Argument Schemes for Normative Practical Reasoning

Nir Oren

Department of Computing Science, University of Aberdeen, AB24 3UE, Scotland
n.oren@abdn.ac.uk

Abstract. This paper describes a framework for practical reasoning in the presence of norms. We describe a formal normative model constructed using Action-based Alternating Transition Systems. This model is able to represent goals; obligations and prohibitions and their violation; and permissions, which are used to derogate the former. Inspired by Atkinson's scheme for practical reasoning, we utilise argument schemes and critical questions to both show, and reason about how goals and obligations lead to preferences over the possible executions of the system. The model then allows us to determine if sufficient information has been provided in order to perform practical reasoning, identify the best courses of action, and explain *why* specific sequences of actions should be executed by agents within the system.

1 Introduction

The violation of a norm, as expressed through obligations, permissions and prohibitions, can result in sanctions being imposed on an agent. Since such sanctions are undesirable, the agent will typically attempt to comply with its norms while pursuing goals. However, it may be the case that the violation of a norm can yield greater rewards than the cost of sanctions to the agent (e.g. if the violation results in the achievement of an important goal). Norms therefore impose *soft constraints* upon an agent, and when performing practical reasoning, an agent must weigh up the penalties (and rewards) involved in violating (or adhering to) norms against the rewards provided by achieving its goals.

Now while practical reasoning frameworks taking norms into account have been previously proposed (e.g. [1]), explaining the decision processes taken by agents when acting in such a system, particularly to non-experts, is a difficult task. In this paper, we propose an argumentation based framework for practical reasoning in the presence of norms, with the longer term aim of investigating how argumentation can be used to contribute to the explanation of the agent's decision processes. While decision and game theory provide processes whereby a rational agent (i.e. one that attempts to maximise some utility, or reach a most preferred state) can identify an optimal sequence of actions, we argue that in the practical reasoning domain, such processes can be more easily understood through argument schemes. The instantiation of such schemes, and their associated critical questions, results in an argument framework which can be

evaluated to identify the appropriate action(s) to pursue. These arguments can then be presented to explain why the specific course of action was selected.

In this paper we propose a semantics for norms and goals that can be used to describe the possible executions of a system. The set of all these executions then forms the core of the practical reasoning process. Building on this formal system, we introduce a set of argument schemes together with the appropriate critical questions, which can in turn be used to identify the most preferred system execution path.

In the next section we describe our formal model in detail. Following this, Section 3 introduces the argument scheme and maps it to our formal model. An example of the approach is provided in Section 4, and we discuss related and future work in Section 5, before concluding the paper in Section 6.

2 The Model

In this section we describe our formal model, which is based on action-based alternating transition systems (AATSs) [2]. Such AATSs are intended to encode all possible evolutions of a system due to the actions of all agents within it, representing the various states through which the system can pass through by means of a branching time tree structure. Since this can be described as a Kripke system, we can reason about the possible trajectories of the system by means of a branching time logic. After introducing the basic concepts of AATSs, we detail how goals and norms, as well as more complex concepts such as violations and the derogation of obligations can be specified using the logic.

2.1 Semantics

Definition 1. ***(AATS, [2])*** *An Action-based alternating transition system (AATS) is a tuple of the form*

$$S = \langle Q, q_0, Ag, Ac_1, \ldots Ac_n, \rho, \tau, \Phi, \pi \rangle$$

Where

- Q *is a finite non-empty set of* states.
- $q_0 \in Q$ *is the* initial state.
- $Ag = \{1, \ldots, n\}$ *is a finite non-empty set of agents.*
- Ac_i*, with* $1 \leq i \leq n$*, is a finite and non-empty set of actions for each agent, where actions for different agents do not overlap.*
- $\rho : Ac_i \rightarrow 2^Q$ *is an action precondition function which identifies the set of states from which some action* $\alpha \in Ac_i$ *can be executed*
- $\tau : Q \times J_{Ag} \rightarrow Q$ *where* $J_{Ag} = \prod_{i \in Ag} Ac_i$*, is the system transition function identifying the state that results from executing a set of actions from within* J_{Ag} *in some state.*
- Φ *is a finite and non-empty set of atomic propositions*

– $\pi : Q \rightarrow 2^{\Phi}$ *is the interpretation function which identifies the set of propositions satisfied in each state.*

Following [2], we define a computation (also referred to as a path) to be an infinite sequence of states $\lambda = q_0, q_1, \ldots$, *where* $q_i \in \tau(q_{i-1}, \alpha)$ *for some* α *for which* $q_i \in \rho(\alpha)$. *We index a state within a path using array notation. Thus, the first element of path* λ *can be referenced via* $\lambda[0]$, *while a sub-path of the path starting at the second element and consisting of the remainder of the path is written* $\lambda[1, \infty]$.

An AATS encodes the possible states of the world that result from executing actions, and can be viewed as a Kripke structure with the transition function τ acting as the accessibility relation. We can therefore represent the AATS using CTL* operators [3], allowing us to refer to both single paths and groups of paths in the structure. We define the semantics of CTL* in two stages, first defining state formulae, following which we describe path formulae. The syntax of CTL* emerges directly from the semantics and is not detailed due to space constraints.

Definition 2. ***(State Formulae)*** *State formulae are evaluated with respect to an AATS* S *and a state* $q \in Q$*:*

$S, q \models \top$
$S, q \not\models \bot$
$S, q \models p$ *iff* $p \in \pi(q)$
$S, q \models \neg\psi$ *iff* $S, q \not\models \psi$
$S, q \models \psi \vee \phi$ *iff* $S, q \models \psi$ *or* $S, q \models \phi$
$S, q \models \mathcal{A}\psi$ *iff* $S, \lambda \models \psi$ *for all paths where* $\lambda[0] = q$
$S, q \models \mathcal{E}\psi$ *iff* $S, \lambda \models \psi$ *for some path where* $\lambda[0] = q$

Definition 3. ***(Path Formulae)*** *Path formulae are evaluated with respect to an AATS* S *and a path* λ.

$S, \lambda \Vdash \psi$ *iff* $S, \lambda[0] \models \psi$ *where* ψ *is a state formula.*
$S, \lambda \Vdash \neg\psi$ *iff* $S, \lambda \not\Vdash \psi$
$S, \lambda \Vdash \psi \vee \phi$ *iff* $S, \lambda \Vdash \psi$ *or* $S, \lambda \Vdash \phi$
$S, \lambda \Vdash \bigcirc\psi$ *iff* $S, \lambda[1, \infty] \Vdash \psi$
$S, \lambda \Vdash \Diamond\psi$ *iff* $\exists u \in \mathbb{N}$ *such that* $S, \lambda[u, \infty] \Vdash \psi$
$S, \lambda \Vdash \Box\psi$ *iff* $\forall u \in \mathbb{N}$ *it is the case that* t $S, \lambda[u, \infty] \Vdash \psi$
$S, \lambda \Vdash \phi \mathcal{U} \psi$ *iff* $\exists u \in \mathbb{N}$ *such that* $S, \lambda[u, \infty] \Vdash \psi$ *and*
$\forall v$ *s.t.* $0 \leq v < u$, $S, \lambda[v, \infty] \Vdash \phi$

Note that state formulae refer only to a single possible world, or state, within a path, even in the case when the state operator then refers to a path formula (c.f. the $\mathcal{A}$ and $\mathcal{E}$ operators). Path formulae always refer to entire paths which begin at some state (e.g. the next state in the case of the $\bigcirc$ operator).

These semantics capture the evolution of a system over time due to agent actions. However, they say nothing about why one path might be followed by agents rather than another in order to effect certain actions and therefore lead to certain states. To capture this notion we define a relation over paths, written

$\succeq^g$ to represent the preferences of some group of agents g with respect to one group of paths over another. This group of paths is specified by means of a path formula. Thus, for example, $\Diamond a \succeq^{\{\alpha\}} \Diamond \neg a$ captures the preference of agent α for those paths in which a is eventually true over those paths where it is eventually false. When dealing with a single agent, or referring to a group by a label, we write $\succeq^\alpha$ instead of $\succeq^{\{\alpha\}}$. Finally, we write $\lambda \succ^g \lambda'$ to represent the case when $\lambda \succeq^g \lambda'$ and $\lambda' \not\succeq^g \lambda$, and abbreviate the situation where both $\lambda \succeq^g \lambda'$ and $\lambda' \succeq^g \lambda$ hold as $\lambda \sim^g \lambda'$.

Now a question arises as to the origin and form of the preference relation, and we propose that the agent's goals, together with the norms found in the system constrain (but do not fully specify) it. For example, if an agent has a goal, then it should prefer those paths where the goal is achieved to those paths where it is not. However, this goal does not impose any preference ordering between those paths in which the goal is achieved (or indeed between those paths where it is not). We begin a more detailed exploration of the preference relationship by examining goals more closely.

2.2 Goals

Goals identify states of affairs in the world that an agent prefers (and should be able to bring about in part due to their action, but we do not formally impose this requirement). In other words, when undertaking practical reasoning, agents prefer those actions forming paths wherein their goals are achieved to those where they are not. We therefore represent goals through path formulae, identifying the state of affairs that must exist for a goal to be considered as *met* or *satisfied*.

We consider both achievement and maintenance goals [4]. The former identifies a state of affairs that must hold at some point in time, while the latter requires some state of affairs to be maintained until some deadline. Both of these goals can be easily represented in our logic, though in this paper we ignore conditional goals (i.e. goals of the form " If X is the case then a goal Y exists").

Definition 4. ***(Goals)*** *An formula γ decribes a path where an achievement goal is met if it is of the form $\neg d \mathcal{U} x$. It describes a maintenance goal path if it takes the form $(\neg d \wedge x) \mathcal{U} d$.*[1]

x represents the state of affairs that the goal aims to satisfy, while d represents the goal's deadline.

Since achieving a goal γ is preferred by some agent or group over not achieving the goal, we can identify a preference ordering over possible paths by the simple rule

$$\gamma \succ^g \neg\gamma$$

[1] The semantics of $\mathcal{U}$ require us to ensure that the deadline does not occur before it actually does.

2.3 Norms

Norms within a system represent obligations, prohibitions and permissions imposed on, or provided to, entities within a society or group. Obligations and prohibitions (respectively) identify the states of affairs that a *target* must ensure do (or do not) occur. If these states of affairs do not (or do) occur, then the norm is *violated*. Following [5–7], we treat permissions as exceptions to obligations and prohibitions: in the case of an obligation, if a state of affairs is ordinarily obliged, but a permission not to achieve the state exists, then even if the state of affairs is not achieved, no violation occurs.

Now we view norms primarily as *social* constructs. That is, an obligation (for example) specifies *who* should behave in some way (i.e. it has a set of target agents), and also identifies which agent — or set of agents — desires that this behaviour occur. The latter form the norm's *creditors* (c.f. the social commitments of Singh [8]).

Following this perspective, we view a norm as expressing a preference over a state of affairs *for its creditors* rather than its target. That is, a creditor prefers those situations in which a norm is not violated to one where it is. Now this implies that a norm, in isolation, has no direct effect on its target's behaviour. Instead, we claim that such behaviour regulation stems from two sources. First, the violation of a norm could (via contrary-to-duties) permit a sanction to be imposed on the violator. Second, social ties could mean that a norm's target takes the norm creditor's preferences into account (e.g. I may fulfil my obligations to my friends because I care about their feelings rather than any threat of sanctions). Note however that in our argument framework, we merge all individual agent preferences into a global preference, limiting the effects of this approach; investigating a more "local" view of preferences forms part of our future work.

Next, we provide a high level overview of the different norm types, before proceeding to formalise them.

Obligations and Prohibitions As mentioned above, obligations identify states of affairs that should be achieved by the *target* of the obligation. Obligations are imposed by some group (the *creditor*) on the target[2]. Furthermore, if an obligation is not fulfilled, then the creditor could potentially sanction the obligation's target. An obligation therefore encodes two concepts, namely the preference by the creditor for paths wherein the obligation is fulfilled over those where it is not. Second, if an obligation is not fulfilled, then a record must be kept that it has been *violated* in order to enable sanctions to be put into place.

In line with goals, we consider two distinct types of obligations, namely *achievement* obligations, which require the target to see to it that some state of affairs holds at some point before some *deadline* occurs, and *maintenance* obligations, which require the target to ensure that the state of affairs holds

[2] Note that this creditor could be the entire society of agents.

at all points before the deadline. Before formally defining obligations, we must examine the notion of a permission, which acts as an exception to an obligation.

Permissions A permission acts as an exception to an obligation (or a prohibition). In other words, given an obligation to achieve some state of affairs, and a permission not to achieve it, not achieving this state of affairs will not result in a violation. As discussed previously, we model prohibitions as negated obligations, and therefore concentrate on the interactions between permissions and obligations. Like other modalities, a permission is given by some creditor to a target, and affects the creditor's concept of a violation. Similarly, permissions identify some (permitted) state of affairs, and a deadline.

Clearly, interpreting a permission in this way makes little sense without an obligation or prohibition being present, and we therefore encode permissions through the presence of a permission proposition, with one such unique proposition being defined for every combination of creditor, target and state of affairs. Since our AATS has only a finite number of agents and propositions, there are a finite number of such proposition symbols. More precisely, we use the proposition $\mathcal{P}^g_{a,x}$ to indicate that agent a has obtained permission from g to see that the state of affairs x *is not the case* in the state where the proposition is true. We can now define a permission through the use of a formula in our logic, writing $P^g_a(x|d)$ as an abbreviation of the formula

$$\mathcal{A}\mathcal{P}^g_{a,x}\mathcal{U}d$$

This formula ensures that a permission is in force over all possible paths in the system until deadline d. Since we must ensure that the permission predicate does not hold when no permission is in force, we require the following axiom in the system:

$$\mathcal{A}\Box(\neg P^g_a(x|d) \rightarrow \neg\mathcal{P}^g_{a,x})$$

Formalising Obligations Obligations identify states of affairs that should hold, and a failure to abide by the requirements of an obligation leads to a violation. We encode such a violation through a violation proposition, in a manner similar to the permission proposition. In other words, the proposition $\mathcal{V}^g_{a,x,d}$ represents a violation by the target a of the obligation, with respect to a creditor g, to see to it that state of affairs x was the case with respect to a deadline d.

An achievement obligation, abbreviated $O^g_a(x|d)$ requiring the target a to ensure that some state of affairs x holds before a deadline d towards a creditor g is represented as follows:

$$\begin{aligned}\mathcal{A}(\neg\mathcal{V}^g_{a,x,d} \wedge \neg d \wedge \neg x)\mathcal{U}\ (((\neg x \wedge d \wedge \neg\mathcal{P}^g_{a,x} \wedge \mathcal{V}^g_{a,x,d})\vee\\ (\neg x \wedge d \wedge \mathcal{P}^g_{a,x} \wedge \neg\mathcal{V}^g_{a,x,d}))\vee\\ (x \wedge \neg\mathcal{V}^g_{a,x,d}))\end{aligned}$$

This obligation therefore requires the following conditions to be met on all possible paths:

1. Before either the deadline or x occurs, the obligation is not considered violated (the first line of the obligation following the $\mathcal{U}$).
2. If the deadline occurs and x is not the case, then if there is no permission allowing this to occur, a violation is recorded. Alternatively, if such a permission exists, then no violation is recorded (this is encoded by the second line of the proposition).
3. Finally, if x is achieved (before the deadline), then no violation is recorded (this is captured by the final line of the proposition).

Therefore, our encoding of an obligation essentially states that if an obligation is in force, it is only violated if the deadline is reached without the desired state of affairs being achieved, assuming that no permission exists allowing the obligation to be ignored. However, nothing in this definition prevents a violation from existing in a state of affairs without an associated obligation. We therefore require that the following axiom hold:

$$\mathcal{A}\Box(\neg O^g_a(x|d) \rightarrow \neg \mathcal{V}^g_{a,x,d})$$

Maintenance obligations requires that a state of affairs be maintained until some deadline[3]. We abbreviate a maintenance obligation on a from g requiring x be the case until deadline d as $O^g_a(m:d)$. This stands for the following formula.

$$\begin{array}{l}\mathcal{A}\ ((\neg x \wedge \neg d \wedge (\neg \mathcal{P}^g_{a,x} \wedge \mathcal{V}^g_{a,x,d}) \vee \\ \quad (\mathcal{P}^g_{a,x} \wedge \neg \mathcal{V}^g_{a,x,d})) \vee (x \wedge \neg d))\mathcal{U}d\end{array}$$

In other words, before the deadline, either x is maintained, or x is not maintained, in which case the obligation is violated if an associated permission does not exist.

The requirement for the lack of a violation, as stated above, is repeated for maintenance obligations:

$$\mathcal{A}\Box(\neg O^g_a(x:d) \rightarrow \neg \mathcal{V}^g_{a,x,d})$$

In discussing obligations so far, we have identified the situations in which they are violated. Detecting these situations allows for the modelling of *contrary to duty* obligations, which come into force when a violation occurs. Such contrary to duties are a form of conditional obligation, which comes into force only when some state of affairs holds in the environment, and generally, such conditionals can be represented via an axiom utilising an implication relation, e.g.

$$\mathcal{A}(\mathcal{V}^g_{a,x,d} \rightarrow O^g_a(x'|d'))$$

We now turn our attention to the second aspect of obligations, namely their interactions with preferences over paths through the system. Informally, the presence of an obligation or prohibition imposed by some creditor leads to that

[3] We assume that this maintenance requirement comes into force with the obligation, ignoring obligations of the form "maintain x between 5pm and 8pm tomorrow".

creditor preferring those paths through the system where the obligation is complied with (i.e. not violated) over those where it is violated. This leads to the following rule within our system:

$$\Box\neg\mathcal{V}^{g}_{a,x,d} \succ^{g} \Diamond\mathcal{V}^{g}_{a,x,d}$$

Note that we do not prefer fewer violations over more violations, as other preferences, for example regarding the interval length of a violation, could affect the preference ordering.

Having formalised permissions and obligations, we now consider prohibitions. In this work we consider only achievement prohibitions, that is, prohibitions on seeing to it that a state of affairs holds (until the prohibition's deadline occurs). Such a prohibition can in fact be modelled as a maintenance obligation — a prohibition on achieving x until some deadline is a maintenance obligation $O^{g}_{a}(\neg x : d)$, requiring the target to ensure x holds until the deadline.

We conclude this section by making several observations regarding our normative system. Unlike many other models, violations in our model do not persist. That is, a violation identifies a single, specific point in time at which an obligation was violated, and is associated with the violated obligation via x and d, the creditor (g) and target (a). Violations are represented as unique propositions in our language.

It should also be noted that our representation of obligations means that an achievement obligation ceases to have force (in the sense of implying a violation) at the moment of deadline; work such as [9] instead specifies that an obligation must still be fulfilled even after it has been violated, and we will investigate this interpretation in future work.

Also note that our preference relation over obligations implies that agents/social groups are, in a sense, "honest", that is, they prefer the outcome implied by compliance with the obligation over one where the obligation is violated.

3 Practical Reasoning via Argumentation

Our formal model contains two distinct aspects. The first aspect consists of the AATS, which identifies all possible evolutions of the system, while the second aspect is associated with the preferences over paths (i.e. sequence of actions) that agents hold. Our aim is to identify whether a most preferred path through the system exists, and explain *why* this is the case. In order to do so,we make use of *argument schemes* [10], defeasible rules expressed in natural language which can be used to justify some conclusion. An argument scheme is associated with a set of *critical questions*, which are used to prevent the inferences of the rule from being made.

The argument schemes we define in the next section are instantiated as arguments within an extended argument framework (EAF) [11]. The evaluation of such an EAF according to a specific argumentation semantics results in a set of *extensions*, each containing a set of arguments. Each of these sets of arguments is, in some sense, justified. We begin by describing our argument schemes

in more detail, following which we describe EAFs and the extension evaluation procedure.

3.1 Argument Schemes

The first scheme we consider puts forth the argument that any sequence of actions through the AATS can be justified. Each path through the AATS thus results in a unique argument which is an instantiation of the following argument scheme.

AS1: In situation S, the sequence of joint actions $A_1, \ldots A_n$ should be executed.

This argument scheme is associated with two critical questions:

CQ1-1 Does some other sequence of actions exist that can be executed?
CQ1-2 Is there a more preferred sequence of actions to this one?

The first critical question will result in symmetric attacks between all instantiations of AS1 for all possible paths (which are instantiations of the sequence of actions) through the system. The second critical question will lead to an asymmetric attack from another AS identifying the more preferred sequence of actions to the less preferred sequence of action. Now a reason for one sequence of actions to be preferred over another is that it achieves a goal, or complies with a norm that is important to the agent. We therefore introduce several additional argument schemes capturing these possible reasons.

AS2: The sequence of joint actions $A_1, \ldots, A_n$ is preferred over $A'_1, \ldots A'_n$ as the former achieves a goal which the latter does not. Critical questions here are as follows:

CQ2-1 Is there some other sequence of actions which achieves a more preferred goal than the one achieved by this action sequence?
CQ2-2 Does the sequence of actions lead to the violation of a norm?

AS3 and AS4 are argument schemes that deal with obligations and permissions:

AS3: The sequence of actions $A_1, \ldots A_n$ should be less preferred than sequence $A'_1, \ldots A'_n$ as, in the absence of permissions, the former violates a norm while the latter does not.

CQ3-1 Is the goal resulting from the sequence of actions more preferred than the violation?
CQ3-2 Does the violation resulting from this norm result in some other, more important violation not occurring?
CQ3-3 Is there a permission that derogates the violation?

AS4: There is a permission that derogates the violation of an obligation.

Note that the separation between AS3 and AS4 is intended purely for explanatory purposes; conceptually, it would be possible to merge both of these schemes into one by considering an argument scheme which deals with violation once permissions are considered.

Finally, we can identify several simple argument schemes that allow an agent to associate preferences between different goals and norms, thereby enabling the instantiation of the critical questions for AS2 and AS3.

AS5: Agent α prefers goal g over goal g'

AS6: Agent α prefers achieving goal g to not violating n

AS7: Agent α prefers not achieving goal g to violating n

AS8: Agent α prefers violating n to violating n'

AS9: Agent α prefers situation A to B

This last argument scheme is intended to allow an agent to express individual preferences with regards to outcomes.

3.2 Argument Scheme Semantics

We now provide a brief formalisation of the argument schemes and critical questions based on our AATS semantics. Above, our argument schemes referred to sequences of actions, which are equivalent to paths through the AATS. As done previously, we label this AATS S below. Our formalisation makes use of the formulae obtained from goals and norms to express preferences over paths. That is, given S, and preferences expressed using CTL* formula ϕ and ψ of the form $\phi \succeq^a \psi$, We specify a set of *path preferences* $\lambda \geq^a \lambda'$ for any paths λ, λ' where $S, \lambda \models \phi$ and $S, \lambda' \models \psi$.

Given a sequence of actions $j_1, \ldots, j_n$, we can obtain a path λ as the path beginning in the initial state $q_0 \in Q$, and for which for all $i = 1 \ldots n$, $\tau(q_{i-1}, j_i) = q_i$.

Given an AATS, we can then identify valid installations of the argument schemes and critical questions, resulting in an argument framework whose evaluation allows us to determine justified action sequences.

AS1: There is a path λ obtained from the sequence of actions $j_1, \ldots j_n$.

AS2: There is a goal γ and two paths λ, λ' obtained from the sequence of joint actions , $j_1, \ldots j_n$ and $j'_1, \ldots j'_m$ respectively, and it is the case that $S, \lambda \models \gamma$ and $S, \lambda' \not\models \gamma$.

AS3: There exist two paths λ, λ' obtained from the sequence of joint actions $j_1, \ldots j_n$ and $j'_1, \ldots j'_m$ respectively, and it is the case that $S \backslash \mathcal{P}^g_{a,x}, \lambda \models \mathcal{V}^g_{a,x,d}$ and $S \backslash \mathcal{P}^g_{a,x}, \lambda' \not\models \mathcal{V}^g_{a,x,d}$

AS4: There is a path λ obtained from the sequence of joint actions $j_1, \ldots j_n$, and $S \backslash \mathcal{P}^g_{a,x}, \lambda \models \mathcal{V}^g_{a,x,d}$ but $S, \lambda \not\models \mathcal{V}^g_{a,x,d}$.

AS5-AS9 express individual agent preferences between goals, and violations. For example, an agent may prefer to achieve one goal over another (AS5), or avoid achieving a goal if it means violating a norm (AS7).

AS5: There are goals γ, γ' where $S, \lambda \models \gamma$ and $S, \lambda' \models \gamma'$ and $\gamma \succeq^\alpha \gamma'$

AS6: There is a goal γ and violation $\mathcal{V}^g_{a,x,d}$ such that $\gamma \succeq^\alpha \neg\mathcal{V}^g_{a,x,d}$

AS7: There is a goal γ and violation $\mathcal{V}^g_{a,x,d}$ such that $\neg\gamma \succeq^\alpha \mathcal{V}^g_{a,x,d}$

AS8: There are two violations $\mathcal{V}^g_{a,x,d}$, $\mathcal{V}^h_{b,y,e}$ such that $\mathcal{V}^g_{a,x,d} \succeq^\alpha \mathcal{V}^h_{b,y,e}$

AS9: $A \succeq^\alpha B$ where A, B are formulae in our language.

Now let us turn our attention to the critical questions, using the same definitions as above.

CQ1-1: There is a sequence of joint actions $j'_1, \ldots j'_n$ such that for some $i \in 1 \ldots n$ $j_i \neq j'_i$.

CQ1-2: There is an instance of AS2 or AS3 whose path λ is created by the sequence of joint actions of this AS1. Alternatively, there is an instance of AS9 whose path B is equivalent to λ created by the sequence of joint actions of this AS1.

CQ2-1: There an instance of AS5 whose less preferred goal is the one identified by this instantiation of AS2.

CQ2-2: There is an instance of AS3 whose path λ is the λ path for AS2.

CQ3-1: There is an instance of AS6 for $S, \lambda \models \gamma$ and $S, \lambda \models \mathcal{V}^g_{a,x,d}$, where λ is the first path of AS3.

CQ3-2: There is an instantiation of AS8 for which this instantiation of AS3 means that $S \backslash \mathcal{P}^g_{a,x}, \lambda \models \mathcal{V}^g_{a,x,d}$ and $S \backslash \mathcal{P}^g_{a,x}, \lambda \not\models \mathcal{V}^h_{b,y,e}$

CQ3-3: There is an instantiation of AS4 referring to a permission $\mathcal{P}^g_{a,x}$ which refers to the same path λ as this instantiation of AS3.

3.3 Instantiating the Framework

We instantiate the framework described above using Modgil's extended argument frameworks (EAF) [11]. Formally, an EAF is defined as follows:

Definition 5. ***(Extended Argument Framework)*** *An EAF is a tuple* $(Args, R, D)$ *such that Args is a set of arguments,* $R \subseteq Args \times Args$*, and* $D \subseteq Args \times R$ *subject to the constraint that if* $(C, (A, B)), (C', (B, A)) \in D$*, then* $(C, C'), (C', C) \in R$

Each instantiation of any of the argument schemes is associated with an argument within our EAF, and each critical question is associated with an attack on the argument scheme instantiation to which this critical question belongs. The constraint imposed on EAFs causes additional attacks to appear that are not described by the critical questions. We describe the process of EAF instantiation informally due to both space concerns and its simplicity.

CQ1-1 arises since only one sequence of actions can ultimately be executed, and results in symmetric attacks being inserted into R between every pair of nodes instantiating $AS1$. CQ1-2 refers to preferences between actions and following [12], is captured via an attack from the node representing the argument to the appropriate attacking edge.

CQ2-1, CQ2-2, CQ3-1 and CQ3-2 capture preferences over goals and norms. That is, they are used to represent the fact that one goal (or norm) is preferred over some other goal (or norm) by entities in the system. All of these link the appropriate argument, as instantiated by AS5-8 via an attack, on the attack from the argument instantiated by the appropriate AS2 or AS3.

Finally, CQ3-3 encompasses the possibility of a violation being derogated by a permission, and in instantiated as an attack from AS8 to the appropriate AS3.

Given the above, CQ1-1 and CQ3-3 result in attacks added to R, while the remaining critical questions result in attacks added to D. Together with the attacks added by the constraint, these attacks between the arguments instantiated from the application of the argument schemes fully specify our EAF.

Given an EAF instantiated as above, all the preferred extensions of the EAF will contain a single argument from argument scheme AS1 for some specific action sequence to be executed iff this action sequence is most preferred by all agents in the system. This sequence of actions is the dominant strategy for all agents in the system. Therefore, each preferred extension of the EAF identifies a single most preferred sequence of action.

The presence of multiple preferred extensions indicates that there are multiple most preferred sequences of action. In most multi-agent situations, this is an undesirable situation, as additional coordination is then required between the agents to ensure that a most desired sequence of joint actions is executed. This would require more refined reasoning about plans (e.g. [13]) to take place.

Finally, an empty set of extensions indicates that there is a preference conflict that must be resolved before a course of action can be agreed on.

4 Example

In this section, we provide a brief example of the framework in action. Due to space constraints, we do not present all details of the system in our example, but instead concentrate on the most important aspects of the system's operation.

Consider two agents, α and β. α can undertake two actions, namely to visit her ill mother in hospital (V), or go to work (W). β, who is α's boss, has two possible actions, namely to fire α (F), or not fire her (N). α has two (conflicting) goals: to visit her mother (vm), and to keep her job (kj), while β would like to see some work done (wd), which can only occur if α goes to work. Finally, β has an obligation to not fire α, but has permission to do so if she does come to work.

The AATS for this example is shown in the top left of Figure 1, and instantiating the EAF results in the main graph of Figure 1. Within this graph, paths from the AATS are indicated through nodes containing the path number; preference information is encoded through the propositions true in the state (e.g. $1 > 3$ kj indicate that path 1 is preferred to path 3 due to α's preference to keep her job; the permission to fire is indicated via the *per* node, and nvl identifies preference nodes instantiated through the prohibition on firing α. Dashed lines indicate attacks due to actions being mutually exclusive, while solid lines capture preference based attacks.

Evaluating the preferred extension of this EAF indicates that multiple actions are possible; for example, paths 1 and 2 are present in two of the extensions. This means that the system's preferences are underspecified. Looking at the situation more closely, this occurs for several reasons. First, α does not have any preferences encoded between going to the hospital or keeping her job; prioritising one of these (by adding attacks on edges between kj and vm via an instantiation of AS9) reduces the number of extensions, for example, if vm is preferred over kj,

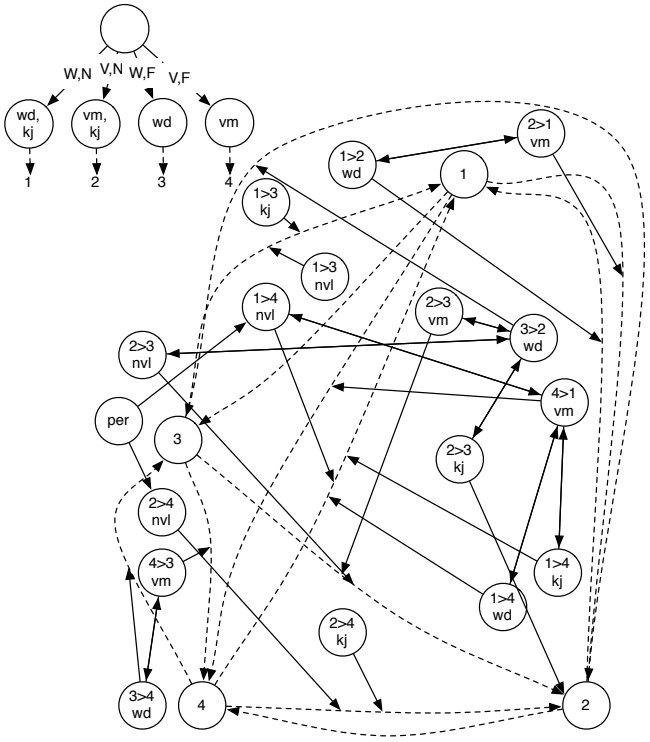

Fig. 1. The AATS (top left) and EAF (main figure) of the example.

only path 2 remains in the extension indicating that α should visit her mother and keep her job.

This odd result arises because while β has permission to fire α if she does not turn up to work, no preferences are expressed over whether β would prefer this situation to one where α keeps her job. Adding an additional preferences over paths, through a new goal for β stating that either the work is done and α keeps her job, or the work is not done and α is fired, will result in α losing her job if she visits her mother in hospital (path 4). Note that due to the permission, there is no need to then express another preference for β between this goal and the norm on not firing α; without the permission, such an additional preference would be necessary.

5 Discussion and Future Work

In practice, there are several ways of using the framework proposed here, each of which poses an avenue for future research. First, as done in the example above, a given AATS could be converted to an EAF and evaluated in order to identify

whether sufficient preference information has been provided in order to reach a decision about a sequence of actions. The potential exponential growth in the number of arguments with respect to AATS size makes this approach practical for only small AATSs.

Second, a dialogue game could be formulated (and verified against an AATS) based on arguments and attacks instantiated from the argument schemes and critical questions. This would involve agents arguing for why some course of action should be taken via utterances regarding their goals, norms and preferences.

Third, and perhaps most novel, an instantiated EAF could be used as the basis of a process to explain why some sequence of actions was followed given agents with some goals and norms. A user could, for example, understand that an action was executed as while a norm was violated, the goal achieved was more important to the agents in the system than the violation.

Our work borrows several ideas from Atkinson's argument scheme for practical reasoning based on values [14]. Atkinson's approach puts both goals and values at the centre of the argumentation scheme, stating that "in situation S, action A should be pursued in order to achieve goal G while promoting values V". This argument scheme is encoded through a VAF, which is used to represent the preferences of different audiences over values. Each argument within the VAF can be associated with several values, but an audience's value ordering must be fully specified and consistent.

In the current work, preferences (which have a similar role to Atkinson's values) are associated with different sequences of action due to the goals that these sequences achieve for the agents as well as the norms violated or complied with by the sequence. Given this, our AS1 argument scheme is much simpler than Atkinson's, stating that (by default) some sequence of actions should be executed, and requiring all possible sequences of actions to be mutually exclusive with each other. Deciding how to act then requires identifying the most preferred sequence of actions. Unlike [14], our work explicitly considers norms in practical reasoning, and considers all possible interactions between norms and goals.

Our representation of preferences within an EAF is based on [12], which applied EAFs to VAFs. While there are many similarities between our instantiation and the VAF instantiation, the requirement of VAFs to have a single consistent preference ordering makes them unsuitable for our needs; as shown above, we explicitly concern ourselves with detecting inconsistent preference orderings.

In one sense, the work presented here takes a global view of norms and actions. We consider joint actions, and require that all agents agree on a path. Such an approach ignores an important nuances of practical reasoning: agents may be force to pursue sub-optimal goals due to the actions of other agents. Thus, while our approach currently finds dominant strategies, it is unable to find other game theoretic solution concepts (e.g. Nash equilibria); we believe that capturing these additional solution concepts is critical, and are currently investigating how these concepts can be captured using our approach. This will make more extensive use of the notion of a norm's creditor and target, and the preferences of each with regards to specific outcomes.

Another interesting avenue of future work involves considering a more dynamic system where new obligations, permissions and prohibitions can be created and removed as the system executes, and agents goals can change over time.

Finally, integrating practical reasoning over norms with reasoning over values would also be useful. This, in combination with the already present capability to reason over goals, should provide an end-to-end practical reasoning formalism.

6 Conclusions

In this paper we proposed a representation for norms built on top of an AATS. Using this representation we described how arguments over norms can be constructed, allowing for the detection of inconsistencies when performing practical reasoning, the explanation of *why* some action was taken, and making a decision about how to act in the presence of both goals and norms.

References

1. Broersen, J., Dastani, M., Hulstijn, J., Huang, Z., van der Torre, L.: The BOID architecture: conflicts between beliefs, obligations, intentions and desires. Proc. Fifth International Conference on Autonomous Agents, (2001) 9–16
2. van der Hoek, W., Roberts, M., Wooldridge, M.: Social laws in alternating time: effectiveness, feasibility, and synthesis. Synthese **156** (2007) 1–19
3. Emerson, E.A., Halpern, J.Y.: 'sometimes' and 'not never' revisited: on branching versus linear time temporal logic. J. ACM **33**(1) (1986) 151–178
4. Hindriks, K.V., van Riemsdijk, M.B.: Satisfying maintenance goals. In: Proc. DALT'07 (2008) 86–103
5. Boella, G., van der Torre, L.: Permissions and obligations in hierarchical normative systems. In: Proceedings of ICAIL 03, Edinburgh, Scotland (2003)
6. Boella, G., van der Torre, L.: Institutions with a hierarchy of authorities in distributed dynamic environments. Artificial Intelligence Law **16** (2008) 53–71
7. Croitoru, M., Oren, N., Miles, S., Luck, M.: Graphical norms via conceptual graphs. Knowledge-Based Systems **29** (2012) 31–43
8. Singh, M.P.: An ontology for commitments in multiagent systems: Toward a unification of normative concepts. Artificial Intelligence and Law **7** (1999) 97–113
9. Governatori, G., Hulstijn, J., Riveret, R., Rotolo, A.: Characterising deadlines in temporal modal defeasible logic. In: Proceedings of AI-2007 (2007).486–496
10. Walton, D.N.: Argumentation Schemes for Presumptive Reasoning. Erlbaum (1996)
11. Modgil, S.: Reasoning about preferences in argumentation frameworks. Artificial Intelligence **173**(9–10) (2009) 901–934
12. Mogdil, S., Bench-Capon, T.: Integrating object and meta-level value based argumentation. In: Proc. COMMA 2008, Amsterdam, IOS Press (2008) 240–251
13. Medellin-Gasque, R., Atkinson, K., Bench-Capon, T.: Arguments over co-operative plans. In: Proc. TAFA'11, Springer (2012) 50–66
14. Atkinson, K., Bench-Capon, T.J.M.: Practical reasoning as presumptive argumentation using action based alternating transition systems. Artif. Intell. **171**(10-15) (2007) 855–874