DELPHIC: Practical DEL Planning via Possibilities (Supplementary Material)

Alessandro Burigana[1][0000−0002−9977−6735], Paolo Felli[2][0000−0001−9561−8775], and Marco Montali[1][0000−0002−8021−3430]

[1] Free University of Bozen-Bolzano, Italy {burigana, montali}@inf.unibz.it
[2] University of Bologna, Italy paolo.felli@unibo.it

A Full Proofs

Lemma 1. *Let $(\mathcal{E}, E_d)$ be an MPEM applicable in the MPKM (M, W_d), with solutions E and W, respectively. Then the possibility spectrum $\mathsf{W}' = \mathsf{W} \bowtie \mathsf{E}$ is the solution of $(M', W'_d) = (M, W_d) \otimes (\mathcal{E}, E_d)$.*

Proof. *Let $M = (W, R, V)$, $\mathcal{E} = (E, Q, pre, post)$ and $M' = (W', R', V')$. Let δ_M and $\delta_{\mathcal{E}}$ be the decorations for M and $\mathcal{E}$, respectively. Let then $(\hat{M}, \hat{W}_d)$ be the picture of W' via the decoration δ, where $\hat{M} = (\hat{W}, \hat{R}, \hat{V})$. By Proposition 1, to prove that W' is the solution of (M', W'_d), we need to show that $(M', W'_d) \underline{\leftrightarrow} (\hat{M}, \hat{W}_d)$. Let $B \subseteq W' \times \hat{W}$ be a relation such that:*

$$(w', \hat{w}) \in B \Leftrightarrow w' = (w, e) \wedge \delta(\hat{w}) = \delta_M(w) \bowtie \delta_{\mathcal{E}}(e).$$

We now show that B is a bisimulation between M' and $\hat{M}$. Let $(w', \hat{w}) \in B$, with $w' = (w, e)$ and let $v' = (v, f) \in W'$. Let $\mathsf{w} = \delta_M(w)$, $\mathsf{e} = \delta_{\mathcal{E}}(e)$, $\mathsf{v} = \delta_M(v)$ and $\mathsf{f} = \delta_{\mathcal{E}}(f)$. Finally, let $\mathsf{w}' = \mathsf{w} \bowtie \mathsf{e} = \delta(\hat{w})$ and $\mathsf{v}' = \mathsf{v} \bowtie \mathsf{f}$.

- *(Atom) Let $p \in \mathcal{P}$ be a propositional atom. Then:*

$$\begin{aligned} w' \in V'(p) &\overset{\text{Def. 6}}{\Leftrightarrow} (M, w) \models post(e)(p) \\ &\overset{\text{Pr. 2, Def. 17}}{\Leftrightarrow} \mathsf{w} \models \mathsf{e}(p) \\ &\overset{\text{Def. 19}}{\Leftrightarrow} \mathsf{w}'(p) = 1 \\ &\overset{\text{Def. 14}}{\Leftrightarrow} \hat{w} \in \hat{V}(p) \end{aligned}$$

- *(Forth/Back) Let $i \in \mathcal{AG}$ be an agent. Then:*

$$\begin{aligned} w' R'_i v' &\overset{\text{Def. 6}}{\Leftrightarrow} wR_iv, eQ_if, (M, w) \models pre(e) \text{ and} \\ &\quad (M, v) \models pre(f) \\ &\overset{\text{Pr. 2, Def. 17}}{\Leftrightarrow} \mathsf{v} \in \mathsf{w}(i), \mathsf{f} \in \mathsf{e}(i), \mathsf{w} \models \mathsf{e}(pre) \text{ and} \\ &\quad \mathsf{v} \models \mathsf{f}(pre) \\ &\overset{\text{Def. 19}}{\Leftrightarrow} \mathsf{v}' \in \mathsf{w}'(i) \\ &\overset{\text{Def. 14}}{\Leftrightarrow} \hat{w} \hat{R}_i \hat{v} \end{aligned}$$

- *(Designated) Let $(w'_d, \hat{w}_d) \in B$, with $w'_d = (w, e)$. Then:*

$$w'_d \in W'_d \overset{\text{Def. 6}}{\Leftrightarrow} w \in W_d, e \in E_d \text{ and } (M, w) \models pre(e)$$
$$\overset{\text{Pr. 2, Def. 17}}{\Leftrightarrow} \mathsf{w} \in \mathsf{W}, \mathsf{e} \in \mathsf{E} \text{ and } \mathsf{w} \models \mathsf{e}(pre)$$
$$\overset{\text{Def. 19}}{\Leftrightarrow} \mathsf{w}' \in \mathsf{W}'$$
$$\overset{\text{Def. 14}}{\Leftrightarrow} \hat{w}_d \in \hat{W}_d$$

B ASP Encodings

In this section, we describes the Answer Set Programming (ASP) encodings of DELPHIC and of the traditional Kripke semantics for DEL. We assume that the reader is familiar with the basics concepts of ASP. Notably, our encodings make use of the *multi-shot* solving strategy provided by the ASP-solver *clingo*, which provides fine-grained control over grounding and solving of ASP programs, and is instrumental to implementing an incremental strategy for solving. For a detailed introduction on multi-shot ASP solving, we refer the reader to [3,4].

We developed our ASP encodings in such a way that DELPHIC objects (*e.g.*, possibility/eventuality spectrums) and DEL objects (*e.g.*, MPKMs/MPEMs) are encoded by closely mirroring each other, and only differing in the two update operators (*i.e.*, union update and product update). This homogeneity is instrumental to obtain a fair experimental comparison of the two encodings, presented in Section 4 of the main paper. At the same time, as detailed later, we stress that the timings exhibited by our ASP encoding of the Kripke semantics are comparable to those shown by the state-of-the-art solver EFP 2.0 [2], which also implements on the Kripke semantics. Due to the similarity of the two encodings, in the following we use generic terms such as "epistemic state/action" and "planning task" to abstract away from which underlying semantics is actually chosen.

The DELPHIC encoding presented in this section constitutes a contribution of this paper of independent interest. In fact, it generalizes the ASP-based epistemic planner PLATO [1], which implements a possibility-based semantics for a fragment of DEL [2]. Moreover, having an ASP-based solver for epistemic planning has several benefits. First, the declarative encoding of the DELPHIC semantics allows for an implementation that is transparent and easier to inspect. Second, we exploited the Python API offered by *clingo* to implement a graphical representation that shows the epistemic states visited by the planner. We thus obtain a practical and useful tool that allows to visualize the evolution of the system. This feature is instrumental in different tasks, such as designing new epistemic planning domains and debugging the correctness of implementations[3]. We concretely show a graphical comparison of the output of the two encodings in the appendix on a concrete domain instance (we do not directly report it here due to space reasons).

The remainder of the section is as follows. First, we briefly describe how *incremental solving* is achieved by the multi-shot solving technique (Section B.1).

[3] The full code and documentation of the ASP encodings are available at https://github.com/a-burigana/delphic_asp.

Then, we illustrate both encodings component by component: *(i)* formulae in Section B.2, *(ii)* planning tasks in Section B.3, *(iii)* epistemic states in Section B.4, *(iv)* truth conditions in Section B.5, and *(v)* update operators in Section B.6).

B.1 Multi-shot Encoding

The multi-shot approach allows one to divide an ASP encoding into sub-programs, then handling grounding and solving of these sub-programs separately. In particular, this technique is useful to implement *incremental solving*, which, at each time step, allows to extend the ASP program in order to look for solutions of increasing size. Intuitively, every step mimics a Breadth-First Search over the planning state space: at each time step `t`, if a solution is not found (*i.e.*, there is no plan of length `t` that satisfies the goal), the ASP program is expanded to look for a longer plan.

To achieve incremental solving, we build on the approach by Gebser et al. [3], splitting our encodings in three subprograms: 1. program `base`, which contains all the static information (*i.e.*, input information on the planning task); 2. program `step(t)`, where `t`>0, which describes the evolution of the system (*i.e.*, updates semantics); and 3. program `check(t)`, where `t`≥ 0, which verify the truth of formulae of the domain and, in particular, of the goal formula. Here, `t` represents the current time step that is being considered. In the reminder of this section, when we describe components of the encodings pertaining to the sub-programs `step(t)` and `check(t)`, we assume `t` fixed.

B.2 Formulae

We represent epistemic formulae through nested ASP predicates. To enhance the performance of grounding and solving, we assume that all input formulae are given in a normal form where occurrences of the $\Box_i$ operators are replaced by the dual representation $\neg\Diamond_i\neg$, and where double negation is simplified. Agents and atoms are represented by the ASP predicates `agent(AG)` and `atom(P)`, respectively. A formula φ is encoded inductively on its structure: 1. $\varphi = p$ is encoded by `p`; 2. $\varphi = \neg\psi$ is encoded by `neg(PSI)`; 3. $\varphi = \psi_1 \wedge \psi_2$ is encoded by `and(PSI`$_1$`,PSI`$_2$`)`; and 4. $\varphi = \Diamond_i\psi$ is encoded by `dia(i,PSI)`.

B.3 Planning Tasks

We now describe our ASP encoding of a planning task.
Initial State. The initial state is given by the following ASP predicates: 1. `w_init(W)`: `W` is an initial possibility/possible world; 2. `r_init(W`$_1$`,W`$_2$`,AG)`: in `W`$_1$, agent `AG` considers `W`$_2$ to be possible; 3. `v_init(W,P)`: atom `P` is true in `W`; 4. `dw_init(W)`: `W` is a designated possibility/world.
Actions. To obtain efficient encodings of the two semantics, and directly support existing benchmarks from the literature, we introduce two features in the

definition of actions, namely *(global) action preconditions* and *observability conditions*.

An action precondition is specified with the ASP predicate `action_pre(ACT, PRE)` and represents the applicability of the action as a whole. This is syntactic sugar: action preconditions do not modify the expressiveness of epistemic actions, and one can always get an equivalent epistemic action that does not employ them.

Observability conditions provide a useful way to compactly represent epistemic actions. Namely, agents are split into *observability groups* classifying different perspectives of agents w.r.t. an action. For instance, in Example 2, agent a is *fully observant*, since it is the one that performs the action, whereas agent b is oblivious, since it does not know that the action is taking place. Then, as we show below, the information regarding which eventuality/event are considered to be possible is lifted to observability groups. Each action must then specify the observability conditions for each agent, assigning each agent to an observability group. This strategy yields an much more succinct representation since each action needs to be substantiated for each group combination, rather than for each agent combination.

We are now ready to describe the ASP encoding of an action `ACT`: 1. `e(ACT, E)`: `E` is an eventuality/event of `ACT`; 2. `q(ACT, E1, E2, GR)`: in `E1`, the observability group `GR` considers `E2` to be possible; 3. `obs(ACT, AG, GR, COND)`: agent `AG` is in the observability group `GR` if the condition `COND` is satisfied by the current epistemic state; 4. `pre(ACT, E, PRE)`: the precondition of `E` is `PRE`; 5. `post(ACT, E, P, POST)`: the postcondition of `P` in `E` is `POST`; 6. `de(ACT, E)`: `E` is a designated eventuality/event of `ACT`.

We also define some auxiliary ASP predicates that will be used in the update encodings (Section B.6), namely `idle(ACT, E)` and `inertia(ACT, E, P)`, which are calculated at the beginning of the ASP computation. The former predicate states that `E` does not affect the worlds in any way (*e.g.*, e_2 in Example 2 is idle). The latter predicate states that atom `P` is not changed by the postconditions of `E`.

Goal. The goal of a planning task is represented by the ASP `goal(F)` predicates. It is possible to declare multiple goal formulae, so that the goal condition of the planning task is the conjunction of all these `goal` predicates.

B.4 Epistemic States

The components of an epistemic state that must be represented (in DELPHIC as well as in the traditional DEL semantics) are four: *(i)* possibilities/possible worlds; *(ii)* information states/accessibility relations; *(iii)* valuation of propositional atoms; *(iv)* designated possibilities/worlds.

Possibilities and Possible Worlds. To describe possibilities and possible worlds, we make use of the ASP predicate `w(t, W, E)`. A possibility/world needs three variable to be univocally identified:

- `t`: represents the time instant when the possibility/world was created. In the initial state, the time is set to `0`;
- `W`: represents the possibility/world that is being updated;
- `E`: represents the eventuality/event that is updating `W`.

As at planning time new possibilities/worlds are created dynamically, we are faced with the challenge of finding a suitable ASP representation that correctly and univocally encodes the worlds that are being updated during each action. This is best explained with an example. Suppose we update the possibility/world $\mathtt{w}(\mathtt{t}-1,\mathtt{W},\mathtt{E})$ with the eventuality/event $\mathtt{e}(\mathtt{ACT},\mathtt{F})$ and let $\mathtt{w}(\mathtt{t},\mathtt{X},\mathtt{F})$ be the result of the update. Intuitively, we can see $\mathtt{X}$ as representing the world being updated (*i.e.*, $\mathtt{w}(\mathtt{t}-1,\mathtt{W},\mathtt{E})$). Thus, the challenge we are facing here is to find a suitable representation for $\mathtt{X}$.

On the one hand, when encoding possibilities, we have to bear in mind that at each time step, we might end up updating possibilities that were previously calculated at any time in the past. As a result, to correctly and univocally encode $\mathtt{X}$, we need to keep track of the following information: *(i)* the time when a possibility was created; *(ii)* the identifier of the possibility; and *(iii)* the eventuality that created it. As a result, we represent $\mathtt{X}$ as the ASP tuple $(\mathtt{t}-1,\mathtt{W},\mathtt{E})$.

On the other hand, when encoding possible worlds, we can always be sure that at each time step we are updating worlds that were created at precisely that time. Thus, the ASP encoding of a possible world is slightly simplified and we represent $\mathtt{X}$ with the ASP tuple $(\mathtt{W},\mathtt{E})$.

Finally, the initial possibilities/worlds are calculated from the initial state representation with the ASP rule $\mathtt{w}(\mathtt{0},\mathtt{W},\mathtt{null})$ `:-` $\mathtt{w\_init}(\mathtt{W})$., where $\mathtt{null}$ is a placeholder that indicates that no action occurred before time $\mathtt{t}$.

Example 1. The possibilities of Example 6 are represented in ASP as follows: *(i)* $\mathtt{w}_1$: $\mathtt{w}(\mathtt{0},\mathtt{w}_1,\mathtt{null})$; *(ii)* $\mathtt{w}_2$: $\mathtt{w}(\mathtt{0},\mathtt{w}_2,\mathtt{null})$; and *(iii)* $\mathtt{v}_3$: $\mathtt{w}(\mathtt{1},(\mathtt{0},\mathtt{w}_1,\mathtt{null}),\mathtt{e}_1)$. Again, notice that, in the ASP encoding of possibilities, we are able to reuse previously calculated information.

Similarly, the possible worlds of Example 3 are represented in ASP as: *(i)* v_1: $\mathtt{w}(\mathtt{1},(\mathtt{w}_1,\mathtt{null}),\mathtt{e}_2)$; *(ii)* v_2: $\mathtt{w}(\mathtt{1},(\mathtt{w}_2,\mathtt{null}),\mathtt{e}_2)$; and *(iii)* v_3: $\mathtt{w}(\mathtt{1},(\mathtt{w}_1,\mathtt{null}),\mathtt{e}_1)$.

Information States and Accessibility Relations. Let $\mathtt{w}(\mathtt{Tw},\mathtt{W},\mathtt{Ew})$ and $\mathtt{w}(\mathtt{Tv},\mathtt{V},\mathtt{Ev})$ be the ASP representations of two possibilities $\mathtt{w}$ and $\mathtt{v}$. Since the possibilities contained in the information states of $\mathtt{w}$ might have been calculated at any time step in the past, in order to encode the fact that $\mathtt{v} \in \mathtt{w}(i)$ (where $i \in \mathcal{AG}$), we need to keep track of the time when $\mathtt{v}$ was created. As a result, the resulting encoding is given by the ASP predicate $\mathtt{r}(\mathtt{Tw},\mathtt{W},\mathtt{Ew},\mathtt{Tv},\mathtt{V},\mathtt{Ev},\mathtt{I})$.

Let now $\mathtt{w}(\mathtt{Tw},\mathtt{W},\mathtt{Ew})$ and $\mathtt{w}(\mathtt{Tv},\mathtt{V},\mathtt{Ev})$ refer to two possible worlds w and v. When representing accessibility relations, we can always be sure that when wR_iv, the ASP representation of the worlds w and v share the same time value (*i.e.*, $\mathtt{Tw} = \mathtt{Tv}$). Thus, we can simplify the encoding as follows: $\mathtt{r}(\mathtt{Tw},\mathtt{W},\mathtt{Ew},\mathtt{V},\mathtt{Ev},\mathtt{I})$.

Example 2. The information states of Example 6 are represented in ASP as follows (we do not further expand the information states of $\mathtt{w}_1$ and $\mathtt{w}_2$ as they refer to previously calculated information):

- $\mathtt{v}_3 \in \mathtt{v}_3(a)$: $\mathtt{r}(\mathtt{1},(\mathtt{0},\mathtt{w}_1,\mathtt{null}),\mathtt{e}_1,\mathtt{1},(\mathtt{0},\mathtt{w}_1,\mathtt{null}),\mathtt{e}_1,\mathtt{a})$;
- $\mathtt{w}_1 \in \mathtt{v}_3(b)$: $\mathtt{r}(\mathtt{1},(\mathtt{0},\mathtt{w}_1,\mathtt{null}),\mathtt{e}_1,\mathtt{0},\mathtt{w}_1,\mathtt{null},\mathtt{b})$; and
- $\mathtt{w}_2 \in \mathtt{v}_3(b)$: $\mathtt{r}(\mathtt{1},(\mathtt{0},\mathtt{w}_1,\mathtt{null}),\mathtt{e}_1,\mathtt{0},\mathtt{w}_2,\mathtt{null},\mathtt{b})$.

The accessibility relations of Example 3 are represented in ASP as follows:

- $v_3 R_a v_3$: $\mathtt{r}(1, (\mathtt{w}_1, \mathtt{null}), \mathtt{e}_1, (\mathtt{w}_1, \mathtt{null}), \mathtt{e}_1, \mathtt{a})$;
- $v_3 R_b v_1$: $\mathtt{r}(1, (\mathtt{w}_1, \mathtt{null}), \mathtt{e}_1, (\mathtt{w}_1, \mathtt{null}), \mathtt{e}_2, \mathtt{b})$;
- $v_3 R_b v_2$: $\mathtt{r}(1, (\mathtt{w}_1, \mathtt{null}), \mathtt{e}_1, (\mathtt{w}_2, \mathtt{null}), \mathtt{e}_2, \mathtt{b})$; and
- $v_x R_i v_y$: $\mathtt{r}(1, (\mathtt{w}_\mathtt{x}, \mathtt{null}), \mathtt{e}_2, (\mathtt{w}_\mathtt{y}, \mathtt{null}), \mathtt{e}_2, \mathtt{i})$, for each $x, y \in \{1, 2\}$ and $i \in \{a, b\}$.

Valuations. For each atom $\mathtt{P}$, we encode the fact that $\mathtt{P}$ is true in the possibility/world $\mathtt{w}(\mathtt{t}, \mathtt{W}, \mathtt{E})$ with the ASP predicate $\mathtt{v}(\mathtt{t}, \mathtt{W}, \mathtt{E}, \mathtt{P})$. We only represent *true* atoms. The initial valuation is calculated from the initial state representation with the ASP rule $\mathtt{v}(0, \mathtt{W}, \mathtt{null}, \mathtt{P})$:- $\mathtt{v\_init}(\mathtt{W}, \mathtt{P})$.

Designated Possibilities and Worlds. A designated possibility/world is represented by the ASP predicate $\mathtt{dw}(\mathtt{t}, \mathtt{W}, \mathtt{E})$, where variables $\mathtt{t}$, $\mathtt{W}$ and $\mathtt{E}$ have the same meaning as in $\mathtt{w}(\mathtt{t}, \mathtt{W}, \mathtt{E})$. The initial designated possibilities/worlds are calculated from the initial state representation with the ASP rule $\mathtt{dw}(0, \mathtt{W}, \mathtt{null})$:- $\mathtt{dw\_init}(\mathtt{W})$.

B.5 Truth Conditions

For space constraints, we abbreviate the representation $\mathtt{Tx}, \mathtt{X}, \mathtt{Ex}$ of a possibility/world as $\bar{\mathtt{X}}$. Truth conditions of formulae are encoded by predicate $\mathtt{holds}(\mathtt{t}, \mathtt{W}, \mathtt{E}, \mathtt{F})$, defined by induction on the structure of formulae as follows:

$$\begin{aligned}
&\mathtt{holds}(\bar{\mathtt{W}}, \mathtt{P}) &&\mathtt{:-}\ \ \mathtt{v}(\bar{\mathtt{W}}, \mathtt{P}), \mathtt{atom}(\mathtt{P}).\\
&\mathtt{holds}(\bar{\mathtt{W}}, \mathtt{neg}(\mathtt{F})) &&\mathtt{:-}\ \ \mathtt{not\ holds}(\bar{\mathtt{W}}, \mathtt{F}).\\
&\mathtt{holds}(\bar{\mathtt{W}}, \mathtt{and}(\mathtt{F}_1, \mathtt{F}_2)) &&\mathtt{:-}\ \ \mathtt{holds}(\bar{\mathtt{W}}, \mathtt{F}_1), \mathtt{holds}(\bar{\mathtt{W}}, \mathtt{F}_2).\\
&\mathtt{holds}(\bar{\mathtt{W}}, \mathtt{dia}(\mathtt{AG}, \mathtt{F})) &&\mathtt{:-}\ \ \mathtt{r}(\bar{\mathtt{W}}, \bar{\mathtt{V}}, \mathtt{AG}), \mathtt{holds}(\bar{\mathtt{V}}, \mathtt{F}).
\end{aligned}$$

B.6 Update Operators

We now describe the ASP encodings of the update operators. As the encodings differ, we present them individually.

Union Update. Let the ASP representations of a possibility spectrum W at time $\mathtt{t}$ and of an eventuality spectrum ACT be given. We now show how the encoding of the update $\mathsf{W}' = \mathsf{W} \bowtie \mathsf{E}$ is obtained. We adopt again the short representation for possibilities/worlds ($\bar{\mathtt{X}}$). We point out that the following ASP rules have a one-to-one correspondence with Definition 19. First, the designated possibilities of W' are determined by the following ASP rules:

$$\begin{aligned}
\mathtt{dw}(\mathtt{t}, \bar{\mathtt{W}}, \mathtt{E}) \mathtt{:-}\ \ &\mathtt{dw}(\bar{\mathtt{W}}), \mathtt{de}(\mathtt{ACT}, \mathtt{E}), \mathtt{pre}(\mathtt{ACT}, \mathtt{E}, \mathtt{PRE}),\\
&\mathtt{holds}(\bar{\mathtt{W}}, \mathtt{PRE}).
\end{aligned}$$

To describe how possibilities are updated, we need the following ASP predicate:

$$\mathtt{qt}(\mathtt{t}, \mathtt{E}, \mathtt{F}, \mathtt{I}) \mathtt{:-}\ \ \mathtt{q}(\mathtt{ACT}, \mathtt{E}, \mathtt{F}, \mathtt{GR}), \mathtt{obs}(\mathtt{t}, \mathtt{I}, \mathtt{GR}).$$

That is, we evaluate the observability conditions of each agent to determine the information states of the action. From this, we can obtain all the updated possibilities recursively:

$$\begin{aligned}
\mathtt{w}(\mathtt{t}, \bar{\mathtt{W}}, \mathtt{E}) \mathtt{:-}\ \ &\mathtt{dw}(\mathtt{t}, \bar{\mathtt{W}}, \mathtt{E}), \mathtt{de}(\mathtt{ACT}, \mathtt{E}).\\
\mathtt{w}(\mathtt{t}, \bar{\mathtt{W}}, \mathtt{E}) \mathtt{:-}\ \ &\mathtt{w}(\mathtt{t}, \bar{\mathtt{V}}, \mathtt{F}), \mathtt{pre}(\mathtt{ACT}, \mathtt{E}, \mathtt{PRE}), \mathtt{-idle}(\mathtt{ACT}, \mathtt{E}),\\
&\mathtt{r}(\bar{\mathtt{V}}, \bar{\mathtt{W}}, \mathtt{I}), \mathtt{qt}(\mathtt{t}, \mathtt{F}, \mathtt{E}, \mathtt{I}), \mathtt{holds}(\bar{\mathtt{W}}, \mathtt{PRE}).
\end{aligned}$$

The first rule states that a designated possibility is a possibility. Let now $\mathsf{v}' = \mathsf{v} \bowtie \mathsf{f}$. Then, the second rule states that for any w and e such that $\mathsf{w} \in \mathsf{v}(i)$, $\mathsf{e} \in \mathsf{f}(i)$ and $\mathsf{w} \models \mathsf{e}(pre)$, we create $\mathsf{w}' = \mathsf{w} \bowtie \mathsf{e}$. Notice that we require that the eventuality e is not *idle*. An eventuality/event is idle when its precondition is $\top$ and when its postconditions are the identity function. In this way, we do not copy redundant information.

Let now $\bar{\mathtt{W}}'$ and $\bar{\mathtt{V}}'$ stand for $\mathtt{t}, \bar{\mathtt{W}}, \mathtt{E}$ and $\mathtt{t}, \bar{\mathtt{V}}, \mathtt{F}$, respectively, and let $\bar{\mathtt{U}}$ stand for $\mathtt{Tu}, \bar{\mathtt{U}}, \mathtt{Eu}$. We encode information states as follows:

$$\mathtt{r}(\bar{\mathtt{W}}', \bar{\mathtt{V}}', \mathtt{I}) \texttt{:-}\ \mathtt{r}(\bar{\mathtt{W}}, \bar{\mathtt{V}}, \mathtt{I}), \mathtt{qt}(\mathtt{t}, \mathtt{E}, \mathtt{F}, \mathtt{I}).$$
$$\mathtt{r}(\bar{\mathtt{W}}', \bar{\mathtt{U}},\ \mathtt{I}) \texttt{:-}\ \mathtt{r}(\bar{\mathtt{W}}, \bar{\mathtt{U}}, \mathtt{I}), \mathtt{qt}(\mathtt{t}, \mathtt{E}, \mathtt{F}, \mathtt{I}), \mathtt{Tu} \leq \mathtt{t}, \mathtt{idle}(\mathtt{ACT}, \mathtt{F}).$$

The first rule states that if $\mathsf{w}' = \mathsf{w} \bowtie \mathsf{e}$ and $\mathsf{v}' = \mathsf{v} \bowtie \mathsf{f}$ are both created at time $\mathtt{t}$ and it holds that $\mathsf{v} \in \mathsf{w}(i)$ and $\mathsf{f} \in \mathsf{e}(i)$, then $\mathsf{v}' \in \mathsf{w}'(i)$. The second rule states that if $\mathsf{w}' = \mathsf{w} \bowtie \mathsf{e}$ is created at time $\mathtt{t}$ and it holds that $\mathsf{u} \in \mathsf{w}(i)$ and there exists an eventuality f such that $\mathsf{f} \in \mathsf{e}(i)$ and $\mathsf{u} = \mathsf{u} \bowtie \mathsf{f}$, then $\mathsf{u} \in \mathsf{w}'(i)$. In this way, we are able to reuse previously calculated information when encoding information states of possibilities.

Finally, we encode the valuation of atoms as follows:

$$\mathtt{v}(\bar{\mathtt{W}}', \mathtt{P}) \texttt{:-}\ \mathtt{post}(\mathtt{ACT}, \mathtt{E}, \mathtt{P}, \mathtt{POST}), \mathtt{holds}(\bar{\mathtt{W}}, \mathtt{POST}).$$
$$\mathtt{v}(\bar{\mathtt{W}}', \mathtt{P}) \texttt{:-}\ \mathtt{inertia}(\mathtt{ACT}, \mathtt{E}, \mathtt{P}), \mathtt{v}(\bar{\mathtt{W}}, \mathtt{P}).$$

The first rule states that if $\mathsf{w} \models \mathsf{e}(p)$, then $\mathsf{w}'(p)$=1. The second rule states that if there are no postconditions associated to an atom (represented by the ASP predicate `inertia`), then in w' we keep the truth value assigned to p in w.

Product Update. Let the ASP representations of a MPKM (M, W_d) at time $\mathtt{t}$ and of a MPEM $(\mathcal{E}, E_d)$ be given. The following ASP rules have a one-to-one correspondence with Definition 6. Designated worlds and valuation of atoms are defined in the same way as in the previous case. As above, let $\bar{\mathtt{W}}'$ and $\bar{\mathtt{V}}'$ stand for $\mathtt{t}, \bar{\mathtt{W}}, \mathtt{E}$ and $\mathtt{t}, \bar{\mathtt{V}}, \mathtt{F}$, respectively. Then, updated possible worlds and accessibility relations are encoded as:

$$\mathtt{w}(\mathtt{t}, \bar{\mathtt{W}}, \mathtt{E})\ \texttt{:-}\ \mathtt{w}(\bar{\mathtt{W}}), \mathtt{e}(\mathtt{ACT}, \mathtt{E}), \mathtt{holds}(\bar{\mathtt{W}}, \mathtt{PRE}).$$
$$\mathtt{r}(\bar{\mathtt{W}}', \bar{\mathtt{V}}', \mathtt{I}) \texttt{:-}\ \mathtt{r}(\bar{\mathtt{W}}, \bar{\mathtt{V}}, \mathtt{I}), \mathtt{qt}(\mathtt{t}, \mathtt{E}, \mathtt{F}, \mathtt{I}).$$

B.7 Epistemic Planning Domains

We overview the planning domains used for the evaluation.

Assemble Line (AL): there are two agents, each responsible for processing a different part of a product. Each agent can fail in processing its part and can inform the other agent of the status of her task. The agents can decide to *assemble* the product or to *restart* the process, depending on their knowledge about the product status. This domain is parametrized on the maximum *modal depth* d of formulae that appear in the action descriptions. The goal in this domain is fixed, *i.e.*, the agents must assemble the product. The aim of this domain is to analyze the impact of the modal depth both in terms of grounding and solving performances.

Coin in the Box (CB): this is a generalization of the domain presented in Example 1. Three agents are in a room where a closed box contains a coin. None of them knows whether the coin lies heads or tails up. To look at the coin, the

agents need to first open the box. Agents can be attentive or distracted: only attentive agents are able to see what actions are executed. Moreover, agents can *signal* others to make them attentive and can also distract them. The goal is for one or more agents to learn the coin position.

Collaboration and Communication (CC): $n \geq 2$ agents move along a corridor with $k \geq 2$ rooms in which $m \geq 1$ boxes are located. Whenever an agent enters a room, it can determine whether a box is there. Agents can then communicate information about the position of the boxes to each other. The goal is to have some agent learn the position of one or more boxes and to know what other agents have learnt.

Grapevine (Gr): $n \geq 2$ agents can move along a corridor with 2 rooms and share their own "secret" to agents in the same room. The goal requires agents to know the secret of one or more agents and to hide their secret to others.

Selective Communication (SC): $n \geq 2$ agents can move along a corridor with $k \geq 2$ rooms. The agents might share some information, represented by an atom q. In each room, only a certain subset of agents is able to listen to what others share. The goal requires for some specific subset of agents to know the information and to hide it to others.

C Graphical Comparison

We briefly report a concrete example on the succinctness of the possibility-based representation, compared to the Kripke-based one (Figures 1 and 2). The figures were generated by our tool. As you can see, even for small plans of length 5, the DELPHIC semantics provide a much more succinct representation. Specifically, in Figure 1 we obtain a total of 15 possibilities, while in Figure 2 we obtain 126 possible worlds. This clearly confirms the benefits of the DELPHIC semantics.

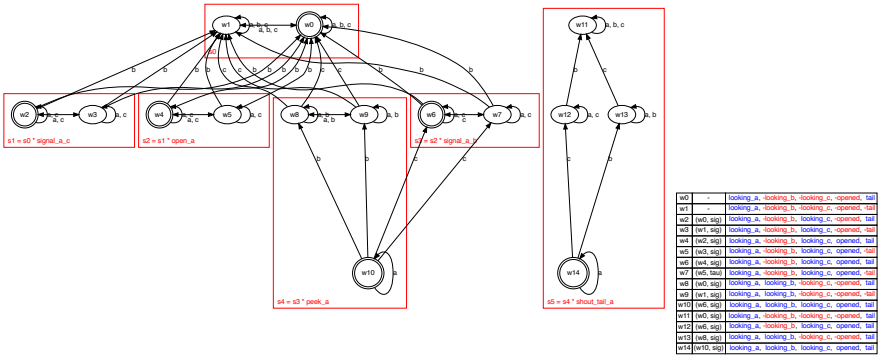

Fig. 1: Graphical representation of a sequence of 5 actions using DELPHIC.

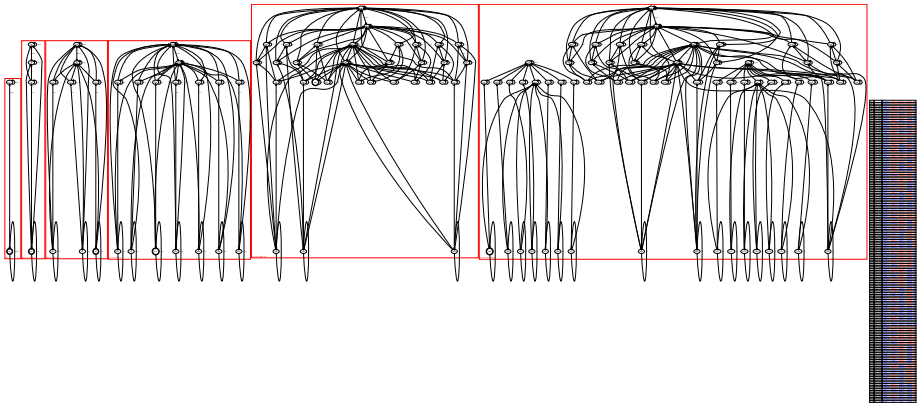

Fig. 2: Graphical representation of a sequence of 5 actions using Kripke semantics.

Assemble									
$\|\mathcal{AG}\|$	$\|\mathcal{P}\|$	$\|W\|$	$\|\mathcal{A}\|$	L	d	Delphic		Kripke	
						Time	Atoms	Time	Atoms
2	4	4	6	5	2	2.560	59332	3.153	123780
					3	2.621	60390	3.606	121194
					4	2.913	61422	4.396	128644
					5	3.117	62478	4.148	125708
					6	3.304	63564	4.774	128328
					7	3.410	64622	4.917	130464
					8	3.372	65680	5.556	138372
					9	3.566	66738	6.161	140804
					10	3.739	67796	6.888	136576
					24	13.103	108000	25.264	218362

Table 1: Results for **AL**.

D Full Experimental Results

We here report the complete tables with the results of our experimental evaluations (Tables 1-5).

References

1. Burigana, A., Fabiano, F., Dovier, A., Pontelli, E.: Modelling multi-agent epistemic planning in ASP. Theory Pract. Log. Program. **20**(5), 593–608 (2020)
2. Fabiano, F., Burigana, A., Dovier, A., Pontelli, E.: EFP 2.0: A multi-agent epistemic solver with multiple e-state representations. In: Beck, J.C., Buffet, O., Hoffmann, J., Karpas, E., Sohrabi, S. (eds.) Proceedings of the Thirtieth International Conference on Automated Planning and Scheduling, Nancy, France, October 26-30, 2020. pp. 101–109. AAAI Press (2020)

Coin in the Box						DELPHIC		Kripke	
$\lvert\mathcal{AG}\rvert$	$\lvert\mathcal{P}\rvert$	$\lvert W\rvert$	$\lvert\mathcal{A}\rvert$	L	d	Time	Atoms	Time	Atoms
3	5	2	21	2	1	0.077	2459	0.098	3094
				3	1	0.215	5828	0.231	8394
				5	3	5.137	77310	7.014	122265
				6	3	27.428	316840	54.091	586037
				7	3	t.o.	-	t.o.	-

Table 2: Results for **CB**.

3. Gebser, M., Kaminski, R., Kaufmann, B., Schaub, T.: Multi-shot ASP solving with clingo. Theory Pract. Log. Program. **19**(1), 27–82 (2019)
4. Kaminski, R., Romero, J., Schaub, T., Wanko, P.: How to build your own asp-based system?! Theory Pract. Log. Program. **23**(1), 299–361 (2023)

Collaboration and Communication									
$\|\mathcal{AG}\|$	$\|\mathcal{P}\|$	$\|W\|$	$\|\mathcal{A}\|$	L	d	Delphic		Kripke	
						Time	Atoms	Time	Atoms
2	10	4	20	3	2	0.129	4579	0.186	8859
				4	1	0.455	10900	0.532	25916
				5	2	2.467	37882	3.599	98226
				6	2	9.435	147183	25.365	385343
				7	2	66.278	636799	254.544	1652783
				8	2	t.o.	-	t.o.	-
3	13	4	30	3	2	0.167	7112	0.328	13532
				4	1	0.774	15873	0.958	37125
				5	2	5.580	57033	9.321	147549
				6	2	18.550	214086	78.667	557746
				7	2	143.178	934859	t.o.	-
				8	2	t.o.	-	t.o.	-
3	15	8	42	3	2	0.905	17288	1.153	36040
				4	1	3.939	46741	4.392	114453
				5	2	14.207	184241	85.423	477169
				6	2	114.014	760234	465.422	1963514
				7	2	t.o.	-	t.o.	-
				8	2	t.o.	-	t.o.	-
2	14	9	28	3	2	0.541	13456	0.723	28514
				4	1	2.778	39077	3.020	96399
				5	2	12.418	152271	38.747	393337
				6	2	57.073	650494	146.581	1681652
				7	2	t.o.	-	t.o.	-
2	17	18	40	3	2	2.688	39310	3.324	87272
				4	1	7.849	128993	14.633	322395
				5	2	52.642	535713	115.531	1376539
				6	2	t.o.	-	t.o.	-
				7	2	t.o.	-	t.o.	-

Table 3: Results for **CC**.

Grapevine									
$\|\mathcal{AG}\|$	$\|\mathcal{P}\|$	$\|W\|$	$\|\mathcal{A}\|$	L	d	Delphic		Kripke	
						Time	Atoms	Time	Atoms
3	9	8	24	2	1	0.104	3644	0.165	9256
				3	1	0.197	5385	0.892	28491
				4	1	0.500	9294	4.142	98558
				5	1	1.934	18691	44.456	372271
				6	2	15.943	43315	t.o.	-
				7	2	52.829	107146	t.o.	-
4	12	16	40	2	1	0.360	10684	0.810	32104
				3	1	1.096	17182	6.613	109032
				4	1	3.694	33024	41.454	412426
				5	1	15.049	73698	t.o.	-
				6	2	87.727	184055	t.o.	-
				7	2	t.o.	-	t.o.	-
5	15	32	60	2	1	1.153	33600	4.593	113362
				3	1	3.695	58527	48.582	417771
				4	1	9.469	123422	t.o.	-
				5	1	77.333	298973	t.o.	-
				6	2	t.o.	-	t.o.	-
				7	2	t.o.	-	t.o.	-

Table 4: Results for **Gr**.

Selective Communication									
$\|\mathcal{AG}\|$	$\|\mathcal{P}\|$	$\|W\|$	$\|\mathcal{A}\|$	L	d	Delphic		Kripke	
						Time	Atoms	Time	Atoms
3	5	2	7	3	2	0.026	943	0.029	1848
				5	1	0.111	5451	0.354	18231
				6	3	0.190	7775	1.948	74916
				8	3	2.063	62367	81.041	1318062
7	5	2	7	5	1	0.188	11342	0.545	30615
				7	2	1.778	67906	20.629	579934
				8	2	4.192	140081	179.118	2617071
8	11	2	13	9	1	0.236	10156	52.347	976904
				10	2	0.339	14519	178.486	1036341
				14	2	17.766	494657	t.o.	-
				15	2	16.430	481155	t.o.	-
9	11	2	13	6	2	4.542	167342	9.196	414842
				8	2	27.503	653357	t.o.	-
				9	2	94.207	1693676	t.o.	-
				12	2	t.o.	-	t.o.	-
9	11	2	13	9	2	0.373	21893	29.649	494173
				10	2	0.723	41257	195.693	1135358
				13	2	1.989	95485	t.o.	-
				17	2	288.827	4023556	t.o.	-
9	12	2	14	4	1	0.084	4712	0.143	12602
				5	1	0.234	15034	0.848	49580
				6	1	0.678	43058	3.560	212899
				7	1	3.165	132163	27.126	1030477
				8	1	15.983	393797	t.o.	-
				9	1	32.373	782700	t.o.	-
				10	1	129.964	2458577	t.o.	-
				11	1	289.298	4209828	t.o.	-

Table 5: Results for **SC**.