Complexity Optimal Decision Procedure for a Propositional Dynamic Logic with Parallel Composition

Joseph Boudou

Complexity Optimal Decision Procedure for a Propositional Dynamic Logic with Parallel Composition

Joseph Boudou(✉)

IRIT – Toulouse University, Toulouse, France
boudou@irit.fr

Abstract. $\mathsf{PPDL}^{\mathrm{det}}$ extends propositional dynamic logic (PDL) with parallel composition of programs. This new construct has separation semantics: to execute the parallel program $(\alpha \,||\, \beta)$ the initial state is separated into two substates and the programs α and β are executed on these substates. By adapting the elimination of Hintikka sets procedure, we provide a decision procedure for the satisfiability problem of $\mathsf{PPDL}^{\mathrm{det}}$. We prove that this decision procedure can be executed in deterministic exponential time, hence that the satisfiability problem of $\mathsf{PPDL}^{\mathrm{det}}$ is EXPTIME-complete.

1 Introduction

Propositional dynamic logic (PDL) is a multi-modal logic designed to reason about behaviors of programs [11,23]. A modal operator $\langle\alpha\rangle$ is associated to each program α, formulas $\langle\alpha\rangle\varphi$ being read "the program α can be executed from the current state to reach a state where the formula φ holds". The set of programs is structured by the following operators: sequential composition $(\alpha;\beta)$ of programs α and β executes β after α; nondeterministic choice $(\alpha\cup\beta)$ of programs α and β executes α or β; test $\varphi?$ on formula φ checks whether the current state satisfies φ; iteration α^* of program α executes α a nondeterministic number of times. The satisfiability problem of PDL is EXPTIME-complete [11,23]. Since PDL programs are abstract, this logic has been successfully adapted to many different domains like knowledge representation or linguistics [9,10,26].

A limitation of PDL is the lack of a construct to reason about concurrency. Different extensions of PDL have been devised to overcome this limitation; let us mention interleaving PDL [1], PDL with intersection [13] and the concurrent dynamic logic [22]. A noteworthy property of these logics is that whenever a parallel program is executable, some of its subprograms are executable too. But in some situations, for example when some agents are forced to cooperate, it may be the case that the parallel composition of some programs is executable while no other programs (but tests) are. The propositional dynamic logic with storing, recovering and parallel composition (PRSPDL) [4] can cope with such situations. This logic extends PDL with parallel compositions of programs and four special programs (for storing and recovering). These five new constructs are

inspired by fork algebras [12] and their semantics rely on a single ternary relation which intuitively models the decomposition of states into two substates. For the parallel program $(\alpha \,||\, \beta)$ to be executed at some state x, x must be decomposed into two states w_1 and w_2 by the ternary relation, then α is executed at w_1 reaching w_3, β is executed at w_2 reaching w_4 and the final state y is obtained by composing w_3 and w_4 by the ternary relation. These semantics for parallel programs are inspired by the concurrent separation logic [6,21] and the ternary relation, called the *separation relation*, is closely related to the Kripke semantics of binary normal modal logics like the Boolean logic of bunched implication [18, 24]. In contrast with unary modalities which usually express relations *between* states, a binary modality can express the internal structure of states, a formula of the form $\varphi \circ \psi$ being read "the current state can be decomposed in two substates, the first one satisfying φ, the other one satisfying ψ". In PRSPDL, the binary modality interpreted by the separation relation can be defined as $\varphi \circ \psi \doteq \langle \varphi? \parallel \psi? \rangle \top$. Hence, PRSPDL embeds both PDL and the minimal binary normal modal logic. Since binary modalities have been used in various fields of logics (see for instance [14,19,27]), combining one with PDL's actions is promising. Despite these interesting features, little is known about PRSPDL. To our knowledge, the only complexity results to date are that the satisfiability problem of PRSPDL is in 2EXPTIME [2] and that variants of PRSPDL interpreted in classes of models where the decomposition of states is deterministic are undecidable [3].

In the present work, we study the logic $\mathsf{PPDL}^{\mathrm{det}}$ which is a variant of PRSPDL. Its language is the fragment of PRSPDL without the four store/recover programs. These special programs were designed to reason about data structures and are of little use to reason about concurrency. The formulas of $\mathsf{PPDL}^{\mathrm{det}}$ are interpreted in the class of models where the composition of states is deterministic: there is at most one way to merge two states. This semantic condition, called $\lhd$-determinism, is quite natural and has been studied in many logics with a binary modality such as Boolean logics of bunched implication [18,24], separation logics [8,25] and arrow logics [19]. Formally, $\lhd$-determinism forces the ternary relation to be a partial binary operator. The satisfiability problem for $\mathsf{PPDL}^{\mathrm{det}}$ has been shown in [5] to be in NEXPTIME, but the exact complexity of this problem was still unknown. In the present paper, we adapt Pratt's elimination of Hintikka sets decision procedure for PDL [23] to prove that the satisfiability problem of $\mathsf{PPDL}^{\mathrm{det}}$ is EXPTIME-complete. Thus, adding a $\lhd$-deterministic parallel composition to PDL does not increase the complexity of the logic. The adaptation of the elimination of Hintikka sets procedure to $\mathsf{PPDL}^{\mathrm{det}}$ is not straightforward. First, as it has been already outlined in [2,5], a comprehensive decomposition of formulas such as the Fischer-Ladner closure is not expressible in $\mathsf{PPDL}^{\mathrm{det}}$. Second, whereas in Pratt [23] states can be considered independently, for $\mathsf{PPDL}^{\mathrm{det}}$ the decomposition path leading to each state must be remembered. Third, like for the filtration [5], $\lhd$-determinism is not preserved by the elimination of Hintikka set procedure.

The paper is organized as follows. In the next section, the language and semantics of $\mathsf{PPDL}^{\mathrm{det}}$ are introduced, along with the $\mathsf{PPDL}^{\mathrm{det}}$ specific notions of threads and twines. In Sect. 3, an adaptation of the Fischer-Ladner closure to

PPDL$^{\mathrm{det}}$ is proposed. Section 4 presents the optimal decision procedure, which is proved to be complete and sound in Sects. 5 and 6 respectively. Section 7 draws a conclusion and proposes perspectives for future works.

2 Language and Semantics of PPDL$^{\mathrm{det}}$

Let Π_0 be a countable set of atomic programs (denoted by $a, b \ldots$) and Φ_0 a countable set of propositional variables (denoted by $p, q \ldots$). The sets Π and Φ of programs and formulas are defined by:

$$\alpha, \beta := a \mid (\alpha \,;\, \beta) \mid (\alpha \cup \beta) \mid \varphi? \mid \alpha^* \mid (\alpha \parallel \beta)$$
$$\varphi := p \mid \bot \mid \neg\varphi \mid \langle\alpha\rangle\varphi$$

Parentheses may be omitted for clarity. We define the dual modalities as usual: $[\alpha]\,\varphi \doteq \neg\langle\alpha\rangle\neg\varphi$. The missing Boolean operators can be defined too, starting with $\varphi \to \psi \doteq [\varphi?]\,\psi$. The syntactic operator $\sim$ is defined such that $\sim\varphi = \psi$ if $\varphi = \neg\psi$ for some ψ and $\sim\varphi = \neg\varphi$ otherwise. We write $|\alpha|$ and $|\varphi|$ for the number of occurrences of symbols in the program α and the formula φ, respectively.

A frame is a tuple $(W, R, \lhd)$ where W is a non-empty set of states (denoted by $w, x, y \ldots$), R is a function assigning a binary relation over W to each atomic program and $\lhd$ is a ternary relation over W called the separation relation. Intuitively, $x\ R(a)\ y$ means that the program a can be executed in state x, reaching state y. Similarly, $x \lhd (y, z)$ means that x can be split into the states y and z. We say that y and z are *substates* of x by the decomposition $(x, y, z) \in\!\lhd$. Equivalently, $x \lhd (y, z)$ means that the substates y and z can be merged to obtain x. When the merging of substates is functional, the frame is said to be $\lhd$-deterministic. This is a common restriction, for instance in Boolean logics of bunched implication [18]. Formally, a frame is $\lhd$-deterministic iff for all $x, y, w_1, w_2 \in W$,

$$\text{if } x \lhd (w_1, w_2) \text{ and } y \lhd (w_1, w_2) \text{ then } x = y \qquad (\lhd\text{-determinism})$$

A model is a tuple $(W, R, \lhd, V)$ where $(W, R, \lhd)$ is a frame and V is a valuation function associating a subset of W to each propositional variable. A model is $\lhd$-deterministic iff its frame is $\lhd$-deterministic. The forcing relation $\vDash$ is defined by parallel induction along with the extension of R to all programs:

$\mathcal{M}, x \vDash p$	iff $x \in V(p)$
$\mathcal{M}, x \vDash \bot$	never
$\mathcal{M}, x \vDash \neg\varphi$	iff $\mathcal{M}, x \nvDash \varphi$
$\mathcal{M}, x \vDash \langle\alpha\rangle\varphi$	iff $\exists y \in W,\ x\ R(\alpha)\ y$ and $\mathcal{M}, y \vDash \varphi$
$x\ R(\alpha \,;\, \beta)\ y$	iff $\exists z \in W,\ x\ R(\alpha)\ z$ and $z\ R(\beta)\ y$
$x\ R(\alpha \cup \beta)\ y$	iff $x\ R(\alpha)\ y$ or $x\ R(\beta)\ y$
$x\ R(\varphi?)\ y$	iff $x = y$ and $\mathcal{M}, x \vDash \varphi$
$x\ R(\alpha^*)\ y$	iff $x\ R(\alpha)^*\ y$ where $R(\alpha)^*$ is the reflexive and transitive closure of $R(\alpha)$
$x\ R(\alpha \parallel \beta)\ y$	iff $\exists w_1, w_2, w_3, w_4 \in W$, $x \lhd (w_1, w_2), w_1\ R(\alpha)\ w_3, w_2\ R(\beta)\ w_4$ and $y \lhd (w_3, w_4)$

$\mathsf{PPDL}^{\mathrm{det}}$ is the logic with language Φ interpreted in the class of $\lhd$-deterministic frames. A formula $\varphi \in \Phi$ is $\mathsf{PPDL}^{\mathrm{det}}$ *satisfiable* iff there exists a $\lhd$-deterministic model $\mathcal{M} = (W, R, \lhd, V)$ and a state $w \in W$ such that $\mathcal{M}, w \vDash \varphi$. The satisfiability problem of $\mathsf{PPDL}^{\mathrm{det}}$ is the decision problem answering whether a formula in Φ is $\mathsf{PPDL}^{\mathrm{det}}$ satisfiable.

Because of the semantics of parallel programs, $\mathsf{PPDL}^{\mathrm{det}}$ does not have the tree-like model property. To overcome this difficulty, $\mathsf{PPDL}^{\mathrm{det}}$ models can be partitioned into parts which have good properties. In [5], *threads* and *twines* were introduced for that purpose. A thread is a set of states which can be reached from each other by some program. Formally, given a model $\mathcal{M} = (W, R, \lhd, V)$, a thread is an equivalence class over W by the symmetric and transitive closure of the relation $\rightsquigarrow$ defined by: $x \rightsquigarrow y$ iff there exists some program α such that $x\ R(\alpha)\ y$. A twine is either a thread which contains no substates of another state or a pair of threads such that whenever a state in one thread is a substate by a decomposition then the other substate by this decomposition belongs to the other thread in the twine. Formally, a twine is an ordered pair $\theta = (t_\ell, t_r)$ of threads such that for all $x, y, z \in W$ if $x \lhd (y, z)$ then $y \notin t_r$, $z \notin t_\ell$ and $y \in t_\ell \Leftrightarrow z \in t_r$. We abusively identify twines with the union of their threads. It has been proved in [5] that whenever a formula is $\mathsf{PPDL}^{\mathrm{det}}$ satisfiable, it is satisfiable in a model $\mathcal{M} = (W, R, \lhd, V)$ such that the set of twines of $\mathcal{M}$ is a partition of W and for any twine θ in $\mathcal{M}$, there is at most two decompositions $(w, x, y) \in \lhd$ such that $\{x, y\} \subseteq \theta$.

3 Fischer-Ladner Closure

The Fischer-Ladner closure [11] is a decomposition of PDL formulas into a comprehensive set of subformulas. This decomposition is used in the elimination of Hintikka sets decision procedure of Pratt [23]. For $\mathsf{PPDL}^{\mathrm{det}}$ we need such a decomposition but parallel compositions of programs cause some difficulties. Firstly, the language of $\mathsf{PPDL}^{\mathrm{det}}$ is not expressive enough for a set of formulas to capture the semantics of formulas of the form $\langle \alpha \parallel \beta \rangle \varphi$. What is missing is some formulas to put after the modalities $\langle \alpha \rangle$ and $\langle \beta \rangle$. For this purpose, we add the special propositional variables L_1, L_2, R_1 and R_2. These new propositional variables identify corresponding left and right components of decompositions by the separation relation: if $x \lhd (y, z)$ then L_t and R_t identify y and z respectively, for some $t \in \{1, 2\}$. These propositional variables are special because we will not include their valuation in the model. Instead, we will allow us to modify their valuation depending on the states we consider (see Sect. 6). Two pairs of new propositional variables are needed because we will consider pairs of decompositions in Sect. 4 and we will need to distinguish them. We write $\Delta^+ = \{L_1, L_2, R_1, R_2\}$, $\Phi_0^+ = \Phi_0 \cup \Delta^+$ and Φ^+ for the formulas over Π_0 and Φ_0^+. Implicitly, Φ denotes the set of formulas over Π_0 and Φ_0, i.e. formulas which do not contain any propositional variables from Δ^+. Secondly, we need to keep track of the level of separation of each subformula. Hence we consider *localized formulas*. A *location* is a word on the alphabet $\{\ell, r\}$, the empty word being

denoted by ϵ. A localized formula is a pair (μ, φ) composed of a location μ and a formula φ.

Then, given a localized formula (μ, φ) we construct the *closure* $\mathrm{Cl}(\mu, \varphi)$ of (μ, φ) as the least set of localized formulas containing (μ, φ) and closed by the rules in Fig. 1. It has to be noted that the closure presented here is an *ad hoc* decomposition devised for the needs of the decision procedure in Sect. 4. More general (and involved) closures for $\mathsf{PPDL}^{\mathrm{det}}$ have been presented in [2,5].

$$\frac{(\mu,\ \langle a\rangle\varphi)}{(\mu,\ \varphi)} \qquad \frac{(\mu,\ \varphi)}{(\mu,\ \sim\varphi)}$$

$$\frac{(\mu,\ \langle\varphi?\rangle\psi)}{(\mu,\ \varphi)\quad(\mu,\ \psi)} \qquad \frac{(\mu,\ \langle\alpha\,;\beta\rangle\varphi)}{(\mu,\ \langle\alpha\rangle\langle\beta\rangle\varphi)}$$

$$\frac{(\mu,\ \langle\alpha^*\rangle\varphi)}{(\mu,\ \langle\alpha\rangle\langle\alpha^*\rangle\varphi)\quad(\mu,\ \varphi)} \qquad \frac{(\mu,\ \langle\alpha\cup\beta\rangle\varphi)}{(\mu,\ \langle\alpha\rangle\varphi)\quad(\mu,\ \langle\beta\rangle\varphi)}$$

$$\frac{(\mu,\ \langle\alpha\parallel\beta\rangle\varphi)}{(\mu.\ell,\ \langle\alpha\rangle L_1)\quad(\mu.\ell,\ \langle\alpha\rangle L_2)\quad(\mu.r,\ \langle\beta\rangle R_1)\quad(\mu.r,\ \langle\beta\rangle R_2)\quad(\mu,\ \varphi)}$$

Fig. 1. Rules of the closure calculus

In the remainder of this paper we will be mainly interested in the closure of localized formulas of the form (ϵ, φ_0) for some formula $\varphi_0 \in \Phi$. We define the abbreviations $\mathrm{Cl}(\varphi_0) = \mathrm{Cl}(\epsilon, \varphi_0)$, $\mathrm{SP}(\varphi_0) = \{\alpha \mid \exists\mu, \exists\varphi, (\mu, \langle\alpha\rangle\varphi) \in \mathrm{Cl}(\varphi_0)\}$ and $\mathrm{Loc}(\varphi_0) = \{\mu \mid \exists\varphi, (\mu, \varphi) \in \mathrm{Cl}(\varphi_0)\}$. The cardinality of $\mathrm{Cl}(\varphi_0)$ is denoted by N_{φ_0}. It can be easily checked that the closure has the two properties expressed by Lemmas 1 and 2.

Lemma 1. *For any location μ, any program α and any formulas $\psi \in \Phi^+$ and $\varphi_0 \in \Phi$, if $(\mu, \langle\alpha\rangle\psi) \in Cl(\varphi_0)$ then $(\mu, \psi) \in Cl(\varphi_0)$.*

Lemma 2. *For any location μ and and any formulas $\psi \in \Phi^+$ and $\varphi_0 \in \Phi$, if $(\mu, \psi) \in Cl(\varphi_0)$ and $\psi? \in \mathrm{SP}(\varphi_0)$ then there are no occurrences of propositional variables from Δ^+ in ψ.*

Moreover, the proof from [11] can be adapted to prove the following lemma:

Lemma 3. *The cardinality of $Cl(\varphi_0)$ is linear in $|\varphi_0|$.*

Proof. For any localized formula (μ, φ), we define the *restricted* closure $\mathrm{rCl}(\mu, \varphi)$ of φ like the closure $\mathrm{Cl}(\mu, \varphi)$ except that the rules for negations, iterations, nondeterministic choices and parallel compositions are replaced with the rules in Fig. 2. The new propositional variables of the form Q_ψ serve the same role as in [11]. Obviously, $\mathrm{Cl}(\epsilon, \varphi_0)$ can be obtained from $\mathrm{rCl}(\epsilon, \varphi_0)$ and the cardinality of $\mathrm{Cl}(\epsilon, \varphi_0)$ is not greater than four times the cardinality of $\mathrm{rCl}(\epsilon, \varphi_0)$. Then, the function γ on programs and formulas is inductively defined by:

$$\begin{aligned}
\gamma(p) &= 1 & \gamma(a) &= 1\\
\gamma(L_1) &= 1 & \gamma(\varphi?) &= \gamma(\varphi) + 1\\
\gamma(R_1) &= 1 & \gamma(\alpha\,;\beta) &= \gamma(\alpha) + \gamma(\beta) + 1\\
\gamma(Q_\varphi) &= 1 & \gamma(\alpha^*) &= \gamma(\alpha) + 2\\
\gamma(\neg\varphi) &= \gamma(\varphi) + 1 & \gamma(\alpha \cup \beta) &= \gamma(\alpha) + \gamma(\beta) + 3\\
\gamma(\langle\alpha\rangle\varphi) &= \gamma(\alpha) + \gamma(\varphi) & \gamma(\alpha \parallel \beta) &= \gamma(\alpha) + \gamma(\beta) + 3
\end{aligned}$$

The following properties can be easily proved by induction on $n > 0$:

1. For any program α, if $n = |\alpha|$ then $\gamma(\alpha) \leq 3n$.
2. For any formula φ, if $n = |\varphi|$ then $\gamma(\varphi) \leq 3n$.
3. For any localized formula (μ, φ), if $\gamma(\varphi) = n$ then the cardinality of $\mathrm{rCl}(\mu, \varphi)$ is less or equal to n. □

$$\frac{(\mu,\ \langle\alpha^*\rangle\varphi)}{(\mu,\ \langle\alpha\rangle Q_{\langle\alpha^*\rangle\varphi}) \quad (\mu,\ \varphi)} \qquad \frac{(\mu,\ \langle\alpha \cup \beta\rangle\varphi)}{(\mu,\ \langle\alpha\rangle Q_\varphi) \quad (\mu,\ \langle\beta\rangle Q_\varphi) \quad (\mu,\ \varphi)}$$

$$\frac{(\mu,\ \neg\varphi)}{(\mu,\ \varphi)} \qquad \frac{(\mu,\ \langle\alpha \parallel \beta\rangle\varphi)}{(\mu.\ell,\ \langle\alpha\rangle L_1) \quad (\mu.r,\ \langle\beta\rangle R_1) \quad (\mu,\ \varphi)}$$

Fig. 2. Replacement rules for the restricted closure calculus

4 Elimination of Hintikka Sets Procedure

In this section we describe a decision procedure for the satisfiability problem of $\mathsf{PPDL}^{\mathrm{det}}$. This decision procedure is based on the elimination of Hintikka set decision procedure devised for PDL by Pratt [23]. The principle of such decision procedures is to first construct a potential finite model for the formula φ_0 being tested for satisfiability. This initial model must somehow embed any possible model which could satisfy φ_0. Then the states of that model not fulfilling some *eventualities* are recursively removed. The procedure succeeds if the final model still contains some states satisfying φ_0. For PDL, states of the initial model are some subsets of the Fischer-Ladner closure called Hintikka sets, an eventuality is a formula of the form $\langle\alpha\rangle\psi$ belonging to a state and a state satisfies a formula if it contains this formula. There are two main difficulties in adapting this decision procedure to $\mathsf{PPDL}^{\mathrm{det}}$. Firstly, Hintikka sets are not sufficient to characterize states of $\mathsf{PPDL}^{\mathrm{det}}$ models. The decomposition path leading to each state is an essential information. Therefore, we introduce *plugs*, which correspond to decompositions by the separation relation, and *sockets*, which are sets of plugs and correspond to twines. A state of the initial model is a pair (H, S) where H is a Hintikka set and S a socket. Secondly, the resulting model is not $\lhd$-deterministic. Hence to prove that whenever the procedure succeeds the formula is satisfiable, a $\lhd$-deterministic model must be constructed from the final

model. This construction is detailed in Sect. 5. In the remainder of the present section, we formally describe the elimination of Hintikka sets procedure for $\mathsf{PPDL}^{\mathrm{det}}$.

Definition 1. *Let $\varphi_0 \in \Phi$ be a formula and μ a location in* $\mathrm{Loc}(\varphi_0)$*. A* Hintikka set *H over φ_0 at μ is any maximal subset of $Cl(\varphi_0)$ verifying all the following conditions:*

1. *If $(\mu', \varphi) \in H$, then $\mu' = \mu$.*
2. *If $(\mu, \neg\varphi) \in Cl(\varphi_0)$, then $(\mu, \neg\varphi) \in H$ iff $(\mu, \varphi) \notin H$.*
3. *If $(\mu, \langle \alpha\,;\beta\rangle\varphi) \in Cl(\varphi_0)$, then $(\mu, \langle \alpha\,;\beta\rangle\varphi) \in H$ iff $(\mu, \langle\alpha\rangle\langle\beta\rangle\varphi) \in H$.*
4. *If $(\mu, \langle \alpha \cup \beta\rangle\varphi) \in Cl(\varphi_0)$, then $(\mu, \langle \alpha \cup \beta\rangle\varphi) \in H$ iff $(\mu, \langle\alpha\rangle\varphi) \in H$ or $(\mu, \langle\beta\rangle\varphi) \in H$.*
5. *If $(\mu, \langle\varphi?\rangle\psi) \in Cl(\varphi_0)$, then $(\mu, \langle\varphi?\rangle\psi) \in H$ iff $(\mu, \varphi) \in H$ and $(\mu, \psi) \in H$.*
6. *If $(\mu, \langle\alpha^*\rangle\varphi) \in Cl(\varphi_0)$, then $(\mu, \langle\alpha^*\rangle\varphi) \in H$ iff $(\mu, \langle\alpha\rangle\langle\alpha^*\rangle\varphi) \in H$ or $(\mu, \varphi) \in H$.*

μ is called the location *of H, denoted by $\lambda(H)$. The set of all Hintikka sets over φ_0 at all $\mu \in \mathrm{Loc}(\varphi_0)$ is denoted by $\mathcal{H}in\,(\varphi_0)$.*

Definition 2. *A* plug *for φ_0 is a triple $P = (H, H_1, H_2)$ of Hintikka sets from $\mathcal{H}in\,(\varphi_0)$ such that:*

1. *$\lambda(H_1) = \lambda(H).\ell$ and $\lambda(H_2) = \lambda(H).r$;*
2. *P has a* type*, which is an index $t \in \{1, 2\}$ such that $(\lambda(H_1), L_t) \in H_1$ and $(\lambda(H_2), R_t) \in H_2$.*

Notice that a plug may have more than one type. Two plugs have different types if there is no $t \in \{1, 2\}$ such that t is a type of both plugs. The *location* of the plug $P = (H, H_1, H_2)$, denoted by $\lambda(P)$, is the location of H.

Definition 3. *A* socket *for φ_0 is a set S of plugs for φ_0 such that:*

1. *S is either the empty set, a singleton or a set $\{P, P'\}$ such that P and P' have the same location but different types;*
2. *for any $(H, H_1, H_2), (H', H_3, H_4) \in S$, any type t' of (H', H_3, H_4) and any α, β, φ such that $(\lambda(H), \langle\alpha \parallel \beta\rangle\varphi) \in Cl(\varphi_0)$,*

$$\begin{aligned} \text{if } (\lambda(H'), \varphi) \in H' \qquad & \text{and} \\ (\lambda(H_1), \langle\alpha\rangle L_{t'}) \in H_1 \qquad & \text{and} \\ (\lambda(H_2), \langle\beta\rangle R_{t'}) \in H_2 & \\ \text{then } (\lambda(H), \langle\alpha \parallel \beta\rangle\varphi) \in H. & \end{aligned}$$

The set of all sockets for φ_0 is denoted by $\mathcal{S}\,(\varphi_0)$. The location set *of a socket S, denoted by $\Lambda(S)$, is defined such that $\Lambda(\emptyset) = \{\epsilon\}$ and for all $S \neq \emptyset$, $\Lambda(S) = \{\lambda(P).\ell, \lambda(P).r \mid P \in S\}$.*

Given a formula $\varphi_0 \in \Phi$ we construct inductively for each $k \in \mathbb{N}$ the tuple $\mathcal{M}_k^{\varphi_0} = (W_k^{\varphi_0}, R_k^{\varphi_0}, \lhd_k^{\varphi_0}, V_k^{\varphi_0})$ where $W_k^{\varphi_0} \subseteq \mathcal{H}in(\varphi_0) \times \mathcal{S}(\varphi_0)$. Each of these tuples is a model iff $W_k^{\varphi_0} \neq \emptyset$. The *restricted accessibility relation* $\widehat{R}_k^{\varphi_0}(\alpha)$ over $W_k^{\varphi_0}$ is inductively defined for all $k \in \mathbb{N}$ and all $\alpha \in \Pi$ by:

- $(H,S)\ \widehat{R}_k^{\varphi_0}(a)\ (H',S')$ iff $(H,S)\ R_k^{\varphi_0}(a)\ (H',S')$,
- $(H,S)\ \widehat{R}_k^{\varphi_0}(\varphi?)\ (H',S')$ iff $(H,S) = (H',S')$ and $(\lambda(H),\varphi) \in H$,
- $(H,S)\ \widehat{R}_k^{\varphi_0}(\alpha\,;\beta)\ (H',S')$ iff $\exists (H'',S'') \in W_k^{\varphi_0}$, $(H,S)\ \widehat{R}_k^{\varphi_0}(\alpha)\ (H'',S'')$ and $(H'',S'')\ \widehat{R}_k^{\varphi_0}(\beta)\ (H',S')$,
- $(H,S)\ \widehat{R}_k^{\varphi_0}(\alpha \cup \beta)\ (H',S')$ iff $(H,S)\ \widehat{R}_k^{\varphi_0}(\alpha)\ (H',S')$ or $(H,S)\ \widehat{R}_k^{\varphi_0}(\beta)\ (H',S')$,
- $(H,S)\ \widehat{R}_k^{\varphi_0}(\alpha^*)\ (H',S')$ iff $(H,S)\,\widehat{R}_k^{\varphi_0}(\alpha)^*\ (H',S')$ where $\widehat{R}_k^{\varphi_0}(\alpha)^*$ is the reflexive and transitive closure of $\widehat{R}_k^{\varphi_0}(\alpha)$,
- $(H,S)\ \widehat{R}_k^{\varphi_0}(\alpha \parallel \beta)\ (H',S')$ iff $S = S'$ and $\exists H_1,H_2,H_3,H_4 \in \mathcal{H}in(\varphi_0)$, $S'' = \{(H,H_1,H_2),(H',H_3,H_4)\} \in \mathcal{S}(\varphi_0)$, $(H_1,S'')\ \widehat{R}_k^{\varphi_0}(\alpha)\ (H_3,S'')$ and $(H_2,S'')\ \widehat{R}_k^{\varphi_0}(\beta)\ (H_4,S'')$.

Initial Step. The initial tuple $\mathcal{M}_0^{\varphi_0} = (W_0^{\varphi_0}, R_0^{\varphi_0}, \lhd_0^{\varphi_0}, V_0^{\varphi_0})$ is constructed as follows:

- $W_0^{\varphi_0}$ is the set of pairs $(H,S) \in \mathcal{H}in(\varphi_0) \times \mathcal{S}(\varphi_0)$ such that $\lambda(H) \in \Lambda(S)$,
- for all $a \in \Pi_0$, $(H,S)\ R_0^{\varphi_0}(a)\ (H',S')$ iff $S = S'$ and $\forall(\mu,\varphi) \in H'$, if $(\mu,\langle a\rangle\varphi) \in \mathrm{Cl}(\varphi_0)$ then $(\mu,\langle a\rangle\varphi) \in H$,
- $(H,S) \lhd_0^{\varphi_0} ((H_1,S_1),(H_2,S_2))$ iff $S_1 = S_2$ and $(H,H_1,H_2) \in S_1$,
- for all $p \in \Phi_0^+$, $V_0^{\varphi_0}(p) = \{(H,S) \in W_0^{\varphi_0} \mid (\lambda(H),p) \in H\}$.

Inductive $(k+1)^{th}$ *Step.* Suppose $\mathcal{M}_k^{\varphi_0} = (W_k^{\varphi_0}, R_k^{\varphi_0}, \lhd_k^{\varphi_0}, V_k^{\varphi_0})$ has already been defined. A state $(H,S) \in W_k^{\varphi_0}$ is *demand-satisfied* in $\mathcal{M}_k^{\varphi_0}$ iff for any program α and any formula φ, if $(\lambda(H),\langle\alpha\rangle\varphi) \in H$ then there exists $(H',S') \in W_k^{\varphi_0}$ such that $(H,S)\ \widehat{R}_k^{\varphi_0}(\alpha)\ (H',S')$ and $(\lambda(H'),\varphi) \in H'$. Define $\mathcal{M}_{k+1}^{\varphi_0} = \left(W_{k+1}^{\varphi_0}, R_{k+1}^{\varphi_0}, \lhd_{k+1}^{\varphi_0}, V_{k+1}^{\varphi_0}\right)$ such that :

- $W_{k+1}^{\varphi_0} = \{(H,S) \in W_k^{\varphi_0} \mid (H,S)$ is demand-satisfied in $\mathcal{M}_k^{\varphi_0}\}$,
- $R_{k+1}^{\varphi_0}(a) = R_k^{\varphi_0}(a) \cap (W_{k+1}^{\varphi_0})^2$ for all $a \in \Pi_0$,
- $\lhd_{k+1}^{\varphi_0} = \lhd_k^{\varphi_0} \cap (W_{k+1}^{\varphi_0})^3$,
- $V_{k+1}^{\varphi_0}(p) = V_k^{\varphi_0}(p) \cap W_{k+1}^{\varphi_0}$ for all $p \in \Phi_0^+$.

It can be easily proved that there is less than $2^{7N_{\varphi_0}+1}$ states in $W_0^{\varphi_0}$. Therefore, there exists $n \leq 2^{7N_{\varphi_0}+1}$ such that $\mathcal{M}_n^{\varphi_0} = \mathcal{M}_{n+k}^{\varphi_0}$ for all $k \in \mathbb{N}$. Let $\mathcal{M}^{\varphi_0} = (W^{\varphi_0}, R^{\varphi_0}, \lhd^{\varphi_0}, V^{\varphi_0}) = \mathcal{M}_n^{\varphi_0}$. Our procedure succeeds iff there is a state $(H_0,S_0) \in W^{\varphi_0}$ such that $(\epsilon,\varphi_0) \in H_0$.

Lemma 4. *Given a formula φ_0, to construct the corresponding model $\mathcal{M}^{\varphi_0}$ and to check whether there is a state $(H_0,S_0) \in W^{\varphi_0}$ such that $(\epsilon,\varphi_0) \in H_0$ can be done in deterministic exponential time.*

Proof. We have already stated that the procedure constructs at most an exponential number of models. The method from [16] can be easily adapted to prove that $\widehat{R}_k^{\varphi_0}(\alpha)$ can be computed in time polynomial in the cardinality of $W_k^{\varphi_0}$. Therefore, the whole procedure can be executed in deterministic exponential time. □

The remainder of this work is devoted to prove that this procedure is a decision procedure for the satisfiability problem of $\mathsf{PPDL}^{\mathrm{det}}$. We use the traditional vocabulary used for the dual problem of validity.

5 Completeness

In this section, we suppose that $\mathcal{M}^{\varphi_0} = (W^{\varphi_0}, R^{\varphi_0}, \lhd^{\varphi_0}, V^{\varphi_0})$ has been constructed, for a given formula $\varphi_0 \in \Phi$, as defined in the previous section and that there exists $(H_0, S_0) \in W^{\varphi_0}$ such that $(\epsilon, \varphi_0) \in H_0$. We will prove that φ_0 is satisfiable in the class of $\lhd$-deterministic models. Obviously, $\mathcal{M}^{\varphi_0}$ is a model. But in the general case, $\mathcal{M}^{\varphi_0}$ is not $\lhd$-deterministic. Therefore we will construct from $\mathcal{M}^{\varphi_0}$ a $\lhd$-deterministic model $\mathcal{M}^{\mathrm{det}} = \left(W^{\mathrm{det}}, R^{\mathrm{det}}, \lhd^{\mathrm{det}}, V^{\mathrm{det}}\right)$ satisfying φ_0. The main idea is to consider the equivalence classes by the relation $\asymp$ over W^{φ_0} defined such that $(H, S) \asymp (H', S')$ iff $S = S'$. It can be proved that, by removing from $\mathcal{M}^{\varphi_0}$ some "unreachable" states, these equivalence classes are twines and that the removed states are not needed in the proofs of the present section. Hence we will abusively call these equivalence classes *twines*. Remark that each such twine corresponds exactly to a socket. The *initial* twine θ_0 is the twine which corresponds to the empty socket $\emptyset$. The model $\mathcal{M}^{\mathrm{det}}$ is constructed inductively as follows. Initially, the model contains only a copy of the initial twine θ_0. Then, whenever two states in $\mathcal{M}^{\mathrm{det}}$ are copies of states reachable in $\mathcal{M}^{\varphi_0}$ by a parallel program, a copy of the twine linking these two states in $\mathcal{M}^{\varphi_0}$ is added to $\mathcal{M}^{\mathrm{det}}$. Since there are no decompositions within twines, we can ensure that $\mathcal{M}^{\mathrm{det}}$ is $\lhd$-deterministic, while preserving the satisfiability of φ_0.

Formally, to be able to copy twines, hence states, the states of $\mathcal{M}^{\mathrm{det}}$ are pairs $(i, (H, S))$ where i is a positive natural number and (H, S) is a state from $\mathcal{M}^{\varphi_0}$. We define the set $PL \subseteq \mathbb{N} \times W^{\varphi_0} \times \mathrm{SP}(\varphi_0) \times W^{\varphi_0}$ of parallel links such that $(n, (H, S), \alpha, (H', S')) \in PL$ iff $(H, S)\ \widehat{R}^{\varphi_0}(\alpha)\ (H', S')$ and there exists $\beta, \gamma \in \Pi$ such that $\alpha = \beta \parallel \gamma$. As both W^{φ_0} and $\mathrm{SP}(\varphi_0)$ are finite, PL can be totally ordered such that $(n_1, (H_1, S_1), \alpha_1, (H_1', S_1')) < (n_2, (H_2, S_2), \alpha_2, (H_2', S_2'))$ implies $n_1 \leq n_2$. If PL is not empty, such an order has a least element, hence the k^{th} element of PL is well defined for all $k \in \mathbb{N}$. Moreover, if $(n, (H, S), \alpha, (H', S'))$ is the k^{th} element of PL, then $n \leq k$. Now, we construct inductively the models $\left(\mathcal{M}_k^{\mathrm{det}}\right)_{k \in \mathbb{N}}$ as follows.

Initial Step. $\mathcal{M}_0^{\det} = \left(W_0^{\det}, R_0^{\det}, \lhd_0^{\det}, V_0^{\det}\right)$ is defined such that:

$$\begin{aligned}
W_0^{\det} &= \{(0,(H,S)) \mid (H,S) \in \theta_0\} \\
R_0^{\det}(a) &= \big\{((i_F,(H_F,S_F)),(i_T,(H_T,S_T))) \in W_0^{\det} \times W_0^{\det} \mid \\
&\qquad\qquad i_F = i_T \text{ and } (H_F,S_F)\ R^{\varphi_0}\ (a)\ (H_T,S_T)\big\} \\
\lhd_0^{\det} &= \emptyset \\
V_0^{\det}(p) &= \big\{(i,(H,S)) \in W_0^{\det} \mid (\lambda(H),p) \in H\big\}
\end{aligned}$$

If PL is empty, let us define $\mathcal{M}_k^{\det} = \mathcal{M}_0^{\det}$ for all $k > 0$. Otherwise, the following step is applied recursively.

Inductive $(k+1)^{th}$ *Step.* Suppose that $\mathcal{M}_k^{\det}$ has already been constructed and $(n,(H,S),\alpha \parallel \beta,(H',S'))$ is the k^{th} tuple in PL. If $(n,(H,S)) \notin W_k^{\det}$ or $(n,(H',S')) \notin W_k^{\det}$ then $\mathcal{M}_{k+1}^{\det} = \mathcal{M}_k^{\det}$. Otherwise, since $(H,S)\ \widehat{R}^{\varphi_0}(\alpha \parallel \beta)\ (H',S')$, there exists $H_1,H_2,H_3,H_4 \in \mathcal{H}in(\varphi_0)$ such that $S'' = \{(H,H_1,H_2),(H',H_3,H_4)\} \in \mathcal{S}(\varphi_0)$, $(H_1,S'')\ \widehat{R}^{\varphi_0}(\alpha)\ (H_3,S'')$ and $(H_2,S'')\ \widehat{R}^{\varphi_0}(\beta)\ (H_4,S'')$. Let θ be the twine corresponding to S''. The model $\mathcal{M}_{k+1}^{\det}$ is defined by:

$$\begin{aligned}
W_{k+1}^{\det} &= W_k^{\det} \cup \{(i,(H''',S''')) \mid i = k+1 \text{ and } (H''',S''') \in \theta\} \\
R_{k+1}^{\det}(a) &= \big\{((i_F,(H_F,S_F)),(i_T,(H_T,S_T))) \in W_{k+1}^{\det} \times W_{k+1}^{\det} \mid \\
&\qquad\qquad i_F = i_T \text{ and } (H_F,S_F)\ R^{\varphi_0}\ (a)\ (H_T,S_T)\big\} \\
\lhd_{k+1}^{\det} &= \lhd_k^{\det} \cup \{((n,(H,S)),(k+1,(H_1,S'')),(k+1,(H_2,S''))), \\
&\qquad\qquad ((n,(H',S')),(k+1,(H_3,S'')),(k+1,(H_4,S'')))\} \\
v_{k+1}^{\det}(p) &= \big\{(i,(H''',S''')) \in W_{k+1}^{\det} \mid (\lambda(H'''),p) \in H'''\big\}
\end{aligned}$$

Finally, the model $\mathcal{M}^{\det}$ is defined as the union of all the models $\mathcal{M}_k^{\det}$ for $k \in \mathbb{N}$. We prove now that $\mathcal{M}^{\det}$ is a $\lhd$-deterministic model satisfying φ_0.

Lemma 5. *$\mathcal{M}^{det}$ is $\lhd$-deterministic.*

Proof. Let us suppose that $(k,(H,S)) \lhd^{\det} ((k_1,(H_1,S_1)),(k_2,(H_2,S_2)))$ and $(k',(H',S')) \lhd^{\det} ((k_1,(H_1,S_1)),(k_2,(H_2,S_2)))$. By construction, $k_1 = k_2$ and $S_1 = S_2$. Moreover, those two tuples have been added to $\lhd^{\det}$ at the ${k_1}^{\text{th}}$ inductive step. Therefore, $k = k'$, $S = S'$ and $\{(H,H_1,H_2),(H',H_1,H_2)\} \in \mathcal{S}(\varphi_0)$. Since the types of (H,H_1,H_2) and (H',H_1,H_2) only depend on H_1 and H_2, these two plugs have the same types. Hence, by Definition 3, $H = H'$. □

To prove that $\mathcal{M}^{\det}$ satisfies φ_0 (Lemma 8), we need the following two lemmas.

Lemma 6. *For all $k \in \mathbb{N}$, all $(H,S),(H',S') \in W_k^{\varphi_0}$ and all programs α, if $(H,S)\ \widehat{R}_k^{\varphi_0}(\alpha)\ (H',S')$, then $S = S'$, $\lambda(H) = \lambda(H')$ and for all $i \leq k$, $(H,S)\ \widehat{R}_i^{\varphi_0}(\alpha)\ (H',S')$.*

Proof. By a simple induction on $|\alpha|$. We detail only the case for parallel compositions. Suppose that $(H,S)\ \widehat{R}_k^{\varphi_0}\ (\alpha \parallel \beta)\ (H',S')$. By definition, $S = S'$ and there exists H_1, H_2, H_3, H_4 such that $S'' = \{(H, H_1, H_2), (H', H_3, H_4)\}$ is a socket, $(H_1, S'')\ \widehat{R}_k^{\varphi_0}\ (\alpha)\ (H_3, S'')$ and $(H_2, S'')\ \widehat{R}_k^{\varphi_0}\ (\beta)\ (H_4, S'')$. Since S'' is a socket, $\lambda(H) = \lambda(H')$. By induction, for all $i \leq k$, $(H_1, S'')\ \widehat{R}_i^{\varphi_0}\ (\alpha)\ (H_3, S'')$ and $(H_2, S'')\ \widehat{R}_i^{\varphi_0}\ (\beta)\ (H_4, S'')$, hence $(H,S)\ \widehat{R}_i^{\varphi_0}\ (\alpha \parallel \beta)\ (H',S')$. □

Lemma 7. *For all* $(H,S), (H',S') \in W^{\varphi_0}$*, all* $\alpha \in \Pi$ *and all* $\varphi \in \Phi$*, if* $(\lambda(H), [\alpha]\,\varphi) \in H$ *and* $(H,S)\ \widehat{R}^{\varphi_0}\ (\alpha)\ (H',S')$ *then* $(\lambda(H'), \varphi) \in H'$*.*

Proof. The proof is by induction on $|\alpha|$. We only prove the case when α is a parallel composition. The other cases are straightforward and left to the reader. Suppose that $(\lambda(H), [\alpha \parallel \beta]\,\varphi) \in H$ and $(H,S)\ \widehat{R}^{\varphi_0}\ (\alpha \parallel \beta)\ (H',S')$. By definition, there exists $H_1, H_2, H_3, H_4 \in \mathcal{H}in\,(\varphi_0)$ such that, $S'' = \{(H, H_1, H_2), (H', H_3, H_4)\} \in \mathcal{S}\,(\varphi_0)$, $(H_1, S'')\ \widehat{R}^{\varphi_0}\ (\alpha)\ (H_3, S'')$ and $(H_2, S'')\ \widehat{R}^{\varphi_0}\ (\beta)\ (H_4, S'')$. As H is a Hintikka set, $(\lambda(H), \langle\alpha \parallel \beta\rangle\neg\varphi) \notin H$. Since (H', H_3, H_4) is a plug, there exists $t' \in \{1,2\}$ such that $(\lambda(H_3), L_{t'}) \in H_3$ and $(\lambda(H_4), R_{t'}) \in H_4$. And since S'' is a socket, by Condition 2 of Definition 3, one of the following statements holds:

$$(\lambda(H_1), \langle\alpha\rangle L_{t'}) \notin H_1 \tag{1}$$

$$(\lambda(H_2), \langle\beta\rangle R_{t'}) \notin H_2 \tag{2}$$

$$(\lambda(H'), \neg\varphi) \notin H' \tag{3}$$

If (1) holds, then $(\lambda(H_1), [\alpha]\,\neg L_{t'}) \in H_1$ and by the induction hypothesis $(\lambda(H_3), \neg L_{t'}) \in H_3$ which is a contradiction. The case when (2) holds is similar. Finally, if (3) holds, then $(\lambda(H'), \varphi) \in H'$. □

We can now state the following truth lemma.

Lemma 8 (Truth lemma). *For all* $(k, (H,S)) \in W^{det}$ *and all* $(\mu, \varphi) \in Cl(\varphi_0)$*,*

$$(\mu, \varphi) \in H \text{ iff } \mathcal{M}^{det}, (k, (H,S)) \vDash \varphi \text{ and } \lambda(H) = \mu$$

Proof. The following two properties are proved by induction on n for all $n \in \mathbb{N}$ and all $(k, (H,S)) \in W^{\mathrm{det}}$:

IH.1 for all $\alpha \in \Pi$ and all $(k', (H',S')) \in W^{\mathrm{det}}$, if $n = |\alpha|$ and $\exists\varphi \in \Phi^+$ such that $(\lambda(H), \langle\alpha\rangle\varphi) \in \mathrm{Cl}\,(\varphi_0)$, then:

$$(k, (H,S))\ R^{\mathrm{det}}\ (\alpha)\ (k', (H',S')) \text{ iff } (H,S)\ \widehat{R}^{\varphi_0}\ (\alpha)\ (H',S') \text{ and } k = k'$$

IH.2 for all $(\mu, \varphi) \in \mathrm{Cl}\,(\varphi_0)$, if $n = |\varphi|$ and $\lambda(H) = \mu$ then:

$$(\mu, \varphi) \in H \text{ iff } \mathcal{M}^{\mathrm{det}}, (k, (H,S)) \vDash \varphi$$

First note that by Lemma 6 and by the construction of $\mathcal{M}^{\text{det}}$, if $(k,(H,S)) \in W^{\text{det}}$ and $(H,S)\ \widehat{R}^{\varphi_0}(\alpha)\ (H',S')$ then $(k,(H',S')) \in W^{\text{det}}$. Then for IH.1, we detail only the case for parallel compositions, the other cases being straightforward. Suppose $\alpha = \beta \parallel \gamma$. For the right-to-left direction, $(k,(H,S),\alpha,(H',S')) \in PL$, hence by construction and by IH.1, $(k,(H,S))\ R^{\text{det}}(\alpha)\ (k',(H',S'))$. For the left-to-right direction, there exists $w_i = (k_i,(H_i,S_i)) \in W^{\text{det}}$ for each $i \in 1..4$ such that $(k,(H,S)) \lhd^{\text{det}} (w_1,w_2)$, $w_1\ R^{\text{det}}(\beta)\ w_3$, $w_2\ R^{\text{det}}(\gamma)\ w_4$ and $(k',(H',S')) \lhd^{\text{det}} (w_3,w_4)$. By IH.1, $k_1 = k_3$, $k_2 = k_4$, $(H_1,S_1)\ \widehat{R}^{\varphi_0}(\beta)\ (H_3,S_3)$ and $(H_2,S_2)\ \widehat{R}^{\varphi_0}(\gamma)\ (H_4,S_4)$. By the construction of $\mathcal{M}^{\text{det}}$, $k = k'$, $S_1 = S_2 = S_3 = S_4$ and $\{(H,H_1,H_2),(H',H_3,H_4)\} \subseteq S_1$. And since any subset of a socket is a socket, $(H,S)\ \widehat{R}^{\varphi_0}(\alpha)\ (H',S')$. For IH.2, the cases for propositional variables and their negation are trivial. For diamond modalities, suppose $\varphi = \langle\alpha\rangle\psi$. By construction of $\mathcal{M}^{\varphi_0}$, there is $(H',S') \in W^{\varphi_0}$ such that $(H,S)\ \widehat{R}^{\varphi_0}(\alpha)\ (H',S')$ and $(\lambda(H'),\psi) \in H'$. And by IH.1 and IH.2, $\mathcal{M}^{\text{det}},(k,(H,S)) \vDash \langle\alpha\rangle\psi$. The case for box modalities is managed by Lemma 7. □

By hypothesis, there exists $(H,S) \in W^{\varphi_0}$ such that $(\epsilon,\varphi_0) \in H$. And by construction, for any state $(H,S) \in W^{\varphi_0}$, if $\lambda(H) = \epsilon$ then $(0,(H,S)) \in W^{\text{det}}$. Therefore, Lemma 8 proves $\mathcal{M}^{\text{det}}$ satisfy φ_0.

6 Soundness

In this section we prove that for any $\mathsf{PPDL}^{\text{det}}$ formula φ_0, if φ_0 is satisfiable then there exists $(H_0,S_0) \in W^{\varphi_0}$ such that $(\epsilon,\varphi_0) \in H_0$, where $\mathcal{M}^{\varphi_0} = (W^{\varphi_0},R^{\varphi_0},\lhd^{\varphi_0},V^{\varphi_0})$ has been obtained by the elimination of Hintikka sets procedure described in Sect. 4. The proof proceeds as follows. First, considering a $\mathsf{PPDL}^{\text{det}}$ model $\mathcal{M}$ satisfying φ_0, a correspondence between the states of $\mathcal{M}$ and some states of $W_0^{\varphi_0}$ is constructed. Then, it is proved that the states of $W_0^{\varphi_0}$ corresponding to states in $\mathcal{M}$ can not be deleted by the procedure and that one of these states $(H_0,S_0) \in W_0^{\varphi_0}$ is such that $(\epsilon,\varphi_0) \in H_0$. The difficulties come from the involved structure of $\mathcal{M}_0^{\varphi_0}$ with locations, Hintikka sets, plugs and sockets and from the new propositional variables L_t and R_t. To overcome these difficulties, we use the following result from [5] which allows us to assume some properties about $\mathcal{M}$.

Proposition 1. *For any formula φ_0, if φ_0 is PPDL^{det} satisfiable then there exists a model $\mathcal{M} = (W,R,\lhd,V)$, a state $x_0 \in W$ and a function λ from W to locations such that for all $x,y,z \in W$ and $\alpha \in \Pi$:*

$$\mathcal{M},x_0 \vDash \varphi_0 \tag{4}$$

$$\lambda(x_0) = \epsilon \tag{5}$$

$$\text{if } x \lhd (y,z) \text{ then } \lambda(y) = \lambda(x).\ell \text{ and } \lambda(z) = \lambda(x).r \tag{6}$$

$$\text{if } x\ R(\alpha)\ y \text{ then } \lambda(x) = \lambda(y) \tag{7}$$

From now on, we assume that $\mathcal{M}$ is as described in Proposition 1. In order to interpret the new propositional variables introduced by the closure defined in Sect. 3, extensions of the valuation V are defined as follows.

Definition 4. *For any model $\mathcal{M} = (W, R, \lhd, V)$ and any formulas φ_0, a valuation extension of $\mathcal{M}$ to φ_0 is a function $\mathcal{V}$ from the new propositional variables L_1, L_2, R_1 and R_2 to subsets of W. We write $\mathcal{M}+\mathcal{V}$ for the model $(W, R, \lhd, V')$ where V' is the function satisfying:*

$$V'(p) = V(p) \text{ iff } p \in \Phi_0$$
$$V'(p) = \mathcal{V}(p) \text{ iff } p \in \Delta^+$$

The set of all valuation extensions of $\mathcal{M}$ to φ_0 is denoted by $\mathfrak{V}(\mathcal{M}, \varphi_0)$.

It has to be noticed that by Lemma 2, there are no occurrences of propositional variables from Δ^+ in any program of $\mathrm{SP}(\varphi_0)$. Therefore, as long as only programs from $\mathrm{SP}(\varphi_0)$ are considered, the extension of R for $\mathcal{M}+\mathcal{V}$ does not depend on $\mathcal{V}$ and the notation R for this extension is not ambiguous.

Then, to define the correspondence between W and $W_0^{\varphi_0}$, we define the functions h_{hin}, h_{plug}, h_{socket} and h_{state} such that for all $x, y, z \in W$, $\mathcal{T} \subseteq \lhd$ and $\mathcal{V}, \mathcal{V}' \in \mathfrak{V}(\mathcal{M}, \varphi_0)$:

$$\begin{aligned}
h_{\text{hin}}(x, \mathcal{V}) &= \{(\mu, \varphi) \in \mathrm{Cl}(\varphi_0) \mid \mu = \lambda(x) \text{ and } \mathcal{M}+\mathcal{V}, x \vDash \varphi\} \\
h_{\text{plug}}((x, y, z), \mathcal{V}, \mathcal{V}') &= (h_{\text{hin}}(x, \mathcal{V}'), h_{\text{hin}}(y, \mathcal{V}), h_{\text{hin}}(z, \mathcal{V})) \\
h_{\text{socket}}(\mathcal{T}, \mathcal{V}, \mathcal{V}') &= \{h_{\text{plug}}(D, \mathcal{V}, \mathcal{V}') \mid D \in \mathcal{T}\} \\
h_{\text{state}}(x, \mathcal{T}, \mathcal{V}, \mathcal{V}') &= (h_{\text{hin}}(x, \mathcal{V}), h_{\text{socket}}(\mathcal{T}, \mathcal{V}, \mathcal{V}'))
\end{aligned}$$

A state $(H, S) \in W_0^{\varphi_0}$ *has a correspondence* if there exists $x \in W$, $\mathcal{T} \subseteq \lhd$ and $\mathcal{V}, \mathcal{V}' \in \mathfrak{V}(\mathcal{M}, \varphi_0)$ such that $h_{\text{state}}(x, \mathcal{T}, \mathcal{V}, \mathcal{V}') = (H, S)$. Obviously, for all $x \in W$ and all $\mathcal{V} \in \mathfrak{V}(\mathcal{M}, \varphi_0)$, $h_{\text{hin}}(x, \mathcal{V})$ is a Hintikka set. The following lemmas prove this correspondence has the desired properties.

Lemma 9. *For some $x \in W$, $\mathcal{T} \subseteq \lhd$ and $\mathcal{V}, \mathcal{V}' \in \mathfrak{V}(\mathcal{M}, \varphi_0)$, $(\epsilon, \varphi_0) \in h_{hin}(x, \mathcal{V})$ and $h_{state}(x, \mathcal{T}, \mathcal{V}, \mathcal{V}') \in W_0^{\varphi_0}$.*

Proof. Define $\mathcal{V}_\emptyset$ such that $\mathcal{V}_\emptyset(p) = \emptyset$ for all $p \in \Delta^+$. By (4) and (5), $(\epsilon, \varphi_0) \in h_{\text{hin}}(x_0, \mathcal{V}_\emptyset)$. Moreover, $h_{\text{socket}}(\emptyset, \mathcal{V}_\emptyset, \mathcal{V}_\emptyset) = \emptyset$ is trivially a socket and since $\Lambda(\emptyset) = \{\epsilon\}$, $h_{\text{state}}(x_0, \emptyset, \mathcal{V}_\emptyset, \mathcal{V}_\emptyset) \in W_0^{\varphi_0}$. □

Lemma 10. *For all $x \in W$, all $\mathcal{T} \subseteq \lhd$ and all $\mathcal{V}, \mathcal{V}' \in \mathfrak{V}(\mathcal{M}, \varphi_0)$,*

$$\text{if } h_{state}(x, \mathcal{T}, \mathcal{V}, \mathcal{V}') \in W_0^{\varphi_0} \text{ then for all } k \in \mathbb{N},\ h_{state}(x, \mathcal{T}, \mathcal{V}, \mathcal{V}') \in W_k^{\varphi_0}.$$

Proof. We prove by induction on k that for all $k \in \mathbb{N}$, $x \in W$, $\mathcal{T} \subseteq \lhd$ and $\mathcal{V}, \mathcal{V}' \in \mathfrak{V}(\mathcal{M}, \varphi_0)$:

IH.1 if $h_{\text{state}}(x, \mathcal{T}, \mathcal{V}, \mathcal{V}') \in W_0^{\varphi_0}$ then $h_{\text{state}}(x, \mathcal{T}, \mathcal{V}, \mathcal{V}') \in W_k^{\varphi_0}$;
IH.2 for all $y \in W$ and all $\alpha \in \Pi$ such that $\exists \varphi, (\lambda(x), \langle\alpha\rangle\varphi) \in \mathrm{Cl}(\varphi_0)$, if $x\ R(\alpha)\ y$ and $h_{\text{state}}(x, \mathcal{T}, \mathcal{V}, \mathcal{V}') \in W_k^{\varphi_0}$ then $h_{\text{state}}(y, \mathcal{T}, \mathcal{V}, \mathcal{V}') \in W_k^{\varphi_0}$ and $h_{\text{state}}(x, \mathcal{T}, \mathcal{V}, \mathcal{V}')\ \widehat{R}_k^{\varphi_0}(\alpha)\ h_{\text{state}}(y, \mathcal{T}, \mathcal{V}, \mathcal{V}')$.

Base case. IH.1 is trivial. For IH.2, we first prove that $h_{\text{state}}(y, \mathcal{T}, \mathcal{V}, \mathcal{V}') \in W_0^{\varphi_0}$. By hypothesis, $h_{\text{socket}}(\mathcal{T}, \mathcal{V}, \mathcal{V}')$ is a socket. Hence it only remains to be proved that $\lambda(y) \in \Lambda(h_{\text{socket}}(\mathcal{T}, \mathcal{V}, \mathcal{V}'))$ which is the case by (7) since $\lambda(x) \in \Lambda(h_{\text{socket}}(\mathcal{T}, \mathcal{V}, \mathcal{V}'))$. The proof that $h_{\text{state}}(x, \mathcal{T}, \mathcal{V}, \mathcal{V}') \; \widehat{R}_0^{\varphi_0}(\alpha) \; h_{\text{state}}(y, \mathcal{T}, \mathcal{V}, \mathcal{V}')$ is by a subinduction on $|\alpha|$. We detail only the case for parallel compositions, the other cases being straightforward. Suppose $x \; R(\beta \parallel \gamma) \; y$. There exists $w_1, w_2, w_3, w_4 \in W$ such that $x \lhd (w_1, w_2)$, $w_1 \; R(\beta) \; w_3$, $w_2 \; R(\gamma) \; w_4$ and $y \lhd (w_3, w_4)$. Let $\mathcal{V}''$ be defined such that $\mathcal{V}''(L_1) = \{w_1\}$, $\mathcal{V}''(R_1) = \{w_2\}$, $\mathcal{V}''(L_2) = \{w_3\}$ and $\mathcal{V}''(R_2) = \{w_4\}$. Since, by hypothesis, there exists φ such that $(\lambda(x), \langle \beta \parallel \gamma \rangle \varphi) \in \text{Cl}(\varphi_0)$, by Lemma 1, $(\lambda(x).\ell, L_1) \in \text{Cl}(\varphi_0)$ and $(\lambda(x).r, R_1) \in \text{Cl}(\varphi_0)$. Therefore, by (6), $h_{\text{plug}}((x, w_1, w_2), \mathcal{V}'', \mathcal{V})$ is a plug of type 1. By a similar reasoning, $h_{\text{plug}}((y, w_3, w_4), \mathcal{V}'', \mathcal{V})$ is a plug of type 2. Let $\mathcal{T}' = \{(x, w_1, w_2), (y, w_3, w_4)\}$, $S' = h_{\text{socket}}(\mathcal{T}', \mathcal{V}'', \mathcal{V})$ and $H_i = h_{\text{hin}}(w_i, \mathcal{V}'')$ for all $i \in 1..4$. By definition, $(H_i, S') = h_{\text{state}}(w_i, \mathcal{T}', \mathcal{V}'', \mathcal{V})$ for all $i \in 1..4$. We prove that S' is a socket. For Condition 1 of Definition 3, suppose first that $h_{\text{plug}}((x, w_1, w_2), \mathcal{V}'', \mathcal{V})$ has both types. Then $(\lambda(w_1), L_2) \in H_1$ and $(\lambda(w_2), R_2) \in H_2$, hence $w_1 = w_3$, $w_2 = w_4$ and $\mathcal{T}'$ is a singleton. The case is similar if $h_{\text{plug}}((y, w_3, w_4), \mathcal{V}'', \mathcal{V})$ has both types. And if the plugs have different types, by (7) they have the same location. For Condition 2 of Definition 3, suppose that $(\lambda(x), \langle \alpha' \parallel \beta' \rangle \varphi') \in \text{Cl}(\varphi_0)$, $(\lambda(y), \varphi') \in h_{\text{hin}}(y, \mathcal{V})$, $(\lambda(w_1), \langle \alpha' \rangle L_2) \in H_1$ and $(\lambda(w_2), \langle \beta' \rangle R_2) \in H_2$, the other case being symmetric. By definition of $\mathcal{V}''$, $w_1 \; R(\alpha') \; w_3$ and $w_2 \; R(\beta') \; w_4$, hence $\mathcal{M} + \mathcal{V}, x \vDash \langle \alpha' \parallel \beta' \rangle \varphi'$ and $(\lambda(x), \langle \alpha' \parallel \beta' \rangle \varphi') \in h_{\text{hin}}(x, \mathcal{V})$. Therefore, S' is a socket.Moreover, since $\Lambda(S') = \{\lambda(x).\ell, \lambda(x).r\}$, by (6) and (7), $\{(H_1, S'), (H_2, S'), (H_3, S'), (H_4, S')\} \subseteq W_0^{\varphi_0}$. By the subinduction hypothesis, we have $H_1 \; \widehat{R}_0^{\varphi_0}(\beta) \; H_3$ and $H_2 \; \widehat{R}_0^{\varphi_0}(\gamma) \; H_4$. Therefore, by definition of the restricted accessibility relation, $h_{\text{state}}(x, \mathcal{T}', \mathcal{V}, \mathcal{V}'') \; \widehat{R}_0^{\varphi_0}(\beta \parallel \gamma) \; h_{\text{state}}(y, \mathcal{T}', \mathcal{V}, \mathcal{V}'')$.

Inductive step. Suppose now that IH.1 and IH.2 hold for a given k. Here the order of the proofs matters since we use IH.1 for $k+1$ to prove IH.2 for $k+1$. To prove IH.1 for $k+1$, suppose that $h_{\text{state}}(x, \mathcal{T}, \mathcal{V}, \mathcal{V}') \in W_k^{\varphi_0}$. Then for any formula $\langle \alpha \rangle \varphi$ such that $(\lambda(x), \langle \alpha \rangle \varphi) \in h_{\text{hin}}(x, \mathcal{V})$, there exists $y \in W$ such that $x \; R(\alpha) \; y$ and $\mathcal{M} + \mathcal{V}, y \vDash \varphi$. And by IH.2, Lemma 1 and (7), $h_{\text{state}}(y, \mathcal{T}, \mathcal{V}, \mathcal{V}') \in W_k^{\varphi_0}$, $h_{\text{state}}(x, \mathcal{T}, \mathcal{V}, \mathcal{V}') \; \widehat{R}_k^{\varphi_0}(\alpha) \; h_{\text{state}}(y, \mathcal{T}, \mathcal{V}, \mathcal{V}')$ and $(\lambda(y), \varphi) \in h_{\text{hin}}(y, \mathcal{V})$. Therefore $h_{\text{state}}(x, \mathcal{T}, \mathcal{V}, \mathcal{V}')$ is demand-satisfied and belongs to $W_{k+1}^{\varphi_0}$. The proof of IH.2 for $k+1$ is similar to the corresponding proof in the base case except that the hypothesis IH.1 for $k+1$ is used. For instance, in the case for parallel compositions, once it has been proved that $h_{\text{state}}(w_i, \mathcal{T}', \mathcal{V}'', \mathcal{V}) \in W_0^{\varphi_0}$ for all $i \in 1..4$, we use IH.1 to state that $h_{\text{state}}(w_i, \mathcal{T}', \mathcal{V}'', \mathcal{V}) \in W_{k+1}^{\varphi_0}$ for all $i \in 1..4$. Thus the subinduction hypothesis can be used to conclude. □

7 Conclusion and Perspectives

In this work, we have presented a procedure for the decision of the satisfiability problem of $\mathsf{PPDL}^{\text{det}}$. This procedure is a nontrivial adaptation of Pratt's elimination of Hintikka set procedure [23]. We have proved that this decision

procedure can be executed in deterministic exponential time. Since PDL can be trivially embedded into $\mathsf{PPDL}^{\mathrm{det}}$, this decision procedure is optimal and the satisfiability problem of $\mathsf{PPDL}^{\mathrm{det}}$ is EXPTIME-complete. This result extends a previous similar result for the iteration-free fragment of $\mathsf{PPDL}^{\mathrm{det}}$ [2]: adding a $\lhd$-deterministic separating parallel composition to PDL does not increase the theoretical complexity of the logic. This result contrasts with the 2EXPTIME complexity of the satisfiability problem of both PDL with intersection [17] and interleaving PDL [20].

Although $\lhd$-determinism is a very natural semantic condition, which turns the ternary separation relation into a partial binary operator, it would be interesting for future research to consider other classes of models. For instance, for the class of all models, only a 2EXPTIME upper bound is currently known [2]. For the class of models where the separation of states is deterministic, even though PRSPDL has been proved to be undecidable [3], it may be possible that in the absence of the four store/recover programs of PRSPDL the logic is decidable. Finally, for many concrete semantics like Petri nets or finite sets of agents, the separation relation would have to be both $\lhd$-deterministic and associative. The minimal associative binary normal modal logic is undecidable [15], therefore the variant of PDL with separating parallel composition interpreted in the class of all associative $\lhd$-deterministic models is undecidable too. But since there exists some decidable associative binary modal logics (for instance the separation logics kSL0 [7]), it would be interesting to search for decidable variants of PDL with separating parallel composition interpreted in special classes of associative $\lhd$-deterministic models.

References

1. Abrahamson, K.R.: Modal logic of concurrent nondeterministic programs. In: Kahn, G. (ed.) Semantics of Concurrent Computation. LNCS, vol. 70, pp. 21–33. Springer, Heidelberg (1979)
2. Balbiani, P., Boudou, J.: Tableaux methods for propositional dynamic logics with separating parallel composition. In: Felty, A.P., Middeldorp, A. (eds.) CADE-25. LNCS, vol. 9195, pp. 539–554. Springer, Switzerland (2015)
3. Balbiani, P., Tinchev, T.: Definability and computability for PRSPDL. In: Goré, R., Kooi, B.P., Kurucz, A. (eds.) Advances in Modal Logic - AiML, pp. 16–33. College Publications, San Diego (2014)
4. Benevides, M.R.F., de Freitas, R.P., Viana, J.P.: Propositional dynamic logic with storing, recovering and parallel composition. In: Hermann Haeusler, E., Farinas del Cerro, L. (eds.) Logical and Semantic Frameworks, with Applications - LSFA. ENTCS, vol. 269, pp. 95–107 (2011)
5. Boudou, J.: Exponential-size model property for PDL with separating parallel composition. In: Italiano, G.F., Pighizzini, G., Sannella, D.T. (eds.) MFCS 2015. LNCS, vol. 9234, pp. 129–140. Springer, Heidelberg (2015)
6. Brookes, S.: A semantics for concurrent separation logic. Theoret. Comput. Sci. **375**(1–3), 227–270 (2007)
7. Calcagno, C., Yang, H., O'Hearn, P.W.: Computability and complexity results for a spatial assertion language for data structures. In: Hariharan, R., Mukund, M., Vinay, V. (eds.) FSTTCS 2001. LNCS, vol. 2245, pp. 108–119. Springer, Heidelberg (2001)

8. Demri, S., Deters, M.: Separation logics and modalities: a survey. J. Appl. Non-Class. Logics **25**(1), 50–99 (2015)
9. van Ditmarsch, H., van der Hoek, W., Kooi, B.P.: Dynamic Epistemic Logic, vol. 337. Springer Science & Business Media, Netherlands (2007)
10. van Eijck, J., Stokhof, M.: The gamut of dynamic logics. In: Gabbay, D.M., Woods, J. (eds.) Logic and the Modalities in the Twentieth Century. Handbook of the History of Logic, vol. 7, pp. 499–600. Elsevier, Amsterdam (2006)
11. Fischer, M.J., Ladner, R.E.: Propositional dynamic logic of regular programs. J. Comput. Syst. Sci. **18**(2), 194–211 (1979)
12. Frias, M.F.: Fork Algebras in Algebra, Logic and Computer Science. Advances in Logic, vol. 2. World Scientific, Singapore (2002)
13. Harel, D.: Recurring dominoes: making the highly undecidable highly understandable (preliminary report). In: Karpinski, M. (ed.) FCT. LNCS, vol. 158, pp. 177–194. Springer, Heidelberg (1983)
14. Herzig, A.: A simple separation logic. In: Libkin, L., Kohlenbach, U., de Queiroz, R. (eds.) WoLLIC 2013. LNCS, vol. 8071, pp. 168–178. Springer, Heidelberg (2013)
15. Kurucz, Á., Németi, I., Sain, I., Simon, A.: Decidable and undecidable logics with a binary modality. J. Logic, Lang. Inf. **4**(3), 191–206 (1995)
16. Lange, M.: Model checking propositional dynamic logic with all extras. J. Appl. Logic **4**(1), 39–49 (2006)
17. Lange, M., Lutz, C.: 2-ExpTime lower bounds for propositional dynamic logics with intersection. J. Symbolic Logic **70**(4), 1072–1086 (2005)
18. Larchey-Wendling, D., Galmiche, D.: The undecidability of boolean BI through phase semantics. In: Logic in Computer Science - LICS, pp. 140–149. IEEE Computer Society (2010)
19. Marx, M., Pólos, L., Masuch, M.: Arrow Logic and Multi-modal Logic. CSLI Publications, Stanford (1996)
20. Mayer, A.J., Stockmeyer, L.J.: The complexity of PDL with interleaving. Theoret. Comput. Sci. **161**(1&2), 109–122 (1996)
21. O'Hearn, P.W.: Resources, concurrency, and local reasoning. Theoret. Comput. Sci. **375**(1–3), 271–307 (2007)
22. Peleg, D.: Concurrent dynamic logic. J. ACM **34**(2), 450–479 (1987)
23. Pratt, V.R.: Models of program logics. In: Foundations of Computer Science - FOCS, pp. 115–122. IEEE Computer Society (1979)
24. Pym, D.J.: The Semantics and Proof Theory of the Logic of Bunched Implications. Applied Logic Series, vol. 26. Springer, Netherlands (2002). Kluwer Academic Publishers
25. Reynolds, J.C.: Separation logic: a logic for shared mutable data structures. In: Logic in Computer Science - LICS, pp. 55–74. IEEE Computer Society (2002)
26. Schild, K.: A correspondence theory for terminological logics: preliminary report. In: Mylopoulos, J., Reiter, R. (eds.) International Joint Conference on Artificial Intelligence - IJCAI, pp. 466–471. Morgan Kaufmann (1991)
27. Venema, Y.: A modal logic for chopping intervals. J. Logic Comput. **1**(4), 453–476 (1991)