines its commitments to ensure that he honours only those that are feasible and necessary and he is not encumbered by any infeasible or unnecessary commitments. The advantage of such a profile is evident in a dynamic, unpredictable environment.

Consider an agent c who has sent its registration form to a conference $conf05$ and thus believes that it has registered for the conference. But it now wishes not to be registered at the conference. It sets itself this goal, and plans for it by generating an action to cancel his registration at $conf05$. Suppose the agent knows that :

> *If it observes that the deadline for cancellation for a conference has reached and it expects to cancel its registration at the conference then it should contact its bank and tell them to stop its credit card payment to the conference.*

Suppose before it has a chance to execute the action of cancellation of its registration it receives a message from the conference secretary telling it that there was a problem with its initial attempt at registration (for example the registration form arrived corrupted) and so it is actually not registered.

An agent with the careful profile will immediately realise that there is no longer any need to cancel its registration and consequently will not contact its bank to tell them to stop the credit card payment. But, under the same circumstances, an agent with a different profile might execute the (unnecessary) acts of contacting the bank and canceling the payment.

2.2 Focused profile

An agent may attempt to plan for multiple goals at once or may plan for one goal at a time. If the agent has limited resources it may be better off trying one goal at a time because typically it may not have enough resources for achieving multiple goals, and attempting to do so would only lead to time-wasting failures. This is the motivation behind our focussed profile. An agent endowed with the focussed profile focuses on one goal at a time.
Suppose an agent has two goals, one to have a particular book and the other to have a particular CD. Suppose the book costs £10 and the CD £15, and the agent has £20 available to spend. This agent cannot achieve both its goals, because due to its financial constraints it cannot form a consistent plan that would achieve both goals. If the agent has the focussed profile it will achieve one of them but if it has any other profile it may not achieve either goal.
In the next two sections we give the background that is necessary in order to formally define the profiles and show their consequences in terms of the behaviour of agents.

3 The KGP Model of Agency

Here we briefly summarise the KGP model for agents. Formal details can be found in [8, 2]. This model relies upon

- an *internal (or mental) state*,
- a set of *reasoning capabilities*, in particular supporting planning, temporal reasoning, reactivity and goal decision,
- a *sensing capability*, allowing agents to observe their environment and actions by other agents,
- a set of *transition rules*, defining how the state of the agent changes, and defined in terms of the above capabilities,
- a set of *selection functions*, to provide appropriate inputs to the transitions,
- a *cycle theory*, for deciding which transitions should be applied when, and defined using the selection functions.

Internal state. This is a tuple $\langle KB, Goals, Plan, TCS\rangle$, where:

- KB is the knowledge base of the agent, and describes what the agent knows (or believes) of itself and of the environment. KB consists of various modules supporting the different reasoning capabilities of agents, and including KB_0, for holding the (dynamic) knowledge of the agent about the external environment in which it is situated.
- $Goals$ is the set of properties that the agent wants to achieve, each one with an associate time variable, possibly constrained by temporal constraints (belonging to TCS), defining when the goals are expected to hold.
- $Plan$ is a set of actions scheduled in order to satisfy goals. Each has an associated time variable, possibly constrained by temporal constraints in TCS, similarly to $Goals$, but defining when the action should be executed and imposing a partial order over actions in $Plan$. Each action is also equipped with the preconditions for its successful execution.

– TCS is a set of constraint atoms (referred to as *temporal constraints*) in some given underlying constraint language with respect to some structure equipped with a notion of *Constraint Satisfaction*. We assume that the constraint predicates include $<, \leq, >, \leq, =, \neq$. These constraints bind the time of goals in $Goals$ and actions in $Plan$. For example, they may specify a time window over which the time of an action can be instantiated, at execution time.

Goals and actions are uniquely identified by their associated time variable, which is implicitly existentially quantified within the overall state.
To aid revision and partial planning, $Goals$ and $Plan$ form a *tree* [2]. The tree is given implicitly by associating with each goal and action its parent. *Top-level* goals and actions are children of the root of the tree, represented by the (arbitrary) symbol $\perp$.

Reasoning capabilities. These include:

– Planning, generating a plan, if one exists in the overall state, for any given set of input goals. These plans are *partial* or *total*. A partial plan consists of (temporally constrained) sub-goals and actions. A *total* plan consists solely of (temporally constrained) actions.
– Reactivity, reacting to perceived changes in the environment by modifying $Goals$, $Plan$, and TCS.
– Goal Decision, revising the top-most level goals by adapting the agent's state to changes in its own preferences and in the environment.
– Temporal Reasoning, reasoning about the evolving environment, and making predictions about properties holding in the environment, based on the partial information the agent acquires.

Transitions. The state of an agent evolves by applying transition rules, which employ capabilities and the Constraint Satisfaction. The transitions are:

– *Goal Introduction (GI)*, changing the top-level goals, and using Goal Decision.
– *Plan Introduction (PI)*, changing $Goals$ and $Plan$, and using Planning.
– *Reactivity (RE)*, changing $Goals$ and $Plan$, and using the Reactivity capability.
– *Sensing Introduction (SI)*, changing $Plan$ by introducing new sensing actions for checking the preconditions of actions already in $Plan$, and using Sensing.
– *Passive Observation Introduction (POI)*, changing KB_0 by introducing unsolicited information coming from the environment, and using Sensing.
– *Active Observation Introduction (AOI)*, changing KB_0 by introducing the outcome of (actively sought) sensing actions, and using Sensing.
– *Action Execution (AE)*, executing actions, and thus changing KB_0.
– *State Revision (SR)*, revising $Goals$ and $Plan$, and using Temporal Reasoning and Constraint Satisfaction.

[2] In the detailed model we actually have two trees, the first containing *non-reactive* goals and actions, the second containing *reactive* goals and actions. All the top-level non-reactive goals are either assigned to the agent by its designer at birth, or they are determined by the Goal Decision capability, via the GI transition (see below). All the top-level reactive goals and actions are determined by the Reactivity capability, via the RE transition (see below). Here for simplicity we overlook the distinction amongst the two trees.

The effect of transitions is dependent on the concrete time of their application. We briefly describe SR, as it will play an important role in section 5. Informally speaking, SR revises a state by removing (i) all timed-out goals and actions, (ii) all executed actions, (iii) all goals that have become obsolete because they are already believed to have been achieved, (iv) siblings (in the tree) of goals and actions deleted in (i), and (v) all descendants (in the tree) of goals deleted in (i)-(iv). A goal or action is *timed-out* if and only if the temporal constraints TCS of the state of the agent at the time of application of SR constrain the time of the goal or action to be less than or equal to the time of application of SR. A goal is *achieved* in a state if and only if it holds according to the Temporal Reasoning capability.

Selection functions. Input to (some of the) transitions is given via selection functions, taking the current state S and time τ as input:

– *action selection function*, $c_{AS}(S, \tau)$, returning the set of actions in S to be executed by AE at time τ;
– *goal selection function*, $c_{GS}(S, \tau)$, returning the set of goals in S to be planned for by PI at time τ;
– *fluent selection function*, $c_{FS}(S, \tau)$, returning the set of properties in S to be sensed by AOI at time τ;
– *precondition selection function*, $c_{PS}(S, \tau)$, returning the set of preconditions of actions in S for which sensing actions are to be introduced by SI at time τ.

4 Cycle Theories

The behaviour of agents results from the application of transitions in sequences, repeatedly changing the state of the agent. These sequences are not fixed a priori, as in conventional agent architectures, but are determined dynamically by reasoning with declarative cycle theories, giving a form of flexible control. Cycle theories are given in the framework of Logic Programming with Priorities (LPP). For the purposes of this paper, we will assume that an *LPP-theory*, referred to as $\mathcal{T}$, consists of four parts:

(i) a low-level part P, consisting of a logic program; each rule in P is assigned a name, which is a term; e.g., one such rule, with name $n(X)$, could be
$n(X) : p(X) \leftarrow q(X, Y), r(Y)$
(ii) a high-level part H, specifying conditional, dynamic priorities amongst rules in P; e.g., one such priority rule, called $h(X)$, could be $h(X) : n(X) \succ m(X) \leftarrow c(X)$, to be read: if (some instance of) the condition $c(X)$ holds, then the rule in P with name (the corresponding instance of) $n(X)$ should be given higher priority than the rule in P with name (the corresponding instance of) $m(X)$.
(iii) an auxiliary part A, defining predicates occurring in the conditions of rules in P and H and not in the conclusions of any rule in P;
(iv) a notion of incompatibility, here given as a set of rules defining the predicate $incompatible$, e.g. $incompatible(p(X), p'(X))$, to be read: any instance of the literal $p(X)$ is incompatible with the corresponding instance of the literal $p'(X)$. We refer to the set of all incompatibility rules as I.

Any concrete LPP framework is equipped with a notion of entailment, that we denote by $\models_{pr}$. Intuitively, $\mathcal{T} \models_{pr} \alpha$ iff α is the "conclusion" of a sub-theory of $P \cup A$ which is "preferred" with respect to $H \cup A$ in $\mathcal{T}$ over any other sub-theory of $P \cup A$ that derives a "conclusion" incompatible with α (with respect to I). Here, we are assuming that the underlying logic programming language is equipped with a notion of "entailment" that allows to draw "conclusions". In [10, 9, 7, 5, 3], $\models_{pr}$ is defined via argumentation.

Formalisation of Cycle theories. Here and in the rest of the paper, we will use notation $T(S, X, S', \tau)$ to represent application of transition T at time τ in state S, given input X, resulting in state S', and notation $*T(S, X)$ to represent that transition T can be potentially chosen as the next transition in state S, with input X.
Formally, a cycle theory $\mathcal{T}_{cycle}$ consists of the following parts.

– An *initial* part $\mathcal{T}_{initial}$, that determines the possible transitions that the agent could perform when it starts to operate. Concretely, $\mathcal{T}_{initial}$ consists of rules of the form
$$*T(S_0, X) \leftarrow C(S_0, \tau, X), now(\tau)$$
which we refer to via the name $\mathcal{R}_{0|T}(S_0, X)$. These rules sanction that, if conditions C hold in the initial state S_0 at the initial time τ, then the initial transition could be T, applied to state S_0 and input X.

– A *basic* part $\mathcal{T}_{basic}$ that determines the possible transitions following given transitions, and consists of rules of the form
$$*T'(S', X') \leftarrow T(S, X, S', \tau), EC(S', \tau', X'), now(\tau')$$
which we refer to via the name $\mathcal{R}_{T|T'}(S', X')$. These rules sanction that, after transition T has been executed, starting at time τ in the state S and resulting in state S', and the conditions EC evaluated in S' at the current time τ' are satisfied, then transition T' could be the next transition to be applied in S', with input X'. EC are called *enabling conditions* as they determine when T' can be applied after T. They also determine input X' for T', via calls to selection functions.

– A *behaviour* part $\mathcal{T}_{behaviour}$ that contains rules describing dynamic priorities amongst rules in $\mathcal{T}_{basic}$ and $\mathcal{T}_{initial}$. Rules in $\mathcal{T}_{behaviour}$ are of the form
$$\mathcal{R}_{T|T'}(S, X') \succ \mathcal{R}_{T|T''}(S, X'') \leftarrow BC(S, X', X'', \tau), now(\tau)$$
with $T' \neq T''$, which we will refer to via the name $\mathcal{P}^{T}_{T' \succ T''}$. Recall that $\mathcal{R}_{T|T'}(\cdot)$ and $\mathcal{R}_{T|T''}(\cdot)$ are (names of) rules in $\mathcal{T}_{basic} \cup \mathcal{T}_{initial}$. Note that, with an abuse of notation, T could be 0 in the case that one such rule is used to specify a priority over the *first* transition to take place, in other words, when the priority is over rules in $\mathcal{T}_{initial}$. These rules in $\mathcal{T}_{behaviour}$ sanction that, at the current time τ, after transition T, if the conditions BC hold, then we prefer the next transition to be T' over T''. The conditions BC are called *behaviour conditions* and give the behavioural profile of the agent.

– An *auxiliary part* including definitions for any predicates occurring in the enabling and behaviour conditions.

– An *incompatibility part*, in effect expressing that only one (instance of a) transition can be chosen at any one time.

Hence, $\mathcal{T}_{cycle}$ is an LPP-theory where: (i) $P = \mathcal{T}_{initial} \cup \mathcal{T}_{basic}$, and (ii) $H = \mathcal{T}_{behaviour}$.

Operational Trace. The cycle theory $\mathcal{T}_{cycle}$ of an agent is responsible for its behaviour, in that it induces an *operational trace* of the agent, namely a (typically infinite) sequence of transitions

$$T_1(S_0, X_1, S_1, \tau_1), \ldots, T_i(S_{i-1}, X_i, S_i, \tau_i), T_{i+1}(S_i, X_{i+1}, S_{i+1}, \tau_{i+1}), \ldots$$

such that

- S_0 is the given initial state;
- for each $i \geq 1$, τ_i is given by the clock of the system ($\tau_i < \tau_{i+i}$);
- $(\mathcal{T}_{cycle} - \mathcal{T}_{basic}) \wedge now(\tau_1) \models_{pr} *T_1(S_0, X_1)$;
- for each $i \geq 1$

  $(\mathcal{T}_{cycle} - \mathcal{T}_{initial}) \wedge T_i(S_{i-1}, X_i, S_i, \tau_i) \wedge now(\tau_{i+1}) \models_{pr} *T_{i+1}(S_i, X_{i+1})$

namely each (non-final) transition in a sequence is followed by the most preferred transition, as specified by $\mathcal{T}_{cycle}$. If, at some stage, the most preferred transition determined by $\models_{pr}$ is not unique, we choose one arbitrarily.

Normal cycle theory. In defining profiles in section 5 we take the *normal cycle theory* as a starting point. This specifies a pattern of operation where the agent prefers to follow a sequence of transitions that allows it to achieve its goals in a way that matches an expected "normal" behaviour. Basically, the "normal" agent first introduces goals (if it has none to start with) via GI, then reacts to them, via RE, and then repeats the process of planning for them, via PI, executing (part of) the chosen plans, via AE, revising its state, via SR, until all goals are dealt with (successfully or revised away). At this point the agent returns to introducing new goals via GI and repeating the above process. Whenever in this process the agent is interrupted via a passive observation, via POI, it chooses to introduce new goals via GI, to take into account any changes in the environment. Whenever it has actions which are "unreliable", in the sense that their preconditions definitely need to be checked, the agent senses them (via SI) before executing the action. Whenever it has actions which are "unreliable", in the sense that their effects definitely need to be checked, the agent actively introduces actions that aim at sensing these effects, via AOI, after having executed the original actions. The full definition of the normal cycle theory is given in the appendix.

5 Behaviour Profiles

In this section we explore how cycle theories can be used to specify different profiles of behaviour. We concentrate on two profiles, the *careful* and the *focussed*.

In the careful profile the behaviour of the agent is such that it would re-examine its commitments in terms of its goals and plans frequently to discard those that are no longer needed or have become infeasible. Intuitively, this profile would be suitable for a changing environment that intervenes in the agent's operations, and the frequent "self-examination" of the agent can help it avoid being occupied with unnecessary activity or activity which is bound to fail. It also ensures that the agent's operations are not hindered by superfluous items in the state and that reactive rules will not be triggered unnecessarily by goals/actions that are timed-out and not achieved/executed.

With the focussed profile the agent concentrates on one (top-level) goal at a time and only moves to other goals when that goal is achieved or is timed out. Intuitively this

profile is useful when the agent has goals that have become mutually unachievable. By being focussed the agent increases its chances of achieving at least some of them.
Below we proceed to define each of the two profiles by giving a formal definition in terms of trace characteristics, followed by specification of cycle theories that will induce such traces. We then proceed to prove the advantages of the profile depending on particular characteristics of the application.

5.1 Careful Profile

Definition 1 (Careful profile: trace-based characterisation). *A careful agent is an agent that will never generate an operational trace with two consecutive transitions that are different from SR.*
In fact, this condition is stronger than strictly necessary: As long as there are no redundant or infeasible goals or actions no revision would be required. However, from a pragmatic point of view, Definition 1 nevertheless provides us with an appropriate characterisation of careful agents. This is so, because *checking* whether or not a state includes redundant or infeasible goals or actions to be revised is just as costly as performing a state revision in the first place.
Our next goal is to define a class of cycle theories that are guaranteed to induce an operational trace where every other transition is an SR. As we shall see this is not as straightforward a goal as it may seem. To illustrate the difficulties and to motivate our choices (which are eventually going to overcome these difficulties), we start by attempting to define a careful cycle theory as an extension of the normal cycle theory.

The normal-careful cycle theory. There are several ways of combining cycle theories (in this case the normal cycle theory with the core rules necessary for characterising the careful profile). One option would be to take the union of the two cycle theories (which are sets of basic and behaviour rules) and then, where necessary, to introduce additional behaviour rules that determine the agent's behaviour in case of conflict between the rules stemming from the different parts. Another way, which gives the profile designer less freedom but which results in much simpler cycle theories, would be to work at the level of basic rules as far as possible and to use suitable enabling conditions to control the agent's behaviour. This is the approach we are going to follow here.
To design a careful agent, we need to ensure that basic rules expressing that *SR* should follow any other transition T get priority over any conflicting rules. Instead of using behaviour rules to this effect, we are simply going to delete such conflicting rules in the first place. Hence, we end up with the following approach:

- **Step 1:** Take the normal cycle theory as a starting point.
- **Step 2:** Remove any basic rules (in $\mathcal{T}_{basic}$) that speak about two consecutive transitions both of which are different from *SR*.
- **Step 3:**Add the following basic rule (to $\mathcal{T}_{basic}$) for each T different from *SR:*

$$\mathcal{R}_{T|SR}(S', \{\}) : *SR(S', \{\}) \leftarrow T(S, X, S', \tau)$$

Note that there cannot be any enabling conditions in this kind of new rule: *SR* needs to be enabled under *any* circumstances. Note also that Step 3 might re-introduce rules which already belong to $\mathcal{T}_{basic}$. This causes no theoretical or practical problem. We thus end up with the following *normal-careful cycle theory:*

- $\mathcal{T}_{initial}$ is as for the normal cycle theory.
- $\mathcal{T}_{basic}$ consists of the above rules of the form $\mathcal{R}_{T|SR}$ and of the following rules:

  $\mathcal{R}_{SR|PI}(S', Gs) : *PI(S', Gs) \leftarrow SR(S, \{\}, S', \tau'), Gs = c_{GS}(S', \tau), Gs \neq \{\}, now(\tau)$
  $\mathcal{R}_{SR|GI}(S', \{\}) : *GI(S', \{\}) \leftarrow SR(S, \{\}, S', \tau'), Gs = c_{GS}(S', \tau), Gs = \{\}, now(\tau)$

  $\mathcal{T}_{basic}$ does not contain any other rules, because all the remaining basic rules in the normal cycle theory speak about transitions that should follow transitions other than *SR* and these are fixed for the careful profile.
- $\mathcal{T}_{behaviour}$ is empty. Indeed, it turns out that also all of the rules in $\mathcal{T}_{behaviour}$ in the normal cycle theory are *redundant*, because they speak about what to do after a transition other than *SR*.

In summary, the normal-careful cycle theory will force an agent to alternate between *SR* and *PI* or *GI* (depending on whether there are currently goals to plan for or not). Such an agent would be careful, but not very useful. Below we improve the cycle theory to overcome this inadequacy.

The core-careful cycle theory. We improve the normal-careful cycle theory by adding that every transition except *SR*, itself, should be enabled after *SR*. Thus, $\mathcal{T}_{basic}$ in the *core-careful cycle theory* contains, in addition to the basic rules in the normal-careful cycle theory, the following rules:

$\mathcal{R}_{SR|RE}(S', \{\}) : *RE(S', \{\}) \leftarrow SR(S, \{\}, S', \tau)$
$\mathcal{R}_{SR|AE}(S', As) : *AE(S', As) \leftarrow SR(S, \{\}, S', \tau'), As = c_{AS}(S', \tau), As \neq \{\}, now(\tau)$
$\mathcal{R}_{SR|SI}(S', Ps) : *SI(S', Ps) \leftarrow SR(S, \{\}, S', \tau'), Ps = c_{PS}(S', \tau), Ps \neq \{\}, now(\tau)$
$\mathcal{R}_{SR|AOI}(S', Fs) : *AOI(S', Fs) \leftarrow SR(S, \{\}, S', \tau'), Fs = c_{FS}(S', \tau), Fs \neq \{\}, now(\tau)$
$\mathcal{R}_{SR|POI}(S', \{\}) : *POI(S', \{\}) \leftarrow SR(S, \{\}, S', \tau)$

The following proposition states the *correspondence* between the *core-careful cycle theory* and the (trace-based characterisation of the) careful profile given in Definition 1:

Proposition 1 (Careful profile). *The core-careful cycle theory induces the careful profile of behaviour: Any agent using this cycle theory will never generate an operational trace with two consecutive transitions that are different from SR.*

Proof. This follows immediately from the fact that the basic part of the cycle theory forces an SR after every other type of transition, and there is exactly one basic rule to determine the follow-up of any transition different from *SR*.

Other careful cycle theories The two careful cycle theories we have considered so far are just two examples; there is a range of cycle theories that conform to the careful behaviour profile. Our second example, the core-careful cycle theory is the most general cycle theory conforming to the careful profile.
For concrete applications, we may wish to combine the features of careful behaviour with other more specific features. We can construct a careful cycle theory of our choice by taking the core-careful cycle theory as a starting point and then imposing additional behaviour constraints using the following means:

- strengthening the enabling conditions in basic rules that determine the follow-up transition for an SR;
- deleting basic rules that determine the follow-up transition for an SR;

– adding any kind of behaviour rules;
– deleting rules that have become redundant due to other changes.

Note, however, that we *cannot* add any enabling conditions to the basic rules that state that *SR* has to follow any other transition. Otherwise, the resulting cycle theory cannot be guaranteed to conform to the careful profile of behaviour anymore. We also cannot delete such a rule, unless it has already become redundant due to other changes in the cycle theory. On the other hand, we do have complete freedom with respect to the behaviour rules we might wish to add, because the basic rules never admit any conflict as to what transition to choose after a transition different from *SR* in the first place.
Clearly, any such careful cycle theory will also induce the careful profile of behaviour in the sense of Proposition 1.

A property of the careful profile. Informally, under certain circumstances:

– Careful agents will never generate a reaction via the reactivity transition to timed-out unachieved goals or timed-out unexecuted actions.
– Careful agents will never generate a reaction via the reactivity transition to actions that may not be timed out yet but which are unexecuted and are no longer necessary.

More formally:

Theorem 1. *The following will never contribute to the generation of a reaction (i.e. an action in $Plan$ or goal in $Goals$) via the RE transition:*

1. *a timed-out unexecuted action,*
2. *a timed-out unachieved goal,*
3. *an unexecuted action whose execution is no longer needed, i.e.*
   (a) *with an ancestor which has already been achieved, or*
   (b) *with a sibling that has been timed-out, or*
   (c) *with an ancestor which has been timed-out,*

provided that no action and no goal is timed out between an SR transition and its immediate successor if that is an RE transition.

Proof. Let the assumption hold that no action and no goal is timed out between an SR transition and its immediate successor if that is an RE transition. Suppose a careful agent applies RE in a state $S = \langle KB, Goals, Plan, TCS\rangle$. Then by Definition 1, because SR must have been applied in the state immediately prior to S, no action or goal of the type specified in 1–3, above exists in state S. Therefore no such action or goal could possibly contribute to the generation of any reaction by RE.

5.2 Focussed Profile

In the *focussed* profile of behaviour an agent does not plan for more than one top-level goal at a time. More specifically, a focussed agent remains committed to a goal amongst its top-level goals until

– that goal has been successfully achieved, or
– that goal has become infeasible, or
– that goal is not preferred by the Goal Decision capability anymore, when invoked by the *GI* transition, or

– that goal has an empty plan in the state.[3]

The advantages of the focussed profile come into effect in highly time-critical domains as well as domains where an agent has several goals with mutually incompatible plans. In such situations, a focussed agent can be expected to achieve, at least, some goals, whereby an unfocussed agent may fail completely. This applies, in particular, to agents that have a preference for total planning. By concentrating planning on a single goal at a time, a focussed agent is likely to be faster and it will also avoid wasting computing resources over incompatible plans for other goals.

Formally, the focussed profile has the following characteristic: A focussed agent, under no circumstances, will generate an operational trace that includes a state with two distinct top-level goals with children, neither of which is either achieved or infeasible. Here, a goal G is called *feasible* iff neither itself nor any of its descendents is timed-out. Note that this notion of infeasibility need not persist. A goal G may, at some point, be infeasible, because an action in its current plan is timed-out, but G may again become feasible later on, after the agent has revised its state and computed a new plan. Therefore, the only way to ensure that switching to a new top-level goal for planning is admissible (under the focussed profile) is to first check that infeasible goals will *stay* infeasible. This requires an SR. Hence, we can give the following alternative definition of the focussed profile, which is simpler than our earlier definition.

Definition 2 (Focussed profile: trace-based characterisation). *A focussed agent is an agent that, under no circumstances, will generate an operational trace that includes a state with two top-level goals with children.*

This definition is stronger (more restrictive) than our first definition, but as argued earlier, it is operationally equivalent to that definition, because an agent can only be sure that switching goals will not violate the focussed profile after having executed an SR (or after having performed an analogous check).

Possible extensions. Note that, according to our definition, focussed agents do not deal with more than one top-level goal at a time, but may switch between top-level goals in some situations, as exemplified by the following example.

Example 1. Consider the following (portion of a) trace:

$$\ldots, SR(S, \{\}, S', \tau), PI(S', Gs, S'', \tau'), \ldots$$

with the top-level goals of S, S', S'' given by $\{G_1, G_2\}$. Assume that G_1 already has got a plan in S, i.e. the set of items in $Goals(S) \cup Plan(S)$ with ancestor G_1 is not empty. Assume also that G_2 has no plan in S, i.e. the set of items in $Goals(S) \cup Plan(S)$ with ancestor G_2 is empty. Suppose that all items in the plan for G_1 in S are timed-out at τ, and thus S' is such that $Goals(S')$ is the set of all top-level goals in S' and $Plan(S') = \{\}$. Suppose also that neither G_1 nor G_2 are timed-out or achieved at τ', but PI is introducing a plan for G_2, so that the set of items in $Goals(S'') \cup Plan(S'')$ with ancestor G_2 is not empty. The agent with this trace is focussed according to definition 2. However, it does switch from dealing with goal G_1 to dealing with goal G_2, despite goal G_1 being still unachieved and feasible.

[3] The need for this last item will become clear in Example 1.

Definition 2 of focussed agent may be modified to prevent goal switching, by comparing successive agent states in traces and force that once an agent has been planning/executing for one top-most level goal in one state, it must stick to that goal in successive states, until the goal has been achieved or has become unachievable. This would amount to getting rid of the last item in the informal description of focussed agent at the beginning of Section 5.2 (and adding some other suitable conditions instead). This stronger definition of focussed agent would however force extending the notion of cycle theory and operational trace, either by looking at histories of transitions rather than individual transitions when deciding on the next transition, or by introducing additional information into cycle theories, such as variables holding the current top-level goal being dealt with. We therefore leave the stronger definition to future work.
Note also that our notion of focussed agent only refers to *top-level* goals, and not to sub-goals or actions. The notion of focussed agent could be extended so as to define agents that are focussed all the way, from top-level goals down.

Focussed Cycle Theories. To achieve the abstract specification, we need a cycle theory that ensures that before any PI an SR has been performed. This is to ensure that we can proceed with planning for a top-level goal even if some of its current children have become infeasible. However, rather than implementing this behaviour directly, we are going to ensure that PI is only enabled with respect to a set of goals that a focussed agent may plan for given its current state according to the Definition 2. (This, in effect, encourages an SR transition when a PI transition is not enabled.)

Definition 3 (Focussed cycle theories). *A cycle theory is called focussed iff the initial rule $\mathcal{R}_{0|PI}(S,Gs)$ (in $\mathcal{T}_{initial}$) and the basic rule (in $\mathcal{T}_{basic}$) $\mathcal{R}_{T|PI}(S,Gs)$ for any transition T include the enabling condition $focussed(Gs',S,Gs)$, where:*

- *given that Gs is the set of goals to which PI will be applied and $Gs' \supseteq Gs$ is the set of goals returned by the goal selection function, then*
- *the predicate $focussed(Gs',S,Gs)$ holds iff all the goals in Gs are descendants of the same top-level goal (possibly including that top-level goal itself) and no other top-level goal has got any children.*

The focussed variant of the normal cycle theory would have in $\mathcal{T}_{initial}$ the rule

$$\mathcal{R}_{0|PI}(S_0,Gs) : *PI(S_0,Gs) \leftarrow Gs' = c_{GS}(S_0,\tau),$$
$$focussed(Gs',S_0,Gs), Gs \neq \{\}, now(\tau)$$

instead of the original rule

$$\mathcal{R}_{0|PI}(S_0,Gs) : *PI(S_0,Gs) \leftarrow Gs = c_{GS}(S_0,\tau), Gs \neq \{\}, now(\tau)$$

Similarly, the focussed variant of the normal cycle theory would have in $\mathcal{T}_{basic}$ the rule

$$\mathcal{R}_{AE|PI}(S',Gs) : *PI(S',Gs) \leftarrow AE(S,As,S',\tau'), Gs' = c_{GS}(S',\tau),$$
$$focussed(Gs',S,Gs), Gs \neq \{\}, now(\tau)$$

instead of the original rule

$$\mathcal{R}_{AE|PI}(S',Gs) : *PI(S',Gs) \leftarrow AE(S,As,S',\tau'), Gs = c_{GS}(S',\tau), Gs \neq \{\}, now(\tau)$$

The *correspondence* between the trace-based characterisation of the *focussed profile* and the class of focussed cycle theories may be stated as follows:

Proposition 2 (Focussed profile). *Any cycle theory that is focussed according to Definition 3 induces the focussed profile of behaviour according to Definition 2.*

Proof. The enabling condition $focussed(Gs', S, Gs)$ restricts the set of goals for which the agent may plan to precisely the set of goals that are available for planning according to the trace-based characterisation of the focussed profile. The claimed correspondence then follows immediately from the fact that PI is the only transition that can add non-top-level goals to a state.

A property of the focussed profile. Let a focussed agent be one equipped with a focussed cycle theory, and a normal agent be one equipped with the normal cycle theory. Then if the two agents have a set of goals for which they have no compatible plans then the focussed agent may be able to achieve at least some of its goals while the normal agent may not be able to achieve any of the goals. The theorem below shows under what conditions the focussed agent is guaranteed to achieve more of its goals compared to the normal agent. Note that conditions 1-6 simply set the scene for the theorem whereas conditions 7-9 restrict features of the environment and the application.

Theorem 2. *Let f be a focussed agent and n be a normal agent. Let f and n be in a state $S = \langle KB, Goals, Plan, TCS \rangle$ at time τ such that all the conditions below hold:*
1. *$Plan$ is empty.*
2. *$Goals$ consists of top-level goals $G_1, \ldots, G_n$, $n > 1$* [4].
3. *The goal selection function, in state S, at all times τ', $\tau' \geq \tau$, selects the same set of k goals for some $1 < k \leq n$, until one or more such goals are achieved. Assume these goals are $\{G_1, \ldots, G_k\}$, without loss of generality.*
4. *The agents' PI transition produces a total plan for all its input goals.*
5. *At all times after τ, given input goals $\{G_1, \ldots, G_k\}$, the agents' PI transition returns no plan, because none exists in the overall state.*
6. *At all times after τ, given input goals $\{G_i\}$, $i = 1, \ldots k$, the agents' PI transition returns a (total) plan.*

Then, f will achieve at least one of the goals amongst $G_1, \ldots, G_n$, while n will achieve none of them, provided that:

7. *The agents' RE transition generates no goals or actions.*
8. *No POI, AOI transitions are performed, and no GI transition is performed after the establishment of top-level goals $G_1, \ldots, G_n$.*
9. *Goals and actions are non-time critical, i.e. no goal or action is timed out.*

Proof. (Sketch) Consider the case of the normal agent n: by conditions 3,5,7,8 the state of n remains the same (although time progresses). In this state, by conditions 3 and 5, n can never make any progress towards achieving any of its top-level goals.
Now consider the case of the focussed agent f: At some time τ_1, $\tau_1 \geq \tau$, f performs PI. By conditions 3 and 6 and the definition of the focussed profile a goal G_i, $i = 1, \ldots k$, is selected and PI succeeds in producing a complete plan for G_i, and updates its state by adding all the produced actions As to its $Plan$ and updating TCS appropriately.

[4] Conditions 1. and 2. can arise, for example, if f and n have just executed GI starting from the same initial state.

These new actions will then all be executed. They will not be timed-out by condition 9. So they may be removed from the state of the agent by SR only if their associated goal is achieved. Any new goals and actions that may be introduced by later applications of PI will not interfere with the execution of the actions in As. Therefore, finally, after all the actions are executed, it will be possible to prove by the Temporal Reasoning that goal G_i which was selected at time τ_1 is achieved.

Note that conditions 7–9 are sufficient but not necessary conditions. For example condition 8 can be replaced with one that requires only that any observation recorded as a result of a POI is "independent" of the goals $G_1, \dots, G_n$, and allows GI transitions but imposes restrictions on their frequency. It is possible to construct examples where some, possibly many, of conditions 7–9 do not hold, but still the focussed agent performs better than the normal one in goal achievement terms.

6 Conclusion

In this paper, building on our earlier work [6], we have further explored the use of cycle theories for declarative control of agents. We showed how in the case of KGP agents we can define concrete and useful agent profiles or personalities by varying the rules in cycle theories. We showed two such profiles in detail, careful and focussed, and exemplified and formally proved their advantages. The cycle theories for these two profiles are no more complicated than the normal cycle theory, and possibly, in the case of the careful profile, the cycle theory is simpler.
The careful profile is best suited to a dynamic unpredictable environment, but one in which the agent does not have strict deadlines. The focussed profile is best suited to resource-bounded agents. The theoretical analysis of the profiles not only allows exploration of heterogeneity of agents, but it can also provide guidelines to designers of agents and implementers, for example those using the PROSOCS platform. There is scope for exploring a number of other profiles, some of which have been introduced in [1]. Exploring other profiles, parameterising their advantages and disadvantages according to factors in the environment and application domains and exploring how profiles can be usefully combined are subjects of current and future research. Currently we see no problem in combining the careful and focussed profiles.
Our work on profiles shares some of the objectives of the work on commitment strategies based on the BDI model [11]. Three commitment strategies have been defined, *blind*, *single minded*, and *open minded*. They are defined by expressing relationships between current and future intentions. A blindly committed agent, for example, maintains its intentions as long as it believes that it has achieved them, while a single minded agent maintains its intentions until it believes they are achievable. Our work on profiles and their consequences goes some way beyond these commitment strategies.
Our approach shares the aims of 3APL [4] to make it possible to program the agent cycle and make the selection mechanisms explicit. But it goes beyond 3APL by abandoning the concept of fixed cycles and replacing it with dynamic programmable cycle theories.

Acknowledgments

This work was partially funded by the IST programme of the EC, FET under the IST-2001-32530 SOCS project, within the GC proactive initiative. We are also grateful to A.C. Kakas and U. Endrich for early discussions regarding this work.

References

1. F. Athienitou, A. Bracciali, U. Endriss, A.C. Kakas, W. Lu, P. Mancarella, F. Sadri, K. Stathis, and F. Toni. Profile related properties. Technical report, SOCS deliverable, 2005.
2. A. Bracciali, N. Demetriou, U. Endriss, A.C. Kakas, W. Lu, P. Mancarella, F. Sadri, K. Stathis, G. Terreni, and F. Toni. The KGP model of agency for global computing: Computational model and prototype implementation. In *Global Computing 2004 Workshop*, page 342. Springer Verlag LNCS 3267, 2005.
3. Y. Dimopoulos and A. C. Kakas. Logic programming without negation as failure. In *Proc. ILPS*, pages 369–384, 1995.
4. K. V. Hindriks, F. S. de Boer, W. van der Hoek, and J. Ch. Meyer. Agent programming in 3APL. *Autonomous Agents and Multi-Agent Systems*, 2(4):357–401, 1999.
5. A. C. Kakas, P. Mancarella, and P. M. Dung. The acceptability semantics for logic programs. pages 504–519, 1994.
6. A. C. Kakas, P. Mancarella, F. Sadri, K. Stathis, and F. Toni. Declarative agent control. In *Proc. CLIMA V*, 2004.
7. A. C. Kakas and P. Moraitis. Argumentation based decision making for autonomous agents. pages 883–890, Melbourne, Victoria, July 14–18 2003.
8. A.C. Kakas, P. Mancarella, F. Sadri, K. Stathis, and F. Toni. The KGP model of agency. In *Proc. ECAI-2004*, 2004.
9. R.A. Kowalski and F. Toni. Abstract argumentation. *Artificial Intelligence and Law Journal, Special Issue on Logical Models of Argumentation*, 4:275–296, 1996.
10. H. Prakken and G. Sartor. A system for defeasible argumentation, with defeasible priorities. In *International Conference on Formal and Applied Practical Reasoning, Springer Lecture Notes in AI 1085*, pages 510–524. 1996.
11. A. S. Rao and M. P. Georgeff. Modeling rational agents within a BDI-architecture. In *Readings in Agents*, pages 317–328. 1997.
12. K. Stathis, A.C. Kakas, W. Lu, N. Demetriou, U. Endriss, and A. Bracciali. PROSOCS: A platform for programming software agents in computational logic. In *Proc. AT2AI*, 2004.

A Normal cycle theory in full

– $\mathcal{T}_{initial}$:
  $\mathcal{R}_{0|GI}(S_0, \{\}) : *GI(S_0, \{\}) \leftarrow empty\_goals(S_0)$
  $\mathcal{R}_{0|PI}(S_0, Gs) : *PI(S_0, Gs) \leftarrow Gs = c_{GS}(S_0, \tau), Gs \neq \{\}, now(\tau)$
  $\mathcal{R}_{0|POI}(S_0, \{\}) : *POI(S_0, \{\}) \leftarrow poi\_pending(\tau), now(\tau)$
– $\mathcal{T}_{basic}$:
  rules for deciding what might follow AE:
  $\mathcal{R}_{AE|PI}(S', Gs) : *PI(S', Gs) \leftarrow AE(S, As, S', \tau'), Gs = c_{GS}(S', \tau), Gs \neq \{\}, now(\tau)$
  $\mathcal{R}_{AE|AE}(S', As') : *AE(S', As') \leftarrow AE(S, As, S', \tau'), As' = c_{AS}(S', \tau), As' \neq \{\}, now(\tau)$
  $\mathcal{R}_{AE|AOI}(S', Fs) : *AOI(S', Fs) \leftarrow AE(S, As, S', \tau'), Fs = c_{FS}(S', \tau), Fs \neq \{\}, now(\tau)$
  $\mathcal{R}_{AE|SR}(S', \{\}) : *SR(S', \{\}) \leftarrow AE(S, As, S', \tau')$
  $\mathcal{R}_{AE|GI}(S', \{\}) : *GI(S', \{\}) \leftarrow AE(S, As, S', \tau')$
  rules for deciding what might follow SR:

$\mathcal{R}_{SR|PI}(S', Gs) : *PI(S', Gs) \leftarrow SR(S, \{\}, S', \tau'), Gs = c_{GS}(S', \tau), Gs \neq \{\}, now(\tau)$
$\mathcal{R}_{SR|GI}(S', \{\}) : *GI(S', \{\}) \leftarrow SR(S, \{\}, S'\tau'), Gs = c_{GS}(S', \tau), Gs = \{\}, now(\tau)$
rules for deciding what might follow PI:
$\mathcal{R}_{PI|AE}(S', As) : *AE(S', As) \leftarrow PI(S, Gs, S', \tau'), As = c_{AS}(S', \tau), As \neq \{\}, now(\tau)$
$\mathcal{R}_{PI|SI}(S', Ps) : *SI(S', Ps) \leftarrow PI(S, Gs, S', \tau'), Ps = c_{PS}(S', \tau), Ps \neq \{\}, now(\tau)$
rules for deciding what might follow GI:
$\mathcal{R}_{GI|RE}(S', \{\}) : *RE(S', \{\}) \leftarrow GI(S, \{\}, S', \tau)$
$\mathcal{R}_{GI|PI}(S', Gs) : *PI(S', Gs) \leftarrow GI(S, \{\}, S', \tau'), Gs = c_{GS}(S', \tau), Gs \neq \{\}, now(\tau)$
rules for deciding what might follow RE:
$\mathcal{R}_{RE|PI}(S', Gs) : *PI(S', Gs) \leftarrow RE(S, \{\}, S', \tau'), Gs = c_{GS}(S', \tau), Gs \neq \{\}, now(\tau)$
$\mathcal{R}_{RE|SI}(S', Ps) : *SI(S', Ps) \leftarrow RE(S, \{\}, S', \tau'), Ps = c_{PS}(S', \tau), Ps \neq \{\}, now(\tau)$
$\mathcal{R}_{RE|AE}(S', As) : *AE(S', As) \leftarrow RE(S, \{\}, S', \tau'), As = c_{AS}(S', \tau), As \neq \{\}, not(\tau)$
$\mathcal{R}_{RE|SR}(S', \{\}) : *SR(S', \{\}) \leftarrow RE(S, \{\}, S', \tau')$
rules for deciding what might follow SI:
$\mathcal{R}_{SI|AE}(S', As) : *AE(S', As) \leftarrow SI(S, Ps, S', \tau'), As = c_{AS}(S', \tau), As \neq \{\}, now(\tau)$
rules for deciding what might follow AOI:
$\mathcal{R}_{AOI|AE}(S', As) : *AE(S', As) \leftarrow AOI(S, Fs, S', \tau'), As = c_{AS}(S', \tau), As \neq \{\}, now(\tau)$
$\mathcal{R}_{AOI|SR}(S', \{\}) : *SR(S', \{\}) \leftarrow AOI(S, Fs, S', \tau')$
$\mathcal{R}_{AOI|SI}(S', Ps) : *SI(S', Ps) \leftarrow AOI(S, Fs, S', \tau') Ps = c_{PS}(S', \tau), Ps \neq \{\}, now(\tau)$
rules for deciding when POI should take place:
$\mathcal{R}_{T|POI}(S', \{\}) : *POI(S', \{\}) \leftarrow T(S, X, S', \tau'), poi\_pending(\tau), now(\tau)$
for all transitions T;
rules for deciding what might follow POI:
$\mathcal{R}_{POI|GI}(S', \{\}) : *GI(S', \{\}) \leftarrow POI(S, \{\}, S', \tau)$
$\mathcal{R}_{POI|RE}(S', \{\}) : *RE(S', \{\}) \leftarrow POI(S, \{\}, S', \tau)$
$\mathcal{R}_{POI|SR}(S', \{\}) : *SR(S', \{\}) \leftarrow POI(S, \{\}, S', \tau)$

– $\mathcal{T}_{behaviour}$:
GI is given higher priority if there are no goals in $Goals$ **and actions in** $Plan$:
$\mathcal{P}^{T}_{GI \succ T'} : \mathcal{R}_{T|GI}(S, \{\}) \succ \mathcal{R}_{T|T'}(S, X) \leftarrow empty\_goals(S), empty\_plan(S)$
for all T, T', with $T' \neq GI$ and T possibly 0;
GI is also given higher priority after a POI:
$\mathcal{P}^{POI}_{GI \succ T} : \mathcal{R}_{POI|GI}(S') \succ \mathcal{R}_{POI|T}(S, S')$ for all $T \neq GI$;
after GI, RE is given higher priority:
$\mathcal{P}^{GI}_{RE \succ T} : \mathcal{R}_{GI|RE}(S, \{\}) \succ \mathcal{R}_{GI|T}(S, X)$ for all $T \neq RE$;
after RE, PI is given higher priority:
$\mathcal{P}^{RE}_{PI \succ T} : \mathcal{R}_{RE|PI}(S, Gs) \succ \mathcal{R}_{RE|T}(S, X)$ for all $T \neq PI$;

after PI, AE is given higher priority, unless there are actions in the actions selected for execution whose preconditions are "unreliable" and need checking, in which case SI will be given higher priority:
$\mathcal{P}^{PI}_{AE \succ T} : \mathcal{R}_{PI|AE}(S, As) \succ \mathcal{R}_{PI|T}(S, X) \leftarrow not\, unreliable\_pre(As)$
for all $T \neq AE$;
$\mathcal{P}^{PI}_{SI \succ T} : \mathcal{R}_{PI|SI}(S, Ps) \succ \mathcal{R}_{PI|T}(S, As) \leftarrow unreliable\_pre(As)$
for all $T \neq SI$;
after SI, AE is given higher priority:
$\mathcal{P}^{SI}_{AE \succ T} : \mathcal{R}_{SI|AE}(S, As) \succ \mathcal{R}_{SI|T}(S, X)$ for all $T \neq AE$;
after AE, AE should be given higher priority until there are no more actions to execute in $Plan$, in which case either AOI or SR should be given higher priority, depending on whether there are actions which are "unreliable", in the sense that their effects need checking, or not:
$\mathcal{P}^{AE}_{AE \succ T} : \mathcal{R}_{AE|AE}(S, As) \succ \mathcal{R}_{AE|T}(S, X)$ for all $T \neq AE$;
$\mathcal{P}^{AE}_{AOI \succ T} : \mathcal{R}_{AE|AOI}(S, Fs) \succ \mathcal{R}_{AE|T}(S, X) \leftarrow empty\_executable\_plan(S), unreliable\_post(S)$
for all $T \neq AOI$;
$\mathcal{P}^{AE}_{SR \succ T} : \mathcal{R}_{AE|SR}(S, \{\}) \succ \mathcal{R}_{AE|T}(S, X) \leftarrow empty\_executable\_plan(S), not\, unreliable\_post(S)$
for all $T \neq SR$;
after SR, PI should have higher priority:
$\mathcal{P}^{SR}_{PI \succ T} : \mathcal{R}_{SR|PI}(S, Gs) \succ \mathcal{R}_{SR|T}(S, \{\})$ for all $T \neq PI$;
after any transition, POI is preferred over all other transitions:
$\mathcal{P}^{T}_{PI \succ T'} : \mathcal{R}_{T||OI}(S) \succ \mathcal{R}_{T|T'}(S, X)$ for all T, T', with $T' \neq POI$ and T possibly 0;
in the initial state, PI is given higher priority:
$\mathcal{P}^{0}_{PI \succ T} : \mathcal{R}_{0|PI}(S, Gs) \succ \mathcal{R}_{0|T}(S, X)$ for all $T \neq PI$;

– The auxiliary part includes definitions for $empty\_goals$, $unreliable\_pre$, $unreliable\_post$, $empty\_executable\_plan$, $poi\_pending$ etc. Note that $poi\_pending(\tau)$ holds when there is an input from the environment pending.