# Planning with Preferences using Logic Programming

Tran Cao Son Enrico Pontelli

Department of Computer Science
New Mexico State University
Las Cruces, NM 88003, USA
{tson|epontell}@cs.nmsu.edu

**Abstract.** We present a declarative language, $\mathcal{PP}$, for the specification of preferences between possible solutions (or trajectories) of a planning problem. This novel language allows users to elegantly express non-trivial, multi-dimensional preferences and priorities over them. The semantics of $\mathcal{PP}$ allows the identification of *most preferred trajectories* of a given goal. We provide a transformation to logic programming with negation as failure, that allows the use of existing logic programming systems to solve planning problems with $\mathcal{PP}$ preferences.

## 1 Introduction

Planning—in its classical sense—is the problem of finding a sequence of actions that achieves a predefined goal. As such, much of the research in AI planning has been focused on methodologies and issues related to the development of efficient planners. To date, several efficient planning systems have been developed (e.g., see [18] for a summary of planners that competed in the International Conference on Artificial Intelligent Planning and Scheduling). These developments can be attributed to the discovery of good domain-independent heuristics, the use of domain-specific knowledge, and the development of efficient data structures used in the implementation of the planning algorithms. Logic programming has played a significant role in this line of research, providing a declarative framework for the encoding of different forms of knowledge and its effective use during the planning process [24].

However, relatively limited effort has been placed on addressing several important aspects in real-world planning domains, such as *plan quality* and *preferences about plans*. In many real world situations, the space of feasible plans to achieve the goal is dense, but many of such plans, even if executable, may present undesirable behavior. In these situations, it may not be difficult to find a solution; rather, the challenge is to produce a solution that is considered satisfactory w.r.t. the needs and preferences of the user. Thus, feasible plans may have a measure of quality and only a subset may be considered acceptable. These issues can be illustrated with the following example:

*Example 1. It is 7 am and* Bob*, a Ph.D. student, is at home. He needs to be at school at 8am to take his qualification exam. His car is broken and he cannot drive to school. He can take a bus, a train, or a taxi to go to school, which will take him 55, 45, or 15 minutes respectively. Taking the bus or the train will require* Bob *to walk to the nearby station, which may take 20 minutes. However, a taxi can arrive in only 5 minutes. When in need of a taxi,* Bob *can call either the* MakeIt50 *or the* PayByMeter *taxi company.* MakeIt50

*will charge a flat rate of $50 for any trip, while* PayByMeter *has a fee schedule of $20 for the trip to school. If he takes the bus or the train, then* Bob *will spend only $2. Furthermore,* Bob*, being a student, prefers to pay less whenever possible.*

*It is easy to see that there are only two* feasible *plans for* Bob *to arrive at the school on time for his exam: calling one of the two taxi companies. However, a* PayByMeter *taxi would be preferable, as* Bob *wants to save money.* In this case, both plans are feasible but *Bob*'s preference is the deciding factor to select which plan he will follow.

The example demonstrates that users' preferences play a deciding role in the choice of a plan. Thus, we need to be able to evaluate plan components at a finer granularity than simply as consistent or violated. In [20], it is argued that users' preferences are likely to be more important in selecting a plan for execution, when a planning problem has too many solutions. It is worth noticing that, with a few exceptions, like the system SIPE-2 with metatheoretic biases [20], most planning systems *do not* allow users to specify their preferences and to use them in finding the plans. As such, the responsibility in selecting the most appropriate plan for their purpose rests solely on the users. It is also important to observe that *preferences* are different from *goals* in a planning problem; they might or might not be satisfied by a plan. The distinction is similar to the separation between *hard* and *soft* constraints [3]. For instance, if *Bob*'s goal is to spend at most $2 to go to school, then he does not have any feasible plans to arrive at school on time.

In this paper, we will investigate the problem of integrating users' preferences into a logic programming-based planner. We will develop a language for the specification of user preferences, and then provide a logic programming implementation of the language, based on answer set programming. As demonstrated in this work, normal logic programs with answer set semantics provide a natural and elegant framework to effectively handle planning with preferences. We divide the preferences that a user might have into different categories:

- *Preference about a state:* the user prefers to be in a state $s$ that satisfies a property $\phi$ rather than a state $s'$ that does not satisfy it, even though both satisfy his/her goal;
- *Preference about an action:* the user prefers to perform the action $a$, whenever it is feasible and it allows the goal to be achieved;
- *Preference about a trajectory:* the user prefers a trajectory that satisfies a certain property $\psi$ over those that do not satisfy this property;
- *Multi-dimensional Preferences:* the user has a *set* of preferences about the trajectory, with an ordering among them. A trajectory satisfying a higher priority preference is preferred over those that satisfy lower priority preferences.

It is important to observe the difference between $\phi$ and $\psi$ in the above definitions. $\phi$ is a *state* property, whereas $\psi$ is a formula over the whole *trajectory* (from the initial state to the state that satisfies the given goal).

The language for the specification of user preferences is developed in successive steps. First, we introduce a simple mechanism to deal with the first three types of preferences, called *desires* (Section 3.1). Preferences about preferences and multi-dimensional preferences are dealt with through the use of *atomic* and *general* preferences—i.e., chains of preferences (Section 3.2) and partial orders between collections of preferences (Section 3.3). A logic programming realization of the language used to express preferences is given in Section 4.

### 1.1 Related Work

This work is a continuation and improvement of our previous work [26], in which we rely on the prioritized default theory framework to express preferences between trajectories in logic programming. Only a few types of preferences can be expressed in [26]. This work is also strongly influenced by other works on exploiting *domain-specific knowledge* in planning (e.g., [2, 24]), in which domain-specific knowledge is expressed as constraints on the trajectories achieving the goal, and hence, are *hard constraints*.

Numerous approaches have been proposed to integrate preferences in the planning process. Eiter et al. introduced a framework for planning with action costs using logic programming [9]. Each action is assigned an integer cost, and plans with the minimal cost are considered optimal. Costs can be either static or relative to the time step in which the action is executed. [9] also presents the encoding of different preferences, such as shortest plan and the cheapest plan. Our approach also emphasizes the use of logic programming, but differs in several aspects. Here, we develop a declarative language for preference representation. Our language can express all of the preferences discussed in [9], but it is more flexible and high-level than the action costs approach. The approach in [9] also does not allow the use of fully general dynamic preferences. Other systems have adopted fixed types of preferences, e.g., shortest plans [6, 4].

Out proposal has similarities with the approach based on metatheories of the planning domain [19, 20], where metatheories provide characterization of semantic differences between the various domain operators and planning variables; metatheories allow the generation of biases to focus the planner towards plans with certain characteristics.

The problem of maintaining and managing preferences has also been investigated in the framework of constraint programming (e.g., through soft constraints [3] or relational optimizations [11]). Constraint solving has also been proposed as the basis for the management of planning in presence of action costs [16].

Considerable effort has been invested in introducing preferences in logic programming. In [14, 7] preferences are expressed at the level of atoms and used for parsing disambiguation in logic grammars. Rule-level preferences have been used in various proposals for selection of preferred answer sets in answer set programming [5, 8, 23].

Our language allows the representation of several types of preferences, similar to those developed in [15] for decision-theoretic planners. The main difference is that we use logic programming while their system is probability based. Our approach also differs from the works on using Markov Decision Processes (MDP) to find optimal plans [22]; in MDPs, optimal plans are functions from states to actions, thus preventing the user from selecting preferred trajectories without changing the MDP specification.

## 2 Preliminary – Answer Set Planning

In this section we review the basics of planning using logic programming with answer set semantics—*Answer Set Planning (or ASP)* [17]. We will assume that the effect of actions on the world and the relationship between fluents in the world are expressed in an appropriate language. In this paper, we will make use of the ontologies of the action description language $\mathcal{B}$ [12]. In $\mathcal{B}$, an action theory is defined over two disjoint sets—the set of actions **A** and the set of fluents **F**; an action theory is a pair $(D, I)$, where $D$ is a set of propositions expressing the effects of actions, the relationship between

fluents (also called static causal laws), and the executability conditions of actions; $I$ is a set of propositions representing the initial state of the world. For example, the action of calling a taxi has the effect of the taxi arriving, and it is represented in $\mathcal{B}$ as follows:

$$call\_taxi \textbf{ causes } taxi\_arrived.$$

Realistically, one has to have enough money to call a taxi. This is expressed in $\mathcal{B}$ by the proposition:

$$call\_taxi \textbf{ executable\_if } has\_enough\_money.$$

The semantics of an action theory is given by the notion of a *state*—a consistent set of fluent literals[1] that satisfies the relationship between fluents —and a *transition function* $\Phi$ that specifies the result of the execution an action $a$ in a state $s$, denoted by $\Phi(a, s)$. A *trajectory* of an action theory $(D, I)$ is a sequence $s_0 a_1 s_1 \ldots a_n s_n$ where $s_i$'s are states, $a_i$'s are actions, and $s_{i+1} \in \Phi(s_i, a_{i+1})$ for $i \in \{0, \ldots, n-1\}$. A state $s$ satisfies a fluent literal $f$, denoted by $s \models f$, if $f \in s$. This is extended over the propositional connectives to define $s \models \varphi$ where $\varphi$ is a fluent formula. Since our main concern in this paper is not the language for representing actions and their effects, we omit here the detailed definitions of $\mathcal{B}$. They can be found in [12].

A planning problem is specified by a triple $\langle D, I, G\rangle$, where $(D, I)$ is an action theory and $G$ is a fluent formula (a propositional formula based on fluent literals) representing the goal. A possible solution to $\langle D, I, G\rangle$ is a trajectory $s_0 a_1 s_1 \ldots a_m s_m$, where $s_0 \models I$ and $s_m \models G$. In this case, we say that the trajectory achieves $G$.

Answer set planning [12, 17] solves a planning problem $\langle D, I, G\rangle$ by translating it into a logic program $\Pi(D, I, G)$ consisting of *domain-dependent* rules that describe $D$, $I$, and $G$ and *domain-independent* rules that generate action occurrences and represent the transitions between states. Besides the planning problem, $\Pi(D, I, G)$ requires an additional parameter: the maximal *length* of the trajectory that the user can accept. The two key predicates of $\Pi(D, I, G)$ are:

- $holds(f, t)$ – the fluent literal $f$ holds at the time moment $t$; and
- $occ(a, t)$ – the action $a$ occurs at the time moment $t$.

$holds(f, t)$ can be extended to define $holds(\phi, t)$ for an arbitrary fluent formula, which states that $\phi$ holds at the time $t$. Details about the program $\Pi(D, I, G)$ can be found in [25, 17]. The key property of the translation of $\langle D, I, G\rangle$ into $\Pi(D, I, G)$ is that it ensures that each trajectory achieving $G$ corresponds to an answer set [13] of $\Pi(D, I, G)$, and each answer set of $\Pi(D, I, G)$ corresponds to a trajectory achieving $G$.

**Theorem 1.** *[25] For a planning problem $\langle D, I, G\rangle$ with a consistent action theory $(D, I)$ and the maximal plan length $n$,*

*(i) if $s_0 a_0 \ldots a_{n-1} s_n$ is a trajectory achieving $G$, then there exists an answer set $M$ of $\Pi(D, I, G)$ such that:* **(a)** *$occ(a_i, i) \in M$ for $i \in \{0, \ldots, n-1\}$ and* **(b)** *$s_i = \{f \mid holds(f, i) \in M\}$ for $i \in \{0, \ldots, n\}$.*

*(ii) if $M$ is an answer set of $\Pi(D, I, G)$, then there exists an integer $0 \leq k \leq n$ such that $s_0 a_0 \ldots a_{k-1} s_k$ is a trajectory achieving $G$, where $occ(a_i, i) \in M$ for $0 \leq i < k$ and $s_i = \{f \mid holds(f, i) \in M\}$ for $i \in \{0, \ldots, n\}$.*

[1] A fluent literal is a fluent $f$ or its negation $\neg f$. A fluent formula is a propositional formula over fluent literals.

Answer sets of the program $\Pi(D, I, G)$ can be computed using answer set solvers such as **smodels** [21], **dlv** [10], **cmodels** [1], or **jsmodels** [27].

# 3 A Language for Planning Preferences Specification

In this section, we introduce the language $\mathcal{PP}$ for planning preference specification. Let $\langle D, I, G\rangle$ be a planning problem, with actions **A** and fluents **F**; let $\mathcal{F}_F$ the set of all fluent formulae. $\mathcal{PP}$'s specifications are defined as special formulae constructed over **A** and **F** of $(D, I)$. We subdivide preferences in different classes: *basic desires, atomic preferences (or chains)*, and *general preferences*.

## 3.1 Basic Desires

A basic desire is a formula expressing a preference about a trajectory. For example, *Bob*'s basic desire is to save money; this implies that he prefers to use the train or the bus to go to school, which in turn means that a preferred trajectory for *Bob* should contain the action $take\_bus$ or $take\_train$. This preference could also be expressed by a formula that forbids the fluent $taxi\_arrived$ to become true in every state of the trajectory. These two alternatives of preference representation are not always equivalent. The first one represents the desire of leaving a state by a specific group of actions while the second one represents the desire of being in certain states. Basic desires are composed of *state desire* and *goal preference*. They are defined next.

**Definition 1 (State Desire).** *A (primitive) state desire is either a formula $\varphi$, where $\varphi \in \mathcal{F}_F$, or a formula of the form $occ(a)$, where $a \in$ **A**.*

Intuitively, a state desire describes a basic user preference to be considered in the context of a specific state. A state desire $\varphi$ implies that we prefer a state $s$ such that $s \models \varphi$. A state desire $occ(a)$ implies that we prefer to leave the state $s$ using the action $a$. In many cases, it is also desirable to talk about the final state of the trajectory. We call this a *goal preference* and it is defined as follows.

**Definition 2 (Goal Preference).** *A* goal preference *is a formula of the form **goal**$(\varphi)$, where $\varphi$ is a formula in $\mathcal{F}_F$.*

We are now ready to define a basic desire that expresses an user's preference over the whole trajectory. As such, in addition to the propositional connectives $\wedge, \vee, \neg, \Rightarrow$, we will also use the temporal connectives **next**, **always**, **until**, and **eventually**.

**Definition 3 (Basic Desire Formula).** *A* Basic Desire Formula *is defined as follows:*

- *$\varphi$—if $\varphi$ is a goal preference;*
- *$\varphi$—if $\varphi$ is a state desire;*
- *if $\varphi_1, \varphi_2$ are basic desire formulae, then the following are basic desire formulae: $\varphi_1 \wedge \varphi_2$, $\varphi_1 \vee \varphi_2$, $\neg\varphi_1$,* **next**$(\varphi_1)$, **until**$(\varphi_1, \varphi_2)$, **always**$(\varphi_1)$, *and* **eventually**$(\varphi)$.

*Only formulae satisfying these conditions are considered to be basic desire formulae.*

To express that *Bob*'s would like to take the train or the bus to school, we can write:

$$\textbf{eventually}(occ(take\_bus) \vee occ(take\_train)).$$

On the other hand, to express the desire that *Bob* does not want to call a taxi:

$$\textbf{always}(\neg occ(call\_taxi)).$$

Alternatively, we could write: $\textbf{always}(\neg taxi\_arrived)$.

We note that this alternative might not reflect the same preference as the last one does.

The definition above is used to develop formulae expressing a desire regarding the structure of trajectories. In the next definition, we will specify when a trajectory satisfies a basic desire formula. In a later section, we will present logic programming rules that can be added to the program $\Pi(D, I, G)$ to compute trajectories that satisfy a basic desire. In the following definitions, given a trajectory $\alpha = s_0 a_1 s_1 \cdots a_n s_n$, the notation $\alpha[i]$ denotes the trajectory $s_i a_{i+1} s_{i+1} \cdots a_n s_n$.

**Definition 4 (Basic Desire Satisfaction).** *Let $\alpha = s_0 a_1 s_1 a_2 s_2 \cdots a_n s_n$ be a trajectory, and let $\varphi$ be a basic desire formula. $\alpha$ satisfies $\varphi$ (written as $\alpha \models \varphi$) iff*

- $\varphi = \textbf{goal}(\psi)$ *and* $s_n \models \psi$
- $\varphi = \psi \in \mathcal{F}_F$ *and* $s_0 \models \psi$
- $\varphi = occ(a)$ *and* $a_1 = a$
- $\varphi = \psi_1 \wedge \psi_2$ *and* $\alpha \models \psi_1$ *and* $\alpha \models \psi_2$
- $\varphi = \psi_1 \vee \psi_2$ *and* $\alpha \models \psi_1$ *or* $\alpha \models \psi_2$
- $\varphi = \neg\psi$ *and* $\alpha \not\models \psi$
- $\varphi = \textbf{next}(\psi)$ *and* $\alpha[1] \models \psi$
- $\varphi = \textbf{always}(\psi)$ *and* $\forall(1 \leq i \leq n)$ *we have that* $\alpha[i] \models \psi$
- $\varphi = \textbf{eventually}(\psi)$ *and* $\exists(1 \leq i \leq n)$ *such that* $\alpha[i] \models \psi$
- $\varphi = \textbf{until}(\psi_1, \psi_2)$ *and* $\exists(1 \leq i \leq n)$ *such that* $\forall(1 \leq j < i)$ *we have that* $\alpha[j] \models \psi_1$, *and* $\alpha[i] \models \psi_2$.

Definition 4 allows us to check whether a trajectory satisfies a basic desire. This will also allow us to compare trajectories. Let us start with the simplest form of trajectory preference, involving a single basic desire.

**Definition 5 (Ordering between Trajectories w.r.t. A Single Basic Desire).** *Let $\varphi$ be a desire formula and let $\alpha$ and $\beta$ be two trajectories. The trajectory $\alpha$ is preferred to the trajectory $\beta$ (denoted by $\alpha \prec_\varphi \beta$) if $\alpha \models \varphi$ and $\beta \not\models \varphi$.*

*We say that $\alpha$ and $\beta$ are indistinguishable (denoted by $\alpha \approx_\varphi \beta$) if one of the two following cases occurs:* (i) $\alpha \models \varphi$ *and* $\beta \models \varphi$, *or* (ii) $\alpha \not\models \varphi$ *and* $\beta \not\models \varphi$

Whenever it is clear from the context, we will omit $\varphi$ from $\prec_\varphi$ and $\approx_\varphi$. We will also allow a weak form of single preference:

**Definition 6 (Weak Single Desire Preference).** *Let $\varphi$ be a desire formula and let $\alpha, \beta$ be two trajectories. The trajectory $\alpha$ is weakly preferred to $\beta$ (denoted as $\alpha \preceq_\varphi \beta$) iff either $\alpha \prec \beta$ or $\alpha \approx \beta$.*

**Proposition 1.** *The relation $\preceq_\varphi$ defines a partial order over the trajectories.*

These definitions are expressive enough to represent a significant portion of preferences that frequently occur in real-world domains. Since some of them are particularly important, we will introduce some syntactic sugars to simplify their description:

- (Strong Desire) given the desire formulae $\varphi_1, \varphi_2$, $(\varphi_1 < \varphi_2)$ denotes $\varphi_1 \wedge \neg\varphi_2$.

- (Weak Desire) given the desire formulae $\varphi_1, \varphi_2$, $(\varphi_1 <^w \varphi_2)$ denotes $\varphi_1 \vee \neg\varphi_2$.
- (Enabled Desire) given two actions $a_1, a_2$, we will denote with $a_1 <^e a_2$ the formula $executable(a_1) \wedge executable(a_2) \Rightarrow occ(a_1) < occ(a_2)$. This can be extended to include disjunction of actions on each side of the formula.

We can prove some simple properties of these syntactic sugars.

**Lemma 1.** *The relation $<^w$ over the set of desire formulae is transitive.*

**Definition 7 (Most Preferred Trajectory w.r.t. A Basic Desire).** *A trajectory $\alpha$ is a* most preferred trajectory *w.r.t. a basic desire $\varphi$, if there is no trajectory $\beta$ s.t. $\beta \prec_\varphi \alpha$.*

*Example 2.* We continue with the theory in Example 1 but enrich it with an action, called *buy_coffee*, which allows $Bob$ to have coffee. He can do it only at the station though. To say that *Bob* prefers to have some coffee before he takes the exams, we can write: **goal**(*have_coffee*). Any plan satisfying this preference requires that *Bob* stops at the station before going to school.

### 3.2 Atomic Preferences and Chains

Basic desire formulae allow users to specify their preferences and can be used in selecting trajectories that satisfy them. From the definition of a basic desire formula, we can assume that users always have a set of desire formulae and that their desire is to find a trajectory that satisfies all formulae. In many cases, this proves to be too strong, and results in situations where no preferred trajectories can be found. For example, *time* and *cost* are often two criteria that a person might have when making a travel plan. This two criteria are often in conflict, i.e., transportation method that takes little time often costs more. It is very unlikely that the user can get a plan that can satisfy both criteria. Consider again Example 1, it is obvious that *Bob* cannot have a plan that costs him only \$2 and still allows him to be on-time. To address this problem, we allow a new type of formulae that we call *atomic preference* which represents an ordering between basic desire formulae.

**Definition 8 (Atomic Preference).** *An* atomic preference formula *is a formula of the form $\varphi_1 \lhd \varphi_2 \lhd \cdots \lhd \varphi_n$ ($n \geq 1$) where $\varphi_1, \ldots, \varphi_n$ are basic desire formulae.*

The intuition behind an atomic preference is to provide an ordering between different desires—i.e., it indicates that trajectories that satisfy the desire $\varphi_1$ are preferable to those that satisfy $\varphi_2$, etc. Observe that basic desire formulae are special cases of atomic preferences ($n = 1$). We now extend the definitions of $\approx$ and $\prec$ to compare trajectories with respect to atomic preferences.

**Definition 9 (Ordering Between Trajectories w.r.t. Atomic Preferences).** *Let $\alpha, \beta$ be two trajectories, and let $\Psi = \varphi_1 \lhd \varphi_2 \lhd \cdots \lhd \varphi_n$ be an atomic preference formula.*

- *$\alpha, \beta$ are indistinguishable w.r.t. $\Psi$ (denoted by $\alpha \approx_\Psi \beta$) if $\forall(1 \leq i \leq n)$ we have that $\alpha \approx_{\varphi_i} \beta$.*
- *$\alpha$ is preferred to $\beta$ w.r.t. $\Psi$ if $\exists(1 \leq i \leq n)$ such that* **(a)** *$\forall(1 \leq j < i)$ we have that $\alpha \approx_{\varphi_j} \beta$, and* **(b)** *$\alpha \prec_{\alpha_i} \beta$.*

*We will say that $\alpha \preceq_\Psi \beta$ if either $\alpha \prec_\Psi \beta$ or $\alpha \approx_\Psi \beta$.*

We can show that this version of $\preceq$ is a partial order (with $\approx$ as underlying equivalence).

**Proposition 2.** *Let $\Psi$ be an atomic preference; then $\preceq_\Psi$ is a partial order.*

We will say that a trajectory $\alpha$ is most preferred if there is no other trajectory that is more preferred than $\alpha$.

*Example 3.* Let us continue with the theory in Example 2. To simplify the representation, we will assume that each action is associated with a degree of safety. We will also write $bus$, $train$, or $taxi_1$, and $taxi_2$ to say that $Bob$ takes the bus, train, taxi with *PayByMeter* or *MakeIt50* company, respectively. The following is a desire expressing that *Bob* prefers to get the fastest possible way to go to school:

$$time = \textbf{always}(taxi_1 \vee taxi_2 <^e bus \vee train \vee walk)$$

On the other hand, when he is not in a hurry, *Bob* prefers to get the cheaper way to go to school: $cost = \textbf{always}(walk \vee bus \vee train <^e taxi_1 \vee taxi_2)$

These two preferences can be combined into an atomic preference

$$time \lhd cost \qquad \text{or} \qquad cost \lhd time.$$

The first one is more appropriate for *Bob* when he is in a hurry while the second one is more appropriate for *Bob* when he has time. The trajectory $\alpha = s_0\ walk\ s_1\ bus\ s_2$ is more preferred than the trajectory $\beta = s_0\ call\_taxi(PayByMeter)\ s'_1\ taxi_1\ s'_2$ with respect to the preference $cost \lhd time$, i.e., $\alpha \prec_{cost \lhd time} \beta$. However, $\beta \prec_{time \lhd cost} \alpha$.

### 3.3 General Preferences

In this section, we will define the most general case of preference formulae. A general preference is constructed from atomic preferences using the propositional connectives $\neg, \vee, \wedge, \Rightarrow$ and the ordering connective $\lhd$.

**Definition 10.** *(General Preferences) A general preference formula is a formula satisfying one of the following conditions:*

- *an atomic preference $\Psi$ is a general preference;*
- *given $\Psi_1$ and $\Psi_2$ general preferences, then $\Psi_1 \wedge \Psi_2$, $\Psi_1 \vee \Psi_2$, $\Psi_1 \Rightarrow \Psi_2$ and $\neg\Psi_1$ are general preferences;*
- *given a collection of general preferences $\Psi_1, \Psi_2, \ldots, \Psi_k$, then $\Psi_1 \lhd \Psi_2 \lhd \cdots \lhd \Psi_k$ is a general preference.*

In the next definition, we extend the definitions of $\prec$ and $\approx$ to compare trajectories with respect to a general preference.

**Definition 11 (Ordering Between Trajectories w.r.t. General Preferences).** *Let $\Psi$ a general preference and $\alpha, \beta$ two trajectories.*

- *The trajectory $\alpha$ is preferred to $\beta$ ($\alpha \prec_\Psi \beta$) if:*
  - *$\Psi$ is an atomic preference and $\alpha \prec_\Psi \beta$*
  - *$\Psi = \Psi_1 \wedge \Psi_2$, $\alpha \prec_{\Psi_1} \beta$ and $\alpha \prec_{\Psi_2} \beta$*
  - *$\Psi = \Psi_1 \vee \Psi_2$ and:* (i) *$\alpha \prec_{\Psi_1} \beta$ and $\alpha \approx_{\Psi_2} \beta$; or* (ii) *$\alpha \prec_{\Psi_2} \beta$ and $\alpha \approx_{\Psi_1} \beta$; or* (iii) *$\alpha \prec_{\Psi_1} \beta$ and $\alpha \prec_{\Psi_2} \beta$*
  - *$\Psi = \neg\Psi_1$ and $\beta \prec_{\Psi_1} \alpha$ or $\alpha \approx_{\Psi_1} \beta$*

– $\Psi = \Psi_1 \lhd \cdots \lhd \Psi_k$, *and there exists* $1 \leq i \leq k$ *such that:* (i) $\forall (1 \leq j < i)$ *we have that* $\alpha \approx_{\Psi_j} \beta$ , *and* (ii) $\alpha \prec_{\Psi_i} \beta$.

- *The trajectory* $\alpha$ *is indistinguishable from the trajectory* $\beta$ *(*$\alpha \approx_\Psi \beta$*) if:*
  – $\Psi$ *is an atomic preference and* $\alpha \approx_\Psi \beta$
  – $\Psi = \Psi_1 \wedge \Psi_2$, $\alpha \approx_{\Psi_1} \beta$, $\alpha \approx_{\Psi_2} \beta$.
  – $\Psi = \Psi_1 \vee \Psi_2$, $\alpha \approx_{\Psi_1} \beta$, *and* $\alpha \approx_{\Psi_2} \beta$.
  – $\Psi = \neg\Psi_1$ *and* $\alpha \approx_{\Psi_1} \beta$.
  – $\Psi = \Psi_1 \lhd \cdots \lhd \Psi_k$, *and for all* $1 \leq i \leq k$ *we have that* $\alpha \approx_{\Psi_i} \beta$.

**Proposition 3.** *Let* $\Psi$ *be a general preference. Then* $\approx_\Psi$ *is an equivalence relation.*

**Lemma 2.** *Let* $\Psi$ *be a general preference. If* $\alpha \prec_\Psi \beta$ *and* $\beta \approx_\Psi \gamma$ *then* $\alpha \prec_\Psi \gamma$. *Similarly, if* $\alpha \prec_\Psi \beta$ *and* $\alpha \approx_\Psi \gamma$ *then* $\gamma \prec_\Psi \gamma$.

**Proposition 4.** *Let* $\Psi$ *be a general preference. Then* $\prec_\Psi$ *is a transitive relation.*

**Corollary 1.** *The relation* $\preceq_\Psi$ *is a partial order (with* $\approx_\Psi$ *as base equivalence).*

A trajectory $\alpha$ is most preferred if there is no trajectory that is more preferred than $\alpha$.

*Example 4.* Let us continue with the theory in Example 3. Assume that the safest transportation mode is either the *train* or the expensive *MakeIt50* cab. The preference

$$safety = \textbf{always}(train \vee walk \vee taxi_2 <^e bus \vee taxi_1)$$

to say that *Bob* prefers to move around using the safest transportation mode. Further, he prefers safety over time and cost, so we write $safety \lhd (time \wedge cost)$.

# 4 Computing Preferred Trajectories

In this section, we address the problem of computing preferred trajectories. Given a planning problem $\langle D, I, G\rangle$ and a preference formula $\varphi$, we are interested in finding a preferred trajectory $\alpha$ for $\varphi$. Notice that because $\wedge, \neg, \vee$ are used in construction of preference formulae, without the lost of generality, we can assume that we only have one preference formula. We will show how this can be done in answer set programming. We achieve that by encoding each basic desire $\varphi$ as a set of rules $\Pi_\varphi$ and developing two sets of rules $\Pi_{sat}$ and $\Pi_{pref}$. $\Pi_{sat}$ checks whether a basic desire is satisfied by a trajectory. $\Pi_{pref}$ consist of rules that, when used with the **maximal** construct of **smodels** will allow us to find a most preferred trajectory with respect to a preference formula. As $\Pi(D, I, G)$ has already been discussed in Section 2 we will begin with defining $\Pi_\varphi$.

## 4.1 Encoding of Desire Formulae

The encoding of a desire formula is similar to the encoding of a fluent formula in [24]. First, each basic desire formula $\varphi$ will be associated with a unique name $n_\varphi$. The set $\Pi_\varphi$ is defined as follows.

– If $\varphi = goal(\phi)$ then $\Pi_\varphi = r_\phi \cup \{goal(n_\phi)\}$;
– If $\varphi$ is a fluent formula then $\Pi_\varphi = r_\varphi \cup \{desire(n_\varphi)\}$;
– If $\varphi = occ(a)$ then $\Pi_\varphi = \{desire(n_\varphi), happend(n_\varphi, a)\}$;

– If $\varphi = \varphi_1 \wedge \varphi_2$ then $\Pi_\varphi = \Pi_{\varphi_1} \cup \Pi_{\varphi_2} \cup \{desire(n_\varphi), and(n_\varphi, n_{\varphi_1}, n_{\varphi_2})\}$;
– If $\varphi = \varphi_1 \vee \varphi_2$ then $\Pi_\varphi = \Pi_{\varphi_1} \cup \Pi_{\varphi_2} \cup \{desire(n_\varphi), or(n_\varphi, n_{\varphi_1}, n_{\varphi_2})\}$;
– If $\varphi = \neg\phi$ then $\Pi_\varphi = \Pi_\phi \cup \{desire(n_\varphi), negation(n_\varphi, n_\phi)\}$;
– If $\varphi =$ **next**$(\phi)$ then $\Pi_\varphi = \Pi_\phi \cup \{desire(n_\varphi), next(n_\varphi, n_\phi)\}$;
– If $\varphi =$ **until**$(\varphi_1, \varphi_2)$ then $\Pi_\varphi = \Pi_{\varphi_1} \cup \Pi_{\varphi_2} \cup \{desire(n_\varphi), until(n_\varphi, n_{\varphi_1}, n_{\varphi_2})\}$;
– If $\varphi =$ **always**$(\phi)$ then $\Pi_\varphi = \Pi_\phi \cup \{desire(n_\varphi), always(n_\varphi, n_\phi)\}$;
– If $\varphi =$ **eventually**$(\phi)$ then $\Pi_\varphi = \Pi_\phi \cup \{desire(n_\varphi), eventually(n_\varphi, n_\phi)\}$.

### 4.2 $\Pi_{sat}$ – Rules for Checking of Basic Desire Formula Satisfaction

We now present the set of rules that check whether a trajectory satisfies a basic desire formula. Recall that an answer set of the program $\Pi(D, I, G)$ will contain a trajectory where action occurrences are record by atoms of the form $occ(a, t)$ and truth value of fluent literals is represented by atoms of the form $holds(f, t)$, where $a \in \mathbf{A}$, $f$ is a fluent literal, and $t$ is a time moment between 0 and $length$. $\Pi_{sat}$ defines the predicate $satisfy(F, T)$ where $F$ and $T$ are variables representing a basic desire and a time moment, respectively. Intuitively, $satisfy(F, T)$ says that the basic desire $F$ is satisfied by the trajectory starting from the time moment $T$. They are defined based on the structure of $F$. Some of the rules of $\Pi_{sat}$ are given next.

$$satisfy(F, T) \leftarrow desire(F), goal(F), satisfy(F, length). \quad (1)$$
$$satisfy(F, T) \leftarrow desire(F), happend(F, A), occ(A, T). \quad (2)$$
$$satisfy(F, T) \leftarrow desire(F), formula(F), holds(F, T). \quad (3)$$
$$satisfy(F, T) \leftarrow desire(F), and(F, F_1, F_2), satisfy(F_1, T), satisfy(F_2, T). \quad (4)$$
$$satisfy(F, T) \leftarrow desire(F), until(F, F_1, F_2), during(F_1, T, T_1), satisfy(F_2, T_1). \quad (5)$$
$$satisfy(F, T) \leftarrow desire(F), always(F, F_1), during(F_1, T, length{+}1). \quad (6)$$
$$satisfy(F, T) \leftarrow desire(F), next(F, F_1), satisfy(F_1, T + 1). \quad (7)$$
$$during(F_1, T, T_1) \leftarrow T < T_1 - 1, desire(F_1), satisfy(F_1, T), during(F_1, T + 1, T_1). \quad (8)$$
$$during(F_1, T, T_1) \leftarrow T = T_1 - 1, desire(F_1), satisfy(F_1, T). \quad (9)$$
$$satisfy(F, T) \leftarrow desire(F), negation(F, F_1), not\ satisfy(F_1, T). \quad (10)$$

In the next theorem, we prove the correctness of $\Pi_{sat}$. We need some additional notation. Let $M$ be an answer set of the program $\Pi(D, I, G)$. By $\alpha_M$ we denote the trajectory $s_0 a_0 \ldots a_{n-1} s_n$, where

- $occ(a_i, i) \in M$ for $i \in \{0, \ldots, n-1\}$ and
- $s_i = \{f \mid holds(f, i) \in M\}$ for $i \in \{0, \ldots, n\}$.

Furthermore, for a trajectory $\alpha = s_0 a_0 \ldots a_{n-1} s_n$, by $\alpha^{-1}$ we denote the set

$$\{occ(a_i, i) \mid i \in \{0, \ldots, n-1\} \cup \{holds(f, i) \mid f \in s_i, i \in \{0, \ldots, n\}.$$

We can prove the following theorem:

**Theorem 2.** *Let $\langle D, I, G\rangle$ be a planning problem and $\varphi$ be a basic desire formula. For every answer set $M$ of $\Pi(D, I, G)$, we have that $\alpha_M \models \varphi$ iff $\Pi_\varphi \cup \Pi_{sat} \cup (\alpha_M)^{-1} \models satisfy(n_\varphi, 0)$.*

This theorem allows us to compute a most preferred trajectory using **smodels**. Let $\Pi(D, I, G, \varphi)$ be the program consisting of the $\Pi(D, I, G) \cup \Pi_\varphi \cup \Pi_{sat}$ and the rule

$$\mathbf{maximize}\{satisfy(n_\varphi, 0) = 1, not\ satisfy(n_\varphi, 0) = 0\}. \quad (11)$$

Notice that rule (11) means that answer sets in which $satisfy(n_\varphi, 0)$ holds are more preferred than those in which $satisfy(n_\varphi, 0)$ does not hold. This is exactly what **smodels** does: it will first try to compute answer sets of $\Pi$ in which $satisfy(n_\varphi, 0)$; only when no answer set with this property exists, other answer sets are considered. We have the following theorem.

**Theorem 3.** *Let $\langle D, I, G\rangle$ be a planning problem and $\varphi$ be a basic desire formula. For every answer set $M$ of $\Pi(D, I, G, \varphi)$, $\alpha_M$ is a most preferred trajectory w.r.t. $\varphi$.*

The above theorem gives us a way for computing a most preferred trajectory with respect to a basic desire. We will now generalize this approach to deal with general preferences using the **maximize** function of **smodels**. The intuition is to associate to the different components of the preference formula a *weight*; these weights are then used to obtain a weight for each trajectory (based on what components of the preference formula are satisfied by the trajectory). The **maximize** function of **smodels** can be used to handle these weights and guide the search of the preferred trajectory. In general, let $\Psi$ be a general preference. We will develop a *weight function*, developed by $w_\Psi$, which maps each trajectory to a number and satisfies the following properties: **(*)** if $\alpha \prec_\Psi \beta$ then $w_\Psi(\alpha) > w_\Psi(\beta)$, and **(**)** if $\alpha \approx_\Psi \beta$ then $w_\Psi(\alpha) = w_\Psi(\beta)$.
A weight function satisfying the two properties (*)-(**) is called an *admissible* weight function. Obviously, if $w_\Psi$ is admissible, we have the following theorem.

**Proposition 5.** *Let $\Psi$ be a general preference formula. If $\alpha$ is a trajectory such that $w_\Psi(\alpha)$ is maximal, then $\alpha$ is a most preferred trajectory w.r.t. $\Psi$.*

*Proof.* Let $w_\Psi(\alpha)$ be maximal. Assume that there exists $\beta$ such that $\beta \prec_\Psi \alpha$. It follows from the admissibility of $w_\Psi$ (from (*)) that $w_\Psi(\beta) > w_\Psi(\alpha)$, which contradicts the hypothesis that $w_\Psi(\alpha)$ is maximal. □

The above theorem implies that we can compute a most preferred trajectory using **smodels** if we can implement an admissible weight function.

### 4.3 Computing An Admissible Weight Function

Let $\Psi$ be a general preference. We will now show how an admissible weight function $w_\Psi$ can be built in the bottom-up fashion. We begin with basic desires.

**Definition 12 (Basic Desire Weight).** *Let $\varphi$ be a basic desire formula and let $\alpha$ be a trajectory. The weight of the trajectory $\alpha$ w.r.t. the desire $\varphi$ is a function defined as*

$$w_\varphi(\alpha) = \begin{cases} 1 & \text{if } \alpha \models \varphi \\ 0 & \text{otherwise} \end{cases}$$

The next proposition shows that for a basic desire $\varphi$, $w_\varphi$ is admissible.

**Proposition 6.** *Let $\varphi$ be a basic desire. Then $w_\varphi$ is an admissible weight function.*

The weight function of an atomic preference is defined on the weight function of basic desires occurring in the preference as follows.

**Definition 13 (Atomic Preference Weight).** *Let $\psi = \varphi_1 \lhd \varphi_2 \lhd \cdots \lhd \varphi_k$ be an atomic preference formula. The weight of a trajectory $\alpha$ w.r.t. $\psi$ is defined as follows:*

$$w_\psi(\alpha) = \sum_{r=1}^{k} (2^{k-r} \times w_{\varphi_r}(\alpha))$$

The next proposition shows that the weight function for atomic preference is also an admissible one.

**Proposition 7.** *Let $\psi = \varphi_1 \lhd \varphi_2 \lhd \cdots \lhd \varphi_k$ be an atomic preference. Then $w_\psi$ is an admissible weight function.*

We are now ready to define an admissible weight function w.r.t. a general preference.

**Definition 14 (General Preference Weight).** *Let $\Psi$ be a general preference formula. The weight of a trajectory $\alpha$ w.r.t. $\Psi$ ($w_\Psi(\alpha)$) is defined as follows:*

- *if $\Psi$ is an atomic preference then the weight is defined as in definition 13.*
- *if $\Psi = \Psi_1 \wedge \Psi_2$ then $w_\Psi(\alpha) = w_{\Psi_1}(\alpha) + w_{\Psi_2}(\alpha)$*
- *if $\Psi = \Psi_1 \vee \Psi_2$ then $w_\Psi(\alpha) = w_{\Psi_1}(\alpha) + w_{\Psi_2}(\alpha)$*
- *if $\Psi = \neg\Psi_1$ then $w_\Psi(\alpha) = max(\Psi_1) - w_{\Psi_1}(\alpha)$ where $max(\Psi_1)$ represents the maximum weight that a trajectory can achieve on the preference formulae $\Psi_1$.*
- *if $\Psi = \Psi_1 \lhd \Psi_2$ then*

$$w_\Psi(\alpha) = \sum_{r=1}^{k} (max(\Psi_2) \times w_{\Psi_1}(\alpha) + w_{\Psi_2}(\alpha))$$

We prove the admissibility of $w_\Psi$ in the next proposition.

**Proposition 8.** *If $\Psi$ is a general preference, then $w_\Psi$ is an admissible weight function.*

Propositions 6-8 show that we can compute an admissible weight function $w_\Psi$ bottom-up from the weight of each basic desire occurring in $\Psi$. We are now ready to define the set of rules $\Pi_{pref}(\Psi)$ which consists of the rules encoding $\Psi$ and the rules encoding the computation of $w_\Psi$. Similar to the encoding of desires, we will assign a new, distinguished name $n_\phi$ to each preference formula $\phi$, which is not a desire[2], occurring in $\Psi$ and encode the preferences in the same way we encode the desires. To save space, we omit here the details of this step. $\Pi_{pref}(\Psi)$ define two predicates, $w(p, n)$ and $max(p, n)$ where $p$ is a preference name and $n$ is the weight of the current trajectory with respect to the preference $p$. $w(p, n)$ (resp. $max(p, n)$) is true if the weight (resp. maximal weight) of the current trajectory with respect to the preference $p$ is $n$.

1. For each desire $d$, $\Pi_{pref}(d)$ contains the rules

$$w(d, 1) \leftarrow satisfy(d). \tag{11}$$
$$w(d, 0) \leftarrow not\ satisfy(d). \tag{12}$$
$$max(d, 2) \leftarrow \tag{13}$$

2. For each atomic preference $\phi = \varphi_1 \lhd \varphi_2 \lhd \cdots \lhd \varphi_k$, $\Pi_{pref}(\phi)$ consists of $\cup_{j=1}^{k} \Pi_{pref}(\varphi_k)$ and the next two rules:

$$w(n_\phi, S) \leftarrow w(n_{\varphi_1}, N_1), w(n_{\varphi_2}, N_2), \ldots, w(n_{\varphi_k}, N_k), S = \Sigma_1^k 2^{k-r} N_r. \tag{14}$$
$$max(n_\phi, 2^{k+1} + 1) \leftarrow \tag{15}$$

[2] This is because we can use the names and encodings of the desires that has been developed in the previous section.

3. For each general preference $\Psi$,
   - if $\Psi$ is an atomic preference then $\Pi_{pref}(\Psi)$ is defined as in the previous item.
   - if $\Psi = \Psi_1 \wedge \Psi_2$ or $\Psi = \Psi_1 \vee \Psi_2$ then $\Pi_{pref}(\Psi)$ consists of $\Pi_{pref}(\Psi_1) \cup \Pi_{pref}(\Psi_2)$ and rules

$$w(n_\Psi, S) \leftarrow w(n_{\Psi_1}, N_1), w(n_{\Psi_2}, N_2), S = N_1 + N_2. \quad (16)$$
$$max(n_\Psi, S) \leftarrow max(n_{\Psi_1}, N_1), max(n_{\Psi_2}, N_2), S = N_1 + N_2. \quad (17)$$

   - if $\Psi = \neg\Psi_1$ then $\Pi_{pref}(\Psi)$ consists of $\Pi_{pref}(\Psi_1)$ and the rules

$$w(n_\Psi, S) \leftarrow w(n_{\Psi_1}, N), max(n_{\Psi_1}, M), S = M + 1 - N. \quad (18)$$
$$max(n_\Psi, S) \leftarrow max(n_{\Psi_1}, M), S = M + 1. \quad (19)$$

   - if $\Psi = \Psi_1 \lhd \Psi_2$ then $\Pi_{pref}(\Psi)$ consists of $\Pi_{pref}(\Psi_1) \cup \Pi_{pref}(\Psi_2)$ and rules

$$w(n_\Psi, S) \leftarrow w(n_{\Psi_1}, N_1), max(n_{\Psi_2}, M_2), w(n_{\Psi_2}, N_2), S = M_2 * N_1 + N_2. \quad (20)$$
$$max(n_\Psi, S) \leftarrow max(n_{\Psi_1}, N_1), max(n_{\Psi_2}, N_2), S = N_2 * N_1 + N_2 + 1. \quad (21)$$

The next theorem proves the correctness of $\Pi_{pref}(\Psi)$.

**Theorem 4.** *Let $\langle D, I, G\rangle$ be a planning problem and $\Psi$ be a general preference. Then, for every answer set $M$ of $\Pi(D, I, G)$, we have that $\Pi_{pref}(\Psi) \cup \Pi_{sat} \cup (\alpha_M)^{-1} \models w(n_\Psi, w)$ iff $w_\Psi(\alpha_M) = w$.*

The above theorem implies that we can compute a most preferred trajectory by (i) adding $\Pi_{pref}(\Psi) \cup \Pi_{sat}$ to $\Pi(D, I, G)$ and (ii) computing an answer set $M$ in which $w(n_\Psi, w)$ is maximal. Since we do have only one value to maximize, this can be implemented in the **smodels** system using the **maximize** command. It is worth noting that the "weak–constraint"feature of `dlv` (also implemented in **jsmodels**) can provide a more direct implementation for computing a most preferred trajectory.

### 4.4 Some Examples of Preferences in $\mathcal{PP}$

We will now present some preferences that are common to many planning problems. Let $\langle D, I, G\rangle$ be a planning problem. In keeping with the notation used in the previous section, we use $\varphi$ to denote $G$ (i.e., $\varphi = G$).

**Preference for shortest trajectory: formula based encoding.** Assume that we are interested in trajectories achieving $\varphi$ whose length is less than or equal $n$. A simple encoding that allows us to accomplish such goal is to make use of basic desires. By $\mathbf{next}^i(\varphi)$ we denote the formula:

$$\underbrace{\mathbf{next}(\mathbf{next}(\mathbf{next} \cdots (\mathbf{next}}_{i}(\varphi)) \cdots))$$

Let us define the formula $\sigma^i(\varphi)$ ($0 \leq i \leq n$) as follows:

$$\sigma^0(\varphi) = \varphi \qquad \sigma^i(\varphi) = \bigwedge_{j=0}^{i-1} \neg\mathbf{next}^j(\varphi) \ \wedge \ \mathbf{next}^i(\varphi)$$

Finally, let us consider the formula $short(n, \varphi)$ defined as

$$short(n, \varphi) = \sigma^0(\varphi) \lhd \sigma^1(\varphi) \lhd \sigma^2(\varphi) \lhd \cdots \lhd \sigma^n(\varphi)$$

**Proposition 9.** *Let $\alpha$ be a most preferred trajectory w.r.t. $short(n, \varphi)$. Then $\alpha$ is a shortest length trajectory satisfying the goal $\varphi$.*

**Preference for shortest trajectory: action based encoding.** The formula based encoding $short(n, \varphi)$ requires the bound $n$ to be given. We now present another encoding that does not require this condition. We introduce two additional fictions actions $stop$ and $noop$ and a new fluent $ended$. The action $stop$ will be triggered when the goal is achieved; $noop$ is used to fill the slot so that we can compare between trajectories; the fluent $ended$ will denote the fact that the goal is achieved. Again, we appeal to the users for the formal representation of these actions. Furthermore, we add the condition $\neg ended$ to the executability condition of any actions in $(D, I)$ and to the initial state $I$. Then we can encoded the condition of shortest length trajectory, denoted by $short$, as

$$\mathbf{always}((stop \vee noop) <^e (a_1 \vee \ldots \vee a_k))$$

where $a_1, \ldots, a_k$ are the actions in the original action theory.

**Proposition 10.** *Let $\alpha$ be a most preferred trajectory w.r.t. $short$. Then $\alpha$ is a shortest length trajectory satisfying the goal $\varphi$.*

**Cheapest Plan.** Let us assume that we would like to associate a cost $c(a)$ to each action $a$ and determine trajectories that have the minimal cost. Since our comparison is done only on equal length trajectories, we will also introduce the two actions $noop$ and $stop$ with no cost and the fluent $ended$ to record the fact that the goal has been achieved. Further, we introduce the fluent $total\_cost$ to denote the cost of the trajectory. Initially, we set the value of $total\_cost$ to 0 and the execution of action $a$ will increase the value of $total\_cost$ by $c(a)$. The preference

$$\mathbf{goal}(total\_cost(m)) \lhd \mathbf{goal}(total\_cost(m+1)) \ldots \lhd \mathbf{goal}(total\_cost(M))$$

where $m$ and $M$ are the estimated minimal and maximal cost of the trajectories, respectively. Note that we can have $m = 0$ and $M = max\{c(a) \mid a \text{ is an action}\} \times length$.

# 5 Conclusion

In this paper we presented a novel declarative language, called $\mathcal{PP}$, for the specification of preferences in the context of planning problems. The language nicely integrates with traditional action description languages (e.g., $\mathcal{B}$) and it allows elegant encoding of complex preferences between trajectories. The language provides a *declarative* framework for the encoding of preferences, allowing users to focus on the high-level description of preferences (more than their encoding—as in the approaches based on utility functions). $\mathcal{PP}$ allows the expression of complex preferences, including multi-dimensional preferences. We also demonstrated that $\mathcal{PP}$ preferences can be effectively and easily handled in a logic programming framework based on answer set semantics.

The implementation of the required cost functions in the **jsmodels** system is almost complete, and this will offer us the opportunity to validate our ideas on large test cases. We also intend to explore the possibility of introducing temporal operators at the level of general preferences. These seem to allow for very compact representation of various types of preferences; for example, a shortest plan preference can be encoded simply as:

$$\mathbf{always}((occ(stop) \vee occ(noop)) \lhd (occ(a_1) \vee \ldots \vee occ(a_k)))$$

if $a_1, \ldots, a_k$ are the possible actions.

# References


1. Y. Babovich. `CMODELS`, `www.cs.utexas.edu/users/tag/cmodels.html`.
2. F. Bacchus and F. Kabanza. Using temporal logics to express search control knowledge for planning. *Artificial Intelligence*, 116(1,2):123–191, 2000.
3. S. Bistarelli et al. Labeling and Partial Local Consistency for Soft Constraint Programming. In *Practical Aspects of Declarative Languages*, Springer Verlag, 2000.
4. A.L. Blum and M.L. Furst. Fast Planning through Planning Graph Analysis. *Artificial Intelligence*, 90:281–300, 1997.
5. G. Brewka and T. Eiter. Preferred answer sets for extended logic programs. *Artificial Intelligence*, 109:297–356, 1999.
6. A. Cimatti and M. Roveri. Conformant Planning via Symbolic Model Checking. *Journal of Artificial Intelligence Research*, 13:305–338, 2000.
7. B. Cui and T. Swift. Preference Logic Grammars: Fixed Point Semantics and Application to Data Standardization. *Artificial Intelligence*, 138(1–2):117–147, 2002.
8. J. Delgrande, T. Schaub, and H. Tompits. A framework for compiling preferences in logic programs. *Theory and Practice of Logic Programming*, 3(2):129–187, March 2003.
9. T. Eiter et al. Answer Set Planning under Action Cost. In *JELIA*. Springer Verlag, 2002.
10. T. Eiter et al. The KR System `dlv`: Progress Report, Comparisons, and Benchmarks. In *Int. Conf. on Principles of Knowledge Representation and Reasoning*, pages 406–417, 1998.
11. F. Fages, J. Fowler, and T. Sola. Handling Preferences in Constraint Logic Programming with Relational Optimization. In *PLILP* Springer Verlag, 1994.
12. M. Gelfond and V. Lifschitz. Action languages. *ETAI*, 3(6), 1998.
13. M. Gelfond et al. On the relationship between CWA, Minimal Model, and Minimal Herbrand Model semantics. *International Journal of Intelligent Systems*, 5(5):549–565, 1990.
14. K. Govindarajan, B. Jayaraman, and S. Mantha. Preference Logic Programming. In *ICLP*, pages 731–746. MIT Press, 1995.
15. P. Haddawy and S. Hanks. Utility Model for Goal-Directed Decision Theoretic Planners. Technical report, University of Washington, 1993.
16. H. Kautz and J.P. Walser. State-space Planning by Integer Optimization. In *AAAI*, pages 526–533, 1999.
17. V. Lifschitz. Answer set planning. In *International Conference on Logic Programming*, pages 23–37, 1999.
18. D. Long, M. Fox, D.E. Smith, D. McDermott, F. Bacchus, and H. Geffner. International Planning Competition.
19. K.L. Myers. Strategic Advice for Hierarchical Planners. In *Principles of Knowledge Representation and Reasoning*. Morgan Kaufmann, 1996.
20. K.L. Myers and T.J. Lee. Generating Qualitatively Different Plans through Metatheoretic Biases. In *AAAI*, 1999.
21. I. Niemelä and P. Simons. Smodels - an implementation of the stable model and well-founded semantics for normal logic programs. In *LPNMR*, Springer, pages 420–429, 1997.
22. M.L. Putterman. *Markov Decision Processes – Discrete Stochastic Dynamic Programming*. John Willey & Sons, Inc., New York, NY, 1994.
23. T. Schaub and K. Wang. A Comparative Study of Logic Programs with Preferences. In *IJCAI*, pages 597–602, 2001.
24. T.C. Son, C. Baral, and S. McIlraith. Domain dependent knowledge in planning - an answer set planning approach. In *LPNMR*, Springer, pages 226–239, 2001.
25. T.C. Son, C. Baral, T. Nam, and S. McIlraith. Domain-Dependent Knowledge in Answer Set Planning. Technical Report CS-2002-007, New Mexico State University, 2002.
26. T.C. Son and E. Pontelli. Reasoning about actions in prioritized default theory. In *JELIA*, pages 369–381. Springer Verlag, 2002.
27. L.V. Hung, E. Pontelli, T.C. Son. A Java Solver for Answer Set Programming, NMSU, 2003.

# Appendix – Proofs[3]

**Proposition 1.** *The relation $\preceq_\varphi$ defines a partial order over the trajectories.*

*Proof.* Let us proof the three properties:

1. *Reflexive:* given a trajectory $\alpha$, we have that either $\alpha \models \varphi$ or $\alpha \not\models \alpha$; this allows us to easily conclude that $\alpha \preceq_\varphi \alpha$;
2. *Anti-symmetry:* consider two trajectories $\alpha, \beta$ and let us assume that $\alpha \preceq_\varphi \beta$ and $\beta \preceq_\varphi \alpha$. First of all, we can observe that from $\alpha \preceq_\varphi \beta$ we have either $\alpha \prec \beta$ or $\alpha \approx \beta$. If $\alpha \prec \beta$ then this means that $\alpha \models \varphi$ and $\beta \not\models \varphi$. But this would automatically imply that $\beta \not\preceq \alpha$. Then we must have that $\alpha \approx \beta$.
3. *Transitivity:* consider three trajectories $\alpha_1, \alpha_2, \alpha_3$ and let us assume

$$\alpha_1 \preceq_\varphi \alpha_2 \wedge \alpha_2 \preceq_\varphi \alpha_3$$

   Let us consider the case of $\prec$. From $\alpha_1 \prec \alpha_2$ we have that $\alpha_1 \models \varphi$ and $\alpha_2 \not\models \varphi$. In turn, since $\alpha_2 \preceq \alpha_3$ then this means $\alpha_3 \not\models \varphi$; this allows us to conclude $\alpha_1 \prec \alpha_3$. If instead we have that $\alpha_1 \approx \alpha_2$, then either $\alpha_1 \models \varphi \wedge \alpha_2 \models \varphi$ or $\alpha_1 \not\models \varphi \wedge \alpha_2 \not\models \varphi$. In the first case, since $\alpha_2 \models \varphi$ we must have that $\alpha_3 \not\models \varphi$. Thus, we have $\alpha_1 \models \varphi \wedge \alpha_3 \not\models \varphi$ that implies $\alpha_1 \preceq \alpha_3$. If instead we have the second possibility, then since $\alpha_2 \not\models \varphi$, we must have $\alpha_2 \approx \alpha_3$ and $\alpha_3 \not\models \varphi$. This allows us to conclude that $\alpha_2 \approx \alpha_3$ and thus $\alpha_2 \preceq \alpha_3$.

□

**Proposition 2.** *Let $\Psi$ be an atomic preference; then $\preceq_\Psi$ is a partial order.*

*Proof.* Let us analyze the three properties.

- *Reflexivity:* Consider a trajectory $\alpha$. It is fairly easy to see that $\alpha \approx_\Psi \alpha$, which leads to $\alpha \preceq_\Psi \alpha$.
- *Antisymmetry:* Let $\alpha \preceq_\Psi \beta$ and $\beta \preceq_\Psi \alpha$. Let us assume, by contradiction, that $\alpha \prec_\Psi \beta$. This means that there is a value of $i$ such that, $\forall(1 \leq j < i)$ we have that $\alpha \approx_{\varphi_j} \beta$ and $\alpha \prec_{\varphi_i} \beta$. But this implies that $\beta \approx_{\varphi_j} \alpha$ for $j < i$ and $\beta \not\prec_{\varphi_i} \alpha$, which ultimately means $\beta \not\preceq_\Psi \alpha$, contradicting the initial assumptions.
- *Transitivity:* let $\alpha_1, \alpha_2, \alpha_3$ be trajectories such that

$$\alpha_1 \preceq_\Psi \alpha_2 \wedge \alpha_2 \preceq_\Psi \alpha_3$$

  Let us consider the possible cases arising from the first component:
  - if $\alpha_1 \approx_\Psi \alpha_2$ and $\alpha_2 \approx_\Psi \alpha_3$ then we can conclude $\alpha_1 \approx_\Psi \alpha_3$ and then $\alpha_1 \preceq_\Psi \alpha_3$.
  - if $\alpha_1 \approx_\Psi \alpha_2$ and $\alpha_2 \prec_\Psi \alpha_3$, then we know that $\forall(1 \leq j \leq n)$ then $\alpha_1 \approx_{\varphi_j} \alpha_2$, and there is $i$ such that $\forall(1 \leq k < i)$ we have that $\alpha_2 \approx_{\varphi_k} \alpha_3$ and $\alpha_2 \prec_{\varphi_i} \alpha_3$. This last formula in particular implies $\alpha_2 \models \varphi_i$ and $\alpha_3 \not\models \varphi_i$. But since $\alpha_1 \approx_{\varphi_i} \alpha_2$, then $\alpha_1 \models \varphi_i$. Putting all together we get $\forall(1 \leq k < i)(\alpha_1 \approx_{\varphi_k} \alpha_3$ and $\alpha_1 \prec_{\varphi_i} \alpha_3$. All this implies $\alpha_1 \preceq_\Psi \alpha_3$.

[3] Proofs are included to facilitate the reviewing process; they will removed from the final version of the paper, if accepted.

**Proposition 3.** *Let $\Psi$ be a general preference formula. Then $\approx_\Psi$ is an equivalence relation.*

*Proof.* Let us prove the result by structural induction on $\Psi$.
*Base:* if $\Psi$ is an atomic preference, then the result derives directly from proposition 2.
*Inductive Step:* let us consider the possible cases for $\Psi$.

Let $\Psi = \Psi_1 \wedge \Psi_2$. Reflexivity and Symmetry are obvious. If $\alpha_1 \approx_\Psi \alpha_2$ and $\alpha_2 \approx_\Psi \alpha_3$, then we have $\alpha_1 \approx_{\Psi_1} \alpha_2$, $\alpha_1 \approx_{\Psi_2} \alpha_2$, $\alpha_2 \approx_{\Psi_1} \alpha_3$, $\alpha_2 \approx_{\Psi_1} \alpha_3$. By inductive hypothesis, we $\alpha_1 \approx_{\Psi_1} \alpha_3$ and $\alpha_1 \approx_{\Psi_2} \alpha_3$, $\alpha_1 \approx_\Psi \alpha_3$.

Let $\Psi = \Psi_1 \vee \Psi_2$. The proof is the same above.

Let $\Psi = \neg\Psi_1$. Reflexivity and symmetry is obvious. Let us now consider $\alpha_1 \approx_\Psi \alpha_2$ and $\alpha_2 \approx_\Psi \alpha_3$. From the definition we have that $\alpha_1 \approx_{\Psi_1} \alpha_2$ and $\alpha_2 \approx_{\Psi_1} \alpha_3$. From the inductive hypothesis we have $\alpha_1 \approx_{\Psi_1} \alpha_3$ and thus $\alpha_1 \approx_\Psi \alpha_3$.

Let $\Psi = \Psi_1 \lhd \cdots \lhd \Psi_k$. Reflexivity and symmetry are obvious from the inductive hypothesis. Let us consider $\alpha_1 \approx_\Psi \alpha_2$ and $\alpha_2 \approx_\Psi \alpha_3$. From the definition this means that $\forall(1 \leq i \leq k)$ we have that $\alpha_1 \approx_{\Psi_i} \alpha_2$ and $\alpha_2 \approx_{\Psi_i} \alpha_3$. From the inductive hypothesis we get $\alpha_1 \approx_{\Psi_i} \alpha_3$. This allows us to conclude $\alpha_1 \approx_\Psi \alpha_3$. □

**Lemma 3.** *Let $\Psi$ be a general preference formula. If $\alpha \prec_\Psi \beta$ and $\beta \approx_\Psi \gamma$ then $\alpha \prec_\Psi \gamma$.*

*Similarly, if $\alpha \prec_\Psi \beta$ and $\alpha \approx_\Psi \gamma$ then $\gamma \prec_\Psi \gamma$.*

*Proof.* let us prove the result by structural induction on $\Psi$.

If $\Psi$ is an atomic preference, and $\Psi = \varphi_1 \lhd \cdots \lhd \varphi_k$ then from $\alpha \prec_\Psi \beta$ we obtain that $\exists(1 \leq i \leq k)$ such that

- $\forall(1 \leq j < i)(\alpha \approx_{\varphi_j} \beta)$
- $\alpha \prec_{\varphi_i} \beta$

Furthermore, from $\beta \approx_\Psi \gamma$ we obtain that $\forall(1 \leq j \leq k)(\beta \approx_{\varphi_j} \gamma)$. Since $\approx_{\varphi_j}$ are all equivalence relations, we obtain that $\forall(1 \leq j < i)(\alpha \approx_{\varphi_j} \gamma$; furthermore, since $\alpha \prec_{\varphi_i} \beta$, then $\alpha \models \varphi_i$ and $\beta \not\models \varphi_i$. Since $\beta \approx_{\varphi_i} \gamma$ then necessarily we have also that $\gamma \not\models \varphi_i$. This allows us to conclude that $\alpha \prec_\Psi \gamma$.

The case where $\alpha \approx_\Psi \gamma$ is analogous.

Let us consider the case of $\Psi = \Psi_1 \wedge \Psi_2$. Since $\alpha \prec_\Psi \beta$ then we have $\alpha \prec_{\Psi_1} \beta$ and $\alpha \prec_{\Psi_2} \beta$. Furthermore, $\beta \approx_\Psi \gamma$ implies $\beta \approx_{\Psi_1} \gamma$ and $\beta \approx_{\Psi_2} \gamma$. From the inductive hypothesis we have $\alpha \prec_{\Psi_1} \gamma$ and $\alpha \prec_{\Psi_2} \gamma$, which leads to $\alpha \prec_\Psi \gamma$.

The case where $\alpha \approx_\Psi \gamma$ is analogous.

Let us consider the case of $\Psi = \Psi_1 \vee \Psi_2$. From $\alpha \prec_\Psi \beta$ we have three possible cases:

1. $\alpha \prec_{\Psi_1} \beta$ and $\alpha \prec_{\Psi_2} \beta$. In this case, since $\beta \approx_\Psi \gamma$ implies $\beta \approx_{\Psi_1} \gamma$ and $\beta \approx_{\Psi_2} \gamma$, then we have $\alpha \prec_{\Psi_1} \gamma$ and $\alpha \prec_{\Psi_2} \gamma$. This implies $\alpha \prec_\Psi \gamma$.
2. $\alpha \prec_{\Psi_1} \beta$ and $\alpha \approx_{\Psi_2} \beta$. From $\beta \approx_\Psi \gamma$ we obtain $\beta \approx_{\Psi_1} \gamma$ and $\beta \approx_{\Psi_2} \gamma$. Since $\approx_{\Psi_2}$ is an equivalence relation, we can infer $\alpha \approx_{\Psi_2} \gamma$. Furthermore, from the inductive hypothesis we obtain $\alpha \prec_{\Psi_1} \gamma$. These two conclusions lead to $\alpha \prec_\Psi \gamma$.
3. $\alpha \prec_{\Psi_2} \beta$ and $\alpha \approx_{\Psi_1} \beta$. This case is symmetrical to the previous one.

The case where $\alpha \approx_{\Psi} \gamma$ is analogous.

Let us consider the case where $\Psi = \neg\Psi_i$. From $\alpha \prec_{\Psi} \beta$ we obtain $\beta \prec_{\Psi_1} \alpha$. Since $\beta \approx_{\Psi} \gamma$ implies $\beta \approx_{\Psi_1} \gamma$, from inductive hypothesis we can conclude $\gamma \prec_{\Psi_1} \alpha$ and ultimately $\alpha \prec_{\Psi} \gamma$.

The case where $\alpha \approx_{\Psi} \gamma$ is analogous.

Let us consider the case of $\Psi = \Psi_1 \lhd \cdots \lhd \Psi_k$. ¿From the definition of $\alpha \prec_{\Psi} \beta$ we obtain that there exists $1 \leq i \leq k$ such that

- $\forall(1 \leq j < i)(\alpha \approx_{\Psi_j} \beta)$
- $\alpha \prec_{\Psi_i} \beta$

Furthermore, from $\beta \approx_{\Psi} \gamma$ we obtain that $\forall(1 \leq j \leq k)(\beta \approx_{\Psi_j} \gamma)$. Since $\approx_{\Psi_j}$ are all equivalence relations, then we have that $\forall(1 \leq j < i)(\alpha \approx_{\Psi_j} \gamma)$. Furthermore, from the inductive hypothesis, from $\alpha \prec_{\Psi_i} \beta$ and $\beta \approx_{\Psi_i} \gamma$ we can conclude $\alpha \prec_{\Psi_i} \gamma$. This finally leads to $\alpha \prec_{\Psi} \gamma$.

The case where $\alpha \approx_{\Psi} \gamma$ is analogous. □

**Proposition 4.** *Let $\Psi$ be a general preference formula. Then $\prec_{\Psi}$ is a transitive relation.*

*Proof:* Let us prove the result by structural induction on $\Psi$.

The base case corresponds to $\Psi$ being an atomic preference. We have already proved in proposition 2 that $\prec_{\Psi}$ is transitive for any atomic preference $\Psi$.

Let us now consider the other possible cases for $\Psi$.

- let $\Psi = \Psi_1 \wedge \Psi_2$ and let $\alpha_1 \prec_{\Psi} \alpha_2$ and $\alpha_2 \prec_{\Psi} \alpha_3$.
  From the definition we have that $\alpha_1 \prec_{\Psi_1} \alpha_2$, $\alpha_1 \prec_{\Psi_2} \alpha_2$, $\alpha_2 \prec_{\Psi_2} \alpha_3$, and $\alpha_2 \prec_{\Psi_2} \alpha_3$.
  From the inductive hypothesis we have $\alpha_1 \prec \Psi_1\alpha_3$ and $\alpha_1 \prec \Psi_2\alpha_3$. This implies $\alpha_1 \prec \Psi\alpha_3$.
- let $\Psi = \Psi_1 \vee \Psi_2$ and let $\alpha_1 \prec_{\Psi} \alpha_2$ and $\alpha_2 \prec_{\Psi} \alpha_3$.
  Let us consider the possible cases that can arise:
  1. $\alpha_1 \prec_{\Psi_1} \alpha_2$ and $\alpha_1 \prec_{\Psi_2} \alpha_2$.
     If we have that $\alpha_2 \prec_{\Psi_1} \alpha_3$ and $\alpha_2 \prec_{\Psi_2} \alpha_3$, then by inductive hypothesis we obtain $\alpha_1 \prec_{\Psi_1} \alpha_3$ and $\alpha_1 \prec_{\Psi_2} \alpha_3$. This leads to $\alpha_1 \prec_{\Psi} \alpha_3$.
     If we have that $\alpha_2 \prec_{\Psi_1} \alpha_3$ and $\alpha_3 \approx_{\Psi_2} \alpha_3$, then from inductive hypothesis we get $\alpha_1 \prec_{\Psi_1} \alpha_3$ while from lemma 3 we obtain $\alpha_1 \prec_{\Psi_2} \alpha_3$. This leads to $\alpha_1 \prec_{\Psi} \alpha_3$.
     If we have that $\alpha_2 \prec_{\Psi_2} \alpha_3$ and $\alpha_3 \approx_{\Psi_1} \alpha_3$, then from inductive hypothesis we get $\alpha_1 \prec_{\Psi_2} \alpha_3$ while from lemma 3 we obtain $\alpha_1 \prec_{\Psi_1} \alpha_3$. This leads to $\alpha_1 \prec_{\Psi} \alpha_3$.
  2. $\alpha_1 \prec_{\Psi_1} \alpha_2$ and $\alpha_1 \approx_{\Psi_2} \alpha_2$.
     If we have that $\alpha_2 \prec_{\Psi_1} \alpha_3$ and $\alpha_2 \prec_{\Psi_2} \alpha_3$, then from inductive hypothesis and from lemma 3 we obtain $\alpha_1 \prec_{\Psi_1} \alpha_3$ and $\alpha_1 \prec_{\Psi_2} \alpha_3$.
     If we have that $\alpha_2 \prec_{\Psi_1} \alpha_3$ and $\alpha_2 \approx_{\Psi_2} \alpha_3$, then from inductive hypothesis we obtain $\alpha_1 \prec_{\Psi_1} \alpha_3$, while from the properties of $\approx_{\Psi_2}$ we obtain $\alpha_1 \approx_{\Psi_2} \alpha_3$.
     If we have that $\alpha_2 \prec_{\Psi_2} \alpha_3$ and $\alpha_2 \approx_{\Psi_1} \alpha_3$, then from inductive hypothesis we obtain $\alpha_1 \prec_{\Psi_2} \alpha_3$, while from the properties of $\approx_{\Psi_1}$ we obtain $\alpha_1 \approx_{\Psi_1} \alpha_3$.
     In all the three cases we can conclude $\alpha_1 \prec_{\Psi} \alpha_3$.

3. the case where $\alpha_1 \prec_{\Psi_2} \alpha_2$ and $\alpha_1 \approx_{\Psi_1} \alpha_2$ is analogous to the previous one.

– Let $\Psi = \neg\Psi_1$ and let $\alpha_1 \prec_\Psi \alpha_2$ and $\alpha_2 \prec_\Psi \alpha_3$.
From the definition we have $\alpha_2 \prec_{\Psi_1} \alpha_1$ and $\alpha_3 \prec_{\Psi_1} \alpha_2$. From the inductive hypothesis we obtain $\alpha_3 \prec_{\Psi_1} \alpha_1$ which implies $\alpha_1 \prec_\Psi \alpha_3$.

– Let $\Psi = \Psi_1 \lhd \cdots \lhd \Psi_k$ and let $\alpha_1 \prec_\Psi \alpha_2$ and $\alpha_2 \prec_\Psi \alpha_3$.
From $\alpha_1 \prec_\Psi \alpha_2$ we obtain that there exists $1 \leq i \leq k$ such that
  - $\forall(1 \leq j < i)(\alpha_1 \approx_{\Psi_j} \alpha_2)$
  - $\alpha_1 \prec_{\Psi_i} \alpha_2$

  Similarly we can find a $1 \leq i' \leq k$ with similar properties for $\alpha_2 \prec_\Psi \alpha_3$.
  If $i < i'$ then from the fact that $\approx_{\Psi_j}$ are equivalence relations, we can conclude that $\forall(1 \leq j < i)(\alpha_1 \approx_{\Psi_j} \alpha_3)$. Furthermore, from lemma 3 and from $\alpha_1 \prec_{\Psi_i} \alpha_2$ and $\alpha_2 \approx_{\Psi_i} \alpha_3$, we can conclude $\alpha_1 \prec_{\Psi_i} \alpha_3$. This leads to $\alpha_1 \prec_\Psi \alpha_3$.
  If $i > i'$, we have that
  - $\forall(1 \leq j < i')(\alpha_1 \approx_{\Psi_j} \alpha_3)$ (since $\approx_{\Psi_j}$ are equivalence relations)
  - since $\alpha_1 \approx_{\Psi_{i'}} \alpha_2$ and $\alpha_2 \prec_{\Psi_{i'}} \alpha_3$, from lemma 3 we obtain $\alpha_1 \prec_{\Psi_{i'}} \alpha_3$.

  This leads to $\alpha_1 \prec_\Psi \alpha_3$.
  Finally, if $i = i'$, then
  - $\forall(1 \leq j < i)(\alpha_1 \approx_{\Psi_j} \alpha_3)$
  - since $\alpha_1 \prec_{\Psi_i} \alpha_2$ and $\alpha_2 \prec_{\Psi_i} \alpha_3$, from the inductive hypothesis we obtain $\alpha_1 \prec_{\Psi_i} \alpha_3$.

  This also leads to $\alpha_1 \prec_\Psi \alpha_3$.

□

**Proposition 5.** *Let $\psi = \varphi_1 \lhd \varphi_2 \lhd \cdots \lhd \varphi_k$ be an atomic preference formula. Then $w_\psi$ is an admissible weight function.*

*Proof.* Let $\alpha_1, \alpha_2$ be two trajectories. Let us start by assuming $\alpha_1 \prec_\psi \alpha_2$. According to the definition, this means that $\exists(1 \leq i \leq k)$ such that

– $\forall(1 \leq j < i)(\alpha_1 \approx_{\varphi_j} \alpha_2)$
– $\alpha_1 \prec_{\varphi_i} \alpha_2$

¿From Proposition 6 we have that $\alpha_1 \approx_{\varphi_j} \alpha_2$ implies $w_{\varphi_j}(\alpha_1) = w_{\varphi_j}(\alpha_2)$. This leads to

$$\sum_{r=1}^{i-1}(2^{k-r} \times w_{\varphi_r}(\alpha_1)) = \sum_{r=1}^{i-1}(2^{k-r} \times w_{\varphi_r}(\alpha_2))$$

In addition, since $\alpha_1 \prec_{\varphi_i} \alpha_2$, then we also have that $1 = w_{\varphi_i}(\alpha_1) > w_{\varphi_i}(\alpha_2) = 0$. Thus, we have

$$\begin{array}{ll}
\sum_{r=1}^{k}(2^{k-r} \times w_{\varphi_r}(\alpha_1)) & = \\
\sum_{r=1}^{i-1}(2^{k-r} \times w_{\varphi_r}(\alpha_1)) + 2^{k-i} + \sum_{r=i+1}^{k}(2^{k-r} \times w_{\varphi_r}(\alpha_1)) & > \\
\sum_{r=1}^{i-1}(2^{k-r} \times w_{\varphi_r}(\alpha_1)) + 2^{k-i} - 1 & \geq \\
\sum_{r=1}^{i-1}(2^{k-r} \times w_{\varphi_r}(\alpha_1)) + \sum_{r=i+1}^{k}(2^{k-r} \times w_{\varphi_r}(\alpha_2)) & = \\
\sum_{r=1}^{k}(2^{k-r} \times w_{\varphi_r}(\alpha_2)) &
\end{array}$$

For similar reasons, it is also easy to see that $\alpha_1 \approx_\psi \alpha_2$ implies $w_\psi(\alpha_1) = w_\psi(\alpha_2)$. □

**Proposition 6.** *Let $\Psi$ be a general preference formula. Then $w_\Psi$ is an admissible weight function.*

*Proof.* Let us prove the result by structural induction on $\Psi$. Let $\alpha, \beta$ be two arbitrary trajectories.

Let us consider the case where $\Psi$ is an atomic preference. The result has been proved in proposition 7.

Let us consider the case where $\Psi = \Psi_1 \wedge \Psi_2$. If $\alpha \prec_\Psi \beta$ then this implies $\alpha \prec_{\Psi_1} \beta$ and $\alpha \prec_{\Psi_2} \beta$. ¿From the inductive hypothesis this leads to

$$w_{\Psi_1}(\alpha) > w_{\Psi_1}(\beta) \text{ and } w_{\Psi_2}(\alpha) > w_{\Psi_2}(\beta)$$

But this clearly implies

$$w_\Psi(\alpha) = w_{\Psi_1}(\alpha) + w_{\Psi_2}(\alpha) > w_{\Psi_1}(\beta) + w_{\Psi_2}(\beta) = w_\Psi(\beta)$$

The situation is analogous if we consider the case of $\alpha \approx_\Psi \beta$.

Let us consider the case where $\Psi = \Psi_1 \vee \Psi_2$. From $\alpha \prec_\Psi \beta$ we have three possible cases:

1. $\alpha \prec_{\Psi_1} \beta$ and $\alpha \prec_{\Psi_2} \beta$; in this case, from the inductive hypothesis we can obtain $w_{\Psi_1}(\alpha) > w_{\Psi_1}(\beta)$ and $w_{\Psi_2}(\alpha) > w_{\Psi_2}(\beta)$. This allows us to conclude

$$w_\Psi(\alpha) = w_{\Psi_1}(\alpha) + w_{\Psi_2}(\alpha) > w_{\Psi_1}(\beta) + w_{\Psi_2}(\beta) = w_\Psi(\beta)$$

2. $\alpha \prec_{\Psi_1} \beta$ and $\alpha \approx_{\Psi_2} \beta$. ¿From the inductive hypothesis we obtain

$$w_{\Psi_1}(\alpha) > w_{\Psi_1}(\beta)$$

Also, from the inductive hypothesis we obtain

$$w_{\Psi_2}(\alpha) = w_{\Psi_2}(\beta)$$

This leads to

$$w_\Psi(\alpha) = w_{\Psi_1}(\alpha) + w_{\Psi_2}(\alpha) > w_{\Psi_1}(\beta) + w_{\Psi_2}(\beta) = w_\Psi(\beta)$$

3. $\alpha \prec_{\Psi_2} \beta$ and $\alpha \approx_{\Psi_1} \beta$. This case is analogous to the previous one.

If we start from $\alpha \approx_\Psi \beta$, then from the definition we have $\alpha \approx_{\Psi_1} \beta$ and $\alpha \approx_{\Psi_2} \beta$. From the inductive hypothesis this means

$$w_{\Psi_1}(\alpha) = w_{\Psi_1}(\beta) \textit{ and } w_{\Psi_2}(\alpha) = w_{\Psi_2}(\beta)$$

which clearly implies that

$$w_\Psi(\alpha) = w_{\Psi_1}(\alpha) + w_{\Psi_2}(\alpha) = w_{\Psi_1}(\beta) + w_{\Psi_2}(\beta) = w_\Psi(\beta)$$

Let $\Psi = \neg\Psi_1$. If $\alpha \prec_\Psi \beta$, then this means $\beta \prec_{\Psi_1} \alpha$; from the inductive hypothesis this leads to $w_{\Psi_1}(\beta) > w_{\Psi_1}(\alpha)$. From this we can obtain that

$$w_\Psi(\alpha) = max(\Psi_1) - w_{\Psi_1}(\alpha) > max(\Psi_1) - w_{\Psi_1}(\beta) = w_\Psi(\beta)$$

If we have $\alpha \approx_\Psi \beta$, then from the definition this means $\alpha \approx_{\Psi_1} \beta$. From the inductive hypothesis we have $w_{\Psi_1}(\alpha) = w_{\Psi_1}(\beta)$ and thus $w_\Psi(\alpha) = w_\Psi(\beta)$.

Let us consider the case $\Psi = \Psi_1 \lhd \Psi_2$ and let us assume $\alpha \prec_\Psi \beta$. There are two possible cases that can occur:

1. $\alpha \approx_{\Psi_1} \beta$ and $\alpha \prec_{\Psi_2} \beta$: in this case by inductive hypothesis we obtain $w_{\Psi_1}(\alpha) = w_{\Psi_1}(\beta)$ and $w_{\Psi_2}(\alpha) < w_{\Psi_2}(\beta)$. This leads to

   $$w_\Psi(\alpha) = max(\Psi_2) \times w_{\Psi_1}(\alpha) + w_{\Psi_2}(\alpha) => max(\Psi_2) \times w_{\Psi_1}(\beta) + w_{\Psi_2}(\beta) = w_\Psi(\beta)$$

2. $\alpha \prec_{\Psi_1} \beta$; in this case from the inductive hypothesis we obtain $w_{\Psi_1}(\alpha) < w_{\Psi_1}(\beta)$. Furthermore, we clearly have that $max(\Psi_2) \geq w_{\Psi_2}(\beta)$. Thus

   $$\begin{array}{ll} w_\Psi(\alpha) = max(\Psi_2) \times w_{\Psi_1}(\alpha) + w_{\Psi_2}(\alpha) & \geq \\ max(\Psi_2) \times (w_{\Psi_1}(\beta) + 1) + w_{\Psi_2}(\alpha) & = \\ max(\Psi_2) \times w_{\Psi_1}(\beta) + (max(\Psi_2) + w_{\Psi_2}(\alpha)) & > \\ max(\Psi_2) \times w_{\Psi_1}(\beta) + w_{\Psi_2}(\beta) & = \\ w_\Psi(\beta) & \end{array}$$

Let us now consider the case $\alpha \approx_\Psi \beta$. From the definitions this means that $\alpha \approx_{\Psi_1} \beta$ and $\alpha \approx_{\Psi_2} \beta$. From the inductive hypothesis we have $w_{\Psi_1}(\alpha) = w_{\Psi_1}(\beta)$ and $w_{\Psi_2}(\alpha) = w_{\Psi_2}(\beta)$. From the definition of the cost we clearly have $w_\Psi(\alpha) = w_\Psi(\beta)$. $\square$