Parallel First-Order Dynamic Logic and Its Expressiveness and Axiomatization*

Zhiguo Zhang and Yunfei Jiang

Faculty of Information Science and Technology
Sun Yatsen University
Guangzhou City, 510275 China
lnszzg@mail.sysu.edu.cn

Abstract. For modeling the parallel actions, the quantified dynamic logic (QDL) is extended to Parallel First-order Dynamic Logic (PaFDL) with parallel action compositions. The composition is introduced as an operator $\cap$ on actions in the same syntax as in Peleg's CQDL but its semantics is defined differently from those of CQDL. The expressive power of PaFDL is proved to be the same as that of QDL. An axiomatic system is given and its first-order soundness and completeness are proved. Compared with other parallel or concurrent Dynamic Logics, PaFDL has a very easy and intuitive understanding for parallel actions as they are in the sequential models.

Keywords: Dynamic logic, First-order logic, Parallel actions, Expressiveness, Axiomatization.

1 Introduction

Propositional dynamic logic (PDL) was first proposed by Fischer and Ladner [FiL79] for describing the sequential program dynamic characteristics such as correctness, termination, and equivalence. It has received considerable attention, and many of its aspects have been thoroughly investigated [Nis79, Har84, HKT00]. Many investigations concern with complexity and axiomatization [Bal01, Dan84, KoP81, Lan05, LaL05, Lei81, Pra78]. PDL has been also extended to first-order level with many deep investigations [BeS01, GrS91, Har79]. Applications of dynamic logic to program verification and reasoning about actions and knowledge are also studied [PrS96, HRS87]. For modeling concurrent behaviors of multiagent systems, propositional dynamic logic has been extended to concurrent propositional dynamic logic (CPDL) and concurrent quantified dynamic logic (CQDL) by Peleg [Pel87a, Pel87b] with an extension of parallel actions. Peleg's approach views concurrency in its purest form as the dual notion of nondeterminism. Nondeterminism introduces splitting at a state into several branches, and letting the process choose between the different possible continuations. Analogously, concurrency means again splitting a node into several branches, but requiring the process to execute all possible

* This work is partially supported by Guangdong Key Laboratory of Information Security.

M. Xu et al. (Eds.): APPT 2007, LNCS 4847, pp. 600 – 607, 2007.

continuations. This is basically the classical concept of and/or decomposition, which occurs widely in logic, game theory, etc. It is shown that CPDL is strictly more expressive than PDL. A complete axiom system and its decidability for CPDL are provided. Some other investigations on concurrency of dynamic logic are presented in [Dan84].

In this paper, we introduce a parallel first-order dynamic logic (short for PaFDL) by adopting the syntax of Peleg's CQDL and give it a different semantics for parallel action compositions. The expressiveness of PaFDL is proved to be the same as QDL which is proposed by Harel et al. in [HKT00]. A sound and complete axiomatic system is provided for a restricted set of formulas of the form $A \rightarrow \langle\alpha\rangle B$. In the following, we give the syntactic definitions in Section 2, define their semantics in Section 3, and discuss the expressiveness in Section 4, the axiomatization in Section 5, and finally conclusion in Section 6.

2 Syntax

The syntax of parallel first-order dynamic logic (PaFDL) is based upon two kinds of symbols: *logical symbols* including the connectives $\neg$ and $\vee$, the punctuation marks (,), $\langle$, $\rangle$, [and], the equality symbol =, the existential qualifier $\exists$ and the universal qualifier symbol $\forall$, a countable set V of variables, the truth symbols *true* and *false*; *extralogical symbols* including a countable set P of predicate symbols, a countable set F of function symbols, and a countable set Π_0 of atomic action symbols. Each of function and predicate symbols has associated with it a natural number which is called its arity. 0-ary function symbols are called constants and 0-ary predicate symbols are called propositional constants. These countable sets constitute the basis for PaFDL. Complex formulas and complex programs over this basis are defined as follows.

Definition 1. (Basis) A basis for PaFDL is $B=(F,P,\Pi_0)$ of sets of symbols, where F, P and Π_0 are understood to be the sets of function symbols, predicate symbols, and action symbols respectively as described above.

Definition 2. (Terms) The set T_B of all terms of PaFDL over a basis $B=(F,P,\Pi_0)$ is inductively defined by:

(1) Every variable from V is a term. ($V \subseteq T_B$)
Every constant from F is a term.

(2) If $t_1, \ldots, t_n$ ($n \geq 1$) are terms and $f \in F$ is an n-ary function symbol, then $f(t_1, \ldots, t_n)$ is also a term.

Definition 3. (Formulas) The set Φ of all well-formed formulas of PaFDL over a basis $B=(F,P,\Pi_0)$ is inductively defined by:

(1) Every propositional constant from P is a formula. ($P \subseteq \Phi$)
The truth symbols *false* and *true* are formulas.
If t_1 and t_2 are terms, then $t_1 = t_2$ is a formula.
If $t_1, \ldots, t_n$ ($n \geq 1$) are terms and $p \in P$ is an n-ary predicate symbol, then $p(t_1, \ldots, t_n)$ is also a formula.

(2) If A is a formula then $(\neg A)$ ("not A") is a formula.
If A and B are formulas then $(A\vee B)$ ("A or B") is a formula.
If A is a formula and x is a variable, then $(\exists xA)$ and $(\forall xA)$ are formulas.
If α is an action and A is a formula then $[\alpha]A$ ("every execution of α from the present state leads to a state where A is true") is a formula

Definition 4. (Actions) The set Π of all actions of PaFDL over a basis B=(F,P,Π_0) is inductively defined by:

(1) Every atomic action is an action. ($\Pi_0 \subseteq \Pi$)
(2) If α and β are actions then $(\alpha;\beta)$ ("do α followed by β") is an action.
If α and β are actions then $(\alpha\cup\beta)$ ("do α or β, nondeterministically") is an action.
If α and β are actions then $(\alpha\cap\beta)$ ("do α and β, in parallel") is an action.
If α is a action then α^* ("repeat α a finite, but nondeterministically determined, number of times") is an action.
If A is a formula then $A?$ ("proceed if A is true, else fail") is an action

The syntax for actions we adapted is exactly the same as Peleg's CQDL. However, we will have a different view of concurrency for parallel actions as described in the following sections.

3 Semantics

First we define a function patching operator as follows: if f: $D\rightarrow E$ is any function, $x\in D$ and $v\in E$, then $f[x/v]$: $D\rightarrow E$ is the function defined by

$$f[x/v](y) \stackrel{\text{def}}{=} \begin{cases} v & \text{if } x = y \\ f(y) & \text{otherwise} \end{cases}$$

We also need to define the domain *Bool* as

$$Bool = \{true, false\}.$$

As default, we always include this domain in our description.

Definition 5. (Interpretation) Let B=(F,P,Π_0) be a basis for PaFDL. An interpretation of B is a pair $\mathcal{J}$=(D,$\mathcal{J}_0$), where D is a non-empty set (called the domain or world of states of $\mathcal{J}$) and $\mathcal{J}_0$ is a mapping which assigns

(1) To every constant $c\in F$ an element $\mathcal{J}_0(c)\in D$;
(2) To every function symbol $f\in F$ of arity $n\geq 1$ a total function $\mathcal{J}_0(f)$: $D^n\rightarrow D$;
(3) To every propositional constant $a\in P$ an element $\mathcal{J}_0(a)\in Bool$, where *Bool* is the domain of truth values;
(4) To every predicate symbol $p\in P$ of arity $n\geq 1$ a predicate $\mathcal{J}_0(p)$: $D^n\rightarrow Bool$.

Definition 6. (Assignment) Let B=(F,P,Π_0) be a basis for PaFDL and $\mathcal{J}$=(D,$\mathcal{J}_0$) be an interpretation of B. A total function σ: $V\rightarrow D$ mapping variables to the domain D of $\mathcal{J}$ is called an assignment. In some context, an assignment is also called state. The set of all assignments for $\mathcal{J}$ is denoted by $\Sigma_{\mathcal{J}}$ or simply by Σ.

The definition of interpretation then can be extended to include:
(5) To every action symbol $\alpha\in\Pi_0$ a binary relation $\mathcal{J}_0(\alpha)\subseteq\Sigma\times\Sigma$.

An interpretation and an assignment together induce a mapping from every term to an element in the domain of the interpretation and from every formula to a truth value and from every action to a binary relation over assignments. It is clear that the interpretation must be extended inductively as follows to supply the intended meanings for the complex terms, actions and formulas:

Definition 7. (Semantics) Let $\mathcal{J}=(D,\mathcal{J}_0)$ be an interpretation of a basis $B=(F,P,\Pi_0)$ for PaFDL. To $\mathcal{J}$ is associated a functional, also denoted by $\mathcal{J}$, which maps every term $t\in T_B$ to a function $\mathcal{J}(t)$: $\Sigma\rightarrow D$ and every formula $A\in\Phi$ to a function $\mathcal{J}(A)$: $\Sigma\rightarrow Bool$ and every action $\alpha\in\Pi$ to a binary relation $\mathcal{J}(\alpha)\subseteq\Sigma\times\Sigma$. Each parts of this functional are defined inductively over T_B, Φ and Π as follows:

Semantics of terms

(1) If $c\in F$ is a constant, then $\mathcal{J}(c)(\sigma)=\mathcal{J}_0(c)$ for all assignments $\sigma\in\Sigma$.
If $x\in V$ is a variable, then $\mathcal{J}(x)(\sigma)=\sigma(x)$ for all assignments $\sigma\in\Sigma$.

(2) If $t_1, \ldots, t_n$ ($n\geq 1$) are terms and $f\in F$ is an n-ary function symbol, then $\mathcal{J}(f(t_1, \ldots, t_n))(\sigma)=\mathcal{J}_0(f)(\mathcal{J}(t_1)(\sigma),\ldots,\mathcal{J}(t_n)(\sigma))$ for all assignments $\sigma\in\Sigma$.

Semantics of actions

(1) For any atomic action a in Π_0, $\mathcal{J}(a)= \mathcal{J}_0(a)$.

(2) $(s,t)\in\mathcal{J}(\alpha;\beta)$ iff there exists a state z such that $(s,z)\in\mathcal{J}(\alpha)$ and $(z,t)\in\mathcal{J}(\beta)$.

(3) $(s,t)\in\mathcal{J}(\alpha\cup\beta)$ iff $(s,t)\in\mathcal{J}(\alpha)$ or $(s,t)\in\mathcal{J}(\beta)$.

(4) $(s,t)\in\mathcal{J}(\alpha\cap\beta)$ iff $(s,t)\in\mathcal{J}(\alpha;\beta)$ and $(s,t)\in\mathcal{J}(\beta;\alpha)$.

(5) $(s,t)\in\mathcal{J}(\alpha^*)$ iff there exists a non-negative integer n and there exist states $z_0, \ldots, z_n$ such that $z_0=s$, $z_n=t$ and for all $k=1..n$, $(z_{k-1}, z_k)\in\mathcal{J}(\alpha)$.

(6) $(s,t)\in\mathcal{J}(A?)$ iff $s=t$ and $\mathcal{J}(A)(t)=true$.

Semantics of formulas

(1) For any propositional constant a in P, then $\mathcal{J}(a)(s)=\mathcal{J}_0(a)$ for all s in Σ.
$\mathcal{J}(false)(s) = false$ and $\mathcal{J}(true)(s) = true$, for any s in Σ.
If t_1, t_2 are terms, then $\mathcal{J}(t_1=t_2)(s)=true$ if $\mathcal{J}(t_1)(s)=\mathcal{J}(t_2)(s)$, $\mathcal{J}(t_1=t_2)(s)=false$ if $\mathcal{J}(t_1)(s)\neq\mathcal{J}(t_2)(s)$, for all s in Σ.
If $t_1, \ldots, t_n$ ($n\geq 1$) are terms and $p\in P$ is an n-ary predicate symbol, then $\mathcal{J}(p(t_1, \ldots, t_n))(s)= \mathcal{J}_0(p)(\mathcal{J}(t_1)(s),\ldots,\mathcal{J}(t_n)(s))$ for all assignments $\sigma\in\Sigma$.

(2) $\mathcal{J}(\neg A)(s)=true$ if $\mathcal{J}(A)(s)=false$, $\mathcal{J}(\neg A)(s)=false$ if $\mathcal{J}(A)(s)=true$, for all s in Σ.
$\mathcal{J}(A\vee B)(s)=\mathcal{J}(A)(s)\vee\mathcal{J}(B)(s)$, for all s in Σ.
$\mathcal{J}([\alpha]A)(s)=true$ if s in $\{r$: for all states t, if $(r,t)\in\mathcal{J}(\alpha)$ then $\mathcal{J}(A)(t)=true\}$,
$\mathcal{J}([\alpha]A)(s)=false$ otherwise, for any s in Σ.
If $A\in T_B$ is a formula and $x\in V$ is a variable, then
$\mathcal{J}((\exists xA))(s)=$

$$\begin{cases} true & \text{if there is an element } d\in D \text{ of the domain such that } \mathcal{J}(A)(s[x/d])=true \\ false & \text{otherwise} \end{cases}$$

for all s in Σ.
If $A\in T_B$ is a formula and $x\in V$ is a variable, then

$$\mathcal{J}((\forall xA))(s)=\begin{cases} true & \text{if for all } d\in D \text{ of the domain such that } \mathcal{J}(A)(s[x/d])=true \\ false & \text{otherwise} \end{cases}$$

for all s in Σ.

We can define a new modal operator $\langle\rangle$ as follows:

$$\langle\alpha\rangle A =^{\text{def}} \neg[\alpha]\neg A.$$

Now consider a formula A. We shall say that A is valid in $\mathcal{J}$ or that $\mathcal{J}$ is a model of A, or "$\mathcal{J}\models A$", iff for all states s in $\Sigma_{\mathcal{J}}$, $\mathcal{J}(A)(s)=true$. A is said to be logically valid, or "$\models A$", iff for all interpretation $\mathcal{J}$, $\mathcal{J}\models A$. We shall say that A is satisfiable in $\mathcal{J}$ or that $\mathcal{J}$ satisfies A, or "$\mathcal{J}\between A$", iff there exists a state t such that $\mathcal{J}(A)(t)=true$. A is said to be logically satisfiable, or "$\between A$", iff there exists a model $\mathcal{J}$ such that $\mathcal{J}\between A$.

4 Expresiveness of PaFDL

We investigate the expressive power of PaFDL relative to quantified dynamic logic (QDL) with no $\cap$ operator. First we introduce a definition that allows us to compare different dynamic logics. If DL_1 and DL_2 are two different dynamic logics over the same basis, we say that DL_2 is as expressive as DL_1 and write $DL_1\leq DL_2$ if for each formula A in DL_1 there is a formula B in DL_2 such that $\mathcal{J}(A\leftrightarrow B)(s)=true$ for all $\mathcal{J}$ and all s. Intuitively, $<$ and $\equiv$ mean "strictly less expressive than" and "of equal expressive power" respectively.

Lemma 1. QDL≤PaFDL.

Proof. This is directly from the syntactic definition of PaFDL. Actually, PaFDL is extended from QDL. □

Lemma 2. PaFDL≤QDL.

Proof. We should prove that for any A in PaFDL there is a formula B in QDL such that $\mathcal{J}(A\leftrightarrow B)(s)=true$ for all $\mathcal{J}$ and all s. Any formula A in PaFDL can be either containing operator $\cap$ or not. If A contains no operator $\cap$, then A is in QDL by the syntactic definition of PaFDL.

Now suppose that A contains the operator $\cap$. Without losing the generality and for simplification we consider only the case A is $[\alpha\cap\beta]B$. Given any interpretation $\mathcal{J}$ and $s\in\Sigma_{\mathcal{J}}$, by the semantic definition of PaFDL,

$$\mathcal{J}([\alpha\cap\beta]B)(s)=true$$
$$\text{iff}$$
$$s\in\{r: \text{for all states } t, \text{ if } (r,t)\in\mathcal{J}(\alpha\cap\beta) \text{ then } \mathcal{J}(B)(t)=true\}$$
$$\text{iff}$$
$$s\in\{r: \text{for all states } t, \text{ if } (r,t)\in\mathcal{J}(\alpha;\beta) \text{ and } (r,t)\in\mathcal{J}(\beta;\alpha) \text{ then } \mathcal{J}(B)(t)=true\}$$
$$\text{iff}$$
$$\mathcal{J}([\alpha;\beta]B)(s)=true \text{ and } \mathcal{J}([\beta;\alpha]B)(s)=true$$

iff

$$\mathcal{J}([\alpha;\beta]B\wedge[\beta;\alpha]B)(s)=true.$$

Clearly $[\alpha;\beta]B\wedge[\beta;\alpha]B$ is in QDL. □

Theorem 1. (Expressiveness) QDL≡PaFDL.

Proof. This is directly from the above two lemmas. □

This theorem shows that PaFDL has the same expressive power as QDL. Intuitively, PaFDL understands parallel actions in just the interleavable way as in QDL.

5 Axiomatization of PaFDL

Here we introduce an axiomatic system for the PaFDL calculus with an interpretation. Let $B=(F,P,\Pi_0)$ be a basis for PaFDL and $\mathcal{J}=(D,\mathcal{J}_0)$ be an interpretation of B. All semantically valid in $\mathcal{J}$ formulas of form $A\rightarrow\langle\alpha\rangle B$ are taken as axioms. This may lead some confusions with calculus because we consider that calculus has nothing to do with semantics. However, we usually can understand intuitively what the interpreted formulas to be true.

Axiom schemes

(A1) All instances of valid PaPDL formulas;
(A2) All instances of valid first-order formulas;
(A3) All formulas of form $A\rightarrow\langle\alpha\rangle B$ which satisfies
"for all s and t such that $(s,t)\in\mathcal{J}(\alpha)$, $\mathcal{J}(A)(s)=true$ implies $\mathcal{J}(B)(t)=true$",
where α is an atomic action.

Inference rule
(MP) modus ponens: from A, $A\rightarrow B$ infer B

If X is a set of formulas and A is a formula then we say that A is deducible from X in $\mathcal{J}$, or "$X \vdash_{\mathcal{J}} A$", if there exists a construction sequence $A_0, A_1, \ldots, A_n=A$ for A from the set of axioms and the inference rule (MP). Further, we say that A is deducible in $\mathcal{J}$ or "$\vdash_{\mathcal{J}} A$" iff $\emptyset \vdash_{\mathcal{J}} A$. X is said to be consistent in $\mathcal{J}$ iff not $X \vdash_{\mathcal{J}} false$.

Theorem 2. (First-order Soundness) For any PaFDL formula of the form $A\rightarrow\langle\alpha\rangle B$, for first-order A and B and action α containing first-order tests only,

$$\vdash_{\mathcal{J}} A\rightarrow\langle\alpha\rangle B \text{ implies } \mathcal{J}\models A\rightarrow\langle\alpha\rangle B$$

Proof. The proof of the soundness proceeds by induction on the compositions of actions. □

Theorem 3. (First-order Completeness) For any PaFDL formula of the form $A\rightarrow\langle\alpha\rangle B$, for first-order A and B and action α containing first-order tests only,

$$\mathcal{J}\models A\rightarrow\langle\alpha\rangle B \text{ implies } \vdash_{\mathcal{J}} A\rightarrow\langle\alpha\rangle B$$

Proof. The proof of the completeness is simplified as the following induction. For atomic α, it is clear that $\mathcal{J} \models A \rightarrow \langle\alpha\rangle B$ implies $\vdash_{\mathcal{J}} A \rightarrow \langle\alpha\rangle B$. For any composed action $\alpha = \alpha_1 \cdot \alpha_2$, where operator "·" indicates one of ";", "∩","∪", we can prove the conclusion for α holds in the condition that the induction hypotheses for α_1 and α_2 both hold. And more for α* we can prove the same conclusion. □

6 Conclusion and Discussion

The Parallel First-order Dynamic Logic (PaFDL) is introduced with the syntactic and semantic definitions. The same syntax as Peleg's CQDL is adopted and semantics of PaFDL is defined differently from those of CQDL. The expressive power of PaFDL is proved to be the same as that of QDL. An axiomatic system is given and its first-order soundness and completeness are proved.

Compared with other parallel or concurrent Dynamic Logics, PaFDL has a very easy and intuitive understanding for parallel actions as they are in sequential models. According to Theorem 1, PaFDL has the same expressive power as its sequential counterpart QDL, not as the concurrent version CQDL. This indicates that PaFDL depicts parallel actions in a much like way the sequential models take.

Many other properties remain to be investigated including complexity, more extended axiomatization, and applications in reasoning about parallel actions and changes.

References

[BeS01] Beckert, B., Schlager, S.: A sequent calculus for first-order dynamic logic with trace modalities. In: Goré, R.P., Leitsch, A., Nipkow, T. (eds.) IJCAR 2001. LNCS (LNAI), vol. 2083, pp. 626–641. Springer, Heidelberg (2001)

[Bal01] Balbiani, P.: A new proof of completeness for a relative modal logic with composition and intersection. Journal of Applied Non-Classical Logics 11, 269–280 (2001)

[Dan84] Danecki, R.: Nondeterministic propositional dynamic logic with intersection is decidable. In: Skowron, A. (ed.) Computation Theory. LNCS, vol. 208, pp. 34–53. Springer, Heidelberg (1985)

[FiL79] Fischer, M., Ladner, R.: Propositional dynamic logic of regular programs. Journal of Computer and System Sciences 18, 194–211 (1979)

[GrS91] Groenendijk, J., Stokhof, M.: Dynamic Predicate Logic. Linguistics and Philosophy 14(1), 31–100 (1991)

[Har84] Harel, D.: Dynamic logic. In: Gabbay, D., Guenthner, F. (eds.) (editors): Handbook of Philosophical Logic, vol. II, pp. 497–604. D. Reidel, Dordrecht (1984)

[Har79] Harel, D.: First-Order Dynamic Logic. LNCS, vol. 68. Springer, Heidelberg (1979)

[HKT00] Harel, D., Kozen, D., Tiuryn, J.: Dynamic Logic. MIT Press, Cambridge (Massachusetts) (2000)

[HRS87] Heisel, M., Reif, W., Stephan, W.: Program verification using dynamic logic. In: Börger, E., Kleine Büning, H., Richter, M.M. (eds.) CSL 1987. LNCS, vol. 329, Springer, Heidelberg (1988)

[KoP81] Kozen, D., Parikh, R.: An elementary proof of the completeness of PDL. Theoretical Computer Science 14, 113–118 (1981)

[Lan05] Lange, M.: A lower complexity bound for propositional dynamic logic with intersection. In: Schmidt, R., Pratt-Hartmann, I., Reynolds, M., Wansing, H. (eds.) Advances in Modal Logic, vol. 5, pp. 133–147. King's College Publications, London (2005)

[LaL05] Lange, M., Lutz, C.: 2-EXPTIME lower bounds for propositional dynamic logics with intersection. Journal of Symbolic Logic 70, 1072–1086 (2005)

[Lei81] Leivant, D.: Proof Theoretic Methodology for Propositional Dynamic Logic. In: Díaz, J., Ramos, I. (eds.) Formalization of Programming Concepts. LNCS, vol. 107, Springer, Heidelberg (1981)

[Nis79] Nishimura, H.: Sequential method in propositional dynamic logic. Acta Informatica 12, 377–400 (1979)

[Pel87a] Peleg, D.: Concurrent dynamic logic. Journal of the ACM 34, 450–479 (1987)

[Pel87b] David, P.: Communication in Concurrent Dynamic Logic. J. Comput. Syst. Sci. 35, 23–58 (1987)

[Pra80] Pratt, V.: A near-optimal method for reasoning about action. Journal of Computer and System Sciences 20, 231–254 (1980)

[Pra78] Pratt, V.: A practical decision method for propositional dynamic logic. In: Proceedings of the 10th Annual ACM Symposium on Theory of Computing, pp. 326–337. ACM Press, New York (1978)

[PrS96] Prendinger, H., Schurz, G.: Reasoning about action and change: A dynamic logic approach. Journal of Logic, Language, and Information 5(2), 209–245 (1996)